

Diffusion Models: Theory and Applications

Lecture 7: Score-Based Generative Models - Learning the Geometry of Data

Shubham Chatterjee

Missouri University of Science and Technology, Department of Computer Science

June 12, 2025

We learned to control what diffusion models generate...

But WHY do these models work so well? ?

- ✓ **Class-conditional:** Simple embedding-based conditioning
- ✓ **Classifier guidance:** External steering with gradients
- ✓ **Classifier-free guidance:** The modern breakthrough
- 🔍 **Today:** The geometric foundation underlying it all!

The Lost Hiker Analogy: Understanding Score Functions



Image source: <https://stock.adobe.com/in/images/mountain-travel-lost-man-with-the-map-in-the-forest/146343754>

Imagine you're lost in a mountainous landscape at night...▲

- You're trying to find the highest peak where a bonfire awaits
- You can't see the entire terrain (the full probability distribution)
- But you have a magical compass that points toward the steepest uphill direction
- By following this compass step by step, you eventually reach the summit

The Lost Hiker Analogy: Understanding Score Functions



Image source: <https://stock.adobe.com/in/images/mountain-travel-lost-man-with-the-map-in-the-forest/146343754>

This is exactly the intuition behind score-based models! ⚡

- **The mountain:** Probability density $p(\mathbf{x})$
- **The peaks:** High-probability regions (data modes)
- **The magical compass:** Score function $\nabla_{\mathbf{x}} \log p(\mathbf{x})$
- **Following the compass:** Langevin dynamics sampling

Score functions provide a navigation system through probability space!

The Lost Hiker Analogy: Understanding Score Functions

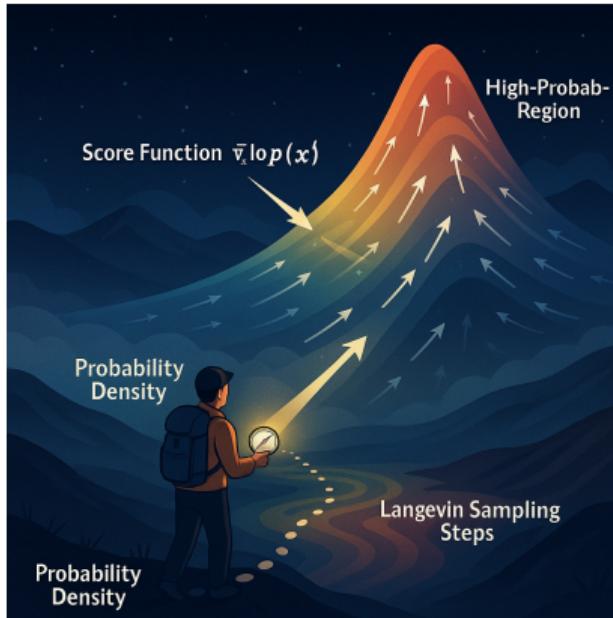


Figure: Score functions provide a navigation system through probability space!

Image generated using GPT-4o



What makes diffusion models fundamentally work?

The Deep Questions:

- Why does noise prediction work?
- What is the model really learning?
- How does this connect to probability theory?
- What's the geometric intuition?

Score-Based Perspective:

- **Score functions:** Gradients of log-probability
- **Geometric view:** Learning the shape of data
- **Equivalence:** Deep connection to diffusion
- **Unified theory:** One framework, multiple views

From algorithms to understanding - the geometry of generation!

A Different Perspective on Generation

The fundamental shift: instead of learning densities, learn directions!

Traditional approaches face the normalization problem

Most probability models have the form: $p(\mathbf{x}) = \frac{1}{Z} \tilde{p}(\mathbf{x})$

Problem: Computing $Z = \int \tilde{p}(\mathbf{x}) d\mathbf{x}$ is intractable!

Score-based insight: The partition function vanishes!

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) = \nabla_{\mathbf{x}} \log \left(\frac{1}{Z} \tilde{p}(\mathbf{x}) \right) \quad (1)$$

$$= \nabla_{\mathbf{x}} \log \tilde{p}(\mathbf{x}) - \nabla_{\mathbf{x}} \log Z \quad (2)$$

$$= \nabla_{\mathbf{x}} \log \tilde{p}(\mathbf{x}) \quad (\text{since } Z \text{ doesn't depend on } \mathbf{x}) \quad (3)$$

This seemingly minor shift has profound implications!

The Score Function: Your Probability Compass



Definition: Score Function 🔑

For a probability density function $p(\mathbf{x})$, the **score function** is:

$$\mathbf{s}(\mathbf{x}) = \nabla_{\mathbf{x}} \log p(\mathbf{x}) \quad (4)$$

Breaking it down 🚀

- $\log p(\mathbf{x})$: Taking logarithm transforms products to sums (more tractable)
- $\nabla_{\mathbf{x}}$: Computes direction and magnitude of steepest increase
- $\mathbf{s}(\mathbf{x})$: Vector field pointing toward higher likelihood regions

Simple example: 1D Gaussian 📈

For $p(x) = \mathcal{N}(x; \mu, \sigma^2)$:

$$s(x) = \frac{d}{dx} \log p(x) = -\frac{x - \mu}{\sigma^2} \quad (5)$$

Intuition: Score points toward mean μ with strength inversely proportional to variance!



Think of probability density as a mountainous landscape...

The Landscape ▲

- **Peaks:** High-probability regions (modes)
- **Valleys:** Low-probability regions
- **Slopes:** Rate of probability change
- **Height:** Probability density $p(x)$

Score Vector Properties ↑

- **Direction:** Points “uphill” toward higher probability
- **Magnitude:** Large where probability changes rapidly (steep slopes)
- **Zero:** Exactly at local maxima (peaks)
- **Roadmap:** Complete navigation system to high-density regions

Key insight ⚡

Score functions provide arrows scattered throughout the landscape, each pointing toward the nearest “realistic” data!

Energy-Based Perspective: Nature's Way ⚡

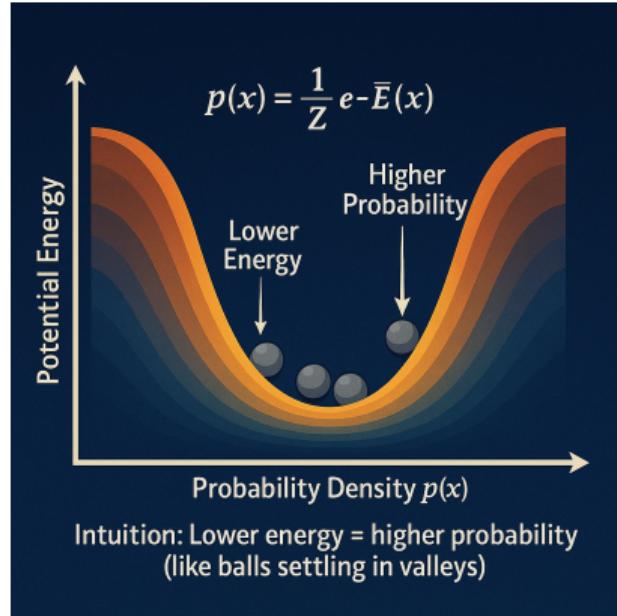


Figure: Another lens: think about data in terms of energy. Any probability distribution can be written as: $p(\mathbf{x}) = \frac{1}{Z} e^{-E(\mathbf{x})}$ *Image generated using GPT-4o*

Energy-Based Perspective: Nature's Way ⚡

Score = Negative energy gradient 

$$\mathbf{s}(\mathbf{x}) = \nabla_{\mathbf{x}} \log p(\mathbf{x}) = -\nabla_{\mathbf{x}} E(\mathbf{x}) \quad (6)$$

Physical interpretation: Score function is the “force” pushing toward lower energy!

Beautiful insight ★

- Natural, realistic data has **low energy**
- Unnatural, unrealistic data has **high energy**
- Score function points toward more natural configurations
- We learn the force field without computing the intractable Z !



Let's build intuition with distributions we understand...

Example 1: Multivariate Gaussian

For $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$:

$$\mathbf{s}(\mathbf{x}) = -\boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}) \quad (7)$$

Beautiful properties:

- Always points toward mean $\boldsymbol{\mu}$
- Strength depends on $\boldsymbol{\Sigma}^{-1}$ (inverse covariance)
- Far points experience strong “pull”, near points gentle guidance



Let's build intuition with distributions we understand...

Example 2: Mixture of Gaussians

For mixture $p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$:

$$\mathbf{s}(\mathbf{x}) = \sum_{k=1}^K \gamma_k(\mathbf{x}) \cdot (-\boldsymbol{\Sigma}_k^{-1}(\mathbf{x} - \boldsymbol{\mu}_k)) \quad (8)$$

where $\gamma_k(\mathbf{x})$ is the posterior probability of component k .

Insight: Weighted combination automatically balances competing influences from multiple modes!



The remarkable property that revolutionized generative modeling...

The profound insight A simple lightbulb icon.

If we can estimate $\nabla_x \log p(x)$ accurately, we can sample from $p(x)$ without ever computing $p(x)$ itself!

This bypasses the intractable partition function that has plagued probabilistic modeling for decades!

But how do we turn this gradient into actual samples?



The solution: Langevin Dynamics

An elegant physics-inspired process that lets us **walk through probability space**.

- **Systematic guidance:** Follow the score field $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ to move uphill toward high-density regions
- **Random exploration:** Inject noise at each step to avoid getting trapped in local modes

This combination enables efficient sampling from complex, high-dimensional distributions!

Back to our hiker analogy

Imagine the hiker now has two tools:

- **Magical compass:** Points toward nearest peak (score function)
- **Random steps:** Occasionally moves in unexpected directions

This combination ensures the hiker:

- ✓ Doesn't get stuck in local valleys
- ✓ Eventually explores all peaks in the mountain range
- ✓ Spends time proportional to peak height (probability)

Key Result:

Systematic + Stochastic



Perfect Sampling

 *No normalization needed!*

The Magic: Generation via Langevin Dynamics

Langevin Dynamics Algorithm

Input: Score function $s(x)$, step size η , steps T

Initialize x_0 randomly (e.g., from $\mathcal{N}(\mathbf{0}, \mathbf{I})$)

for $t = 0, 1, \dots, T - 1$ **do**

 Sample $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

$x_{t+1} = x_t + \eta s(x_t) + \sqrt{2\eta}\epsilon_t$

end for

Return: x_T

Understanding the update rule

- $\eta s(x_t)$: **Drift term** - systematic move toward higher probability
- $\sqrt{2\eta}\epsilon_t$: **Diffusion term** - controlled noise for exploration
- Balance prevents getting stuck in local modes while making progress

The Learning Challenge: How to Learn the Compass?

The fundamental chicken-and-egg problem ?

- We want to learn $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ to sample from $p(\mathbf{x})$
- But to compute $\nabla_{\mathbf{x}} \log p(\mathbf{x})$, we need to know $p(\mathbf{x})$
- We don't know $p(\mathbf{x})$ —learning it is the whole point!

Naive approach (doesn't work!)

Train neural network $\mathbf{s}_{\theta}(\mathbf{x})$ to minimize:

$$\mathcal{L}_{\text{naive}} = \mathbb{E}_{p(\mathbf{x})} [\|\mathbf{s}_{\theta}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2] \quad (9)$$

Fatal flaw: We don't know $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ —that's what we're trying to learn!

Solution: Score matching - a mathematical miracle! 



Question: Can we transform this intractable loss into something we *can* compute?

Answer: Yes! Through integration by parts, we can eliminate the unknown true score entirely.

Let's see this mathematical magic in action...

Score Matching: The Mathematical Transformation



Step-by-step derivation of the miracle...

- Starting with the intractable naive loss:

$$\mathcal{L}_{\text{naive}} = \mathbb{E}_{p(\mathbf{x})} [\|\mathbf{s}_\theta(\mathbf{x}) - \nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2] \quad (10)$$

- Expanding the squared norm:

$$\mathcal{L}_{\text{naive}} = \mathbb{E}_{p(\mathbf{x})} [\|\mathbf{s}_\theta(\mathbf{x})\|^2 - 2\mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x}) + \|\nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2] \quad (11)$$

Key observation

- First term:** $\mathbb{E}[\|\mathbf{s}_\theta(\mathbf{x})\|^2]$ - depends only on our learned function
- Second term:** $\mathbb{E}[2\mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x})]$ - involves unknown true score
- Third term:** $\mathbb{E}[\|\nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2]$ - constant w.r.t. θ (can ignore for optimization)

The magic happens in transforming the problematic second term...

The Challenge: One Problematic Term

After expanding, we identified our roadblock...

What we learned from expansion 

$$\mathcal{L}_{\text{naive}} = \mathbb{E}_{p(\mathbf{x})} \left[\|\mathbf{s}_\theta(\mathbf{x})\|^2 \right] \quad (12)$$

$$-2\mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x}) \quad (13)$$

$$+ \|\nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2 \quad (14)$$

The bottleneck 

Second term: $\mathbb{E}[\mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x})]$ contains the unknown true score $\nabla_{\mathbf{x}} \log p(\mathbf{x})$

Question: Can we somehow transform this term to eliminate the unknown?

We need a mathematical trick to “move” the unknown... 

The Strategy: Move Derivatives Around

The core idea 

Current situation: We have derivatives of the unknown $p(x)$

Goal: Transform to get derivatives of the known $s_\theta(x)$

Mathematical tool: Integration by parts — swaps which function gets differentiated!

Integration by parts intuition

General principle: $\int u \frac{dv}{dx} dx = uv - \int v \frac{du}{dx} dx$

What this does:

- **Original integral:** u multiplied by derivative of v
- **Transformed integral:** v multiplied by derivative of u
- **Key insight:** Swaps which function gets differentiated!

Plan: Make the unknown $p(x)$ “plain” and differentiate the known $s_\theta(x)$! 

Why This Strategy Will Work ✓

The beautiful consequence of our plan...

What integration by parts will give us ⏮

Before transformation:

$$\int \mathbf{s}_\theta(\mathbf{x}) \cdot \frac{\partial}{\partial x_i} [\text{unknown}] d\mathbf{x} \quad (15)$$

After transformation:

$$\int \frac{\partial}{\partial x_i} [\mathbf{s}_\theta(\mathbf{x})] \cdot \text{unknown} d\mathbf{x} \quad (16)$$

Why this is revolutionary ✎

- **New derivative term:** $\frac{\partial \mathbf{s}_\theta(\mathbf{x})}{\partial x_i}$ — we can compute this!
- **Unknown part:** Becomes just $p(\mathbf{x})$ (no derivatives)
- **Final step:** The $p(\mathbf{x})$ becomes our expectation — computable from data!

Step 1: From Expectation to Integral →

Converting our problematic term to integral form...

Our target: the problematic term

$$\mathbb{E}_{p(\mathbf{x})} [\mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x})] \quad (17)$$

Using the definition of expectation 

By definition, $\mathbb{E}_{p(\mathbf{x})}[f(\mathbf{x})] = \int f(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}$

Applying this to our case:

$$\mathbb{E}_{p(\mathbf{x})} [\mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x})] = \int \mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x} \quad (18)$$

Now we have an integral we can work with! 

Step 2: Understanding the Chain Rule for Logarithms



The key mathematical identity we need...

Recall: Chain rule for logarithmic derivatives

For any function $f(\mathbf{x})$, the chain rule gives us:

$$\frac{d}{d\mathbf{x}} \log f(\mathbf{x}) = \frac{1}{f(\mathbf{x})} \cdot \frac{df(\mathbf{x})}{d\mathbf{x}} \quad (19)$$

Applying to our probability density

For $f(\mathbf{x}) = p(\mathbf{x})$:

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) = \frac{1}{p(\mathbf{x})} \cdot \nabla_{\mathbf{x}} p(\mathbf{x}) = \frac{\nabla_{\mathbf{x}} p(\mathbf{x})}{p(\mathbf{x})} \quad (20)$$

This is just basic calculus - but it's the key to our transformation!



Step 3: The Crucial Algebraic Manipulation =

Starting from the chain rule result →

We just showed that:

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) = \frac{\nabla_{\mathbf{x}} p(\mathbf{x})}{p(\mathbf{x})} \quad (21)$$

Multiply both sides by $p(\mathbf{x})$ 

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) \cdot p(\mathbf{x}) = \frac{\nabla_{\mathbf{x}} p(\mathbf{x})}{p(\mathbf{x})} \cdot p(\mathbf{x}) = \nabla_{\mathbf{x}} p(\mathbf{x}) \quad (22)$$

The beautiful result 

$$\boxed{\nabla_{\mathbf{x}} \log p(\mathbf{x}) \cdot p(\mathbf{x}) = \nabla_{\mathbf{x}} p(\mathbf{x})} \quad (23)$$

We've converted log-derivatives to plain derivatives!

Step 4: Applying the Transformation

Substituting our identity into the integral...

Our integral before transformation ←

$$\int \mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x} \quad (24)$$

Using our identity: $\nabla_{\mathbf{x}} \log p(\mathbf{x}) \cdot p(\mathbf{x}) = \nabla_{\mathbf{x}} p(\mathbf{x})$ ↓

$$\int \mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x} = \int \mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} p(\mathbf{x}) d\mathbf{x} \quad (25)$$

Perfect! Now we have derivatives of $p(\mathbf{x})$, not $\log p(\mathbf{x})!$ ✓
Ready for integration by parts! ↗

Setting Up Integration by Parts 🔧

Breaking it down component-wise 📈

Since $\mathbf{s}_\theta(\mathbf{x})$ is a vector, we can write:

$$\mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} p(\mathbf{x}) = \sum_{i=1}^d s_{\theta,i}(\mathbf{x}) \frac{\partial p(\mathbf{x})}{\partial x_i} \quad (26)$$

So our integral becomes:

$$\int \sum_{i=1}^d s_{\theta,i}(\mathbf{x}) \frac{\partial p(\mathbf{x})}{\partial x_i} d\mathbf{x} = \sum_{i=1}^d \int s_{\theta,i}(\mathbf{x}) \frac{\partial p(\mathbf{x})}{\partial x_i} d\mathbf{x} \quad (27)$$

Now we can apply integration by parts to each component separately! ↓

Integration by Parts Formula Reminder



The basic integration by parts formula

For functions $u(x)$ and $v(x)$ on interval $[a, b]$:

$$\int_a^b u(x) \frac{dv}{dx} dx = u(x)v(x) \Big|_a^b - \int_a^b v(x) \frac{du}{dx} dx \quad (28)$$

In multivariable form (Divergence Theorem)

For integration over all space \mathbb{R}^d :

$$\int_{\mathbb{R}^d} u(\mathbf{x}) \frac{\partial v(\mathbf{x})}{\partial x_i} d\mathbf{x} = \underbrace{\lim_{R \rightarrow \infty} \int_{\partial B_R} u(\mathbf{x})v(\mathbf{x})n_i dS}_{\text{surface integral}} - \int_{\mathbb{R}^d} v(\mathbf{x}) \frac{\partial u(\mathbf{x})}{\partial x_i} d\mathbf{x} \quad (29)$$

Where B_R is a ball of radius R and n_i is the i -th component of the outward normal.

The key insight

Effect: Transfers the derivative from v to u

- **Before:** $u \times (\text{derivative of } v)$
- **After:** $v \times (\text{derivative of } u)$ (plus surface integral)
- **Surface integral:** Vanishes under regularity conditions

Choosing u and v for Our Problem

Our integral for component i

$$\int s_{\theta,i}(\mathbf{x}) \frac{\partial p(\mathbf{x})}{\partial x_i} d\mathbf{x} \quad (30)$$

Strategic choice 

Let:

- $u = s_{\theta,i}(\mathbf{x})$ (the function we know and can differentiate)
- $v = p(\mathbf{x})$ (the unknown density)

Why this choice?

- We want to get rid of $\frac{\partial p(\mathbf{x})}{\partial x_i}$ (derivative of unknown)
- We're okay with getting $\frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i}$ (derivative of known)

Applying Integration by Parts

Using the formula with our choices →

Formula: $\int_{\mathbb{R}^d} u \frac{\partial v}{\partial x_i} d\mathbf{x} = \lim_{R \rightarrow \infty} \int_{\partial B_R} uv n_i dS - \int_{\mathbb{R}^d} v \frac{\partial u}{\partial x_i} d\mathbf{x}$

Our substitution: $u = s_{\theta,i}(\mathbf{x})$, $v = p(\mathbf{x})$

The transformation

$$\int_{\mathbb{R}^d} s_{\theta,i}(\mathbf{x}) \frac{\partial p(\mathbf{x})}{\partial x_i} d\mathbf{x} = \lim_{R \rightarrow \infty} \int_{\partial B_R} s_{\theta,i}(\mathbf{x}) p(\mathbf{x}) n_i dS \quad (31)$$

$$- \int_{\mathbb{R}^d} p(\mathbf{x}) \frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i} d\mathbf{x} \quad (32)$$

What we achieved ★

- **Eliminated:** $\frac{\partial p(\mathbf{x})}{\partial x_i}$ (derivative of unknown)
- **Introduced:** $\frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i}$ (derivative of known)
- **Plus:** A surface integral we need to handle

Why the Surface Integral Vanishes 🕵️

The surface integral in question ?

$$\lim_{R \rightarrow \infty} \int_{\partial B_R} s_{\theta,i}(\mathbf{x}) p(\mathbf{x}) n_i dS \quad (33)$$

This integrates $s_{\theta,i}(\mathbf{x}) p(\mathbf{x})$ over the surface of a sphere of radius R as $R \rightarrow \infty$.

Why it vanishes: Decay rates ✅

- Probability densities: $p(\mathbf{x}) \rightarrow 0$ exponentially fast as $\|\mathbf{x}\| \rightarrow \infty$
- Score networks: $s_{\theta,i}(\mathbf{x})$ grows at most polynomially with $\|\mathbf{x}\|$
- Surface area: Grows like R^{d-1} (polynomial in R)
- Mathematical result: Exponential decay dominates polynomial growth

Intuitive explanation 🎠

- Far from data (large $\|\mathbf{x}\|$): probability $p(\mathbf{x}) \approx 0$
- Even if score is large, $s_{\theta,i}(\mathbf{x}) \times \text{tiny } p(\mathbf{x}) \approx 0$
- The surface integral goes to zero faster than the surface area grows
- Result:** Surface integral vanishes in the limit $R \rightarrow \infty$

The Final Result: Component i

For component i , we started with 

$$\int s_{\theta,i}(\mathbf{x}) \frac{\partial p(\mathbf{x})}{\partial x_i} d\mathbf{x} \quad (34)$$

After integration by parts 

$$\int s_{\theta,i}(\mathbf{x}) \frac{\partial p(\mathbf{x})}{\partial x_i} d\mathbf{x} = - \int p(\mathbf{x}) \frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i} d\mathbf{x} \quad (35)$$

(boundary term vanished)

Converting back to expectation 

$$- \int p(\mathbf{x}) \frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i} d\mathbf{x} = - \mathbb{E}_{p(\mathbf{x})} \left[\frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i} \right] \quad (36)$$

Beautiful! No unknown functions remain - only derivatives of our learned score! 

Summing Over All Components +

Recall: We had a sum over all components $\stackrel{1}{\equiv}$

$$\int \mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} p(\mathbf{x}) d\mathbf{x} = \sum_{i=1}^d \int s_{\theta,i}(\mathbf{x}) \frac{\partial p(\mathbf{x})}{\partial x_i} d\mathbf{x} \quad (37)$$

Each component gave us $\stackrel{2}{\equiv}$

$$\int s_{\theta,i}(\mathbf{x}) \frac{\partial p(\mathbf{x})}{\partial x_i} d\mathbf{x} = -\mathbb{E}_{p(\mathbf{x})} \left[\frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i} \right] \quad (38)$$

Summing Over All Components +

Summing over all $i = 1, 2, \dots, d$ 

$$\sum_{i=1}^d \int s_{\theta,i}(\mathbf{x}) \frac{\partial p(\mathbf{x})}{\partial x_i} d\mathbf{x} = - \sum_{i=1}^d \mathbb{E}_{p(\mathbf{x})} \left[\frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i} \right] \quad (39)$$

$$= -\mathbb{E}_{p(\mathbf{x})} \left[\sum_{i=1}^d \frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i} \right] \quad (40)$$

$$= -\mathbb{E}_{p(\mathbf{x})} [\text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_{\theta}(\mathbf{x}))] \quad (41)$$

But what is this sum of partial derivatives? ?

What is $\sum_{i=1}^d \frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i}$? ?

Understanding the mathematical object we've created...

The Jacobian matrix 

For vector function $\mathbf{s}_\theta(\mathbf{x}) = [s_{\theta,1}(\mathbf{x}), s_{\theta,2}(\mathbf{x}), \dots, s_{\theta,d}(\mathbf{x})]^T$:

$$\text{Jacobian } J = \nabla_{\mathbf{x}} \mathbf{s}_\theta(\mathbf{x}) = \begin{bmatrix} \frac{\partial s_{\theta,1}}{\partial x_1} & \frac{\partial s_{\theta,1}}{\partial x_2} & \cdots & \frac{\partial s_{\theta,1}}{\partial x_d} \\ \frac{\partial s_{\theta,2}}{\partial x_1} & \frac{\partial s_{\theta,2}}{\partial x_2} & \cdots & \frac{\partial s_{\theta,2}}{\partial x_d} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial s_{\theta,d}}{\partial x_1} & \frac{\partial s_{\theta,d}}{\partial x_2} & \cdots & \frac{\partial s_{\theta,d}}{\partial x_d} \end{bmatrix} \quad (42)$$

What is $\sum_{i=1}^d \frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i}$? ?

The trace of the Jacobian

$$\text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_{\theta}(\mathbf{x})) = \sum_{i=1}^d \frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i} \quad (43)$$

Trace = Sum of diagonal elements = each output component's derivative w.r.t. its corresponding input



The geometric meaning behind our mathematical result...

The divergence of a vector field

For any vector field $\mathbf{v}(\mathbf{x}) = [v_1(\mathbf{x}), v_2(\mathbf{x}), \dots, v_d(\mathbf{x})]^T$:

$$\text{div}(\mathbf{v}) = \nabla \cdot \mathbf{v} = \sum_{i=1}^d \frac{\partial v_i(\mathbf{x})}{\partial x_i} \quad (44)$$

Geometric meaning: How much the vector field “spreads out” or “contracts” at each point

Our Score Function Is A Vector Field!



- $s_\theta(x)$ assigns a vector (direction + magnitude) to each point x
- The trace $\text{tr}(\nabla_x s_\theta(x))$ is the divergence of this vector field
- It measures how much our learned score field “expands” or “contracts” locally

Why this matters for score matching

The divergence term ensures our learned score field has the right “flow properties” - it can’t be too wild or unrealistic compared to true probability score fields.

The Final Integration by Parts Result

What we computed 

$$\mathbb{E}_{p(x)} \left[\mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x}) \right] = -\mathbb{E}_{p(x)} [\text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_\theta(\mathbf{x}))] \quad (45)$$

In terms we now understand 

Left side: Problematic term with unknown true score

Right side: Negative expectation of the divergence of our learned score field

Beautiful result: We transformed unknown derivatives into known derivatives!

Why this is exactly what we wanted 

- **Computable:** Only involves our learned function $\mathbf{s}_\theta(\mathbf{x})$
- **Geometric:** Has clear interpretation as vector field divergence
- **Regularizing:** Prevents unrealistic score field behavior
- **Equivalent:** Mathematically identical to the original impossible loss

Putting It All Together: The Complete Transformation

Substituting our integration by parts result back into the original loss...

Recall

- We wanted to transform the problematic term: $\mathbb{E}_{p(\mathbf{x})} [\mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x})]$
- What integration by parts gave us:
$$\mathbb{E}_{p(\mathbf{x})} [\mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x})] = -\mathbb{E}_{p(\mathbf{x})} [\text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_\theta(\mathbf{x}))]$$

Substituting back into our expanded naive loss 

$$\mathcal{L}_{\text{naive}} = \mathbb{E}_{p(\mathbf{x})} [\|\mathbf{s}_\theta(\mathbf{x})\|^2 - 2\mathbf{s}_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x}) + \|\nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2] \quad (46)$$

$$= \mathbb{E}_{p(\mathbf{x})} [\|\mathbf{s}_\theta(\mathbf{x})\|^2 + 2\text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_\theta(\mathbf{x})) + \|\nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2] \quad (47)$$

We're almost there - one more step! 

The Score Matching Miracle: Final Result ✨

Ignoring constants for optimization →

The term $\mathbb{E}_{p(x)} [\|\nabla_x \log p(x)\|^2]$ doesn't depend on θ , so we can ignore it.

The Score Matching objective ⚡

$$\mathcal{L}_{\text{SM}} = \mathbb{E}_{p(x)} \left[\text{tr}(\nabla_x \mathbf{s}_\theta(x)) + \frac{1}{2} \|\mathbf{s}_\theta(x)\|^2 \right] \quad (48)$$

What we achieved ★

- **Eliminated:** All dependence on unknown true score $\nabla_x \log p(x)$
- **Computable:** Everything depends only on our learned function $\mathbf{s}_\theta(x)$
- **Tractable:** Can estimate this loss from data samples
- **Equivalent:** Minimizing this = minimizing the original impossible loss

Mathematical miracle complete! ✓

Score Matching: Beautiful but Expensive !

The elegant solution has a computational catch...

The computational bottleneck 

Computing $\text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_{\theta}(\mathbf{x}))$ requires:

$$\text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_{\theta}(\mathbf{x})) = \sum_{i=1}^d \frac{\partial s_{\theta,i}(\mathbf{x})}{\partial x_i} \quad (49)$$

That's d second-order derivatives per training sample!

The scaling nightmare 

- **1D data:** 1 derivative - Easy
- **CIFAR (32×32×3):** 3,072 derivatives - Expensive
- **ImageNet (224×224×3):** 150,528 derivatives - Intractable

Next Session Preview: Generative Models for IR, NLP & RAG

Now that we understand the geometric foundations, let's explore cutting-edge research opportunities:

- How can we revolutionize information retrieval with generative models? 
- What breakthrough opportunities exist in generative RAG architectures? 
- How do we apply diffusion models to personalized knowledge systems? 
- Which research directions offer the highest impact potential for your career? 