Score-based generative models for calorimeter shower simulation

Vinicius Mikuni[1,*] and Benjamin Nachman[2,3,†]

[1]*National Energy Research Scientific Computing Center, Berkeley Lab, Berkeley, California 94720, USA*
[2]*Physics Division, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA*
[3]*Berkeley Institute for Data Science, University of California, Berkeley, California 94720, USA*

(Received 6 July 2022; accepted 10 November 2022; published 28 November 2022)

Score-based generative models are a new class of generative algorithms that have been shown to produce realistic images even in high dimensional spaces, currently surpassing other state-of-the-art models for different benchmark categories and applications. In this work we introduce CaloScore, a score-based generative model for collider physics applied to calorimeter shower generation. Three different diffusion models are investigated using the Fast Calorimeter Simulation Challenge 2022 dataset. CaloScore is the first application of a score-based generative model in collider physics and is able to produce high-fidelity calorimeter images for all datasets, providing an alternative paradigm for calorimeter shower simulation.

DOI: 10.1103/PhysRevD.106.092009

I. INTRODUCTION

Detailed detector simulations are an essential component of data analysis in particle and nuclear physics. These simulations are used to fold particle-level predictions for comparisons with data and are used to unfold data for detector-effects in order to compare with predictions and other experiments. Simulations are also a critical input for the design of future experiments.

The most widely used detector simulations are built on the program GEANT [1–3]. Achieving precision requires significant computing time as propagating particles through materials results in a large number of secondary particles each undergoing electromagnetic and/or nuclear interactions. For this reason, the most complex detectors to emulate are calorimeters, whose purpose is to stop particles and measure the deposited energy. An $\mathcal{O}(1)$ fraction of all computing in HEP goes toward simulating particle propagation inside dense materials with GEANT.

The experiments at the Large Hadron Collider (LHC) generate billions of events per run, each of which has hundreds to thousands of individual calorimeter showers. Within the experiments' computing budgets, it is not possible to run GEANT-based ('full') simulation for all events. Therefore, all of the experiments have developed fast simulation methods that replace physics models with simpler parametric models that are tuned to the full simulation. The fast simulation models are constructed with relatively few parameters in order to facilitate efficient optimization and validation. This fundamentally limits their precision, in particular in the ability to model complex correlations in high-dimensions. These correlations may also not be explicitly part of an optimization that uses only a relatively small number of one-dimensional observables.

Deep learning offers a complementary approach to engineered parameteric models. Flexible neural networks are used to transform random numbers (called a *latent space* in machine learning) into structured data. So far, there have been three main strategies for deep generative modeling. Generative adversarial networks (GANs) [4] optimize the generator network by means of an auxiliary network ("discriminator") that tries to classify generated examples from real examples. Variational autoencoders (VAEs) [5] learn a stochastic map from the data space to a latent space and back, preserving the statistics of the latent space and data space. Normalizing flows (NFs) [6] use invertible transformations so that the probability density can be computed and the generator is optimized using the log likelihood. A number of deep generative models have been proposed for emulating calorimeter showers and other particle detectors [7–45].

The first proposals for generating calorimeter showers with deep generative models used GANs (starting with CaloGAN [7–9]). The evaluation of GANs is fast and there are no constraints on the form of the generator function. However, GAN optimization is challenging as it is a minimax problem due to the competition between the generator and discriminator networks. Furthermore, GANs are known to suffer from *mode collapse* where the generator learns to produce only a subset of the possible

*vmikuni@lbl.gov
†bpnachman@lbl.gov

showers. Despite these challenges, the ATLAS Collaboration becomes the first experiment to replace part of their fast simulation with a GAN [46] and are already using it to generate billions of events for Run 3. Other experiments are also exploring the use of GANs for detector simulation [12,18,20].

Since the first GAN studies, there have been a number of innovations to improve the precision of deep generative calorimeter simulation. This includes variations/modifications to the GAN setup including Wasserstein GANs [20,47] to help with training stability and mode collapse and refining networks [25,26] to correct the spectra of an initial generative model. Beyond GANs, recent work with NFs has shown great promise [44,45]. Normalizing flows tend to be robust to mode collapse and are minimized with a convex loss function. Furthermore, the normalization of the resulting generative model seems to have beneficial regularization properties for the training. The authors of the CaloFlow model [44,45] showed that a *post-hoc* classifier struggled to distinguish showers generated from their neural network and showers from GEANT 4, a critical milestone in this field.

While NFs show great promise for fast calorimeter simulation, they have difficulty scaling to higher-dimensional datasets. Such datasets are important for the upgraded CMS forward calorimeter [48] as well as ultrafine calorimeters proposed for future detectors [49]. Two of the three datasets in the recent *Fast Calorimeter Simulation Challenge 2022* [46,50–52] have dimensionality at least an order of magnitude beyond what has been studied with NFs. The NF training time for these data is prohibitive and would require significant research and development.

In this paper, we examine the potential of a new class of deep generative algorithms named score-based generative models [53–55]. Like NFs, score-based models minimize a convex loss function with a single generator network that also provides access to the full data likelihood after training (see Appendix A). However, unlike NFs, the gradient of the data density (the "score") is learned instead of the density. This choice introduces more flexibility to the network architecture, since the Jacobian of the transformation does not need to be computed during training. This additional flexibility, contrary to normalizing flows, also allows the introduction of bottleneck layers (layers with fewer neurons than the previous layer), greatly reducing the number of trainable parameters and improving the scalability of the model. We demonstrate this explicitly on the Fast Calorimeter Simulation Challenge 2022 datasets [56].

This paper is organized as follows. Section II introduces how deep generative models can be constructed using the likelihood gradients instead of likelihoods directly. Different choices of drift and diffusion functions investigated in this work are introduced in Sec. III. Section IV describes how new samples are generated from the trained model. Description of the Fast Calorimeter Simulation Challenge 2022 datasets and network architecture are presented in Secs. V and VI as well as a GAN model used for comparison. Finally, numerical results are discussed in Sec. VII. The paper ends with conclusions and outlook in Sec. VIII.

II. GENERATIVE MODELS USING GRADIENTS OF THE DATA

Our approach most closely follows Ref. [55]. Before providing the technical details, we briefly describe the key method components. First, a neural network is learned to approximate the data score, $\nabla_x \log p_{\text{data}}$ for some high-dimensional distribution $x \in \mathbb{R}^D$ described by the probability density p_{data}. In many high energy physics examples, including calorimeter simulation, we do not have access to p_{data} analytically—we can only sample from it. Without access to p_{data}, we cannot use a loss function like the mean squared error to directly approximate ("match") the score. NFs circumvent this problem by maximizing the likelihood of the data. There is no analog of maximum likelihood for score matching and so a different innovation is required.

For data that are smeared, it can be shown that matching the score of the smeared data is equivalent to matching the score of the smearing function [57]. For data purposefully smeared by an analytically tractable smearing function (e.g., a Gaussian), this means that all of components required to compute the loss function are known. The methods explored in this paper make use of purposeful smearing, where the amount of smearing is increased/decreased to estimate the density or generate samples, respectively. A schematic of the idea is shown in Fig. 1.

A stochastic process that continuously corrupts data is described by the following stochastic differential equation (SDE):

$$\mathrm{d}x = f(x,t)\mathrm{d}t + g(t)\mathrm{dw}. \tag{1}$$

The initial data $x(t=0) \coloneqq x_0 \in \mathbb{R}^d$ sampled from the distribution p_{data} evolve over time given a set of drift and diffusion coefficients $f(x,t):\mathbb{R}^d \to \mathbb{R}^d$ and $g(t):\mathbb{R} \to \mathbb{R}$, respectively. The Wiener process, or Brownian motion, $w(t):\mathbb{R} \to \mathbb{R}$ is indexed by the time parameter $t \in [0,1]$. The goal of the generative model is to reverse this process, generating new observations starting from a noise distribution and solving the reverse SDE defined as:

$$\mathrm{d}x = [f(x,t) - g(t)^2 \nabla_x \log p_t(x)]\mathrm{d}t + g(t)\mathrm{d}\bar{w}, \tag{2}$$

where $\bar{w}$ is the Wiener process in the reverse time direction [58] with the gradient $\nabla_x \log p_t(x)$ of the time-dependent density $p_t(x)$ named the *data score*. For fixed choices of the drift and diffusion coefficients, the only term in Eq. (2) that needs to be estimated is the time-dependent score function. However, since the score-function is not tractable, we use neural networks that aim to minimize the Frobenius norm of the difference

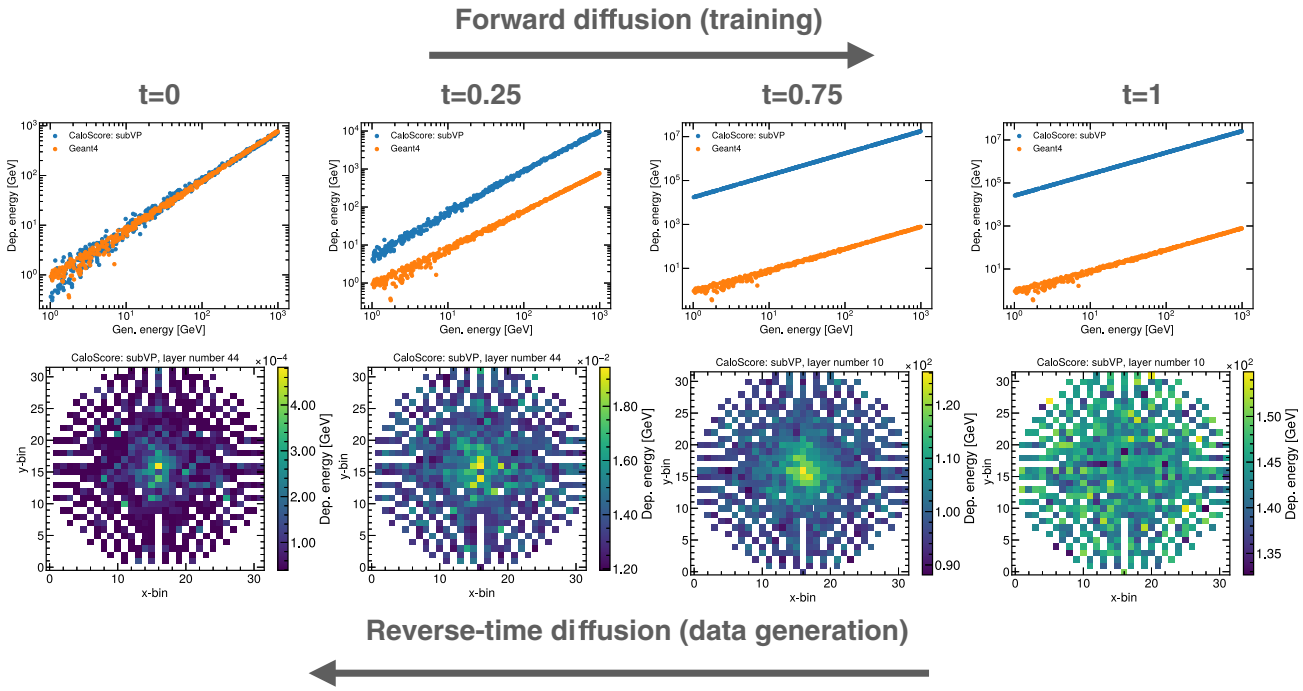

FIG. 1. The score-based generative model is trained using a diffusion process that slowly perturbs the data. Generation of new samples is carried out by reversing the diffusion process using the learned score-function, or the gradient of the data density. For different time steps, we show the distribution of deposited energies versus generated particle energies (top) and the energy deposition in a single layer of a calorimeter (bottom), generated with our proposed CaloScore model.

$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(x)}[\|s_\theta(x) - \nabla_x \log p_{\text{data}}\|_2^2], \tag{3}$$

with $s_\theta(x)$ the output of a network with trainable parameters θ. Since the true score function is not known, Eq. (3) cannot be readily computed. Instead, a denoising score matching [57] strategy is used. In this strategy, instead of learning the score function of the data, we aim to learn the score function of data that have been perturbed with a known perturbation function since it is sufficient to match the score of the perturbation function. Note that while matching the score of the smeared data requires computing an expectation value over the smeared data, matching the score of the smearing function requires computing the smeared data and an expectation value over the smeared data.

Given a Gaussian perturbation kernel $p_\sigma(\tilde{x}|x) \coloneqq \mathcal{N}(x, \sigma^2)$ and $p_\sigma(\tilde{x}) \coloneqq \int p_{\text{data}}(x) p_\sigma(\tilde{x}|x)\mathrm{d}x$, the probability density of the perturbed data, the loss function minimized during training is

$$\frac{1}{2}\mathbb{E}_{p_\sigma(\tilde{x}|x)p_{\text{data}}}[\|s_\theta(\tilde{x}) - \nabla_{\tilde{x}} \log p_\sigma(\tilde{x}|x)\|_2^2]. \tag{4}$$

The advantage of this strategy is that we can directly estimate the last term in Eq. (4), since:

$$\nabla_{\tilde{x}} \log p_\sigma(\tilde{x}|x) = \frac{x - \tilde{x}}{\sigma^2} \sim \frac{\mathcal{N}(0,1)}{\sigma} \tag{5}$$

The time component can be made explicit by rewriting the loss function in Eq. (4) as:

$$\frac{1}{2}\mathbb{E}_t\mathbb{E}_{p(x_t|x_0)p(x_0)}[\lambda(t)\|s_\theta(x,t) - \nabla_{x_t} \log p_t(x_t|x_0)\|_2^2]. \tag{6}$$

The weighting function $\lambda(t)\colon \mathbb{R} \to \mathbb{R}$ ensures the loss function has the same order of magnitude at all times and is chosen to be inversely proportional to $\mathbb{E}[\|\nabla_{x_t} \log p_t(x_t|x_0)\|_2^2]$. When the drift coefficient $f(x,t)$ is chosen to be an affine function of x, the resulting perturbation kernel is always Gaussian [59] and can be chosen such that both mean and variance are known in closed form, making Eq. (6) efficient to compute during training.

III. CHOICE OF DRIFT AND DIFFUSION COEFFICIENTS

In this work we investigate three different choices of drift and diffusion coefficients that result in perturbation kernels that are easy to calculate in closed form. The first SDE, initially proposed in [53], is defined as:

$$\mathrm{d}x = \sqrt{\frac{\mathrm{d}[\sigma^2(t)]}{\mathrm{d}t}}\mathrm{d}w. \tag{7}$$

TABLE I. Perturbation kernel induced by different SDE choices.

SDE	Perturbation kernel
VE	$\mathcal{N}(x(0), \sigma^2(t) - \sigma^2(0))$
VP	$\mathcal{N}(x(0)e^{-\frac{1}{2}\int_0^t \beta(s)\mathrm{d}s}, 1 - e^{-\int_0^t \beta(s)\mathrm{d}s})$
subVP	$\mathcal{N}(x(0)e^{-\frac{1}{2}\int_0^t \beta(s)\mathrm{d}s}, (1 - e^{-\int_0^t \beta(s)\mathrm{d}s})^2)$

The parameter $\sigma(t) = \sigma_{\min}(\frac{\sigma_{\max}}{\sigma_{\min}})^t$ is defined with $\sigma_{\min} = 0.01$ and $\sigma_{\max} = 50$ to ensure $x(1) \sim \mathcal{N}(0, \sigma^2_{\mathrm{MAX}})$ is independent from $x(0)$. Since the time-dependent variance of the resulting perturbation explodes when $t \to \infty$, this SDE is often referred to variance exploding (VE) SDE.

The second SDE is a continuous version of the discrete perturbation introduced in [54], defined as:

$$\mathrm{d}x = -\frac{1}{2}\beta(t)x\mathrm{d}t + \sqrt{\beta(t)}\mathrm{dw}. \tag{8}$$

The parameter $\beta(t) = \beta_{\min} + t(\beta_{\max} - \beta_{\min})$ with $\beta_{\min} = 0.1$ and $\beta_{\max} = 20$ is used, resulting in $x(1) \sim \mathcal{N}(0, 1)$. The variance of this process is fixed to one when the initial distribution also has unit variance, hence its is referred to as variance preserving (VP) SDE.

The last SDE, introduced in [55], is defined as:

$$\mathrm{d}x = -\frac{1}{2}\beta(t)x\mathrm{d}t + \sqrt{\beta(t)(1 - e^{-2\int_0^t \beta(s)\mathrm{d}s})}\mathrm{dw}. \tag{9}$$

When the parameter $\beta(t)$ is chosen to be identical as the one used in Eq. (8), the variance of the stochastic process is always smaller than the variance of the VP SDE, hence receiving the name subVP. Similarly to Eq. (8), $x(1)$ also follows a normal distribution.

The resulting perturbation kernels induced by each SDE are listed in Table I.

IV. SAMPLE GENERATION

New samples are generated by solving the reverse diffusion process defined in Eq. (2) using a numerical SDE solver. In this work, we use the Euler-Maruyama algorithm [60] followed by an additional corrector step that uses the Langevin MCMC approach [61,62] to increase the sampling quality. For each time decrement Δt, the updated state of the system is described as:

$$x_{t-\Delta t} = x_t + [f(x_t, t) - g^2(t)s_\theta(x_t, t)]\Delta t + g(t)\sqrt{|\Delta t|}z, \tag{10}$$

where $z \sim \mathcal{N}(0, 1)$ is sampled at each time step. The corrector step takes the updated state from Eq. (10) and applies the correction

$$x'_{t-\Delta t} = x_{t-\Delta t} + \epsilon s_\theta(x_{t-\Delta t}, t)g(t) + \sqrt{2\epsilon}z, \tag{11}$$

where ϵ is a tunable parameter that determines the strength of the correction applied. A dimension-independent expression for ϵ is defined as a function of a signal-to-noise ratio r parameter as:

$$\epsilon = 2r^2\frac{\|z\|_2^2}{\|s_\theta\|_2^2}, \tag{12}$$

where $\|z\|_2$ and $\|s_\theta\|_2$ are the batch-average norms of the Gaussian noise and trained score function.

High fidelity samples require the time interval Δt to be small, possibly leading to hundreds of iterative steps and consequently hundreds of network evaluations. We decrease the number of function evaluations by reusing the score function evaluated in Eq. (10) during the calculation of the corrector step in Eq. (11). This approach effectively decreases the number of function evaluations by a factor two while no decrease in generation quality is observed.

V. FAST CALORIMETER SIMULATION CHALLENGE 2022

CaloScore is trained on the datasets created by the Fast Calorimeter Simulation Challenge 2022 [46,50–52]. A total of three datasets are provided, representing calorimeter shower simulations with GEANT of different geometries and granularities. Dataset 1 [51] is based on the ATLAS open dataset [46,63] and is similar to the current ATLAS detector calorimeter geometry. Showers are generated at the calorimeter surface in the pseudorapidity range $\eta \in [0.20, 0.25]$. While samples consisting of both photons and pions are provided, we evaluate our model using only the photon dataset. The voxelization procedure is defined such that it minimizes the amount of empty voxels, while maintaining high fidelity compared to the full simulation. This strategy results in different number of voxels per calorimeter layer and a total of 368 voxels to represent the full detector slice. Photon energies are provided in this configuration for 15 incident energies from 256 MeV up to 4 TeV in steps given by powers of two. For each generated energy, 10k samples are provided with this number reduced at higher energies due to long simulation times, resulting in a total of 121k used during training.

Datasets 2 [52] and 3 [50] contain each 100k examples and are simulated using a common detector layout but with different voxelization granularity. The detector simulated has a concentric cylinder geometry with 45 layers, where each layer consists of active (silicon) and passive (tungsten) material, simulated with GEANT 4. Electrons are generated at the detector surface with initial energy sampled from a log-uniform distribution ranging from 1 GeV to 1 TeV. In dataset 2, each layer consists of 144 readout cells, with 9 in

the radial and 16 in the angular directions. Dataset 3 is more granular, consisting of 900 readout cells in each layer, with 18 in the radial and 50 in the angular directions.

Even though the initial voxelization is provided in Cylindrical coordinates, we found it beneficial to convert the voxels in datasets 2 and 3 to Cartesian coordinates. This change allows the effective usage of 3D convolutional neural networks (CNNs) to build the score model. While CNNs are applicable in polar coordinates, they struggle to learn periodic boundary conditions. Convolutional operations are also less effective, since the majority of the energy depositions are located near $r = 0$, or near the corners of the image. Since the datasets are only available after voxelization, the transformation from cylindrical to Cartesian coordinates inevitably leads to loss of information.[1] Nevertheless, we observe improved generation quality after changing the coordinate system. See Appendix B for more details of the transformation. The total number of voxels after the transformation is chosen to be similar to the original number, resulting in $12 \times 12 = 144$ and $32 \times 32 = 1024$ voxels per layer for datasets 2 and 3, respectively.

Energies deposited in each voxel span multiple orders of magnitude, motivating yet another transformation of the inputs before training the generative model. First, each voxel energy E_v is normalized by the value of the generated energy of the particle E_0 times a factor f that ensures the normalized voxel energy $E'_v = \frac{E_v}{fE_0}$ lies between 0 and 1. Naively, the factor f could be taken as 1, since energy conservation should ensure the sum of deposited energies to not exceed the initial particle's energy. However, the sampling fraction of the calorimeter may lead to a mismatch between these numbers. This effect is particularly important in dataset 1, when particle energies as low as 256 MeV are considered. We take f as the highest deviation between total deposited and generated energies, fixed to $f = 3.1$ for dataset 1 and $f = 2$ for datasets 2 and 3.

The normalized energy depositions are then transformed to log-space, similarly to the strategy used in CaloFLOW. The log-transformed value u_v is defined as:

$$u_v = \log\left(\frac{x}{1-x}\right), \qquad x = \alpha + (1-2\alpha)E'_v. \quad (13)$$

The value α is set to 10^{-6} and avoids a discontinuity when $E'_v = 0$. The generated particle energy, used as a conditional input to the model, is also transformed before training. The transformed conditional energy u_0 is defined as:

$$u_0 = \frac{e_0 - e_{\min}}{e_{\max} - e_{\min}}, \quad (14)$$

where $e_{\min}$ and $e_{\max}$ are the minimum and maximum energies available in the dataset.

Finally, dataset 1 is also modified by adding an extra dimension that encodes an overall energy normalization. Before applying the log-transformation in Eq. (13), the total deposited energy is calculated. Instead of normalizing each voxel by the generated energy, the total deposited energy is used, ensuring the sum of all voxels is equal to 1. The additional entry is then defined as the total deposited energy normalized by the initial particle energy times the factor f. This strategy improves the estimation of the total energy deposition during training, now encoded as a single entry rather than the sum of all voxels.

VI. MODEL ARCHITECTURE AND TRAINING DETAILS

The score function is built from a modified version of the U-net model [64] where and encoder-decoder architecture with skip connections is used. 3D convolution operations are used as the basic layers in CaloScore for datasets 2 and 3, leveraging the regular geometry. Dataset 1, on the other hand, is irregular and consists of different number of voxels per detector layer. In this case the score function is built based on 1D convolutional operations. This approach leads to dataset 1 requiring bigger kernel and layer sizes compared to datasets 2 and 3, resulting in a bigger model architecture.

Each convolutional operation uses the swish [65] nonlinear activation function, with a kernel size of 5 (dataset 1) and 3 (datasets 2 and 3). The number of dimensions is reduced in the encoder section of the network through average pooling operations, reducing the total number of dimensions by a factor 4 for dataset 1 and a factor $3 \times 2 \times 2 = 12$ for datasets 2 and 3 after each pooling layer. In the opposite direction, upsampling layers are used to increase the dimensionality by repeating entries multiple times.

The different network architectures used for each dataset are shown in Fig. 2.

Conditional inputs used to train the model, namely the time component and generated energy, are first transformed using random Fourier features [66]. The transformed features for each conditional input are then concatenated and passed over 2 fully connected layers of sizes 256 (dataset 1) or 128 (datasets 2 and 3), both followed by a swish activation function. This set of *conditional embeddings* are used during multiple stages of the model architecture. In particular, conditional convolutional operations are created by adding the conditional embeddings as an additional bias at the output of convolutional layers.

Additionally, we train a Wasserstein GAN (WGAN) [67] using an additional gradient penalty (GP) term [68] to

[1] A one-to-one assignment between the two sets of coordinates is possible, but requires the distance interval in Cartesian coordinates to follow a nonlinear function since the transformation of coordinates is itself nonlinear.

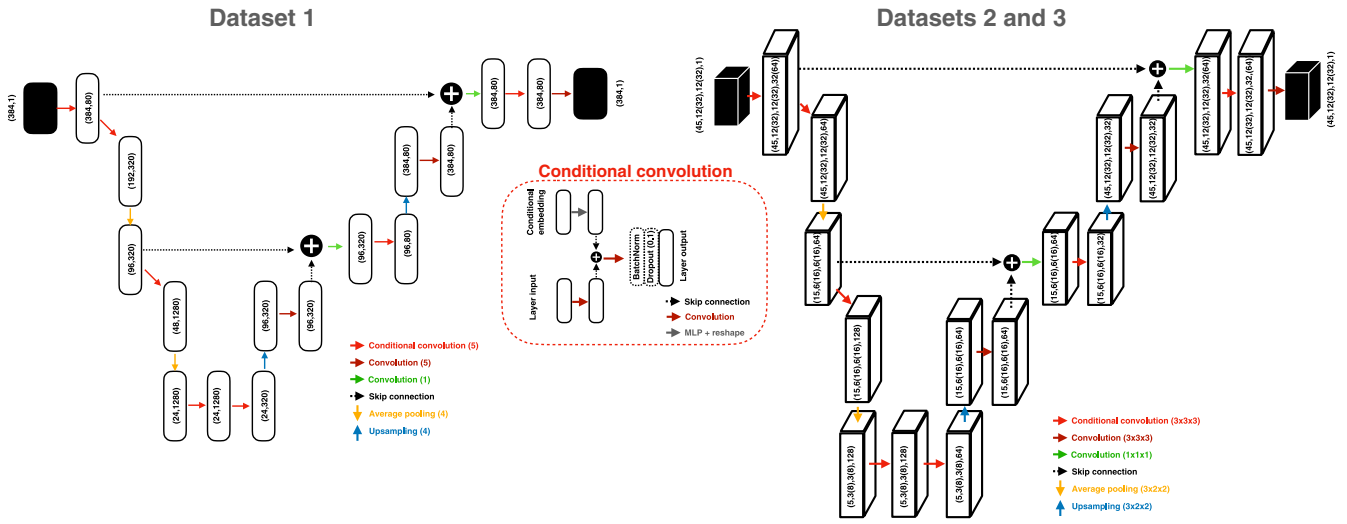

FIG. 2. Network architectures used for datasets 1 (left), 2 and 3 (right) based on the U-Net architecture. Values in parentheses represent the dimensionality of the data at each layer of the network. Parenthesis after convolutional operations represent the kernel size used. The swish nonlinear activation function is used after each convolution operation. Conditional inputs (time and energy) are used multiple times in the model to define conditional convolution operations described in the middle.

enforce the Lipschitz constraint of the critic model. WGAN-GP is a well-established model for calorimeter detector simulation and currently used in the ATLAS Collaboration [46] and studied in the context of different detector geometries applied to different experimental collaborations [12,20,69].

A similar network architecture to CaloScore is used to facilitate the comparison between approaches. The generator network takes as inputs an uniform distribution with dimensions 50, 100, 150 for datasets 1, 2, and 3 respectively. The noise vector is then concatenated with the conditional energy before passing through a fully connected layer with swish activation function. The output size of the fully connected layer is set to be the same size as last bottleneck layer of the U-Net model used in CaloScore. After reshaping, the same architecture that follows the last bottleneck layer in the right-hand side of the U-net is used. Similarly, the critic network takes as inputs real or generated samples and uses the same convolutional layers as the ones used in left-hand side of the U-Net model before the last bottleneck layer. From the last bottleneck layer, an additional fully connected layer is used with output size of 1 and no activation function. This choice results in all CaloScore models and WGAN-GP with similar number of trainable weights.

The implementation of all models is carried out with TensorFlow [70] optimized with ADAM [71] in the case of CaloScore and RMSProp in the WGAN-GP implementation, all with initial learning rate set to 10^{-4}. In CaloScore, the learning rate is reduced by a factor 2 if the loss function does not improve for a period of 100 consecutive epochs, evaluated using an independent dataset. The evaluation dataset is taken as 20% of the total amount of available training events. The models are trained for a total of 2000 epochs. The epoch with lowest evaluation loss is saved for further inspection. For all models, 16 NVIDIA A100 GPUs are used simultaneously interfaced with the HOROVOD package [72] on the Perlmutter supercomputer. The batch size in each GPU is set to 128 (datasets 1 and 2) and 64 (dataset 3). The WGAN-GP model is trained for 10000 epochs with fixed learning rate. The last training epoch is saved for further evaluation, but others epochs before the last were checked to produce similar results as the ones presented.

Sample generation is performed in CaloScore as described in Sec. IV, with signal-to-noise ratio fixed to 0.2 and total number of time steps set to 100, in dataset 1, and 200, in datasets 2 and 3. See Appendix C for differences in generation quality for other parameter choices. The total number of trainable weights and the time required to generate 100 calorimeter showers in a single GPU with batch size fixed to 100 are listed in Table II.

The lack of a regular geometry resulted in CaloScore requiring almost 20 times more trainable parameters for dataset 1 compared to datasets 2. On the other hand, dataset 3 has only ∼20% more trainable weights than dataset 2, even though dataset 3 has 7 times more voxels. This observation suggests that the model complexity is mostly determined by the network architecture rather than the number of voxels present in the dataset, contrary to normalizing flows where the complexity from the Jacobian determinant calculation increases at least linearly with the number of voxels.

Although the ultimate goal is for the generation time to be significantly faster than for GEANT, the time to generate new calorimeter showers with CaloScore is comparatively

TABLE II. Number of dimensions, trainable parameter, and time to generate 100 new calorimeter showers for each dataset studied in this work. Generation times for GEANT are based on the average time required to generate samples over the energy range provided.

			Time to 100 showers [s]		
Dataset	N. of voxels	N. of weights	CaloScore	WGAN-GP	GEANT
Dataset 1	384	32M	4.0	1.3	$\mathcal{O}(10^2$–$10^3)$
Dataset 2	6480	1.4M	5.8	1.33	$\mathcal{O}(10^4)$
Dataset 3	46080	1.7M	33.4	2.06	$\mathcal{O}(10^4)$

slow to other machine learning-based models as a result of the hundreds of function evaluations. In this paper, we have focused on modeling the complex data distributions and we leave explorations of accelerating inference to future studies.

VII. RESULTS

Multiple distributions are used to evaluate the quality of generated samples using different CaloScore SDE implementations and additional WGAN-GP model. The total energy deposited in the detector and the number of calorimeter hits are shown in Fig. 3. A hit is defined as any voxel with energy deposition above a certain energy threshold. The energy thresholds are taken as the minimum energy observed in each challenge dataset, set to 0.01 keV for dataset 1 and 15.1 keV for datasets 2 and 3. The 1-Wasserstein distance between distributions, referred as the Earth mover's distance (EMD), is also calculated between the GEANT and different generative model implementations.

A good agreement between the GEANT and generated samples is observed for all diffusion models and datasets. At low deposited energies, the difference between CaloScore and GEANT increase and is most noticeable for dataset 3. The subVP implementation shows a better agreement overall, followed by VP and VE, indicating that bounding the variance of the diffusion process is beneficial, specially as the number of voxels increase. Conversely, the

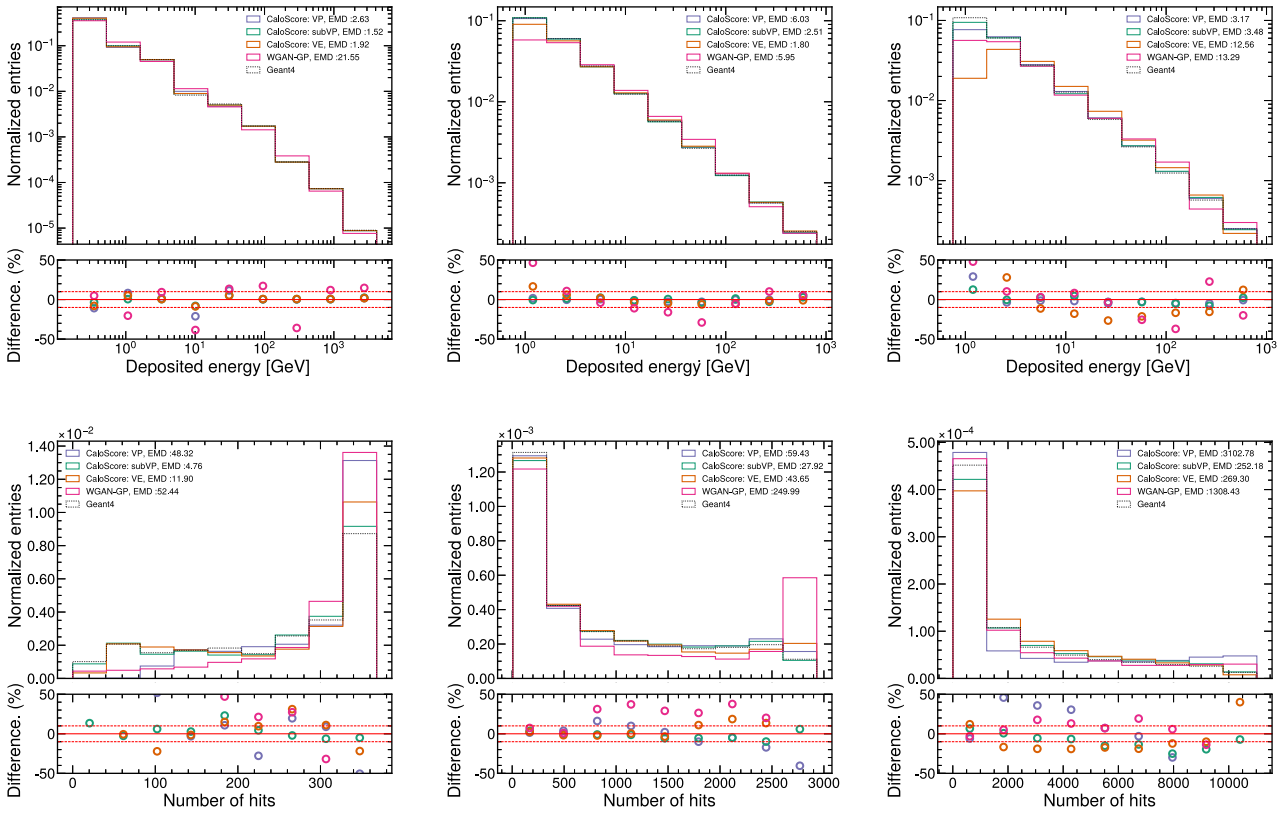

FIG. 3. Comparison of the sum of all voxel energies (top) and number of hits (bottom) for datasets 1 (left), 2 (middle), and 3 (right). Dashed red bands represent the 10% deviation interval of the generated samples when compared to GEANT predictions. The Earth mover's distance (EMD) between each distribution and the GEANT distribution is also provided.

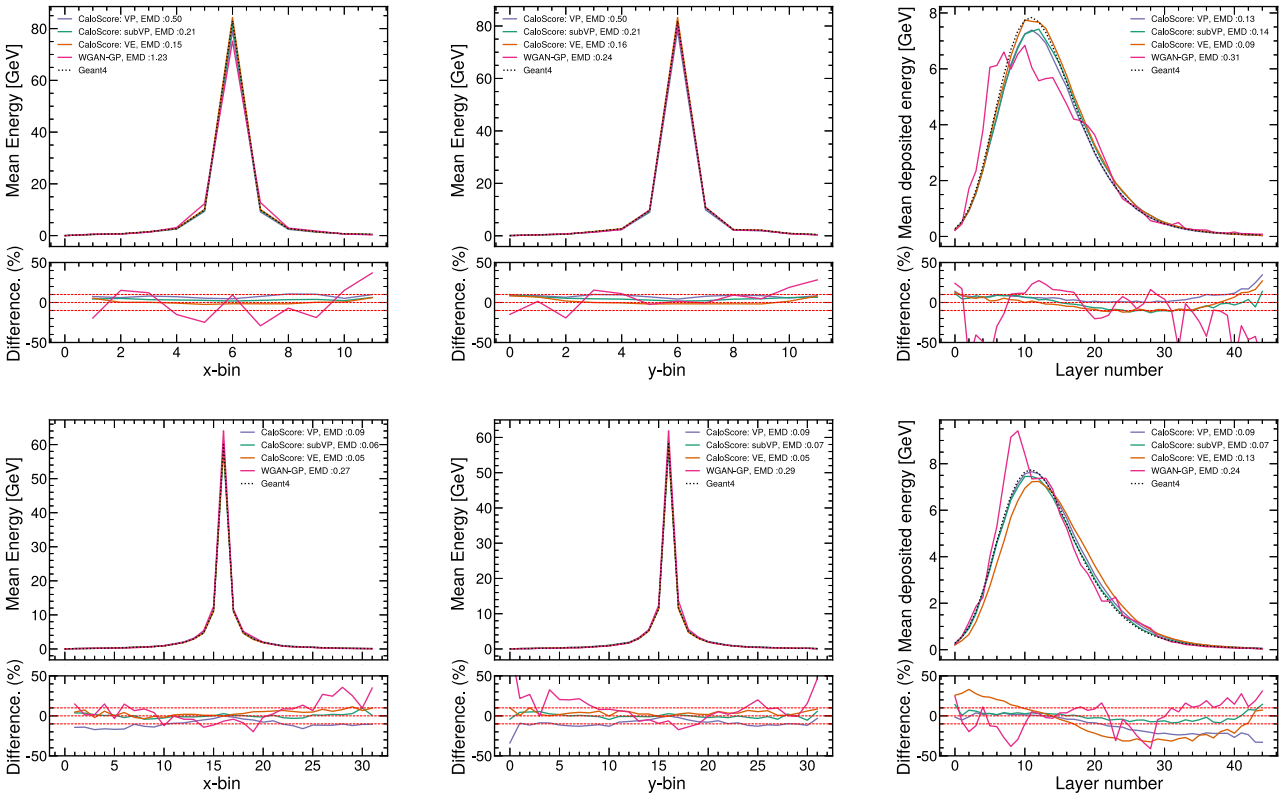

FIG. 4. Comparison of the average deposited energies in the x- (left), y- (middle), and z-coordinates (right) for datasets 2 (top) and 3 (bottom). Dashed red bands represent the 10% deviation interval of the generated samples when compared to GEANT predictions. The Earth mover's distance (EMD) between each distribution and the GEANT distribution is also provided.

WGAN-GP implementation shows a higher EMD value compared to CaloScore.

A similar conclusion is derived from the average energy deposition as a function of the layer number and Cartesian coordinates x and y shown in Fig. 4.

While the VE implementation agrees with the simulation response in dataset 2, the prediction for dataset 3 is shifted as seen from the layer-dependent distribution. In the case of the WGAN-GP, the predictions are also shifted, but in the opposite direction, predicting a higher energy fraction at the initial layers of the detector.

Similarly, the maximum energy deposited in a single voxel normalized to the total deposited energy is shifted in the VE implementation for dataset 3 as shown in Fig. 5. While the low energy fraction region for dataset 1 is well described by all CaloScore implementations, the high energy

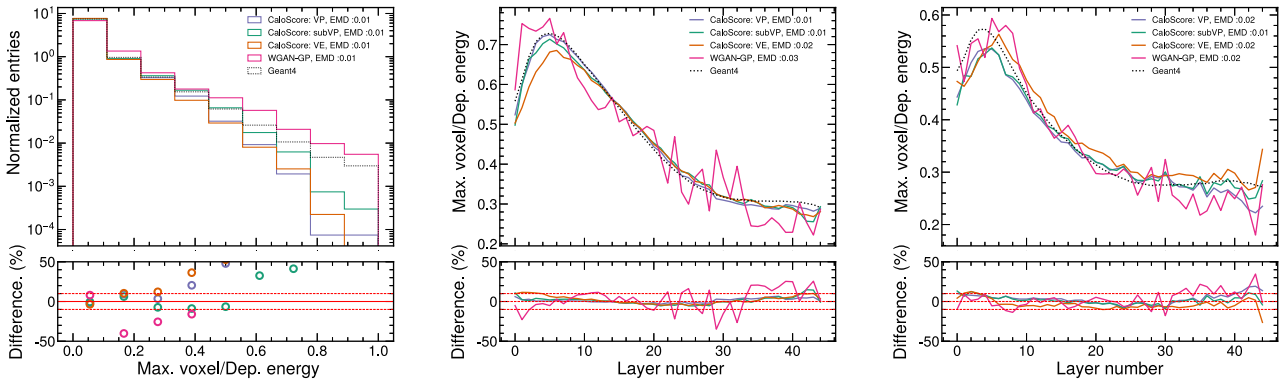

FIG. 5. Comparison of the maximum deposited energy in a single voxel divided by the sum of deposited energies in datasets 1 (left), 2 (middle), and 3 (right). Dashed red bands represent the 10% deviation interval of the generated samples when compared to GEANT predictions. The Earth mover's distance (EMD) between each distribution and the GEANT distribution is also provided.

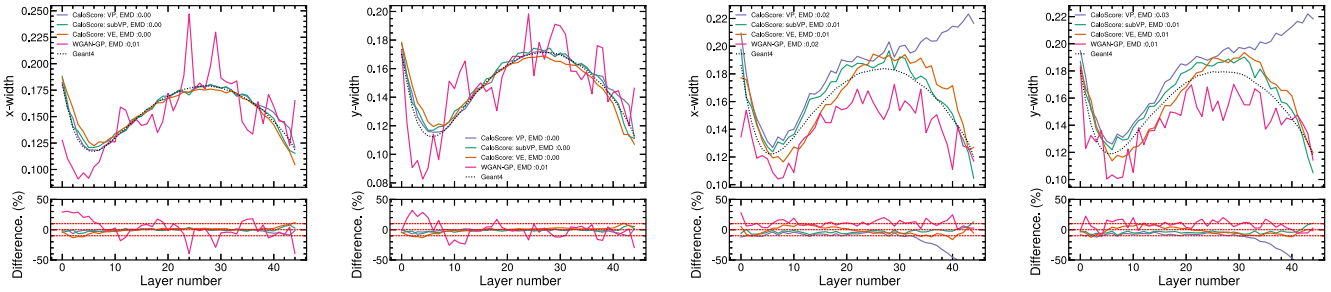

FIG. 6. Comparison of the particle shower width in the x- and y- directions in datasets 2 (first two figures from the left) and 3 (last two figures from the left). Dashed red bands represent the 10% deviation interval of the generated samples when compared to GEANT predictions. The Earth mover's distance (EMD) between each distribution and the GEANT distribution is also provided.

fraction region starts to show deviations from the GEANT predictions and is best described by the WGAN-GP model. The maximum energy fraction as a function of the layer number for datasets 2 and 3 shows a good agreement between the different CaloScore implementations, with most of the distributions showing deviations within the 10% interval.

The angular distribution of the calorimeter shower is investigated in datasets 2 and 3 in terms of the shower width, shown in Fig. 6. The shower width σ_i with $x_i, i \in [1, 2]$ representing the x- and y-coordinates is calculated as:

$$\sigma_i = \sqrt{\langle x_i^2 \rangle - \langle x_i \rangle^2}, \tag{15}$$

with energy-weighted mean defined as

$$\langle x_i \rangle = \frac{\sum_j x_{i,j} E_j}{\sum_j E_j}. \tag{16}$$

A good agreement between all CaloScore implementations and GEANT predictions is observed in dataset 2, with all

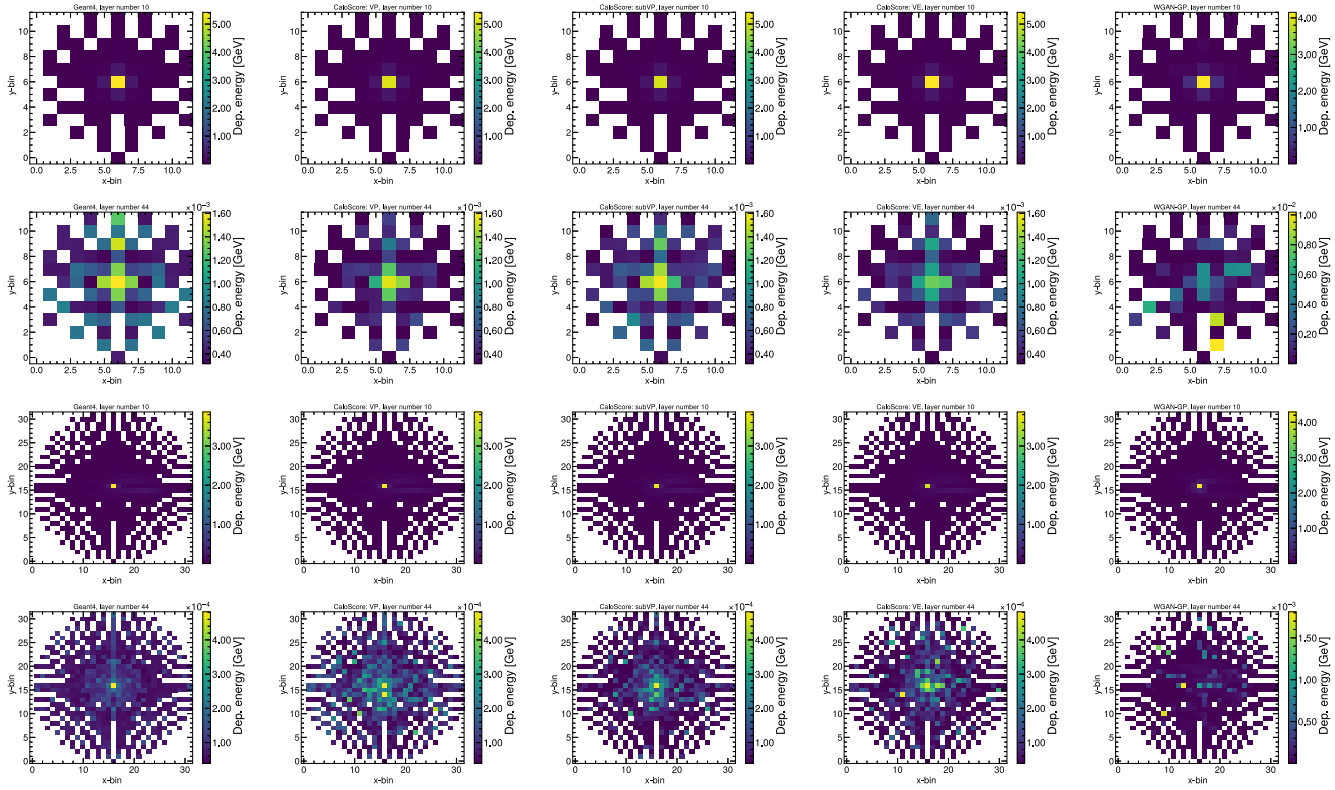

FIG. 7. The 2-dimensional distribution of the mean deposited energy in layers with highest (first and third rows) and lowest (second and fourth rows) mean energy depositions in datasets 2 (first two rows) and 3 (last two rows). Simulated samples from GEANT are shown in the first column, compared with different diffusion models: VP (second column), subVP (third column), and VE (fourth column). The WGAN-GP results are shown in the fourth column.

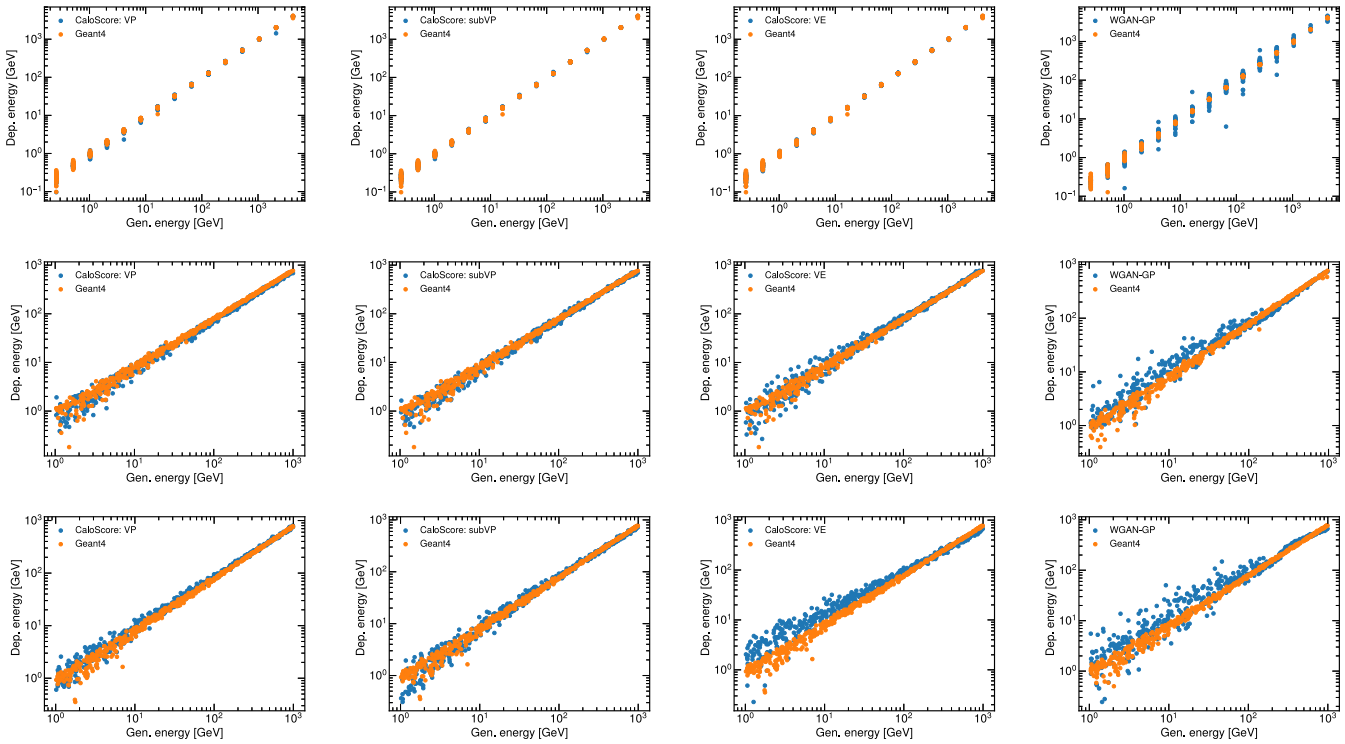

FIG. 8. Deposited energy versus generated energy in CaloScore (blue) and GEANT (orange) for the three different diffusion models: VP (first column), subVP(second column), and VE (third column). WGAN-GP results are shown in the fourth column. First row samples are generated using energies from dataset 1, second row from dataset 2 and third row from dataset 3.

implementations showing less than 10% deviation in all calorimeter layers. However, for dataset 3, the VP implementations shows a disagreement at the last layers of the detector while the shift observed in Fig. 4 for the VE implementation leads to a similar shift in the shower width. Nevertheless, the subVP implementation maintains the same level of agreement as observed in dataset 2. The WGAN-GP implementation however shows larger fluctuations compared to CaloScore and consistently smaller shower widths in dataset 3.

A qualitative assessment of the generation is shown in Fig. 7 for datasets 2 and 3. The 2-dimensional distribution of the average energy deposition is shown in the detector layers with highest (layer 10) and lowest (layer 44) mean energy depositions. Empty entries in the GEANT simulation are a result of the initial voxelization combined with the following transformation to Cartesian coordinates. All voxels with an expected energy deposition above 0 are populated in all CaloScore and WGAN-GP implementations, an indication that the models are able to reproduce the shower diversity from the training set. Images at layer 10 are identical for all generative models, dominated by the central voxel. Layer 44; however, has more voxels sharing a significant fraction of the layer energy. The subVP implementation shows a visually similar average to GEANT compared to the other diffusion implementations, capturing the high energy depositions along the y-axis in dataset 2 and the isotropic pattern around the center in dataset 3. The WGAN-GP implementation, on the other hand, shows higher energy fractions away from the center of the image, where lower energy fractions are expected.

Finally, the assessment of generated samples using different conditional energies is investigated in Fig. 8, by comparing the total deposited energy versus the generated particle energy.

All CaloScore models show similar mean and spread compared to GEANT, with the exception of the VE implementation that shows a wider spread for dataset 2 and higher mean in dataset 3. The WGAN-GP model shows a wider spread in all datasets compared to all CaloScore besides the VE implementation.

We have also explored the classifier metric introduced in CalowFlow whereby a *post-hoc* classifier is trained to distinguish generated showers from GEANT 4 examples. While the classifier could not perfectly identify fake from real showers, it did have an area under the receiver operating characteristic curve (AUC) of about 0.98 for all three models. The classifier was trained using the generated calorimeter images. While this suggests that further (hyperparameter)optimization would be beneficial, it already

serves as an important baseline for other methods. It should also be noted that compared to the dataset introduced in the CaloGAN paper, the datasets considered in this work are either more realistic (dataset 1) or higher dimensional (a factor 10 for dataset 2 and 100 for dataset 3) and would be interesting to see the classifier metric obtained from models such as CaloFLOW.

VIII. CONCLUSIONS AND OUTLOOK

In this paper we introduced CaloScore, a novel generative model for calorimeter shower simulation based on score-matching, applied for the first time in the context of collider physics.

The performance of CaloScore is studied using the Fast Calorimeter Simulation Challenge 2022 datasets and compared to three different model implementations based on different drift and diffusion coefficients. An additional comparison is also provided using a Wasserstein GAN with gradient penalty. This is the first method to produce results for all three datasets of the challenge and we look forward to comparisons with other models as they become available.

For lower number of voxels, all models are capable of producing realistic calorimeter showers, showing a good agreement with the GEANT simulation in all datasets and for a variety of observables. At the highest dimensional dataset, CaloScore with diffusion process described by the subVP stochastic differential equation is able to produce realistic calorimeter showers while VP and VE SDEs show larger deviations. CaloScore also shows overall better results compared to the WGAN-GP implementation in all distributions investigated in this work with exception to generation time, where the WGAN-GP implementation is 3–16 times faster than CaloScore. CaloScore is also shown to be scalable, with number of trainable parameters sensitive to the overall model architecture rather than the total dimensionality of the dataset.

While the voxelization strategy used in dataset 1 minimizes the number of empty voxels, the irregular voxelization reduces the geometrical information present in the calorimeter shower. The geometrical information is included as an important inductive bias for datasets 2 and 3 and leads to an order of magnitude fewer trainable parameters in the model compared to the one used in dataset 1. Moreover, a regular voxelization amenable to convolutional operations is obtained in datasets 2 and 3 only after an additional transformation of coordinates. Since the transformation is applied on the voxelized inputs, nonphysical artifacts are introduced, such as empty voxels in regions that are not expected to be empty. All of these issues could be addressed if alternative voxelization schemes were available or if access to the datasets prior to any voxelization was possible.

The addition of inductive biases to the model is also expected to improve the generation capabilities of CaloScore, possibly leading to better and even smaller model architectures. Energy conservation in particular is challenging to enforce, since generated samples are not produced during training time, but only at generation time when the reverse stochastic differential equation is solved. We partially address this issue by increasing the dimensionality of dataset 1, introducing an additional entry that stores an overall normalization. Since datasets 2 and 3 rely on the geometrical description of the voxels, this strategy is not readily applicable and would instead benefit from a two-step approach as used in CaloFLOW, where the overall normalization is determined separately and used as a conditional input to a second model that learns the normalized detector response.

The major challenge to be addressed in CaloScore is the generation time, currently requiring hundreds of model evaluations to solve the reverse SDE. While the total generation time of CaloScore is still faster compared to the GEANT simulation, we envision future works targeting high fidelity generation with lower number of function evaluations. Indeed, since this limitation is also observed in applications of diffusion models in general, a number of different attempts are currently being proposed to accelerate the generation procedure [73–77], with feasibility for collider physics applications yet to be studied.

Finally, CaloScore introduces a new generative paradigm to collider physics with scalable training strategy and able to generate realistic calorimeter showers consisting of tens of thousands of dimensions. While the generation time represents the main challenge to be overcome, CaloScore is able to incorporate different advantages from other generative models while addressing some of their limitations. These include scalable and stable training schedule, based on the minimization of the convex score-matching loss, and exact likelihood estimation, previously only available with methods such as normalizing flows.

Scripts used to reproduce the results shown in this document are available at [56].

ACKNOWLEDGMENTS

We thank Jean-Roch Vlimant, David Shih, and Daniel Britzger for feedback on the manuscript. V. M. and B. N. are supported by the U.S. Department of Energy (DOE), Office of Science under Contract No. DE-AC02-05CH11231. This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under

Contract No. DE-AC02-05CH11231 using NERSC Grant No. HEP-ERCAP0021099.

APPENDIX A: SCORE FUNCTION AND THE CONNECTION WITH CONTINUOUS NORMALIZING FLOWS

Full data likelihood access is obtained from the trained score-matching model by first identifying the deterministic ordinary differential equation (ODE) associated to the SDE in Eq. (2) that reads:

$$\mathrm{d}x = \left[f(x,t) - \frac{1}{2}g(t)^2 s_\theta(x,t)\right]\mathrm{d}t = \tilde{f}_\theta(x_t,t)\mathrm{d}t. \quad \text{(A1)}$$

This ODE, named probability flow ODE by [55], has the property of having all trajectories sharing the same marginal probability densities as the SDE in Eq. (2) and is fully determined once the score function is estimated by the generative model. The time evolution of the density is given by the instantaneous change of variables defined in [78]:

$$\log p_0(x_0) = \log p_T(x_T) + \int_0^T \nabla.\tilde{f}_\theta(x_t,t)\mathrm{d}t. \quad \text{(A2)}$$

Equation (A2) is equivalent to the change of variables often used in continuous normalizing flows. This expression can be estimated efficiently by first noticing that

$$\nabla.\tilde{f}_\theta(x_t,t) = \mathrm{Tr}(\nabla\tilde{f}_\theta(x_t,t)), \quad \text{(A3)}$$

where $\nabla\tilde{f}_\theta(x_t,t)$ represents the Jacobian of $\tilde{f}_\theta$ and using algorithms such as the Skilling-Hutchinson trace estimator [79,80] to approximate the trace calculation.

APPENDIX B: CYLINDRICAL TO CARTESIAN COORDINATE TRANSFORMATION

The initial voxelization provided for datasets 2 and 3 are in cylindrical coordinates (r,α,z'). While this set of coordinates reflect the detector symmetry, we found beneficial to convert the voxelization to Cartesian coordinates (x,y,z). A voxel initially described in cylindrical coordinates (r_i,α_i,z_i) is then converted as:

$$x_i = r_i \cos\alpha_i \quad \text{(B1)}$$

$$y_i = r_i \sin\alpha_i \quad \text{(B2)}$$

$$z_i = z'_i, \quad \text{(B3)}$$

where for simplicity we assume $r_i \in [0,1]$ and $\alpha_i \in [0,2\pi]$, resulting in $x \in [-1,1]$ and $y \in [-1,1]$. Since the overall transformation is not linear, some voxels of the new set of coordinates will always be empty, while others contain the sum of multiple voxels in the initial set of coordinates.

APPENDIX C: GENERATION QUALITY FOR DIFFERENT SAMPLING PARAMETERS

Results presented in this work are derived using the value of 0.2 for signal-to-noise-ratio r of the Langevin corrector [Eq. (11)] and fixed number of times steps of 100 for dataset 1 and 200 for datasets 2 and 3. This choice of parameters is used to balance the generation quality and the generation time. In Fig. 9 different choices of r are compared while maintaining the same number of steps as before for all datasets and diffusion models, evaluated on the distribution of maximum energy fraction in a single voxel for dataset 1 and average energy deposition per layer for datasets 2 and 3, the distributions that show were observed to be more sensitive to the choice of generation parameters used.

Different choices of r yield similar results for all diffusion models in both datasets 1 and 2. On the other hand, the corrector step has a stronger effect on dataset 3 and is crucial to achieve good generation quality with minimal additional computational complexity.

Contrary to the corrector step, increasing the number of time steps directly affect the generation time, dominated by the number of score function evaluations. Different choices of number of times steps are shown in Fig. 10 with r value fixed to the baseline value.

In all cases using fewer time steps deteriorate the agreement with GEANT while additional steps are able to improve the agreement in both datasets 1 and 3 while dataset 2 shows similar results compared to the baseline. In Table III, the time required to generate 100 calorimeter showers using different number of time steps is listed. Even though the additional time steps improve the simulation quality, the time to generate the same amount of new observations increase by more than a factor 2.

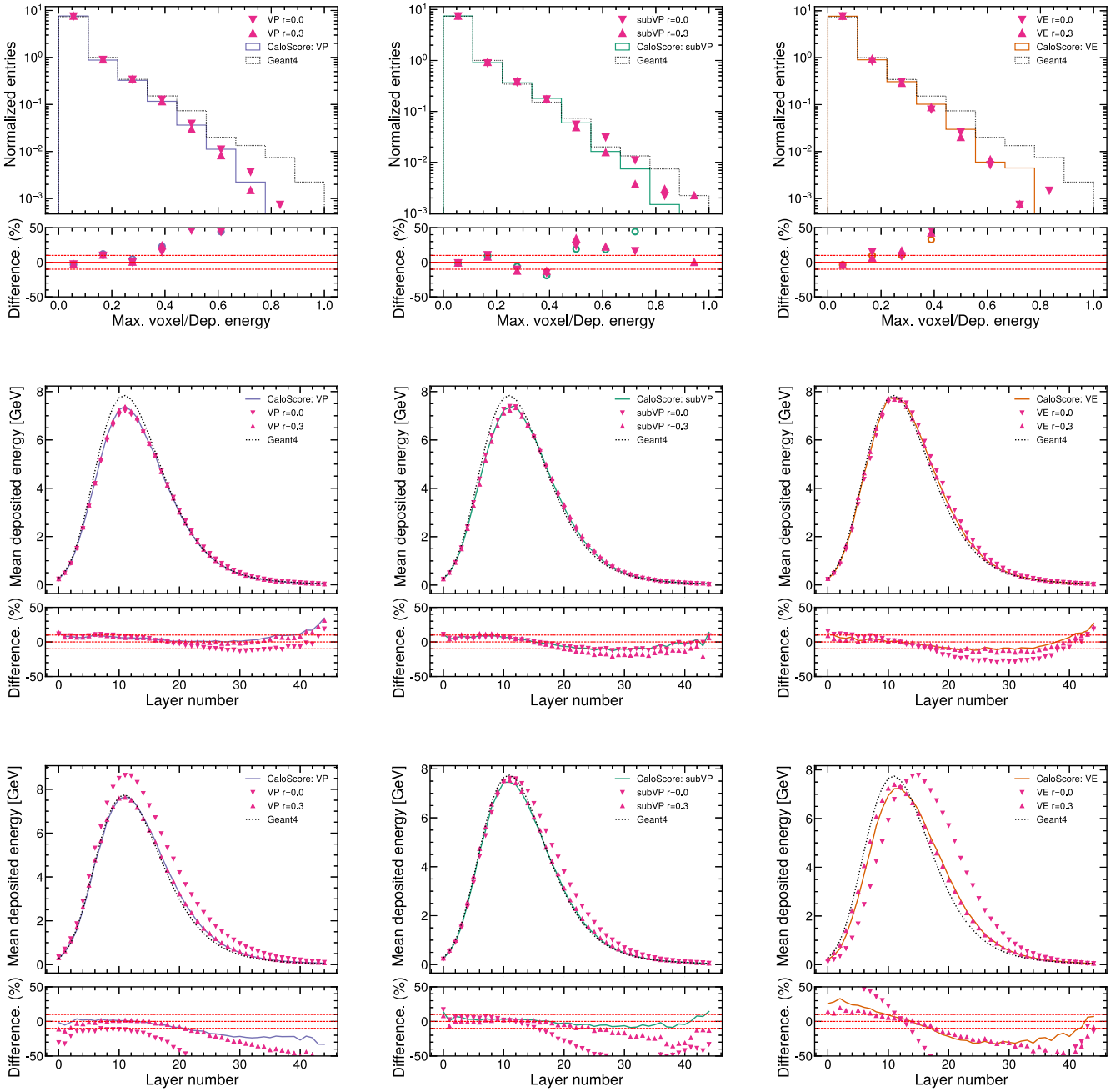

FIG. 9. Comparison of different signal-to-noise choices on the maximum energy fraction deposited in a single voxel for dataset 1 (top) and average energy deposition per layer for datasets 2 (middle) and 3 (bottom) for the three different diffusion models: VP (left), subVP (middle), and VE (right).

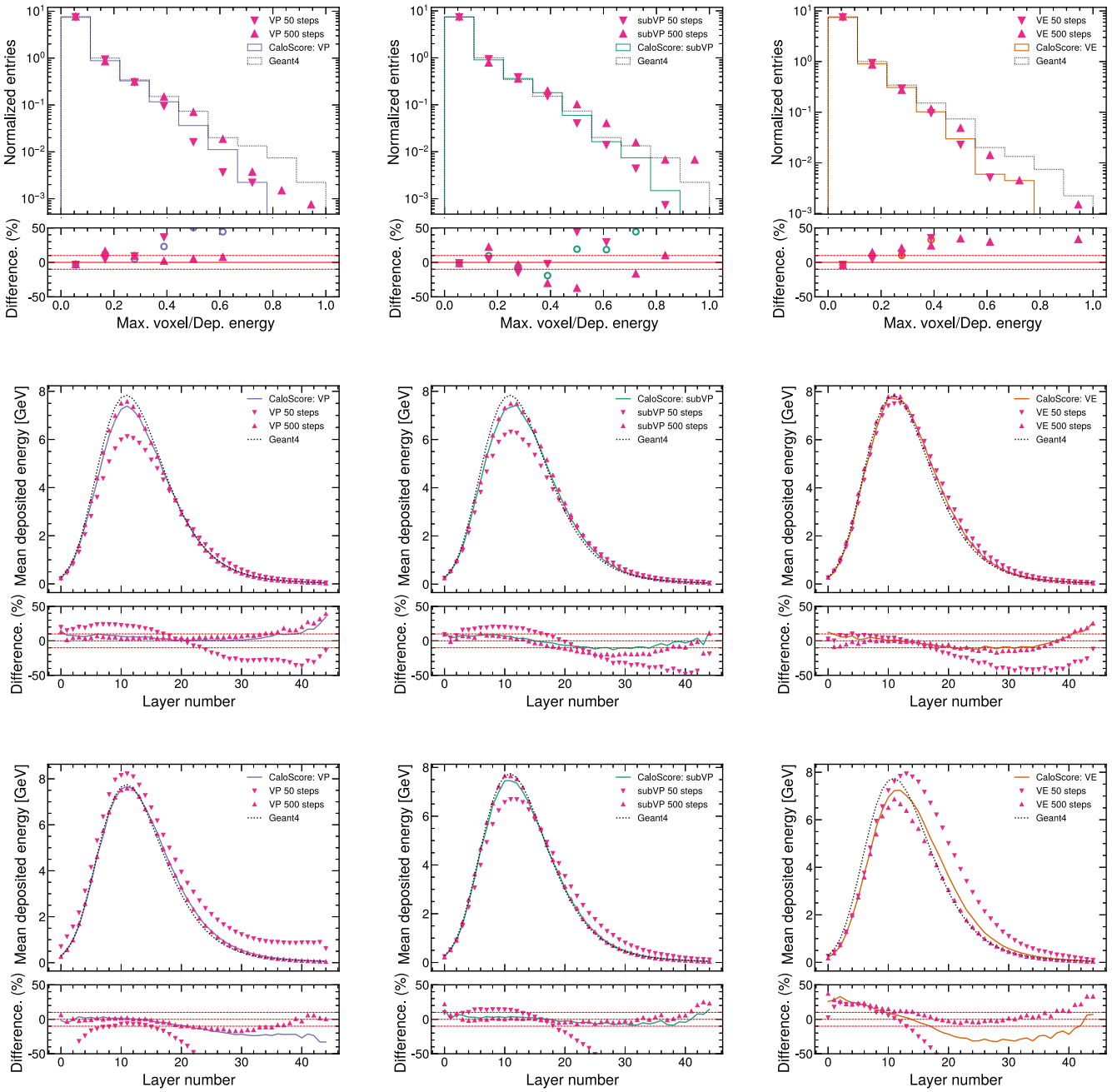

FIG. 10. Comparison of different number of time steps on the maximum energy fraction deposited in a single voxel for dataset 1 (top) and average energy deposition per layer for datasets 2 (middle) and 3 (bottom) for the three different diffusion models: VP (left), subVP (middle), and VE (right).

TABLE III. Time comparison to generate 100 calorimeter showers using the baseline model and different number of time steps.

Dataset	Baseline [s]	N = 50 [s]	N = 500 [s]
Dataset 1	4.0	2.9	14.8
Dataset 2	5.8	2.7	13.1
Dataset 3	33.4	10.3	80.2

[1] S. Agostinelli *et al.*, Nucl. Instrum. Methods Phys. Res., Sect. A **506**, 250 (2003).
[2] J. Allison *et al.*, IEEE Trans. Nucl. Sci. **53**, 270 (2006).
[3] J. Allison *et al.*, Nucl. Instrum. Methods Phys. Res., Sect. A **835**, 186 (2016).
[4] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, arXiv:1406.2661.
[5] D. P. Kingma and M. Welling, arXiv:1312.6114.
[6] D. Rezende and S. Mohamed, Proc. Int. Conf. Mach. Learn. **37**, 1530 (2015).
[7] L. de Oliveira, M. Paganini, and B. Nachman, Comput. Software Big Sci. **1**, 4 (2017).
[8] M. Paganini, L. de Oliveira, and B. Nachman, Phys. Rev. Lett. **120**, 042003 (2018),
[9] M. Paganini, L. de Oliveira, and B. Nachman, Phys. Rev. D **97**, 014021 (2018),
[10] S. Vallecorsa, F. Carminati, and G. Khattak, EPJ Web Conf. **214**, 02010 (2019).
[11] C. Ahdida *et al.* (SHiP Collaboration), J. Instrum. **14**, P11028 (2019).
[12] V. Chekalina, E. Orlova, F. Ratnikov, D. Ulyanov, A. Ustyuzhanin, and E. Zakharov, EPJ Web Conf. **214**, 02034 (2019).
[13] The ATLAS Collaboration, Report No. ATL-SOFT-PUB-2018-001 (2018).
[14] F. Carminati, A. Gheata, G. Khattak, P. Mendez Lorenzo, S. Sharan, and S. Vallecorsa, EPJ Web Conf. **1085**, 032016 (2018).
[15] S. Vallecorsa, EPJ Web Conf. **1085**, 022005 (2018).
[16] P. Musella and F. Pandolfi, Comput. Software Big Sci. **2**, 8 (2018).
[17] M. Erdmann, L. Geiger, J. Glombitza, and D. Schmidt, Comput. Software Big Sci. **2**, 4 (2018).
[18] K. Deja, T. Trzcinski, and u. Graczykowski, EPJ Web Conf. **214**, 06003 (2019).
[19] D. Derkach, N. Kazeev, F. Ratnikov, A. Ustyuzhanin, and A. Volokhova, Nucl. Instrum. Methods Phys. Res., Sect. A **952**, 161804 (2020).
[20] M. Erdmann, J. Glombitza, and T. Quast, Comput. Software Big Sci. **3**, 4 (2019).
[21] L. de Oliveira, M. Paganini, and B. Nachman, Tips and Tricks for Training GANs with Physics Constraints (2017), https://dl4physicalsciences.github.io/files/nips_dlps_2017_26.pdf.
[22] L. de Oliveira, M. Paganini, and B. Nachman, J. Phys. Conf. Ser. **1085**, 042017 (2018).
[23] B. Hooberman, A. Farbin, G. Khattak, V. Pacela, M. Pierini, J.-R. Vlimant, M. Spiropulu, W. Wei, M. Zhang, and S. Vallecorsa, Calorimetry with Deep Learning: Particle Classification, Energy Regression, and Simulation for High-Energy Physics (2017), https://dl4physicalsciences.github.io/files/nips_dlps_2017_15.pdf.
[24] D. Belayneh *et al.*, Eur. Phys. J. C **80**, 688 (2020).
[25] E. Buhmann, S. Diefenbacher, E. Eren, F. Gaede, G. Kasieczka, A. Korol, and K. Kruger, arXiv:2005.05334.
[26] S. Diefenbacher, E. Eren, G. Kasieczka, A. Korol, B. Nachman, and D. Shih, J. Instrum. **15**, P11004 (2020).
[27] R. Kansal, J. Duarte, B. Orzari, T. Tomei, M. Pierini, M. Touranakou, J.-R. Vlimant, and D. Gunopoulos, *34th Conference on Neural Information Processing Systems* (2020), arXiv:2012.00173.
[28] A. Maevskiy, F. Ratnikov, A. Zinchenko, and V. Riabov, Eur. Phys. J. C **81**, 599 (2021).
[29] F. Rehm, S. Vallecorsa, V. Saletore, H. Pabst, A. Chaibi, V. Codreanu, K. Borras, and D. Krücker, arXiv:2103.10142.
[30] F. Rehm, S. Vallecorsa, K. Borras, and D. Krücker, arXiv:2103.13698.
[31] F. Rehm, S. Vallecorsa, K. Borras, and D. Krücker, EPJ Web Conf. **251**, 03042 (2021).
[32] R. Kansal, J. Duarte, H. Su, B. Orzari, T. Tomei, M. Pierini, M. Touranakou, J.-R. Vlimant, and D. Gunopulos, arXiv:2106.11535.
[33] G. R. Khattak, S. Vallecorsa, F. Carminati, and G. M. Khan, Eur. Phys. J. C **82**, 386 (2022).
[34] L. Anderlini, arXiv:2110.07925.
[35] E. Buhmann, S. Diefenbacher, E. Eren, F. Gaede, D. Hundhausen, G. Kasieczka, W. Korcari, K. Krüger, P. McKeown, and L. Rustige, Mach. Learn. Sci. Technol. **3**, 025014 (2022).
[36] S. Bieringer, A. Butter, S. Diefenbacher, E. Eren, F. Gaede, D. Hundhausen, G. Kasieczka, B. Nachman, T. Plehn, and M. Trabs, J. Instrum. **17**, P09028 (2022).
[37] J. W. Monk, J. High Energy Phys. 12 (2018) 021.
[38] K. Dohi, arXiv:2009.04842.
[39] E. Buhmann, S. Diefenbacher, E. Eren, F. Gaede, G. Kasieczka, A. Korol, and K. Krüger, EPJ Web Conf. **251**, 03003 (2021).
[40] A. Hariri, D. Dyachkova, and S. Gleyzer, arXiv:2104.01725.
[41] C. Fanelli and J. Pomponi, Mach. Learn. Sci. Technol. **1**, 015010 (2019).
[42] B. Orzari, T. Tomei, M. Pierini, M. Touranakou, J. Duarte, R. Kansal, J.-R. Vlimant, and D. Gunopulos, in *38th International Conference on Machine Learning Conference* (2021), arXiv:2109.15197.
[43] M. Touranakou, N. Chernyavskaya, J. Duarte, D. Gunopulos, R. Kansal, B. Orzari, M. Pierini, T. Tomei, and J.-R. Vlimant, Mach. Learn. Sci. Technol. **3**, 035003 (2022).
[44] C. Krause and D. Shih, arXiv:2106.05285.
[45] C. Krause and D. Shih, arXiv:2110.11377.
[46] G. Aad *et al.* (ATLAS Collaboration), Comput. Software Big Sci. **6**, 7 (2022).
[47] M. Arjovsky, S. Chintala, and L. Bottou, arXiv.1701.07875.
[48] The Phase-2 Upgrade of the CMS Endcap Calorimeter, Technical Report No. CERN-LHCC-2017-023, CERN, Geneva, 2017.
[49] J. Repond *et al.* (CALICE Collaboration), J. Instrum. **13**, P12022 (2018).
[50] M. Faucci Giannelli, G. Kasieczka, C. Krause, B. Nachman, D. Salamani, D. Shih, and A. Zaborowska, Fast Calorimeter Simulation Challenge 2022—Dataset 3 (2022).
[51] M. F. Giannelli, G. Kasieczka, C. Krause, B. Nachman, D. Salamani, D. Shih, and A. Zaborowska, Fast Calorimeter Simulation Challenge 2022—Dataset 1 (2022).
[52] M. Faucci Giannelli, G. Kasieczka, C. Krause, B. Nachman, D. Salamani, D. Shih, and A. Zaborowska, Fast Calorimeter Simulation Challenge 2022—Dataset 2 (2022).
[53] Y. Song and S. Ermon, arXiv:1907.05600.

[54] J. Ho, A. Jain, and P. Abbeel, Adv. Neural Inf. Process. Syst. **33**, 6840 (2020).
[55] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, arXiv:abs/2011.13456.
[56] V. Mikuni and B. Nachman, v1.0 (2022), https://github.com/ViniciusMikuni/CaloScore.
[57] P. Vincent, Neural Comput. **23**, 1661 (2011).
[58] B. D. Anderson, Stoch. Proc. Appl. **12**, 313 (1982).
[59] S. Särkkä and A. Solin, *Applied Stochastic Differential Equations* (Cambridge University Press, Cambridge, England, 2019), Vol. 10.
[60] P. E. Kloeden and E. Platen, in *Numerical Solution of Stochastic Differential Equations* (Springer, New York, 1992), pp. 103–160.
[61] G. Parisi, Nucl. Phys. **B180**, 378 (1981).
[62] U. Grenander and M. I. Miller, J. R. Stat. Soc. **56**, 549 (1994).
[63] Fast simulation of the ATLAS calorimeter system with Generative Adversarial Networks, Technical Report No. ATL-SOFT-PUB-2020-006, CERN, Geneva, 2020.
[64] O. Ronneberger, P. Fischer, and T. Brox, in *International Conference on Medical Image Computing and Computer-Assisted Intervention* (Springer, New York, 2015), pp. 234–241.
[65] P. Ramachandran, B. Zoph, and Q. V. Le, arXiv:1710.05941.
[66] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng, Adv. Neural Inf. Process. Syst. **33**, 7537 (2020).
[67] M. Arjovsky, S. Chintala, and L. Bottou, in *International Conference on Machine Learning* (PMLR, Sydney, Australia, 2017), pp. 214–223.
[68] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, Adv. Neural Inf. Process. Syst. **30**, 5769 (2017).
[69] E. Buhmann, S. Diefenbacher, E. Eren, F. Gaede, G. Kasieczka, A. Korol, and K. Krüger, Comput. Software Big Sci. **5**, 13 (2021).
[70] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, in OSDI'16: Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation (2016), Vol. **16**, pp. 265–283, https://dl.acm.org/doi/10.5555/3026877.3026899.
[71] D. Kingma and J. Ba, arXiv:1412.6980.
[72] A. Sergeev and M. D. Balso, arXiv:1802.05799.
[73] A. Jolicoeur-Martineau, K. Li, R. Piché-Taillefer, T. Kachman, and I. Mitliagkas, arXiv:2105.14080.
[74] Q. Zhang and Y. Chen, arXiv:2204.13902.
[75] Z. Lyu, X. Xu, C. Yang, D. Lin, and B. Dai, arXiv:2205.12524.
[76] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu, arXiv:2206.00927.
[77] Z. Xiao, K. Kreis, and A. Vahdat, in *International Conference on Learning Representations (ICLR)* (International Conference on Learning Representations (ICLR), 2022).
[78] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, arXiv:abs/1806.07366.
[79] J. Skilling, The eigenvalues of mega-dimensional matrices, in *Maximum Entropy and Bayesian Methods: Cambridge, England, 1988*, edited by J. Skilling (Springer Netherlands, Dordrecht, 1989), pp. 455–466.
[80] M. Hutchinson, Commun. Stat. Simul. Comput. **19**, 433 (1990).