Bayesian Theory and Computation

Lecture 19: Energy-based and Score-based Generative Models

Cheng Zhang
School of Mathematical Sciences, Peking University

May 14, 2025

How to Parameterize a Distribution?

Probability densities $p(x)$ need to satisfy

- non-negative: $p(x) \geq 0$.
- sum-to-one: $\sum_x p(x) = 1$ or $\int p(x)dx = 1$ for continuous variables

Coming up with a non-negative function $p_\theta(x)$ is not hard

- $p_\theta(x) = f_\theta(x)^2$
- $p_\theta(x) = \exp(f_\theta(x))$
- $p_\theta(x) = |f_\theta(x)|$

Sum to one is the key. Although many models allow analytical integration (e.g., autoregressive models, normalizing flows), what if the analytical integration is not available?

Energy-based Model

$$p_\theta(x) = \frac{\exp(f_\theta(x))}{Z(\theta)}, \quad Z(\theta) = \int \exp(f_\theta(x))dx$$

The normalizing constant $Z(\theta)$ is also called the partition function. Why exponential (and not e.g. $f_\theta(x)^2$)?

- Want to capture very large variations in probability. log-probability is the nature scale we want to work with. Otherwise need highly non-smooth f_θ.
- Exponential families. Many common distributions can be written in this form.
- These distributions arise under fairly general assumptions in statistical physics (maximum entropy, second law of thermodynamics).
  - $f_\theta(x)$ is called the energy, hence the name.
  - Intuitively, configurations x with low energy (high $f_\theta(x)$) are more likely.

Energy-based Model

$$p_\theta(x) = \frac{\exp(f_\theta(x))}{Z(\theta)}, \quad Z(\theta) = \int \exp(f_\theta(x))dx$$

Pros:

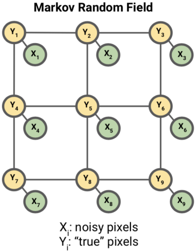
- extreme flexibility. pretty much any function $f_\theta(x)$ you want to use

Cons:

- Samping from $p_\theta(x)$ is hard
- Evaluating and optimizing likelihood $p_\theta(x)$ is hard (learning is hard)
- No feature learning (but can add latent variables)

Curse of dimensionality: The fundamental issue is that computing $Z(\theta)$ numerically (when no analytic solution is available) scales exponentially in the number of dimensions of x.

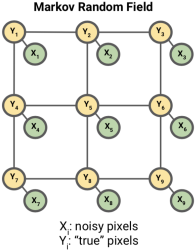
Example: Ising Model



- There is a true image $y \in \{0,1\}^{3\times 3}$, and a corrupted image $x \in \{0,1\}^{3\times 3}$. We know x, and want to somehow recover y.
- We model the joint probability distribution $p(y, x)$ as

$$p(y,x) \propto \exp\left(\sum_i \psi_i(x_i, y_i) + \sum_{i,j\in E} \psi_{i,j}(y_i, y_j)\right)$$

Example: Ising Model



The energy is $\sum_i \psi_i(x_i, y_i) + \sum_{i,j \in E} \psi_{i,j}(y_i, y_j)$

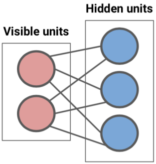
- $\psi_i(x_i, y_i)$: the i-th corrupted pixel depends on the i-th original pixel
- $\psi_{ij}(y_i, y_j)$: neighboring pixels tend to have the same value

How did the original image y look like? Solution: maximize $p(y|x)$. Or equivalently, maximize $p(y, x)$.

Example: Restricted Boltzmann Machine (RBM)

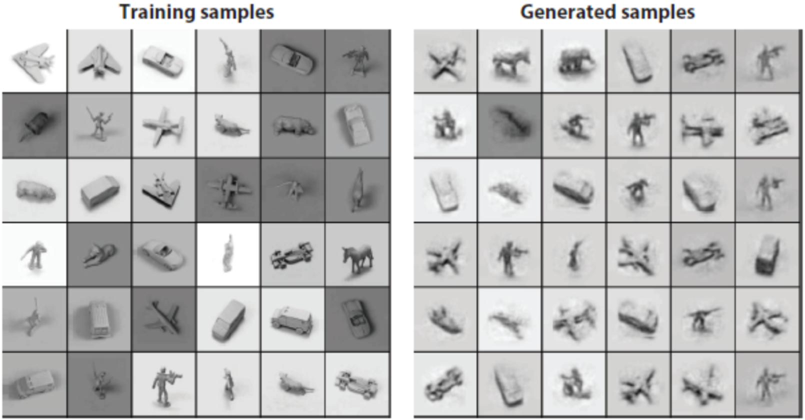
- RBM: energy-based model with latent variables
- Two types of variables:
  - $x \in \{0,1\}^n$ are visible variables (e.g., pixel values)
  - $z \in \{0,1\}^m$ are latent ones
- The joint distribution is

$$p_{W,b,c}(x,z) \propto \exp(x^T W z + b^T x + c^T z)$$



- Restricted as there are no within-class connections.
- Can be stacked together to make deep RBMs (one of the first generative models).

Deep RBMs: Samples



Adapted from Salakhutdinov and Hinton, 2009.

- Learning by maximizing the likelihood function

$$\max_\theta \mathbb{E}_{x\sim p_{\text{data}}} \log p_\theta(x) = \max_\theta \left(\mathbb{E}_{x\sim p_{\text{data}}} f_\theta(x) - \log Z(\theta)\right)$$

- Gradient of log-likelihood:

$$\begin{aligned}
\mathbb{E}_{x\sim p_{\text{data}}} \nabla_\theta f_\theta(x) - \nabla_\theta \log Z(\theta) &= \mathbb{E}_{x\sim p_{\text{data}}} \nabla_\theta f_\theta(x) - \frac{\nabla_\theta Z(\theta)}{Z(\theta)} \\
&= \mathbb{E}_{x\sim p_{\text{data}}} \nabla_\theta f_\theta(x) - \int \frac{\exp(f_\theta(x))}{Z(\theta)} \nabla_\theta f_\theta(x) dx \\
&= \mathbb{E}_{x\sim p_{\text{data}}} \nabla_\theta f_\theta(x) - \int p_\theta(x) \nabla_\theta f_\theta(x) dx \\
&= \mathbb{E}_{x\sim p_{\text{data}}} \nabla_\theta f_\theta(x) - \mathbb{E}_{x\sim p_\theta(x)} \nabla_\theta f_\theta(x)
\end{aligned}$$

- **Contrastive Divergence**: sample $x_{\text{sample}} \sim p_\theta$, take gradient step on $\nabla_\theta f_\theta(x_{\text{train}}) - \nabla_\theta f_\theta(x_{\text{sample}})$.

Sampling From EBMs

$$p_\theta(x) = \frac{\exp(f_\theta(x))}{Z(\theta)}, \quad Z(\theta) = \int \exp(f_\theta(x))dx$$

- No direct way to sample like in autoregressive or flow models.
- Can use gradient-based MCMC methods, e.g., SGLD

$$x^{t+1} = x^t + \epsilon \nabla_x \log p_\theta(x^t) + \sqrt{2\epsilon}\eta^t, \quad \eta^t \sim \mathcal{N}(0, I)$$

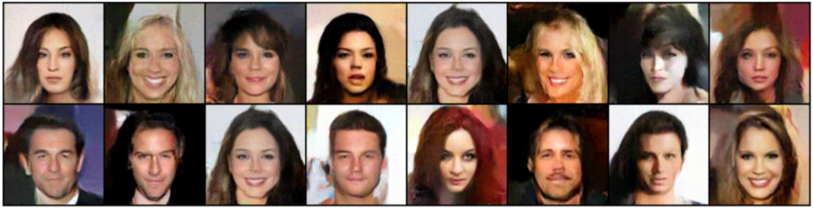
- Note that for energy-based models

$$s_\theta(x) = \nabla_x \log p_\theta(x) = \nabla_x f_\theta(x) - \nabla_x \log Z(\theta) = \nabla_x f_\theta(x)$$

The score function does not depend on $Z(\theta)$!

Modern EBMs

Langevin sampling



Face samples

Adapted from Nijkamp et al. 2019

Training Without Sampling

$$x^{t+1} = x^t + \epsilon \nabla_x \log p_\theta(x^t) + \sqrt{2\epsilon}\eta^t, \quad \eta^t \sim \mathcal{N}(0, I)$$

- MCMC sampling converges slowly in high dimensional spaces, and repetitive sampling for each training iteration would be expensive.
- Can we train without sampling?
- Note that to generate samples from an EBM, we only need the score function $\nabla_x \log p_\theta(x)$.
- Can we properly train the score function without sampling?

- A key observation: two distributions are identical iff their scores are the same

$$p(x) = q(x) \Leftrightarrow \nabla_x \log \tilde{p}(x) = \nabla_x \log \tilde{q}(x)$$

where $\tilde{p}, \tilde{q}$ are the unnormalized densities of p, q.

- Match the scores of the data distribution and EBMs by minimizing

$$\begin{aligned} &\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}\|\nabla_x \log p_{\text{data}}(x) - s_\theta(x)\|^2 \\ =&\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}\|\nabla_x \log p_{\text{data}}(x) - \nabla_x f_\theta(x)\|^2 \end{aligned}$$

This is also known as Fisher divergence.

- Using integration by parts, we have

$$
\begin{aligned}
&\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}\|\nabla_x \log p_{\text{data}}(x) - \nabla_x f_\theta(x)\|^2 \\
=&\mathbb{E}_{x\sim p_{\text{data}}}\left(\operatorname{Tr}\left(\nabla_x^2 f_\theta(x)\right) + \frac{1}{2}\|\nabla_x f_\theta(x)\|^2\right) + \text{Const}
\end{aligned}
$$

- Sample a mini-batch of datapoints

$$
\{x_1, x_2, \ldots, x_n\} \sim p_{\text{data}}(x)
$$

- Estimate the score matching loss with the empirical mean

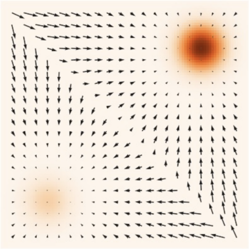
$$
\frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{2}\|\nabla_x f_\theta(x_i)\|^2 + \operatorname{Tr}\left(\nabla_x^2 f_\theta(x_i)\right)\right)
$$

- Minimize the score matching loss via stochastic gradient descent.
- No need to sample from the EBM!
- Note that computing the trace of Hessian $\text{Tr}\left(\nabla_x^2 f_\theta(x)\right)$ is in general very expensive for large models.
- Scalable score matching methods: denoising score matching (Vincent 2010) and sliced score matching (Song et al. 2019).

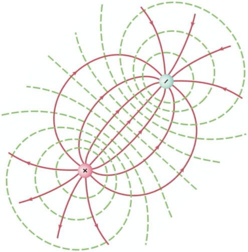
Score-based Models

- When the pdf is differentiable, we can compute the gradient of a probability density, and use it to represent the distribution.

Score function $\nabla_x \log p(x)$



(pdf and score)



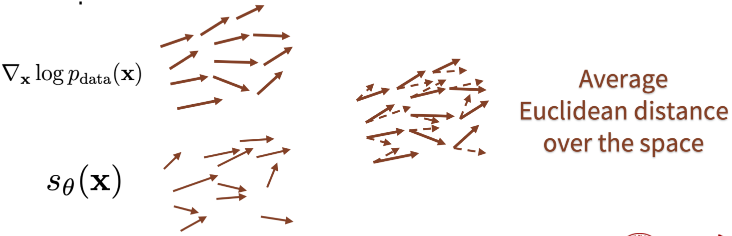
(Electrical potentials and fields)

How to Train Score-based Models

- Given i.i.d. samples $\{x_1, \ldots, x_N\} \sim p(x)$
- We want to estimate the score $\nabla_x \log p_{\text{data}}(x)$
- Score model: a learnable vector-valued function

$$s_\theta(x) : \mathbb{R}^D \to \mathbb{R}^D$$

- Goal: $s_\theta(x) \approx \nabla_x \log p_{\text{data}}(x)$
- How to compare two vector fields of scores?



- Objective: Average Euclidean distance over the whole space.

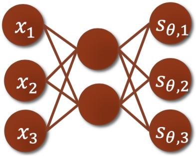
$$\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}\|\nabla_x \log p_{\text{data}}(x) - s_\theta(x)\|^2$$

- **Score matching**:

$$\mathbb{E}_{x\sim p_{\text{data}}}\left(\frac{1}{2}\|s_\theta(x)\|^2 + \text{Tr}(\nabla_x s_\theta(x))\right)$$

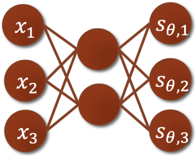
- Requirements:
  - The score model must be efficient to evaluated.
  - Do we need the score model to be a proper score function?

- We can use deep neural networks for more expressive score models



Score Matching is Not Scalable

- We can use deep neural networks for more expressive score models



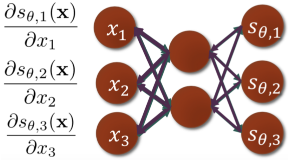
- However, $\mathrm{Tr}(\nabla_x s_\theta(x))$ can be a problem.

$O(D)$ Backprops!

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

Score Matching is Not Scalable

- We can use deep neural networks for more expressive score models

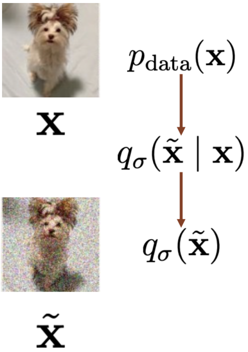


- However, $\mathrm{Tr}(\nabla_x s_\theta(x))$ can be a problem.

$O(D)$ 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



- Denoising score matching (Vincent 2011) used a noise-perturbed data distribution

$$
\begin{aligned}
&\frac{1}{2}\mathbb{E}_{\tilde{x}\sim q_\sigma}\|\nabla_{\tilde{x}}\log q_\sigma(\tilde{x}) - s_\theta(\tilde{x})\|^2 \\
=&\frac{1}{2}\int q_\sigma(\tilde{x})\|\nabla_{\tilde{x}}\log q_\sigma(\tilde{x}) - s_\theta(\tilde{x})\|^2 d\tilde{x} \\
=&\frac{1}{2}\int q_\sigma(\tilde{x})\|s_\theta(\tilde{x})\|^2 d\tilde{x} \\
&-\int q_\sigma(\tilde{x})\nabla_{\tilde{x}}\log q_\sigma(\tilde{x})^T s_\theta(\tilde{x}) d\tilde{x} + \text{Const}
\end{aligned}
$$

- The second term can be rewritten as

$$
\begin{aligned}
-\int q_\sigma(\tilde{x}) \nabla_{\tilde{x}} \log q_\sigma(\tilde{x})^T s_\theta(\tilde{x}) d\tilde{x} &= -\int \nabla_{\tilde{x}} q_\sigma(\tilde{x})^T s_\theta(\tilde{x}) d\tilde{x} \\
&= -\int \nabla_{\tilde{x}} \left(\int p_{\text{data}}(x) q_\sigma(\tilde{x}|x) dx \right)^T s_\theta(\tilde{x}) d\tilde{x} \\
&= -\int \left(\int p_{\text{data}}(x) \nabla_{\tilde{x}} q_\sigma(\tilde{x}|x) dx \right)^T s_\theta(\tilde{x}) d\tilde{x} \\
&= -\int \int p_{\text{data}}(x) q_\sigma(\tilde{x}|x) \nabla_{\tilde{x}} \log q_\sigma(\tilde{x}|x)^T s_\theta(\tilde{x}) dx d\tilde{x} \\
&= -\mathbb{E}_{x \sim p_{\text{data}}(x), \tilde{x} \sim q_\sigma(\tilde{x}|x)} \nabla_{\tilde{x}} \log q_\sigma(\tilde{x}|x)^T s_\theta(\tilde{x})
\end{aligned}
$$

- Plug it back we have

$$
\begin{aligned}
&\frac{1}{2}\mathbb{E}_{\tilde{x}\sim q_\sigma}\|\nabla_{\tilde{x}}\log q_\sigma(\tilde{x}) - s_\theta(\tilde{x})\|^2 \\
=&\frac{1}{2}\mathbb{E}_{x\sim p_{\mathrm{data}}(x),\tilde{x}\sim q_\sigma(\tilde{x}|x)}\|s_\theta(\tilde{x}) - \nabla_{\tilde{x}}\log q_\sigma(\tilde{x}|x)\|^2 + \mathrm{Const}
\end{aligned}
$$

- The noise score $\nabla_{\tilde{x}}\log q_\sigma(\tilde{x}|x)$ is easy to compute. For example, when use Gaussian noise $q_\sigma(\tilde{x}|x) = \mathcal{N}(\tilde{x}|x, \sigma^2 I)$, the score is

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

- Pros: efficient to optimize even for very high dimensional data, and useful for optimal denoising.
- Cons: cannot estimate the score of clean data (noise-free)

- Sample a minibatch of datapoints $\{x_1, \ldots, x_n\} \sim p_{\text{data}}(x)$.
- Sample a minibatch of perturbed datapoints

$$\tilde{x}_i \sim q_\sigma(\tilde{x}_i | x_i), \quad i = 1, 2, \ldots, n$$

- Estimate the denoising score matching loss with empirical means

$$\frac{1}{2n} \sum_{i=1}^{n} \| s_\theta(\tilde{x}) - \nabla_{\tilde{x}} \log q_\sigma(\tilde{x}_i | x) \|^2$$

- Stochastic gradient descent
- Need to choose a very small σ! However, the loss variance would also increase drastically as $\sigma \to 0$!

- Denoising score matching is suitable for optimal denoising
- Given $p(x)$, $q_\sigma(\tilde{x}|x) = \mathcal{N}(\tilde{x}|x, \sigma^2 I)$, we can define the posterior $p(x|\tilde{x})$ with Bayes' rule

$$p(x|\tilde{x}) = \frac{p(x)q_\sigma(\tilde{x}|x)}{q_\sigma(\tilde{x})}$$

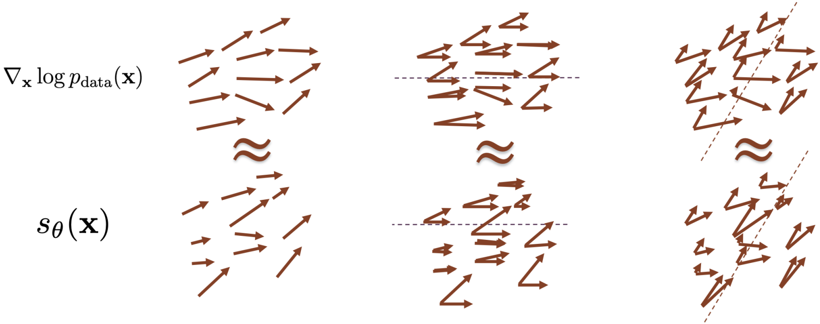
where

$$q_\sigma(\tilde{x}) = \int p(x)q_\sigma(\tilde{x}|x)dx$$

- Tweedie's formula:

$$\begin{aligned}\mathbb{E}_{x\sim p(x|\tilde{x})}[x] &= \tilde{x} + \sigma^2 \nabla_{\tilde{x}} \log q_\sigma(\tilde{x}) \\ &\approx \tilde{x} + \sigma^2 s_\theta(\tilde{x})\end{aligned}$$

- One dimensional problems should be easier.
- Consider projections onto random directions.
- Sliced score matching (Song et al 2019).



- Objective: Sliced Fisher Divergence

$$\frac{1}{2}\mathbb{E}_{v\sim p_v}\mathbb{E}_{x\sim p_{\text{data}}}\left(v^T\nabla_x \log p_{\text{data}}(x) - v^T s_\theta(x)\right)^2$$

- Similarly, we can do integration by parts

$$\mathbb{E}_{v\sim p_v}\mathbb{E}_{x\sim p_{\text{data}}}\left(v^T\nabla_x s_\theta(x)v + \frac{1}{2}(v^T s_\theta(x))^2\right)$$

- Computing Jacobian-vector products is scalable

$$v^T\nabla_x s_\theta(x)v = v^T\nabla_x(s_\theta(x)^T v)$$

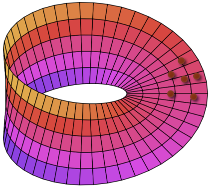
This only requires one backpropagation!

- Sample a minibatch of datapoints $\{x_1, \ldots, x_n\} \sim p_{\text{data}}(x)$
- Sample a minibatch of projection directions $\{v_i \sim p_v\}_{i=1}^n$
- Estimate the sliced score matching loss with empirical means

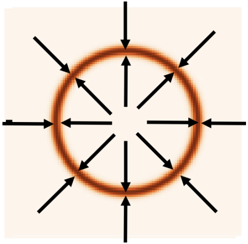
$$\frac{1}{n}\sum_{i=1}^{n}\left(v_i^T \nabla_x s_\theta(x_i) v_i + \frac{1}{2}(v_i^T s_\theta(x_i))^2\right)$$

- The perturbation distribution is typically Gaussian or Rademacher. When $\mathbb{E}vv^T = I$, this is equivalent to the Hutchinson's trick.
- Can use $\|s_\theta(x)\|^2$ instead of $(v^T s_\theta(x))^2$ to reduce variance.
- Can use more projections per datapoint to boost performance.

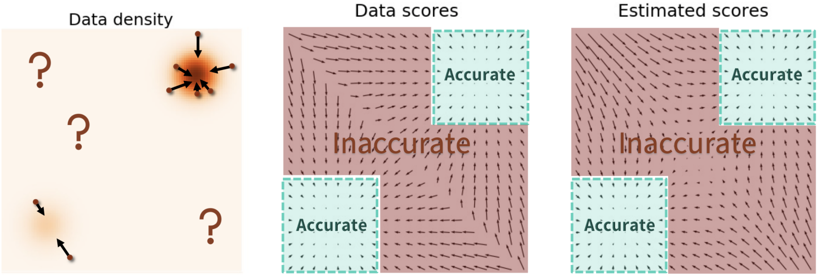
- Datapoints would lie on a lower dimensional manifold.



- Data score hence would be undefined.

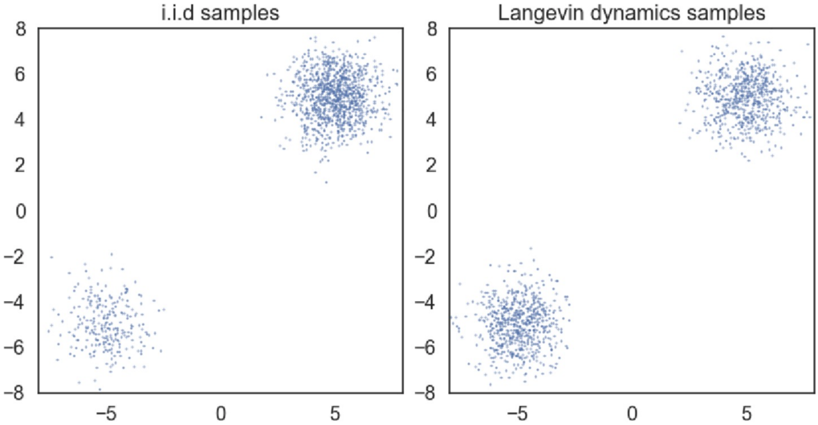


$$\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}\|\nabla_x \log p_{\text{data}}(x) - s_\theta(x)\|^2$$



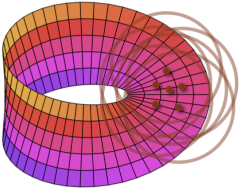
- Poor score estimation in low data density regions.
- Langevin MCMC will also have trouble exploring low density regions.

Pitfalls: Slow Mixing Between Data Modes

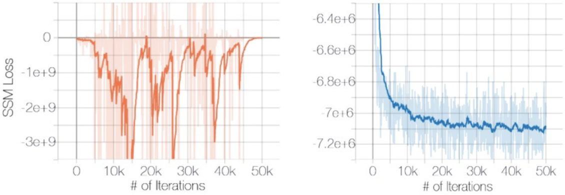


Adapted from Song et al 2019.

- The solution to all pitfalls: Gaussian perturbation!
- Inflate the flat manifold with noise.



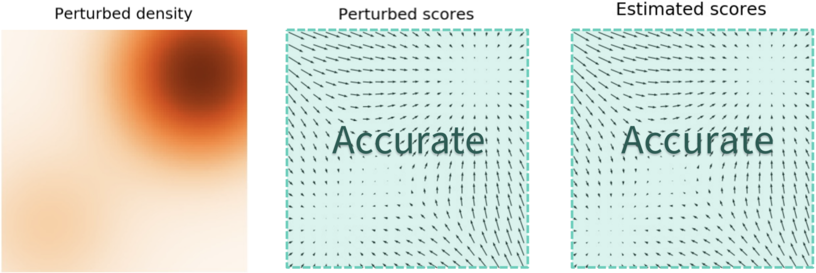
- Score matching on noise data



CIFAR-10 Noisy CIFAR-10

Noisy Data Score Estimation

$$\frac{1}{2}\mathbb{E}_{\tilde{x}\sim q_\sigma}\|\nabla_{\tilde{x}}\log q_\sigma(\tilde{x}) - s_\theta(\tilde{x})\|^2$$

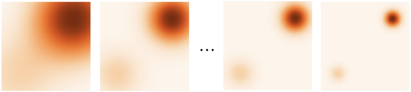


- Noisy score can provide useful directional information for Langevin MCMC.

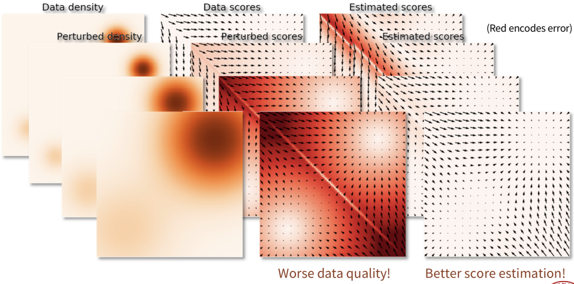
Multi-scale Noise Perturbation

- Multi-scale noise perturbations.

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

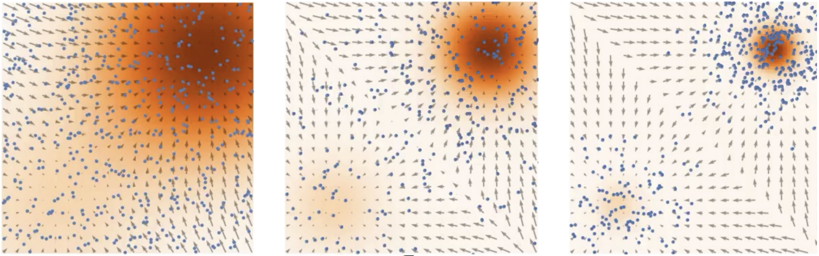


- Trading off data quality and estimator accuracy



Annealed Langevin Dynamics

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



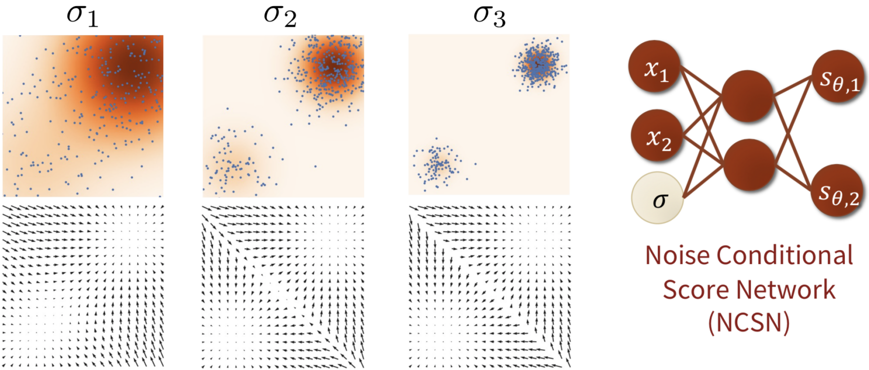
σ_1 σ_2 σ_3

Algorithm 1 Annealed Langevin dynamics.

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

Noise Conditional Score Networks

- Learning score functions jointly with noise conditional score networks!



- As the goal is to estimate the score of perturbed data distributions, we can use denoising score matching for training.
- Assign different weights to combine denoising score matching losses for different noise levels.

$$\frac{1}{L}\sum_{i=1}^{L}\lambda(\sigma_i)\mathbb{E}_{\tilde{x}\sim q_{\sigma_i}(\tilde{x})}\|\nabla_{\tilde{x}}\log q_{\sigma_i}(\tilde{x}) - s_\theta(\tilde{x},\sigma_i)\|^2$$

$$= \frac{1}{L}\sum_{i=1}^{L}\lambda(\sigma_i)\mathbb{E}_{x\sim p_{\text{data}},\tilde{x}\sim q_{\sigma_i}(\tilde{x}|x)}\|\nabla_x \log q_{\sigma_i}(\tilde{x}|x) - s_\theta(\tilde{x},\sigma_i)\|^2 + \text{Const}$$

$$= \frac{1}{L}\sum_{i=1}^{L}\lambda(\sigma_i)\mathbb{E}_{x\sim p_{\text{data}},z\sim\mathcal{N}(0,I)}\|s_\theta(x+\sigma_i z,\sigma_i) + \frac{z}{\sigma_i}\|^2 + \text{Const.}$$

Noise Scales and Weighting Functions

- Adjacent noise scales should have sufficient overlap to ease transitioning across noise scales in annealed Langevin dynamics.
- For example, a geometric progression

$$\frac{\sigma_i}{\sigma_{i+1}} = \alpha > 1, \quad i = 1, \ldots, L-1$$

- What about the weighting function λ?
- Use $\lambda(\sigma) = \sigma^2$ to balance different score matching losses

$$\begin{aligned}
&\frac{1}{L}\sum_{i=1}^{L} \sigma_i^2 \mathbb{E}_{x\sim p_{\text{data}}, z\sim\mathcal{N}(0,I)} \|s_\theta(x+\sigma_i z, \sigma_i) + \frac{z}{\sigma_i}\|^2 \\
=&\frac{1}{L}\sum_{i=1}^{L} \mathbb{E}_{x\sim p_{\text{data}}, z\sim\mathcal{N}(0,I)} \|\sigma_i s_\theta(x+\sigma_i z, \sigma_i) + z\|^2
\end{aligned}$$

- Sample a mini-batch of datapoints $\{x_1, \ldots, x_n\} \sim p_{\text{data}}$.
- Sample a mini-batch of noise scale indices

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

- Sample a mini-batch of Gaussian noise

$$\{z_1, \ldots, z_n\} \sim \mathcal{N}(0, I)$$

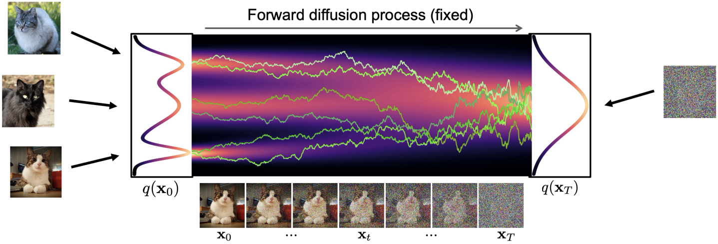
- Estimate the weighted mixture of score matching losses

$$\frac{1}{n} \sum_{k=1}^{n} \|\sigma_{i_k} s_\theta(x_k + \sigma_{i_k} z_k, \sigma_{i_k}) + z_k\|^2$$

- As efficient as training one single non-conditional score-based model.

A Continuous Version via SDEs

Consider the case of infinitely many noise levels

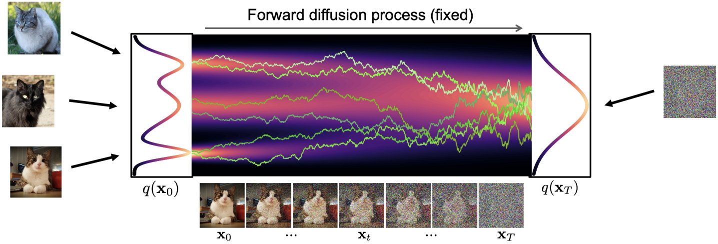


Forward diffusion SDE: $dX_t = f(X_t, t)dt + g(t)dB_t.$

Examples:

- Variance Exploding: $f(X_t, t) = 0, \quad g(t) = \sqrt{\frac{d\sigma_t^2}{dt}}.$
- Variance Preserving: $f(X_t, t) = -X_t, \quad g(t) = \sqrt{2}.$

The Generative Reverse SDE



- Forward diffusion SDE:

$$d\bar{X}_t = f(\bar{X}_t, t)dt + g(t)dB_t, \quad \bar{X}_t \sim q_t.$$

- Reverse diffusion SDE: let $\bar{X}_t^{\leftarrow} := \bar{X}_{T-t}, 0 \le t \le T$

$$d\bar{X}_t^{\leftarrow} = (g(T-t)^2 \nabla \log q_{T-t}(\bar{X}_t^{\leftarrow}) - f(\bar{X}_t^{\leftarrow}, T-t))dt + g(T-t)dB_t.$$

A Concrete Example via OU Process

- Let q be the data distribution. Consider the OU forward process:
$$d\bar{X}_t = -\bar{X}_t dt + \sqrt{2} dB_t, \quad q_0 \sim q.$$
- The condition distribution is
$$\bar{X}_t | \bar{X}_0 \sim \mathcal{N}(e^{-t}\bar{X}_0, (1-e^{-2t}) I_d).$$
- The corresponding reverse process is
$$d\bar{X}_t^{\leftarrow} = (\bar{X}_t^{\leftarrow} + 2\nabla \log q_{T-t}(\bar{X}_t^{\leftarrow}))dt + \sqrt{2} dB_t.$$
where q_t is the law of the forward process.
- Denoising score matching:
$$\min_s \mathbb{E}_{\bar{x}_0 \sim q, \bar{x}_t \sim q(\bar{x}_t|\bar{x}_0)} \|s_t(\bar{x}_t) - \nabla_{\bar{x}_t} \log q(\bar{x}_t|\bar{x}_0)\|^2.$$

Estimated Reverse SDE and Discretization

- Reverse SDE with estimated score

$$d\bar{X}_t^{\leftarrow} = (\bar{X}_t^{\leftarrow} + 2s_{T-t}(\bar{X}_t^{\leftarrow}))dt + \sqrt{2}dB_t.$$

- Let $h > 0$ be the step size. Assume that we have score estimates s_{kh} for each time $k = 0, 1, \ldots, N$, where $T = Nh$.
- Discretize the reverse SDE using an exponential integrator

$$d\bar{X}_t^{\leftarrow} = (\bar{X}_t^{\leftarrow} + 2s_{T-kh}(\bar{X}_{kh}^{\leftarrow}))dt + \sqrt{2}dB_t, \quad t \in [kh, (k+1)h]$$

- How well can the data distribution be approximated if the score estimation is accurate enough?

Main Theorem

Assumptions:

- A1:$\forall t \geq 0$, the score function $\nabla \log q_t$ L-Lipschitz.
- A2: For some $\eta > 0$, $\mathbb{E}_q \| \cdot \|^{2+\eta}$ is finite, and

$$m_2^2 := \mathbb{E}_q \| \cdot \|^2.$$

- A3: For all $k = 1, N$, $\mathbb{E}_{q_{kh}} \| s_{kh} - \nabla \log q_{kh} \|^2 \leq \epsilon^2$.

Theorem (Chen et al., 2023)

Suppose A1-3 hold. Let p_T be the output of the discretized reverse SDE at time T with $\bar{X}_0^{\leftarrow} \sim \gamma^d$, and suppose $h \lesssim 1/L$, where $L \geq 1$. Then it holds that

$$\mathrm{TV}(p_T, q) \lesssim \sqrt{\mathrm{KL}(q \| \gamma^d)} \exp(-T) + (L\sqrt{dh} + L m_2 h)\sqrt{T} + \epsilon\sqrt{T}$$

Proof

- Let $Q_T^{\leftarrow}$ be the path measure of the exact reverse process

$$d\bar{X}_t^{\leftarrow} = (\bar{X}_t^{\leftarrow} + 2\nabla \log q_{T-t}(\bar{X}_t^{\leftarrow}))dt + \sqrt{2}dB_t.$$

- Let $P_T^{q_T}$ be the path measure of the approximated reverse process

$$d\bar{X}_t^{\leftarrow} = (\bar{X}_t^{\leftarrow} + 2s_{T-kh}(\bar{X}_{kh}^{\leftarrow}))dt + \sqrt{2}dB_t, \quad t \in [kh, (k+1)h]$$

- Girsanov's theorem: a more general case

$$\mathrm{KL}(Q_T^{\leftarrow} \| P_T^{q_T}) \leq \sum_{k=0}^{N-1} \mathbb{E}_{Q_T^{\leftarrow}} \int_{kh}^{(k+1)h} \|s_{T-kh}(X_{kh}) - \nabla \log q_{T-t}(X_t)\|^2 dt.$$

- Bounding the discretization error. $\forall t \in [kh, (k+1)h]$

$$
\begin{aligned}
&\mathbb{E}_{Q_T^{\leftarrow}} \|s_{T-kh}(X_{kh}) - \nabla \log q_{T-t}(X_t)\|^2 \\
&\quad \lesssim \mathbb{E}_{Q_T^{\leftarrow}} \big(\|s_{T-kh}(X_{kh}) - \nabla \log q_{T-kh}(X_{kh})\|^2 \\
&\qquad\qquad + \|\nabla \log q_{T-kh}(X_{kh}) - \nabla \log q_{T-t}(X_{kh})\|^2 \\
&\qquad\qquad + \|\nabla \log q_{T-t}(X_{kh}) - \nabla \log q_{T-t}(X_t)\|^2\big) \\
&\quad \lesssim \epsilon^2 + \mathbb{E}_{Q_T^{\leftarrow}} \left\|\nabla \log \frac{q_{T-kh}}{q_{T-t}}(X_{kh})\right\|^2 + L^2\, \mathbb{E}_{Q_T^{\leftarrow}} \|X_{kh} - X_t\|^2.
\end{aligned}
$$

Proof

- Bounding the change of score along the forward process

$$\left\|\nabla \log \frac{q_{T-kh}}{q_{T-t}}(X_{kh})\right\|^2 \lesssim L^2 dh + L^2h^2\|X_{kh}\|^2 + L^2h^2\|\nabla \log q_{T-t}(X_{kh})\|^2.$$

- For the last term

$$\begin{aligned}\|\nabla \log q_{T-t}(X_{kh})\|^2 \lesssim & \|\nabla \log q_{T-t}(X_t)\|^2 + \\ & \|\nabla \log q_{T-t}(X_{kh}) - \nabla \log q_{T-t}(X_t)\|^2 \\ \lesssim & \|\nabla \log q_{T-t}(X_t)\|^2 + L^2\|X_{kh} - X_t\|^2.\end{aligned}$$

Proof

- put these together

$$
\begin{aligned}
&\mathbb{E}_{Q_T^{\leftarrow}}\|s_{T-kh}(X_{kh}) - \nabla \log q_{T-t}(X_t)\|^2 \\
&\quad\lesssim\ \epsilon^2 + L^2 dh + L^2 h^2 \mathbb{E}_{Q_T^{\leftarrow}}\|X_{kh}\|^2 \\
&\qquad + L^2 h^2 \mathbb{E}_{Q_T^{\leftarrow}}\|\nabla \log q_{T-t}(X_t)\|^2 + L^2 \mathbb{E}_{Q_T^{\leftarrow}}\|X_{kh} - X_t\|^2.
\end{aligned}
$$

- apply moment bounds for the forward process

$$
\begin{aligned}
&\mathbb{E}_{Q_T^{\leftarrow}}\|s_{T-kh}(X_{kh}) - \nabla \log q_{T-t}(X_t)\|^2 \\
&\quad\lesssim \epsilon^2 + L^2 dh + L^2 h^2 (d + m_2^2) + L^3 h^2 d + L^2 (m^2 h^2 + dh) \\
&\quad\lesssim \epsilon^2 + L^2 dh + L^2 m_2^2 h^2.
\end{aligned}
$$

Proof

- According to Girsanov's theorem

$$\mathrm{KL}(Q_T^{\leftarrow} \| P_T^{q_T}) \lesssim (\epsilon^2 + L^2 dh + L^2 m_2^2 h^2) T.$$

- By data processing inequality

$$\begin{aligned} \mathrm{TV}(p_T, q) \leq & \mathrm{TV}(p_T^{\gamma^d}, p_T^{q_T}) + \mathrm{TV}(p_T^{q_T}, Q_T^{\leftarrow}) \\ \leq & \mathrm{TV}(q_T, \gamma^d) + \mathrm{TV}(p_T^{q_T}, Q_T^{\leftarrow}). \end{aligned}$$

- Using the convergence of the OU process in KL divergence and Pinsker inequality, we have

$$\mathrm{TV}(p_T, q) \lesssim \sqrt{\mathrm{KL}(q \| \gamma^d)} \exp(-T) + (L\sqrt{dh} + L m_2 h)\sqrt{T} + \epsilon\sqrt{T}$$

References

- Salakhutdinov, R. and Hinton, G. Deep boltzmann machines. In Artificial intelligence and statistics, 2009.
- Aapo Hyvarinen. Estimation of non-normalized statistical models by score matching. Journal of Machine Learning Research, 6(Apr):695–709, 2005.
- Pascal Vincent. A connection between score matching and denoising autoencoders. Neural computation, 23(7):1661–1674, 2011.

- Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019.
- Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In Advances in Neural Information Processing Systems, pp. 11895–11907, 2019.
- Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru Zhang. Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions. In The Eleventh International Conference on Learning Representations, 2023