Denoising Score Matching and Diffusion Models

Kevin Wardakhan, Romain Planchon, Arthur Gilles

Goal and approach

We aim to generate new samples from an unknown distribution given only observed data. For example, given a dataset of face images, we want to synthesize realistic unseen faces. Our approach has two steps:

- Learn the *score* of the data distribution, i.e. $\nabla_x \log p(x)$.
- Generate samples using either Langevin-based MCMC or SDE-based samplers.

Score matching

Suppose we are trying to learn a distribution $p(x) = \frac{e^{-f(x)}}{Z}$. The main problem with learning a distribution is its normalizing constant. Often, it is easy to learn a distribution up to a normalizing constant, and computing this constant can be very hard in a high-dimensional problem.
To bypass this constant, we instead try to learn its score, which does not depend on Z.

Sampling (naive Langevin dynamics)

Suppose we know the score field of a distribution, or we have trained a score network s_θ to approximate it (as we will explain later using Eq (2)). How do we actually draw samples from the distribution? The answer: Langevin dynamics!

$$x_{t+1} = x_t + \tau \nabla_x \log p(x_t) + \sqrt{2\tau}\, z_t, \qquad z_t \sim \mathcal{N}(0, I_d) \tag{1}$$

However, for multimodal data such as a Gaussian mixture, this naive approach often fails:

- Far from the data, the score can be undefined.
- The chain quickly collapses to a nearby mode and rarely crosses low-density regions.

This motivates learning and sampling from *smoothed* distributions p_σ obtained by adding Gaussian noise.

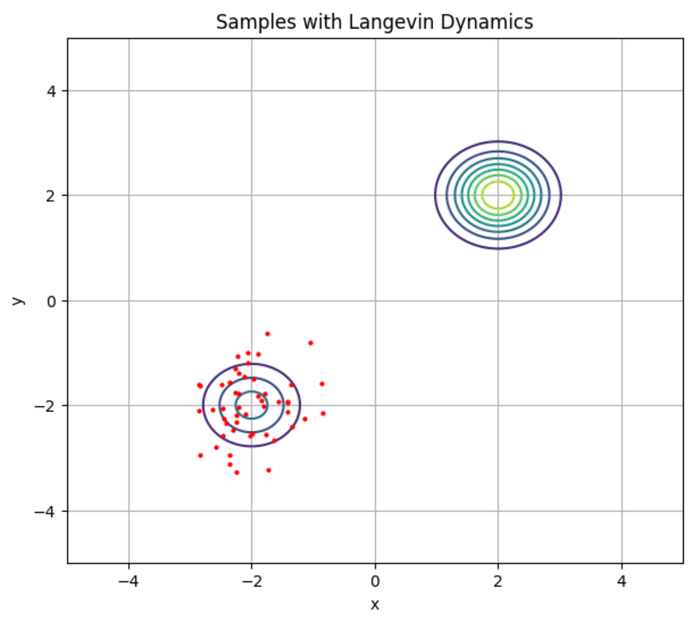


Fig. 1: Langevin dynamics fails to sample from a GMM.

Noise perturbation and score matching

More specifically, suppose we have an underlying data distribution $p(x)$ and consider a sequence of noise scales $\{\sigma_i\}_{i=1}^L$ such that $\sigma_1 > \sigma_2 > \cdots > \sigma_L$. For any $\sigma > 0$, define the Gaussian perturbation kernel

$$p_\sigma(\tilde{x} \mid x) = \mathcal{N}(\tilde{x} \mid x, \sigma^2 I_d)$$

and the corresponding perturbed distribution

$$p_\sigma(\tilde{x}) = \int p_\sigma(\tilde{x} \mid x)\, p(x)\, \mathrm{d}x$$

We train a noise-conditional score network $s_\theta(\tilde{x}, \sigma) \approx \nabla_{\tilde{x}} \log p_\sigma(\tilde{x})$ using denoising score matching:

$$\mathcal{L}(\theta) = \frac{1}{2L} \sum_{i=1}^{L} \mathbb{E}_{x\sim p}\, \mathbb{E}_{\tilde{x}\sim p_{\sigma_i}(\cdot|x)} \left[\left\| s_\theta(\tilde{x}, \sigma_i) - \nabla_{\tilde{x}} \log p_{\sigma_i}(\tilde{x} \mid x) \right\|_2^2 \right] \tag{2}$$

Since $p_{\sigma_i}(\tilde{x} \mid x)$ is Gaussian,

$$\nabla_{\tilde{x}} \log p_{\sigma_i}(\tilde{x} \mid x) = \frac{x - \tilde{x}}{\sigma_i^2}$$

so the target is available in closed form.

Bibliography

[1] A. Hyvärinen. *Estimation of Non-Normalized Statistical Models by Score Matching.* JMLR, 2005.

[2] Y. Song and S. Ermon. *Generative Modeling by Estimating Gradients of the Data Distribution.* NeurIPS, 2019.

[3] Y. Song et al. *Score-Based Generative Modeling through Stochastic Differential Equations.* ICLR, 2021.

[4] P. Vincent. *A Connection Between Score Matching and Denoising Autoencoders.* Neural Computation, 2011.

Annealed Langevin Dynamics

Once we have trained a noise-conditional score network $s_\theta(x, \sigma) \approx \nabla_x \log p_\sigma(x)$, we can sample by starting from a large noise level and progressively decreasing σ. This is *Annealed Langevin Dynamics*.
For a geometric noise schedule $\sigma_1 > \sigma_2 > \cdots > \sigma_L$, the sampler reads:

Algorithm: Annealed Langevin Dynamics

Require: $\{\sigma_i\}_{i=1}^L$, ϵ, T

```
Initialize x̃_0
for i ← 1 to L do
    τ_i ← ε · σ_i^2 / σ_L^2
    for t ← 1 to T do
        Draw z_t ~ N(0, I)
        x̃_t ← x̃_{t-1} + τ_i s_θ(x̃_{t-1}, σ_i) + √(2τ_i) z_t
    end for
    x̃_0 ← x̃_T
end for
return x̃_T
```

This combination of denoising score matching (2) and Annealed Langevin dynamics is known as the SMLD framework.

SDE

This problem can also be formulated with stochastic differential equations. Starting from a point x_0 in the data distribution, we can progressively add noise to it.
With $g(t) = \sqrt{2\,log\,\sigma}\,\sigma^t$, the forward process is written:

$$dx_t = g(t) dw_t$$

We can then train a time-dependent neural network with the following loss by simulating the time-forward SDE:

$$\mathcal{L}_{\mathrm{SDE}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,T]} \left[\lambda(t)\, \mathbb{E}_{x_0\sim p} \mathbb{E}_{x_t\sim p_{0t}(\,\cdot\,|x_0)} \left\| s_\theta(x_t, t) - \nabla_{x_t} \log p_{0t}(x_t \mid x_0) \right\|_2^2 \right] \tag{3}$$

To simulate a new sample, we now have to simulate the SDE:

$$\mathrm{d}x_t = -g(t)^2 s_{\theta^\star}(x_t, t)\, \mathrm{d}t + g(t)\, \mathrm{d}\bar{w}_t$$

In practice, this is done by with by simulating the equation by discretizing it.

Signal-To-Noise ratio

Rather than tuning the step size in the Annealed Langevin Dynamics, we can adjust the Signal-To-Noise ratio, which is defined as:

$$\mathrm{SNR}^2(x^{(k)}, \sigma; \tau) = \frac{\|\Delta x_{\mathrm{det}}\|_2^2}{\mathbb{E}\|\Delta x_{\mathrm{noise}}\|_2^2} = \frac{\|\tau\, s_\theta(x^{(k)}, \sigma)\|_2^2}{2\tau\, \mathbb{E}\|z^{(k)}\|_2^2} = \frac{\tau\, \|s_\theta(x^{(k)}, \sigma)\|_2^2}{2d}$$

Fixing a target $SNR(x^{(k)}, \sigma; \tau) = \rho$ yields an adaptive step size $\tau(x^{(k)}, \sigma) = \frac{2d\,\rho^2}{\|s_\theta(x^{(k)}, \sigma)\|_2^2}$

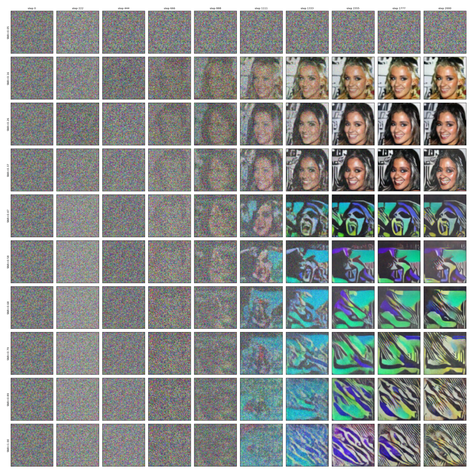
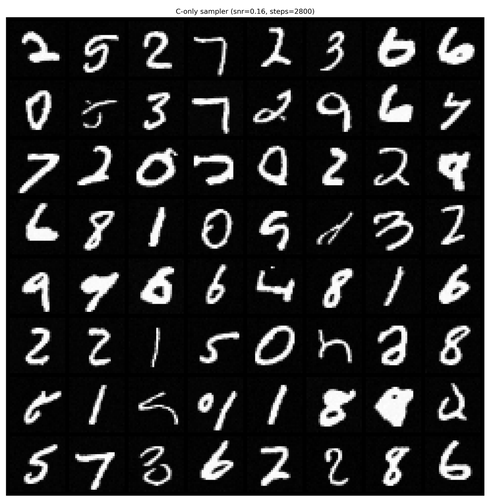


Fig. 2: Different trajectories, with SNR varying between 0.05 (top) and 1 (bottom)

Results

CelebA



MNIST (SMLD)

Fig. 3: Image sampling results (left: VE-SDE/PC on CelebA, right: SMLD on MNIST).

LaTeX TikZposter