A Practical Guide to Unbinned Unfolding

Florencia Canelli,[1, *] Kyle Cormier,[1, *] Andrew Cudd,[2, †] Dag Gillberg,[3, ‡] Roger G. Huang,[4, †] Weijie Jin,[1, *] Sookhyun Lee,[5, §] Vinicius Mikuni,[4, 6, ¶] Laura Miller,[7] Benjamin Nachman,[4, 8, 9, **] Jingjing Pan,[4, 10, **] Tanmay Pani,[11, ††] Mariel Pettee,[4, ‡] Youqi Song,[10, ††] and Fernando Torales Acosta[4, ¶]

[1]*Department of Physics, University of Zurich, 8006 Zürich, Switzerland*
[2]*Department of Physics, University of Colorado Boulder, Boulder, CO 80309, USA*
[3]*Department of Physics, Carleton University, Ottawa, ON K1S 5B6, Canada*
[4]*Physics Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA*
[5]*Department of Physics & Astronomy, University of Tennessee at Knoxville, TN 37996, USA*
[6]*National Energy Research Scientific Computing Center, Berkeley Lab, Berkeley, CA 94720, USA*
[7]*TRIUMF, Vancouver, BC V6T 2A3, Canada*
[8]*Department of Particle Physics and Astrophysics, Stanford University, Stanford, CA 94305, USA*
[9]*Fundamental Physics Directorate, SLAC National Accelerator Laboratory, Menlo Park, CA 94025, USA*
[10]*Wright Laboratory, Yale University, New Haven, CT 06511, USA*
[11]*Department of Physics and Astronomy, Rutgers University, New Brunswick, NJ 08901, USA*
(Dated: July 15, 2025)

Unfolding, in the context of high-energy particle physics, refers to the process of removing detector distortions in experimental data. The resulting unfolded measurements are straightforward to use for direct comparisons between experiments and a wide variety of theoretical predictions. For decades, popular unfolding strategies were designed to operate on data formatted as one or more binned histograms. In recent years, new strategies have emerged that use machine learning to unfold datasets in an unbinned manner, allowing for higher-dimensional analyses and more flexibility for current and future users of the unfolded data. This guide comprises recommendations and practical considerations from researchers across a number of major particle physics experiments who have recently put these techniques into practice on real data.

CONTENTS

---

* CMS Collaboration
† T2K Collaboration
‡ ATLAS Collaboration
§ LHCb Collaboration
¶ H1 Collaboration
** ATLAS & H1 Collaborations
†† STAR Collaboration

I. INTRODUCTION

Science is driven by the pursuit of deeper truths about our Universe, but scientists must extract these truths using datasets collected by instruments that carry inherent measurement errors and resolution limits. To assess the validity of a theory by directly comparing it with experimental data, scientists must therefore account for the expected changes that will occur when a pristine theoretical prediction is altered by the imperfections of the measurement device. If the physics process to be measured and the model to be tested are both known *a priori*, the standard approach used by researchers is "forward modeling": taking a theoretical prediction and then modifying it using a carefully-constructed, high-accuracy detector simulation that requires expert knowledge of the detector geometry and its calibrations in order to directly compare the prediction with real data. Each time one wants to test a new theory or physics process, however, this time- and compute-intensive process must be fully re-run, making it impractical for testing multiple theories, particularly if a new hypothesis is developed long after the measurements have already taken place.

As an alternative to this paradigm, a real-world dataset can instead be corrected to try to remove any detector distortions from the data, thereby resulting in measurements that can be directly compared between experiments and theoretical predictions. Such a dataset could then be tested against any number of theoretical hypotheses without having to repeat the detector simulation each time. The nuanced process of removing de-

tector effects from data is widespread across many areas of science and can be variously referred to as "deconvolution", "denoising", "reconstruction", or "inverse modeling". In High-Energy Physics (HEP), it has historically been called "unfolding".

For decades, unfolding in HEP has been performed on binned datasets of no more than a handful of variables at a time. Algorithms such as Iterative Bayesian Unfolding [1] frame the experimental process as a series of transformations including a detector response matrix applied to the "true" dataset, i.e. an observable prior to the interaction with the detector — or, equivalently, what a detector with perfect resolution and detection efficiency would measure. The detector response matrix encodes bin-by-bin measurement effects such as reconstruction errors, energy smearing, and background noise. Unfolding, in this framing, means effectively inverting this detector response matrix so that it can be applied instead to experimental data and thereby produce a truth-level dataset. Approximating this inverted matrix can produce numerical instabilities that can yield unreliable results, particularly as the number of variables — and therefore the number of independent degrees of freedom — increases. As a result, unfolded results from traditional binned strategies are restricted to fixed histogram bins for only a few simultaneous observables measured at once.

Over the past few years, HEP researchers have proposed new unfolding techniques that use machine learning pipelines to perform unfolding in an unbinned manner [2–18], allowing for entire Monte Carlo simulated datasets to be corrected event-by-event such that the entire dataset aligns with the target. In addition to the inherent flexibility of removing a predetermined choice of bins from a measurement, these methods use neural networks that can readily process high-dimensional inputs, meaning that the unfolding can be performed for dozens or more variables at once without significant computational overhead. One such method that has gained recent prominence in the field is called OmniFold [7, 19]. OmniFold is a density reweighting method that determines a set of weights that can transform a simulated dataset event-by-event to match the target, i.e. the data one would see with an idealized detector. These weights are derived from the ratio of the likelihoods of two samples. These ratios can be directly approximated by harnessing neural network classifiers, as these methods are known to naturally learn likelihood ratios in order to optimally discriminate between classes.

Unbinned unfolding has now progressed well beyond simulation-only and proof-of-concept results. Between mid-2021 and mid-2025, a number of public results have made use of unbinned unfolding methods for measurements on datasets from at least five different HEP detectors: ATLAS, CMS, H1, LHCb, and STAR, as well as a study on highly realistic simulation from one accelerator-based neutrino experiment, T2K. This white paper is designed to further facilitate broader adoption of unbinned unfolding in the field by synthesizing the lessons learned from researchers involved in each of these measurements. In Section II, we summarize the core methodology of OmniFold, the unbinned unfolding technique that enabled each of these public measurements. In Section III, we summarize the specific choices made by the various analyses in areas including hyperparameter optimization, ensembling, uncertainty calculation, validation, and presentation of the final results. Finally, in Section IV, we conclude and propose some future directions for unbinned unfolding in HEP using machine learning.

II.   UNBINNED UNFOLDING

Each unbinned analysis summarized in this work makes use of the classifier-based machine learning method called OmniFold [7, 19], which is illustrated in Figure 1 and briefly reviewed here. OmniFold requires two data samples that provide distributions of three different sets of observables in total:

1. $\vec{x}_{\text{true}}^{\text{MC}}$: a Monte Carlo simulation sample containing a list of events with a number of "truth-level" observables, i.e. particle-level properties as they would be captured by an idealized detector;

2. $\vec{x}_{\text{reco}}^{\text{MC}}$: a Monte Carlo simulation sample containing that same list of events, but instead with "reco-level" observables, i.e. the properties of those same events as they would be captured by a realistic detector; and

3. $\vec{x}_{\text{reco}}^{\text{data}}$: the experimental data that, by definition, only has "reco-level" observables.

Every dataset also specifies a "weight" for each event. For the Monte Carlo datasets $\vec{x}_{\text{true}}^{\text{MC}}$ and $\vec{x}_{\text{reco}}^{\text{MC}}$, these weights are generally given by the Monte Carlo generator, but for the real data $\vec{x}_{\text{reco}}^{\text{data}}$ these weights are all set to 1. An event indexed i in the MC sample can hence typically be written as $(w_i, \vec{x}_{\text{true},i}^{\text{MC}}, \vec{x}_{\text{reco},i}^{\text{MC}})$, and the task of OmniFold is to use these three different sets of event properties to adjust the weight w_i for each MC event such that the weighted sample of $\vec{x}_{\text{reco}}^{\text{MC}}$ agrees with data. At that point, the weighted sample of $\vec{x}_{\text{true}}^{\text{MC}}$ will provide an approximation (or measurement) of the target, i.e. $\vec{x}_{\text{true}}^{\text{data}}$. OmniFold and other likelihood-based methods do not change the observables themselves in the dataset — only the weights. This is in contrast to unfolding based on generative models that change both the dataset observables *and* the weights.

OmniFold makes frequent use of a useful feature of neural network binary classifiers: their learned decision functions can be used to construct a smooth reweighting function between the two datasets that participate in the binary classification. This property holds for a broad range of loss functions (see e.g. Ref. [20]), but the binary cross-entropy loss function is a popular choice. With this

Step 1 $\vec{x}_{\text{reco}}^{\text{MC}}$ $\vec{x}_{\text{reco}}^{\text{data}}$ $w_1(\vec{x}_{\text{reco}}^{\text{MC}})$ x

Step 2 $\vec{x}_{\text{true}}^{\text{MC}}$ $w_1(\vec{x}_{\text{true}}^{\text{MC}})$ Final iteration $w_2(\vec{x}_{\text{true}}^{\text{MC}})$ x

$\vec{x}_{\text{true}}^{\text{MC}}$ **OmniFold** measurement $w_2(\vec{x}_{\text{true}}^{\text{MC}})$ x

Repeat with $\vec{x}_{\text{reco}}^{\text{MC}} \to w_2(\vec{x}_{\text{reco}}^{\text{MC}})$
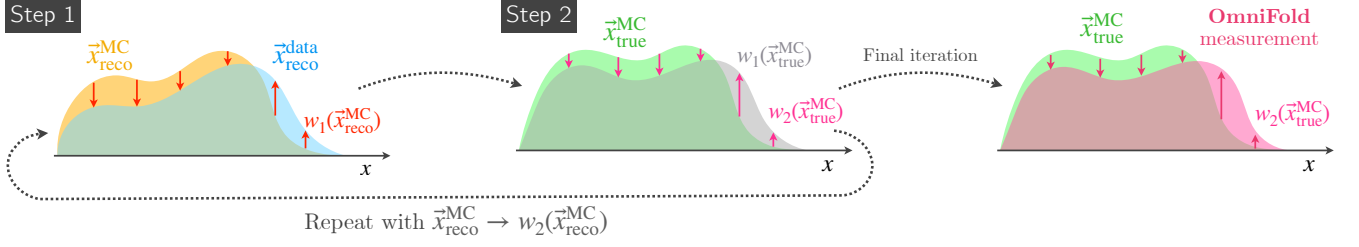
FIG. 1. Illustration of the OmniFold method. In Step 1, MC is corrected to match data at the detector level, producing a weighting function $w_1(\vec{x}_{\text{reco}}^{\text{MC}})$. In Step 2, a new function $w_2(\vec{x}_{\text{true}}^{\text{MC}})$ is learned based only on particle-level quantities. The method proceeds iteratively, refining the function $w_2(\vec{x}_{\text{true}}^{\text{MC}})$ such that the particle-level MC can be reweighted to give event yields and kinematics that match those observed in the data.

choice of loss function, a classifier trained to distinguish whether a set of events $\vec{x}$ are drawn from either probability distribution $p_A(\vec{x})$ or $p_B(\vec{x})$ will learn a function $f(\vec{x})$ that minimizes:

$$\mathcal{L}_{\text{BCE}}[f] = -\int d\vec{x} \left(p_A(\vec{x})\log(f(\vec{x})) + p_B(\vec{x})\log(1 - f(\vec{x}))\right)$$

The function that minimizes this expression will satisfy, for any small variation $\delta f(\vec{x})$:

$$\delta\mathcal{L}_{\text{BCE}} = \int d\vec{x} \left(-\frac{p_A(\vec{x})}{f(\vec{x})} + \frac{p_B(\vec{x})}{1 - f(\vec{x})}\right)\delta f(\vec{x}) = 0,$$

therefore the integrand must be identically zero:

$$-\frac{p_A(\vec{x})}{f(\vec{x})} + \frac{p_B(\vec{x})}{1 - f(\vec{x})} = 0.$$

Solving this equation, we find that we can reconstruct the likelihood ratio:

$$\frac{f(\vec{x})}{1 - f(\vec{x})} = \frac{p_A(\vec{x})}{p_B(\vec{x})},$$

This function of the neural network classifier $f(x)$ can then be used to reweight one sample to look like the other by adjusting the event weight of each data point appropriately. In practice, this method will asymptotically approximate the likelihood ratio $p_A(\vec{x})/p_B(\vec{x})$ [21, 22].

Like some binned unfolding methods that have been widely used in HEP for decades, OmniFold operates through multiple iterations. In fact, when applied to binned data, OmniFold is mathematically equivalent to Iterative Bayesian Unfolding [1]. Each iteration of Omni-Fold consists of two steps both of which involve estimating a reweighting function using the procedure described above:

- **Step 1:** First, a classifier is trained to distinguish between distributions of $\vec{x}_{\text{reco}}^{\text{MC}}$ and $\vec{x}_{\text{reco}}^{\text{data}}$. This classifier is used to construct a reweighting function $w_1(\vec{x}_{\text{reco}}^{\text{MC}})$ that can be applied to make $\vec{x}_{\text{reco}}^{\text{MC}}$ match $\vec{x}_{\text{reco}}^{\text{data}}$ at the detector level.

- **Step 2:** Next, a new classifier is trained to distinguish between distributions of $\vec{x}_{\text{true}}^{\text{MC}}$ and the adjusted events from the previous step taken at the particle level: $\vec{x}_{\text{true}}^{\text{MC}}$ weighted by $w_1(\vec{x}_{\text{reco}}^{\text{MC}})$. A new reweighting function $w_2(\vec{x}_{\text{true}}^{\text{MC}})$ is learned that can be applied to make $\vec{x}_{\text{true}}^{\text{MC}}$ statistically match the target $\vec{x}_{\text{true}}^{\text{data}}$ at the particle level.

These two steps are then repeated for multiple iterations, where the next iteration instead takes $\vec{x}_{\text{reco}}^{\text{MC}}$ weighted by $w_2(\vec{x}_{\text{true}}^{\text{MC}})$ as its input to compare with data. The updated reweighting function becomes a product of the previous one until a predefined number of iterations are performed and the method stops.

III. PRACTICAL CONSIDERATIONS

A. Experimental Overview

Table I summarizes the eleven analyses considered.

B. Hyperparameter Optimization

The unbinned unfolding process includes a number of hyperparameters that affect either the core unfolding methodology (e.g. number of unfolding iterations) or the neural network training (e.g. batch size). The OmniFold codebase [34] includes a suite of recommended default hyperparameters that were generally sufficient for some analyses. Other analyses chose to optimize hyperparameters based on the particular needs of their experiments. Those who did choose to perform a hyperparameter optimization generally tracked the performance of the model in unfolding with two different datasets of MC simulations.

1. Unfolding Hyperparameters

a. Number of iterations: OmniFold [19], like its binned analog IBU [1], is an iterative method. However,

Experiment	Paper Link	Dimensions	Final State	Momentum Selection
ATLAS [23]	2405.20041	24	Z+jets	$p_{\rm T}^{\ell\ell} > 200$ GeV
ATLAS [24]	2502.02062	6	Dijets	$p_{\rm T}^{j_1} > 240$ GeV & $p_{\rm T}^{j_1} < 1.5\ p_{\rm T}^{j_2}$
CMS [25]	2505.17850	8	Minimum bias	> 2 charged particles with $p_{\rm T} > 0.5$ GeV
H1 [26]	2108.12376	8*	High Q^2 DIS	$Q^2 > 150$ GeV^2
H1 [27]	2303.13620	10	High Q^2 DIS	$Q^2 > 150$ GeV^2
H1 [28]	2412.14092	8*	High Q^2 DIS	$Q^2 > 150$ GeV^2
H1 [29]	H1prelim-25-031	Variable	High Q^2 DIS	$Q^2 > 150$ GeV^2
LHCb [30]	2208.11691	4	Z+hadrons in jets	$20 < p_{\rm T}^{j} < 100$ GeV and $p_{\rm T}^{h} > 0.25$ GeV
STAR [31]	2307.07718	6	Jets	$20 < p_{\rm T}^{j} < 50$ GeV
STAR [32]	2403.13921	7	Jets in heavy ions	$20 < p_{\rm T}^{j} < 45$ GeV
T2K [33]	2504.06857	6	Muon + Proton	$p^p > 450$ MeV for single transverse variables

TABLE I. An overview of the recently-published experimental results that use unbinned unfolding methods. For most results, the unfolded dimensionality is the same at reconstruction-level and at truth-level, but * indicates the analyses for which the unfolded dimensionality at reconstruction-level was the full phase space, but the truth-level unfolded result is 8-dimensional. Full details of the phase space for each measurement, including η selections, are listed in the individual papers.

the number of iterations needed to see good performance can vary. To estimate this number, most analyses used dedicated studies with a known target to measure the convergence of the unfolded dataset with the target over a large number of iterations, sometimes reaching up to 70 or 100. In practice, far fewer iterations were used for the actual analysis result, typically around 5, due to limitations in the detector resolution. One notable exception is the T2K study [33], which preferred as many as 20 to 40 iterations. It also explored using per-event weight changes after each iteration to deal with the still open question of how to choose the appropriate number of iterations in an unbinned manner. The default choice in the OmniFold software is set to 3 iterations. Analyzers can also report the unfolding results for different numbers of iterations to assess the impact of this choice, as in Refs. [35, 36].

b. *Network initialization:* Most analyses chose to train the networks for each unfolding step from scratch with the onset of each new iteration, which is also the default choice in the OmniFold software. However, using a pre-trained network as a starting point for training subsequent steps can potentially reduce the overall training time and improve the stability of the result, particularly in cases for which there is a large imbalance in statistics between simulation and data [29], or where the available data statistics are just generally low [33].

c. *Step 2 task:* Step 2 of the unfolding procedure is designed to learn a function that maps the truth-level MC to the target of unfolded data. At each iteration, however, this task can be framed either as performing this mapping in full or in part. "In part", in this case, means learning to map from the previous iteration's reweighted MC to the target and then composing each of these mappings together after training to learn the full mapping from initial MC simulation to the target. The default behavior in OmniFold is to learn this mapping in full. In the CMS analysis [25], researchers found that learning the mapping from the previous iteration's reweighted MC sample yielded worse performance and larger variance in

the OmniFold weights, though this behavior has not yet been tested extensively.

2. Neural Network Hyperparameters

a. *Network architecture:* The classifiers across the various analyses took the form of dense neural networks with a small number of of hidden layers—typically three, although some analyses also explored architectures with two or four layers. The T2K study [33] found measurably worse performance with smaller networks. Hidden layers mostly used the `ReLU` activation function and the final output layer used a sigmoid activation. Hidden layers generally had a size of $\mathcal{O}(100)$, e.g. 100 or 200 nodes.

b. *Batch size:* Larger batch sizes of $\mathcal{O}(10^3)$ events are preferable in order to maximize GPU usage and improve stability of the results. The default recommendation is 128, but in practice, different analyses selected batch sizes ranging from 1,028 [25] to 50,000 [31].

c. *Number of training epochs:* Instead of using a fixed number of epochs, most analyses implemented early stopping using either a pre-determined patience of e.g. 10 epochs or a minimum Δ, i.e. a lower bound on the change in the loss that counts as an improvement to the model. Early stopping was the most common form of regularization used, but a few analyses also employed dropout and batch normalization.

C. Preprocessing

Data preprocessing can have major effects on the ultimate performance of a neural network training. Each analysis had to determine not only how to represent the features used as inputs, but also how to standardize the inputs and weights.

a. *Input features:* Many analyses used jet features represented as four-vectors, e.g. p_T, η, and ϕ. In some

cases, these features were represented relative to another reference frame such as the recoiling electron [29]. A few analyses chose to apply Graph Neural Networks (GNNs) in order to process the particles as 3D point clouds [27, 29]. The ATLAS Z+jets analysis [23] chose to represent ϕ coordinates as a combination of $\sin(\phi)$ and $\cos(\phi)$ after a dedicated study that suggested this representation was more effective at reducing discontinuities in the final results. In addition to raw input features, summary statistics such as moments and cumulants can also be successfully unfolded, even higher-order ones with relatively small values (e.g. $\mathcal{O}(10^{-5})$) [24]. Inputs were generally standardized using their z-scores, i.e. $z = (x - \mu)/\sigma$, before using them in the training process.

The T2K study [33] identified two outgoing particles for each event, the muon and proton, and used the angles and forward momentum of these particles as inputs. However, in events for which there was no outgoing proton — whether because it was not reconstructed or because no proton was kicked out at truth level — placeholder values were used for the proton kinematics. Additionally, this analysis separated events into different samples in detector space based on the locations within the detector that the particles were reconstructed, and separated them into different reaction topologies in truth space. This qualitative information was provided as one-hot encoded input to the neural network.

b. Negative weights: Simulated samples of high-energy physics data can sometimes be associated with MC weights representing wide magnitude ranges, even including negative values, which can introduce difficulties when using these weighted samples in training neural networks. In the ATLAS Z+jets measurement [23], several of the input MC samples suffered from a significant fraction of negative event weights and large spread of weights. For these samples, the original MC weight was replaced by a new weight following a procedure similar to the method described in Ref. [37]. The resulting samples had positive weights with a reduced spread (standard deviation), yielding datasets that are statistically compatible to those produced using the original weights.

c. Calculating the differential cross-section: Most analyses performed the unfolding task on normalized samples such that the neural networks learned the relative shape differences between the datasets. This choice allows for the use of standardization in the weights themselves as well as the input features. The measured differential cross-sections can then estimated by scaling the weights based on a calculation using the efficiency and ofiducial factor from the nominal MC sample as well as the fiducial cross-section and number of data events:

$$\frac{\mathcal{L} \cdot \sigma_{\text{fiducial}} \cdot \epsilon}{f_{\text{fiducial}}} = n_{\text{data}}.$$

These corrections are implicitly accounted for if the unfolding is done without normalizing the samples, as was done in the T2K study [33]. Alternatively, if the goal is to measure the shape of an observable, the measurement can be normalized again after unfolding such that the weighted area sums to 1.

d. Simultaneously unfolding event- and particle-level features: Unfolded features can potentially include both aggregate and individual characteristics of event constituents, e.g. a jet and its constituent particles. In the LHCb measurements [30], to account for this, an initial weight of unity for each particle was scaled down by the particle multiplicity within the jet to allow the network learn the substructure of jets and consistently normalize the synthetic data by the number of jets.

D. Background and Acceptance Effects

These analyses broadly benefited from a lack of major background sources that would require a detailed background subtraction process. However, their methodologies can in some cases be extended for analyses with nontrivial backgrounds. We note that "backgrounds" can comprise irreducible backgrounds, i.e. physics processes with the same final states as the signal, as well as reducible backgrounds, i.e. events that have been misclassified as signal events due to detector artifacts or other errors during particle reconstruction and identification.

a. Effects from irreducible backgrounds: Backgrounds were estimated to be small or negligible for the analyses covered in this report, and were generally either ignored or were treated by assigning an overall uncertainty. For example, the ATLAS Z+jets measurement [23] had a primary background of top events, but these were small enough ($< 0.25\%$ of total events) to simply subtract them from a set of pseudodata, run the full unfolding procedure, and then take the difference between this and the nominal result as an uncertainty. The CMS measurement [25] targeted minimum bias proton-proton collisions, and the fiducial space for the measurement is as inclusive as possible. In the STAR heavy ions jets analysis [32], backgrounds were modeled using an embedding simulation to remove residual effects. To mitigate any discrepancies between data and MC due to choice of embedding strategy, an extra weight was then multiplied to the reco-level jets to make their multiplicity and luminosity distributions look similar to real minimum-bias data. For future users dealing with nontrivial irreducible backgrounds, we recommend incorporating these backgrounds into the initial MC dataset as MC events with negative weights such that in total, the initial MC dataset corresponds to data with background subtracted.

b. Acceptance and reducible background effects: These effects emerge due to events that pass the truth-level selections but not reconstruction-level selections, or vice-versa. Generally, the former is the dominant acceptance effect due to tracking or trigger inefficiencies. The default OmniFold [19] procedure handles these events by assigning them the average weight in their region of

phase space, which has been found to yield better convergence than directly including them in the unfolding procedure [33]. Unfolding a larger fiducial region than the one reported in the final measurement can also help mitigate acceptance effects—for instance, the ATLAS Z+jets measurement [23] measures the phase space $p_T^{\ell\ell} > 190$ GeV, but only reports results with $p_T^{\ell\ell} > 200$ GeV in order to mitigate acceptance effects due to migrations across this threshold.

In the LHCb measurements of charged particles in jets [30], the track reconstruction efficiencies along with particle identification corrections were applied to each reconstructed particle in data being unfolded based on a look-up table prepared prior to unfolding.

In the STAR measurement of jet substructure variables [31], this correction is instead done after the unfolding, and involves applying an efficiency correction by dividing the normalized unfolded dataset over the efficiency function bin-by-bin. Backgrounds due to misidentified jets were estimated using simulations and were used to construct initial weights for the data by subtracting the fake rates [31].

The minor background from the minimum-bias CMS measurement [25] came from fake tracks induced by detector noise or misidentification of the algorithms, which resulted in events that passed the detector-level selection but failed the truth-level selection. Acceptance effects came from tracking inefficiency corresponding to events that passed the truth-level selection but failed the detector-level selection. Both effects were taken into account by adding two steps[1] to each iteration of the unfolding, as described in Ref. [7].

The T2K study [33] dealt with both significant inefficiencies from acceptance effects and significant backgrounds, which are common features of measurements in neutrino detectors due to their large volumes and relatively low fidelity. The backgrounds and effects from non-uniform detector acceptance were dealt with by splitting the detector-level events into distinct samples based on the topology with which they were reconstructed in the detector, including background-enhanced samples to constrain background contributions to the overall signal. This qualitative sample information was included as additional input to the neural network used in the unfolding. Additionally, results on single transverse variables

were only reported for events with proton momentum > 450 MeV, but the unfolding was performed over the entire available phase space without this restriction.

E. Ensembling

Most analyses found that ensembling improved the stability of the unfolded result. "Ensembling" refers to training multiple independent instances of the same model and then using the mean or median weight. In principle, this ensembling could occur at the level of binned observables, but ensembling the weights themselves facilitates publishing a fully unbinned dataset that does not depend on any particular choice of binning. The standard error on this quantity across the trained models is then included as a source of uncertainty. This uncertainty can be thought of as measuring the impact of the stochastic nature of the neural network training — i.e. choice of random seed — on the final unfolded result. The magnitude of this uncertainty is generally small ($< 2\%$). Most analyses used some form of ensembling, with numbers of ensembles typically ranging from 4 to 10. Both the ATLAS Z+jets measurement [23] and the ATLAS jet track functions measurement [24] used an ensemble of 100 models, which helped to mitigate the compounding effects of these statistical fluctuations over a large number of uncertainties. Ensembling was usually performed by re-running the entire result, but as an alternative, the ensembling could instead be done independently for Step 1 and Step 2 of each unfolding iteration [38].

F. Uncertainties

Despite the introduction of neural networks underlying the unfolding procedures in these analyses, the procedure of estimating uncertainties is, with a few exceptions, relatively standard for particle physics measurements. These uncertainty estimation strategies include bootstrapping for estimating statistical uncertainties and calculating the effects of varying the MC sample for estimating systematic uncertainties. One new uncertainty included in these measurements was an uncertainty associated with the neural network initialization, i.e. the inherent variation in repeating the neural network training with a different random seed but otherwise an identical configuration. These variations are typically small, but they affect every other reported uncertainty, so they are important to constrain.

1. ATLAS unfolding uncertainty

In the ATLAS Z+jets measurement, the unfolding uncertainty consisted of two parts: an uncertainty due to the truth-level prior as well as an uncertainty due to the potential mismodeling of the detector effects [23]. These

---

[1] After Step 1, a Step 1b is added in which the weighting function $w_1(\vec{x}_{\mathrm{reco}}^{\mathrm{MC}})$ is pulled back to the truth-level $w_{1\mathrm{b}}(\vec{x}_{\mathrm{true}}^{\mathrm{MC}})$ to correct the detector acceptance. $w_{1\mathrm{b}}$ is derived by training a classifier at the truth level and can weight the original MC events passing the selections on both truth and detector levels to the same events with weights from Step 1. $w_{1\mathrm{b}}(\vec{x}_{\mathrm{true}}^{\mathrm{MC}})$ is applied to MC events not passing the truth-level selection. After Step 2, a Step 2b is added, in which the weighting function $w_2(\vec{x}_{\mathrm{true}}^{\mathrm{MC}})$ is pushed to the detector-level $w_{2\mathrm{b}}(\vec{x}_{\mathrm{reco}}^{\mathrm{MC}})$ to correct the detector background. $w_{2\mathrm{b}}$ is from training a classifier at the detector level and weights the original MC events passing both selections to the same events weighted by Step 2. Then $w_{2\mathrm{b}}(\vec{x}_{\mathrm{reco}}^{\mathrm{MC}})$ is applied to MC events not passing the detector-level selection.

effects were estimated separately and reported as two separate sources of unfolding uncertainty. Together, the unfolding uncertainty represented the dominant uncertainty across most of the measured phase space.

The first component of the ATLAS unfolding uncertainty, which measures the sensitivity of the result to the modeling of the particle-level sample, is estimated via a data-driven process: the MC is reweighted at truth-level such that all of its reco-level observables agree with data, and the result of unfolding using this dataset is compared with the nominal unfolding result. Crucially, the reweighting function used here is a sequence of one-dimensional Gaussian-kernel functions, not the OmniFold procedure that is itself being probed. The second component of the ATLAS unfolding uncertainty, which measures the sensitivity of the result to potential mismodeling of the detector response for features not included in the unfolding, is sometimes called a "hidden variable" uncertainty. To estimate this component, a MC sample from a different generator is used in the unfolding, but it is first reweighted at truth-level to match the MC sample from the original generator as closely as possible.

2. CMS unfolding uncertainty

In the CMS measurement [25], uncertainty templates are implemented as weighting functions applied to nominal MC events. These functions are learned by classifiers trained to distinguish nominal and alternative MC samples, either at truth- or reco-level, depending on the source of the uncertainty. The templates for the CMS unfolding bias, which originate from the mismodeled truth-level prior on the target variables to be unfolded, are constructed as weighting functions $w_\alpha(\vec{x}_{\text{true}}^{\text{MC}})$ at the truth level from the nominal MC sample to the MC sample from each alternative model α. When applied to the nominal MC events, $w_\alpha(\vec{x}_{\text{true}}^{\text{MC}})$ makes their truth-level samples match the ones from the alternative model. Similarly, the templates for mismodeled detector responses, which come from the discrepancies between the reconstruction in the simulation and truth, are constructed as weighting functions $w_{\text{det}}(\vec{x}_{\text{reco}}^{\text{MC}})$ at the detector level, mapping the nominal MC sample to the MC sample from the same truth-level model but alternative detector simulation.

A two-stage reweighting scheme handles mismodeled priors in kinematic variables not directly unfolded. First, a truth-level weighting $v_\alpha^1(\vec{x}_{\text{true}}^{\text{MC}})$ is applied to MC events from the alternative model α to match their truth-level samples to the ones from the nominal model. Then, a joint weighting of both truth and detector level, $v_\alpha(\vec{x}_{\text{true}}^{\text{MC}}, \vec{x}_{\text{reco}}^{\text{MC}})$, is applied to the nominal MC events to match their samples at both levels to the ones from the alternative samples weighted by $v_\alpha^1(\vec{x}_{\text{true}}^{\text{MC}})$. The resulting weighting function keeps the truth-level of the nominal sample while mapping its detector response to that of the alternative model. These reweighted templates al-

low smooth interpolation between models via nuisance parameters and enable uncertainty estimation via bootstrapping.

G. Validation

Many analyses were performed "blinded", i.e. not considering the real data while configuring the analysis parameters. This meant using a different MC sample in place of the real data while developing the analysis, and then only using the real data after the analysis passed a number of predetermined validation criteria. Even if the analysis itself was not formally blinded, all analyses validated their unfolding procedure using alternate MC samples.

a. Pseudodata: The ATLAS Z+jets analysis [23] constructed a realistic set of "pseudodata" in place of the actual data to validate their procedure before obtaining results using the data. The pseudodata has a known underlying dataset $\vec{x}_{\text{true}}$, so the measured OmniFold result of any quantity can be compared to the desired "target" value. To produce the pseudodata, alternate signal and background MC samples were reweighted to match data using a dedicated reweighting algorithm. The full analysis was performed using this pseudodataset, and the unfolded results were evaluated by analyzing the p-value distributions between the unfolded result and the target truth pseudodata. The agreement was checked using not only the 24 input observables, but also on some select composite observables (i.e. observables that can be calculated using the 24 but are not included in the unfolding), for a series of 2D distributions and for a series of phase space cuts.

The CMS analysis [25] used a few sets of "pseudodata" from detector simulations of alternative models that are not used in unfolding or uncertainty estimation. The pseudodata sets are generated from distinct generators, parton distribution functions, or underlying-event tunes. The unfolding results of pseudodata sets are compared with their corresponding truth-level samples. The agreements are used for hyperparameter optimization and have been confirmed for all the variables to be unfolded and their multi-dimensional distributions.

b. Validation: Technical closure tests and stress tests served to validate the unfolding methodology itself using different sets of MC samples. In the ATLAS Z+jets measurement, the stress tests were performed using a single MC sample to unfold itself, but the half used for data had additional stress weights applied. Two tests were performed: one where the weights were based on a deterministic function of the observables, and another where the weights were based on a stochastic function of the observables. In the T2K study [33], the test was performed using a set of fake data with reweightings that were functions of non-observable parameters. This resulted in the unfolding needing to learn a reweighting that is not a simple function of the quantities made available to it –

a type of stress test that is frequently done for neutrino cross-section measurements.

In the CMS result, bias and coverage tests were performed to evaluate the bias of the unfolded results compared to the truth and to validate the coverage of the unfolding uncertainties. The tests were based on unfolding pseudodata using a frequentist framework. Multiple toy experiments of unfolding were performed, in which both the MC and pseudodata samples were randomly deviated from the original ones according to the considered uncertainties. The cumulative behaviors of these toy experiments indicate the fluctuations if the measurement is repeated several times. The bias was evaluated as the deviations between the central values from the toy experiments and the pseudodata truth, and the coverage was estimated as the frequencies that the toy experiment results are compatible with the pseudodata truth, considering the uncertainties.

Furthermore, the CMS analysis performed a bottom-line test to compare the differences between the unfolded results and the truth-level nominal MC sample to the ones between the (pseudo)data and the detector-level nominal MC sample. Since the unfolded results cannot be more discriminative to the models than the detector-level data, the former should always be smaller than the latter ones. A failure of the bottom-line test could come from the algorithm attributing the data-MC discrepancies from statistical fluctuations to the truth-level mismodeling and overfitting the given samples.

H. Presentation of Results

Some community-sourced guidelines have already proposed how to best present unbinned measurements [3]. Across all of the results presented in this paper, however, though the unfolding procedure itself is performed unbinned, the unfolded results are presented in a binned histogram format for all but the ATLAS Z+jets measurement, which was presented in an unbinned manner.

a. Binned format: Measurements are presented as individual histograms corresponding to each unfolded dimension. The specifications and bin counts of these histograms can then be uploaded to HEPData.

b. Unbinned format: In the ATLAS Z+jets measurement [23], the final result is presented as a set of Pandas DataFrames in which each DataFrame row corresponds with a truth-level Monte Carlo event, inspired by the example set by Ref. [36] using CMS Open Data. More than one DataFrame is needed for the full ATLAS result in order to account for all sources of systematic uncertainty. Columns correspond with the 24 unfolded dimensions, i.e. truth-level particle quantities, along with hundreds of weights used to construct the nominal measurement and calculate uncertainties. These datasets are published on Zenodo [39] and are paired with a detailed codebase illustrating how to use them to replicate the results in the paper as well as how to produce new mea-

surements (see Figure 2). This presentation format was closely informed by the recent white paper that discusses the presentation of unbinned results [3]. The CMS results will also be released in a similar unbinned format.

I. Computing Requirements

Across each of these measurements, an individual unfolding procedure could, in most cases, be run on a single NVIDIA A100 GPU with 40 GB or 80 GB of RAM in $\mathcal{O}(1\ \text{hour})$, generally between 1 and 4 hours. The main factors influencing this training time include dimensionality of the unfolding, number of trainable model parameters, and number of unfolding iterations. If ensembling is used, identical copies of the unfolding setup are run up to $\mathcal{O}(100)$ times in order to reduce the uncertainty due to the stochastic nature of the training process. These ensembling runs are able to be fully trained in parallel, if compute availability permits.

A full physics measurement, however, includes the calculation of a number of sources of uncertainty, and the number of uncertainties included can directly scale the computational resources required. Computational complexity for systematic uncertainties depends on the uncertainty sources and estimation strategies, requiring $\mathcal{O}(100)$ unfolding runs if bootstrapping is used. Statistical uncertainties estimated through bootstrapping can also require about 50 to 100 individual bootstrap runs.

In total, most analyses reporting significant computational needs for the unfolding portion of their experiments estimated that they used between approximately 500 and 10,000 GPU hours in total to perform their measurement once the full procedure had been fixed. Necessary computational resources depended heavily on the number of models used for ensembling and ranged from the STAR heavy ions analysis [32], which did not use any ensembling and was able to run on a single GPU, to the 24-dimensional ATLAS Z+jets measurement [23], which used an ensemble of 100 models to calculate each weight and took approximately 25,000 GPU hours in total.

IV. CONCLUSIONS AND OUTLOOK

The recent emergence of more than 10 different high-energy physics publications using unbinned unfolding methods underscores that these methodologies are undoubtedly publication-ready. These results cover a variety of final states, six different experimental detectors, a range of unfolded dimensions (from four to 24 to the full phase space). Importantly, these methods have also been tested and validated in a number of different scenarios and have yielded unbinned cross-section measurements using large-scale physics datasets and full sets of uncertainties. As this field continues to expand, we expect that many more analyses will opt to both perform and publish their measurements in an unbinned fashion using
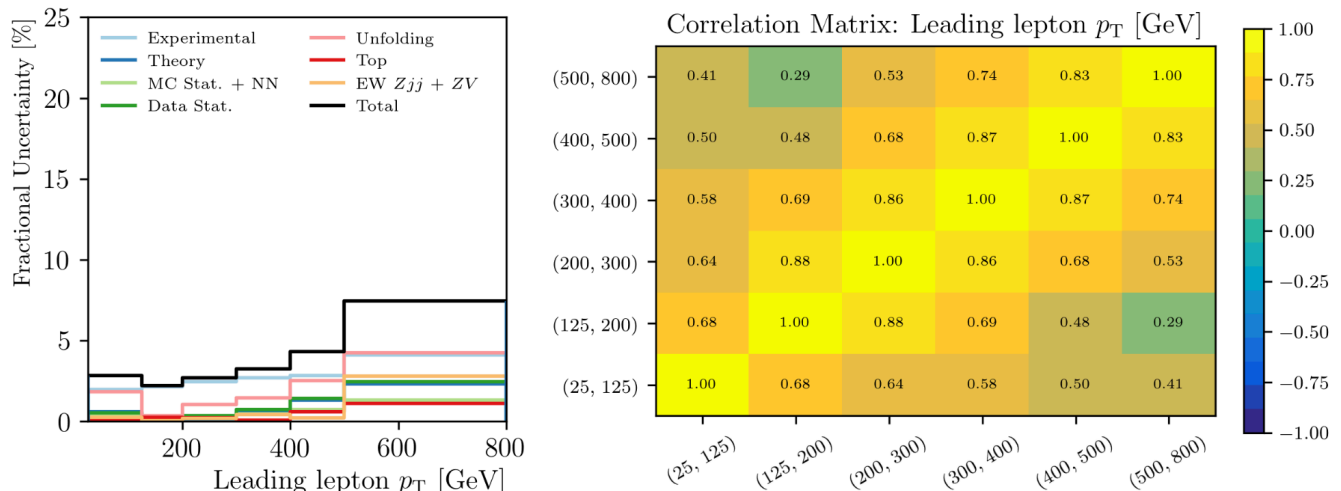
FIG. 2. An example of the documentation from the ATLAS Z+jets measurement [23] shows how to use the unbinned data to configure plots of not only the unfolded result, but also the uncertainty breakdown (left) and correlation matrix (right) using bins of one's choice for a given observable such as leading lepton p_T.

the frameworks these first publications have established as a guide.

A. Future Directions

While unbinned unfolding has now been proven effective for a range of particle physics applications, there remain several interesting directions for the development of future work in this area, including:

- **Pre-trained models:** Would fine-tuning pre-trained models instead of training from scratch each time help improve the stability of the result as well as save computational resources, and should this become the new default methodology? (See Ref. [38] for some additional discussion.)

- **Statistical uncertainties:** Are there viable alternatives to bootstrapping, to reduce the computational requirements needed in calculating statistical uncertainties? How can we best capture the statistical uncertainty not only in terms of the limitation of the total number of data events, but also in terms of the resulting instability in the neural network training?

- **Unfolding after many iterations:** In some cases, unfolding performance suddenly plummeted after many ($\mathcal{O}(100)$) iterations. Though this behavior did not affect the quality of the results ultimately presented here, which all relied on small numbers of iterations (5 or fewer), this behavior could merit future study.

- **Unfolding the whole phase space:** While the methodology for unfolding variable-length events is well-established in theory, we are only recently starting to see examples of this put into practice (e.g. Ref. [29]). To what extent do these recommendations apply to unfolding the whole phase space? Some researchers observed that when unfolding the whole phase space with datasets that were too similar, the unfolding performance could oscillate with each iteration — what is responsible for this behavior, and how could it be mitigated?

- **Goodness-of-fit tests for unbinned data:** How can we best measure closure with a known target in an unbinned manner, therefore bypassing the typical χ^2 and p-value binned statistical tests? Existing metrics include the Wasserstein distance [40] and permutation tests [41], among others, but many open questions still remain in this area.

- **Generative methods:** Each of these measurements has employed a version of OmniFold [7, 19]. Could other ML-based unbinned unfolding methods, including generative methods, yield competitive results?

- **Nontrivial background subtraction:** While there are proposed methods for putting nontrivial background estimation into practice, there is a strong need for researchers to put these methods to the test and explore which are most effective.

[1] G. D'Agostini, A multidimensional unfolding method based on Bayes' theorem, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **362**, 487 (1995).

[2] A. Glazov, Machine learning as an instrument for data unfolding, (2017), arXiv:1712.01814 [physics.data-an].

[3] M. Arratia et al., Presenting Unbinned Differential Cross Section Results, (2021), arXiv:2109.13243 [hep-ph].

[4] M. Bunse, N. Piatkowski, T. Ruhe, W. Rhode, and K. Morik, Unification of deconvolution algorithms for Cherenkov astronomy, in *5th International Conference on Data Science and Advanced Analytics (DSAA)* (IEEE, 2018) pp. 21–30.

[5] T. Ruhe, T. Voigt, M. Wornowizki, M. Borner, W. Rhode, and K. Morik, Mining for Spectra - The Dortmund Spectrum Estimation Algorithm, in *Astronomical Data Analysis Software and Systems XXVI*, Astronomical Society of the Pacific Conference Series, Vol. 521 (2019) p. 394.

[6] A. Andreassen, P. T. Komiske, E. M. Metodiev, B. Nachman, and J. Thaler, OmniFold: A Method to Simultaneously Unfold All Observables, Phys. Rev. Lett. **124**, 182001 (2020), arXiv:1911.09107 [hep-ph].

[7] A. Andreassen, P. T. Komiske, E. M. Metodiev, B. Nachman, A. Suresh, and J. Thaler, Scaffolding Simulations with Deep Learning for High-dimensional Deconvolution, (2021), arXiv:2105.04448 [stat.ML].

[8] M. Arratia, D. Britzger, O. Long, and B. Nachman, Optimizing observables with machine learning for better unfolding, JINST **17** (07), P07009, arXiv:2203.16722 [hep-ex].

[9] J. Chan and B. Nachman, Unbinned profiled unfolding, Phys. Rev. D **108**, 016002 (2023), arXiv:2302.05390 [hep-ph].

[10] K. Datta, D. Kar, and D. Roy, Unfolding with Generative Adversarial Networks, (2018), arXiv:1806.00433 [physics.data-an].

[11] M. Bellagente, A. Butter, G. Kasieczka, T. Plehn, and R. Winterhalder, How to GAN away Detector Effects 10.21468/SciPostPhys.8.4.070 (2019), arXiv:1912.00477 [hep-ph].

[12] M. Bellagente, A. Butter, G. Kasieczka, T. Plehn, A. Rousselot, R. Winterhalder, L. Ardizzone, and U. Köthe, Invertible Networks or Partons to Detector and Back Again, SciPost Phys. **9**, 074 (2020), arXiv:2006.06685 [hep-ph].

[13] M. Vandegar, M. Kagan, A. Wehenkel, and G. Louppe, Neural Empirical Bayes: Source Distribution Estimation and its Applications to Simulation-Based Inference, in *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, Vol. 130, edited by A. Banerjee and K. Fukumizu (PMLR, 2021) pp. 2107–2115, arXiv:2011.05836 [stat.ML].

[14] J. N. Howard, S. Mandt, D. Whiteson, and Y. Yang, Foundations of a Fast, Data-Driven, Machine-Learned Simulator, (2021), arXiv:2101.08944 [hep-ph].

[15] M. Backes, A. Butter, M. Dunford, and B. Malaescu, An unfolding method based on conditional Invertible Neural Networks (cINN) using iterative training, (2022), arXiv:2212.08674 [hep-ph].

[16] A. Shmakov, K. Greif, M. Fenton, A. Ghosh, P. Baldi, and D. Whiteson, End-To-End Latent Variational Diffusion Models for Inverse Problems in High Energy Physics, (2023), arXiv:2305.10399 [hep-ex].

[17] T. Alghamdi et al., Toward a generative modeling analysis of CLAS exclusive 2π photoproduction, (2023), arXiv:2307.04450 [hep-ph].

[18] S. Diefenbacher, G.-H. Liu, V. Mikuni, B. Nachman, and W. Nie, Improving Generative Model-based Unfolding with Schrödinger Bridges, (2023), arXiv:2308.12351 [hep-ph].

[19] A. Andreassen, P. T. Komiske, E. M. Metodiev, B. Nachman, and J. Thaler, OmniFold: A Method to Simultaneously Unfold All Observables, Phys. Rev. Lett. **124**, 182001 (2020).

[20] S. Rizvi, M. Pettee, and B. Nachman, Learning Likelihood Ratios with Neural Network Classifiers, Journal of High Energy Physics **2024**, 10.1007/jhep02(2024)136 (2024).

[21] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer Series in Statistics (Springer New York Inc., New York, NY, USA, 2001).

[22] M. Sugiyama, T. Suzuki, and T. Kanamori, *Density Ratio Estimation in Machine Learning* (Cambridge University Press, Cambridge, UK, 2012).

[23] ATLAS Collaboration, Simultaneous Unbinned Differential Cross-Section Measurement of Twenty-Four Z+jets Kinematic Observables with the ATLAS Detector, Phys. Rev. Lett. **133**, 261803 (2024).

[24] ATLAS Collaboration, Measurement of jet track functions in pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector (2025), arXiv:2502.02062 [hep-ex].

[25] CMS Collaboration, *Measurement of event shapes in minimum bias events from* pp *collisions at* 13 TeV, Tech. Rep. (CERN, Geneva, 2024).

[26] H1 Collaboration, Measurement of Lepton-Jet Correlation in Deep-Inelastic Scattering with the H1 Detector Using Machine Learning for Unfolding, Phys. Rev. Lett. **128**, 132002 (2022).

[27] H1 Collaboration, Unbinned Deep Learning Jet Substructure Measurement in High Q^2 ep collisions at HERA, Physics Letters B **844**, 138101 (2023), arXiv:2303.13620 [hep-ex].

[28] H1 Collaboration, Machine Learning-Assisted Measurement of Lepton-Jet Azimuthal Angular Asymmetries in Deep-Inelastic Scattering at HERA (2024), arXiv:2412.14092 [hep-ex].

[29] H1 Collaboration, Towards Unfolding All Particles in High Q^2 DIS Events, https://www-h1.desy.de/h1/www/publications/htmlsplit/H1prelim-25-031.long.html (2025), preliminary result.

[30] LHCb Collaboration, Multidifferential study of identified charged hadron distributions in Z-tagged jets in proton-proton collisions at $\sqrt{s} = 13$ TeV, Phys. Rev. D **108**, L031103 (2023).

[31] Y. Song, Measurement of CollinearDrop jet mass and its correlation with SoftDrop groomed jet substructure observables in $\sqrt{s} = 200$ GeV pp collisions by STAR, https://arxiv.org/abs/2307.07718 (2023), arXiv:2307.07718 [nucl-ex].

[32] T. Pani, Generalized angularities measurements from STAR at $\sqrt{s_{NN}} = 200$ GeV, https://arxiv.org/abs/2403.13921 (2024), arXiv:2403.13921 [nucl-ex].

[33] R. G. Huang, A. Cudd, M. Kawaue, T. Kikawa, B. Nachman, V. Mikuni, and C. Wilkinson, Machine learning assisted unfolding for neutrino cross-section measurements with the OmniFold technique, Phys. Rev. D **112**, 012008 (2025), arXiv:2504.06857 [hep-ex].

[34] V. Mikuni, OmniFold Codebase, https://github.com/ViciniusMikuni/omnifold (2020).

[35] P. T. Komiske, S. Kryhin, and J. Thaler, Disentangling quarks and gluons in CMS open data, Physical Review D **106**, 10.1103/physrevd.106.094021 (2022).

[36] P. Komiske, S. Kryhin, and J. Thaler, OmniFold Weights — CMS 2011A Open Data — Jet Primary Dataset — pT 375-700 GeV , 10.5281/zenodo.6519307 (2022).

[37] B. Nachman and J. Thaler, Neural resampler for Monte Carlo reweighting with preserved uncertainties, Physical Review D **102**, 10.1103/physrevd.102.076004 (2020).

[38] F. T. Acosta, T. Wamorkar, V. Mikuni, and B. Nachman, Stabilizing Neural Likelihood Ratio Estimation, https://arxiv.org/abs/2503.20753 (2025), arXiv:2503.20753 [hep-ph].

[39] ATLAS Collaboration, ATLAS OmniFold 24-Dimensional Z+jets Open Data, 10.5281/zenodo.11507450 (2024).

[40] R. L. Dobrushin, Prescribing a system of random variables by conditional distributions, Theory of Probability & Its Applications **15**, 458 (1970), https://doi.org/10.1137/1115049.

[41] M. Williams, How good are your fits? Unbinned multivariate goodness-of-fit tests in high energy physics, JINST **5**, P09004, arXiv:1006.3019 [hep-ex].