Eur. Phys. J. C (2023) 83:843
https://doi.org/10.1140/epjc/s10052-023-11989-8

THE EUROPEAN PHYSICAL JOURNAL C

Regular Article - Theoretical Physics

ELSA: enhanced latent spaces for improved collider simulations

Benjamin Nachman[1,2,a], **Ramon Winterhalder**[3,b]

[1] Physics Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA
[2] Berkeley Institute for Data Science, University of California, Berkeley, CA 94720, USA
[3] CP3, Université Catholique de Louvain, 1348 Louvain-la-Neuve, Belgium

Received: 23 May 2023 / Accepted: 29 August 2023

Abstract Simulations play a key role for inference in collider physics. We explore various approaches for enhancing the precision of simulations using machine learning, including interventions at the end of the simulation chain (reweighting), at the beginning of the simulation chain (preprocessing), and connections between the end and beginning (latent space refinement). To clearly illustrate our approaches, we use W + jets matrix element surrogate simulations based on normalizing flows as a prototypical example. First, weights in the data space are derived using machine learning classifiers. Then, we pull back the data-space weights to the latent space to produce unweighted examples and employ the Latent Space Refinement (LASER) protocol using Hamiltonian Monte Carlo. An alternative approach is an augmented normalizing flow, which allows for different dimensions in the latent and target spaces. These methods are studied for various pre-processing strategies, including a new and general method for massive particles at hadron colliders that is a tweak on the widely-used RAMBOONDIET mapping. We find that modified simulations can achieve sub-percent precision across a wide range of phase space.

Contents

1 Introduction

A cornerstone of modern high-energy collider physics is first-principles simulations that encode the underlying physical laws. These simulations are crucial for performing and interpreting analyses, linking theoretical predictions and experimental measurements. However, the most precise simulations are computationally demanding and may become prohibitively expensive for data analysis in the era of the High-Luminosity Large Hadron Collider (HL-LHC) [1,2]. As a result, various fast simulation approaches have been developed to mimic precise simulations with only a fraction of the computational overhead.

In the language of machine learning (ML), fast simulations are called *surrogate models*. While most fast simulations use some flavor of machine learning, a growing number of proposed surrogate models are based on deep learning. Deep generative models include Generative Adversarial Networks (GAN) [3,4], Variational Autoencoders [5,6], Normalizing Flows [7,8], and Diffusion Models [9–11] which often make use of score matching [12–14].

In collider physics, deep learning-based surrogate models have been proposed for all steps in the simulation chain, including phase-space sampling, amplitude evaluation, hard-scatter processes, minimum bias generation, parton shower

[a] e-mail: bpnachman@lbl.gov
[b] e-mail: ramon.winterhalder@uclouvain.be (corresponding author)

Published online: 22 September 2023

Monte Carlo, and detector effects (see e.g. Refs. [15,16] for recent reviews). In this paper, we focus on the emulation of hard-scatter event generation [17–21], although the approaches are more general.

Simulations can be viewed as deterministic functions from a set of fixed and random numbers into structured data. The fixed numbers represent physics parameters like masses and couplings, while the random numbers give rise to outgoing particle properties sampled from various phase space distributions. For deep generative models, the random numbers are typically sampled from simple probability densities like the Gaussian or uniform probability distributions. The random numbers going into a simulation constitute the *latent space* of the generator. A key challenge with deep generative models is achieving precision. One way to improve the precision is to modify the map from random numbers to structure. A complementary approach is modifying the probability density of the generated data space or the latent space. This approach is what we investigate in this paper.

For a variety of applications, normalizing flows have shown impressive precision for HEP applications [21–27].However, these models suffer from topological obstructions due to their bijective nature [28–30] which for instance becomes relevant for multi/resonant processes and at sharp phase-space boundaries that are often introduced by kinematic cuts. Topological obstructions are not just affecting normalizing flows but can also limit the performance of autoencoders [31]. If the phase-space structure is known, the problematic phase-space regions can be either smoothed by a local transformation [21] or the flow can be directly defined on the correct manifold [32]. However, we generally do not know where these problematic regions lie in the highly complex and multi-dimensional phase space. Moreover, even if we identify and fix problematic patterns in a lower-dimensional subspace, we are not guaranteed to cure all higher-dimensional topological structures. Therefore, we study two complementary strategies to improve the precision of generative networks affected by topological obstructions and other effects. These strategies are both generalizable and computationally inexpensive.

As a first method, we use the LASER protocol [30] which relies on the refinement of the latent space of the generative model. The LASER protocol is an extension of the DCTRGAN [33] method that is not specific to GANs and circumvents the limited statistical power of weighted events by propagating the weights into the latent space. While this idea has also been proposed solely for GANs in Ref. [34], the LASER approach works for any generative model and further extends Ref. [34] by allowing more sophisticated sampling techniques.

In the second method, we employ augmented normalizing flows [28,35–37]. In contrast to standard normalizing flows, the feature space is augmented with additional variables such that the latent space dimension can be chosen to have higher dimensions than the original feature space dimension. As a result, possibly disconnected patches in the feature space, which are difficult to describe by deep generative models, can be connected by augmenting additional dimensions and thus simplify the mapping the network has to learn. In order to obtain the original feature space, we use a lower/dimensional slice of the generated feature space.

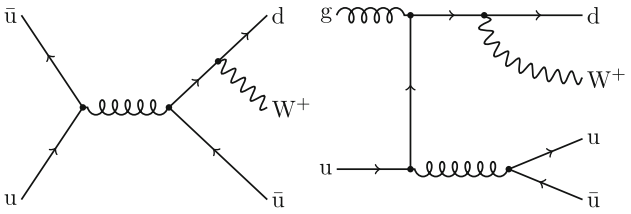

Fig. 1 Examples of LO Feynman diagrams contributing to the $pp \to W^+ + \text{jets}$ process

The LASER protocol and the augmented flow show excellent results when applied to simple toy examples. In this paper, we apply both methods to a complicated high-energy physics example for the first time and show how they alleviate current limitations in precision while being conservative regarding network sizes and the number of parameters. Note that augmentation only describes one possible incarnation of a surjective transformation, of which some have already applied on HEP data successfully in Ref. [26]. In our studies, we consider W-boson production with associated jets [38–45] at the LHC, i.e.

$$pp \to W^+ + \text{jets}, \tag{1.1}$$

as illustrated in Fig. 1. Owing to the high multiplicity of the multi-jet process and phase space restrictions (e.g. jet p_T thresholds), the associated phase space is large and topologically complex, with many non-trivial features that need to be correctly addressed by the generative models.

This paper is organized as follows. In Sect. 2, we introduce and compare both methods and describe their implementation. We define and list different data representations in Sect. 3 and introduce the MAHAMBO algorithm. As a first application, we show simple 2/dimensional toy examples to illustrate our approaches in Sect. 4. Afterwards, we employ both methods on a fully comprehensive LHC example and investigate the impact of various data representations on the models' performance. Finally, we conclude in Sect. 5.

2 Methods

The baseline method for both of our models is a normalizing flows [7,8]. Normalizing flows rely on the change of variables formula

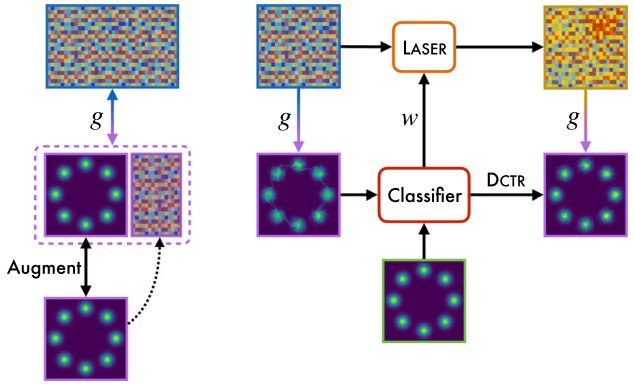

Fig. 2 A schematic diagram illustrating the augmented flow (left) and the LASER protocol (right)

$$\log p_{g(\mathcal{Z})}(x) = \log p_{\mathcal{Z}}(g^{-1}(x)) + \log \left| \frac{\partial g^{-1}(x)}{\partial x} \right|, \qquad (2.1)$$

which explicitly encodes an estimate of the probability density $p_{g(\mathcal{Z})} \approx p_{\mathcal{X}}$ by introducing a bijective mapping g (parameterized as a neural network) from a latent space $\mathcal{Z} = \{z \in \mathbb{R}^N | z \sim p_{\mathcal{Z}}\}$ to a feature space with data described by $\mathcal{X} = \{x \in \mathbb{R}^N | x \sim p_{\mathcal{X}}\}$. In contrast, for VAEs and GANs, the mapping g does not need to be bijective. However, VAEs only yield a lower bound of the density (evidence lower bound, or ELBO) and for GANs the posterior is only defined implicitly.

We study two methods that aim to improve the precision and expressiveness of the generative model g by either modifying the latent space prior $p_{\mathcal{Z}}$ or by post-processing the data space, either with importance weights or by augmenting the feature space $\mathcal{X}$ with an additional variable $r \sim q(r|x)$. These methods are illustrated in Fig. 2 and are described in more detail in the following.

2.1 Latent space refinement

As the first method, we use the LASER protocol [30], illustrated on the right of Fig. 2. The LASER method acts as a post-hoc procedure modifying the latent space prior $p_{\mathcal{Z}}$ to improve the generated samples x while keeping the generative model g fixed. For this method, the baseline generator $g(z)$ can be any generative model or physics simulator (as long as we have access to the input random numbers) and does not have to be bijective in general. Throughout this paper, we choose g as a normalizing flow for simplicity and to make this method better comparable with the second method.

As a first step in the LASER protocol, we train a classifier f to distinguish samples x generated by g and samples drawn from the truth data probability distribution, $p_{\mathcal{X}}$. If the generative model g is part of a GAN, it is possible to use the discriminator d as the classifier [34], but it might be more appropriate to train a new unbiased network from scratch [33]. If this classifier is trained with the binary cross entropy (BCE) and converges to its optimum, it obeys the well-known (see e.g. Refs. [46,47]) property

$$f(x) \approx \frac{p_{\mathcal{X}}}{p_{g(\mathcal{Z})}(x) + p_{\mathcal{X}}}. \qquad (2.2)$$

This can be used to assign a weight

$$w(x) = \frac{f(x)}{1 - f(x)} \approx \frac{p_{\mathcal{X}}}{p_{g(\mathcal{Z})}(x)}, \qquad (2.3)$$

to each generated sample. This idea has been widely studied in HEP for simulation reweighting and inference (see e.g. Refs. [48–60]) and is the core of the DCTR protocol [57] that has also been proposed for simulation surrogate refinement [33,61] so we will refer to events weighted in this way as DCTR.

If weighted events are acceptable, then one could stop here and use the weights in the data space (as was proposed in Ref. [33]). For LASER, we next backpropagate this weight to the corresponding latent space point $g(z_i) = x_i$.

If g is not bijective, then the weights pulled back from the data space to the latent space can be made a proper function of the phase space via Step 2 of the OMNIFOLD algorithm [30,58]. Finally, we sample from the new weighted latent space $(z, \bar{w}(z))$. There are three possible ways proposed by the LASER protocol:

- **Rejection sampling:** The weights $w(z)$ can be used to perform rejection sampling on the latent space. In general, however, the weights are broadly distributed which makes this method inefficient and least favored.
- **Markov chain Monte Carlo:** The weights $\bar{w}(z)$ induce a modified probability distribution

$$q(z, \bar{w}) = p_{\mathcal{Z}}(z)\, \bar{w}(z)/Z_0, \qquad (2.4)$$

  with some normalization constant Z_0. If both the probability q and its derivative $\partial q/\partial z$ are tractable, we can employ a Hamiltonian Monte Carlo (HMC) [62,63] algorithm for sampling. Differentiability is natural when w is a neural network.
- **Refiner network** Φ: Finally, if the derivative $\partial q/\partial z$ is not tractable the weighted latent space $(z, w(z))$ can instead be converted into a unweighted latent space. For this, another generative model Φ is trained on the weighted latent space [22,64] while generating unweighted samples following the same probability distribution $p_\Phi(z) \approx q(z, \bar{w})$. The output of the refinement network Φ can then be used as an input to the generative model g.

In contrast to the MCMC algorithm, the refiner network Φ requires the training of an additional generative model which

might be affected by the same topological obstructions or even evokes new type of problems. Therefore, when applying the LASER protocol in our experiments, we always use a HMC algorithm to sample from the refined latent space when possible.

2.2 Augmented normalizing flows

Augmented normalizing flows [28,35–37] are bridging the gap between VAEs and NFs. While they allow for (partially) bijective mappings they also tackle the bottleneck problem [36] by increasing the dimensionality of the original data. As it has been pointed out in great detail in Ref. [37], augmented flows only represent a particular example of surjective normalizing flows. In general, there are multiple and potentially interesting surjective transformations that might enhance the overall performance of the network [26]. In our studies, we restrict ourselves to only investigate the effect of an augmentation transformation. As all necessary formulae for surjective transformations have been derived and documented exhaustively in the literature, we only explain briefly how the augmentation transformation is defined and refer for details to Ref. [37].

Augmentation (or slicing in the inverse direction) starts by augmenting the data x with an additional D_k dimensional random variable $z_2 \in \mathbb{R}^{D_k}$ which is drawn from a distribution $q(z_2)$. While not necessary in our application, the more general case allows for z_2 to be conditional on the data x. In all our examples, $q(z_2)$ is simply given by a normal distribution, i.e. $q(z_2) = \mathcal{N}(0, 1)$. This means that the forward and the inverse passes are given by

$$x \quad \underset{\leftarrow \text{slice}}{\overset{\text{augment}\rightarrow}{\longleftrightarrow}} \quad (z_1, z_2), \qquad \text{with} \quad z_1 = x, \quad \text{and} \quad z_2 \sim \mathcal{N}(0, 1). \tag{2.5}$$

As this mapping is surjective in the inverse direction but stochastic in forward direction, the full likelihood of this transformation is not tractable. Consequently, the augmented flow is optimized using only the tractable likelihood contribution which is the ELBO, and connects surjective flows with VAEs.

We note that a similar idea was also introduced in Ref. [65] where a noise-extended INN was used to match the dimensions of parton-level and detector-level events to better describe the stochastic nature of detector effects.

3 Data representation

While it is well known that proper data preprocessing can increase the accuracy and reliability of a neural network, it is often difficult to find the best data representation for any particular dataset.

In order to better understand which data representations are well suited in the context of event generation, we investigate several different proposals from the literature and develop our own parametrizations, which are built to enhance the sensitivity of the neural network dependent on its specific task. In detail, we consider parton-level events from a hadronic $2 \to n$ scattering process with an intrinsic dimensionality of $d = 3n - 2$. We implemented the following preprocessings:

1. **4-Mom**: We parametrize each final state particle in terms of its four momentum and represent the entire event by $4n$ features

$$x = \{p_1, p_2, \ldots, p_n\} \tag{3.1}$$

where n_f denotes the number of final-state particles, and standardize each feature according to

$$\tilde{x}_i = \frac{x_i - \mu_i}{\sigma_i}. \tag{3.2}$$

2. **MinRep**: We start by considering the 3-momenta of all final state particles, neglecting the energy component due to on-shell conditions. Further, in order to obey momentum conservation in the transverse plane, we reject the p_x and p_y components of the last particle and treat them as dependent quantities. This reduces the dataset to a $3n-2$ dimensional data representation. Again, we normalize all features using the standardization in Eq. (3.2).
3. **PRECISESIAST**: In this preprocessing, we follow the preprocessing suggested in Ref. [21] and thus denote it as *precision enthusiast* (PRECISESIAST) parameterization. For parton level events, this preprocessing starts by representing each finals state in terms of

$$\{p_T, \eta, \phi\}, \tag{3.3}$$

followed by the additional transformations

$$p_T \to \tilde{p}_T = \log(p_T - p_{T,\min}), \tag{3.4}$$
$$\phi_{j_i} \to \widetilde{\phi}_{j_i} = \operatorname{atanh}(\Delta\phi_{Wj_i}/\pi), \tag{3.5}$$
$$\phi_W \to \widetilde{\phi}_W = \operatorname{atanh}(\phi_W/\pi), \tag{3.6}$$

to Gaussianize all features. This leads to $3n$/dimensional representation.
4. **MAHAMBO**: In this preprocessing, we map the phase space onto a unit-hypercube with $3n - 2$ dimensions. Hence, we reduce the representation to the relevant degrees of freedom similar to the MinRep approach. While the MinRep parametrization is dependent on the

ordering of the momenta and hence potentially prone to instabilities, the MAHAMBO preprocessing relies on the RAMBO [66,67] algorithm which populates the phase space uniformly (at least for massless final states). The details about the MAHAMBO algorithm are given in more detail in Sect. 3.1.

5. **ELFS**: This starts from a PRECISESIAST inspired representation

$$\{\log p_T, \widetilde{\phi}, \eta\} \tag{3.7}$$

and additionally augments with the features

$$\{\text{atanh}(2\Delta\phi_{ij}/\pi - 1), \log(\Delta\eta_{ij}), \log(\Delta R_{ij})\} \tag{3.8}$$

for all possible jets pairs ij, where we define

$$\begin{aligned}\Delta\eta_{ij} &= |\eta_i - \eta_j| \\ \Delta\phi_{ij} &= \begin{cases} |\phi_i - \phi_j|, & \text{if } |\phi_i - \phi_j| < \pi \\ 2\pi - |\phi_i - \phi_j|, & \text{otherwise} \end{cases} \\ \Delta R_{ij} &= \sqrt{(\Delta\phi_{ij})^2 + (\Delta\eta_{ij})^2}. \end{aligned} \tag{3.9}$$

This yields a $3n + k$ dimensional representation, where $k = \{3, 9\}$ for 2- or 3-jet events considered in our studies, respectively. Thus, this is denoted as *enlarged feature space* (ELFS) representation.

The list of possible parametrizations above is not exhaustive, but they cover common representations and thus we expect them to be representative.

3.1 The MAHAMBO algorithm

In order to reduce the dataset to the relevant degrees of freedom while avoiding the limitations of the MinRep parametrization, which is sensitive to the order of the particle momenta, we employ a modified version of the RAMBO [66,67] algorithm which allows for an invertible mapping between momenta p and random numbers r living on the unit-hypercube $U = [0, 1]^d$. While the original RAMBO [66] algorithm is not invertible as it requires $4n$ random numbers to parametrize n particle momenta, the RAMBOONDIET [67] algorithm solves this problem and only requires $3n - 4$ random numbers. By construction, the RAMBOONDIET algorithm is defined for a fixed partonic center-of-mass (COM) energy and provides momenta within this frame.

However, as we are considering scattering processes at hadron colliders the corresponding events only obey momentum conservation in the transverse plane and are only observed in the lab frame. Consequently, we need two additional random numbers r_{3n-3}, r_{3n-2} that parametrize the momentum fractions x_1 and x_2 of the scattering partons, which are relevant for the PDFs and connect the lab frame with the partonic COM frame via a boost along the z-axis. This boost can be parametrized in terms of a rapidity parameter

$$\zeta = \frac{1}{2}\log\frac{x_1}{x_2} = \text{atanh}\,\frac{q^3}{q^0}, \quad \text{with } q = \sum_i p_i. \tag{3.10}$$

In particular, we choose the following parametrization

$$\tau = \tau_{\min}^{r_{3n-3}}, \quad x_1 = \tau^{r_{3n-2}}, \quad x_2 = \tau^{1-r_{3n-2}}, \tag{3.11}$$

with $\tau_{\min}$ reflecting a possible cut on the squared partonic center-of-mass energy $\hat{s} > \hat{s}_{\min} = \tau_{\min} s$. We denote the modified algorithm as MAHAMBO.

The derivation and proofs for the unmodified parts adopted from RAMBOONDIET are given in Ref. [67].[1]

Algorithm 1: The MAHAMBO algorithm.

$\tau \leftarrow \tau_{\min}^{r_{3n-3}}, x_1 \leftarrow \tau^{r_{3n-2}}, x_2 \leftarrow \tau^{1-r_{3n-2}}, \mathcal{W} \leftarrow \sqrt{\tau s}$
$Q \leftarrow (\mathcal{W}, 0, 0, 0), Q_1 \leftarrow Q, M_1 \leftarrow \sqrt{Q^2}, M_n \leftarrow 0$
for $i = 2, \ldots, n$ **do**
  if $i \neq n$ **then**
    solve $r_{i-1} = (n+1-i)u_i^{2(n-i)} - (n-i)u_i^{2(n+1-i)}$ for u_i
    $M_i \leftarrow u_2 \ldots u_i \sqrt{Q^2}$
  end if
  $\cos\theta_i \leftarrow 2r_{n-5+2i} - 1, \sin\theta_i \leftarrow \sqrt{1-\cos^2\theta_i}$, $\phi_i \leftarrow 2\pi r_{n-4+2i}$
  $q_i \leftarrow 4M_{i-1}\rho(M_{i-1}, M_i, 0)$
  $\mathbf{p}_{i-1} \leftarrow q_i(\cos\phi_i \sin\theta_1, \sin\phi_i \sin\theta_i, \cos\theta_i)$
  $p_{i-1} \leftarrow (q_i, -\mathbf{p}_{i-1}), Q_i \leftarrow (\sqrt{q_i^2 + M_i^2}, -\mathbf{p}_{i-1})$
  boost p_{i-1} and Q_i by $\mathbf{Q}_{i-1}/Q_{i-1}^0$
end for
$p_n \leftarrow Q_n$
solve $\mathcal{W} = \sum_{i=1}^n \sqrt{\xi^2(p_i^0)^2 + m_i^2}$ for ξ
for $i = 1, \ldots, n$ **do**
  $p_i^0 \leftarrow \sqrt{\xi^2(p_i^0)^2 + m_i^2}, \mathbf{p}_i \leftarrow \xi\mathbf{p}_i$
  boost p_i by $\Lambda_z(\zeta)$ with rapidity $\zeta \leftarrow \frac{1}{2}\log\frac{x_1}{x_2}$
end for
return $\{p_1, \ldots, p_n\}$

The MAHAMBO procedure to generate a single phase-space x point starting from random numbers r passes the following steps:

1. Sample the partonic momentum fractions x_1 and x_2 according to Eq. (3.11).

[1] Note that there is an error in the original paper in the transformation from Eq. (8) to Eq. (9). Thus, the algorithm has some missing squares in the algorithm when solving for the variable u_i. These are fixed in our implementation.

2. Define the partonic COM energy $\hat{s} = \sqrt{x_1 x_2 s}$ and define the Lorentz boost parameter ζ according to Eq. (3.10).
3. Pass $\hat{s}$ to the RAMBOONDIET algorithm to generate n massless 4-momenta in the partonic COM frame.
4. Possibly reshuffle these four/momenta following the procedure in Ref. [66] to obtain massive final states.
5. Boost all momenta into the lab frame with $\Lambda_z(\zeta)$.

The full details of the mapping between $3n-2$ random numbers r_i and n 4-momenta is given in Algorithm 1, and its inverse in Algorithm 2.

Algorithm 2: The inverse MAHAMBO algorithm.

1: $q \leftarrow \sum_{j=1}^{n} p_j, \tau \leftarrow q^2/s, \zeta \leftarrow \operatorname{atanh}(q^3/q^0)$
2: $\mathcal{W} \leftarrow \sqrt{\tau s}, \log x_1 \leftarrow \frac{1}{2}\left[\log\tau + \log\left(\frac{1-\zeta}{1+\zeta}\right)\right]$
3: $r_{3n-3} \leftarrow \log\tau/\log\tau_{\min}, r_{3n-2} \leftarrow \log x_1/\log\tau$
4: solve $\mathcal{W} = \sum_{i=1}^{n}\sqrt{\left(\left(p_i^0\right)^2 - m_i^2\right)/\xi^2}$ for ξ
5: **for** $i = 1, \ldots, n$ **do**
6: $\quad p_i^0 \leftarrow \sqrt{\left(\left(p_i^0\right)^2 - m_i^2\right)/\xi^2}, \mathbf{p}_i \leftarrow \mathbf{p}_i/\xi$
7: $\quad$ boost p_i by $\Lambda_z(-\zeta)$ with rapidity ζ
8: **end for**
9: $M_1 \leftarrow \sqrt{\left(\sum_{j=1}^{n} p_j\right)^2}, Q_n \leftarrow p_n$
10: **for** $i = n, \ldots, 2$ **do**
11: $\quad M_i \leftarrow \sqrt{\left(\sum_{j=i}^{n} p_j\right)^2}$
12: $\quad$ **if** $i \neq n$ **then**
13: $\quad\quad u_i \leftarrow M_i/M_{i-1}$
14: $\quad\quad r_{i-1} \leftarrow (n+1-i)u_i^{2(n-i)} - (n-i)u_i^{2(n+1-i)}$
15: $\quad$ **end if**
16: $\quad Q_{i-1} \leftarrow Q_i + p_{i-1}$
17: $\quad$ boost p_{i-1} by $-\mathbf{Q}_{i-1}/Q_{i-1}^0$
18: $\quad r_{n-5+2i} \leftarrow (p_{i-1}^3/|\mathbf{p}_{i-1}| + 1)/2$
19: $\quad \phi \leftarrow \arctan(p_{i-1}^2/p_{i-1}^1), r_{n-4+2i} \leftarrow \phi/2\pi + \Theta(-\phi)$
20: **end for**
21: **return** $\{r_1, \ldots, r_{3n-2}\}$

To solve the equations in line 5 and 15 of Algorithm 1 and line 4 of Algorithm 2 we use Brent's method [68]. The function $\rho(M_{i-1}, M_i, m_{i-1})$ in line 9 corresponds to the two-body decay factor defined as

$$\rho(a,b,c) = \frac{\sqrt{(a^2-(b+c)^2)(a^2-(b-c)^2)}}{8a^2}, \tag{3.12}$$

which simplifies in the massless ($m_i = 0$) case to

$$\rho(M_{i-1}, M_i, 0) = \frac{M_{i-1}^2 - M_i^2}{8M_{i-1}^2}. \tag{3.13}$$

4 Experiments

We demonstrate the performance of LASER and augmented flows in two numerical experiments. We start with two illustrative toy examples in Sect. 4.1, before turning to a particle physics problem in Sect. 4.2. For the more complicated examples, we further perform a dedicated analyses of choosing an optimal parametrization for both the generative model as well as the classifier needed for the DCTR or LASER approach.

In order to have full control of all parameters and inputs we implemented our own version of Hamiltonian Monte Carlo in PYTORCH, using its automatic differentiation module to compute the necessary gradients to run the algorithm. For all examples, we initialize a total of 100 Markov Chains running in parallel to reduce computational overhead and solve the equation of motions numerically using the leapfrog algorithm [63] with a step size of $\epsilon = 0.01$ and 30 leapfrog steps. To avoid artifacts originating from the random initialization of the chains, we start with a burn-in phase and discard the first 5000 points from each chain.

4.1 Toy examples

We consider two different toy examples commonly used in the literature: a probability density defined by thin concentric circles and another density described by circular array of two-dimensional Gaussian random variables. These are illustrated pictorially in the leftmost column of Fig. 3.

Network architecture

We use a normalizing flow with affine coupling blocks as the baseline model in all toy examples. Our implementation relies partially on SURVAE [37]. The NF consists of 8 coupling blocks, each consisting of a fully connected multi-layer perceptron (MLP) with two hidden layers, 32 units per layer, and ReLU activation functions. The model is trained for 100 epochs by minimizing the negative log-likelihood with an additional weight decay of 10^{-5}. The optimization is performed using Adam [69] with default β parameters and $\alpha_0 = 10^{-3}$. We employ an exponential learning rate schedule, i.e. $\alpha_n = \alpha_0\gamma^n$, with $\gamma = 0.995$ which decays the learning rate after each epoch.

In the scenario using the augmented flow, we extend the dimensionality from $2 \to 4$ by augmenting an additional 2-dimensional random number $r \sim \mathcal{N}(0,1)$ to the feature vector x.

For the classifier, we use a simple feed-forward neural network which consists of 8 layers with 80 units each and leaky ReLU activation [70] in the hidden layers. We note that, none of these hyperparameters have been extensively optimized in our studies.

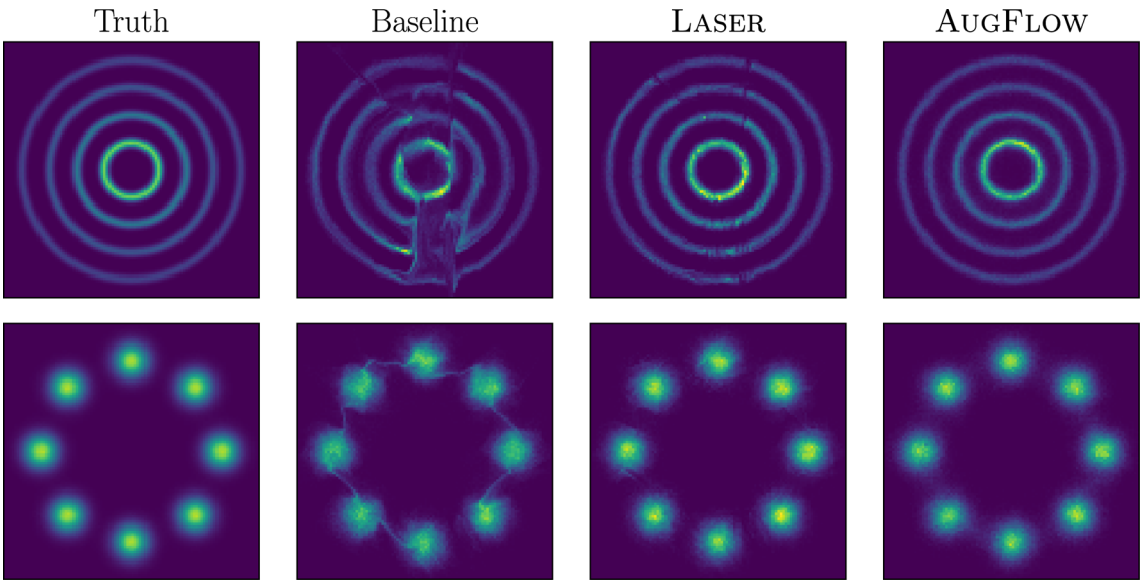

Fig. 3 Comparison of the baseline flow model, the LASER refined output and the augmented flow for the Circle (top) and Gaussians (bottom) toy examples

Probability distance measures

In the 2-dimensional toy examples, we can quantify the performance gain of our LASER-flow and the augmented flow compared to our baseline flow model. For this, we implemented different f-divergences [71]. All f-divergences used in this paper are shown in Table 1.

Except for the total variation (V), the squared Hellinger distance (H^2) and the Jensen–Shannon divergence (JS) are not proper metrics as they do not obey the triangle-inequality. In contrast to the Kullback–Leibler (KL) divergence, which is also commonly used, we only considered f-divergences which are symmetric in p and q.

Additionally, we implemented the earth mover's distance (EMD) [72] between two probability distributions $p(x)$ and $q(x)$. The EMD is directly connected to an optimal transport problem. If the two distributions are represented as a set of tuples $P = \{(x_i, p(x_i)\}_{i=1}^N$ and $Q = \{(y_j, q(y_j)\}_{j=1}^M$, the EMD is the minimum cost required to transform P into Q. This transformation is parameterized by discrete probability flows f_{ij} from $x_i \in P$ to $y_j \in Q$ and reads

$$\begin{aligned} \mathrm{EMD}(p,q) &= \min_{\{f_{ij}>0\}} \sum_{i=1}^{N}\sum_{i=1}^{M} f_{ij}\,||x_i - y_j||_2, \\ \sum_{j=1}^{M} f_{ij} &\le p(x_i), \quad \sum_{i=1}^{N} f_{ij} \le q(y_j), \\ \sum_{i=1}^{N}\sum_{i=1}^{M} f_{ij} &= \min\left(\sum_{i=1}^{N} p(x_i), \sum_{j=1}^{M} q(y_j)\right), \end{aligned} \tag{4.1}$$

where $||\cdot||_2$ is the Euclidean norm and the optimal flow f_{ij} is found by solving the optimization problem. For an efficient numerical implementation we use the POT library [73]. An advantage of the EMD over the f-divergences is that it also takes the metric space of the points x_i and y_i into account.

All f-divergences introduced have been evaluated in a histogram-based way using 1M data points for all models and examples. By comparing the same metrics for two equally large, but independent samples drawn from the truth distribution, we can estimate the uncertainty of the quantities. The extracted error estimates are indicated in parenthesis in Table 2.

We note that we used a total of $10^4 = 100 \times 100$ bins to provide a detailed test of the phase space for evaluating all metrics while still providing stable results for the given statistics. We did not optimize the number of bins, but found that the qualitative conclusions are unaffected by the details of this choice.

Results

In the first toy example, illustrated in the top panel of Fig. 3, we considered multiple concentric circles representing a multimodal distribution that is topologically different than the simple latent space given by a Gaussian random variable. Consequently, the baseline flow fails to properly learn the truth density and generates fake samples which result in a blurred distribution that is not able to properly separate the modes. In contrast, the LASER improved generator as well as the AUGFLOW show highly improved distributions which is both easily detectable by eye as well as quantified by the

Table 1 A comparison of different f-divergences and the earth mover distance (EMD)

Name	Formula $f(p,q)$	Proper metric
Total variation (V)	$\frac{1}{2}\int \mathrm{d}x\ \lvert p(x)-q(x)\rvert$	✓
Squared Hellinger (H^2)	$\frac{1}{2}\int \mathrm{d}x\ \left(\sqrt{p(x)}-\sqrt{q(x)}\right)^2$	×
Jensen–Shannon (JS)	$\frac{1}{2}\int \mathrm{d}x\ p(x)\log\left(\frac{2p(x)}{p(x)+q(x)}\right)+(p\leftrightarrow q)$	×
Earth mover distance (EMD)	Eq. (4.1)	✓

Table 2 Earth mover distance (EMD), total variation (V), squared Hellinger distance (H^2) and Jensen–Shannon divergence (JS) on test data for the baseline model and the proposed methods for various two-dimensional examples. The best results are written in bold face. The errors on the scores are indicated in parenthesis

Method		EMD	V	H^2	JS
Circle	Baseline	0.0153(2)	0.24(3)	0.069(1)	0.0610(9)
	LASER	0.0075(2)	**0.10(3)**	**0.012(1)**	**0.0115(9)**
	AUGFLOW	**0.0041(2)**	0.12(3)	0.024(1)	0.0211(9)
Gaussians	Baseline	0.0048(9)	0.10(3)	0.013(1)	0.012(1)
	LASER	**0.0035(9)**	**0.05(3)**	**0.005(1)**	**0.004(1)**
	AUGFLOW	0.0046(9)	0.07(3)	0.007(1)	0.007(1)

lower divergences given in Table 2. Overall, when looking at the given scores in Table 2, the performance of the LASER improved model is superior over both the baseline model as well as the AUGFLOW, except for the EMD metric where the AUGFLOW yields the best score.

Similarly to the first example, the baseline model is again not expressive enough to adequately describe the second toy example, illustrated in the lower panel of Fig. 3. Both LASER and AUGFLOW are nearly indistinguishable from the truth by eye, which is quantified by even lower scores numerically than for the circle example in Table 2. In this case, the LASER improved flow gives the best score for all evaluated metrics and outperforms the AUGFLOW. This indicates that the LASER protocol seems to typically perform better than the augmentation of additional dimensions, an observation we will make again in the higher dimensional physics examples in Sect. 4.2.

Finally, we want to remark that simply ramping up the number of parameters in the baseline model would also improve its performance in these simple low/dimensional toy examples. However, this parameter growth scales poorly for more complicated and higher/dimensional distributions, leading to increasingly minor improvements. Further, as Ref. [30] noted, making a bijective generative model larger will not fundamentally overcome the topology problem.

4.2 Physics example

As a representative HEP example, we consider the

$$\mathrm{pp} \to \mathrm{W}^+ + \{2,3\}\mathrm{jets} \tag{4.2}$$

scattering process. For all simulations in this paper, we consider a centre-of-mass energy of $\sqrt{s} = 14\,\mathrm{TeV}$. All events used as training and test data are generated at parton level using MADGRAPH5_AMC@NLO [74] (v3.4.1). We use the NNPDF 3.0 NLO PDF set [75] with a strong coupling constant $\alpha_s(M_Z) = 0.119$ in the four flavor scheme provided by LHAPDF6 [76]. In order to stay in the perturbative regime of QCD we apply a cut on the transverse momenta of the jets as

$$p_{\mathrm{T,j}} > 20\,\mathrm{GeV}. \tag{4.3}$$

Additionally, we require the jets to fulfill

$$|\eta_\mathrm{j}| < 6, \quad \Delta R_{\mathrm{jj}} > 0.4. \tag{4.4}$$

We have 1M data points for each jet multiplicity and train on 800k events each. The network is trained on each multiplicity separately. In general, it is also possible to set up an architecture which can be trained on different multiplicities simultaneously [21,26].

Network architecture

For the physics case considered in this work, the affine coupling blocks are replaced with rational quadratic spline (RQS) blocks [77] with 10 spline bins unless mentioned otherwise. In order to cope with the higher dimensions we now use 14 coupling blocks, where each block has 2 hidden layers, 80 units per layer and ReLU activation functions. We now employ a one-cycle training schedule with a starting and

maximum learning rate of $\alpha_0 = 5 \cdot 10^{-4}$ and $\alpha_{\max} = 3 \cdot 10^{-3}$, respectively. The networks are trained for 50 epochs. In contrast to the toy examples, the classifier now consists of 10 layers[2] with 128 units each and all other parameters unchanged.

A possible bottleneck, when running the LASER protocol in terms of a HMC on the latent space, is the necessary fine tuning of the step size ϵ. Usually, the average acceptance rate of a proposed phase-space point is a good indication about the quality of the chosen hyperparameters. In an optimal scenario, the acceptance rate of an HMC should be around ~ 0.65 [63]. Consequently, during the initial burn-in phase, we adapt the step-size to achieve the optimal acceptance rate of 0.65 on average. We found, that this additional adaptive phase was only important for the physics examples considered.

Further, for the augmented flow, we have a freedom in choosing the dimensionality of the augmented random number $r \in \mathbb{R}^{D_r}$. We find that for both physics examples considered we obtain the best performance when choosing $D_r = d$, where d is the dimension of the used data representation. We note, however, that we did not do any extensive tuning of this parameter.

Two-jet events

Before diving into the results of the various tested methods, we want to emphasize that the structure and storyline of what is coming next are pedagogical. We will start by using a very naive model to highlight and explain the reason for its limitations. Only then will we move forward by alleviating most of these issues and finish this section with a recommended workflow for multi-dimensional event generation.

Naive baseline model In our first study, shown in Fig. 4, we tested a simple affine flow as baseline model which has been trained on $W^+ + 2j$ events given in the naive 4-Mom parametrization. This parametrization represents the most simple and straightforward parametrization possible and consequently serves as a benchmark point for all further improvements. Thus, we also kept the affine blocks in this example to keep the architecture as close to the ones used in the toy examples as possible. Within this parametrization, we train our classifier model which subsequently yields the necessary weights to perform the LASER protocol. As we can see in Fig. 4, both LASER[3] and the AUGFLOW improve the precision of the baseline network. However, the improvement obtained by the AUGFLOW is minimal compared to the large gain in precision by using the LASER protocol. In particular, we find that the more expressive the baseline model is, the less effective the AUGFLOW performance is in these high-dimensional examples. Consequently, we omit showing further comparisons of the AUGFLOW with the baseline model for all other tests performed and focus on techniques that are more relevant for reaching a higher precision.

Different parametrizations in the generator Next, we study the effect of using different data preprocessings on the quality of the generated events. In particular, we compare the parametrizations 4-Mom, MinRep, PRECISESIAST and MAHAMBO which have been introduced in Sect. 3. In order to be more compatible with state-of-the-art generators, we now replace the affine blocks with RQS blocks and keep them throughout the rest of our analyses. In the various distributions shown in Fig. 5, we see that the previously tested 4-Mom parametrization is strictly the worst, likely because it has the highest dimensionality. This leaves the network with the task of learning the likelihood of a lower-dimensional manifold within this higher/dimensional representation, making it harder to match the redundant features. Consequently, the naive MinRep parametrization, which merely omits all redundant features in the data representation, outperforms the 4-Mom preprocessing and achieves higher precision in all observables.

The results become even better when picking either the PRECISESIAST or MAHAMBO preprocessing, as can be seen in Fig. 5. At first glance, this seems to somewhat conflict with our conjecture about dimensionality since the PRECISESIAST parametrization only reduces the data representation to $3n$ dimensions and is thus not minimal. However, this parametrization is deliberately designed to perform well at the otherwise hard-to-learn phase-space boundaries introduced by the $p_{T,j}$ cuts. As a result, this parametrization yields the best precision in all transverse-momentum observables. In contrast, the PRECISESIAST parametrization fails to correctly describe the ΔR_{jj} distribution. This is in line with the observations made in Ref. [21], where the authors introduced a 'magic' transformation to smooth out the hard phase-space cut. In comparison, within the MAHAMBO preprocessing, the flow renders this phase-space boundary automatically and is generally more precise in all connected angular observables such as $\Delta\phi_{jj}$ and $\Delta\eta_{jj}$, shown in the lower panel of Fig. 5. While there is indeed still room for some improvement even for the flow using the PRECISESIAST and MAHAMBO representation, we postpone the combination of better preprocessing with refinement (via the LASER protocol) to the $W^+ + 3j$ process, as the impact will be more important.

Three-jet events

The three-jet case is much more complex than the two-jet case because the dimensionality of the data manifold is relatively smaller than the data space dimensionality.

[2] We wanted to be sure in results presented later that the network expressiveness was not a limiting factor.

[3] The data-space DCTR results are nearly indistinguishable from the LASER results and are therefore omitted.

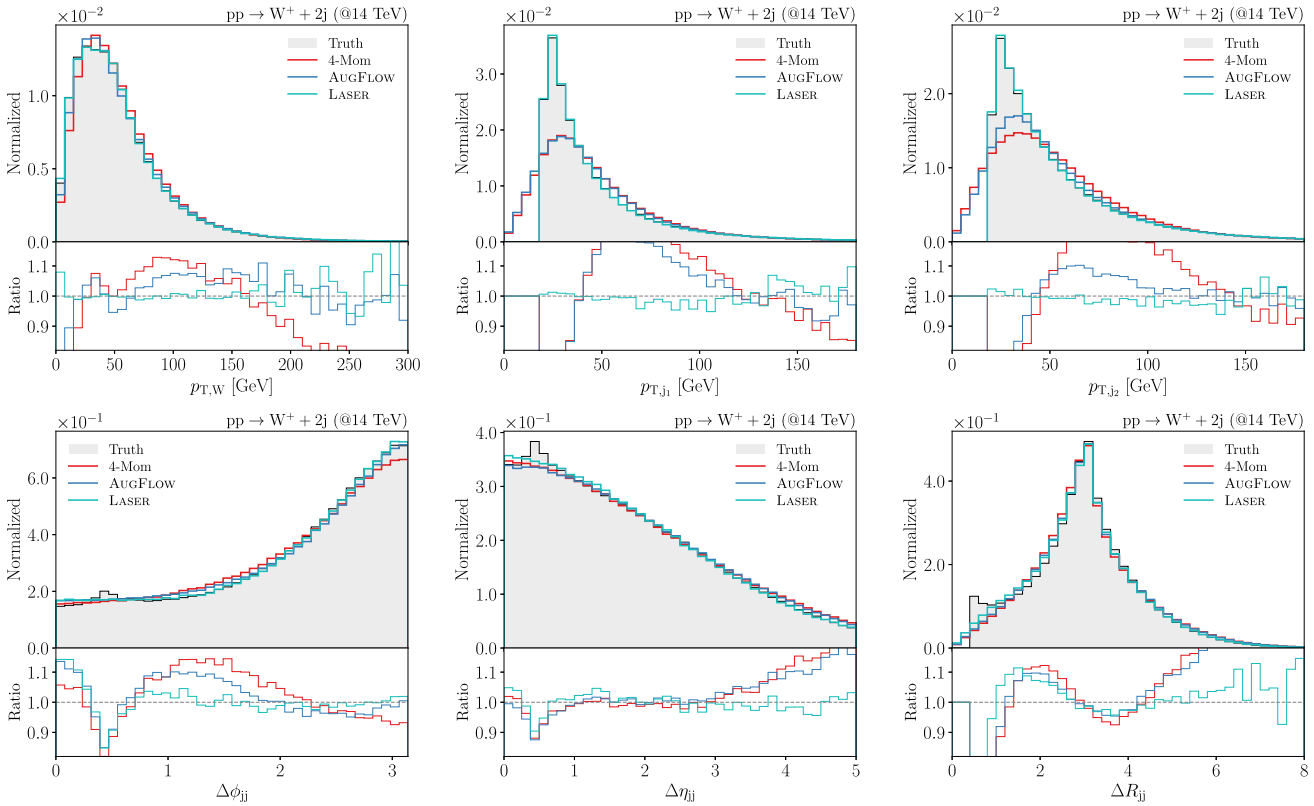

Fig. 4 Transverse momentum distributions for the W-boson and the two jets (top), and high-level angular distributions (bottom) for the pp → W^+ + 2j process. We show the baseline model as well as AugFlow and a Laser improved version. The baseline model uses the 4-vector parametrization introduced in Eq. (3.1)

However, as the relevant phase space is bigger than in the two-jet case, this allows us to study the scaling behavior of our surrogate model for increasing multiplicities and hence phase-space dimensions. This study is relevant as the expected precision of the surrogate model, which is trained on data following a given underlying distribution, solely scales with the dimension and is not affected by the intrinsic precision of the training data. In other words, given a fixed phase-space dimensionality, the surrogate model's performance will not degrade by replacing the LO training data with more accurate NLO or even NNLO simulation data. Hence, we only test our methods on easy-to-generate LO data for simplicity and without loss of generality. Finally, we expect the effect effective gain of the surrogate model to be more significant for more complicated processes without losing precision. This has been shown in Ref. [61] and can be explained as the surrogate model is generally much faster to evaluate than event weights derived from first principles. This speed advantage is becoming even bigger for NLO and NNLO scenarios where the evaluation time of the matrix elements becomes the limiting aspect.

Different parametrizations in the generator reloaded First, we study the impact of preprocessing in Fig. 6 (the three-jet analog of Fig. 5). We omit the 4-Mom parameterization and observe that the MinRep parameterization is already within 10% across the phase space and often better than or at least as good as PrecisESIAST and MAHAMBO. By design, PrecisESIAST is excellent for the jet transverse momenta but struggles with some of the ΔR observables. MAHAMBO is the worse for the second jet transverse momentum but the best at the ΔR and $\Delta\eta$ between the subleading two jets. Elsewhere, MinRep and MAHAMBO are comparable.

Classifier improved event generation The rest of the figures show how reweighting can improve the precision in both the data space and from the latent space.

Initially, the classifier was fed directly by the events generated in the MAHAMBO preprocessing, meaning the discriminator acts on a 10/dimensional hypercube. However, as shown in Fig. 7 by the red curve, the classifier is not learning any sensible weights and sharply peaks at $w(x) \approx 1$. This means the classifier cannot distinguish between the generated data and the truth using the MAHAMBO parametrization even though we tested a rather large network size to ensure that too few network parameters are not limiting the performance. Therefore, to better use the classifier, we change the data parametrization before feeding into the network to

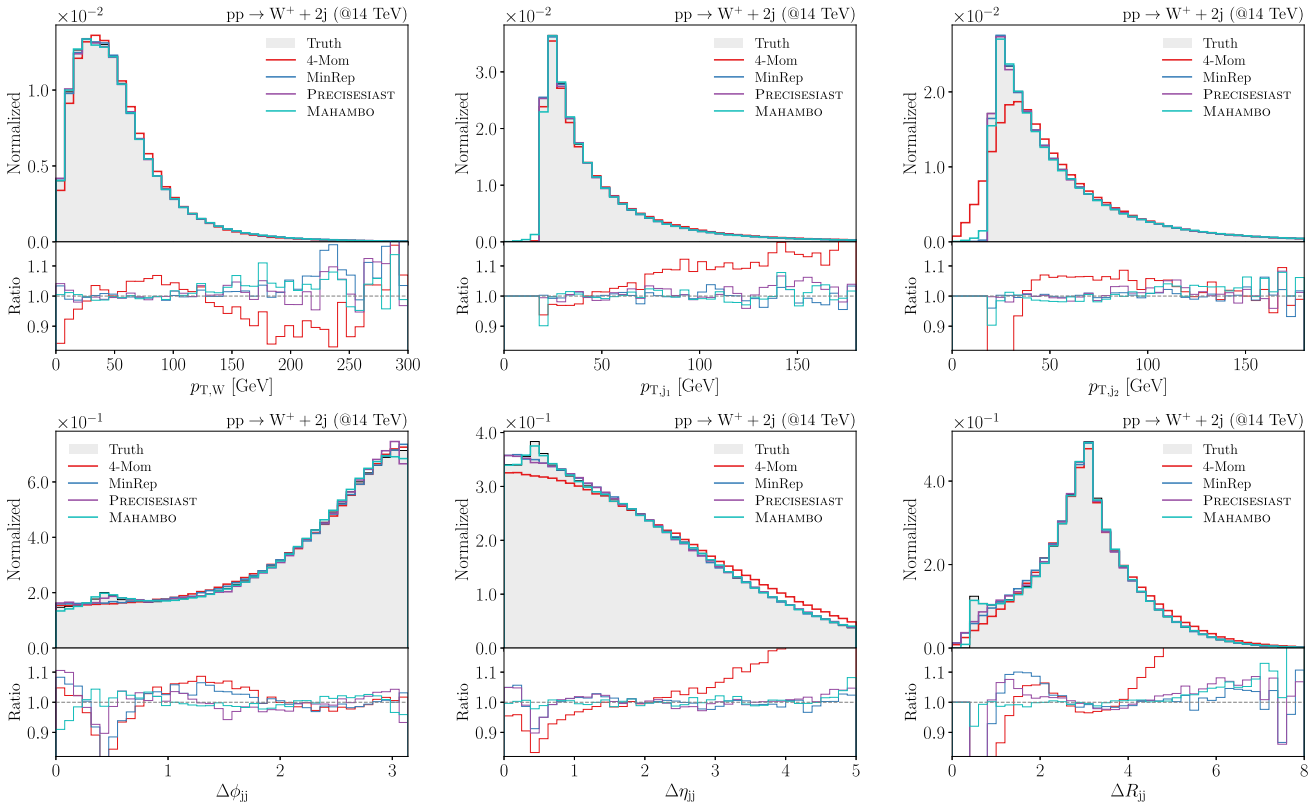

Fig. 5 Transverse momentum distributions (top) for the final state particles and high-level angular distributions (bottom) for the $pp \to W^+ + 2j$ process. We show the results for different parametrizations used for the generation

enhance the classification capabilities as much as possible. In particular, if the classifier inputs directly contain critical features such as $p_{T,j}$, ΔR, $\Delta\eta$ and $\Delta\phi$ the obtained weights provide much more information, as can be seen by the weight distributions for the PRECISESIAST and ELFS parametrization shown in Fig. 7.

First, Fig. 8 demonstrates the reweighting in the data space when generating events in the MAHAMBO parametrization and feeding the data into the classifier in the ELFS parametrization. In this case, the weights defined by the classifier are no longer proper functions of the latent space, which naively does not allow to employ the LASER protocol. Since the data space and latent space are not having the same dimensionalities, we have to use the OMNIFOLD protocol to pull back the weights to the latent space. However, it is as tricky for OMNIFOLD to pull back the weights onto the latent space as obtaining useful classifier weights when training the classifier in the MAHAMBO representation. Apparently, it is difficult to detect the non-trivial deformations to the Gaussian latent space required to alleviate the small deficiencies in the MAHAMBO representation.

Instead, we can use these weights to reweight the data space directly, which is the bases of the DCTR protocol. As was observed in many other studies that employ classifier-/based reweighting, including for ML/based simulation surrogate refinement [33,61], we find that the weighted data-/space is nearly indistinguishable from the truth and can capture sharp and subtle features.

However, the weighted samples in the data space have less statistical power due to a variance in the weights. Figure 9 explores how we can pull back the data-space weights to the latent space to produce unweighted examples in the data space. We start with ELFS instead of MAHAMBO as we found that pulling back MAHAMBO weights was ineffective as it requires OMNIFOLD. Instead, if we use ELFS in both the generator and discriminator, the obtained weights are already proper weights of the latent space and allow using the LASER protocol. Without any refinement, ELFS cannot capture many of the key features of the phase space. However, DCTR works well, and the improvements directly translate almost verbatim when pulled back to the latent space and integrated into LASER. This includes all sharp and subtle features, especially in the variables that describe angles between jets. Even though ELFS starts further from the truth, the refinement makes it closer than MAHAMBO.

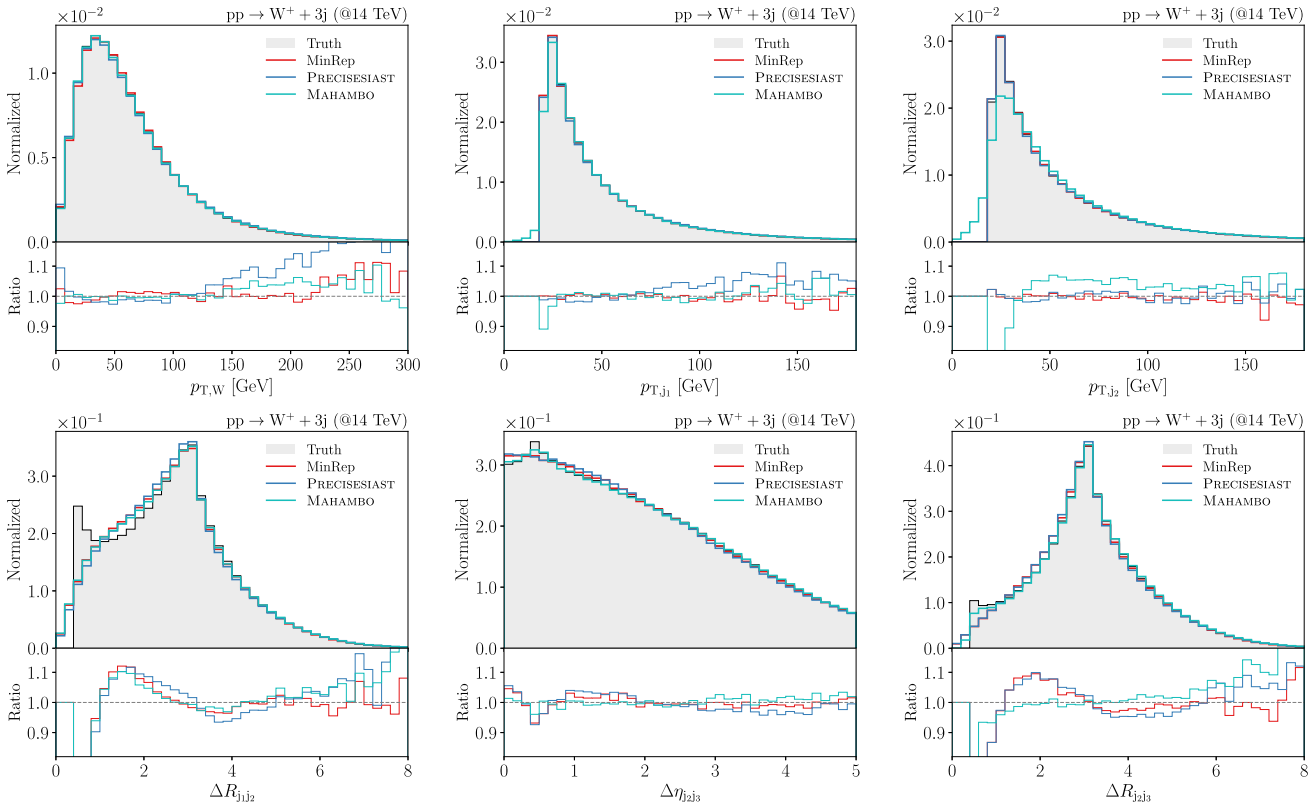

Fig. 6 Transverse momentum distributions for the W-boson and the two leading jets (top) and high-level angular distributions (bottom) for the $pp \to W^+ + 3j$ process. We show the results for different parametrizations used for the generation

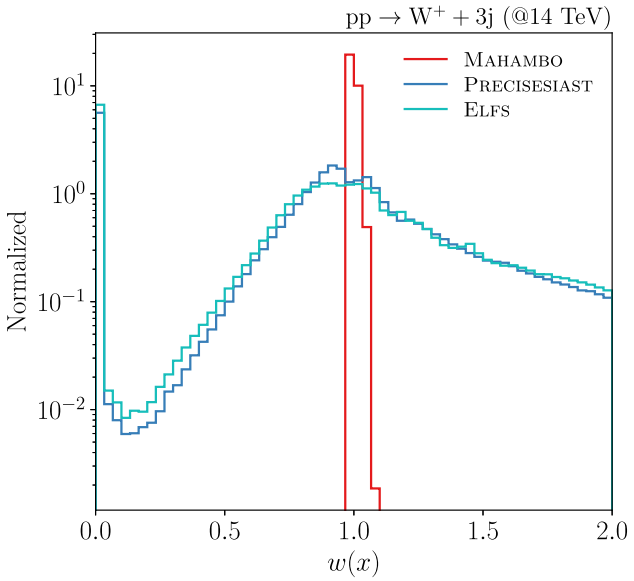

Fig. 7 Learned classifier weight $w(x)$ for different parametrizations applied to events generated in the MAHAMBO parametrization

5 Conclusions and outlook

Neural network surrogate models are entering an era of precision, where a qualitative description is replaced with quantitative agreement across the full phase space. Using W+jets matrix element simulations as an example, we explore how modifying the data before, during, and after generation can enhance the precision.

Unsurprisingly, preprocessing is found to have a significant impact on downstream precision. Therefore, we proposed MAHAMBO, a variation of RAMBOONDIET, to automatically reduce the data to its minimal/dimensional representation without needing process-specific interventions. This approach matched or outperformed other parameterizations across most of the phase space.

In agreement with previous studies, we find that classifier-based reweighting is able to precisely match the truth since the support of the probability density for our surrogate models is a superset of the truth.

We also study two latent space refinement methods to avoid weighted samples. The best approach for the W + jets example was LASER, where the data-space weights are used in the latent space to produce unweighted examples.

While our primary focus has been on matrix element generation, the presented ML approaches have the potential for

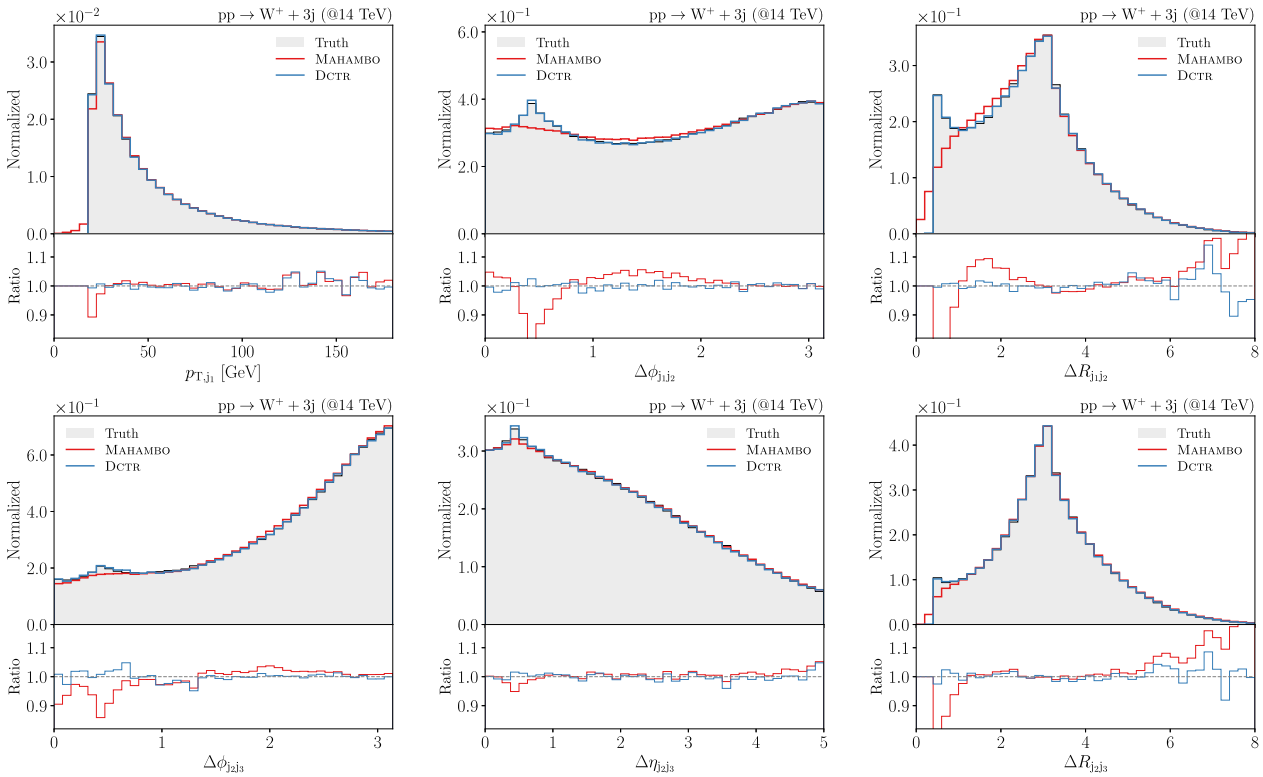

Fig. 8 Transverse momentum distributions for the W-boson and the two leading jets (top) and high-level angular distributions (bottom) for the $pp \to W^+ + 3j$ process. We show the results for pure MAHAMBO parametrization and including DCTR reweighting

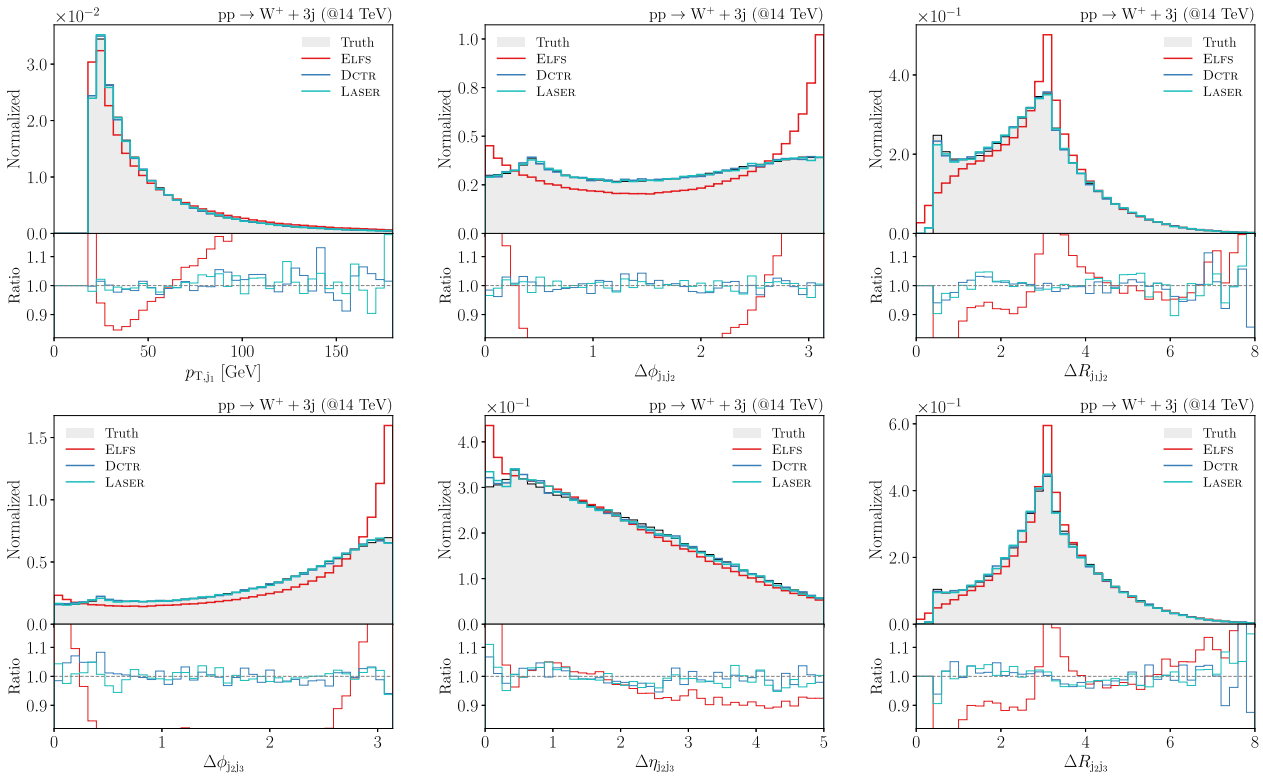

Fig. 9 Transverse momentum distributions for the W-boson and the two leading jets (top) and high-level angular distributions (bottom) for the $pp \to W^+ + 3j$ process. We show the results for pure ELFS parametrization and including DCTR and LASER refinement

broader applications in the study of surrogate models. Furthermore, these techniques can be adapted for physics-based simulators, particularly in cases where refining the latent space allows for accessing the random numbers within the program. Finally, improving surrogate models to align accurately with physics-based simulations or observational data holds immense potential for enhancing various downstream inference tasks. We eagerly anticipate further exploration of these connections in future endeavors.

Acknowledgements We want to thank Tilman Plehn for fruitful discussions about classifiers and Theo Heimel for his help on the PRECISESIAST parametrization. Further, we would like to thank Rob Verheyen for fruitful discussions about non-bijective flows and Marco Bellagente for his work earlier in this project. Finally, we thank Didrik Nielsen for his help on the SURVAE framework. BPN was supported by the Department of Energy, Office of Science under contract number DE-AC02-05CH11231. RW was supported by FRS-FNRS (Belgian National Scientific Research Fund) IISN projects 4.4503.16. In addition, computational resources have been provided by the supercomputing facilities of the Université catholique de Louvain (CISM/UCL) and the Consortium des Équipements de Calcul Intensif en Fédération Wallonie Bruxelles (CÉCI) funded by the Fond de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under convention 2.5020.11 and by the Walloon Region.

Data Availability Statement The code and data for this paper can be found at https://github.com/ramonpeter/elsa. This allows the reader to either fully reproduce all our results with ease or adopt and implement our methods within other software tools.

Funded by SCOAP[3]. SCOAP[3] supports the goals of the International Year of Basic Sciences for Sustainable Development.

References

1. ATLAS Collaboration, Technical design report for the phase-II upgrade of the ATLAS TDAQ system. Technical report. CERN, Geneva, Sept 2017
2. CMS Collaboration, The phase-2 upgrade of the CMS DAQ interim technical design report. Technical report. CERN, Geneva, Sept 2017
3. I.J. Goodfellow et al., Generative adversarial nets, in *Conference on Neural Information Processing Systems*, vol. 2 (2014), p. 2672
4. A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, A.A. Bharath, Generative adversarial networks: an overview. IEEE Signal Process. Mag. **35**, 53 (2018). https://doi.org/10.1109/msp.2017.2765202
5. D.P. Kingma, M. Welling, Auto-encoding variational bayes. arXiv:1312.6114
6. D.P. Kingma, M. Welling, An introduction to variational autoencoders. Found. Trends Mach. Learn. **12**, 307 (2019). https://doi.org/10.1561/2200000056
7. D.J. Rezende, S. Mohamed, Variational inference with normalizing flows, in *International Conference on Machine Learning*, vol. 37 (2015), p. 1530
8. I. Kobyzev, S. Prince, M. Brubaker, Normalizing flows: an introduction and review of current methods. IEEE Trans. Pattern Anal. Mach. Intell. (2020). https://doi.org/10.1109/tpami.2020.2992934
9. J. Sohl-Dickstein, E.A. Weiss, N. Maheswaranathan, S. Ganguli, Deep unsupervised learning using nonequilibrium thermodynamics. arXiv:1503.03585
10. J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models. arXiv:2006.11239
11. P. Dhariwal, A. Nichol, Diffusion models beat GANs on image synthesis. arXiv:2105.05233
12. A. Hyvärinen, Estimation of non-normalized statistical models by score matching. J. Mach. Learn. Res. **6**, 695–709 (2005)
13. Y. Song, S. Ermon, Generative modeling by estimating gradients of the data distribution. arXiv:1907.05600
14. Y. Song, S. Ermon, Improved techniques for training score-based generative models. arXiv:2006.09011
15. S. Badger et al., Machine learning and LHC event generation. SciPost Phys. **14**, 079 (2023). https://doi.org/10.21468/SciPostPhys.14.4.079. arXiv:2203.07460
16. A. Adelmann et al., New directions for surrogate models and differentiable programming for High Energy Physics detector simulation, in *Snowmass 2021* (2022). arXiv:2203.08806
17. A. Butter, T. Plehn, R. Winterhalder, How to GAN LHC events. SciPost Phys. **7**, 075 (2019). https://doi.org/10.21468/SciPostPhys.7.6.075. arXiv:1907.03764
18. Y. Alanazi et al., Simulation of electron–proton scattering events by a feature-augmented and transformed generative adversarial network (FAT-GAN). arXiv:2001.11103
19. Y. Alanazi et al., Machine learning-based event generator for electron–proton scattering. Phys. Rev. D **106**, 096002 (2022). https://doi.org/10.1103/PhysRevD.106.096002. arXiv:2008.03151
20. C. Bravo-Prieto, J. Baglio, M. Cè, A. Francis, D.M. Grabowska, S. Carrazza, Style-based quantum generative adversarial networks for Monte Carlo events. Quantum **6**, 777 (2022). https://doi.org/10.22331/q-2022-08-17-777. arXiv:2110.06933
21. A. Butter, T. Heimel, S. Hummerich, T. Krebs, T. Plehn, A. Rousselot et al., Generative networks for precision enthusiasts. SciPost Phys. **14**, 078 (2023). https://doi.org/10.21468/SciPostPhys.14.4.078. arXiv:2110.13632
22. B. Stienen, R. Verheyen, Phase space sampling and inference from weighted events with autoregressive flows. SciPost Phys. **10**, 038 (2021). https://doi.org/10.21468/SciPostPhys.10.2.038. arXiv:2011.13445
23. C. Krause, D. Shih, Fast and accurate simulations of calorimeter showers with normalizing flows, Phys. Rev. D **107**, 113003 (2023). arXiv:2106.05285
24. C. Krause, D. Shih, Accelerating accurate simulations of calorimeter showers with normalizing flows and probability density distillation, Phys. Rev. D **107**, 113004 (2023). arXiv:2110.11377
25. R. Winterhalder, V. Magerya, E. Villa, S.P. Jones, M. Kerner, A. Butter et al., Targeting multi-loop integrals with neural networks. SciPost Phys. **12**, 129 (2022). https://doi.org/10.21468/SciPostPhys.12.4.129. arXiv:2112.09145
26. R. Verheyen, Event generation and density estimation with surjective normalizing flows. SciPost Phys. **13**, 047 (2022). https://doi.org/10.21468/SciPostPhys.13.3.047. arXiv:2205.01697

27. T. Heimel, R. Winterhalder, A. Butter, J. Isaacson, C. Krause, F. Maltoni et al., *MadNIS – Neural Multi-Channel Importance Sampling*, arXiv:2212.06172
28. E. Dupont, A. Doucet, Y. W. Teh, *Augmented neural odes*, arXiv:1904.01681
29. H. Wu, J. Köhler, F. Noé, Stochastic normalizing flows. arXiv:2002.06707
30. R. Winterhalder, M. Bellagente, B. Nachman, Latent space refinement for deep generative models. arXiv:2106.00792
31. J. Batson, C.G. Haaf, Y. Kahn, D.A. Roberts, Topological obstructions to autoencoding. JHEP **04**, 280 (2021). https://doi.org/10.1007/JHEP04(2021)280. arXiv:2102.08380
32. J. Brehmer, K. Cranmer, Flows for simultaneous manifold learning and density estimation. arXiv:2003.13913
33. S. Diefenbacher, E. Eren, G. Kasieczka, A. Korol, B. Nachman, D. Shih, DCTRGAN: improving the precision of generative models with reweighting. J. Instrum. **15**, P11004 (2020). https://doi.org/10.1088/1748-0221/15/11/p11004. arXiv:2009.03796
34. T. Che, R. Zhang, J. Sohl-Dickstein, H. Larochelle, L. Paull, Y. Cao et al., Your gan is secretly an energy-based model and you should use discriminator driven latent sampling. arXiv:2003.06060
35. C.-W. Huang, L. Dinh, A. Courville, Augmented normalizing flows: bridging the gap between generative flows and latent variable models. arXiv:2002.07101
36. J. Chen, C. Lu, B. Chenli, J. Zhu, T. Tian, Vflow: more expressive generative flows with variational data augmentation. arXiv:2002.09741
37. D. Nielsen, P. Jaini, E. Hoogeboom, O. Winther, M. Welling, Survae flows: surjections to bridge the gap between vaes and flows. arXiv:2007.02731
38. CMS Collaboration, S. Chatrchyan et al., Jet production rates in association with W and Z bosons in pp collisions at $\sqrt{s} = 7$ TeV. JHEP **01**, 010 (2012). https://doi.org/10.1007/JHEP01(2012)010. arXiv:1110.3226
39. ATLAS Collaboration, G. Aad et al., Study of jets produced in association with a W boson in pp collisions at $\sqrt{s} = 7$ TeV with the ATLAS detector. Phys. Rev. D **85**, 092002 (2012). arXiv:1201.1276. https://doi.org/10.1103/PhysRevD.85.092002
40. S. Höche, F. Krauss, M. Schönherr, F. Siegert, W + n-jet predictions at the Large Hadron Collider at next-to-leading order matched with a parton shower. Phys. Rev. Lett. **110**, 052001 (2013). https://doi.org/10.1103/PhysRevLett.110.052001. arXiv:1201.5882
41. R. Boughezal, X. Liu, F. Petriello, W-boson plus jet differential distributions at NNLO in QCD. Phys. Rev. D **94**, 113009 (2016). https://doi.org/10.1103/PhysRevD.94.113009. arXiv:1602.06965
42. LHCb Collaboration, R. Aaij et al., Measurement of forward W and Z boson production in association with jets in proton–proton collisions at $\sqrt{s} = 8$ TeV. JHEP **05**, 131 (2016). https://doi.org/10.1007/JHEP05(2016)131. arXiv:1605.00951
43. CMS Collaboration, A.M. Sirunyan et al., Measurement of the differential cross sections for the associated production of a W boson and jets in proton–proton collisions at $\sqrt{s} = 13$ TeV. Phys. Rev. D **96**, 072005 (2017). https://doi.org/10.1103/PhysRevD.96.072005. arXiv:1707.05979
44. ATLAS Collaboration, M. Aaboud et al., Measurement of differential cross sections and W^+/W^- cross-section ratios for W boson production in association with jets at $\sqrt{s} = 8$ TeV with the ATLAS detector. JHEP **05**, 077 (2018). https://doi.org/10.1007/JHEP05(2018)077. arXiv:1711.03296
45. A. Gehrmann-De Ridder, T. Gehrmann, N. Glover, A.Y. Huss, D. Walker, NNLO QCD corrections to W+jet production in NNLOJET. PoS **LL2018**, 041 (2018). https://doi.org/10.22323/1.303.0041. arXiv:1807.09113
46. T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning*. *Springer Series in Statistics* (Springer New York Inc., New York, 2001)
47. M. Sugiyama, T. Suzuki, T. Kanamori, *Density Ratio Estimation in Machine Learning* (Cambridge University Press, Cambridge, 2012). https://doi.org/10.1017/CBO9781139035613
48. D. Martschei, M. Feindt, S. Honc, J. Wagner-Kuhr, Advanced event reweighting using multivariate analysis. J. Phys.: Conf. Ser. **368**, 012028 (2012). https://doi.org/10.1088/1742-6596/368/1/012028
49. K. Cranmer, J. Pavez, G. Louppe, Approximating likelihood ratios with calibrated discriminative classifiers. arXiv:1506.02169
50. A. Rogozhnikov, Reweighting with boosted decision trees. J. Phys. Conf. Ser. **762**, 012036 (2016). https://doi.org/10.1088/1742-6596/762/1/012036. arXiv:1608.05806
51. LHCb Collaboration, R. Aaij et al., Observation of the decays $\Lambda_b^0 \to \chi_{c1} p K^-$ and $\Lambda_b^0 \to \chi_{c2} p K^-$. Phys. Rev. Lett. **119**, 062001 (2017). https://doi.org/10.1103/PhysRevLett.119.062001. arXiv:1704.07900
52. ATLAS Collaboration, M. Aaboud et al., Search for pair production of higgsinos in final states with at least three b-tagged jets in $\sqrt{s} = 13$ TeV pp collisions using the ATLAS detector. Phys. Rev. D **98**, 092002 (2018). https://doi.org/10.1103/PhysRevD.98.092002. arXiv:1806.04030
53. J. Brehmer, K. Cranmer, G. Louppe, J. Pavez, Constraining effective field theories with machine learning. Phys. Rev. Lett. **121**, 111801 (2018). https://doi.org/10.1103/PhysRevLett.121.111801. arXiv:1805.00013
54. J. Brehmer, K. Cranmer, G. Louppe, J. Pavez, A guide to constraining effective field theories with machine learning. Phys. Rev. D **98**, 052004 (2018). https://doi.org/10.1103/PhysRevD.98.052004. arXiv:1805.00020
55. J. Brehmer, G. Louppe, J. Pavez, K. Cranmer, Mining gold from implicit models to improve likelihood-free inference. Proc. Natl. Acad. Sci. (2020). https://doi.org/10.1073/pnas.1915980117. arXiv:1805.12244
56. A. Andreassen, I. Feige, C. Frye, M.D. Schwartz, JUNIPR: a framework for unsupervised machine learning in particle physics. Eur. Phys. J. C **79**, 102 (2019). https://doi.org/10.1140/epjc/s10052-019-6607-9. arXiv:1804.09720
57. A. Andreassen, B. Nachman, Neural networks for full phase-space reweighting and parameter tuning. Phys. Rev. D **101**, 091901 (2020). https://doi.org/10.1103/PhysRevD.101.091901. arXiv:1907.08209
58. A. Andreassen, P.T. Komiske, E.M. Metodiev, B. Nachman, J. Thaler, OmniFold: a method to simultaneously unfold all observables. Phys. Rev. Lett. **124**, 182001 (2020). https://doi.org/10.1103/PhysRevLett.124.182001. arXiv:1911.09107
59. E. Bothmann, L. Del Debbio, Reweighting a parton shower using a neural network: the final-state case. J. High Energy Phys. **2019**, 33 (2019). https://doi.org/10.1007/JHEP01(2019)033
60. A. Andreassen, B. Nachman, D. Shih, Simulation assisted likelihood-free anomaly detection. Phys. Rev. D **101**, 095004 (2020). https://doi.org/10.1103/PhysRevD.101.095004. arXiv:2001.05001
61. K. Danziger, T. Janßen, S. Schumann, F. Siegert, Accelerating Monte Carlo event generation—rejection sampling using neural network event-weight estimates. SciPost Phys. **12**, 164 (2022). https://doi.org/10.21468/SciPostPhys.12.5.164. arXiv:2109.11964
62. S. Duane, A. Kennedy, B.J. Pendleton, D. Roweth, Hybrid Monte Carlo. Phys. Lett. B **195**, 216–222 (1987). https://doi.org/10.1016/0370-2693(87)91197-X
63. R.M. Neal, MCMC using Hamiltonian dynamics. arXiv:1206.1901
64. M. Backes, A. Butter, T. Plehn, R. Winterhalder, How to GAN event unweighting. SciPost Phys. **10**, 089 (2021). https://doi.org/10.21468/SciPostPhys.10.4.089. arXiv:2012.07873
65. M. Bellagente, A. Butter, G. Kasieczka, T. Plehn, A. Rousselot, R. Winterhalder et al., Invertible networks or partons to detector and

back again. SciPost Phys. **9**, 074 (2020). https://doi.org/10.21468/SciPostPhys.9.5.074. arXiv:2006.06685
66. R. Kleiss, W. Stirling, S. Ellis, A new Monte Carlo treatment of multiparticle phase space at high energies. Comput. Phys. Commun. **40**, 359–373 (1986). https://doi.org/10.1016/0010-4655(86)90119-0
67. S. Plätzer, RAMBO on diet. arXiv:1308.2922
68. R. Brent, *Algorithms for minimization without derivatives*, vol. 19 (Prentice Hall, Englewood Cliffs, 2002). https://doi.org/10.2307/2005713
69. D.P. Kingma, J. Ba, Adam: a method for stochastic optimization. arXiv:1412.6980
70. A.L. Maas, A.Y. Hannun, A.Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing* (2013)
71. F. Nielsen, R. Nock, On the chi square and higher-order chi distances for approximating f-divergences. IEEE Signal Process. Lett. **21**, 10–13 (2014). https://doi.org/10.1109/lsp.2013.2288355. arXiv:1309.3029
72. Y. Rubner, C. Tomasi, L.J. Guibas, The earth mover's distance as a metric for image retrieval. Int. J. Comput. Vis. **40**, 99–121 (2000). https://doi.org/10.1023/A:1026543900054
73. R. Flamary, N. Courty, A. Gramfort, M.Z. Alaya, A. Boisbunon, S. Chambon et al., Pot: Python optimal transport. J. Mach. Learn. Res. **22**, 1–8 (2021)
74. J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer et al., The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations. JHEP **07**, 079 (2014). arXiv:1405.03011/JHEP07(2014)079
75. NNPDF Collaboration, R.D. Ball et al., Parton distributions for the LHC Run II. JHEP **04**, 040 (2015). https://doi.org/10.1007/JHEP04(2015)040. arXiv:1410.8849
76. A. Buckley, J. Ferrando, S. Lloyd, K. Nordström, B. Page, M. Rüfenacht et al., LHAPDF6: parton density access in the LHC precision era. Eur. Phys. J. C **75**, 132 (2015). https://doi.org/10.1140/epjc/s10052-015-3318-8. arXiv:1412.7420
77. C. Durkan, A. Bekasov, I. Murray, G. Papamakarios, Neural spline flows. Adv. Neural Inf. Process. Syst. **32**, 7511–7522 (2019). arXiv:1906.04032