XI. Score-based Method

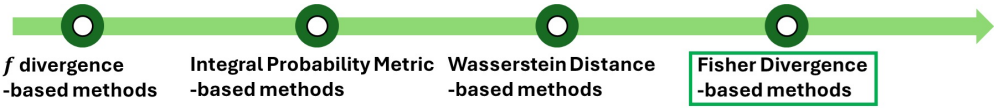
Young-geun Kim

Department of Statistics and Probability

STT 997 (SS 2025)

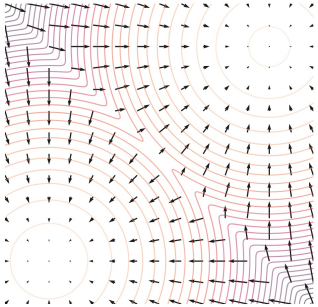
MICHIGAN STATE
UNIVERSITY

Introduction



- IPM and Wasserstein distance-based methods have alleviated optimization issues; however, adversarial training is still practically difficult.
- Recent works have focused on score functions instead of densities, using estimated scores to generate data.
- Useful materials include: `https://yang-song.net/blog/2021/score/`

Fisher Divergence

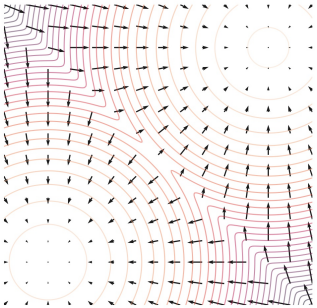


- Fisher divergence (Johnson, 2004) is the expected difference between the (Stein) scores (Liu et al., 2016) of two distributions. It can be expressed as:

$$\mathrm{FD}(p_n \parallel p_\theta) = \int \left\| \nabla_{\vec{x}} \log p_n(\vec{x}) - \nabla_{\vec{x}} \log p_\theta(\vec{x}) \right\|^2 p_n(\vec{x})\, d\vec{x}. \tag{1}$$

Visualization of $\nabla_{\vec{x}} \log p(\vec{x})$ of a mixture of two Gaussians, from `https://yang-song.net/blog/2021/score/`.

Fisher Divergence



- It is zero if and only if $p = q$.
  Proof:

$$\nabla_{\vec{x}} \log p(\vec{x}) = \nabla_{\vec{x}} \log q(\vec{x}) \implies p(\vec{x}) = Cq(\vec{x}) \tag{2}$$

  and $C = 1$ because $\int p(\vec{x}) d\vec{x} = \int q(\vec{x}) d\vec{x} = 1$.

Visualization of $\nabla_{\vec{x}} \log p(\vec{x})$ of a mixture of two Gaussians, from `https://yang-song.net/blog/2021/score/`.

Fisher Divergence

- The Fisher divergence is useful for learning energy-based models (Teh et al., 2003), such as Boltzmann distributions:

$$p_\theta(\vec{x}) = \frac{1}{C(\theta)} \exp(-E_\theta(\vec{x})) \tag{3}$$

where $C(\theta) := \int \exp(-E_\theta(\vec{x}))\, d\vec{x}$. In this case,

$$\nabla_{\vec{x}} \log p_\theta(\vec{x}) = -\nabla_{\vec{x}} E_\theta(\vec{x}) \tag{4}$$

holds, and the normalizing constant disappears.

Fisher Divergence and Kullback-Leibler Divergence

- Note that $\nabla_{\vec{x}} \log p_\theta(\vec{x})$ differs from $\nabla_\theta \log p_\theta(\vec{x}) (= \partial \log p_\theta(\vec{x})/\partial\theta)$ in maximum likelihood estimation.
- While MLEs minimize the KL-divergence between model and true distributions, score matching estimators minimize the change of the KL-divergence w.r.t. a white noise:

$$\text{FD}(p_n(\vec{x})||p_\theta(\vec{x})) = \lim_{t \to 0^+} \frac{\text{KL}(p_n(\vec{x} + \sqrt{t}\vec{w})||p_\theta(\vec{x} + \sqrt{t}\vec{w})) - \text{KL}(p_n(\vec{x})||p_\theta(\vec{x}))}{t} \tag{5}$$

  where $\vec{w}$ represents a white Gaussian noise.
- See Theorem 1 in Lyu (2009) for details. The authors also generalized score matching methods by utilizing generalized Fisher divergences.
- Under certain conditions, the minimizer of the Fisher-divergence is consistent. See Hyvärinen (2005) for further details on asymptotic properties of score-based estimators.

Outline

Data Generation using Score Function

- When we model $p_\theta(\vec{x})$ as a parametric family distribution and use the minimizer of $\text{FD}(p_n||p_\theta)$ as an estimator, we can directly generate data from $p_\theta(\vec{x})$ with the estimator.
- How to generate data using score functions $S(\vec{x})$ in general?
- The key idea is to introduce *Langevin dynamics* in the sampling process.

Data Generation using Score Function

- Langevin dynamics describes the stochastic movement of a fluid particle located at $\vec{X}(t)$:

$$m\frac{d^2\vec{X}(t)}{dt^2} = -\nabla_{\vec{x}=\vec{X}(t)}U(\vec{x}) - \lambda\frac{d\vec{X}(t)}{dt} + \sqrt{2\lambda k_B T}\vec{B}(t), \tag{6}$$

  where m is the mass, U is the potential functions, λ is the damping coefficient, k_B is the Boltzmann constant, T is the temperature, and $\vec{B}(t)$ represents the Brownian motion.

- In the overdamped case, where the inertial force is negligible, when $\lambda = 1$, we get

$$d\vec{X}(t) = -\nabla_{\vec{x}=\vec{X}(t)}U(\vec{x})\,dt + \sqrt{2k_B T}\,d\vec{B}(t) \tag{7}$$

  where $d\vec{B}(t) \sim N(0, dtI_m)$.[1] Its stationary distribution is the Boltzmann distribution with energy $U/(k_B T)$, $p(\vec{x}(\infty)) \propto \exp\Big(-U(\vec{x}(\infty))/(k_B T)\Big)$.

[1] This is a special case of the Itô drift-diffusion process.

Data Generation using Score Function

- By substituting $U(\vec{x}) = -\log p_n(\vec{x})$ and setting $T = 1/k_B$, we obtain:

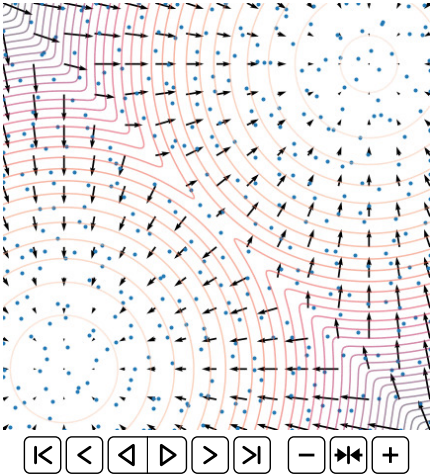
$$d\vec{X}(t) = \nabla_{\vec{x}} \log p_n(\vec{x})\, dt + \sqrt{2dt}\, \vec{\mathcal{E}}(t), \tag{8}$$

  where $\vec{\mathcal{E}}(t) \sim N(0, I_m)$, and the corresponding stationary distribution is $p_n(\vec{x})$.
- The discrete approximation with $dt = \eta/2$ and $S_{\theta^*}(\vec{x})$ results in the following iterative sampling process:

$$\vec{X}(t) = \vec{X}(t-1) + (\eta/2) S_{\theta^*}(\vec{X}(t-1)) + \sqrt{\eta}\vec{\mathcal{E}}(t), \tag{9}$$

  where $\vec{X}(T)$ approximately follows $p_n(\vec{x})$.
- See the following materials for further details:
  1. A lecture by Dr. Kirill Neklyudov `https://www.youtube.com/watch?v=3-KzIjoFJy4`
  2. Section 'Recovering Boltzmann Statistics' at `https://en.wikipedia.org/wiki/Langevin_equation`

Data Generation using Score Function



The video is from https://yang-song.net/blog/2021/score/.

Score Matching Estimation

- Score matching estimation (Hyvärinen, 2005) was proposed targeting Fisher divergence in learning distributions.
- Let $S_\theta(\vec{x}) := \nabla_{\vec{x}} \log p_\theta(\vec{x})$. Then,

$$\mathrm{FD}(p_n||p_\theta) = \int \left(\mathrm{tr}(\nabla_{\vec{x}} S_\theta(\vec{x})) + \frac{1}{2}\|S_\theta(\vec{x})\|^2\right) p_n(\vec{x})\, d\vec{x} \tag{10}$$

up to a constant addition and sign-preserving multiplication. We assume that $S_\theta(\vec{x})p_n(\vec{x})$ vanishes at the boundary, e.g., $(x_1, \ldots, x_{i-1}, \pm\infty, x_{i+1}, \ldots, x_m)$.

Score Matching Estimation

Proof:

$$\begin{aligned} \mathrm{FD}(p_n||p_\theta) &:= \int \|\nabla_{\vec{x}} \log p_n(\vec{x}) - S_\theta(\vec{x})\|^2 p_n(\vec{x})\, d\vec{x} \\ &= C - 2\int \Big(S_\theta^T(\vec{x})\nabla_{\vec{x}} \log p_n(\vec{x})\Big) p_n(\vec{x})\, d\vec{x} + \int \|S_\theta(\vec{x})\|^2 p_n(\vec{x})\, d\vec{x}. \end{aligned} \tag{11}$$

Here, $\int \Big(S_\theta^T(\vec{x})\nabla_{\vec{x}} \log p_n(\vec{x})\Big) p_n(\vec{x})\, d\vec{x}$ equals $-\int \mathrm{tr}(\nabla_{\vec{x}} S_\theta(\vec{x})) p_n(\vec{x})\, d\vec{x}$.

Score Matching Estimation

Proof (Cont.): Let $\vec{X}_{-i} := (X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_m)^T$. Then,

$$\begin{aligned}\int \left(S_\theta^T(\vec{x})\nabla_{\vec{x}} \log p_n(\vec{x})\right) p_n(\vec{x})\, d\vec{x} &= \int S_\theta^T(\vec{x})\nabla_{\vec{x}} p_n(\vec{x})\, d\vec{x} \\ &= \sum_{i=1}^m \int \left(\int S_\theta(\vec{x})_i \frac{\partial}{\partial x_i} p_n(\vec{x})\, dx_i \right) d\vec{x}_{-i}.\end{aligned} \tag{12}$$

Since $S_\theta(\vec{x})p_n(\vec{x})$ vanishes at the boundary, by partial integration, we have

$$\int S_\theta(\vec{x})_i \frac{\partial}{\partial x_i} p_n(\vec{x})\, dx_i = -\int \left(\frac{\partial}{\partial x_i} S_\theta(\vec{x})_i \right) p_n(\vec{x})\, dx_i. \tag{13}$$

Thus, $\mathrm{FD}(p_n||p_\theta) = C + \int \left(2\mathrm{tr}(\nabla_{\vec{x}} S_\theta(\vec{x})) + \|S_\theta(\vec{x})\|^2\right) p_n(\vec{x}) d\vec{x}$, which concludes the proof.

Sliced Score Matching

- In the objective of score matching estimation, $\int \left(\text{tr}(\nabla_{\vec{x}} S_\theta(\vec{x})) + \frac{1}{2}\|S_\theta(\vec{x})\|^2\right) p_n(\vec{x})\, d\vec{x}$, the Hessian term poses another computational challenge.
- Sliced score matching (Song et al., 2020) targets sliced Fisher divergence (SFD),

$$\text{SFD}(p_n||p_\theta) := \int \left\| \vec{v}^T \nabla_{\vec{x}} \log p_n(\vec{x}) - \vec{v}^T \nabla_{\vec{x}} \log p_\theta(\vec{x}) \right\|^2 p_n(\vec{x}) p(\vec{v}) d\vec{x} d\vec{v}, \qquad (14)$$

  to overcome this limitation.
- The SFD is the average difference between randomly projected scores.

Sliced Score Matching

- In a similar way used in score matching estimation,

$$\text{SFD}(p_n||p_\theta) = \int \left(\vec{v}^T \nabla_{\vec{x}} S_\theta(\vec{x})\vec{v} + \frac{1}{2}(\vec{v}^T S_\theta(\vec{x}))^2\right) p_n(\vec{x})p(\vec{v})d\vec{x}d\vec{v} \tag{15}$$

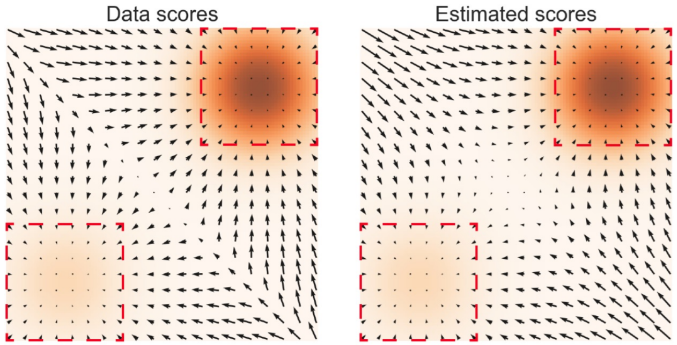
  up to a constant addition and sign-preserving multiplication.
- By changing the target statistical distances from FD to SFD, the computational bottleneck shifts from computing $\text{tr}\left(\nabla_{\vec{x}} S_\theta(\vec{x})\right)$ to computing $\vec{v}^T \nabla_{\vec{x}} S_\theta(\vec{x}) = \nabla_{\vec{x}}\left(\vec{v}^T S_\theta(\vec{x})\right)$, which is numerically less demanding.
- When $p(\vec{v})$ is the multivariate standard Gaussian distribution, the equation $\int \left(\vec{v}^T S_\theta(\vec{x})\right)^2 d\vec{v} = \|S_\theta(\vec{x})\|^2$ holds, further reducing the computational cost.
- Furthermore, under certain conditions, the minimizer of SFD is consistent and holds asymptotic normality.

Outline

Noise Conditional Score Network

- Noise Conditional Score Network (NCSN) (Song and Ermon, 2019) are score-based generative models that use estimated scores $S_\theta(\vec{x})$ to generate data.
- NCSNs introduce a U-Net-based network to model $S(\vec{x})$. Note that the shape of $S(\vec{x}) := \nabla_{\vec{x}} \log p(\vec{x})$ is the same as that of $\vec{x}$.

Noise Conditional Score Network



- The initial points are likely to lie in low-density regions, and estimation performances on those regions may poor.
  Q: The existence of low-density regions is not a new phenomenon compared to other generative models. Why can it be critical in score-based approaches?

The figure is from Song and Ermon (2019). Two red dashed-rectangles display regions where $S_\theta(\vec{x}) \approx S(\vec{x})$.

Noise Conditional Score Network: Training

- NCSNs employ the denoising score matching method (Vincent, 2011). They add noise to the data, $\vec{X} + \sigma\vec{\mathcal{E}}$, learn its score $S_\theta(\vec{x}; \sigma)$, and use $S_\theta(\vec{x}; \sigma)$ with a sufficiently small σ for effective sampling.
- For a given noise level σ, the objective can be expressed as:

$$l(\theta; \sigma) := \int \|S_\theta(\vec{x}, \sigma) - \nabla_{\vec{x}+\sigma\vec{\epsilon}} \log p(\vec{x} + \sigma\vec{\epsilon}|\vec{x})\|^2 p(\vec{\epsilon}; \vec{0}, I_p) p_n(\vec{x}) d\vec{\epsilon} d\vec{x}. \tag{16}$$

  Here, $\nabla_{\vec{x}+\sigma\vec{\epsilon}} \log p(\vec{x} + \sigma\vec{\epsilon}|\vec{x}) = -\vec{\epsilon}/\sigma^2$.
- The final objective can be expressed as:

$$\sum_{l=1}^{L} \lambda_l l(\theta; \sigma_l) \tag{17}$$

  where $(\lambda_l)_{l=1}^L$ represents coefficients corresponding to noise levels $(\sigma_l := \sigma_1 \gamma^{l-1})_{l=1}^L$.

Noise Conditional Score Network: Generation

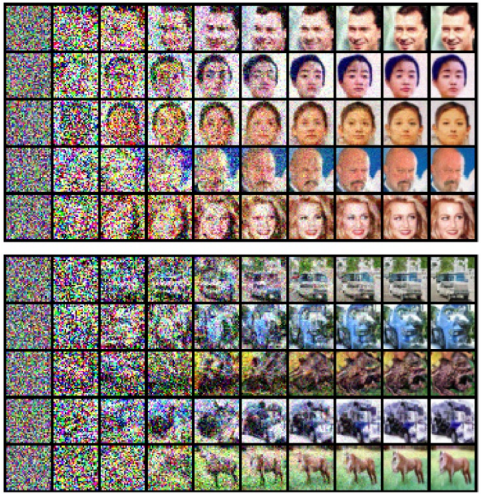
Algorithm 1 Annealed Langevin dynamics.

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 $\tilde{\mathbf{x}}_T$

The algorithm is from Song and Ermon (2019).

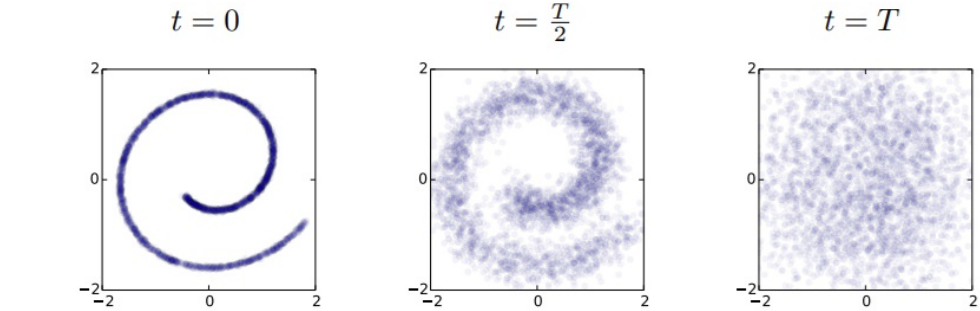
Noise Conditional Score Network: Result



The figure is from Song and Ermon (2019).

Outline

Motivation: Diffusion



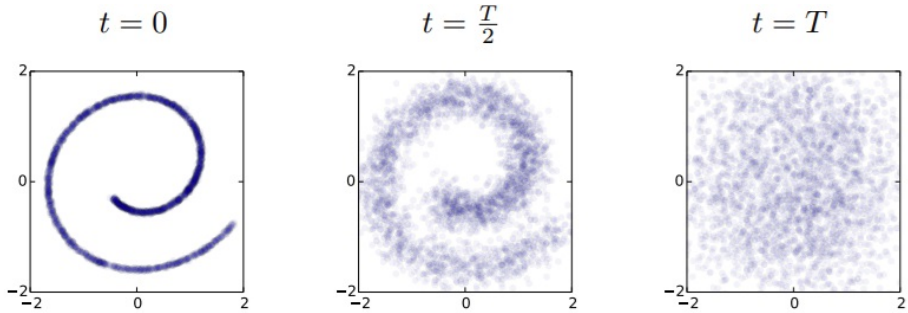
- For a given datum from $p_n(\vec{x})$, let's gradually mix it with Gaussian noises:

$$\vec{x}(t) := \sqrt{1-\beta_t}\vec{x}(t-1) + \sqrt{\beta_t}\vec{\epsilon}(t) \tag{18}$$

where $\vec{x}(0)$ represents the initial datum sampled from $p_n(\vec{x})$ and $\vec{\epsilon}(t) \sim N(0, I_p)$.

The figure is from Sohl-Dickstein et al. (2015).

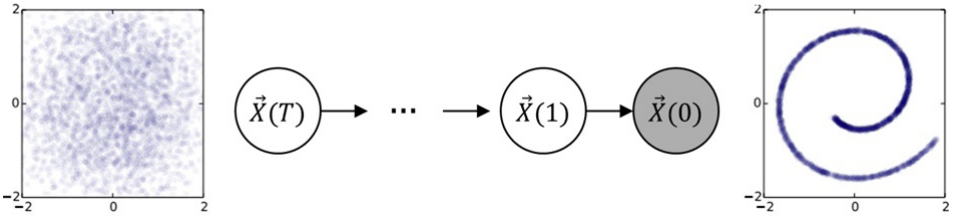
Motivation: Diffusion



- Then, the distribution of $\vec{x}(T)$ converges to $N(0, I_p)$ as $T \to \infty$ and $\beta_t \to 0$.
- The key idea of diffusion model is to learn $p(\vec{x}(0)|\vec{x}(T))$ to generate data.

The figure is from Sohl-Dickstein et al. (2015).

Denoising Diffusion Probabilistic Model: Generation



- The generation process (called "Backward Diffusion Process") can be formulated as:

$$p_\theta(\vec{x}(0), \ldots, \vec{x}(T)) = p(\vec{x}(T)) \prod_{t=1}^{T} p_\theta(\vec{x}(t-1)|\vec{x}(t)) \tag{19}$$

where $p(\vec{x}(T))$ is the density of $N(0, I_p)$ and $p_\theta(\vec{x}(t-1)|\vec{x}(t))$ is that of $N(\vec{\mu}_\theta(\vec{x}(t), t), \Sigma_\theta(\vec{x}(t), t))$.

Denoising Diffusion Probabilistic Model: Inference

- Denoising Diffusion Probabilistic Model (DDPM) (Ho et al., 2020) is a generative model based on this idea.
- The data likelihood $p_\theta(\vec{x}(0))$ is highly intractable. DDPMs apply variational inference.
- The diffusion process gives a closed-form expression of the optimal posterior:[2]

$$\begin{aligned} & q(\vec{x}(t)|\vec{x}(0), \ldots, \vec{x}(t-1))(= q(\vec{x}(t)|\vec{x}(t-1))) \\ & = p(\vec{x}(t); \sqrt{1-\beta_t}\vec{x}(t-1), \beta_t I_p). \end{aligned} \tag{20}$$

It is called "Forward Diffusion Process". This implies

$$q(\vec{x}(1), \ldots, \vec{x}(T)|\vec{x}(0)) = \prod_{t=1}^{T} q(\vec{x}(t)|\vec{x}(t-1)). \tag{21}$$

[2] In other words, we already know the ground-truth inference process.

Denoising Diffusion Probabilistic Model: Loss

- The loss function is the negative ELBO w.r.t. $\log p_\theta(\vec{x}(0))$:

$$-\log p_\theta(\vec{x}(0)) + \mathrm{KL}(q(\vec{x}(1), \ldots, \vec{x}(T)|\vec{x}(0))||p_\theta(\vec{x}(1), \ldots, \vec{x}(T)|\vec{x}(0))). \tag{22}$$

- This negative ELBO can be expressed as:

$$\begin{aligned} &-\int (\log p_\theta(\vec{x}(0)|\vec{x}(1)))\, q(\vec{x}(1)|\vec{x}(0)) d\vec{x}(1) + \mathrm{KL}(q(\vec{x}(T)|\vec{x}(0))||p(\vec{x}(T))) \\ &+\sum_{t=2}^{T} \int \mathrm{KL}(q(\vec{x}(t-1)|\vec{x}(0), \vec{x}(t))||p_\theta(\vec{x}(t-1)|\vec{x}(t)))q(\vec{x}(t)|\vec{x}(0))d\vec{x}(t). \end{aligned} \tag{23}$$

Denoising Diffusion Probabilistic Model: Loss

Proof:

$$
\begin{aligned}
&\int \left(\log p_\theta(\vec{x}(0)|\vec{x}(1),\ldots,\vec{x}(T))\right) q(\vec{x}(1),\ldots,\vec{x}(T)|\vec{x}(0)) d\vec{x}(1)\cdots d\vec{x}(T) \\
&\quad - \mathrm{KL}(q(\vec{x}(1),\ldots,\vec{x}(T)|\vec{x}(0))||p_\theta(\vec{x}(1),\ldots,\vec{x}(T))) \\
&= \int \left(\log p_\theta(\vec{x}(0)|\vec{x}(1))\right) q(\vec{x}(1)|\vec{x}(0)) d\vec{x}(1) \\
&\quad - \mathrm{KL}(q(\vec{x}(1),\ldots,\vec{x}(T)|\vec{x}(0))||p_\theta(\vec{x}(1),\ldots,\vec{x}(T))).
\end{aligned} \tag{24}
$$

In the last KL term, the second argument can be expressed as $p(\vec{x}(T)) \prod_{t=2}^{T} p_\theta(\vec{x}(t-1)|\vec{x}(t))$.

Denoising Diffusion Probabilistic Model: Loss

Proof (Conti.):

- In the last KL term, the first argument can be expressed as $\prod_{t=1}^{T} q(\vec{x}(t)|\vec{x}(t-1))$. Here, by Bayes' Theorem,

$$q(\vec{x}(t-1)|\vec{x}(0),\vec{x}(t)) = \frac{q(\vec{x}(t)|\vec{x}(0),\vec{x}(t-1))q(\vec{x}(t-1)|\vec{x}(0))}{q(\vec{x}(t)|\vec{x}(0))}. \tag{25}$$

Note that, the Forward Diffusion Process structure implies $q(\vec{x}(t)|\vec{x}(0),\vec{x}(t-1)) = q(\vec{x}(t)|\vec{x}(t-1))$, yielding:

$$q(\vec{x}(t)|\vec{x}(t-1)) = q(\vec{x}(t-1)|\vec{x}(0),\vec{x}(t))\frac{q(\vec{x}(t)|\vec{x}(0))}{q(\vec{x}(t-1)|\vec{x}(0))} \tag{26}$$

for $t > 1$.

Denoising Diffusion Probabilistic Model: Loss

Proof (Conti.):

- These imply:

$$
\begin{aligned}
&\log \frac{q(\vec{x}(1), \ldots, \vec{x}(T)|\vec{x}(0))}{p_\theta(\vec{x}(1), \ldots, \vec{x}(T))} \\
&= \log \frac{\prod_{t=1}^{T} q(\vec{x}(t)|\vec{x}(t-1))}{p(\vec{x}(T)) \prod_{t=2}^{T} p_\theta(\vec{x}(t-1)|\vec{x}(t))} \\
&= \log \frac{q(\vec{x}(1)|\vec{x}(0)) \prod_{t=2}^{T} q(\vec{x}(t-1)|\vec{x}(0), \vec{x}(t)) q(\vec{x}(t-1)|\vec{x}(0))/q(\vec{x}(t)|\vec{x}(0))}{p(\vec{x}(T)) \prod_{t=2}^{T} p_\theta(\vec{x}(t-1)|\vec{x}(t))} \\
&= \log \frac{q(\vec{x}(T)|\vec{x}(0))}{p(\vec{x}(T))} + \sum_{t=2}^{T} \log \frac{q(\vec{x}(t-1)|\vec{x}(0), \vec{x}(t))}{p_\theta(\vec{x}(t-1)|\vec{x}(t))}.
\end{aligned} \tag{27}
$$

Denoising Diffusion Probabilistic Model: Loss

Proof (Conti.):

- Thus,

$$
\begin{aligned}
&\mathrm{KL}(q(\vec{x}(1),\ldots,\vec{x}(T)|\vec{x}(0))||p_\theta(\vec{x}(1),\ldots,\vec{x}(T))) \\
&= \int \left(\log \frac{q(\vec{x}(T)|\vec{x}(0))}{p(\vec{x}(T))}\right) q(\vec{x}(T)|\vec{x}(0))d\vec{x}(T) \\
&\quad + \sum_{t=2}^{T} \int \left(\log \frac{q(\vec{x}(t-1)|\vec{x}(0),\vec{x}(t))}{p_\theta(\vec{x}(t-1)|\vec{x}(t))}\right) q(\vec{x}(t-1),\vec{x}(t)|\vec{x}(0))d\vec{x}(t-1)d\vec{x}(t) \qquad (28) \\
&= \mathrm{KL}(q(\vec{x}(T)|\vec{x}(0))||p(\vec{x}(T))) \\
&\quad + \sum_{t=2}^{T} \int \mathrm{KL}(q(\vec{x}(t-1)|\vec{x}(0),\vec{x}(t))||p_\theta(\vec{x}(t-1)|\vec{x}(t)))q(\vec{x}(t)|\vec{x}(0))d\vec{x}(t).
\end{aligned}
$$

Denoising Diffusion Probabilistic Model: Loss

- Again, the loss function, the negative ELBO, can be expressed as:

$$-\int (\log p_\theta(\vec{x}(0)|\vec{x}(1)))\, q(\vec{x}(1)|\vec{x}(0)) d\vec{x}(1) + \mathrm{KL}(q(\vec{x}(T)|\vec{x}(0))||p(\vec{x}(T)))$$
$$+\sum_{t=2}^{T}\int \mathrm{KL}(q(\vec{x}(t-1)|\vec{x}(0),\vec{x}(t))||p_\theta(\vec{x}(t-1)|\vec{x}(t)))q(\vec{x}(t)|\vec{x}(0))d\vec{x}(t).$$

- Here, $q(\vec{x}(t-1)|\vec{x}(0),\vec{x}(t))$ is a density of Gaussian, so we can use closed-form expressions of KL between Gaussians in computing the loss function. Specifically, it is the density of $N(\vec{\mu}_t(\vec{x}(0),\vec{x}(t)),\bar{\beta}_t I_p)$ (Ho et al., 2020), where:

$$\vec{\mu}_t(\vec{x}(0),\vec{x}(t)) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\vec{x}(0) + \frac{\sqrt{\alpha}_t(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\vec{x}(t), \tag{29}$$

$\bar{\beta}_t := (1-\bar{\alpha}_{t-1})\beta_t/(1-\bar{\alpha}_t)$, and $\bar{\alpha}_t := \prod_{s=1}^{t}(1-\beta_s)$.

Denoising Diffusion Probabilistic Model: Loss

- When $p_\theta(\vec{x}(t-1)|\vec{x}(t))$ is the density of $N(\vec{\mu}_\theta(\vec{x}(t),t), \bar{\beta}_t I_p)$, the KL term can be expressed as:

$$\mathrm{KL}(q(\vec{x}(t-1)|\vec{x}(0),\vec{x}(t))||p_\theta(\vec{x}(t-1)|\vec{x}(t))) = \frac{1}{2\bar{\beta}_t}\|\vec{\mu}_t(\vec{x}(0),\vec{x}(t)) - \vec{\mu}_\theta(\vec{x}(t),t)\|^2 \quad (30)$$

  up to constant addition.
- Note that $\vec{x}(t) = \sqrt{\bar{\alpha}_t}\vec{x}(0) + \sqrt{1-\bar{\alpha}_t}\vec{\epsilon}$ for some Gaussian noise $\vec{\epsilon}$, implying that:

$$\vec{\mu}_t(\vec{x}(0),\vec{x}(t)) = \frac{1}{\sqrt{\alpha_t}}\left(\vec{x}(t) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\vec{\epsilon}\right). \quad (31)$$

  Thus, learning $\vec{\mu}_\theta(\vec{x}(t),t)$ can be re-formulated as estimating $\vec{\epsilon}$ using $(\vec{x}(t),t)$.

Denoising Diffusion Probabilistic Model: Loss

- Motivated by this, DDPMs introduce noise prediction network $\vec{\epsilon}_\theta(\vec{x}(t), t) = \vec{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\vec{x}(0) + \sqrt{1-\bar{\alpha}_t}\vec{\epsilon}, t)$. See Section 3.2. in Ho et al. (2020) for further details.
- Though DDPMs utilize variational inference, placing them in the category of likelihood-based approaches, they are equivalent to score-based generative models.
- The estimated $\vec{\epsilon}_\theta(\vec{x}_t, t)$ can be interpreted as an estimate of score functions of data with Gaussian noises, so ϵ_θ can be used to sample data via Langevin dynamics.

Denoising Diffusion Probabilistic Model: Algorithm

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

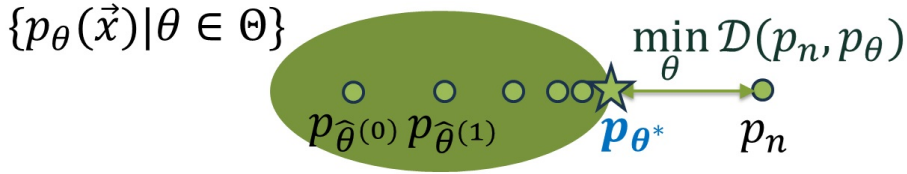
Algorithm tables are from Ho et al. (2020).

Denoising Diffusion Probabilistic Model: Result

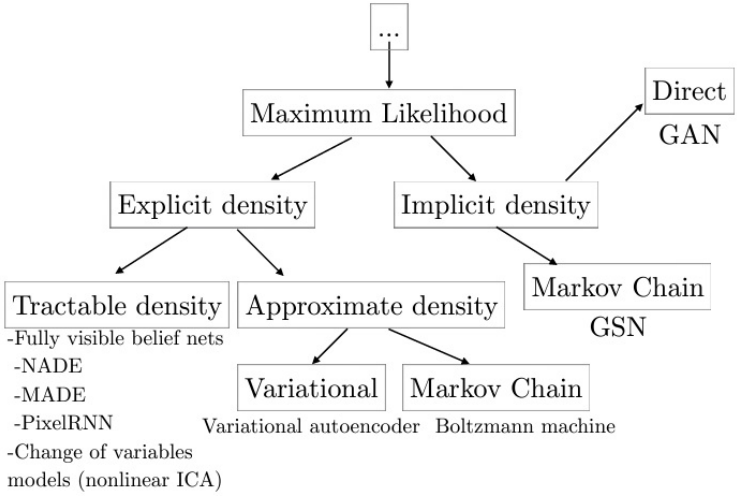
Images are from Ho et al. (2020).

Outline

Summary


$\{p_\theta(\vec{x})|\theta \in \Theta\}$
$\min_\theta \mathcal{D}(p_n, p_\theta)$
$p_{\hat{\theta}^{(0)}}$
$p_{\hat{\theta}^{(1)}}$
p_{θ^*}
p_n

Summary



The figure is from Goodfellow (2016).

Summary

- We have reviewed recent developments in deep generative models, with a particular focus on targeted statistical distances.
- There are many ways to explore recent advancements in deep generative models. Examples include developing model classes for specific data types, analyzing the properties of model classes and estimators, or adapting large pre-trained models to new datasets.

Concluding Remark

- Such changes may not always be exciting for everyone.
- Rapid advancements inevitably bring anxiety and uncertainty as the expectations of academia and industry continue to evolve.[3]
- I hope that this course has helped you find joy and beauty in deep generative models and has also encouraged you to continue exploring new waves in this evolving field.

[3] *"i sensed anxiety and frustration at NeurIPS'24"*
`https://kyunghyuncho.me/i-sensed-anxiety-and-frustration-at-neurips24/`

References I

Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851.

Hyvärinen, A. (2005). Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4).

Johnson, O. (2004). *Information theory and the central limit theorem*. World Scientific.

Liu, Q., Lee, J., and Jordan, M. (2016). A kernelized stein discrepancy for goodness-of-fit tests. In *International conference on machine learning*, pages 276–284. PMLR.

Lyu, S. (2009). Interpretation and generalization of score matching. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 359–366.

References II

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR.

Song, Y. and Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32.

Song, Y., Garg, S., Shi, J., and Ermon, S. (2020). Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*, pages 574–584. PMLR.

Teh, Y. W., Welling, M., Osindero, S., and Hinton, G. E. (2003). Energy-based models for sparse overcomplete representations. *Journal of Machine Learning Research*, 4(Dec):1235–1260.

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