Deconvolving Detector Effects for Distribution Moments

Krish Desai
Department of Physics
University of California, Berkeley
Berkeley, CA 94720
krish.desai@berkeley.edu

Benjamin Nachman
Physics Division
Lawrence Berkeley National Laboratory
Berkeley, CA 94720
bpnachman@lbl.gov

Jesse Thaler
Center for Theoretical Physics, Massachusetts Institute of Technology
The NSF AI Institute for Artificial Intelligence and Fundamental Interactions
Cambridge, MA 02139
jthaler@mit.edu

Abstract

Deconvolving ('unfolding') detector distortions is a critical step in the comparison of cross section measurements with theoretical predictions in particle and nuclear physics. However, most extant unfolding approaches require histogram binning while many theoretical predictions are at the level of moments. We develop a new approach to directly unfold distribution moments as a function of other observables without having to first discretize the data. Our Moment Unfolding technique uses machine learning and is inspired by Generative Adversarial Networks (GANs). We demonstrate the performance of this approach using jet substructure measurements in collider physics.

1 Introduction

Studying the dependence of the statistical moments of physical observables on various quantities like the energy scale offers a rich probe into complex scaling relationships essential to the dyanamics of fundamental physical theories. Summarizing a probability distribution with a small number of moments makes visualization and interpretation tractable and lends itself to more accurate theoretical predictions. For example, the full probability densities of hadronic jets cannot be computed from first principles in perturbative quantum chromodynamics (QCD), but the energy dependence of their moments can be accurately predicted.

Unfolding (also known as *deconvolution*) is the process of correcting detector distortions in experimental data, and is necessary for accurately comparing data between experiments and with theoretical predictions. Typically, entire spectra are unfolded and then moments are computed afterward. In order to capture the dependence of an observable X's moments on another quantity Y, the two features must be simultaneously unfolded. Current unfolding approaches (see e.g. Ref. [1]) discretize the (X, Y) support and then unfold the two-dimensional histogram. Through this procedure, the moments of X can be computed in bins of Y. This binning procedure, however, introduces discretization artifacts.

One possible solution is to unfold without binning. A number of unbinned unfolding methods have been proposed [2–7], including approaches that are powered by machine learning [4–7]. First measurements [8–10] and studies [11] with collider data show that unbinned unfolding is also possible in practice. By construction, these approaches do not introduce binning artifacts. However, these methods offer a generic solution to unfolding entire spectra and therefore may compromise precision for any particular aspect of the spectrum, such as a small set of moments. We introduce a dedicated machine learning-based unfolding method to directly unfold the observable moments. This *moment unfolding* technique is motivated by the Boltzman equation from statistical mechanics and uses a structure that is similar to that of a Generative Adversarial Network (GAN) [12]. In particular, we learn a reweighting function (the analog of the generator) whose form is inspired by the Boltzman equation so that its parameters can be identified with the observable moments. This function is optimized by requiring that the reweighted simulation be as similar as possible to the target data (determined by a discriminator). The approach is similar to the `OmniFold` method [5] in that it is based on reweighting simulation, but it is fundamentally different because it is not iterative. That is to say, in methods such as `Omnifold` a distinct pair of neural networks is trained each time, "on each iteration," whereas in the proposed Moment Unfolding method, one pair of neural networks is trained, only once. We restrict our attention here to a small number of moments. In principle, this approach could be extended to full distributions, but there are complications due to the partition function that we leave for future studies.

2 Method

We consider the situation common in particle and nuclear physics where inference is enabled with a synthetic dataset that includes an emulated detector. Each synthetic collision event comes as a pair (X_T, X_D), for $X_T \in \mathbb{R}^{N_T}$ and $X_D \in \mathbb{R}^{N_D}$, where X_T is the pre-detector version of the event ('generation') and X_D is the post-detector observation of the event ('simulation'). In experimental data, we only have access to the detector-level version ('data') so we use the simulation to infer the pre-detector *truth*.

To achieve this goal, we propose an unbinned, non–iterative, reweighting based method to unfold the statistical moments of observables inspired by Boltzmann's approach used to construct the Maxwell–Boltzmann distribution [13]. The Maxwell–Boltzmann distribution is one which maximizes the entropy of an ensemble while holding mean energy constant. Similarly, our method aims to construct a distribution which maximizes the cross entropy with a target distribution while matching a fixed set of moments. The proposed generator then, in analogy to the Boltzman factor is

$$g(x) = e^{\lambda_1 x + \cdots + \lambda_n x^n}, \tag{1}$$

where n is the number of moments being computed. The moments of the resultant distribution are linear combinations of λ_i. The weights of this generator are updated until it can reweight the moments of the generation dataset so that the corresponding simulation dataset is statistically indistinguishable from the observed data. This method is schematically represented in Fig. 1.

2.1 Machine Learning Implementation

To implement the Moment Unfolding procedure, we modify the learning setup of a GAN [12]. In a typical GAN, the generator g surjects a latent space onto a data space. Then a discriminator, d, distinguishes generated examples from real examples. In this case, the latent space probability density is the simulation density, as illustrated in Fig. 1. These two neural networks are then trained simultaneously to optimize the binary cross entropy loss functional, where the generator tries to maximize the loss with respect to g and the discriminator tries to minimize the loss with respect to d. The weighted binary cross entropy loss functional is

$$L[g, d] = -\frac{1}{N} \sum_{x_{\text{data}}} \Big[\log\big(d(x_{\text{data}})\big) + \sum_{(x_{\text{gen}}, x_{\text{sim}})} g(x_{\text{gen}}) \log\big(1 - d(x_{\text{sim}})\big) \Big], \tag{2}$$

where x_{data} are the detector-level data examples, x_{sim} are the detection level simulation examples, and x_{gen} are the particle-level simulation (generation) examples. For the empirical studies here, all neural networks are implemented using the KERAS [14] high-level API with the TENSORFLOW2 backend [15] and optimized with ADAM [16]. The generator function g is parametrized as the exponential of a polynomial, as described in Eq. (1) above. It is a relatively simple 'neural network' with only n trainable parameters, $\{\lambda_i\}_{i=1}^{n}$ The discriminator function d is parametrized with three hidden layers, using 50 nodes per layer. Rectified Linear Unit (ReLU) activation functions are used for the intermediate layers and a sigmoid function is used for the last layer.

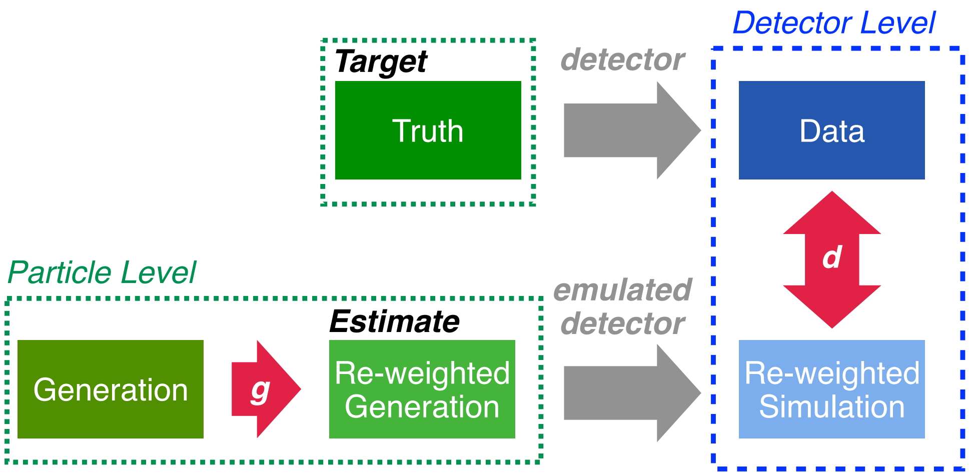
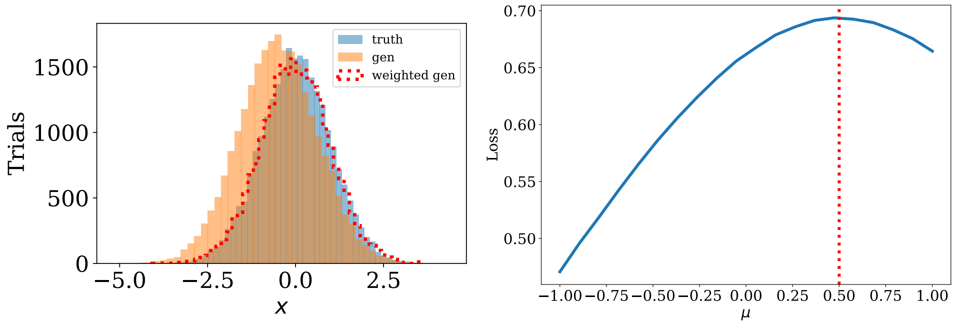


Figure 1: A schematic diagram of the training setup for the moment unfolding discussed in this paper for automatically unfolding statistical moments. Here, g is the generator and d is the discriminator. The re-weighted simulation dataset inherits its weight from the matching generation dataset. The detector emulations are only run once since a new simulated dataset is created via importance weights and not by changing the features themselves.

3 Case Studies

3.1 Gaussian Example

We begin by attempting to unfold Gaussian data with Gaussian distortions. The truth is drawn from $\mathcal{N}(0, 1)$ and the generation is drawn from $\mathcal{N}(-0.5, 1)$. Detector effects are represented by additive noise that is also Gaussian with distribution $\mathcal{N}(0, 5)$. 10^6 samples are used, with a train–test split of $3 : 1$. Since for Gaussian examples there are only finitely many moments to unfold, moment unfolding is in fact equivalent to unfolding the entire probability density. Figure 2a shows the numerical results of this unfolding procedure. The success of this procedure may be verified by observing that the maximum of the loss function is at the true mean, as shown in Fig. 2b. The training is relatively short, discriminator converges well within 10 epochs of training.



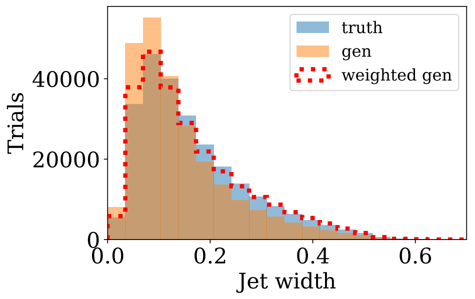
(a) (b)

Figure 2: (a) Histograms of truth, generation, and the reweighted generation for the Gaussian example. (b) The binary cross entropy loss Eq. (2) for a fixed value of g given by the mean value shown on the horizontal axis. The correct value is indicated by a vertical red dotted line.

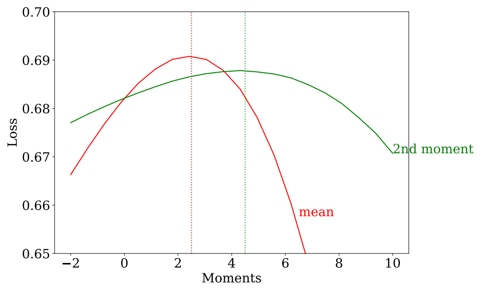
3.2 Jet Substructure

In this section, we study hadronic jets from simulations of the Large Hadron Collider (LHC) published in Ref. [5], including detector effects [17]. In order to have access to the truth, both the 'real' and 'synthetic' datasets are from simulations. One of these simulations plays the role of 'data' and one serves as the synthetic dataset. For simplicity, we consider a one-dimensional case using the jet width w [18] defined as $w = \frac{1}{p_{T,\text{jet}}} \sum p_{T,i} \Delta R_i$, where the sum runs over all the constituents of a jet and $p_{T,i}$, and ΔR_i are the transverse momentum and angular distance from the jet axis, respectively. Although we focus on the jet width observable, this method can be applied to other jet observables as well and can readily be conditioned on kinematic properties such as the jet momentum.

In this experiment we attempt to unfold two moments of detector distorted jet substructure data. The results are presented in Fig. 3a with the corresponding loss function scan shown in Fig. 3b. Radiation within jets is enhanced at low values of ΔR which leads to the uni-modal distribution of the jet width that falls off rapidly after about 0.1. These features are present in both the truth and generation, albeit with a longer tail in truth. Even though the first and second moments of the reweighted generation match the truth well, the full distributions are not statistically identical. This is because higher moments are relevant and are not the same between truth and generation.



(a)



(b)

Figure 3: (a) Histograms of the jet width at particle-level. (b) The binary cross entropy loss Eq. (2) for fixed values of g given by the moments indicated on the horizontal axis. These are two dimensional slices of a three dimensional plot, with the slice taken at the true value of the other moment. The correct values, known prior to the training, are indicated by a vertical dotted lines. The vertical dotted lines (true moments) coincide with the peaks of the loss curves (predicted moments) with a mean absolute error of $\leq 0.02\%$.

4 Conclusions and Outlook

In this paper, we analyzed the impediments to accurately unfolding detector data to facilitate comparisons with theoretical calculations without introducing binning artifacts. To that end we proposed Moment Unfolding as a novel, flexible, unbinned, and non-iterative reweighting technique to unfold the statistical moments of jet observables from detector data. Moment Unfolding showed promising results when applied to both Gaussian datasets as well as detector data from the LHC, although future work will be required to study higher-dimensional examples as well as to compare with other unfolding approaches.

A key question for future work is whether this method could be used to unfold infinitely many moments, thereby unfolding the entire probability density. This would involve answering important questions about partition function normalization, stability, and overlapping support, which become salient in the infinite moment limit. We hope that future algorithmic and implementation developments will enable more effective strategies for moment unfolding in particle physics and beyond.

5 Broader Impacts

Deconvolution is an essential operation in any signal processing or image processing procedure that involves a signal convoluted with well modelled background effects. Effective and accurate deconvolution methods find utility in fields ranging from biology, physiology, and medical device technology [19], to seismology [20], to spectral astronomy [21]. Deconvolution is also a critical step in various industry areas such as computer vision [22], and finance [23]. The moment unfolding algorithm we suggest is data set agnostic, and can be applied to data from a diversity of fields.

Code and Data

The code for this paper can be found at `https://github.com/hep-lbdl/MomentUnfolding`, which makes use of JUPYTER notebooks [24] employing NUMPY [25] for data manipulation and MATPLOTLIB [26] to produce figures. All of the machine learning was performed on an Nvidia RTX6000 Graphical Processing Unit (GPU) and reproducing the entire notebook takes less than five minutes. The physics data sets are hosted on Zenodo at [5, 27].

Acknowledgments and Disclosure of Funding

BN would like to thank Stefan Kluth for useful discussions about moment unfolding nearly a decade ago in the context of Ref. [28]. KD and BN are supported by the U.S. Department of Energy, Office of Science under contract DE-AC02-05CH11231. JT is supported by the National Science Foundation under Cooperative Agreement PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, `http://iaifi.org/`), and by the U.S. DOE Office of High Energy Physics under grant number DE-SC0012567.

References

[1] L. Brenner *et al.*, “Comparison of unfolding methods using RooFitUnfold,” *Int. J. Mod. Phys. A*, vol. 35, no. 24, p. 2 050 145, 2020. DOI: `10.1142/S0217751X20501456`. arXiv: `1910.14654 [physics.data-an]`.

[2] G. Zech and B. Aslan, “Binning-Free Unfolding Based on Monte Carlo Migration,” *PHYSTAT*, 2003.

[3] L. Lindemann and G. Zech, “Unfolding by weighting Monte Carlo events,” *Nucl. Instrum. Meth. A*, vol. 354, pp. 516–521, 1995. DOI: `10.1016/0168-9002(94)01067-6`.

[4] K. Datta, D. Kar, and D. Roy, “Unfolding with Generative Adversarial Networks,” 2018. arXiv: `1806.00433 [physics.data-an]`.

[5] A. Andreassen, P. T. Komiske, E. M. Metodiev, B. Nachman, and J. Thaler, “OmniFold: A Method to Simultaneously Unfold All Observables,” *Phys. Rev. Lett.*, vol. 124, no. 18, p. 182 001, 2020. DOI: `10.1103/PhysRevLett.124.182001`. arXiv: `1911.09107 [hep-ph]`.

[6] M. Arratia *et al.*, “Presenting Unbinned Differential Cross Section Results,” Sep. 2021. arXiv: `2109.13243 [hep-ph]`.

[7] 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*, A. Banerjee and K. Fukumizu, Eds., ser. Proceedings of Machine Learning Research, vol. 130, PMLR, Nov. 2021, pp. 2107–2115. arXiv: `2011.05836 [stat.ML]`. [Online]. Available: `https://proceedings.mlr.press/v130/vandegar21a.html`.

[8] V. Andreev *et al.*, “Measurement of lepton-jet correlation in deep-inelastic scattering with the H1 detector using machine learning for unfolding,” Aug. 2021. arXiv: `2108.12376 [hep-ex]`.

[9] H1 Collaboration, “Multi-differential jet substructure measurement in high Q^2 dis events with hera-ii data,” *H1prelim-22-034*, 2022. [Online]. Available: `https://www-h1.desy.de/h1/www/publications/htmlsplit/H1prelim-22-034.long.html`.

[10] H1 Collaboration, “Machine learning-assisted measurement of multi-differential lepton-jet correlations in deep-inelastic scattering with the h1 detector,” *H1prelim-22-031*, 2022. [Online]. Available: `https://www-h1.desy.de/h1/www/publications/htmlsplit/H1prelim-22-031.long.html`.

[11] P. T. Komiske, S. Kryhin, and J. Thaler, “Disentangling Quarks and Gluons with CMS Open Data,” May 2022. arXiv: 2205.04459 [hep-ph].

[12] I. J. Goodfellow *et al.*, “Generative Adversarial Networks,” 2014. arXiv: 1406.2661 [stat.ML].

[13] K. Sharp and F. Matschinsky, “Translation of ludwig boltzmann’s paper “on the relationship between the second fundamental theorem of the mechanical theory of heat and probability calculations regarding the conditions for thermal equilibrium” sitzungberichte der kaiserlichen akademie der wissenschaften. mathematisch-naturwissen classe. abt. ii, lxxvi 1877, pp 373-435 (wien. ber. 1877, 76:373-435). reprinted in wiss. abhandlungen, vol. ii, reprint 42, p. 164-223, barth, leipzig, 1909,” *Entropy*, vol. 17, no. 4, pp. 1971–2009, 2015, ISSN: 1099-4300. DOI: 10.3390/e17041971. [Online]. Available: https://www.mdpi.com/1099-4300/17/4/1971.

[14] F. Chollet, *Keras*, https://github.com/fchollet/keras, 2017.

[15] M. Abadi *et al.*, “Tensorflow: A system for large-scale machine learning.,” in *OSDI*, vol. 16, 2016, pp. 265–283.

[16] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014. arXiv: 1412.6980 [cs].

[17] J. de Favereau *et al.*, “DELPHES 3, A modular framework for fast simulation of a generic collider experiment,” *JHEP*, vol. 02, p. 057, 2014. DOI: 10.1007/JHEP02(2014)057. arXiv: 1307.6346 [hep-ex].

[18] D. Krohn, M. D. Schwartz, T. Lin, and W. J. Waalewijn, “Jet Charge at the LHC,” *Phys. Rev. Lett.*, vol. 110, no. 21, p. 212 001, 2013. DOI: 10.1103/PhysRevLett.110.212001. arXiv: 1209.2421 [hep-ph].

[19] G. Sparacino and C. Cobelli, “Reconstruction of insulin secretion rate by deconvolution: Domain of validity of a monoexponential c-peptide impulse response model.,” *Technology and health care : official journal of the European Society for Engineering and Medicine*, vol. 4 1, pp. 87–95, 1996.

[20] B. Russell, “Machine learning and geophysical inversion — a numerical study,” *The Leading Edge*, vol. 38, pp. 512–519, Jul. 2019. DOI: 10.1190/tle38070512.1.

[21] M. Molnar, K. P. Reardon, C. Osborne, and I. Milić, “Spectral deconvolution with deep learning: Removing the effects of spectral PSF broadening,” *Frontiers in Astronomy and Space Sciences*, vol. 7, Jun. 2020. DOI: 10.3389/fspas.2020.00029. [Online]. Available: https://doi.org/10.3389%2Ffspas.2020.00029.

[22] M. Makarkin and D. Bratashov, “State-of-the-art approaches for image deconvolution problems, including modern deep learning architectures,” *Micromachines*, vol. 12, no. 12, 2021, ISSN: 2072-666X. DOI: 10.3390/mi12121558. [Online]. Available: https://www.mdpi.com/2072-666X/12/12/1558.

[23] T. Meinl and E. Sun, “Methods of denoising financial data,” in Aug. 2015, pp. 519–538. DOI: 10.1007/978-1-4614-7750-1_18.

[24] T. Kluyver *et al.*, “Jupyter notebooks – a publishing format for reproducible computational workflows,” in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, F. Loizides and B. Schmidt, Eds., IOS Press, 2016, pp. 87–90.

[25] C. R. Harris *et al.*, “Array programming with NumPy,” *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. DOI: 10.1038/s41586-020-2649-2. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2.

[26] J. D. Hunter, “Matplotlib: A 2d graphics environment,” *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007. DOI: 10.1109/MCSE.2007.55.

[27] A. Andreassen, P. Komiske, E. Metodiev, B. Nachman, and J. Thaler, *Pythia/Herwig + Delphes Jet Datasets for OmniFold Unfolding*, version v0, Zenodo, Nov. 2019. DOI: 10.5281/zenodo.3548091. [Online]. Available: https://doi.org/10.5281/zenodo.3548091.

[28] G. Aad *et al.*, “Measurement of jet charge in dijet events from $\sqrt{s}$=8 TeV pp collisions with the ATLAS detector,” *Phys. Rev. D*, vol. 93, no. 5, p. 052 003, 2016. DOI: 10.1103/PhysRevD.93.052003. arXiv: 1509.05190 [hep-ex].

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] We did not require significant computing resources for the example shown in this extended abstract.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]