Generative Modeling by Estimating Gradients of the Data Distribution

Hanna Benarroch
Mines Paris PSL
Paris, France
hanna.benarroch@etu.minesparis.psl.eu

Antoine Li
Mines Paris PSL
Paris, France
antoine.li@etu.minesparis.psl.eu

Virgile Richard
Mines Paris PSL
Paris, France
vrigile.richard@etu.minesparis.psl.eu

Abstract

Published in 2018, "Generative Modeling by Estimating the Gradients of the Data Distribution" by Y. Song et al. has played a seminal role in challenging the superiority of Generative Adversarial Networks (GANs) in generative modeling. It indeed introduced a new score-based method that would later prove scalable and lead to high-quality samples. In this paper, we first revisit the proposed method while providing new visualizations, additional context, proofs, and empirical evidence. We then highlight some intrinsic limitations of the method and discuss ways of mitigating them.

Contents

1 Introduction

"Generative Modeling by Estimating the Gradients of the Data Distribution" by Y. Song et al. introduced a new method for generative modeling [4]. Based on observations coming from an unknown data distribution, it aims to produce synthetic samples that follow the same underlying distribution. Target applications notably include generating artificial yet realistic images or sounds, and enabling data augmentation for self-supervised algorithms.

Two main approaches are used for generative modeling: adversarial training-based methods (and most notably, Generative Adversarial Networks) and likelihood-based models. However, both have intrinsic limitations.

1.1 Generative Adversarial Networks

GANs are made of two main components: a generator G that seeks to produce realistic images, and a discriminator D which tries to discriminate samples produced by G from genuine samples. These components are trained to minimize an objective of the form

$$\min_G \max_D \ \mathbb{E}_{x\sim p_{data}}\left[\ell_1(D(x))\right] + \mathbb{E}_{z\sim p_z}\left[\ell_2(D(G(z))\right]$$

where ℓ_1 and ℓ_2 are convex loss functions, e.g. $\ell_1 = \log$ and $\ell_2 = \log(1-\cdot)$, p_{data} is the data distribution and p_z a chosen noise distribution, e.g. $\mathcal{N}(0, \mathrm{I})$, that is fed into the generator.

Though sometimes leading to very capable models, this adversarial "minimax" objective makes the training of GANs unstable. Indeed, a common training technique is to sequentially train each component for a few iterations while freezing the other. This leads to an ever-changing objective function for each component and has empirically been found to induce training instabilities.

Moreover, as GANs' objectives depend on their respective discriminator, GANs require additional metrics to be compared and evaluated.

1.2 Likelihood-based models

On the other hand, Likelihood-based models aim to learn a probability distribution $p_{model}(\mathbf{x}) \approx p_{data}(\mathbf{x})$ by maximizing the likelihood of the observed data under the model. Likelihood-based models then sample from the learned distribution to generate new data points.

However, most likelihood-based models must rely on specialized architectures to construct normalized probability distributions (e.g. invertible maps for flow models) or utilize surrogate objectives, like the Evidence Lower Bound (ELBO) in Variational Autoencoders (which tend to produce blurry images) or contrastive divergence-based objectives in energy-based models[1], to enable effective training.

Some explicit likelihood-based models do not directly estimate $p_{data}(\mathbf{x})$ but rather its variations via $\nabla_{\mathbf{x}} \log p(\mathbf{x})$; we call them *Score-Based Generative Models*. This paper presents a new method

1. Energy-based models learn distributions of the form $p_\theta(\mathbf{x}) = \frac{e^{-E_\theta(\mathbf{x})}}{Z_\theta}$. They often resort to contrastive divergence to sample from p_θ and estimate the update $\nabla_\theta \log p_\theta(\mathbf{x}) = -\nabla_\theta E_\theta(\mathbf{x}) - \nabla_\theta \log Z_\theta = -\nabla_\theta E_\theta(\mathbf{x}) - \mathbb{E}_{\mathbf{x}\sim p_\theta}\left[\nabla_\theta E_\theta(\mathbf{x})\right]$.

for score-based generative modeling that circumvents the aforementioned limitations.

1.3 Two examples of Sampling Dynamics for Score-Based Generative models

Score-based generative models rely on score-based sampling methods like the Langevin or Hamilton Monte-Carlo dynamics.

1.3.1 Langevin dynamics. The Langevin dynamics is a discretization of Langevin's stochastic differential equation whose equilibrium distribution is the posterior distribution denoted by p in (1).

Given a fixed step size $\epsilon > 0$, and an initial value $\tilde{\mathbf{x}}_0 \sim \pi(x)$ with π being a prior distribution, the Langevin method recursively computes

$$\tilde{x}_t = \tilde{x}_{t-1} + \frac{\epsilon}{2}\nabla_x \log p(\tilde{x}_{t-1}) + \sqrt{\epsilon} z_t \tag{1}$$

where $z_t \sim \mathcal{N}(0, I)$. The distribution of $\tilde{\mathbf{x}}_T$ equals $p(\mathbf{x})$ when $\epsilon \to 0$ and $T \to \infty$ under some regularity conditions (see Appendix A).

Physical interpretation. In physics, the Langevin approach is a commonly used method for modeling the random motion (or brownian motion) of a particle in a fluid. The Langevin equation takes into account both deterministic forces, such as the force exerted by an external field, and random forces due to collisions with fluid molecules. The particle follows the equation below,

$$\frac{dx(t)}{dt} = -\frac{\epsilon}{2}\nabla U(x) + \sqrt{\epsilon k_B T}\, z(t) \tag{2}$$

where

- $-\frac{\epsilon}{2}\nabla U(x)$ is a force term influenced by the potential field $U(x)$ where the particle evolves and the mobility ϵ of the particle
- $\sqrt{\epsilon k_B T}\, z(t)$ is a diffusion term influenced by ϵ the mobility of the particle, T the temperature and $z(t)$ the thermal noise, a gaussian noise with mean 0 and variance 1.

Transposed into the probability theory, the idea behind the Langevin dynamics is that it gradually moves an initial random sample to regions with high probability by moving along the vector field of scores, and at each iteration, it injects noise in such a way that the trajectory will converge to the full posterior distribution, not only the regions with minimal energy.

1.3.2 Hamilton Monte Carlo dynamics. Hamiltonian Monte Carlo (HMC) is a method based on Hamiltonian dynamics, which introduces auxiliary momentum variables to explore the parameter space. In HMC, the target distribution $p(x)$ is augmented with a momentum variable p, forming a joint distribution:

$$p(x, p) \propto \exp\left(-H(x, p)\right),$$

where $H(x, p)$ is the *Hamiltonian*:

$$H(x, p) = U(x) + K(p).$$

Here, $U(x) = -\log p(x)$ is the potential energy, and $K(p) = \frac{1}{2}p^T M^{-1} p$ is the kinetic energy, where M is the mass matrix (often the identity matrix). The Hamiltonian dynamics evolve the system using the equations of motion:

$$\frac{dx}{dt} = \frac{\partial H}{\partial p}, \quad \frac{dp}{dt} = -\frac{\partial H}{\partial x}.$$

To approximate these continuous dynamics, HMC uses the *leapfrog integrator*, which discretizes the equations of motion with a step size ϵ. The discretized updates consist of three steps:

(1) Half-step momentum update: $p_t = p_{t-1} - \frac{\epsilon}{2}\nabla_x U(x_{t-1})$.
(2) Full-step position update: $x_t = x_{t-1} + \epsilon M^{-1} p_t$.
(3) Second half-step momentum update: $p_t = p_t - \frac{\epsilon}{2}\nabla_x U(x_t)$.

These updates simulate the Hamiltonian dynamics, ensuring the exploration of the probability distribution is efficient and approximately conserves the total Hamiltonian $H(x, p)$.

After integrating for a fixed number of steps, a Metropolis acceptance step ensures the new state is consistent with the target distribution. The acceptance probability is given by:

$$\alpha = \min\left(1, \exp\left(-\Delta H\right)\right)$$

where ΔH is the change in Hamiltonian, so that updates leading to too-large variations of H are likely to be rejected.

2 Score-learning

To sample under the Langevin or HMC dynamics, we need an estimation of the scores of the data distribution.

2.1 Conventional score-estimating objective

A neural network $s_\theta(\mathbf{x})$ is trained to estimate $\nabla_{\mathbf{x}} \log p(\mathbf{x})$. The objective is to minimize the following expectation:

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

In practice, the expectation is empirically evaluated using i.i.d data samples $\{\mathbf{x}_i\}_{i=1}^N \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$.

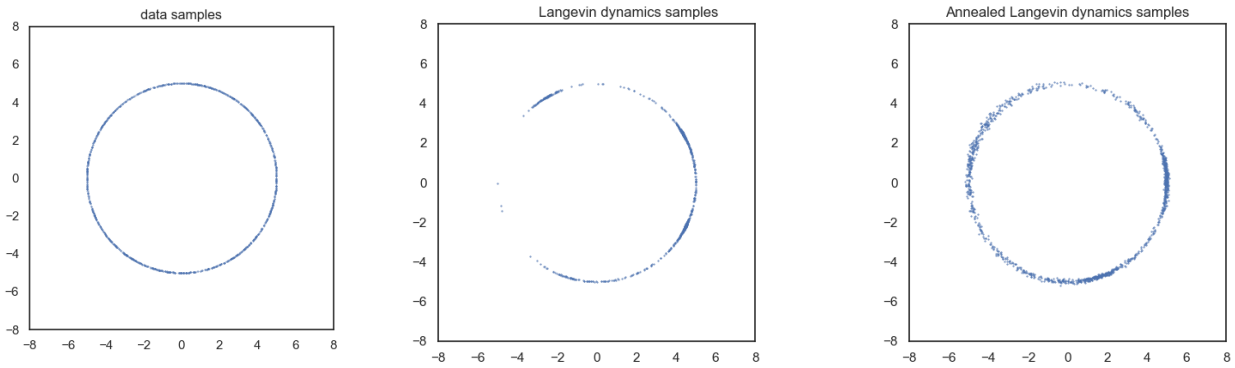
2.2 Challenges in learning the score

Learning the score of a data distribution finds two main issues.

2.2.1 Manifold hypothesis $p(\mathbf{x}) = 0$. Due to the manifold hypothesis, the support of the data distribution is not the whole space. Therefore, outside $\text{supp}(p)$, $\nabla_x \log p$ is not defined. This means that when a given $\mathbf{x}$ is not in the manifold, the learned score estimating $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ will not accurately point in the direction of the manifold. In particular, the initial points of the Langevin dynamics that are outside the manifold will not converge to the real data distribution (see an example in Figure 1).

2.2.2 Low-density regions $p(\mathbf{x}) \approx 0$. Another issue is that in low-density regions (where $p(\mathbf{x}) \approx 0$), we will likely not have enough data points in the $\{\mathbf{x}_i\}_{i=1}^N$ to correctly estimate the score $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ (see an example in Figure 2).

The negative effect of this can be seen in a phenomenon called *slow mixing*. When the data distribution possesses multiple modes with disjoint supports or connected with very low density regions, the Langevin dynamics will not recover the relative weights of the modes (see an example in Figure 3). Intuitively, given that we can generate faithful samples from the Langevin dynamics only when the initial sample belongs to the support of a mode (see 2.2.1), the proportion of samples ending up in one of the modes will only depend on their initialization, which is random (see an example in Figure 3).

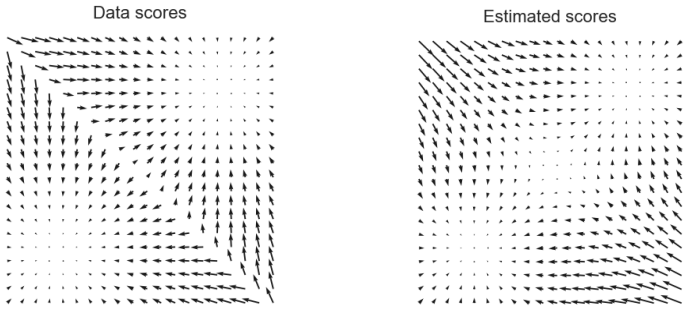


(a) Exact sampling of a circle, an example of a data distribution in $\mathbb{R}^2$ with a 1-dimensional manifold.

(b) Sampling using Langevin dynamics with naively estimated scores.

(c) Sampling using annealed Langevin dynamics.

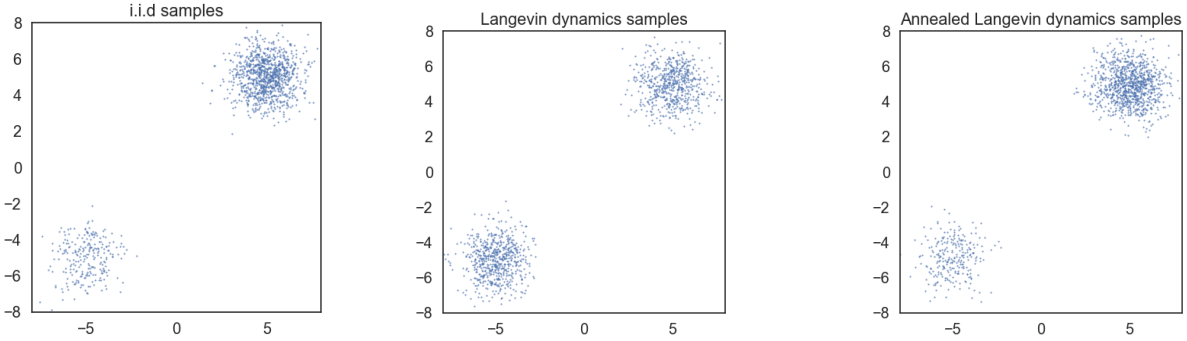
Figure 1: Comparison of the Langevin dynamics and the annealed Langevin dynamics for a distribution with a manifold (a circle in $\mathbb{R}^2$). The conventional Langevin dynamics does not converge to the actual data distribution whereas the new method introduced (annealed Langevin dynamics) recovers the full distribution.



(a) Exact scores $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ of two gaussians with weights 0.2 and 0.8 respectively.

(b) Estimated scores $s_\theta(\mathbf{x})$ with a simple score matching network.

Figure 2: Comparison of the ground-truth scores and estimated scores for a distribution with two modes (two gaussians). The estimated scores are wrong in low-density regions (in the middle of the map).



(a) Exact sampling of two gaussians with centers (-5,5) and (5,5) and weights 0.2 and 0.8 respectively.

(b) Sampling using conventional Langevin dynamics.

(c) Sampling using annealed Langevin dynamics.

Figure 3: Comparison of the Langevin dynamics and the annealed Langevin dynamics for a distribution with two modes (two gaussians with different weights). The conventional Langevin dynamic cannot recover the weights whereas the annealed Langevin dynamics can.

2.3 Theoretical solution

2.3.1 Building a particular sequence of distributions... To solve the problems mentioned above, the authors build a sequence of perturbed probability distributions $q_{\sigma_1}, q_{\sigma_2}, \dots, q_{\sigma_L}$, with $\{\sigma_i\}_{i=1}^L$ a geometrically decreasing sequence of noise scales satisfying $\frac{\sigma_1}{\sigma_2} = \dots = \frac{\sigma_{L-1}}{\sigma_L} > 1$, which converges to the actual probability distribution p_{data} we want to sample from. The last noise scale σ_L is chosen small enough so that $q_{\sigma_L}(\mathbf{x}) \approx p_{\text{data}}(\mathbf{x})$.

Each perturbed distribution $q_\sigma(\tilde{\mathbf{x}})$ is obtained by perturbing the input data point $\mathbf{x}$ with a pre-specified noise distribution $q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})$, chosen by the authors to be $\mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma\mathbf{I})$, an isotropic gaussian with mean $\mathbf{x}$ and variance $\sigma^2 I$.

2.3.2 ...That solves our (2.2.1) and (2.2.2) problems. The sequence $(q_{\sigma_i})_i$ verifies the following property. For all $i = 1, \dots, L$, we can initialize the Langevin dynamics in a region where $\nabla_{\mathbf{x}} \log q_{\sigma_i}(\mathbf{x})$ is well defined, thus resolving problems (2.2.1) and (2.2.2). The justification is done by induction:

- for $i = 1$, the noise σ_1 is large enough so that $\text{supp}(q_{\sigma_1}) \approx \mathbb{R}^d$, implying that any random point will be on $\text{supp}(q_{\sigma_1})$. Hence $\nabla_{\mathbf{x}} \log q_{\sigma_1}(\mathbf{x})$ is well defined at initialization.
- for all $i = 1, \dots, L-1$, by choosing $\sigma_i \approx \sigma_{i+1}$, we have $q_{\sigma_i}(\mathbf{x}) \approx q_{\sigma_{i+1}}(\mathbf{x})$ and therefore $\text{supp}(q_{\sigma_i}) \approx \text{supp}(q_{\sigma_{i+1}})$. A terminal sample from the Langevin Dynamics (LD) applied to $q_{\sigma_i}(\mathbf{x})$, for which we assume the LD can be properly initialized, will be a good initialization to run the LD for $q_{\sigma_{i+1}}(\mathbf{x})$.

When $i = L$, the property tells us that we can correctly sample from $q_{\sigma_L}(\mathbf{x}) \approx p_{\text{data}}(\mathbf{x})$ with the Langevin dynamics. An intuitive way of understanding the method is that when a data distribution is only supported on a manifold, the noisy distributions add "information" outside of this manifold.

2.3.3 ... And solves the mixing problem as well. In low-density regions between modes, where previously $p(\mathbf{x}) \approx 0$, now in q_{σ_i}, the contributions of each mode will appear with a strength following their weights in p_{data}, and thus the learned gradient will guide towards each mode with the correct likelihood (see an example of gradient maps in Figure 4).

3 Implementation

3.1 Reformulating the objective for score estimation

To sample from q_{σ_i} with Langevin dynamics, we need to train a "noise-conditional" neural network $s_\theta(\mathbf{x}, \sigma)$, which will estimate the score of the perturbed data distribution $q_\sigma(\tilde{\mathbf{x}})$ at a certain noise level σ. The objective to minimize is the following:

$$\frac{1}{2}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}})}\left[\|s_\theta(\tilde{\mathbf{x}}, \sigma) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})\|_2^2\right] \tag{4}$$

While the term $\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})$ is hardly tractable, minimizing this objective is exactly the same as minimizing the denoising score matching objective from Eq. (5) (see proof in Appendix B):

$$\frac{1}{2}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})p_{\text{data}}(\mathbf{x})}\left[\|s_\theta(\tilde{\mathbf{x}}, \sigma) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})\|_2^2\right] \tag{5}$$

This makes our objective much more tractable. Indeed, with gaussian noise, the objective becomes:

$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{\tilde{\mathbf{x}}\sim\mathcal{N}(\mathbf{x},\sigma^2 I)}\left[\left\|s_\theta(\tilde{\mathbf{x}}, \sigma) - \frac{\mathbf{x} - \tilde{\mathbf{x}}}{\sigma^2}\right\|_2^2\right] \tag{6}$$

In order to minimize the objective, the neural network, given the noise level and the perturbed distribution $q_\sigma(\tilde{\mathbf{x}})$, has to estimate the direction $\frac{\mathbf{x}-\tilde{\mathbf{x}}}{\sigma^2}$, which means moving from $\tilde{\mathbf{x}}$ to the actual data point $\mathbf{x}$.

3.2 Noise conditional score networks (NCSN)

3.2.1 Training a single NCSN. The authors chose to train a single network to estimate the score given the noise. Indeed, common information will be shared between different noisy representations of the distribution p_{data}, and training one single network could be more data efficient.

The objective for a single σ_i is of the form:

$$\ell(\theta; \sigma_i) = \frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{\tilde{\mathbf{x}}\sim\mathcal{N}(\mathbf{x},\sigma_i^2 I)}\left[\left\|s_\theta(\tilde{\mathbf{x}}, \sigma_i) - \frac{\mathbf{x} - \tilde{\mathbf{x}}}{\sigma_i^2}\right\|_2^2\right] \tag{7}$$

The unified objective for the noise conditional neural network (NCSN) is a weighted sum of the individual objectives:

$$\mathcal{L}(\theta; \{\sigma_i\}_{i=1}^L) = \frac{1}{L}\sum_{i=1}^L \lambda(\sigma_i)\ell(\theta; \sigma_i), \tag{8}$$

where the factors $\lambda(\sigma_i)$ are positive. Hence, $s_{\theta^*}(\mathbf{x}, \sigma)$ achieves a zero loss in (8) if and only if $s_{\theta^*}(\mathbf{x}, \sigma_i) = \nabla_{\mathbf{x}} \log q_{\sigma_i}(\mathbf{x})$ almost surely for all $i \in \{1, 2, \dots, L\}$.

The authors chose $\lambda(\sigma_i)$ so that all the terms of the sum have a similar order of magnitude, regardless of σ. We observe empirically that $\|s_\theta(\mathbf{x}, \sigma)\|_2 \propto 1/\sigma$ (see Figure 5). Moreover, for $\lambda(\sigma) \coloneqq \sigma^2$,

$$\lambda(\sigma)\ell(\theta; \sigma) = \sigma^2\ell(\theta; \sigma) = \frac{1}{2}\mathbb{E}\left[\left\|\sigma s_\theta(\tilde{\mathbf{x}}, \sigma) + \frac{\tilde{\mathbf{x}} - \mathbf{x}}{\sigma}\right\|_2^2\right],$$

and since $\frac{\tilde{\mathbf{x}}-\mathbf{x}}{\sigma} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\|\sigma s_\theta(\mathbf{x}, \sigma)\|_2 \propto 1$, the order of magnitude of $\lambda(\sigma)\ell(\theta; \sigma)$ does not depend on σ. The unified objective becomes:

$$\mathcal{L}(\theta; \{\sigma_i\}_{i=1}^L) = \frac{1}{L}\sum_{i=1}^L \sigma_i^2\ell(\theta; \sigma_i) \tag{9}$$

3.2.2 NCSN architecture. For generating images, the neural network is built on a U-Net architecture [2], which is a convolutional neural network architecture initially designed for image segmentation. It features an encoder-decoder architecture and uses skip connections to retain high-resolution features for accurate localization.

3.2.3 Conditional instance normalization. Instance normalization is applied on every feature map for each input $\mathbf{x}$ to facilitate the training of the network $s_\theta(\mathbf{x}, \sigma)$ [6]. The authors adopt conditional (with respect to σ) instance normalization so that $s_\theta(\mathbf{x}, \sigma)$ takes into account the level of noise in its prediction. If $\mathbf{x}$ has C input features, let μ_k and s_k be the mean and standard deviation of the k^{th} feature map of $\mathbf{x}$. The learnable parameters of the normalization are $\gamma \in \mathbb{R}^{L\times C}$, $\beta \in \mathbb{R}^{L\times C}$ and $\alpha \in \mathbb{R}^{L\times C}$. The new conditional instance normalization introduced by the authors is:

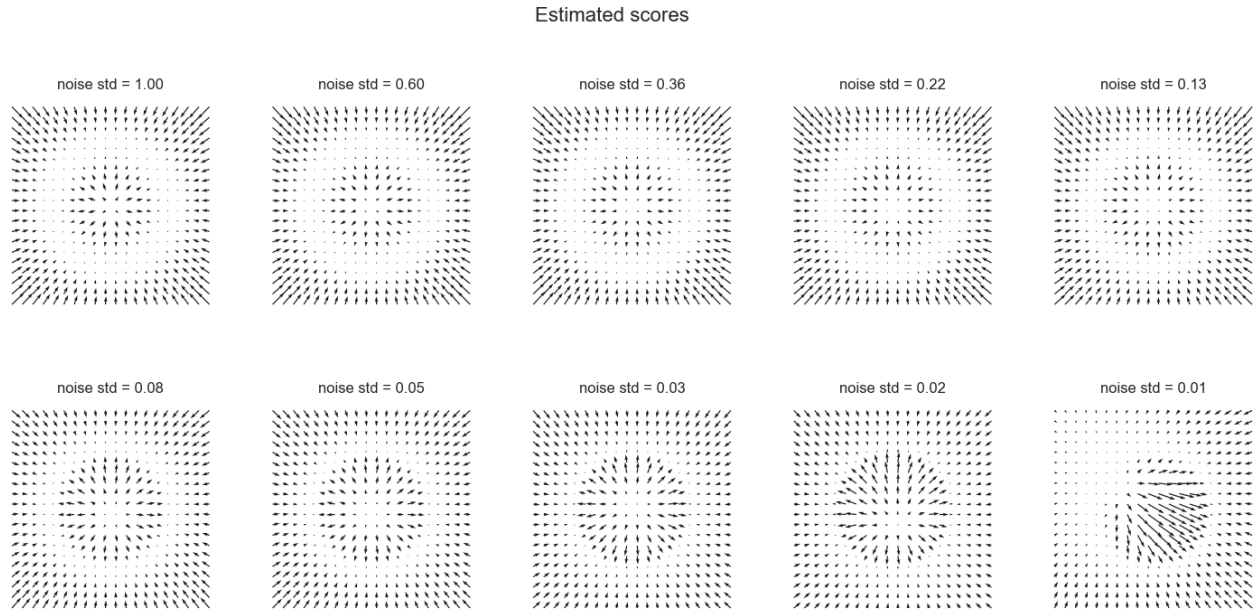


Figure 4: Estimated scores $s_\theta(\mathbf{x}, \sigma_i)$ for $\sigma_1 = 1$ to $\sigma_L = 10^{-2}$. For σ_i large, the gradients globally push the samples towards the manifold, and as σ_i decreases, the gradients are more accurate close to the manifold. The last gradient map is anisotropic, but the samples are supposed to be closed to the manifold, so most gradients will not be used.

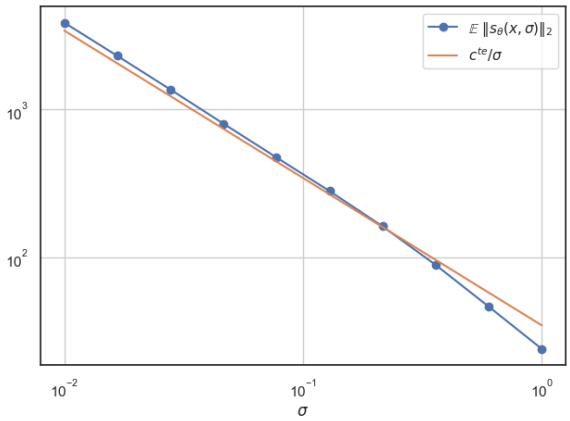


Figure 5: Expected norm $\mathbb{E}\left[\|s_\theta(x,\sigma)\|_2\right]$ of the score network as a function of σ. This empirically validates that $s_\theta(x,\sigma) \propto \frac{1}{\sigma}$, and thus motivates the unified objective from Eq. (9). The expectation is computed for the score network trained by Y. Song et al. on MNIST. Inputs are sampled uniformly from $[0,1]^{28\times 28}$.

$$z_k = \gamma[i,k]\frac{x_k - \mu_k}{s_k} + \beta[i,k] + \alpha[i,k]\frac{\mu_k - m}{v} \tag{10}$$

where k is the index of feature maps, and i the index of σ in $\{\sigma_i\}_{i=1}^L$.

The dependence on σ is thus in the parameters of γ, β and α. The novelty of this formula is to add a factor $\alpha[i,k]\frac{\mu_k - m}{v}$ that takes into account the information provided by the mean μ_k of the feature map k. Especially in images, the mean of a feature map can encode important information, for example, overall brightness or tint: removing this information can generate images with shifted colors.

3.3 Annealed Langevin dynamics

Once the scores are estimated for all perturbed distributions, we sequentially sample from all perturbed probability densities $q_{\sigma_1}, q_{\sigma_2}, ...q_{\sigma_L}$, each time using a Langevin dynamics.

Let $\tilde{\mathbf{x}}_i^t$ denote the t-th step of the Langevin dynamics ran while sampling from q_{σ_i}. In particular, $\tilde{\mathbf{x}}_1^1$ is chosen randomly, and each initialization of the Langevin dynamics uses the final sample of the previous iteration:

$$\tilde{\mathbf{x}}_i^1 = \tilde{\mathbf{x}}_{i-1}^T.$$

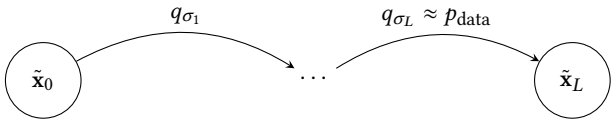
For each $i \in \{1, ..., L\}$, we run a Langevin sampling as follows. For $t \in \{1, ..., T\}$, sample $\mathbf{z}_t \sim \mathcal{N}(0, \mathbf{I})$ and compute:

$$\tilde{\mathbf{x}}_i^t = \tilde{\mathbf{x}}_i^{t-1} + \frac{\epsilon_i}{2} s_\theta(\tilde{\mathbf{x}}_i^{t-1}, \sigma_i) + \sqrt{\epsilon_i}\mathbf{z}_t \tag{11}$$

The first Langevin dynamics (sampling under q_{σ_1}) will move $\tilde{\mathbf{x}}_0^1$ to $\tilde{\mathbf{x}}_0^T = \tilde{\mathbf{x}}_1^1$, then the second Langevin dynamics (sampling under q_{σ_2}) will move $\tilde{\mathbf{x}}_1^1$ to $\tilde{\mathbf{x}}_1^T = \tilde{\mathbf{x}}_2^1$... and the last Langevin dynamics (sampling under $q_{\sigma_L} \approx p_{\text{data}}$) will move $\tilde{\mathbf{x}}_L^1$ to $\tilde{\mathbf{x}}_L^T$ which should be a representative sample from p_{data} (see Figure 6 for a graph representation of the annealed Langevin dynamics).

The parameter ϵ_i must be chosen for each Langevin sampling corresponding to q_{σ_i}. The authors chose it so that the "signal-to-noise ratio" in Langevin dynamics does not depend on σ_i. Here, the "signal-to-noise ratio" is:

Figure 6: In the following graph, we see the path of one single random sample. The initial input point $\tilde{\mathbf{x}}_0$ is random. The final output $\tilde{\mathbf{x}}_L$ is sampled with probability distribution $q_{\sigma_L} \approx p_{\text{data}}$.



$$\frac{\epsilon_i s_\theta(x, \sigma_i)}{2\sqrt{\epsilon_i} z},$$

and we have:

$$\mathbb{E}\left[\left\|\frac{\epsilon_i s_\theta(x, \sigma_i)}{2\sqrt{\epsilon_i} z}\right\|_2^2\right] \approx \mathbb{E}\left[\frac{\epsilon_i \|s_\theta(x, \sigma_i)\|_2^2}{4}\right]$$

Empirically (see Figure 5), $\|s_\theta(x, \sigma)\|_2 \propto 1/\sigma$ when the score network is trained close to optimal. When choosing $\epsilon_i \propto \sigma_i^2$,

$$\mathbb{E}\left[\frac{\epsilon_i \|s_\theta(x, \sigma_i)\|_2^2}{4}\right] \propto \frac{1}{4}\mathbb{E}\left[\|\sigma_i s_\theta(x, \sigma_i)\|_2^2\right] \propto \frac{1}{4}.$$

does not depend on σ_i.

Finally, the authors take $\epsilon_i = \sigma_i^2 \frac{\epsilon}{\sigma_L^2}$ to have

- $\epsilon_i \propto \sigma_i^2$
- ϵ_i small (as the constant ϵ is small) so that the Langevin dynamics assumptions for convergence are verified
- and ϵ_i homogeneous to ϵ, so that only ϵ has to be tuned.

The efficiency of choosing an ϵ_i depending on the noise level is demonstrated in Figure 7.

Figure 8 shows a spatial representation of the annealed Langevin dynamics: since the step size ϵ_i decreases with the noise levels, the sample moves more slowly as it gets closer to the actual distribution; this is appropriate to have a higher precision close to p_{data}.

3.4 Link and difference with diffusion models

The approach detailed above has similarities with other denoising-based methods and, in particular, with Diffusion Models [3]. Indeed, diffusion models also learn a sequence of transformations to iteratively turn random noise into samples that could have been drawn from p_{data}. In a nutshell, diffusion models use a Markov chain—usually with a Gaussian transition kernel such that $X_{t+1} \sim \mathcal{N}(\sqrt{1-\beta_t}X_t, \sqrt{\beta_t}I)$ where $(\beta_t)_t$ defines the noise schedule—to iteratively add noise to the collected data samples. They then try to learn the reverse transitions to reconstruct the original image.

In our paper, the denoising transitions are actually the combination of the learned score for a given noise level, and its associated Langevin Dynamics. On the one hand, unlike diffusion models, this approach does not allow for exact sampling. On the other hand, this decouples the score learning step from the sampling step, thus giving the choice of the sampling dynamic. Moreover, diffusion models usually require a few thousands of transitions in order to produce high-fidelity images. Although this computational limitation has since then been mitigated, it used to be a major bottleneck that prevented the scaling of such models to high-resolution images.

4 Limitations / Additional empirical evidence

The main constraint of this new method is to correctly choose the newly introduced hyperparameters: ϵ, T, and most importantly, the noises $\{\sigma_i\}_{i=1}^L$.

4.1 Choice of σ's

4.1.1 Challenges.

σ is sensible to the diameter of the distribution. Figure 9 shows an experiment with a very high diameter (in the mathematical sense of the largest distance between two points of the dataset). When keeping the same noise levels as before ($\sigma_1 = 1$ and $\sigma_L = 10^{-2}$), the Langevin dynamics does not converge towards the true distribution, even when varying parameters ϵ and T. Intuitively, we understand that if the first noise is too small, it will not mitigate the problem of low-density regions and the support of the noisy distributions will not be the whole ambient space as needed.

4.1.2 Heuristics.

Choosing σ_1 depending on data diameter. To correct the problem highlighted in 4.1.1, we must choose σ_1 conditionally to the data distribution. In [5], the authors propose σ_1 to be as large as the maximum Euclidean distance between all pairs of training data points. Indeed, when we have N points $\{x^{(i)}\}_{i=1}^N$ in the training dataset:

$$\hat{p}_{\text{data}}(x) = \frac{1}{N}\sum_{i=1}^N \delta(x = x^{(i)}),$$

where $\delta(\cdot)$ denotes a point mass distribution. When perturbed with $\mathcal{N}(0, \sigma_1^2 I)$, the empirical distribution becomes:

$$q_{\sigma_1}(x) = \frac{1}{N}\sum_{i=1}^N q^{(i)}(x),$$

where $q^{(i)}(x) = \mathcal{N}(x \mid x^{(i)}, \sigma_1^2 I)$.

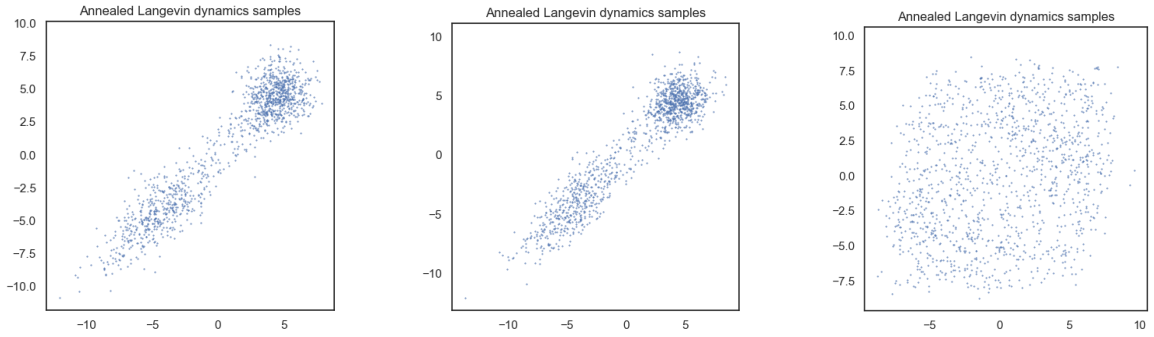
The Langevin dynamics must be able to generate samples regardless of the initialization, which means that it can explore any component $q^{(i)}(x)$ when initialized from any other component $q^{(j)}(x)$, where $i \neq j$.
Let $q_{\sigma_1}(x) = \frac{1}{N}\sum_{i=1}^N q^{(i)}(x)$, where $q^{(i)}(x) = \mathcal{N}(x \mid x^{(i)}, \sigma_1^2 I)$. With $r^{(i)}(x) = \frac{q^{(i)}(x)}{\sum_{k=1}^N q^{(k)}(x)}$ the relative weight of point $q^{(i)}(x)$, the score function is:

$$\nabla_x \log q_{\sigma_1}(x) = \sum_{i=1}^N r^{(i)}(x) \nabla_x \log q^{(i)}(x).$$

Moreover, one can show that:

$$\mathbb{E}_{q^{(i)}(x)}\left[r^{(j)}(x)\right] \leq \frac{1}{2}\exp\left(-\frac{\|x^{(i)} - x^{(j)}\|_2^2}{8\sigma_1^2}\right).$$

In order for Langevin dynamics to transition from $q^{(i)}(x)$ to $q^{(j)}(x)$ easily for $i \neq j$, $\mathbb{E}_{q^{(i)}(x)}\left[r^{(j)}(x)\right]$ has to be relatively large, because otherwise the term corresponding to $q^{(j)}(x)$ in $\nabla_x \log q_{\sigma_1}(x) = \sum_{k=1}^N r^{(k)}(x) \nabla_x \log q^{(k)}(x)$ will be ignored (on average) when initialized with $x \sim q^{(i)}(x)$, and in such case $q^{(j)}(x)$ will not have any influence in Langevin dynamics. The bound



(a) $\epsilon = 0.01$ **(b)** $\epsilon = 10^{-3}$ **(c)** $\epsilon = 10^{-4}$

Figure 7: Plot of the Langevin dynamics with different constant step sizes ϵ. None of the Langevin dynamics converge, which highlights the importance of an ϵ_i depending on σ_i.

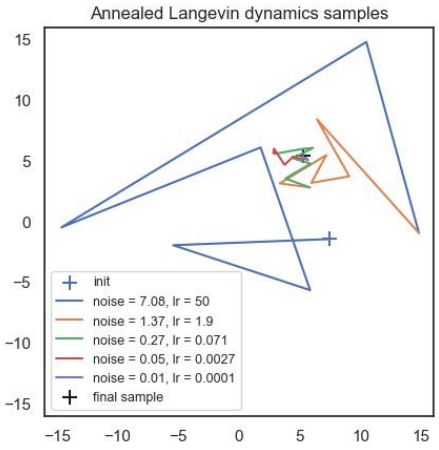


Figure 8: Path of a random sample passing through several Langevin dynamics and converging to the true data distribution. We represent only six iterations per Langevin dynamic.

above indicates that the decrease of $\mathbb{E}_{q^{(i)}(x)}[r^{(j)}(x)]$ must be controlled and is exponential unless σ_1 is numerically comparable to $\|x^{(i)} - x^{(j)}\|_2^2$.

Geometric progression for $(\sigma_i)_i$. In the above algorithm, the sequence $(\sigma_i)_{i\in[\![1,L]\!]}$ was chosen in geometric progression without much justification. This choice was mostly driven by convincing empirical results and a search space reduction. Indeed, σ_1, σ_L, and L are parameters to tune, and we do have heuristics for those parameters: σ_1 must be as large as the diameter of the problem, σ_L must be small enough so that the last distribution is not too perturbed, and L is given by our compute budget. This geometric progression is thus very practical. In this section, we show that this choice is also theoretically motivated [5].

We indeed seek to construct a sequence of perturbed distributions $(q_{\sigma_i})_i$ such that consecutive distributions overlap enough so that one can initialize the Langevin Dynamics (LD) for q_{σ_i} from the final sample from the LD from $q_{\sigma_{i-1}}$.

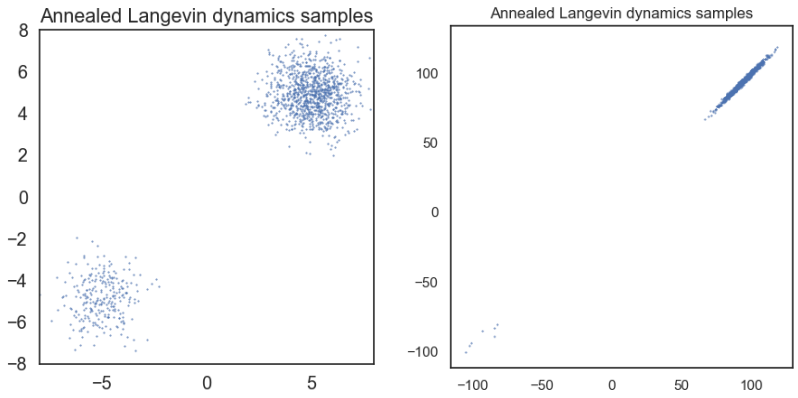
Now, if we look at the very simple case in which our dataset only consists of one point, the perturbed version q_{σ_i} of the empirical distribution δ_x is $q_{\sigma_i} = \mathcal{N}(x, \sigma_i I)$. We can decompose q_{σ_i} into hyperspherical coordinates $q^{\phi}(\phi)q^r_{\sigma_i}(r)$, and since all our perturbed distributions are isotropic, q^{ϕ} is the same across all σs and will not influence the overlap between distributions. Moreover, in high dimension, which is the case in practice when dealing with images, the radial component $q^r_{\sigma_i}$ can be approximated by a Gaussian $\mathcal{N}(\sigma_i\sqrt{D}, \frac{\sigma_i^2}{2})$, where D is the problem dimension.

Then, to have a significant enough overlap, we want $q^r_{\sigma_{i-1}}$ to have a high density in the interval in which $q^r_{\sigma_{i-1}}$ concentrates most of its density, i.e. $\mathcal{I}_{i-1} \coloneqq [\sigma_{i-1}\sqrt{D} - 3\frac{\sigma_{i-1}^2}{2}, \sigma_{i-1}\sqrt{D} + 3\frac{\sigma_{i-1}^2}{2}]$, using the 3-$\sigma$s rule of thumb. One can easily show that $\mathbb{P}(q^r_{\sigma_i} \in \mathcal{I}_{i-1}) = \Phi(\sqrt{2D}(\gamma_i - 1) + 3\gamma_i) - \Phi(\sqrt{2D}(\gamma_i - 1) - 3\gamma_i)$, where Φ is the CDF of a standard gaussian and $\gamma_i \coloneqq \frac{\sigma_{i-1}}{\sigma_i}$. Thus, fixing $\mathbb{P}(q^r_{\sigma_i} \in \mathcal{I}_{i-1})$ to some constant close enough to 1 to guarantee the overlap of consecutive distributions directly implies that all γ_i should be equal, and thus, that the sequence $(\sigma_i)_i$ is in geometric progression.

4.2 Choice of ϵ and T

For each Langevin dynamics, we must choose a small step ϵ, and a large number of Langevin iterations T (to verify the convergence conditions). However, T is constrained by our compute budget, and thus ϵ must not be too small if we want samples to actually get close to the data's manifold in a fixed number T of iterations.

4.2.1 Challenges. We created a complex distribution to test the limits of the algorithm in Figure 10: two gaussians, one with large support but small weight in the mixture and one with very small support but large weight, which is a hard case of slow mixing. The score maps give us some interesting insights into the method: if the sample was given only the real score map, it would not be attracted to the peaky gaussian unless initialized really close to it; instead, when the score map corresponding to the biggest noise is used, the domain of attraction of the peaked gaussian is made larger than it is in reality to attract samples, and then the gradients become more peaked around it. To sample from the annealed Langevin dynamics, we used these score maps, but we changed the parameters T and



(a) Sampling using annealed Langevin dynamics with estimated scores ($\sigma_1 = 1$ and $\sigma_L = 10^{-2}$) for relatively close gaussians.

(b) Sampling using annealed Langevin dynamics with estimated scores ($\sigma_1 = 1$ and $\sigma_L = 10^{-2}$) for far apart gaussians.

Figure 9: Comparison of annealed Langevin dynamics with two gaussians separated by different distances. For the same noise levels and parameters, the problem breaks with the diameter of the problem.

	$T = 1000$	$T = 10\,000$	$T = 30\,000$
$\epsilon = 10^{-2}$	0.07	0.46	0.78
$\epsilon = 10^{-3}$	0.47	0.86	0.94
$\epsilon = 3.10^{-4}$	0.46	0.83	0.93
$\epsilon = 10^{-4}$	0.36	0.6	0.77

Table 1: Table showing the relative weight of the peaked gaussian from the example in 10. Real weight is 0.8.

ϵ and noticed that the method is very sensitive to the choice of hyperparameters.

Table 1 shows the results of our experiments. We notice that :

- Unless the number of iterations T is high enough, we do not recover the correct weights, regardless of the chosen step size ϵ. This highlights the fact that T must be chosen according to the complexity of the problem: indeed, in the previous 2D experiments we ran on gaussian data, $T = 1000$ was enough.
- given a high enough T, the success of the experiment highly depends on the choice of the parameter ϵ, and there is no obvious way to tune it.

At first sight, this seems like a weakness of the algorithm, because the hyperparameters T and ϵ must be fine-tuned for each problem.

4.2.2 Heuristics. However, the authors in [5] give a heuristic to choose ϵ given the geometric reason γ and the number of iterations per Langevin dynamics T. They advise to choose ϵ so that the ratio

$$\left(1 - \frac{\epsilon}{\sigma_L^2}\right)^{2T} \left(\gamma^2 - \frac{2\epsilon}{\sigma_L^2 - \sigma_L^2\left(1 - \frac{\epsilon}{\sigma_L^2}\right)^2}\right) + \frac{2\epsilon}{\sigma_L^2 - \sigma_L^2\left(1 - \frac{\epsilon}{\sigma_L^2}\right)^2}.$$

is close to 1, and to this end recommend performing a grid search over ϵ. This theoretical result meets the results we get in Table 1.

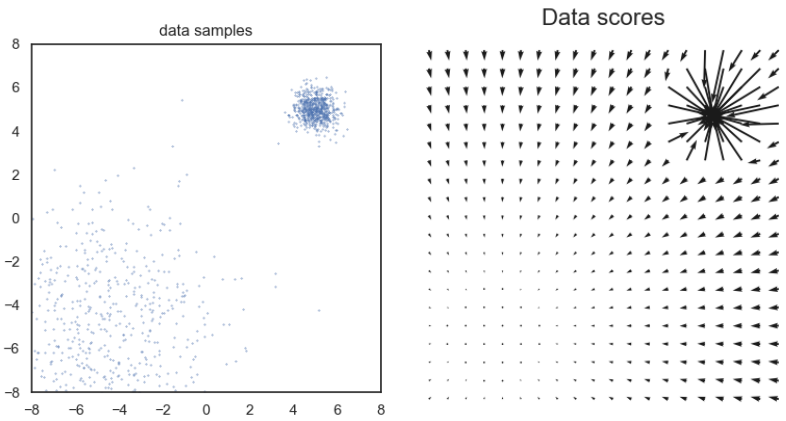
For images, this also seems to produce good samples (see Figure 11). However, when running the method on MNIST, we notice that the images' fidelity improves very fast at first - so that after only half of the iterations, one can easily recognize the sampled digits - but that as many iterations are then needed to improve over very slight details and make images sharp (11). This leads us to think that the choice of T and ϵ might not be optimal. However, it is also an intrinsic limitation of the method, which works on levels of details of exponentially decreasing scale. This high inference cost will be prohibitive in many applications, and in such cases, GANs or flow models will be preferred.

In this section, we highlighted the fact that choosing T and ϵ is paramount, and yet tricky:

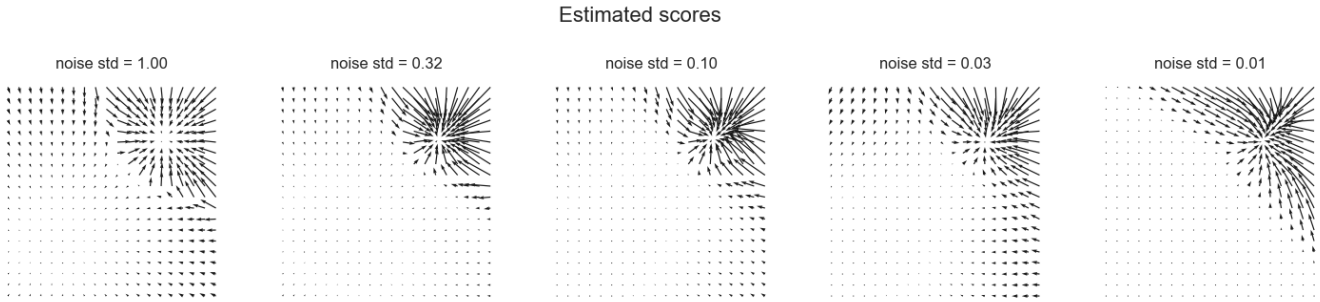
- T must be high enough for the method to converge regardless of ϵ, as our example highlighted. We cannot just take the best computation allocation we have.
- when T is given, there exists a specific range of ϵ that is necessary to produce good samples (the rule given by the authors). It means that in some cases, an additional computation for grid-search must be added to find a good ϵ.
- even when T and ϵ produce good samples, there might be ways to fine-tune the model and reduce the number of iterations.

4.3 Conclusion

To conclude on the limitations, we believe the method relies on a significant amount of hyperparameters fine-tuning. It might not be an issue when there are a lot of available computational resources and when there is a clear way of validating the results (for example, in image generation). However, when there is no easy validation, fine-tuning this method becomes particularly tricky.



(a) Real samples of the data distribution. **(b)** Real scores of the distribution.



(c) Estimated scores by the NCSN for $\sigma_1 = 1$ to $\sigma_L = 10^{-2}$.

Figure 10: Samples, scores and estimated scores of a complex problem consisting of two gaussians. One has parameters : center (-5, -5), $\sigma = 3I$, weight=0.2 and one has parameters : center (5, 5), $\sigma = 0.5I$ and weight=0.8.

(a) Initial state. **(b)** Iteration 100. **(c)** Iteration 200. **(d)** Iteration 300. **(e)** Iteration 400. **(f)** Iteration 600. **(g)** Iteration 1000.

Figure 11: Running the experience on MNIST.

5 Extensions

5.1 Changing sampling method

We tried using the Hamilton Monte Carlo dynamics instead of the Langevin dynamics. However, the method did not converge, even after changing the parameters. We think this might come from the fact that we cannot implement the Metropolis Hasting acceptance step (which requires knowing $\log p_{\text{data}}(\mathbf{x})$). The sampling method thus loses its theoretical guarantees, leading to incorrect results.

This highlights the fact that, contrary to what the authors state, the Langevin dynamics is much more adapted to their method than the HMC. Indeed, although the MH acceptance step is needed in the conventional Langevin dynamics as well, the fact that it is not computed here does not prevent the method from producing accurate samples.

References

[1] T Chaing, Cr Hwang, and Shuenn-Jyi Sheu. Diffusion for global optimization in n r. *Siam Journal on Control and Optimization - SIAM*, 01 1987.

[2] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[3] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *CoRR*, abs/1503.03585, 2015.

[4] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *CoRR*, abs/1907.05600, 2019.

[5] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *CoRR*, abs/2006.09011, 2020.

[6] D Ulyanov. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[7] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23:1661–74, 07 2011.

A Convergence of the Langevin Dynamics

Below are some notations and assumptions required to state the Theorem A.1 below, which we take from the paper [1]. This theorem will give us the theoretical setting in which sampling via the Langevin Dynamics allows us to actually sample from the learned data density. This theorem will consider the stochastic differential equation :

$$dx(t) = -\nabla U(x(t))\,dt + \sigma(t)\,dz(t),$$

where $z(t)$ is standard Brownian motion.

In this equation, let U be a twice continuously differentiable function from $\mathbb{R}^n$ to $[0,\infty)$ such that the following assumptions hold:

$$\min_{x\in\mathbb{R}^n} U(x) = 0,$$

(A1) $U(x) \xrightarrow[\|x\|\to+\infty]{} +\infty$, $|\nabla U(x)| \xrightarrow[\|x\|\to+\infty]{} +\infty$, and $\lim_{|x|\to\infty} |\nabla U(x)|^2 - \Delta U(x) > -\infty$.

(A2)

For all $0 < \epsilon < 1$, $\pi^\epsilon(x) := \frac{1}{c(\epsilon)} \exp\left(-\frac{2U(x)}{\epsilon^2}\right)$ is well-defined, where

$$c(\epsilon) = \int_{\mathbb{R}^n} \exp\left(-\frac{2U(x)}{\epsilon^2}\right) dx < \infty.$$

(A3) The measure associated with the density π^ϵ has a unique weak limit π as $\epsilon \downarrow 0$.

For simplicity, let us assume that $\sigma^2(t) < 1$, $\sigma^2(t) = c/\log t$ for large t, and the process $x(t)$ starts at $x(0) = x_0$.

Let S denote the set of all stationary points of U, i.e.,

$$S = \{x \mid \nabla U(x) = 0\}.$$

For any $\eta > 0$, $\xi > 0$, we define the following:

$$\begin{aligned}
&S(\eta) := \{x \mid d(x,S) < \eta\},\\
&K(\eta) := \text{the set containing all the solutions}\\
&\text{of the dynamical system (1.3) with starting}\\
&\text{points in } S(\eta),\\
&K(\eta,\xi) := \{x \mid d(x,K(\eta)) \le \xi\},\\
&I(t,x,y) := \inf_{\substack{\psi(0)=x\\ \psi(t)=y}} \frac{1}{2}\int_0^t |\dot\psi(s) + \nabla U(\psi(s))|^2\,ds,\\
&J(t,\eta,\xi) := \sup_{x,y\in K(\eta,\xi)} \big(I(t,x,y) - 2U(y)\big),\\
&J(\infty,\eta,\xi) := \lim_{t\to\infty} J(t,\eta,\xi),\\
&c_0 := \frac{3}{2}\inf_\eta\left(\inf_\xi J(\infty,\eta,\xi)\right).
\end{aligned}$$

For a measure μ, define $\mu(f) := \int f d\mu$.

Theorem A.1. *Let $p(s,x,t,\cdot)$ denote the transition probability of*

$$dx(t) = -\nabla U(x(t))\,dt + \sigma(t)\,dz(t),$$

Assume (A1), (A2), and (A3) and $c > c_0$; then for any bounded continuous function f

$$p(0,x,t,f) \to \pi(f) \quad \text{as } t\to\infty,$$

and the convergence is uniform for x in a compact set.

Let us now link this theorem with our Langevin Dynamics from Eq. 1. If one defines $U := -\frac{\varepsilon}{2}\log p$ and if one discretizes the above stochastic differential equation, one obtains

$$x_{t+1} = x_t + \frac{\varepsilon}{2}\nabla \log p(x) + \sigma_t z_t,$$

where $z_t \sim \mathcal{N}(0,1)$.

For a fixed σ_t equal to $\sqrt{\varepsilon}$, we recover the Langevin Dynamics used in Eq. 1. However, σ_t being fixed, we are not in the exact setting from Theorem A.1 even if we disregard the above assumptions on U. This theoretical limitation is acknowledged by the authors, who argue that a correction is not needed. Indeed, one known correction when p is known is the Metropolis-adjusted Langevin algorithm, and several papers have found that this correction does not lead to significant performance improvements. Based on this empirical evidence, the authors argue that a similar correction would not necessarily be beneficial. Note that as we are learning the score only, we do not have a direct density estimate and thus would require a different method to validate the authors' hypothesis.

B Equivalent objective for denoising score matching

The following proof is taken from [7]. The explicit score matching criterion defined in Eq. 4 is

$$J_{ESM}(\theta,\sigma,q_\sigma) := \frac{1}{2}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}})}\left[\|s_\theta(\tilde{\mathbf{x}},\sigma) - \nabla_{\tilde{\mathbf{x}}}\log q_\sigma(\tilde{\mathbf{x}})\|_2^2\right] \tag{12}$$

which we can develop as

$$J_{ESM}(\theta,\sigma,q_\sigma) = \mathbb{E}_{q_\sigma(\tilde{x})}\left[\frac{1}{2}\|s_\theta(\tilde{x},\sigma)\|^2\right] - S(\theta) + C_1 \tag{13}$$

where $C_1 := \mathbb{E}_{q_\sigma(\tilde{x})}\left[\frac{1}{2}\left\|\frac{\partial \log q_\sigma(\tilde{x})}{\partial \tilde{x}}\right\|^2\right]$ is a constant that does not depend on θ, and

$$S(\theta) := \mathbb{E}_{q_\sigma(\tilde{x})}\left\langle s_\theta(\tilde{x},\sigma), \frac{\partial \log q_\sigma(\tilde{x})}{\partial \tilde{x}}\right\rangle$$

Expanding $S(\theta)$, we get:

$$\begin{aligned}
S(\theta) &= \int_{\tilde{x}} q_\sigma(\tilde{x}) \left\langle s_\theta(\tilde{x}, \sigma), \frac{\partial \log q_\sigma(\tilde{x})}{\partial \tilde{x}} \right\rangle d\tilde{x} \\
&= \int_{\tilde{x}} \left\langle s_\theta(\tilde{x}, \sigma), \frac{\partial q_\sigma(\tilde{x})}{\partial \tilde{x}} \right\rangle d\tilde{x} \\
&= \int_{\tilde{x}} \left\langle s_\theta(\tilde{x}, \sigma), \frac{\partial}{\partial \tilde{x}} \left(\int_x p_{data}(x) q_\sigma(\tilde{x} \mid x) dx \right) \right\rangle d\tilde{x} \\
&= \int_{\tilde{x}} \left\langle s_\theta(\tilde{x}, \sigma), \int_x p_{\text{data}}(x) \frac{\partial q_\sigma(\tilde{x} \mid x)}{\partial \tilde{x}} dx \right\rangle d\tilde{x} \\
&= \int_{\tilde{x}} \left\langle s_\theta(\tilde{x}, \sigma), \int_x p_{\text{data}}(x) q_\sigma(\tilde{x} \mid x) \frac{\partial \log q_\sigma(\tilde{x} \mid x)}{\partial \tilde{x}} dx \right\rangle d\tilde{x} \\
&= \int_{\tilde{x}} \int_x p_{\text{data}}(x) q_\sigma(\tilde{x} \mid x) \left\langle s_\theta(\tilde{x}, \sigma), \frac{\partial \log q_\sigma(\tilde{x} \mid x)}{\partial \tilde{x}} \right\rangle dx d\tilde{x} \\
&= \int_{\tilde{x}} \int_x q_\sigma(\tilde{x}, x) \left\langle s_\theta(\tilde{x}, \sigma), \frac{\partial \log q_\sigma(\tilde{x} \mid x)}{\partial \tilde{x}} \right\rangle dx d\tilde{x} \\
&= \mathbb{E}_{q_\sigma(\tilde{x}, x)} \left[\left\langle s_\theta(\tilde{x}, \sigma), \frac{\partial \log q_\sigma(\tilde{x} \mid x)}{\partial \tilde{x}} \right\rangle \right]
\end{aligned}$$

Substituting this expression for $S(\theta)$ in Eq. (13) yields:

$$J_{ESM}(\theta, \sigma, q_\sigma) = \mathbb{E}_{q_\sigma(\tilde{x})} \left[\frac{1}{2} \left\| s_\theta(\tilde{x}, \sigma) \right\|^2 \right] - \mathbb{E}_{q_\sigma(\tilde{x}, x)} \left\langle s_\theta(\tilde{x}, \sigma), \frac{\partial \log q_\sigma(\tilde{x} \mid x)}{\partial \tilde{x}} \right\rangle + C_1$$

We also defined in Eq. (5) the denoising score matching objective

$$J_{DSM}(\theta, \sigma, q_\sigma) \coloneqq \frac{1}{2} \mathbb{E}_{q_\sigma(x, \tilde{x})} \left[\left\| s_\theta(\tilde{x}, \sigma) - \frac{\partial \log q_\sigma(\tilde{x} \mid x)}{\partial \tilde{x}} \right\|^2 \right],$$

which we can develop as

$$J_{DSMq_\sigma}(\theta) = \mathbb{E}_{q_\sigma(\tilde{x})} \left[\frac{1}{2} \left\| s_\theta(\tilde{x}, \sigma) \right\|^2 \right] - \mathbb{E}_{q_\sigma(x, \tilde{x})} \left\langle s_\theta(\tilde{x}, \sigma), \frac{\partial \log q_\sigma(\tilde{x} \mid x)}{\partial \tilde{x}} \right\rangle + C_2$$

where $C_2 = \mathbb{E}_{q_\sigma(x, \tilde{x})} \left[\frac{1}{2} \left\| \frac{\partial \log q_\sigma(\tilde{x} \mid x)}{\partial \tilde{x}} \right\|^2 \right]$ is a constant that does not depend on θ.

Putting everything together, we have:

$$J_{ESMq_\sigma}(\theta) = J_{DSMq_\sigma}(\theta) + C_1 - C_2.$$

Therefore, it is equivalent to minimize the denoising score matching objective, which turns out to be easily tractable.