Diffusion Processes and Generative Models

Evgeny Burnaev

Skoltech, Moscow, Russia

Skoltech

Skolkovo Institute of Science and Technology

Outline

Forward gaussian diffusion process

Let $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x})$, $\beta_t \in (0, 1)$. Define the Markov chain

$$\mathbf{x}_t = \sqrt{1 - \beta_t} \cdot \mathbf{x}_{t-1} + \sqrt{\beta_t} \cdot \boldsymbol{\epsilon}, \quad \text{where } \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I});$$

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t | \sqrt{1 - \beta_t} \cdot \mathbf{x}_{t-1}, \beta_t \cdot \mathbf{I}).$$

Statement 1

Let denote $\alpha_t = 1 - \beta_t$ and $\overline{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. Then

$$\begin{aligned}
\mathbf{x}_t &= \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1 - \alpha_t}\boldsymbol{\epsilon}_t = \\
&= \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1 - \alpha_{t-1}}\boldsymbol{\epsilon}_{t-1}) + \sqrt{1 - \alpha_t}\boldsymbol{\epsilon}_t = \\
&= \sqrt{\alpha_t \alpha_{t-1}}\mathbf{x}_{t-2} + (\sqrt{\alpha_t(1 - \alpha_{t-1})}\boldsymbol{\epsilon}_{t-1} + \sqrt{1 - \alpha_t}\boldsymbol{\epsilon}_t) = \\
&= \sqrt{\overline{\alpha}_t} \cdot \mathbf{x}_0 + \sqrt{1 - \overline{\alpha}_t} \cdot \boldsymbol{\epsilon}, \quad \text{where } \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}).
\end{aligned}$$

$$q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t | \sqrt{\overline{\alpha}_t} \cdot \mathbf{x}_0, (1 - \overline{\alpha}_t) \cdot \mathbf{I}).$$

We could sample from any timestamp using only $\mathbf{x}_0$!

See `http://proceedings.mlr.press/v37/sohl-dickstein15.pdf`, **Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015**

Forward gaussian diffusion process

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t|\sqrt{1-\beta_t}\cdot\mathbf{x}_{t-1}, \beta_t\cdot\mathbf{I});$$
$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t|\sqrt{\overline{\alpha}_t}\cdot\mathbf{x}_0, (1-\overline{\alpha}_t)\cdot\mathbf{I}).$$

At each step we

- scale magnitude of the signal at rate $\sqrt{1-\beta_t}$;
- add noise with variance β_t.

Statement 2

Applying the Markov chain to samples from any $\pi(\mathbf{x})$ we will get $\mathbf{x}_\infty \sim p_\infty(\mathbf{x}) = \mathcal{N}(0,1)$. Here $p_\infty(\mathbf{x})$ is a **stationary** and **limiting** distribution:

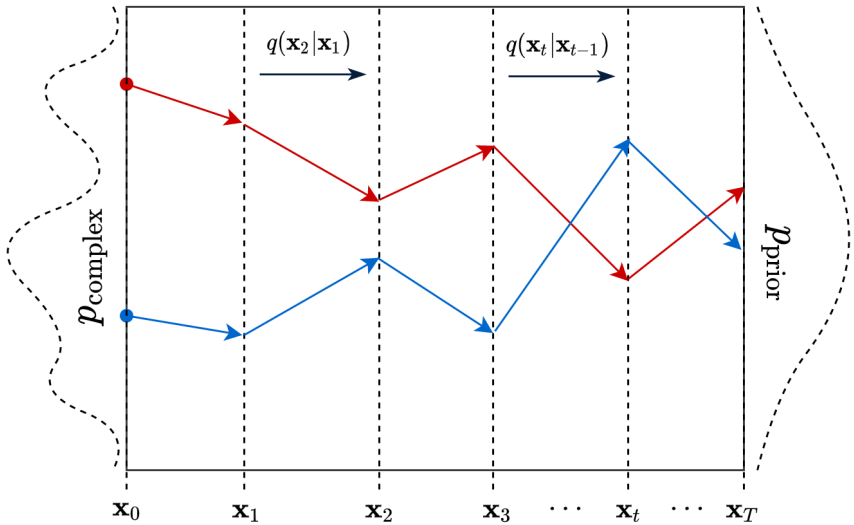
$$p_\infty(\mathbf{x}) = \int q(\mathbf{x}|\mathbf{x}')p_\infty(\mathbf{x}')d\mathbf{x}'.$$

$$p_\infty(\mathbf{x}) = \int q(\mathbf{x}_\infty|\mathbf{x}_0)\pi(\mathbf{x}_0)d\mathbf{x}_0 \approx \mathcal{N}(0,\mathbf{I})\int \pi(\mathbf{x}_0)d\mathbf{x}_0 = \mathcal{N}(0,\mathbf{I})$$

See `http://proceedings.mlr.press/v37/sohl-dickstein15.pdf`, **Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015**

Forward gaussian diffusion process

Diffusion refers to the flow of particles from high-density regions towards low-density regions.

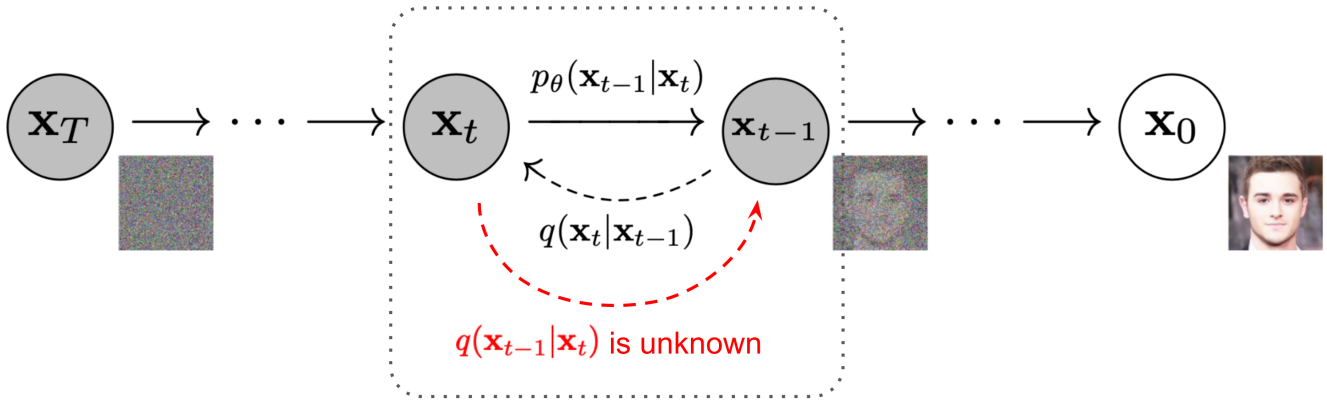


1. $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x})$;
2. $\mathbf{x}_t = \sqrt{1-\beta_t} \cdot \mathbf{x}_{t-1} + \sqrt{\beta_t} \cdot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$, $t \geq 1$;
3. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, 1)$, where $T >> 1$.

If we are able to invert this process, we will get the way to sample $\mathbf{x} \sim \pi(\mathbf{x})$ using noise samples $p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$.
Now our goal is to revert this process.

See `https://ayandas.me/blog-tut/2021/12/04/diffusion-prob-models.html`, **Das A. An introduction to Diffusion Probabilistic Models, blog post, 2021**

Reverse gaussian diffusion process



Forward process

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t|\sqrt{1-\beta_t}\cdot\mathbf{x}_{t-1}, \beta_t\cdot\mathbf{I}).$$

Reverse process

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})q(\mathbf{x}_{t-1})}{q(\mathbf{x}_t)} \approx p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})$$

- $q(\mathbf{x}_{t-1})$, $q(\mathbf{x}_t)$ are intractable.
- If β_t is small enough, $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ will be Gaussian (Feller, 1949).

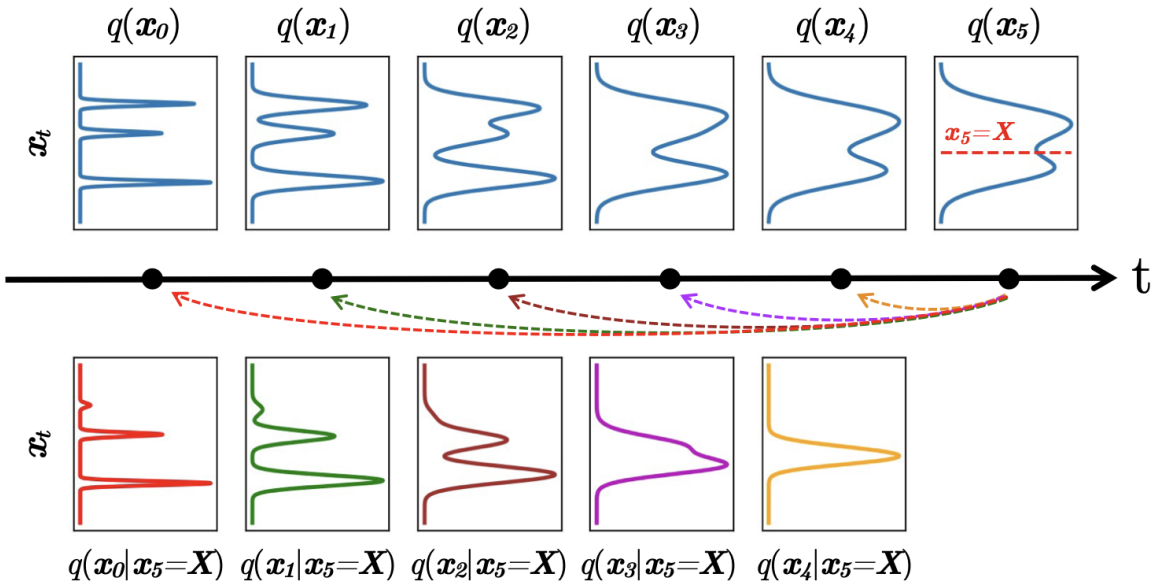
See Feller W. On the theory of stochastic processes, with particular reference to applications, 1949

Reverse gaussian diffusion process

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})q(\mathbf{x}_{t-1})}{q(\mathbf{x}_t)}$$

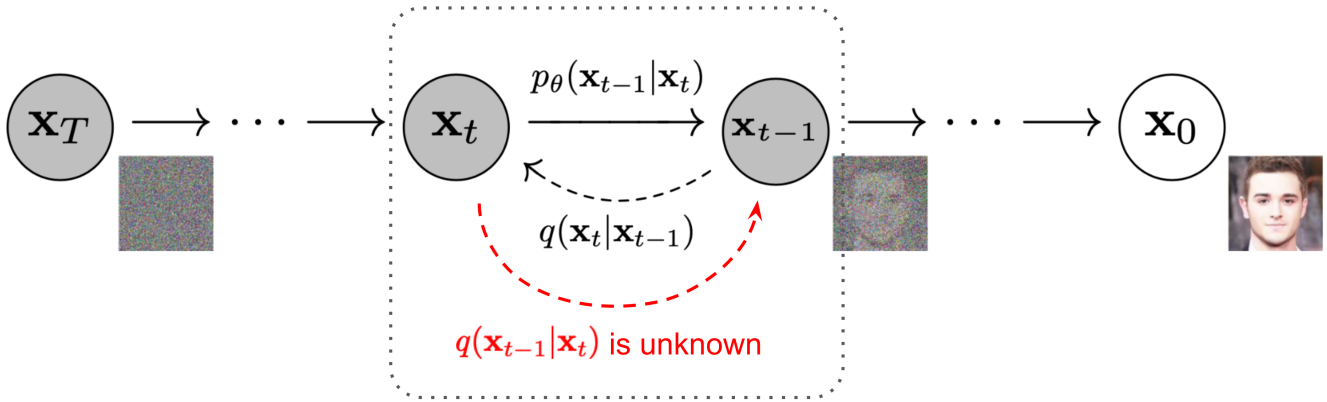
$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} = \mathcal{N}(\widetilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \widetilde{\beta}_t\mathbf{I})$$

- $q(\mathbf{x}_{t-1})$, $q(\mathbf{x}_t)$ are intractable.
- If β_t is small enough, $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ will be Gaussian (Feller, 1949).



See https://arxiv.org/abs/2112.07804, **Xiao Z., Kreis K., Vahdat A. Tackling the generative learning trilemma with denoising diffusion GANs, 2021**

Reverse gaussian diffusion process



Let define the reverse process

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) \approx p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta}) = \mathcal{N}\left(\mathbf{x}_{t-1}|\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \boldsymbol{\sigma}^2_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right)$$

Forward process:

1. $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x})$;
2. $\mathbf{x}_t = \sqrt{1-\beta_t} \cdot \mathbf{x}_{t-1} + \sqrt{\beta_t} \cdot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$, $t \geq 1$;
3. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$.

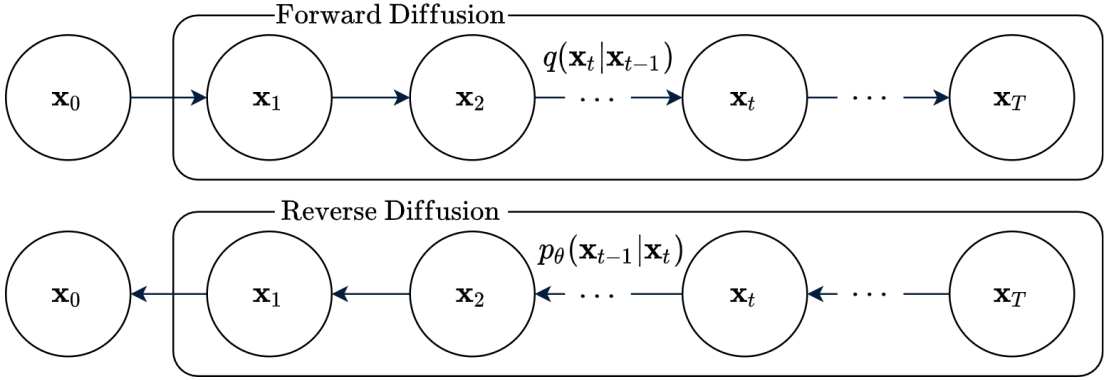
Reverse process:

1. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$;
2. $\mathbf{x}_{t-1} = \boldsymbol{\sigma}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \cdot \boldsymbol{\epsilon} + \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$;
3. $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x})$;

Note: The forward process does not have any learnable parameters!

See https://lilianweng.github.io/posts/2021-07-11-diffusion-models/, **Weng L. What are Diffusion Models?, blog post, 2021**

Gaussian diffusion model as VAE



- Let treat $\mathbf{z} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$ as a latent variable (**note:** each $\mathbf{x}_t$ has the same size).
- Variational posterior distribution (**note:** there is no learnable parameters)

$$q(\mathbf{z}|\mathbf{x}) = q(\mathbf{x}_1, \ldots, \mathbf{x}_T|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}).$$

- Probabilistic model

$$p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) = p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})p(\mathbf{z}|\boldsymbol{\theta})$$

- Generative distribution and prior

$$p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}); \quad p(\mathbf{z}|\boldsymbol{\theta}) = \prod_{t=2}^{T} p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta}) \cdot p(\mathbf{x}_T)$$

See `https://ayandas.me/blog-tut/2021/12/04/diffusion-prob-models.html`, **Das A. An introduction to Diffusion Probabilistic Models, blog post, 2021**

ELBO for gaussian diffusion model

Standard ELBO

$$\log p(\mathbf{x}|\boldsymbol{\theta}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})} = \mathcal{L}(q, \boldsymbol{\theta}) \to \max_{q, \boldsymbol{\theta}}$$

Derivation

$$\begin{aligned}
\mathcal{L}(q, \boldsymbol{\theta}) &= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \log \frac{p(\mathbf{x}_0, \mathbf{x}_{1:T}|\boldsymbol{\theta})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \\
&= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \log \frac{p(\mathbf{x}_T) \prod_{t=1}^{T} p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})}{\prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})} \\
&= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \log \frac{p(\mathbf{x}_T) p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) \prod_{t=2}^{T} p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})}{q(\mathbf{x}_1|\mathbf{x}_0) \prod_{t=2}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})} \\
&= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \log \frac{p(\mathbf{x}_T) p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) \prod_{t=2}^{T} p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})}{q(\mathbf{x}_1|\mathbf{x}_0) \prod_{t=2}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)}
\end{aligned}$$

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0) q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} = \mathcal{N}(\widetilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \widetilde{\beta}_t \mathbf{I})$$

See https://arxiv.org/abs/2006.11239, **Ho J. Denoising Diffusion Probabilistic Models, 2020**

Derivation (continued)

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \log \frac{p(\mathbf{x}_T)p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) \prod_{t=2}^{T} p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})}{q(\mathbf{x}_1|\mathbf{x}_0) \prod_{t=2}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)} =$$

$$= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \log \frac{p(\mathbf{x}_T)p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) \prod_{t=2}^{T} p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})}{q(\mathbf{x}_1|\mathbf{x}_0) \prod_{t=2}^{T} \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}} =$$

$$= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \log \frac{p(\mathbf{x}_T)p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) \prod_{t=2}^{T} p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})}{q(\mathbf{x}_T|\mathbf{x}_0) \prod_{t=2}^{T} q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} =$$

$$= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[\log p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) + \log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T|\mathbf{x}_0)} + \sum_{t=2}^{T} \log \left(\frac{p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} \right) \right] =$$

$$= \mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)} \log p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) + \mathbb{E}_{q(\mathbf{x}_T|\mathbf{x}_0)} \log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T|\mathbf{x}_0)} +$$

$$+ \sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_{t-1}, \mathbf{x}_t|\mathbf{x}_0)} \log \left(\frac{p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} \right)$$

See https://arxiv.org/abs/2006.11239, **Ho J. Denoising Diffusion Probabilistic Models, 2020**

ELBO for gaussian diffusion model

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)} \log p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) + \mathbb{E}_{q(\mathbf{x}_T|\mathbf{x}_0)} \log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T|\mathbf{x}_0)} +$$

$$+ \sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_{t-1}, \mathbf{x}_t|\mathbf{x}_0)} \log \left(\frac{p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} \right) =$$

$$= \mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)} \log p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) - KL\big(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T)\big) -$$

$$- \sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} KL\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})\big)}_{\mathcal{L}_t}$$

- First term is a decoder distribution

$$\log p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) = \log \mathcal{N}\big(\mathbf{x}_0|\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_1, t), \boldsymbol{\sigma}^2_{\boldsymbol{\theta}}(\mathbf{x}_1, t)\big).$$

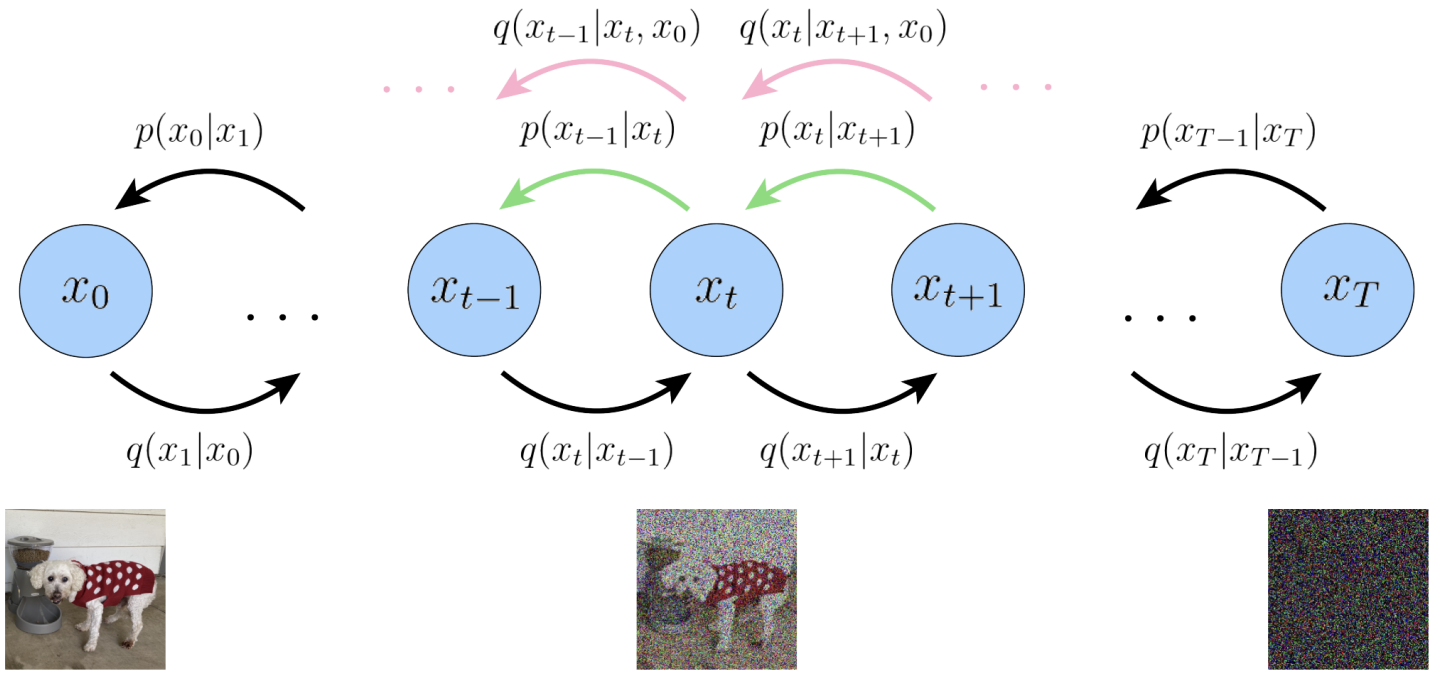
- Second term is constant ($p(\mathbf{x}_T)$ is a standard Normal, $q(\mathbf{x}_T|\mathbf{x}_0)$ is a non-parametrical Normal).

See https://arxiv.org/abs/2006.11239, **Ho J. Denoising Diffusion Probabilistic Models, 2020**

ELBO for gaussian diffusion model

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)} \log p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) - KL\big(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T)\big) - \\ - \sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} KL\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})\big)}_{\mathcal{L}_t}$$

$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ defines how to denoise a noisy image $\mathbf{x}_t$ with access to what the final, completely denoised image $\mathbf{x}_0$ should be.



See https://arxiv.org/abs/2208.11970, **Luo C. Understanding Diffusion Models: A Unified Perspective, 2022**

Summary

- Gaussian diffusion process is a Markov chain that injects special form of Gaussian noise to the samples.
- Reverse process allows to sample from the real distribution $\pi(\mathbf{x})$ using samples from noise.
- Diffusion model is a VAE model which reverts gaussian diffusion process using variational inference.
- ELBO of DDPM could be represented as a sum of KL terms.

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} KL\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})\big)$$

$\mathcal{L}_t$ is the mean of KL between two normal distributions:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}|\widetilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \widetilde{\beta}_t \mathbf{I}),$$
$$p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta}) = \mathcal{N}\big(\mathbf{x}_{t-1}|\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \boldsymbol{\sigma}^2_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\big)$$

Here

$$\widetilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\alpha_t}(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\overline{\alpha}_{t-1}}(1-\alpha_t)}{1-\overline{\alpha}_t}\mathbf{x}_0;$$
$$\widetilde{\beta}_t = \frac{(1-\alpha_t)(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t} = \text{const.}$$

Let assume

$$\boldsymbol{\sigma}^2_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = \widetilde{\beta}_t \mathbf{I} \quad \Rightarrow \quad p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta}) = \mathcal{N}\big(\mathbf{x}_{t-1}|\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \widetilde{\beta}_t \mathbf{I}\big).$$

See https://arxiv.org/abs/2006.11239, **Ho J. Denoising Diffusion Probabilistic Models, 2020**

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}|\widetilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \widetilde{\beta}_t\mathbf{I});$$
$$p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}_{t-1}|\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \widetilde{\beta}_t\mathbf{I}).$$

Use the formula for KL between two normal distributions:

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} KL\Big(\mathcal{N}\big(\widetilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \widetilde{\beta}_t\mathbf{I}\big)||\mathcal{N}\big(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \widetilde{\beta}_t\mathbf{I}\big)\Big)$$
$$= \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\frac{1}{2\widetilde{\beta}_t}\left\|\widetilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right\|^2\right]$$

$$\mathbf{x}_t = \sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon} \quad \Rightarrow \quad \mathbf{x}_0 = \frac{\mathbf{x}_t - \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}}{\sqrt{\overline{\alpha}_t}}$$

$$\widetilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\alpha_t}(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\overline{\alpha}_{t-1}}(1-\alpha_t)}{1-\overline{\alpha}_t}\mathbf{x}_0$$
$$= \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{\alpha_t(1-\overline{\alpha}_t)}}\boldsymbol{\epsilon}$$

See https://arxiv.org/abs/2006.11239, **Ho J. Denoising Diffusion Probabilistic Models, 2020**

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\left[\frac{1}{2\widetilde{\beta}_t}\left\|\widetilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right\|^2\right]$$

Reparametrization

$$\widetilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{\alpha_t(1-\overline{\alpha}_t)}}\boldsymbol{\epsilon}$$

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{\alpha_t(1-\overline{\alpha}_t)}}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$$

$$\mathcal{L}_t = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\left[\frac{(1-\alpha_t)^2}{2\widetilde{\beta}_t\alpha_t(1-\overline{\alpha}_t)}\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right\|^2\right]$$

$$= \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\left[\frac{(1-\alpha_t)^2}{2\widetilde{\beta}_t\alpha_t(1-\overline{\alpha}_t)}\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}\left(\sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}, t\right)\right\|^2\right]$$

At each step of reverse diffusion process we try to predict the noise $\boldsymbol{\epsilon}$ that we used in the forward diffusion process!

See https://arxiv.org/abs/2006.11239, **Ho J. Denoising Diffusion Probabilistic Models, 2020**

Reparametrization of DDPM

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)} \log p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) - KL\big(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T)\big) - \\ - \sum_{t=2}^{T} \underbrace{\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} KL\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta})\big)}_{\mathcal{L}_t}$$

$$\mathcal{L}_t = \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})} \left[\frac{(1-\alpha_t)^2}{2\widetilde{\beta}_t \alpha_t (1-\overline{\alpha}_t)} \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}} \left(\sqrt{\overline{\alpha}_t} \mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t} \boldsymbol{\epsilon}, t \right) \right\|^2 \right]$$

Simplified objective

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{t \sim U[2,T]} \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})} \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}} \left(\sqrt{\overline{\alpha}_t} \mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t} \boldsymbol{\epsilon}, t \right) \right\|^2$$

See https://arxiv.org/abs/2006.11239, **Ho J. Denoising Diffusion Probabilistic Models, 2020**

Denoising diffusion probabilistic model (DDPM)

DDPM is a VAE model

- Encoder is a fixed Gaussian Markov chain $q(\mathbf{x}_1, \ldots, \mathbf{x}_T | \mathbf{x}_0)$.
- Latent variable is a hierarchical (in each step the dim. of the latent equals to the dim of the input).
- Decoder is a simple Gaussian model $p(\mathbf{x}_0 | \mathbf{x}_1, \boldsymbol{\theta})$.
- Prior distribution is given by parametric Gaussian Makov chain $p(\mathbf{x}_{t-1} | \mathbf{x}_t, \boldsymbol{\theta})$.

Forward process:

1. $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x})$;
2. $\mathbf{x}_t = \sqrt{1 - \beta_t} \cdot \mathbf{x}_{t-1} + \sqrt{\beta_t} \cdot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$, $t \geq 1$;
3. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$.

Reverse process:

1. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$;
2. $\mathbf{x}_{t-1} = \boldsymbol{\sigma}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \cdot \boldsymbol{\epsilon} + \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$;
3. $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x})$;

Denoising diffusion probabilistic model (DDPM)

Training

1. Get the sample $\mathbf{x}_0 \sim \pi(\mathbf{x})$.
2. Sample timestamp $t \sim U[1, T]$ and the noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$.
3. Get noisy image $\mathbf{x}_t = \sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \overline{\alpha}_t}\boldsymbol{\epsilon}$.
4. Compute loss $\mathcal{L}_{\text{simple}} = \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\|^2$.

Sampling

1. Sample $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$.
2. Compute mean of $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \widetilde{\beta}_t\mathbf{I})$:

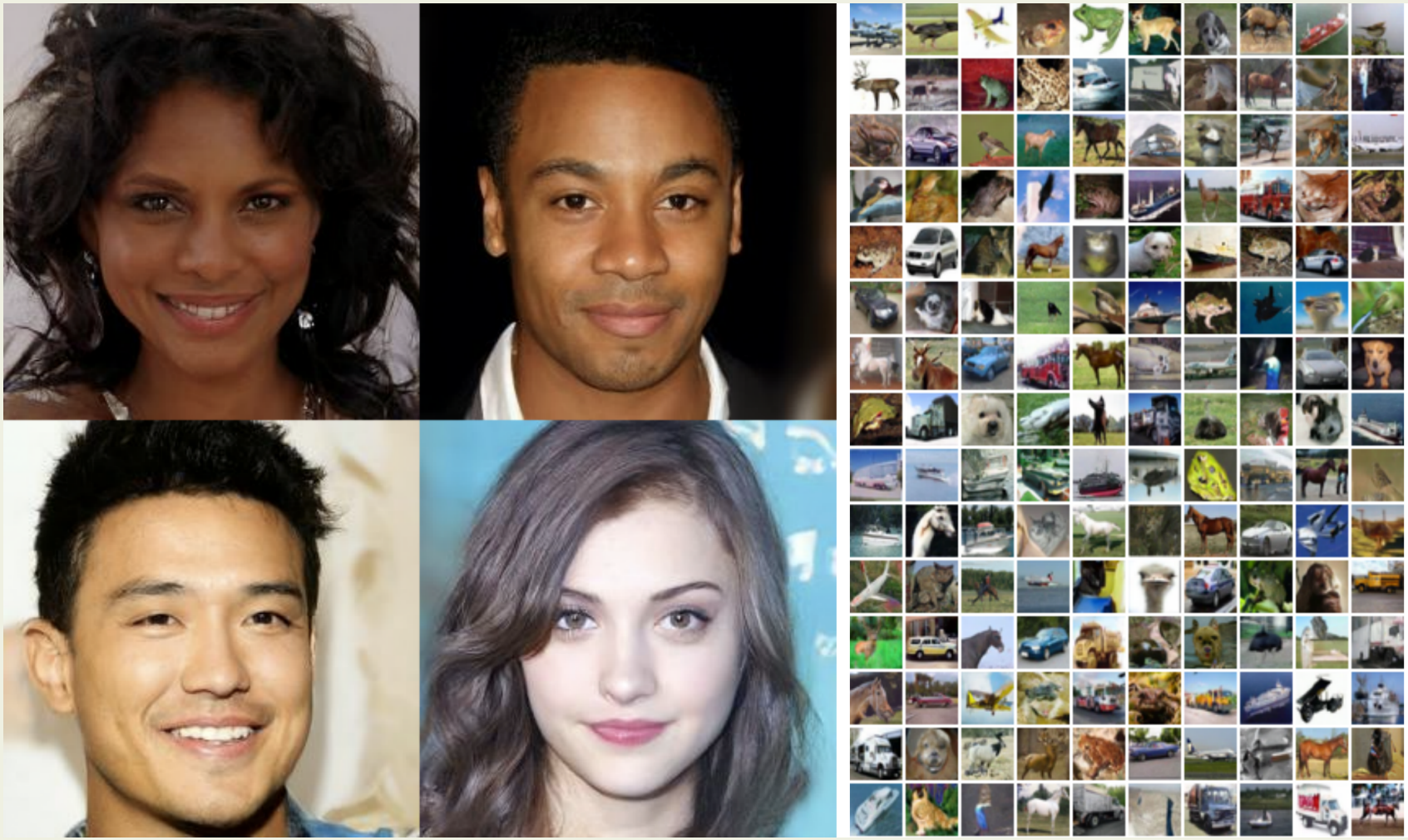
$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{\alpha_t(1 - \overline{\alpha}_t)}}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$$

3. Get denoised image $\mathbf{x}_{t-1} = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) + \widetilde{\beta}_t \cdot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$.

See https://arxiv.org/abs/2006.11239, **Ho J. Denoising Diffusion Probabilistic Models, 2020**

Denoising diffusion probabilistic model (DDPM)

Samples



See https://arxiv.org/abs/2006.11239, **Ho J. Denoising Diffusion Probabilistic Models, 2020**

Langevin dynamic

Imagine that we have some generative model $p(\mathbf{x}|\boldsymbol{\theta})$.

Statement

Let $\mathbf{x}_0$ be a random vector. Then under mild regularity conditions for small enough η samples from the following dynamics

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \eta\frac{1}{2}\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\boldsymbol{\theta}) + \sqrt{\eta}\cdot\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}).$$

will comes from $p(\mathbf{x}|\boldsymbol{\theta})$.

What do we get if $\boldsymbol{\epsilon} = \mathbf{0}$?

Energy-based model

$$p(\mathbf{x}|\boldsymbol{\theta}) = \frac{\widehat{p}(\mathbf{x}|\boldsymbol{\theta})}{Z_{\boldsymbol{\theta}}}, \quad \text{where } Z_{\boldsymbol{\theta}} = \int \widehat{p}(\mathbf{x}|\boldsymbol{\theta})d\mathbf{x}$$

$$\nabla_{\mathbf{x}}\log p(\mathbf{x}|\boldsymbol{\theta}) = \nabla_{\mathbf{x}}\log\widehat{p}(\mathbf{x}|\boldsymbol{\theta}) - \nabla_{\mathbf{x}}\log Z_{\boldsymbol{\theta}} = \nabla_{\mathbf{x}}\log\widehat{p}(\mathbf{x}|\boldsymbol{\theta})$$

Gradient of normalized density equals to gradient of unnormalized density.

See https://www.stats.ox.ac.uk/~teh/research/compstats/WelTeh2011a.pdf, **Welling M. Bayesian Learning via Stochastic Gradient Langevin Dynamics, 2011**

Stochastic differential equation (SDE)

Let define stochastic process $\mathbf{x}(t)$ with initial condition $\mathbf{x}(0) \sim p_0(\mathbf{x})$:

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

- $\mathbf{f}(\mathbf{x}, t)$ is the **drift** function of $\mathbf{x}(t)$.
- $g(t)$ is the **diffusion** coefficient of $\mathbf{x}(t)$.
- If $g(t) = 0$ we get standard ODE.
- $\mathbf{w}(t)$ is the standard Wiener process (Brownian motion):
  1. $\mathbf{w}(0) = 0$ (almost surely);
  2. $\mathbf{w}(t)$ has independent increments;
  3. $\mathbf{w}(t) - \mathbf{w}(s) \sim \mathcal{N}(0, t - s)$.
- $d\mathbf{w} = \mathbf{w}(t + dt) - \mathbf{w}(t) = \mathcal{N}(0, dt) = \boldsymbol{\epsilon} \cdot \sqrt{dt}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$.

Note: In contrast to ODE, initial condition $\mathbf{x}(0)$ does not uniquely determine the process trajectory.

Stochastic differential equation (SDE)

$$d\mathbf{x} = \mathbf{f}(\mathbf{x}, t)dt + g(t)d\mathbf{w}, \quad d\mathbf{w} = \boldsymbol{\epsilon} \cdot \sqrt{dt}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}).$$

- At each moment t we have the density $p(\mathbf{x}(t), t)$.
- How to get distribution $p(\mathbf{x}, t)$ for $\mathbf{x}(t)$?

Theorem (Kolmogorov-Fokker-Planck)

Evolution of the distribution $p(\mathbf{x}, t)$ is given by the following ODE:

$$\frac{\partial p(\mathbf{x}, t)}{\partial t} = \text{tr}\left(-\frac{\partial}{\partial \mathbf{x}}\left[\mathbf{f}(\mathbf{x}, t)p(\mathbf{x}, t)\right] + \frac{1}{2}g^2(t)\frac{\partial^2 p(\mathbf{x}, t)}{\partial \mathbf{x}^2}\right)$$

Note: This is the generalization of KFP theorem that we used in continuous-in-time NF.

Langevin SDE (special case)

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

$$d\mathbf{x} = \frac{1}{2}\frac{\partial}{\partial \mathbf{x}}\log p(\mathbf{x}, t)dt + 1 \cdot d\mathbf{w}$$

Langevin SDE (special case)

$$d\mathbf{x} = \frac{1}{2}\frac{\partial}{\partial \mathbf{x}} \log p(\mathbf{x}, t) dt + 1 \cdot d\mathbf{w}$$

Let apply KFP theorem.

$$\frac{\partial p(\mathbf{x}, t)}{\partial t} = \text{tr}\left(-\frac{\partial}{\partial \mathbf{x}} \left[p(\mathbf{x}, t) \frac{1}{2} \frac{\partial}{\partial \mathbf{x}} \log p(\mathbf{x}, t) \right] + \frac{1}{2} \frac{\partial^2 p(\mathbf{x}, t)}{\partial \mathbf{x}^2} \right) =$$

$$= \text{tr}\left(-\frac{\partial}{\partial \mathbf{x}} \left[\frac{1}{2} \frac{\partial}{\partial \mathbf{x}} p(\mathbf{x}, t) \right] + \frac{1}{2} \frac{\partial^2 p(\mathbf{x}, t)}{\partial \mathbf{x}^2} \right) = 0$$

The density $p(\mathbf{x}, t) = \text{const}(t)$!

Discretized Langevin SDE

$$\mathbf{x}_{t+1} - \mathbf{x}_t = \eta \frac{1}{2} \frac{\partial}{\partial \mathbf{x}} \log p(\mathbf{x}, t) + \sqrt{\eta} \cdot \boldsymbol{\epsilon}, \quad \eta \approx dt.$$

Langevin dynamic

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \eta \frac{1}{2} \nabla_{\mathbf{x}} \log p(\mathbf{x}|\boldsymbol{\theta}) + \sqrt{\eta} \cdot \boldsymbol{\epsilon}, \quad \eta \approx dt.$$

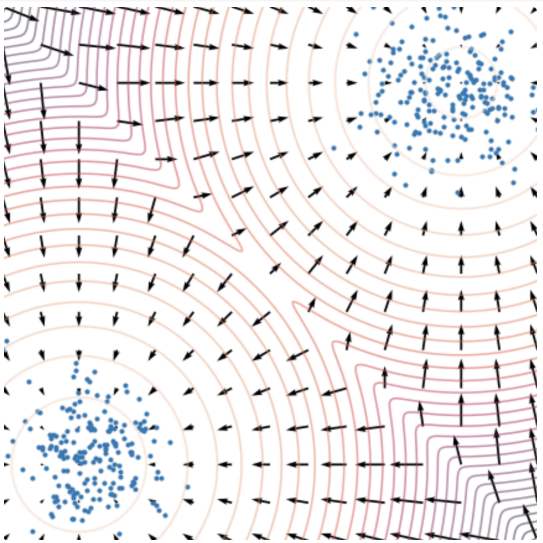
Statement

Let $\mathbf{x}_0$ be a random vector. Then samples from the following dynamics

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \eta \frac{1}{2} \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\boldsymbol{\theta}) + \sqrt{\eta} \cdot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}).$$

will come from $p(\mathbf{x}|\boldsymbol{\theta})$ under mild regularity conditions for small enough η and large enough t.

The density $p(\mathbf{x}|\boldsymbol{\theta})$ is a **stationary** distribution for this SDE.



See https://yang-song.github.io/blog/2021/score/, **Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021**

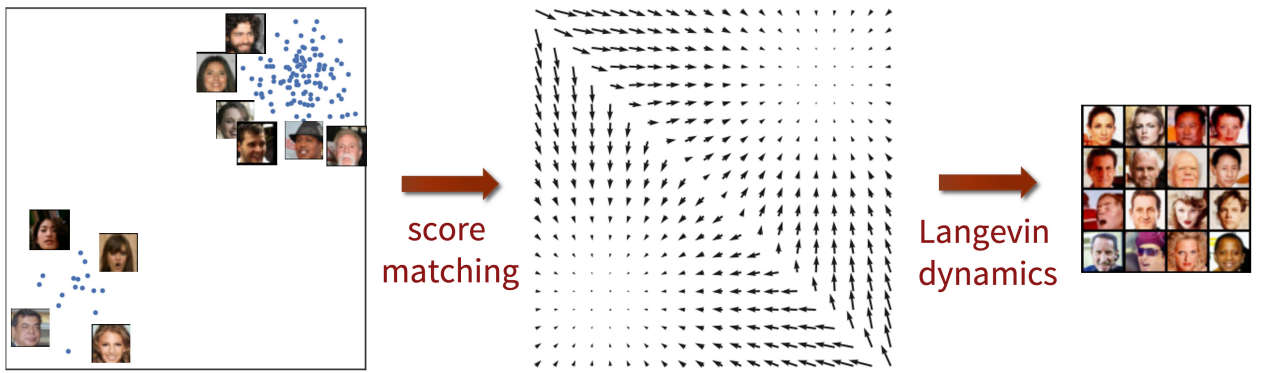
Score matching

We could sample from the model using Langevin dynamics if we have $\nabla_{\mathbf{x}} \log p(\mathbf{x}|\boldsymbol{\theta})$.

Fisher divergence

$$D_F(\pi, p) = \frac{1}{2}\mathbb{E}_{\pi}\left\|\nabla_{\mathbf{x}} \log p(\mathbf{x}|\boldsymbol{\theta}) - \nabla_{\mathbf{x}} \log \pi(\mathbf{x})\right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Let introduce **score function** $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) = \nabla_{\mathbf{x}} \log p(\mathbf{x}|\boldsymbol{\theta})$.



Problem: we do not know $\nabla_{\mathbf{x}} \log \pi(\mathbf{x})$.

See https://yang-song.github.io/blog/2021/score/, **Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021**

Implicit score matching

Theorem

Under some regularity conditions, it holds

$$\frac{1}{2}\mathbb{E}_{\pi}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}}\log\pi(\mathbf{x})\right\|_2^2 = \mathbb{E}_{\pi}\left[\frac{1}{2}\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2 + \operatorname{tr}\left(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\right)\right] + \text{const}$$

Proof (only for 1D)

$$\mathbb{E}_{\pi}\left\|s(x) - \nabla_x\log\pi(x)\right\|_2^2 = \mathbb{E}_{\pi}\left[s(x)^2 + (\nabla_x\log\pi(x))^2 - 2[s(x)\nabla_x\log\pi(x)]\right]$$

$$\mathbb{E}_{\pi}[s(x)\nabla_x\log\pi(x)] = \int \pi(x)s(x)\nabla_x\log\pi(x)dx = \int \underbrace{\nabla_x\log p(x)}_{g}\underbrace{\nabla_x\pi(x)}_{\nabla f}dx$$

$$= \underbrace{\nabla_x\log p(x)}_{g}\underbrace{\pi(x)}_{f}\Big|_{-\infty}^{+\infty} - \int \underbrace{\nabla_x\left(\nabla_x\log p(x)\right)}_{\nabla g}\underbrace{\pi(x)}_{f}dx$$

$$= -\mathbb{E}_{\pi}\nabla_x s(x)$$

$$\frac{1}{2}\mathbb{E}_{\pi}\left\|s(x) - \nabla_x\log\pi(x)\right\|_2^2 = \mathbb{E}_{\pi}\left[\frac{1}{2}s(x)^2 + \nabla_x s(x)\right] + \text{const.}$$

See https://jmlr.org/papers/volume6/hyvarinen05a/old.pdf, **Hyvarinen A. Estimation of non-normalized statistical models by score matching, 2005**

Score matching

Theorem (implicit score matching)

$$\frac{1}{2}\mathbb{E}_{\pi}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}}\log\pi(\mathbf{x})\right\|_2^2 = \mathbb{E}_{\pi}\left[\frac{1}{2}\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|_2^2 + \mathrm{tr}\big(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\big)\right] + \text{const}$$

Here $\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) = \nabla_{\mathbf{x}}^2\log p(\mathbf{x}|\boldsymbol{\theta})$ is a Hessian matrix.

1. The right hand side is complex due to Hessian matrix – **sliced score matching**.
2. The left hand side is intractable due to unknown $\pi(\mathbf{x})$ – **denoising score matching**.

Sliced score matching (Hutchinson's trace estimation)

$$\mathrm{tr}\big(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\big) = \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[\boldsymbol{\epsilon}^T\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\boldsymbol{\epsilon}\right]$$

See https://yang-song.net/blog/2021/ssm/, **Song Y. Sliced Score Matching: A Scalable Approach to Density and Score Estimation, 2019**; https://yang-song.net/blog/2021/score/, **Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021**

Denoising score matching

Let perturb original data $\mathbf{x} \sim \pi(\mathbf{x})$ by random normal noise

$$\mathbf{x}' = \mathbf{x} + \sigma \cdot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}), \quad p(\mathbf{x}'|\mathbf{x}, \sigma) = \mathcal{N}(\mathbf{x}'|\mathbf{x}, \sigma^2\mathbf{I})$$

$$\pi(\mathbf{x}'|\sigma) = \int p(\mathbf{x}'|\mathbf{x}, \sigma)\pi(\mathbf{x})d\mathbf{x}.$$

Assumption

The solution of

$$\frac{1}{2}\mathbb{E}_{\pi(\mathbf{x}'|\sigma)}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma) - \nabla_{\mathbf{x}'}\log\pi(\mathbf{x}'|\sigma)\right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

satisfies $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma) \approx \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', 0) = \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$ if σ is small enough.

- $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma)$ tries to **denoise** a corrupted sample $\mathbf{x}'$.
- Score function $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma)$ parametrized by σ.

See `http://www.iro.umontreal.ca/~vincentp/Publications/smdae_techreport.pdf`, **Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010**

Denoising score matching

Theorem

$$\mathbb{E}_{\pi(\mathbf{x}'|\sigma)} \left\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma) - \nabla_{\mathbf{x}'} \log \pi(\mathbf{x}'|\sigma) \right\|_2^2 =$$

$$= \mathbb{E}_{\pi(\mathbf{x})} \mathbb{E}_{p(\mathbf{x}'|\mathbf{x},\sigma)} \left\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma) - \nabla_{\mathbf{x}'} \log p(\mathbf{x}'|\mathbf{x}, \sigma) \right\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof

$$\mathbb{E}_{\pi(\mathbf{x}'|\sigma)} \left\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma) - \nabla_{\mathbf{x}'} \log \pi(\mathbf{x}'|\sigma) \right\|_2^2 =$$

$$= \mathbb{E}_{\pi(\mathbf{x}'|\sigma)} \left[\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma)\|^2 + \underbrace{\|\nabla_{\mathbf{x}'} \log \pi(\mathbf{x}'|\sigma)\|_2^2}_{\text{const}(\boldsymbol{\theta})} - 2\mathbf{s}_{\boldsymbol{\theta}}^T(\mathbf{x}', \sigma) \nabla_{\mathbf{x}'} \log \pi(\mathbf{x}'|\sigma) \right]$$

$$\mathbb{E}_{\pi(\mathbf{x}'|\sigma)} \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma)\|^2 = \int \pi(\mathbf{x}'|\sigma) \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma)\|^2 d\mathbf{x}' =$$

$$= \int \left(\int p(\mathbf{x}'|\mathbf{x}, \sigma) \pi(\mathbf{x}) d\mathbf{x} \right) \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma)\|^2 d\mathbf{x}' = \mathbb{E}_{\pi(\mathbf{x})} \mathbb{E}_{p(\mathbf{x}'|\mathbf{x},\sigma)} \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma)\|^2 d\mathbf{x}'$$

See http://www.iro.umontreal.ca/~vincentp/Publications/smdae_techreport.pdf, **Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010**

Denoising score matching

Theorem

$$\mathbb{E}_{\pi(\mathbf{x}'|\sigma)} \left\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma) - \nabla_{\mathbf{x}'} \log \pi(\mathbf{x}'|\sigma) \right\|_2^2 =$$

$$= \mathbb{E}_{\pi(\mathbf{x})} \mathbb{E}_{p(\mathbf{x}'|\mathbf{x},\sigma)} \left\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma) - \nabla_{\mathbf{x}'} \log p(\mathbf{x}'|\mathbf{x}, \sigma) \right\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof (continued)

$$\mathbb{E}_{\pi(\mathbf{x}'|\sigma)} \left[\mathbf{s}_{\boldsymbol{\theta}}^T(\mathbf{x}', \sigma) \nabla_{\mathbf{x}'} \log \pi(\mathbf{x}'|\sigma) \right] = \int \pi(\mathbf{x}'|\sigma) \left[\mathbf{s}_{\boldsymbol{\theta}}^T(\mathbf{x}', \sigma) \frac{\nabla_{\mathbf{x}'} \pi(\mathbf{x}'|\sigma)}{\pi(\mathbf{x}'|\sigma)} \right] d\mathbf{x}' =$$

$$= \int \left[\mathbf{s}_{\boldsymbol{\theta}}^T(\mathbf{x}', \sigma) \nabla_{\mathbf{x}'} \left(\int p(\mathbf{x}'|\mathbf{x}, \sigma) \pi(\mathbf{x}) d\mathbf{x} \right) \right] d\mathbf{x}' =$$

$$= \int \int \pi(\mathbf{x}) \left[\mathbf{s}_{\boldsymbol{\theta}}^T(\mathbf{x}', \sigma) \nabla_{\mathbf{x}'} p(\mathbf{x}'|\mathbf{x}, \sigma) \right] d\mathbf{x}' d\mathbf{x} =$$

$$= \int \int \pi(\mathbf{x}) p(\mathbf{x}'|\mathbf{x}, \sigma) \left[\mathbf{s}_{\boldsymbol{\theta}}^T(\mathbf{x}', \sigma) \nabla_{\mathbf{x}'} \log p(\mathbf{x}'|\mathbf{x}, \sigma) \right] d\mathbf{x}' d\mathbf{x} =$$

$$= \mathbb{E}_{\pi(\mathbf{x})} \mathbb{E}_{p(\mathbf{x}'|\mathbf{x},\sigma)} \left[\mathbf{s}_{\boldsymbol{\theta}}^T(\mathbf{x}', \sigma) \nabla_{\mathbf{x}'} \log p(\mathbf{x}'|\mathbf{x}, \sigma) \right]$$

See http://www.iro.umontreal.ca/~vincentp/Publications/smdae_techreport.pdf, **Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010**

Denoising score matching

Theorem

$$\mathbb{E}_{\pi(\mathbf{x}'|\sigma)} \left\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma) - \nabla_{\mathbf{x}'} \log \pi(\mathbf{x}'|\sigma) \right\|_2^2 =$$

$$= \mathbb{E}_{\pi(\mathbf{x})} \mathbb{E}_{p(\mathbf{x}'|\mathbf{x},\sigma)} \left\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma) - \nabla_{\mathbf{x}'} \log p(\mathbf{x}'|\mathbf{x}, \sigma) \right\|_2^2 + \text{const}(\boldsymbol{\theta})$$

Proof (continued)

$$\mathbb{E}_{\pi(\mathbf{x}'|\sigma)} \left\| \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma) - \nabla_{\mathbf{x}'} \log \pi(\mathbf{x}'|\sigma) \right\|_2^2 =$$

$$= \mathbb{E}_{\pi(\mathbf{x}'|\sigma)} \left[\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma)\|^2 - 2\mathbf{s}_{\boldsymbol{\theta}}^T(\mathbf{x}', \sigma) \nabla_{\mathbf{x}'} \log \pi(\mathbf{x}'|\sigma) \right] + \text{const}(\boldsymbol{\theta}) =$$

$$= \mathbb{E}_{\pi(\mathbf{x})} \mathbb{E}_{p(\mathbf{x}'|\mathbf{x},\sigma)} \left[\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}', \sigma)\|^2 - 2\mathbf{s}_{\boldsymbol{\theta}}^T(\mathbf{x}', \sigma) \nabla_{\mathbf{x}'} \log p(\mathbf{x}'|\mathbf{x}, \sigma) \right] + \text{const}(\boldsymbol{\theta})$$

Gradient of the noise kernel

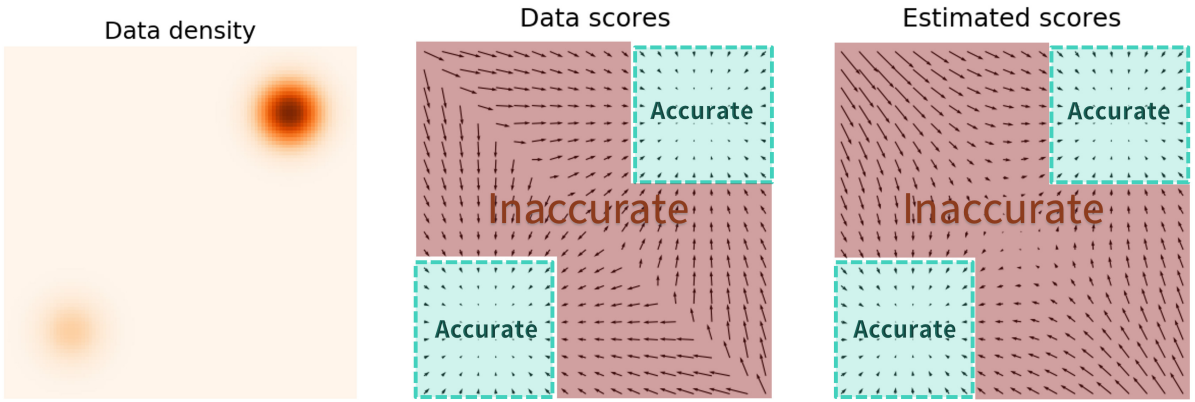
$$\nabla_{\mathbf{x}'} \log p(\mathbf{x}'|\mathbf{x}, \sigma) = \nabla_{\mathbf{x}'} \log \mathcal{N}(\mathbf{x}'|\mathbf{x}, \sigma^2 \mathbf{I}) = -\frac{\mathbf{x}' - \mathbf{x}}{\sigma^2}$$

The RHS does not need to compute $\nabla_{\mathbf{x}'} \log \pi(\mathbf{x}'|\sigma)$ and even $\nabla_{\mathbf{x}'} \log \pi(\mathbf{x}')$.

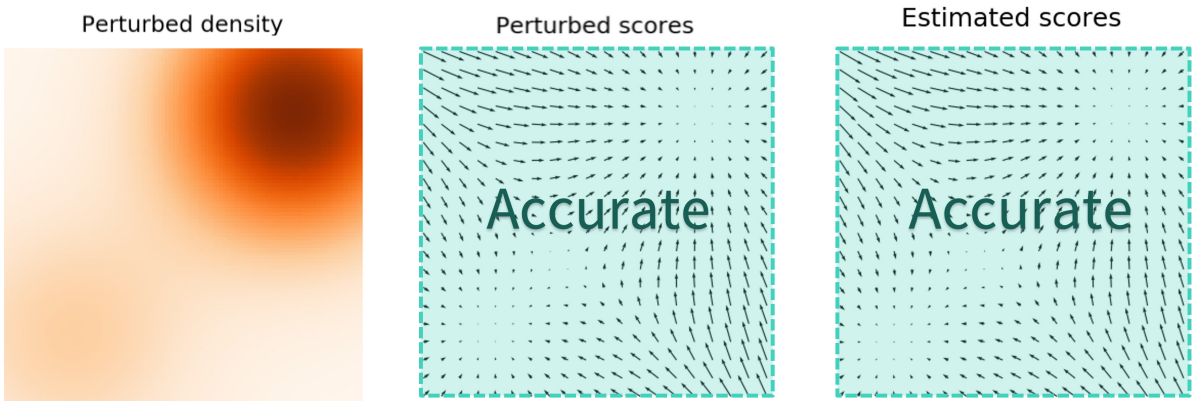
See `http://www.iro.umontreal.ca/~vincentp/Publications/smdae_techreport.pdf`, **Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010**

Denoising score matching

- If σ is **small**, the score function is not accurate and Langevin dynamics will probably fail to jump between modes.



- If σ is **large**, it is good for low-density regions and multimodal distributions, but we will learn too corrupted distribution.



See https://yang-song.github.io/blog/2021/score/, **Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021**

Summary

- DDPM is a VAE model with hierarchical latent variables.
- At each step DDPM predicts the noise that was used in the forward diffusion process.
- Langevin dynamics allows to sample from the energy-based model using the score function (due to the existence of stationary distribution of SDE).
- Score matching proposes to minimize Fisher divergence to get score function.
- Implicit score matching tries to avoid the value of original distribution $\pi(\mathbf{x})$. Sliced score matching makes implicit score matching scalable.
- Denoising score matching minimizes Fisher divergence on noisy samples.

References

- Current lecture is mainly a compilation of materials from the course of Roman Isachenko, Deep Generative Models course, MIPT, 2023
- http://proceedings.mlr.press/v37/sohl-dickstein15.pdf, Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015
- https://ayandas.me/blog-tut/2021/12/04/diffusion-prob-models.html, Das A. An introduction to Diffusion Probabilistic Models, blog post, 2021
- https://arxiv.org/abs/2112.07804, Xiao Z., Kreis K., Vahdat A. Tackling the generative learning trilemma with denoising diffusion GANs, 2021
- https://lilianweng.github.io/posts/2021-07-11-diffusion-models/, Weng L. What are Diffusion Models?, blog post, 2021
- https://arxiv.org/abs/2006.11239, Ho J. Denoising Diffusion Probabilistic Models, 2020
- https://arxiv.org/abs/2208.11970, Luo C. Understanding Diffusion Models: A Unified Perspective, 2022
- https://www.stats.ox.ac.uk/~teh/research/compstats/WelTeh2011a.pdf, Welling M. Bayesian Learning via Stochastic Gradient Langevin Dynamics, 2011
- https://yang-song.github.io/blog/2021/score/, Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021
- https://jmlr.org/papers/volume6/hyvarinen05a/old.pdf, Hyvarinen A. Estimation of non-normalized statistical models by score matching, 2005
- https://yang-song.net/blog/2021/ssm/, Song Y. Sliced Score Matching: A Scalable Approach to Density and Score Estimation, 2019
- http://www.iro.umontreal.ca/~vincentp/Publications/smdae_techreport.pdf, Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010