ML HW7

Date: 10/15/2025
Week: 7
Author: Alvin B. Lin
Student ID: 112652040

Problem 1: Understanding Score Matching

Explain the concept of score matching and describe how it is used in score-based (diffusion) generative models.

Solution:

1. **Background:** The **MNIST** (Modified National Institute of Standards and Technology) dataset is a standard benchmark for generative models. It consists of thousands of small, grayscale images of handwritten digits (0 through 9).

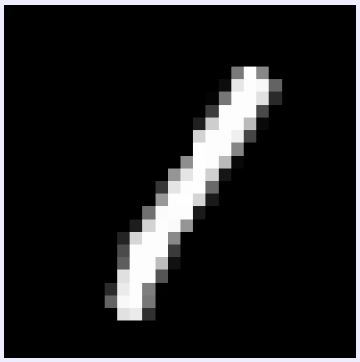

Figure 1: 28×28 Handwritten Digit 1 from MNIST Dataset

   - **The Data Point x Dimensionality:** Each image is a 28×28 grayscale image. The total dimension of a single data point $\mathbf{x}$ is $28 \times 28 = 784$.
   - **Value Space:** Each pixel intensity x_i can be treated as a value between 0 (black) and 1 (white). Thus, $\mathbf{x}$ is a vector in a 784-dimensional space, $\mathcal{X} \subset \mathbb{R}^{784}$.

   The distribution $p(\mathbf{x})$ is the true, fixed, **unknown** probability density function that governs the formation of all possible handwritten digits.

   - **Definition:** $p(\mathbf{x})$ assigns a probability density to every possible image $\mathbf{x}$.
   - **Structure:** This distribution is highly complex:
     - It has very high density only in tiny, manifold-like regions of the 784-dimensional space (where the points look like well-formed digits).
     - It has near-zero density everywhere else (e.g., in regions where images are pure noise or unintelligible scribbles).

2. **The Traditional Goal—Maximum Likelihood Estimation:** The optimal parameters θ^* are found by maximising the Log-Likelihood of the data, which is equivalent to minimising the Negative Log-Likelihood (NLL):

$$\text{NLL}(\theta) = -\mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})} \log p(\mathbf{x}; \theta),$$

where $p(\mathbf{x})$ is the actual distribution, $p(\mathbf{x};\theta)$ is our model/hypothesis function. So,

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\ \mathrm{NLL}(\theta) = \underset{\theta}{\operatorname{argmax}}\ \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\left[\log p(\mathbf{x};\theta)\right]$$

3. **The Failure of Traditional MLE for High-Dimensional Data:** When the model $p(\mathbf{x};\theta)$ is a **deep neural network**, it is often defined through an unnormalised probability density function $q(\mathbf{x};\theta)$:

$$p(\mathbf{x};\theta) = \frac{q(\mathbf{x};\theta)}{Z(\theta)},$$

where $q(\mathbf{x};\theta)$ is a function that neural network is easy to compute, *e.g.* exponential, $Z(\theta)$ be the normalisation constant:

$$Z(\theta) = \int_{\mathbb{R}^{784}} p(\mathbf{x};\theta)\ \mathrm{d}\mathbf{x} = \int_{\mathcal{X}} p(\mathbf{x};\theta)\ \mathrm{d}\mathbf{x}$$

Therefore our goal becomes:

$$\theta^* = \underset{\theta}{\operatorname{argmax}}\ \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\left[\log q(\mathbf{x};\theta) - \log Z(\theta)\right]$$

The Bottleneck: The Intractable Normalisation Constant $\mathbf{Z}(\theta)$: For MNIST, the space $\mathcal{X}$ is $\mathbb{R}^{784}$. Computing the integral $Z(\theta)$ to get the model's true probability $p(\mathbf{x};\theta)$ is **computationally intractable**.

Therefore, when we try to find θ^*, we need to compute $\nabla_\theta \mathrm{NLL}(\theta)$:

$$\nabla_\theta \mathrm{NLL}(\theta) = \nabla_\theta \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\left[\log q(\mathbf{x};\theta) - \log Z(\theta)\right] = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\left[\nabla_\theta \log q(\mathbf{x};\theta) - \nabla_\theta \log Z(\theta)\right]$$

the term $\nabla_\theta \log \mathbf{Z}(\theta)$ makes the entire MLE optimisation process impossible using standard gradient descent.

4. **The Score Matching Is Born:** Score Matching ignores the intractable probability function $p(\mathbf{x};\theta)$ and instead focuses on the score function,

$$S(\mathbf{x};\theta) := \nabla_{\mathbf{x}} \log p(\mathbf{x};\theta).$$

Hence,

$$S(\mathbf{x};\theta) = \nabla_{\mathbf{x}} \log \mathbf{p}(\mathbf{x};\theta) = \nabla_{\mathbf{x}} \left(\log q(\mathbf{x};\theta) - \log Z(\theta)\right) = \nabla_{\mathbf{x}} \log q(\mathbf{x};\theta),$$

which is always **computable**. We then define two **score matchings:**

> **Definition 1: ESM & ISM**
>
> The **explicit score matching** is defined as:
>
> $$L_{\mathrm{ESM}} = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})} \|S(\mathbf{x};\theta) - \nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2$$
>
> And the **implicit score matching** is defined as:
>
> $$L_{\mathrm{ISM}} = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})} \left(\|S(\mathbf{x};\theta)\|^2 + 2\nabla_{\mathbf{x}} \cdot S(\mathbf{x};\theta)\right)$$

We can see that it is impossible to calculate **ESM** directly due to the unknown term $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ when we don't know the actual distribution; hence we need the aid of **ISM**.

5. **Why Does Score Matching Works?**

To answer this question, **if our model is accurate**, we have the following relation:

$$\underset{\theta}{\operatorname{argmin}}\ \mathrm{NNL}(\theta) \overset{(1)}{=} \underset{\theta}{\operatorname{argmin}}\ D_{\mathrm{KL}}(p(\mathbf{x})\|p(\mathbf{x};\theta)) \overset{(2)}{=} \underset{\theta}{\operatorname{argmin}}\ L_{\mathrm{ESM}}(\theta) \overset{(3)}{=} \underset{\theta}{\operatorname{argmin}}\ L_{\mathrm{ISM}}(\theta)$$

We will show equation (1) and (2), equation (3) has been shown in the lecture note[1].

Theorem 1: Demonstration of Equation (1)

By definition of the Kullback–Leibler divergence

$$D_{\mathrm{KL}}(p(\mathbf{x})\|p(\mathbf{x};\theta)) = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\left(\log p(\mathbf{x}) - \log p(\mathbf{x};\theta)\right)$$

Hence

$$\underset{\theta}{\operatorname{argmin}}\ D_{\mathrm{KL}}(p(\mathbf{x})\|p(\mathbf{x};\theta)) = -\underset{\theta}{\operatorname{argmin}}\ \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})} \log p(\mathbf{x}) = \underset{\theta}{\operatorname{argmin}}\ \mathrm{NNL}(\theta) \qquad \square$$

Definition 2: Almost Everywhere

We say a **Borel-measurable** function $f(x) = K$ **a.e.** (almost everywhere) on $\mathbb{R}$, when

$$\mu\left(\{x \in \mathbb{R} |\ f(x) \neq K\}\right) = 0,$$

where $\mu(\cdot)$ is the **Borel/Lebesgue** measure.

Lemma 1

If $\mathbf{f},\ \mathbf{g} : \mathbb{R}^n \to \mathbb{R}$, it follows that

$$\mathbf{f} - \mathbf{g} = K \quad \textbf{a.e.} \iff \nabla_{\mathbf{x}}\mathbf{f} - \nabla_{\mathbf{x}}\mathbf{g} = \mathbf{0} \quad \textbf{a.e.}$$

Proof. By fundamental theorem of calculus, for any path C with starting point and end point fixed $\mathbf{a},\ \mathbf{b}$,

$$\mathbf{h}(\mathbf{b}) - \mathbf{h}(\mathbf{a}) = \int_C \nabla_{\mathbf{x}}\mathbf{h} \cdot \mathrm{d}\mathbf{x}$$

($\Rightarrow$) Take $\mathbf{h} = \mathbf{f} - \mathbf{g}$, we get $\mathbf{h}(\mathbf{b}) - \mathbf{h}(\mathbf{a}) = K - K = 0 \quad \textbf{a.e.}\ \forall\ \mathbf{a},\ \mathbf{b} \in \mathbb{R}^n$, then

$$\int_C \nabla_{\mathbf{x}}\mathbf{h} \cdot \mathrm{d}\mathbf{x} = 0 \quad \textbf{a.e.} \quad \forall \text{ path } C \in \mathbb{R}^n \iff \nabla_{\mathbf{x}}\mathbf{h} = \nabla_{\mathbf{x}}\mathbf{f} - \nabla_{\mathbf{x}}\mathbf{g} = \mathbf{0} \quad \textbf{a.e.}$$

($\Leftarrow$) Take $\mathbf{h} = \mathbf{f} - \mathbf{g}$ again, therefore $\nabla_{\mathbf{x}}\mathbf{h} = \mathbf{0} \quad \textbf{a.e.}$; then $\forall\ \mathbf{a},\ \mathbf{b}$, we have

$$\mathbf{h}(\mathbf{b}) - \mathbf{h}(\mathbf{a}) = \int_C \nabla_{\mathbf{x}}\mathbf{h} \cdot \mathrm{d}\mathbf{x} = 0 \implies \mathbf{f} - \mathbf{g} = \mathbf{h} = K \quad \textbf{a.e.},$$

which completes the proof. $\square$

Theorem 2: Demonstration of Approximation/Equation (2)

Given our model is **accurate enough**, *i.e.* $\exists\ \theta^* \in \Theta,\ s.t.\ p(\mathbf{x}) = p(\mathbf{x};\theta^*)$ **a.e.**, then it follows that

$$\operatorname*{argmin}_\theta D_{\mathrm{KL}}(p(\mathbf{x})\|p(\mathbf{x};\theta)) = \operatorname*{argmin}_\theta L_{\mathrm{ESM}}(\theta).$$

Proof. We first know that $0 \le D_{\mathrm{KL}}(p(\mathbf{x})\|p(\mathbf{x};\theta))$ and $0 \le L_{\mathrm{ESM}}(\theta)$, by definition. Also, by definition of the two divergence, we have:

$$D_{KL}(p(\mathbf{x})\|p(\mathbf{x};\theta)) = 0 \iff p(\mathbf{x};\theta) = p(\mathbf{x}) \quad \text{a.e.} \iff \log p(\mathbf{x};\theta) = \log p(\mathbf{x}) \quad \text{a.e.}$$

and

$$L_{\mathrm{ESM}}(\theta) = 0 \iff \nabla_{\mathbf{x}} \log p(\mathbf{x};\theta) = \nabla_{\mathbf{x}} \log p(\mathbf{x}) \quad \text{a.e.}$$

By assigning $\log p(\mathbf{x}) = \mathbf{f}$ and $\log p(\mathbf{x};\theta) = \mathbf{g}$ and applying to **Lemma 1**, we get:

$$\log p(\mathbf{x}) - \log p(\mathbf{x};\theta) = K \quad \textbf{a.e.} \iff \nabla_{\mathbf{x}} \log p(\mathbf{x}) - \nabla_{\mathbf{x}} \log p(\mathbf{x};\theta) = \mathbf{0} \quad \textbf{a.e.}$$

So if $\log p(\mathbf{x}) - \log p(\mathbf{x};\theta) = K$ **a.e.**, we have $p(\mathbf{x}) = e^K p(\mathbf{x};\theta)$ **a.e.**.
Recall that $p(\mathbf{x})$ and $p(\mathbf{x};\theta)$ are both **p.d.f.**, hence

$$e^K = \frac{\int_{\mathcal{X}} p(\mathbf{x})\mathrm{d}\mathbf{x}}{\int_{\mathcal{X}} p(\mathbf{x};\theta)\mathrm{d}\mathbf{x}} = \frac{1}{1} = 1 \implies K = 0$$

Therefore we have:

$$\log p(\mathbf{x}) = \log p(\mathbf{x};\theta) \quad \textbf{a.e.} \iff \nabla_{\mathbf{x}} \log p(\mathbf{x}) = \nabla_{\mathbf{x}} \log p(\mathbf{x};\theta) \quad \textbf{a.e.}$$

Finally, combine everything together, we get:

$$L_{\mathrm{ESM}}(\theta) = 0 \iff D_{KL}(p(\mathbf{x})\|p(\mathbf{x};\theta)) = 0$$

Also by the initial assumption that $p(\mathbf{x}) = p(\mathbf{x};\theta^*)$ **a.e.**, we get the final result:

$$\operatorname*{argmin}_\theta D_{KL}(p(\mathbf{x})\|p(\mathbf{x};\theta)) = 0 = \operatorname*{argmin}_\theta L_{\mathrm{ESM}}(\theta)$$

□

So in short, for an **accurate enough** hypothesis function $p(\mathbf{x};\theta)$, we heve:

$$\operatorname*{argmin}_\theta \mathrm{NNL}(\theta) = \operatorname*{argmin}_\theta L_{\mathrm{ISM}}(\theta),$$

which means our definition to $S(\mathbf{x};\theta) = \log p(\mathbf{x};\theta)$ is **reasonable** and can make the computation to the loss **computable and tractable** based on the lecture note.

6. **The Practical Solution: Denoising Score Matching**
   Although $L_{\mathrm{ISM}}(\theta)$ is theoretically tractable, but it is generally computationally inefficient ("intractable for large-scale problems") due to its reliance on the **Hessian trace**.

The General Perturbation Process

- Original Data $\mathbf{x}_0$: We start with a clean data point $\mathbf{x}_0 \sim p_0(\mathbf{x}_0)$.
- Perturbation $\mathbf{x}|\mathbf{x}_0$: We corrupt $\mathbf{x}_0$ by a general noise kernel, $p(\mathbf{x}|\mathbf{x}_0)$, to generate a noisy data point, $\mathbf{x}$.
  - $\mathbf{x}$ is the noisy data.
  - $p(\mathbf{x}|\mathbf{x}_0)$ is the known conditional (noise) distribution.
- Noisy Data Distribution ($p_\sigma(\mathbf{x})$): The resulting distribution of noisy points is the marginal distribution, $p_\sigma(\mathbf{x})$, found by integrating over all possible clean inputs:

$$p_\sigma(\mathbf{x}) = \int_{\mathbb{R}^d} p(\mathbf{x}|\mathbf{x}_0)p_0(\mathbf{x}_0)\mathrm{d}\mathbf{x}_0$$

The objective becomes training $S_\theta(\mathbf{x})$ to match the score of this noisy distribution, $\nabla_{\mathbf{x}} \log p_\sigma(\mathbf{x})$, which is still intractable in the ESM form.

The key mathematical step is to use the **Hyvärinen Score Matching theorem** on the noisy distribution $p_\sigma(\mathbf{x})$, which proves that minimizing the intractable $L_{\text{ESM}}(p_\sigma \| p_\theta)$ is equivalent to minimising the practical $L_{\text{DSM}}(\theta)$ loss:

$$L_{\text{DSM}}(\theta) := \mathbb{E}_{\mathbf{x}_0 \sim p_0(\mathbf{x}_0)} \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x}|\mathbf{x}_0)} \left[\| S_\sigma(\mathbf{x}; \theta) - \nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{x}_0) \|^2 \right]$$

Both the mentioned theorem and the lecture note[1] have shown that:

$$\operatorname*{argmin}_\theta L_{\text{DSM}}(\theta) = \operatorname*{argmin}_\theta L_{\text{ESM}}(\theta)$$

Fortunately, the calculation of $L_{\text{DSM}}(\theta)$ is tractable, meaning that we don't need to calculate the **trace of the Hessian matrix**.

7. **The Practical Form: Gaussian Noise**

In nearly all modern score-based generative models (diffusion models), the noise kernel $p(\mathbf{x}|\mathbf{x}_0)$ is chosen to be a simple **isotropic Gaussian distribution** with mean $\mathbf{x}_0$ and variance $\sigma^2 I$.

$$\mathbf{x} = \mathbf{x_0} + \varepsilon, \quad \text{where } \varepsilon \sim \mathcal{N}(\mathbf{0}, \sigma^\mathbf{2}\mathbf{I})$$

Under this specific Gaussian perturbation:

- The conditional distribution is defined:

$$p(\mathbf{x}|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}; \mathbf{x}_0, \sigma^2 I)$$

- The tractable target is derived: Gradient of the conditional log-density:

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{x}_0) = \nabla_{\mathbf{x}} \left(C - \frac{1}{2\sigma^2} \|\mathbf{x} - \mathbf{x}_0\|^2 \right) = -\frac{\mathbf{x} - \mathbf{x}_0}{\sigma^2}$$

- The DSM Loss Simplifies: Since the noise is $\varepsilon = \mathbf{x} - \mathbf{x}_0$, the loss becomes:

$$L_{\text{DSM}}(\theta) = \mathbb{E}_{\mathbf{x}_0 \sim p_0(\mathbf{x}_0)} \mathbb{E}_{\varepsilon \sim \mathcal{N}(0, \sigma^2 I)} \left[\left\| S_\theta(\mathbf{x}_0 + \varepsilon) - \left(-\frac{\varepsilon}{\sigma^2} \right) \right\|^2 \right]$$

8. **How Score Matching is Used in Generative Models**

Score Matching is the engine behind modern Score-Based Generative Models (SGM), which are primarily built on the Diffusion Model architecture.

I've run a simple implementation on the concept of **DSM**, instructed from the note on Song Yang's website[2], as a **scored-based generative model** to help understand the full process and how **DSM** can be used. Here is the detail:

(a) **The Setup: Multi-Level Perturbation Process**

The simulation begins by creating a multi-modal 2D dataset ($\mathbf{x}_0$) composed of two distinct Gaussian clusters (the "Two Modes").

- **Original Data** $p_0(\mathbf{x}_0)$: The initial plot shows the true, unknown density function, $p_0(\mathbf{x}_0)$.

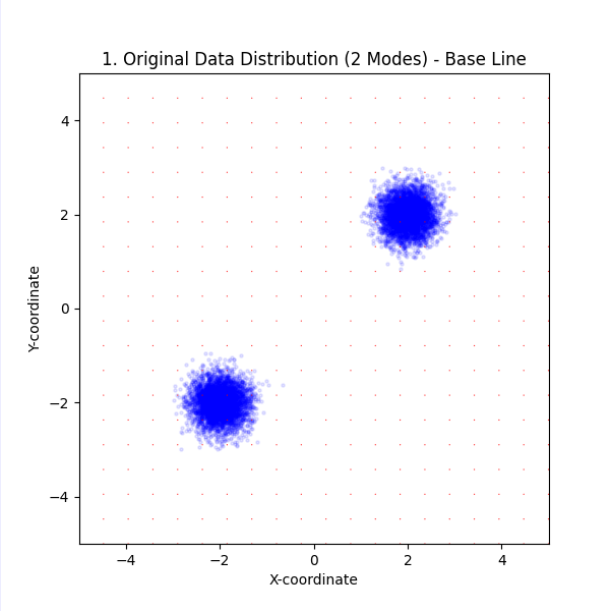

Figure 2: Original Data Distribution

This represents the complex, high-density regions we want the model to learn.The code then simulates the general perturbation process described in **section 7**, using Gaussian Noise $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$. Instead of just one σ, the code uses a discrete noise schedule $\sigma \in \{0.75, 0.5, \ldots, 0.01\}$. This is the foundation of **Score-Based Generative Models**.

(b) **Training with DSM**

The central objective is to **find a neural network** like our `ScoreModel2D` or S_θ that accurately **approximates the true score function** $\nabla_{\mathbf{x}} \log p_\sigma(\mathbf{x})$, the full **training process** has been drawn as a **flow chart** and appended to the **appendix** 6.

i. **Tractable Loss Calculation:** The code explicitly minimises the DSM Loss, which is made tractable by using the specific Gaussian noise kernel:

$$L_{DSM}(\theta) \propto \mathbb{E}_{\mathbf{x}_0,\epsilon}\left[\left\| S_\theta(\mathbf{x}_0+\epsilon,\sigma) - \left(\frac{-\epsilon}{\sigma^2}\right)\right\|^2\right]$$

- The term $\mathbf{x} = \mathbf{x}_0 + \epsilon$ is the noisy data.
- The term $-\dfrac{\epsilon}{\sigma^2}$ is the tractable target score $\nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{x}_0)$.

ii. **Learning Multi-Scale Structure (NCSN):** The model is trained to learn the correct score field for every σ in the schedule, giving it two distinct "skills" visualised in the plots:

Table 1: Interpretation of The Trained Model with Different Noises

Fig. #	Noise Level	Interpretation by the Model S_θ
2	High, $\sigma = 0.75$	Inward toward the centre, Global, Rough Guide
3	Low, $\sigma = 0.01$	Directly into the nearest cluster centre, Local, Fine-Tuning Guide

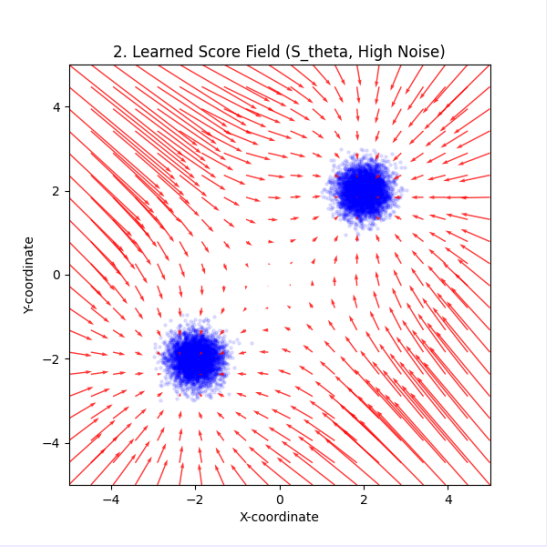

Figure 3: Learned High Noise Score Field

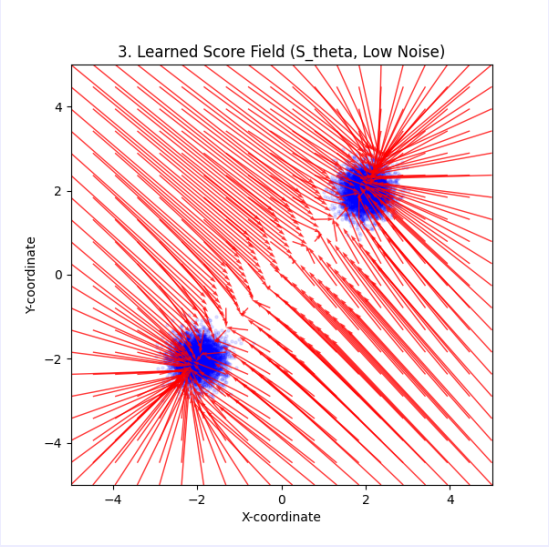

Figure 4: Learned Low Noise Score Field

(c) The Application: Annealed Langevin Dynamics

The model is used for generation by applying the **Annealed Langevin Dynamics** [2] sampling process. This process starts with pure noise and then iteratively reduces the noise level σ, using the corresponding learned score function $S_\theta(\mathbf{x}, \sigma)$ at each level.

The update rule simulated in the code is:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \alpha \cdot S_\theta(\mathbf{x}_t, \sigma) + \sqrt{2\alpha} \cdot \mathbf{z},$$

where α is the step size and $\mathbf{z} \sim \mathcal{N}(0, I)$.

The process starts by initialising samples from large Gaussian noise and then sequentially runs the **Langevin Dynamics** update for every σ in the schedule, from largest to smallest (annealing). In each step, the sample $\mathbf{x}$ is moved toward regions of **higher probability density**, guided by the **score** $S_\theta(\mathbf{x}, \sigma)$, while a small amount of **diffusion noise is added**. The final samples, once the noise level reaches its minimum, represent the generated data from the **target distribution**.

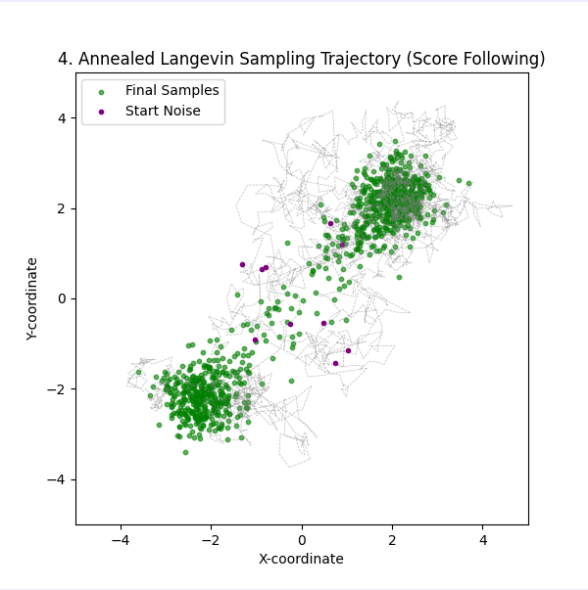

Figure 5: Final Sampling Visualisation

Around 50,000 iterations for the annealing Langevin Dynamics, we get the following result, which shows that the score learned is fairly close to the original distribution in **Figure** 2, which means the process is successful.

In particular, we can see some random starting noises eventually follow the **grey trajectory** to the clusters, which is a clear visualisation of the validation of the method.

References

[1] Tesheng Lin. 2025_ml_week_7. https://hackmd.io/@teshenglin/2025_ML_week_7, 2025. Accessed: 15 October 2025.

[2] Yang Song. Generative modeling by estimating gradients of the data distribution. https://yang-song.net/blog/2021/score/, 2021. Accessed: 15 October 2025.

A Training Process Flowchart

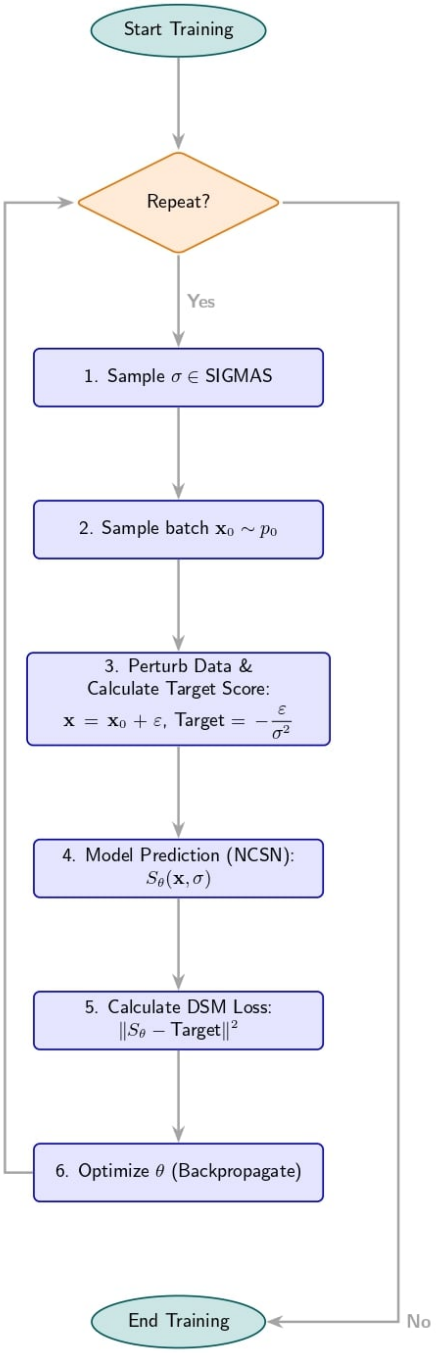

Figure 6: Flow Chart of The Training Process