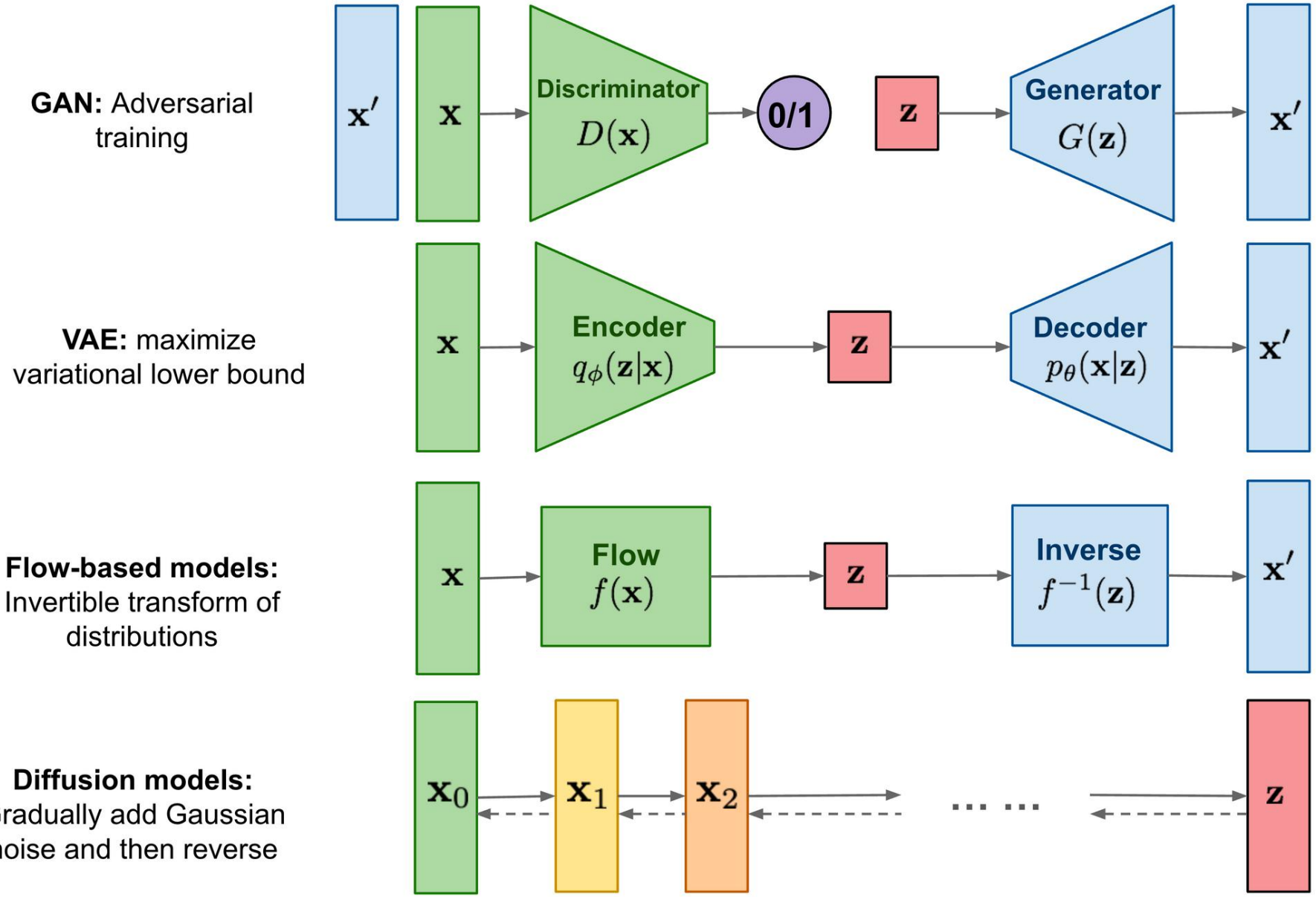
Diffusion Model

Xinyang Liu
Xidian University



Demo Images from https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

DALL-E 2

DALL-E 2: An astronaut riding a horse in a photorealistic style

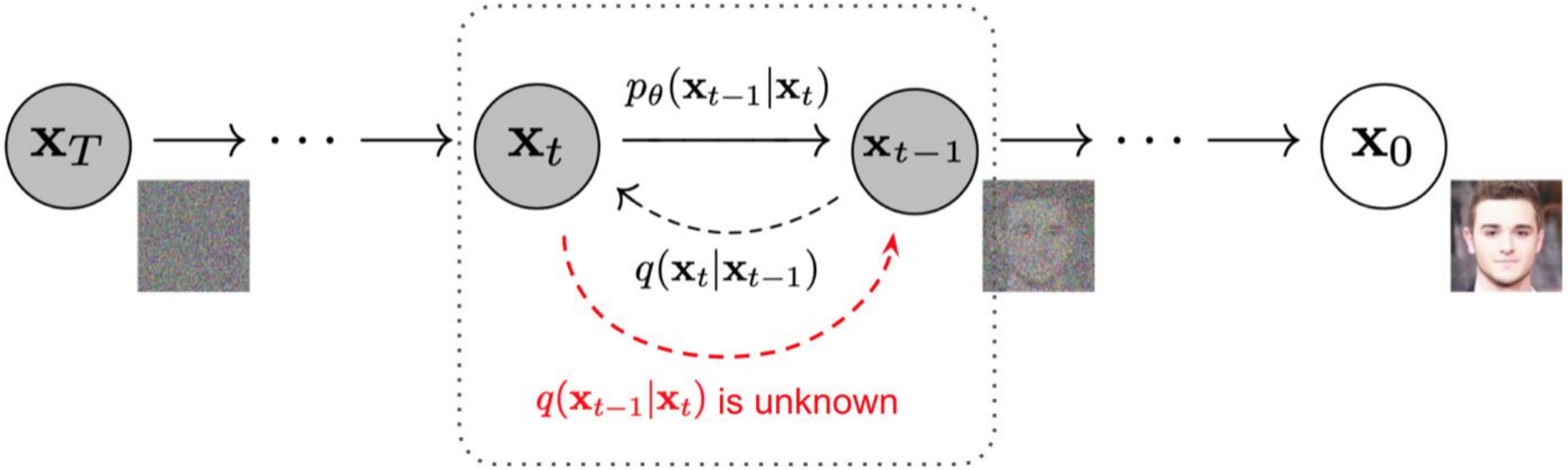
DALL-E 2: Teddy bears mixing sparkling chemicals as mad scientists as a 1990s Saturday morning cartoon

Denoising Diffusion Probabilistic Models

Jonathan Ho
UC Berkeley
jonathanho@berkeley.edu

Ajay Jain
UC Berkeley
ajayj@berkeley.edu

Pieter Abbeel
UC Berkeley
pabbeel@cs.berkeley.edu

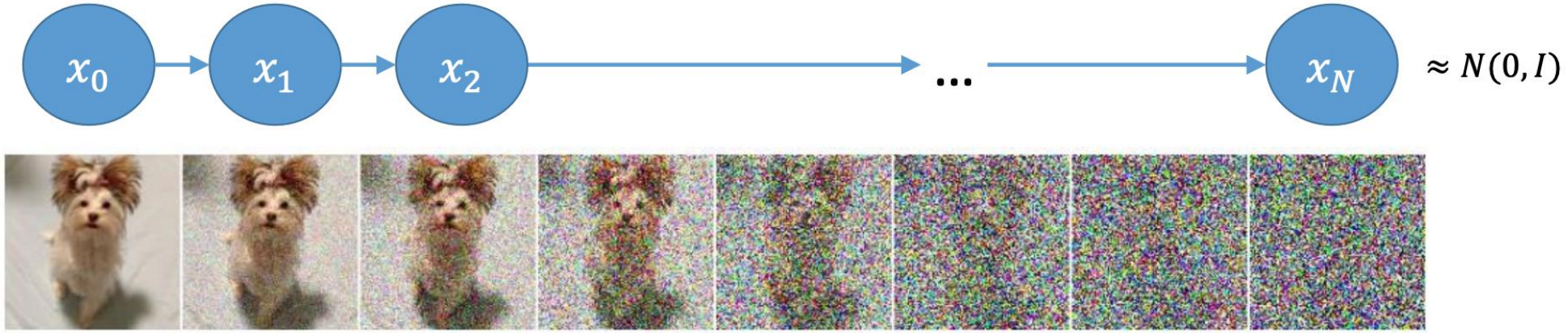


Ho et al. Denoising Diffusion Probabilistic Models (DDPM), NIPS 2020

Forward/Diffusion process

- Diffusion process gradually injects noise to data
- Described by a Markov chain: $q(x_0, \ldots, x_T) = q(x_0)q(x_1|x_0)q(x_2|x_1)\ldots q(x_T|x_{T-1})$

Transition of diffusion: $q(x_t|x_{t-1}) := \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, \beta_t \mathrm{I})\alpha_t = 1 - \beta_t$

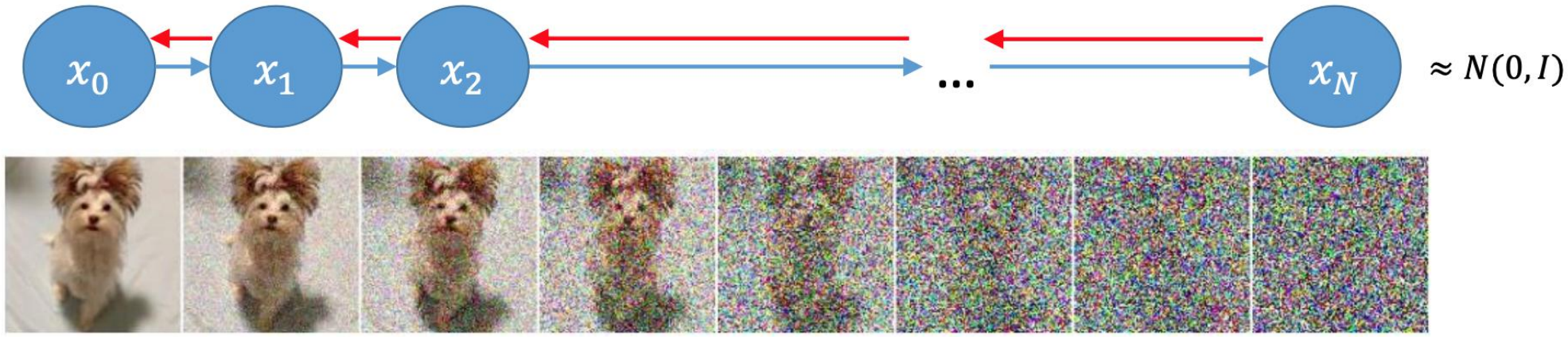


Demo Images from Song et al. Score-based generative modeling through stochastic differential equations, ICLR 2021

Reverse process

- Diffusion process in the reverse direction ⟺ denoise process
- Reverse factorization: $q(x_0, \ldots, x_T)$
  $= q(x_0|x_1)q(x_1|x_2)\ldots q(x_{T-1}|x_T)q(x_T)$

  Transition of reverse: $q(x_{t-1}|x_t)$
  $= ?$



Diffusion process: $q(x_0, \ldots, x_T)$
$= q(x_0)q(x_1|x_0)q(x_2|x_1)\ldots q(x_T|x_{T-1})$
$= q(x_0|x_1)q(x_1|x_2)\ldots q(x_{T-1}|x_T)q(x_T)$

Approximation

- Approximate diffusion process in the reverse direction

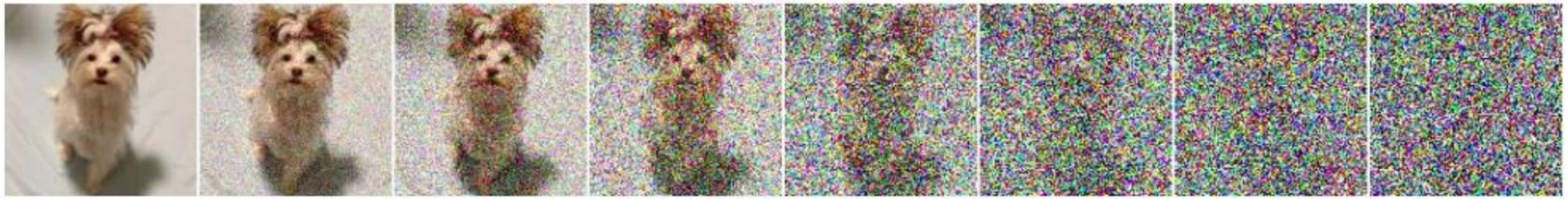
Model Transition: $p_\theta(x_{t-1}|x_t)$

$= \mathcal{N}(\mu_{t,\theta}(x_t), \Sigma_{t,\theta}(x_t))$

approximate

Transition of reverse: $q(x_{t-1}|x_t)$

$= ?$



Diffusion process: $q(x_0, \ldots, x_T)$

$= q(x_0)q(x_1|x_0)q(x_2|x_1)\ldots q(x_T|x_{T-1})$

$= q(x_0|x_1)q(x_1|x_2)\ldots q(x_{T-1}|x_T)q(x_T)$

The model: $p_\theta(x_0, \ldots, x_T)$

$= p(x_T)p_\theta(x_0|x_1)p_\theta(x_1|x_2)\ldots p_\theta(x_{T-1}|x_T)p(x_T)$

$= \mathcal{N}(x_T; 0, I)$

Approximation

- We hope $q(x_0, \ldots, x_T) \approx p_\theta(x_0, \ldots, x_T)$ $\quad p_\theta(x_{t-1}|x_t) = \mathcal{N}(\mu_t(x_t); \Sigma_t(x_t))$
- Achieved by minimizing their KL divergence (i.e., maximizing the ELBO)

min KL $\qquad\qquad$ max ELBO

$$\min_{\mu_t, \Sigma_t} KL(q(x_{0:T}) \| p_\theta(x_{0:T})) \Leftrightarrow \max_{\mu_t, \Sigma_t} E_q \log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}$$

Approximation (simplify)

$q(x_{t-1}|x_t)$ is intractable

Using Bayes' rule: $q(x_{t-1}|x_t, x_0) = q(x_t|x_{t-1}, x_0)\frac{q(x_{t-}}{q(x_t}$

$$L_{VLB}$$

$$= \mathbb{E}_{q(x_{0:T})}\left[\log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})}\right]$$

$$= \dots$$

$$= \mathbb{E}_q\left[-\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{q(x_t|x_{t-1})}{p_\theta(x_{t-1}|x_t)} + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}\right]$$

$$= \dots$$

$$= \mathbb{E}_q\left[-\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{q(x_{t-1}|x_t, x_0)}{p_\theta(x_{t-1}|x_t)} \cdot \frac{q(x_t|x_0)}{q(x_{t-1}|x_0)}\right.$$

$$\left. + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}\right]$$

$$= \mathbb{E}_q[D_{KL}(q(x_T|x_0) \| p(x_T))$$

$$+ \sum_{t=2}^{T} D_{KL}(q(x_t|x_{t-1}, x_0) \| p_\theta(x_{t-1}|x_t)) - \log p_\theta(x_0|x_1)]$$

L_T L_{t-1} L_0

Simple Derivation

- $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{\alpha_t}x_0,\ \beta_t \mathrm{I})$ Set $\alpha_t + \beta_t = 1, \overline{\alpha}_t = \prod_{s=1}^{t} \alpha_t$

★ • $q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\overline{\alpha}_t}x_0, (1-\overline{\alpha}_t)\mathrm{I}) \Leftrightarrow x_t = \sqrt{\overline{\alpha}_t}x_0 + \sqrt{1-\overline{\alpha}_t}\epsilon_t$

$x_0 = \frac{1}{\sqrt{\overline{\alpha}_t}}(x_t - \sqrt{1-\overline{\alpha}_t}\epsilon_t)$

★ • $q(x_{t-1}|x_t, x_0) = \mathcal{N}\left(x_{t-1}; \widetilde{\mu}_t(x_t, x_0), \widetilde{\beta}_t \mathrm{I}\right)$

Where $\widetilde{\mu}_t(x_t, x_0) := \frac{\sqrt{\overline{\alpha}_{t-1}}\beta_t}{1-\overline{\alpha}_t}x_0 + \frac{\sqrt{\overline{\alpha}_t}(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}x_t$ and $\widetilde{\beta}_t := \frac{1-\overline{\alpha}_{t-1}}{1-\overline{\alpha}_t}\beta_t$

- $p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t); \Sigma_\theta(x_t, t))$,

$\Sigma_\theta(x_t, t) = \sigma_t^2 \mathrm{I}$
$= \widetilde{\beta}_t \mathrm{I}$

Approximation (simplify)

$$x_t \to x_t(x_0, \epsilon)$$
$$x_0 \to x_t^{-1}(x_0, \epsilon)$$

$$L_{t-1} = D_{KL}(q(x_t|x_{t-1}, x_0) \| p_\theta(x_{t-1}|x_t))$$
$$= \mathbb{E}_q\left[\frac{1}{2\sigma^2}\| \tilde{\mu}_t\ (x_t, x_0) - \mu_\theta\ (x_t, t) \|^2\right]$$

$$L_{t-1} - C + C$$

$$= \mathbb{E}_{x_0,\epsilon}\left[\frac{1}{2\sigma^2}\left\|\tilde{\mu}_t\ \left(x_t(x_0,\epsilon), \frac{1}{\sqrt{\overline{\alpha}_t}}(x_t(x_0,\epsilon) - \sqrt{1-\overline{\alpha}_t}\epsilon)\right) - \mu_\theta\ (x_t(x_0,\epsilon), t)\right\|^2\right]$$

$$= \mathbb{E}_{x_0,\epsilon}\left[\frac{1}{2\sigma^2}\left\|\frac{1}{\sqrt{\overline{\alpha}_t}}(x_t(x_0,\epsilon) - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon)\right.\right.$$
$$\left.\left. - \mu_\theta\ (x_t(x_0,\epsilon), t)\right\|^2\right]$$

$$= \mathbb{E}_{x_0,\epsilon}\left[\frac{\beta_t^2}{2\sigma^2\alpha_t(1-\overline{\alpha}_t)}\|\epsilon - \epsilon_\theta(\sqrt{\overline{\alpha}_t}x_0 + \sqrt{1-\overline{\alpha}_t}\epsilon, t)\|^2\right]$$

$$L_{simple}(\theta) = \mathbb{E}_{t,x_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(\sqrt{\overline{\alpha}_t}x_0 + \sqrt{1-\overline{\alpha}_t}\epsilon, t)\|^2\right]$$

Training & Sampling algorithms

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** $\mathbf{x}_0$

Setting

$T = 1000$

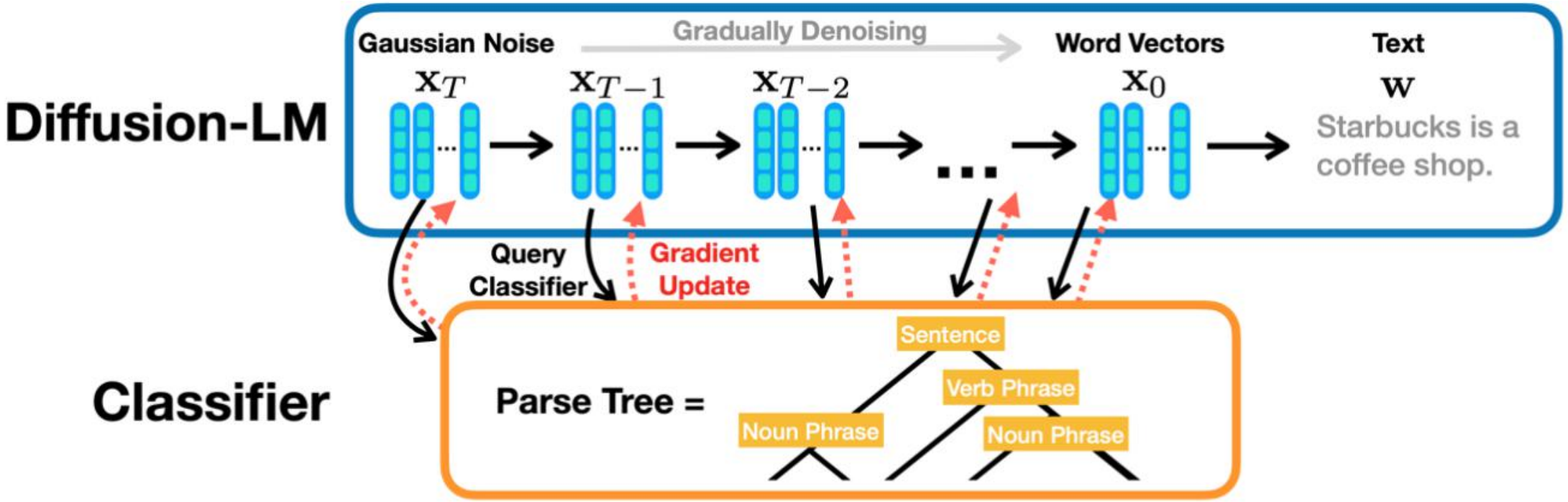
$\beta_1 = 10^{-4}$ to $\beta_T = 0.02 \Rightarrow D_{KL}(q(x_T|x_0)\|\mathcal{N}(\mathbf{0}, \mathbf{I})) \approx 10^{-5}$

$\mu_\theta(x_t, t)$: U-Net

Applications (Conditioned/Controllable Generation)

[NLP] Test Generation

Li et al. Diffusion-LM Improves Controllable Text Generation, NIPS 2023



$$\nabla_{\mathbf{x}_{t-1}} \log p(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{c}) = \nabla_{\mathbf{x}_{t-1}} \log p(\mathbf{x}_{t-1} \mid \mathbf{x}_t) + \nabla_{\mathbf{x}_{t-1}} \log p(\mathbf{c} \mid \mathbf{x}_{t-1})$$

Applications

[Vision] (Controllable) Image Generation

- Text-to-image generation: DALL-E 2
- Image-to-image translation
- Image restoration: DDRM
- …

[Graph] Molecule Generation

Xu et al. GEODIFF: A GEOMETRIC DIFFUSION MODEL FOR MOLECULAR CONFORMATION GENERATION, ICLR 2022

Quick Summary

Pros:

- Tractability and flexibility
- High quality

Cons:

- Expensive and inefficiency (Depends on a long Markov chain of diffusion steps)
- Inconvenience on discrete data

Works on discrete data (categorical data)

Multinomial Diffusion $x_t \in \{0,1\}^K$ (one-hot format)

$$C(x|p) = \prod_{i=1}^{k} p_i^{[x=i]}$$

Forward process $q(x_t|x_{t-1}) = C(x_t; (1-\beta_t)x_{t-1} + \beta_t/K)$

$$q(x_t|x_0) = C(x_t;\ \overline{\alpha}_t x_0 + (1-\overline{\alpha}_t)/K)$$

$$q(x_{t-1}|x_t, x_0) = C(x_{t-1}; \theta_{post}(x_t, x_0))$$

where $\theta_{post}(x_t, x_0) = \tilde{\theta}/\sum_{k=1}^{K}\tilde{\theta}_k$ and $\tilde{\theta} = [\alpha_t x_t + (1-\alpha_t)/K] \odot [\overline{\alpha}_{t-1} x_0 + (1-\overline{\alpha}_{t-1})/K]$

$p(x_0|x_1) = C(x_0; \hat{x}_0)$ and $p(x_{t-1}|x_t) = C(x_{t-1}; \theta_{post}(x_t, \hat{x}_0))$ where $\hat{x}_0 = \mu(x_t, t)$

Then we can calculate the KL terms:

$$D_{KL}(q(x_t|x_{t-1}, x_0) \,\|\, p_\theta(x_{t-1}|x_t))$$
$$= D_{KL}(C(x_{t-1}; \theta_{post}(x_t, x_0)) \,\|\, C(x_{t-1}; \theta_{post}(x_t, \hat{x}_0)))$$

Hoogeboom et al. Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions, NIPS 2021

Works on discrete data (categorical data) ★

DP3M $x_t \in \{0,1\}^K$ (one-hot format)

Forward process $q(x_t|x_{t-1}) = C(x_t; p = x_{t-1}Q_t)$

$q(x_t|x_0) = C(x_t; p = x_0\overline{Q}_t)$, with $\overline{Q}_t = Q_1 Q_2 \ldots Q_t$

$$q(x_{t-1}|x_t, x_0) = C\left(x_{t-1}; p = \frac{x_t Q_t^T \odot x_0 Q_{t-1}^T}{x_0 \overline{Q}_t x_t^T}\right)$$

Noise schedules

- Uniform
- Absorbing state (MASK)
- Discretized Gaussian
- ...

Austin et al. Structured Denoising Diffusion Models in Discrete State-Spaces NIPS 2021

Works on Speeding up sampling (DDIM)

DDPM: $q(x_t|x_{t-1}) \xrightarrow{\text{derivate}} q(x_t|x_0) \xrightarrow{\text{derivate}} q(x_{t-1}|x_t, x_0) \xrightarrow{\text{approximate}} q(x_{t-1}|x_t)$

- Loss $\longleftarrow q(x_t|x_0)$
- Sampling $\longleftarrow q(x_{t-1}|x_t)$

$$L_{t-1} = D_{KL}(q(x_t|x_{t-1}, x_0) \| p_\theta(x_{t-1}|x_t))$$

$$\int q(x_{t-1}|x_t, x_0) q(x_t|x_0)\, dx_t = q(x_{t-1}|x_0)$$

$$q_\sigma(x_{1:T}|x_0) := q_\sigma(x_T|x_0) \prod_{t=2}^{T} q_\sigma(x_{t-1}|x_t, x_0)$$

where $q_\sigma(x_T|x_0) = \mathcal{N}\left(\sqrt{\alpha_T}x_0, (1-\alpha_T)\boldsymbol{I}\right)$

Ensure that $q_\sigma(x_t|x_0) = \mathcal{N}\left(x_t; \sqrt{\alpha_t}x_0, (1-\alpha_t)\boldsymbol{I}\right)$

$$q_\sigma(x_{t-1}|x_t, x_0) = \mathcal{N}\left(\sqrt{\alpha_{t-1}}x_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \frac{x_t - \sqrt{\alpha_t}x_0}{\sqrt{1-\alpha_t}}, \sigma_t^2 \boldsymbol{I}\right)$$

Song et al. DENOISING DIFFUSION IMPLICIT MODELS, ICLR 2021

Works on Speeding up sampling (DDIM)

$$p_\theta^{(t)}(x_{t-1}|x_t) = \begin{cases} \mathcal{N}\left(f_\theta^{(t)}(x_t), \sigma_1^2 \boldsymbol{I}\right) & \text{if } t = 1 \\ q_\sigma\left(x_{t-1}|x_t, f_\theta^{(t)}(x_t)\right) & \text{otherwise} \end{cases}$$

$$f_\theta^{(t)}(x_t) := \frac{1}{\sqrt{\alpha_t}}\left(x_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(x_t)\right)$$

Train process: $\left\|\epsilon - \epsilon_\theta(\sqrt{\alpha_t}x_0 + \sqrt{1-\alpha_t}\epsilon, t)\right\|^2$

$$x_{t-1} = \sqrt{\alpha_{t-1}}\left(\frac{x_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(x_t)}{\sqrt{\alpha_t}}\right) + \sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \epsilon_\theta^{(t)}(x_t) + \sigma_t \epsilon_t$$

$\tau = [\tau_1, \tau_2, \ldots, \tau_{S-1}, \tau_S], \tau_S = T, \ \bar{\tau} := \{1, \ldots T\} \setminus \tau$ ★

Inference process: $q_{\sigma,\tau}(x_{1:T}|x_0) := q_{\sigma,\tau}(x_{\tau_S}|x_0)\prod_{i=1}^{S} q_{\sigma,\tau}(x_{\tau_{i-1}}|x_{\tau_i}, x_0)\prod_{t\in\bar{\tau}} q_$

$q_{\sigma,\tau}(x_{\tau_i-1}|x_{\tau_i}, x_0)$

$$= \mathcal{N}\left(x_t; \sqrt{\alpha_{\tau_i-1}}x_0 + \sqrt{1-\alpha_{\tau_i-1}-\sigma_{\tau_i}^2} \cdot \frac{x_{\tau_i} - \sqrt{\alpha_{\tau_i}}x_0}{\sqrt{1-\alpha_{\tau_i}}}, \sigma_{\tau_i}^2 \boldsymbol{I}\right)$$

Generative process: $p_\theta(x_{0:T}) :=$

$$p_\theta(x_T)\prod_{i=1}^{S} p_\theta^{(\tau_i)}(x_{\tau_{i-1}}|x_{\tau_i}) \times \prod_{t\in\bar{\tau}} p_\theta^{(t)}(x_0|x_t)$$

Use to produce samples

In variational objective

Simple Derivation (DDIM)

$$x_{t-1} = \sqrt{\alpha_{t-1}}\left(\frac{x_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(x_t)}{\sqrt{\alpha_t}}\right) + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\epsilon_\theta^{(t)}(x_t) + \sigma_t\epsilon_t$$

When $\sigma_t = g(\alpha_t, \alpha_{t-1}) \Longrightarrow$ DDPM

$\sigma_t = 0 \Longrightarrow$ deterministic decoding $\Longrightarrow$ Denoising Diffusion Implicit Model (DDIM)

- Interpolation
- $\frac{x_{t-\Delta t}}{\sqrt{\alpha_{t-\Delta t}}} = \frac{x_t}{\sqrt{\alpha_t}} + \left(\sqrt{\frac{1-\alpha_{t-\Delta t}}{\alpha_{t-\Delta t}}} - \sqrt{\frac{1-\alpha_t}{\alpha_t}}\right)\epsilon_\theta^{(t)}(x_t)$

$$\mathrm{d}\frac{x_t}{\sqrt{\alpha_t}} = \epsilon_\theta^{(t)}\left(\frac{x_t}{\sqrt{\sigma^2+1}}\right)\mathrm{d}\sigma_t \Longrightarrow \text{ODE}$$

Score-Based Model

Xinyang Liu
Xidian University

Statistical foundations of generative models

Example

First pixel

121

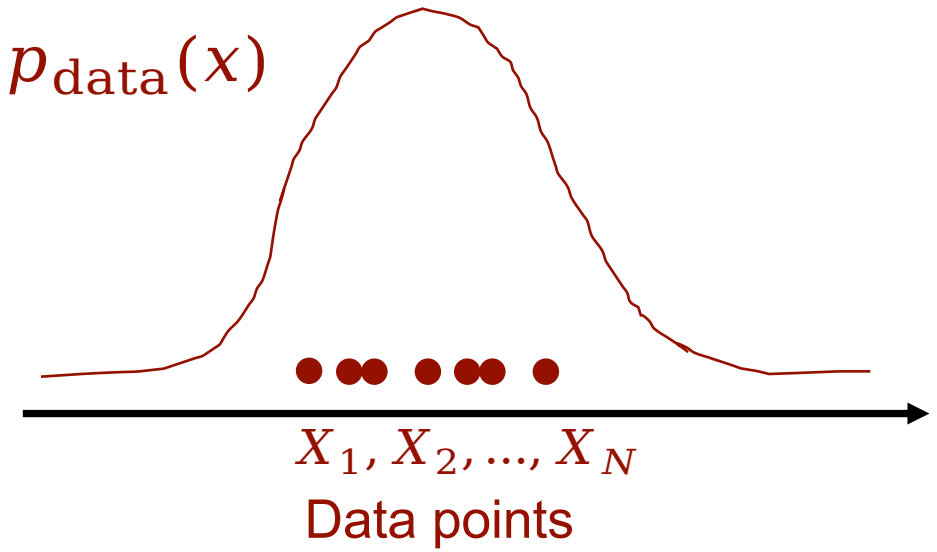
137

108

149

Model family

$$p_{\mu,\sigma}(x) = \mathcal{N}(x|\mu, \sigma^2)$$



Statistical foundations of generative models

Example

First pixel

121

137

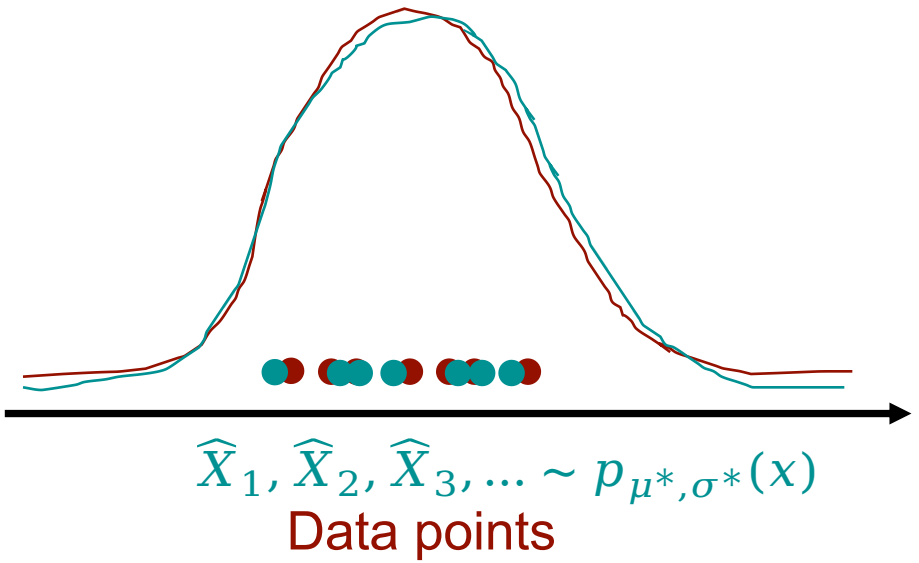
108

149

Model family

$$p_{\mu,\sigma}(x) = \mathcal{N}(x|\mu, \sigma^2)$$

Generative models → $p_{\mu^*,\sigma^*}(x) \approx p_{\text{data}}(x)$



Statistical foundations of generative models

Example

First pixel

121

137

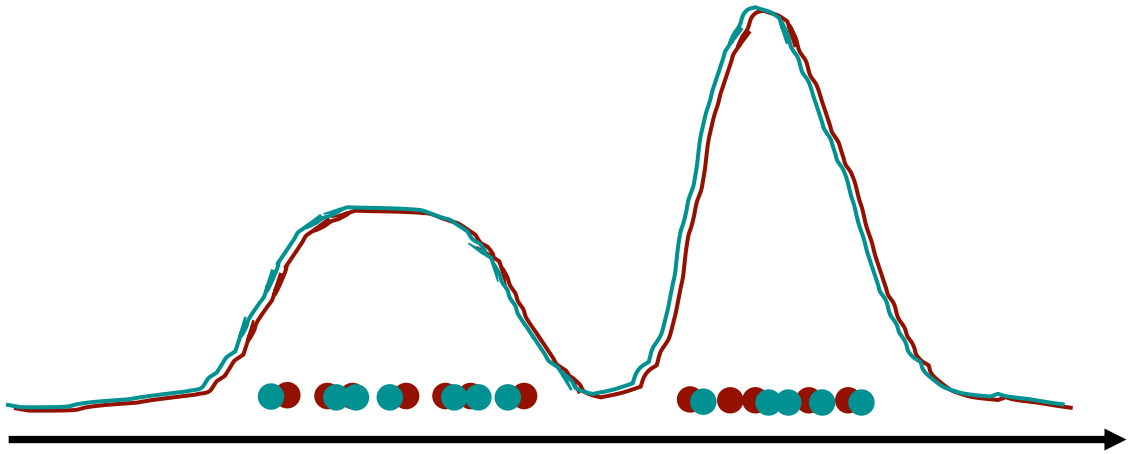
108

149

Model family

$$p_{\mu,\sigma}(x) = \mathcal{N}(x|\mu, \sigma^2)$$

Generative models



Statistical foundations of generative models

Well studied in statistical inference & density estimation:

- Maximum likelihood estimation (Ronald Fisher 1912-1922)
- Method of moments (Chebyshev 1887)
- Pseudo-likelihood maximization (Besag 1975)

New challenges in modeling high dimensional data:

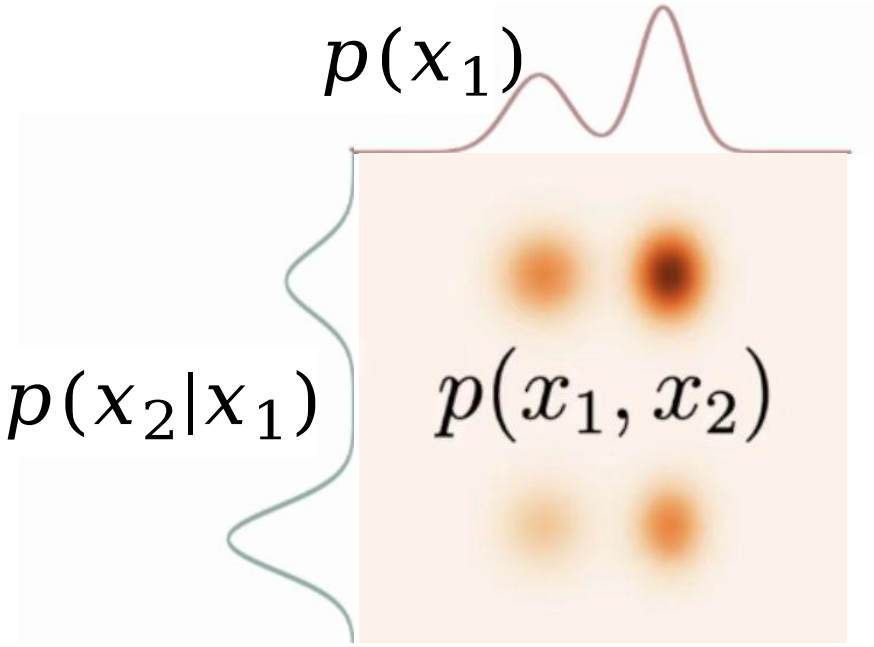
x_1

Statistical foundations of generative models

Well studied in statistical inference & density estimation:

- Maximum likelihood estimation (Ronald Fisher 1912-1922)
- Method of moments (Chebyshev 1887)
- Pseudo-likelihood maximization (Besag 1975)

New challenges in modeling high dimensional data:

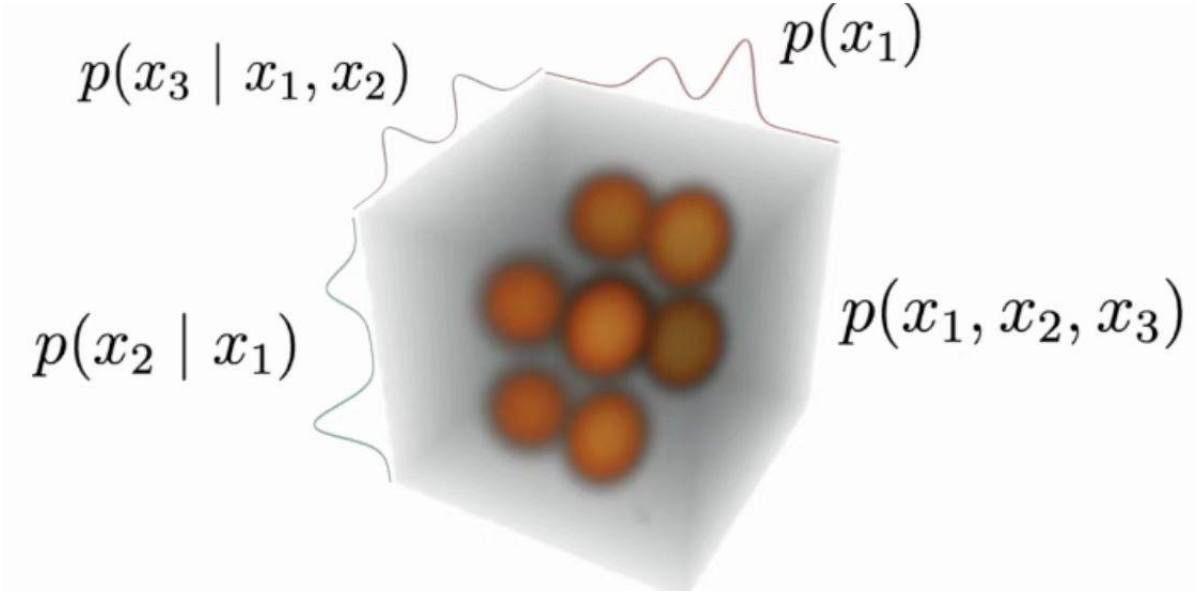


Statistical foundations of generative models

Well studied in statistical inference & density estimation:

- Maximum likelihood estimation (Ronald Fisher 1912-1922)
- Method of moments (Chebyshev 1887)
- Pseudo-likelihood maximization (Besag 1975)

New challenges in modeling high dimensional data:

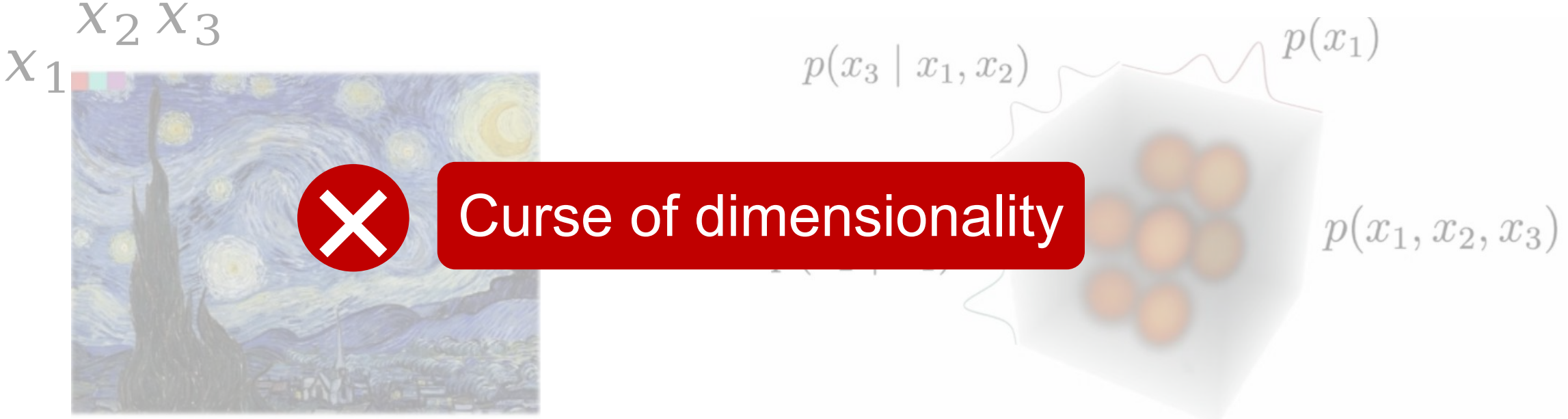


Statistical foundations of generative models

Well studied in statistical inference & density estimation:

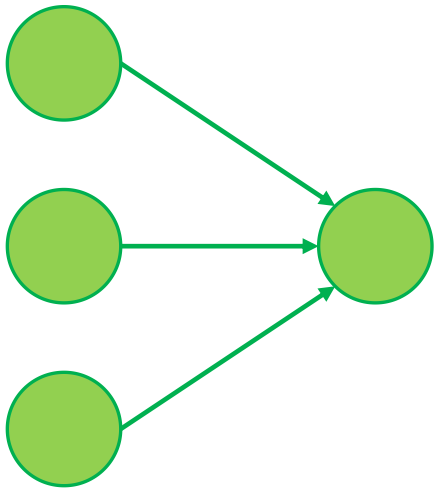
- Maximum likelihood estimation (Ronald Fisher 1912-1922)
- Method of moments (Chebyshev 1887)
- Pseudo-likelihood maximization (Besag 1975)

New challenges in modeling high dimensional data:



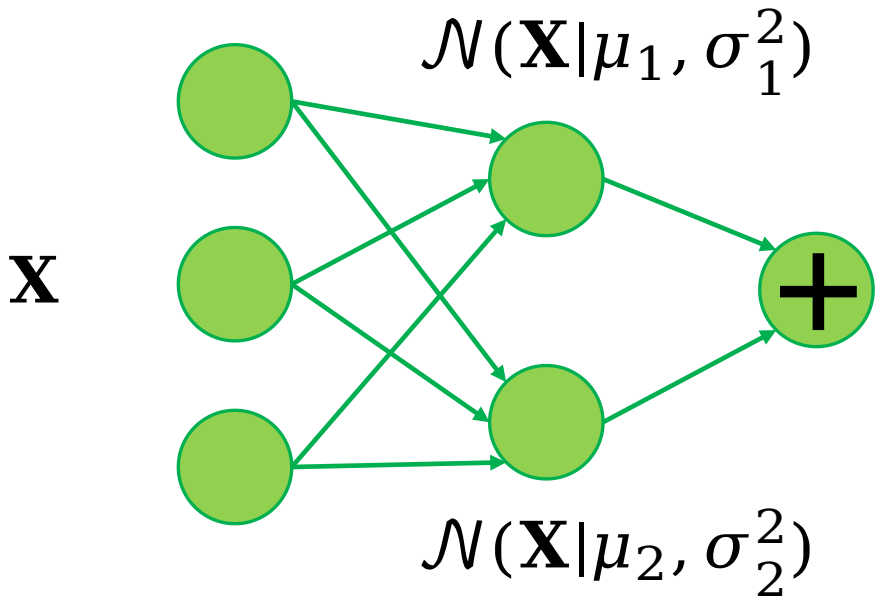
Towards more expressive generative models

$\mathbf{x}$

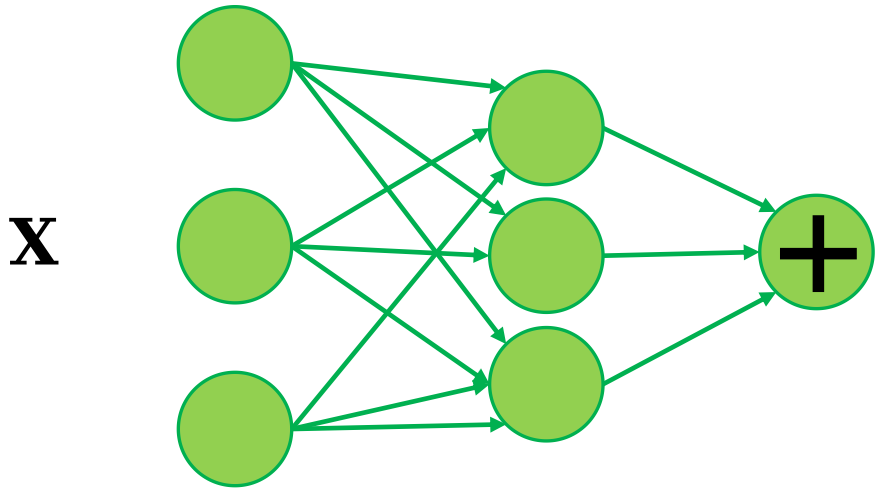


$\mathcal{N}(x|\mu, \sigma^2)$

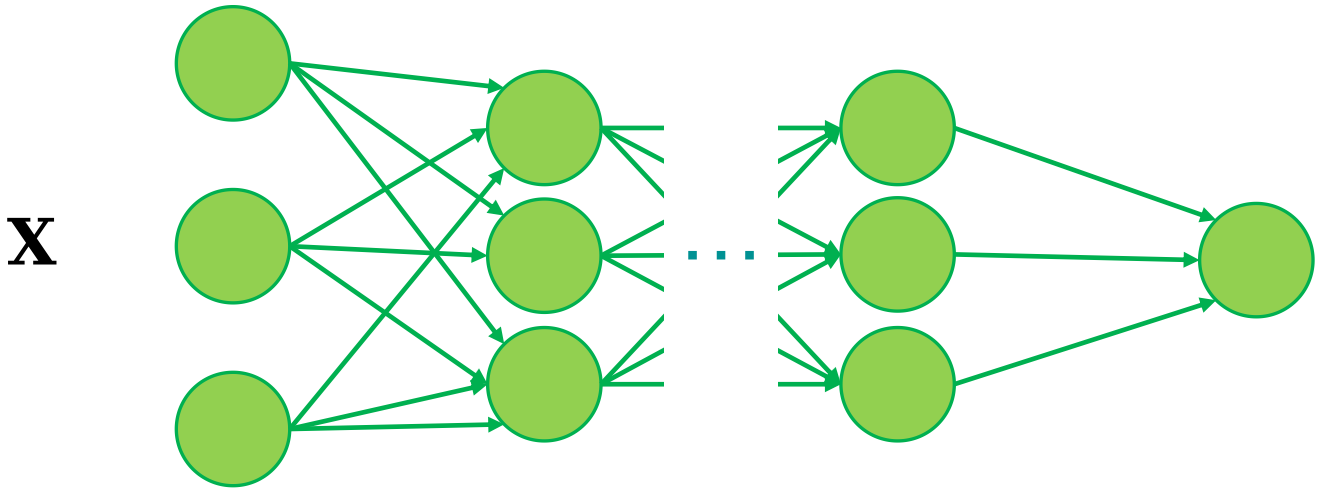
Towards more expressive generative models



Towards more expressive generative models



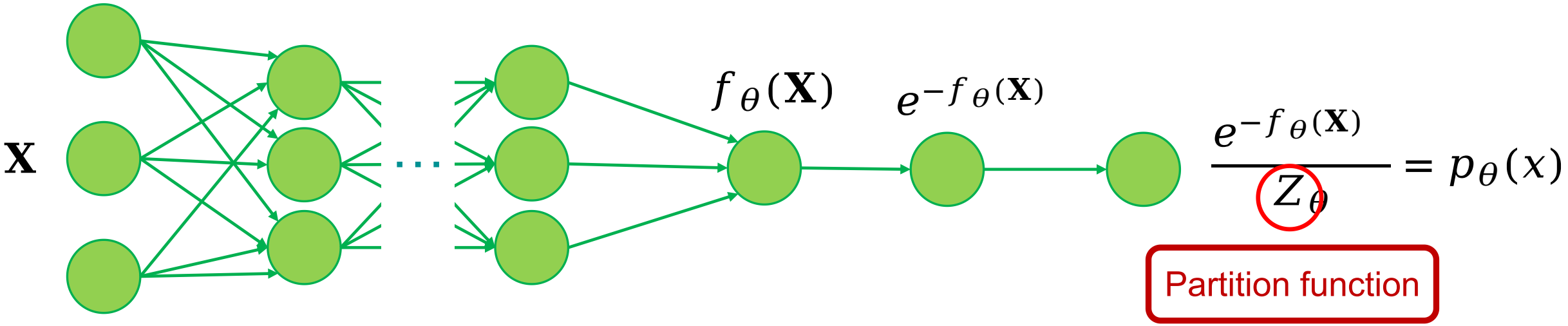
Towards more expressive generative models



Deep Neural Network

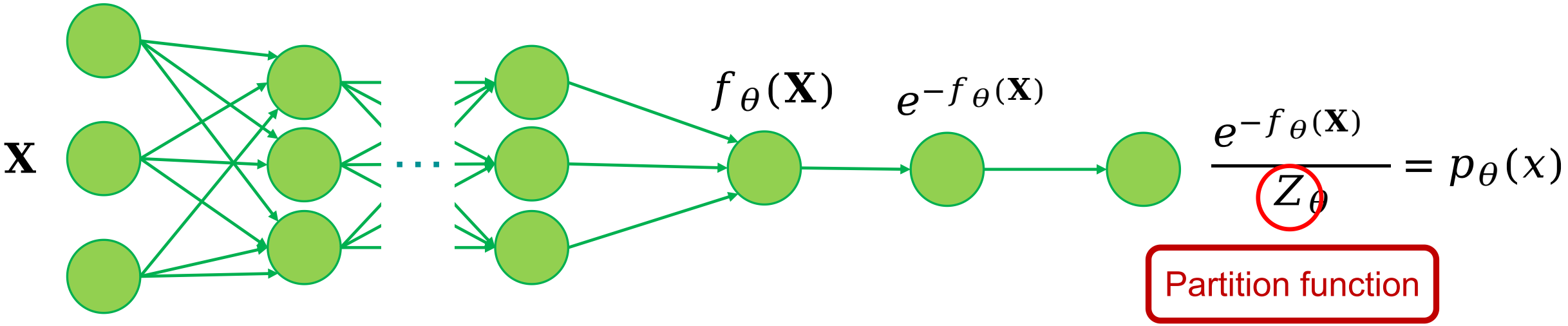
Deep Generative Models

Towards more expressive generative models



$$Z_\theta = \iint \dots \int e^{-f_\theta(x)} \, dx_1 dx_2 \dots dx_d$$

Towards more expressive generative models



$$Z_\theta = \iint \dots \int e^{-f_\theta(x)} \, dx_1 dx_2 \dots dx_d$$

Approximation the Partition function:

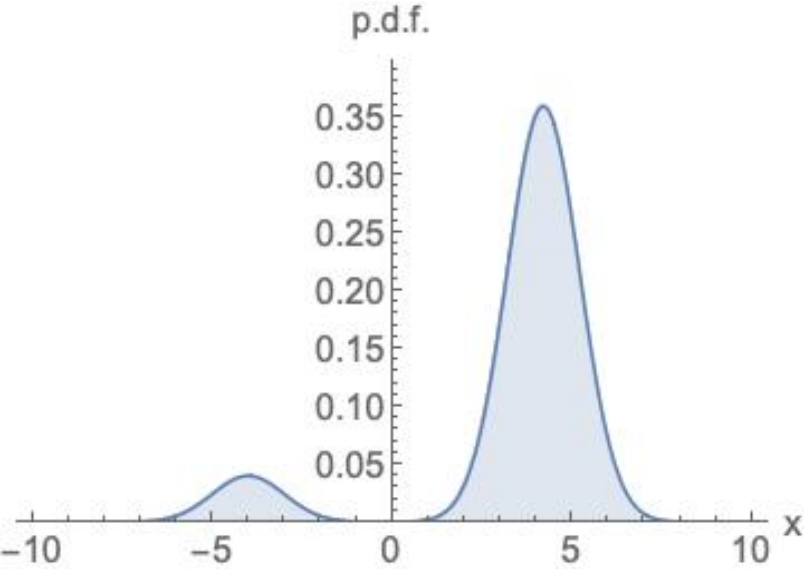
- Variational auto-encoders (Kingma & Welling 2014, Rezende et al. 2014)
- Contrastive divergence (Hinton 2002)

Using **restricted** model architectures:

- Autoregressive models (Bengio & Bengio 2000)
- Normalizing flow models (Dinh et al. 2014)

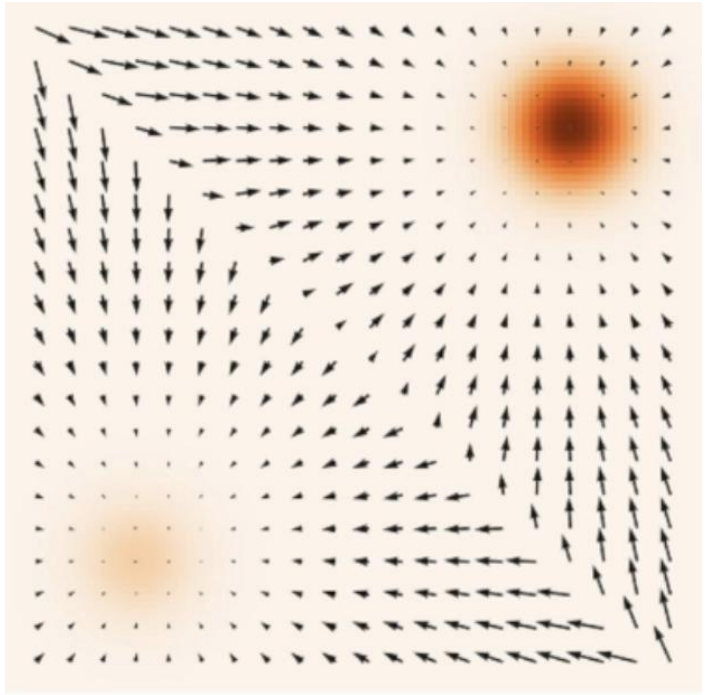
Bypassing the partition function

$p(x)$

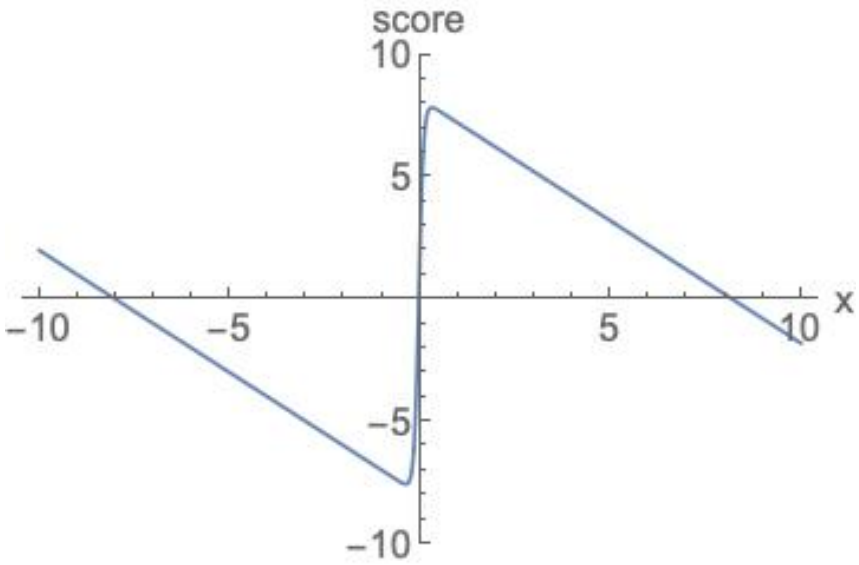


Density function

$$p_\theta(x) = \frac{e^{-f_\theta(\mathbf{x})}}{Z_\theta}$$



$\nabla_x \log p(x)$



(Stein 1986) Score function

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

$$= -\nabla_x f_\theta(x)$$

How to estimate score function ★

- Given: i.i.d. samples $\{x_1, x_2, \ldots x_N\} \sim p_{\text{data}}(x)$
- Goal: Estimating the score $\nabla_x \log p_{\text{data}}(x)$
- Score Model: A trainable vector-valued function $s_\theta(x): \mathbb{R}^D \to \mathbb{R}^D$
- Objective: How to compare two vector fields of score?

$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(x)}[\|\nabla_x \log p_{\text{data}}(x) - s_\theta(x)\|_2^2]$$ (Fisher divergence)

Integration by parts → **Score Matching** (Hyvärinen 2005)

$$\mathbb{E}_{p_{\text{data}}(x)}[\frac{1}{2}\|s_\theta(x)\|_2^2 + \text{trace}(\nabla_x s_\theta(x))]$$

$$\approx \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\|s_\theta(x_i)\|_2^2 + \text{trace}(\nabla_x s_\theta(x_i))]$$

Not scalable

Sliced score matching ★★

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

- **Intuition:** one dimensional problems should be easier
- **Idea:** project onto random directions
- **Randomized objective: Sliced Fisher Divergence**

$$\frac{1}{2} \mathbb{E}_{p_v} \mathbb{E}_{p_{\text{data}}(x)} [\| v^T \nabla_x \log p_{\text{data}}(x) - v^T s_\theta(x) \|_2^2]$$

Integration by parts → **Sliced Score Matching**

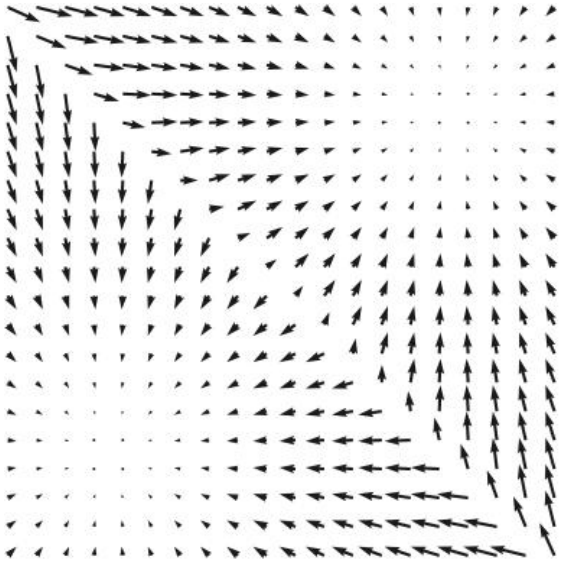
$$\mathbb{E}_{p_v} \mathbb{E}_{p_{\text{data}}(x)} [v^T \nabla_x s_\theta(x) v + \frac{1}{2} (v^T s_\theta(x))^2]$$

Scalable!

Yang Song et al. Sliced Score Matching: A Scalable Approach to Density and Score Estimation, UAI 2019 Oral

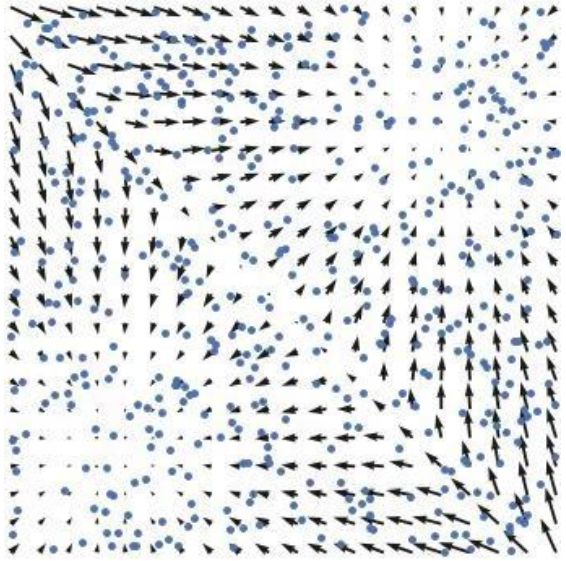
Score-based generative modeling

$$s_\theta(x) \approx \nabla_x \log p_{\text{data}}(x)$$



Scores

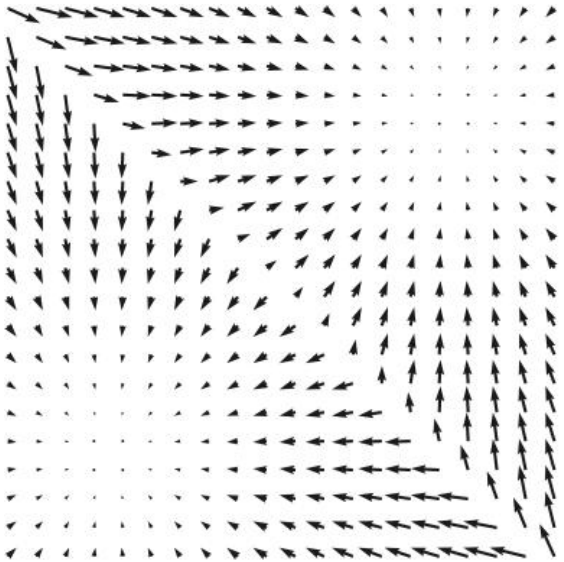
$s_\theta(x)$



Follow the scores

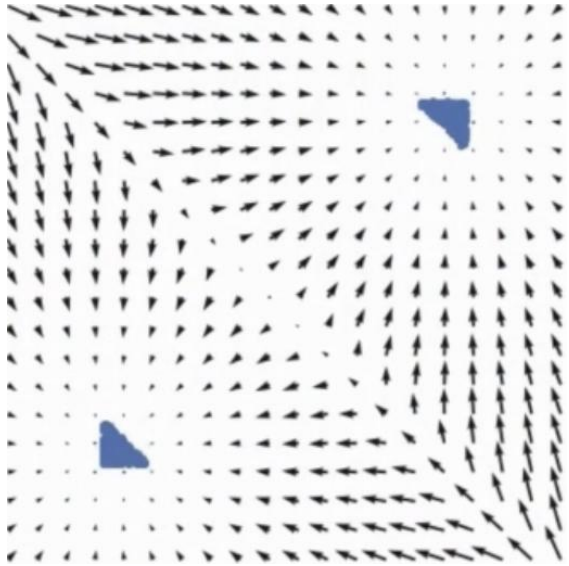
Score-based generative modeling

$$s_\theta(x) \approx \nabla_x \log p_{\text{data}}(x)$$

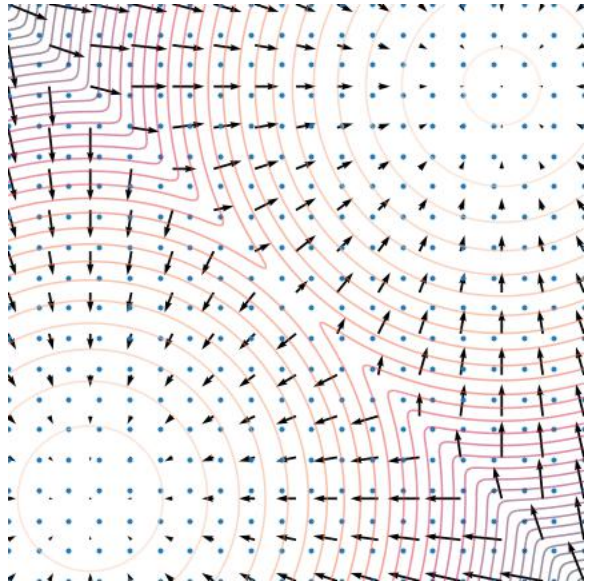


Scores

$s_\theta(x)$



Follow the scores



Follow noisy scores: Langevin dynamics (Parisi 1981)

Langevin dynamics

$$x_0 \sim \pi(x)$$

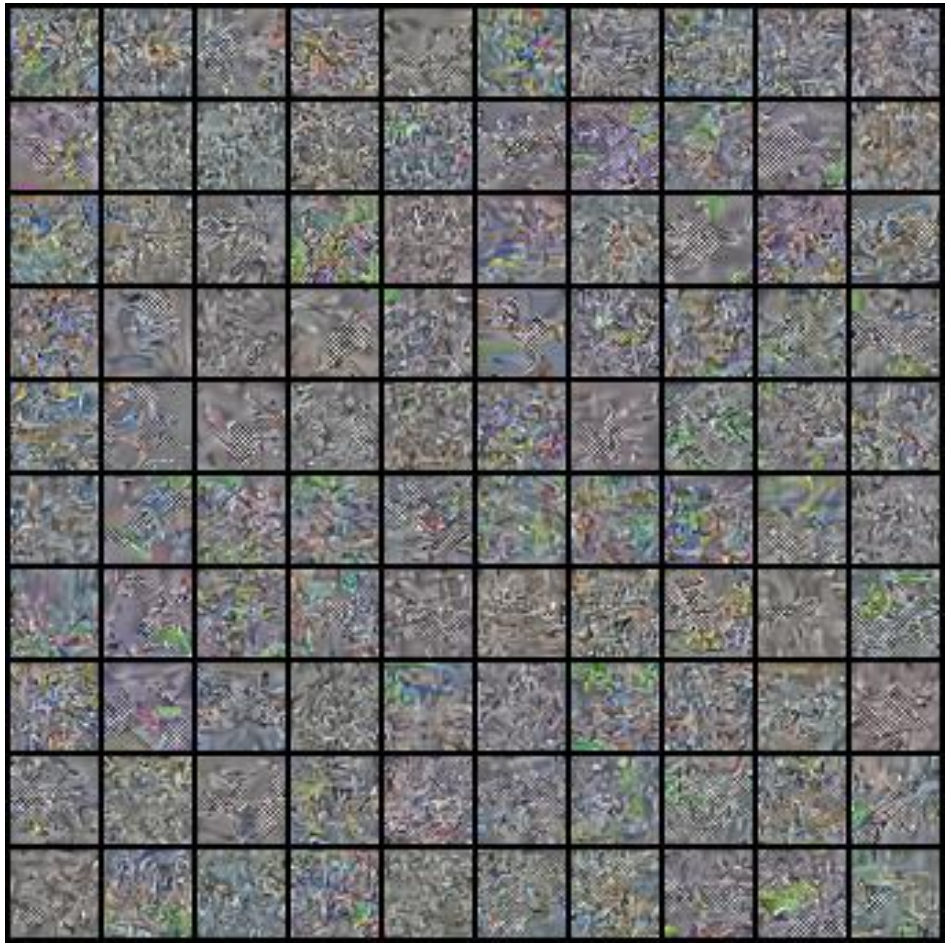
$$x_{i+1} \leftarrow x_i + \epsilon \nabla_x \log p(x) + \sqrt{2\epsilon} z_i, \qquad i = 0,1,\ldots K$$

$$z_i \sim \mathcal{N}(0, I)$$

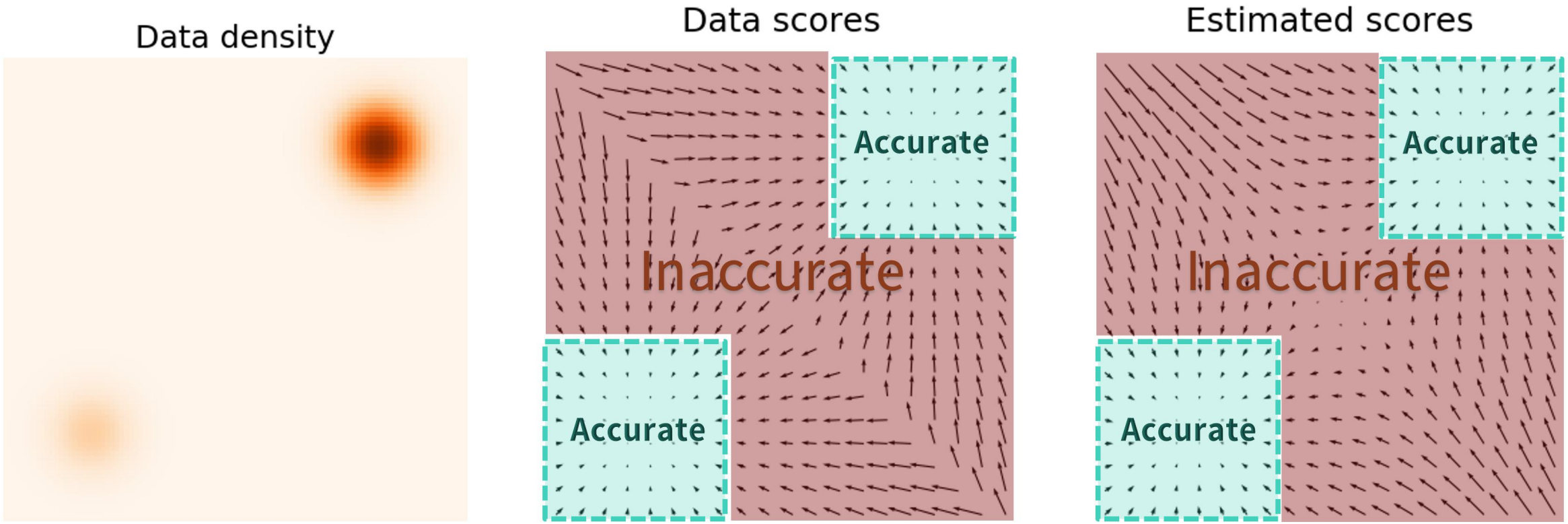
$\epsilon \to 0$ and $K \to \infty$ converge

Naive score-based generative modeling

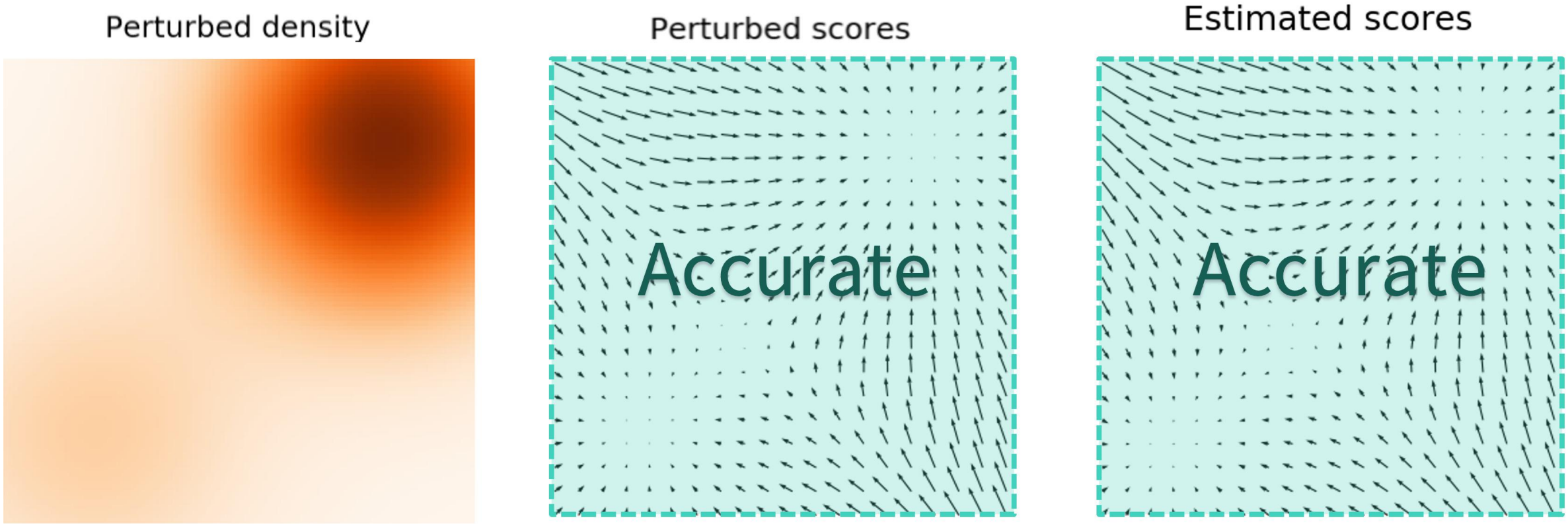
Model samples



Challenge in low data density regions



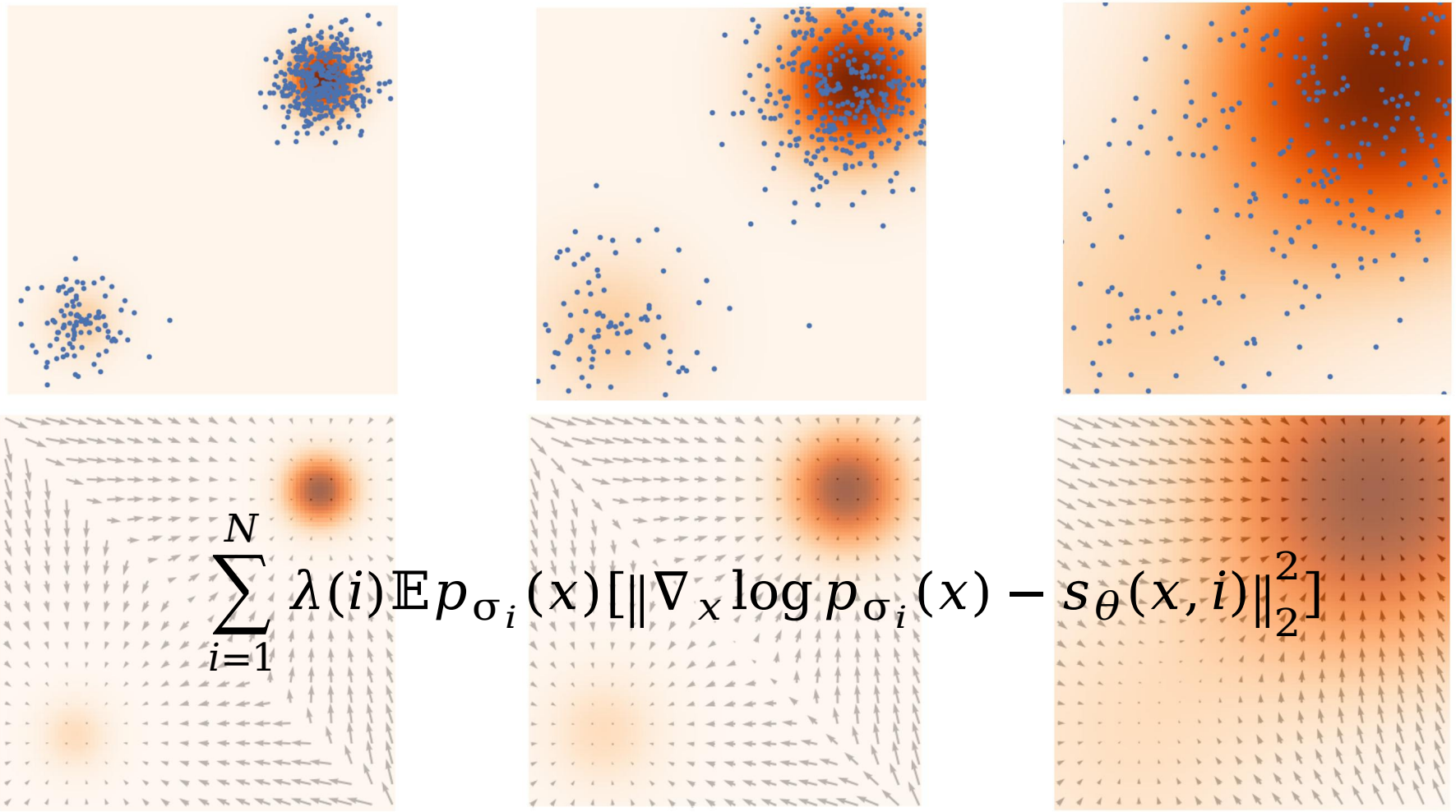
Improving score estimation by add noise

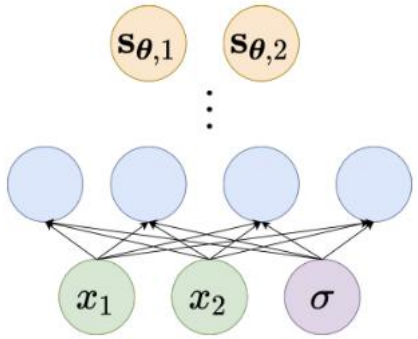


Using multiple noise scales

$$\sigma_1 < \sigma_2 < \sigma_3$$

Data





Noise Conditional Score Networks (NCSN)

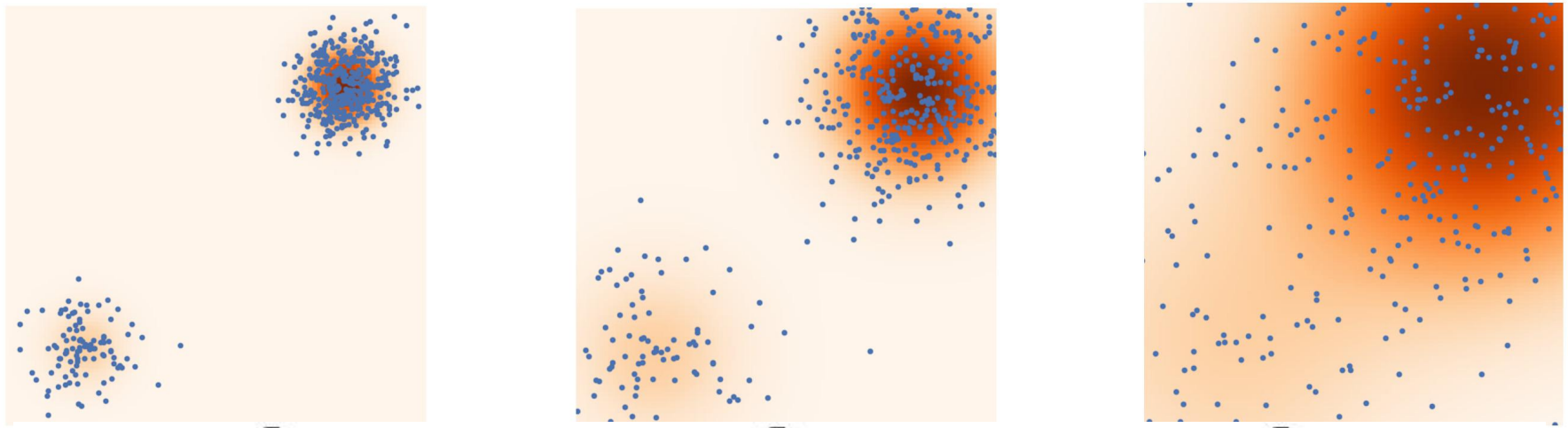
$$\sum_{i=1}^{N} \lambda(i) \mathbb{E} p_{\sigma_i}(x) [\| \nabla_x \log p_{\sigma_i}(x) - s_\theta(x, i) \|_2^2]$$

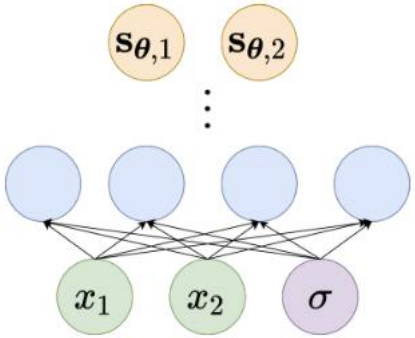
Yang Song et al. Generative Modeling by Estimating Gradients of the Data Distribution, NIPS 2019 Oral

Using multiple noise scales

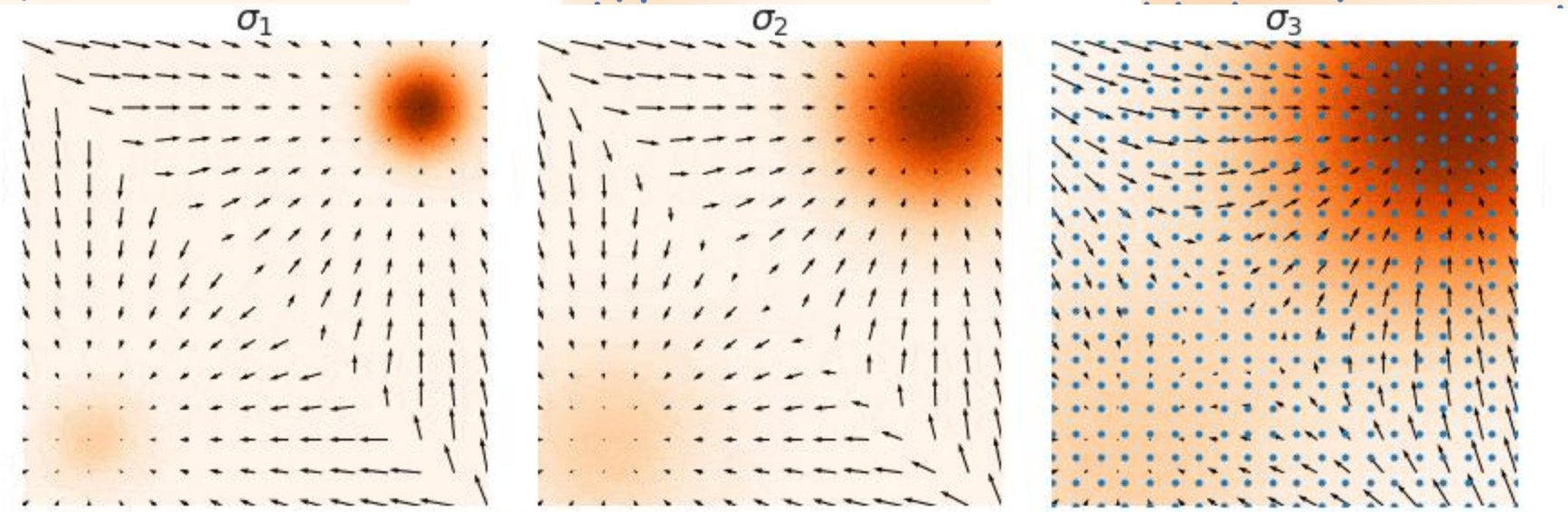
$\sigma_1 < \sigma_2 < \sigma_3$

Data





Noise Conditional Score Networks (NCSN)



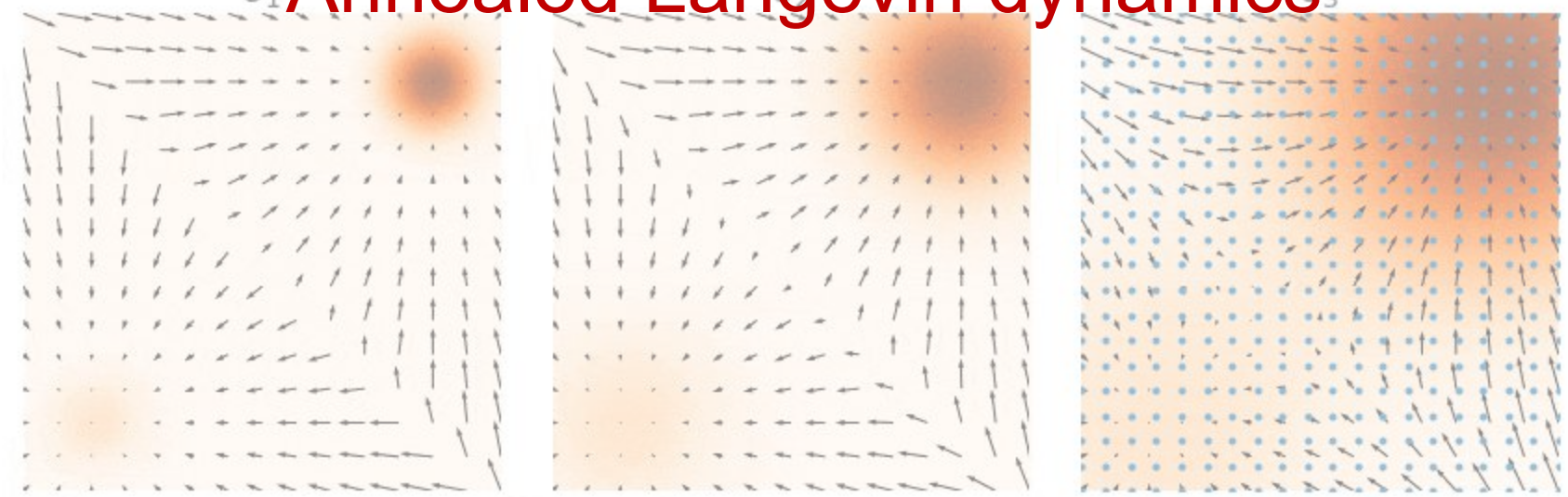
Yang Song et al. Generative Modeling by Estimating Gradients of the Data Distribution, NIPS 2019 Oral

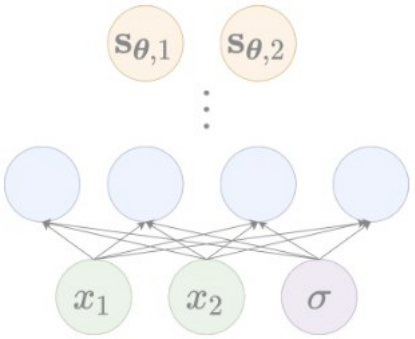
Using multiple noise scales

$\sigma_1 < \sigma_2 < \sigma_3$

Data

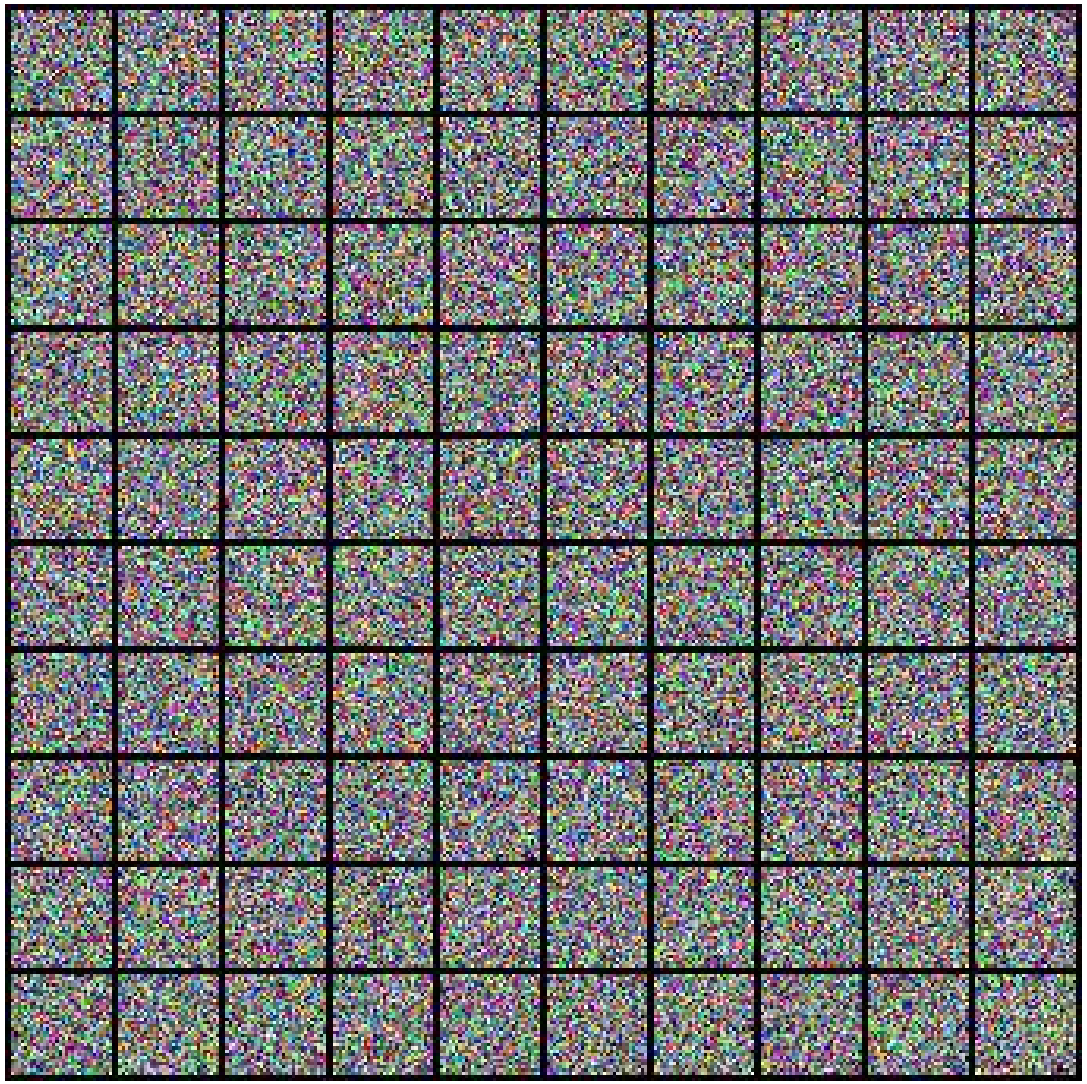
Annealed Langevin dynamics



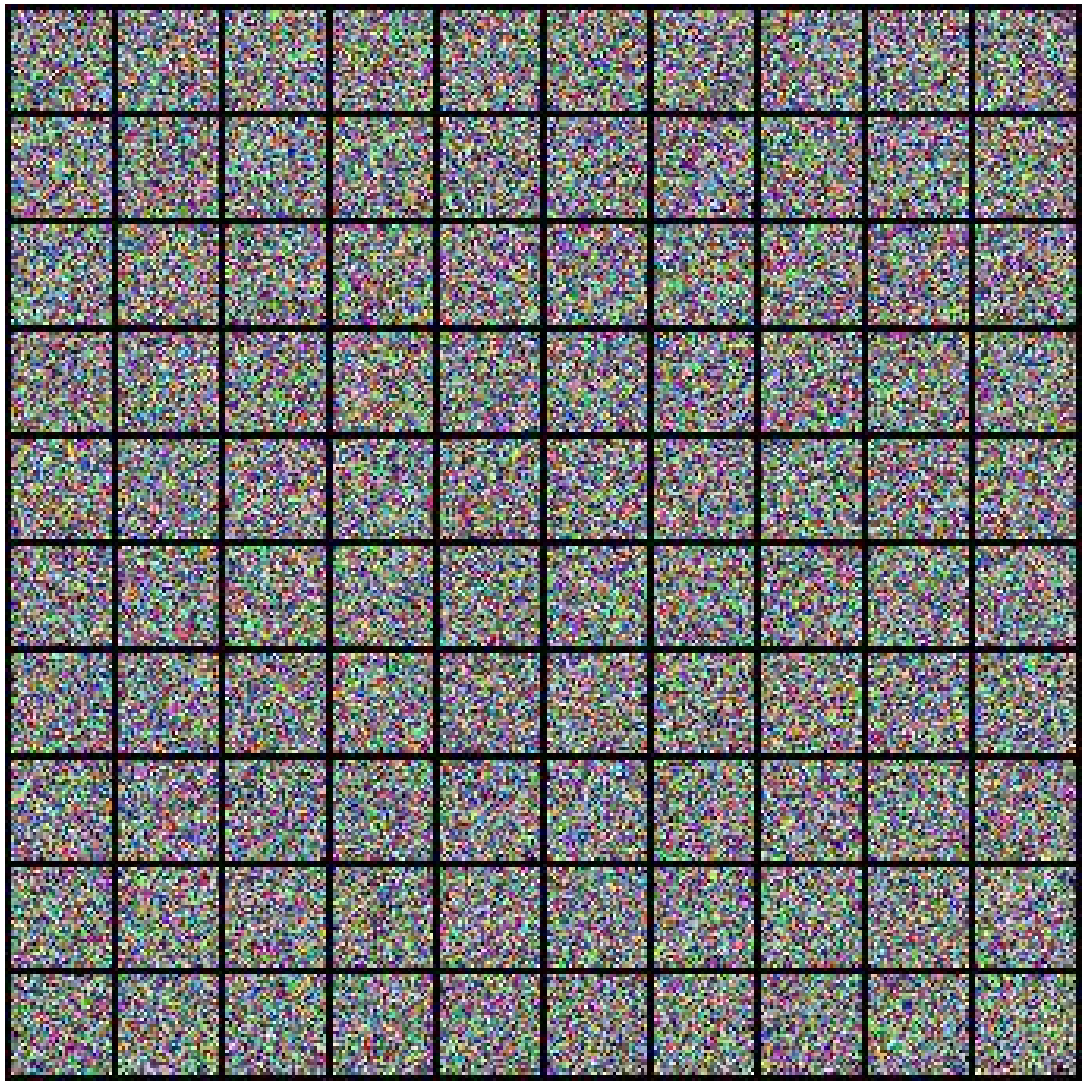


Noise Conditional Score Networks (NCSN)

Annealed Langevin dynamics for the Noise Conditional Score Network (NCSN) model

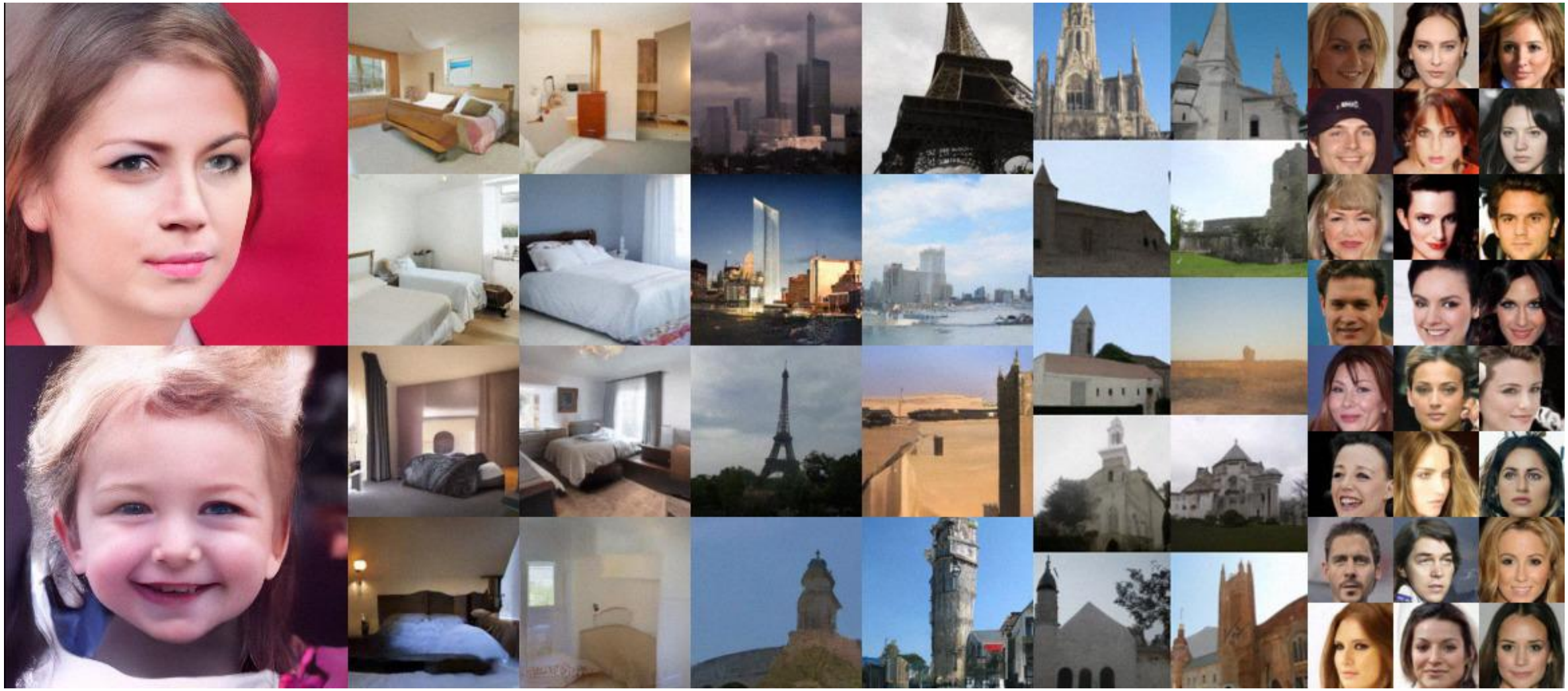


Trained on CelebA



Trained on CIFAR-10
(Beat GANs for the first time)

High resolution image generation



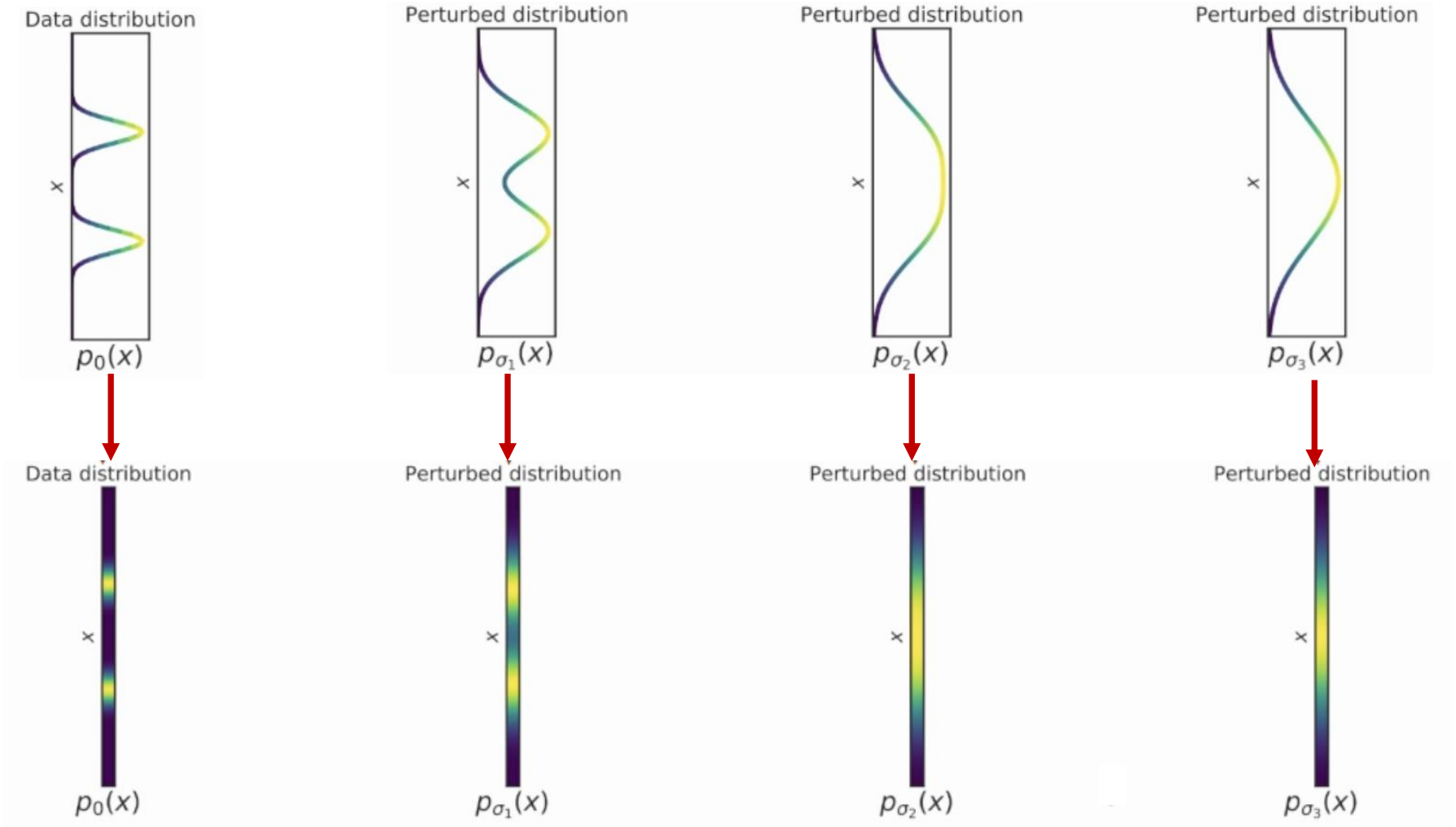
256 × 256

128 × 128

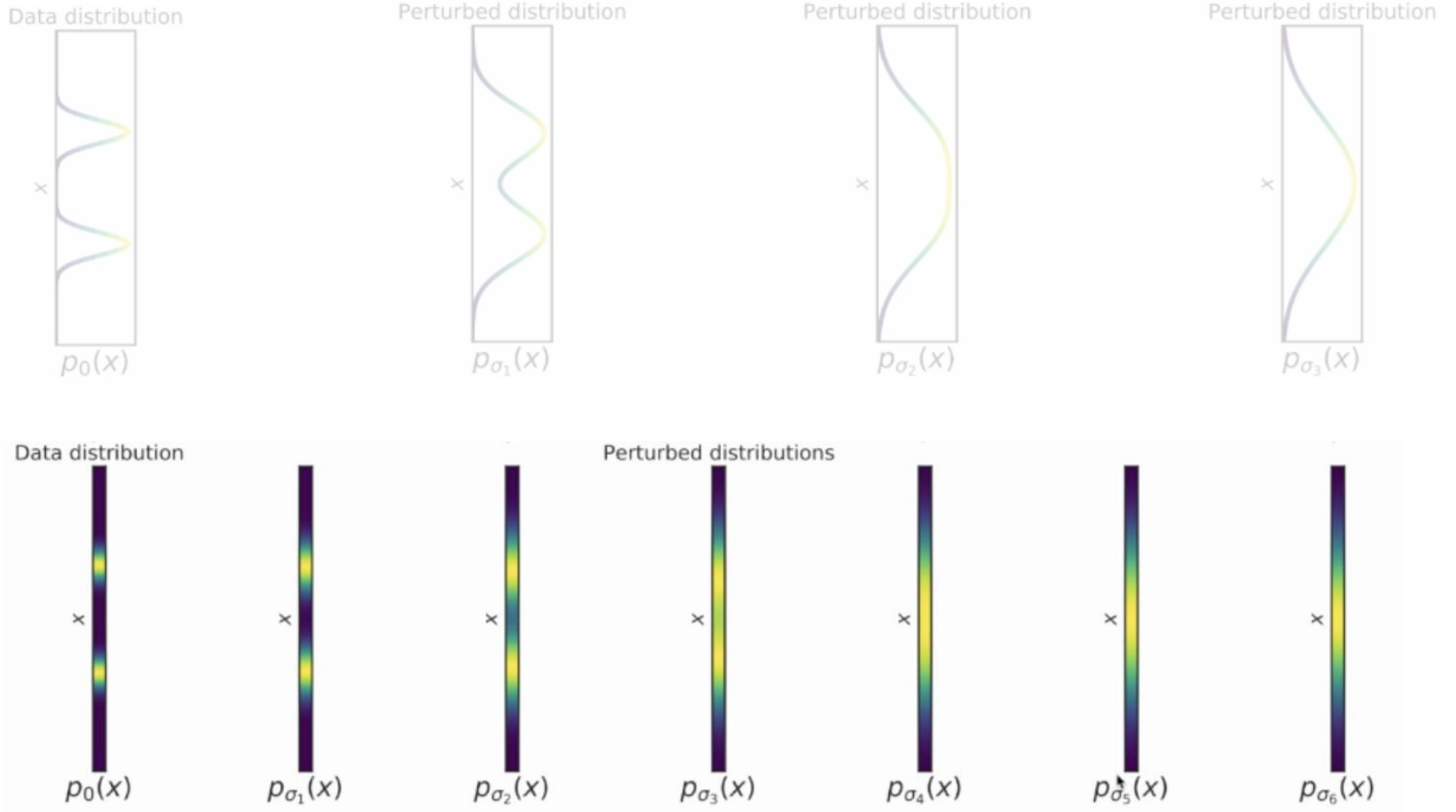
96 × 96

64 × 64

Infinitely many noise scales

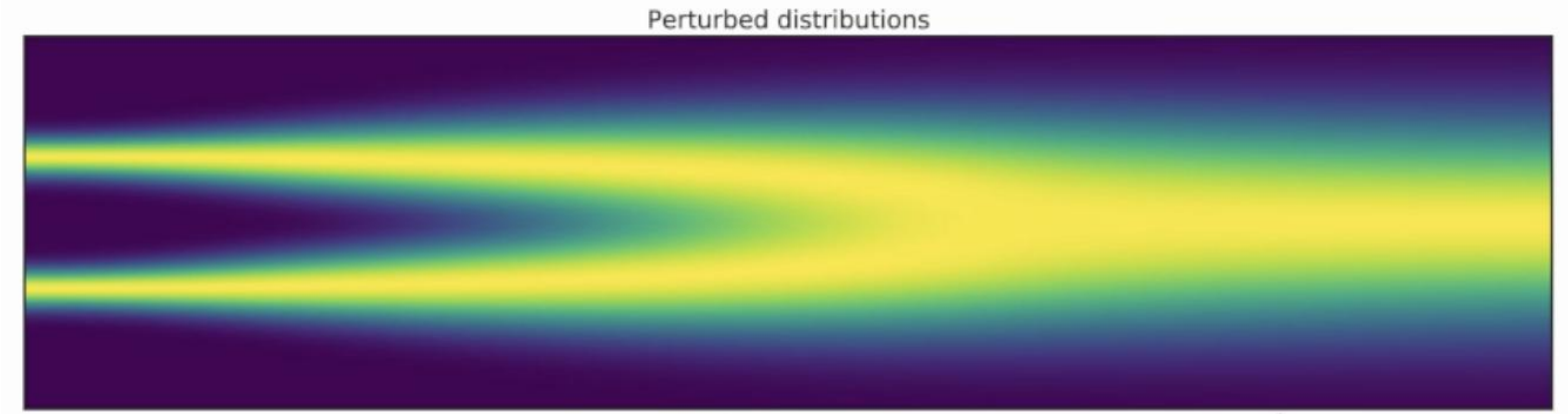


Infinitely many noise scales



Infinitely many noise scales

$p_0(x)$
=
$p\_\mathrm{data}(x)$

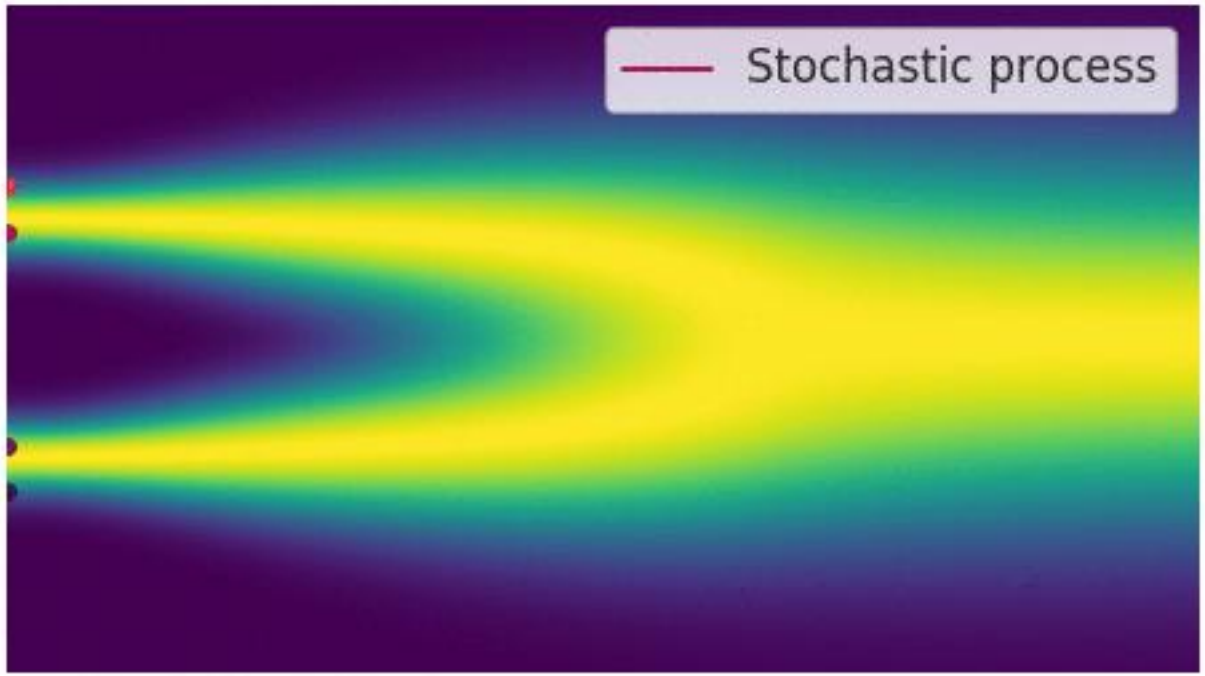


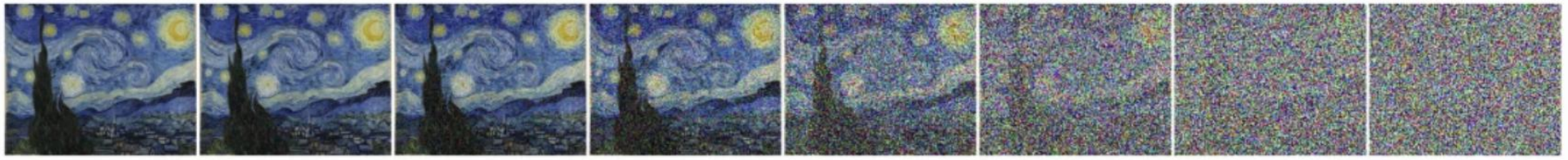
$p_T(x)$

$p_t(x)$

$t \in [0, T]$: continuous index of perturbed distributions

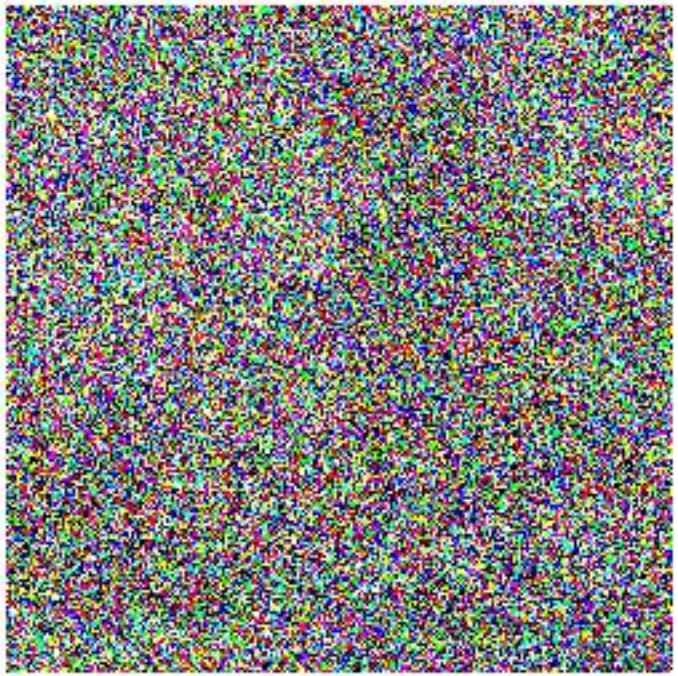
Perturbing data with an SDE

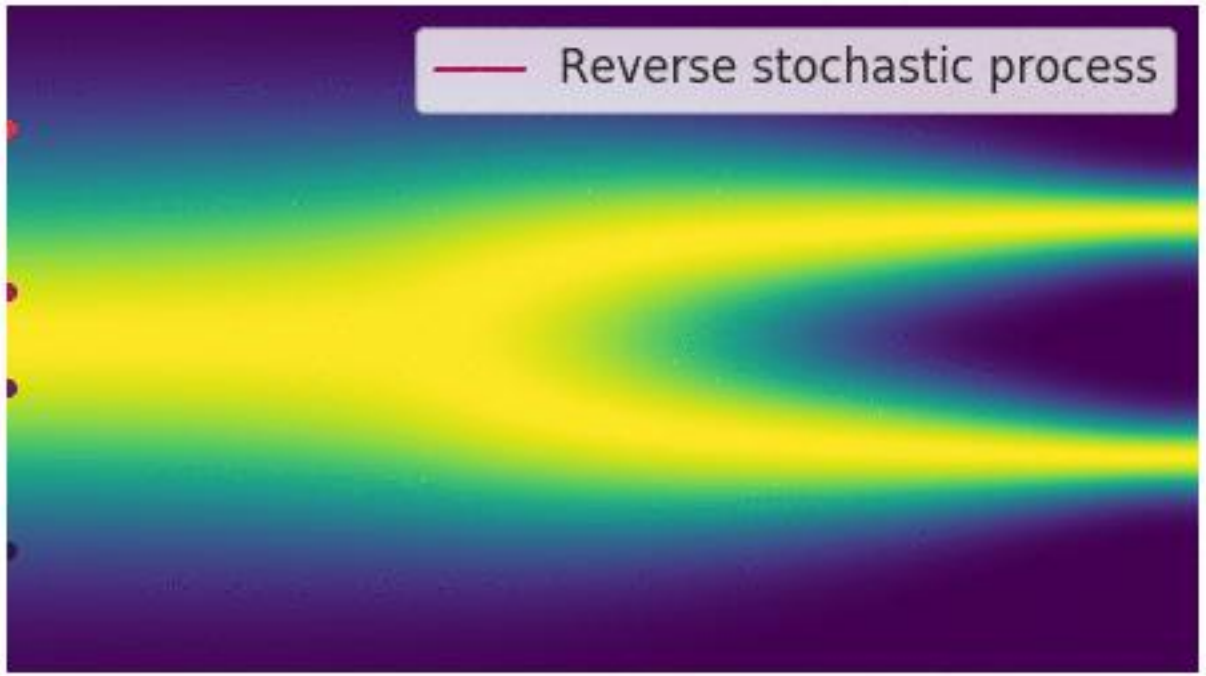




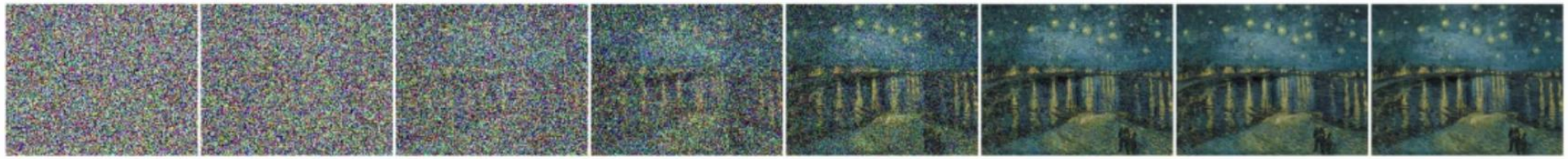
$$\mathrm{d}x = \sigma(t)\mathrm{d}w$$

Score-based generative modeling via SDEs





$\pi(x) \approx p_T(x)$ $p_\theta(x)$



$$\mathrm{d}x = \sigma(t)\mathrm{d}w$$

Time reversal

$$\mathrm{d}x = -\sigma^2(t)\nabla_x \log p_t(x)\mathrm{d}t + \sigma(t)\mathrm{d}w$$

Score function

Score-based generative modeling via SDEs

Time-dependent score model:

$$s_\theta(x,t) \approx \nabla_x \log p_t(x)$$

Training:

$$\mathbb{E}_{t\sim \mathrm{Uniform}[0,T]}[\lambda(t)\mathbb{E}_{p_t(x)}[\|\nabla_x \log p_t(x) - s_\theta(x,t)\|_2^2]$$

Reverse-time SDE:

$$\mathrm{d}x = -\sigma^2(t)s_\theta(x,t)\mathrm{d}t + \sigma(t)\mathrm{d}w$$

Numerical SDE solver for sample generation.

Unifies diffusion probabilistic models into the framework of score-based generative models

Reference

DDPM and its extension:

DDPM: Jouathan Ho et al. Denoising Diffusion Probabilistic Models, ICML 2020 (UCB)

DDIM: Jiaming Song et al. DENOISING DIFFUSION IMPLICIT MODELS, ICLR 2021 (Stanford, Ermon's group)

Multinomial Diffusion: Emiel Hoogeboom et al. Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions NIPS 2021 (Welling's group)

DP3M: Jacob Austin et al. Structured Denoising Diffusion Models in Discrete State-Spaces, NIPS 2021(Google Research, Brain Team)

Score-based model:

Sliced score matching: Yang Song et al. Sliced Score Matching: A Scalable Approach to Density and Score Estimation, UAI 2019 Oral (Stanford, Ermon's group)

NCSN: Yang Song et al. Generative Modeling by Estimating Gradients of the Data Distribution, NIPS 2019 Oral (Stanford, Ermon's group)

Score-based model: Yang Song et al. SCORE-BASED GENERATIVE MODELING THROUGH STOCHASTIC DIFFERENTIAL EQUATIONS, ICML 2021 Oral, Outstanding paper (Stanford, Ermon's group, Google Brain)

Other interesting works:

Analytic-DPM: Fan Bao et al. Analytic-DPM: an Analytic Estimate of the Optimal Reverse Variance in Diffusion Probabilistic Models, ICLR 2022, Outstanding paper (Tsinghua, Zhu's group)

CARD: CARD: Xizewen Han et al. Classification and Regression Diffusion Models NIPS 2022(UT, Zhou's group)

Blog:

https://lilianweng.github.io/posts/2021-07-11-diffusion-models/#nice

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

Appendix (Derivation-1)

$$
\begin{aligned}
\mathbf{x}_t &= \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\mathbf{z}_{t-1} && ;\text{where } \mathbf{z}_{t-1}, \mathbf{z}_{t-2}, \cdots \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\mathbf{z}}_{t-2} && ;\text{where } \bar{\mathbf{z}}_{t-2} \text{ merges two Gaussians } (*). \\
&= \dots \\
&= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\mathbf{z} \\
q(\mathbf{x}_t|\mathbf{x}_0) &= \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})
\end{aligned}
$$

Appendix (Derivation-2)

$$
\begin{aligned}
q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) &= q(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{x}_0)\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \\
&\propto \exp\Big(-\frac{1}{2}\big(\frac{(\mathbf{x}_t-\sqrt{\alpha_t}\mathbf{x}_{t-1})^2}{\beta_t}+\frac{(\mathbf{x}_{t-1}-\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0)^2}{1-\bar{\alpha}_{t-1}}-\frac{(\mathbf{x}_t-\sqrt{\bar{\alpha}_t}\mathbf{x}_0)^2}{1-\bar{\alpha}_t}\big)\Big) \\
&= \exp\Big(-\frac{1}{2}\big(\frac{\mathbf{x}_t^2-2\sqrt{\alpha_t}\mathbf{x}_t\mathbf{x}_{t-1}+\alpha_t\mathbf{x}_{t-1}^2}{\beta_t}+\frac{\mathbf{x}_{t-1}^2-2\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0\mathbf{x}_{t-1}+\bar{\alpha}_{t-1}\mathbf{x}_0^2}{1-\bar{\alpha}_{t-1}}-\frac{(\mathbf{x}_t-\sqrt{\bar{\alpha}_t}\mathbf{x}_0)^2}{1-\bar{\alpha}_t}\big)\Big) \\
&= \exp\Big(-\frac{1}{2}\big((\frac{\alpha_t}{\beta_t}+\frac{1}{1-\bar{\alpha}_{t-1}})\mathbf{x}_{t-1}^2-(\frac{2\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t+\frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0)\mathbf{x}_{t-1}+C(\mathbf{x}_t,\mathbf{x}_0)\big)\Big)
\end{aligned}
$$

Appendix (Derivation-3)

$$\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon})-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}\right)-\boldsymbol{\mu}_\theta(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon}),t)\right\|^2\right] \quad \bigstar$$

$$\mu_\theta \rightarrow \frac{1}{\sqrt{\bar{\alpha}_t}}\left(x_t-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right),\ \text{given } \mathrm{x_t}$$

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

Appendix

Uniform diffusion

$$[\boldsymbol{Q}_t]_{ij} = \begin{cases} 1 - \frac{K-1}{K}\beta_t & \text{if} \quad i = j \\ \frac{1}{K}\beta_t & \text{if} \quad i \neq j \end{cases}$$

Diffusion with an absorbing state

$$[\boldsymbol{Q}_t]_{ij} = \begin{cases} 1 & \text{if} \quad i = j = m \\ 1 - \beta_t & \text{if} \quad i = j \neq m \\ \beta_t & \text{if} \quad j = m, i \neq m \end{cases}$$

Discretized Gaussian transition matrices

$$[\boldsymbol{Q}_t]_{ij} = \begin{cases} \frac{\exp\left(-\frac{4|i-j|^2}{(K-1)^2\beta_t}\right)}{\sum_{n=-(K-1)}^{K-1}\exp\left(-\frac{4n^2}{(K-1)^2\beta_t}\right)} & \text{if} \quad i \neq j \\ 1 - \sum_{l=0, l\neq i}^{K-1}[\boldsymbol{Q}_t]_{il} & \text{if} \quad i = j \end{cases}$$

Appendix

Illustration of Multinomial Diffusion

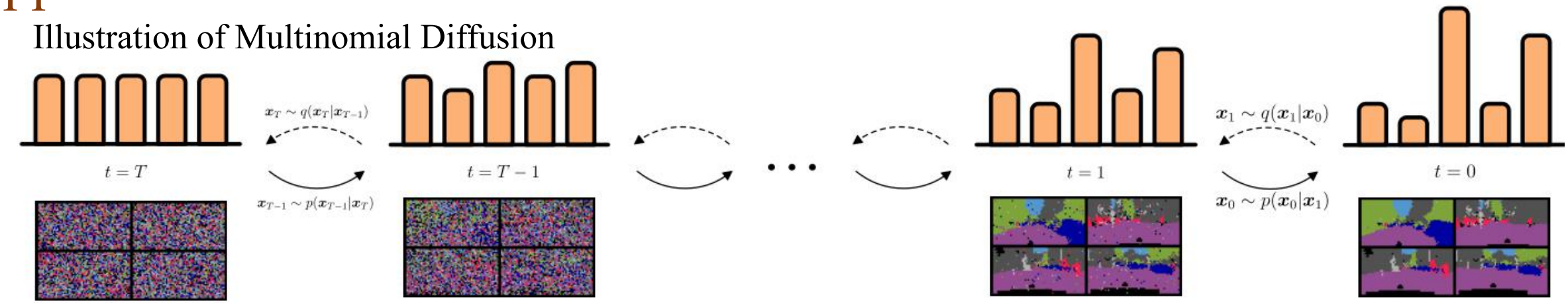
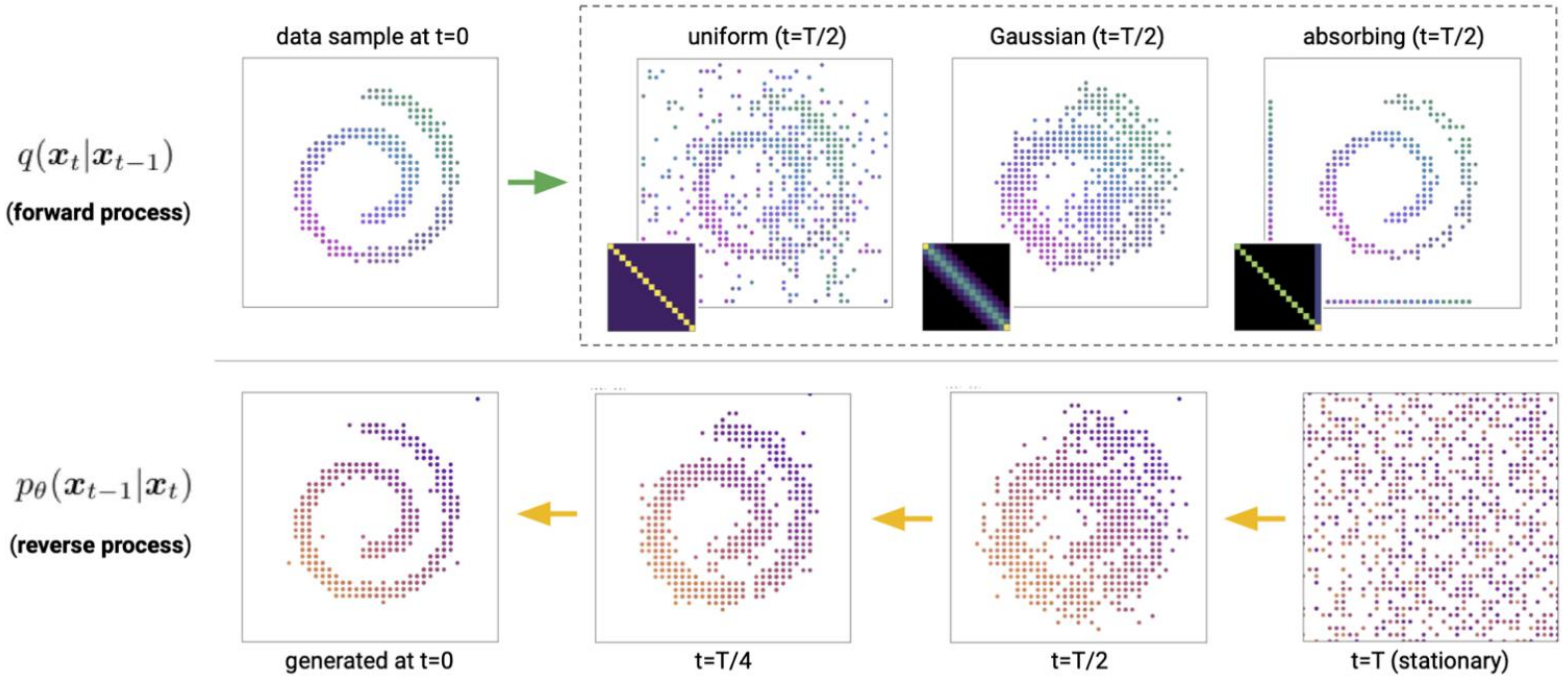
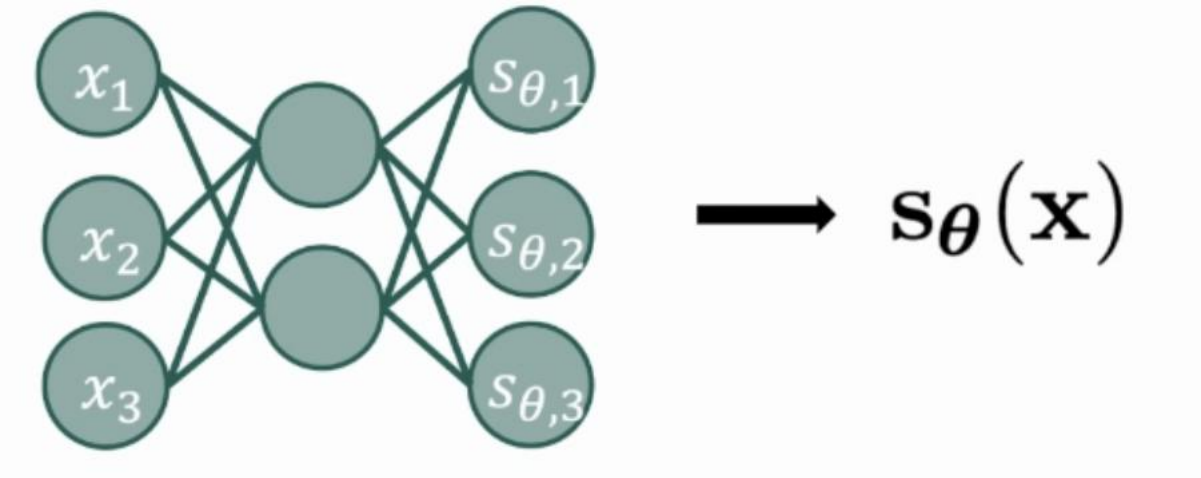


Illustration of DP3M

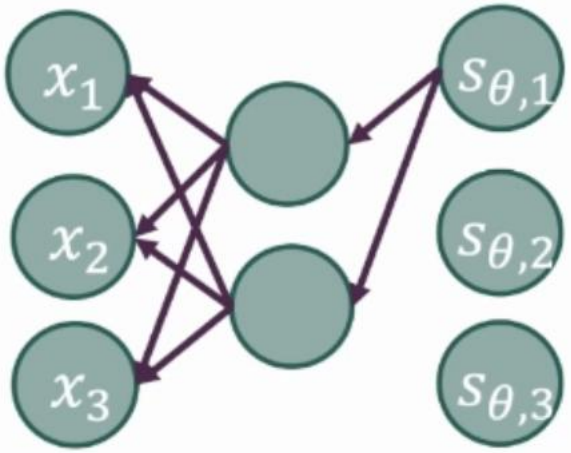


Appendix


x_1
x_2
x_3
$s_{\theta,1}$
$s_{\theta,2}$
$s_{\theta,3}$
$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$

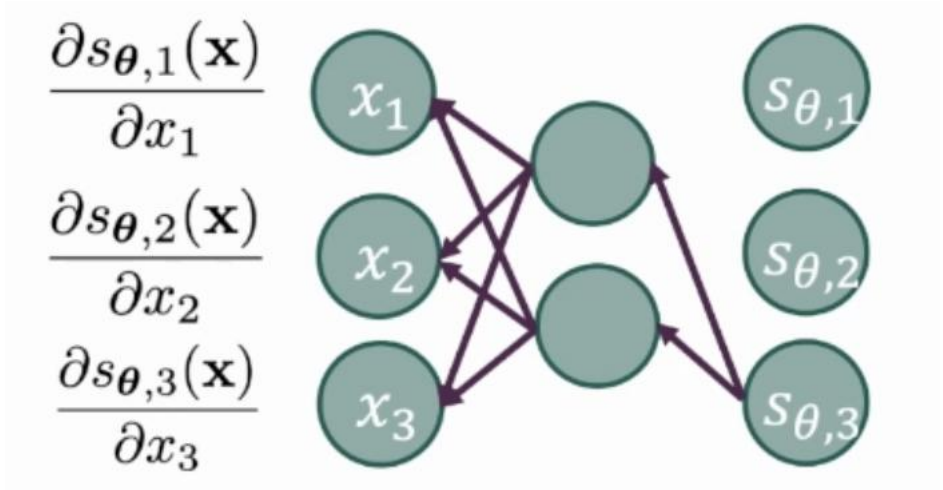
Appendix

$$\frac{\partial s_{\boldsymbol{\theta},1}(\mathbf{x})}{\partial x_1}$$



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

Appendix

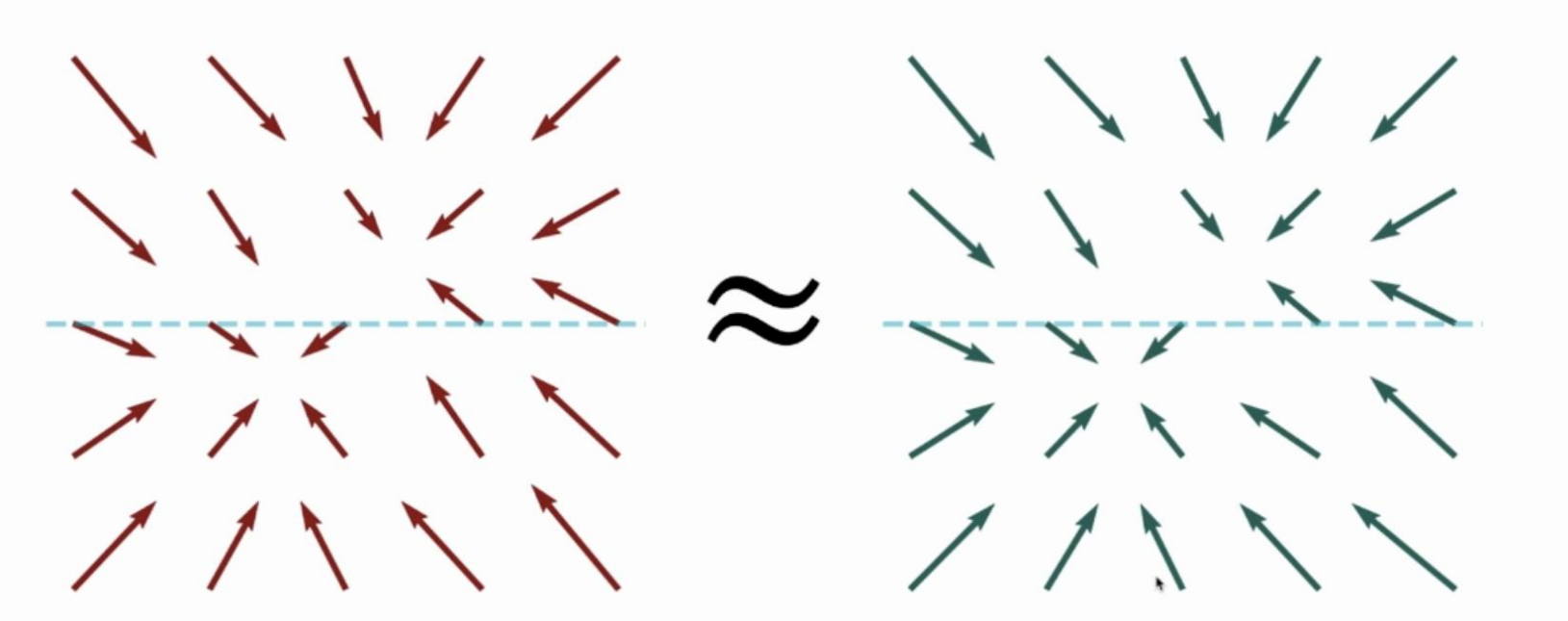


$$O(\#\text{dimensions of } \mathbf{x})$$

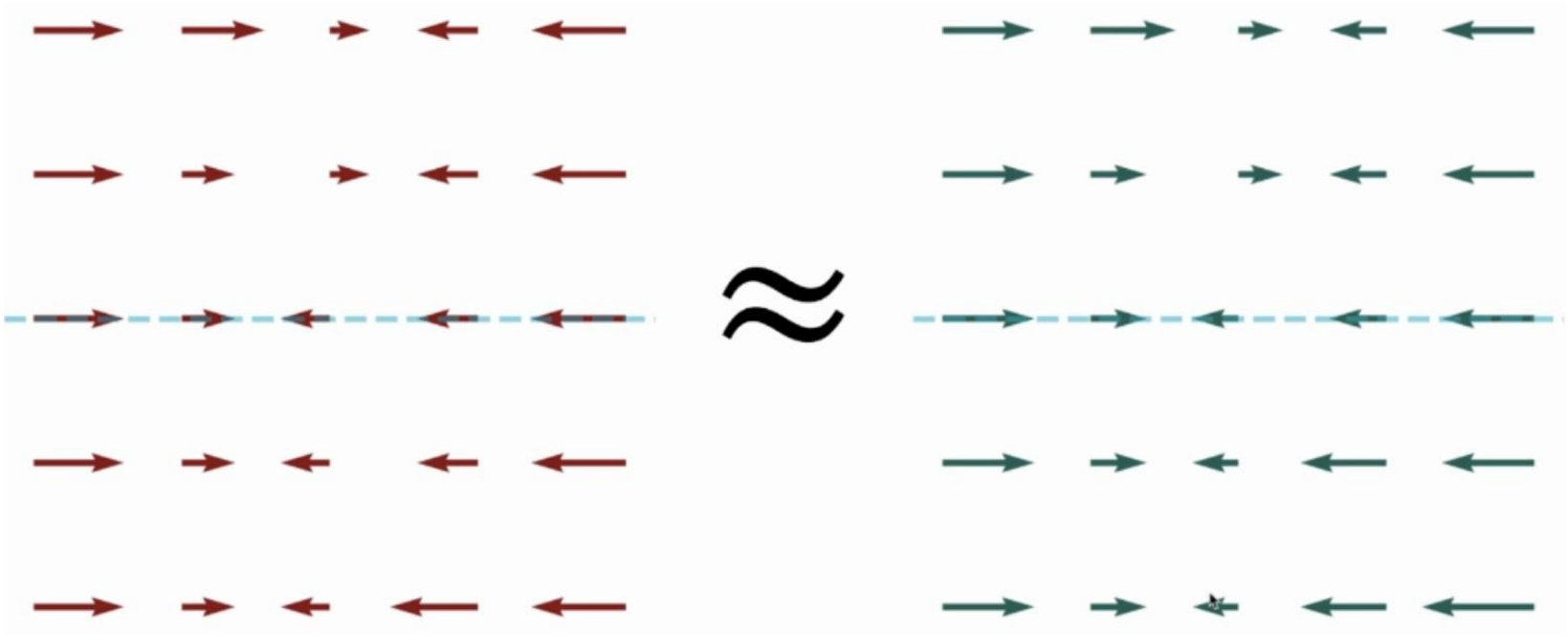
Backprops!

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

Appendix

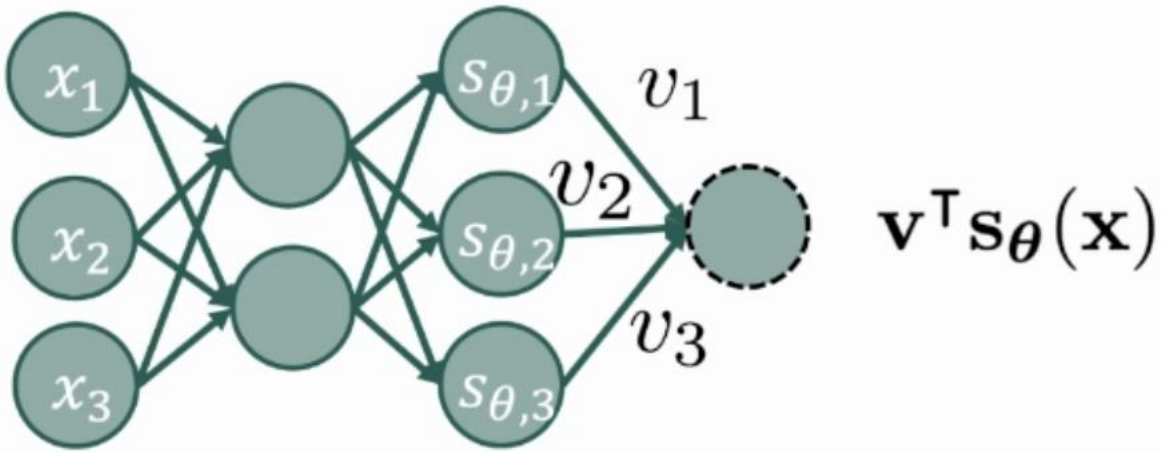


Appendix



Appendix

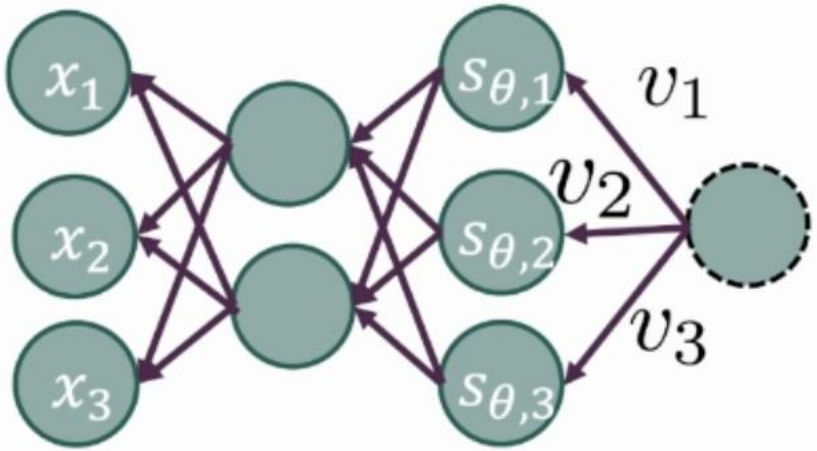
$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\mathbf{v} = \mathbf{v}^\intercal \nabla_{\mathbf{x}} \left(\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\right)$$



Appendix

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \mathbf{v} = \mathbf{v}^\intercal \nabla_{\mathbf{x}} \left(\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \right)$$

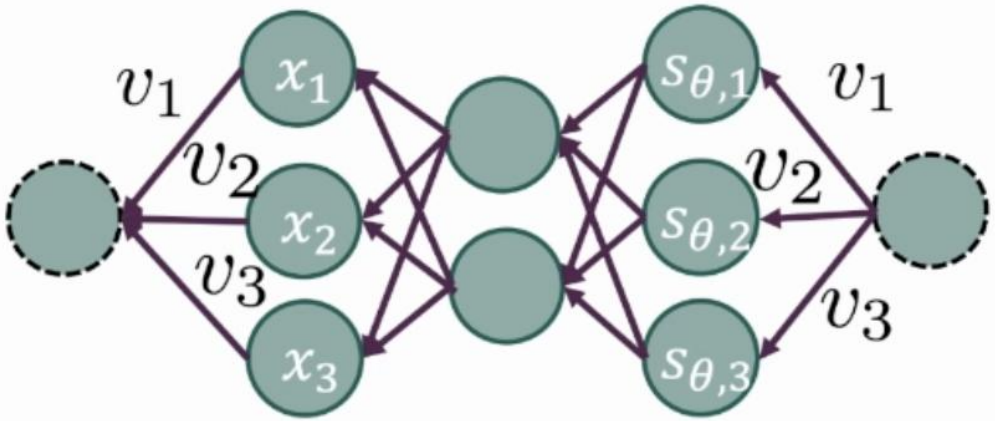
$\nabla_{\mathbf{x}} \left(\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) \right)$



Appendix

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\mathbf{v} = \nabla_{\mathbf{x}}(\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))\mathbf{v}$$

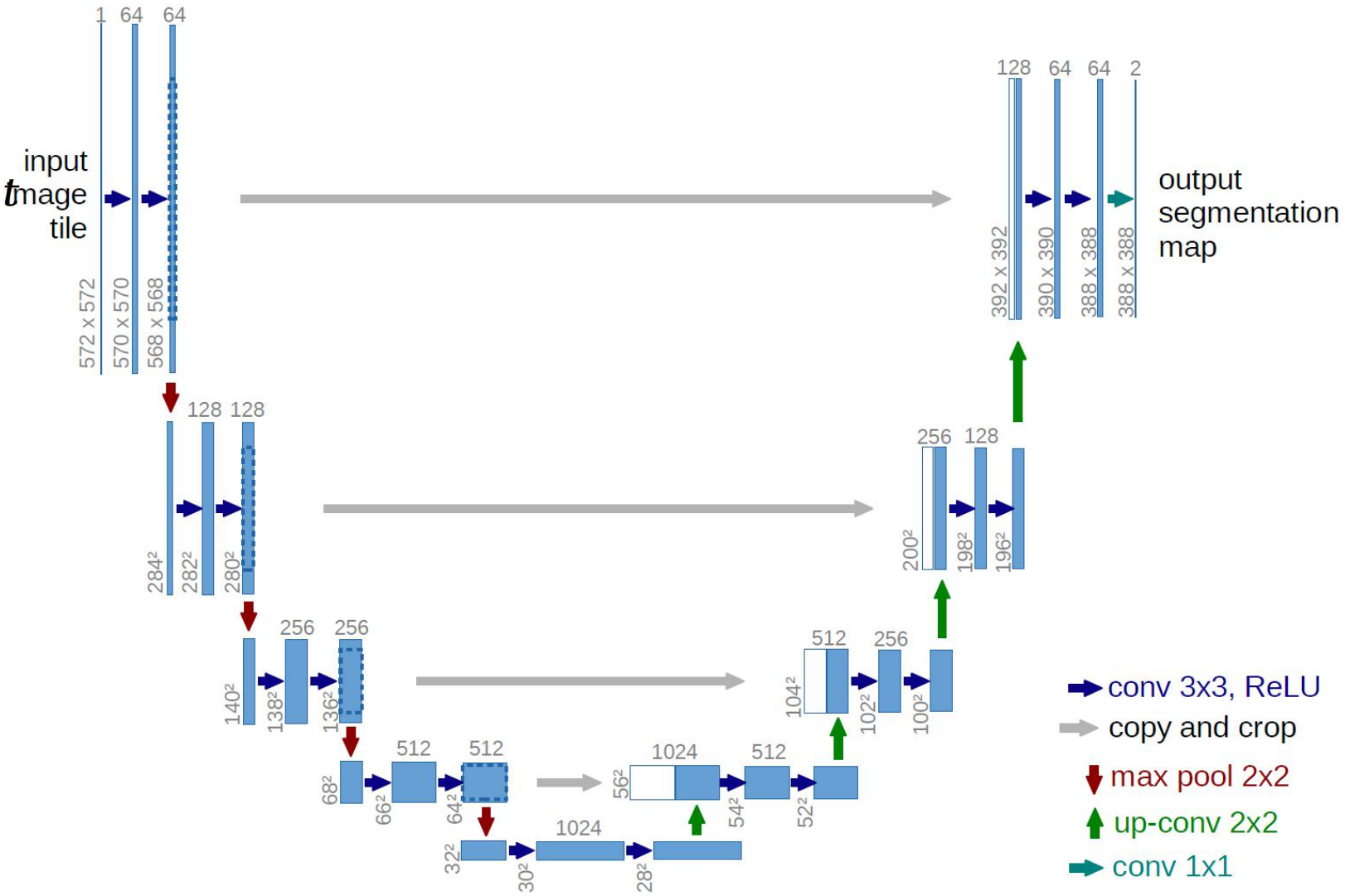
$\mathbf{v}^\intercal \nabla_{\mathbf{x}}(\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$



Appendix

$\mu_\theta\ (x_t, t)$: U-Net

$Dim^{input} = Dim^{output}$



Appendix

DDPM