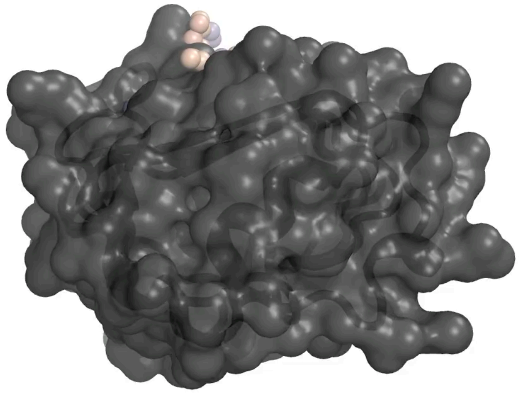
Outline:

- Denoising Diffusion Probabilistic model
- Score-based model
- Connections between DDPM and Score-based model
- Conditional generation

Dall-E

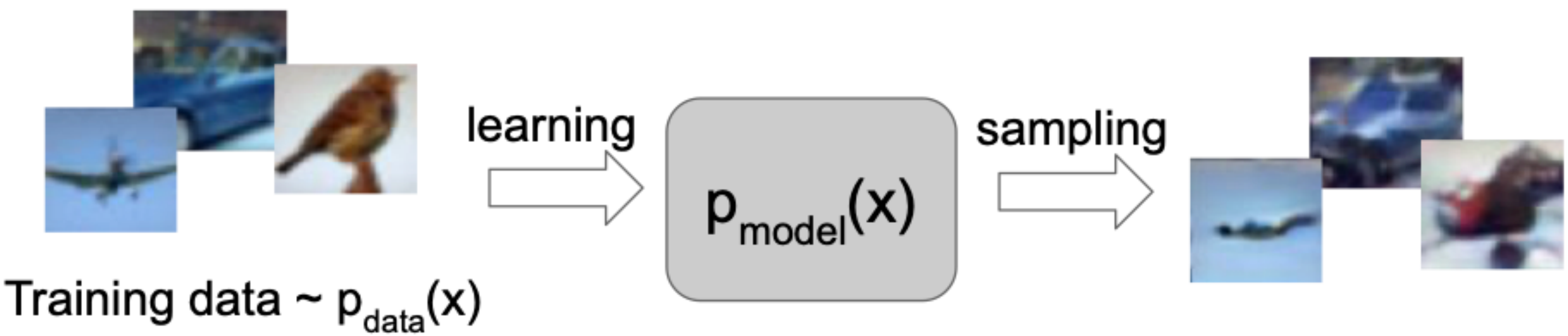


RFdiffusion

DDPM: Denoising Diffusion Probabilistic models

NIPS'20

- DDPM is a generative model



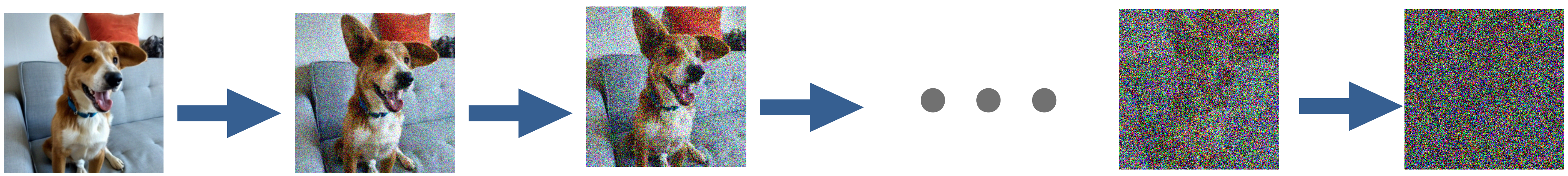
Objectives:

1. Learn $p_{model}(x)$ that approximates $p_{data}(x)$
2. **Sampling new x from $p_{model}(x)$**

Two processes in DDPM: diffusion and denoising

Diffusion

$$p(\mathbf{x}^{(0)}) \longrightarrow p(\mathbf{x}^{(1)}) \longrightarrow \cdots \longrightarrow p(\mathbf{x}^{(T)})$$



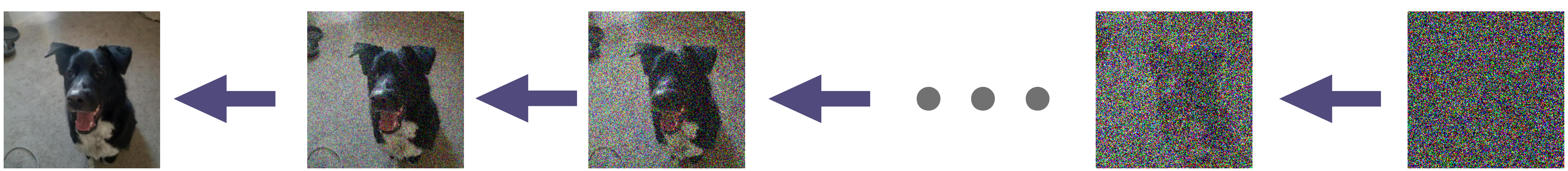
$\sim p(\mathbf{x}^{(0)})$ $\sim p(\mathbf{x}^{(1)})$ $\sim p(\mathbf{x}^{(2)})$ $\sim p(\mathbf{x}^{(T-1)})$ $\sim \mathcal{N}(\mathbf{x}^{(t)}; \mathbf{0}, \mathbf{I})$

<u>Easier to sample noise</u>

<u>But how to convert noise back to dogs?</u>

Reverse (denoising)

$$q(\mathbf{x}^{(0)}) \longleftarrow q(\mathbf{x}^{(1)}) \longleftarrow \cdots \longleftarrow q(\mathbf{x}^{(T)})$$



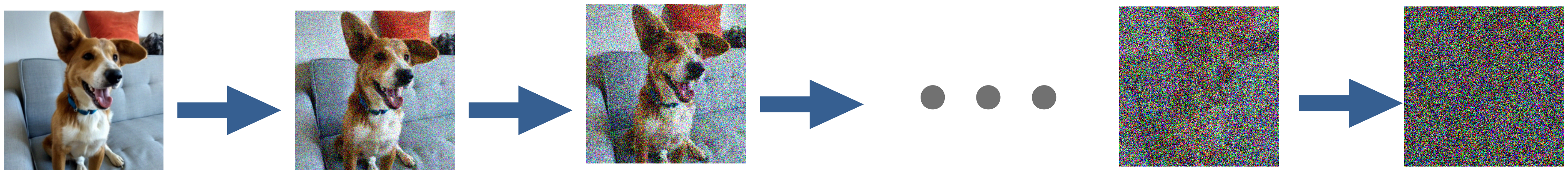
$\sim q(\mathbf{x}^{(0)})$ $\sim q(\mathbf{x}^{(1)})$ $\sim q(\mathbf{x}^{(2)})$ $\sim q(\mathbf{x}^{(T-1)})$ $\sim p(\mathbf{x}^{(T)})$

Backward diffusion kernel: $q_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$

Sohl-Dickstein, Surya Ganguli and co. "Deep unsupervised learning using nonequilibrium thermodynamics." *ICML*. PMLR, 2015.
Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." *arXiv:2006.11239,* 2020.

Diffusion

$$q(\mathbf{x}^{(0)}) \longrightarrow q(\mathbf{x}^{(1)}) \longrightarrow \cdots \longrightarrow q(\mathbf{x}^{(T)})$$



$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$\beta : 10^{-4} \rightarrow 2 \times 10^{-2} \text{ linearly changing} \quad T = 2000$$

$$\xrightarrow{\alpha_i = 1-\beta_i} \quad \mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad \bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$$

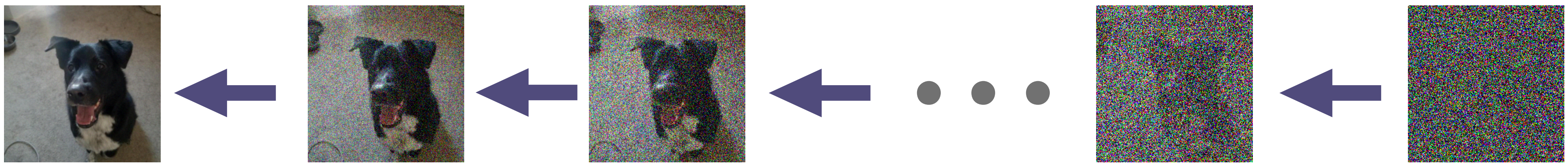
$$\mathbf{x}_T = \sqrt{\bar{\alpha}_T}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_T}\epsilon$$

$$\bar{\alpha}_T \approx 0$$

Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." *arXiv:2006.11239,* 2020.

Reverse (denoising)

$$q(\mathbf{x}^{(0)}) \longleftarrow q(\mathbf{x}^{(1)}) \longleftarrow \cdots \longleftarrow q(\mathbf{x}^{(T)})$$



$\sim q(\mathbf{x}^{(0)})$ $\sim q(\mathbf{x}^{(1)})$ $\sim q(\mathbf{x}^{(2)})$ $\sim q(\mathbf{x}^{(T-1)})$ $\sim p(\mathbf{x}^{(T)})$

Backward diffusion kernel: $q_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_\theta(\mathbf{x}_t, t))$

$p(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \dfrac{p(\mathbf{x}_t \mid \mathbf{x}_{t-1})p(\mathbf{x}_{t-1})}{p(\mathbf{x}_t)}$ is not analytically tractable

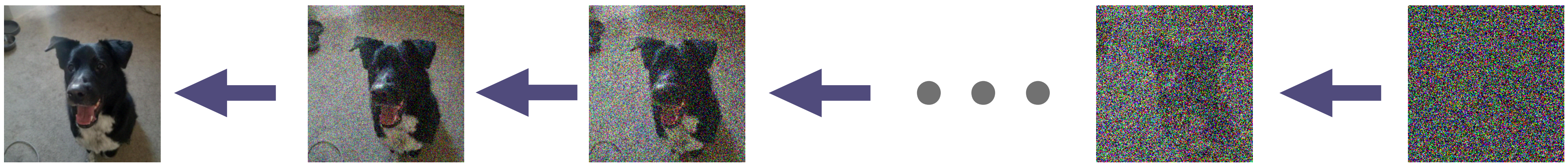
However, the reverse process conditioned on $\mathbf{x}_0$ is tractable:

$$p(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \frac{p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{x}_0)p(\mathbf{x}_{t-1} \mid \mathbf{x}_0)}{p(\mathbf{x}_t \mid \mathbf{x}_0)} = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_0, \mathbf{x}_t, t), \sigma_\theta(\mathbf{x}_0, \mathbf{x}_t, t))$$

Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." *arXiv:2006.11239,* 2020.

Reverse (denoising)

$$q(\mathbf{x}^{(0)}) \longleftarrow q(\mathbf{x}^{(1)}) \longleftarrow \cdots \longleftarrow q(\mathbf{x}^{(T)})$$



$\sim q(\mathbf{x}^{(0)})$ $\sim q(\mathbf{x}^{(1)})$ $\sim q(\mathbf{x}^{(2)})$ $\sim q(\mathbf{x}^{(T-1)})$ $\sim p(\mathbf{x}^{(T)})$

$$p(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_0, \mathbf{x}_t, t), \sigma_\theta(\mathbf{x}_0, \mathbf{x}_t, t))$$

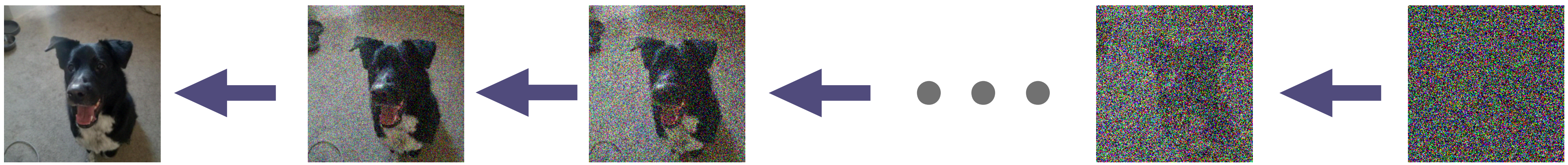
$$\sigma^2(\mathbf{x}_0, \mathbf{x}_t, t) = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \cdot \beta_t$$

$$\mu(\mathbf{x}_0, \mathbf{x}_t, t) = \sqrt{\beta_t} \cdot \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t}\mathbf{x}_t + \sqrt{\beta_t} \cdot \frac{\bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t}\mathbf{x}_0$$

Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." *arXiv:2006.11239,* 2020.

Reverse (denoising)

$$q(\mathbf{x}^{(0)}) \longleftarrow q(\mathbf{x}^{(1)}) \longleftarrow \cdots \longleftarrow q(\mathbf{x}^{(T)})$$



$\sim q(\mathbf{x}^{(0)})$ $\sim q(\mathbf{x}^{(1)})$ $\sim q(\mathbf{x}^{(2)})$ $\sim q(\mathbf{x}^{(T-1)})$ $\sim p(\mathbf{x}^{(T)})$

$$\mu(\mathbf{x}_0, \mathbf{x}_t, t) = \sqrt{\beta_t} \cdot \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\mathbf{x}_t + \sqrt{\beta_t} \cdot \frac{\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\mathbf{x}_0$$

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$$

$$\mathbf{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}\mathbf{x}_t - \frac{\sqrt{1-\bar{\alpha}_t}\tilde{\epsilon}}{\sqrt{\bar{\alpha}_t}}$$

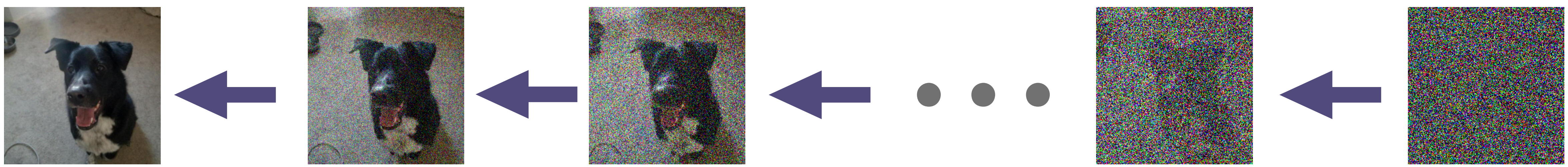
Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
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

Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." *arXiv:2006.11239,* 2020.

Reverse (denoising)

$$q(\mathbf{x}^{(0)}) \longleftarrow q(\mathbf{x}^{(1)}) \longleftarrow \cdots \longleftarrow q(\mathbf{x}^{(T)})$$



$\sim q(\mathbf{x}^{(0)})$ $\sim q(\mathbf{x}^{(1)})$ $\sim q(\mathbf{x}^{(2)})$ $\sim q(\mathbf{x}^{(T-1)})$ $\sim p(\mathbf{x}^{(T)})$

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\tilde{\epsilon}\right) + \sqrt{\frac{\beta_t}{1-\bar{\alpha}_t}}\beta_t\epsilon$$

$$\tilde{\epsilon} = \mathrm{Net}(\mathbf{x_t}, t) \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Algorithm 2 Sampling

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

Sohl-Dickstein, Surya Ganguli and co. "Deep unsupervised learning using nonequilibrium thermodynamics." *ICML*. PMLR, 2015.
Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." *arXiv:2006.11239,* 2020.

Summary

Diffusion

$$\mathbf{x}_t \sim \mathcal{N}(\sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I})$$

$$\mathbf{x}_t \sim \mathcal{N}(\sqrt{\alpha_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$$

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\epsilon_t$$

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$$

Reverse (denoising)

$$\mathbf{x}_{t-1} \sim \mathcal{N}(\mu, \sigma^2)$$

$$\sigma^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

$$\mu = \sqrt{\alpha_t}\left(\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}\beta_t}}{1-\bar{\alpha}_t}\mathbf{x}_0\right)$$

$$\mu = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\tilde{\epsilon}\right)$$

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\tilde{\epsilon}\right) + \sqrt{\frac{\beta_t}{1-\bar{\alpha}_t}}\beta_t\epsilon$$

Algorithm 1 Training

repeat
- $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
- $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
- $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
- Take gradient descent step on $\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$

until converged

Algorithm 2 Sampling

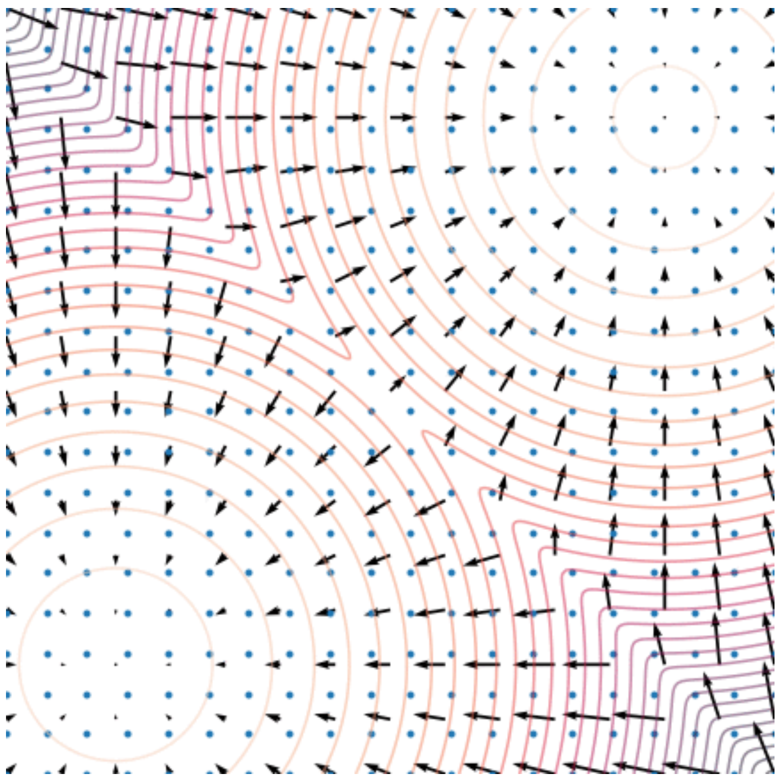
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

for $t = T, \ldots, 1$ **do**
- $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
- $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t\mathbf{z}$

end for

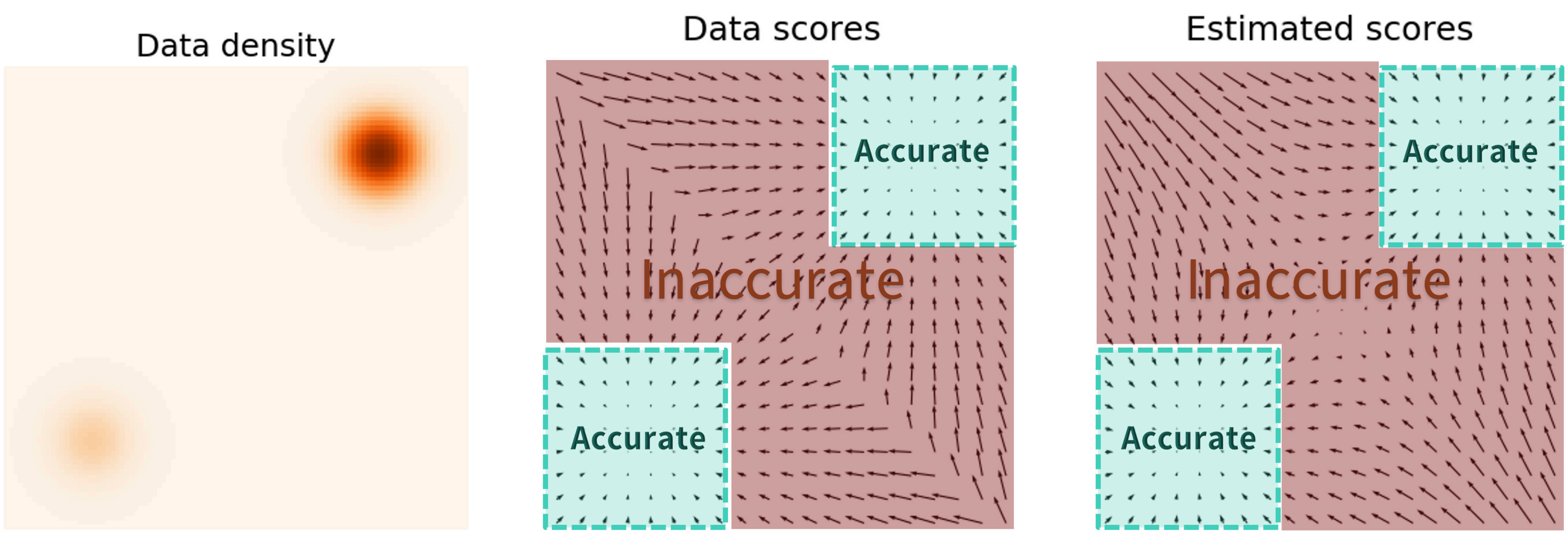
return $\mathbf{x}_0$

Score-Based Generative Modeling



- Model the gradient of the log probability density function (**score function**)
- Generate samples with Langevin-type sampling

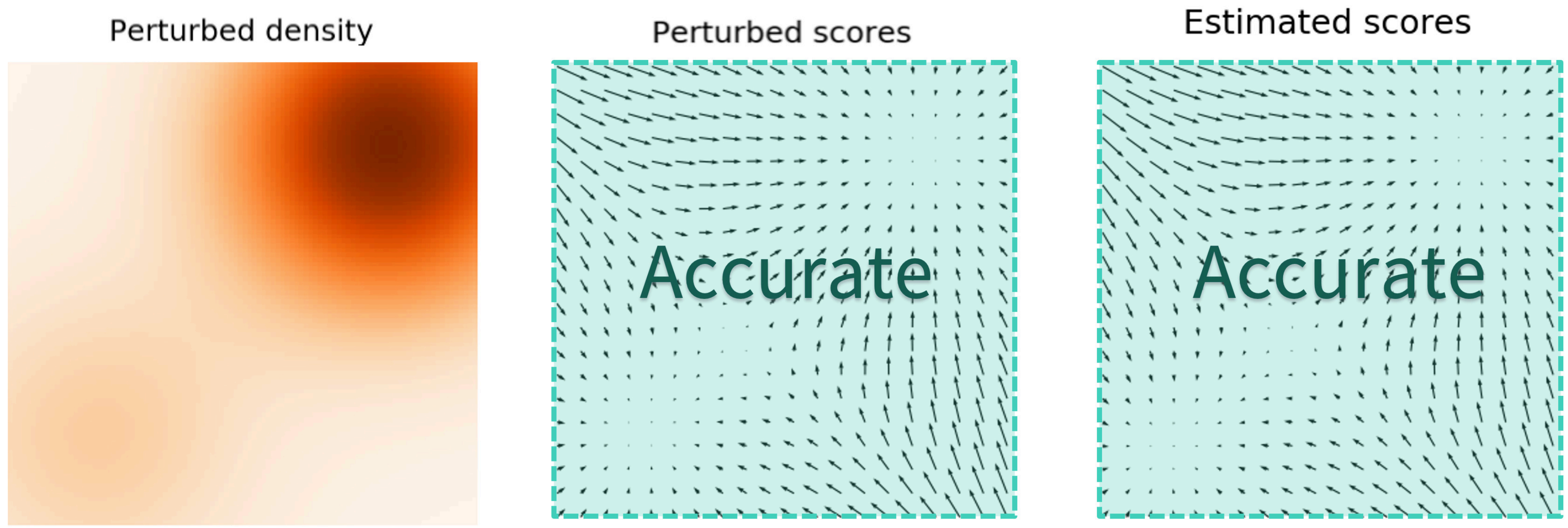
Challenge: An inaccurate score-based model will derail Langevin dynamics from the very beginning of the procedure



https://yang-song.net/blog/2021/score/
Song, Yang, and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution." *NIPS* (2019)

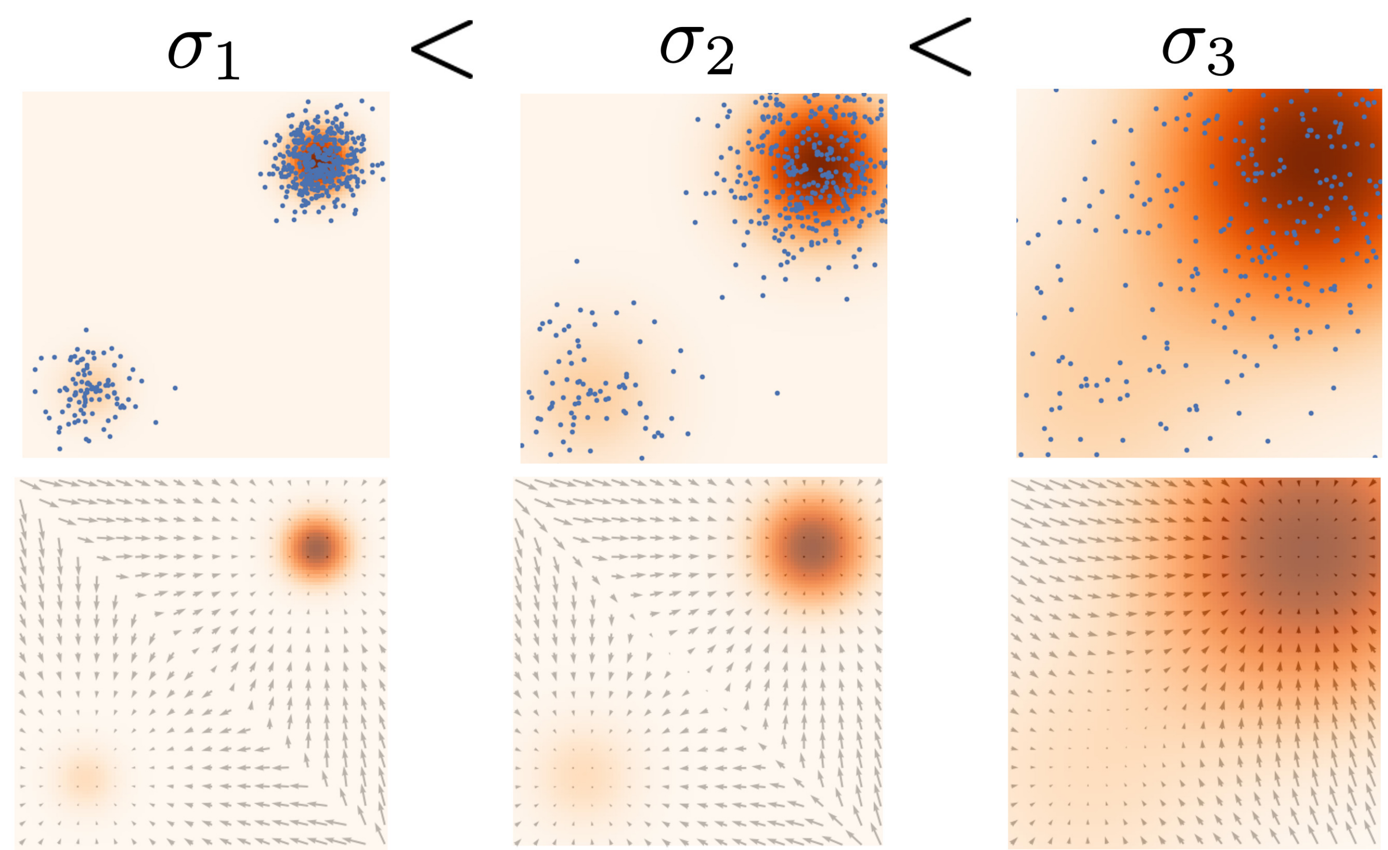
Score-Based Generative Modeling

perturb data points with noise and train score-based models on the noisy data points instead.

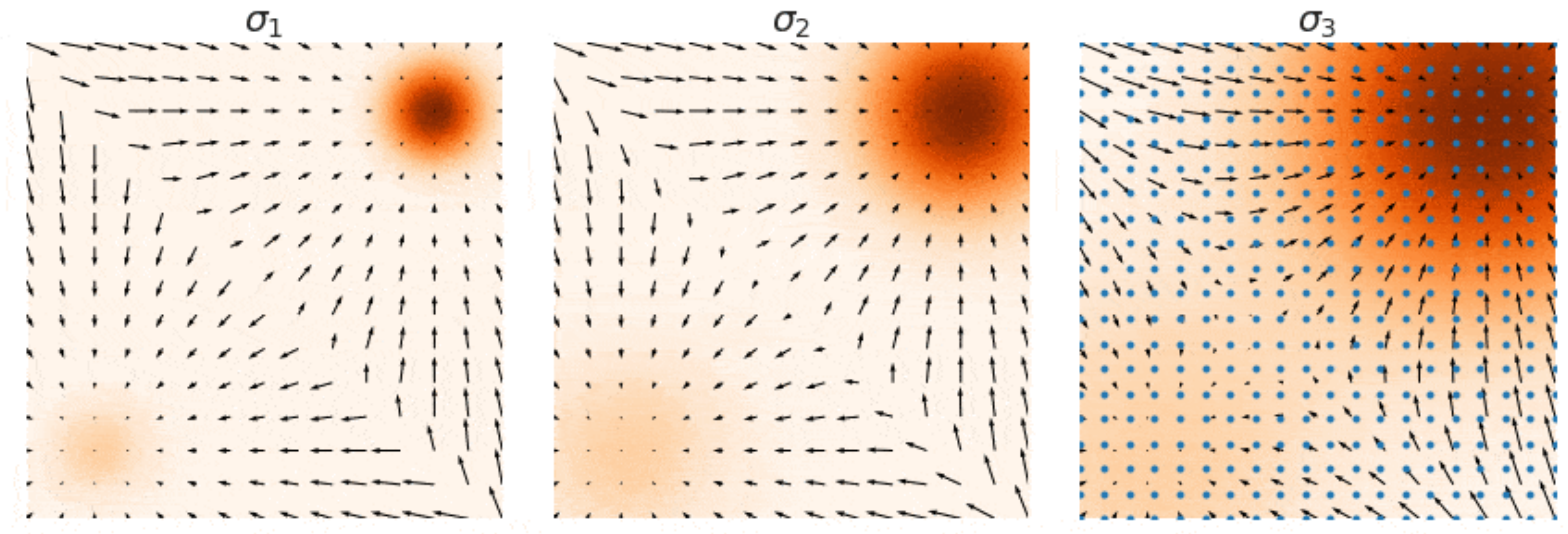


Larger noise:
- can cover more low-density regions
- over-corrupts the data

Smaller noise:
- less corruption of the original data distribution
- does not cover the low-density regions



Annealed Langevin Dynamics



Algorithm 1 Annealed Langevin dynamics.

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

Initialize $\tilde{\mathbf{x}}_0$
for $i \leftarrow 1$ to L **do**
    $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2 / \sigma_L^2$ $\triangleright$ 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 $\tilde{\mathbf{x}}_T$

- Tuning down the step size α_i gradually
- When T = 1, it becomes similar to to DDPM

https://yang-song.net/blog/2021/score/
Song, Yang, and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution." *NIPS* (2019)

Score-Based Generative Modeling

How to calculate the score?

$$\ell(\boldsymbol{\theta};\sigma) \triangleq \frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{\tilde{\mathbf{x}}\sim\mathcal{N}(\mathbf{x},\sigma^2 I)}\left[\left\|\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}},\sigma)+\frac{\tilde{\mathbf{x}}-\mathbf{x}}{\sigma^2}\right\|_2^2\right]$$

$$\tilde{\mathbf{x}} \sim \mathcal{N}(\mathbf{x},\sigma^2\mathbf{I})$$

$$p(\tilde{\mathbf{x}}) \propto \exp\left\{-\frac{(\tilde{\mathbf{x}}-\mathbf{x})^2}{2\sigma^2}\right\}$$

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

https://yang-song.net/blog/2021/score/

Song, Yang, and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution." *NIPS* (2019)

A unified view of DDPM and score-based model

$X_T, X_{T-1}, X_{T-2} \cdots X_1, X_0$ is a stochastic trajecctory $\longrightarrow$ stochastic process

SDE-based diffusion process

$$dx = f(x, t)\, dt + g(t)\, dw$$

$f(x, t)$: drift coefficient

$g(t)$: diffusion coefficient

w : Brownian motion

Discretized Form:

$$\mathbf{x}_{t+\Delta t} - \mathbf{x}_t = f(\mathbf{x}_t, t)\Delta t + g(t)\sqrt{\Delta t}\epsilon \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Song, Yang, et al. "Score-based generative modeling through stochastic differential equations." *arXiv:2011.13456* (2020).

A unified view of DDPM and score-based model

$$\mathbf{x}_{t+\Delta t} = \mathbf{x}_t + f(\mathbf{x}_t, t)\Delta t + g(t)\sqrt{\Delta t}\epsilon \longrightarrow \mathbf{x}_t \to \mathbf{x}_{t+\Delta t} \xrightarrow{?} \mathbf{x}_{t+\Delta t} \to \mathbf{x}_t$$

Reverse:

$$\mathbf{x}_{t+\Delta t} \to \mathbf{x}_t \qquad p(\mathbf{x}_t \mid \mathbf{x}_{t+\Delta t}) = \frac{p(\mathbf{x}_{t+\Delta t} \mid \mathbf{x}_t)\, p(\mathbf{x}_t)}{p(\mathbf{x}_{t+\Delta t})}$$

$$= p(\mathbf{x}_{t+\Delta t} \mid \mathbf{x}_t) \exp\left\{\log p(\mathbf{x}_t) - \log p(\mathbf{x}_{t+\Delta t})\right\}$$

$$\log p(\mathbf{x}_{t+\Delta t}) \approx \log p(\mathbf{x}_t) + (\mathbf{x}_{t+\Delta t} - \mathbf{x}_t)\, \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) + \Delta t \frac{\partial}{\partial t} \log p(\mathbf{x}_t)$$

$$p(\mathbf{x}_t \mid \mathbf{x}_{t+\Delta t}) \overset{\Delta t \to 0}{\approx} \exp\left\{ -\frac{1}{2g^2(t+\Delta t)\Delta t} \left\| (\mathbf{x}_{t+\Delta t} - \mathbf{x}_t) - \left(f(\mathbf{x}_{t+\Delta t}, t+\Delta t) - g^2(t+\Delta t)\, \nabla_{\mathbf{x}_{t+\Delta t}} \log p(\mathbf{x}_{t+\Delta t}) \Delta t \right) \right\|^2 \right\}$$

Song, Yang, et al. "Score-based generative modeling through stochastic differential equations." *arXiv:2011.13456* (2020).

A unified view of DDPM and score-based model

$$p(\mathbf{x}_t \mid \mathbf{x}_{t+\Delta t}) \approx \exp\left\{ -\frac{1}{2g^2(t+\Delta t)\Delta t} \left\| (\mathbf{x}_{t+\Delta t} - \mathbf{x}_t) - \Big(f(\mathbf{x}_{t+\Delta t}, t+\Delta t) - g^2(t+\Delta t)\, \nabla_{\mathbf{x}_{t+\Delta t}} \log p(\mathbf{x}_{t+\Delta t}) \Delta t \Big) \right\|^2 \right\}$$

$$\mathbf{x}_{t+\Delta t} - \mathbf{x}_t = \Big[f(\mathbf{x}_{t+\Delta t}, t+\Delta t) - g^2(t+\Delta t)\, \nabla_{\mathbf{x}_{t+\Delta t}} \log p(\mathbf{x}_{t+\Delta t}) \Big] \Delta t + g(t+\Delta t)\sqrt{\Delta t}\epsilon$$

$$d\mathbf{x} = \Big[f(\mathbf{x}, t) - g^2(t)\, \nabla_{\mathbf{x}} \log p(\mathbf{x}, t) \Big]\, dt + g(t) dw$$

$$\begin{cases} d\mathbf{x} = f(\mathbf{x}, t)\, dt + g(t)\, dw \\ d\mathbf{x} = \Big[f(\mathbf{x}, t) - g^2(t)\, \nabla_{\mathbf{x}} \log p(\mathbf{x}) \Big]\, dt + g(t)\, dw \end{cases}$$

Song, Yang, et al. "Score-based generative modeling through stochastic differential equations." *arXiv:2011.13456* (2020).

VE-SED <—> score-based model

(Variance Exploring)

$$x_t = x_0 + \sigma_t \epsilon$$

$$x_{t+1} = x_t + \sqrt{\sigma_{t+1}^2 - \sigma_t^2}\,\epsilon$$

VP-SED <—> DDPM

(Variance Preserving)

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\,\epsilon$$

$$x_{t+1} = \sqrt{1 - \beta_{t+1}} x_t + \sqrt{\beta_{t+1}}\,\epsilon$$

$$x_{t+\Delta t} - x_t = f(x_t, t)\Delta t + g(t)\sqrt{\Delta t}\epsilon$$

Song, Yang, et al. "Score-based generative modeling through stochastic differential equations." *arXiv:2011.13456* (2020).

VE-SED <—> score-based model

(Variance Exploring)

$$x_t = x_0 + \sigma_t \epsilon$$

$$x_{t+1} = x_t + \sqrt{\sigma_{t+1}^2 - \sigma_t^2}\, \epsilon$$

VP-SED <—> DDPM

(Variance Preserving)

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$$

$$x_{t+1} = \sqrt{1 - \beta_{t+1}} x_t + \sqrt{\beta_{t+1}}\, \epsilon$$

$$x_{t+\Delta t} - x_t = f(x_t, t)\Delta t + g(t)\sqrt{\Delta t}\epsilon$$

VE-SED:

$$\begin{aligned} x_{t+\Delta t} &= x_t + \sqrt{\sigma_{t+\Delta t}^2 - \sigma_t^2}\, \epsilon \\ &= x_t + \sqrt{\frac{\sigma_{t+\Delta t}^2 - \sigma_t^2}{\Delta t}} \cdot \sqrt{\Delta t}\, \epsilon \\ &= x_t + \sqrt{\frac{\Delta \sigma_t^2}{\Delta t}} \sqrt{\Delta t}\, \epsilon \end{aligned}$$

$$f(x_t, t) = 0$$

$$g(t) = \frac{d}{dt}\sigma_t^2$$

Song, Yang, et al. "Score-based generative modeling through stochastic differential equations." *arXiv:2011.13456* (2020).

VE-SED <—> score-based model

(Variance Exploring)

$$x_t = x_0 + \sigma_t \epsilon$$

$$x_{t+1} = x_t + \sqrt{\sigma_{t+1}^2 - \sigma_t^2}\, \epsilon$$

VP-SED <—> DDPM

(Variance Preserving)

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$$

$$x_{t+1} = \sqrt{1 - \beta_{t+1}} x_t + \sqrt{\beta_{t+1}}\, \epsilon$$

$$x_{t+\Delta t} - x_t = f(x_t, t)\Delta t + g(t)\sqrt{\Delta t}\epsilon$$

VP-SED (DDPM):

$$\{T\beta_i\}_{i=1}^{T} \to \beta(t) \quad t \in [0,1]$$

$$\beta\left(\frac{i}{T}\right) = T\beta_i \quad \Delta t = \frac{1}{T}$$

$$\begin{aligned} X_{t+\Delta t} &= \sqrt{1 - \beta(t + \Delta t)\Delta t}\, X_t + \sqrt{\beta(t + \Delta t)\Delta t}\, \epsilon \\ &\approx \left(1 - \frac{1}{2}\beta(t + \Delta t)\Delta t\right) X_t + \sqrt{\beta(t + \Delta t)\Delta t}\, \epsilon \\ &\approx \left(1 - \frac{1}{2}\beta(t)\Delta t\right) X_t + \sqrt{\beta(t)\Delta t}\, \epsilon \end{aligned}$$

$$f(X_t, t) = -\frac{1}{2}\beta(t)X_t$$

$$g(t) = \sqrt{\beta(t)}$$

Song, Yang, et al. "Score-based generative modeling through stochastic differential equations." *arXiv:2011.13456* (2020).

VE-SED <—> score-based model

(Variance Exploring)

$$x_t = x_0 + \sigma_t \epsilon$$

$$x_{t+1} = x_t + \sqrt{\sigma_{t+1}^2 - \sigma_t^2}\, \epsilon$$

$$f(x_t, t) = 0$$

$$g(t) = \frac{d}{dt}\sigma_t^2$$

$$p(x_t) \propto \exp\left\{ -\frac{\|x_t - \sqrt{\bar{\alpha}_t} x_0\|_2^2}{2(1 - \bar{\alpha}_t)} \right\}$$

VP-SED <—> DDPM

(Variance Preserving)

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$$

$$x_{t+1} = \sqrt{1 - \beta_{t+1}} x_t + \sqrt{\beta_{t+1}}\, \epsilon$$

$$f(x_t, t) = -\frac{1}{2}\beta(t) x_t$$

$$g(t) = \sqrt{\beta(t)}$$

$$\epsilon_\theta(x_t, t) = \frac{x_t - \sqrt{\bar{\alpha}_t} x_0}{\sqrt{1 - \bar{\alpha}_t}}$$

$$s_\theta(x_t, t) \approx \nabla_{x_t} \log p(X_t) = -\frac{x_t - \sqrt{\bar{\alpha}_t} x_0}{1 - \bar{\alpha}_t} = -\frac{1}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(x_t, t)$$

Song, Yang, et al. "Score-based generative modeling through stochastic differential equations." *arXiv:2011.13456* (2020).

Exact likelihood computation

Sampling:

$$d\mathbf{x} = \left[f(\mathbf{x}, t) - g^2(t)\,\nabla_{\mathbf{x}} \log p(\mathbf{x})\right] dt + g(t)\, dw$$

There exists a corresponding ***deterministic*** *process* whose trajectories share the same marginal probability densities as the SDE:

$$d\mathbf{x} = \left[f(\mathbf{x}, t) - \frac{1}{2} g(t)^2\,\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] dt$$

Conditional generation

“Inpainting”

Algorithm 2 Inpainting with annealed Langevin dynamics.

Require: $\{\sigma_i\}_{i=1}^{L}, \epsilon, T$ $\triangleright$ ϵ is smallest step size; T is the number of iteration for each noise level.

Require: $\mathbf{m}, \mathbf{x}$ $\triangleright$ $\mathbf{m}$ is a mask to indicate regions not occluded; $\mathbf{x}$ is the given image.

Initialize $\tilde{\mathbf{x}}_0$

for $i \leftarrow 1$ to L **do**

  $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2/\sigma_L^2$ $\triangleright$ α_i is the step size.

  Draw $\tilde{\mathbf{z}} \sim \mathcal{N}(0, \sigma_i^2)$

  $\mathbf{y} \leftarrow \mathbf{x} + \tilde{\mathbf{z}}$

  for $t \leftarrow 1$ to T **do**

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

    $\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \dfrac{\alpha_i}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t-1}, \sigma_i) + \sqrt{\alpha_i}\,\mathbf{z}_t$

    $\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_t \odot (1 - \mathbf{m}) + \mathbf{y} \odot \mathbf{m}$

  end for

  $\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$

end for

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

https://yang-song.net/blog/2021/score/

Song, Yang, and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution." *NIPS* (2019)

Conditional generation

Guided diffusion

$$\nabla_{x_t} \log p(x_t) \Rightarrow \nabla_{x_t} \log p(x_t \mid y)$$, y is some abstract labels

$$p(x_t \mid y) = \frac{p(y \mid x_t) p(x_t)}{p(y)}$$

$$\nabla_{x_t} \log p(x_t \mid y) = \nabla_{x_t} \log p(y \mid x_t) + \underbrace{\boxed{\nabla_{x_t} \log p(x_t)}}_{\text{score}}$$

$p_\theta(y \mid x_t)$ is a classifier trained on noisy images

Dhariwal, Prafulla, and Alexander Nichol. "Diffusion models beat gans on image synthesis." *NIPS* (2021)

Conditional generation

Guided diffusion

$$\nabla_{x_t} \log p(x_t \mid y) = \nabla_{x_t} \log p(y \mid x_t) + \nabla_{x_t} \log p(x_t)$$

Figure 3: Samples from an unconditional diffusion model with classifier guidance to condition on the class "Pembroke Welsh corgi". Using classifier scale 1.0 (left; FID: 33.0) does not produce

Dhariwal, Prafulla, and Alexander Nichol. "Diffusion models beat gans on image synthesis." *NIPS* (2021)