Score based generative modeling

Geometric Data Analysis Project Report

Kshitij Ambilduke
Nemanja Vujadinovic
Bruny Soutarson
ENS Paris-Saclay, Gif-sur-Yvette

Abstract

In this project, we implemented the Score based generative modeling [12] framework, highlighting some of the shortcomings particularly related to the choice of the variance schedule. In particular, we explore using linear and square-root variance schedules along with the original geometric schedule. Besides this, we also run a proof-of-concept experiment on the usage of continuous variances instead of using a discrete sequence. To mitigate the computational limitations of the framework, we also find that using score estimation on a latent space results in faster training and inference and to this end, we train a model on the PCA latent space and show the benefits of training in the latent space rather than training on the raw data representation. All our training and inference code is available on Github.[1]

1 Introduction

Score matching, initially conceptualized to estimate non-normalized probabilistic models [6], has gained a lot of traction recently due to its application to image generation, particularly in the context of Diffusion models [4]. Originally, score matching was formalized by optimizing the expected mean square error between the true score $\nabla_x \log p_{data}(x)$ and the estimated score $s_\theta(x)$.

$$J_1(\theta) = \frac{1}{2}\mathbb{E}_{x\sim p_{data}}\left[\|\nabla_x \log p_{data}(x) - s_\theta(x)\|_2^2\right]$$

Using the property of partial derivatives [6], it was shown that optimizing the $J_1(\theta)$ objective is equivalent to optimizing the following objective $J_2(\theta)$.

$$J_2(\theta) = \mathbb{E}_{x\sim p_{data}}\left[\mathrm{tr}(\nabla_x s_\theta(x)) + \|s_\theta(x)\|_2^2\right]$$

Since calculating the trace of the Jacobian of $s_\theta(x)$ is computationally expensive, subsequently, several works tried to work around this by either taking projections of the score functions onto random vectors [13] or estimating scores of images perturbed with gaussian noise [14], which results in an objective ($J_3(\theta)$) which can be practically implemented without requiring to compute the Jacobian. The later objective, mentioned below, has gained a lot of traction in the image generation community due to its connection to Diffusion models [4].

$$J_3(\theta) = \frac{1}{2}\mathbb{E}_{\tilde{x}\sim q_\sigma(\tilde{x}|x)p_{data}(x)}\left[\|s_\theta(\tilde{x}) - \nabla_{\tilde{x}} \log q_\sigma(\tilde{x}|x)\|_2^2\right]$$

Note that in the objective $J_3(\theta)$, $q_\sigma(\tilde{x}|x)$ denotes the conditional distribution of the noisy image given the original image for the noise level σ. For example, if we consider zero mean additive gaussian noise ($\epsilon \sim \mathcal{N}(0, \sigma^2 I)$), then, $\tilde{x} = x + \epsilon$ and hence, $\tilde{x} \sim \mathcal{N}(x, \sigma^2 I) = q_\sigma(\tilde{x}|x)$. In this particular case, where we consider zero mean additive gaussian noise, the score $\nabla_{\tilde{x}} \log q_\sigma(\tilde{x}|x)$ takes the form $\frac{x-\tilde{x}}{\sigma^2} = -\frac{\epsilon}{\sigma^2}$. This loss function can be very efficiently implemented by basically augmenting the original image with a specific noise level and training a neural network (score network) to estimate the score of this perturbed image after which $J_3(\theta)$ can be directly optimized by backpropagation.

[1]https://github.com/Kshitij-Ambilduke/ScoreBasedGenerativeModels

Once we have trained the score network, we can utilize the Langevin dynamics to generate new datapoints starting from a random point in the data space. The intuition behind Langevin dynamics is to start from any random point in the data space and then gradually move in the direction of the score to end up in a high likelihood state which corresponds to a plausible datapoint.

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \frac{\alpha}{2}\nabla_\mathbf{x} \log p(\mathbf{x}_t) + \sqrt{\alpha}\,\mathbf{z}_t \tag{1}$$

Here, $z_t \sim \mathcal{N}(0, I)$ and $\alpha > 0$ is the step size which depends on the noise level of x_t. Note that during inference, the true score $\nabla_x \log p(x_t)$ is replaced by the score estimated by our score network, $s_\theta(x_t)$, for the current data point x_t.

Direct application of score matching is hindered because the standard $J_2(\theta)$ objective fails on lower-dimensional data manifolds, and score estimation remains inaccurate in unobserved, low-density regions. To resolve this, the network is trained on noise-perturbed images across multiple scales [12], effectively expanding the support of the distribution to cover the whole space. Correspondingly, the sampling algorithm employs Annealed Langevin Dynamics, which utilizes large initial steps to traverse low-density gaps and gradually decreases the step size to converge faithfully into high-density regions.

The remainder of this report is organized as follows. Section 2 analyzes the impact of variance scheduling on model performance, comparing linear and square-root schedules with the traditional geometric schedule, and introduces a toy experiment for a continuous variance framework. In Section 3, we investigate the efficiency of performing score estimation in a lower-dimensional latent space. Section 4 details our implementation specifics, including the model architectures and datasets used. Section 5 presents our experimental results, comparing the efficacy of the different variance schedules and the latent space approach. Finally, Sections 6 and 7 provides concluding remarks and discusses the limitations of score based generative modeling.

2 Variance Schedule

One of the main decisions taken during training the score network is about noise levels. Although using geometric sequence as noise levels seems to work well [12], recent works trying to replicate these results have shown that linear schedule is also effective [10].

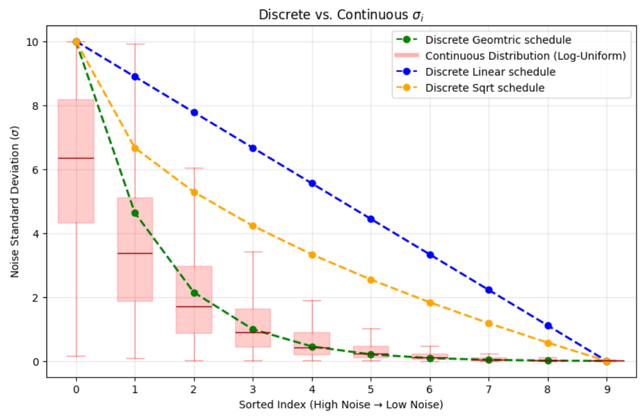

Figure 1: Continuous and discrete noise schedules.

In addition to these, using other sets of variances for training is relatively underexplored. Furthermore, we hypothesize that training on discrete levels limits the performance of the score network only to these specific noise levels. This means that even during sampling we have to use the same set of variances. On the other hand, if we train a model to deal with any noise level, we can perhaps tune the variances during sampling so that much fewer total steps are required to achieve the same quality of images. Hence, in the following sections, we examine two settings related to the variance schedule, first we try out different discrete variance schedules, and secondly, we try out the continuous alternative on a mixture-of-gaussian toy dataset.

2.1 Discrete variance schedule

In the original setup [12], the score-network is trained on discrete geometric sequence of noise levels $\{\sigma_i\}_{i=1}^{L}$. These noise levels follow a decreasing geometric progression that satisfies $\frac{\sigma_1}{\sigma_2} = \cdots = \frac{\sigma_{L-1}}{\sigma_L}$. A single noise level can therefore be written as:

$$\sigma_i = \sigma_1 \left(\frac{\sigma_L}{\sigma_1}\right)^{(i-1/L-1)}, \qquad i = 1, \ldots, L. \tag{2}$$

During the sampling phase (Eq. 1), the step-size α is directly tied to this variance schedule. Specifically, [12] proposes using $\alpha_i = \epsilon \cdot \sigma_i^2/\sigma_L^2$ for the i-th noise level. The intuition behind this schedule relies on the relative position of the image and modes of the data distribution. When initializing sampling at a random point in the high-dimensional space, the sample is likely to fall within the support of the Gaussian with the largest variance (σ_1), but far from the data mode. Consequently, a larger σ_i dictates a larger step size α_i, allowing the sampling algorithm to traverse low-density regions quickly. As the process iterates and the sampler approaches the mode, σ_i (and thus the step size) is annealed to refine the sample.

However, as argued in [10], geometric annealing may decrease the noise too aggressively, which can potentially emphasize fine-scale details before global features are captured. Inspired by their findings, we also examine alternative scheduling methods beyond the original geometric one.

We first model a linear discrete schedule as:

$$\sigma_i = \sigma_1 + (\sigma_L - \sigma_1)\,\frac{i-1}{L-1}, \qquad i = 1, \ldots, L.$$

This schedule distributes noise levels uniformly between σ_1 and σ_L, thereby assigning more levels to the medium- and high-noise regions. Our hypothesis is that this can allow the sampler to learn global structure more effectively. In addition to the linear schedule, we also investigate a square-root schedule, defined as:

$$\sigma_i = \sigma_1 + (\sigma_L - \sigma_1)\sqrt{\frac{i-1}{L-1}}, \qquad i = 1, \ldots, L.$$

As shown in Fig. 1, this schedule decays more slowly than the geometric schedule in the high-noise regions, yet also decreases faster than the linear schedule in low-noise regions. Therefore, our hypothesis for square-root schedule is that it may combine the benefits of both previous schedules and propose a balance between capturing global features and refining smaller details.

2.2 Continuous variance schedule

A significant limitation of the discrete geometric method described in Section 2.1 is the coupling between training and sampling hyperparameters. Noise level is intrinsically a continuous parameter. Relying on a discrete set requires the sampling procedure to adhere to the specific σ_i's seen during training. A continuous approach where we condition the model on continuous noise levels, would decouple these phases, allowing for the selection of an optimal, arbitrary variance schedule during sampling, which could potentially be with fewer steps than the total discrete levels (L), without the need to retrain the model.

Building on this insight, we evaluate the continuous noise conditioning strategy by sampling σ from a log-uniform distribution:

$$\sigma = \sigma_1 \cdot \left(\frac{\sigma_L}{\sigma_1}\right)^{u}, \quad \text{where } u \sim \mathcal{U}(0,1). \tag{3}$$

This distribution is chosen specifically to approximate the density of the discrete geometric progression used in the original work [12]. The comparison between discrete schedule with geometric rule and continuous schedule with log-linear rule can be seen in Figure 1. The green markers denote the fixed geometric sequence used in the discrete formulation whereas the red box plots (aggregated over 1,000 trials) illustrate the distribution of sorted noise levels sampled from the continuous log-uniform distribution. On linear scale, the steep curve highlights that both strategies concentrate the majority of the model's capacity on lower noise levels ($\sigma \approx 0$), where fine-grained image details are refined. The continuous approach covers the intervals between the discrete steps, allowing for arbitrary variance schedules during inference.

3 Latent generation

While score-based generative modeling was originally formulated in the data space, recent studies have demonstrated several advantages of performing generation in a latent space instead [11]. Motivated by this, we conduct an additional experiment in which data are generated in the PCA latent space rather than directly in the image space.

Given an input image x, we first obtain its latent representation via a linear projection $\tilde{x} = \tilde{U}x$ where $\tilde{U} \in \mathbb{R}^{l \times d}$ contains the l

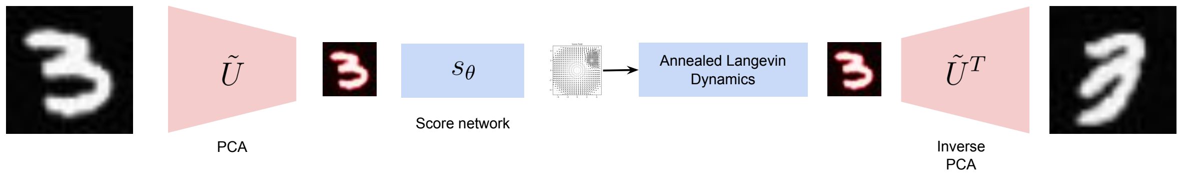

Figure 2: Illustration of how we train and sample from the Latent PCA model.

eigenvectors of the PCA projection matrix U corresponding to the largest eigenvalues. We then treat $\tilde{x}$ as the data representation and apply the same score-based learning procedure as before, but now entirely in the latent space. At sampling time, after training a model to estimate the score function over these latent variables, we generate a new latent sample $\hat{\tilde{x}}$ starting from pure Gaussian noise using the annealed Langevin dynamics. Finally, the generated latent is mapped back to the data space using the inverse PCA transform $\hat{x} = \tilde{U}^\top \hat{\tilde{x}}$ to obtain a synthesized image $\hat{x}$. The overall framework is shown in Figure 2.

It is worth noting that this framework provides a general approach for latent-space generation. In this work, we employ a simple linear model as encoder and decoder. However, the same methodology readily extends to nonlinear autoencoder-based architectures.

Generating in the latent space offers several important advantages over direct data-space generation. First, since latent representations typically lie in a significantly lower-dimensional space than the original data, both training and sampling become computationally more efficient, resulting in substantial speedups. Second, from a geometric perspective, the latent space occupies a substantially smaller volume than the original data space. Assuming the encoder is injective (or approximately injective), the induced latent distribution concentrates probability mass in a more compact region with fewer low-density areas. As a result, score estimation in latent space is typically more stable and accurate than in the full data space, leading to improved generative performance.

4 Implementation details

4.1 Model and Datasets

For the score-network, we use RefineNet [9] as our backbone. Specifically, following the original work [12], we replace the instance norm layers in RefineNet with a modified Conditional instance norm layers in-order to input the noise level of the images. Note that for image generation we rely on discrete schedule and hence the noise level can be input using its corresponding index in the schedule sequence and a trainable embedding matrix. Besides this, for the experiment on latent generation, we fix the size of the latent to 64 resulting into a 8×8 latent image. This corresponds to roughly 91.61% of the total magnitude of the eigenvalues.

Owing to computational limitations restricted to the free T4 GPU on `Google Colab`, we limit our experiments to MNIST [2] and FashionMNIST [15] datasets. We further restrict the number of classes for the FashionMNIST dataset to 3 by using only the images of {T-shirt, Trousers, Pullover}. Inspired by the success of this small-scale experiment, we then extend to all classes of the MNIST dataset. We train our model using the Adam optimizer [7] on a total of 5 epochs using a batch size of 32. As mentioned before, we use the discrete geometric schedule for noise with $L = 10$ levels between $\sigma_1 = 1$ and $\sigma_L = 0.1$. Note that we used the same setup while testing the different discrete noise schedules.

4.2 Continuous noise schedule

To quantify the advantage of continuous schedule against discrete, we perform a toy experiment by generating scores from a 2D mixture of Gaussians dataset. We use the following density function to generate our data with 3 modes

$$p_{\text{data}}(\mathbf{x}) = \frac{1}{5}\mathcal{N}\left(\mathbf{x}; \begin{bmatrix}-5\\-5\end{bmatrix}, \mathbf{I}\right) + \frac{3}{5}\mathcal{N}\left(\mathbf{x}; \begin{bmatrix}5\\5\end{bmatrix}, \mathbf{I}\right) + \frac{1}{5}\mathcal{N}\left(\mathbf{x}; \begin{bmatrix}5\\-5\end{bmatrix}, \mathbf{I}\right)$$

For the score network in this 2D toy experiment, we employ a lightweight Multi-Layer Perceptron. Specifically, the network consists of three fully connected layers with Softplus activations. To accommodate the continuous noise schedule, we condition the network by concatenating the input coordinates with the logarithm of the noise level. This architecture is inspired by the toy experiments in the original work [12].

5 Results

5.1 Discrete schedules

To better understand the effect of the variance schedule on the denoising process, we begin by qualitatively examining the sampling results across noise levels for three discrete schedules introduced in Section 2.1. By visualizing these results, we aim to show how quickly coarse structure emerges, how reliably global shape is captured, and how much capacity remains for refining smaller details. As shown in Figure 3, the geometric schedule produces a recognizable structure of the object already at an early stage (around L=3). The remaining levels are spent for local refinement, but differ only minimally from one another. In contrast, the linear schedule shows the slowest transition from noise to meaningful structure. It spends most of the steps (up to L=7, L=8) on recovering the global structure and only a few for fine-tuning the details. Finally, the square-root schedule results in a more even progression and spends approximately the same number of steps on refining both the global and local structure. These observations support our initial hypothesis

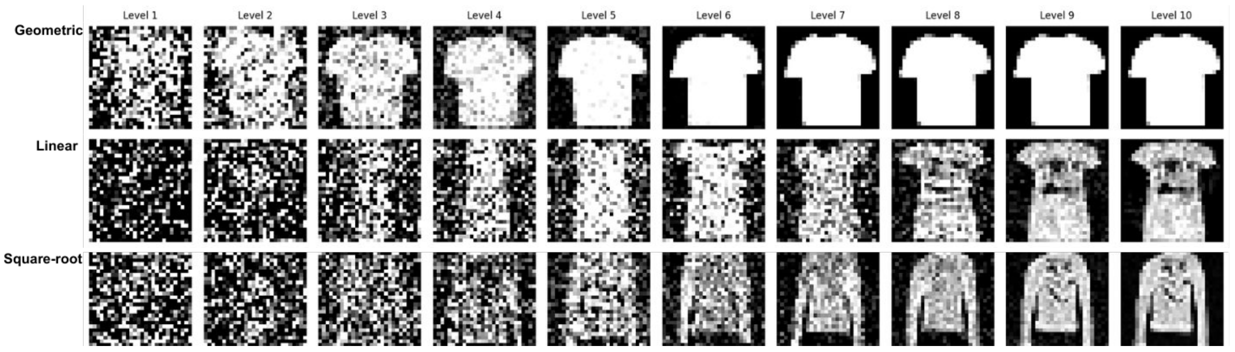

Figure 3: Comparison of images generated at each noise level for geometric, linear, and square-root variance schedules.

on how different schedules emphasize different phases of the denoising process. In particular, we have shown that the square-root schedule offers the most balanced progression, while the geometric and linear schedules are biased toward the earliest and latest stages of sampling, respectively.

Next, we evaluate and compare the final samples produced under each scheduling method. From Figure 7, we see that the geometric schedule produces the cleanest results. The generated objects are recognizable and stable, yet they appear slightly over-smoothed. This behavior is likely influenced by the simplicity of the dataset together with the high proportion of low noise levels used during training and sampling. On the other hand, the linear schedule performs reasonably well and generates diverse and recognizable shapes. The global structure, which was our main interest, is captured clearly and remains consistent across samples. The results are, however, noticeably noisier, which can (inversely) be attributed to the reduced number of noise levels in the lower-noise regions. Lastly, the square-root schedule performs the worst and often produces fragmented or irregular shapes. Our initial assumptions regarding its balanced behavior therefore do not hold in this setting. This may be due to its intermediate assignment of noise levels, or simply due to the limited amount of training we were able to perform, which we discuss in the following paragraph.

We have nevertheless shown that an alternative schedule can produce useful results, even though the geometric schedule remains the strongest baseline in our experiments. Unlike reports in earlier work [10], we are not able to claim superiority of any schedule over the geometric method because our computational resources limited the scale of our analysis and experiments. Both the linear and square-root schedules still provided valuable insights in the evaluation of intermediate sampling steps, while the linear schedule also produced encouraging final samples.

5.2 Continuous schedule

To empirically validate the advantages of continuous noise conditioning over the discrete baseline, we propose two evaluation protocols. First, we examine the generalization capability of the score network by visualizing the score vector field at a noise level σ that does not coincide with any step in the fixed geometric series used during discrete training. By visualizing the score field for particular values of noise and comparing it against the ground truth score field, we can gauge into the generalization capabilities of the score network trained on discrete sequence for unseen noise levels. As seen in Figure 4, for high noise levels, as expected, the ground truth score vector field predominantly points towards the largest mode (the Gaussian at $[5, 5]^T$ with weight $3/5$). A closer inspection of the region immediately above the $[5, -5]^T$ mode reveals a significant discrepancy. The discrete schedule exhibits sparse score predictions in this area. In contrast, the continuous model maintains a vector field coherent with the ground truth. This observation reinforces our belief that training on a continuous schedule enables the model to effectively interpolate and generalize to intermediate noise levels that were not explicitly seen during training.

For the second evaluation setup, we test the robustness of the generation process by restricting the sampling trajectory to a subpar number of total steps. For the discrete case, this constraint is implemented by reducing the number of Langevin steps per σ_i level. Conversely, for the continuous approach, we sample the specific sequence of noise levels from the proposed log-linear distribution and iterate from noise to the data manifold. This is achieved by linearly interpolating in the log space. We visualize the resulting density via heatmaps of 1,000 data points, averaged over 5 independent sampling runs in Figure 5. The qualitative comparison demonstrates that samples generated via the discrete method exhibit a significantly larger spread around the modes. The continuous case, however, produces tighter clusters, leading us to conclude that continuous sigma generation results in samples that converge more faithfully to the expected data distribution, even under a constrained computational budget.

5.3 Latent generation

We evaluate the effectiveness of the latent space approach by first comparing the convergence rates of the two methods. We observe that latent generation yields plausible image structures as early as the first training epoch. In stark contrast, the full data generation model requires training until at least the third epoch to produce recognizable fashion items. This accelerated learning suggests that the topology of the lower-dimensional PCA subspace allows the score network to generate much more commendable score fields at each noise level which leads to, during sampling, capturing the underlying data distribution significantly faster than in the high-dimensional pixel space.

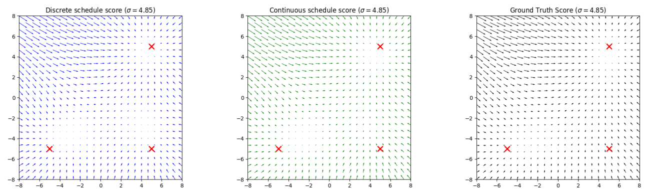

Figure 4: We clearly see a patch near the mode [5,-5] in discrete case but not in continuous. This is because sigma=4.85 was not in the geometric sequence.

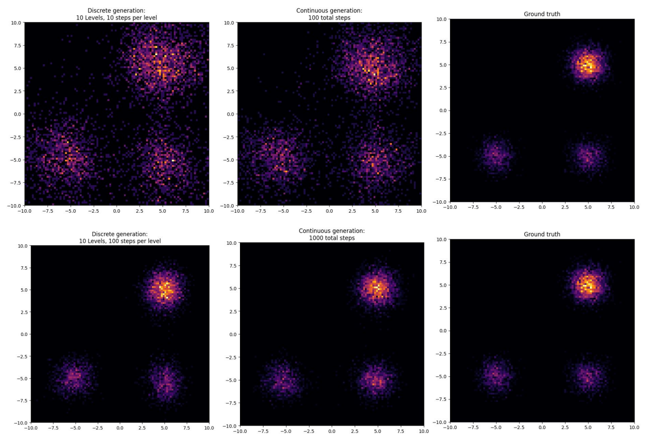

Figure 5: Heat map of 5 times generations of 1000 samples. We see clearly that discrete generation in more spreadout than continuous when we decrease the number of sampling steps whereas both discrete and continuous match performance on increasing the number of steps.

Furthermore, this efficiency extends to the inference phase. We find that the latent generative process is capable of producing plausible samples with fewer sampling iterations. Specifically, recognizable features begin to emerge as early as the second or third noise level step in the reverse dynamics (corresponding to approximately 200 to 300 Langevin iterates). We attribute this robustness to the nature of the decoding step. Since the inverse PCA projection maps the latent vector back to the data space using a linear combination of principal components, it effectively acts as a regularizer. Even approximate latent representations, which might still contain residual noise, are projected onto the manifold of valid images. We hypothesize that employing a stronger, non-linear decoding architecture, such as a deep Autoencoder, would further enhance this error-correcting capability and result in even higher fidelity samples at earlier steps.

Finally, we quantify the computational advantages in terms of wall-clock time. Due to the reduced dimensionality of the input (8×8 latent vs. 28×28 original), the training speed for latent generation is markedly faster, clocking in at approximately 210 seconds per epoch, compared to 617 seconds for full data generation. Similarly, the sampling throughput sees a substantial boost, with latent generation requiring only 27 seconds to synthesize a batch of samples, whereas the complete data generation process takes 68 seconds. Note that the duration for training is clocked in T4 GPU whereas the duration for sampling is clocked on a locally on a NVIDIA GTX-1650 GPU. These results confirm that projecting to a lower-dimensional latent space not only stabilizes training but also drastically reduces the computational budget required for both learning and inference.

Encouraged by the strong qualitative performance observed after only a few training epochs (see Appendix 8), we extended our analysis to the full MNIST dataset and trained both the latent-space and data-space models for a substantially longer schedule of 200 epochs, corresponding to approximately 100k Langevin iterations. For this experiment, we also refined the choice of latent dimensionality by selecting the number of PCA components via the elbow method; see Appendix 8.1 for details.

The resulting quantitative scores are reported in Table 1 and Table 2. These results further support our initial observations: the PCA latent model maintains a clear computational advantage, with significantly reduced training and sampling times compared to its data-space counterpart. However, the performance gap in terms of sample quality becomes less pronounced at this larger training scale. While latent generation continues to benefit from faster convergence and more stable score estimation, the full data model progressively catches up in visual fidelity as training proceeds.

Overall, these findings reinforce the value of latent-space generation as an efficient alternative to direct score estimation in pixel space. Yet they also highlight an important limitation: as training time increases, improvements in data-space modeling eventually diminish the relative qualitative advantage of the latent approach, even though the computational savings remain substantial.

Sample	Inception score
MNIST Dataset	2.466 ± 0.04
Raw data generations	2.351 ± 0.04
Latent generations	2.232 ± 0.012

Table 1: Inception scores for MNIST and generated samples

Sample A	Sample B	FID Score
MNIST Dataset	Raw data generations	41.34
Recon. MNIST Dataset	Latent generations	52.08
MNIST Dataset	Latent generations	63.49

Table 2: FID scores on MNIST

6 Conclusion

In this report, we present the main results from [12] concerning score-based generative modeling and emphasize that they are qualitatively reproducible, yielding visually plausible images on MNIST and FashionMNIST datasets. We present a few additional experiments particularly pertaining to (a) the arbitrary variance schedule utilized in the original work [12] and (b) score generation in latent space.

When it comes to discrete variance schedules, we observe that while the geometric schedule produces the cleanest results, alternative schedules also reveal interesting characteristics. The linear schedule captures global structure effectively, but produces grainier final samples, mainly due to insufficient low-noise refinement steps [10]. The square-root schedule however, despite our hypothesis of balanced progression, underperformed compared to both geometric and linear schedule. Building on these findings, we investigate continuous variance conditioning on a 2D Gaussian mixture toy experiment. This experiment showed clear advantages in generalizing to unseen noise levels and improved generated sample quality under constrained sampling budget. Furthermore, the visualization of the score vector fields confirms that continuous training enables better interpolation between noise levels.

Latent generation yielded substantial computational improvements over raw data generation. The former generated recognizable images after just one training epoch (compared to three for raw data), training time decreased from 617 to 210 seconds per epoch, and sampling time decreased from 68 to 27 seconds per image. While we used a simple linear PCA encoder-decoder model, our results suggest that employing more expressive non-linear architectures could improve quality of the generated image. Due to computational constraints we resorted to simplified datasets and we acknowledge that our findings require validation on larger and possibly more complex datasets.

7 Limitations

While score-based generative modeling has been one of the most influential frameworks for image generation, its application is fundamentally limited to data which adhere to Euclidean geometry. The formulation of both training and sampling relies on flat vector space operations, making it non-trivial to extend to other geometries, such as Riemannian manifolds. While this gap has started to gain attention in the community with works on Riemannian Score-Based Generative Models [1, 5], these methods have not yet scaled as effectively as their Euclidean counterparts.

Furthermore, compared to other established training paradigms like Variational Autoencoders (VAEs) [8] and Generative Adversarial Networks (GANs) [3], score-based models rely on an iterative sampling procedure. This makes them inherently slower than the aforementioned methods, which typically generate new samples in a single forward pass. This computational bottleneck poses significant challenges for real-time applications and for scaling the synthesis of synthetic training data where the volume of required images is large.

References

[1] Valentin De Bortoli, Emile Mathieu, Michael Hutchinson, James Thornton, Yee Whye Teh, and Arnaud Doucet. 2022. Riemannian Score-Based Generative Modelling. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 2406–2422. https://proceedings.neurips.cc/paper_files/paper/2022/file/105112d52254f86d5854f3da734a52b4-Paper-Conference.pdf

[2] Li Deng. 2012. The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]. *IEEE Signal Processing Magazine* 29, 6 (2012), 141–142. doi:10.1109/MSP.2012.2211477

[3] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger (Eds.), Vol. 27. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2014/file/f033ed80deb0234979a61f95710dbe25-Paper.pdf

[4] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS '20)*. Curran Associates Inc., Red Hook, NY, USA, Article 574, 12 pages.

[5] Chin-Wei Huang, Milad Aghajohari, Joey Bose, Prakash Panangaden, and Aaron C Courville. 2022. Riemannian Diffusion Models. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 2750–2761. https://proceedings.neurips.cc/paper_files/paper/2022/file/123d3e814e257e0781e5d328232ead9b-Paper-Conference.pdf

[6] Aapo Hyvärinen. 2005. Estimation of Non-Normalized Statistical Models by Score Matching. *Journal of Machine Learning Research* 6, 24 (2005), 695–709. http://jmlr.org/papers/v6/hyvarinen05a.html

[7] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG] https://arxiv.org/abs/1412.6980

[8] Diederik P Kingma and Max Welling. 2022. Auto-Encoding Variational Bayes. arXiv:1312.6114 [stat.ML] https://arxiv.org/abs/1312.6114

[9] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. 2017. RefineNet: Multi-path Refinement Networks for High-Resolution Semantic Segmentation . In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 5168–5177. doi:10.1109/CVPR.2017.549

[10] Antonio Matosevic, Eliisabet Hein, and Francesco Nuzzo. 2020. Reproducibility Challenge – Generative Modeling by Estimating Gradients of the Data Distribution. In *NeurIPS 2019 Reproducibility Challenge*. https://openreview.net/forum?id=SkxCSTqG6H

[11] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv:2112.10752 [cs.CV] https://arxiv.org/abs/2112.10752

[12] Yang Song and Stefano Ermon. 2019. Generative Modeling by Estimating Gradients of the Data Distribution. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf

[13] Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. 2019. Sliced Score Matching: A Scalable Approach to Density and Score Estimation. arXiv:1905.07088 [cs.LG] https://arxiv.org/abs/1905.07088

[14] Pascal Vincent. 2011. A connection between score matching and denoising autoencoders. *Neural Comput.* 23, 7 (July 2011), 1661–1674. doi:10.1162/NECO_a_00142

[15] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. arXiv:cs.LG/1708.07747 [cs.LG]

8 Appendix

8.1 PCA explained variance and elbow method

To motivate our choice of latent dimensionality on MNIST, we report in Figure 6 the cumulative explained variance of the PCA components computed on the training set. The curve displays a clear elbow at 91 components, after which the marginal gain in explained variance becomes negligible.

While 91 would be the PCA-optimal dimensionality, our score model requires the latent vector to be reshaped into a square grid. We therefore adopt 100 dimensions in practice, as it is the closest perfect square to 91.

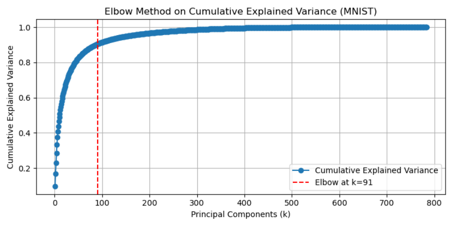

Figure 6: Cumulative variance of PCA components on MNIST.

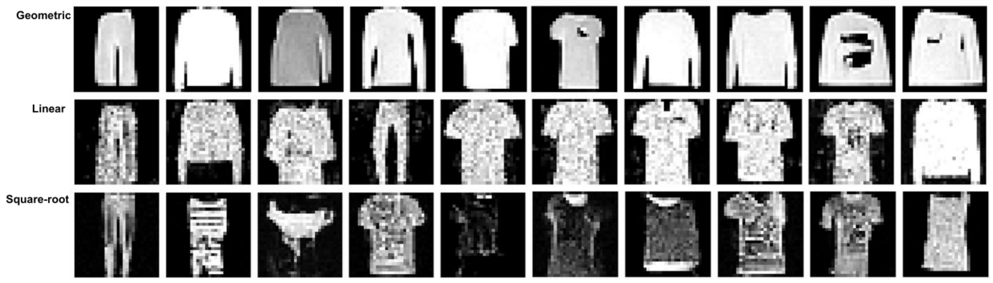

Figure 7: Comparison of final samples generated for geometric, linear, and square-root variance schedules.

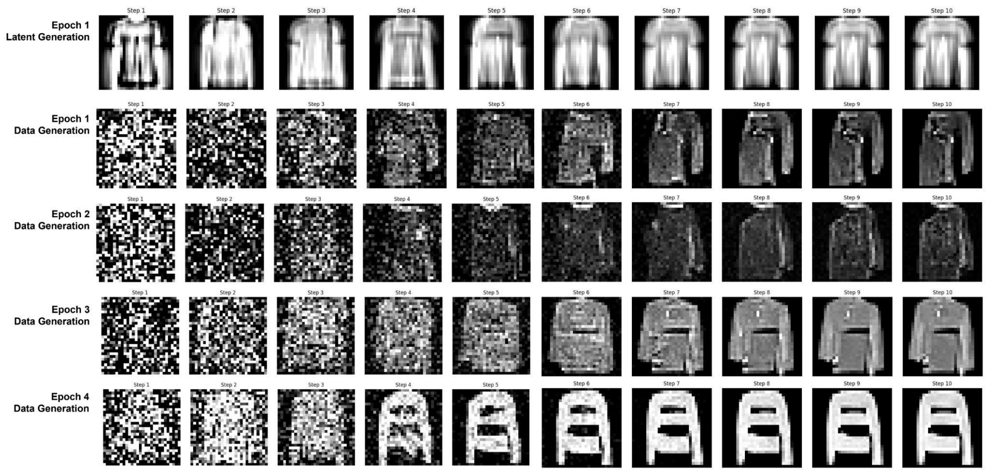

Figure 8: Comparison of images generated after training for one epoch by latent generation and data generation method.

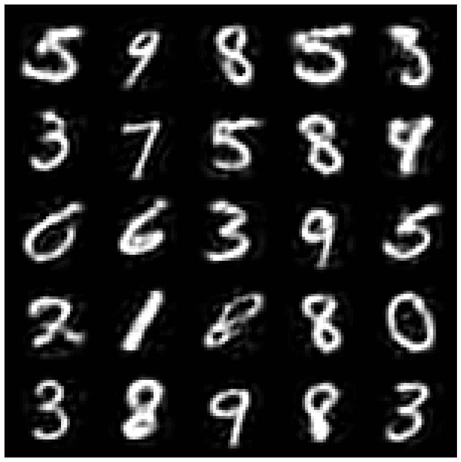

Figure 9: MNIST Digits generated by our latent PCA model.

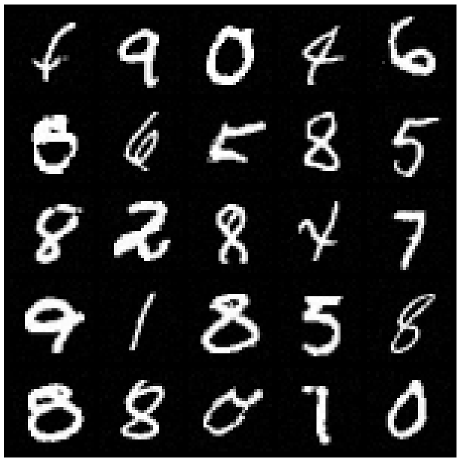

Figure 10: MNIST Digits generated by the classical approach.