Score-based Generative Models for Calorimeter Shower Simulation

Vinicius M. Mikuni, Ben Nachman

Detector simulation

- Calorimeters **expensive to simulate**:
  - Full detector simulation of a particle can take up to **a minute** and we still need **billions of particles simulated**
- For previous LHC runs, detector simulation used around **40% of all computing resources** and may go beyond the available budget for future runs

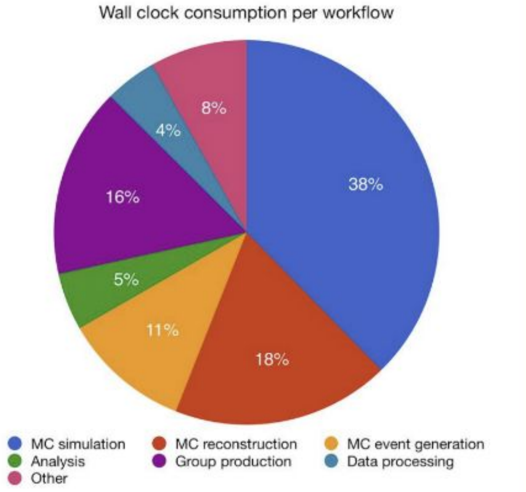
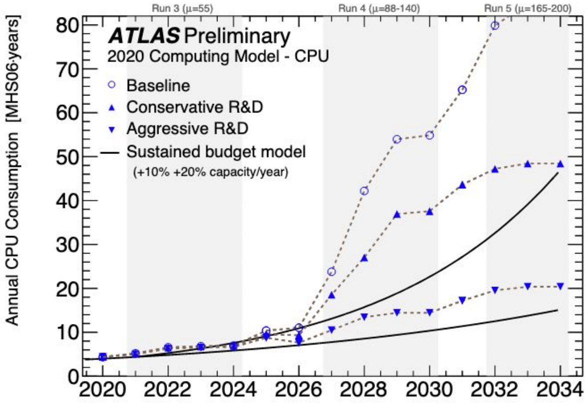
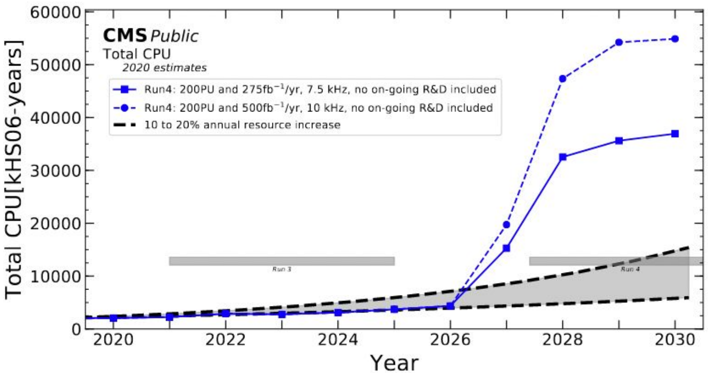


Figure 1: ATLAS CPU hours used by various activities in 2018

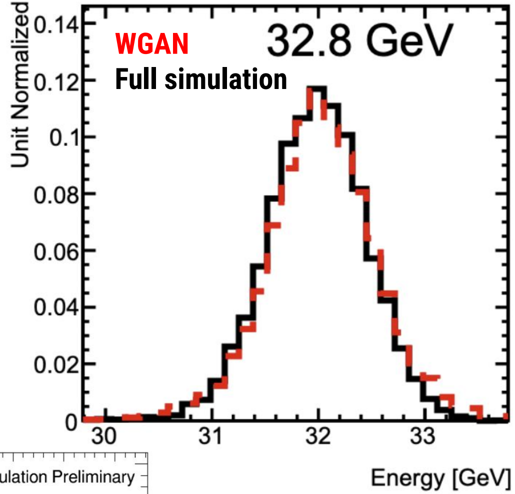


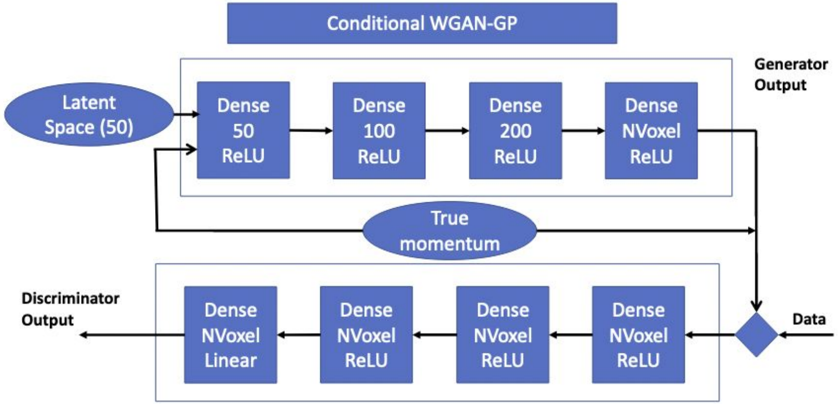


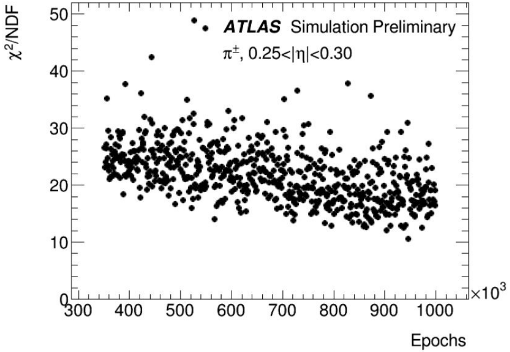
FastCaloGAN

See Michele's talk for updates!

- The **ATLAS collaboration** already has a WGAN-GP planned to replace the full simulation routine
- **Fully-connected** architecture that leads to orders of magnitude faster generation compared to full simulation
- Best epoch chosen based on the χ^2 of the energy distribution (sum of the energy depositions)







Diffusion models

"An astronaut lounging in a tropical resort in space in a photorealistic style"

https://openai.com/dall-e-2/

Diffusion models

Salmon in the river

Machine learning for jets

Score matching/denoising/diffusion

Denoise diffusion models are the newest state-of-the-art generative models for image generation.

Pros:

- **Stable training**: convex loss function
- **Scalability**: Network complexity is more sensitive to the architecture than the dimensionality
- **Access to data likelihood after training**: similar to NFs, but overall normalization is not required during training

Cons:

- **Slow sampling**: Possibly 1000s of model evaluations to generate realistic images

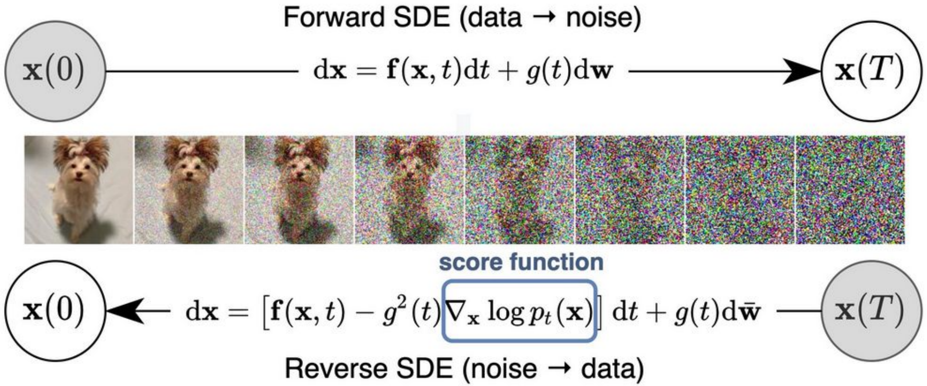
Image Generation on CIFAR-10

Rank	Model	FID	Inception score	bits/dimension	FID-10k-test	Paper	Code	Result	Year	Tags
1	**StyleGAN-XL**	1.85				StyleGAN-XL: Scaling StyleGAN to Large Diverse Datasets			2022	
2	**LSGM** (FID)	2.10		3.43		Score-based Generative Modeling in Latent Space			2021	VAE Score-based
3	**Subspace Diffusion** (NSCN++)	2.17	9.99			Subspace Diffusion Generative Models			2022	Score-based
4	**LSGM** (balanced)	2.17		2.95		Score-based Generative Modeling in Latent Space			2021	VAE Score-based
5	**NCSN++**	2.20	9.73			Score-Based Generative Modeling through Stochastic Differential Equations			2020	Score-based

How it works?

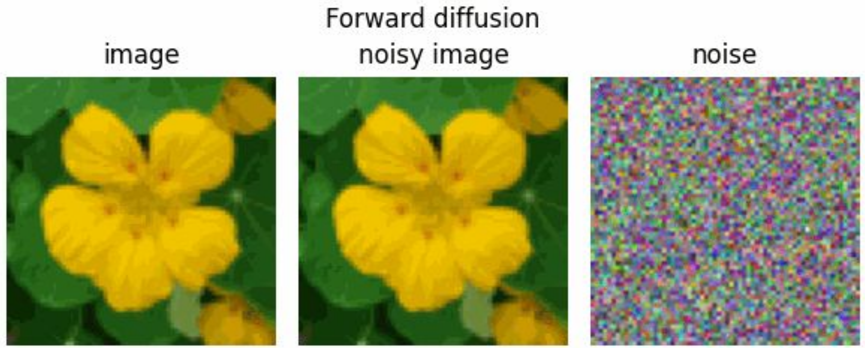
Starting from an image, we can define a **diffusion** process that add small perturbations to the data until transforming it to a tractable distribution

- If we can reverse this process, we can start from the noise distribution and **denoise** to generate a new image
- Since the **drift f(x,t)** and **diffusion g(t)** coefficients are known, the tricky part is to estimate the **data gradient**, also known as the **score function**
- **Goal of the network:** estimate the data score



How it works?

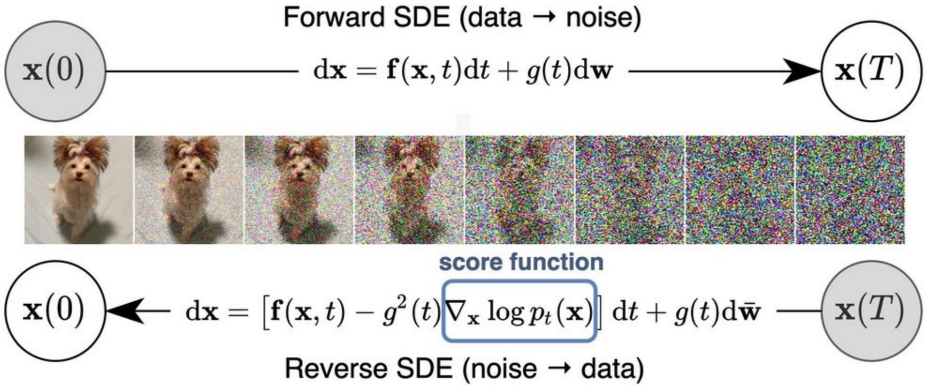
- We would like to find $\mathbf{s}_\theta(\mathbf{x})$ that minimizes: $\frac{1}{2}\mathbb{E}_{p_{\text{data}}}[\|\mathbf{s}_\theta(\mathbf{x}) - \nabla_\mathbf{x} \log p_{\text{data}}(\mathbf{x})\|_2^2]$
- Without knowing the density, we can instead, use a small data perturbation $q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})$
- And estimate the score of the perturbed data by minimizing $\frac{1}{2}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})p_{\text{data}}(\mathbf{x})}[\|\mathbf{s}_\theta(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})\|_2^2]$
- Example: **Gaussian perturbation** $q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) = \mathcal{N}(\tilde{\mathbf{x}} \mid \mathbf{x}, \sigma^2 I)$ $\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) = -(\tilde{\mathbf{x}}-\mathbf{x})/\sigma^2$
- We minimize $\frac{1}{2}\mathbb{E}[\|\sigma \mathbf{s}_\theta(\tilde{\mathbf{x}}, \sigma) + \frac{\tilde{\mathbf{x}}-\mathbf{x}}{\sigma}\|_2^2]$
- with $\frac{\tilde{\mathbf{x}}-\mathbf{x}}{\sigma} \sim \mathcal{N}(0, I)$



Gif from:
https://keras.io/examples/generative/ddim/

Perturbation kernels

- Let's go back to the diffusion equation
- In principle, we can choose any function for **f** and **g** but the common ones are those in which the transition kernel $p(x_t|x)$ is gaussian. That can be accomplished if **f is an affine function**



Variance preserving (VP):
$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\mathbf{w}.$$

Variance exploding (VE):
$$\mathrm{d}\mathbf{x} = \sqrt{\frac{\mathrm{d}\left[\sigma^2(t)\right]}{\mathrm{d}t}}\mathrm{d}\mathbf{w}.$$

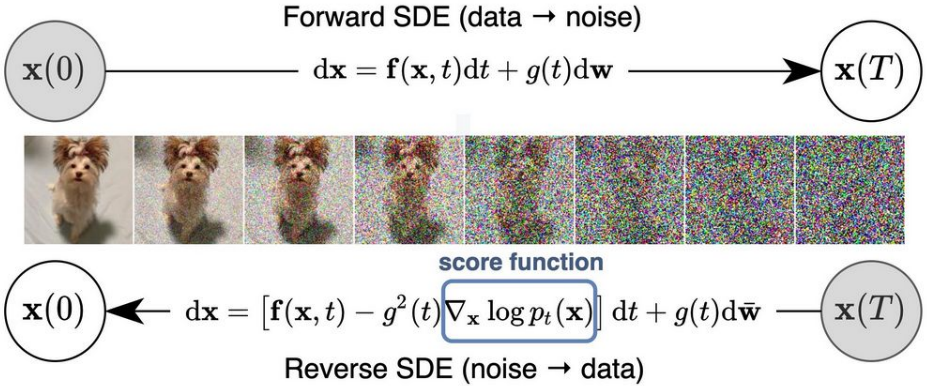
Sub Variance preserving(subVP):
$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\,\mathrm{d}t + \sqrt{\beta(t)(1 - e^{-2\int_0^t \beta(s)\mathrm{d}s})}\mathrm{d}\mathbf{w}.$$

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	$\mathcal{N}(x(0)e^{-\frac{1}{2}\int_0^t \beta(s)\mathrm{d}s}, (1 - e^{-\int_0^t \beta(s)\mathrm{d}s})^2)$

Generation?

- Generation of new samples is done by solving the **reverse SDE**
- Langevin dynamics is used to draw samples from **p(x)** using only the **score function**
- High fidelity samples require small time steps, possibly leading to **1000s** of network evaluations to produce a new sample
- For Calorimeter generation, **O(100)** evaluations are enough to produce precise results

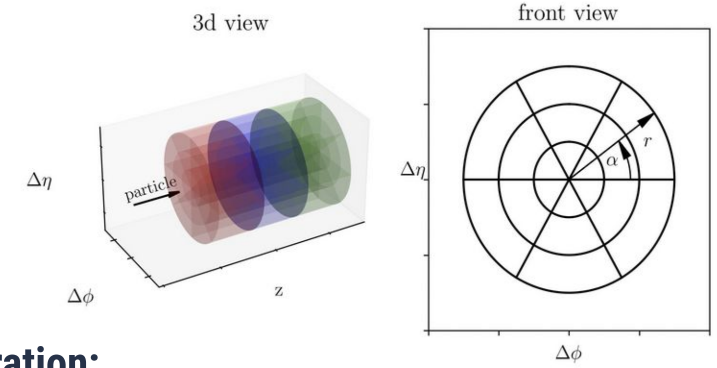


$$\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \epsilon \nabla_{\mathbf{x}} \log p(\mathbf{x}) + \sqrt{2\epsilon}\, \mathbf{z}_i, \quad i = 0, 1, \cdots, K,$$

Calorimeter shower generation

Let's use a realistic example: **Fast Calorimeter Simulation Challenge 2022**

- Converting initial sets voxelized in (alpha,r) coordinates to (eta,phi) coordinates
  - **Dataset 1**: 368
  - **Dataset 2**: 45x12x12 = **6480**
  - **Dataset 3**: 45x32x32 = **46080**
- **Datasets 2 and 3: 3D convolutional layers**.
  - Number of trainable parameters **~2M**
- **Dataset 1: 1D convolutional layers**
  - Number of trainable parameters **~32M**
- **For comparison:** Multiple normalizing flow model implemented in a **30x10x10** dataset used **72M parameters**

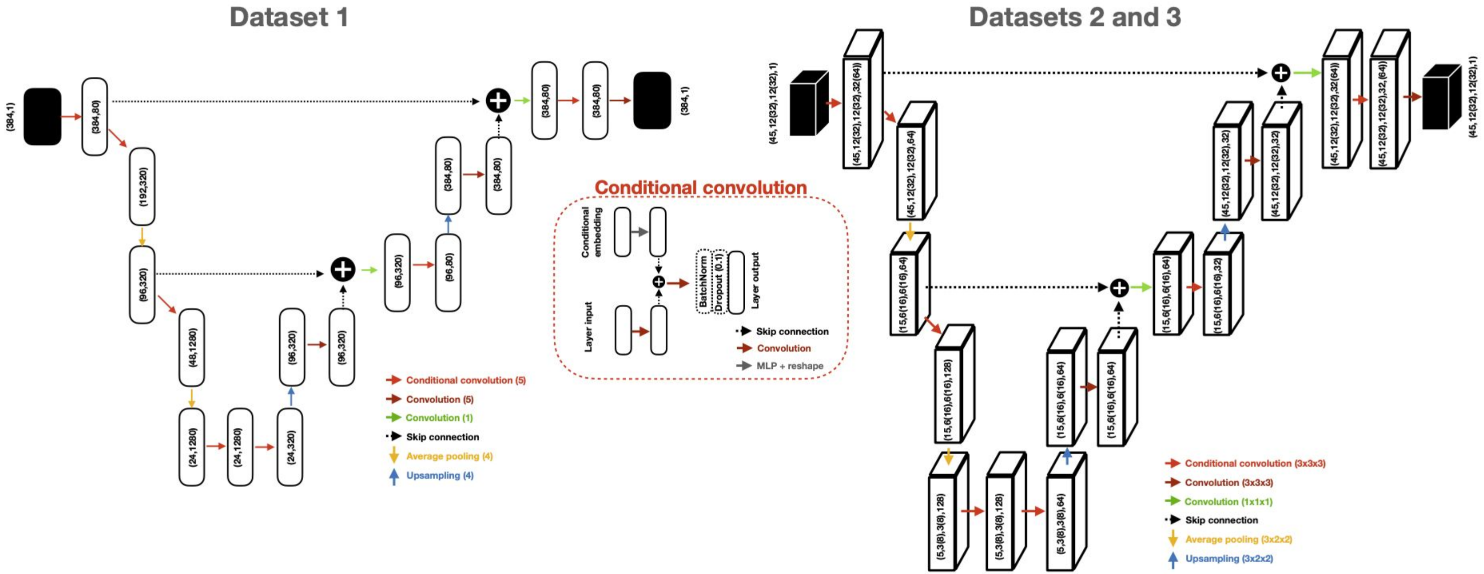


Data curation:

- Each energy deposition $\mathbf{E_i}$ is normalized by the generated energy **E** and transformed to log space: $\mathbf{u = E_i/E}$ and $u_{\text{logit}} = \log \frac{x}{1-x},$

$$x = \alpha + (1 - 2\alpha)u \quad \text{and} \quad \alpha = 10^{-6}.$$

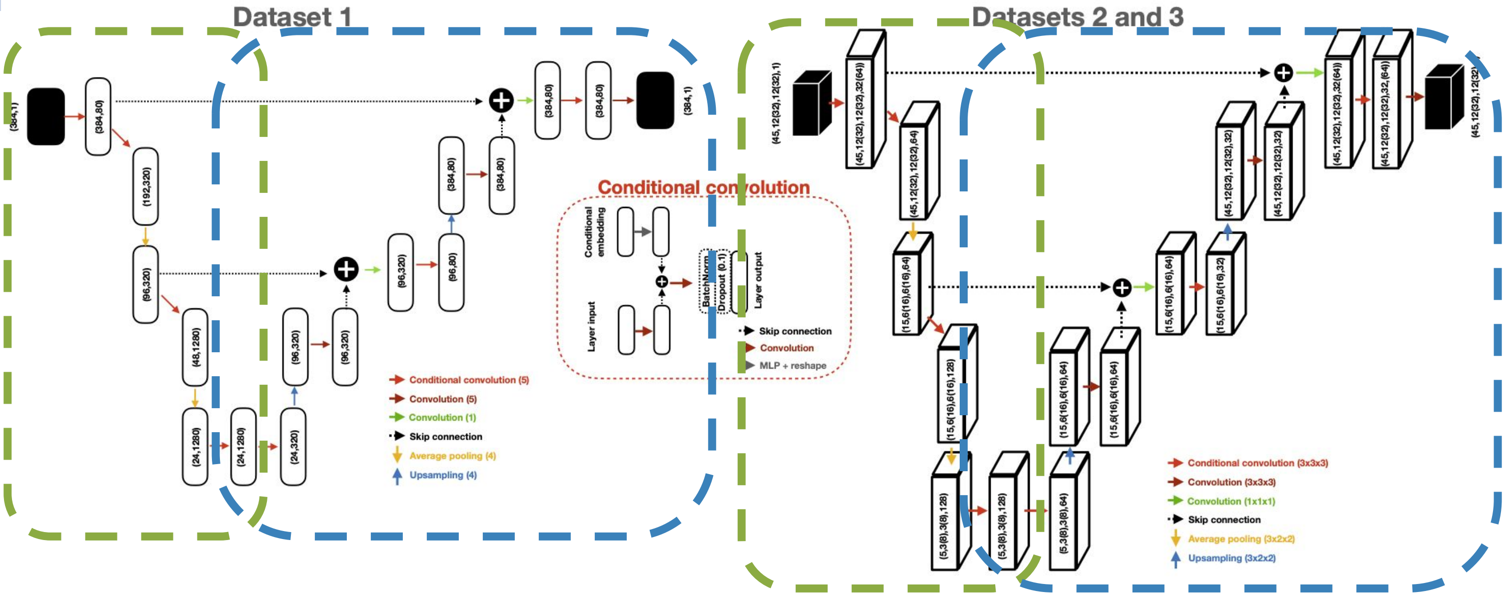
Calorimeter shower generation



Very simple **U-NET** model used to build the score function

- Lots of new developments over the years, adding attention between layers, additional skip connections, but kept it simple for this application
- **Conditional information** is added to convolutional layers as a **bias term**

Calorimeter shower generation

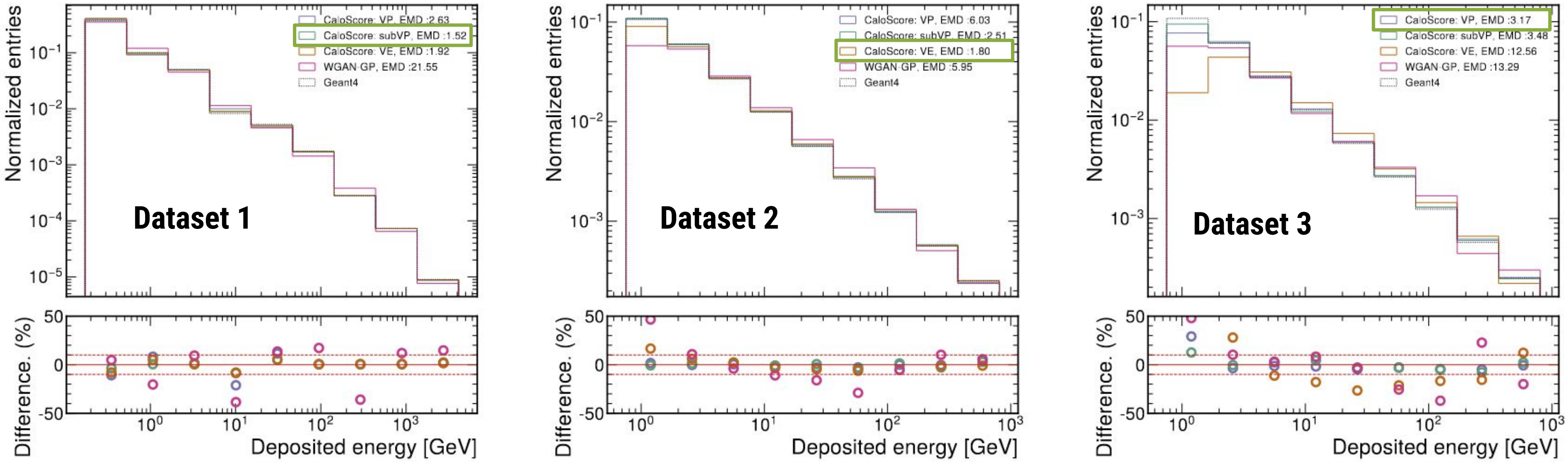


WGAN-GP:
Critic
Generator

Very simple **U-NET** model used to build the score function

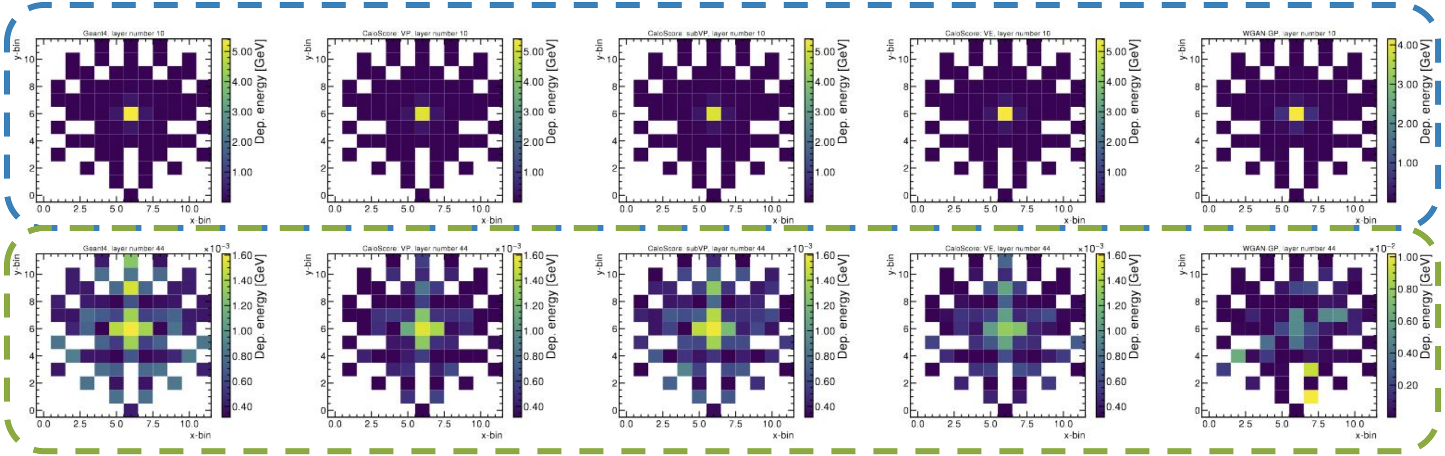
- Lots of new developments over the years, adding attention between layers, additional skip connections, but kept it simple for this application
- **Conditional information** is added to convolutional layers as a **bias term**

Results

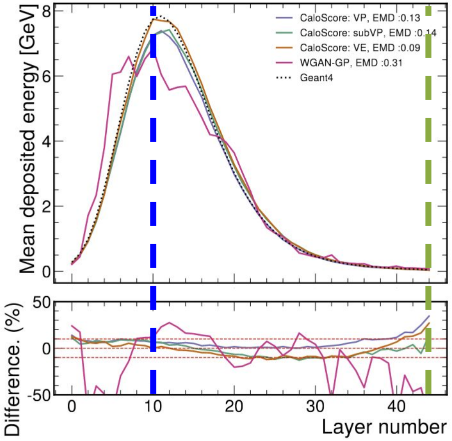


- **Total energy deposited in the calorimeter material**
- The 1-Wasserstein distance (**EMD**) between each generative model and Geant4 are shown for comparison

Results: Visualization



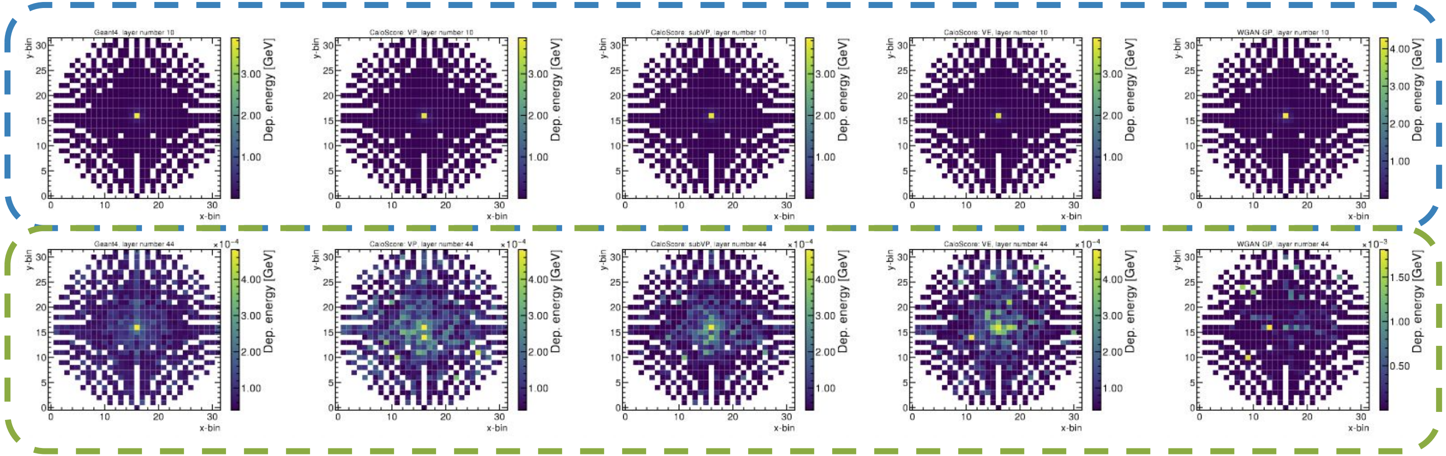
Geant4 VP subVP VE WGAN-GP



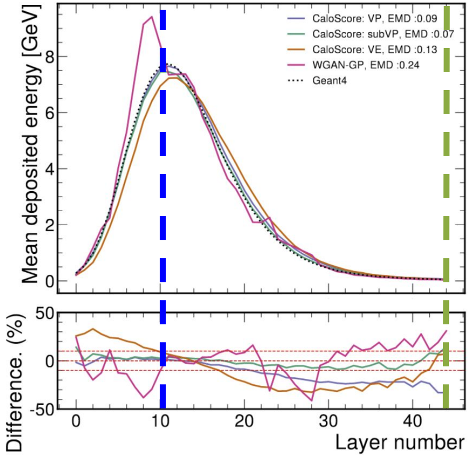
Weird shapes are a result of the coordinate transformation

- **Mean deposited energy** for each calorimeter layer in **dataset 2**
- Visualize the energy deposition in the layers with highest (**10**) and lowest (**44**) expected energies
- Layer 10 is similar for all models, but layer 44 shows discrepancies compared to Geant

Results: Visualization



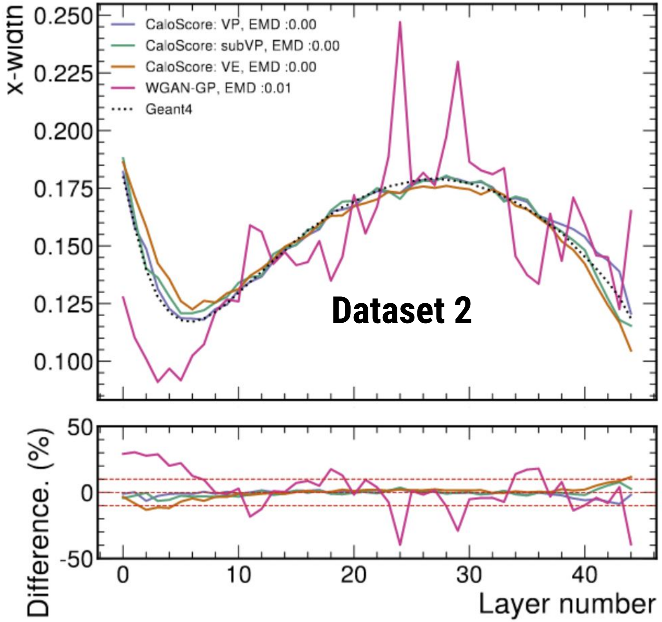
Geant4 **VP** **subVP** **VE** **WGAN-GP**

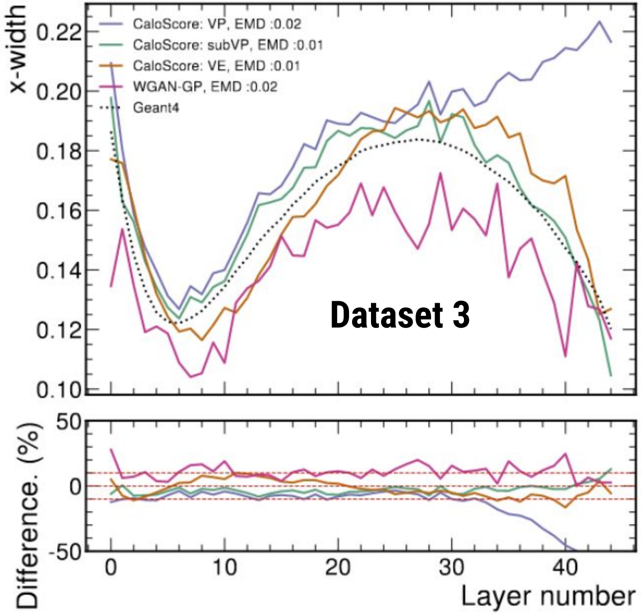


Weird shapes are a result of the coordinate transformation

- **Mean deposited energy** for each calorimeter layer in **dataset 2**
- Visualize the energy deposition in the layers with highest (**10**) and lowest (**44**) expected energies
- Layer 10 is similar for all models, but layer 44 shows discrepancies compared to Geant

Results

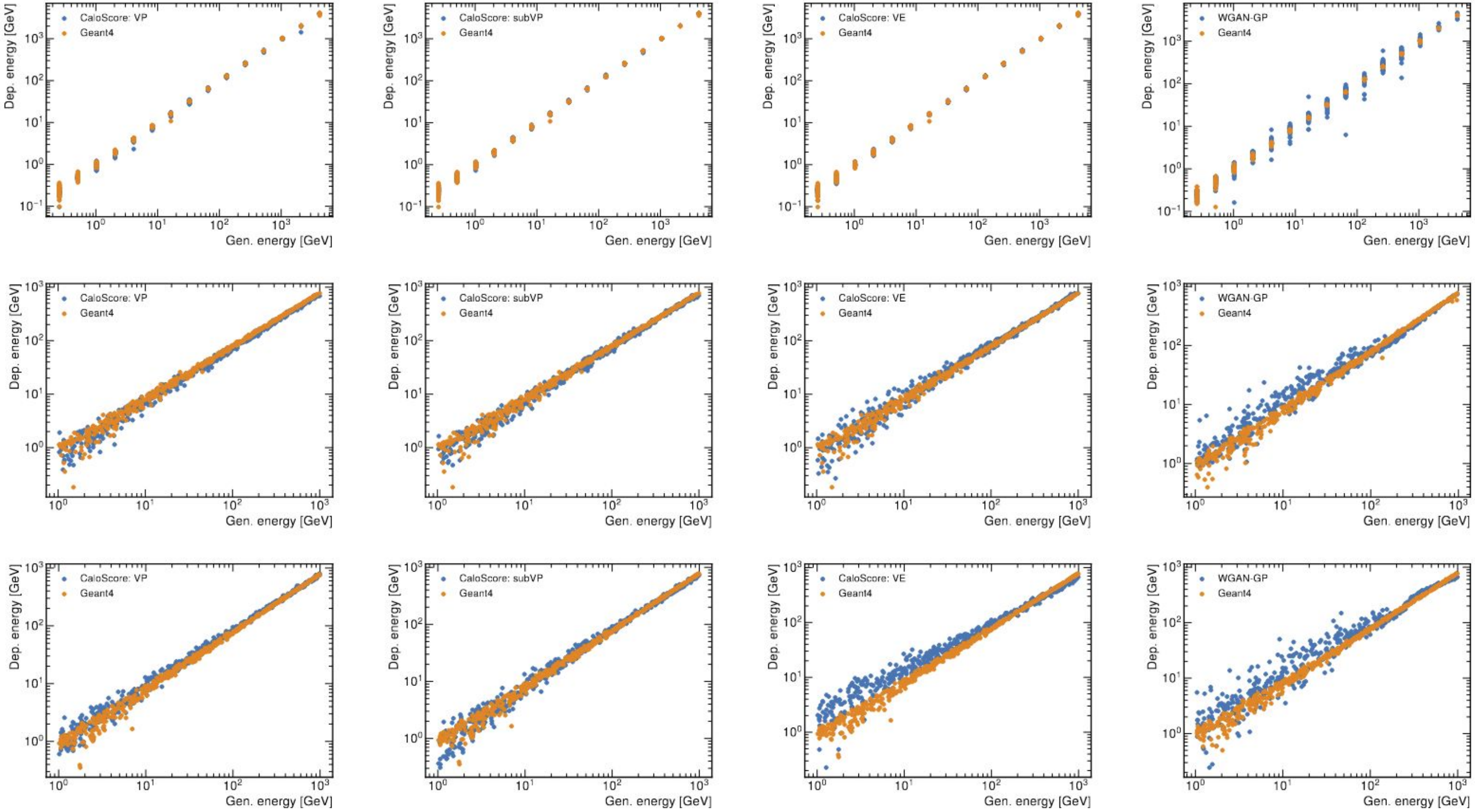




$$\sigma_i = \sqrt{\langle x_i^2 \rangle - \langle x_i \rangle^2},$$

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

- **Width of the particle shower** is particularly challenging to model
- **subVP** describes well all regions and datasets



Dataset 1

Dataset 2

Dataset 3

VP **subVP** **VE** **WGAN-GP**

Geant4

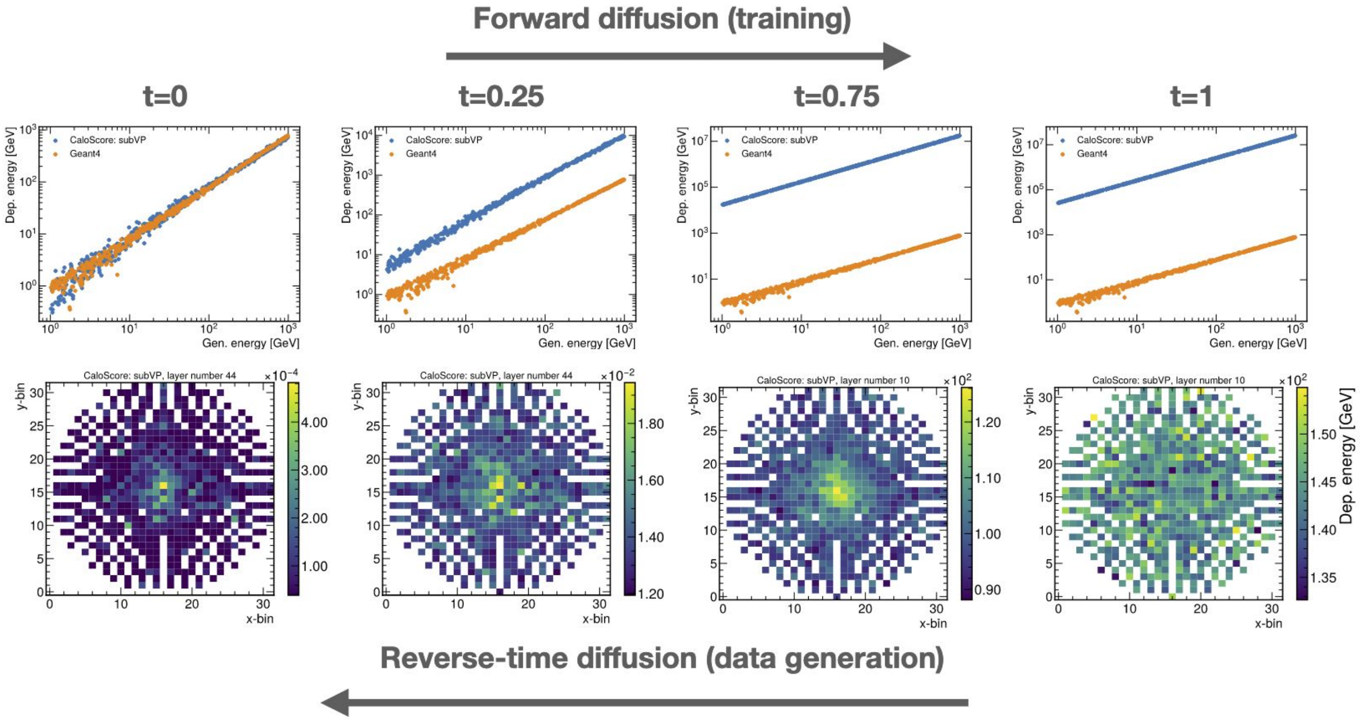
Generative models

Results

Dataset	N. of voxels	N. of weights	Time to 100 showers [s]		
			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	$\mathcal{O}(10^4)$

- Total number of trainable weights are more sensitive to the **model architecture** rather than the total number of dimensions
- Compared to the WGAN, CaloScore is **3-16 times slower**
- Compared to the full simulation, the generation time is **2-3 orders of magnitude faster**

Conclusion



Score-based generative models are an exciting new option for generative models

- https://scorebasedgenerativemodeling.github.io/

arXiv posting:

- https://arxiv.org/abs/2206.11898

You can take a look at the implementation: https://github.com/ViniciusMikuni/CaloScore

Backup

Likelihood estimation?

- Data generation can also be achieved by solving the **associated ODE**
  - Often leads to **worse** samples compared to Langevin dynamics generation
- On the other hand, we can also use the deterministic ODE recover the **data density!**

SDE

$$d\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})]dt + g(t)d\mathbf{w}$$

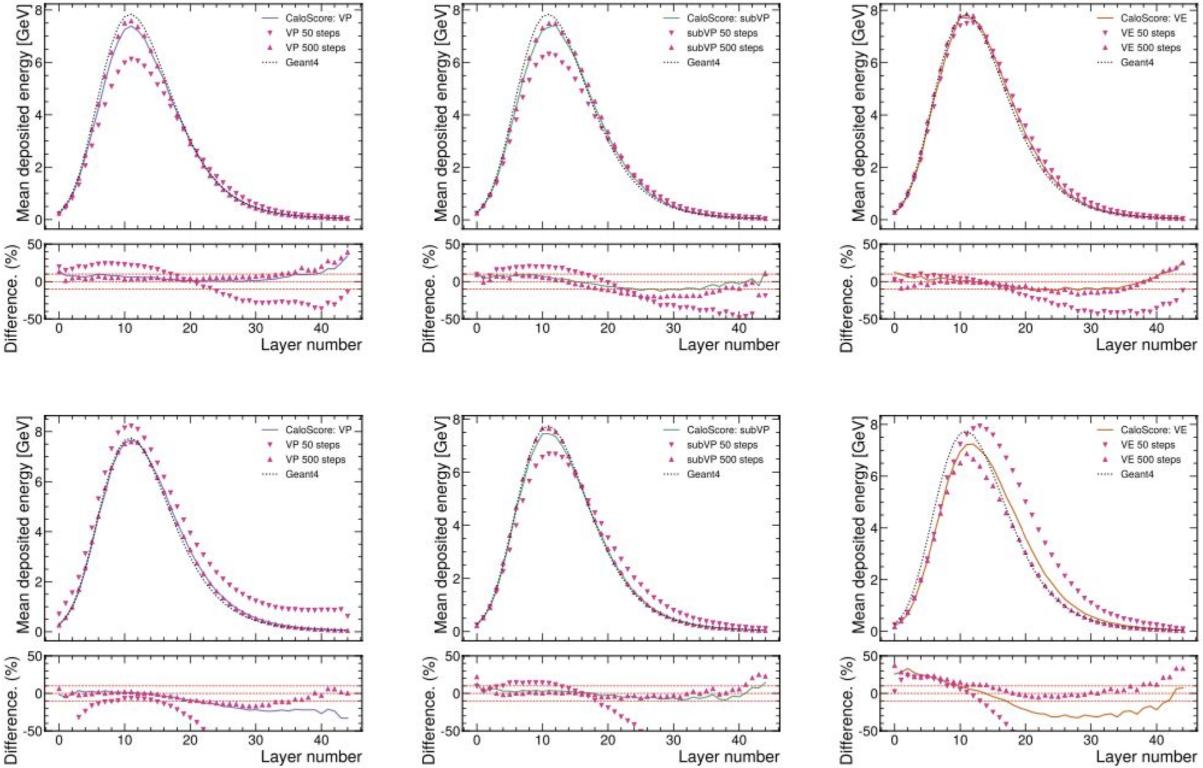
ODE

$$d\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - \frac{1}{2}g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right]dt$$

$$d\mathbf{x} = \tilde{\mathbf{f}}(\mathbf{x}, t)dt,$$

$$\log p_0(\mathbf{x}(0)) = \log p_T(\mathbf{x}(T)) + \int_0^T \nabla \cdot \tilde{\mathbf{f}}_{\boldsymbol{\theta}}(\mathbf{x}(t), t)dt,$$

Results



- Sample quality can be improved if more time steps are used, however the time increase for sampling is the bottleneck

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