Denoising Score Matching for Diffusion Models

Halvard BARILLER
halvard.bariller@ens-paris-saclay.fr
ENS Paris-Saclay
Paris, France

Franki NGUIMATSIA TIOFACK
franki.nguimatsiatiofack@ensae.fr
ENSAE
Paris, France

Junior Cedric TONGA
junior.tonga@ens-paris-saclay.fr
ENS Paris-Saclay
Paris, France

Abstract

In this work, we consider the class of generative models, and more specifically score-based models. With a focus on the work of Song and Ermon [2019] in the article *Generative Modeling by Estimating Gradients of the Data Distribution* and of Vincent [2011] in the article *A Connection Between Score Matching and Denoising Autoencoders*, we provide a comprehensive investigation of the possibilities that this approach offers for data generation as compared to existing techniques. We first introduce the score-based approach and present how one can use the score function to generate data using Langevin dynamics. We then expand on mathematical results that allow to derive the score function from the data distribution, and move on to present the main contributions of Song and Ermon [2019] when it comes to improving the generating process building on Vincent [2011]. Throughout this paper, we will punctuate the methodology used with comments on choices and potential limitations, supported by a series of toy experiments. Finally, we conclude by discussing extensions of the described methodology.

Keywords: Generative Model, Score Matching, Langevin Dynamics

1 Introduction

Since the advent of ChatGPT, generative models have become the latest craze as far as Machine Learning is concerned. For the past year, *"GenAI"* has flooded the news, becoming a trending topic as well in non-technical audience as in the scientific community. New models are being released nearly on a daily basis, with numerous applications ranging from text generation [Devlin et al. 2018; Scao et al. 2022; Touvron et al. 2023; Vaswani et al. 2017] to image generation [Goodfellow et al. 2014; Ho et al. 2020; Shaham et al. 2019], by way of music or video generation [Chan et al. 2019]. The latest developments are showing impressive results in terms of multi-modal generation [Copet et al. 2023; Kreuk et al. 2023; Ramesh et al. 2021].

In spite of the recent hype, Generative Modeling is not a new concept. Indeed, it has been around for quite some time in the Statistical and Machine Learning community [Ng and Jordan 2001] and several mathematical results that we will present can be traced back to the early 2010's. The latest advances have been however made possible as technical progresses have brought a so-far unmatched computational power, leveraging the possibilities offered by efficient GPU programming for example [Abadi et al. 2016; Charlier et al. 2021; Paszke et al. 2017].

We note $\mathcal{D}_N = \{(\mathbf{x}_n, y_n), 1 \leq n \leq N\}$ a set composed of data instances and their associated labels. Formally, while discriminative models are trying to capture the conditional probability $p(y|\mathbf{x})$, generative models estimate the joint probability $p(\mathbf{x}, y)$ (or solely $p(\mathbf{x})$ in the absence of labels). We consider N samples $\{\mathbf{x}_n \in \mathbb{R}^d\}_{1 \leq n \leq N}$ independent and identically distributed according to an unknown probability distribution $\mathbf{x}_n \sim p_{data}(\mathbf{x})$. In the classical statistical approach, we consider a family of parameterized models $\mathcal{P} = (p_\theta)_{\theta \in \mathbb{R}^d}$ and we want to choose a model p_θ such that $p_\theta(\mathbf{x}) \approx p_{data}(\mathbf{x})$. One can do so by comparing distances between probability distributions [Feydy 2020], using metrics such as the Kullback-Leibler divergence yielding maximum likelihood [Arjovsky et al. 2017] or the Fisher-Rao distance if the geometry is propitious. Modelling a probability distribution is however more delicate than performing a classification task: one of the main challenge has been that as the model becomes more complex, density estimation rapidly becomes impractical due to an intractable normalizing term. This issue only worsens with high-dimensional data. In such setting, the problem can no longer be directly considered and needs clever techniques to circumvent this intractable partition function.

Behind the catch-all term generative models, we can distinguish between two main different types: likelihood-based models and implicit generative models. The former category contains models such as autoregressive models [Larochelle and Murray 2011; Van Den Oord et al. 2016], normalizing flow models [Dinh et al. 2014; Rezende and Mohamed 2015], energy-based models (EBMs), and variational auto-encoders (VAEs) [Doersch 2021; Kingma and Welling 2022]. These explicitly learn the probability density function of the distribution by maximizing the likelihood of observed data (or an approximation for VAEs), and their parameters hold an interpretable probabilistic significance. However, due to the issue of intractable partition function, their application is either limited to specific architectures (for autoregressive and normalizing flow models), or rely on an adapted objective function (evidence-lower bound for VAEs and contrastive divergence in EBMs). In contrast, implicit generative models, as generative adversarial networks (GANs) [Goodfellow et al. 2014; Jolicoeur-Martineau 2018], represent the probability distribution only implicitly as they focus on modelling the sampling process. For instance, to go on with GANs, they

utilize a game-theoretic framework where a generator and discriminator respectively learn to produce and differentiate between real and synthetic data. As a consequence, the parameters of implicit generative models lack a direct probabilistic interpretation, as the emphasis lies on generating realistic samples rather than explicitly defining the probability distribution. Although they avoid the pitfall of the normalizing term, GANs often grapple with issues related to training stability as they require adversarial training.

In this context, we introduce score-based generative models as an alternative to the aforementioned models. We will define in Section 2 the mathematical object and place it in a generative context using Langevin dynamics. In Section 3, we will present several techniques and results making possible the estimation of this score function (i.e., score matching). We will state in Section 4 the two main contributions discussed in Song and Ermon [2019], namely the addition of Gaussian noise on the data and annealed Langevin dynamics, an adapted sampling algorithm. Finally, we will make some comments about the presented approach in Section 5 and conclude in Section 6.

2 Score-Based Generative Modeling

We consider a parametrized density model $p_\theta(\mathbf{x})$. We want to estimate the vector of parameters θ such that $p_\theta(\mathbf{x}) \approx p_{data}(\mathbf{x})$. Using energy-based formulation, the density can be written as follows:

$$p_\theta(\mathbf{x}) = \frac{e^{q_\theta(\mathbf{x})}}{Z(\theta)}$$

where $Z(\theta) = \int_{\mathbf{x}} e^{q_\theta(\mathbf{x})} d\mathbf{x}$ is the (usually) intractable normalizing constant.

2.1 Score function

We define (*Stein) score* [Liu et al. 2016] function as follows:

$$\mathbf{s}_\theta(\mathbf{x}) \triangleq \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}) \tag{1}$$

We emphasize that this score function is different from Fisher's Information Score which we more commonly come across in statistics: the latter is obtained by taking the gradient of the log-likelihood function w.r.t. the parameters θ, whereas Stein's score is computed by differentiating w.r.t. the data. We can then write:

$$\mathbf{s}_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} q_\theta(\mathbf{x}) - \underbrace{\nabla_{\mathbf{x}} \log Z(\theta)}_{=0}$$

Therefore, the pivotal advantage of score-based models lies in their ability to circumvent the intractable normalizing term.

2.2 Langevin Monte-Carlo Sampling

We now place the score function in a context of generative modeling. Since we no longer estimate distribution densities, we need to use an adapted method in order to sample from the distribution knowing solely its score function.

In the statistical community, this matter has been extensively studied as part of Monte-Carlo Markov Chain (MCMC) methods [Roberts and Rosenthal 1998; Welling and Teh 2011]. We recall that these methods aim at neatly-building stochastic processes under the form of ergodic Markov chains, for which the stationary distribution is the target distribution from which we want to sample [Roberts and Rosenthal 2004]. Therefore, by letting such-built chains run for enough time, we obtain samples from the target distribution. Among the numerous algorithms available, a range of methods have been developed with the objective of incorporating the structure of the target density in the process. Building on the well-known idea behind gradient ascent, these algorithms have integrated the gradient of the distribution in the MCMC procedure: this term will have the effect of pushing the Markov chain towards values having higher density. Besides the mere intuition, such processes relies on strong theoretical justifications based on Langevin's diffusions.

We define the overdamped Langevin Itô diffusion as follows:

$$d\mathbf{x}_t \triangleq \frac{1}{2}\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}_t) dt + d\mathbf{B}_t \tag{2}$$

where $\mathbf{B}_t$ is a Brownian motion. It has been proven that the stationary distribution of this diffusion is the density p_θ [Roberts and Tweedie 1996].

2.2.1 Metropolis-Adjusted Langevin Algorithm.

In practice, we consider discrete-time approximations of this diffusion to approach its paths. Using Euler–Maruyama method, we obtain the following discrete-time stochastic process:

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \frac{\epsilon}{2}\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}_{t-1}) + \sqrt{\epsilon}\mathbf{z}_t \tag{3}$$

where $\mathbf{z}_t \sim \mathcal{N}(0, I_d)$ is a standard Gaussian noise and $\epsilon > 0$ is the step size. Because of the bias introduced by the discrete-time approximation, we note that (3) no longer targets p_θ. To overcome this shift, methods have been proposed to correct the resolution by adding a Metropolis-Hasting step: the update rule becomes a proposal step followed by an acceptance step. The proposal step is accepted with probability:

$$\alpha(\mathbf{x}_{t-1}, \mathbf{x}_t) = \min\left(1, \frac{p_\theta(\mathbf{x}_t) q(\mathbf{x}_{t-1}|\mathbf{x}_t)}{p_\theta(\mathbf{x}_{t-1}) q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right) \tag{4}$$

where $q(\mathbf{x}_t|\mathbf{x}_{t-1}) \propto \exp\left(-\frac{1}{2\epsilon}||\mathbf{x}_t - \mathbf{x}_{t-1} - \frac{\epsilon}{2}\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}_{t-1})||_2^2\right)$ is the proposal distribution. We note here that as in classical Metropolis-Hasting, we can proceed to the acceptance step even when solely knowing the density up to a constant as it will vanish in the quotient.

This method is known as Metropolis-adjusted Langevin algorithm (MALA) [Roberts and Tweedie 1996].

We performed numerical experiments to simulate the MALA and compare it with the unnadjusted algorithm. The toy distributions used are described in Appendix A, and the parameters used are given in Appendix B.

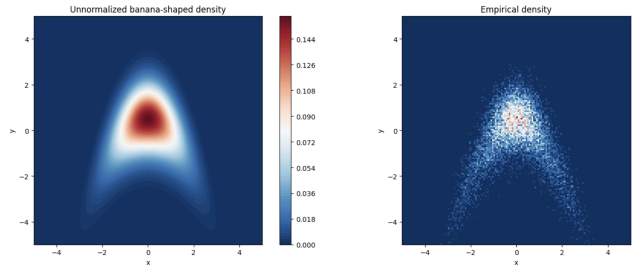

Figure 1. Left: Original Unnormalized Density; **Right**: Estimated Density with Metropolis-Adjusted Langevin Algorithm using the exact score functions.

2.2.2 Unadjusted Langevin Algorithm.

In the original paper, the Metropolis-Hasting step is discarded and the update is accepted at each iteration. As the bias fade out when $\epsilon \to 0$ and $t \to \infty$, this choice was explained by assuming the error negligible when ϵ is small and t is large. This equation is known as the Unadjusted Langevin Algorithm (ULA) and recent theoretical work has showed that it enjoyed convergence bounds without the correction step [Dalalyan 2017; De Bortoli et al. 2021; Durmus and Moulines 2017]. Numerical results obtained with this version of the algorithm matched really closely the ones with the acceptance step.

3 Score Matching

To take advantage of these sampling algorithms, we now turn to the estimation of the score function. While the numerical results have been obtained using the exact score functions, there is no reason why one could obtain such closed-form formulas when facing a dataset. We therefore need to adapt to our problem the classical statistical approach detailed in the introduction. We consider a set of N i.i.d. samples $\{\mathbf{x}_n\}_{1\leq n\leq N}$ drawn from the unknown data distribution $\mathbf{x} \sim p_{data}$. We want to estimate the score function $\mathbf{s}_\theta : \mathbb{R}^d \to \mathbb{R}^d$ such that $\mathbf{s}_\theta(\mathbf{x}) \approx \nabla_\mathbf{x} \log p_{data}(\mathbf{x})$. While we could use maximum likelihood for example to estimate θ when dealing with density functions, here we no longer compare probability distributions but rather vector fields. We thus introduce the Fisher divergence as a new objective function:

$$J_{ESM}(\theta) \triangleq \frac{1}{2}\mathbb{E}_{p_{data}(\mathbf{x})}\left[||\mathrm{s}_\theta(\mathbf{x}) - \nabla_\mathbf{x} \log p_{data}(\mathbf{x})||_2^2\right] \quad (5)$$

This approach is called *Score Matching*, and we will discuss three different methods to estimate this score function. Numerical estimations can be found in Appendix (**??**).

3.1 Implicit Score Matching

The first main result when it comes to score matching is the following theorem [Hyvärinen and Dayan 2005]:

Theorem 3.1 (Implicit Score Matching)**.** *Assuming that the score function is differentiable and under some regularity conditions ensuring existence of all the terms, the Fisher divergence (5) can be written as:*

$$J_{ISM}(\theta) \triangleq \mathbb{E}_{p_{data}(\mathbf{x})}\left[tr\left(\nabla_\mathbf{x} s_\theta(\mathbf{x})\right) + \frac{1}{2}||s_\theta(\mathbf{x})||_2^2\right] + c \quad (6)$$

where c is a constant term that does not depend on θ.

This formulation is quite spectacular as it yields an *Implicit Score Matching* (ISM) objective equivalent to the *Explicit Score Matching* (5) that no longer depends on the score of the unknown data distribution $p_{data}(\mathbf{x})$. In practice, the computation of the Jacobian $\nabla_\mathbf{x} s_\theta(\mathbf{x})$ becomes really expensive when facing high-dimensional data (e.g., images) thereby limiting this approach.

3.2 Denoising Score Matching

To overcome the issue of high-dimensional data, Vincent [2011] exhibits an unexpected connection between score matching and denoising auto-encoders. Denoising autoencoders are a particular type of auto-encoders that are trained to reconstruct the original data from a corrupted version of it. We thus consider pairs of clean and corrupted data $(\mathbf{x}, \tilde{\mathbf{x}})$ where $\tilde{\mathbf{x}}$ is obtained by adding Gaussian noise $\tilde{\mathbf{x}} \sim \mathcal{N}(\mathbf{x}, \sigma^2 I_d)$. This density can be written using the multivariate isotropic Gaussian kernel with variance σ^2:

$$q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) \triangleq \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{1}{2\sigma^2}||\tilde{\mathbf{x}} - \mathbf{x}||_2^2\right) \quad (7)$$

Using kernel density estimation, we can define the perturbed data distribution as follows:

$$q_\sigma(\tilde{\mathbf{x}}) \triangleq \int_\mathbf{x} q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) p_{data}(\mathbf{x}) d\mathbf{x} = \int_\mathbf{x} q_\sigma(\tilde{\mathbf{x}}, \mathbf{x}) dx \quad (8)$$

The explicit score matching objective for the perturbed data distribution thus write as follows:

$$J_{ESM_\sigma}(\theta) \triangleq \frac{1}{2}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}})}\left[||\mathrm{s}_\theta(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})||_2^2\right] \quad (9)$$

Vincent [2011] then proceeds to demonstrate the following result:

Theorem 3.2 (Denoising Score Matching)**.** *Under weak conditions of log-differentiability, the explicit score matching objective for the perturbed data distribution (9) is equivalent to the following Denoising Score Matching (DSM) objective:*

$$J_{DSM_\sigma}(\theta) = \frac{1}{2}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}},\mathbf{x})}\left[||s_\theta(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})||_2^2\right] \quad (10)$$

over the joint distribution of clean and corrupted data as previously expressed in (8).

While the score of perturbed data distribution $\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})$ has no reason to be explicitly known, the considered quantity in Denoising Score Matching possesses the appreciable following property:

$$\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) = -\frac{1}{\sigma^2}(\tilde{\mathbf{x}} - \mathbf{x}) \quad (11)$$

We can thus write the Denoising Score Matching objective as follows:

$$J_{DSM_\sigma}(\theta) = \frac{1}{2}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}},\mathbf{x})}\left[||\mathbf{s}_\theta(\tilde{\mathbf{x}}) + \frac{1}{\sigma^2}(\tilde{\mathbf{x}} - \mathbf{x})||_2^2\right] \tag{12}$$

Finally, Vincent [2011] shows that performing score matching on the Parzen density estimate defined in (8) is equivalent to training a denoising autoencoder on data with conditional density defined in (7). The main advantage of this method is that it is computationally efficient as it only requires to train a denoising auto-encoder. However, choosing σ must be done cautiously: to ensure that the perturbed data distribution is close to the original data distribution, σ must be sufficiently small.

3.3 Sliced Score Matching

The last score matching technique considered in the original article relies on Song et al. [2020]. In this paper, the authors circumvent the heavy computation required in implicit score matching with a technique called *sliced score matching* which consists in solely computing projections of the Jacobian along random vectors. The objective then writes as follows:

$$J_{SSM}(\theta) \triangleq \mathbb{E}_{p_\mathbf{v}}\mathbb{E}_{p_{data}(\mathbf{x})}\left[||\mathbf{v}^T\nabla_\mathbf{x}\mathbf{s}_\theta(\mathbf{x})\mathbf{v} + \frac{1}{2}||\mathbf{s}_\theta(\mathbf{x})||_2^2\right] \tag{13}$$

where $p_\mathbf{v}$ is a distribution over random vectors (e.g., $\mathbf{v} \sim \mathcal{N}(0, I_d)$). This estimator has been proven to be consistent and asymptotically normal. While this technique has the advantage of approximating the unperturbed data distribution, the authors mention that it is more computationally expensive than denoising score matching.

4 Improving Score-Matching with Perturbed Data and Annealed Langevin Dynamics

Despite clever techniques for approximating the score function of unknown data distribution, Song and Ermon [2019] highlight several shortcomings with the generating process as it is.

4.1 Noise Conditional Score Network

The first consideration made builds on a discussion element we mentioned when defining denoising score matching: the choice of σ^2. On one hand, the objective defined by denoising score matching needs a small value for σ^2 to ensure that the perturbed data distribution is close to the original data distribution. On the other hand, as noise is added to the data, the perturbed data distribution will be more spread out, leading to better estimations of the score function in the low-densities regions of the support of the probability distribution. Since the score matching is made through empirical mean, we would not be able otherwise to estimate the score function in these scarcely-filled regions (Figure **??**).

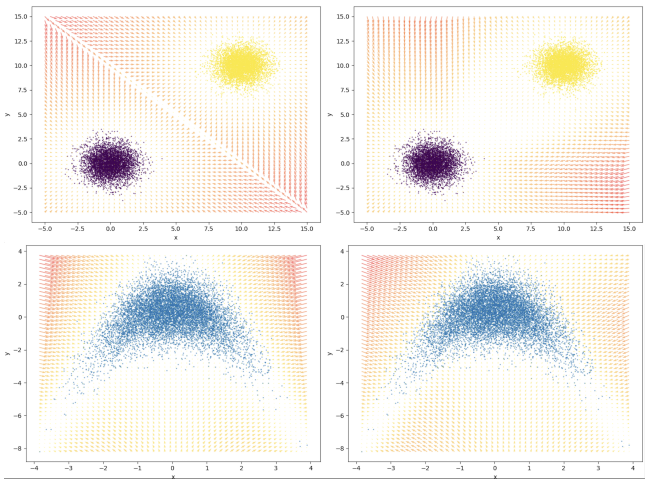

Figure 2. Left: $\nabla_\mathbf{x}\log p_{data}(\mathbf{x})$; **Right**: $\mathbf{s}_\theta(\mathbf{x})$ (Sliced Score Matching). The scores have been normalized and are encoded using a warm colormap: warmer color implies a higher value for the norm of score. As one gets closer to the data point cloud, the scores' norm decrease and the estimation obtained by score matching improves.

Secondly, it has been shown that most of the high dimensional data distributions are actually concentrated on a low dimensional manifold [Bengio et al. 2013; Cayton 2005; McInnes et al. 2018], itself embedded in the high-dimensional space. This phenomenon is known as the *manifold hypothesis*. In our context, this would mean that the score function is not defined in the whole space but rather on this lower-dimensional manifold. This is problematic as the hypotheses allowing to go from *explicit* score matching to *implicit* score matching are no longer verified. Indeed, the score function is no longer differentiable everywhere and the Jacobian is no longer defined.

To simultaneously overcome these issues, Song and Ermon [2019] propose to perturb the data with a geometric decreasing sequence of Gaussian noise defined as follows:

$$\{\sigma_i \in \mathbb{R}_+^*\}_{1\leq i\leq L} \text{ s.t. } \frac{\sigma_1}{\sigma_2} = \cdots = \frac{\sigma_{L-1}}{\sigma_L} > 1 \tag{14}$$

The rationale behind this choice is that the first levels of noise will allow to estimate the score function in the low-densities regions of the distribution support, while the last levels of noise will become sufficiently small to allow to accurately approximate the unknown score function. As each noise level will require a different score function, it seems impractical to train a different model for each level of noise as we might need a large number of noise levels to properly estimate the score function. The authors then propose to train a *Noise Conditional Score Network* (NCSN) to simultaneously estimate the score functions for each level of noise. The NCSN takes as input a data point and a level of noise and outputs an estimation of the score function.

The Denoising Score Matching objective is then adapted to optimize over all noise levels:

$$J_{NCSN}(\theta) \triangleq \frac{1}{L}\sum_{i=1}^{L} \mathbb{E}_{q_{\sigma_i}(\tilde{\mathbf{x}},\mathbf{x})}\left[||\mathrm{s}_\theta(\tilde{\mathbf{x}},\sigma_i) + \frac{1}{\sigma_i^2}(\tilde{\mathbf{x}}-\mathbf{x})||_2^2\right] \quad (15)$$

In practice, the authors have introduced a term correcting the magnitude orders, yielding the following objective:

$$J_{NCSN}(\theta) \triangleq \frac{1}{L}\sum_{i=1}^{L} \lambda(\sigma_i)\mathbb{E}_{q_{\sigma_i}(\tilde{\mathbf{x}},\mathbf{x})}\left[||\mathrm{s}_\theta(\tilde{\mathbf{x}},\sigma_i) + \frac{(\tilde{\mathbf{x}}-\mathbf{x})}{\sigma_i^2}||_2^2\right] \quad (16)$$

where $\lambda(\sigma_i) = \sigma_i^2$. The authors have empirically observed that this correction term allows to obtain better results and improves the convergence of the algorithm.

4.2 Annealed Langevin Dynamics

Secondly, as noticed when reproducing experiments (Appendix D.1, Figure 8), the Langevin dynamics is not able to properly estimate the mixing weights in the case of multimodal distributions. This is no news as this issue was already known in the context of MCMC methods: a larger step size ϵ will lead to a better mixing, allowing the chain to "jump" between modes, but will also lead to a less accurate approximation of the target distribution, with an acceptance rate that will tend to 0 in a high-dimensional setting. Conversely, a smaller step size will lead to a better approximation of the target distribution, but will also lead to a poor mixing as the chain won't be able to efficiently explore the support of the distribution. To bypass these issues, the idea of perturbing the data seems promising in this context as well: by adding noise to the samples, the chain will be able to explore the support of the distribution more efficiently, while still being able to approximate the target distribution. In this regard, Song and Ermon [2019] propose a modified version of the Langevin dynamics, called *annealed Langevin dynamics*. The algorithm will sequentially run L different chains, one for each noise level σ_i, and use the samples obtained at the previous level of noise as initial states for the next level. Concomitantly, the step size used in the Langevin dynamics will be decreased as the noise level increases (we recall that a large noise level actually corresponds to a small perturbation of the data, c.f. (14)). This algorithm is called *annealed* as a parallel can be made with a cooling process: the initial distribution is heated up by adding noise to the data, making it smoother over an extended support, and then gradually cooled down as the noise level decreases, allowing to obtain samples from the original distribution.

Algorithm 1 Annealed Langevin dynamics.

Require: $\{\sigma_i\}_{i=1}^{L}, \epsilon, T$.
Initialize $\tilde{\mathbf{x}}_0$
for $i \leftarrow 1$ to L **do**
  $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2/\sigma_L^2$ ▷ α_i is the step size.
  for $t \leftarrow 1$ to T **do**
    Draw $\mathbf{z}_t \sim \mathcal{N}(0, I)$
    $\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \frac{\alpha_i}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t-1},\sigma_i) + \sqrt{\alpha_i}\,\mathbf{z}_t$
  end for
  $\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$
end for
return $\tilde{\mathbf{x}}_T$

The authors have observed that this algorithm allows to obtain better results, especially when dealing with multimodal distributions.

5 Discussions and Experiments

Reflecting on the proposed work, we can make several comments. First of all, building on the theoretical contributions brought by Hyvärinen and Dayan [2005] in a first time and Vincent [2011] in a second time, Song and Ermon [2019] have proposed an innovative approach to generative modeling by leveraging the strengths of score-based models. This approach has the advantage of circumventing the intractable normalizing term, while still being able to sample from the target distribution after having estimated a score function. Facing issues to sample efficiently from image distributions, the authors have proposed to perturb the data with Gaussian noise of various magnitudes, and to train a Noise Conditional Score Network to simultaneously estimate the score functions for each level of noise. This approach has the advantage of being computationally efficient, while still being able to explore the distribution support. Finally, the authors have proposed to use an annealed version of Langevin dynamics, which allows to obtain better results, especially when dealing with multimodal distributions. When the paper was published, the authors showed that their approach obtained state-of-the-art results on the CIFAR-10 dataset, generated very realistic samples from the MNIST and CelebA datasets and was able to learn meaningful representations when performing image inpainting. At the time, GANs were the ones-to-beat in terms of image generation. The authors therefore stress the fact that their method is able to produce samples of comparable quality while not requiring any adversarial training, making it supposedly less prone to training instability.

During our work, we designed experiments so as to assert the justifications and choices made by the authors. In the following, we extend on three aspects: the choice of the noise parameters, their and the computational aspect of the proposed approach.

5.1 Noise levels

In Song and Ermon [2019], some recommendations are made regarding the choice of the noise levels. The experiment they describe for images with pixel values between 0 and 1 used $L = 10$, with $\sigma_1 = 1$ and $\sigma_{10} = 0.01$.

5.1.1 Number of levels. First of all, the authors recommend to choose the noise levels such that $\frac{\sigma_1}{\sigma_2} = \cdots = \frac{\sigma_{L-1}}{\sigma_L} > 1$. This choice is motivated by the fact that the noise levels should be chosen diverse enough to allow to estimate the score function in the low-densities regions of the distribution support, while still being able to accurately approximate the unknown score function. However, the authors do not provide any theoretical justification on how to appropriately select the number of noise levels. Using the advised range of noise, we considered several number of levels $L = \{1, 2, 5, 10, 20\}$ for the NCSN and compared the sampling performances. Our results show that the sampling improves up until $L = 10$, but the quality of our score estimation for $L = 20$ deteriorated considerably, leading to an useless sampling.

5.1.2 Range of levels. Another consideration we had concerns the value of the noise levels themselves. In early experiments, we have observed that the upper bound proposed did not necessarily seem optimal. Indeed, even by adding a Gaussian noise of variance $\sigma_1 = 1$, there were still areas of the support that were poorly estimated with denoising score matching. We considered a different approach deriving from the class of Interacting MCMC methods, and more specifically from an adapted version of the Parallel Tempering algorithm [Geyer et al. 1991]. This algorithm is based on the idea of running several chains in parallel, each of them targeting a different temperature. The chains are then allowed to stochastically swap their temperatures according to an acceptance criterion, allowing to explore the distribution support more efficiently. Usually, these temperatures (similar to an amount of noise) can be quite high. We tried to adapt the original algorithm so as to take into account the score for the candidate proposals. In this case, the method seemed however limited as our implementation needs to be reworked: the score seemed to impact too strongly the proposals, and while proportions between clusters in multimodal distribution seemed to be well-estimated, samples seemed constrained along the gradient direction in the Gaussian Mixture Model.

5.1.3 Scaled loss. A second parameter linked to the noise levels is the scaling factor introduced in the loss function. The authors have empirically observed that the loss function was more stable when using a scaling factor $\lambda(\sigma_i) = \sigma_i^2$. This choice was motivated by the fact that it allowed to observe orders of magnitude for $\lambda(\sigma_i)\mathbb{E}_{q_{\sigma_i}(\tilde{\mathbf{x}},\mathbf{x})}\left[||s_\theta(\tilde{\mathbf{x}}, \sigma_i) + \frac{(\tilde{\mathbf{x}}-\mathbf{x})}{\sigma_i^2}||_2^2\right]$ that were similar for all noise levels. However, one might consider to weight the loss function differently, with the idea of penalizing more the noise levels that are closer to the original data distribution. This could be done by introducing a scaling factor $\lambda(\sigma_i) = \frac{1}{\sigma_i^2}$ for example. This would allow to better estimate the score function in the high-densities regions of the distribution support.

5.2 Discrete distributions

We considered an extension of the generating process described in the papers to discrete distributions. To do so we considered such a discrete distribution, and widened its support by defining it as a Gaussian mixture with as many clusters as elements in the discrete support. Each cluster was centered on the latter, and we associated isotropic covariance matrices to each with a variance parameter of 0.01 to make sure to have very narrow clusters similar to a Dirac centered at each mean. While the score seemed to be well-estimated using a NCSN, we however did not manage to sample efficiently from the distribution using the considered algorithms. Numerical results can be found in Appendix C.3 for the score, and in the notebooks for the sampling.

6 Conclusion

To conclude, we have presented the main ideas behind score-based generative modeling. Rendered possible by results presented by Vincent [2011] (and Hyvärinen and Dayan [2005] before him), this technique was then made effective thanks to the considerable contributions of Song and Ermon [2019]. We have then discussed several aspects of the proposed approach, and supported our remarks with numerical experiments.

The authors mention having trained their models on powerful GPUs. In our experiments, we were unfortunately short of such resources, and thus restrained ourselves to toy experiments. The drawback from this is that many considerations and potential intuitions could not be supported. Indeed, the curse of dimensionality renders useless considerations made on toy data, as unexpected phenomena arise when dealing with high-dimensional data such as images.

It is important to note that, since 2019, the authors wrote two follow-up papers for this article that have known great recognition in the community. In Song and Ermon [2020], the authors have addressed several issues related to the training process. They provided a series of recommendations for hyperparameter tuning that showed significant improvements in terms of sample quality, especially for high-quality images (up to 256×256 resolution). In Song et al. [2021], the authors have proposed a way to considerably improve the sampling process: instead of a discrete diffusion process, they propose a method that consists in reversing a stochastic differential equation to obtain a continuous-time diffusion process. This method is equivalent to having an infinite number of noise's level, and thus allows to sample images from pure noise.

This work was realised as part of the *Probabilistic Graphical Models and Deep Generative Models* class given by Prs. Pierre Latouche and Pierre-Alexandre Mattei. (MVA Master, December 2023).

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A system for large-scale machine learning. arXiv:1605.08695 [cs.DC]

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein Generative Adversarial Networks. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 214–223. https://proceedings.mlr.press/v70/arjovsky17a.html

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation Learning: A Review and New Perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828. http://arxiv.org/abs/1206.5538 cite arxiv:1206.5538.

Lawrence Cayton. 2005. Algorithms for manifold learning. https://api.semanticscholar.org/CorpusID:408872

Caroline Chan, Shiry Ginosar, Tinghui Zhou, and Alexei A Efros. 2019. Everybody Dance Now. In *IEEE International Conference on Computer Vision (ICCV)*.

Benjamin Charlier, Jean Feydy, Joan Alexis Glaunes, François-David Collin, and Ghislain Durif. 2021. Kernel operations on the gpu, with autodiff, without memory overflows. *The Journal of Machine Learning Research* 22, 1 (2021), 3457–3462.

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. 2023. Simple and Controllable Music Generation. arXiv:2306.05284 [cs.SD]

Arnak S Dalalyan. 2017. Theoretical guarantees for approximate sampling from smooth and log-concave densities. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 79, 3 (2017), 651–676.

Valentin De Bortoli, Alain Durmus, Marcelo Pereyra, and Ana F Vidal. 2021. Efficient stochastic optimisation by unadjusted Langevin Monte Carlo: Application to maximum marginal likelihood and empirical Bayesian estimation. *Statistics and Computing* 31 (2021), 1–18.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

Laurent Dinh, David Krueger, and Yoshua Bengio. 2014. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516* (2014).

Carl Doersch. 2021. Tutorial on Variational Autoencoders. arXiv:1606.05908 [stat.ML]

Alain Durmus and Eric Moulines. 2017. Nonasymptotic convergence analysis for the unadjusted Langevin algorithm. (2017).

Jean Feydy. 2020. Geometric data analysis, beyond convolutions. *Applied Mathematics* (2020).

Charles J Geyer et al. 1991. Computing science and statistics: Proceedings of the 23rd Symposium on the Interface. *American Statistical Association, New York* 156 (1991).

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Networks. (June 2014). https://arxiv.org/abs/1406.2661

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* 33 (2020), 6840–6851.

Aapo Hyvärinen and Peter Dayan. 2005. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research* 6, 4 (2005).

Alexia Jolicoeur-Martineau. 2018. The relativistic discriminator: a key element missing from standard GAN. arXiv:1807.00734 [cs.LG]

Diederik P Kingma and Max Welling. 2022. Auto-Encoding Variational Bayes. arXiv:1312.6114 [stat.ML]

Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. 2023. AudioGen: Textually Guided Audio Generation. arXiv:2209.15352 [cs.SD]

Hugo Larochelle and Iain Murray. 2011. The neural autoregressive distribution estimator. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 29–37.

Qiang Liu, Jason D. Lee, and Michael I. Jordan. 2016. A Kernelized Stein Discrepancy for Goodness-of-fit Tests and Model Evaluation. arXiv:1602.03253 [stat.ML]

L. McInnes, J. Healy, and J. Melville. 2018. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints* (Feb. 2018). arXiv:1802.03426 [stat.ML]

Andrew Ng and Michael Jordan. 2001. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems* 14 (2001).

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. (2017).

Gabriel Peyré, Marco Cuturi, et al. 2019. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning* 11, 5-6 (2019), 355–607.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-Shot Text-to-Image Generation. *CoRR* abs/2102.12092 (2021). arXiv:2102.12092 https://arxiv.org/abs/2102.12092

Danilo Rezende and Shakir Mohamed. 2015. Variational inference with normalizing flows. In *International conference on machine learning*. PMLR, 1530–1538.

Gareth O. Roberts and Jeffrey S. Rosenthal. 1998. Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 60 (1998). https://api.semanticscholar.org/CorpusID:5831882

Gareth O. Roberts and Jeffrey S. Rosenthal. 2004. General state space Markov chains and MCMC algorithms. *Probability Surveys* 1, none (2004), 20 – 71. https://doi.org/10.1214/154957804100000024

Gareth O Roberts and Richard L Tweedie. 1996. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli* (1996), 341–363.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100* (2022).

Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. 2019. Singan: Learning a generative model from a single natural image. In *Proceedings of the IEEE/CVF international conference on computer vision*. 4570–4580.

Yang Song and Stefano Ermon. 2019. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems* 32 (2019).

Yang Song and Stefano Ermon. 2020. Improved techniques for training score-based generative models. *Advances in neural information processing systems* 33 (2020), 12438–12448.

Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. 2020. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*. PMLR, 574–584.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-Based Generative Modeling

through Stochastic Differential Equations. (2021). https://openreview.net/forum?id=PxTIG12RRHS

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. http://arxiv.org/abs/2302.13971 cite arxiv:2302.13971.

Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. 2016. Pixel recurrent neural networks. In *International conference on machine learning*. PMLR, 1747–1756.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

Pascal Vincent. 2011. A connection between score matching and denoising autoencoders. *Neural computation* 23, 7 (2011), 1661–1674.

Max Welling and Yee W Teh. 2011. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*. 681–688.

A Stein's Score for Toy Distributions

Some developments are made for probability distribution defined on $\mathbb{R}^d$. However, in practice we restrained ourselves to the case $d = 2$ for simplicity and visualisation purposes.

A.1 Gaussian Mixture Model

Let p be the density of a Gaussian Mixture composed of c clusters. We note $\theta = (\mu_i, \Sigma_i)_{1\leq i\leq c}$ the parameters, and $\alpha = (\alpha_1, \ldots, \alpha_c)^T$ the vector of mixing weights lying in the standard $(c-1)$-simplex. We can then write:

$$p(x|\alpha, \theta) = \sum_{i=1}^{c} \alpha_i \mathcal{N}(x; \mu_i, \Sigma_i)$$

where $\mathcal{N}(x; \mu_k, \Sigma_k)$ denotes the multivariate Gaussian density with parameters μ_k and Σ_k evaluated at $x \in \mathbb{R}^d$.

$$\mathcal{N}(x; \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{d}{2}}} \frac{1}{|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

We consider fixed and known the parameters and the mixing weights. We proceed to explicit Stein's score for this probability distribution.

$$\nabla_x \log p(x) = \frac{p(x)^{-1}}{2\pi} \left[\sum_{i=1}^{c} -\frac{\alpha_i}{|\Sigma_i|^{1/2}} \Sigma_i^{-1}(x-\mu_i) \exp\left(-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)\right)\right] \tag{17}$$

A.2 "Banana"-Shaped Density

Let π be the target density of the considered probability distribution that we note $\mathcal{P}$. It is based on a Gaussian distribution in $\mathbb{R}^d$ with mean 0 and covariance matrix $\Sigma = \mathrm{diag}(\sigma_1^2, 1, ..., 1)$. This density defined on $\mathbb{R}^d$ is twisted by changing the second coordinate x_2 to $x_2 + b(x_1^2 - \sigma_1^2)$ for $b \in \mathbb{R}$. We denote $\mathcal{N}(x; \mu, \Sigma)$ the density function of the d-dimensional Gaussian with mean μ and covariance Σ. We thus have, up to a normalizing constant:

$$\forall x \in \mathbb{R}^d, \quad \pi(x) \propto \mathcal{N}\left(\left(x_1, x_2 + b(x_1^2 - \sigma_1^2), \ldots, x_d\right); \mu, \Sigma\right)$$

In order to sample from such a distribution, we introduce a few definitions and notations.

Definition A.1 (Push-forward measure [Peyré et al. 2019])**.** Let $\mathcal{M}(\mathcal{X})$ be the set of Radon measures on the space $\mathcal{X}$ and $C(\mathcal{X})$ the set of continuous functions of $\mathcal{X}$. For a continuous map $T : \mathcal{X} \mapsto \mathcal{Y}$, we define its push-forward operator $\beta = T_\# \alpha \in \mathcal{M}(\mathcal{Y})$ of some $\alpha \in \mathcal{M}(\mathcal{X})$ as the measure satisfying :

$$\forall h \in C(\mathcal{Y}), \quad \int_{\mathcal{Y}} h(y) d\beta(y) = \int_{\mathcal{X}} h(T(x)) d\alpha(x) \tag{18}$$

Equivalently, for any measurable set $B \subset \mathcal{Y}$, one has

$$\beta(B) = \alpha(\{x \in \mathcal{X} : T(x) \in B\}) = \alpha(T^{-1}(B)) \tag{19}$$

Corollary A.2 (Push-forward for multivariate densities [Peyré et al. 2019])**.** *Explicitly doing the change of variables in formula 18 for measures with densities $(\rho_\alpha, \rho_\beta)$ on $\mathbb{R}^d$ (assuming T is smooth and bijective) shows that a push-forward acts on densities linearly as a change of variables in the integration formula. Indeed, one has:*

$$\rho_\alpha(x) = |\det \partial T(x)| \rho_\beta(T(x)) \tag{20}$$

Circling back to our case, we have defined our target density as follows:

$$\pi(x) \propto T_\# \mathcal{N}(x) = |\det \nabla_x (T)^{-1}(x)| \times \mathcal{N} \circ (T)^{-1}(x) = \mathcal{N} \circ (T)^{-1}(x)$$

where

$$(T)^{-1}(x_1, x_2) = \begin{pmatrix} x_1 \\ x_2 + b(x_1^2 - \sigma_1^2) \end{pmatrix}, \quad \det \nabla_x (T)^{-1}(x_1, x_2) = \det \begin{pmatrix} 1 & 0 \\ 2b & 1 \end{pmatrix} = 1.$$

We can finally write the map T as follows:

$$\begin{aligned} T : \quad \mathbb{R}^2 &\to \mathbb{R}^2 \\ (x, y) &\mapsto \left(x, y - b(x^2 + \sigma_1^2)\right) \end{aligned}$$

In practice, this expression of T allows us to obtain samples from the "banana" distribution $\mathcal{P}$ by applying T to Gaussian samples. Formally, for $X \sim \mathcal{N}\left(0, \text{diag}(\sigma_1^2, 1)\right)$, we have:

$$T \circ \mathcal{N}(x) \sim \mathcal{P}.$$

As extensively detailed in the core part of this work, score-based models do not need to have a closed-form expression of the density: it is sufficient here to know it up to a normalizing term. We can then proceed to explicit Stein's score for this probability distribution for $d = 2$. First, we have:

$$\begin{aligned}\log \phi(x) &\propto -\log 2\pi - \frac{1}{2}\log|\Sigma| \left(-\frac{1}{2}(T^{-1}(x) - \mu)^T \Sigma^{-1}(T^{-1}(x) - \mu)\right) \\ &\propto -\log 2\pi - \frac{1}{2}\log|\Sigma| - \frac{1}{2}\left(\frac{(x_1 - \mu)^2}{\sigma_1^2} + \frac{(x_2 + b(x_1^2 - \sigma_1^2) - \mu)^2}{\sigma_2^2}\right)\end{aligned}$$

As a consequence, we can write:

$$\nabla_x \phi(x) = \begin{pmatrix} -\frac{(x_1 - \mu)}{\sigma_1^2} - \frac{2bx_1}{\sigma_2^2}(x_2 + b(x_1^2 - \sigma_1^2) - \mu) \\ -\frac{x_2 + b(x_1^2 - \sigma_1^2) - \mu}{\sigma_2^2} \end{pmatrix} \tag{21}$$

B Parametrization of numerical experiments

All the implementation details are made available at https://github.com/JuniorTonga/Score-matching-project-.

B.1 Toy Data

The datasets are created using the following parameters.

GMM:

- Number of samples: $N = 10000$
- Number of clusters: $c = 2$
- Mixing weights: $\alpha = (0.5, 0.5)$ for most, $\alpha = (0.2, 0.8)$ for highlighting the shortcoming of the MALA algorithm in estimating proportions
- Means: $\mu_1 = (0, 0)$, $\mu_2 = (10, 10)$
- Covariance matrices: $\Sigma_1 = \Sigma_2 = I_2$

Banana distribution:

- Number of samples: $N = 10000$
- Mean: $\mu = (0, 0)$
- Correlation parameter: $b = 0.5$
- Covariance matrix: $\Sigma = I_2$

Star distribution:

- Number of samples: $N = 10000$
- Number of clusters: $c = 250$
- Mixing weights: Chosen at random
- Center of the star: $m = (0, 0)$
- Discrete support: $E = \{(m_1 + r_i \cos\theta_i, m_2 + r_i \sin\theta_i) \in \mathbb{R}^2\}_{1 \le i \le c}$, where $\begin{cases} \theta_i = \frac{2\pi i}{c} \\ r_i = 1 + \sin(5\theta_i) \end{cases}$
- Means: $\mu = E$
- Scale: $s = 0.01$
- Covariance matrices: $\forall i \in [\![1, \ldots, c]\!], \quad \Sigma_i = s \times I_2$

The means are determined using polar coordinates, ensuring an even distribution along the star's arms. When sampling from this GMM, the small covariances cause the samples to cluster near these means, visually creating a distinct star shape.

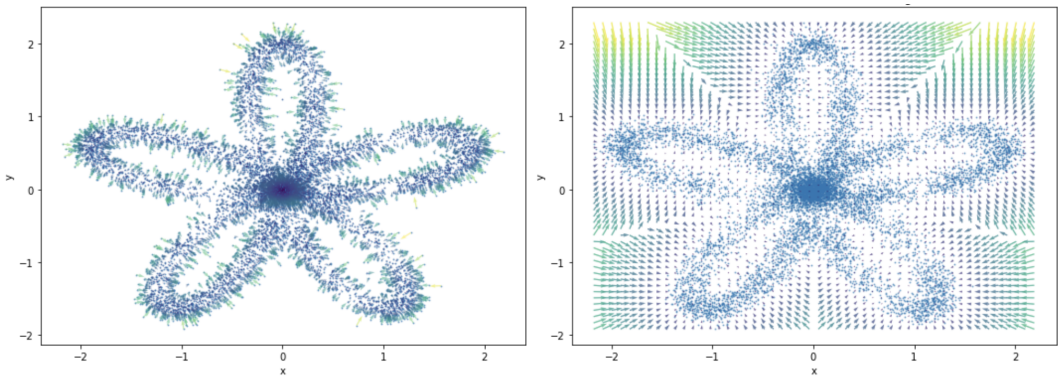

Figure 3. Star-shaped distribution

B.2 Score Networks

For Score Matching, we implemented two small neural networks (score networks) following the details given in the original paper [Song and Ermon 2019].

More specifically, the simplest version is a fully-connected neural network with 2 hidden layers of 128 neurons each, and Softplus activation functions. It is used when for basic Score Matching (Implicit, Denoising and Sliced Score Matching).

We also implemented a Noise-Conditional Score Network that consists in two hidden Conditional Linear layers of 128 neurons each (conditioned on noise level), and Softplus activation functions. The conditional layer consists of an unbiased linear layer and an embedding layer.

C Score Matching

In the core of the article, we presented some results obtained with Sliced Score Matching. We gather here the results obtained when training the score network with implicit score matching, denoising score matching, and with a noise-conditional score network.

C.1 Implicit Score Matching

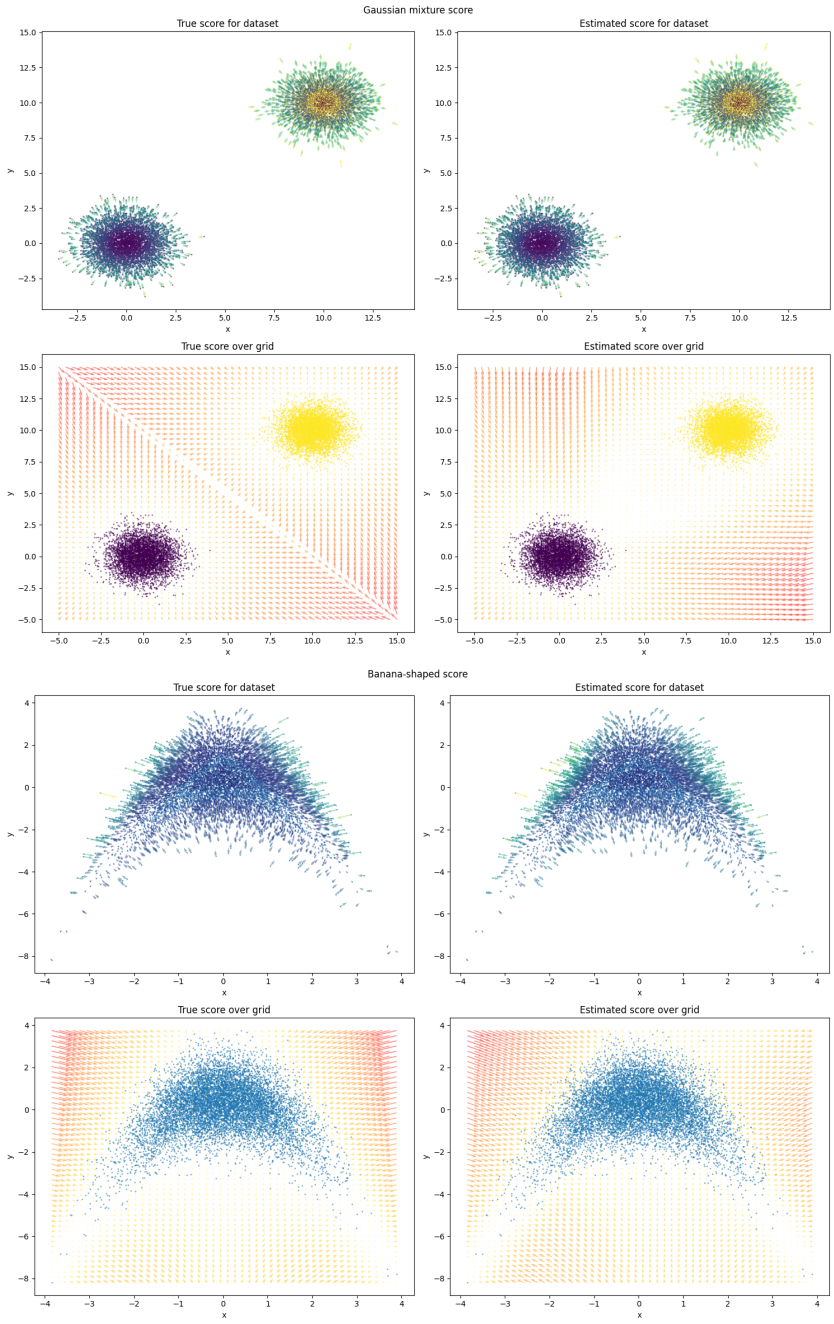

Figure 4. Left: $\nabla_{\mathbf{x}} \log p_{data}(\mathbf{x})$; **Right**: $\mathbf{s}_\theta(\mathbf{x})$ (ISM). The scores have been normalized and are encoded using a warm colormap: warmer color implies a higher value for the norm of score. As one gets closer to the data point cloud, the scores' norm decrease and the estimation obtained by score matching improves.

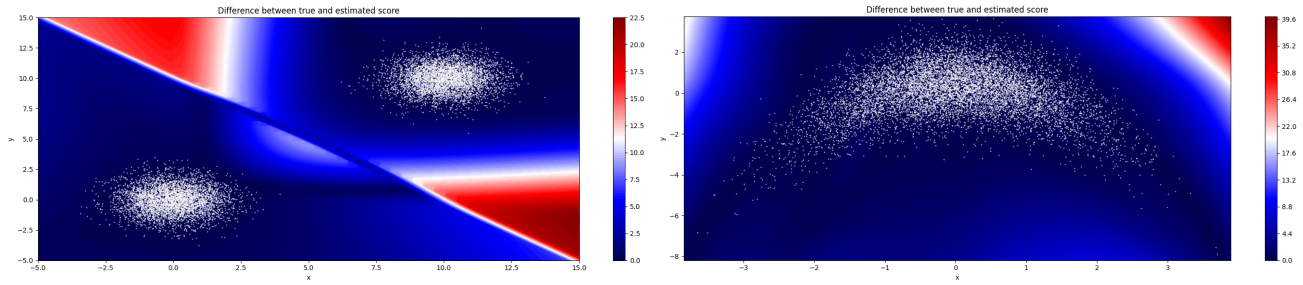

Figure 5. $||\nabla_{\mathbf{x}} \log p_{data}(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x})||_2^2$. (ISM). The norms are encoded using a warm heatmap: warmer color implies a higher value for the norm of the difference between the scores. Supporting the observation made in Figure 2, score matching is most accurate when approaching high-density regions (data represented as white point cloud).

C.2 Denoising Score Matching for $\sigma^2 = 1$ and $\sigma^2 = 0.2$

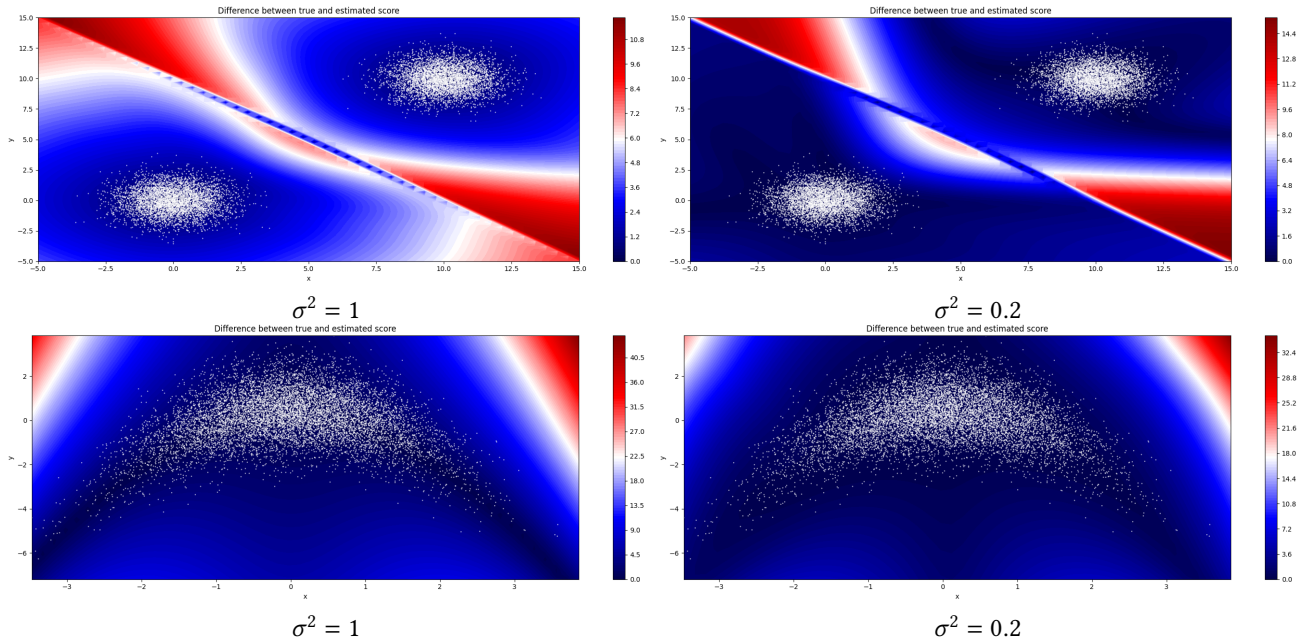

Figure 6. $||\nabla_{\mathbf{x}} \log p_{data}(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x})||_2^2$. (DSM). The norms are encoded using a warm heatmap: warmer color implies a higher value for the norm of the difference between the scores (data represented as white point cloud). Taking into account the slight changes in scale, we notice that as compared to a small noise, a higher level of noise in the Gaussian mixture case leads to a less precise score estimation overall but allows improvements in low-density areas. It seems less effective for the second distribution though.

C.3 Noise-Conditional Score Matching

Following the implementation advice given by Song and Ermon [2019], we trained a Noise-Conditional Score Network with $L = 10$ levels of noise between $\sigma_1 = 1$ and $\sigma_L = 0.01$.

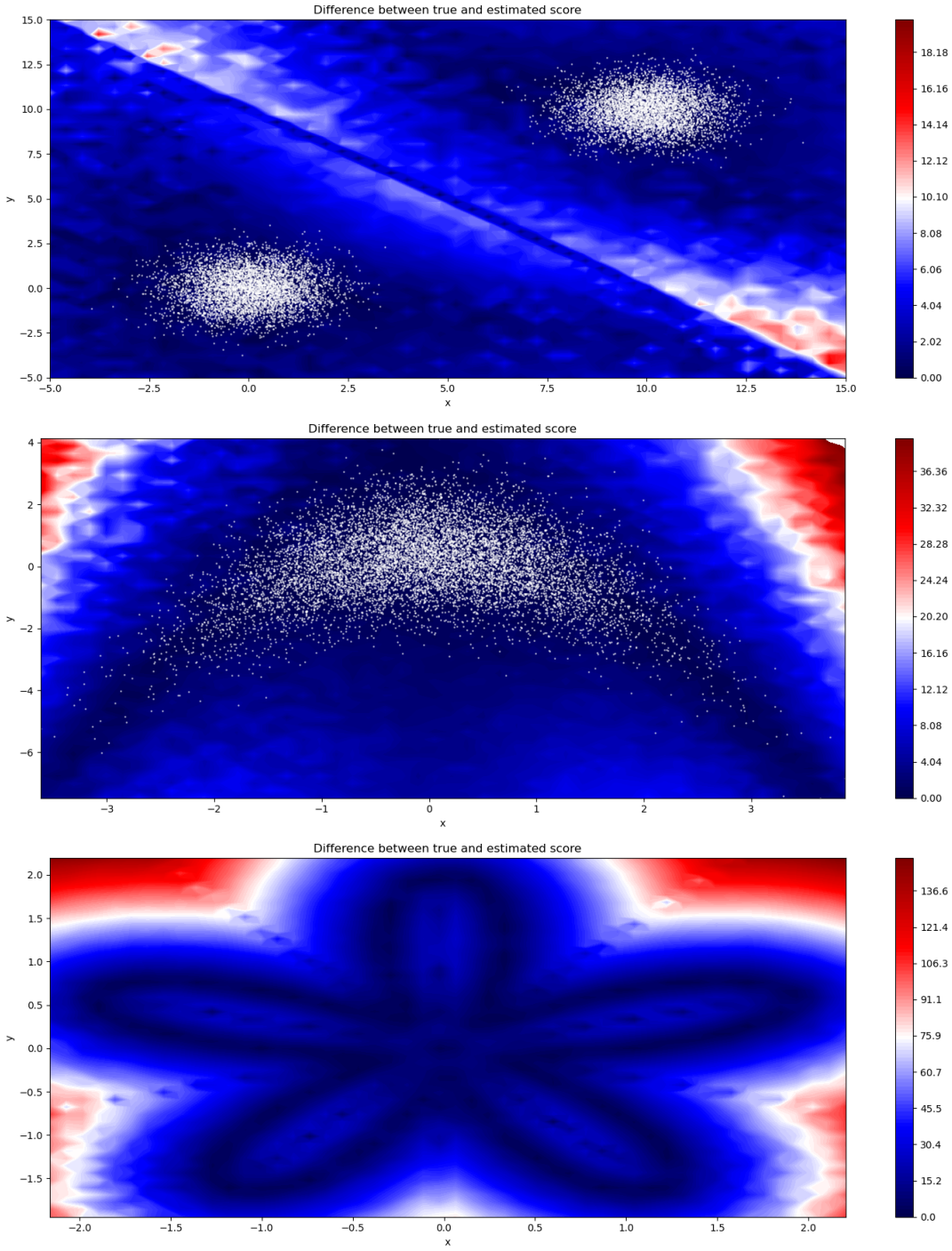

Figure 7. $||\nabla_{\mathbf{x}} \log p_{data}(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x})||_2^2$. (NCSN). The norms are encoded using a warm heatmap: warmer color implies a higher value for the norm of the difference between the scores (data represented as white point cloud).

D Sampling

D.1 Annealed Langevin

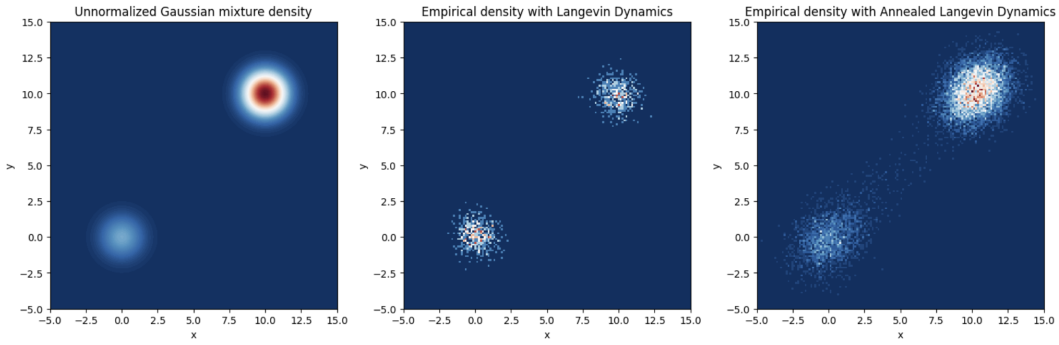

Figure 8. Left: Original Density; **Middle**: Estimated Density with Metropolis-Adjusted Langevin Algorithm using the exact score functions; **Right**: Estimated Density with Annealed Langevin algorithm using a NCSN. In accordance with Song and Ermon [2019], we observe that despite having added a Metropolis-Hasting correction, the Langevin dynamics cannot estimate properly the mixing weights in the case of multimodal distribution whereas the annealed version seem to better reconcile the original weights.

D.2 Impact of noise levels

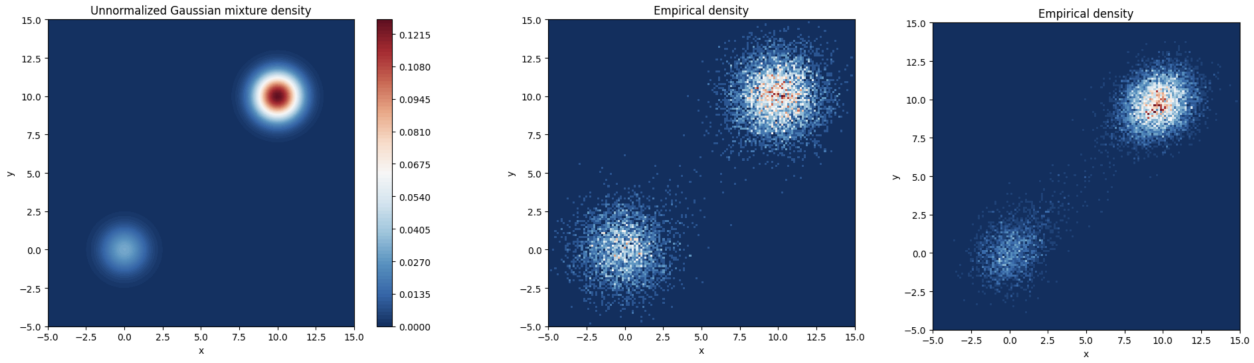

Figure 9. Left: Original Unnormalized Density; **Middle**: Estimated Density with Annealed Langevin algorithm using a NCSN trained on $L = 2$ noise levels; **Right**: Estimated Density with Annealed Langevin algorithm using a NCSN trained on $L = 10$ noise levels. We observe that increasing the number of levels helps us in better estimating both the localisation and the weights of the clusters.