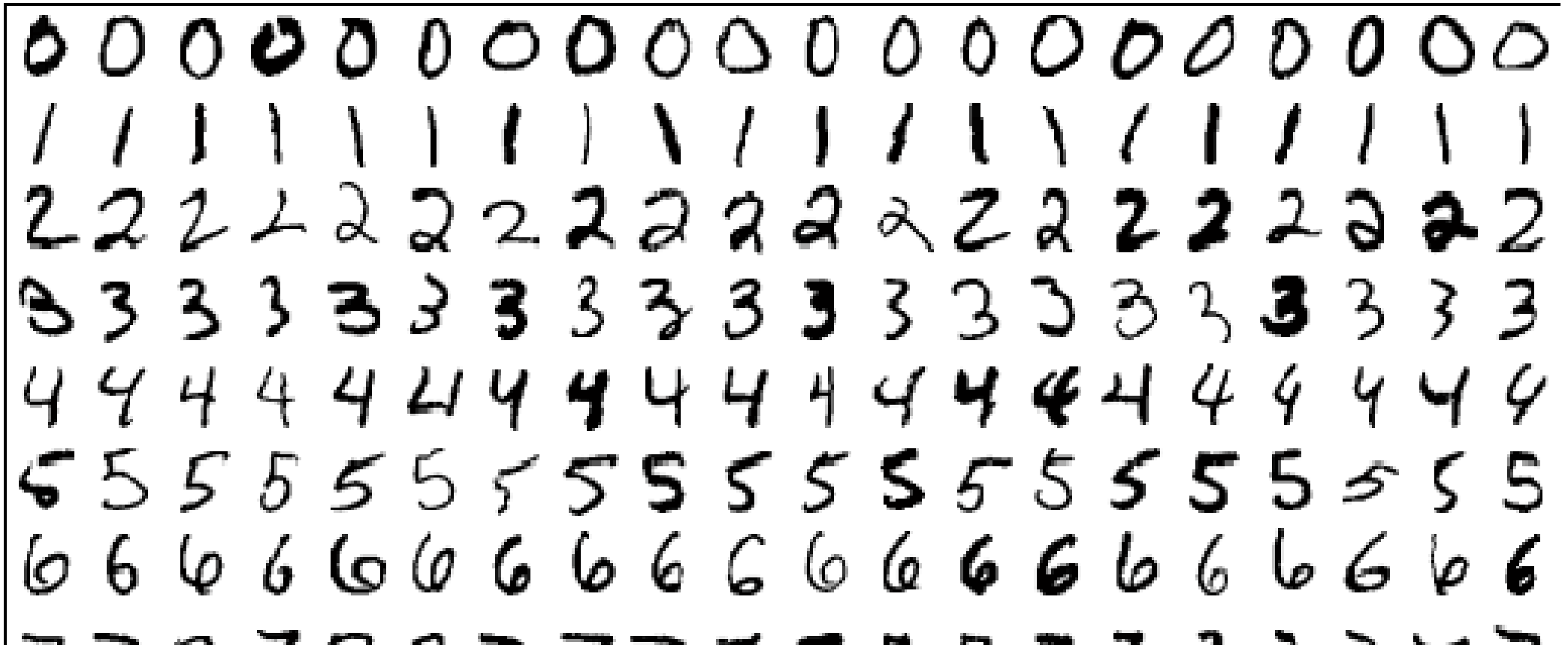
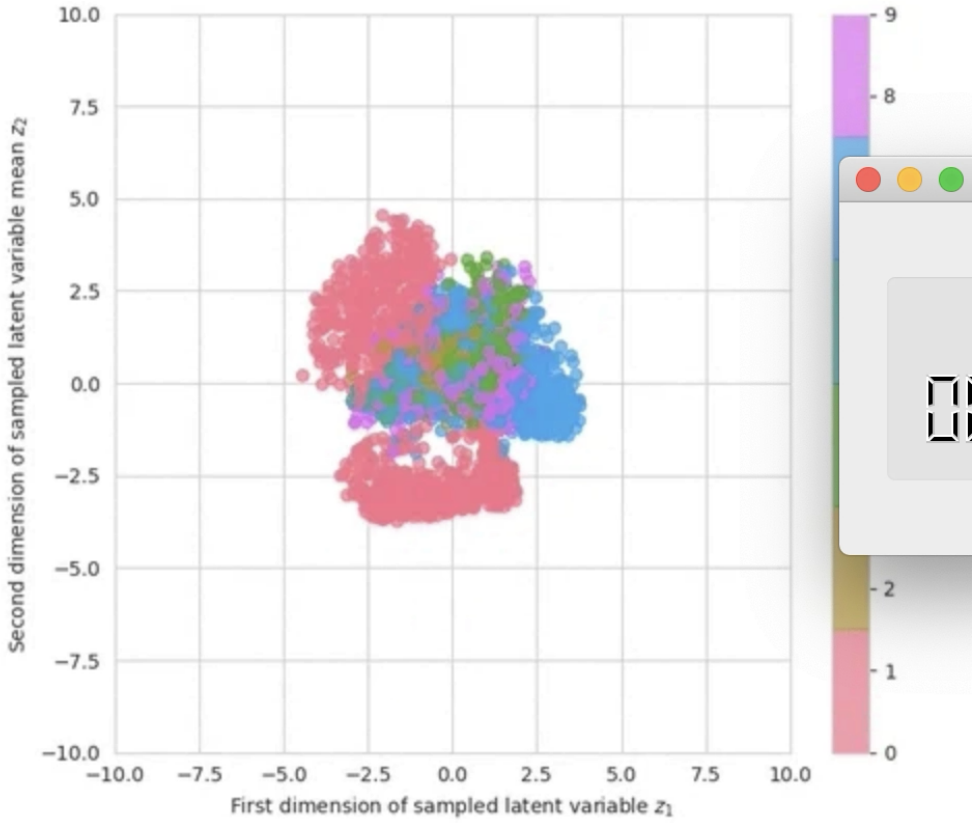
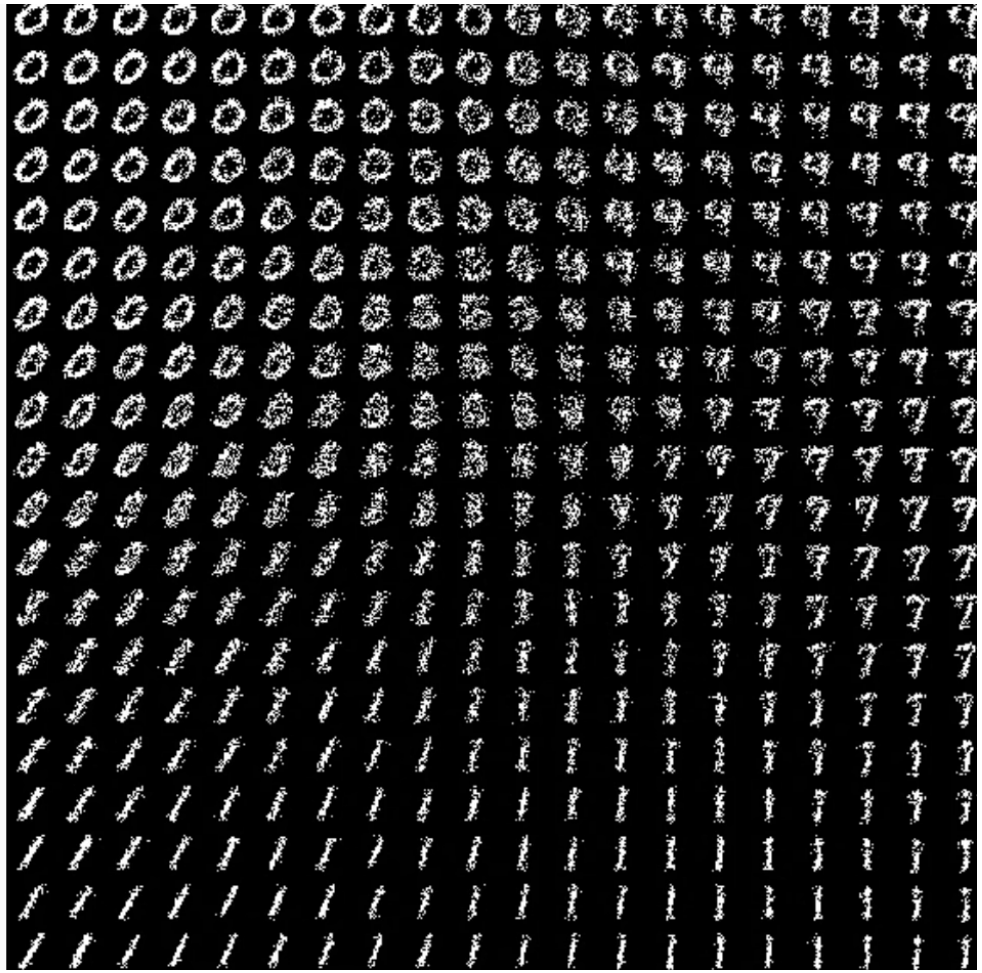
Results



Results

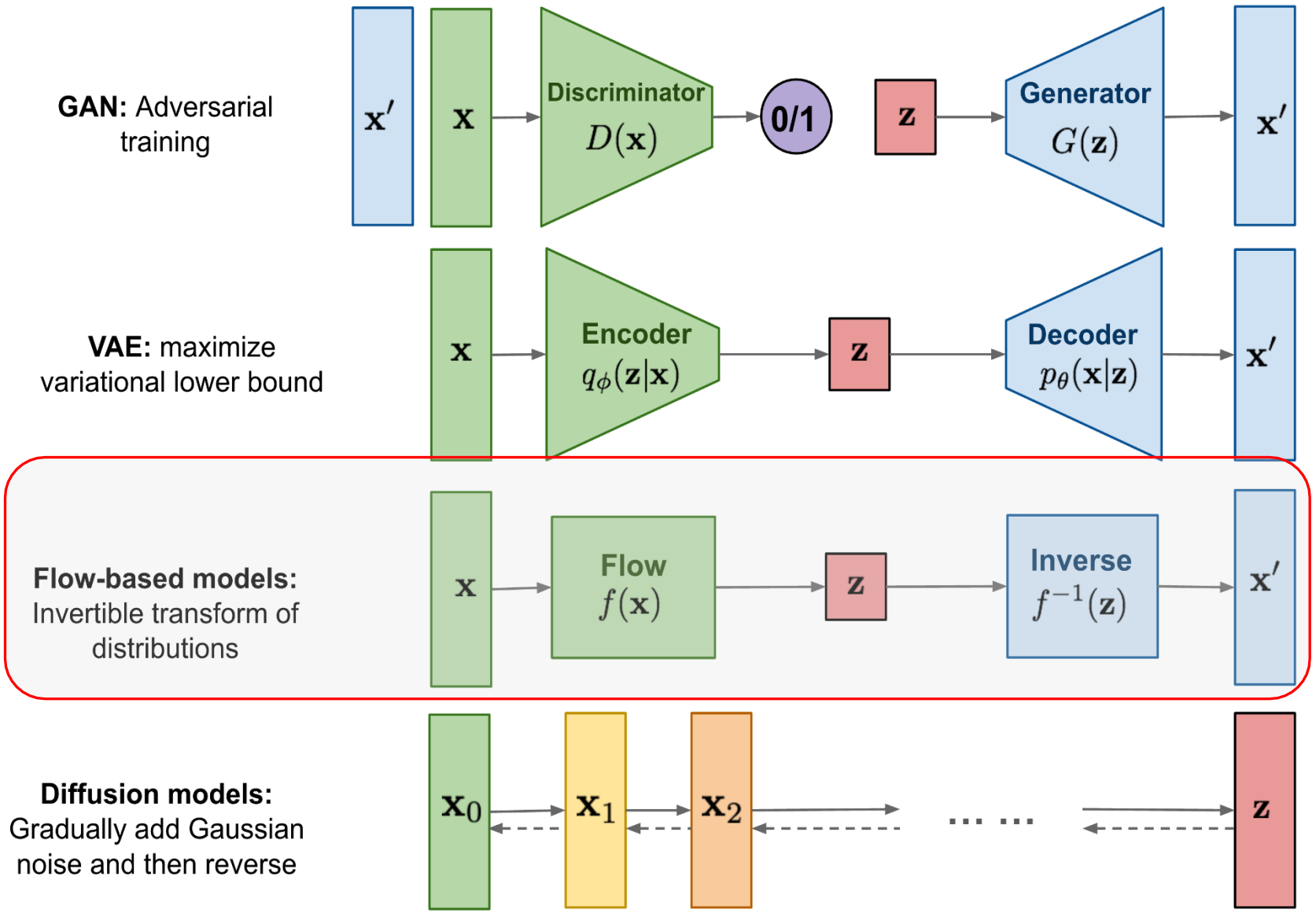

10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
Second dimension of sampled latent variable mean z_2
−10.0 −7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0
First dimension of sampled latent variable z_1
9
8
2
1
0



VAE的优缺点

- 优点：
  - 生成高质量的样本。
  - 自动编码器
  - 潜在空间
  - 变分推断
  - 应用广泛
- 缺点：
  - 训练困难：VAE的训练过程比较困难，需要设置合适的超参数和优化算法，避免出现梯度消失和模式崩溃的问题。
  - 模型复杂：VAE的模型比较复杂，需要大量的计算资源和时间来训练。
  - 生成样本有限：VAE生成的样本数量是有限的，因为它只能从潜在空间中采样有限的样本。
  - 模糊的潜在空间：VAE学习的潜在空间可能是模糊的，因为它只是对数据的一个近似表示，可能存在信息损失。

Generative Model Zoo

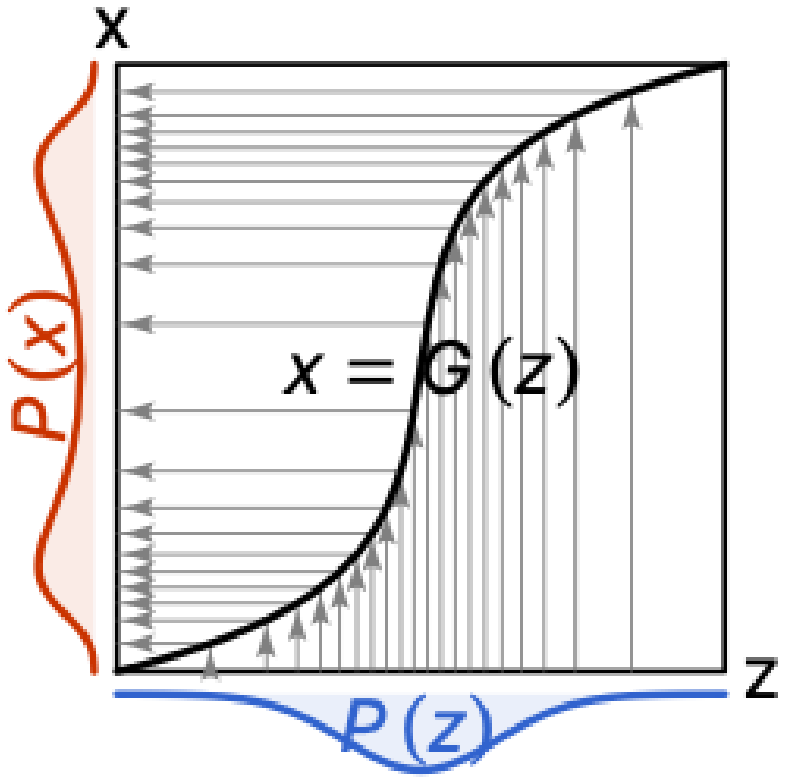


https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

Normalizing Flow

- For finite discrete variable
  - Softmax function
- For continuous with gaussian distribution
  - VAE
- For continuous with any distribution
  - Normalizing Flow
  - Diffusion model

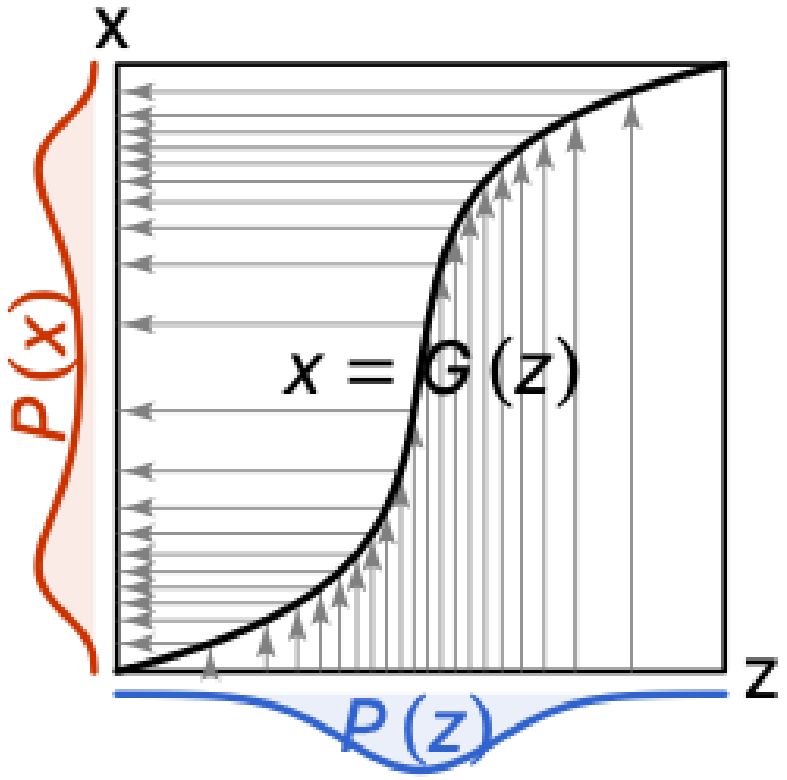
Density function of inverse function



$$Z \sim P(z)$$
$$X = G(Z)$$
$$X \sim ?$$

G is a bijector

Density function of inverse function

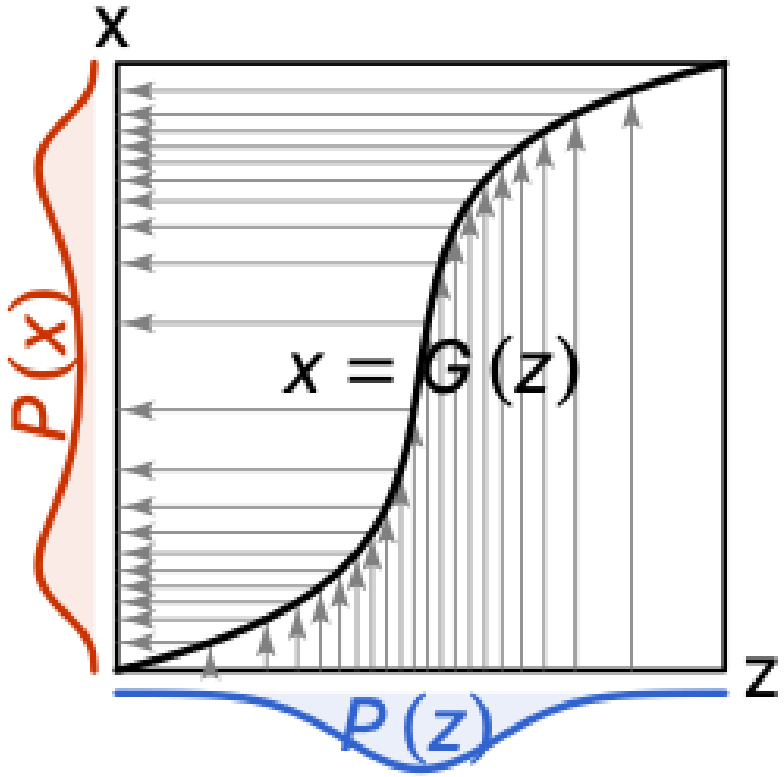


$$Z \sim P(z)$$
$$X = G(Z)$$
$$X \sim ?$$

$$F(x) \equiv \Pr(X < x)$$
$$= \Pr(G(Z) < x)$$
$$= \Pr\left(Z < G^{-1}(x)\right) \equiv F(G^{-1}(x))$$

$$P(x) \equiv \frac{\partial F(x)}{\partial x} = \left|\frac{\partial G^{-1}}{\partial x}\right|_x \cdot \left.\frac{\partial F(z)}{\partial z}\right|_{z=G^{-1}(x)}$$
$$= \left|\left(\frac{\partial G}{\partial x}\right)^{-1}\right| \cdot P(z)$$

For general case



$$Z \sim P(z)$$
$$X = G(Z)$$
$$X \sim ?$$

$$x, z \in \Re^n$$

$$P(x) = \left| \left(\frac{\partial G}{\partial x} \right)^{-1} \right| \cdot P(z)$$

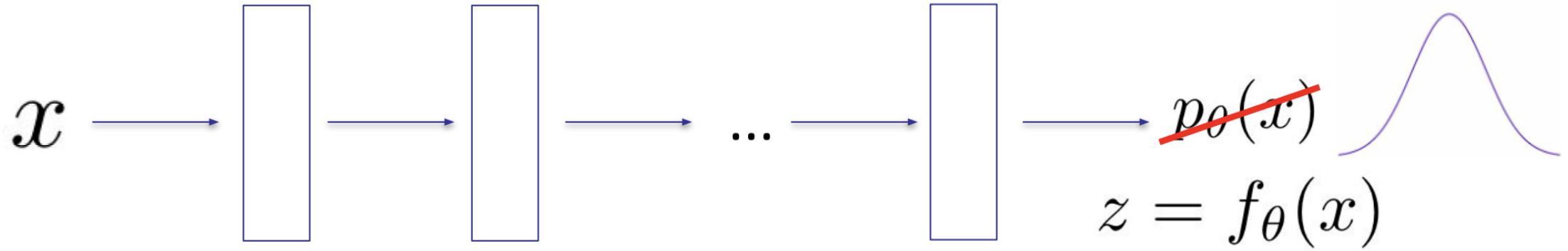
Jacobian:

$$\frac{\partial G}{\partial x} = \begin{pmatrix} \frac{\partial G_1}{\partial x_1} & \frac{\partial G_2}{\partial x_1} \cdots & \frac{\partial G_n}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial G_n}{\partial x_1} & \frac{\partial G_n}{\partial x_2} \cdots & \frac{\partial G_n}{\partial x_n} \end{pmatrix}$$

$|x|$: *abusolute value of determinant*

Flows: main idea

Invertible function Invertible function Invertible function



Generally: $z \sim p_Z(z)$

Normalizing Flow: $z \sim \mathcal{N}(0, 1)$

How to train? How to evaluate $p_\theta(x)$? How to sample?

Flows: main idea

$$f_\theta = f_n \circ f_{n-1} \circ \cdots \circ f_1$$

Invertible function Invertible function Invertible function

x → f_1 → f_2 → ... → f_n → $z = f_\theta(x)$

$$z \sim p_Z(z)$$

$$\max_\theta \sum_i \log p_\theta(x^{(i)})$$

Training

$$\max_\theta \sum_i \log p_\theta(x^{(i)})$$

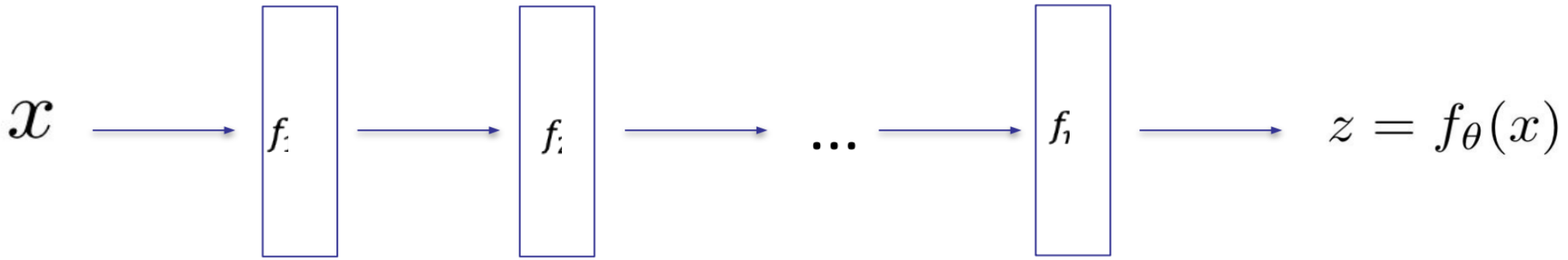
$$z^{(i)} = f_\theta(x^{(i)})$$

$$\begin{aligned} p_\theta(x^{(i)}) &= p_Z(z^{(i)}) \left|\frac{\partial z}{\partial x}(x^{(i)})\right| \\ &= p_Z(f_\theta(x^{(i)})) \left|\frac{\partial f_\theta}{\partial x}(x^{(i)})\right| \end{aligned}$$

$$\max_\theta \sum_i \log p_\theta(x^{(i)}) = \max_\theta \sum_i \log p_Z(f_\theta(x^{(i)})) + \log \left|\frac{\partial f_\theta}{\partial x}(x^{(i)})\right|$$

$\longrightarrow$ assuming we have an expression for p_Z ,
this can be optimized with Stochastic Gradient Descent

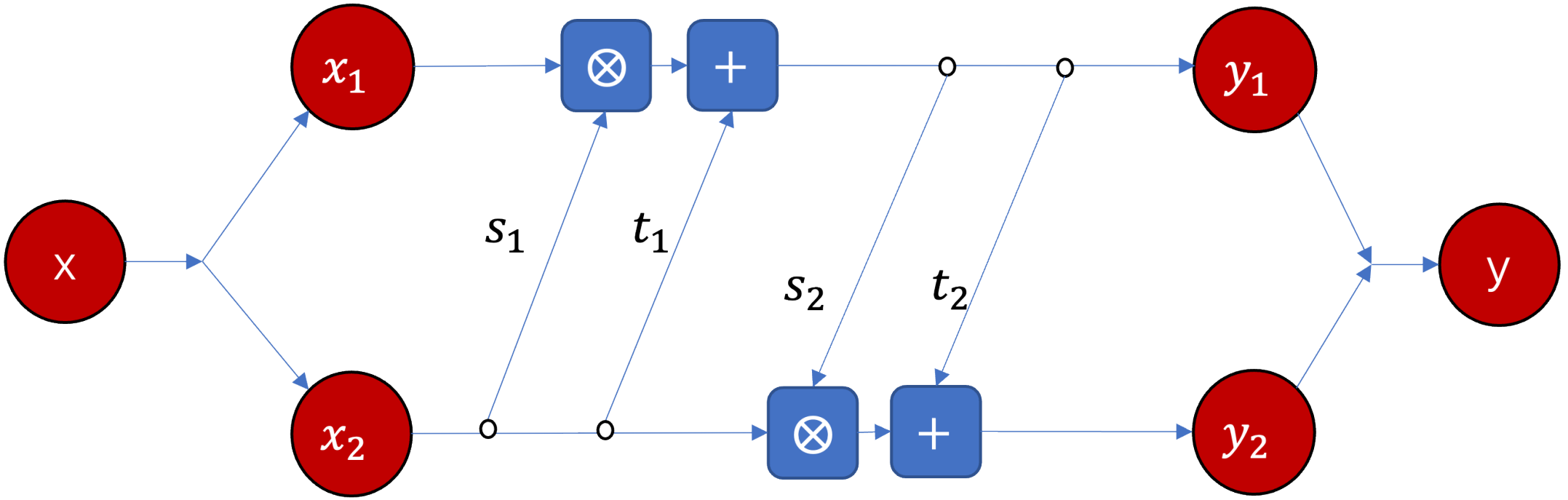
Sampling



Step 1: sample $z \sim p_Z(z)$

Step 2: $x = f_\theta^{-1}(z)$

Invertible Neural Network - RealNVP

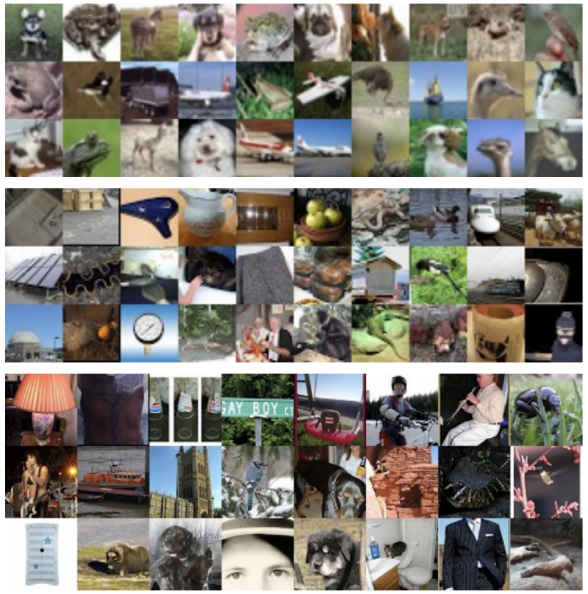


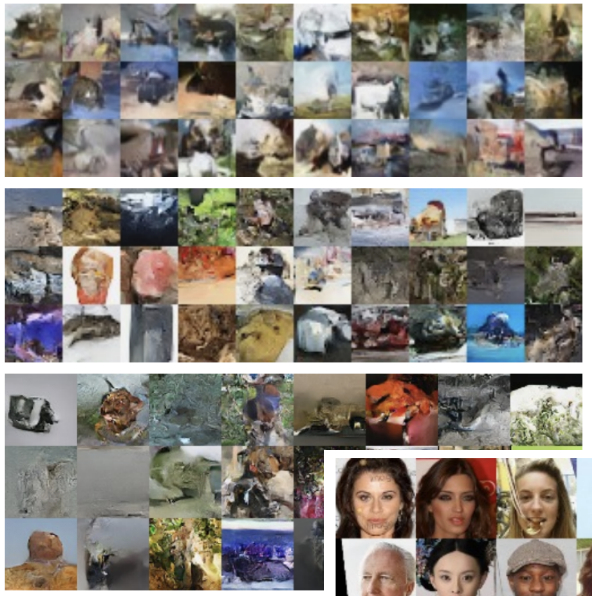
$$y_1 = x_1 \otimes \exp(s_1(x_2)) + t_1(x_2)$$
$$y_2 = x_2 \otimes \exp(s_2(y_1)) + t_2(y_1)$$

$$x_2 = (y_2 - t_2(y_1)) \otimes \exp(-s_2(y_1))$$
$$x_1 = (y_1 - t_1(x_2)) \otimes \exp(-s_1(x_2))$$

https://arxiv.org/pdf/1605.08803.pdf, DENSITY ESTIMATION USING REAL NVP

Results



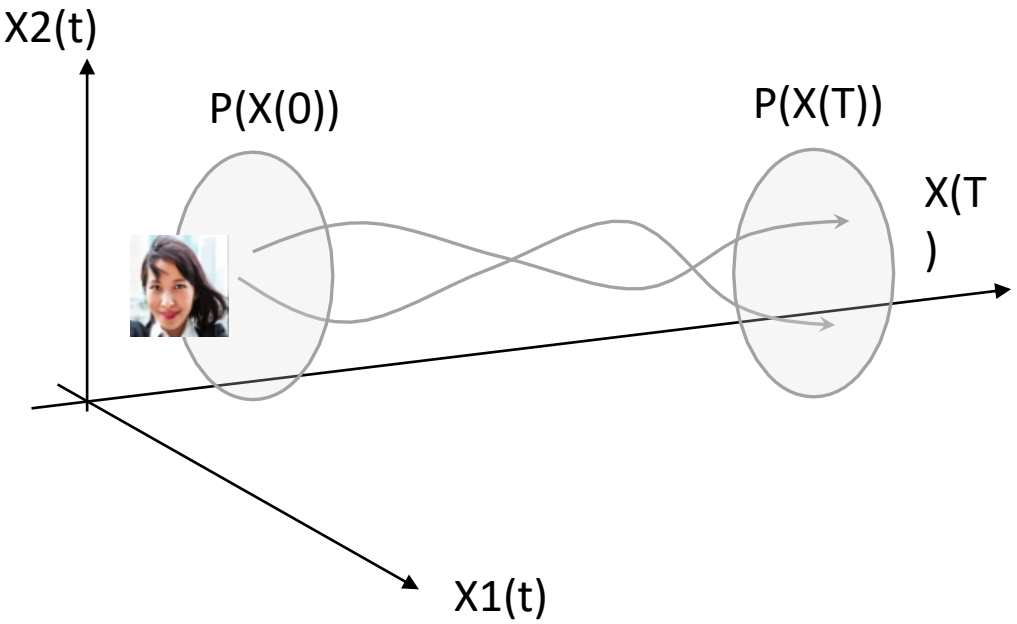


https://arxiv.org/pdf/1605.08803.pdf， DENSITY ESTIMATION USING REAL NVP

Differential Equation and Flows

$$\frac{dX}{dt} = f(X, t)$$

$$X(t+\varepsilon) = X(t) + \int_t^{t+\varepsilon} f(X(s), s)ds = F(X(t), \varepsilon)$$



https://arxiv.org/pdf/1605.08803.pdf, DENSITY ESTIMATION USING REAL NVP

Continuous Normalizing Flows

Theorem 1 (Instantaneous Change of Variables). *Let $\mathbf{z}(t)$ be a finite continuous random variable with probability $p(\mathbf{z}(t))$ dependent on time. Let $\frac{d\mathbf{z}}{dt} = f(\mathbf{z}(t), t)$ be a differential equation describing a continuous-in-time transformation of $\mathbf{z}(t)$. Assuming that f is uniformly Lipschitz continuous in $\mathbf{z}$ and continuous in t, then the change in log probability also follows a differential equation,*

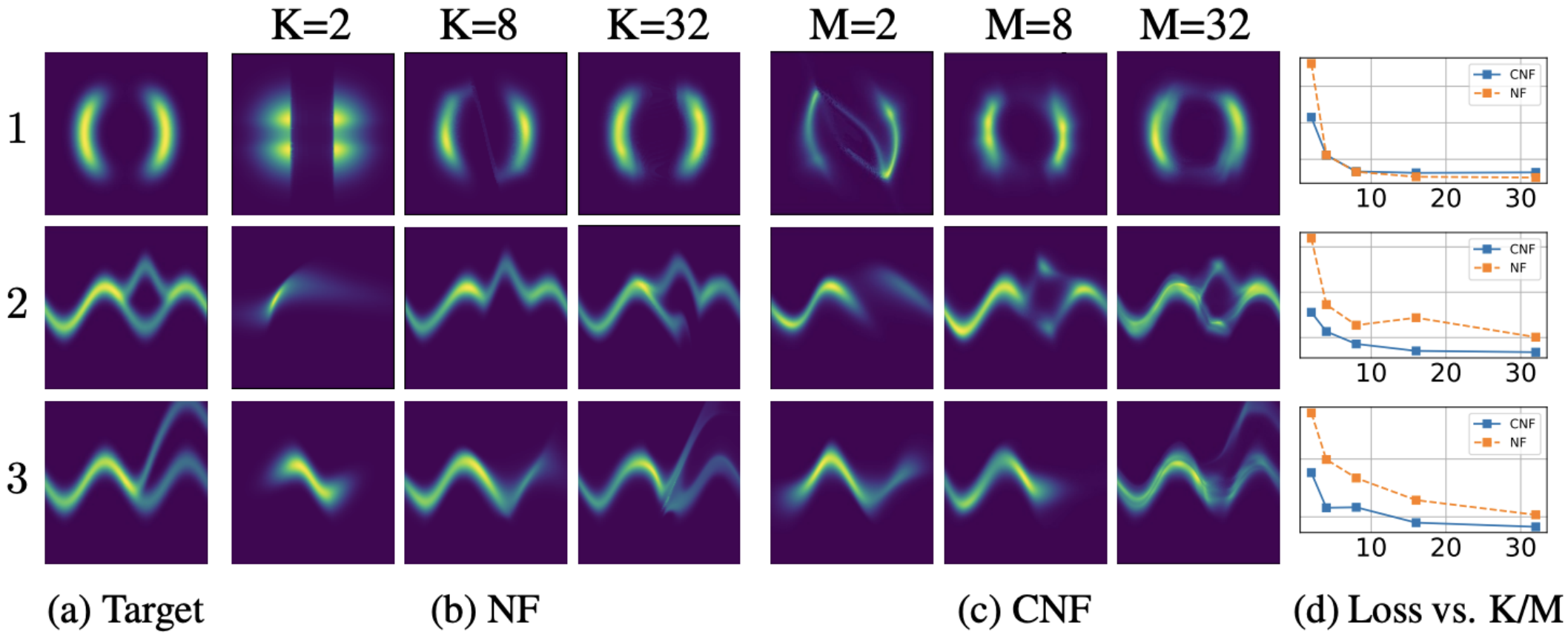
$$\frac{\partial \log p(\mathbf{z}(t))}{\partial t} = -\mathrm{tr}\left(\frac{df}{d\mathbf{z}(t)}\right) \tag{8}$$

$$\frac{d\mathbf{z}(t)}{dt} = uh(w^{\mathsf{T}}\mathbf{z}(t) + b), \qquad \frac{\partial \log p(\mathbf{z}(t))}{\partial t} = -u^{\mathsf{T}}\frac{\partial h}{\partial \mathbf{z}(t)}$$

$$\frac{d\mathbf{z}(t)}{dt} = \sum_{n=1}^{M} f_n(\mathbf{z}(t)), \qquad \frac{d \log p(\mathbf{z}(t))}{dt} = \sum_{n=1}^{M} \mathrm{tr}\left(\frac{\partial f_n}{\partial \mathbf{z}}\right)$$

https://arxiv.org/pdf/1806.07366.pdf， Neural ODE

Continuous Normalizing Flows - Results

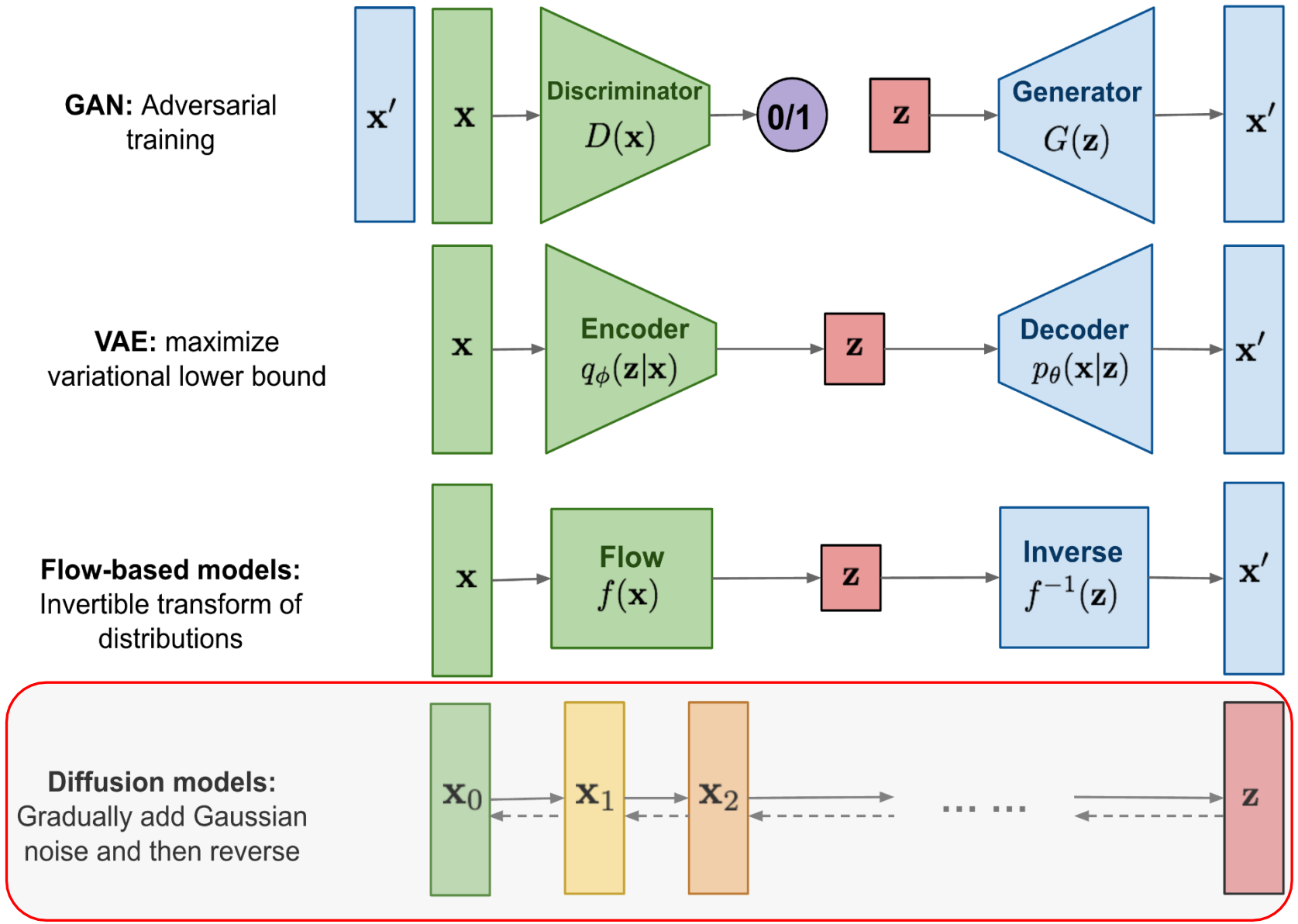


https://arxiv.org/pdf/1806.07366.pdf， Neural ODE

Normalizing flow的优缺点

- 优点：
  - 高灵活性：Normalizing flow模型的灵活性很高，可以使用各种变换来构建生成模型，包括仿射变换、非线性变换等，因此可以适用于不同的数据集。
  - 直接可逆：Normalizing flow模型的变换是直接可逆的，因此可以轻松地计算样本的概率密度函数，不需要使用复杂的采样算法。
  - 高效性：Normalizing flow模型的训练过程比较高效，可以使用反向传播算法进行优化，因此可以应用于大规模数据集。
  - 高质量的生成样本：Normalizing flow模型可以生成高质量的样本，这些样本与原始数据非常相似，因此可以应用于图像生成、语音生成等领域。
- 缺点：
  - 模型复杂：Normalizing flow模型比较复杂，需要使用多层非线性变换来构建生成模型，因此需要更多的计算资源和时间来训练。
  - 变换的选择：Normalizing flow模型的性能很大程度上取决于变换的选择，需要选择合适的变换来构建生成模型，而这个过程比较困难。
  - 模型的可解释性：Normalizing flow模型的可解释性较差，因为它是一种黑盒模型，不容易解释模型中的变换和参数的含义。

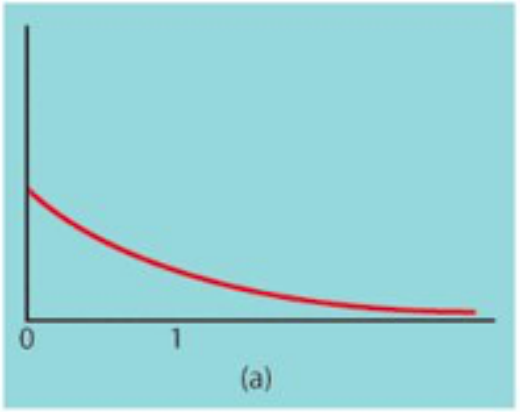
Generative Model Zoo

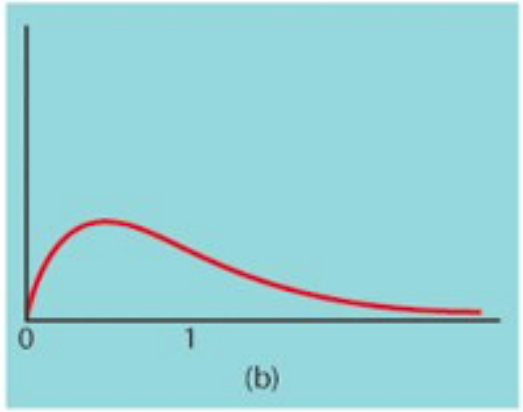


https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

Central Limit Theorem & Diffusion Process

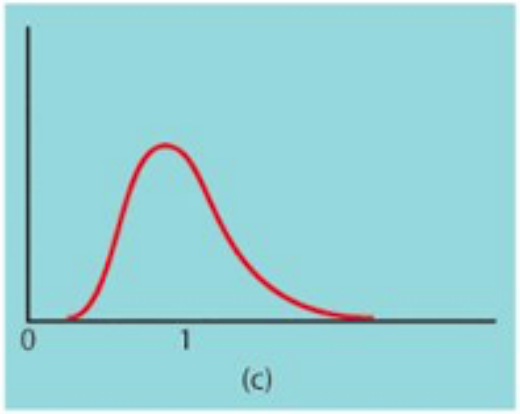
Population with strongly skewed distribution

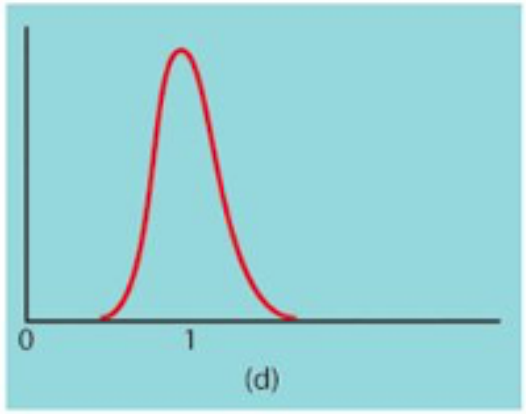




Sampling distribution of $\bar{x}$ for $n = 2$ observations

Sampling distribution of $\bar{x}$ for $n = 10$ observations

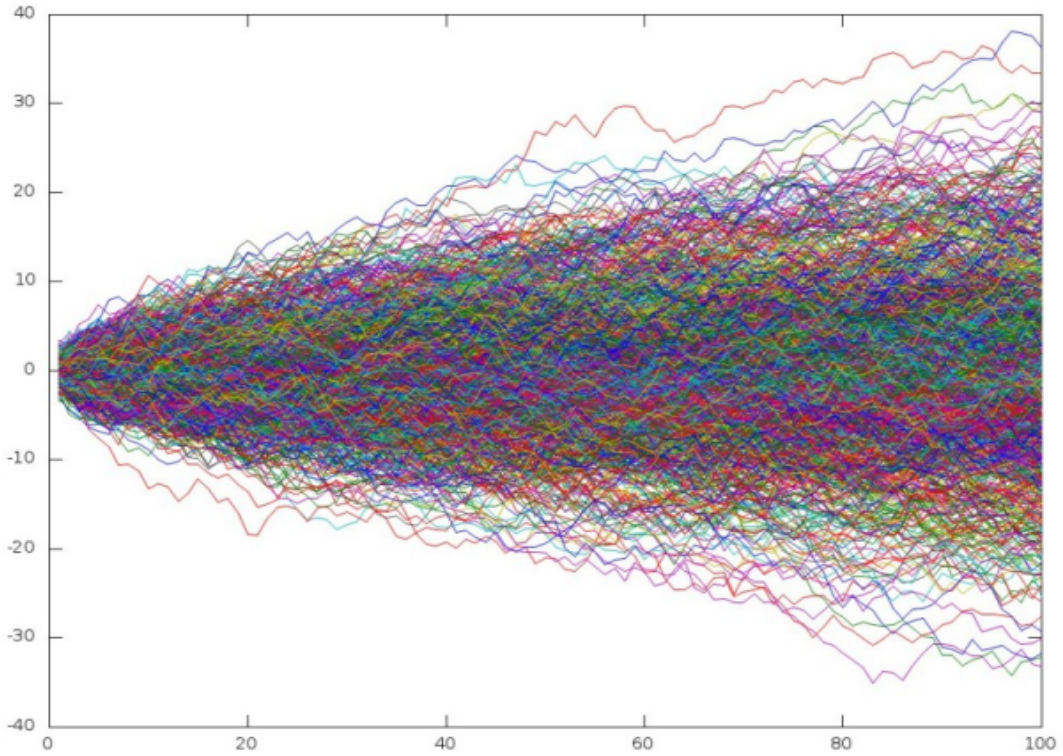


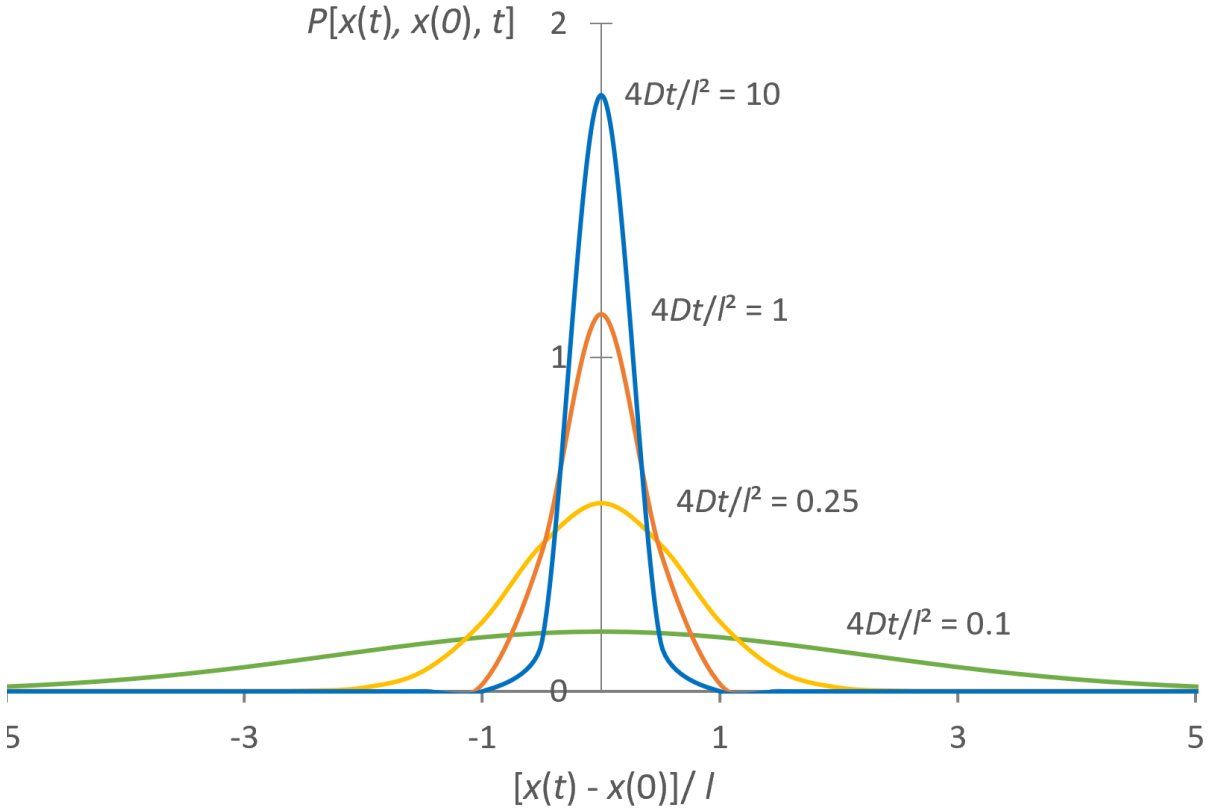


Sampling distribution of $\bar{x}$ for $n = 25$ observations

$$Y = \frac{X_1 + X_2 + \cdots + X_n}{n} \xrightarrow{n\to\infty} N(\mu, \sigma^2)$$

Central Limit Theorem & Diffusion Process





Diffusion Models

- Several diffusion-based generative models
  - *diffusion probabilistic models* (Sohl-Dickstein et al., 2015 https://arxiv.org/abs/1503.03585),
  - *noise-conditioned score network* (**NCSN**; Yang & Ermon, 2019 https://arxiv.org/abs/1907.05600),
  - *denoising diffusion probabilistic models* (**DDPM**; Ho et al. 2020 https://arxiv.org/abs/2006.11239).

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

Diffusion

Deep Unsupervised Learning using Nonequilibrium Thermodynamics

Jascha Sohl-Dickstein JASCHA@STANFORD.EDU
Stanford University

Eric A. Weiss EAWEISS@BERKELEY.EDU
University of California, Berkeley

Niru Maheswaranathan NIRUM@STANFORD.EDU
Stanford University

Surya Ganguli SGANGULI@STANFORD.EDU
Stanford University

Abstract

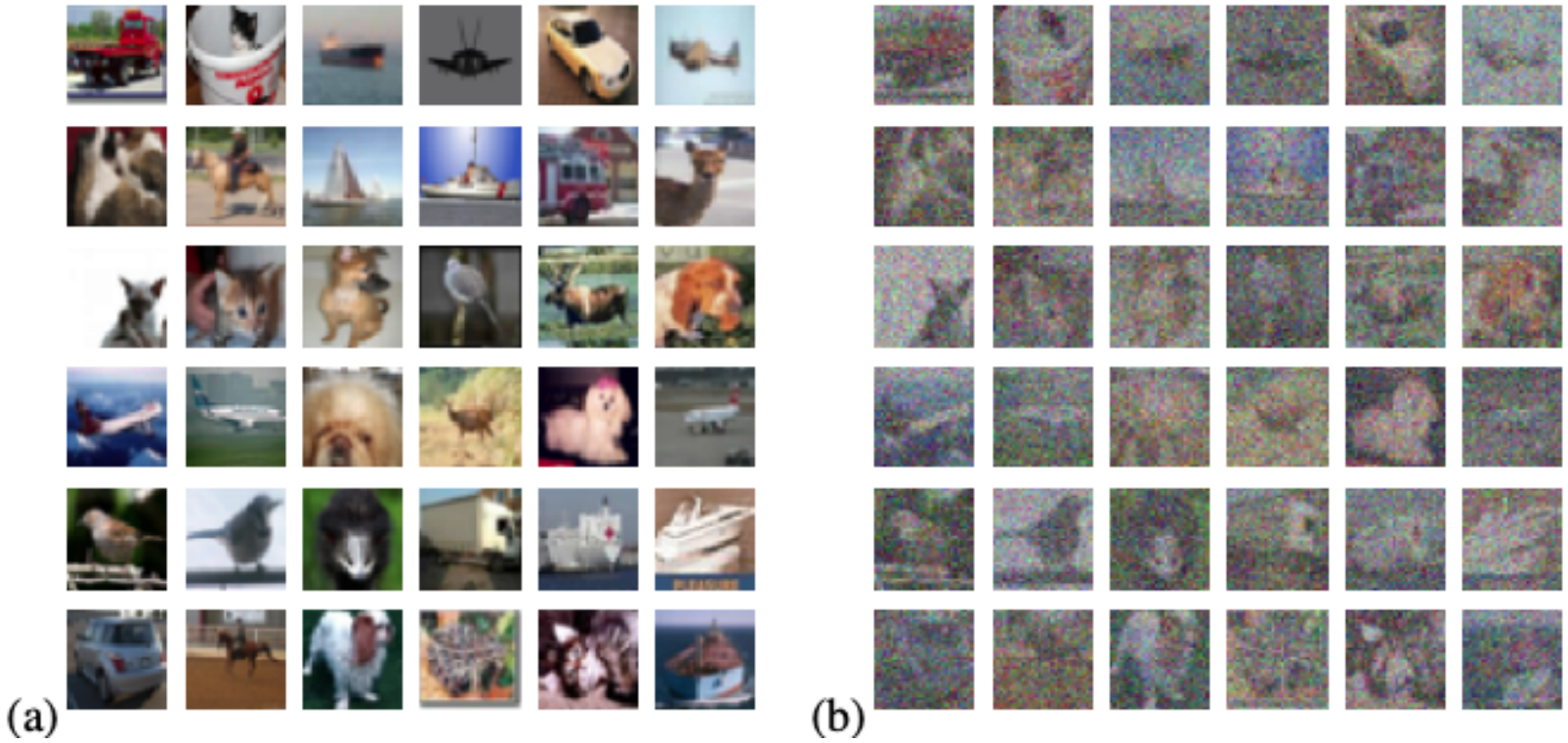
A central problem in machine learning involves modeling complex data-sets using highly flexible families of probability distributions in which learning, sampling, inference, and evaluation are still analytically or computationally tractable. Here, we develop an approach that simultaneously achieves both flexibility and tractability. The essential idea, inspired by non-equilibrium statistical physics, is to systematically and slowly destroy structure in a data distribution through an iterative forward diffusion process. We then learn a reverse diffusion process that restores structure in data, yielding a highly flexible and tractable generative model of the data. This approach allows us to rapidly learn, sample from, and evaluate probabilities in deep generative models with thousands of layers or time steps, as well as to compute conditional and posterior probabilities under the learned model. We additionally release an open source reference implementation of the algorithm.

these models are unable to aptly describe structure in rich datasets. On the other hand, models that are *flexible* can be molded to fit structure in arbitrary data. For example, we can define models in terms of any (non-negative) function $\phi(\mathbf{x})$ yielding the flexible distribution $p(\mathbf{x}) = \frac{\phi(\mathbf{x})}{Z}$, where Z is a normalization constant. However, computing this normalization constant is generally intractable. Evaluating, training, or drawing samples from such flexible models typically requires a very expensive Monte Carlo process.

A variety of analytic approximations exist which ameliorate, but do not remove, this tradeoff–for instance mean field theory and its expansions (T, 1982; Tanaka, 1998), variational Bayes (Jordan et al., 1999), contrastive divergence (Welling & Hinton, 2002; Hinton, 2002), minimum probability flow (Sohl-Dickstein et al., 2011b;a), minimum KL contraction (Lyu, 2011), proper scoring rules (Gneiting & Raftery, 2007; Parry et al., 2012), score matching (Hyvärinen, 2005), pseudolikelihood (Besag, 1975), loopy belief propagation (Murphy et al., 1999), and many, many more. Non-parametric methods (Gershman & Blei, 2012) can also be very effective[1].

https://arxiv.org/pdf/1503.03585.pdf

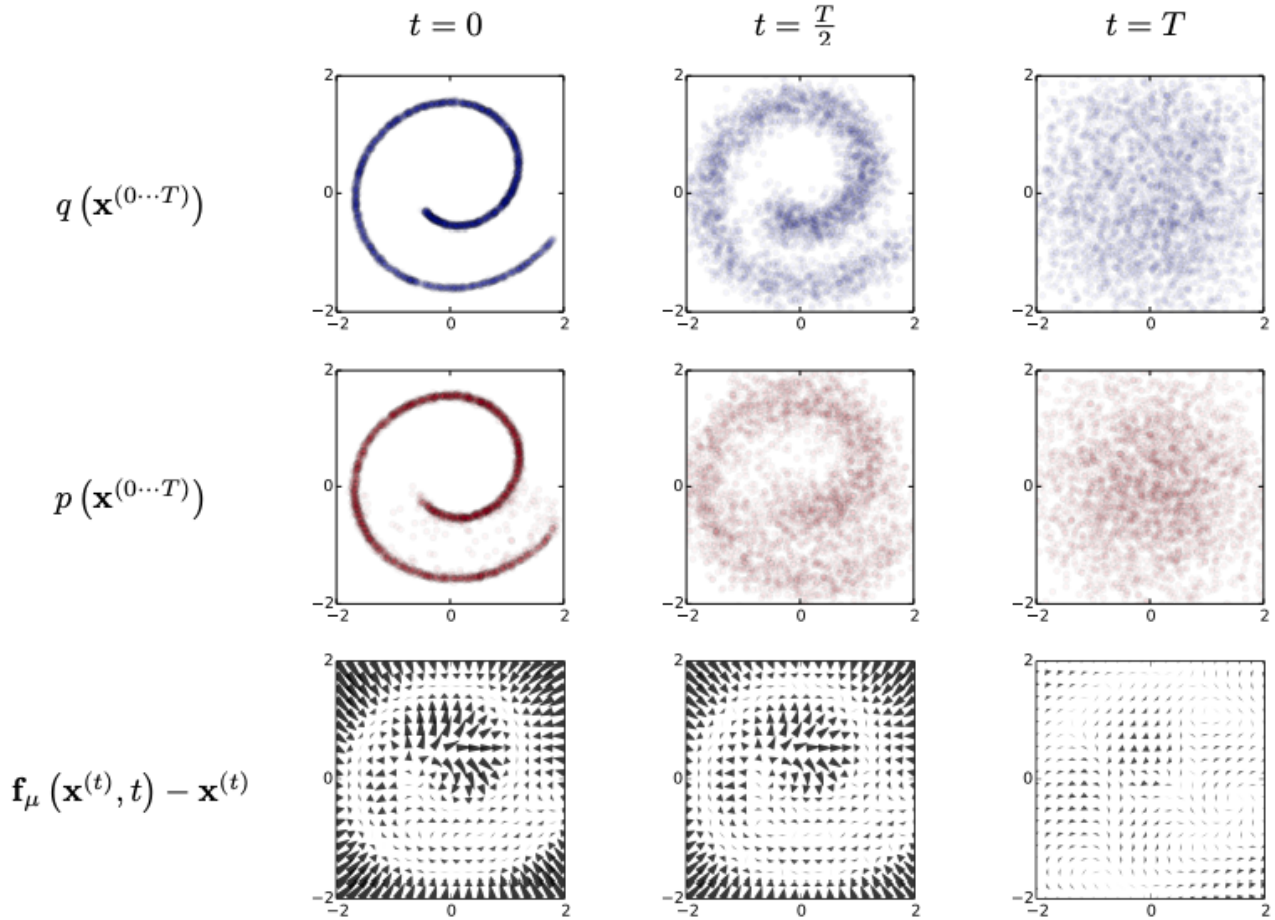
Basic Idea



How can you denoise an image if you have a large amount of image data

https://arxiv.org/pdf/1503.03585.pdf

Basic Idea



Forward process

Backward process

https://arxiv.org/pdf/1503.03585.pdf

Score function

Generative Modeling by Estimating Gradients of the Data Distribution

Yang Song
Stanford University
yangsong@cs.stanford.edu

Stefano Ermon
Stanford University
ermon@cs.stanford.edu

Abstract

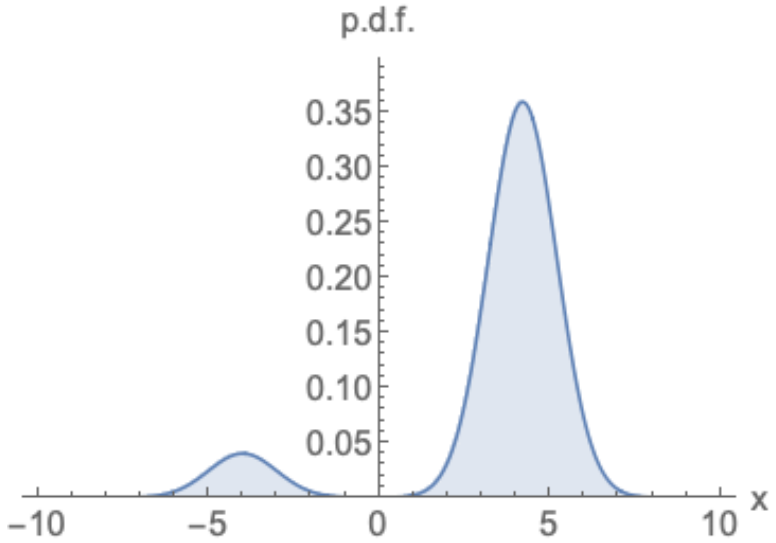
We introduce a new generative model where samples are produced via Langevin dynamics using gradients of the data distribution estimated with score matching. Because gradients can be ill-defined and hard to estimate when the data resides on low-dimensional manifolds, we perturb the data with different levels of Gaussian noise, and jointly estimate the corresponding scores, *i.e.*, the vector fields of gradients of the perturbed data distribution for all noise levels. For sampling, we propose an annealed Langevin dynamics where we use gradients corresponding to gradually decreasing noise levels as the sampling process gets closer to the data manifold. Our framework allows flexible model architectures, requires no sampling during training or the use of adversarial methods, and provides a learning objective that can be used for principled model comparisons. Our models produce samples comparable to GANs on MNIST, CelebA and CIFAR-10 datasets, achieving a new state-of-the-art inception score of 8.87 on CIFAR-10. Additionally, we demonstrate that our models learn effective representations via image inpainting experiments.

https://arxiv.org/pdf/1907.05600.pdf

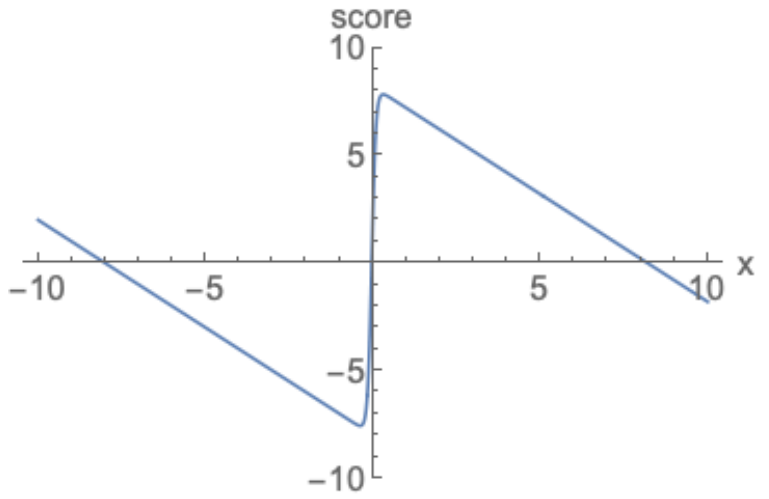
Score function

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

intractable



$$\mathbf{s}_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}) = -\nabla_{\mathbf{x}} f_\theta(\mathbf{x}) - \underbrace{\nabla_{\mathbf{x}} \log Z_\theta}_{=0} = -\nabla_{\mathbf{x}} f_\theta(\mathbf{x}).$$



$s_\theta(x)$: Score function

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

Basic Idea

Add noise

$$p_{\sigma_i}(\mathbf{x}) = \int p(\mathbf{y})\mathcal{N}(\mathbf{x};\mathbf{y},\sigma_i^2 I)\mathrm{d}\mathbf{y}.$$

$$x \sim p(x), \qquad x + \sigma_i z \qquad z \sim N(0, I)$$

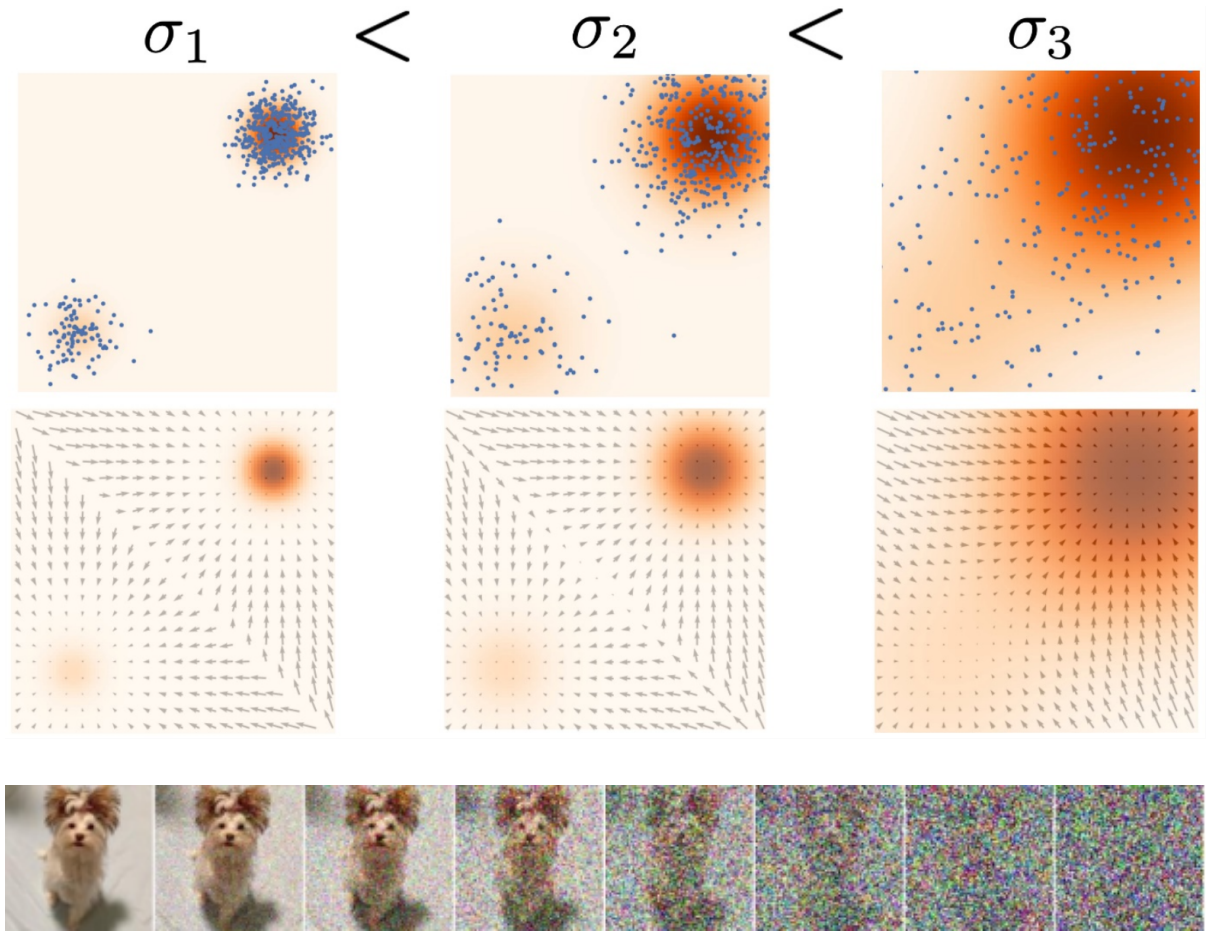
Learning:

$$s_\theta(x, i) \simeq \nabla_x \log p_{\sigma_i}(x)$$

For i=1,2,…L

Objective:

$$\sum_{i=1}^{L} \lambda(i)\mathbb{E}_{p_{\sigma_i}(\mathbf{x})}[\|\nabla_{\mathbf{x}} \log p_{\sigma_i}(\mathbf{x}) - \mathbf{s}_\theta(\mathbf{x}, i)\|_2^2],$$



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

How to estimate gradient of likelihood?

Journal of Machine Learning Research 6 (2005) 695–709 Submitted 11/04; Revised 3/05; Published 4/05

Estimation of Non-Normalized Statistical Models by Score Matching

Aapo Hyvärinen AAPO.HYVARINEN@HELSINKI.FI
Helsinki Institute for Information Technology (BRU)
Department of Computer Science
FIN-00014 University of Helsinki, Finland

Editor: Peter Dayan

Abstract

One often wants to estimate statistical models where the probability density function is known only up to a multiplicative normalization constant. Typically, one then has to resort to Markov Chain Monte Carlo methods, or approximations of the normalization constant. Here, we propose that such models can be estimated by minimizing the expected squared distance between the gradient of the log-density given by the model and the gradient of the log-density of the observed data. While the estimation of the gradient of log-density function is, in principle, a very difficult non-parametric problem, we prove a surprising result that gives a simple formula for this objective function. The density function of the observed data does not appear in this formula, which simplifies to a sample average of a sum of some derivatives of the log-density given by the model. The validity of the method is demonstrated on multivariate Gaussian and independent component analysis models, and by estimating an overcomplete filter set for natural image data.

Keywords: statistical estimation, non-normalized densities, pseudo-likelihood, Markov chain Monte Carlo, contrastive divergence

$$\sum_{i=1}^{L} \lambda(i) \mathbb{E}_{p_{\sigma_i}(\mathbf{x})} [\| \nabla_{\mathbf{x}} \log p_{\sigma_i}(\mathbf{x}) - \mathbf{s}_{\theta}(\mathbf{x}, i) \|_2^2],$$

$$L_i = \int_{x \in \mathbb{R}^n} p_{\sigma_i}(x) \| s_{\theta}(x, i) - \psi(x) \|^2 \, dx$$

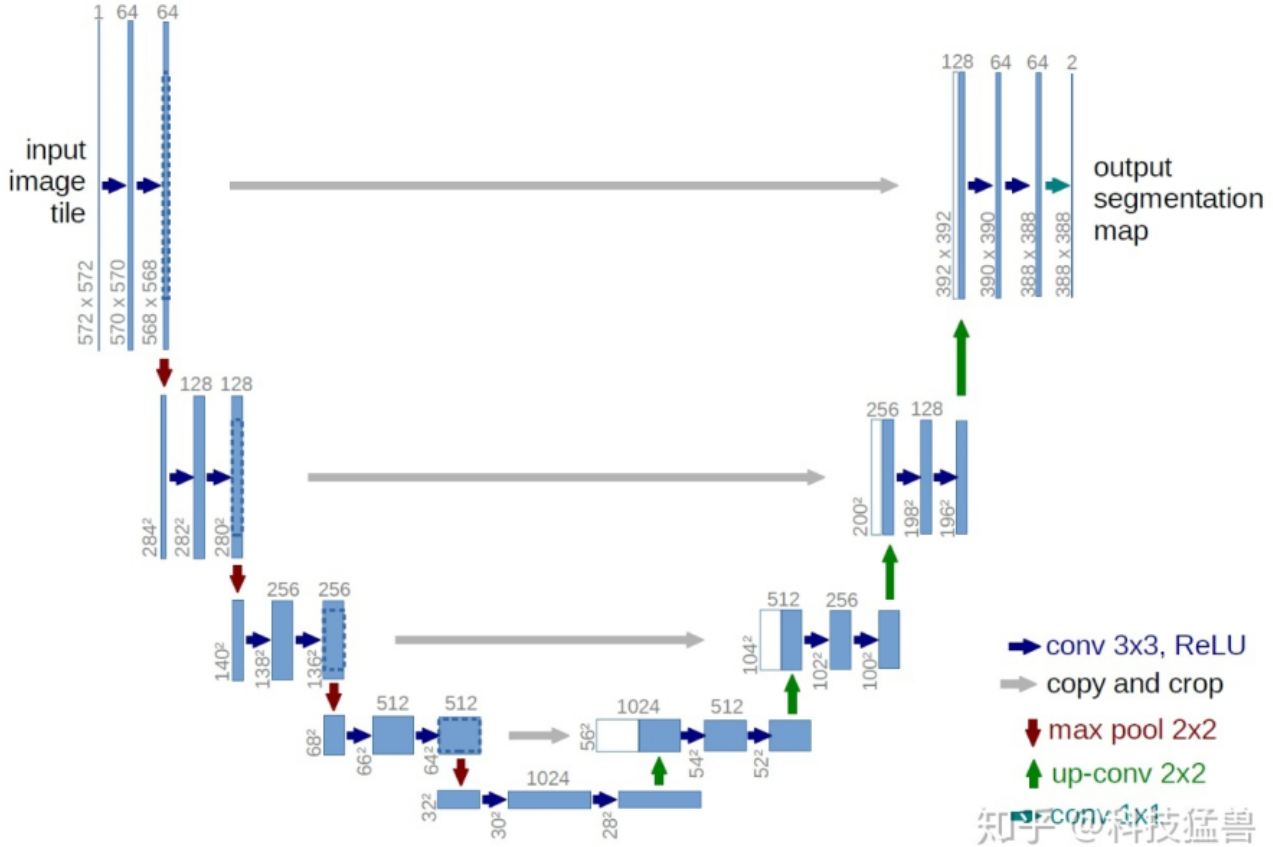
$$\psi(x) = \nabla_x \log p_{\sigma_i}(x)$$

It can be proved:

$$L_i = \int_{x \in \mathbb{R}^n} p_{\sigma_i}(x) \sum_{j=1}^{n} \left[\partial_j s_{\theta,j}(x) + \frac{1}{2} s_{\theta,j}(x) \right] dx + c$$

https://www.cs.helsinki.fi/u/ahyvarin/papers/JMLR05.pdf

How to model score function?



U-Net: Convolutional Networks for Biomedical Image Segmentation

Olaf Ronneberger, Philipp Fischer, and Thomas Brox

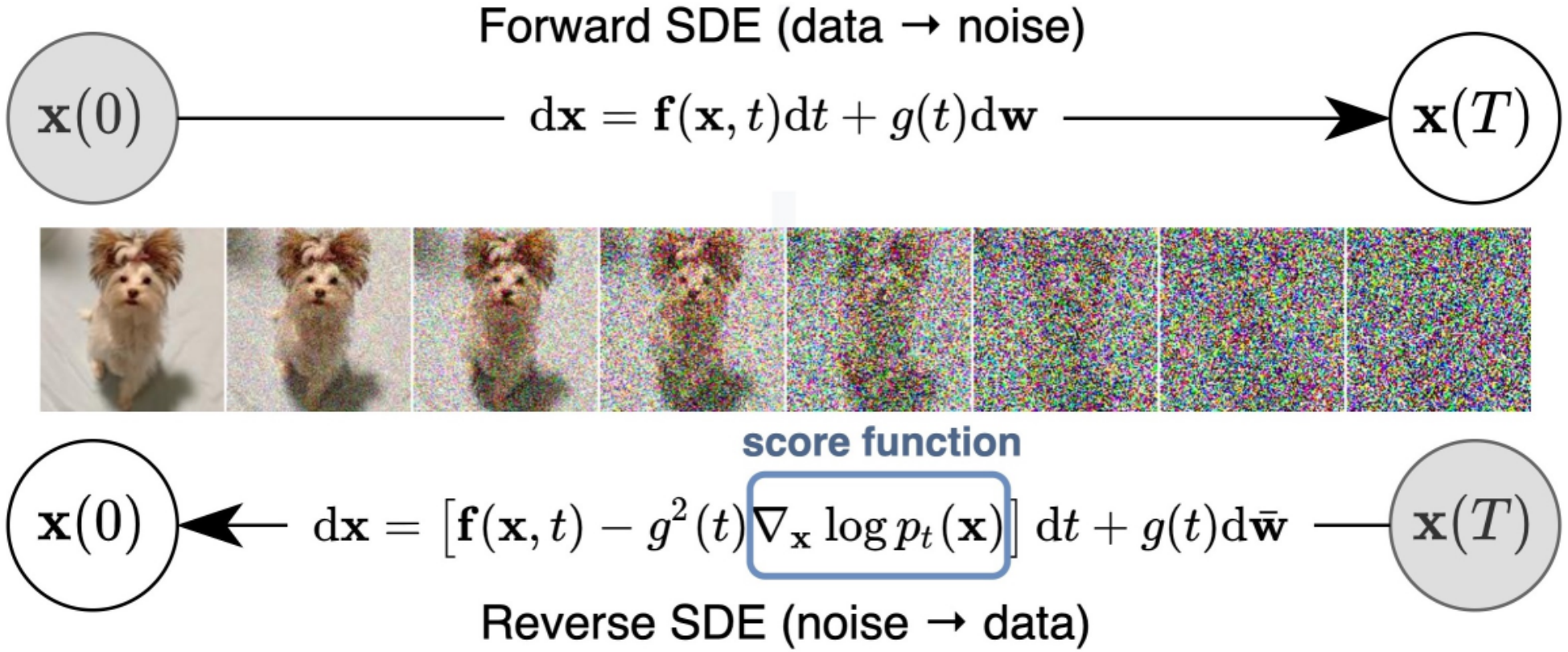
Computer Science Department and BIOSS Centre for Biological Signalling Studies, University of Freiburg, Germany
ronneber@informatik.uni-freiburg.de,
WWW home page: http://lmb.informatik.uni-freiburg.de/

Abstract. There is large consent that successful training of deep networks requires many thousand annotated training samples. In this paper, we present a network and training strategy that relies on the strong use of data augmentation to use the available annotated samples more efficiently. The architecture consists of a contracting path to capture context and a symmetric expanding path that enables precise localization. We show that such a network can be trained end-to-end from very few images and outperforms the prior best method (a sliding-window convolutional network) on the ISBI challenge for segmentation of neuronal structures in electron microscopic stacks. Using the same network trained on transmitted light microscopy images (phase contrast and DIC) we won the ISBI cell tracking challenge 2015 in these categories by a large margin. Moreover, the network is fast. Segmentation of a 512x512 image takes less than a second on a recent GPU. The full implementation (based on Caffe) and the trained networks are available at http://lmb.informatik.uni-freiburg.de/people/ronneber/u-net.

1 Introduction

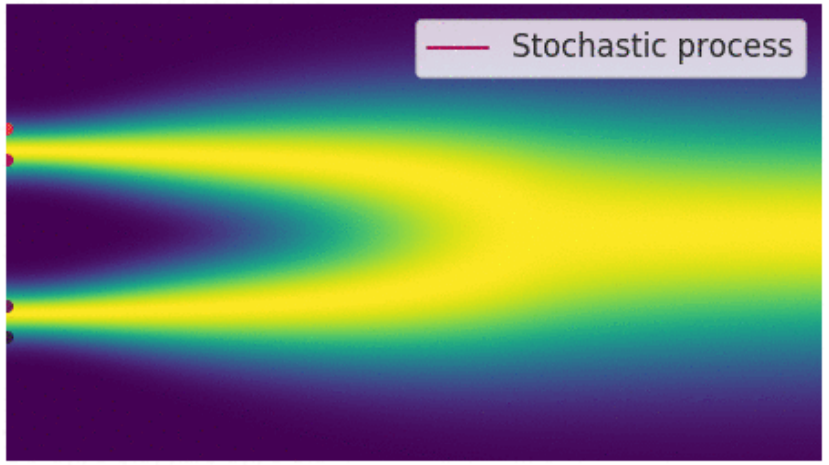
https://arxiv.org/pdf/1505.04597.pdf

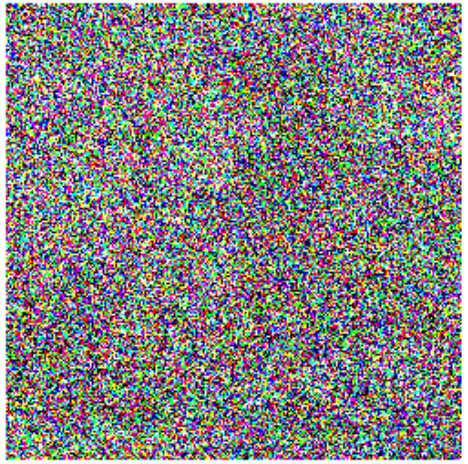
Stochastic Differential Equation

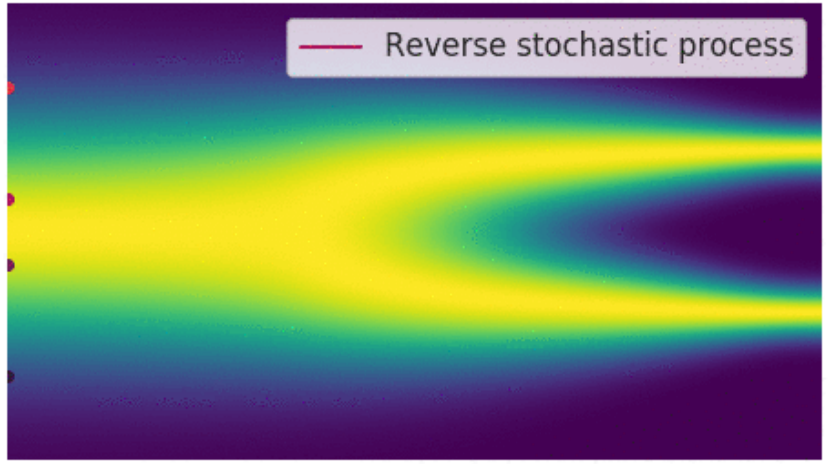


https://yang-song.net/blog/2021/score/

Stochastic Differential Equation

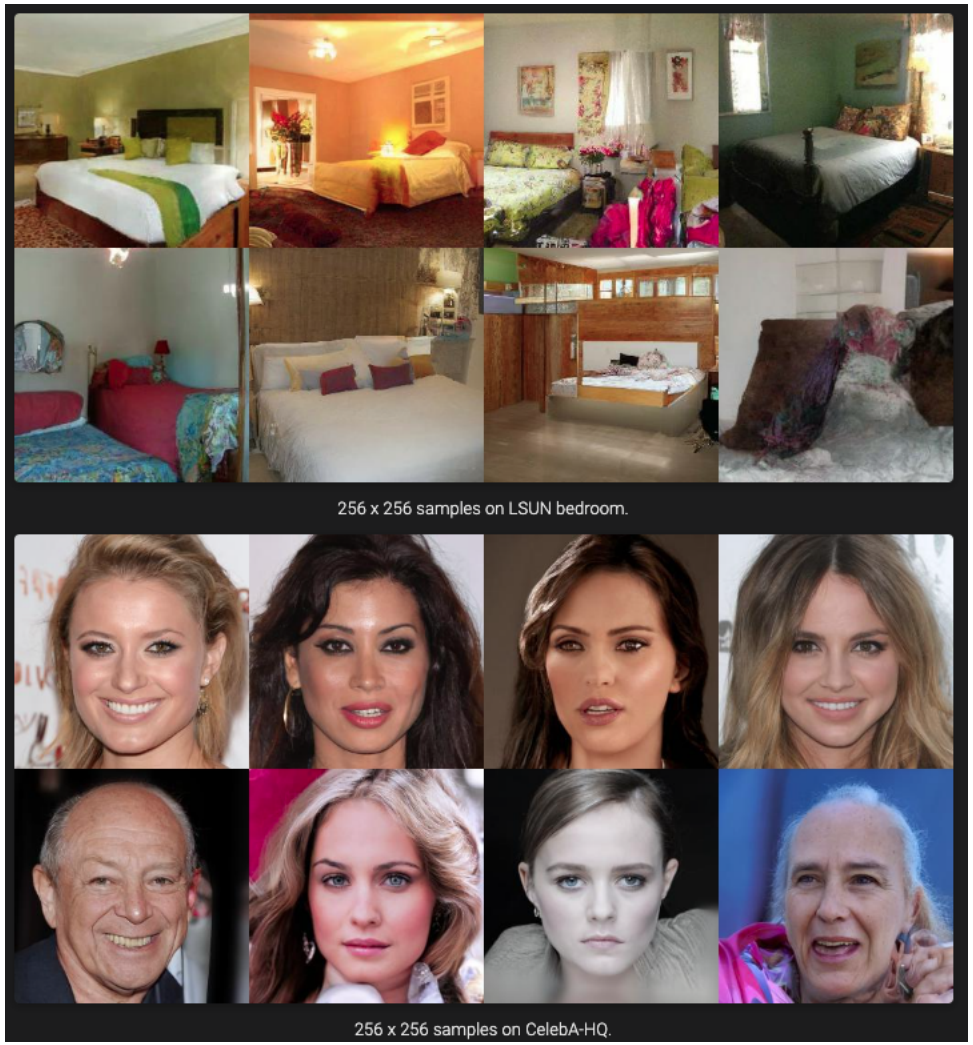






https://yang-song.net/blog/2021/score/

Results



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

DDPM

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

Abstract

We present high quality image synthesis results using diffusion probabilistic models, a class of latent variable models inspired by considerations from nonequilibrium thermodynamics. Our best results are obtained by training on a weighted variational bound designed according to a novel connection between diffusion probabilistic models and denoising score matching with Langevin dynamics, and our models naturally admit a progressive lossy decompression scheme that can be interpreted as a generalization of autoregressive decoding. On the unconditional CIFAR10 dataset, we obtain an Inception score of 9.46 and a state-of-the-art FID score of 3.17. On 256x256 LSUN, we obtain sample quality similar to ProgressiveGAN. Our implementation is available at https://github.com/hojonathanho/diffusion.

https://arxiv.org/pdf/2006.11239.pdf