1 Explaining the Concept of Score Matching

1.1 Definition of the Score Function

For a given data probability density function $p(x)$, its **Score Function** $S(x)$ is defined as the **gradient of the log-probability density function with respect to the data** x.

$$S(x) \equiv \nabla_x \log p(x)$$

This function $S(x)$ is a vector field. At any point x in the data space, it points in the direction of the **steepest ascent** for the probability density $p(x)$.

1.2 Explicit Score Matching (ESM)

The goal of "Score Matching" is to train a parameterized model $S(x; \theta)$ (e.g., a neural network) to approximate the true score function $S(x)$.

The most intuitive method is to minimize the L2 distance between them. This is known as **Explicit Score Matching (ESM)**, and its loss function L_{ESM} is defined as:

$$L_{ESM}(\theta) = \mathbb{E}_{x \sim p(x)} \left[\|S(x; \theta) - \nabla_x \log p(x)\|_2^2 \right]$$

The Core Problem: This loss function is **intractable** (impossible to compute). This is because we only have data samples from $p(x)$ and do not know the true score $\nabla_x \log p(x)$.

1.3 Denoising Score Matching (DSM)

Denoising Score Matching (DSM) was proposed to solve the problem of ESM. Its core idea is: instead of matching the score of the original data x_0, it aims to match the score of **perturbed** (noisy) data x.

1. **Define the Perturbation Process:** We first define a perturbation process $p_\sigma(x|x_0)$, typically Gaussian noise. Let $x_0 \sim p_0(x_0)$ be the original data. The perturbed data x is:

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

   This x is drawn from the "perturbed data distribution" $p_\sigma(x) = \int p_\sigma(x|x_0)p_0(x_0)dx_0$.

2. **The DSM Objective Function:** The goal of DSM is to train a model $S_\sigma(x; \theta)$ to match the **conditional score** of this perturbed data, $\nabla_x \log p_\sigma(x|x_0)$. Its loss function L_{DSM} is defined as:

   $$L_{DSM}(\theta) = \mathbb{E}_{x_0 \sim p_0(x_0)} \mathbb{E}_{x \sim p_\sigma(x|x_0)} \left[\|S_\sigma(x; \theta) - \nabla_x \log p_\sigma(x|x_0)\|_2^2 \right]$$

3. **The Feasibility of DSM (The Key Insight):** This L_{DSM} is **tractable**! This is because the perturbation process $p_\sigma(x|x_0)$ is a Gaussian distribution defined by us:

$$p_\sigma(x|x_0) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{1}{2\sigma^2}\|x - x_0\|_2^2\right)$$

We can easily compute its score (the "ground truth"):

$$\nabla_x \log p_\sigma(x|x_0) = \nabla_x \left(-\frac{1}{2\sigma^2}\|x - x_0\|_2^2 + \text{const}\right) = -\frac{1}{\sigma^2}(x - x_0)$$

4. **The Final, Tractable Loss Function:** Plugging this known "ground truth" score into L_{DSM}, we get:

$$L_{DSM}(\theta) = \mathbb{E}_{x_0 \sim p_0(x_0)} \mathbb{E}_{x \sim p_\sigma(x|x_0)} \left[\left\| S_\sigma(x;\theta) + \frac{x - x_0}{\sigma^2} \right\|_2^2 \right]$$

Since $x = x_0 + \epsilon_\sigma$, we have $x - x_0 = \epsilon_\sigma$. The loss can be rewritten as:

$$L_{DSM}(\theta) = \mathbb{E}_{x_0 \sim p_0(x_0)} \mathbb{E}_{\epsilon_\sigma \sim \mathcal{N}(0,\sigma^2 I)} \left[\left\| S_\sigma(x_0 + \epsilon_\sigma;\theta) + \frac{\epsilon_\sigma}{\sigma^2} \right\|_2^2 \right]$$

Important Conclusion: Training the DSM model S_σ to match the score is mathematically equivalent to (via re-parameterization, e.g., $S_\sigma(x;\theta) \approx -\frac{\epsilon_\theta(x;\theta)}{\sigma}$) training a denoising network ϵ_θ to **predict the added noise** ϵ.

2 Application of Score Matching in Score-Based (Diffusion) Generative Models

Score-based generative models (or diffusion models) use Score Matching to learn how to progressively restore real data from pure noise.

2.1 Forward Process (Diffusion)

This is a fixed, progressive **noising** process. From $t = 0$ to $t = T$, we add varying levels of noise σ_t to the original data x_0, obtaining a sequence of noisy samples x_t.

$$x_0 \to x_1 \to \cdots \to x_t \to \cdots \to x_T$$

The distribution of x_t is $p(x_t)$, and the distribution of x_T, $p(x_T)$, approaches a known prior distribution, such as pure noise $\mathcal{N}(0, I)$.

2.2 Reverse Process (Generation)

This is the process the model must **learn**. We want to start from pure noise $x_T \sim \mathcal{N}(0, I)$ and progressively **denoise** it, reversing the forward process to eventually generate a real data sample x_0.

$$x_T \to x_{T-1} \to \cdots \to x_t \to \cdots \to x_0$$

2.3 The Core Application of Score Matching

The Key Question: In the reverse process, when we are at x_t, how do we know the "correct denoising direction" to get x_{t-1}?

The Answer: Mathematical theory (e.g., the reverse of a Stochastic Differential Equation) shows that this reverse step depends entirely on knowing the **score function of the data distribution at time** t, i.e., $\nabla_{x_t} \log p(x_t)$.

This is precisely where Score Matching is applied:

1. **Training Objective:** Our goal is to train a **single, time-dependent** neural network $S_\theta(x_t, t)$ that can estimate the true score $\nabla_{x_t} \log p(x_t)$ at **any noise level** t.

2. **Training Method (using DSM):** Learning $\nabla_{x_t} \log p(x_t)$ directly is hard. However, $p(x_t)$ is just the result of noising $p(x_0)$ with the process $p(x_t|x_0)$. This perfectly fits the DSM framework.

   Instead of matching $\nabla_{x_t} \log p(x_t)$, we match the **known conditional score** $\nabla_{x_t} \log p(x_t|x_0)$.

   The total loss function is the expectation of the L_{DSM} loss over all timesteps t:

   $$L(\theta) = \mathbb{E}_{t \sim \mathcal{U}(0,T)} \left[L_{DSM}^{(t)}(\theta) \right]$$

   Where $L_{DSM}^{(t)}$ is the DSM loss at time t (using the corresponding noise level σ_t):

   $$L_{DSM}^{(t)}(\theta) = \mathbb{E}_{x_0, x_t} \left[\| S_\theta(x_t, t) - \nabla_{x_t} \log p(x_t|x_0) \|_2^2 \right]$$

3. **Simplification to Noise Prediction:** As shown before, $\nabla_{x_t} \log p(x_t|x_0)$ is known (e.g., in the DDPM framework, it is $-\frac{\epsilon}{\sigma_t}$, where ϵ is the added standard Gaussian noise).

   Therefore, in practice, the score model S_θ is often re-parameterized as a "noise predictor" $\epsilon_\theta(x_t, t)$. Training S_θ to match the score becomes equivalent to training ϵ_θ to match the noise ϵ.

The loss function becomes the well-known form:

$$L(\theta) = \mathbb{E}_{t,x_0,\epsilon}\left[\|\epsilon_\theta(x_t,t) - \epsilon\|_2^2\right] \quad \text{where } x_t = \alpha_t x_0 + \sigma_t \epsilon$$

2.4 Generation (Sampling)

Once the model $S_\theta(x_t,t)$ (or $\epsilon_\theta(x_t,t)$) is trained:

1. We start with a sample from the prior distribution, $x_T \sim \mathcal{N}(0,I)$.
2. We iterate from $t = T$ down to $t = 1$.
3. At each step t, we feed x_t and t into the model to get an estimate of the score $\hat{S} = S_\theta(x_t,t)$.
4. We use this $\hat{S}$ as a "denoising guide" in the reverse process update rule (e.g., an SDE solver or the DDPM sampling step) to calculate x_{t-1}.
5. The final x_0 is a new sample generated by the model.

The content is written with reference to chatgpt, and chatgpt is requested to modify the format.

3 Questions

1. **Why is "predicting noise" equivalent to "matching the score"?**

   In The class notes show that the L_{DSM} loss can be rewritten. Intuitively, one model is $S_\sigma(x;\theta)$ (the score model), and the other target is ϵ (the noise). Why did practice (e.g., DDPM) converge on training a model $\epsilon_\theta(x_t,t)$ to "predict the noise ϵ" instead of directly "matching the score $\nabla_x \log p(x|x_0)$"? Is there a concrete difference in training stability or performance between these two approaches?