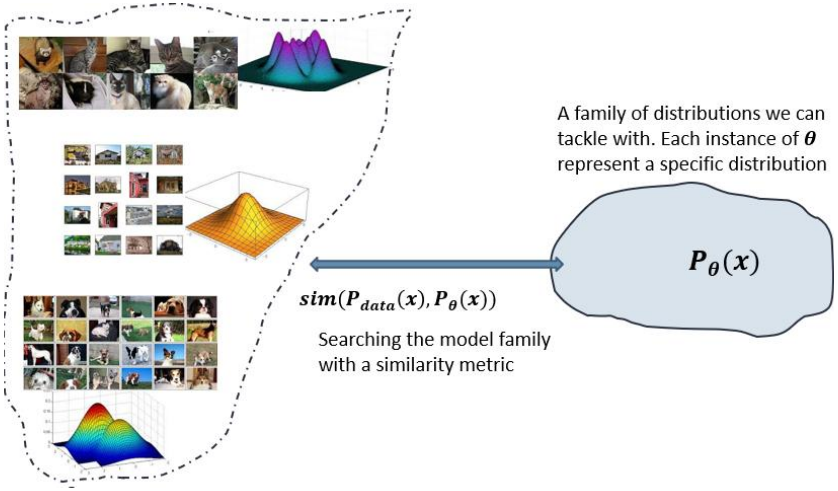
Score-based Models

22-808: Generative models
Sharif University of Technology
Fall 2025

Fatemeh Seyyedsalehi

Slides of this lecture are selected from Yong Song's and Stanford CS236

Recap



- We need a framework to interact with distributions for statistical generative models.
  - Probabilistic generative models
  - Deep generative models
    - Autoregressive models
    - Variational Autoencoders
    - Generative adversarial networks
    - Normalizing Flow
    - Energy-based models
    - **Score-based models**

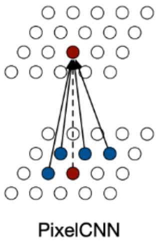
How to represent probability distributions?

- Probability density function (p.d.f.) or probability mass function (p.m.f.)

$$p(\mathbf{x})$$

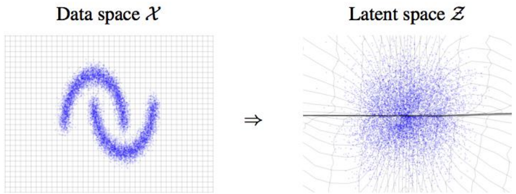
- Autoregressive models

$$p_\theta(\mathbf{x}) = \prod_{i=1}^{d} p_\theta(\mathbf{x}_i \mid \mathbf{x}_{<i})$$



- Flow models

$$p_\theta(\mathbf{x}) = p(\mathbf{z}) |\det(J_{f_\theta}(\mathbf{x}))|, \quad \mathbf{z} = f_\theta(\mathbf{x})$$



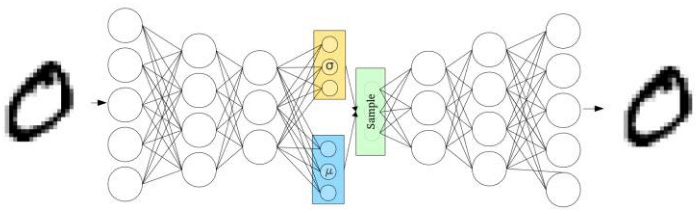
How to represent probability distributions?

- Probability density function (p.d.f.) or probability mass function (p.m.f.)

$$p(\mathbf{x})$$

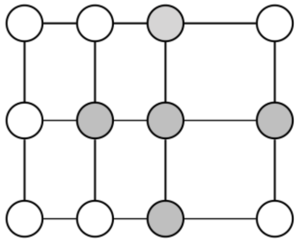
- Variational autoencoders

$$p_\theta(\mathbf{x}) = \int p(\mathbf{z}) p_\theta(\mathbf{x} \mid \mathbf{z}) \, \mathrm{d}\mathbf{z}$$



- Energy-based models

$$p_\theta(\mathbf{x}) = \frac{e^{f_\theta(\mathbf{x})}}{Z(\theta)}$$



How to represent probability distributions?

- Probability density function (p.d.f.) or probability mass function (p.m.f.)

$$p(\mathbf{x})$$

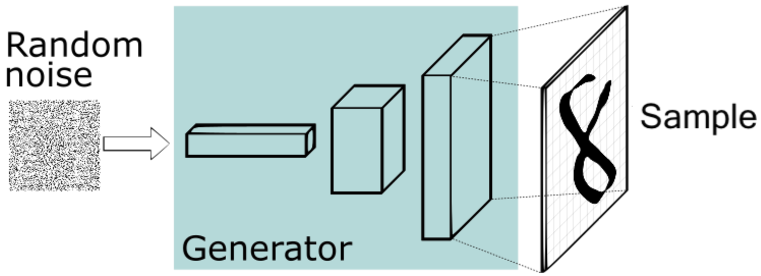
- Pros
  - Maximum likelihood training
  - Principled model comparison via likelihoods
- Cons
  - Special architectures or surrogate losses to deal with intractable partition functions

How to represent probability distributions?

- Sampling process

- Generative adversarial networks (GANs)

$$\mathbf{z} \sim p(\mathbf{z})$$
$$\mathbf{x} = g_\theta(\mathbf{z})$$



GAN

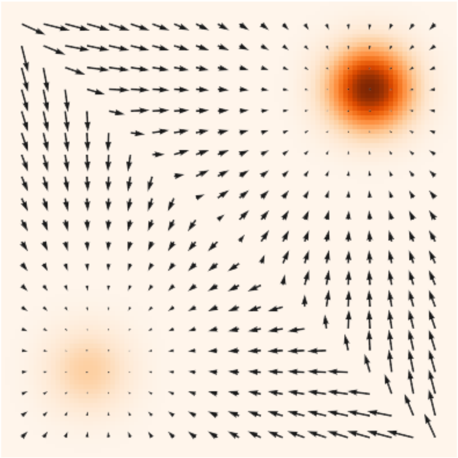
How to represent probability distributions?

- Sampling process
- Pros
  - Samples typically have better quality
- Cons
  - Require adversarial training. Training instability and mode collapse.
  - No principled way to compare different models
  - No principled termination criteria for training

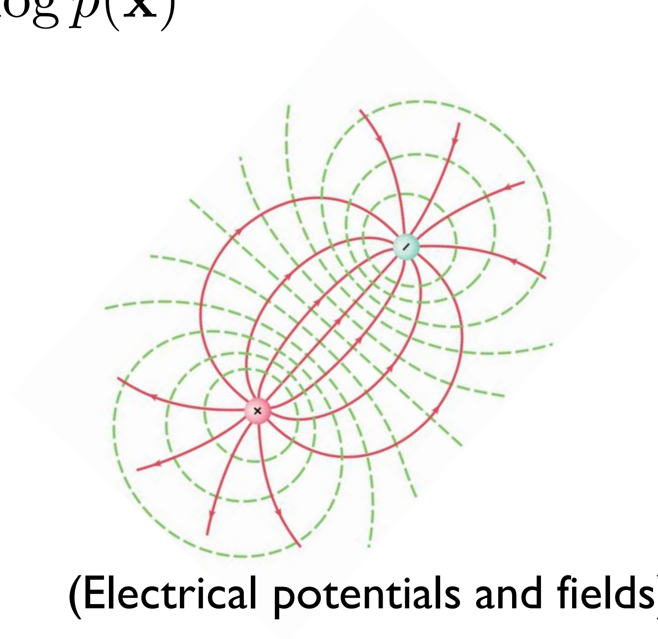
How to represent probability distributions?

- When the pdf is differentiable, we can compute the gradient of a probability density.

Score function $\nabla_{\mathbf{x}} \log p(\mathbf{x})$



(pdf and score)

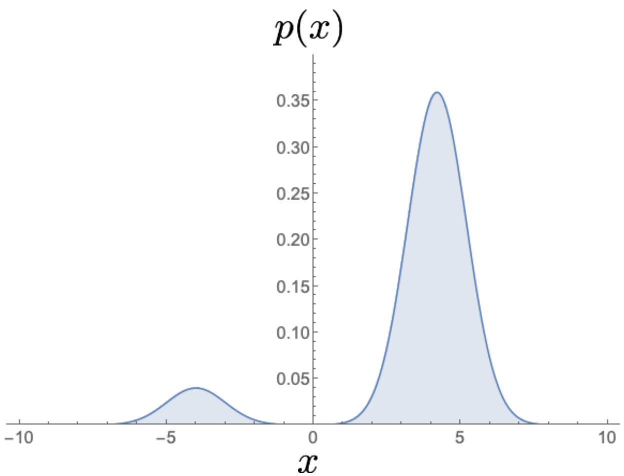


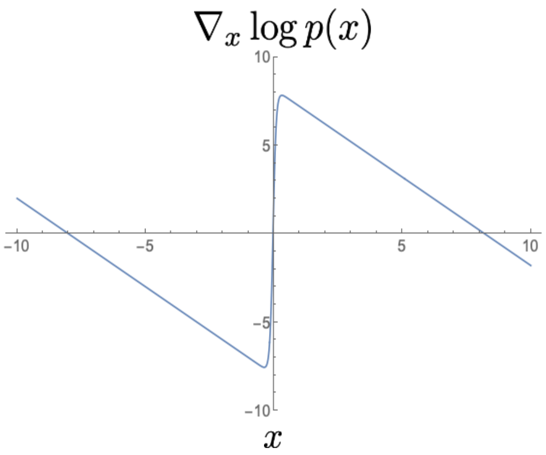
(Electrical potentials and fields)

How to represent probability distributions?

- When the pdf is differentiable, we can compute the gradient of a probability density.

Score function $\nabla_{\mathbf{x}} \log p(\mathbf{x})$



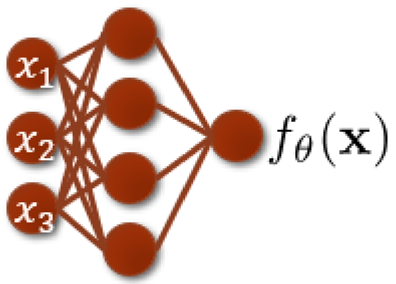


Recap on energy-based models

- Deep Energy-Based models (EBMs)

$$f_\theta(\mathbf{x}) \in \mathbb{R}$$

$$p_\theta(\mathbf{x}) = \frac{e^{f_\theta(\mathbf{x})}}{Z(\theta)}$$



- Maximum likelihood training: $\max_\theta f_\theta(\mathbf{x}_{\text{train}}) - \log Z(\theta)$
  - Contrastive divergence

$$\nabla_\theta f_\theta(\mathbf{x}_{\text{train}}) - \nabla_\theta \log Z(\theta) \approx \nabla_\theta f_\theta(\mathbf{x}_{\text{train}}) - \nabla_\theta f_\theta(\mathbf{x}_{\text{sample}})$$

  - Requires iterative sampling during training

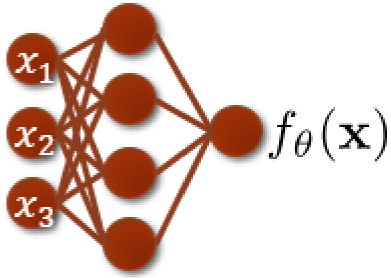
$$\mathbf{x}_{\text{sample}} \sim p_\theta(\mathbf{x})$$

Recap on energy-based models

- Deep Energy-Based models (EBMs)

$$f_\theta(\mathbf{x}) \in \mathbb{R}$$

$$p_\theta(\mathbf{x}) = \frac{e^{f_\theta(\mathbf{x})}}{Z(\theta)}$$



- Minimizing Fisher divergence:

$$\min_\theta \frac{1}{2} E_{\mathbf{x}\sim p_{\text{data}}}[\|\nabla_\mathbf{x} \log p_{\text{data}}(\mathbf{x}) - \nabla_\mathbf{x} \log p_\theta(\mathbf{x})\|_2^2]$$

  - Score matching

$$\frac{1}{2} E_{\mathbf{x}\sim p_{\text{data}}}[\|\nabla_\mathbf{x} \log p_{\text{data}}(\mathbf{x}) - \nabla_\mathbf{x} \log p_\theta(\mathbf{x})\|_2^2]$$

$$= \frac{1}{2} E_{\mathbf{x}\sim p_{\text{data}}}\left[\frac{1}{2}\|\nabla_\mathbf{x} \log p_\theta(\mathbf{x})\|_2^2 + \text{tr}(\nabla_\mathbf{x}^2 \log p_\theta(\mathbf{x}))\right] + \text{const.}$$

Score matching for training EBMs

- Score function of EBMs

$$\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} f_\theta(\mathbf{x}) - \underbrace{\nabla_{\mathbf{x}} \log Z(\theta)}_{=0} = \nabla_{\mathbf{x}} f_\theta(\mathbf{x})$$

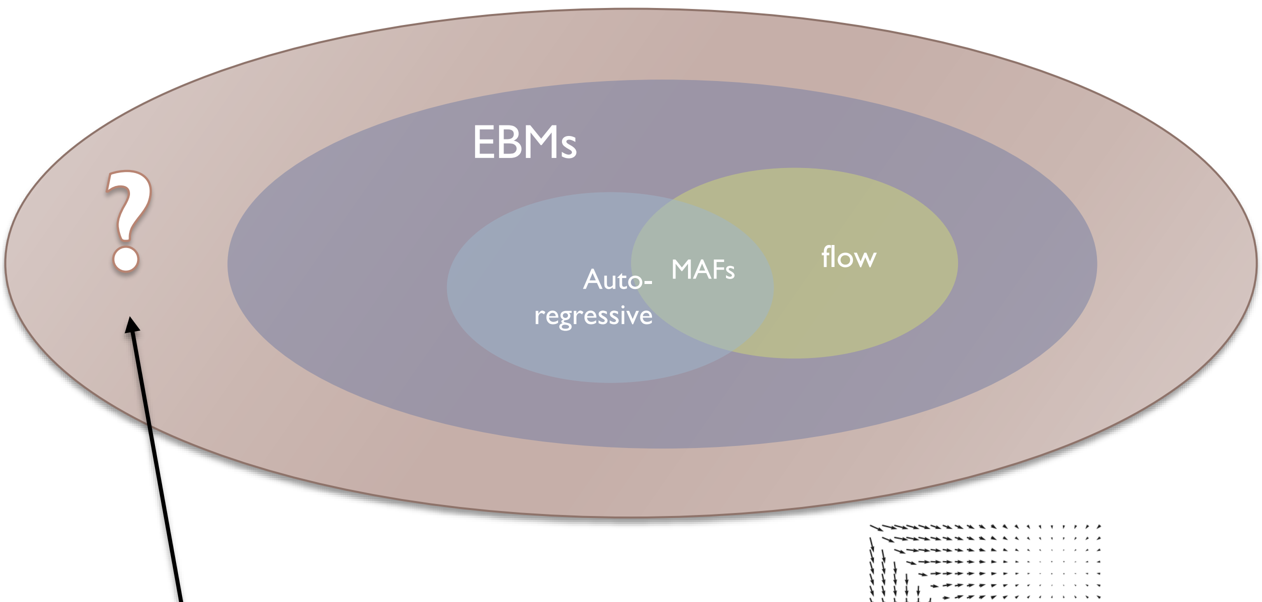
- Score matching for EBMs:

$$\begin{aligned} & E_{\mathbf{x}\sim p_{\text{data}}}\left[\frac{1}{2}\,\|\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})\|_2^2 + \operatorname{tr}(\nabla_{\mathbf{x}}^2 \log p_\theta(\mathbf{x}))\right] \\ =& E_{\mathbf{x}\sim p_{\text{data}}}\left[\frac{1}{2}\,\|\nabla_{\mathbf{x}} f_\theta(\mathbf{x})\|_2^2 + \operatorname{tr}(\nabla_{\mathbf{x}}^2 f_\theta(\mathbf{x}))\right] \end{aligned}$$

- Is score matching limited to EBMs?
  - Autoregressive models
  - Normalizing flow models

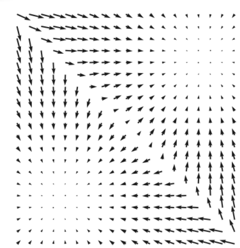
Score-based models

- What's the most general model that can be efficiently trained by score matching?



- Score-based model

$$\boldsymbol{s}_\theta(\mathbf{x})$$



$\mathbb{R}^d \rightarrow \mathbb{R}^d$

Directly model the vector field of gradients!

Score estimation by training score-based models

Probability density

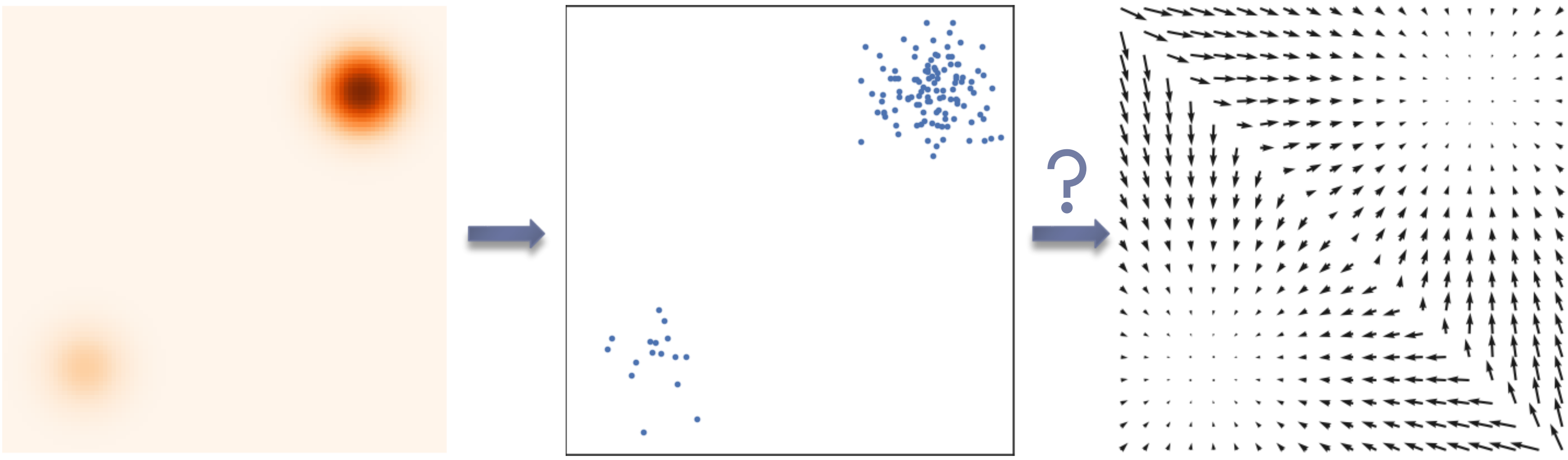
$p_{\text{data}}(\mathbf{x})$

i.i.d. samples

$\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$

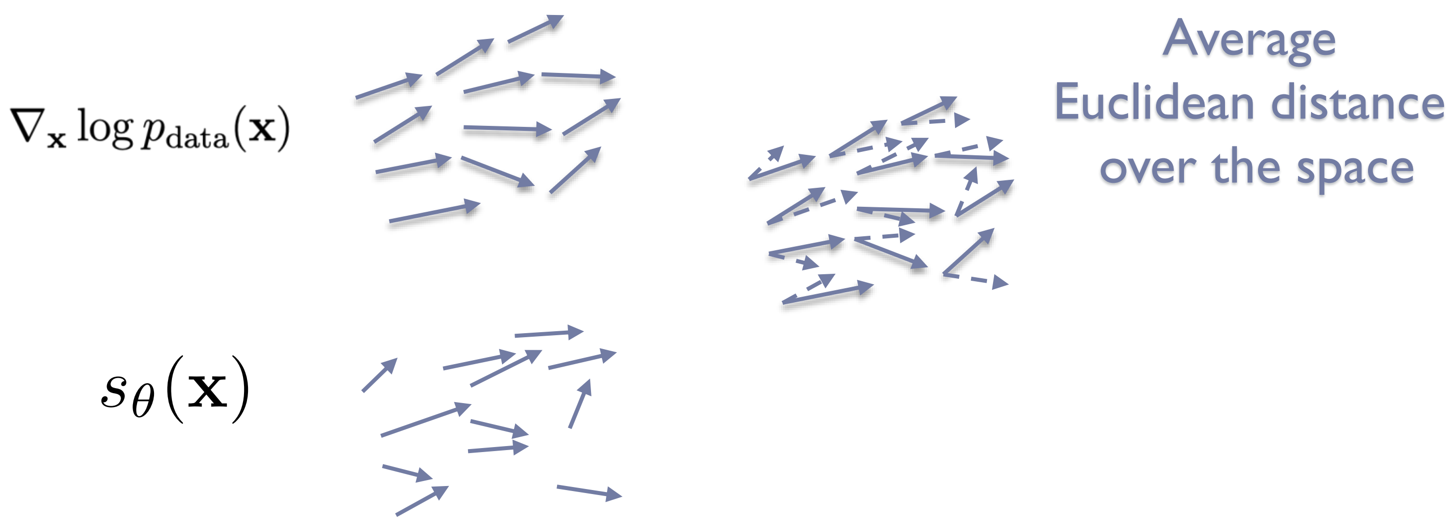
Score function

$\boldsymbol{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$



Score estimation by training score-based models

- **Given:** i.i.d. samples $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\} \sim p_{\text{data}}(\mathbf{x})$
- **Task:** Estimating the score $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$
- **Score Model:** A learnable vector-valued function
- **Goal:** $\boldsymbol{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$ $\boldsymbol{s}_\theta(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}^d$
- How to compare two vector fields of scores?



Score estimation by training score-based models

- **Objective:** Average Euclidean distance over the whole space.

$$\frac{1}{2} E_{\mathbf{x} \sim p_{\text{data}}}[\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \boldsymbol{s}_\theta(\mathbf{x})\|_2^2]$$

(Fisher divergence)

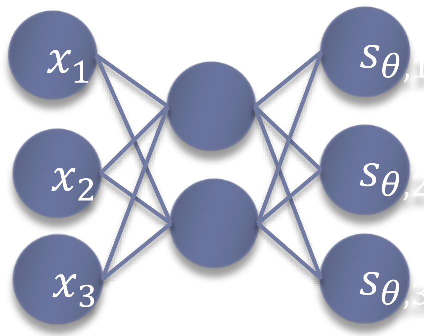
- **Score matching:**

$$E_{\mathbf{x} \sim p_{\text{data}}}\left[\frac{1}{2}\|\boldsymbol{s}_\theta(\mathbf{x})\|_2^2 + \operatorname{tr}(\underbrace{\nabla_{\mathbf{x}} \boldsymbol{s}_\theta(\mathbf{x})}_{\text{Jacobian of } \boldsymbol{s}_\theta(\mathbf{x})})\right]$$

- **Requirements:**
  - The score model must be efficient to evaluate.
  - Do we need the score model to be a proper score function (i.e., gradient of a scalar "energy" function)?

Score matching is not scalable

- Deep neural networks as more expressive score models



$\longrightarrow$ $s_\theta(\mathbf{x})$

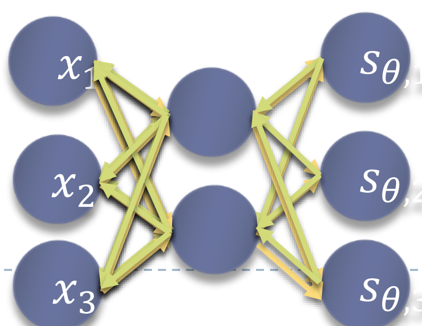
Score Matching
is not Scalable!

- Compute $\|s_\theta(\mathbf{x})\|_2^2$ and $\mathrm{tr}(\nabla_{\mathbf{x}} s_\theta(\mathbf{x}))$ ✗ 

$\frac{\partial s_{\theta,1}(\mathbf{x})}{\partial x_1}$

$\frac{\partial s_{\theta,2}(\mathbf{x})}{\partial x_2}$

$\frac{\partial s_{\theta,3}(\mathbf{x})}{\partial x_3}$



$s_{\theta,1}(\mathbf{x})$

$s_{\theta,2}(\mathbf{x})$ $s_\theta(\mathbf{x})$

$s_{\theta,3}(\mathbf{x})$

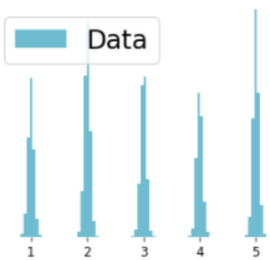
$O(d)$ Backprops!

$$\nabla_{\mathbf{x}} s_\theta(\mathbf{x}) = \begin{pmatrix} \frac{\partial s_{\theta,1}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\theta,1}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\theta,1}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\theta,2}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\theta,2}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\theta,2}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\theta,3}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\theta,3}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\theta,3}(\mathbf{x})}{\partial x_3} \end{pmatrix}$$

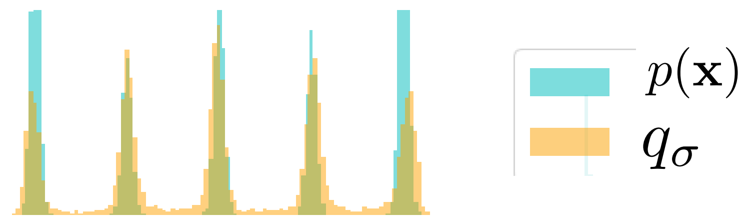
Denoising Score Matching (Vincent, 2011)

- Consider the perturbed distribution

$$q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) = \mathcal{N}(\mathbf{x}; \sigma^2 I) \qquad q_\sigma(\tilde{\mathbf{x}}) = \int p(\mathbf{x}) q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) \mathrm{d}\mathbf{x}$$



$p(\mathbf{x})$



- Score estimation for $\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})$ is easier $q_\sigma(\tilde{\mathbf{x}}) \approx p(\tilde{\mathbf{x}})$
- If the noise level is small, this is a good approximation

Denoising score matching

$p_{\text{data}}(\mathbf{x})$

$\mathbf{x}$

$q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})$

$q_\sigma(\tilde{\mathbf{x}})$

$\tilde{\mathbf{x}}$

- Denoising score matching (Vincent 2011): matching the score of a noise-perturbed distribution

$$\frac{1}{2} E_{\tilde{\mathbf{x}} \sim q_\sigma}[\|\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}) - \boldsymbol{s}_\theta(\tilde{\mathbf{x}})\|_2^2]$$

$$= \frac{1}{2} \int q_\sigma(\tilde{\mathbf{x}}) \, \|\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}) - \boldsymbol{s}_\theta(\tilde{\mathbf{x}})\|_2^2 \, \mathrm{d}\tilde{\mathbf{x}}$$

$$= \frac{1}{2} \int q_\sigma(\tilde{\mathbf{x}}) \, \|\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})\|_2^2 \, \mathrm{d}\tilde{\mathbf{x}} + \frac{1}{2} \int q_\sigma(\tilde{\mathbf{x}}) \, \|\boldsymbol{s}_\theta(\tilde{\mathbf{x}})\|_2^2 \, \mathrm{d}\tilde{\mathbf{x}}$$

$$- \int q_\sigma(\tilde{\mathbf{x}}) \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})^\mathsf{T} \boldsymbol{s}_\theta(\tilde{\mathbf{x}}) \, \mathrm{d}\tilde{\mathbf{x}}$$

$$= \text{const.} + \frac{1}{2} E_{\tilde{\mathbf{x}} \sim q_\sigma}[\|\boldsymbol{s}_\theta(\tilde{\mathbf{x}})\|_2^2] - \int q_\sigma(\tilde{\mathbf{x}}) \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})^\mathsf{T} \boldsymbol{s}_\theta(\tilde{\mathbf{x}}) \, \mathrm{d}\tilde{\mathbf{x}}$$

Denoising score matching

$\mathbf{x}$

$p_{\text{data}}(\mathbf{x})$

$\downarrow$

$q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})$

$\downarrow$

$q_\sigma(\tilde{\mathbf{x}})$

$\tilde{\mathbf{x}}$

$$
\begin{aligned}
& -\int q_\sigma(\tilde{\mathbf{x}}) \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})^\top \boldsymbol{s}_\theta(\tilde{\mathbf{x}}) \, \mathrm{d}\tilde{\mathbf{x}} \\
&= -\int q_\sigma(\tilde{\mathbf{x}}) \frac{1}{q_\sigma(\tilde{\mathbf{x}})} \nabla_{\tilde{\mathbf{x}}} q_\sigma(\tilde{\mathbf{x}})^\top \boldsymbol{s}_\theta(\tilde{\mathbf{x}}) \, \mathrm{d}\tilde{\mathbf{x}} \\
&= -\int \nabla_{\tilde{\mathbf{x}}} q_\sigma(\tilde{\mathbf{x}})^\top \boldsymbol{s}_\theta(\tilde{\mathbf{x}}) \, \mathrm{d}\tilde{\mathbf{x}} \\
&= -\int \nabla_{\tilde{\mathbf{x}}} \left(\int p_{\text{data}}(\mathbf{x}) q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) \, \mathrm{d}\mathbf{x} \right)^\top \boldsymbol{s}_\theta(\tilde{\mathbf{x}}) \, \mathrm{d}\tilde{\mathbf{x}} \\
&= -\int \left(\int p_{\text{data}}(\mathbf{x}) \nabla_{\tilde{\mathbf{x}}} q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) \, \mathrm{d}\mathbf{x} \right)^\top \boldsymbol{s}_\theta(\tilde{\mathbf{x}}) \, \mathrm{d}\tilde{\mathbf{x}} \\
&= -\int \left(\int p_{\text{data}}(\mathbf{x}) q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) \, \mathrm{d}\mathbf{x} \right)^\top \boldsymbol{s}_\theta(\tilde{\mathbf{x}}) \, \mathrm{d}\tilde{\mathbf{x}} \\
&= -\iint p_{\text{data}}(\mathbf{x}) q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})^\top \boldsymbol{s}_\theta(\tilde{\mathbf{x}}) \, \mathrm{d}\mathbf{x} \, \mathrm{d}\tilde{\mathbf{x}} \\
&= -E_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x}), \tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})} \left[\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})^\top \boldsymbol{s}_\theta(\tilde{\mathbf{x}}) \right]
\end{aligned}
$$

Denoising score matching

$\mathbf{x}$

$\tilde{\mathbf{x}}$

$p_{\text{data}}(\mathbf{x})$

$q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})$

$q_\sigma(\tilde{\mathbf{x}})$

$$\frac{1}{2} E_{\tilde{\mathbf{x}} \sim q_\sigma}[\|\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}) - \boldsymbol{s}_\theta(\tilde{\mathbf{x}})\|_2^2]$$

$$= \text{const.} + \frac{1}{2} E_{\tilde{\mathbf{x}} \sim q_\sigma}[\|\boldsymbol{s}_\theta(\tilde{\mathbf{x}})\|_2^2] - \int q_\sigma(\tilde{\mathbf{x}}) \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})^\mathsf{T} \boldsymbol{s}_\theta(\tilde{\mathbf{x}}) \, \mathrm{d}\tilde{\mathbf{x}}$$

$$= \text{const.} + \frac{1}{2} E_{\tilde{\mathbf{x}} \sim q_\sigma}[\|\boldsymbol{s}_\theta(\tilde{\mathbf{x}})\|_2^2] - E_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x}), \tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}[\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})^\mathsf{T} \boldsymbol{s}_\theta(\tilde{\mathbf{x}})]$$

$$= \text{const.} + \frac{1}{2} E_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x}), \tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}[\|\boldsymbol{s}_\theta(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})\|_2^2]$$

$$- \frac{1}{2} E_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x}), \tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}[\|\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})\|_2^2]$$

$$= \text{const.} + \frac{1}{2} E_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x}), \tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}[\|\boldsymbol{s}_\theta(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})\|_2^2] + \text{const.}$$

$$= \frac{1}{2} E_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x}), \tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}[\|\boldsymbol{s}_\theta(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})\|_2^2] + \text{const.}$$

Denoising score matching

- Estimate the score of a noise-perturbed distribution

$$
\begin{aligned}
&\frac{1}{2} E_{\tilde{\mathbf{x}} \sim p_{\text{data}}}[\|\boldsymbol{s}_\theta(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})\|_2^2] \\
=&\frac{1}{2} E_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x}), \tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})}[\|\boldsymbol{s}_\theta(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})\|_2^2] + \text{const.}
\end{aligned}
$$

- $\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})$ is easy to compute
  - $q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) = \mathcal{N}(\tilde{\mathbf{x}} \mid \mathbf{x}, \sigma^2 \boldsymbol{I})$
  - $\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) = -\dfrac{\tilde{\mathbf{x}} - \mathbf{x}}{\sigma^2}$
- **Pros:** efficient to optimize even for very high dimensional data, and useful for optimal denoising.
- **Con:** cannot estimate the score of clean data (noise-free)

Denoising score matching

- Sample a minibatch of datapoints $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\} \sim p_{\text{data}}(\mathbf{x})$
- Sample a minibatch of perturbed datapoints

$$\{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \cdots, \tilde{\mathbf{x}}_n\} \sim q_\sigma(\tilde{\mathbf{x}})$$

- Estimate the denoising score matching loss with empirical means

$$\tilde{\mathbf{x}}_i \sim q_\sigma(\tilde{\mathbf{x}}_i \mid \mathbf{x}_i)$$

$$\frac{1}{2n}\sum_{i=1}^{n}[\|\boldsymbol{s}_\theta(\tilde{\mathbf{x}}_i) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}_i \mid \mathbf{x}_i)\|_2^2]$$

- If Gaussian perturbation

$$\frac{1}{2n}\sum_{i=1}^{n}\left[\left\|\boldsymbol{s}_\theta(\tilde{\mathbf{x}}_i) + \frac{\tilde{\mathbf{x}}_i - \mathbf{x}_i}{\sigma^2}\right\|_2^2\right]$$

- Stochastic gradient descent
- Need to choose a very small σ!

Denoising Score Matching (Vincent, 2011)

- Consider the perturbed distribution

$$q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) = \mathcal{N}(\mathbf{x}; \sigma^2 I) \qquad q_\sigma(\tilde{\mathbf{x}}) = \int p(\mathbf{x}) q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) \mathrm{d}\mathbf{x}$$

- Score estimation for $\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})$ is easier

Score matching $\frac{1}{2}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}})}[\|\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}) - s_\theta(\tilde{\mathbf{x}})\|_2^2]$

Denoising $= \frac{1}{2}\mathbb{E}_{p(\mathbf{x})}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}[\| \frac{1}{\sigma^2}(\mathbf{x} - \tilde{\mathbf{x}}) - s_\theta(\tilde{\mathbf{x}})\|_2^2] + \text{const.}$

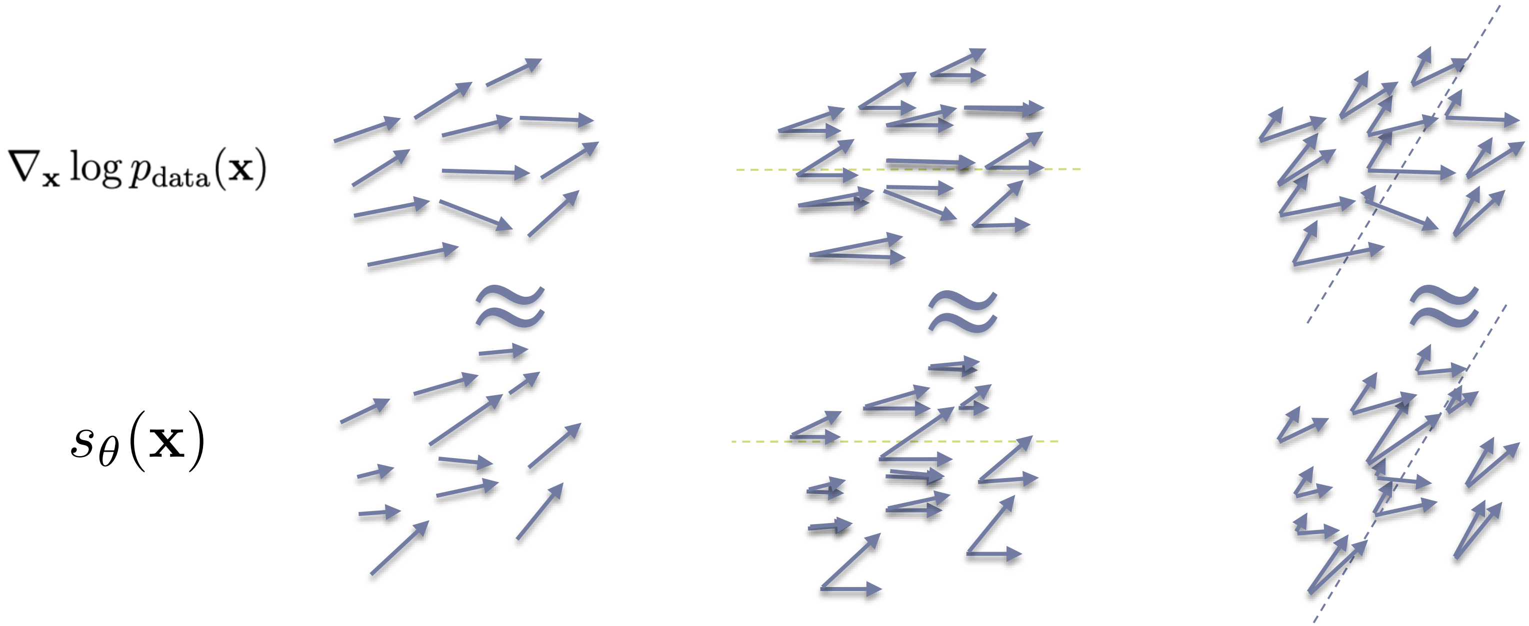
x

$\tilde{\mathbf{x}}$ = x + noise

$s_\theta(\tilde{\mathbf{x}})$ tries to estimate the noise that was added to produce $\tilde{\mathbf{x}}$

Sliced score matching

- One dimensional problems should be easier.
- Consider projections onto random directions.



Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

Sliced score matching

- **Objective:** Sliced Fisher Divergence

$$\frac{1}{2}E_{\mathbf{v}\sim p_{\mathbf{v}}}E_{\mathbf{x}\sim p_{\text{data}}}[(\mathbf{v}^\intercal \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{v}^\intercal \boldsymbol{s}_\theta(\mathbf{x}))^2]$$

- **Integration by parts**

$$E_{\mathbf{v}\sim p_{\mathbf{v}}}E_{\mathbf{x}\sim p_{\text{data}}}\left[\mathbf{v}^\intercal \nabla_{\mathbf{x}} \boldsymbol{s}_\theta(\mathbf{x})\mathbf{v} + \frac{1}{2}(\mathbf{v}^\intercal \boldsymbol{s}_\theta(\mathbf{x}))^2\right]$$

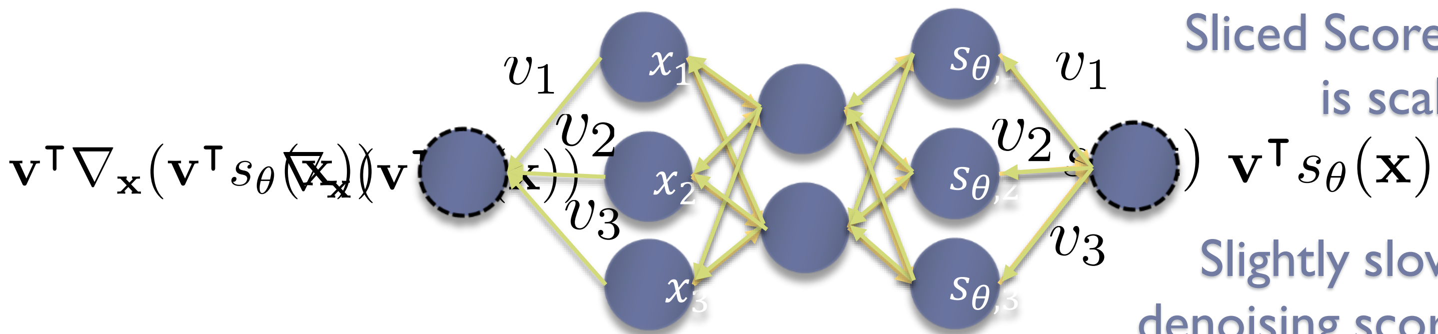
$$\begin{pmatrix} v_1 & v_2 & v_3 \end{pmatrix} \begin{pmatrix} \frac{\partial s_{\theta,1}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\theta,1}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\theta,1}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\theta,2}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\theta,2}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\theta,2}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\theta,3}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\theta,3}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\theta,3}(\mathbf{x})}{\partial x_3} \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$$

Sliced Score Matching

fast

Computing Jacobian-vector products is scalable

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} s_\theta(\mathbf{x}) \mathbf{v} = \mathbf{v}^\intercal \nabla_{\mathbf{x}} (\mathbf{v}^\intercal s_\theta(\mathbf{x}))$$



One Backprop!

Sliced Score Matching is scalable

Slightly slower than denoising score matching

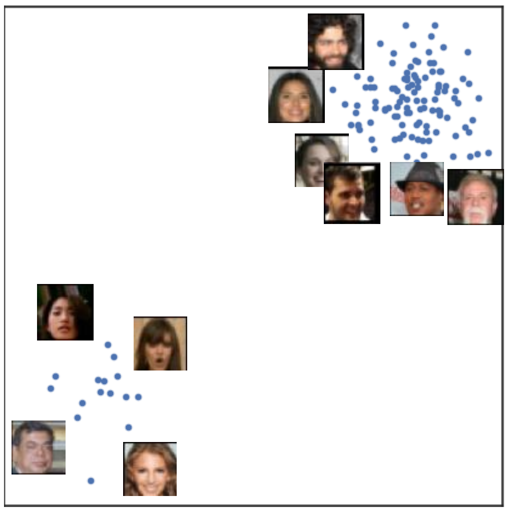
Sliced score matching

- Sample a minibatch of datapoints $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\} \sim p_{\text{data}}(\mathbf{x})$
- Sample a minibatch of projection directions $\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n\} \sim p_{\mathbf{v}}$
- Estimate the sliced score matching loss with empirical means

$$\frac{1}{n}\sum_{i=1}^{n}\left[\mathbf{v}_i^{\mathsf{T}}\nabla_{\mathbf{x}}\boldsymbol{s}_\theta(\mathbf{x}_i)\mathbf{v}_i + \frac{1}{2}\left(\mathbf{v}_i^{\mathsf{T}}\boldsymbol{s}_\theta(\mathbf{x}_i)\right)^2\right]$$

- The projection distribution is typically Gaussian or Rademacher
- Stochastic gradient descent
- Can use more projections per datapoint to boost performance

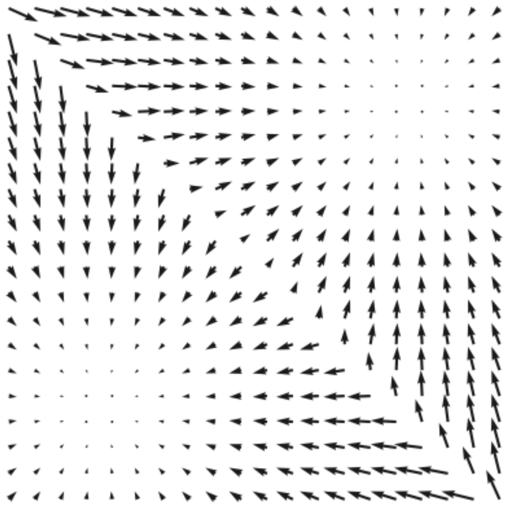
Score-based generative modeling



Data samples

$$\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\} \sim p_{\text{data}}(\mathbf{x})$$

Score Matching



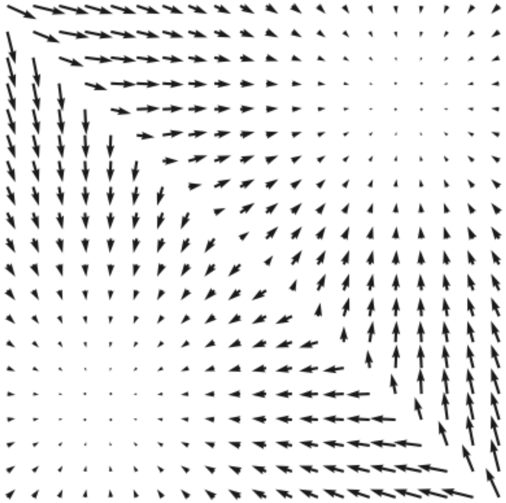
Scores

$$\boldsymbol{s}_\theta(\mathbf{x}) \approx \nabla_\mathbf{x} \log p_{\text{data}}(\mathbf{x})$$

?

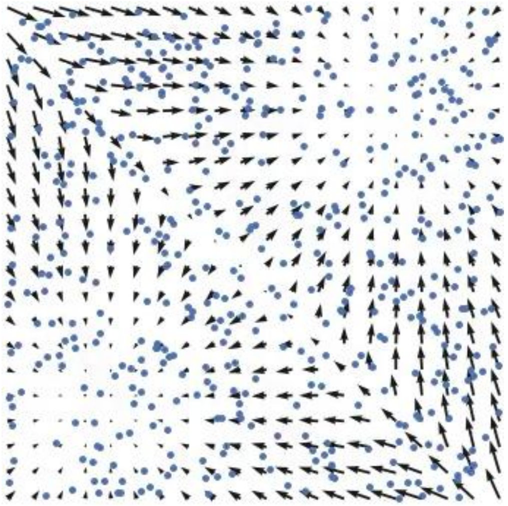
New samples

From scores to samples: Langevin MCMC



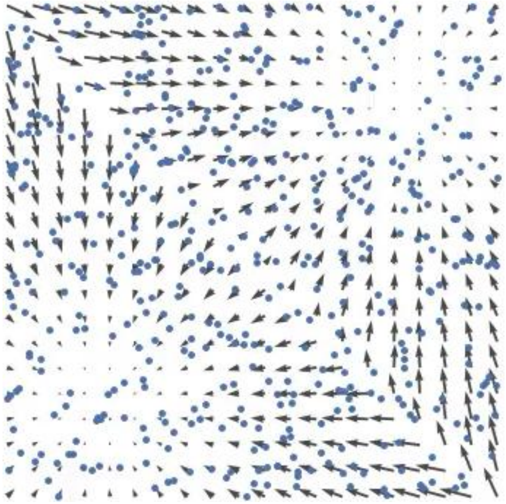
Scores

$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$



Follow the scores

$$\tilde{\mathbf{x}}_{t+1} \leftarrow \tilde{\mathbf{x}}_t + \frac{\epsilon}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t)$$



Follow noisy scores: Langevin MCMC

$$\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$\tilde{\mathbf{x}}_{t+1} \leftarrow \tilde{\mathbf{x}}_t + \frac{\epsilon}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t) + \sqrt{\epsilon}\,\mathbf{z}_t$$

Langevin dynamics sampling

Sample from $p(\mathbf{x})$ using only the score $\nabla_{\mathbf{x}} \log p(\mathbf{x})$

- Initialize $\mathbf{x}^0 \sim \pi(\mathbf{x})$
- Repeat for $t \leftarrow 1, 2, \cdots, T$

$$\mathbf{z}^t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$$
$$\mathbf{x}^t \leftarrow \mathbf{x}^{t-1} + \frac{\epsilon}{2}\nabla_{\mathbf{x}} \log p(\mathbf{x}^{t-1}) + \sqrt{\epsilon}\, \mathbf{z}^t$$

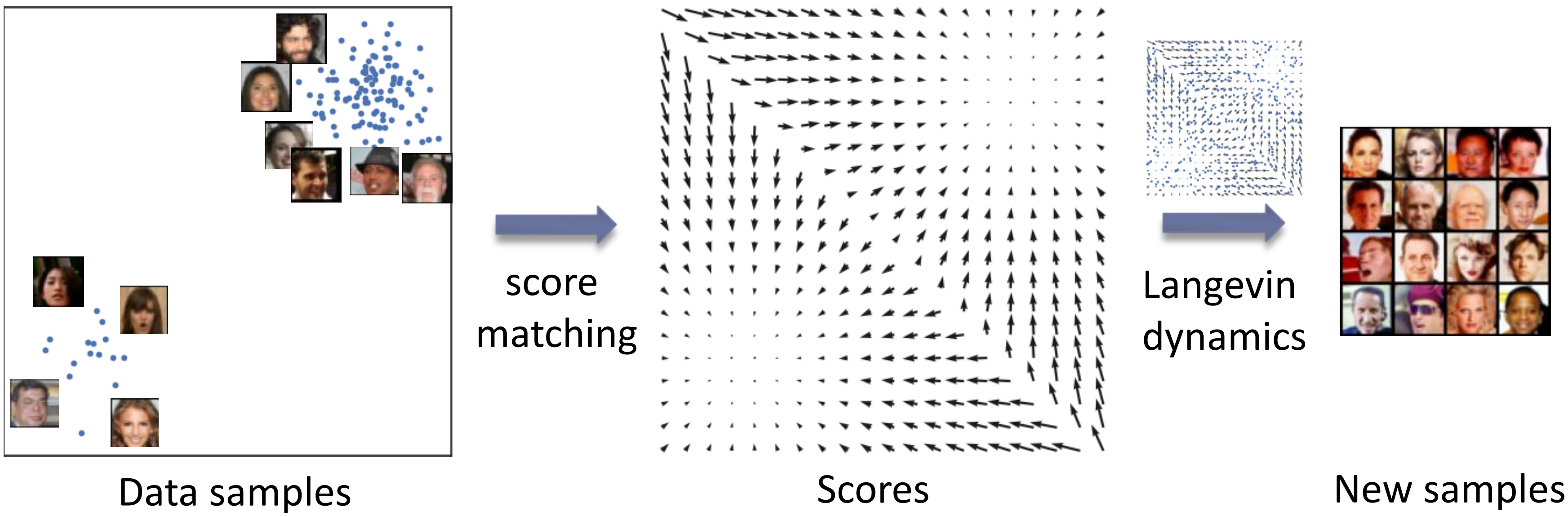
- If $\epsilon \rightarrow 0$ and $T \rightarrow \infty$, we are guaranteed to have

$$\mathbf{x}^T \sim p(\mathbf{x})$$

- Langevin dynamics + score estimation

$$\boldsymbol{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p(\mathbf{x})$$

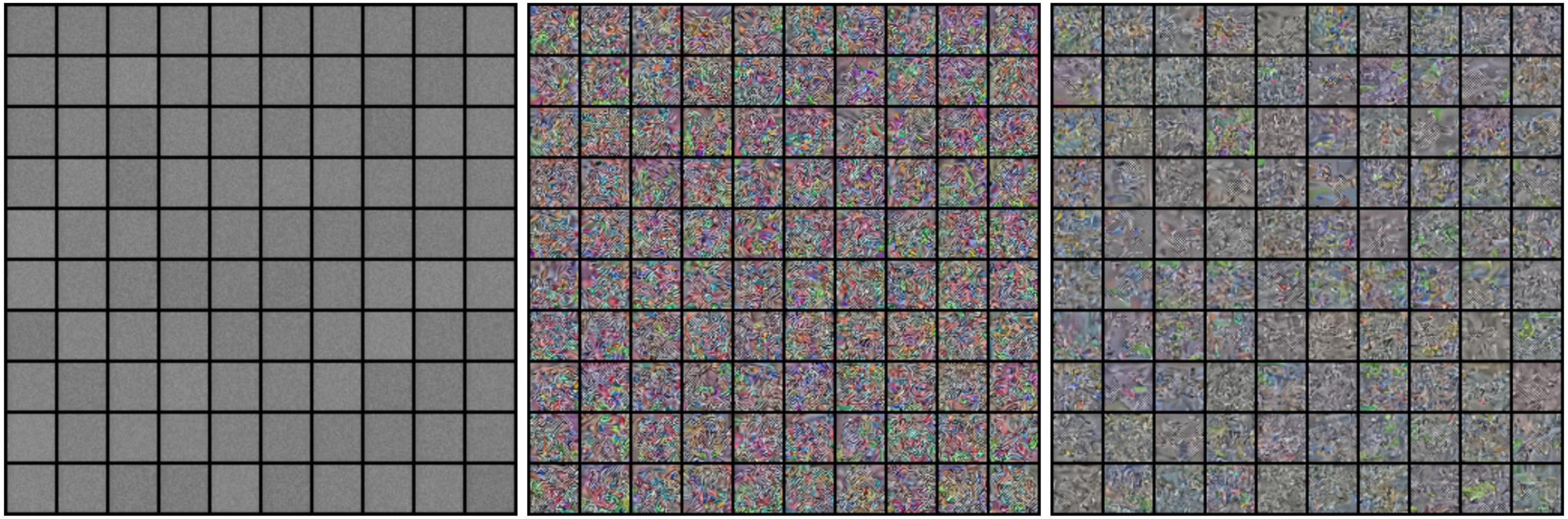
Score-based generative modeling



$$\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\} \sim p_{\text{data}}(\mathbf{x})$$

$$\boldsymbol{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$$

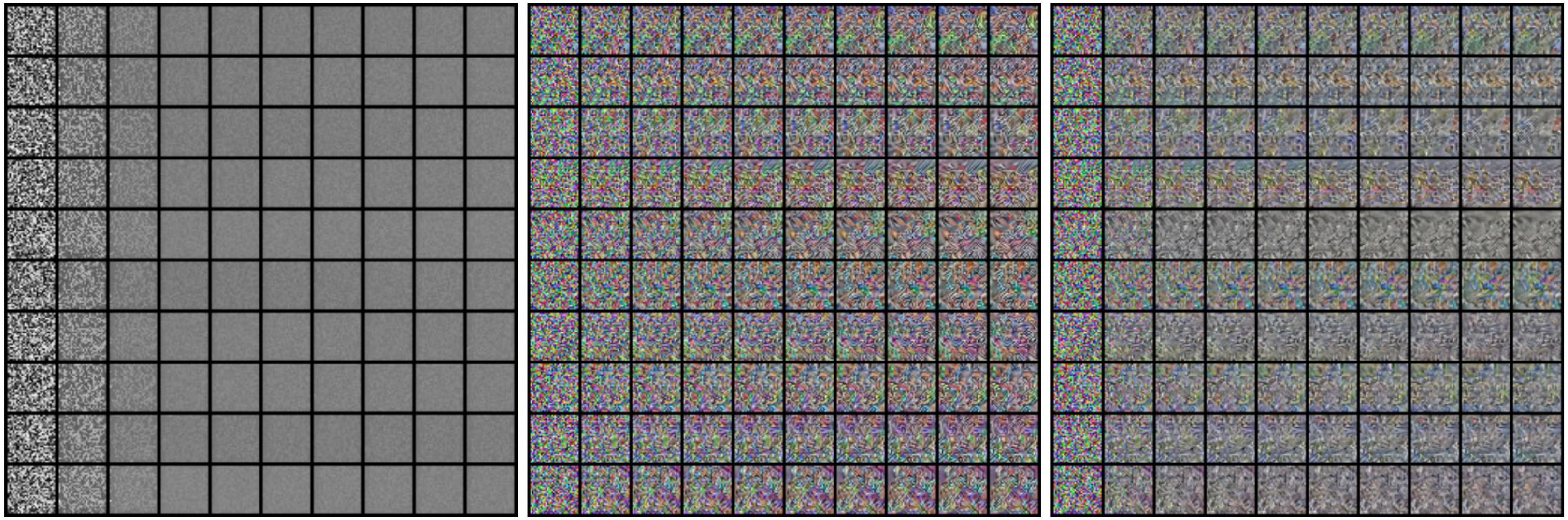
Score-based generative modeling: results



(a) MNIST (b) CelebA (c) CIFAR-10

Final samples

Score-based generative modeling: results

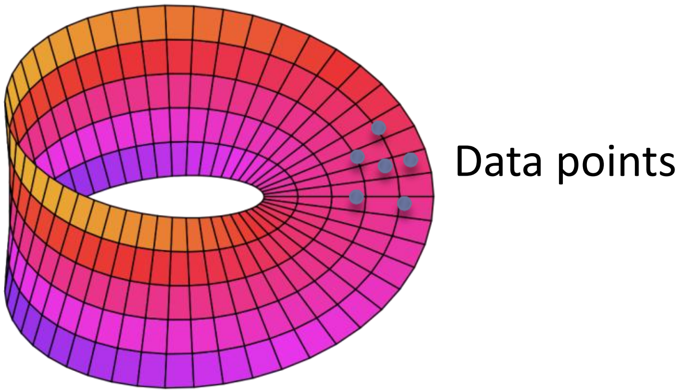


(a) MNIST (b) CelebA (c) CIFAR-10

Langevin sampling process

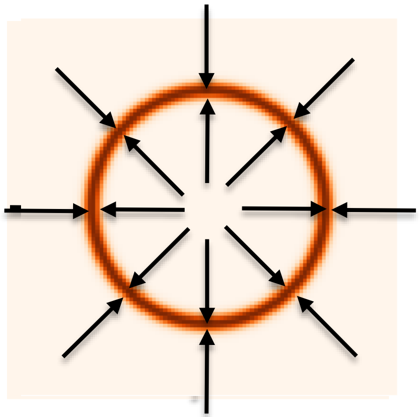
Pitfall 1: manifold hypothesis

- Manifold hypothesis.



- Data score is undefined.

$$\nabla_{\mathbf{x}} \log p_{\mathrm{data}}(\mathbf{x})$$

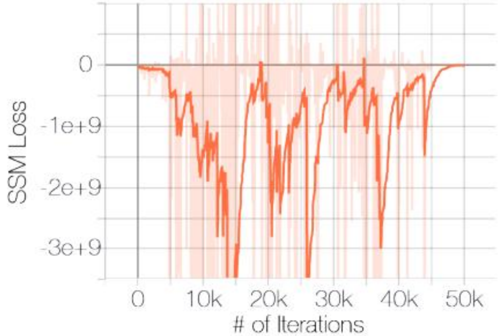


Pitfall 1: manifold hypothesis

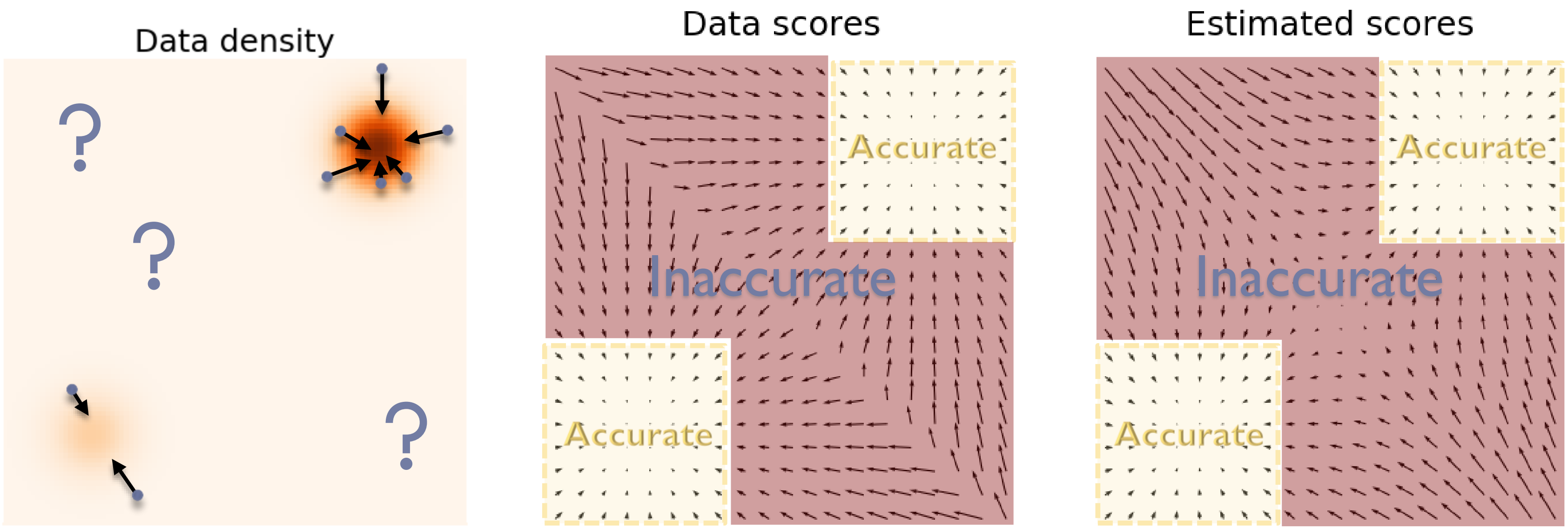
- Fitting the data with a low-dimensional linear manifold (PCA)

3072
Dim
2165

- Score estimation on CIFAR-10.



Challenge in low data density regions



$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2]$$

Langevin MCMC will have trouble exploring low density regions

Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Pitfall 3: slow mixing of Langevin dynamics between data modes

- Suppose the data distribution has two modes with disjoint supports:

$$p_{\text{data}}(\mathbf{x}) = \pi p_1(\mathbf{x}) + (1-\pi) p_2(\mathbf{x})$$

$$\mathcal{A} \cap \mathcal{B} = \varnothing \qquad p_{\text{data}}(\mathbf{x}) = \begin{cases} \pi p_1(\mathbf{x}), & \mathbf{x} \in \mathcal{A} \\ (1-\pi) p_2(\mathbf{x}), & \mathbf{x} \in \mathcal{B} \end{cases}$$

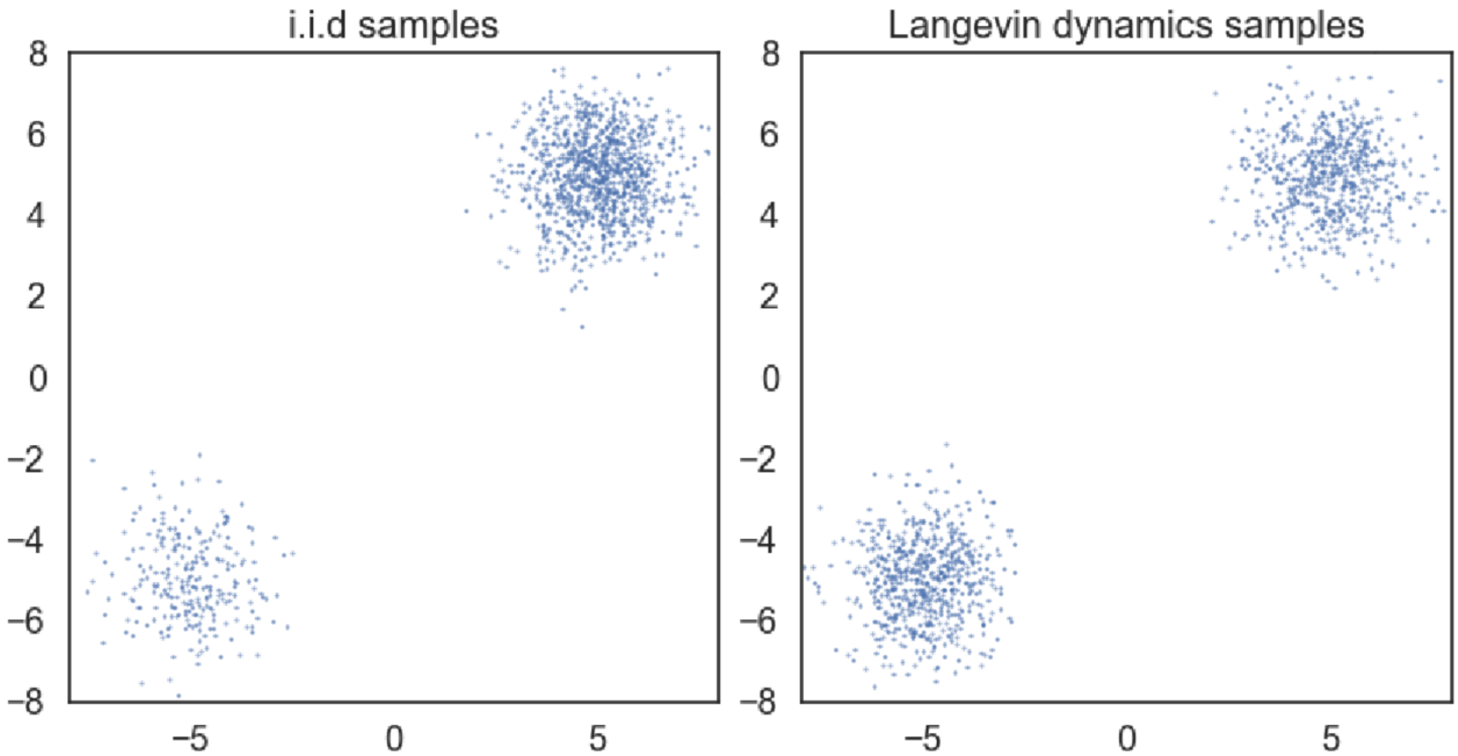
- Data score function:

$$\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) = \begin{cases} \nabla_{\mathbf{x}}[\log \pi + \log p_1(\mathbf{x})], & \mathbf{x} \in \mathcal{A} \\ \nabla_{\mathbf{x}}[\log(1-\pi) + \log p_2(\mathbf{x})], & \mathbf{x} \in \mathcal{B} \end{cases}$$

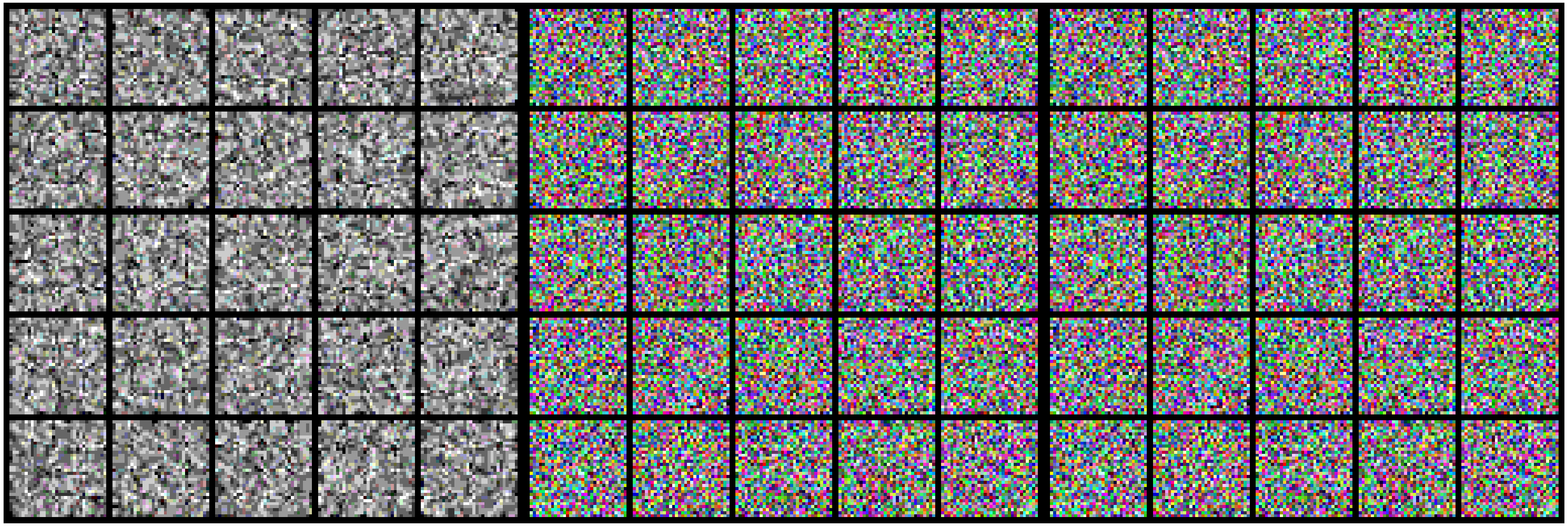
$$= \begin{cases} \nabla_{\mathbf{x}} \log p_1(\mathbf{x}), & \mathbf{x} \in \mathcal{A} \\ \nabla_{\mathbf{x}} \log p_2(\mathbf{x}), & \mathbf{x} \in \mathcal{B} \end{cases}$$

- The score function has no dependence on the mode weighting π at all!
- Langevin sampling will not reflect π

Pitfall 3: slow mixing of Langevin dynamics between data modes



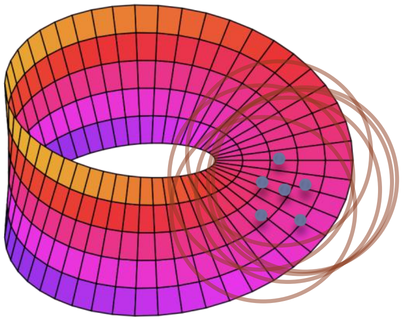
After fixing these pitfalls

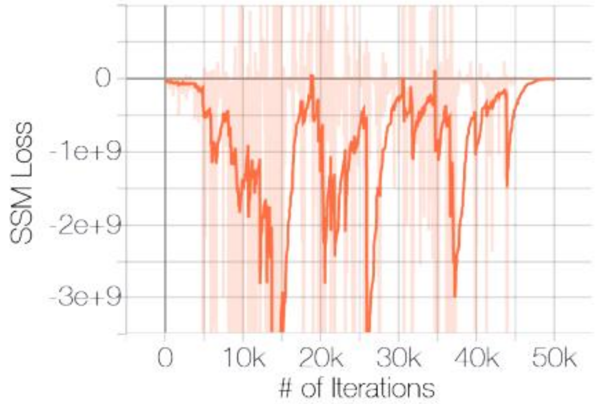


Song, Yang, and Stefano Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

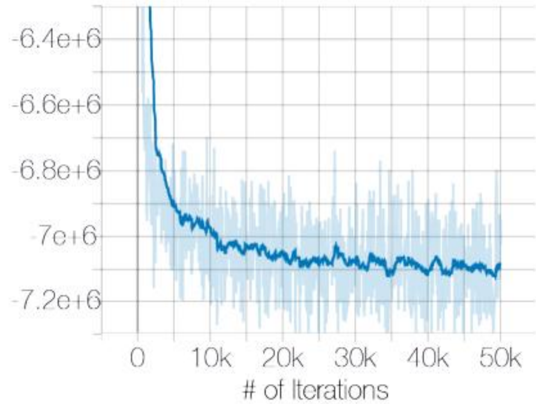
Gaussian perturbation

- The solution to all pitfalls: **Gaussian perturbation!**
- Manifold + noise
- Score matching on noisy data.





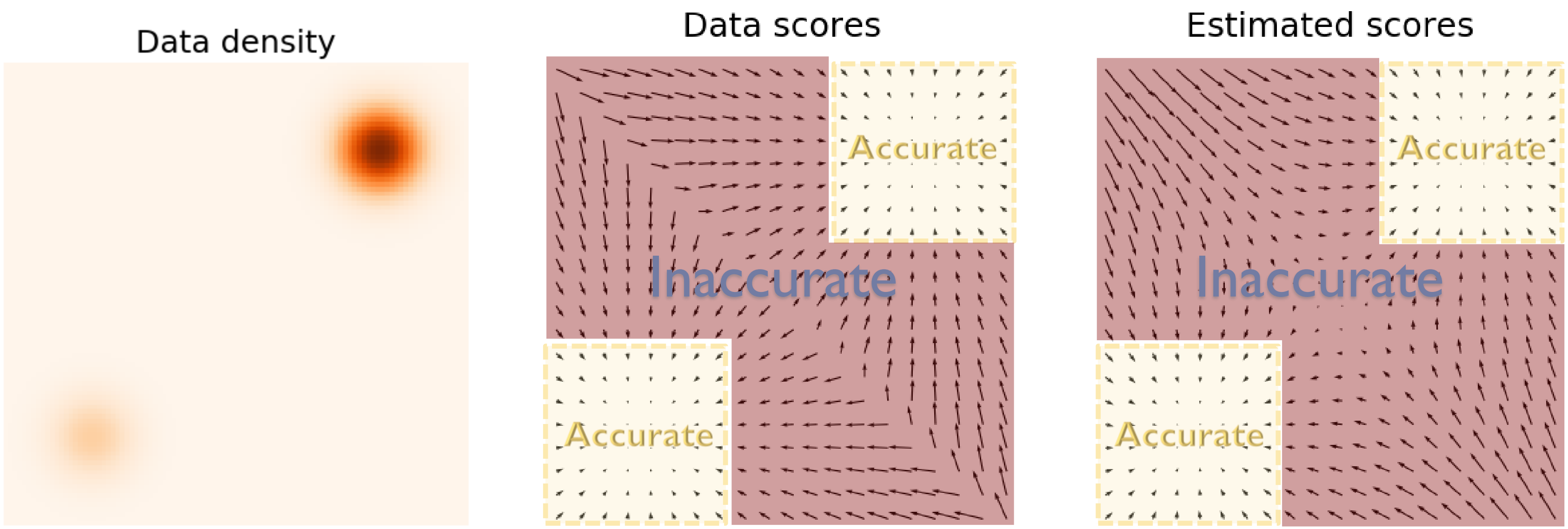
CIFAR-10



Noisy CIFAR-10

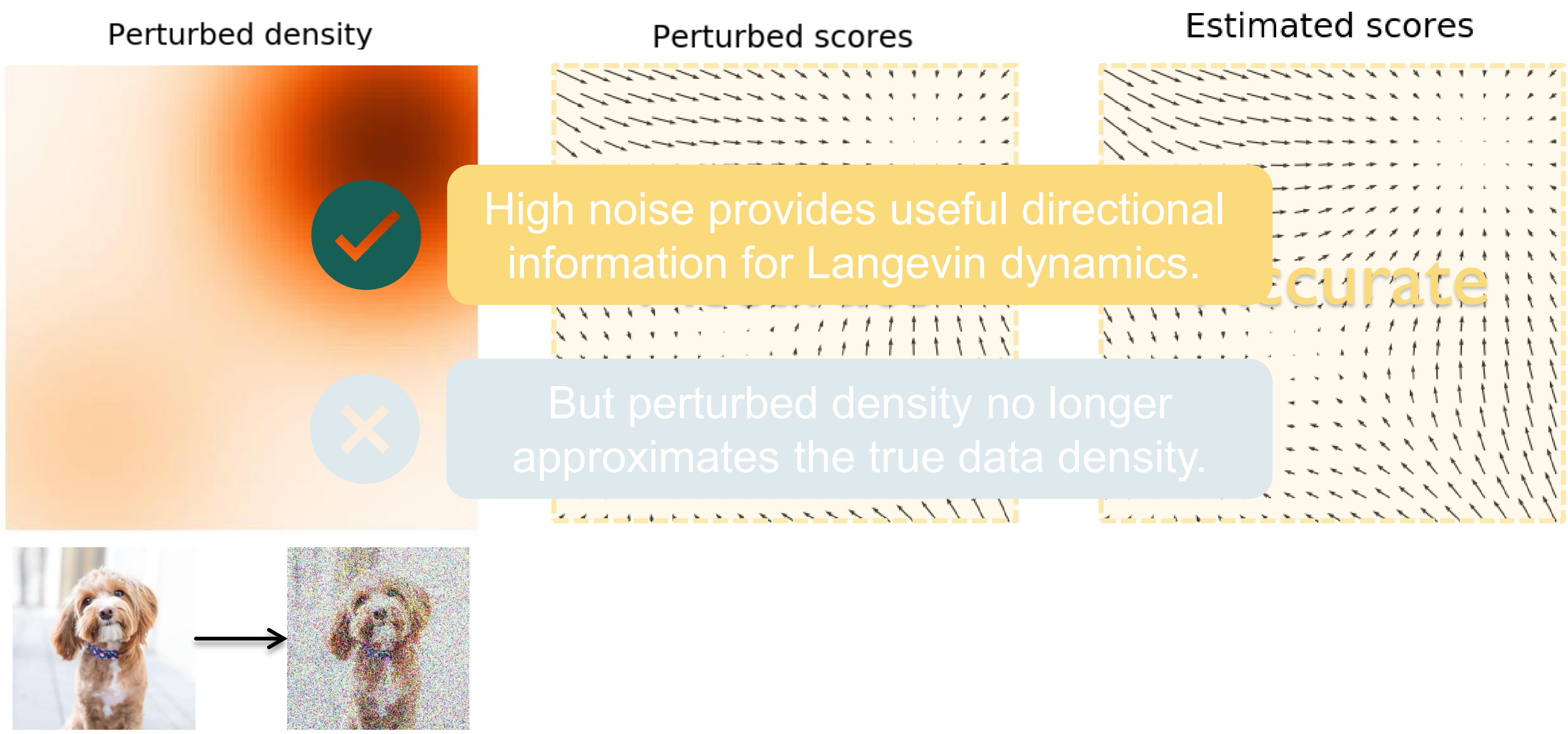
$$\mathcal{N}(0; 0.0001)$$

Challenge in low data density regions



Song and Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution." NeurIPS 2019.

Improving score estimation by adding noise

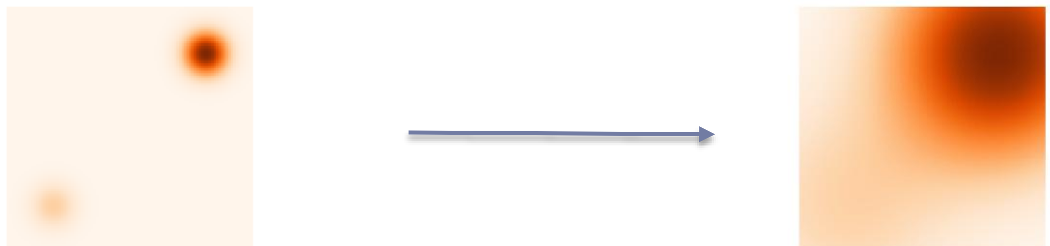


High noise provides useful directional information for Langevin dynamics.

But perturbed density no longer approximates the true data density.

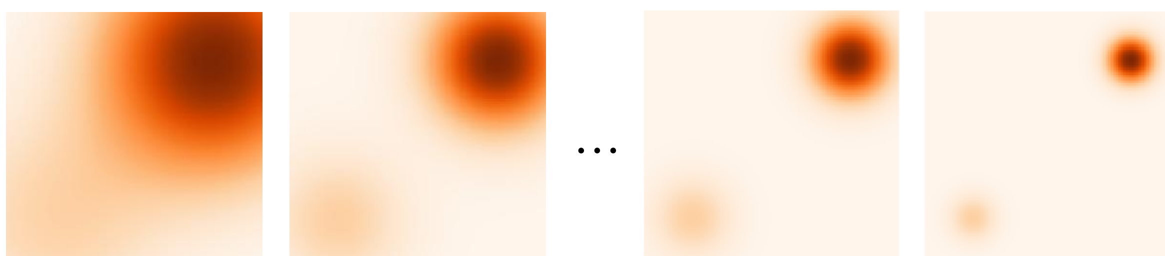
Multi-scale Noise Perturbation

- How much noise to add?

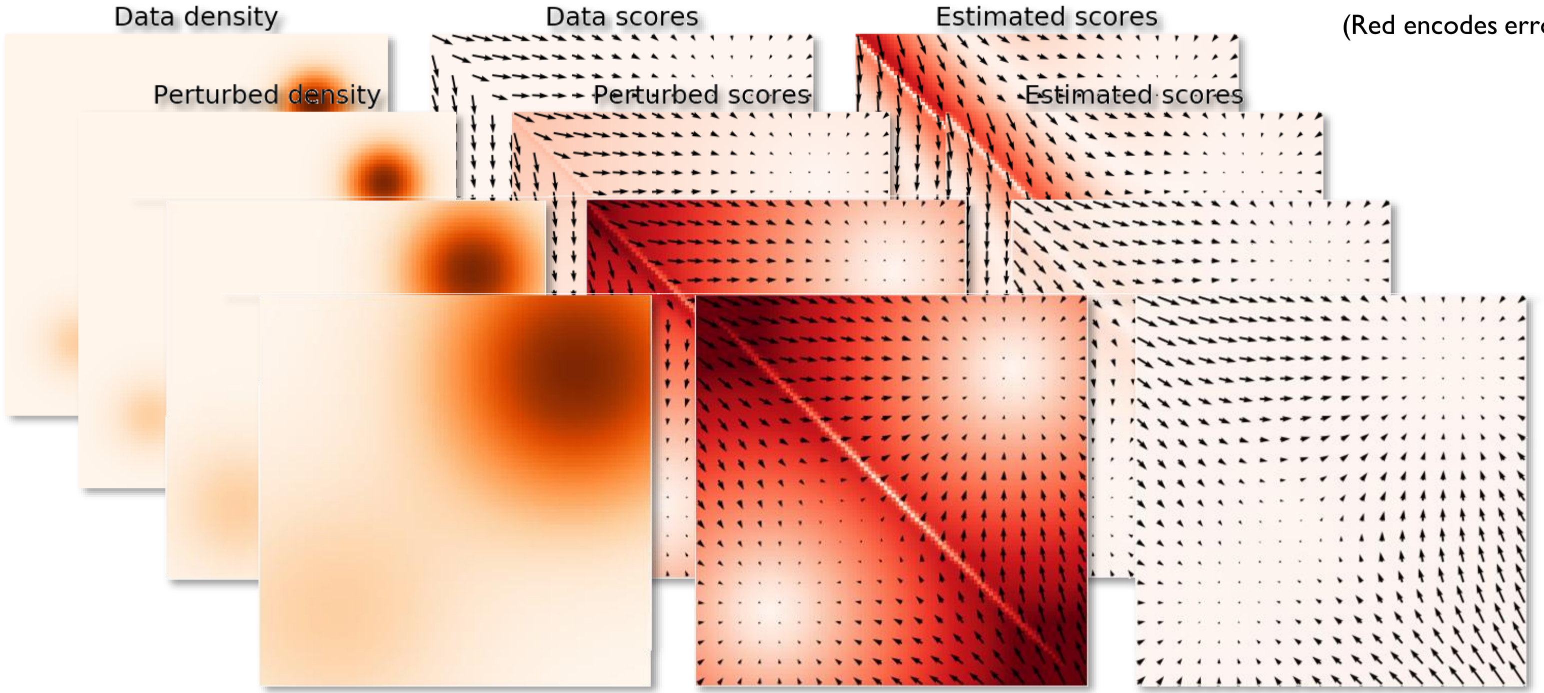


- Multi-scale noise perturbations.

$$\sigma_1 > \sigma_2 > \cdots > \sigma_{L-1} > \sigma_L$$



Trading off Data Quality and Estimation Accuracy



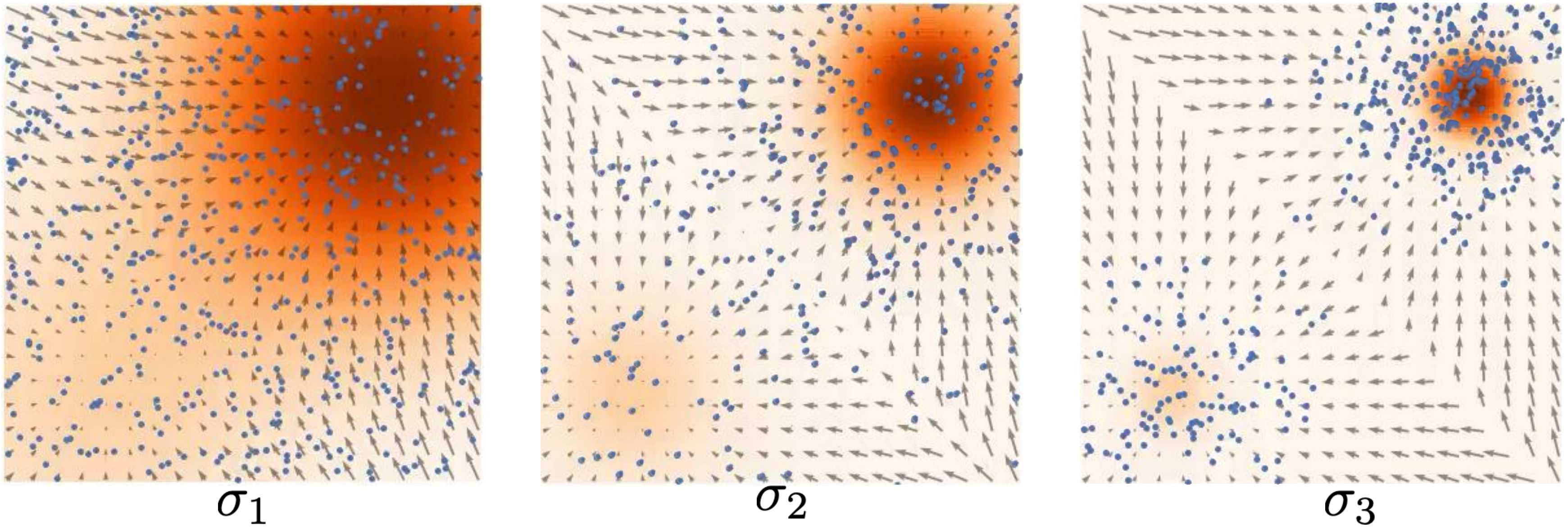
(Red encodes error)

Worse data quality!

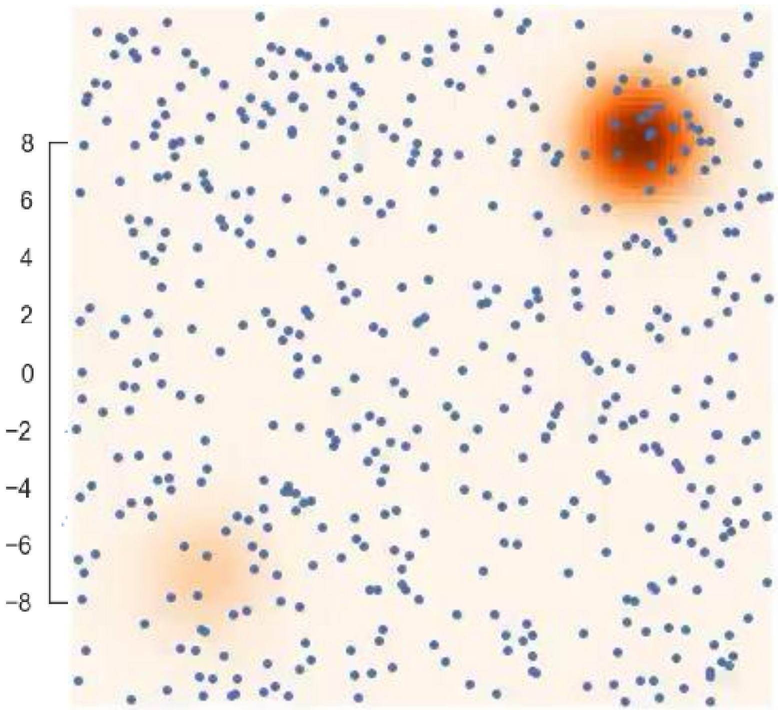
Better score estimation!

Annealed Langevin Dynamics: Joint Scores to Samples

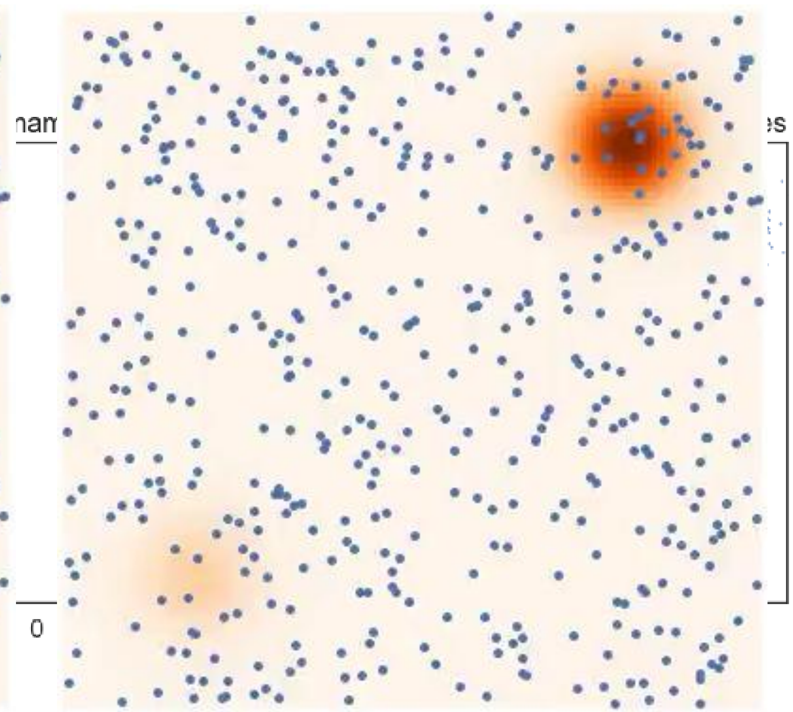
- Sample using $\sigma_1, \sigma_2, \cdots, \sigma_L$ sequentially with Langevin dynamics.
- Anneal down the noise level.
- Samples used as initialization for the next level.



Comparison to the vanilla Langevin dynamics

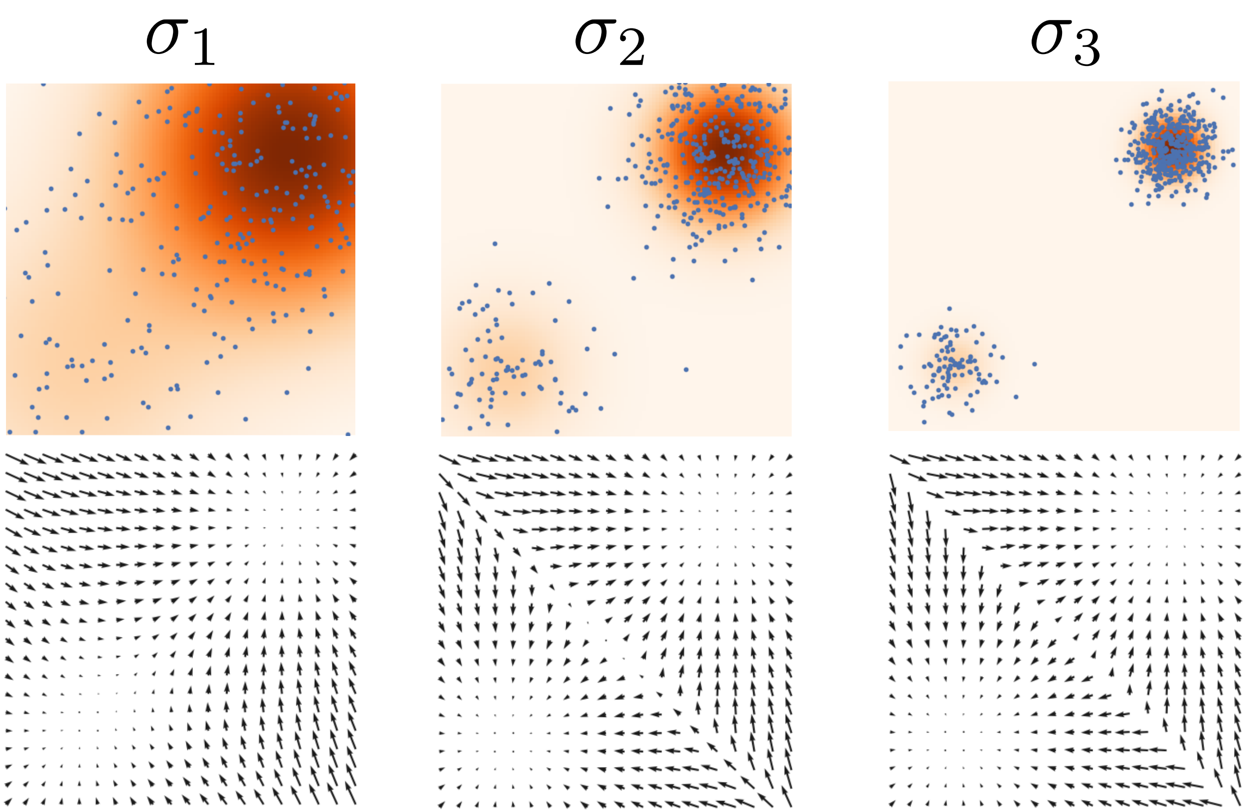


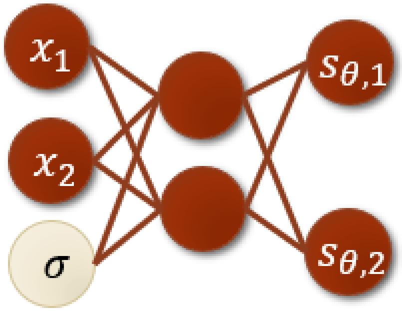
Langevin dynamics



Annealed Langevin dynamics

Joint Score Estimation via Noise Conditional Score Networks





Noise Conditional Score Network (NCSN)

Training noise conditional score networks

- Which score matching loss?
  - Sliced score matching?
  - Denoising score matching?
- Denoising score matching is naturally suitable, since the goal is to estimate the score of perturbed data distributions.
- Weighted combination of denoising score matching losses

$$\frac{1}{L}\sum_{i=1}^{L}\lambda(\sigma_i)E_{q_{\sigma_i}(\mathbf{x})}[\|\nabla_{\mathbf{x}}\log q_{\sigma_i}(\mathbf{x})-\boldsymbol{s}_\theta(\mathbf{x},\sigma_i)\|_2^2]$$

$$=\frac{1}{L}\sum_{i=1}^{L}\lambda(\sigma_i)E_{\mathbf{x}\sim p_{\text{data}},\mathbf{z}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}[\|\nabla_{\tilde{\mathbf{x}}}\log q_{\sigma_i}(\tilde{\mathbf{x}}\mid\mathbf{x})-\boldsymbol{s}_\theta(\tilde{\mathbf{x}},\sigma_i)\|_2^2]+\text{const.}$$

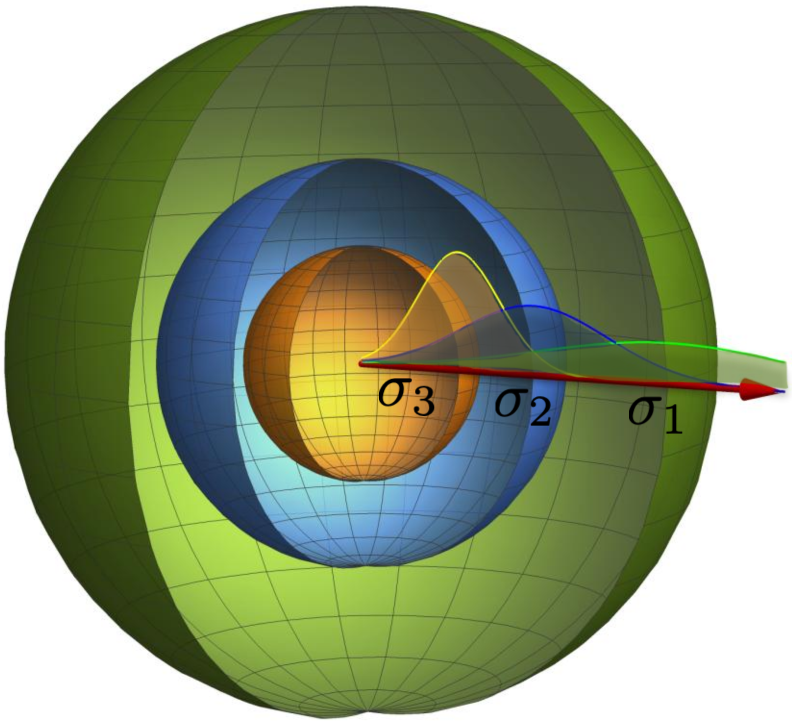
$$=\frac{1}{L}\sum_{i=1}^{L}\lambda(\sigma_i)E_{\mathbf{x}\sim p_{\text{data}},\mathbf{z}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[\left\|\boldsymbol{s}_\theta(\mathbf{x}+\sigma_i\mathbf{z},\sigma_i)+\frac{\mathbf{z}}{\sigma_i}\right\|_2^2\right]+\text{const.}$$

Choosing noise scales

- **Key intuition:** adjacent noise scales should have sufficient overlap to facilitate transitioning across noise scales in annealed Langevin dynamics.
- A geometric progression with sufficient length.

$$\sigma_1 > \sigma_2 > \sigma_3 > \cdots > \sigma_{L-1} > \sigma_L$$

$$\frac{\sigma_1}{\sigma_2} = \frac{\sigma_2}{\sigma_3} = \cdots = \frac{\sigma_{L-1}}{\sigma_L}$$



Choosing the weighting function

- Weighted combination of denoising score matching losses

$$\frac{1}{L}\sum_{i=1}^{L}\lambda(\sigma_i)E_{\mathbf{x}\sim p_{\text{data}},\mathbf{z}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[\left\|\boldsymbol{s}_\theta(\mathbf{x}+\sigma_i\mathbf{z},\sigma_i)+\frac{\mathbf{z}}{\sigma_i}\right\|_2^2\right]$$

$$\lambda:\mathbb{R}_{>0}\rightarrow\mathbb{R}_{>0}$$

- How to choose the weighting function: $\lambda(\sigma_i)=\sigma_i^2$

- **Goal:** balancing different score matching losses in the sum →

$$\frac{1}{L}\sum_{i=1}^{L}\sigma_i^2E_{\mathbf{x}\sim p_{\text{data}},\mathbf{z}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[\left\|\boldsymbol{s}_\theta(\mathbf{x}+\sigma_i\mathbf{z},\sigma_i)+\frac{\mathbf{z}}{\sigma_i}\right\|_2^2\right]$$

$$=\frac{1}{L}\sum_{i=1}^{L}E_{\mathbf{x}\sim p_{\text{data}},\mathbf{z}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[\|\sigma_i\boldsymbol{s}_\theta(\mathbf{x}+\sigma_i\mathbf{z},\sigma_i)+\mathbf{z}\|_2^2\right]$$

$$=\frac{1}{L}\sum_{i=1}^{L}E_{\mathbf{x}\sim p_{\text{data}},\mathbf{z}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[\|\boldsymbol{\epsilon}_\theta(\mathbf{x}+\sigma_i\mathbf{z},\sigma_i)+\mathbf{z}\|_2^2\right]$$

$$[\ \boldsymbol{\epsilon}_\theta(\cdot,\sigma_i):=\sigma_i\boldsymbol{s}_\theta(\cdot,\sigma_i)\ ]$$

Training noise conditional score networks

- Sample a mini-batch of datapoints $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\} \sim p_{\text{data}}$
- Sample a mini-batch of noise scale indices

$$\{i_1, i_2, \cdots, i_n\} \sim \mathcal{U}\{1, 2, \cdots, L\}$$

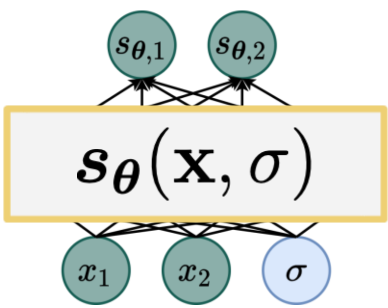
- Sample a mini-batch of Gaussian noise $\{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n\} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
- Estimate the weighted mixture of score matching losses

$$\frac{1}{n} \sum_{k=1}^{n} \left[\| \sigma_{i_k} \boldsymbol{s}_\theta(\mathbf{x}_k + \sigma_{i_k} \mathbf{z}_k, \sigma_{i_k}) + \mathbf{z}_k \|_2^2 \right]$$

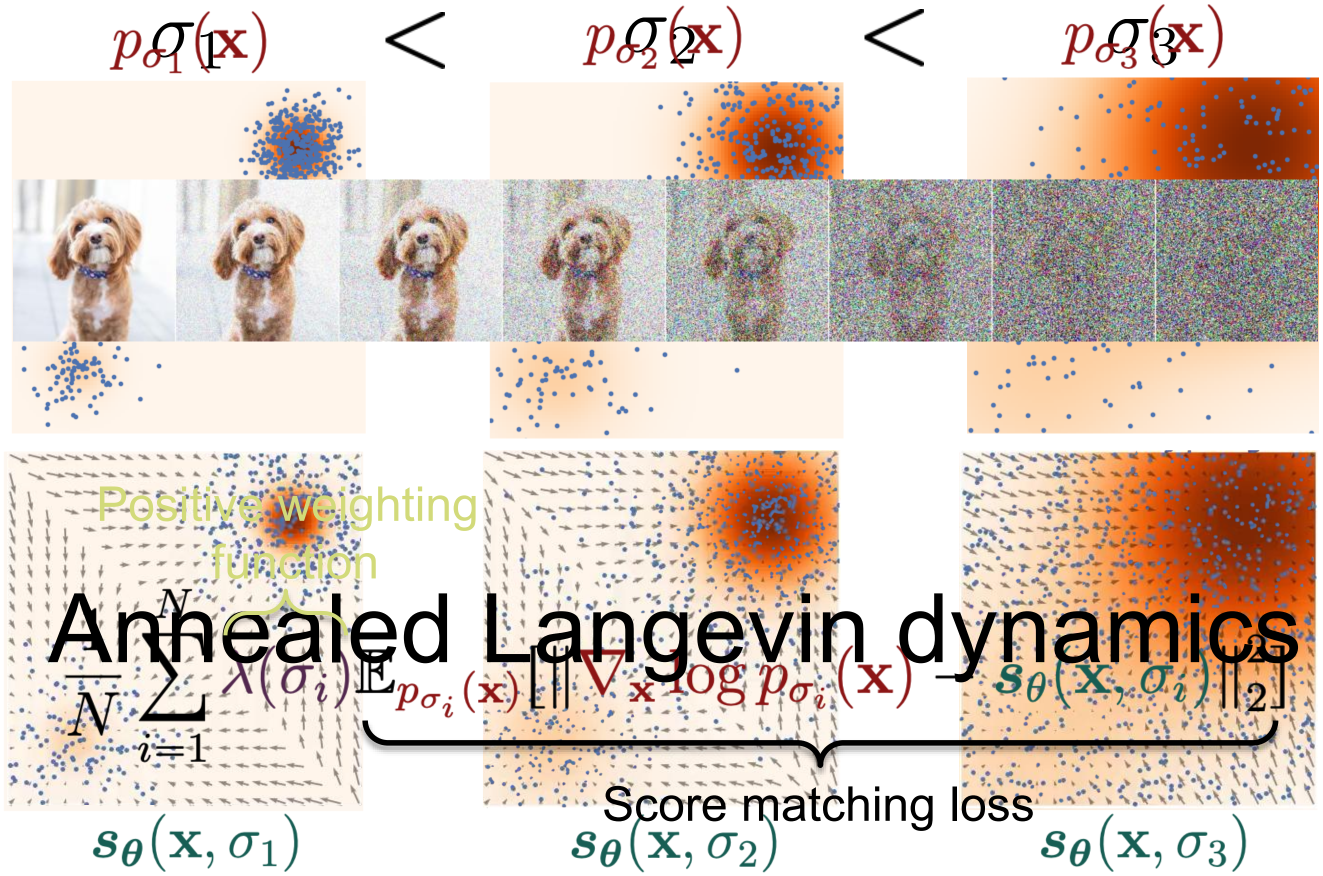
- Stochastic gradient descent
- As efficient as training one single non-conditional score-based model

Using multiple noise levels

Data



Noise Conditional Score Model



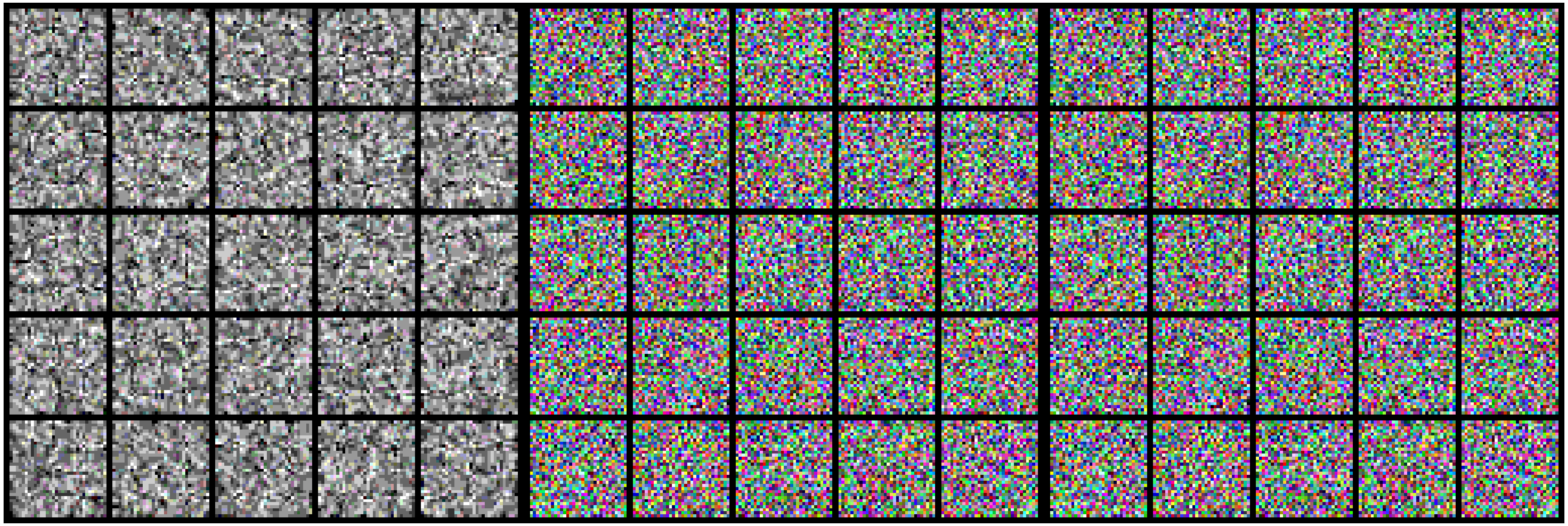
Positive weighting function

$$\frac{1}{N}\sum_{i=1}^{N} \lambda(\sigma_i)\mathbb{E}_{p_{\sigma_i}(\mathbf{x})}\left[\|\nabla_{\mathbf{x}} \log p_{\sigma_i}(\mathbf{x}) - s_\theta(\mathbf{x}, \sigma_i)\|_2^2\right]$$

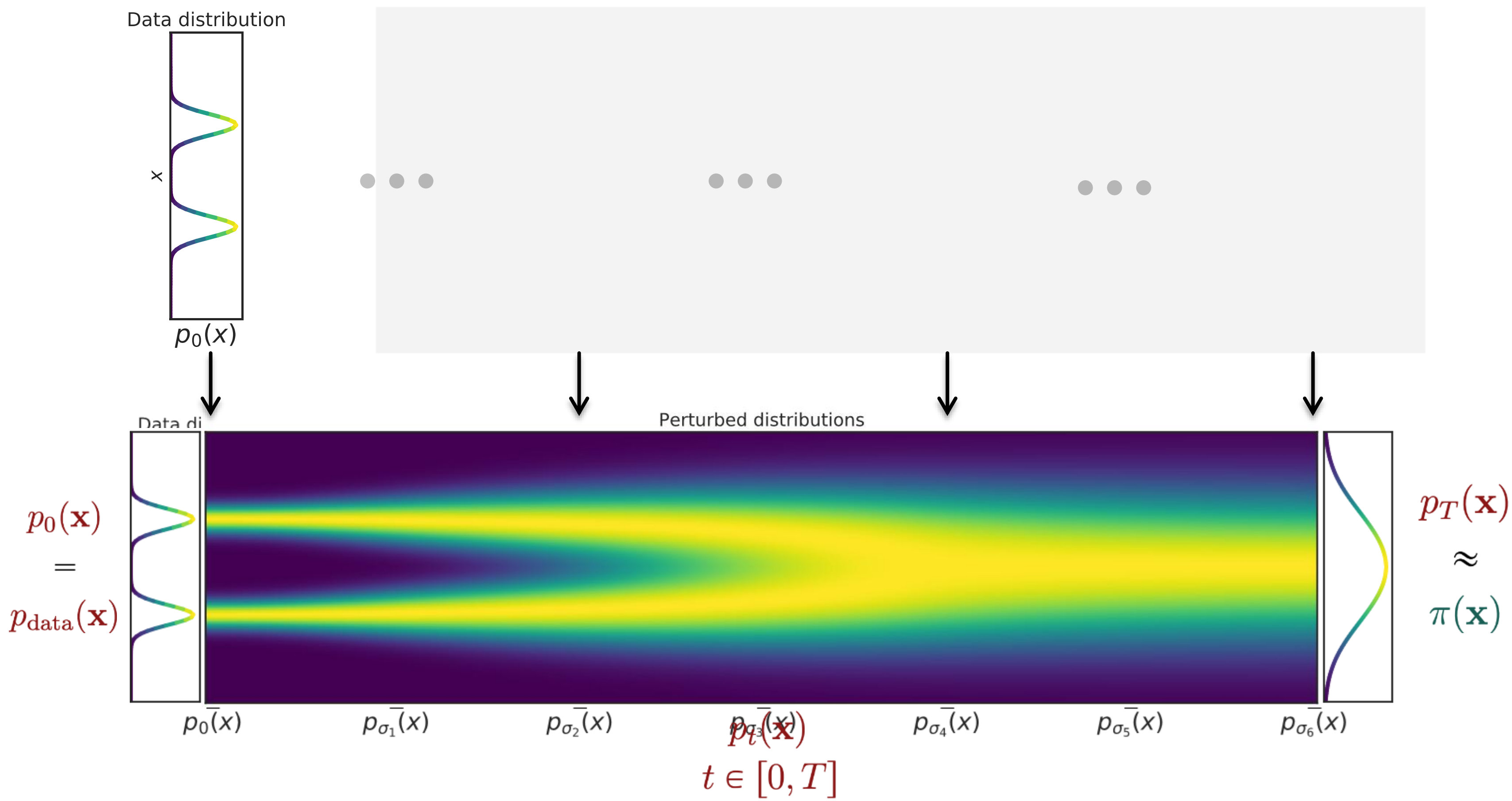
Score matching loss

Annealed Langevin dynamics

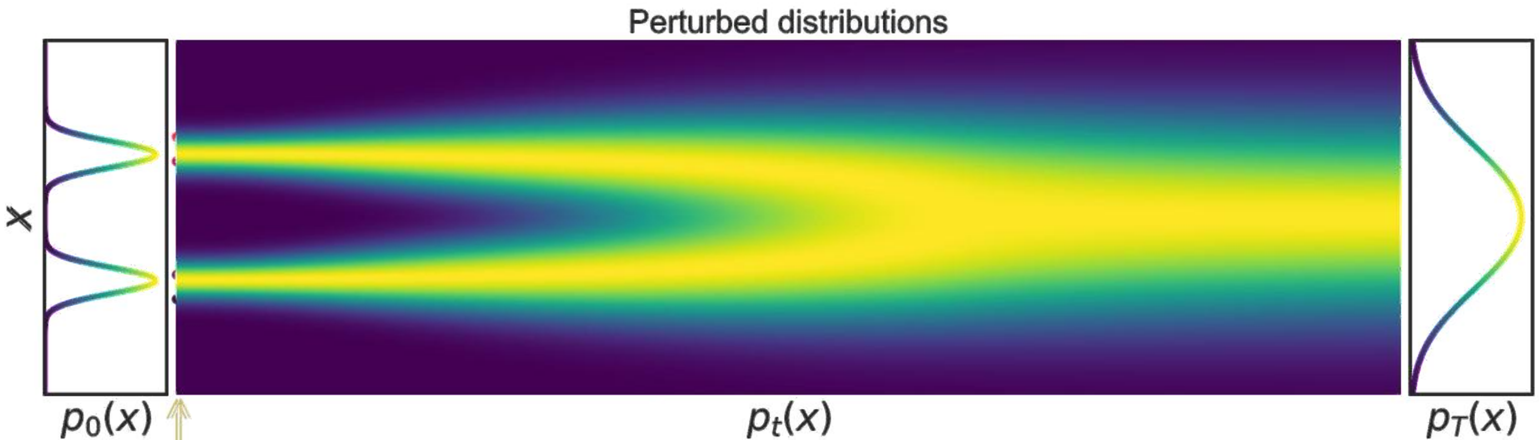
Experiments: Sampling



Infinite noise levels



Perturbing data with stochastic processes



Stochastic process

$\{\mathbf{x}_t\}_{t\in[0,T]}$

Probability densities

$\{p_t(\mathbf{x})\}_{t\in[0,T]}$

Stochastic differential equation (SDE)

$$\mathrm{d}\mathbf{x}_t = \boldsymbol{f}(\mathbf{x}_t, t)\,\mathrm{d}t + g(t)\,\mathrm{d}\mathbf{w}_t$$

Deterministic drift

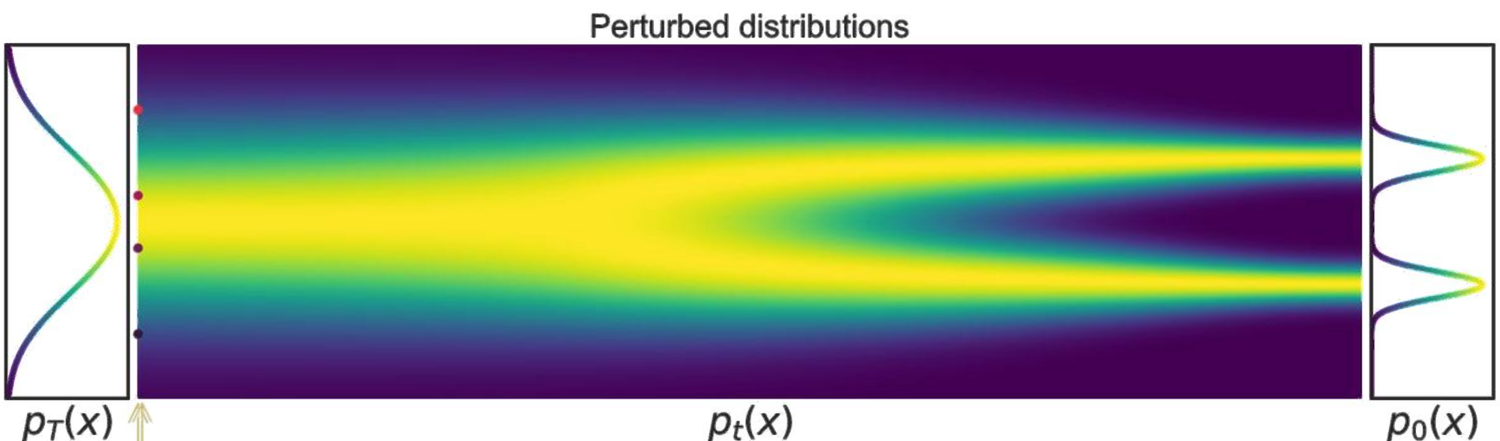
Infinitesimal noise

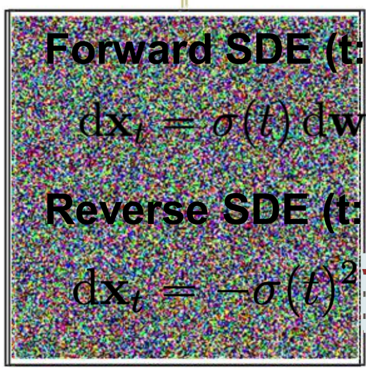
WLOG: Toy SDE

$$\mathrm{d}\mathbf{x}_t = \sigma(t)\,\mathrm{d}\mathbf{w}_t$$

$$p_T(\mathbf{x}) \approx \pi(\mathbf{x})$$

Generation via reverse stochastic processes





$\pi(\mathbf{x}) \approx p_T(\mathbf{x})$

Forward SDE (t: 0→T)

$$\mathrm{d}\mathbf{x}_t = \sigma(t)\,\mathrm{d}\mathbf{w}_t$$

Reverse SDE (t: T→0)

$$\mathrm{d}\mathbf{x}_t = -\sigma(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x}_t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}\bar{\mathbf{w}}_t$$

Infinitesimal noise in the reverse time direction

Score function!

Score-based generative modeling via SDEs

- Time-dependent score-based model

$$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t) \approx \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$$

- Training:

$$\mathbb{E}_{t \in \mathcal{U}(0,T)}[\lambda(t)\mathbb{E}_{p_t(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)\|_2^2]]$$

- Reverse-time SDE

$$\mathrm{d}\mathbf{x} = -\sigma^2(t)\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)\mathrm{d}t + \sigma(t)\mathrm{d}\bar{\mathbf{w}}$$

- Euler-Maruyama (analgous to Euler for ODEs)

$$\mathbf{x} \leftarrow \mathbf{x} - \sigma(t)^2 \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)\Delta t + \sigma(t)\ \mathbf{z} \quad (\mathbf{z} \sim \mathcal{N}(\mathbf{0}, |\Delta t|\ \mathbf{I}))$$

$$t \leftarrow t + \Delta t$$

Song, Sohl-Dickstein, Kingma, Kumar, Ermon, Poole. "Score-Based Generative Modeling through Stochastic Differential Equations." ICLR 2021.

Predictor-Corrector sampling methods

- Predictor-Corrector sampling.
  - **Predictor:** Numerical SDE solver
  - **Corrector:** Score-based MCMC