D3PU: Denoising Diffusion Detector Probabilistic Unfolding in High-Energy Physics

Camila Pazos
Department of Physics and Astronomy
Tufts University, USA
camila.pazos@tufts.edu

Shuchin Aeron
Department of Electrical and Computer Engineering
Tufts University, USA
shuchin@eecs.tufts.edu

Pierre-Hugues Beauchemin
Department of Physics and Astronomy
Tufts University, USA
hugo.beauchemin@tufts.edu

Vincent Croft
Leiden Institute for Advanced Computer Science
Leiden University, The Netherlands
vincent.croft@cern.ch

Martin Klassen
Department of Physics and Astronomy
Tufts University, USA
martin.klassen@tufts.edu

Taritree Wongjirad
Department of Physics and Astronomy
Tufts University, USA
taritree.wongjirad@tufts.edu

Abstract

Correcting for detector effects in experimental data, particularly through unfolding, is critical for enabling precision measurements in high-energy physics. However, traditional unfolding methods face challenges in scalability, flexibility, and dependence on simulations. We introduce a novel approach to multidimensional object-wise unfolding using conditional Denoising Diffusion Probabilistic Models (cDDPM). Our method utilizes the cDDPM for a non-iterative, flexible posterior sampling approach, incorporating distribution moments as conditioning information, which exhibits a strong inductive bias that allows it to generalize to unseen physics processes without explicitly assuming the underlying distribution. Our results highlight the potential of this method as a step towards a "universal" unfolding tool that reduces dependence on truth-level assumptions.

1 Introduction

Experimental data in high-energy physics (HEP) presents a distorted picture of the true physics processes due to detector effects. Unfolding is an inverse problem solved through statistical inference that aims to correct the detector distortions of the observed data to recover the true distribution of particle properties. This process is essential for the validation of theories, new discoveries, precision measurements, and comparison of experimental results between different experiments.

A standard approach to unfolding [1] begins with a simulated particle distribution $f_{\text{true}}(\mathbf{x})$ that characterizes the underlying physics process of interest, and a detailed detector simulation that describes how detector effects distort the particle property distributions. These distortions affect the kinematic quantities of particles incident to the detector, resulting in an altered particle distribution $f_{\text{det}}(\mathbf{y})$. Mathematically, this relationship can be written as a Fredholm integral equation of the first kind,

Machine Learning and the Physical Sciences Workshop, NeurIPS 2024.

$$f_{\text{det}}(\mathbf{y}) = \int d\mathbf{x}\, P(\mathbf{y}|\mathbf{x})\, f_{\text{true}}(\mathbf{x}) \tag{1}$$

where $P(\mathbf{y}|\mathbf{x})$ is the conditional probability distribution describing the detector effects. One approach to unfolding is to estimate the inverse process $P(\mathbf{x}|\mathbf{y})$, which can be expressed using Bayes' theorem:

$$P(\mathbf{x}|\mathbf{y}) = \frac{P(\mathbf{y}|\mathbf{x}) f_{\text{true}}(\mathbf{x})}{f_{\text{det}}(\mathbf{y})}\,. \tag{2}$$

In this context, a detector dataset can be unfolded by sampling from the posterior $P(\mathbf{x}|\mathbf{y})$ to recover the distribution $f_{\text{true}}(\mathbf{x})$. The detector effects $P(\mathbf{y}|\mathbf{x})$ are assumed to be the same for any physics process, and it is clear that the posterior $P(\mathbf{x}|\mathbf{y})$ depends on the prior distribution $f_{\text{true}}(\mathbf{x})$. Although one can sample from $f_{\text{true}}(\mathbf{x})$ through the use of particle generators, there is no guarantee that any particular assumed $f_{\text{true}}(\mathbf{x})$ accurately represents the underlying physics of the specific data to unfold. Consequently, unfolding results can be significantly influenced by the assumed underlying distribution, potentially introducing bias or limiting the method's ability to detect unexpected phenomena. This reveals one of the main challenges in developing a *universal* unfolder, which aims to remove detector effects from any set of measured data agnostic of the process of interest, ideally with no bias towards any prior distribution.

Related Work: Traditional unfolding methods, based on the linearization of the problem, face limitations such as requiring binned histograms and an inability to unfold multiple observables simultaneously. Various machine learning approaches have emerged in recent years to address these challenges. These include re-weighting methods like OmniFold [2] [3], as well as several generative approaches. Among the generative techniques are those using Generative Adversarial Networks (GANs) [4] [5], conditional invertible neural networks [6] [7], and latent variational diffusion models [8] [9]. Additionally, distribution mapping techniques have been developed, such as Schrödinger bridges [10] and direct diffusion models [11]. For a comprehensive overview of these methods, see the recent survey by [12]. Each new method has made further strides in unfolding and shown the advantages in machine learning based approaches compared to traditional techniques.

Objectives: This work seeks to overcome the limitations of traditional unfolding methods while expanding upon the benefits offered by machine learning-based approaches. The proposed approach builds upon the advantages of object-wise unfolding, a technique common in machine learning-based unfolding methods, which reconstructs the properties of individual particles or physics objects rather than operating on binned distributions. Through object-wise unfolding, some of the challenges posed by traditional methods can be addressed: the impact of the experimenter's selections and cuts on the unfolded results can be minimized, while underlying correlations between the unfolded distributions are preserved.

We first present a "dedicated" unfolder, an approach similar to many existing machine learning-based methods, which learns and applies a specific posterior distribution for a particular physics process. This approach serves as an effective solution for well-understood processes and provides a benchmark for subsequent work. Building upon this foundation, the aim is to develop a "generalizable" unfolder to handle a wide range of physics processes and observables, including those not explicitly seen during training. This generalization capability is crucial for enhancing the method's applicability across various physics scenarios, while ideally avoiding dependence on specific physics generator models. This amounts to addressing both the bias and generalization problems in the solution to unfolding. Such a method would enable the unfolding of distributions for a wide range of processes, including those involving yet-undiscovered particles in new physics searches at high-energy colliders.

An effective new unfolding method should achieve an accuracy that falls within the typical uncertainty range of measurements where unfolding is applied, while simultaneously preserving the benefits of object-wise unfolding, such as maintaining correlations between kinematic quantities, and offering generalization capabilities. With these objectives, the goal is to contribute a more flexible, accurate, and widely applicable unfolding tool to the high-energy physics community.

Our Contribution: We introduce a novel approach using conditional Denoising Diffusion Probabilistic Models (cDDPM) to unfold detector effects in HEP data. We demonstrate that a single

cDDPM, trained on diverse particle data and incorporating statistical moments of various distributions, can serve as a "generalized" unfolder by performing multidimensional object-wise unfolding for multiple physics processes without explicit assumptions about the underlying distribution, thereby minimizing bias. Figure 1 illustrates the effectiveness of this approach in two key scenarios. The left panel shows an "unknown" process created by combining data from multiple known processes (40% $t\bar{t}$, 35% W+jets, and 25% leptoquark). Here, the generalized unfolder outperforms the dedicated unfolder, which is designed to unfold only a single specific physics process (in this case $t\bar{t}$, selected because it forms the largest component of the unknown process). The right panel provides further evidence of the generalized unfolder's flexibility, demonstrating its ability to accurately unfold data from graviton production (generated in the context of large extra-dimension scenarios [13]) accompanied by jets, a completely new physics process absent from the training phase. The accuracy of the generalized approach illustrates its ability to handle previously unseen physics processes without assuming an underlying distribution. This flexibility demonstrated by the generalized unfolder is beneficial for new physics searches and studying processes not accurately modeled by current theories, providing an unfolding solution to the bulk of the data analyses performed at high-energy colliders.

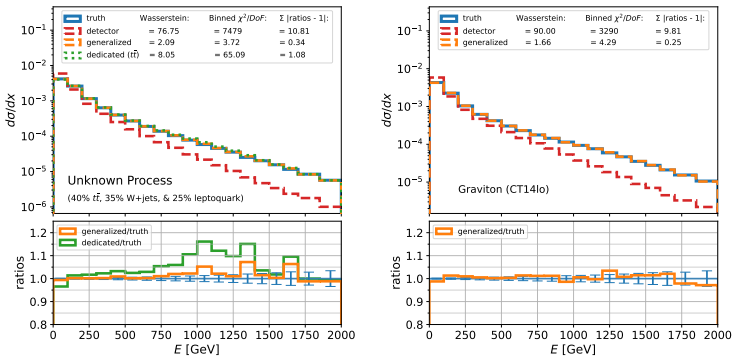

Figure 1: Unfolding results using the generalized cDDPM unfolder. The left panel shows performance on data from an "unknown" physics process combining multiple processes. The generalized unfolder (orange) demonstrates superior performance compared to the dedicated unfolder (green), which was trained assuming a specific physics process. The right panel shows the generalized unfolder successfully handling data from graviton production accompanied by jets, a new physics process completely absent from the training data.

2 Methods

2.1 Our Unfolding Approach

While an ideal universal unfolder cannot be achieved, this approach seeks to enhance the inductive bias of the unfolding method to improve generalization to cover various posteriors pertaining to different physics data distributions. The posteriors for two different physics processes i and j are related by a ratio of the probability density functions of each process,

$$\frac{P_i(\mathbf{x}|\mathbf{y})}{P_j(\mathbf{x}|\mathbf{y})} = \frac{f^i_{\text{true}}(\mathbf{x})\, f^j_{\text{det}}(\mathbf{y})}{f^i_{\text{det}}(\mathbf{y})\, f^j_{\text{true}}(\mathbf{x})} \,. \tag{3}$$

If the posterior for a given physics process can be learned, extrapolation to unseen posteriors becomes possible if the priors $f_{\text{true}}(\mathbf{x})$ and detector distributions $f_{\text{det}}(\mathbf{y})$ can be approximated or written in a closed form. Although these functions have no analytical form, key features can be approximated using the first moments of these distributions. By making use of these moments as conditionals, a more flexible unfolder can be created that is not strictly tied to a selected prior distribution, and enables interpolation and extrapolation to unseen posteriors based on the provided moments. Consequently, this unfolding tool gains the ability to handle a wider range of physics processes and enhances the generalization capabilities, making it a more versatile tool for unfolding in various high energy physics applications.

2.2 Denoising Diffusion Probabilistic Models

The proposed unfolding approach calls for a flexible generative model, and denoising diffusion probabilistic models (DDPMs) [14] lend themselves naturally to this task. DDPMs learn via a reversible generative process which can be conditioned directly on the detector data values and on the moments of the distribution $f_{\text{det}}(\mathbf{y})$, providing a natural way to sample from $P(\mathbf{x}|\mathbf{y})$ for unfolding. A DDPM comprises two parts: a fixed forward process that gradually adds Gaussian noise to data samples, and a learned reverse process that denoises the data.

The implementation uses a conditional DDPM with direct conditioning, where sampling is done according to the learned conditional distribution. The loss function for the conditional denoising process is:

$$L(\theta) = \mathbb{E}_{t,\,\boldsymbol{\epsilon},\,\mathbf{x}_t,\,\mathbf{y}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta, (t, \mathbf{x}_t, \mathbf{y})\right\|^2\right]. \tag{4}$$

where $\boldsymbol{\epsilon}$ is the noise added during the forward process and $\boldsymbol{\epsilon}_\theta(t, \mathbf{x}_t, \mathbf{y})$ is the noise predicted by the model. More details on cDDPMs and a derivation of this loss can be found in Appendix A.

This approach differs from the commonly used guided conditioning methods, where the predicted noise is a weighted combination of the conditional and unconditional predictions: $\tilde{\boldsymbol{\epsilon}}_\theta(t, \mathbf{x}_t, \mathbf{y}) = (1 + w)\,\boldsymbol{\epsilon}_\theta(t, \mathbf{x}_t, \mathbf{y}) - w\,\boldsymbol{\epsilon}_\theta(t, \mathbf{x}_t)$ [15]. The cDDPM can be seen as a special case of guided conditioning with the guidance weight $w = 0$, allowing sampling from $P(\mathbf{x}|\mathbf{y})$ without explicitly evaluating the prior distribution over the data space. This makes the cDDPM a natural choice for applications like unfolding where the prior is unknown or difficult to model.

2.3 Unfolding with cDDPMs

Our study focuses on QCD jets, narrow streams of hadrons produced by quark or gluon hadronization in high-energy particle collisions. Using the PYTHIA event generator [16], jet datasets are generated for various physics processes ($t\bar{t}$, W+jets, Z+jets, dijet, leptoquark, and graviton) under different settings. These "truth-level" jets are then passed through a detector simulation framework to produce "detector-level" jets, mimicking particle interactions within a detector.

Part 1: Dedicated Unfolder We first consider how to setup a *dedicated* cDDPM unfolder (without use of the distributional moments) that can achieve multidimensional object-wise unfolding for a single physics process. The jet kinematic information is defined with a vector that includes the transverse momentum (p_T), pseudorapidity (η), azimuthal angle (ϕ), and 4-momentum vector (E, p_x, p_y, p_z). These jet vectors are defined both at truth-level as $\mathbf{x}$ and detector-level as $\mathbf{y}$. A cDDPM can be trained with data pairs $(\mathbf{x}, \mathbf{y})$ as input to learn the posterior distribution $P(\mathbf{x}|\mathbf{y})$. To unfold, the detector data $\mathbf{y}$ is given as input and the cDDPM acts as a posterior sampler of $P(\mathbf{x}|\mathbf{y})$.

Part 2: Generalized Unfolder We aim to enhance the inductive bias through use of the distributional moments to attain a *generalized* cDDPM unfolder that encompasses a broader range of posteriors, enabling the unfolding of data from diverse physics processes. To achieve this, the training dataset is expanded to include jets from multiple different physics simulations. For each simulation, the first 6 moments of the p_T distribution are calculated and appended to the corresponding jet vectors. In slight abuse of notation, we now denote these augmented jet vectors (including distribution moments) as $\mathbf{x}$ at truth-level and $\mathbf{y}$ at detector-level. By training with these diverse data pairs $(\mathbf{x}, \mathbf{y})$, the cDDPM is able to represent multiple posteriors corresponding to the distributions in the expanded training dataset, distinguishable through the added distributional information provided by the moments. More details on the training dataset, procedure, cDDPM parameters, and pseudocode are provided in Appendix B.

3 Results and Discussion

The Wasserstein-1 distance [17], "Binned χ^2/DoF", and "$\sum |\text{ratios} - 1|$" are metrics employed to evaluate the unfolding performance, with all three metrics appearing in the figures.

In Figure 1, the left panel showcases results from an "unknown" process dataset, created by combining jets from the $t\bar{t}$, W+jets, and leptoquark test datasets. Here, the moments used for conditioning are

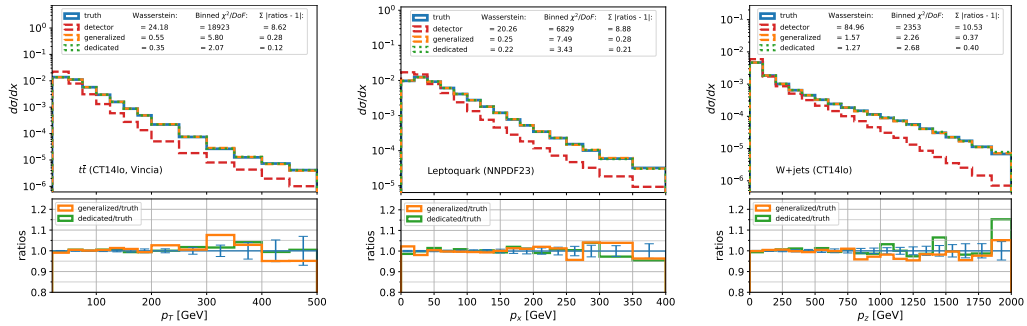

Figure 2: Unfolding results of jet vector components across diverse physics processes. We compare our generalized cDDPM unfolder (orange) against process-specific dedicated unfolders (green).

calculated from the combined dataset as a whole, presenting the generalized unfolder with previously unseen distributional characteristics. The generalized unfolder demonstrates superior performance when unfolding this unknown process compared to a dedicated unfolder assuming a similar, yet incorrect, underlying $t\bar{t}$ process. The right panel demonstrates that the generalized unfolder successfully reconstructs true distributions from graviton production data (a physics process entirely absent from its training data), showing its ability to handle completely new physics scenarios. While the generalized unfolder's advantage is expected for unknown processes, comparable performance to dedicated unfolders on known processes is also desired. To validate the framework's effectiveness, both unfolders are compared across various test datasets, and Table 1 presents the resulting multidimensional Wasserstein distances to their true distributions.

Figure 2 illustrates unfolding results for various jet observables across different physics processes, showcasing the generalized unfolder's versatility. In Figure 3, the model's efficacy is further demonstrated with two tests: (1) reconstructing jet mass from unfolded results, indicating well-preserved correlations among jet vector components, and (2) reconstructing event-level observables from unfolded quantities, achieved by tracking event numbers through object-wise unfolding.

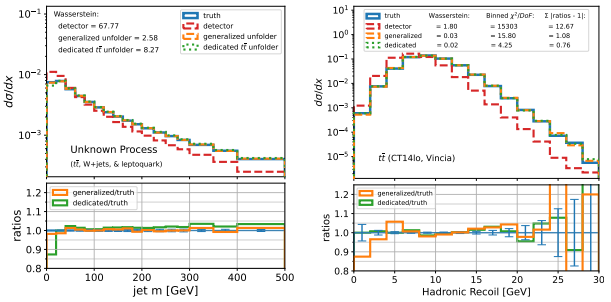

Figure 3: Reconstruction of jet mass and hadronic recoil (event-level observable) from unfolded data.

Process	Wasserstein Distances		
	Det.	Gen.	Ded.
Graviton	31.35	0.64	N/A
Unknown	28.20	0.744	2.677
$t\bar{t}$	26.43	0.565	0.196
LQ	32.72	0.457	0.155
W+jets	31.15	0.304	0.353

Table 1: Comparison of Wasserstein distances for detector-level data and unfolded results using generalized and dedicated unfolders across different physics processes.

While this approach shows promise, we acknowledge key limitations. Addressing particles outside detector thresholds and accounting for systematic and experimental uncertainties are crucial improvements needed to fully realize the method's potential in practical applications. An important constraint of our current implementation is that while correlations between object vector components are preserved, the model lacks access to event-wise information. This limitation impacts the reconstruction accuracy of certain event-level observables, since inter-object relationships within an event are not captured. We leave these improvements for future work.

To conclude, our results confirm the versatility of the generalized cDDPM unfolder across diverse physics processes. This non-iterative and flexible posterior sampling approach exhibits a strong inductive bias that allows the cDDPM to generalize to unseen processes without explicitly assuming the underlying physics distribution, setting it apart from other unfolding techniques so far.

Acknowledgments and Disclosure of Funding

This work has been made possible thanks to the support of the Department of Energy Office of Science through the Grant DE-SC0023964. Shuchin Aeron and Taritree Wonhjirad would also like to acknowledge support by the National Science Foundation under Cooperative Agreement PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, http://iaifi.org/).

References

[1] Volker Blobel. An unfolding method for high energy physics experiments, 2002. URL https://arxiv.org/abs/hep-ex/0208022.

[2] 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: 182001, 5 2020. doi: 10.1103/PhysRevLett.124.182001. URL https://link.aps.org/doi/10.1103/PhysRevLett.124.182001.

[3] Anders Andreassen, Patrick T. Komiske, Eric M. Metodiev, Benjamin Nachman, Adi Suresh, and Jesse Thaler. Scaffolding simulations with deep learning for high-dimensional deconvolution, 2021. URL https://arxiv.org/abs/2105.04448.

[4] Marco Bellagente, Anja Butter, Gregor Kasieczka, Tilman Plehn, and Ramon Winterhalder. How to gan away detector effects. SciPost Physics, 8(4), April 2020. ISSN 2542-4653. doi: 10.21468/scipostphys.8.4.070. URL http://dx.doi.org/10.21468/SciPostPhys.8.4.070.

[5] Kaustuv Datta, Deepak Kar, and Debarati Roy. Unfolding with generative adversarial networks, 2018. URL https://arxiv.org/abs/1806.00433.

[6] Mathias Backes, Anja Butter, Monica Dunford, and Bogdan Malaescu. An unfolding method based on conditional invertible neural networks (cinn) using iterative training, 2024. URL https://arxiv.org/abs/2212.08674.

[7] Marco Bellagente, Anja Butter, Gregor Kasieczka, Tilman Plehn, Armand Rousselot, Ramon Winterhalder, Lynton Ardizzone, and Ullrich Köthe. Invertible networks or partons to detector and back again. SciPost Physics, 9(5), November 2020. ISSN 2542-4653. doi: 10.21468/scipostphys.9.5.074. URL http://dx.doi.org/10.21468/SciPostPhys.9.5.074.

[8] Alexander Shmakov, Kevin Greif, Michael Fenton, Aishik Ghosh, Pierre Baldi, and Daniel Whiteson. End-to-end latent variational diffusion models for inverse problems in high energy physics, 2023. URL https://arxiv.org/abs/2305.10399.

[9] Alexander Shmakov, Kevin Greif, Michael James Fenton, Aishik Ghosh, Pierre Baldi, and Daniel Whiteson. Full event particle-level unfolding with variable-length latent variational diffusion, 2024. URL https://arxiv.org/abs/2404.14332.

[10] Sascha Diefenbacher, Guan-Horng Liu, Vinicius Mikuni, Benjamin Nachman, and Weili Nie. Improving generative model-based unfolding with schrödinger bridges, 2023. URL https://arxiv.org/abs/2308.12351.

[11] Anja Butter, Tomas Jezo, Michael Klasen, Mathias Kuschick, Sofia Palacios Schweitzer, and Tilman Plehn. Kicking it off(-shell) with direct diffusion, 2024. URL https://arxiv.org/abs/2311.17175.

[12] Nathan Huetsch, Javier Mariño Villadamigo, Alexander Shmakov, Sascha Diefenbacher, Vinicius Mikuni, Theo Heimel, Michael Fenton, Kevin Greif, Benjamin Nachman, Daniel Whiteson, Anja Butter, and Tilman Plehn. The landscape of unfolding with machine learning, 2024. URL https://arxiv.org/abs/2404.18807.

[13] S. Ask, I.V. Akin, L. Benucci, A. De Roeck, M. Goebel, and J. Haller. Real emission and virtual exchange of gravitons and unparticles in pythia8. Computer Physics Communications, 181(9):1593–1604, September 2010. ISSN 0010-4655. doi: 10.1016/j.cpc.2010.05.013. URL http://dx.doi.org/10.1016/j.cpc.2010.05.013.

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. URL https://arxiv.org/abs/2006.11239.

[15] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

[16] Christian Bierlich, Smita Chakraborty, Nishita Desai, Leif Gellersen, Ilkka Helenius, Philip Ilten, Leif Lönnblad, Stephen Mrenna, Stefan Prestel, Christian T. Preuss, Torbjörn Sjöstrand, Peter Skands, Marius Utheim, and Rob Verheyen. A comprehensive guide to the physics and usage of pythia 8.3, 2022. URL https://arxiv.org/abs/2203.11601.

[17] Cédric Villani. The wasserstein distances, 2009. URL https://doi.org/10.1007/978-3-540-71050-9_6.

[18] S. Kullback and R. A. Leibler. On Information and Sufficiency. The Annals of Mathematical Statistics, 22(1):79 – 86, 1951. doi: 10.1214/aoms/1177729694. URL https://doi.org/10.1214/aoms/1177729694.

Appendices

A cDDPM Loss Derivation

In the proposed cDDPM, the forward process is a Markov chain that gradually adds Gaussian noise to the data according to a variance schedule β.

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t\,; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}\ ,\ \beta_t\,\mathbf{I}) \tag{5}$$

To recover the original sample from a Gaussian noise input, this process needs to be reversed. This can be achieved through the use of a model p_θ which corresponds to the joint distribution $p_\theta(x_{0:T}|y) = p_\theta(x_0, x_1, ...x_T|y)$, and it is defined as a Markov chain with learned Gaussian transitions starting at $p(x_T|y) = \mathcal{N}(x_T; 0, I)$

$$p_\theta\left(\mathbf{x}_{0:T}|\mathbf{y}\right) := p(\mathbf{x}_T|\mathbf{y}) \prod_{t=1}^{T} p_\theta\left(\mathbf{x}_{t-1}|\mathbf{x}_t\,, \mathbf{y}\right) \tag{6}$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t\,, \mathbf{y}) := \mathcal{N}\big(\mathbf{x}_{t-1}\,; \boldsymbol{\mu}_\theta(t, \mathbf{x}_t, \mathbf{y}),\, \Sigma_\theta(t, \mathbf{x}_t, \mathbf{y})\big) \tag{7}$$

where μ_θ represents the learned mean, and Σ_θ represents the learned covariance of the Gaussian transitions, which vary with time step t:

$$\Sigma_\theta(t, \mathbf{x}_t, \mathbf{y}) = \sigma^2\mathbf{I},\ \sigma^2 = \beta_t. \tag{8}$$

Training involves learning the reverse Markovian transitions that maximize the likelihood of the training samples, which is equivalent to minimizing the variational upper bound on the negative log likelihood. This negative log likelihood can be expressed in terms of the Kullback-Leibler (KL) divergence [18], a statistical measure of the difference between two probability distributions P and Q:

$$D_{KL}(P\|Q) = \sum_{x\in X} P(x)\left(\log\frac{P(x)}{Q(x)}\right) \tag{9}$$

Applying this, the variational bound on the negative log likelihood can be expressed as:

$$\begin{aligned}
\mathbb{E}\big[-\log\, p_\theta\left(\mathbf{x}_0|\mathbf{y}\right)\big] &\leq \mathbb{E}\big[-\log\, p_\theta\left(\mathbf{x}_0|\mathbf{y}\right)\big] + D_{KL}\left(q(\mathbf{x}_{1:T}|\mathbf{x}_0)\|p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0,\mathbf{y})\right) \\
&= \mathbb{E}\big[-\log\, p_\theta\left(\mathbf{x}_0|\mathbf{y}\right)\big] + \mathbb{E}_q\left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta\left(\mathbf{x}_{1:T}|\mathbf{x}_0,\mathbf{y}\right)}\right] \\
&= \mathbb{E}\big[-\log\, p_\theta\left(\mathbf{x}_0|\mathbf{y}\right)\big] + \mathbb{E}_q\left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta\left(\mathbf{x}_{0:T}|\mathbf{y}\right)/p_\theta(\mathbf{x}_0|\mathbf{y})}\right] \\
&= \mathbb{E}\big[-\log\, p_\theta\left(\mathbf{x}_0|\mathbf{y}\right)\big] + \mathbb{E}_q\left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta\left(\mathbf{x}_{0:T}|\mathbf{y}\right)}\right] + \mathbb{E}\big[\log\, p_\theta\left(\mathbf{x}_0|\mathbf{y}\right)\big] \\
&= \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T}|\mathbf{y})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] \\
&= \mathbb{E}_q\left[-\log\, p(\mathbf{x}_T|\mathbf{y})\, - \sum_{t\geq 1} \log\, \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{y})}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] := L
\end{aligned} \tag{10}$$

Following the similar derivation provided in [14], this loss can then be rewritten using the KL-divergence

$$\begin{aligned}
L &= \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T}|\mathbf{y})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] \\
&= \mathbb{E}_q\left[-\log\, p(\mathbf{x}_T|\mathbf{y})\, - \sum_{t\geq 1} \log\, \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{y})}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] \\
&= \mathbb{E}_q\left[-\log\, p(\mathbf{x}_T|\mathbf{y})\, - \sum_{t>1} \log\, \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{y})}{q(\mathbf{x}_t|\mathbf{x}_{t-1})} - \log\, \frac{p_\theta(\mathbf{x}_0|\mathbf{x}_1,\mathbf{y})}{q(\mathbf{x}_1|\mathbf{x}_0)}\right] \\
&= \mathbb{E}_q\left[-\log\, p(\mathbf{x}_T|\mathbf{y})\, - \sum_{t>1} \log\, \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{y})}{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)} \cdot \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} - \log\, \frac{p_\theta(\mathbf{x}_0|\mathbf{x}_1,\mathbf{y})}{q(\mathbf{x}_1|\mathbf{x}_0)}\right] \\
&= \mathbb{E}_q\left[-\log\, \frac{p(\mathbf{x}_T|\mathbf{y})}{q(\mathbf{x}_T|\mathbf{x}_0)}\, - \sum_{t>1} \log\, \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{y})}{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}\, - \log\, p_\theta(\mathbf{x}_0|\mathbf{x}_1,\mathbf{y})\right] \\
&= \mathbb{E}_q\left[\underbrace{D_{KL}\big(q(\mathbf{x}_T|\mathbf{x}_0)\parallel p(\mathbf{x}_T|\mathbf{y})\big)}_{L_T} + \sum_{t>1} \underbrace{D_{KL}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)\parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{y})\big)}_{L_{1:T-1}} - \underbrace{\log p_\theta(\mathbf{x}_0|\mathbf{x}_1,\mathbf{y})}_{L_0}\right]
\end{aligned} \tag{11}$$

The term L_T is a constant, as it is the KL-divergence between two distributions of pure noise, and the L_0 term is a final denoising step with no comparison to the forward process posteriors. For the term $L_{1:T-1}$, the forward process posteriors can be written as

$$q(\mathbf{x}_{t+1}|\mathbf{x}_t,\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t+1};\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0),\tilde{\beta}_t I) \tag{12}$$

$$\begin{aligned}
\text{where } \tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t,\ \bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s) \\
\text{and } \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0) = \left(\frac{\beta_t\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\bar{\alpha}_t}(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)}\mathbf{x}_t\right).
\end{aligned} \tag{13}$$

Using this forward process posterior together with the reverse process posterior defined in Equation A.3, a parametrization for $\boldsymbol{\mu}_\theta(\mathbf{x}_t,t,\mathbf{y})$ is introduced that aims to predict $\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0)$. With this the loss becomes

$$L_{t-1} = \mathbb{E}\left[\frac{1}{2\sigma_t^2}\parallel \tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t,\mathbf{x}_0\right) - \boldsymbol{\mu}_\theta\left(t,\mathbf{x}_t,\mathbf{y}\right)\parallel^2\right] + C \tag{14}$$

where C is a constant, and $\tilde{\boldsymbol{\mu}}_t$ and $\boldsymbol{\mu}_\theta$ can be reparametrized using $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}$ and reduced to

$$L_{t-1} = \mathbb{E}_{\boldsymbol{\epsilon},\mathbf{x}_0,\mathbf{y}}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)} \parallel \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(t, \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \mathbf{y}\right) \parallel^2\right]. \tag{15}$$

Finally we can write a simplified version of the loss with the terms differentiable in θ as

$$\begin{aligned} L_{simple}(\theta) &= \mathbb{E}_{\boldsymbol{\epsilon},\mathbf{x}_t,\mathbf{y}}\left[\parallel \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(t, \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \mathbf{y}\right) \parallel^2\right] \\ &= \mathbb{E}_{\boldsymbol{\epsilon},\mathbf{x}_t,\mathbf{y}}\left[\parallel \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(t, \mathbf{x}_t, \mathbf{y}\right) \parallel^2\right]. \end{aligned} \tag{16}$$

This derivation shows that in the cDDPM formulation, the task of learning a posterior distribution reduces to minimizing a simple mean squared error between added and predicted noise. This allows for estimation of the posterior without requiring explicit evaluation of the prior distribution.

B Model Details and Pseudocode

During inference, the inputs are given to the denoising process are the vector $\mathbf{y}$ and random noise values $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The denoising process removes noise from $\mathbf{x}_T$ in T steps according to the learned conditional distribution $p_\theta(\mathbf{x}_{0:T}|\mathbf{y})$. Pseudocode for the training and sampling algorithms can be seen in Figures 4 and 5.

The cDDPM architecture consists of a Multi-Layer Perceptron (MLP), a feedforward neural network, with approximately 1 million trainable parameters. It comprises three main components: an initial linear layer with Gaussian Error Linear Unit (GELU) activation, which provides smooth non-linear transformations, a time step embedding layer, and a series of linear layers with GELU activations. The network takes as input the noised data and the time step. It first processes the input through a 256-unit hidden layer, then adds a learned time step embedding. This combined representation is passed through four 512-unit hidden layers, followed by a 256-unit layer. Skip connections are employed between the input and output of the main block. The final output layer predicts the noise at the given time step. Dropout (rate 0.01) is applied after each linear layer to prevent overfitting during training.

The diffusion process employs a linear variance schedule over T = 500 time steps. The schedule starts with an initial noise level β_1 = 1e-4 at the first step and increases linearly to β_T = 0.02 at the final step. The model is trained using the Adam optimizer with an initial learning rate of 3e-4. To improve convergence and performance, a linear learning rate scheduler is employed. It starts at the initial rate and linearly decreases to 1% of the initial rate (3e-6) by the end of training.

The model is trained for 5000 epochs with a batch size of 2048. Using an NVIDIA A100 GPU, the training procedure on our full dataset or 1.8 million data points completes in approximately 3 hours. Once trained, the model demonstrates efficient inference capabilities. Unfolding a dataset of 1 million data points takes approximately 3 minutes on the A100 GPU, with processing time scaling linearly with the number of jets. Notably, this model functions as a generalizing unfolder, eliminating the need for retraining when applying it to various different datasets.

Algorithm 1 Conditional DDPM: Training

Input: dataset $\{\mathbf{x}_0, \mathbf{y}\}$, variance schedule β_1, ... β_T
$t \leftarrow$ Uniform$(\{1, \ldots, T\})$
$\bar{\alpha}_t \leftarrow \prod_{s=1}^{t}(1-\beta_s)$
$\boldsymbol{\epsilon} \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$
Repeat

a) $\mathbf{x}_t \leftarrow \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}$

b) Calculate loss, $L = ||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(t, \mathbf{x}_t, \mathbf{y}\right)||^2$

c) Update θ via $\nabla_\theta L$

Until converged

Figure 4: The training procedure for the conditional DDPM unfolding model is presented. The algorithm trains on data samples $\{\mathbf{x}_0, \mathbf{y}\}$. In step (a) Gaussian noise $\boldsymbol{\epsilon}$ is added to $\mathbf{x}_0$ over T timesteps according to the variance schedule. The model parameterized by θ is trained to estimate this added noise by observing the noisy states $\mathbf{x}_t$ at a timestep t and the condition $\mathbf{y}$.

Algorithm 2 Conditional DDPM: Sampling

Input: detector-level data vector $\mathbf{y}$, variance schedule β_1, ... β_T
$\mathbf{x}_T \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$
For $t = T, ..., 1$ **do**

a) $\alpha_t \leftarrow 1 - \beta_t$, $\bar{\alpha}_t \leftarrow \prod_{s=1}^{t}\alpha_s$, $\sigma_t \leftarrow \sqrt{\beta_t}$

b) $\mathbf{z} \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} \leftarrow 0$

c) $\mathbf{x}_{t-1} \leftarrow \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\,\boldsymbol{\epsilon}_\theta\left(t, \mathbf{x}_t, \mathbf{y}\right)\right) + \sigma_t\,\mathbf{z}$

Return $\mathbf{x}_0$

Figure 5: The trained conditional DDPM model serves as a posterior sampler, generating unfolded truth-level samples $\mathbf{x}_0$ given condition $\mathbf{y}$. Starting from pure noise $\mathbf{x}_T$, the conditioned reverse process denoises $\mathbf{x}_t$ at each timestep by removing the estimated injected noise. Here $\sigma_t \equiv \sqrt{\beta_t}$ since this choice is optimal for a non-deterministic $\mathbf{x}_0$.