UC Berkeley
UC Berkeley Previously Published Works

Title

Neural Posterior Unfolding

Permalink

https://escholarship.org/uc/item/5mg3n583

Authors

Desai, Krish
Nachman, benjamin
Pan, jingjing
et al.

Publication Date

2024-12-12

Copyright Information

This work is made available under the terms of a Creative Commons Attribution License, available at https://creativecommons.org/licenses/by/4.0/

Peer reviewed

eScholarship.org Powered by the California Digital Library
University of California

Neural Posterior Unfolding

Fernando Torales Acosta
Physics Division
Lawrence Berkeley National Laboratory
Berkeley, CA 94720

Jay Chan
Physics Division
Lawrence Berkeley National Laboratory
Berkeley, CA 94720

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

Vinicius Mikuni
Physics Division
Lawrence Berkeley National Laboratory
Berkeley, CA 94720

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

Jingjing Pan
Wright Laboratory
Yale University
New Haven, CT 06511
jingjing.pan@yale.edu

Abstract

Differential cross section measurements are the currency of scientific exchange in particle and nuclear physics. The key challenge for these analyses is the correction for detector distortions known as deconvolution or *unfolding*. In the case of binned cross section measurements, there are many tools for regularized matrix inversion where the matrix governs the detector response going from pre- to post-detector observables. In this paper, we show how normalizing flows and neural posterior estimation can be used for unfolding. This approach has many potential advantages, including implicit regularization from the neural networks and fast inference from amortized training. We demonstrate this approach using simple Gaussian examples as well as a simulated jet substructure measurement at the Large Hadron Collider.

1 Introduction

In particle and nuclear physics, theories are connected with experiments through cross section measurements. Given some observables (what data scientists might call 'features'), we want to predict and measure how often a given reaction produces a certain value of the observable. The key challenge is that we only measure the value of the observable after detector distortions while first-principles calculations do not account for such effects. Removing these distortions is thus a critical step in performing a cross section measurement. This task is known as unfolding. The detector response refers to the conditional probability density

$$p(\text{measured} = m|\text{truth} = t). \tag{1}$$

Since the detector response can be simulated extremely precisely for collider experiments, one might seek to circumvent the ill-posed inverse problem entirely. In some cases, one could, instead, "forward fold" the theoretical prediction through the detector response, and compare with the experimental result at detector–level. However, it is not always possible to avoid the inverse problem. For example, a theory–agnostic comparison between experiments requires that at least one of the measurements be

Machine Learning and the Physical Sciences Workshop, NeurIPS 2024.

unfolded, since detector responses differ significantly between experiments. Furthermore, unfolding has a future-proofing effect on the data. Unfolded data can be compared to theoretical predictions years later, even if the detector simulation is no longer available.

Thousands of unfolded cross section measurements have been performed, nearly all of them using classical methods [1]. Nearly all classical methods discretize the problem, representing the differential cross section as a histogram. While there is a growing interest in using machine learning for unbinned unfolding [2, 3], we focus on the (currently) more prevalent binned methods.

Particle and nuclear physicists are typically frequentists and thus employ maximum likelihood estimation (MLE) for unfolding. Directly maximizing the likelihood is numerically unstable, especially when detector distortions are non-negligible, because the detector response can have eigenvalues very close to 0. As a result, all extant unfolding methods in particle and nuclear physics perform a variation of regularized maximum likelihood estimation to regularize the otherwise unacceptable numerical instability introduced by the ill-posedness of the inverse problem.

One strategy is to use Bayesian methods with a uniform prior. Fully Bayesian Unfolding [4] (FBU) has been used for a number of measurements, with a uniform prior and Markov Chain Monte Carlo (MCMC). In this paper, we explore an alternative approach to MCMC by using normalizing flows [5] and amortized machine learning. The idea is that we use pairs of (truth, measured) histograms to learn the density of truth given measured (note that the causal structure is in the opposite direction). The potential advantages over MCMC are that we have a function approximation of the likelihood and we do not need to rerun the algorithm if we change the dataset (e.g. add more data or determine statistical uncertainties with bootstrapping [6]). Furthermore, as a neural network, normalizing flows are inherently regularized through the limited complexity of the architecture and training protocol. We call this approach *neural posterior unfolding* (NPU).

2 Statistics of Unfolding

For binned unfolding, we approximate the detector response in Eq. 1 with the response matrix

$$R_{ij} = \Pr(m_i|t_j) \tag{2}$$

where m_i indicates that the observable is measured in bin i at detector–level and t_j indicates that the observable is in bin j at particle–level. The response matrix for collider experiments can be computed extremely precisely using simulations. The task is then, to infer the cross section t_i per particle-level bin i given observed counts m_j in detector-level bin j and response matrix. It follows from the definition of the response matrix that, on average, $m = R\,t$. The simplest solution is therefore $\hat{t} = R^{-1}\,d$. While unbiased, this simple solution has many issues. First, it need not be non-negative definite, as required for physical cross sections. Second, matrix inversion amplifies statistical fluctuations due to (often large) off-diagonal components of R. A number of regularized matrix inversion solutions have been proposed to address both of these issues, including the widely-used Iterative Bayesian Unfolding (IBU) (also known as Lucy-Richardson deconvolution) [7–9]. Even though 'Bayes' is in the name of IBU, the approach is strictly frequentist; it is an expectation-maximization technique that converges to a (local) MLE.

IBU will serve as our baseline. One way of describing it is through the following iterative protocol:

$$t_j^{(n)} = \sum_i \Pr_{n-1}(\text{truth } j|\text{measure } i)\,\Pr(\text{measure } i) = \sum_i \frac{R_{ij}t_j^{(n-1)}}{\sum_k R_{ik}t_k^{(n-1)}} \times m_i\,, \tag{3}$$

where n is the iteration number and stopping $n < \infty$ is a form of regularization. IBU only provides a point estimate, but it is possible to estimate statistical uncertainties through asymptotic formulae or through bootstrapping. This already highlights potential advantages of NPU: the statistical uncertainty is part of the result and the regularization is implicit in the training and automated through model selection (via the validation loss). Another difference between NPU and IBU is their behavior in unconstrained regions of phase space. Cross sections in the particle-level phase space that are un- or poorly constrained by detector-level measurements should be highly uncertain. However, this is not always the case. Suppose that there are two bins α and β such that $R_{\kappa\alpha} = R_{\kappa\beta} = 1$ and $R_{\kappa\omega} = 0$ for $\omega \notin \{\alpha, \beta\}$ for some detector-level bin κ. In other words, the detector cannot differentiate between

two truth bins. Following Eq. 3, IBU would return:

$$t_\alpha^{(n)} = \frac{t_\alpha^{(n-1)} m_\kappa}{t_\alpha^{(n-1)} + t_\beta^{(n-1)}} \quad t_\beta^{(n)} = \frac{t_\beta^{(n-1)} m_\kappa}{t_\alpha^{(n-1)} + t_\beta^{(n-1)}}, \tag{4}$$

which means that $t_\alpha^{(n)}/(t_\alpha^{(n)} + t_\beta^{(n)}) = t_\alpha^{(0)}/(t_\alpha^{(0)} + t_\beta^{(0)})$, i.e. is the same as the 'prior'. Importantly this quantity is independent of the observations m. The problem is thus that the uncertainty on $\hat{t}_\alpha/(\hat{t}_\alpha + \hat{t}_\beta)$ would be zero, even though it should be maximal. This uncertainty is currently somewhat covered through systematic uncertainties estimated by comparing different priors, usually with a small set of simulations. These are not sufficient and in fact the uncertainty could still be zero if the two priors happen to agree in these bins. A similar analysis holds for other regularized matrix inversion techniques. Unfolding methods such as FBU and the method this paper proposes, NPU, circumvent this issue by returning a wide posterior.

3 Machine Learning Implementation

We implement neural posterior unfolding using TENSORFLOW [10] and TENSORFLOW PROBABILITY packages [11]. The normalizing flow implementation is based on the MADE [12, 13] implementation consisting of an invertible transformation using two fully connected layers with 16 nodes and SWISH [14] activation functions. The conditional inputs, based on pre-detector observables, are included with an additional fully connected layer. The model is then trained for between 1000 and 1500 epochs. We used the ADAM [15] optimizer with a learning rate parameter of 0.001. After training, we determine the unfolded response by performing the MLE. This is carried out by minimizing the negative log-likelihood of the data based on the conditional inputs, also with ADAM.

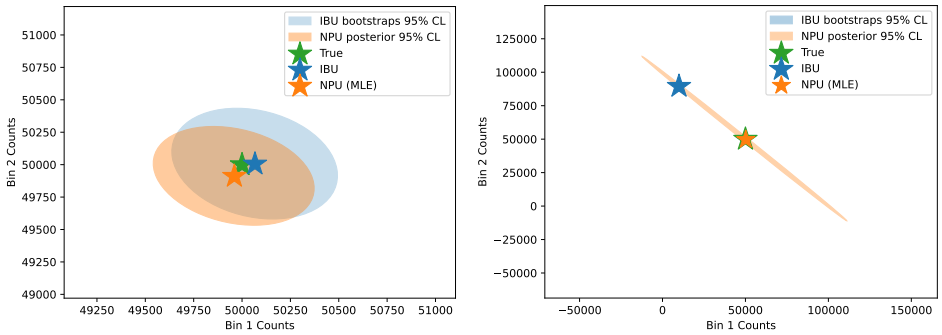
4 Numerical Results

4.1 2-bin Degenerate Response Example

We first start with a simple two-bin example. In this case, $t, m \in \mathbb{R}^2$ and R is a 2×2 matrix

$$R = \begin{bmatrix} \sigma & \rho \\ \rho & \sigma \end{bmatrix}, \tag{5}$$

where in this example, $\sigma = \sigma_1 = \sigma_2 = 0.8$ and $\rho = \mathrm{Cov}_{12}$ We fix $t_0 = t_1 = 5 \times 10^4$. First, Fig. 1(a) shows the case where $\rho = 1$. With small migrations, all methods perform about the same, and the true answer is well-contained within the confidence regions of both IBU (determined via boostrapping) and NPU. However, once we set $\rho \approx 0$, Fig. 1(b) shows the challenges with IBU highlighted in the previous section. In particular, IBU returns a single value while NPU returns a broad uncertainty region (all values consistent with the total counts).



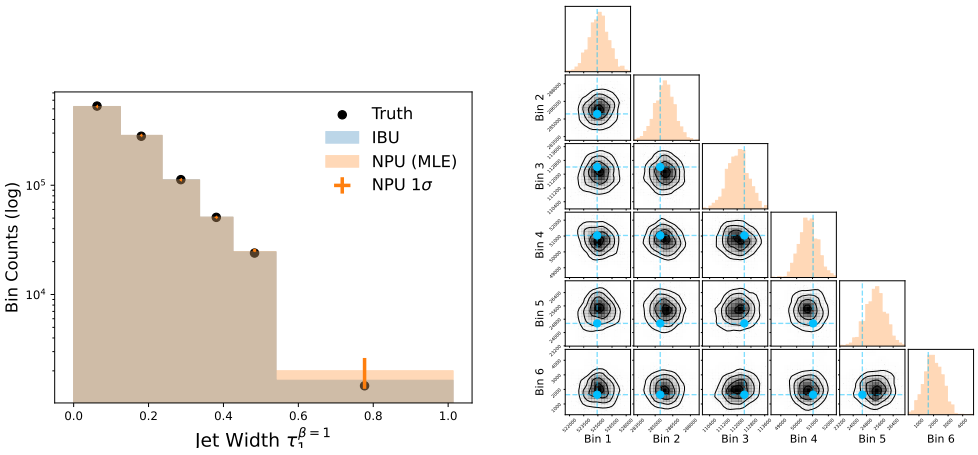
(a) 2-bin example (b) 2-bin example with degenerate response

Figure 1: Demonstration of NPU using a two-bin example with (a) a nearly diagonal response matrix *v.s.* (b) a response with degeneracy. The MLE of NPU shows good agreement with the truth values.

4.2 Particle Physics Example

Our study is based on proton-proton collisions generated at $\sqrt{s} = 14$ TeV; the dataset is the same as used in Ref. [16, 17] and is briefly described below. The dataset used the default tune of HERWIG 7.1.5 [18] and Tune 26 [19] of PYTHIA 8.243 [20] in order to study a challenging setting where the 'natural' and 'simulated' distributions are significantly different. As a proxy for detector effects and a full detector simulation, we use the DELPHES 3.4.2 [21] fast simulation of the CMS detector [22], which uses particle flow reconstruction. Jets with radius parameter $R = 0.4$ are clustered using either all particle flow objects (detector-level) or stable non-neutrino truth particles (particle-level) with the anti-k_T algorithm [23] implemented in FastJet 3.3.2 [24]. One of the simulations (HERWIG) plays the role of data and truth, while the other (PYTHIA) is used to derive the unfolding corrections. To reduce acceptance effects, the leading jets are studied in events with a Z boson with transverse momentum $p_T^Z > 200$ GeV. After applying the selections, we obtain approximately 1.5 million events from each generator.

To investigate the unfolding performance, we investigated a number of jet substructure observables. For brevity, we show the results from the jet width ($\tau_1^{\beta=1}$), which is representative. The jet width is the transverse-momentum-weighted first radial moment of the radiation within a jet. Gluon-initiated jets tend to be wider than quark jets. The unfolding performance of NPU is shown in Fig. 2(a) and compared to IBU with 10 iterations. The corresponding corner plot is in Fig. 2(b). NPU continues to succeed in recovering the truth distribution inside the high-energy jets produced in the complicated real-world LHC environment, despite the challenging long tail of this widely used observable.



(a) Gaussian result (b) Gaussian result in corner plot

Figure 2: The unfolding result for jet width $\tau_1^{\beta=1}$, using HERWIG 7.1.5 (data & truth) and PYTHIA 8.243 tune 26 (simulations to prepare the response matrix). The errorbars in orange provide the standard deviation of the posterior from NPU, in addition to the MLE of NPU that continues to succeed in matching the truth distribution (particle-level) marked by black dots. Corner plot for a physics example [25]: The corner plot is also shown with comparison to IBU in blue.

5 Conclusions and Outlook

In this paper, we proposed Neural Posterior Unfolding (NPU), a new ML-based unfolding method that can provide fast[1] access to the full posterior without resorting to MCMC through neural posterior estimation. The method uses normalizing flows as a density estimator to directly learn $\Pr(t_j|m_i)$ from the priors that were forward folded with the response matrix. The unfolded distribution is then obtained by performing maximum likelihood estimation, where we minimize the negative log-likelihood

[1]Unlike cases of computationally expensive forward models, the generative model here is simple (product of Poissons). However, the computational differences may be non-negligible when many bootstraps are required for classical uncertainty quantification.

of the observed data based on the conditional inputs. Namely, we minimize $-\log(\Pr(t_j|m_i))$ with respect to t_j and plot t'_j after minimization as the unfolded result that we expect to match the truth (pre-detector) distribution.

In a two-bin example, we validated the coverage of the posterior given by NPU through comparing with IBU using bootstraps when the response matrix is non-degenerate. When the response matrix contains degeneracy, or physically when the particle-level phase space is not uniquely constrained by detector-level measurements, we demonstrated that the NPU method can still provide the correct unfolded values when IBU breaks down.

To test the new method in a numerically more challenging and physically more relevant example, we applied NPU to the cross section measurements of a jet substructure observable namely, the jet width ($\tau_1^{\beta=1}$). This observable has been broadly used in both particle and nuclear physics, especially for the task of jet tagging. We showed that NPU accurately and precisely recovers the truth distribution of the jet substructure variable while providing the full posterior, despite the challenging shape of the probability density function of the jet width, as well as the complicated detector distortions in the real-world LHC environment.

Unlike classical unfolding methods, the regularization in NPU is implicit. It would be interesting to explore this difference in more detail in the future in order to understand its benefits and disadvantages. The NPU framework introduced here may also provide a starting point for full statistical uncertainty quantification in the case of unbinned results. Lastly, the code for NPU is written with modern Python and ML libraries; this is not the case for FBU and thus may connect more researchers to this alternative method in the future.

References

[1] Tim Adye. Unfolding algorithms and tests using RooUnfold. In *PHYSTAT 2011*, pages 313–318, Geneva, 2011. CERN. doi: 10.5170/CERN-2011-006.313.

[2] Miguel Arratia et al. Publishing unbinned differential cross section results. *JINST*, 17(01): P01024, 2022. doi: 10.1088/1748-0221/17/01/P01024.

[3] Nathan Huetsch et al. The Landscape of Unfolding with Machine Learning. 4 2024.

[4] Georgios Choudalakis. Fully bayesian unfolding, 2012.

[5] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2016.

[6] B. Efron. Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7 (1):1 – 26, 1979. doi: 10.1214/aos/1176344552. URL `https://doi.org/10.1214/aos/1176344552`.

[7] G. D'Agostini. A Multidimensional unfolding method based on Bayes' theorem. *Nucl. Instrum. Meth.*, A362:487–498, 1995. doi: 10.1016/0168-9002(95)00274-X.

[8] William Hadley Richardson. Bayesian-based iterative method of image restoration. *J. Opt. Soc. Am.*, 62(1):55–59, Jan 1972. doi: 10.1364/JOSA.62.000055. URL `http://www.osapublishing.org/abstract.cfm?URI=josa-62-1-55`.

[9] L. B. Lucy. An iterative technique for the rectification of observed distributions. *Astronomical Journal*, 79:745, June 1974. doi: 10.1086/111605.

[10] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

[11] Joshua V. Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matthew D. Hoffman, and Rif A. Saurous. Tensorflow distributions. *CoRR*, abs/1711.10604, 2017. URL `http://arxiv.org/abs/1711.10604`.

[12] Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. MADE: Masked Autoencoder for Distribution Estimation. 2015.

[13] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked Autoregressive Flow for Density Estimation. 2018.

[14] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[15] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. 2015.

[16] Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler. OmniFold: A Method to Simultaneously Unfold All Observables. *Phys. Rev. Lett.*, 124 (18):182001, 2020. doi: 10.1103/PhysRevLett.124.182001.

[17] Anders Andreassen, Patrick Komiske, Eric Metodiev, Benjamin Nachman, and Jesse Thaler. Pythia/Herwig + Delphes Jet Datasets for OmniFold Unfolding, November 2019. URL `https://doi.org/10.5281/zenodo.3548091`.

[18] Johannes Bellm et al. Herwig 7.1 Release Note. 5 2017.

[19] ATLAS Collaboration. ATLAS Pythia 8 tunes to 7 TeV data. 11 2014. `https://cds.cern.ch/record/1966419`.

[20] Torbjörn Sjöstrand, Stefan Ask, Jesper R. Christiansen, Richard Corke, Nishita Desai, Philip Ilten, Stephen Mrenna, Stefan Prestel, Christine O. Rasmussen, and Peter Z. Skands. An introduction to PYTHIA 8.2. *Comput. Phys. Commun.*, 191:159–177, 2015. doi: 10.1016/j.cpc.2015.01.024.

[21] J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi. DELPHES 3, A modular framework for fast simulation of a generic collider experiment. *JHEP*, 02:057, 2014. doi: 10.1007/JHEP02(2014)057.

[22] S. Chatrchyan et al. The CMS Experiment at the CERN LHC. *JINST*, 3:S08004, 2008. doi: 10.1088/1748-0221/3/08/S08004.

[23] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. The anti-k_t jet clustering algorithm. *JHEP*, 04:063, 2008. doi: 10.1088/1126-6708/2008/04/063.

[24] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. FastJet User Manual. *Eur. Phys. J.*, C72: 1896, 2012. doi: 10.1140/epjc/s10052-012-1896-2.

[25] Daniel Foreman-Mackey. corner.py: Scatterplot matrices in python. *The Journal of Open Source Software*, 1(2):24, jun 2016. doi: 10.21105/joss.00024. URL `https://doi.org/10.21105/joss.00024`.