# Lecture 03 - Score-Based Perspective: From EBMs to NCSN

Yu Xiang

Topics on Diffusion Models

School of Data Science, Fudan University

# Table of Contents

# Introduction: From Likelihood to Score

- **Recap:** In VAEs, we optimized the *Evidence Lower Bound (ELBO)* to approximate the likelihood.

- **Energy-Based Models (EBMs): [Ackley et al., 1985, LeCun et al., 2006]** Define the probability density using an energy function $E_\theta(\mathrm{x})$:

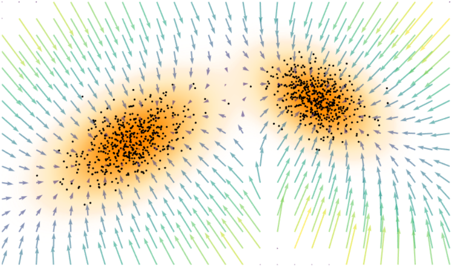
$$p_\theta(\mathrm{x}) = \frac{e^{-E_\theta(\mathrm{x})}}{Z_\theta}, \quad \text{where } Z_\theta = \int e^{-E_\theta(\mathrm{x})} d\mathrm{x}$$

- **The Challenge:** Computing the $Z_\theta$ is generally intractable.

- **Key Insight (The Score):** The gradient of the log-density (the Score) is independent of $Z_\theta$:

$$\nabla_\mathrm{x} \log p_\theta(\mathrm{x}) = -\nabla_\mathrm{x} E_\theta(\mathrm{x}) - \underbrace{\nabla_\mathrm{x} \log Z_\theta}_{=0} = -\nabla_\mathrm{x} E_\theta(\mathrm{x})$$

# Introduction: Score-Based Generation



The score function $\nabla_x \log p(x)$ defines a vector field pointing toward high-density data regions.

- **Langevin Dynamics:** Generate samples by starting from random noise and following the score vectors toward the data manifold.
- **From EBM to NCSN:**
  - Scores are ill-defined in low-density regions (where no data exists).
  - Perturb data with multiple noise levels for robust, progressive denoising.

# EBMs: Definition and Formalism

- EBMs define a probability density $p_\phi(\mathrm{x})$ using an energy function $E_\phi(\mathrm{x})$.
- The energy function is parameterized by $\phi$, and assigns lower energy to more likely data configurations.

## The Boltzmann Distribution

The resulting distribution is a form of the Gibbs/Boltzmann distribution:

$$p_\phi(\mathrm{x}) := \frac{\exp(-E_\phi(\mathrm{x}))}{Z_\phi}$$

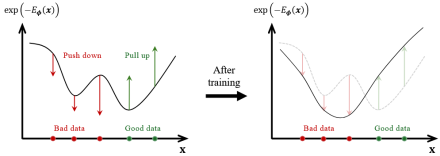
where $Z_\phi$ is the partition function ensuring normalization:

$$Z_\phi := \int_{\mathbb{R}^D} \exp(-E_\phi(\mathrm{x}))dx$$

- $\int_{\mathbb{R}^D} p_\phi(\mathrm{x})dx = 1.$

# EBMs: Energy Landscape and Global Trade-offs

- **Intuition:** The data points lie in the valleys of the energy landscape, much like a ball rolling down to equilibrium.
- **Relative Energy:** Only the relative energy values matter; adding a constant to all energies does not change the distribution $p_\phi(\mathrm{x})$.




EBM training lowers energy (pushes down) at good data and raises energy (pulls up) at bad data. This enforces a strict global trade-off.

- **Global Trade-off:** Decreasing the energy in one region must lead to a decrease in probability elsewhere, as the total mass must sum to one.

# Challenges of Maximum Likelihood Training

- In principle, EBMs can be trained by maximizing the log-likelihood:

$$\mathcal{L}_{\text{MLE}}(\phi) = \mathbb{E}_{p_{\text{data}}(\mathrm{x})}[\log p_\phi(\mathrm{x})]$$

## Decomposition and Intractability

The MLE objective is decomposed as:

$$\mathcal{L}_{\text{MLE}}(\phi) = \underbrace{-\mathbb{E}_{p_{\text{data}}}[E_\phi(\mathrm{x})]}_{\text{1. Lowers Energy of Data}} - \underbrace{\log Z_\phi}_{\text{2. Global Regularization}}$$

The gradient of $\log Z_\phi$ is $\mathbb{E}_{p_\phi(\mathrm{x})}[-\nabla_\phi E_\phi(\mathrm{x})]$.

- **The Problem:** Computing $\log Z_\phi$ and its gradient is intractable in high dimensions, requires sampling from the complex distribution $p_\phi(\mathrm{x})$.
- **Motivation:** **Avoid** the partition function entirely.

# The Score Function: Definition and Intuition

- **Definition:** For a probability density $p(\mathrm{x})$ on $\mathbb{R}^D$, the score function is defined as the gradient of the log-density:

$$\mathrm{s}(\mathrm{x}) := \nabla_{\mathrm{x}} \log p(\mathrm{x}), \quad \mathrm{s} : \mathbb{R}^D \to \mathbb{R}^D$$

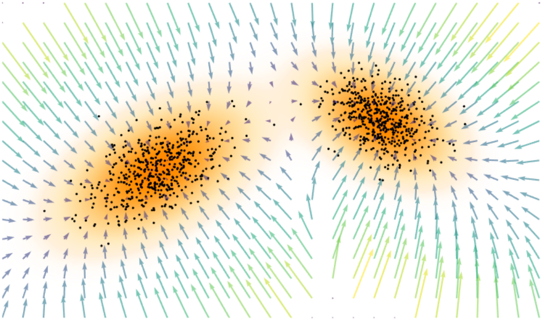


Illustration of score vector fields. The vectors $\nabla_{\mathrm{x}} \log p(\mathrm{x})$ point toward regions of increasing density (the mode), providing a local guide.

# Why Model Scores? 1. Freedom from Normalization

- **The EBM Bottleneck:** Many distributions (like EBMs) are defined up to an unnormalized density $\tilde{p}(x) = \exp(-E_\phi(x))$:

$$p(x) = \frac{\tilde{p}(x)}{Z}, \quad \text{where } Z = \int \tilde{p}(x)dx$$

- **The Score Advantage:** The score is **independent** of $Z$:

$$\begin{aligned}\nabla_x \log p(x) &= \nabla_x \log \left(\frac{\tilde{p}(x)}{Z}\right) \\ &= \nabla_x \log \tilde{p}(x) - \underbrace{\nabla_x \log Z}_{=0 \text{ (constant w.r.t } x)}\end{aligned}$$

- Key Takeaway: We can train score-based models without ever evaluating the partition function $Z$.

# Why Model Scores? 2. A Complete Representation

- Does modeling the gradient lose information compared to modeling the density? **No.**
- **A Complete Representation:** The score function fully characterizes the underlying distribution. The density can be recovered (up to a constant) via integration:

$$\log p(\mathrm{x}) = \log p(\mathrm{x}_0) + \int_0^1 \mathrm{s}(\mathrm{x}_0 + t(\mathrm{x} - \mathrm{x}_0))^\top (\mathrm{x} - \mathrm{x}_0)\, dt$$

- **Implication:** Modeling the score is as expressive as modeling $p(\mathrm{x})$ itself, but often computationally more tractable for generative tasks.

# Score Matching: The Objective

- **Recap:** Max Likelihood estimation for EBMs is intractable due to the partition function $Z_\phi$.
- **Key Observation:** The model score is independent of $Z_\phi$:

$$s_\phi(x) = \nabla_x \log p_\phi(x) = -\nabla_x E_\phi(x)$$

**Explicit Score Matching Objective [Hyvärinen and Dayan, 2005]**

Instead of fitting probabilities, we align the model score with the data score by minimizing the Fisher Divergence:

$$\mathcal{L}_{\mathrm{SM}}(\phi) = \frac{1}{2}\mathbb{E}_{p_{\mathrm{data}}(x)}\left[\|\nabla_x \log p_\phi(x) - \nabla_x \log p_{\mathrm{data}}(x)\|_2^2\right]$$

- The Problem: The data score $\nabla_x \log p_{\mathrm{data}}(x)$ is unknown and inaccessible.

# Implicit Score Matching

- **The Solution:** Using integration by parts, we can rewrite the objective to eliminate the unknown data score.
- This yields an equivalent expression dependent only on the energy function and its derivatives.

## Implicit Score Matching Loss

$$\mathcal{L}_{\mathrm{SM}}(\phi) = \mathbb{E}_{p_{\mathrm{data}}(\mathrm{x})}\left[\underbrace{\mathrm{Tr}(\nabla_{\mathrm{x}}^2 E_\phi(\mathrm{x}))}_{\text{Hessian Trace}} + \frac{1}{2}\underbrace{\|\nabla_{\mathrm{x}} E_\phi(\mathrm{x})\|_2^2}_{\text{Squared Gradient}}\right] + C$$

where $\nabla_{\mathrm{x}}^2 E_\phi(\mathrm{x})$ is the Hessian of $E_\phi$ and $C$ is a constant independent of $\phi$.

- **Implication:** Can train EBMs effectively using only samples from $p_{\mathrm{data}}$!

# Analysis

- **Advantages:**
  - No Partition Function: $Z_\phi$ is completely eliminated.
  - No MCMC Sampling: Unlike Contrastive Divergence, we do not need to sample from the model $p_\phi(\mathrm{x})$ during training.

- **Limitations:**
  - Computational Cost: The objective requires computing the trace of the Hessian matrix $\mathrm{Tr}(\nabla_{\mathrm{x}}^2 E_\phi(\mathrm{x}))$.
  - For high-dimensional data (e.g., images), calculating second-order derivatives is computationally prohibitive (scaling with $D^2$ or requiring multiple backward passes).

**Outlook:** We will address this scalability issue using *Denoising Score Matching* and *Sliced Score Matching* later.

# Discrete-Time Langevin Dynamics

- **Objective:** Sample from an unnormalized density $p_\phi(\mathrm{x}) \propto e^{-E_\phi(\mathrm{x})}$.
- **Mechanism:** Langevin dynamics defines an iterative stochastic process that evolves a random initialization $\mathrm{x}_0$ over time.

## The Energy-Based Update Rule

Given a step size $\eta > 0$ and Gaussian noise $\epsilon_n \sim \mathcal{N}(0, \mathrm{I})$:

$$\mathrm{x}_{n+1} = \mathrm{x}_n - \underbrace{\eta \nabla_{\mathrm{x}} E_\phi(\mathrm{x}_n)}_{\text{Gradient Descent}} + \underbrace{\sqrt{2\eta}\epsilon_n}_{\text{Brownian Motion}}$$

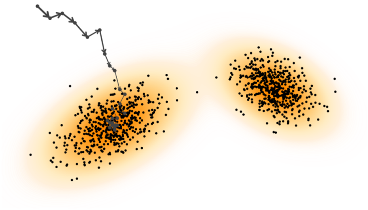
- The Gradient pushes the sample "downhill" toward low-energy (stable) states.
- The Noise prevents the sample from getting trapped in local minima.
- The factor $\sqrt{2}$ is to ensure the stationary distribution is exactly $p_\phi(\mathrm{x})$.

# Discrete-Time Langevin Dynamics

- **Connecting to Scores:** $\nabla_x \log p_\phi(x) = -\nabla_x E_\phi(x)$.

## The Score-Based Update Rule

$$x_{n+1} = x_n + \eta \underbrace{\nabla_x \log p_\phi(x_n)}_{\text{Guides to High Density}} + \sqrt{2\eta}\epsilon_n$$



The score field guides the noisy trajectory toward the data manifold.

# Continuous-Time Langevin SDE

- As the step size $\eta \to 0$, the discrete update converges to a Stochastic Differential Equation (SDE).

**Langevin SDE**

$$dx(t) = \nabla_x \log p_\phi(x(t))dt + \sqrt{2}dw(t)$$

where $w(t)$ is a standard Brownian motion.

- **Stationarity:** Under standard regularity conditions, the distribution of $x(t)$ converges to $p_\phi(x)$ as $t \to \infty$.
- The discrete update is essentially the *Euler-Maruyama discretization* of this SDE.

# Physical Intuition: Escaping Local Minima

- **Deterministic Gradient Descent (ODE):**

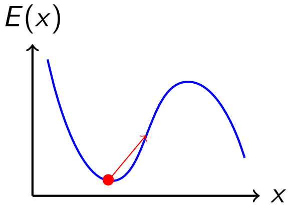
$$dx = -\nabla E(x)dt$$

Particles roll downhill and get trapped in the nearest local minimum.

- **Langevin Dynamics (SDE):**

$$dx = -\nabla E(x)dt + \sqrt{2}dw$$

The injected noise allows particles to escape local minima by crossing energy barriers.



# Inherent Challenges: Mixing Time

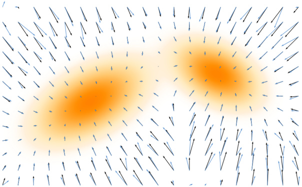
- While theoretically sound, Langevin dynamics faces serious practical limitations in high dimensions.
- **The Mixing Problem:**
  - Real data distributions have isolated modes (distant valleys in the energy landscape).
  - To jump between modes, the noise must be large enough to cross barriers, but small enough to remain accurate.
  - In high dimensions, the empty space between modes is vast, leading to prohibitively slow convergence.
- **Conclusion:** Standard Langevin dynamics struggles to explore the full diversity of the data distribution efficiently.
- **Solution:** This motivates the use of **multiple noise levels** (NCSN), which we discuss next.

# Score Matching: The Objective

- **Goal:** Approximate the true data score $s(x) = \nabla_x \log p_{\mathrm{data}}(x)$ using a neural network $s_\phi(x)$.
- **Loss Function:** Minimize the MSE between the model and the score:

**Explicit Score Matching Objective**

$$\mathcal{L}_{\mathrm{SM}}(\phi) := \frac{1}{2}\mathbb{E}_{x\sim p_{\mathrm{data}}}\left[\|s_\phi(x) - s(x)\|_2^2\right] \tag{3.2.1}$$



The neural network $s_\phi(x)$ is trained to match the ground truth vector field.

# HyvÃď'rinen's Tractable Form

- Rewrite the objective to depend only on the model $s_\phi$ and data samples.

**Theorem 1 (Tractable Score Matching)**

*The score matching objective is equivalent to:*

$$\mathcal{L}_{\mathrm{SM}}(\phi) = \tilde{\mathcal{L}}_{\mathrm{SM}}(\phi) + C \tag{3.1}$$

$$\tilde{\mathcal{L}}_{\mathrm{SM}}(\phi) := \mathbb{E}_{x \sim p_{\mathrm{data}}(x)}\left[\mathrm{Tr}(\nabla_x s_\phi(x)) + \frac{1}{2}\|s_\phi(x)\|_2^2\right] \tag{3.2.2}$$

*and $C$ is a constant independent of $\phi$.*

- This eliminates the need for the true score $s(x)$.
- **Note:** $\mathrm{Tr}(\nabla_x s_\phi(x))$ involves the Jacobian of the score network.

# Proof of Proposition 3.2.1 (Part I)

**Step 1: Expand the MSE.**

$$\begin{aligned}\mathcal{L}_{\mathrm{SM}}(\phi) &= \frac{1}{2}\mathbb{E}_{p_{\mathrm{data}}}\left[\|\mathsf{s}_\phi(\mathsf{x})\|_2^2 - 2\langle \mathsf{s}_\phi(\mathsf{x}), \mathsf{s}(\mathsf{x})\rangle + \|\mathsf{s}(\mathsf{x})\|_2^2\right] \\ &= \underbrace{\frac{1}{2}\mathbb{E}_{p_{\mathrm{data}}}\left[\|\mathsf{s}_\phi(\mathsf{x})\|_2^2\right]}_{\text{computable}} - \underbrace{\mathbb{E}_{p_{\mathrm{data}}}[\langle \mathsf{s}_\phi(\mathsf{x}), \mathsf{s}(\mathsf{x})\rangle]}_{\text{cross-term}} + \underbrace{C}_{\text{constant}}\end{aligned}$$

**Step 2: Analyze the Cross-Term.** Using $\mathsf{s}(\mathsf{x}) = \nabla_{\mathsf{x}} \log p_{\mathrm{data}}(\mathsf{x}) = \frac{\nabla_{\mathsf{x}} p_{\mathrm{data}}(\mathsf{x})}{p_{\mathrm{data}}(\mathsf{x})}$:

$$\begin{aligned}\mathbb{E}_{p_{\mathrm{data}}}[\langle \mathsf{s}_\phi(\mathsf{x}), \mathsf{s}(\mathsf{x})\rangle] &= \int \mathsf{s}_\phi(\mathsf{x})^\top \frac{\nabla_{\mathsf{x}} p_{\mathrm{data}}(\mathsf{x})}{p_{\mathrm{data}}(\mathsf{x})} p_{\mathrm{data}}(\mathsf{x}) d\mathsf{x} \\ &= \sum_{i=1}^{D} \int s_\phi^{(i)}(\mathsf{x}) \frac{\partial p_{\mathrm{data}}(\mathsf{x})}{\partial x_i} d\mathsf{x}\end{aligned}$$

# Proof of Proposition 3.2.1 (Part II)

**Step 3: Integration by Parts.** Recall the formula for differentiable functions $u, v$ vanishing at boundaries:

$$\int u(\mathrm{x}) \frac{\partial v(\mathrm{x})}{\partial x_i} d\mathrm{x} = -\int v(\mathrm{x}) \frac{\partial u(\mathrm{x})}{\partial x_i} d\mathrm{x}$$

Setting $u = s_\phi^{(i)}(\mathrm{x})$ and $v = p_{\mathrm{data}}(\mathrm{x})$:

$$\int s_\phi^{(i)}(\mathrm{x}) \partial_{x_i} p_{\mathrm{data}}(\mathrm{x}) d\mathrm{x} = -\int p_{\mathrm{data}}(\mathrm{x}) \partial_{x_i} s_\phi^{(i)}(\mathrm{x}) d\mathrm{x} \quad = -\mathbb{E}_{p_{\mathrm{data}}} \left[ \frac{\partial s_\phi^{(i)}(\mathrm{x})}{\partial x_i} \right]$$

Summing over all $i = 1 \dots D$:

$$\mathbb{E}_{p_{\mathrm{data}}}[\langle \mathrm{s}_\phi, \mathrm{s} \rangle] = -\mathbb{E}_{p_{\mathrm{data}}} \left[ \sum_{i=1}^{D} \partial_{x_i} s_\phi^{(i)} \right] = -\mathbb{E}_{p_{\mathrm{data}}}[\mathrm{Tr}(\nabla_{\mathrm{x}} \mathrm{s}_\phi(\mathrm{x}))]$$

# Intuition: Shaping the Landscape

The loss $\tilde{\mathcal{L}}_{\mathrm{SM}}(\phi)$ has two competing terms:

$$\mathbb{E}_{p_{\mathrm{data}}}\left[\underbrace{\mathrm{Tr}(\nabla_{\mathsf{x}} \mathsf{s}_\phi(\mathsf{x}))}_{\text{Divergence}} + \frac{1}{2}\underbrace{\|\mathsf{s}_\phi(\mathsf{x})\|_2^2}_{\text{Magnitude}}\right]$$

1. **Magnitude Term** ($\|\mathsf{s}_\phi\|^2$):
   - Forces the score to be **zero** (stationary) in regions where data density $p_{\mathrm{data}}$ is high.
2. **Divergence Term** ($\mathrm{Tr}(\nabla_{\mathsf{x}} \mathsf{s}_\phi)$):
   - Encourages **negative divergence** (sink) at these high-density regions.
   - Vectors point *inward* towards the data, making the stationary points stable attractors.

*Result: Data points become stable "sinks" in the vector field.*

# Sampling with the Learned Score

- Once the score network $s_{\phi^*}(x)$ is trained (by minimizing Eq. 3.2.2), we use it to replace the unknown oracle score.
- **Sampling Algorithm:** Initialize $x_0$ from a prior (e.g., Gaussian) and iterate:

**Discrete Langevin Sampling**

$$x_{n+1} = x_n + \eta s_{\phi^*}(x_n) + \sqrt{2\eta}\varepsilon_n, \quad \varepsilon_n \sim \mathcal{N}(0, I) \tag{3.2.3}$$

- This allows us to generate data samples solely using the learned vector field, without ever normalizing a probability density.

# Continuous Connection and Stability

- **SDE Limit:** As $\eta \to 0$, the discrete update becomes the Euler Maruyama discretization of the continuous Langevin SDE:

$$d\mathsf{x}(t) = \mathsf{s}_{\phi^*}(\mathsf{x}(t))dt + \sqrt{2}d\mathsf{w}(t)$$

## Technical Remark: Trace vs. Definiteness

The training objective minimizes $\mathrm{Tr}(\nabla_{\mathsf{x}} \mathsf{s}_{\phi})$.

- A negative trace implies the *sum* of eigenvalues is negative.
- It does **not** guarantee that *all* eigenvalues are negative (which is required for a strict local maximum).
- **Implication:** The dynamics might stabilize at saddle points rather than true peaks, though stochastic noise helps escape these.

# Prologue: The Rise of Score-Based Models

- **A Shift in Perspective:**
  - Initially, the score function was just a "trick" to train Energy-Based Models (EBMs) without calculating partition functions.
  - It has evolved into the central component of modern **Diffusion Models**.
- **The Core Paradigm:**
  - Instead of modeling the density $p(\mathrm{x})$ directly, we model the vector field $\nabla_{\mathrm{x}} \log p(\mathrm{x})$.
  - This offers a principled framework for data generation via **Stochastic Differential Equations (SDEs)**.
- **Coming Up:**
  - We will now explore **Denoising Score Matching** and **NCSN**, which solve the practical scalability issues of the methods discussed so far.

# The Computational Bottleneck

- **Recap:** The tractable objective (Eq. 3.2.2) eliminated the true score but introduced a new problem:

$$\tilde{\mathcal{L}}_{\mathrm{SM}}(\phi) = \mathbb{E}_{p_{\mathrm{data}}}\left[\mathrm{Tr}(\nabla_{\mathsf{x}} \mathsf{s}_\phi(\mathsf{x})) + \frac{1}{2}\|\mathsf{s}_\phi(\mathsf{x})\|_2^2\right]$$

## The Complexity Issue

- Computing $\mathrm{Tr}(\nabla_{\mathsf{x}} \mathsf{s}_\phi(\mathsf{x}))$ requires the Jacobian of the output with respect to the input.
- For high-dimensional data (e.g., dimension $D$), this scales with $O(D^2)$.
- This requires determining $D$ backward passes, which is computationally prohibitive for deep networks.

# A Partial Solution: Sliced Score Matching

- **Idea:** Replace the expensive Trace calculation with a stochastic estimate using random projections [Song et al., 2020].
- **Hutchinson's Estimator:** For a random vector $\mathbf{u} \sim \mathcal{N}(0, \mathbf{I})$:

$$\mathrm{Tr}(A) = \mathbb{E}_{\mathbf{u}}[\mathbf{u}^\top A \mathbf{u}]$$

### Sliced Score Matching Objective

$$\mathcal{L}_{\mathrm{SSM}}(\phi) = \mathbb{E}_{\mathbf{x},\mathbf{u}}\left[\mathbf{u}^\top(\nabla_{\mathbf{x}} \mathbf{s}_\phi(\mathbf{x}))\mathbf{u} + \frac{1}{2}(\mathbf{u}^\top \mathbf{s}_\phi(\mathbf{x}))^2\right]$$

- **Benefit:** The term $\mathbf{u}^\top(\nabla_{\mathbf{x}} \mathbf{s}_\phi)\mathbf{u}$ can be computed via *Jacobian-Vector Products (JVP)*, which is efficient in auto-diff libraries (scaling with $O(D)$).

# From Sliced to Denoising Score Matching

- Sliced SM solves the computation speed, but fundamental theoretical issues remain.

## The Manifold Hypothesis

- Real-world data (like images) typically lies on a **low-dimensional manifold** embedded in high-dimensional space.
- Problem: The score $\nabla_{\mathrm{x}} \log p_{\mathrm{data}}(\mathrm{x})$ is undefined or unstable when $\mathrm{x}$ is not on the manifold.

- **Consequence:** Score matching only constrains the model on the data points. The score field in the ambient space (where we start sampling noise) remains undefined.
- **Solution:** Denoising Score Matching (DSM) [Vincent, 2011].
  - By adding noise to the data, we "fill" the ambient space, making the score defined everywhere.

# Overcoming Intractability via Conditioning

- **The Problem:** Explicit Score Matching requires the data score $\nabla_{\mathrm{x}} \log p_{\mathrm{data}}(\mathrm{x})$, which is unknown.
- **Vincent's Solution (2011):** Inject noise into the data $\mathrm{x} \sim p_{\mathrm{data}}$ using a known conditional distribution $p_\sigma(\tilde{\mathrm{x}}|\mathrm{x})$.

## The Goal

Train a network $\mathrm{s}_\phi(\tilde{\mathrm{x}}; \sigma)$ to approximate the score of the **perturbed marginal distribution**:

$$p_\sigma(\tilde{\mathrm{x}}) = \int p_\sigma(\tilde{\mathrm{x}}|\mathrm{x}) p_{\mathrm{data}}(\mathrm{x}) d\mathrm{x}$$

- **The Intractable Objective ($L_{\mathrm{SM}}$):** Trying to match the marginal score directly is still hard:

$$\mathcal{L}_{\mathrm{SM}}(\phi; \sigma) := \frac{1}{2} \mathbb{E}_{\tilde{\mathrm{x}} \sim p_\sigma} \left[ \| \mathrm{s}_\phi(\tilde{\mathrm{x}}; \sigma) - \nabla_{\tilde{\mathrm{x}}} \log p_\sigma(\tilde{\mathrm{x}}) \|_2^2 \right]$$

# The Denoising Score Matching (DSM) Objective

- While $\nabla_{\tilde{x}} \log p_\sigma(\tilde{x})$ is intractable, the conditional score $\nabla_{\tilde{x}} \log p_\sigma(\tilde{x}|x)$ is known (since we define the noise model).

## DSM Loss [Vincent, 2011]

Conditioning on x yields a tractable equivalent objective:

$$\mathcal{L}_{\mathrm{DSM}}(\phi;\sigma) := \frac{1}{2}\mathbb{E}_{x\sim p_{\mathrm{data}},\tilde{x}\sim p_\sigma(\cdot|x)}\left[\|s_\phi(\tilde{x};\sigma) - \nabla_{\tilde{x}} \log p_\sigma(\tilde{x}|x)\|_2^2\right] \quad (3.3.2)$$

## Insight: The Conditioning Technique

This mirrors the variational view in DDPM (Theorem 2.2.1). Conditioning on a clean data point x turns an intractable marginal loss into a tractable conditional one.

# Equivalence of Objectives

**Theorem 2 (Equivalence of LSM and LDSM)**

*For any fixed noise scale* $\sigma > 0$,

$$\mathcal{L}_{\mathrm{SM}}(\phi;\sigma) = \mathcal{L}_{\mathrm{DSM}}(\phi;\sigma) + C \tag{3.3.3}$$

*where* $C$ *is a constant independent of* $\phi$. *Furthermore, the minimizer* $\mathrm{s}^*$ *satisfies* $\mathrm{s}^*(\tilde{\mathrm{x}};\sigma) = \nabla_{\tilde{\mathrm{x}}} \log p_\sigma(\tilde{\mathrm{x}})$ *almost everywhere.*

**Significance:** Minimizing the tractable DSM loss (Eq 3.3.2) yields the optimal score for the marginal distribution $p_\sigma(\tilde{\mathrm{x}})$, which is exactly what we need for generation.

# Proof of Equivalence

**Sketch:** We expand the squared norms in both objectives.

- focus on the cross-terms dependent on $\phi$:

1. **In $\mathcal{L}_{\mathrm{SM}}$:** $\mathbb{E}_{\tilde{\mathsf{x}}\sim p_\sigma}[\langle \mathsf{s}_\phi, \nabla \log p_\sigma(\tilde{\mathsf{x}})\rangle] = \int \mathsf{s}_\phi(\tilde{\mathsf{x}})^\top \nabla p_\sigma(\tilde{\mathsf{x}}) d\tilde{\mathsf{x}}$
2. **In $\mathcal{L}_{\mathrm{DSM}}$:**

$$
\begin{aligned}
&\mathbb{E}_{\mathsf{x}\sim p_{\mathrm{data}}}\mathbb{E}_{\tilde{\mathsf{x}}\sim p_\sigma(\cdot|\mathsf{x})}[\langle \mathsf{s}_\phi, \nabla_{\tilde{\mathsf{x}}} \log p_\sigma(\tilde{\mathsf{x}}|\mathsf{x})\rangle] \\
&= \int p_{\mathrm{data}}(\mathsf{x}) \int p_\sigma(\tilde{\mathsf{x}}|\mathsf{x}) \mathsf{s}_\phi(\tilde{\mathsf{x}})^\top \frac{\nabla_{\tilde{\mathsf{x}}} p_\sigma(\tilde{\mathsf{x}}|\mathsf{x})}{p_\sigma(\tilde{\mathsf{x}}|\mathsf{x})} d\tilde{\mathsf{x}} d\mathsf{x} \\
&= \int \mathsf{s}_\phi(\tilde{\mathsf{x}})^\top \left( \int p_{\mathrm{data}}(\mathsf{x}) \nabla_{\tilde{\mathsf{x}}} p_\sigma(\tilde{\mathsf{x}}|\mathsf{x}) d\mathsf{x} \right) d\tilde{\mathsf{x}} \\
&= \int \mathsf{s}_\phi(\tilde{\mathsf{x}})^\top \nabla_{\tilde{\mathsf{x}}} \left( \int p_{\mathrm{data}}(\mathsf{x}) p_\sigma(\tilde{\mathsf{x}}|\mathsf{x}) d\mathsf{x} \right) d\tilde{\mathsf{x}} = \int \mathsf{s}_\phi(\tilde{\mathsf{x}})^\top \nabla p_\sigma(\tilde{\mathsf{x}}) d\tilde{\mathsf{x}}
\end{aligned}
$$

Since the gradient terms match, the objectives differ only by a constant $C$.
■

# Special Case: Additive Gaussian Noise

Consider perturbing data with: $\tilde{x} = x + \sigma\varepsilon$, where $\varepsilon \sim \mathcal{N}(0, I)$.

$$p_\sigma(\tilde{x}|x) = \mathcal{N}(\tilde{x}; x, \sigma^2 I)$$

$$\nabla_{\tilde{x}} \log p_\sigma(\tilde{x}|x) = \frac{x - \tilde{x}}{\sigma^2} = \frac{-\sigma\varepsilon}{\sigma^2} = -\frac{\varepsilon}{\sigma}$$

## Gaussian DSM Loss

Substituting this into Eq (3.3.2):

$$\mathcal{L}_{\mathrm{DSM}}(\phi; \sigma) = \frac{1}{2}\mathbb{E}_{x,\varepsilon}\left[\left\| s_\phi(x + \sigma\varepsilon; \sigma) + \frac{\varepsilon}{\sigma}\right\|_2^2\right] \tag{3.3.4}$$

- This is a simple regression problem!
- For small $\sigma$, $\nabla \log p_\sigma(\tilde{x}) \approx \nabla \log p_{\mathrm{data}}(x)$.

# Sampling with DSM

- Once trained, we have a score model $s_{\phi^*}(\tilde{x}; \sigma)$ that approximates $\nabla_{\tilde{x}} \log p_\sigma(\tilde{x})$.
- We generate samples using Langevin dynamics by plugging in this learned score.

## Langevin Update for DSM

For a fixed noise level $\sigma$ (assumed small) and step size $\eta$:

$$\tilde{x}_{n+1} = \tilde{x}_n + \eta \underbrace{s_{\phi^*}(\tilde{x}_n; \sigma)}_{\approx \nabla \log p_\sigma(\tilde{x}_n)} + \sqrt{2\eta}\varepsilon_n, \quad \varepsilon_n \sim \mathcal{N}(0, I) \tag{3.3.5}$$

- If $\sigma$ is sufficiently small, $p_\sigma \approx p_{\mathrm{data}}$, so samples approximate the real data distribution.

# Why Inject Noise? Two Key Advantages

Compared to vanilla score matching, defining the target on $p_\sigma$ (the perturbed distribution) solves two major problems [Song and Ermon, 2019]:

1. **Well-Defined Gradients (Manifold Hypothesis):**
   - Data often lies on a low-dimensional manifold. The score is undefined off-manifold.
   - Gaussian noise "fills" the ambient space, ensuring $p_\sigma$ has full support on $\mathbb{R}^D$.
   - **Result:** The score $\nabla_{\tilde{x}} \log p_\sigma(\tilde{x})$ is defined everywhere.
2. **Improved Coverage (Mixing):**
   - Without noise, regions between data modes have near-zero density (vanishing gradients).
   - Noise "bridges" these gaps, allowing Langevin dynamics to traverse low-density regions and mix between modes effectively.

# Tweedie's Formula: The Link to Denoising

- How does learning a gradient (score) relate to removing noise?
- **Tweedie's Formula [Efron, 2011]** provides the answer: the score of the marginal distribution implicitly points to the clean data mean.

**Lemma 3 (Tweedie's Formula)**

*Assume* $\mathrm{x} \sim p_{\text{data}}$ *and* $\tilde{\mathrm{x}} \sim \mathcal{N}(\cdot; \alpha\mathrm{x}, \sigma^2\mathrm{I})$ *with* $\alpha \neq 0$. *Then:*

$$\alpha\mathbb{E}[\mathrm{x} \mid \tilde{\mathrm{x}}] = \tilde{\mathrm{x}} + \sigma^2 \nabla_{\tilde{\mathrm{x}}} \log p_\sigma(\tilde{\mathrm{x}}) \tag{3.3.6}$$

*where the expectation is over the posterior* $p(\mathrm{x} \mid \tilde{\mathrm{x}})$.

- **Insight:** The expected clean signal is obtained by nudging the noisy observation $\tilde{\mathrm{x}}$ in the direction of the score.

# Proof of Tweedie's Formula (Part I)

**Step 1: Expand the Score of the Marginal.** $p_\sigma(\tilde{x}) = \int p(\tilde{x}|x)p_{\text{data}}(x)dx$.

$$\nabla_{\tilde{x}} \log p_\sigma(\tilde{x}) = \frac{\nabla_{\tilde{x}} p_\sigma(\tilde{x})}{p_\sigma(\tilde{x})} = \frac{1}{p_\sigma(\tilde{x})} \int \nabla_{\tilde{x}} p(\tilde{x}|x)p_{\text{data}}(x)dx$$

**Step 2: Compute the Gradient of the Likelihood.**

$$p(\tilde{x}|x) = \mathcal{N}(\tilde{x}; \alpha x, \sigma^2 I) \propto \exp\left(-\frac{\|\tilde{x} - \alpha x\|^2}{2\sigma^2}\right)$$

$$\begin{aligned}\nabla_{\tilde{x}} p(\tilde{x}|x) &= p(\tilde{x}|x) \cdot \nabla_{\tilde{x}} \left(-\frac{\|\tilde{x} - \alpha x\|^2}{2\sigma^2}\right) \\ &= p(\tilde{x}|x) \cdot \left(-\frac{(\tilde{x} - \alpha x)}{\sigma^2}\right) = \frac{\alpha x - \tilde{x}}{\sigma^2} p(\tilde{x}|x)\end{aligned}$$

# Proof of Tweedie's Formula (Part II)

**Step 3: Substitute and Identify the Posterior.**

$$\nabla_{\tilde{x}} \log p_\sigma(\tilde{x}) = \frac{1}{p_\sigma(\tilde{x})} \int \frac{(\alpha x - \tilde{x})}{\sigma^2} p(\tilde{x}|x) p_{\mathrm{data}}(x) dx$$
$$= \frac{1}{\sigma^2} \int (\alpha x - \tilde{x}) \underbrace{\frac{p(\tilde{x}|x) p_{\mathrm{data}}(x)}{p_\sigma(\tilde{x})}}_{\text{Bayes' Rule: } p(x|\tilde{x})} dx$$

**Step 4: Separate Terms.**

$$\sigma^2 \nabla_{\tilde{x}} \log p_\sigma(\tilde{x}) = \underbrace{\int \alpha x \, p(x|\tilde{x}) dx}_{\alpha \mathbb{E}[x|\tilde{x}]} - \tilde{x} \underbrace{\int p(x|\tilde{x}) dx}_{=1 \text{ (Normalization)}}$$

$$\alpha \mathbb{E}[x \mid \tilde{x}] = \tilde{x} + \sigma^2 \nabla_{\tilde{x}} \log p_\sigma(\tilde{x})$$

# Interpretation: Score Estimation $\iff$ Denoising

- Tweedie's formula establishes a fundamental link between the Score Function and the Optimal Denoiser.

**The Denoiser**

The posterior mean $\hat{x} = \mathbb{E}[x|\tilde{x}]$ is the optimal Minimum Mean MSE denoiser. Using Tweedie's formula:

$$\hat{x}(\tilde{x}) = \frac{1}{\alpha}\left(\tilde{x} + \sigma^2 \nabla_{\tilde{x}} \log p_\sigma(\tilde{x})\right)$$

- **Intuition:** A single gradient ascent step on the noisy log-likelihood (with step size $\sigma^2$) recovers the expected clean signal.
- **Connection to DDPM:** If we train a model $s_\phi \approx \nabla \log p_\sigma$, we are implicitly learning a denoiser. This explains why predicting noise $\epsilon$ (in DDPM) is equivalent to predicting the score.

# Extension: Higher Order Tweedie's Formula

- **Recap:** The first derivative (Score) $\nabla \log p_\sigma$ gives the posterior mean (Clean Data).
- **Extension:** Higher derivatives of the log-density relate to higher-order cumulants of the posterior (e.g., uncertainty).

**Second-Order Tweedie**

For Gaussian noise, the posterior covariance is given by the Hessian of the log-density:

$$\mathrm{Cov}[\mathrm{x} \mid \tilde{\mathrm{x}}] = \sigma^2 \mathrm{I} + \sigma^4 \nabla^2_{\tilde{\mathrm{x}}} \log p_\sigma(\tilde{\mathrm{x}})$$

- **Significance:** By learning higher-order scores, we can estimate not just the denoised image, but also the *uncertainty* of that estimate.

# Why DSM is Denoising: The SURE Perspective

- **Problem:** To train a denoiser $D(\tilde{x})$, we typically need clean data pairs $(x, \tilde{x})$ to minimize MSE.
- **SURE [Stein, 1981]** Allows estimating the MSE of a denoiser using *only* noisy data $\tilde{x}$.

## SURE Objective

For additive Gaussian noise $\tilde{x} = x + \sigma\varepsilon$:

$$\mathrm{SURE}(D; \tilde{x}) = \| \underbrace{D(\tilde{x}) - \tilde{x}}_{\text{Residual}} \|_2^2 + 2\sigma^2 \underbrace{\nabla_{\tilde{x}} \cdot D(\tilde{x})}_{\text{Divergence}} - D\sigma^2 \quad (3.3.7)$$

- **Key Property:** $\mathbb{E}_{\tilde{x}|x}[\mathrm{SURE}(D; \tilde{x})] = \mathbb{E}_{\tilde{x}|x}[\|D(\tilde{x}) - x\|^2]$.
- Minimizing SURE $\iff$ Minimizing true MSE.

# Equivalence: SURE and Score Matching

- Let's parameterize a denoiser using a score field via Tweedie's formula:

$$D(\tilde{\mathrm{x}}) = \tilde{\mathrm{x}} + \sigma^2 \mathrm{s}_\phi(\tilde{\mathrm{x}})$$

- Plugging this into the SURE objective yields:

**The Connection**

$$\frac{1}{2\sigma^4}\mathrm{SURE}(D) \equiv \underbrace{\mathbb{E}_{p_\sigma}\left[\mathrm{Tr}(\nabla \mathrm{s}_\phi) + \frac{1}{2}\|\mathrm{s}_\phi\|^2\right]}_{\text{Implicit Score Matching (Eq 3.2.2)}} + C$$

- **Conclusion:** Minimizing SURE (denoising without clean data) is mathematically equivalent to Score Matching!
- They both lead to the optimal Bayes denoiser: $D^*(\tilde{\mathrm{x}}) = \mathbb{E}[\mathrm{x}|\tilde{\mathrm{x}}]$.

# Generalized Score Matching

- Can we unify all these methods? Yes, via **Generalized Fisher Divergence**.
- Let $\mathcal{L}$ be a linear operator. We want to match the "generalized score" $\frac{\mathcal{L}p(\mathrm{x})}{p(\mathrm{x})}$.

## GSM Objective

Using integration by parts with the adjoint operator $\mathcal{L}^\dagger$:

$$\mathcal{L}_{\mathrm{GSM}}(\phi) = \mathbb{E}_{\mathrm{x}\sim p}\left[\frac{1}{2}\|\mathrm{s}_\phi(\mathrm{x})\|^2 - (\mathcal{L}^\dagger \mathrm{s}_\phi)(\mathrm{x})\right]$$

# Examples of Operators

| **Method** | **Operator $\mathcal{L}$** | **Target** |
|---|---|---|
| Classical Score Matching | $\nabla_x$ | $\nabla \log p(x)$ |
| Denoising Score Matching | $\tilde{x} + \sigma^2 \nabla_{\tilde{x}}$ | $\mathbb{E}[x \mid \tilde{x}]$ (Tweedie) |
| Higher Order Matching | $\nabla\nabla \ldots$ | Cumulants |

- **Takeaway:** This operator view unifies SM, DSM, and SURE into a single framework, allowing us to design new objectives by choosing suitable operators.

# The Dilemma of Single Noise Level

- **Recap:** In DSM, we perturb data with noise level $\sigma$.
- Training score model at a **fixed** $\sigma$ introduces a fundamental trade-off:

### Low Noise ($\sigma \approx 0$)

- **Pros:** The distribution $p_\sigma \approx p_{\mathrm{data}}$. Samples have high fidelity and fine details.
- **Cons:** Data resides on disjoint manifolds. Gradients vanish in low-density regions.
- **Result:** Langevin dynamics gets stuck in local modes (poor mixing).

### High Noise ($\sigma \gg 0$)

- **Pros:** The distribution is smooth; modes merge. Gradients are defined everywhere.
- **Cons:** The distribution $p_\sigma$ is far from $p_{\mathrm{data}}$.
- **Result:** Langevin dynamics mixes well, but produces blurry, coarse samples.

# Inaccurate Scores in Low-Density Regions

- Score estimation is only accurate in regions covered by data samples.
- In high-dim space, the volume of "empty space" (low density) is vast.

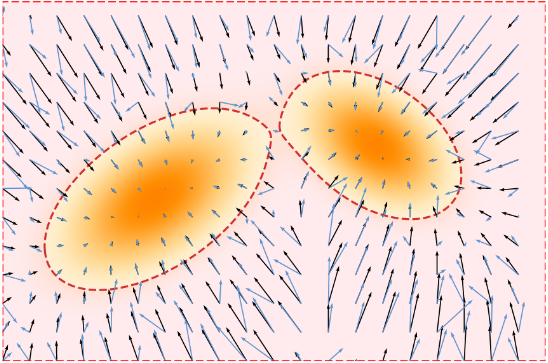


Illustration of SM inaccuracy. High-density regions (data) yield accurate scores, while low-density regions (red) yield random/inaccurate estimates.

# Noise Conditional Score Networks (NCSN)

## Key Idea: Multi-Scale Noise Perturbation

Instead of a single $\sigma$, consider a sequence of noise levels $\{\sigma_i\}_{i=1}^{L}$ such that:

$$\sigma_1 > \sigma_2 > \cdots > \sigma_L > 0$$

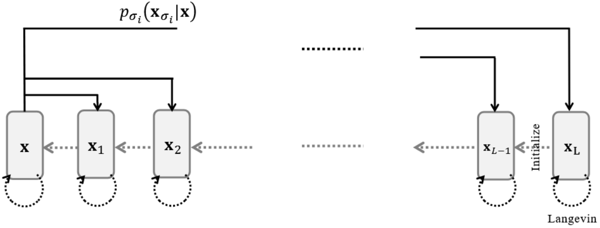
- $\sigma_1$ is large enough to smooth out all modes (easy exploration).
- $\sigma_L$ is small enough to approximate the clean data (fine details).

- **Joint Training:** We train a single Noise Conditional Score Network (NCSN) $s_\theta(\mathrm{x}, \sigma)$ to estimate the scores for **all** levels simultaneously:

$$s_\theta(\mathrm{x}, \sigma_i) \approx \nabla_{\mathrm{x}} \log p_{\sigma_i}(\mathrm{x})$$

# Annealed Langevin Dynamics

- **Generation Strategy:** Coarse-to-fine generation.




The sampler starts at high noise ($\sigma_1$) to find the general mode, then uses that result to initialize the next level ($\sigma_2$), gradually refining details down to $\sigma_L$.

- By the time we reach the tricky low-noise levels ($\sigma_L$), the sample is already close to the data manifold, avoiding the "trapped in low-density" problem.

# NCSN: Formal Setup

- **Noise Sequence:** geometric sequence of $L$ noise levels $\{\sigma_i\}_{i=1}^{L}$ such that:

$$0 < \sigma_1 < \sigma_2 < \cdots < \sigma_L$$

  - $\sigma_1$: Small enough to preserve fine details (approx. clean data).
  - $\sigma_L$: Large enough to smooth the distribution and bridge modes.

- **Perturbation Kernel:** For a clean data point $\mathrm{x} \sim p_{\mathrm{data}}$, the perturbed sample at level $i$ is $\mathrm{x}_{\sigma_i} = \mathrm{x} + \sigma_i \varepsilon$.

$$p_{\sigma_i}(\mathrm{x}_{\sigma_i}|\mathrm{x}) := \mathcal{N}(\mathrm{x}_{\sigma_i}; \mathrm{x}, \sigma_i^2 \mathrm{I})$$

- **Marginal Distribution:**

$$p_{\sigma_i}(\mathrm{x}_{\sigma_i}) = \int p_{\sigma_i}(\mathrm{x}_{\sigma_i}|\mathrm{x}) p_{\mathrm{data}}(\mathrm{x}) d\mathrm{x}$$

# Training Objective of NCSN

- **Goal:** Train a single Noise-Conditional Score Network $s_\phi(x, \sigma)$ to estimate the scores $\nabla_x \log p_{\sigma_i}(x)$ for all $i$.
- **Loss Function:** We minimize the weighted sum of Denoising Score Matching (DSM) objectives across all levels:

## NCSN Loss Function

$$\mathcal{L}_{\mathrm{NCSN}}(\phi) := \sum_{i=1}^{L} \lambda(\sigma_i)\mathcal{L}_{\mathrm{DSM}}(\phi; \sigma_i) \tag{3.4.1}$$

$$\mathcal{L}_{\mathrm{DSM}}(\phi; \sigma_i) = \frac{1}{2}\mathbb{E}_{x,\tilde{x}}\left[\left\| s_\phi(\tilde{x}, \sigma_i) - \left(\frac{x - \tilde{x}}{\sigma_i^2}\right)\right\|_2^2\right]$$

- $\lambda(\sigma_i) > 0$ balances the magnitude of the loss (typically $\lambda(\sigma_i) \propto \sigma_i^2$).
- The minimizer satisfies $s^*(\cdot, \sigma_i) = \nabla \log p_{\sigma_i}(\cdot)$.

# Relationship with DDPM Loss

- Is there a difference between predicting the Score (NCSN) and predicting the Noise (DDPM)? **Mathematically, no.**

## Equivalence via Tweedie's Formula

Let $x_\sigma = x + \sigma\varepsilon$. Tweedie's formula relates the score to the posterior noise expectation:

$$\nabla_{x_\sigma} \log p_\sigma(x_\sigma) = -\frac{1}{\sigma}\mathbb{E}[\varepsilon | x_\sigma]$$

- **NCSN Optimum:** $s^*(x_\sigma, \sigma) = \nabla \log p_\sigma(x_\sigma)$.
- **DDPM Optimum:** $\epsilon^*(x_\sigma, \sigma) = \mathbb{E}[\varepsilon | x_\sigma]$.
- **Conclusion:** The models are equivalent up to a scaling factor:

$$s^*(x_\sigma, \sigma) = -\frac{1}{\sigma}\epsilon^*(x_\sigma, \sigma)$$

# Algorithm: Annealed Langevin Dynamics

**Algorithm 1** Annealed Langevin Dynamics

**Input** Trained score model $\mathrm{s}_{\phi^*}$, step sizes $\{\eta_l\}$, steps per level $N_l$.
  Initialize $\mathrm{x}_{\sigma_L} \sim \mathcal{N}(0, \mathrm{I})$ ▷ Start at highest noise level
  **for** $l = L$ **down to** 2 **do**
    $\tilde{\mathrm{x}}_0 \leftarrow \mathrm{x}_{\sigma_l}$ ▷ Initialize from previous level's output
    **for** $n = 0$ **to** $N_l - 1$ **do**
      $\varepsilon_n \sim \mathcal{N}(0, \mathrm{I})$
      $\tilde{\mathrm{x}}_{n+1} \leftarrow \tilde{\mathrm{x}}_n + \eta_l \mathrm{s}_{\phi^*}(\tilde{\mathrm{x}}_n, \sigma_l) + \sqrt{2\eta_l}\varepsilon_n$
    **end for**
    $\mathrm{x}_{\sigma_{l-1}} \leftarrow \tilde{\mathrm{x}}_{N_l}$
  **end for**
  **Output** $\mathrm{x}_{\sigma_1}$ ▷ Final sample at lowest noise level

# Annealed Langevin Dynamics: The Procedure

- **Setup:** We have trained score networks for a sequence of noise levels:

$$\mathbf{s}_{\phi^*}(\cdot, \sigma_1), \ldots, \mathbf{s}_{\phi^*}(\cdot, \sigma_L) \quad \text{where } \sigma_L > \cdots > \sigma_1 \approx 0$$

- **Initialization:** Start from pure Gaussian noise: $\tilde{\mathbf{x}}_0 \sim \mathcal{N}(0, \mathbf{I})$ (corresponding to $\sigma_L$).

## Progressive Denoising

Applies Langevin dynamics at each level $\sigma_l$ to sample from $p_{\sigma_l}(\mathbf{x})$.

- **Crucial Step:** The final sample from level $\sigma_l$ is used as the initialization for the next, lower noise level $\sigma_{l-1}$.
- This "hand-off" strategy ensures the sampler is always initialized in a high-density region of the next distribution.

# Step Size Schedule and Intuition

- **Update Rule:** At each level $l$, we perform $K$ steps:

$$\tilde{x}_{n+1} = \tilde{x}_n + \eta_l s_{\phi^*}(\tilde{x}_n, \sigma_l) + \sqrt{2\eta_l}\varepsilon_n$$

- **Step Size Scaling:** The step size is not constant. It is scaled by the noise variance to maintain a constant Signal-to-Noise Ratio (SNR):

$$\eta_l = \delta \cdot \frac{\sigma_l^2}{\sigma_1^2} \propto \sigma_l^2$$

where $\delta > 0$ is a hyperparameter.

## Why Annealing Works

- **High Noise ($\sigma_L$):** Large steps traverse the space globally, finding the general location of data modes.
- **Low Noise ($\sigma_1$):** Small steps refine the local structure and texture.
- This prevents the sampler from getting trapped in isolated modes.

# The Bottleneck: Slow Sampling Speed

- While effective, NCSN suffers from significant computational costs.
- **Total Complexity:** $L$ noise levels $\times$ $K$ steps per level $= O(LK)$ sequential network evaluations.

**Why so many steps?**

1. **Local Accuracy & Stability:** The score network is only accurate for small perturbations. We need small step sizes (and thus many steps) to avoid integration errors or divergence.
2. **Slow Mixing:** Langevin dynamics is an MCMC process. In high dimensions, random walks explore the space inefficiently, requiring many iterations to converge to the target distribution at each level.

**Implication:** Generating a single image can take hundreds or thousands of forward passes, making real-time generation difficult compared to GANs.

# Comparison: Formulation (NCSN vs. DDPM)

- While NCSN originates from **Score Matching** (EBMs) and DDPM from **Variational Inference** (VAEs), they share striking similarities.

| Feature | NCSN | DDPM |
|---|---|---|
| **Marginal** $q(\mathrm{x}_i \mid \mathrm{x}_0)$ | $\mathrm{x} + \sigma_i \varepsilon$ | $\sqrt{\bar{\alpha}_i}\mathrm{x} + \sqrt{1 - \bar{\alpha}_i^2}\varepsilon$ |
| **Forward** $q(\mathrm{x}_{i+1} \mid \mathrm{x}_i)$ | $\mathrm{x}_i + \sqrt{\sigma_{i+1}^2 - \sigma_i^2}\varepsilon$ | $\sqrt{1 - \beta_i}\mathrm{x}_i + \sqrt{\beta_i}\varepsilon$ |
| **Prior** $p(\mathrm{x}_L)$ | $\mathcal{N}(0, \sigma_L^2 \mathrm{I})$ | $\mathcal{N}(0, \mathrm{I})$ |

Table 3.1: Comparison of the Forward Process formulations.

- **Key Difference:** NCSN explicitly defines the noise scale of marginals, while DDPM defines the incremental Markov transition.

# Comparison: Training and Sampling

| Feature | NCSN | DDPM |
|---|---|---|
| **Loss** | $\mathbb{E}\left[\left\|\mathrm{s}_\phi(\mathrm{x}_i, \sigma_i) + \frac{\varepsilon}{\sigma_i}\right\|^2\right]$ (Score Matching) | $\mathbb{E}\left[\left\|\epsilon_\phi(\mathrm{x}_i, i) - \varepsilon\right\|^2\right]$ (Noise Prediction) |
| **Sampling** | **Annealed Langevin:** Repeat $K$ steps of $\mathrm{x} + \eta \mathrm{s}_\phi + \dots$ at each level to mix. | **Reverse Markov Chain:** Single step prediction $p_\theta(\mathrm{x}_{i-1} \mid \mathrm{x}_i)$ to traverse the chain. |

Table 3.2: Comparison of Optimization and Generation.

*Note: The objectives are equivalent via* $\mathrm{s}_\phi \approx -\frac{\epsilon_\phi}{\sigma}$.

# A Shared Bottleneck

- Despite the differences in derivation, both models rely on **dense time discretization** (many noise levels).

**The Computational Cost**

- **NCSN:** Requires $L \times K$ Langevin steps (often thousands).
- **DDPM:** Requires traversing $T$ timesteps (often $T = 1000$).
- **Result:** Sampling is slow and computationally intensive compared to GANs or VAEs.

**Question 3.5.1**

*How can we accelerate sampling in diffusion models?*

**Outlook:** We will revisit advanced acceleration techniques (e.g., ODE solvers, Distillation) in Chapters 9 and 10.

# Closing Remarks: The Score-Based Journey

- **From EBMs to Scores:**
  - We identified the intractable partition function of EBMs as the core challenge.
  - The Score Function $\nabla_x \log p(x)$ circumvented this by modeling gradients instead of densities.
- **Tractability via Noise (DSM):**
  - **Denoising Score Matching (DSM)** turned the intractable score matching objective into a simple regression problem by conditioning on data.
  - **Tweedie's Formula** established a profound link: estimating the score is mathematically equivalent to learning an **optimal denoiser**.
- **NCSN:**
  - We extended this to a continuum of noise scales, enabling robust generation via **Annealed Langevin Dynamics**.

# Convergence and Limitations

- **A Unified View Emerges:**
  - Despite distinct origins (Variational Inference vs. Score Matching), **DDPM** and **NCSN** share a strikingly similar structure.
  - Both rely on sequential denoising/Langevin updates.
- **The Shared Bottleneck:**
  - Both suffer from slow, sequential sampling due to the dense discretization of time (steps).
  - This limitation suggests that discrete-time models are just approximations of a more general underlying process.

# Looking Ahead: The Continuous-Time Perspective

In the next chapter, we will take the crucial step toward unification:

**Lecture 04: The Score SDE Framework**

1. **Continuous Unification:** We will show that DDPMs and NCSNs are different discretizations of a single Stochastic Differential Equation (SDE).
2. **Generative ODEs:** We will recast generation as solving a differential equation, unlocking advanced numerical solvers.
3. **Acceleration:** This framework will provide the theoretical tools needed to tackle the sampling speed problem.

# Practical Resources

## Tutorial 8: Deep Energy-Based Models

- **Documentation:**
  https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial_notebooks/tutorial8/Deep_Energy_Models.html
- **GitHub Repository:**
  https://github.com/phlippe/uvadlc_notebooks
- **Google Colab:**
  ▸ Open in Colab https://colab.research.google.com/drive/...

1. **EBM Architecture:** Implementing an Energy Model using simple CNNs on MNIST.
2. **Sampling:** Practical implementation of Langevin Dynamics (SGLD).
3. **Training Tricks:** Using **Replay Buffers** to stabilize Contrastive Divergence training.
4. **Applications:** Image generation, denoising, and Out-of-Distribution (OOD) detection.

# References I

David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169, 1985.

Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011.

Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4): 695–709, 2005.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

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

# References II

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

Charles M Stein. Estimation of the mean of a multivariate normal distribution. *The Annals of Statistics*, pages 1135–1151, 1981.

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