Deep Generative Models

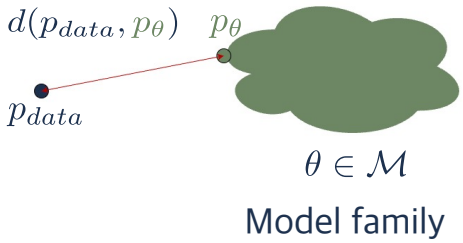
13. Score-Based Models

• 국가수리과학연구소 산업수학혁신센터 김민중

Summary

- **Representation**: how do we model the joint distribution of many random variables?
  - Need compact representation
- **Learning**: what is the right way to compare probability distributions?

$\mathbf{x}_i \sim p_{data}$
$i = 1, 2, ..., N$

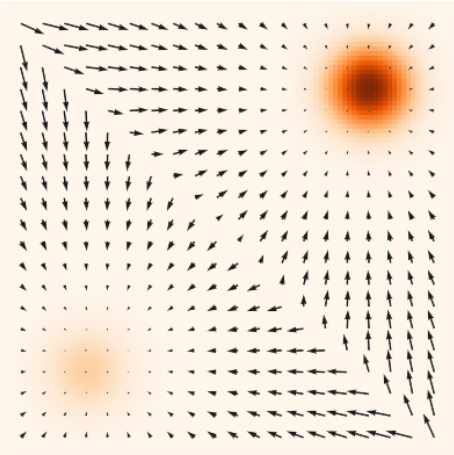


- **Inference**: how do we invert the generation process

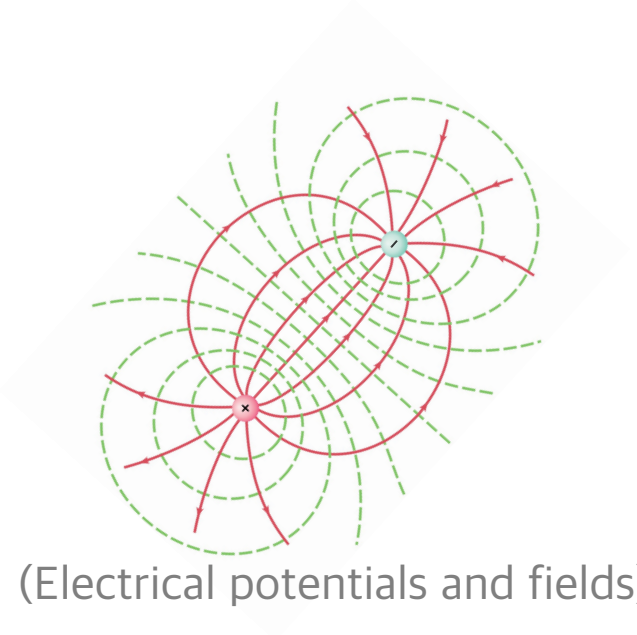
Representation: score functions

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

$$\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$$



(pdf and score)



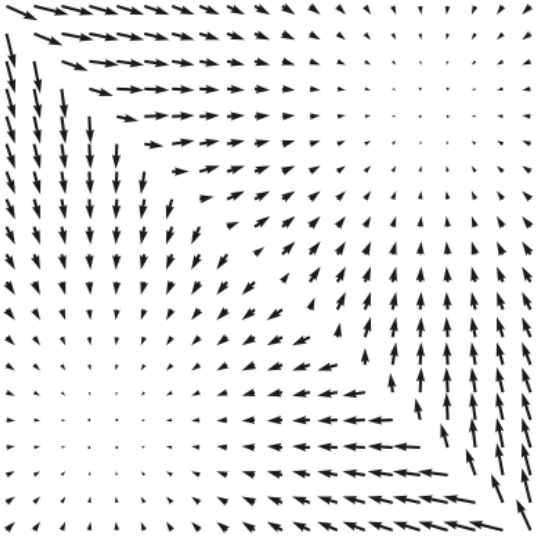
(Electrical potentials and fields)

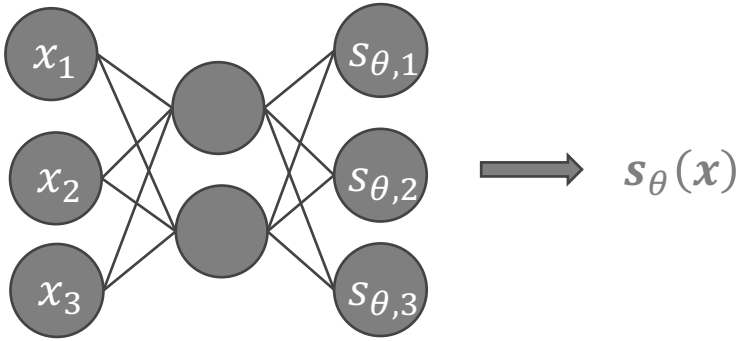
Score-based models

- Directly model the vector field of gradients

$$\boldsymbol{s}_\theta(\boldsymbol{x}): \mathbb{R}^{\mathrm{d}} \rightarrow \mathbb{R}^d$$

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





Score estimation

- Training the score-based model from data points
- **Score matching**

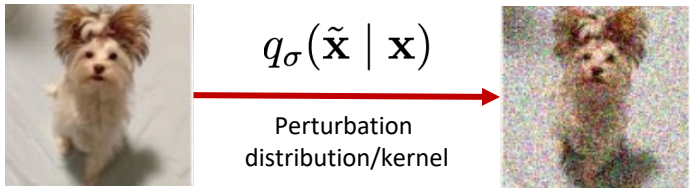
$$\frac{1}{2} E_{\boldsymbol{x}\sim p_{data}}[\, \|\nabla_{\boldsymbol{x}} \log p_{data}(\boldsymbol{x}) - \boldsymbol{s}_\theta(\boldsymbol{x})\|_2^2]$$

$$= E_{\boldsymbol{x}\sim p_{data}}\left[\frac{1}{2}\|\boldsymbol{s}_\theta(\boldsymbol{x})\|_2^2 + tr\big(\nabla_{\boldsymbol{x}}\boldsymbol{s}_\theta(\boldsymbol{x})\big)\right] + \text{conts.}$$

- Not scalable for deep score-based models and high dimensional data

Denoising score matching

$x \sim p_{data}(x)$
Data distribution



$\tilde{x} \sim q_\sigma(\tilde{x})$
Noise-perturbed data distribution

$$E_{\tilde{x}\sim q_\sigma}[\,\|\nabla_{\tilde{x}} \log q_\sigma(\tilde{x}) - s_\theta(\tilde{x})\|_2^2]$$
$$= E_{x\sim p_{data}(x)} E_{\tilde{x}\sim q_\sigma(\tilde{x}|x)}[\|\nabla_{\tilde{x}} \log q_\sigma(\tilde{x}|x) - s_\theta(\tilde{x})\|_2^2] + \text{const.}$$
$$= E_{x\sim p_{data}(x)} E_{z\sim N(0,I)}\left[\left\|\frac{1}{\sigma} z + s_\theta(x + \sigma z)\right\|_2^2\right] + \text{const.}$$

- **Pros**
  - more scalable than score matching
  - reduces score estimation to a denoising task
- **Con**: cannot estimate the score of clean data (noise-free)

$$s_\theta(x) \approx \nabla_x \log q_\sigma(x) \neq \nabla_x \log p_{data}(x)$$

Sliced score matching

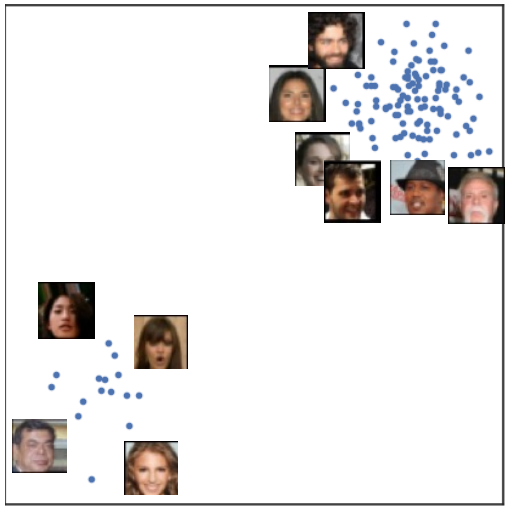
- **Objective**: Sliced Fisher Divergence

$$\frac{1}{2} E_{\boldsymbol{v}\sim p_v} E_{\boldsymbol{x}\sim p_{data}} \left[\left(\boldsymbol{v}^T \nabla_{\boldsymbol{x}} \log p_{data}(\boldsymbol{x}) - \boldsymbol{v}^T \boldsymbol{s}_\theta(\boldsymbol{x}) \right)^2 \right]$$

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

- Projection distribution $p_{\boldsymbol{v}}$ can be Gaussian or Rademacher
- Pros
  - Much more scalable than score matching
  - Estimates the true data score
- Con
  - Slower than denoising score matching

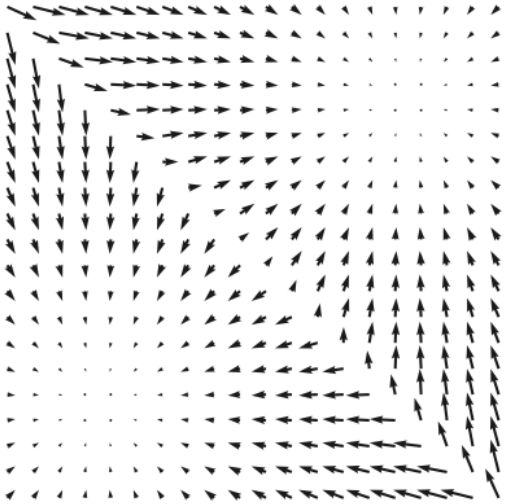
Score-based generative modeling



Data samples

$\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(N)}$

Score Matching



Scores function

$\boldsymbol{s}_\theta(\boldsymbol{x}) \approx \nabla_x \log p_{data}(\boldsymbol{x})$

Langevin dynamics

New samples

Langevin dynamics sampling

- Sample from $p_{data}(\boldsymbol{x})$ using only the scores $\nabla_{\boldsymbol{x}} \log p_{data}(\boldsymbol{x})$
  - Initialize $\boldsymbol{x}^0 \sim \pi(\boldsymbol{x})$
  - Repeat for $t = 0, \ldots, T-1$
    - $\boldsymbol{z} \sim N(0, I)$
    - $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t + \frac{\epsilon}{2} \nabla_{\boldsymbol{x}} \log p_{data}(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}^t} + \sqrt{\epsilon}\boldsymbol{z}$
  - If $\epsilon \to 0$ and $T \to \infty$, then we have $\boldsymbol{x}^T$ converges to a sample from p_{data}
- Langevin dynamics + score estimation

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

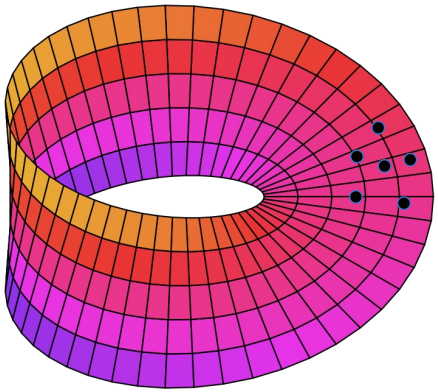
Langevin dynamics sampling

- Using

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

  - Initialize $\boldsymbol{x}^0 \sim \pi(\boldsymbol{x})$
  - Repeat for $t = 0, \ldots, T-1$
    - $\boldsymbol{z} \sim N(0, I)$
    - $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t + \frac{\epsilon}{2}\boldsymbol{s}_\theta(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}^t} + \sqrt{\epsilon}\boldsymbol{z}$

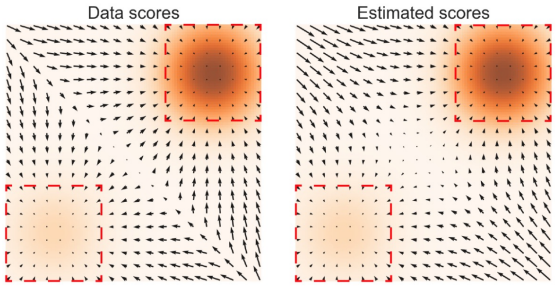
Pitfalls



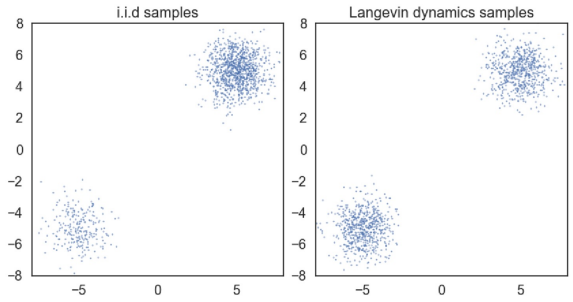
- Manifold hypothesis: data score is undefined

$$\nabla_{\boldsymbol{x}} \log p_{data}(\boldsymbol{x})$$

- Score matching fails in low data density regions

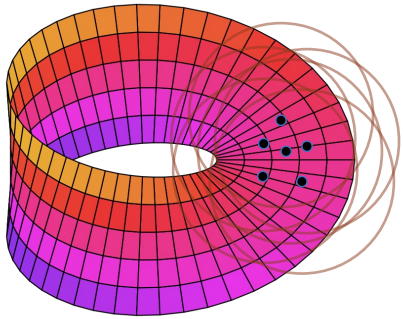


- Langevin dynamics converges very slowly

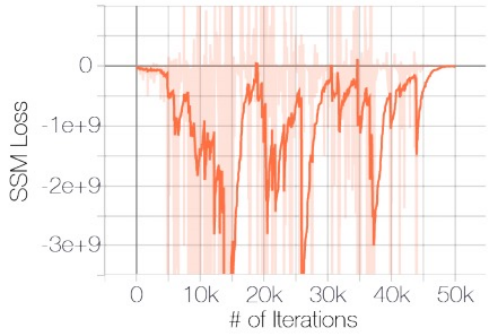


Gaussian perturbation

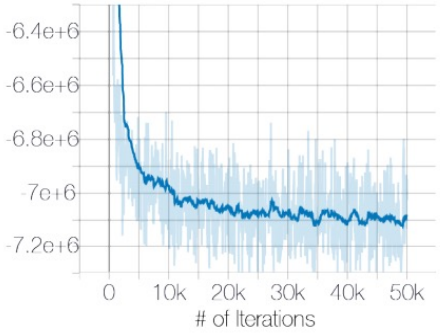
- The solution to all pitfalls: **Gaussian perturbation**
- Manifold + noise



- Score matching on noisy data



CIFAR-10

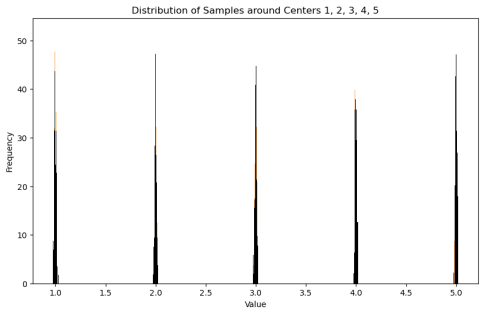


Noisy CIFAR-10

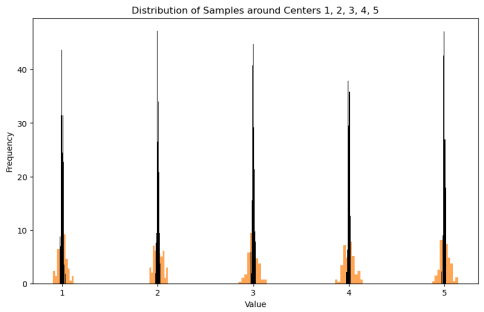
$N(\mathbf{0}, 10^{-4}\boldsymbol{I})$

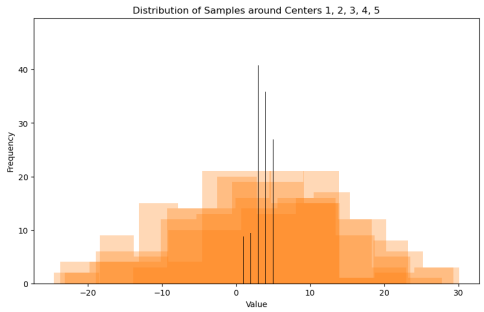
Gaussian perturbation

$$q_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x}) \coloneqq N(\tilde{\boldsymbol{x}}|\boldsymbol{x}, \sigma^2 \boldsymbol{I}), \qquad q_\sigma(\tilde{\boldsymbol{x}}) = \int p_{data}(\boldsymbol{x}) q_\sigma(\tilde{\boldsymbol{x}}|\boldsymbol{x}) d\boldsymbol{x}$$



small σ



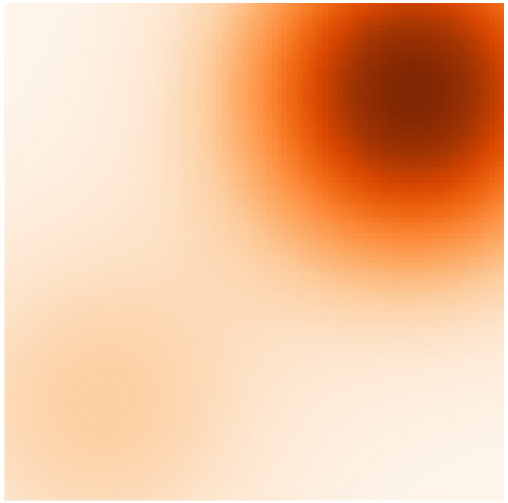


large σ

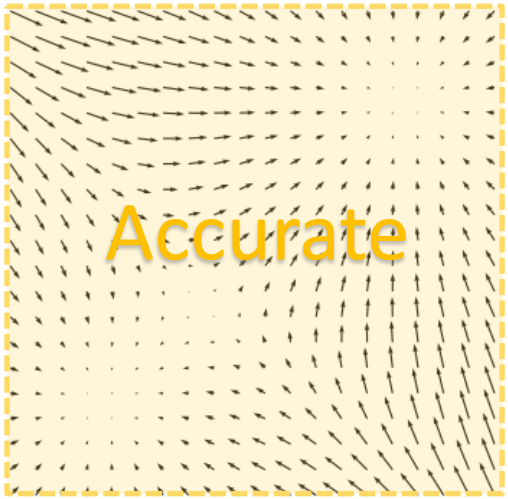
$p_{data}(\boldsymbol{x})$ $q_\sigma(\boldsymbol{x})$

Improving score estimation by adding noise

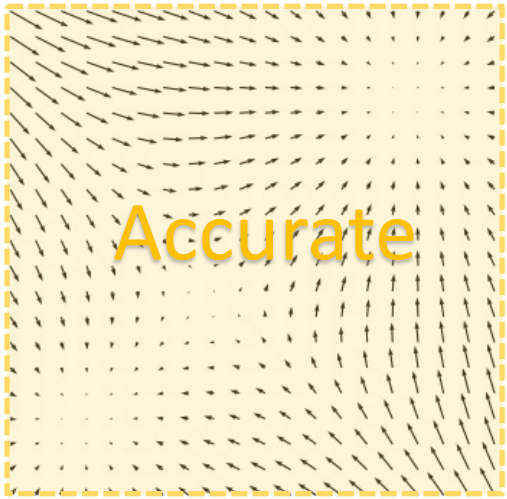
Perturbed density



Perturbed scores



Estimated scores



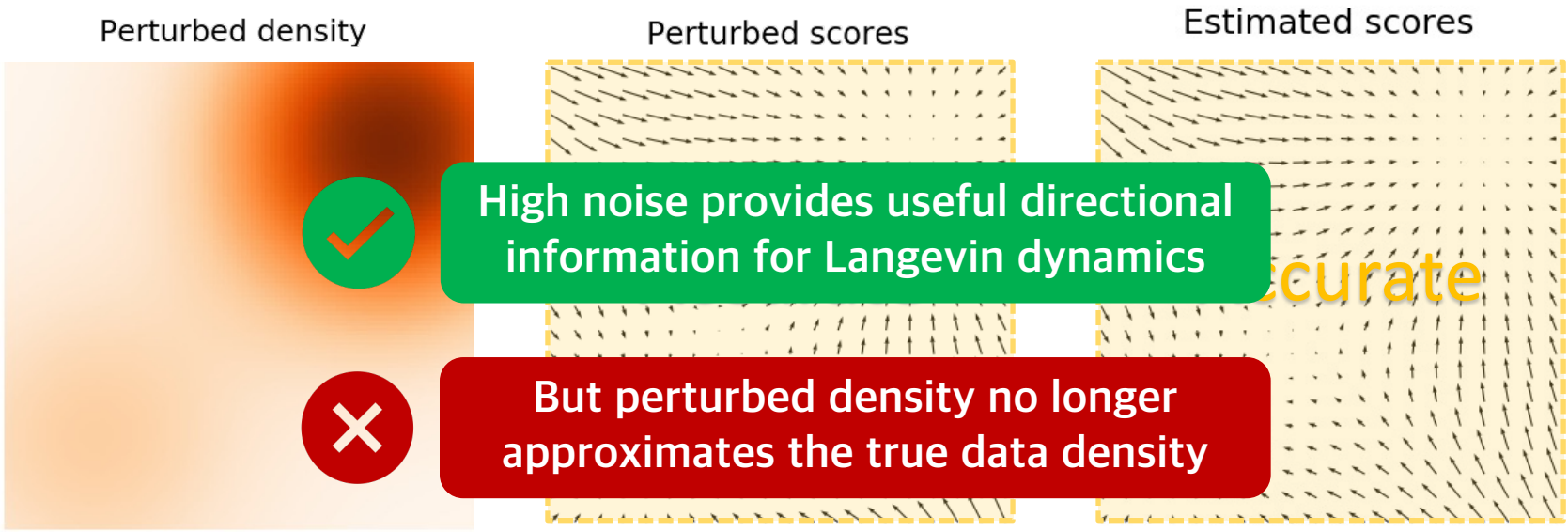
$x \sim p_{data}(x)$
Data distribution

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

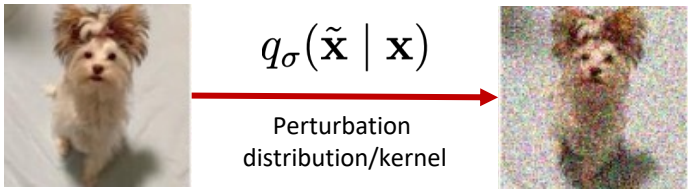
Perturbation distribution/kernel

$\tilde{x} \sim q_\sigma(\tilde{x})$
Noise-perturbed data distribution

Improving score estimation by adding noise



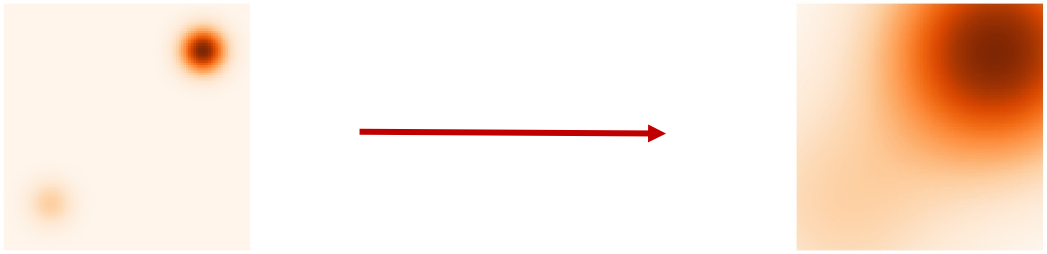
$x \sim p_{data}(x)$
Data distribution



$\tilde{x} \sim q_\sigma(\tilde{x})$
Noise-perturbed data distribution

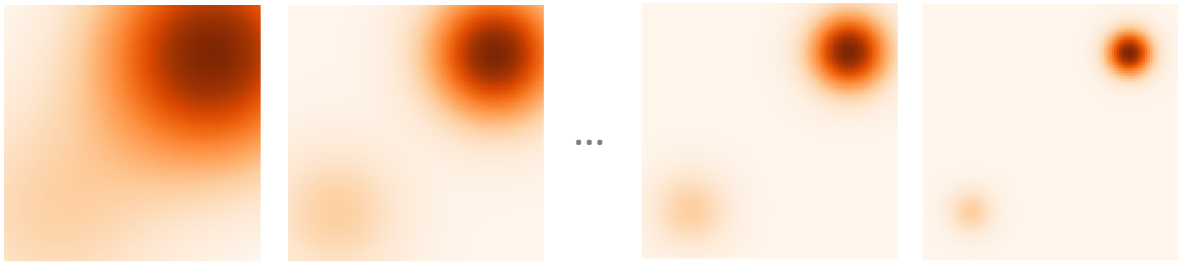
Multi-scale noise perturbation

- How much noise to add?

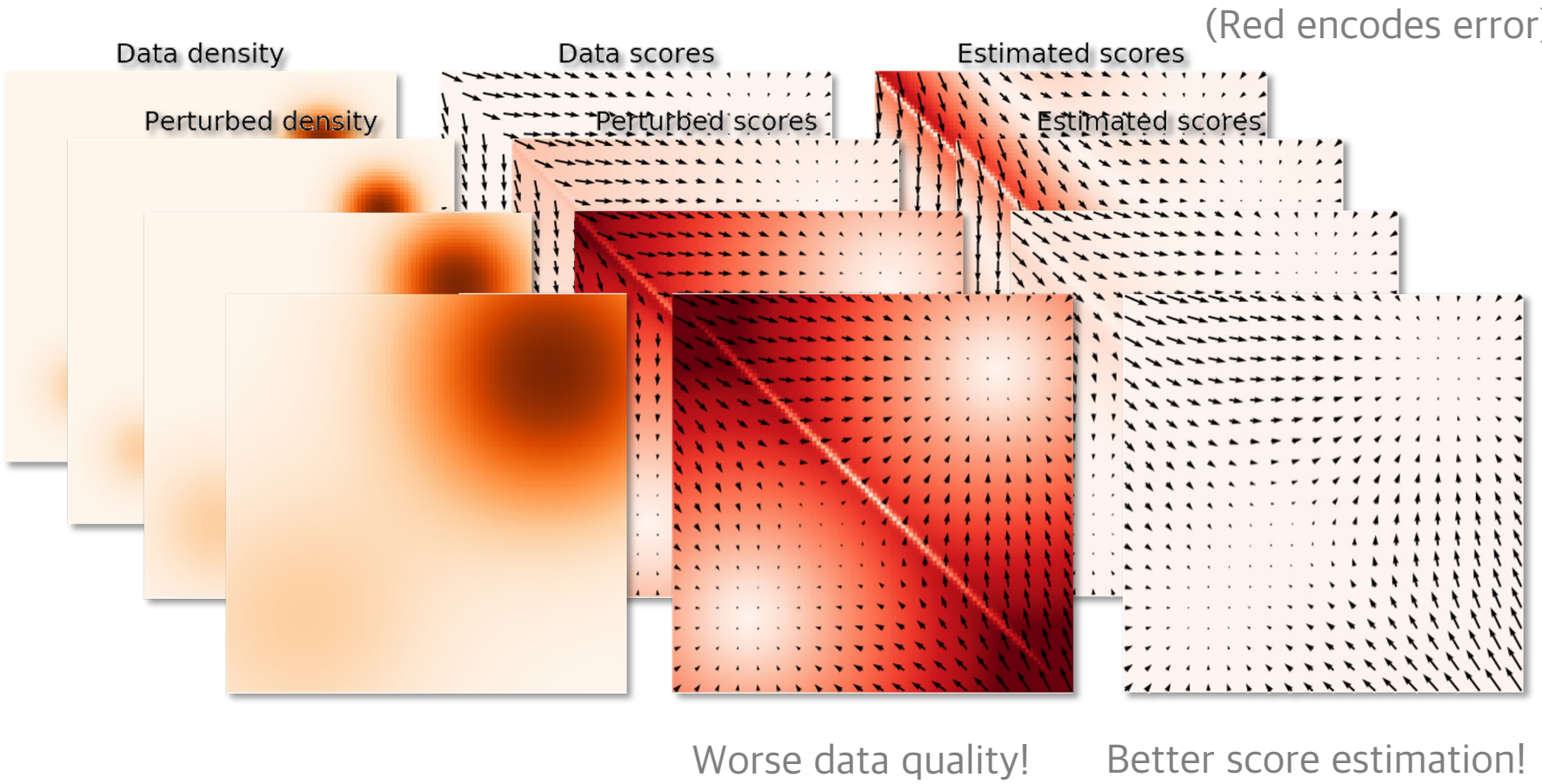


- Multi-scale noise perturbations

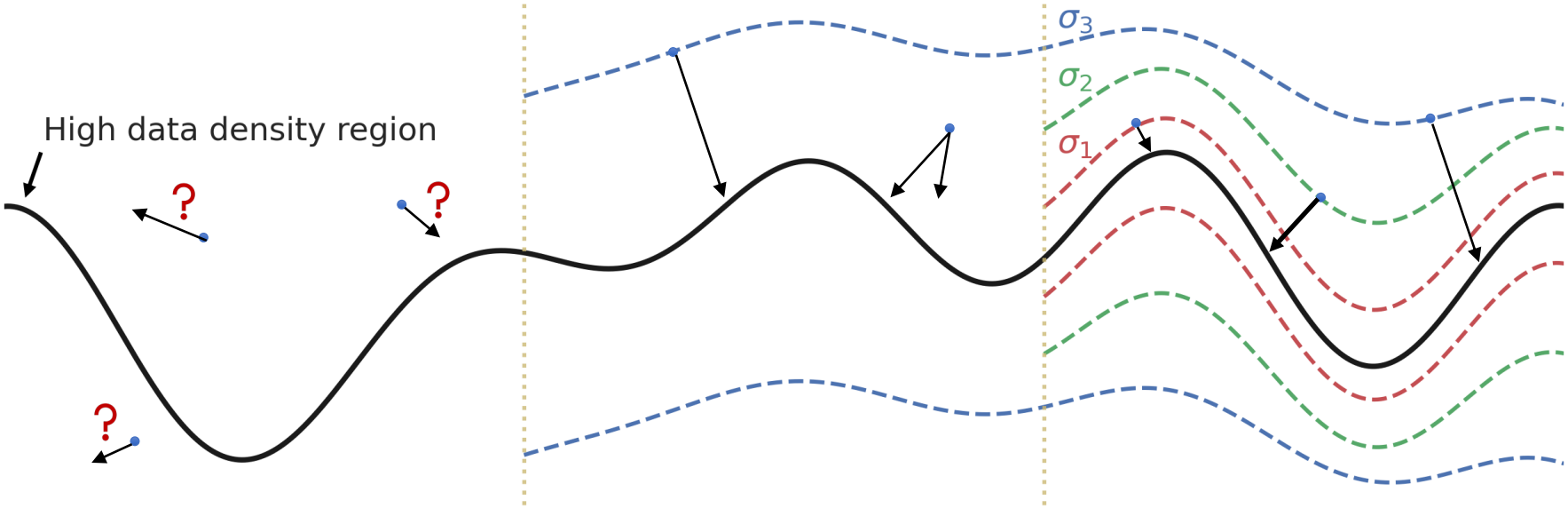
$$\sigma_L > \sigma_{L-1} > \cdots > \sigma_2 > \sigma_1$$



Trading off data quality and estimation accuracy

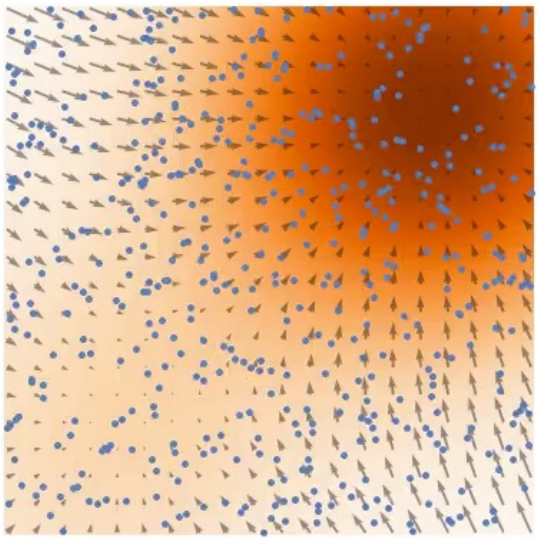


Using multiple noise scales

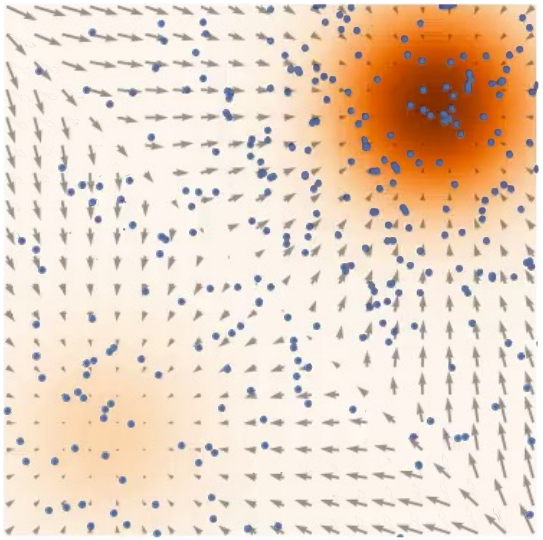


Annealed Langevin dynamics: Joint scores to samples

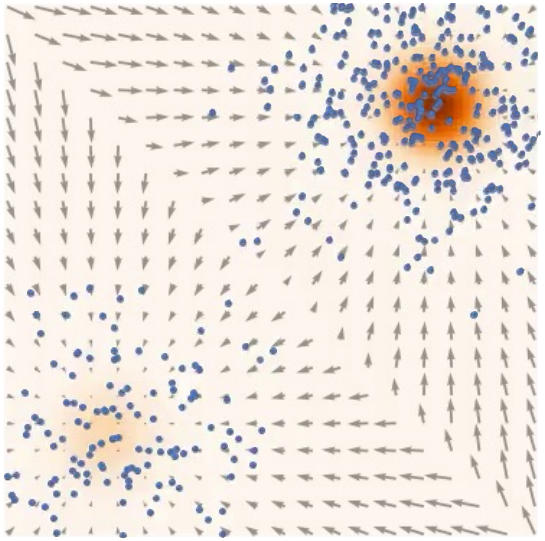
- Sample using $\sigma_L > \sigma_{L-1} > \cdots > \sigma_2 > \sigma_1$ sequentially with Langevin dynamics
- Anneal down the noise level
- Samples used as initialization for the next level



σ_3

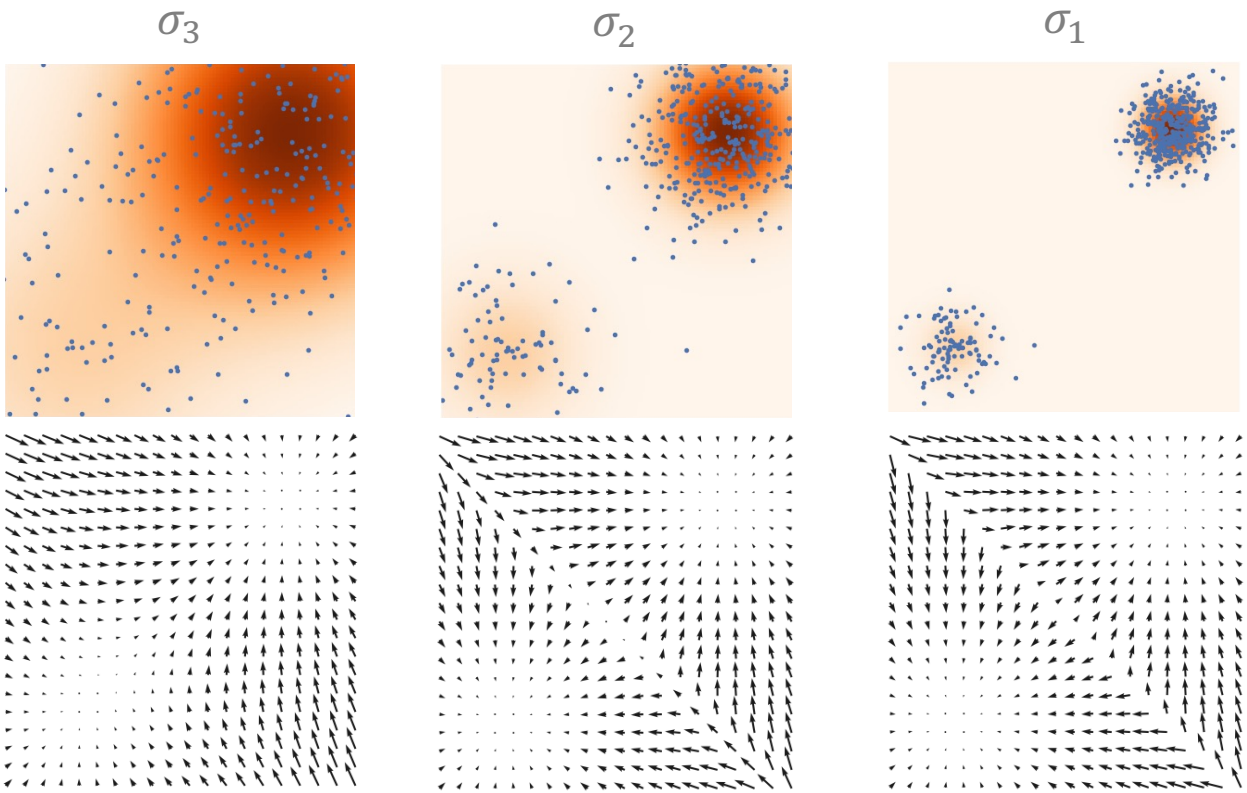


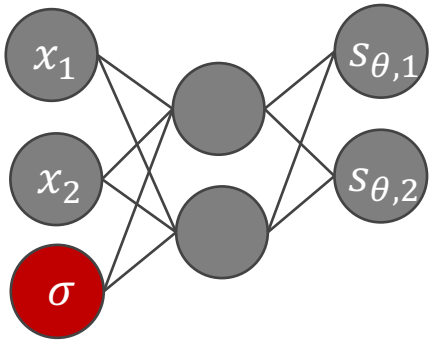
σ_2



σ_1

Joint score estimation via noise conditional score networks





Noise Conditional Score Network (NCSN)

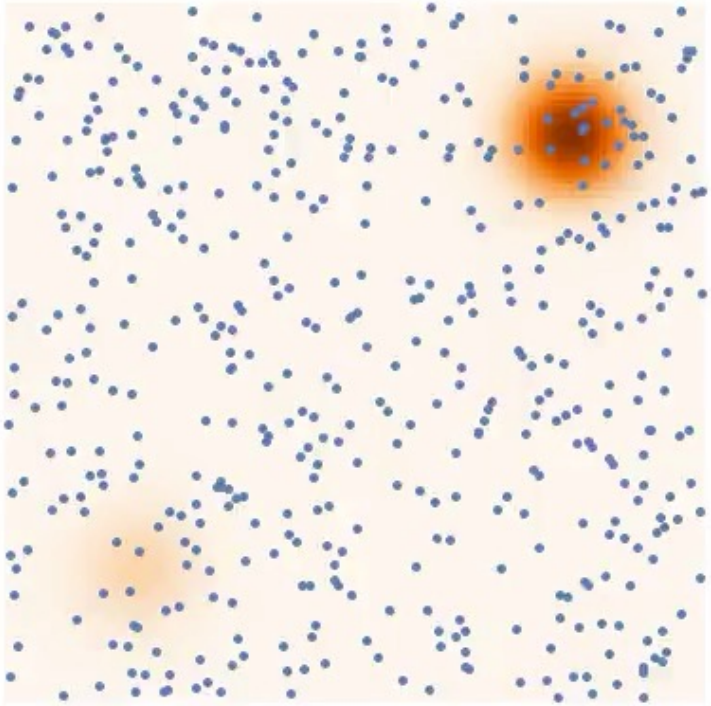
Annealed Langevin dynamics

Algorithm 1 Annealed Langevin dynamics.

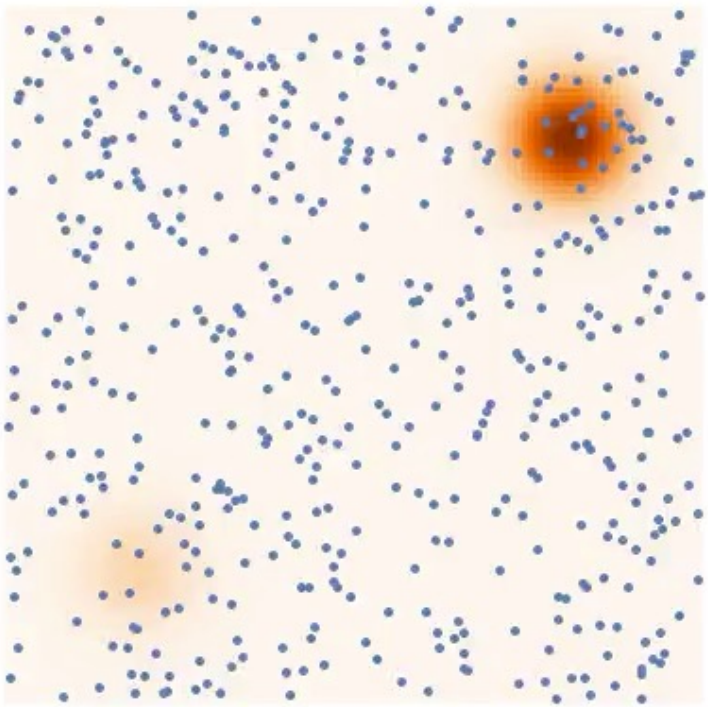
Require: $\{\sigma_i\}_{i=1}^{L}, \epsilon, T$.

Initialize $\tilde{\mathbf{x}}_0$
for $i \leftarrow 1$ to L **do**
  $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2 / \sigma_L^2$ ▷ α_i is the step size.
  for $t \leftarrow 1$ to T **do**
    Draw $\mathbf{z}_t \sim \mathcal{N}(0, I)$
    $\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \frac{\alpha_i}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t-1}, \sigma_i) + \sqrt{\alpha_i}\,\mathbf{z}_t$
  end for
  $\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$
end for
return $\tilde{\mathbf{x}}_T$

Comparison to the vanilla Langevin dynamics

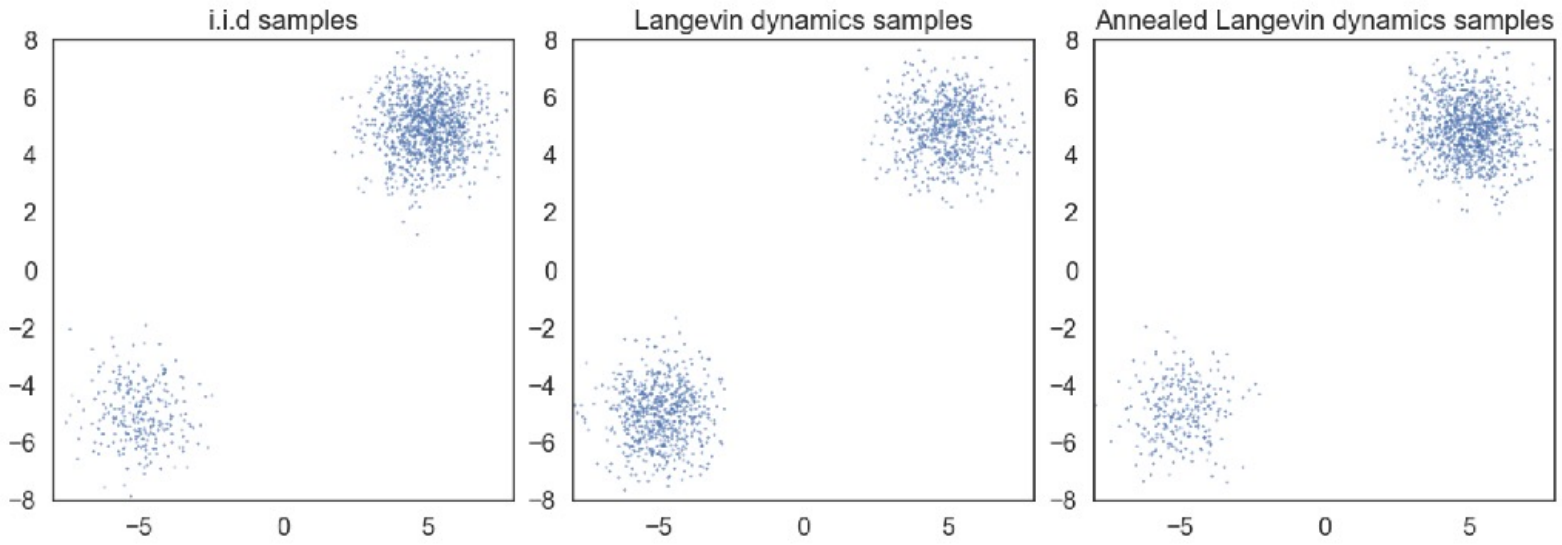


Langevin dynamics



Annealed Langevin dynamics

Comparison to the vanilla Langevin dynamics



Langevin dynamics

Annealed Langevin dynamics

Training noise conditional score networks

- 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

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

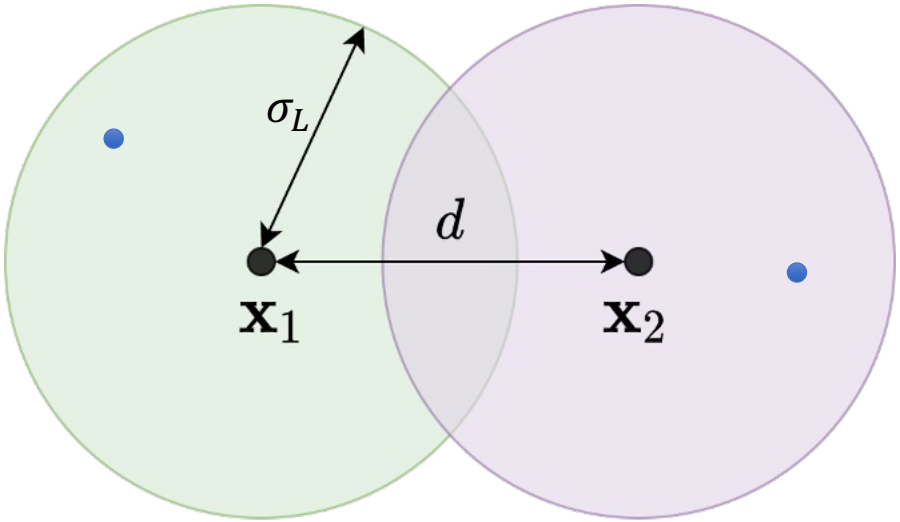
$$=\frac{1}{L}\sum_{i=1}^{L}\lambda(\sigma_i)E_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})}E_{\tilde{\boldsymbol{x}}\sim q_{\sigma_i}(\tilde{\boldsymbol{x}}|\boldsymbol{x})}\left[\left\|\nabla_{\tilde{\boldsymbol{x}}}\log q_{\sigma_i}(\tilde{\boldsymbol{x}}|\boldsymbol{x})-\boldsymbol{s}_\theta(\tilde{\boldsymbol{x}},\sigma_i)\right\|_2^2\right]$$

$$+\text{const.}$$

Choosing noise scales

- **Maximum noise scale**

  $\sigma_L \approx$ maximum pairwise distance between datapoints



- **Minimum noise scale**: σ_1 should be sufficiently small so that noise in final samples is negligible. I.e., $q_{\sigma_1}(\boldsymbol{x}) \approx p_{data}(\boldsymbol{x})$

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

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

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

Choosing the weighting function

- Weighted combination of denoising score matching losses

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

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

- How to choose the weighting function $\lambda: \mathbb{R}_+ \to \mathbb{R}_+$?
- **Goal**: balancing different score matching losses in the sum

Choosing the weighting function

- How to choose the weighting function $\lambda: \mathbb{R}_+ \to \mathbb{R}_+$?
- **Goal**: balancing different score matching losses in the sum
- Since $\sigma_i^2 \propto 1/E\left[\left\|\nabla_{\tilde{\boldsymbol{x}}} \log p_{\sigma_i}(\tilde{\boldsymbol{x}}|\boldsymbol{x})\right\|_2^2\right]$, $\lambda(\sigma_i) := \sigma_i^2$

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

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

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

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

Training noise conditional score networks

- Sample a mini-batch of datapoints $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots, \boldsymbol{x}^{(n)}$ from p_{data}
- Sample a mini-batch of noise scale indices

$$i_1, i_2, \ldots, i_n \sim U(1,2,\cdots,L)$$

- Sample a mini-batch of Gaussian noise $\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \cdots, \boldsymbol{z}^{(n)}$ from $N(\mathbf{0}, \boldsymbol{I})$
- Estimate the weighted mixture of score matching losses

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

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

Generation with annealed Langevin dynamics

- For each $q_{\sigma_i}(\boldsymbol{x})$ with $\sigma_1 < \sigma_2 < \cdots < \sigma_L$, Song & Ermond run T steps of Langevin MCMC to get a sample sequentially

$$\boldsymbol{x}_i^t \coloneqq \boldsymbol{x}_i^{t-1} + \frac{\alpha_i}{2}\boldsymbol{s}_{\theta^*}\left(\boldsymbol{x}_i^{t-1}, \sigma_i\right) + \sqrt{\alpha_i}\boldsymbol{z}, \qquad t = 1,2,\ldots,T$$

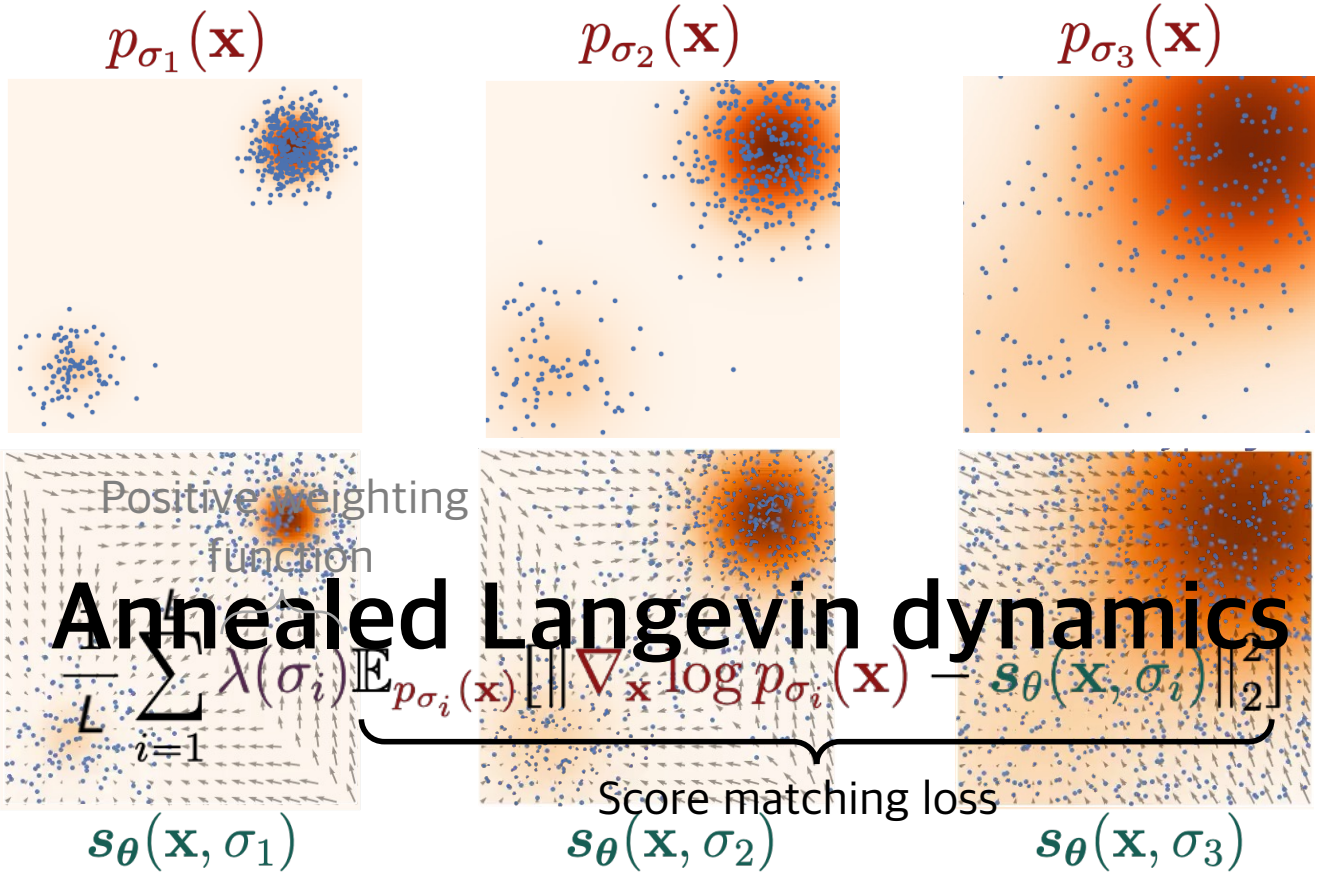
- where $\alpha_i > 0$ is the step size and $\boldsymbol{z} \sim N(\boldsymbol{0}, \boldsymbol{I})$

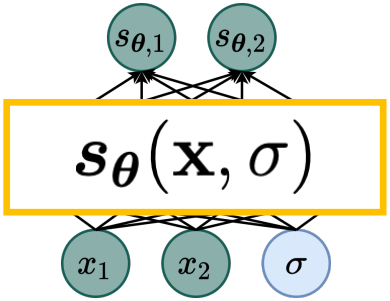
$$\alpha_i \coloneqq \epsilon \frac{\sigma_i^2}{\sigma_1^2}$$

- $\epsilon > 0$

Using multiple noise levels

Data





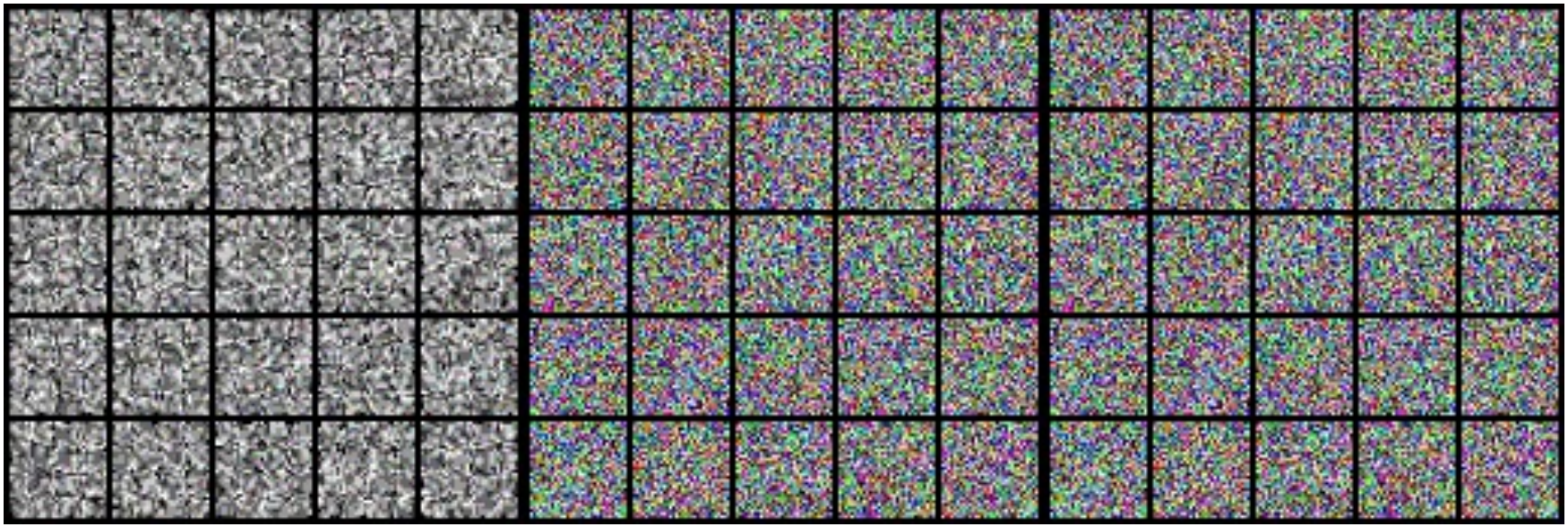
Noise Conditional Score Model

Positive weighting function

Annealed Langevin dynamics

$$\mathcal{L} \triangleq \sum_{i=1}^{L} \lambda(\sigma_i) \underbrace{\mathbb{E}_{p_{\sigma_i}(\mathbf{x})}\left[\left\| \nabla_{\mathbf{x}} \log p_{\sigma_i}(\mathbf{x}) - s_\theta(\mathbf{x}, \sigma_i) \right\|_2^2\right]}_{\text{Score matching loss}}$$

Experiments: sampling



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

Thanks