Introduction to diffusion models

François Rousseau

Ressources

Diffusion Models: A Comprehensive Survey of Methods and Applications

LING YANG*, Peking University, China
ZHILONG ZHANG*, Peking University, China
YANG SONG, OpenAI, USA
SHENDA HONG, Peking University, China
RUNSHENG XU, University of California, Los Angeles, USA
YUE ZHAO, Carnegie Mellon University, USA
YINGXIA SHAO, Beijing University of Posts and Telecommunications, China
WENTAO ZHANG, Mila - Quebec AI Institute, HEC Montréal, Canada
BIN CUI, Peking University, China
MING-HSUAN YANG, University of California at Merced, USA

Diffusion models have emerged as a powerful new family of deep generative models with record-breaking performance in many applications, including image synthesis, video generation, and molecule design. In this survey, we provide an overview of the rapidly expanding body of work on diffusion models, categorizing the research into three key areas: efficient sampling, improved likelihood estimation, and handling data with special structures. We also discuss the potential for combining diffusion models with other generative models for enhanced results. We further review the wide-ranging applications of diffusion models in fields spanning from computer vision, natural language processing, temporal data modeling, to interdisciplinary applications in other scientific disciplines. This survey aims to provide a contextualized, in-depth look at the state of diffusion models, identifying the key areas of focus and pointing to potential areas for further exploration. Github: https://github.com/YangLing0818/Diffusion-Models-Papers-Survey-Taxonomy.

CVPR 2022 Tutorial:

Denoising Diffusion-based Generative Modeling: Foundations and Applications

Date: Sunday June 19, 8:30 am - 12:10 pm (CDT)

CVPR JUNE 19-24 2022 NEW ORLEANS LOUISIANA

Diffusers

Yang Song

Generative Modeling by Estimating Gradients of the Data Distribution

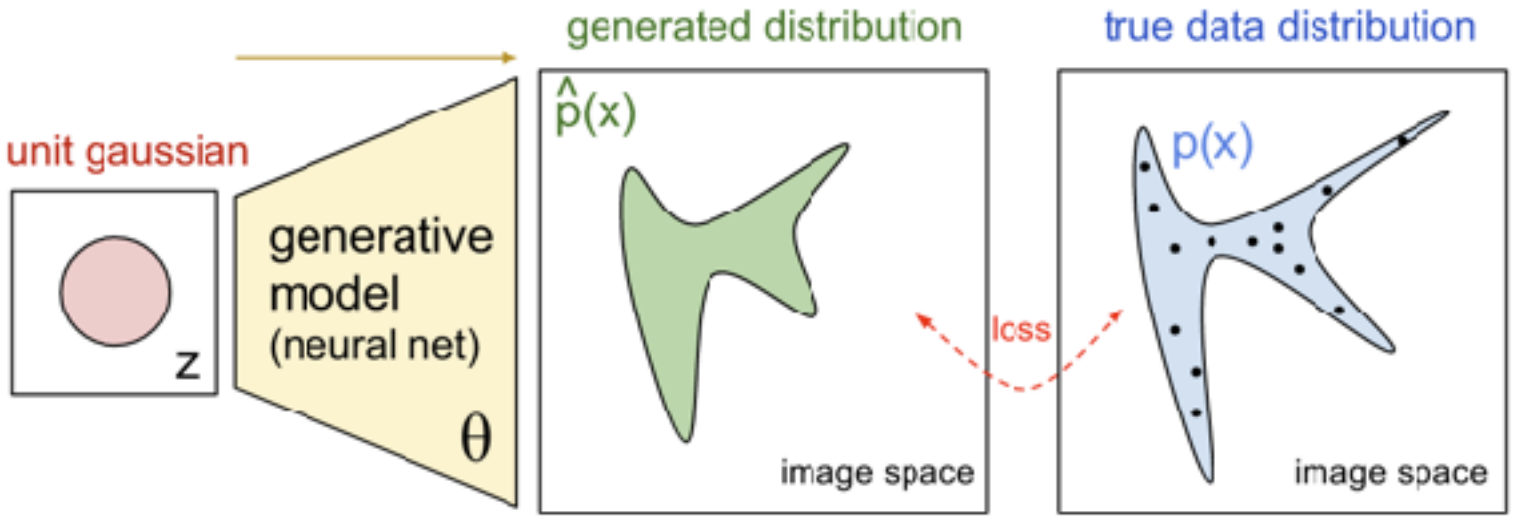
LilLog

What are Diffusion Models?

Stochastic Differential Equations and Diffusion Models

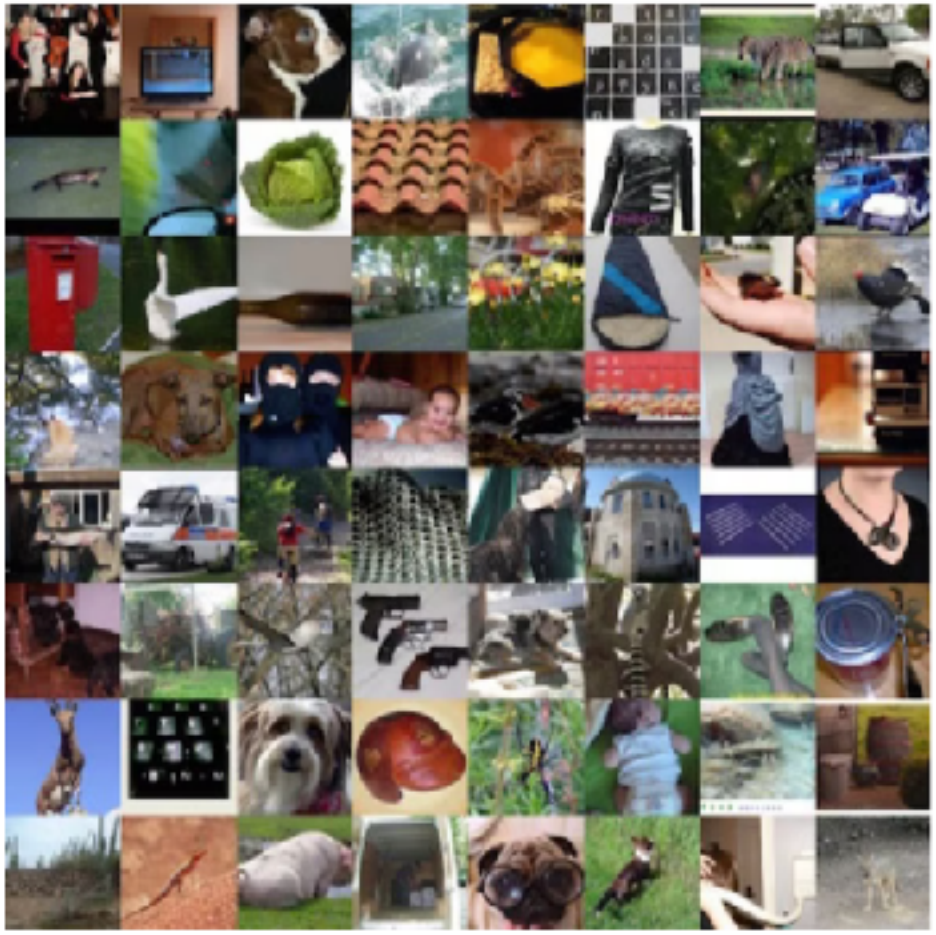
Generative modeling

Objective : draw a random sample z to generate a data x



[openAI]

Generative modeling using GAN

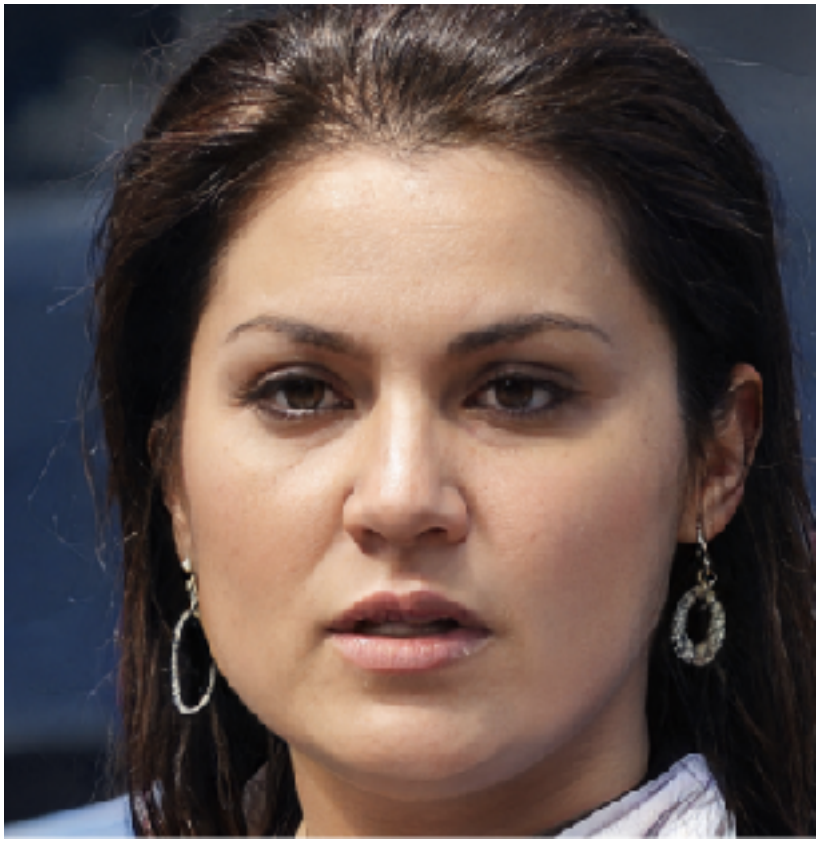


Real images (ImageNet)

Generated images

[openAI]

Generative modeling using GAN



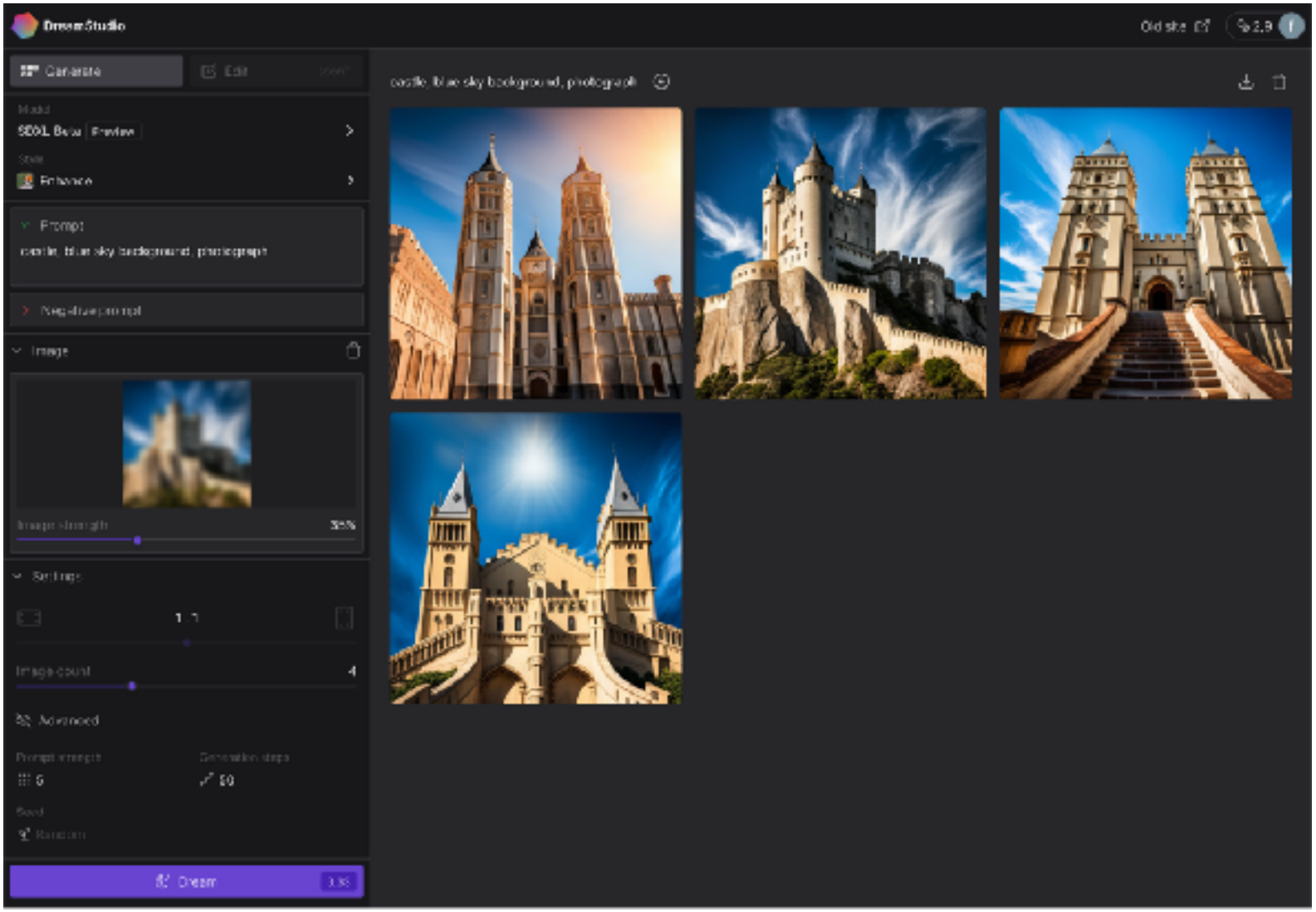
[This person does not exist !]

[Chan et al. pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis. CVPR 2021]

Generative modeling with diffusion models

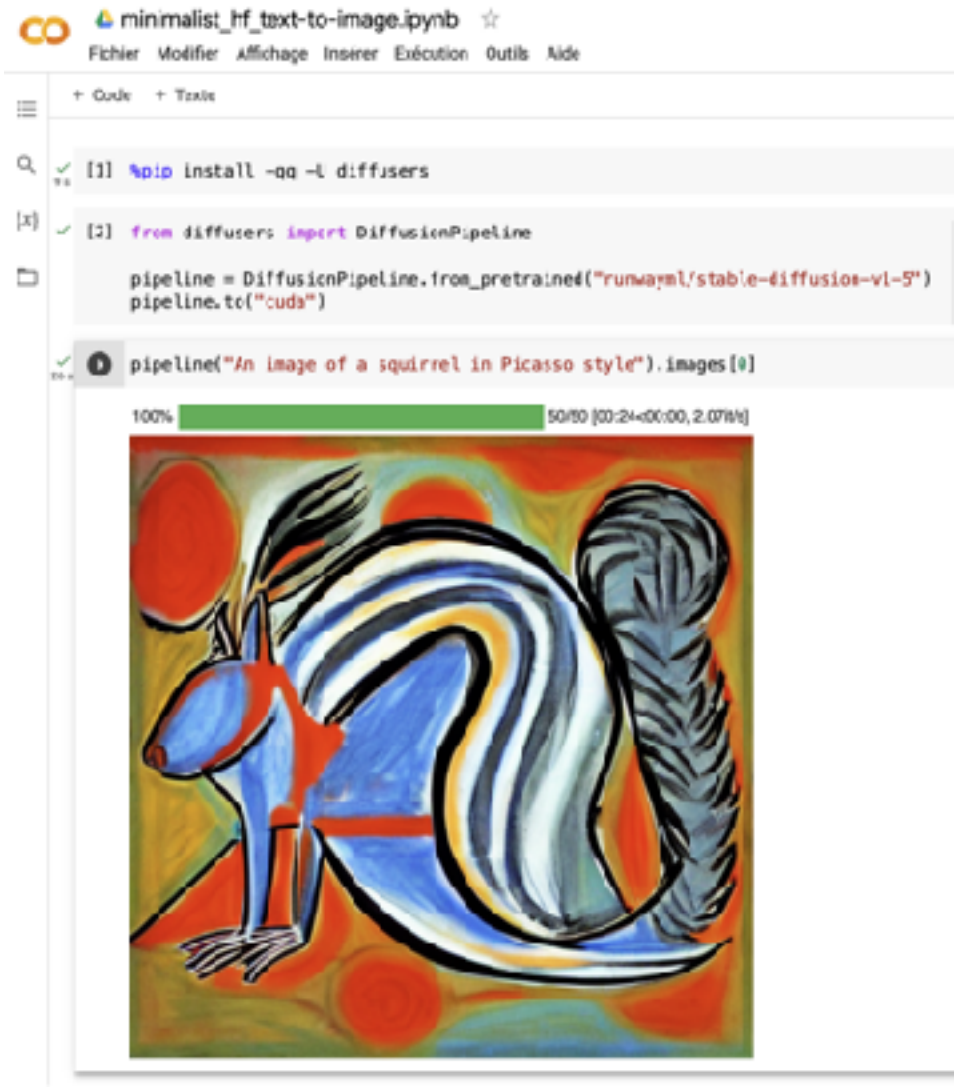
[Rombach et al. High-resolution image synthesis with latent diffusion models, CVPR 2022]

Diffusion models for everyone

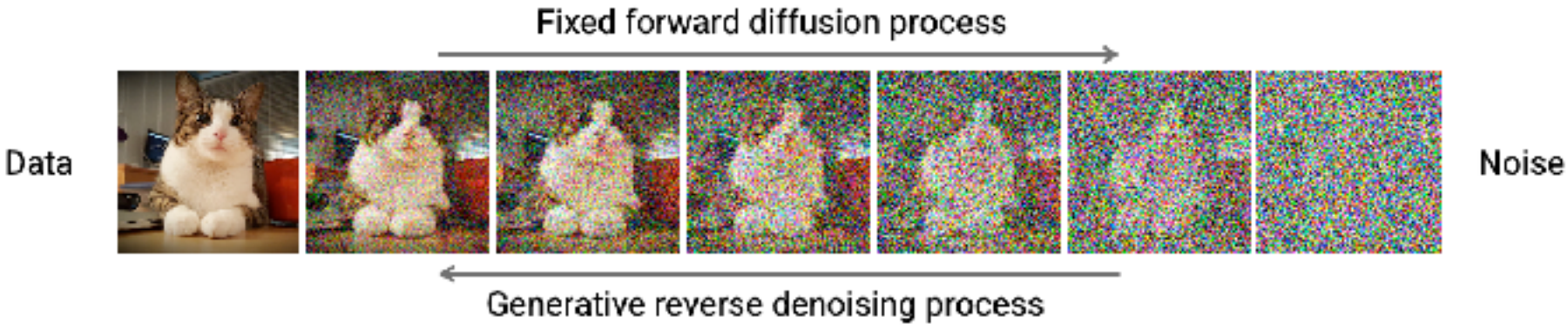


[stability.ai]

Diffusion models and Hugging Face Diffusers API

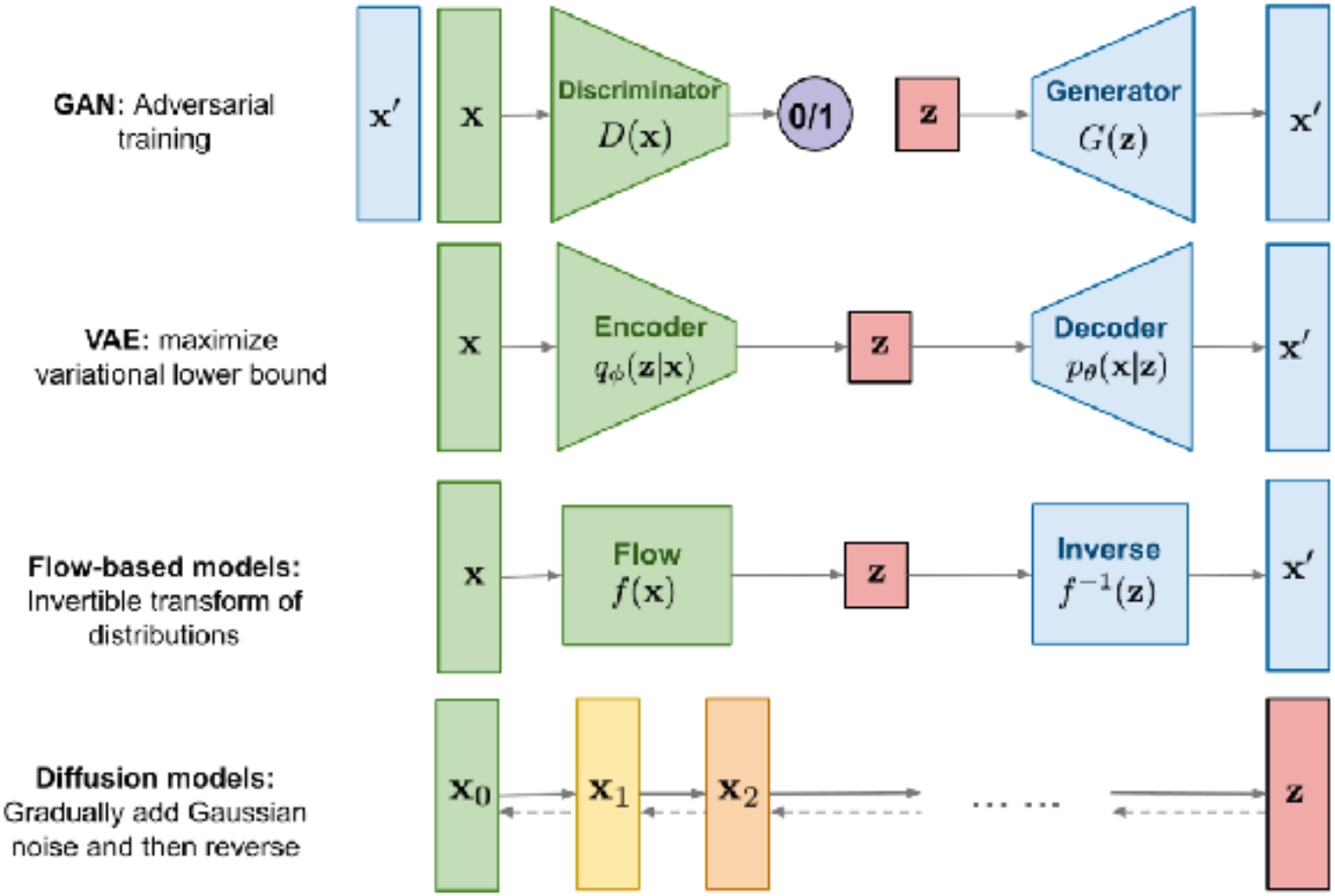


Overview of diffusion models



[CVPR 2022 Tutorial]

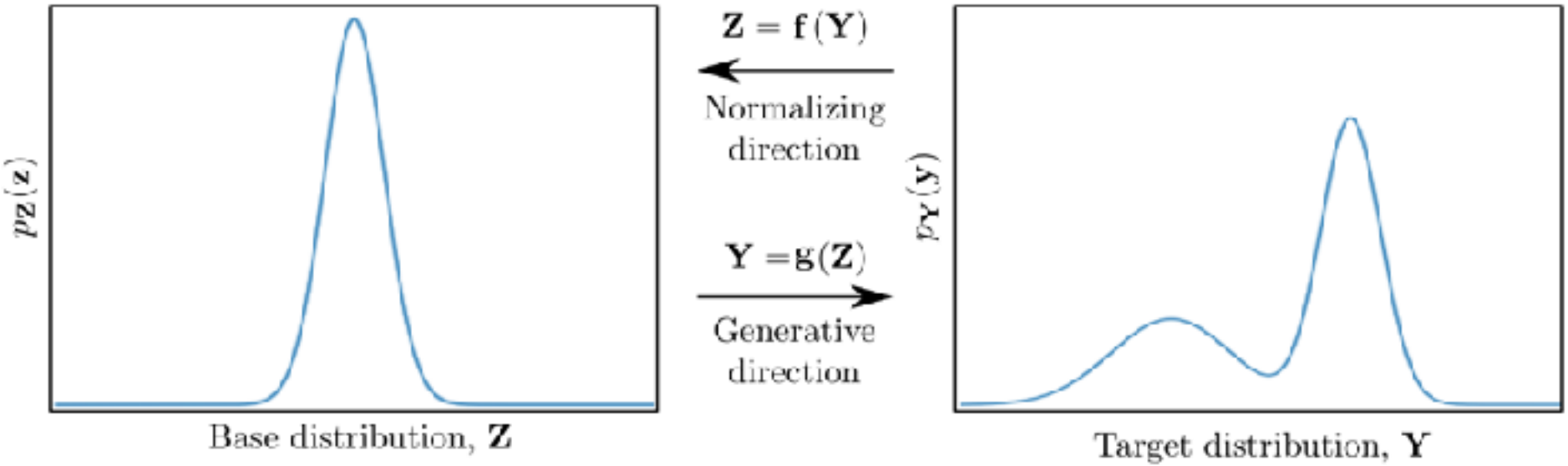
Overview of generative models



[Lilian Weng, https://lilianweng.github.io]

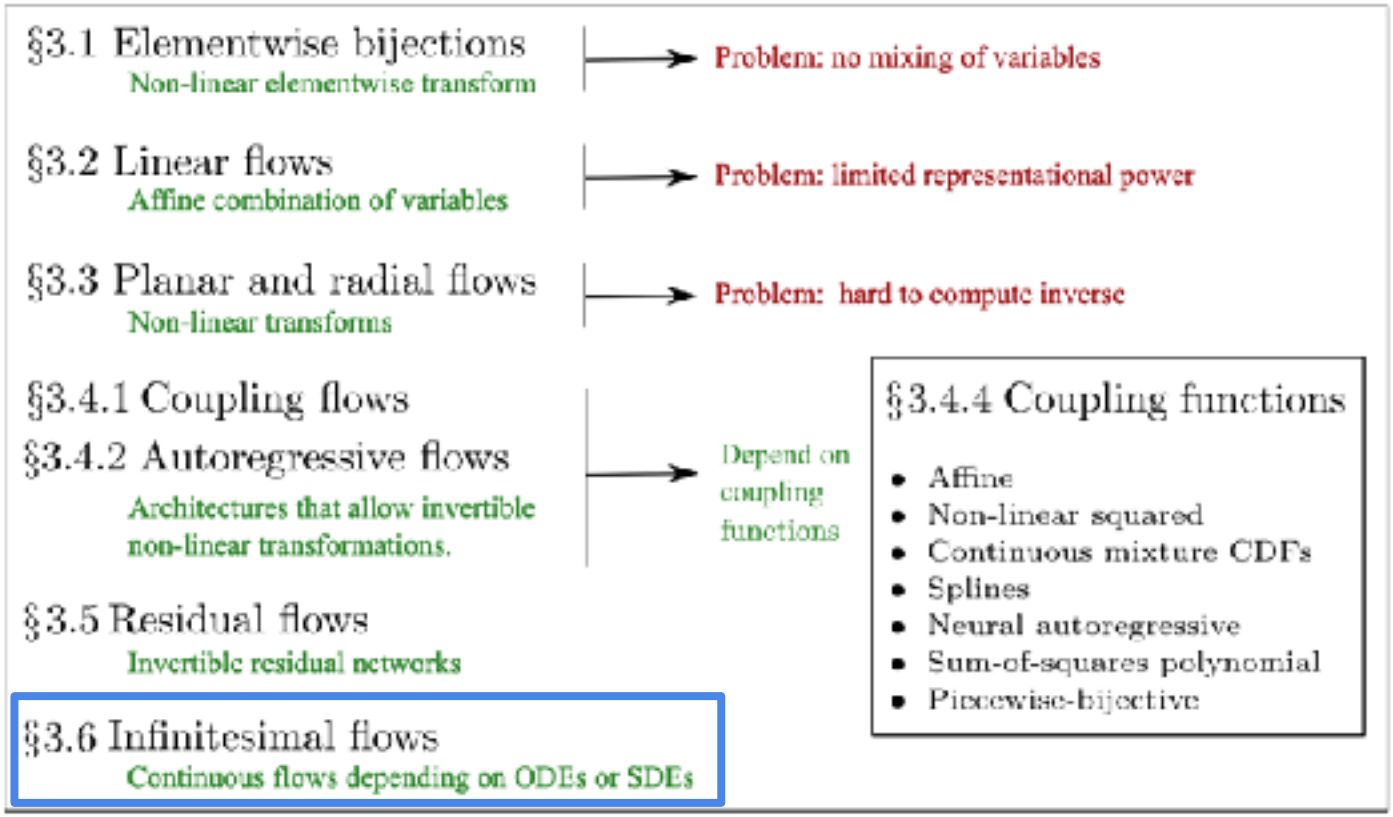
Density mapping

Objective of generative modeling : draw a random sample z to generate a data



[Kobyzev et al. Normalizing flows: an introduction and review of currents methods. IEEE PAMI 2020]

Normalizing flows



"Note, that even though Langevin flows manifest nice mathematical properties, they have not found practical applications."

[Kobyzev et al. Normalizing flows: an introduction and review of currents methods. IEEE PAMI 2020]

Discrete-Time Markov Processes

Sequence of random variables, with Markov properties:

$$P(X_0 = x_0, X_1 = x_1, X_2 = x_2, \dots) = P(X_0 = x_0) \prod_{t \in \{1,2,\dots\}} P(X_t = x_t | X_{t-1} = x_{t-1})$$

Such a Markov process is completely defined by the initial state and the transition probabilities.

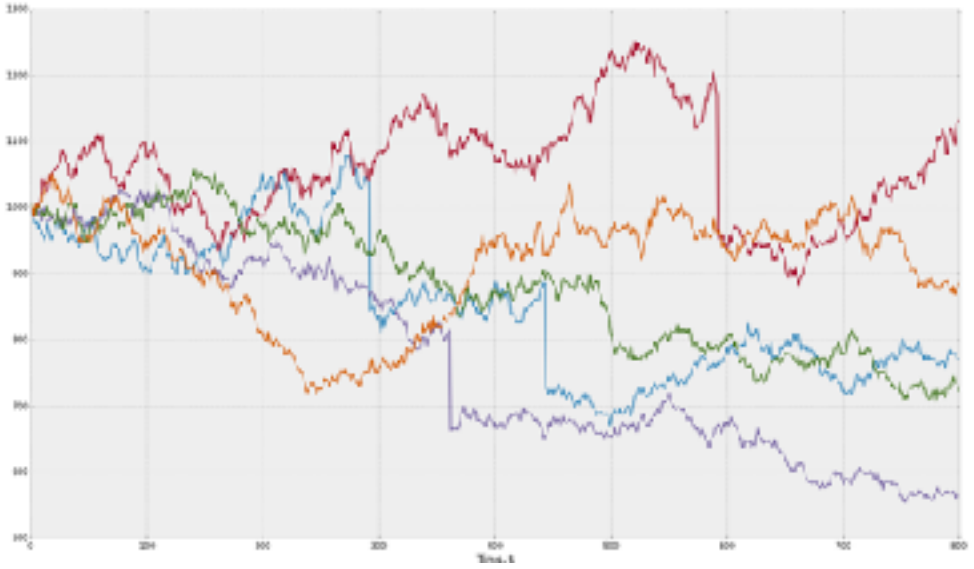
Given two states, one can write:

$$P(X_t = x_t | X_s = x_s) = \sum_k P(X_t = x_t | X_m = k) P(X_m = k | X_s = x_s)$$

Continuous-time Markov processes

In 1931, Kolmogorov found that there are two kinds of continuous time Markov processes, depending on the assumed behavior over small intervals of time:

- Jump processes (like Poisson process)
- Diffusion processes (represented by diffusion and by Brownian motion)



The Kolmogorov forward equation

Evolution of the conditional density function over a small interval:

$$p(x; t + \mathrm{d}t|y; s) = \int_{-\infty}^{\infty} p(x; t + \mathrm{d}t|m; t)p(m; t|y; s)\mathrm{d}m$$

Let define: $\phi_t(\Delta; z) = p(z + \Delta; t + \mathrm{d}t|z; t)$

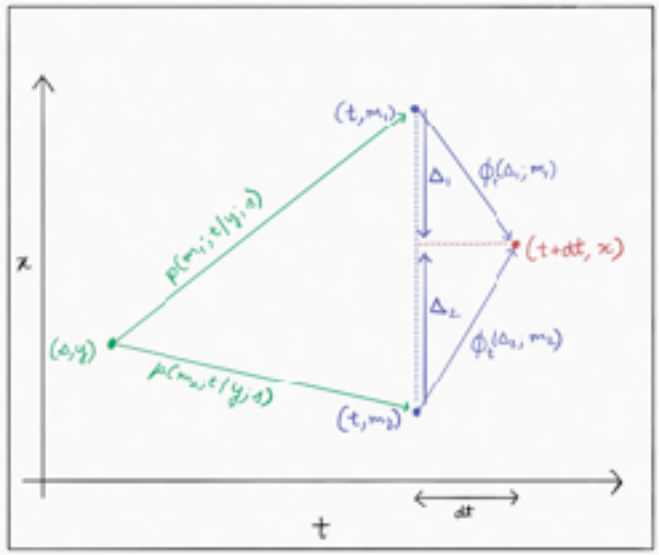
Rewrite using: $m = x - \Delta$

$$\Rightarrow \mathrm{d}m = -\mathrm{d}\Delta$$

$$m = \pm\infty \Rightarrow \Delta = \mp\infty$$

$$\Rightarrow p(x; t + \mathrm{d}t|y; s) = -\int_{\infty}^{-\infty} \phi_t(\Delta; m)p(m; t|y; s)\mathrm{d}\Delta$$

$$= \int_{-\infty}^{+\infty} \phi_t(\Delta; m)p(m; t|y; s)\mathrm{d}\Delta$$



[T. S. Dabral, 2021]

The Kolmogorov forward equation

Taylor's theorem: $f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f^{(2)}(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(n)}(a)}{n!}(x-a)^n + R_n(x)$

Using Taylor's theorem around the point x:

$$\begin{aligned} p(x; t+\mathrm{d}t|y; s) = & \int_{-\infty}^{+\infty} \phi_t(\Delta; x) p(x; t|y; s) \mathrm{d}\Delta \\ & - \int_{-\infty}^{+\infty} \Delta \frac{\partial}{\partial x} \phi_t(\Delta; x) p(x; t|y; s) \mathrm{d}\Delta \\ & + \int_{-\infty}^{+\infty} \frac{\Delta^2}{2} \frac{\partial^2}{\partial x^2} \phi_t(\Delta; x) p(x; t|y; s) \mathrm{d}\Delta \\ & \vdots \end{aligned}$$

We get:

$$\begin{aligned} p(x; t+\mathrm{d}t|y; s) - p(x; t|y; s) = & -\frac{\partial}{\partial x} \left(\mathbb{E}_{\Delta \sim \phi_t(;x)}[\Delta] p(x; t|y; s) \right) \\ & + \frac{1}{2} \frac{\partial^2}{\partial x^2} \left(\mathbb{E}_{\Delta \sim \phi_t(;x)} \left[\Delta^2\right] p(x; t|y; s) \right) \\ & \vdots \end{aligned}$$

The Kolmogorov forward equation

Let define:

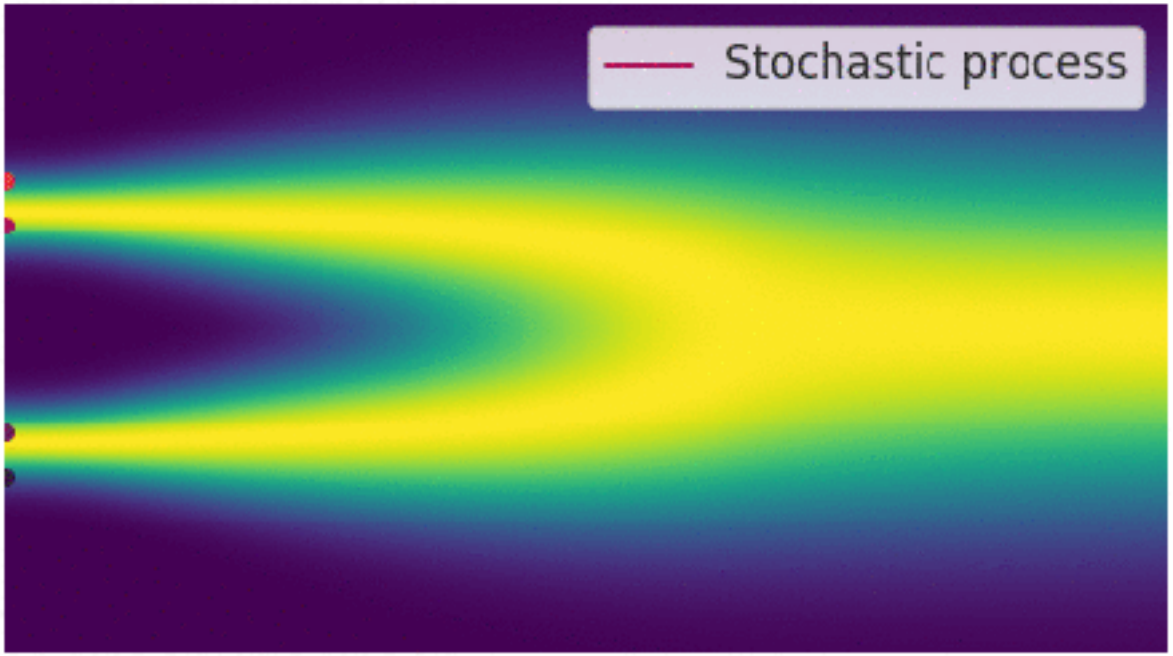
$$\mathbb{E}_{\Delta \sim \phi_t(;x)}[\Delta] := f(x,t)\mathrm{d}t$$
$$\mathbb{E}_{\Delta \sim \phi_t(;x)}\left[\Delta^2\right] := g^2(x,t)\mathrm{d}t$$

By truncating the Taylor series up to the second moment, we get the following partial differential equation:

$$\frac{\partial}{\partial t}p(x;t|y;s) = -\frac{\partial}{\partial x}\left(f(x,t)p(x;t|y;s)\right) + \frac{1}{2}\frac{\partial^2}{\partial x^2}\left(g^2(x;t)p(x;t|y;s)\right)$$

This is the Kolmogorov forward equation, also known as the Fokker-Planck equation.
F is called the *drift coefficient* of the diffusion process and g is called the *diffusion coefficient*.

Diffusion models and SDE



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

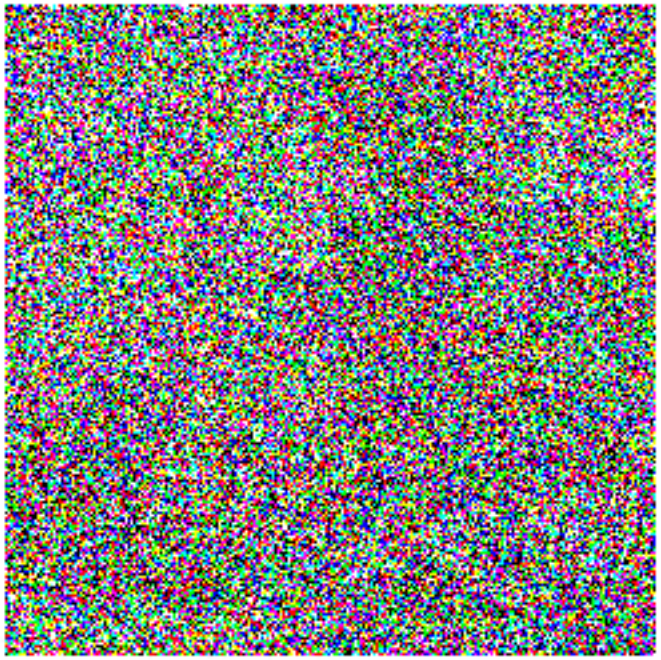
drift coefficient

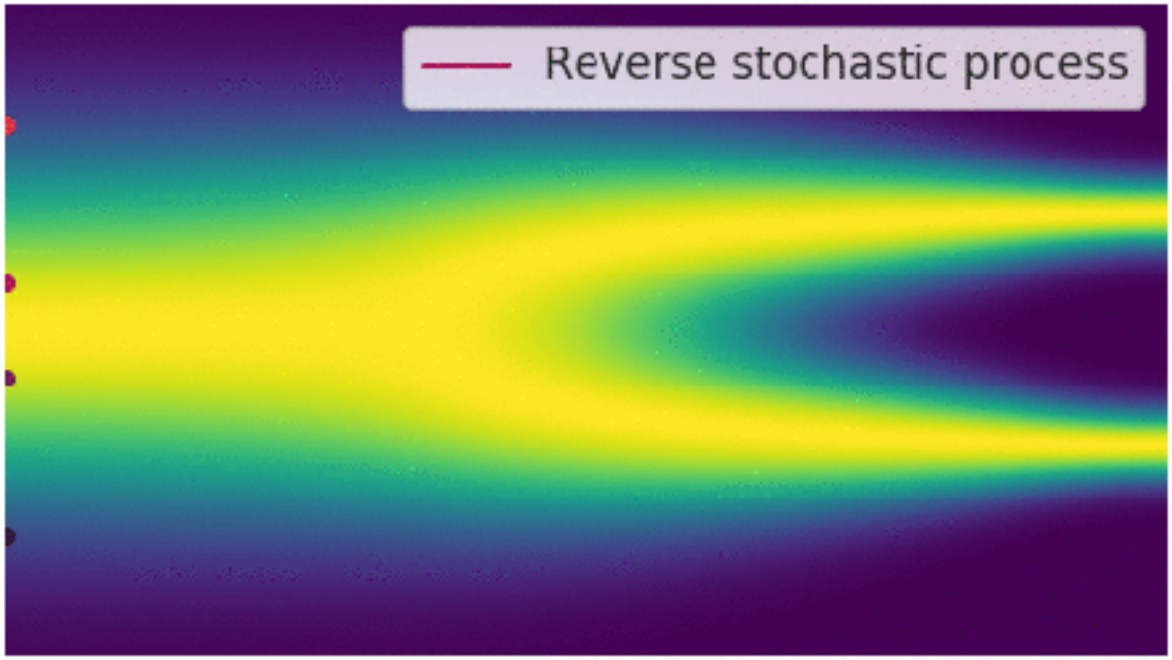
diffusion coefficient

Brownian motion

[Song et al., Score-based generative modeling through stochastic differential equations, ICLR 2021]

Sample generation by reversing the SDE



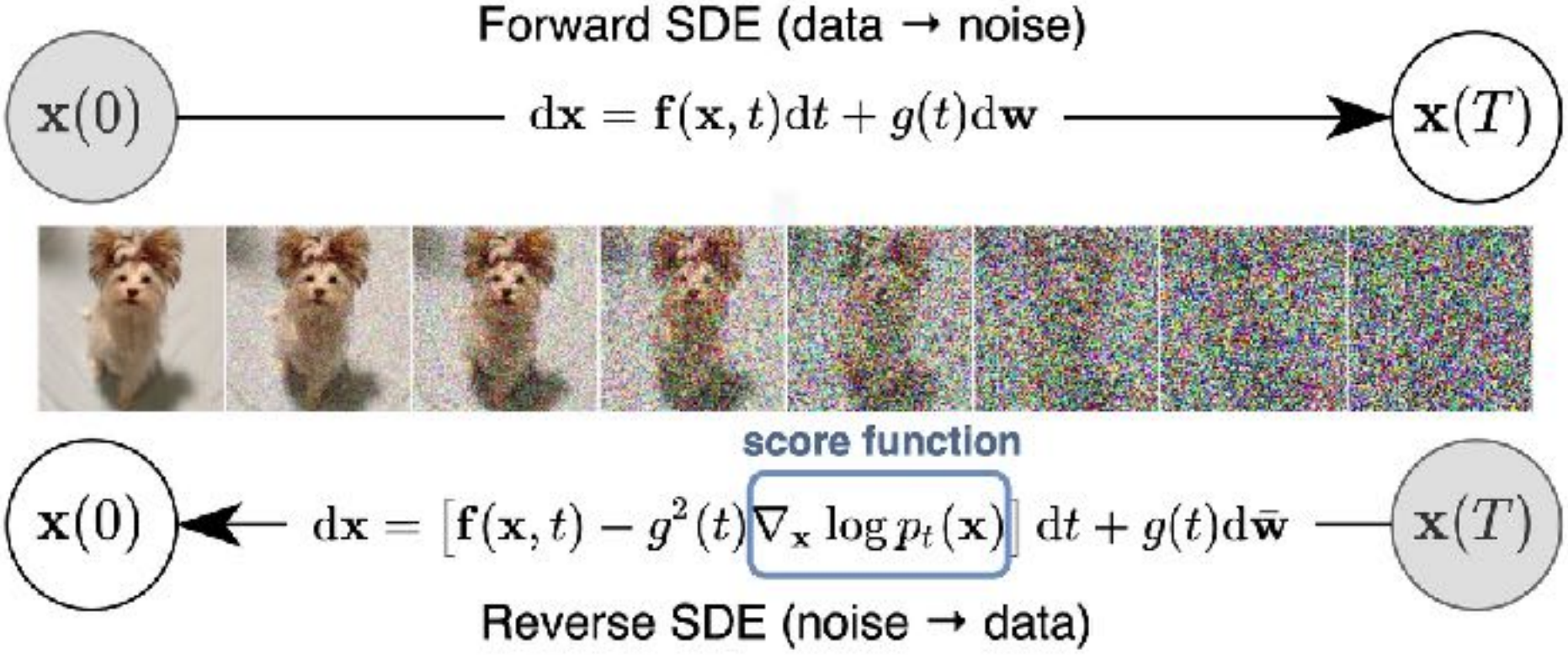


$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})]\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$$

Score function
(to be estimated)

[Anderson, Reverse-time diffusion equations models, 1982; Song et al., ICLR 2021]

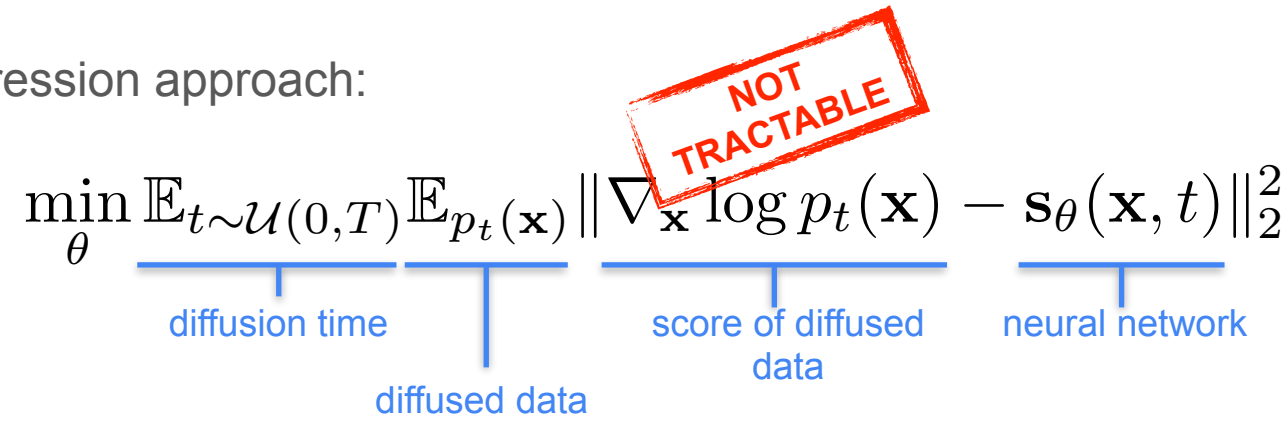
Overview of the SDE-based generative modeling



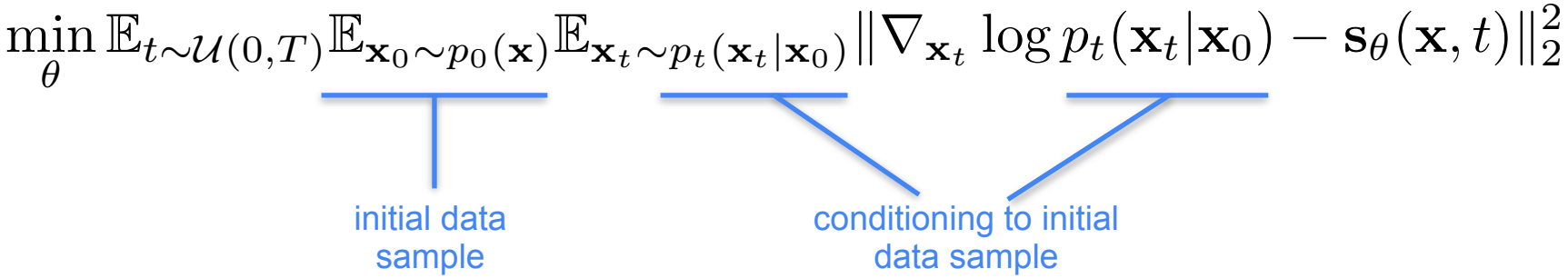
[Song et al., Score-based generative modeling through stochastic differential equations, ICLR 2021]

How to learn the score function?

Direct regression approach:



Conditioning to initial state:



It can be shown that, after expectations, $\mathbf{s}_\theta(\mathbf{x}, t) \approx \nabla_{\mathbf{x}} \log p_t(\mathbf{x})$

Variance preserving SDE

Consider the following discrete Markov chain:

$$\mathbf{x}_i = \sqrt{1-\beta_i}\mathbf{x}_{i-1} + \sqrt{\beta_i}\mathbf{z}_{i-1}, \mathbf{z}_{i-1} \sim \mathcal{N}(0, I), \beta_i \in [0, 1]$$

It corresponds to the following SDE:

$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\ \mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{w}$$

It is called the variance preserving SDE.

Variance preserving SDE

$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\,\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{w}$$

$$\gamma_t = e^{-\frac{1}{2}\int_0^t \beta(s)ds}$$
$$\sigma_t^2 = 1 - e^{-\int_0^t \beta(s)ds}$$

Re-parametrized sampling: $\mathbf{x}_t = \gamma_t \mathbf{x}_0 + \sigma_t \boldsymbol{\epsilon} \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

Score function: $$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{x}_0) = -\nabla_{\mathbf{x}_t} \frac{(\mathbf{x}_t - \gamma_t\mathbf{x}_0)^2}{2\sigma_t^2} = -\frac{\mathbf{x}_t - \gamma_t\mathbf{x}_0}{\sigma_t^2} = -\frac{\epsilon}{\sigma_t}$$

Neural network model: $$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) := -\frac{\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\sigma_t}$$

Overall loss: $$\min_\theta \mathbb{E}_{t\sim\mathcal{U}(0,T)}\mathbb{E}_{\mathbf{x}_0\sim p_0(\mathbf{x})}\mathbb{E}_{\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})} \frac{1}{\sigma_t^2}\|\epsilon - \epsilon_\theta(\mathbf{x}_t, t)\|_2^2$$

Denoising Diffusion Models

Implementation considerations

1. Network architecture for score function estimation

2. Noise scheduler

3. Weighting of the loss for different timesteps:

$$\min_{\theta} \mathbb{E}_{t\sim\mathcal{U}(0,T)} \mathbb{E}_{\mathbf{x}_0\sim p_0(\mathbf{x})} \mathbb{E}_{\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})} \frac{\lambda(t)}{\sigma_t^2} \|\epsilon - \epsilon_\theta(\mathbf{x}_t, t)\|_2^2$$

Different loss weightings trade off between model with good perceptual quality vs high log-likelihood:

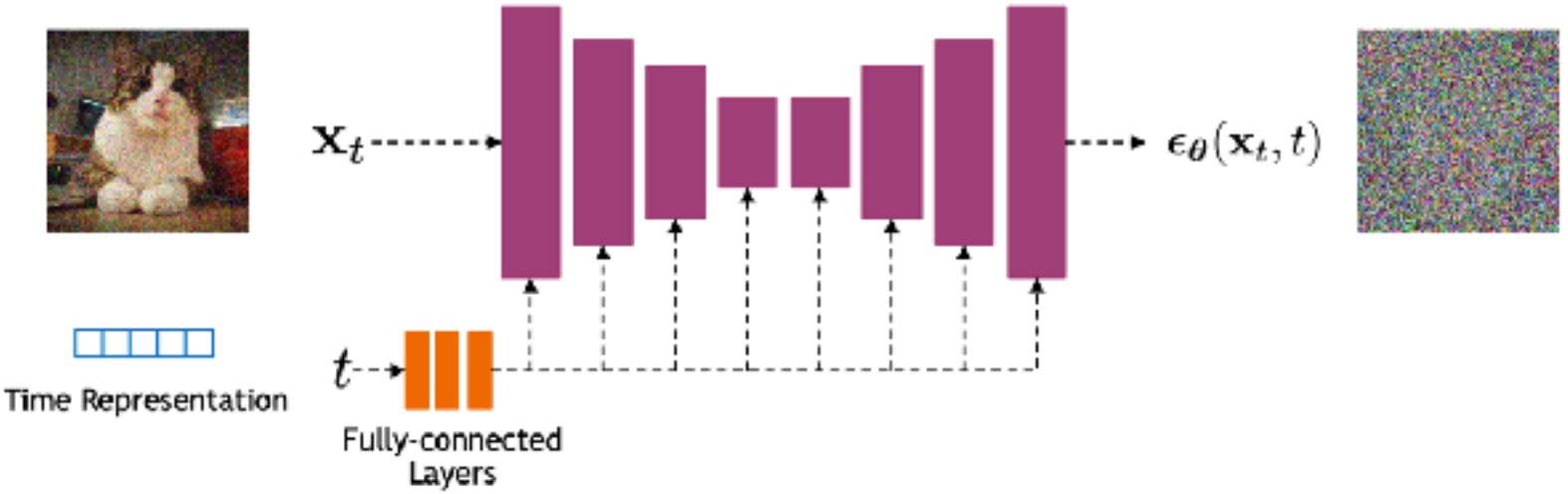
Perceptual quality: $\lambda(t) = \sigma_t^2$

Maximum log-likelihood: $\lambda(t) = \beta(t)$ *(negative ELBO)*

[Kreis, Gao and Vahdat, CVPR tutorial 2022]

Network architecture

Diffusion models often use U-Net architectures with ResNet blocks and self-attention layers to represent $\epsilon_\theta(\mathbf{x}_t, t)$



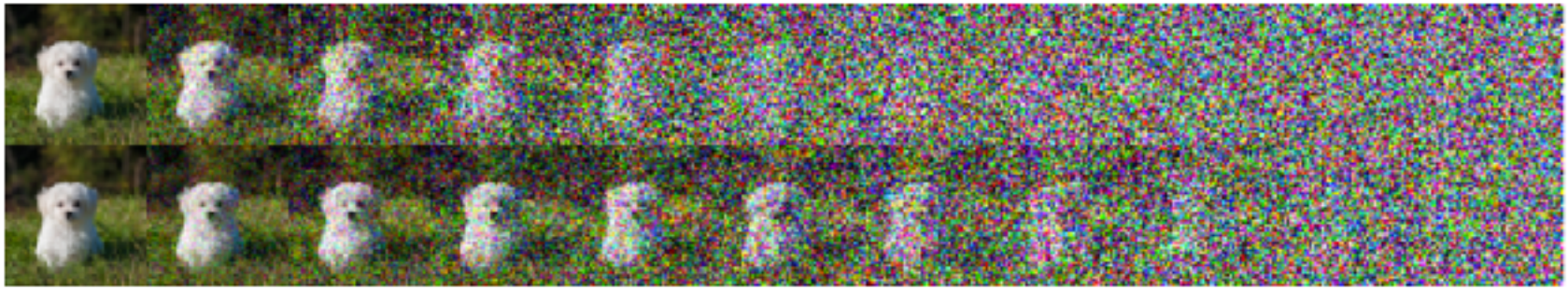
Time representation: sinusoidal positional embeddings or random Fourier features.

[Kreis, Gao and Vahdat, CVPR tutorial 2022]

Noise scheduler

Linear

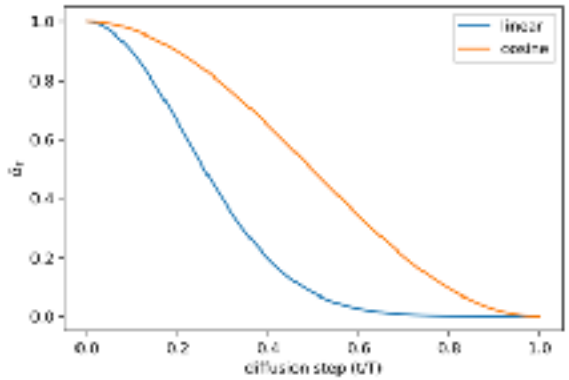
Cosine



Linear schedule from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$

Cosine schedule: $\beta_t = \text{clip}(1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}, 0.999) \quad \bar{\alpha}_t = \frac{f(t)}{f(0)} \quad \text{where } f(t) = \cos\left(\frac{t/T + s}{1+s} \cdot \frac{\pi}{2}\right)^2$

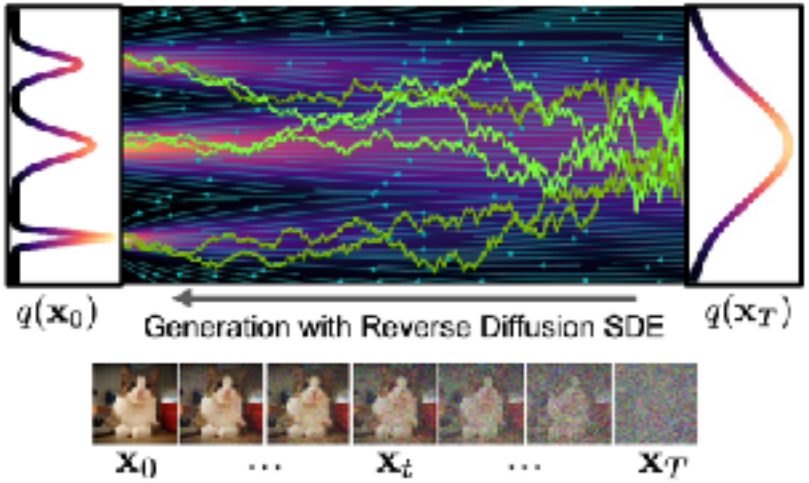
$\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=0}^{t} \alpha_s$



[Ho et al. 2020, Nichol and Dhariwal, 2021]

Sampling data, i.e. how to solve the reverse SDE?

Numerical solvers provide approximate trajectories from SDE.



Generative Diffusion SDE:

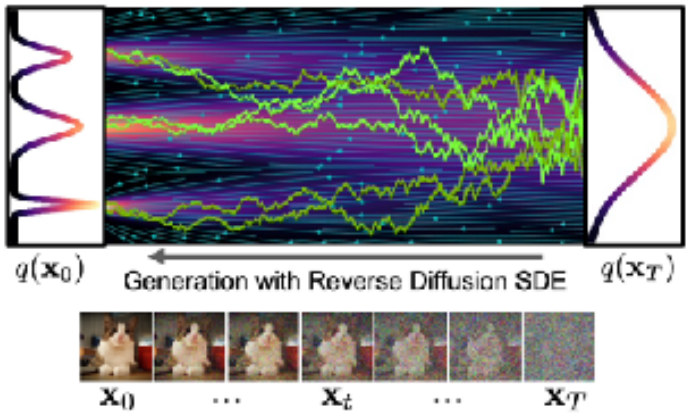
$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + 2\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{\omega}}_t$$

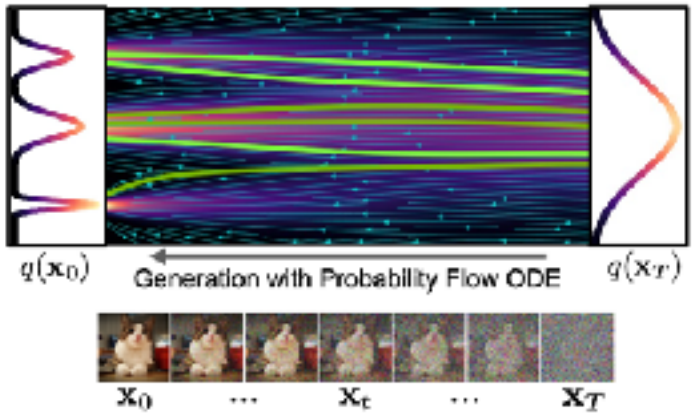
➡ *Euler-Maruyama:*

$$\mathbf{x}_{t-1} = \mathbf{x}_t + \frac{1}{2}\beta(t)\left[\mathbf{x}_t + 2\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\Delta t + \sqrt{\beta(t)\Delta t}\,\mathcal{N}(\mathbf{0}, \mathbf{I})$$

[Kreis, Gao and Vahdat, CVPR tutorial 2022]

Sampling: SDE vs ODE





Generative Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + 2\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{\omega}}_t$$

➡ *Euler-Maruyama*:

$$\mathbf{x}_{t-1} = \mathbf{x}_t + \frac{1}{2}\beta(t)\left[\mathbf{x}_t + 2\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\Delta t + \sqrt{\beta(t)\Delta t}\,\mathcal{N}(\mathbf{0}, \mathbf{I})$$

Probability Flow ODE:

$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\mathrm{d}t$$

➡ *Euler's Method*:

$$\mathbf{x}_{t-1} = \mathbf{x}_t + \frac{1}{2}\beta(t)\left[\mathbf{x}_t + \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\Delta t$$

[Kreis, Gao and Vahdat, CVPR tutorial 2022, Song et al., ICLR 2021]

What about denoising diffusion probabilistic models (DDPM)?

DDPM proposed by Ho et al. in 2020 is equivalent to variance preserving SDE with ancestral sampling.

Algorithm 1 Training

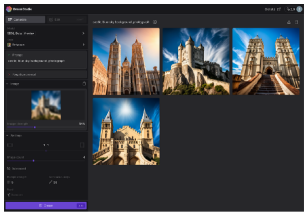
1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0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

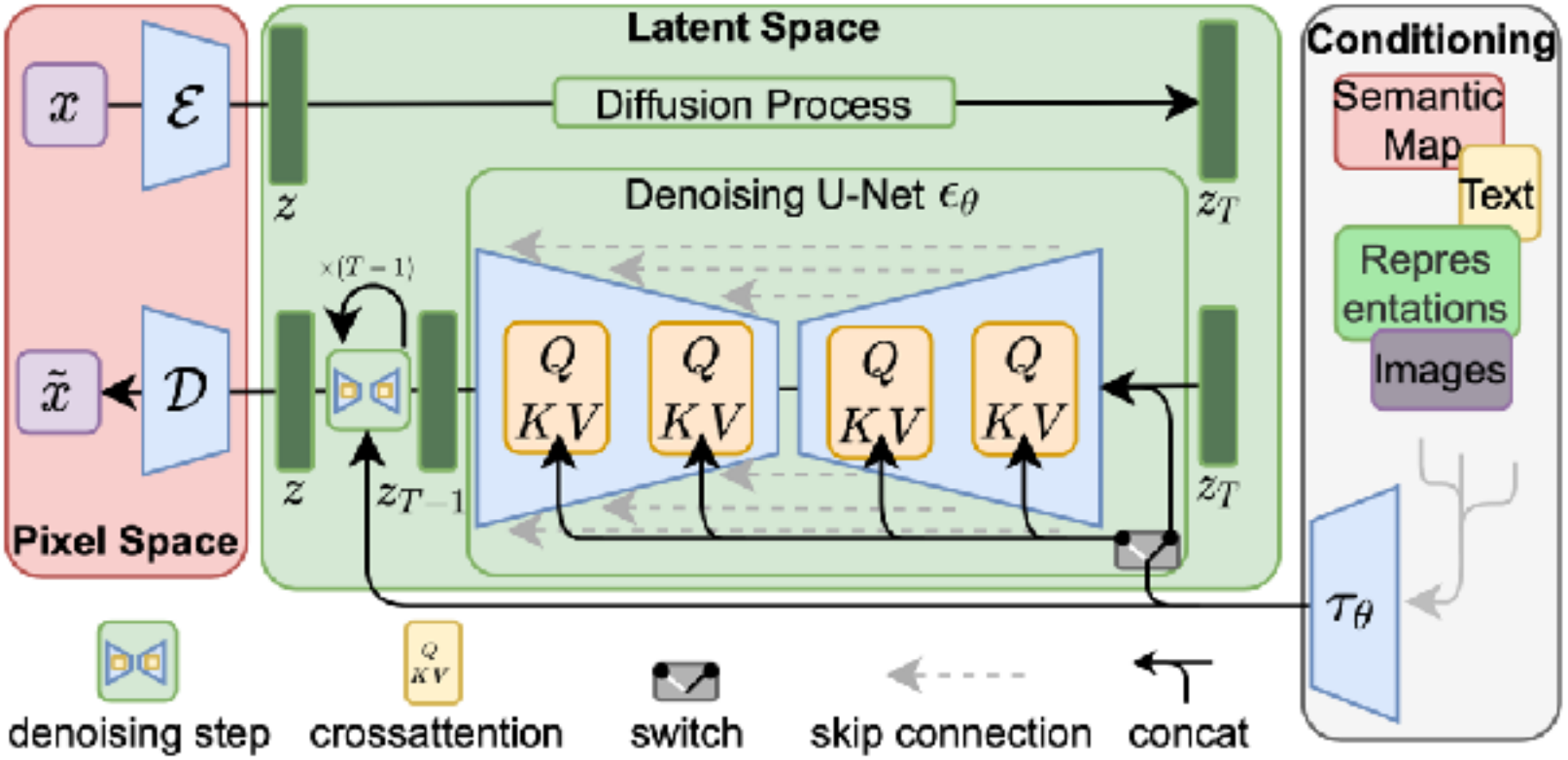
Algorithm 2 Sampling

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** $\mathbf{x}_0$

[Sohl-Dickstein et al. 2015, Ho et al. 2020, Song et al., ICLR 2021]

What about stable diffusion?





[Rombach et al. High-resolution image synthesis with latent diffusion models, CVPR 2022]

A not exhaustive list of key works

- Anderson, Reverse-time diffusion equation models, 1982
- Vincent, A connection between score matching and denoting auto encoders, Neural computation, 2011
- Sohl-Dickstein et al., Deep unsupervised learning using non equilibrium thermodynamics, ICML, 2015
- Ho et al. Denoising diffusion probabilistic models, 2020
- Song et al. Score-based generative modeling through stochastic differential equations, ICLR 2021
- Rombach et al. High-resolution image synthesis with latent diffusion models, CVPR 2022
- Yang et al., Diffusion models: a comprehensive survey of methods and applications, 2022

Take home messages

Diffusion models are very powerful and very easy to use

SDE formulation prodiges a clear mathematical framework

Connections to VAEs, neural ODEs, continuous normalizing flows

Key elements for implementations: denoising network (Unet) and a scheduler

Long to train and « slow » to sample

What about other data types? (meshes, text, etc.)