TECHNISCHE UNIVERSITÄT DRESDEN

Applications of Diffusion Models in Communications

Technische Universität Dresden
Chair of Information Theory and Machine Learning

Presenters: Muah Kim, Rick Fritschek, Rafael F. Schaefer

IEEE International Conference on Machine Learning for Communication and Networking
04.05.2024 @ Stockholm, Sweden

Too many generative models. What to use for my research?

Let’s say we buy diffusion models, the top seller.

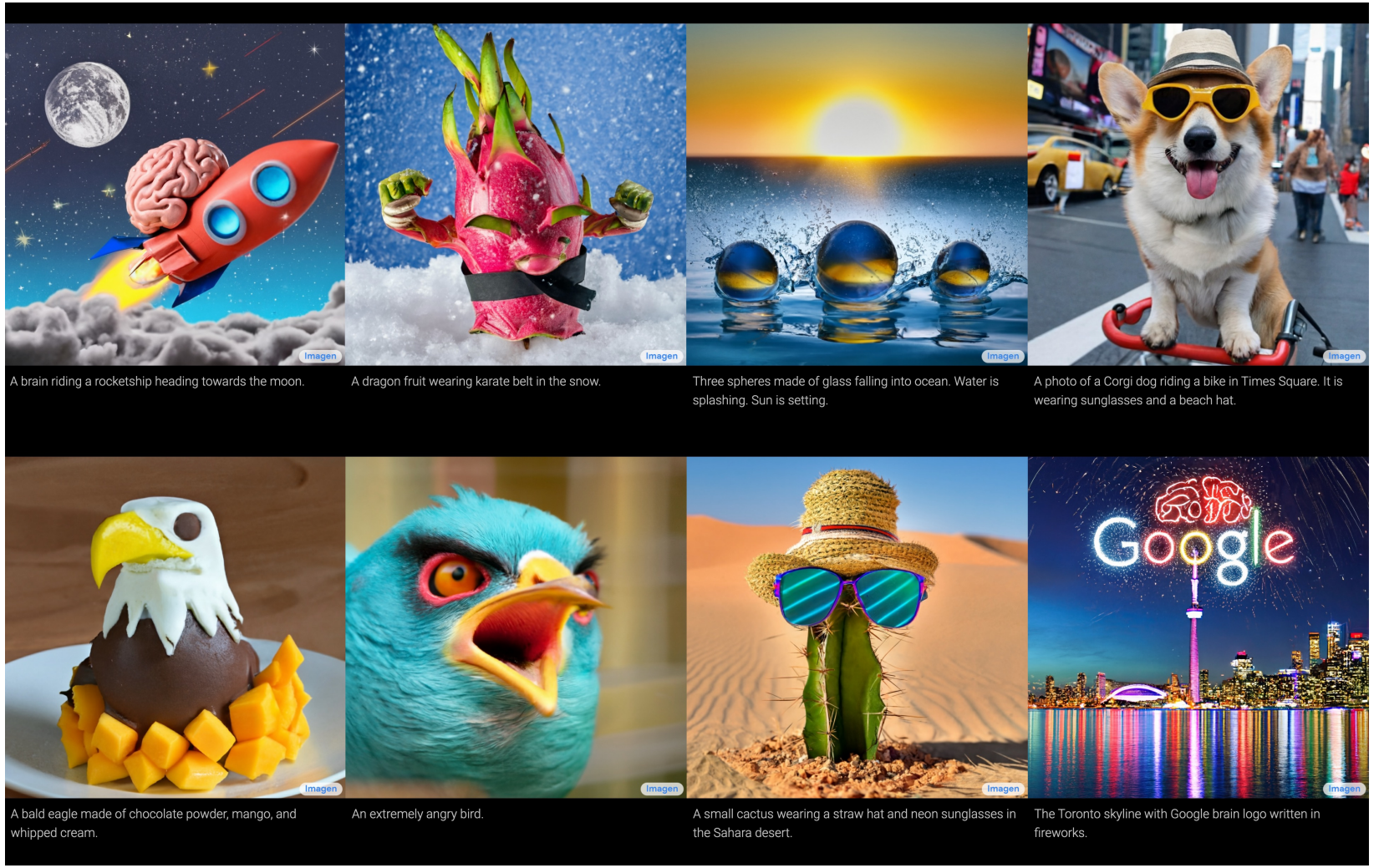
How to use diffusion models now?

What’s already done with diffusion models for communications?

Overview

- Part 1. Introduction to Diffusion Denoising Probabilistic Models (DDPM)
- Part 2. Advanced Techniques of DDPM
- Part 3. Applications of Diffusion Models in Communications

Diffusion Models for Text-to-Image Generation: Imagen



Diffusion Models for Text-to-Video Generation: SORA

The camera directly faces colorful buildings in Burano Italy. An adorable dalmatian looks through a window on a building on the ground floor…

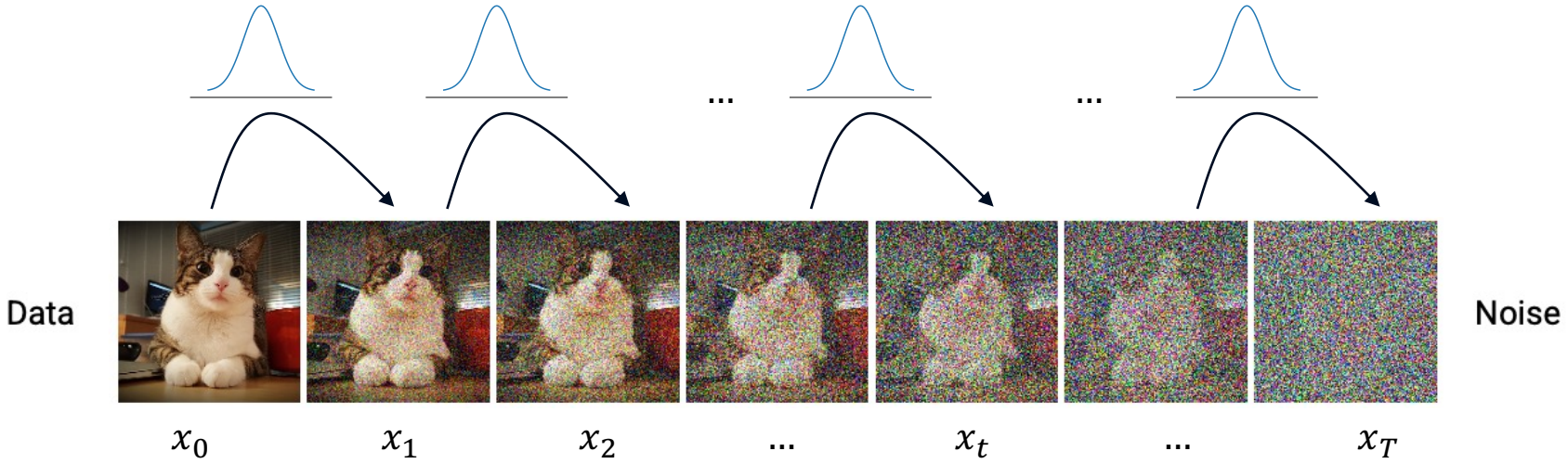
A little golden retriever puppies playing in the snow. Their heads pop out of the snow covered in.

An adorable happy otter confidently stands on a surfboard wearing a yellow lifejacket, riding along turquoise tropical waters near lush tropical islands, 3D digital render art style.

Extreme close up of a 24 year old woman’s eye blinking, standing in Marrakech during magic hour, cinematic film shot in 70mm, depth of field, vivid colors, cinematic.

Source: OpenAI, https://openai.com/index/sora

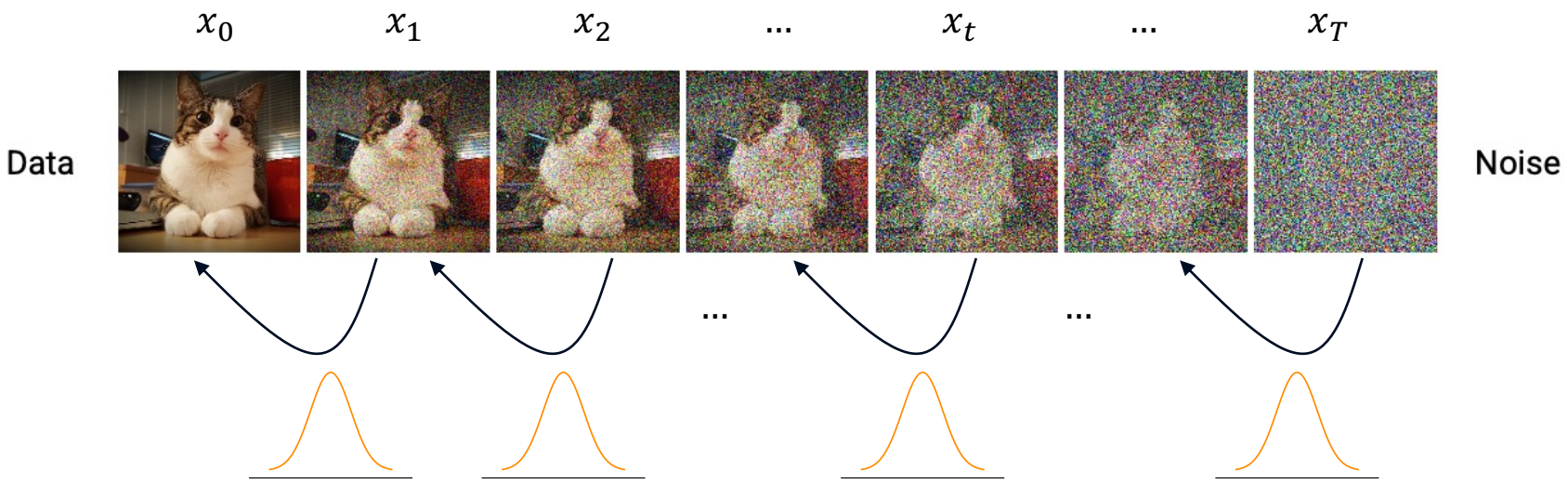
Diffusion Models: Gradual Diffusion



Diffusion Process with Defined Gaussian Distributions

Kreis, Gao, and Vahdat. (2022). Diffusion and Denoising Processes [Image]. Retrieved from https://cvpr2022-tutorial-diffusion-models.github.io/

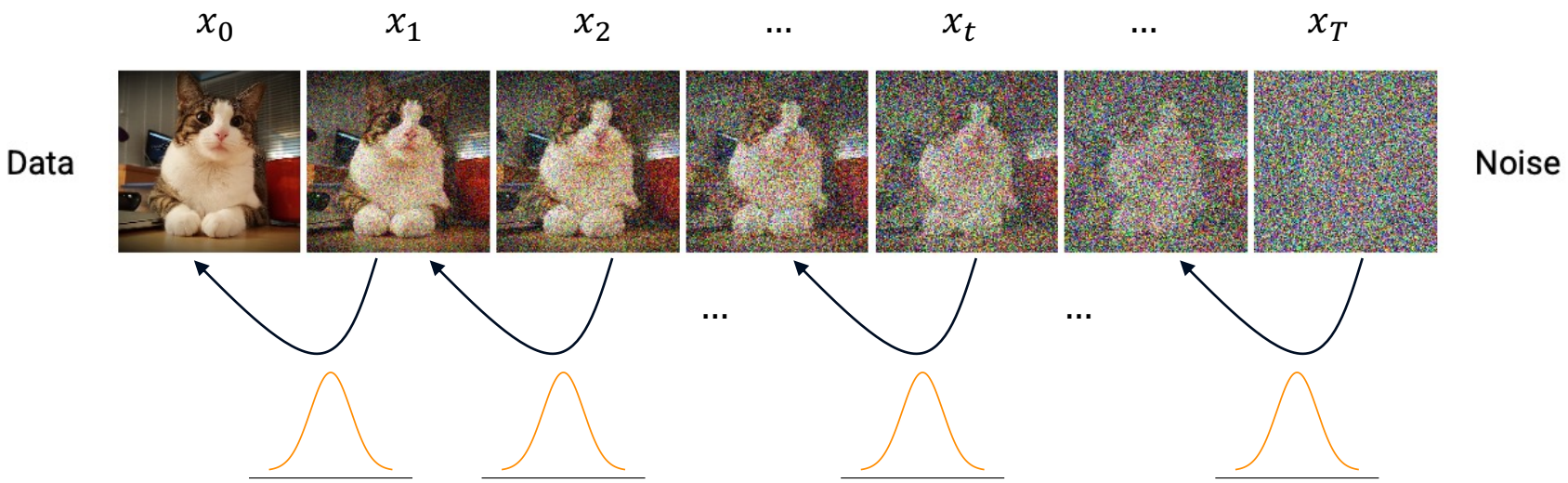
Diffusion Models: Gradual Denoising



Denoising Process by Parametric Estimation of Noise

Kreis, Gao, and Vahdat. (2022). Diffusion and Denoising Processes [Image]. Retrieved from https://cvpr2022-tutorial-diffusion-models.github.io/

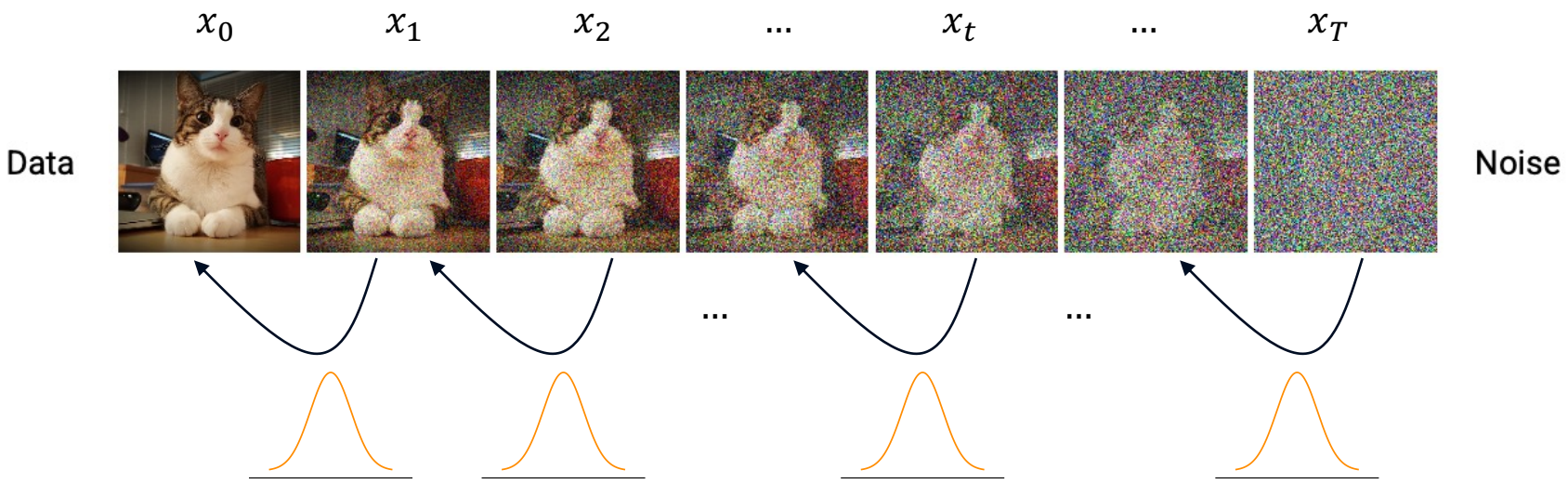
Diffusion Models: Gradual Denoising



$\Rightarrow$ Generating samples out of noise.

Kreis, Gao, and Vahdat. (2022). Diffusion and Denoising Processes [Image]. Retrieved from https://cvpr2022-tutorial-diffusion-models.github.io/

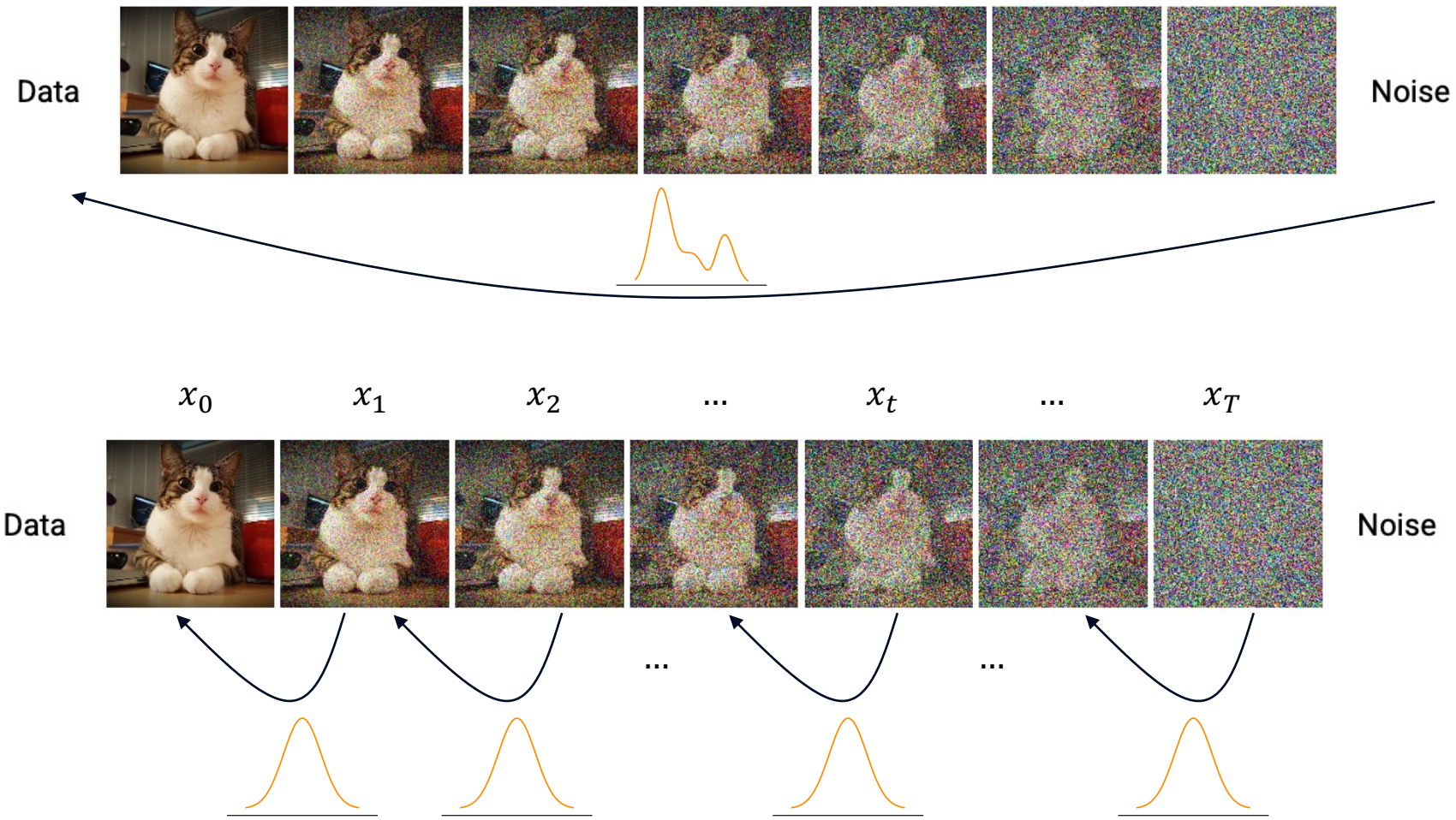
Diffusion Models: Gradual Denoising



Q. Why do we use gradual denoising instead of doing it in one step?

Kreis, Gao, and Vahdat. (2022). Diffusion and Denoising Processes [Image]. Retrieved from https://cvpr2022-tutorial-diffusion-models.github.io/

Diffusion Models: Gradual Denoising



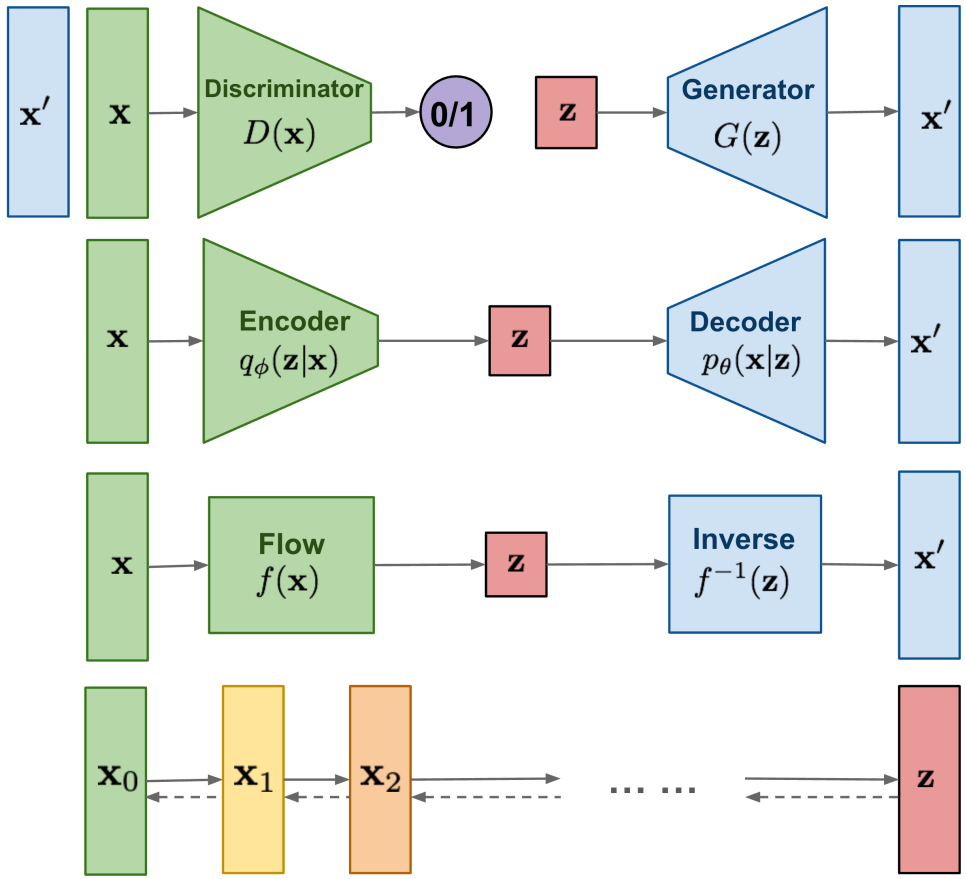
Comparison to other generative models

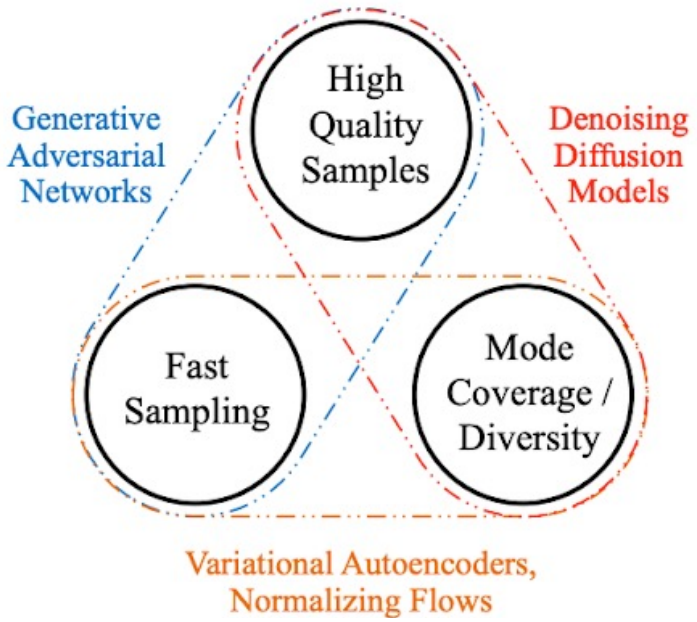
GAN: Adversarial training

VAE: maximize variational lower bound

Flow-based models: Invertible transform of distributions

Diffusion models: Gradually add Gaussian noise and then reverse

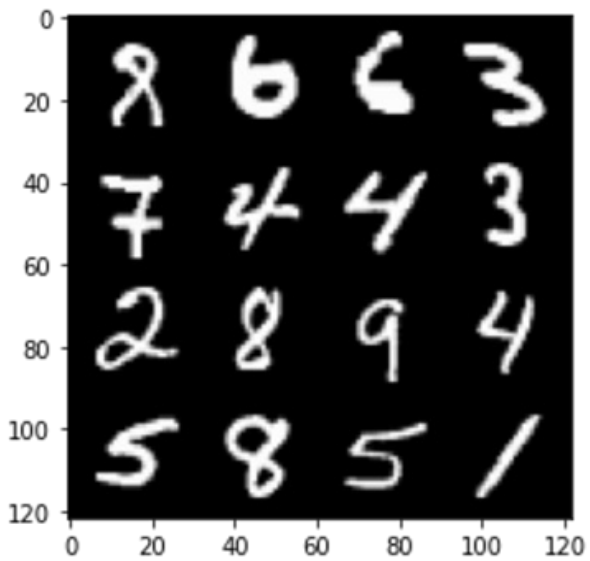




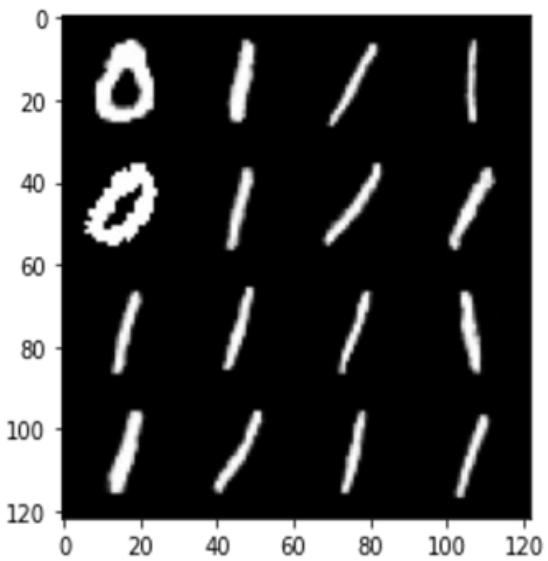
Wang, "What are Diffusion Models?", 2021
Vahdat and Kreis, Generative learning trilemma, 2022.

GANs' mode collapse

Generating only a few types of samples



Train Data Point



GAN Generated Data Point

Jaramillo, Addressing Mode Collapse in Generative Adversarial Networks (GANs): Enhancing the Diversity and Quality of Generated Outputs, 2023.

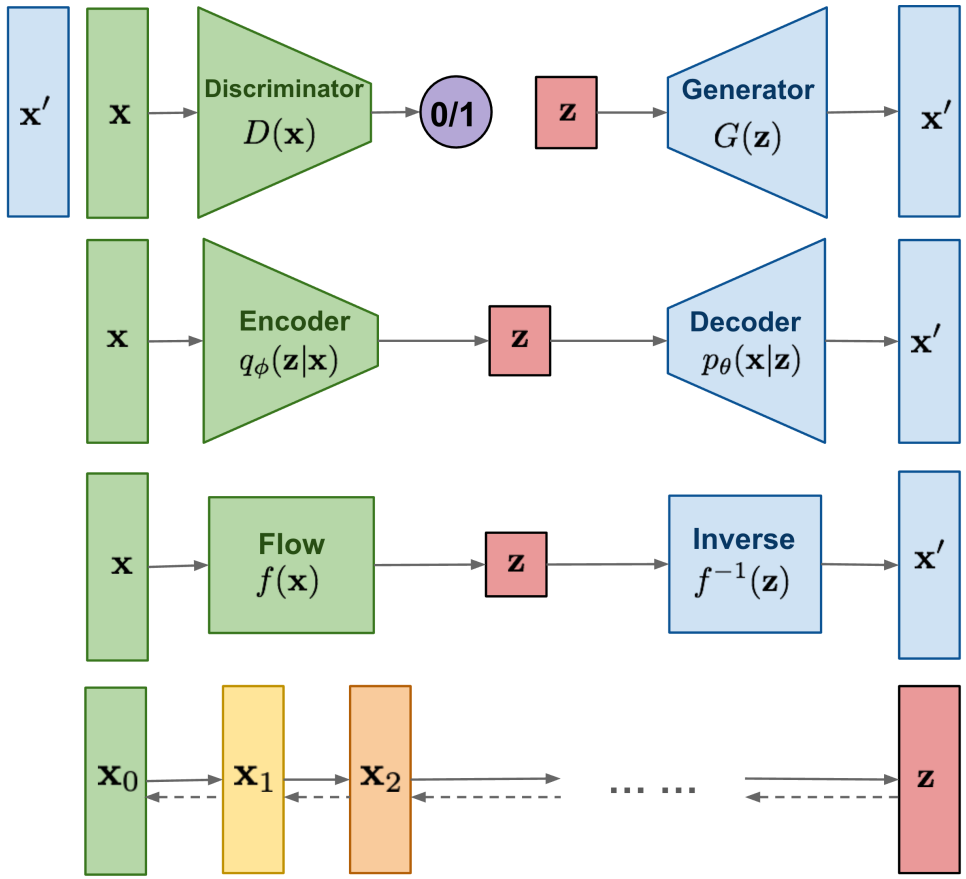
Comparison to other generative models

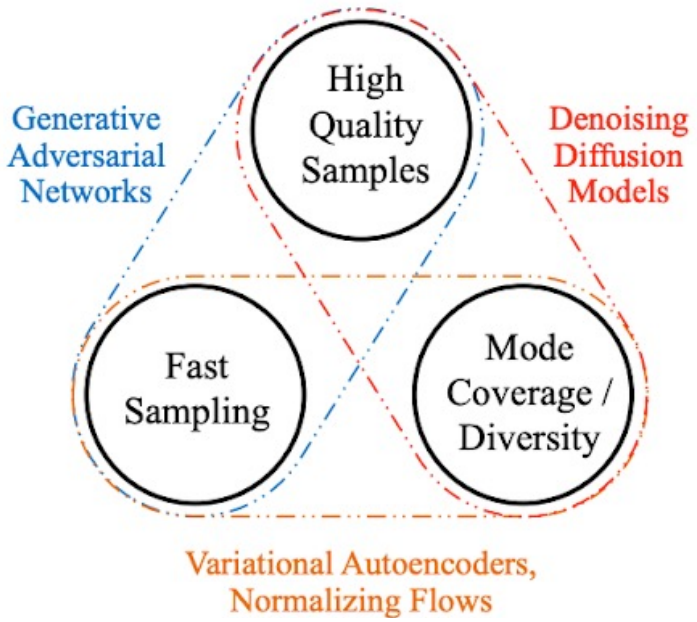
GAN: Adversarial training

VAE: maximize variational lower bound

Flow-based models: Invertible transform of distributions

Diffusion models: Gradually add Gaussian noise and then reverse





Wang, "What are Diffusion Models?", 2021
Vahdat and Kreis, Generative learning trilemma, 2022.

Diffusion Models

1. Diffusion Denoising Probabilistic Model (DDPM)

Sohl-Dickstein, Weiss, Maheswaranathan, and Ganguli, “Deep unsupervised learning using non-equilibrium thermodynamics,” ICML, 2015
Ho, Jain, and Abbeel, “Denoising diffusion probabilistic models,” NeurIPS, 2020.

2. Noise Conditional Score Network (NCSN)

Song and Ermon, “Generative modeling by estimating gradients of the data distribution,” NeurIPS, 2019.

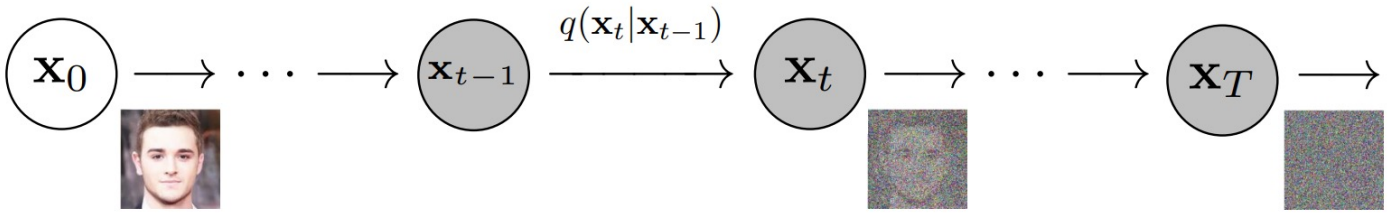
3. Stochastic-Differential Equation (SDE)

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

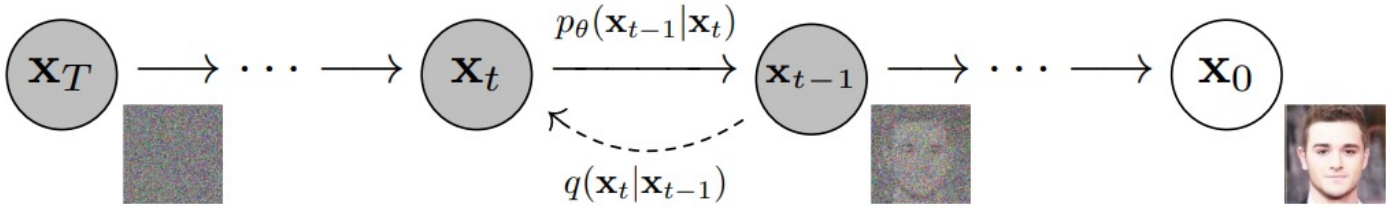
Different diffusion process, loss functions, parametrization technique, etc..

Diffusion Denoising Probabilistic Models (DDPMs)

Diffusion Process: add small Gaussian noise until the data follows almost $\sim N(\mathbf{0}, \boldsymbol{I})$



Denoising Process: learn the distribution of denoising each step

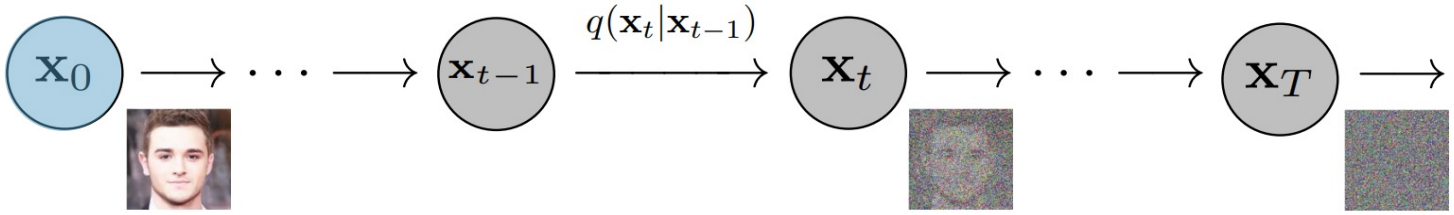


Ho, Jain, and Abbeel, “Denoising diffusion probabilistic models,” NeurIPS, 2020.

For detailed proofs: Luo. “Understanding Diffusion Models: A Unified Perspective”, arXiv, 2022

Definition of Diffusion Process

Diffusion Process



$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = N(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}) \qquad 0 < \beta_t \ll 1$$

$$\begin{aligned}
\boldsymbol{x}_t &= \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}, && \boldsymbol{\epsilon} \sim N(\boldsymbol{0}, \boldsymbol{I}) \\
&= \sqrt{\alpha_t}\boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}, && \alpha_t = 1 - \beta_t \\
&= \text{... (solving recursion)} \\
&= \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, && \bar{\alpha}_t = \Pi_{i=1}^{t}\alpha_i
\end{aligned}$$

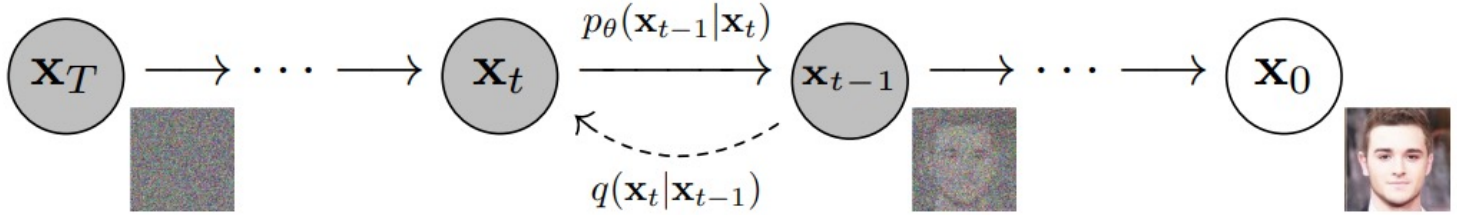
Sample from the target source

Ho, Jain, and Abbeel, “Denoising diffusion probabilistic models,” NeurIPS, 2020.

For detailed proofs: Luo. “Understanding Diffusion Models: A Unified Perspective”, arXiv, 2022

Unknown Denoising Process to be Learned

Denoising Process



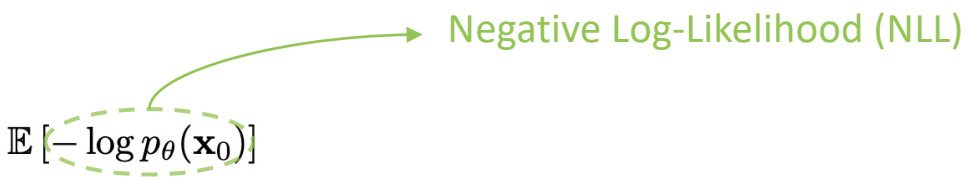
The true denoising distribution $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ is unknown.

$\Rightarrow$ Estimated by deep learning, $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$!

Ho, Jain, and Abbeel, "Denoising diffusion probabilistic models," NeurIPS, 2020.

For detailed proofs: Luo. "Understanding Diffusion Models: A Unified Perspective", arXiv, 2022

Training Loss for Denoising: Maximizing Likelihood

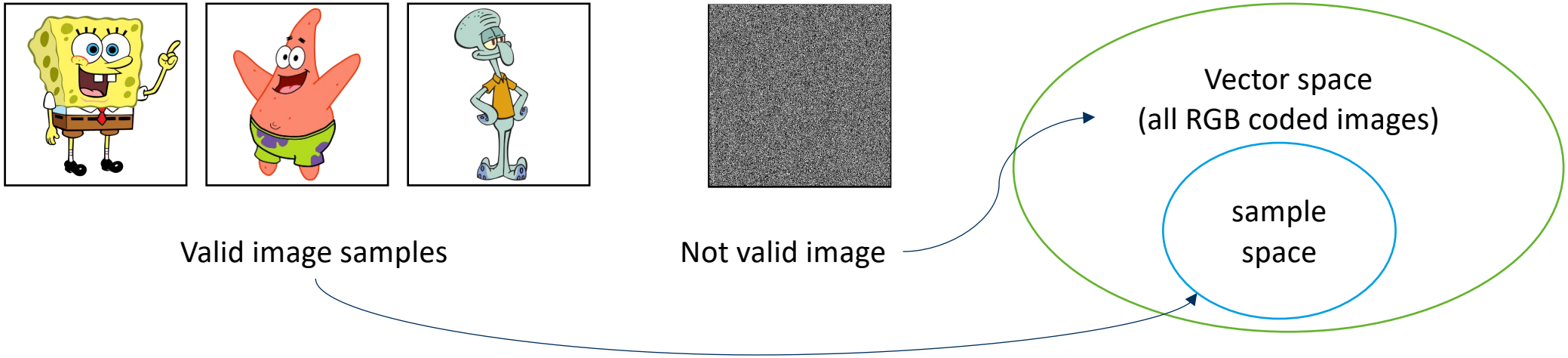


Training Loss for Denoising: Maximizing Likelihood

Negative Log-Likelihood (NLL)

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right]$$

Likelihood: the probability of the valid samples for the given parameters.



Maximizing likelihood? Maximizing the probability of valid samples!
(among generated samples by the NN associated with $\boldsymbol{\theta}$.)

Training Loss for Denoising: Maximizing Likelihood

Negative Log-Likelihood (NLL)

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right]$$

Calculating the exact likelihood is prohibitive and not practical for training.

- Need the marginal probability $p_\theta(\boldsymbol{x}_0)$.
- Expectation over all possible samples $\{\boldsymbol{x}_0\}$.

Training Loss for Denoising: Evidence Lower Bound (ELBO)

Negative Log-Likelihood (NLL)

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$$

Evidence lower bound (ELBO): Worst case

💡 $D_{KL}(p||q) := \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx$

$$L = \mathbb{E}_q\Bigg[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T))}_{L_T} + \sum_{t>1} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\Bigg]$$

Ho, Jain, and Abbeel, "Denoising diffusion probabilistic models," *NeurIPS*, 2020.

Training Loss for Denoising: Evidence Lower Bound (ELBO)

Negative Log-Likelihood (NLL)

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$$

Evidence lower bound (ELBO): Worst case

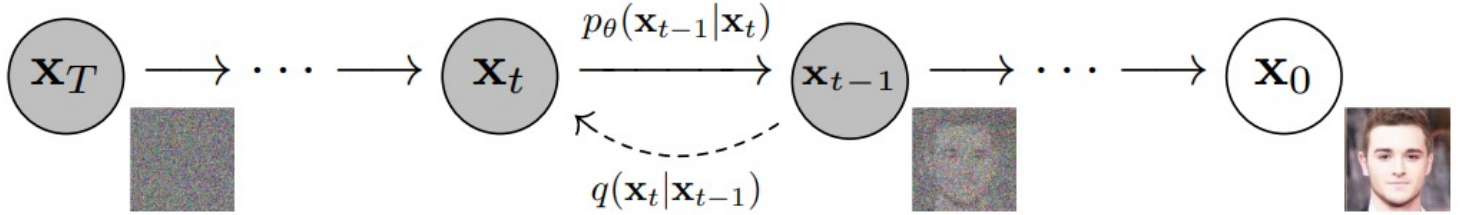
💡 $D_{KL}(p||q) := \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx$

$$L = \mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T))}_{L_T} + \sum_{t>1} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\right]$$

Ho, Jain, and Abbeel, "Denoising diffusion probabilistic models," *NeurIPS*, 2020.

Denoising Process for Training – Parametrization and Cost Function

Denoising Process



Given $\boldsymbol{x}_0$, $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ is known:

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = N\left(\boldsymbol{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0), \tilde{\beta}_t \boldsymbol{I}\right)$$

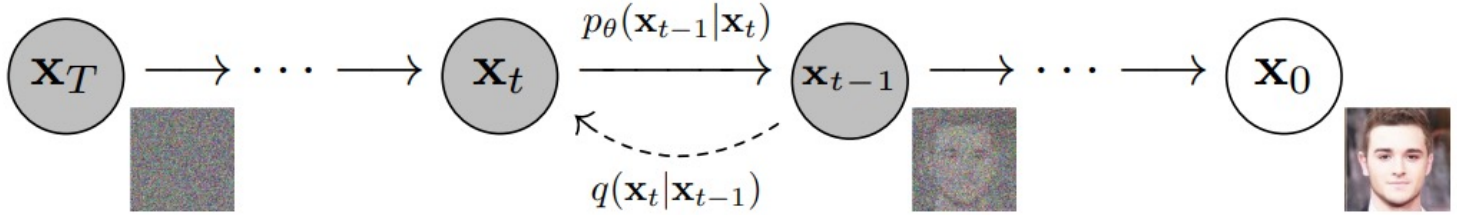
$$\tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) \coloneqq \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t(\boldsymbol{x}_0) - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right) \text{ with } \boldsymbol{\epsilon}_t \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{I}),$$

$$\tilde{\beta}_t \coloneqq \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t.$$

Ho, Jain, and Abbeel, "Denoising diffusion probabilistic models," *NeurIPS*, 2020.

Denoising Process for Training – Parametrization and Cost Function

Denoising Process



Given $\boldsymbol{x}_0$, $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ is known:

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = N\left(\boldsymbol{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0), \tilde{\beta}_t \boldsymbol{I}\right)$$

$$\tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) \coloneqq \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t(\boldsymbol{x}_0) - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right) \text{ with } \boldsymbol{\epsilon}_t \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{I}),$$

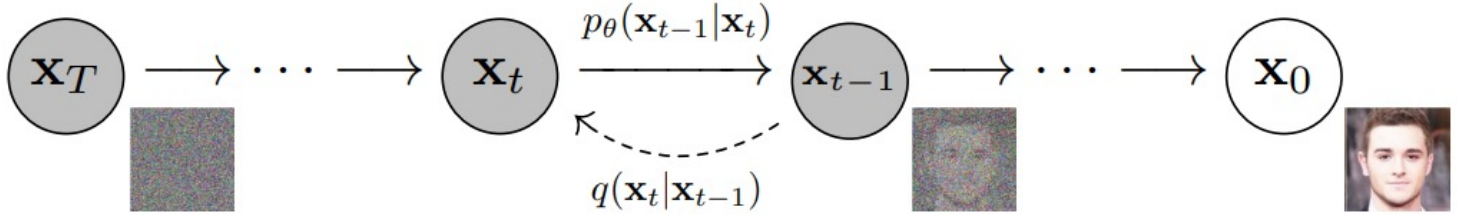
$$\tilde{\beta}_t \coloneqq \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t.$$

Random noise in the sample $\boldsymbol{x}_t$

Ho, Jain, and Abbeel, "Denoising diffusion probabilistic models," *NeurIPS*, 2020.

Denoising Process for Training – Parametrization and Cost Function

Denoising Process

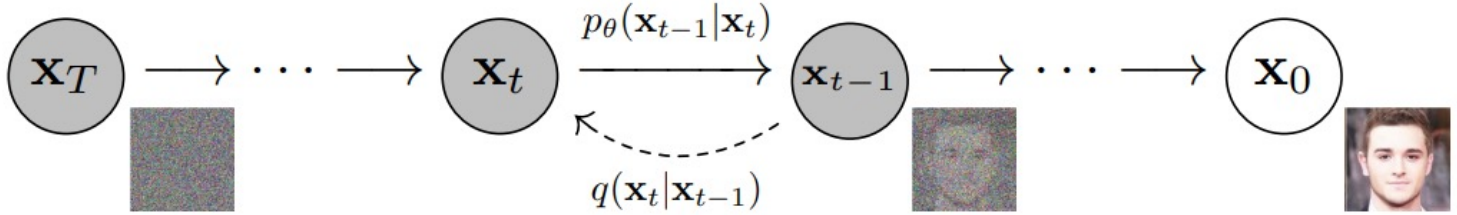


True Distribution (when $\boldsymbol{x}_0$ is known)	Learned Distribution
$q(\boldsymbol{x}_{t-1}\|\boldsymbol{x}_t, \boldsymbol{x}_0) = N\left(\boldsymbol{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0), \tilde{\beta}_t \boldsymbol{I}\right)$	$p_\theta(\boldsymbol{x}_{t-1}\|\boldsymbol{x}_t) := N\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \tilde{\beta}_t \boldsymbol{I}\right)$
$\tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t(\boldsymbol{x}_0) - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right),$ $\boldsymbol{\epsilon}_t \sim N(\boldsymbol{0}, \boldsymbol{I})$	$\boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t) := \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)\right)$

Ho, Jain, and Abbeel, "Denoising diffusion probabilistic models," *NeurIPS*, 2020.

Denoising Process for Training – Parametrization and Cost Function

Denoising Process



Estimating $\boldsymbol{x}_{t-1}$ for given $\boldsymbol{x}_t$
$\Rightarrow$ Estimating the noise $\boldsymbol{\epsilon}_t$ in $\boldsymbol{x}_t$ in the parametric form.

$q(\boldsymbol{x}_{t-1}\|\boldsymbol{x}_t,\boldsymbol{x}_0)$	$p_\theta(\boldsymbol{x}_{t-1}\|\boldsymbol{x}_t)$
$= N\left(\boldsymbol{x}_{t-1};\tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t,\boldsymbol{x}_0),\tilde{\beta}_t\boldsymbol{I}\right)$	$:= N\left(\boldsymbol{x}_{t-1};\boldsymbol{\mu}_\theta(\boldsymbol{x}_t,t),\tilde{\beta}_t\boldsymbol{I}\right)$
$\tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t,\boldsymbol{x}_0) = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t(\boldsymbol{x}_0) - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right),$ $\boldsymbol{\epsilon}_t \sim N(\boldsymbol{0},\boldsymbol{I})$	$\boldsymbol{\mu}_\theta(\boldsymbol{x}_t,t) := \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t,t)\right)$

Ho, Jain, and Abbeel, "Denoising diffusion probabilistic models," *NeurIPS*, 2020.

Training Loss for Denoising: Evidence Lower Bound (ELBO)

Negative Log-Likelihood (NLL)

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$$

Evidence lower bound (ELBO): Worst case

💡 $D_{KL}(p||q) := \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx$

$$L = \mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T))}_{L_T} + \sum_{t>1} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\right]$$

Ho, Jain, and Abbeel, "Denoising diffusion probabilistic models," *NeurIPS*, 2020.

Training Loss for Denoising: Evidence Lower Bound (ELBO)

Negative Log-Likelihood (NLL)

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$$

Evidence lower bound (ELBO): Worst case

💡 $D_{KL}(p||q) := \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx$

$$L = \mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T))}_{L_T} + \sum_{t>1} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\right]$$

not-trainable for fixed diffusion

Ho, Jain, and Abbeel, "Denoising diffusion probabilistic models," *NeurIPS*, 2020.

Training Loss for Denoising: Evidence Lower Bound (ELBO)

Negative Log-Likelihood (NLL)

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$$

Evidence lower bound (ELBO)

$D_{KL}(p||q) := \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx$

$$L = \mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T))}_{L_T} + \sum_{t>1} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\right]$$

For $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I})$ and $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \sigma_t^2 \mathbf{I})$,

$$L_{t-1} = \mathbb{E}_q\left[\frac{1}{2\sigma_t^2} \|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2\right] + C$$

Boiled down to a simple MSE for Gaussian distributions

Ho, Jain, and Abbeel, “Denoising diffusion probabilistic models,” NeurIPS, 2020.

Training Loss for Denoising: Evidence Lower Bound (ELBO)

Negative Log-Likelihood (NLL)

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$$

Evidence lower bound (ELBO)

$D_{KL}(p||q) := \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx$

$$L = \mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T))}_{L_T} + \sum_{t>1} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\right]$$

In practice, a simple loss stemmed from L_{t-1} is used for convenience.

$$\begin{aligned}
&\min_\theta E_q[L_{t-1}] \\
&= \min_\theta E_q[D_{KL}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)|p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t))] \\
&= \min_\theta E_q\left[\frac{1}{2\sigma_t^2} \parallel \tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) - \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t) \parallel^2\right] + C \\
&= \min_\theta E_{\boldsymbol{x}_0, \boldsymbol{\epsilon}}\left[\Sigma_{t>0} \frac{\beta_t}{2\sigma_t^2 \alpha_t(1-\bar{\alpha}_t)} \parallel \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) \parallel^2\right] \\
&= \min_\theta E_{\boldsymbol{x}_0, \boldsymbol{\epsilon}}\left[\Sigma_{t>0} \frac{\beta_t}{\sigma_t^2 \alpha_t(1-\bar{\alpha}_t)} \parallel \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t\right) \parallel^2\right]
\end{aligned}$$

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

Simplified by
- Uniformly distributed t instead of the sum
- Removing the scaling constant in front

Ho, Jain, and Abbeel, "Denoising diffusion probabilistic models," NeurIPS, 2020.

DDPM Training and Sampling Algorithms

Training Algorithm

repeat until converged

Step 1. Sample $\boldsymbol{x}_0 \sim q(x_0), t \sim Uniform\{1,2,\dots,T\}, \boldsymbol{\epsilon} \sim N(\mathbf{0}, \boldsymbol{I})$

Step 2. Diffuse for t times: $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$

Step 3. Take gradient descent step on

$$\nabla_\theta \| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) \|^2.$$

Sampling Algorithm

Step 1. Sample $\boldsymbol{x}_T \sim N(\mathbf{0}, \boldsymbol{I})$

Step 2. For $t = T, T-1, \dots, 1,$

$$\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)\right) + \sqrt{\tilde{\beta}_t}\boldsymbol{z}, \quad \boldsymbol{z} \sim N(\mathbf{0}, \boldsymbol{I})$$

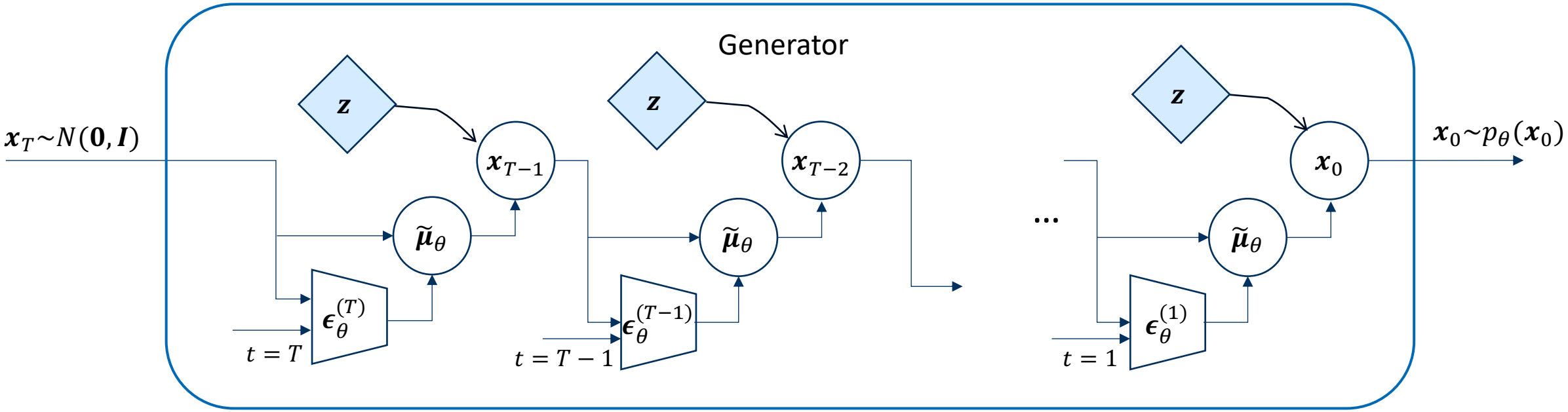
The output of the NN.

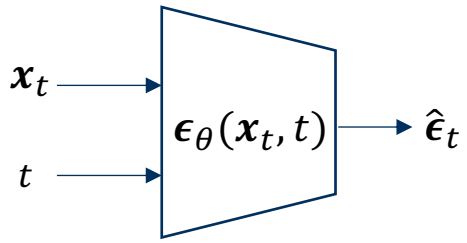
$\mu_\theta(x_t, t)$

Step 3. Return $\boldsymbol{x}_0$

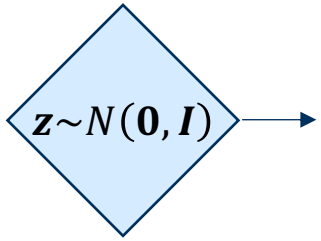
[10] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," Adv. Neural Inf. Process. Syst., vol. 33, pp. 6840–6851, Dec. 2020.

A Diagram of Sampling Process

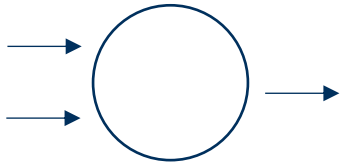




Neural network



Random sample

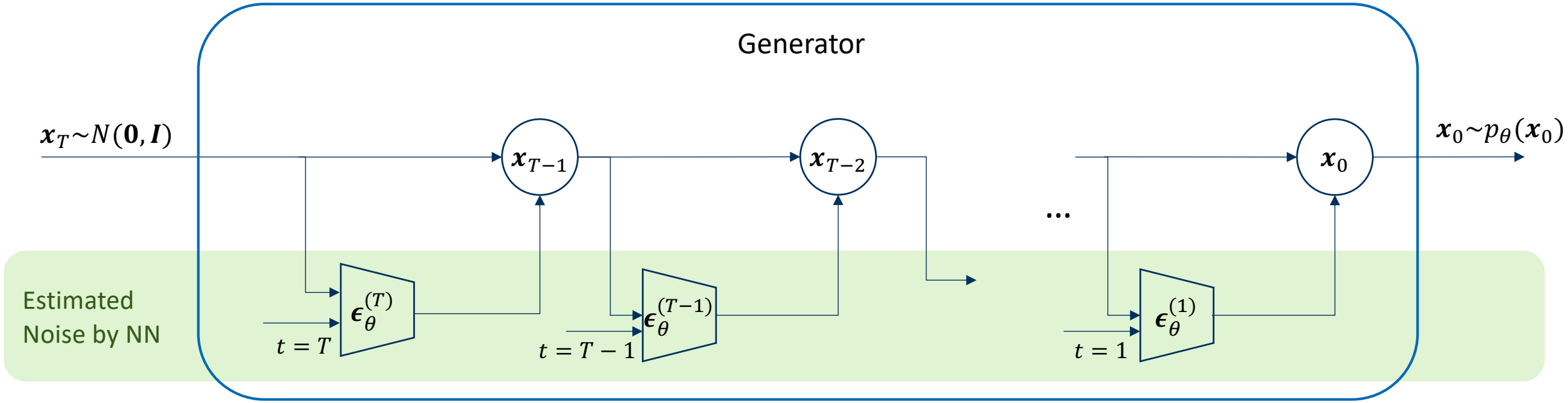


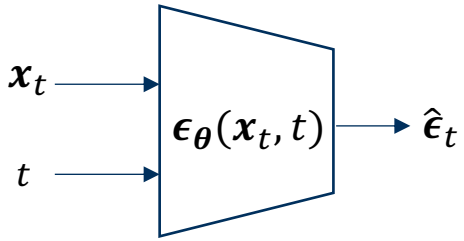
Equations

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

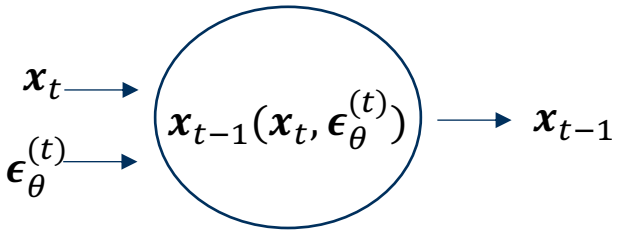
$$\boldsymbol{x}_{t-1}(\tilde{\boldsymbol{\mu}}_\theta) = \frac{1}{\sqrt{\alpha_t}} \tilde{\boldsymbol{\mu}}_\theta + \sqrt{\tilde{\beta}_t} \boldsymbol{z}$$

A Diagram of Sampling Process – Simplified



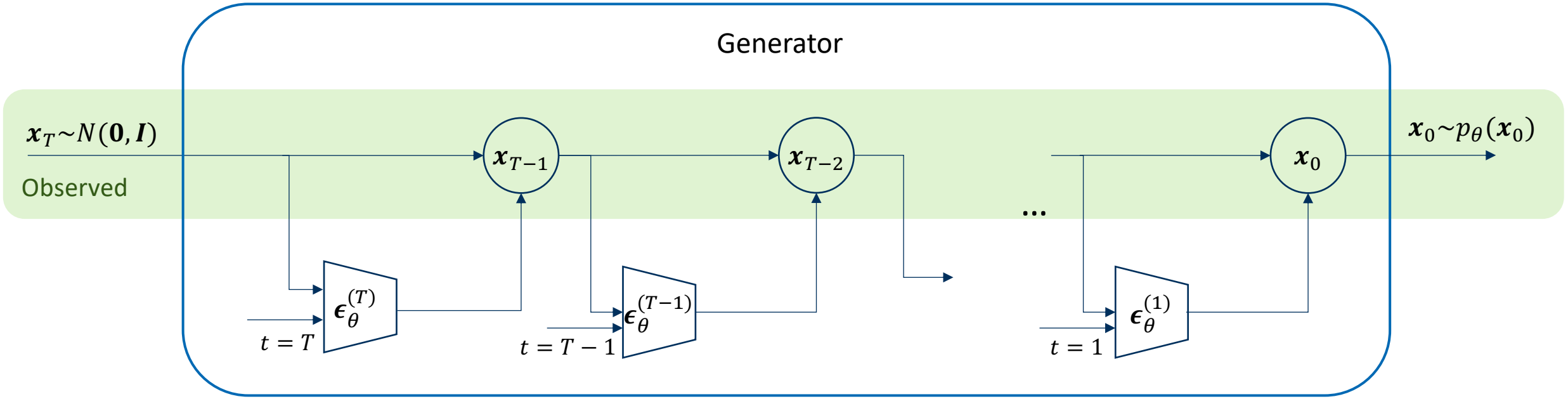


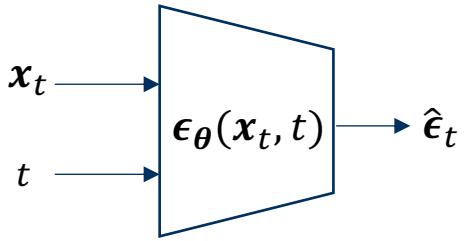
Neural network



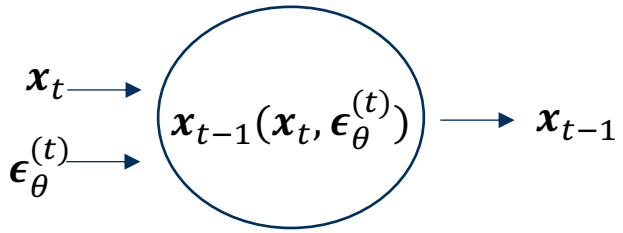
Denoising eq.

A Diagram of Sampling Process – Simplified





Neural network

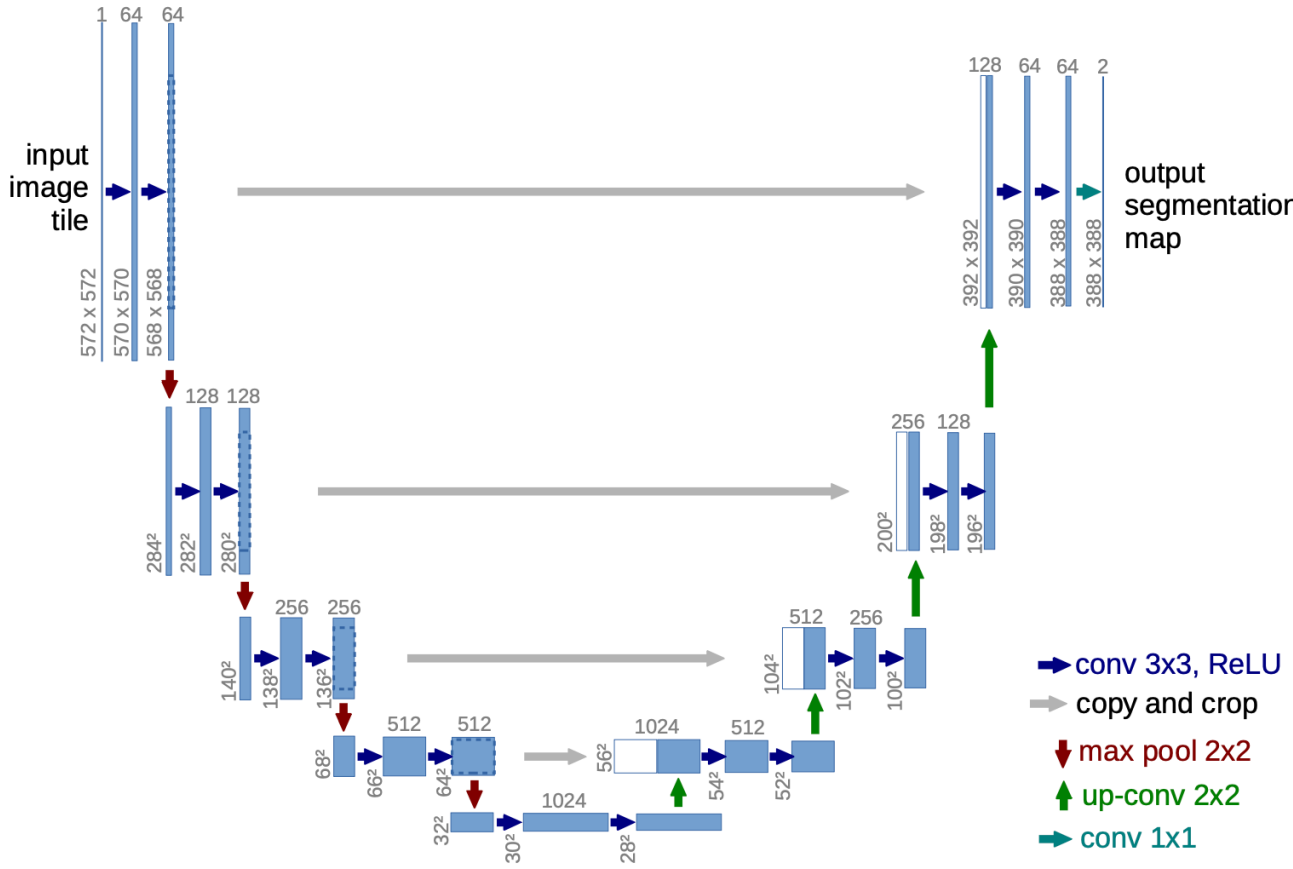


Denoising eq.

Jupyter Notebook Demo

Source code: https://github.com/Fritschek/MinDiffusion

Neural Network Architecture: U-Net Structure for Images



Originally suggested for image segmentation

Suitable for diffusion models

- Size-preserving
- Residual skipped connection

Slight modification with time embedding needed

Ronneberger, Fischer, Brox, U-Net: Convolutional Networks for Biomedical Image Segmentation, *MICCAI*, 2015

Part 2. Advanced Techniques of DDPM

Advanced Variants of DDPM

1. Fast sampling

- Learning free: Non-Markovian model and diffusion denoising implicit modes (DDIMs)
- Learning based:
  - Optimized Discretization
  - Truncated Diffusion
  - Diffusion Distillation

2. Improving sample quality and mode coverage

- Learning free: noise schedule optimization, parametrization techniques.
- Learning-based: Variational diffusion models.

Yang et al, "Diffusion Models: A Comprehensive Survey of Methods and Applications," *ACM Computing Surveys*, 2023

Advanced Variants of DDPM

1. Fast sampling

- Learning free: **Non-Markovian model and diffusion denoising implicit modes (DDIMs)**
- Learning based:
  - Optimized Discretization
  - Truncated Diffusion
  - Diffusion Distillation

2. Improving sample quality and mode coverage

- Learning free: **noise schedule optimization**, **parametrization techniques**.
- Learning-based: Variational diffusion models.

Yang et al, "Diffusion Models: A Comprehensive Survey of Methods and Applications," *ACM Computing Surveys*, 2023

Advanced Variants of DDPM – Learning-free Methods

1. Non-Markovian model – DDIM and Skipped Sampling

   Song, Meng, and Ermon, “Diffusion Denoising Implicit Model”, *ICLR* 2021

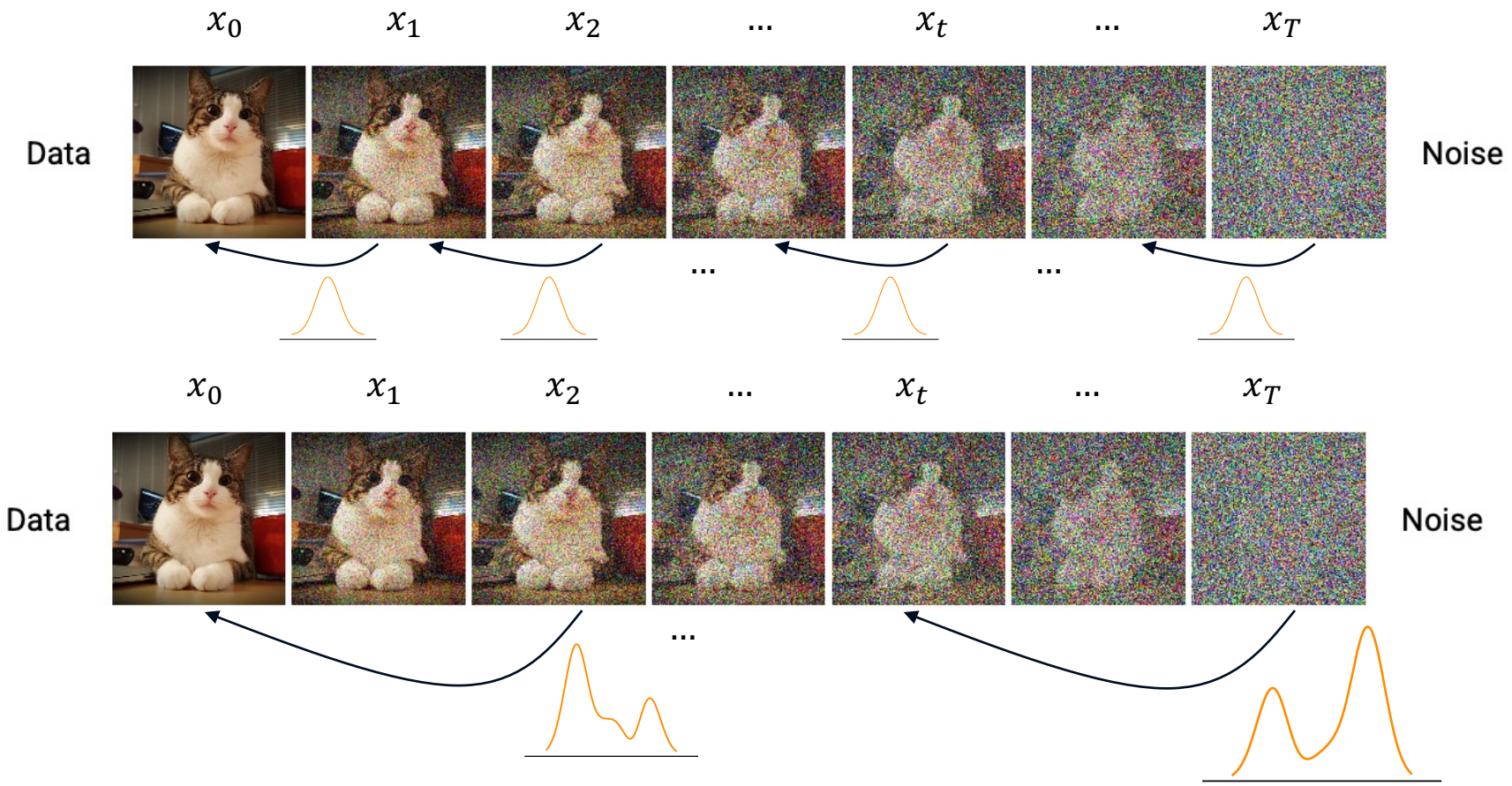
2. Cosine Noise Scheduling – Improving Likelihood of DDPM

   Nichol and Dhariwal, “Improved Denoising Diffusion Probabilistic Models”, *ICML* 2021

3. Perfect Diffusion and Stable Parametrization – Robustness to Skipped Sampling

   Salimans and Ho, “Progressive Distillation for Fast Sampling of Diffusion Models”, *ICLR* 2022

1. Fast sampling: skipped sampling and trade-off



Reducing the number of iterations

$\Rightarrow$ requiring high-functioning noise approximation

1. Fast Sampling: Non-Markovian Inference Process

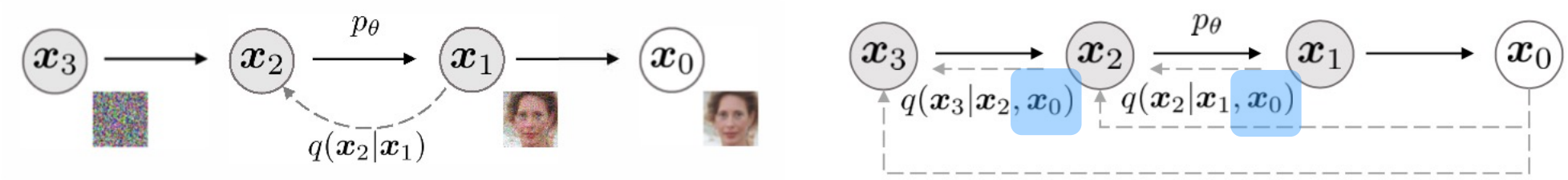
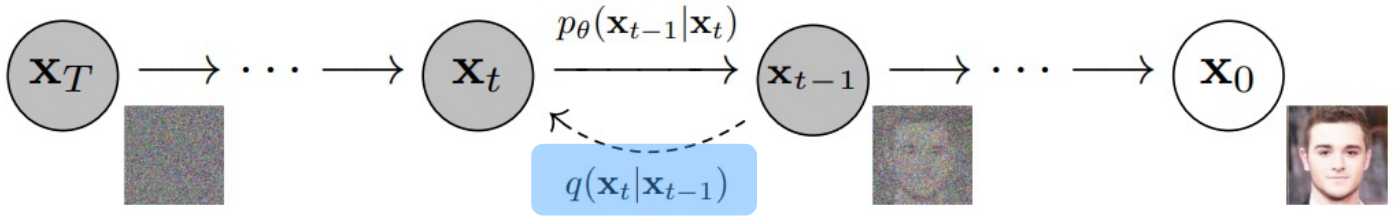


Figure 1: Graphical models for diffusion (left) and non-Markovian (right) inference models.



Cf.) Markovian inference model of DDPM.

$\Rightarrow$ How to denoise depends on the sample that it's generating.

$\Rightarrow$ Generalized version of DDPM.

Song, Meng, and Ermon, "Diffusion Denoising Implicit Model", *ICLR* 2021

1. Fast Sampling: Non-Markovian Inference Process

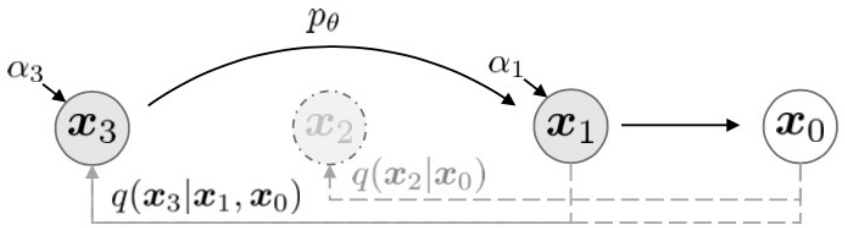


Figure 2: Graphical model for accelerated generation, where $\tau = [1, 3]$.

Why non-Markovian for fast sampling??

A special case of the non-Markovian process is robust to skipped sampling.
That is diffusion denoising implicit models (DDIMs) using deterministic denoising without randomness.

Song, Meng, and Ermon, "Diffusion Denoising Implicit Model", *ICLR* 2021

1. Fast Sampling: Generalization of DDPM

Recall) ELBO of NLL in DDPM

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$$

Some math and simplification

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

The simplified loss function only depends on the marginal $q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_0)$ instead of the joint $q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)$ such that

$$q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\sqrt{\alpha_t}\boldsymbol{x}_0, (1-\alpha_t)\boldsymbol{I}) .$$

$\Rightarrow$ The set of $q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0)$ having the same marginal $q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_0)$ can be also considered, in addition to the Markovian case with $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$.

$\Rightarrow$ Generalization to non-Markovian inference process.

Song, Meng, and Ermon, "Diffusion Denoising Implicit Model", *ICLR* 2021

⚠ α_t in DDIM paper is equivalent to $\bar{\alpha}_t$ in DDPM paper.

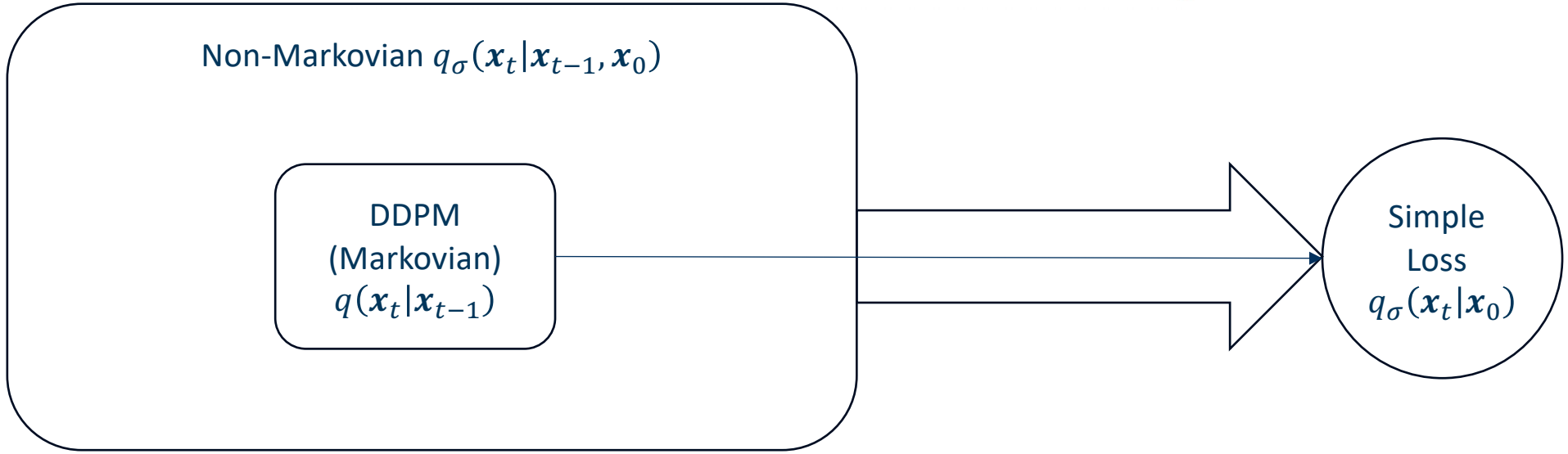
1. Fast Sampling: Generalization of DDPM

Recall) ELBO of NLL in DDPM

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$$

Some math and simplification

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$



Song, Meng, and Ermon, "Diffusion Denoising Implicit Model", *ICLR* 2021

⚠ α_t in DDIM paper is equivalent to $\bar{\alpha}_t$ in DDPM paper.

1. Fast Sampling: Generalization of DDPM

For non-Markovian process, the diffusion is defined as

$$q_\sigma(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) := q_\sigma(\boldsymbol{x}_T|\boldsymbol{x}_0)\prod_{t=2}^{T} q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0).$$

A set of $q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ yielding the marginal distribution, $q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\sqrt{\alpha_t}\boldsymbol{x}_0, (1-\alpha_t)\boldsymbol{I})$,
is defined by a stochasticity parameter σ_t^2:

Generalized diffusion model.

$$q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}\left(\sqrt{\alpha_{t-1}}\boldsymbol{x}_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\frac{\boldsymbol{x}_t - \sqrt{\alpha_t}\boldsymbol{x}_0}{\sqrt{1-\alpha_t}}, \sigma_t^2\boldsymbol{I}\right).$$

Q. For this new definition, how can we derive the denoising equation for sampling?

Song, Meng, and Ermon, "Diffusion Denoising Implicit Model", *ICLR* 2021

⚠ α_t in DDIM paper is equivalent to $\bar{\alpha}_t$ in DDPM paper.

1. Fast Sampling: Sampling for the non-Markovian Process

$$q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}\left(\sqrt{\alpha_{t-1}}\boldsymbol{x}_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\frac{\boldsymbol{x}_t - \sqrt{\alpha_t}\boldsymbol{x}_0}{\sqrt{1-\alpha_t}}, \sigma_t^2\boldsymbol{I}\right)$$

During sampling, $\boldsymbol{x}_0$ is not available.

$\Rightarrow$ Use predicted $\boldsymbol{x}_0$ corresponding to $\boldsymbol{\epsilon}_\theta^{(t)}$, which satisfies, for given $\boldsymbol{x}_t$,

$$q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\sqrt{\alpha_t}\boldsymbol{x}_0, (1-\alpha_t)\boldsymbol{I}) \text{ , i.e., } \quad \boldsymbol{x}_t = \sqrt{\alpha_t}\boldsymbol{x}_0 + \sqrt{1-\alpha_t}\epsilon, \quad \text{where} \quad \epsilon \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}).$$

That leads to the denoising step equation

$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}}\underbrace{\left(\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(\boldsymbol{x}_t)}{\sqrt{\alpha_t}}\right)}_{\text{“ predicted } \boldsymbol{x}_0\text{”}} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\epsilon_\theta^{(t)}(\boldsymbol{x}_t)}_{\text{“direction pointing to } \boldsymbol{x}_t\text{”}} + \underbrace{\sigma_t\epsilon_t}_{\text{random noise}}$$

Song, Meng, and Ermon, “Diffusion Denoising Implicit Model”, *ICLR* 2021

⚠ α_t in DDIM paper is equivalent to $\bar{\alpha}_t$ in DDPM paper.

1. Fast Sampling: DDIM - deterministic denoising

Denoising distribution of the non-Markovian process

$$q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}\left(\sqrt{\alpha_{t-1}}\boldsymbol{x}_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\frac{\boldsymbol{x}_t - \sqrt{\alpha_t}\boldsymbol{x}_0}{\sqrt{1-\alpha_t}}, \sigma_t^2\boldsymbol{I}\right).$$

Special cases

1. DDPM (satisfying Markovian process) when

$$\sigma_t = \sqrt{\frac{1-\alpha_{t-1}}{1-\alpha_t}\cdot\left(1-\frac{\alpha_t}{\alpha_{t-1}}\right)}.$$

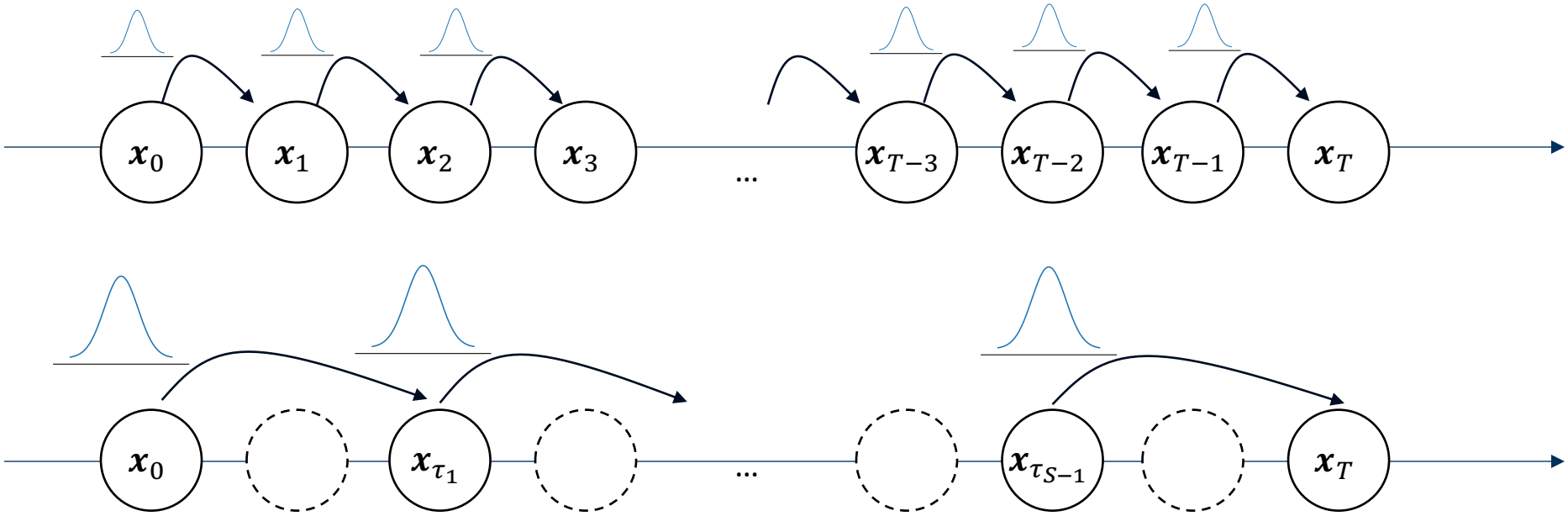
2. Diffusion Denoising Implicit Model (DDIM) when $\sigma_t = 0$. $\rightarrow$ deterministic denoising

DDIM is robust to skipped sampling.

Song, Meng, and Ermon, "Diffusion Denoising Implicit Model", *ICLR* 2021

1. Fast Sampling: Skipped Sampling

The diffusion process

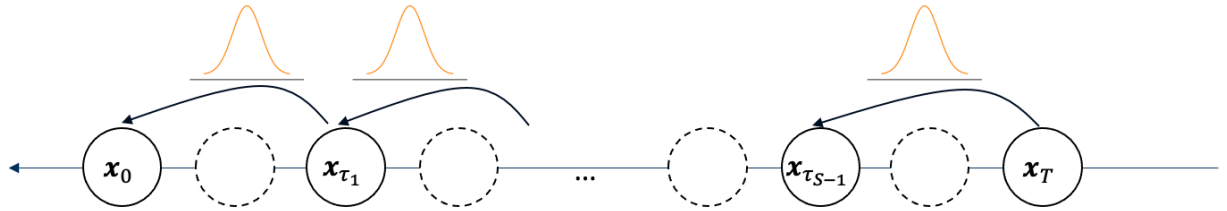


Imagine a skipped diffusion process whose marginal is equal to the diffusion process.

A trajectory $\boldsymbol{\tau} = (\tau_1, \tau_2, \dots, \tau_S)$ is a length S sub-sequence of $(1,2, \dots, T)$, where $\tau_S = T$.
For τ_i, it holds $q(\boldsymbol{x}_{\tau_i}|\boldsymbol{x}_0) = N(\sqrt{\alpha_{\tau_i}}\boldsymbol{x}_0, (1-\alpha_{\tau_i})\boldsymbol{I})$.

Song, Meng, and Ermon, "Diffusion Denoising Implicit Model", *ICLR* 2021

1. Fast Sampling: Skipped Sampling



Sampling as if the skipped sequence is the full sequence by using

$$\boldsymbol{x}_{\tau_{i-1}}(\eta) = \sqrt{\alpha_{\tau_{i-1}}}\left(\frac{\boldsymbol{x}_{\tau_i} - \sqrt{1-\alpha_{\tau_i}}\epsilon_\theta^{(\tau_i)}(\boldsymbol{x}_{\tau_i})}{\sqrt{\alpha_{\tau_i}}}\right) + \sqrt{1-\alpha_{\tau_{i-1}}-\sigma_{\tau_i}(\eta)^2}\cdot\epsilon_\theta^{(\tau_i)}(\boldsymbol{x}_{\tau_i}) + \sigma_{\tau_i}(\eta)\epsilon.$$

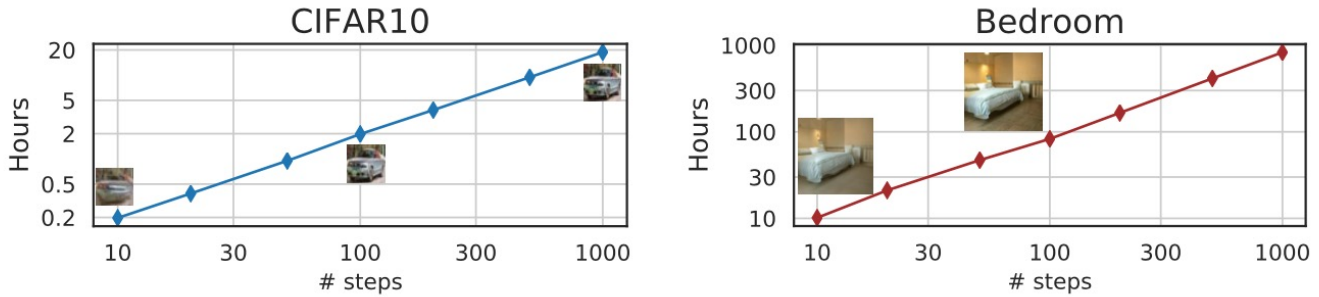


Figure 4: Hours to sample 50k images with one Nvidia 2080 Ti GPU and samples at different steps.

Song, Meng, and Ermon, "Diffusion Denoising Implicit Model", *ICLR* 2021

1. Fast Sampling: DDIM's Robustness to Skipped Sampling

Table 1: CIFAR10 and CelebA image generation measured in FID. $\eta = 1.0$ and $\hat{\sigma}$ are cases of DDPM (although Ho et al. (2020) only considered $T = 1000$ steps, and $S < T$ can be seen as simulating DDPMs trained with S steps), and $\eta = 0.0$ indicates DDIM.

		CIFAR10 (32×32)					CelebA (64×64)				
	S	10	20	50	100	1000	10	20	50	100	1000
η	0.0	**13.36**	**6.84**	**4.67**	**4.16**	4.04	**17.33**	**13.73**	**9.17**	**6.53**	3.51
	0.2	14.04	7.11	4.77	4.25	4.09	17.66	14.11	9.51	6.79	3.64
	0.5	16.66	8.35	5.25	4.46	4.29	19.86	16.06	11.01	8.09	4.28
	1.0	41.07	18.36	8.01	5.78	4.73	33.12	26.03	18.48	13.93	5.98
$\hat{\sigma}$		367.43	133.37	32.72	9.99	**3.17**	299.71	183.83	71.71	45.20	**3.26**

$\eta = \frac{\sigma_t}{\sqrt{\frac{1-\alpha_{t-1}}{1-\alpha_t}\left(1-\frac{\alpha_t}{\alpha_{t-1}}\right)}}$ is 0 for DDIM and 1 for DDPM. And, $\hat{\sigma}$ is a case where $\eta > 1$, and $\sigma_t = \sqrt{1 - \alpha_{\tau_i}/\alpha_{\tau_{i-1}}}$.

$\Rightarrow$ The sample quality is the best for DDPM without skipped sampling.

$\Rightarrow$ DDIM is much more robust to skipped sampling, where as DDPM breaks down dramatically.

$\Rightarrow$ As the training loss is defined the same regardless of η, DDIM can be used as a robust skipped sampling method.

Song, Meng, and Ermon, "Diffusion Denoising Implicit Model", *ICLR* 2021

1. Fast Sampling: DDIM's Robustness to Skipped Sampling

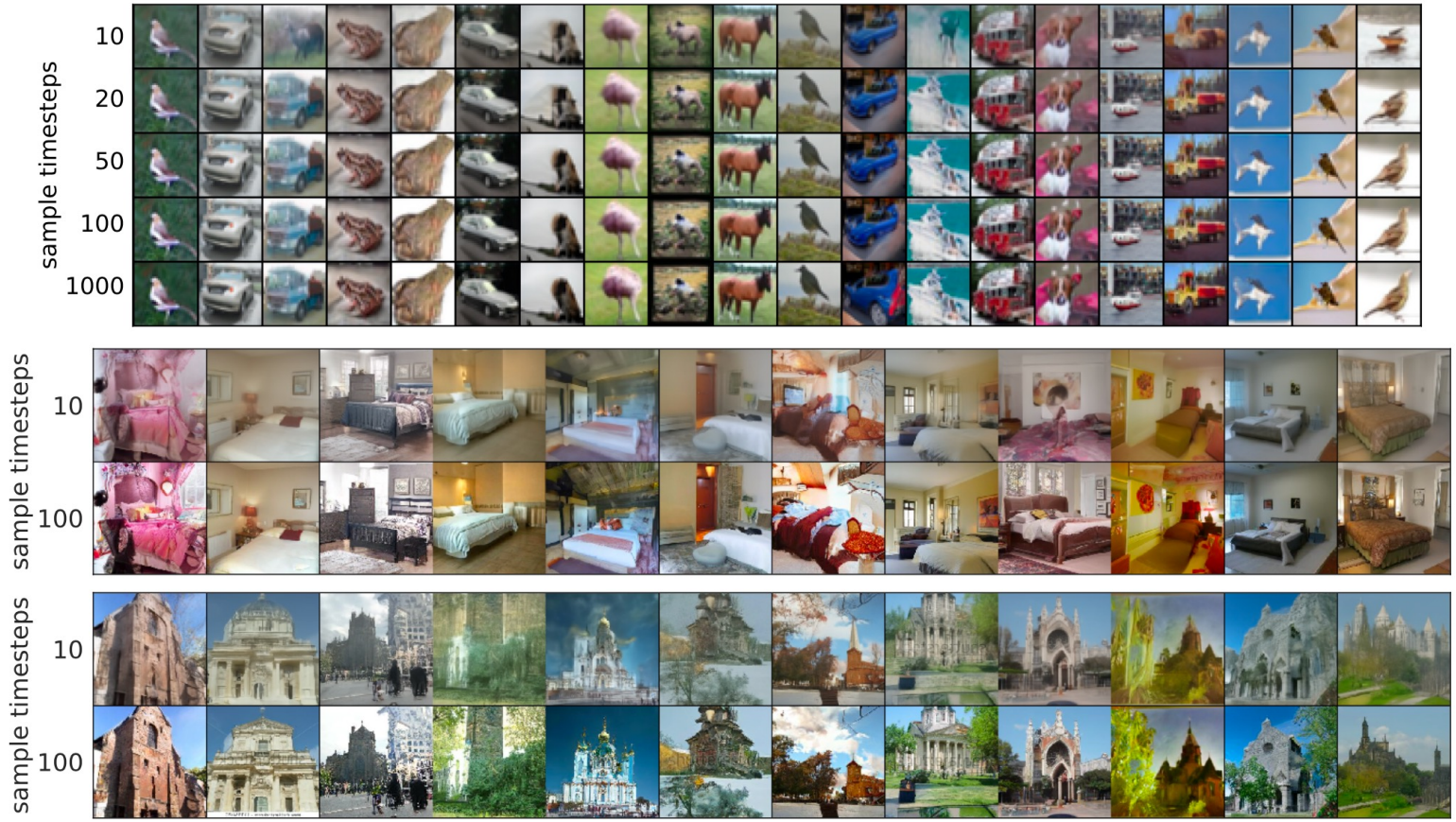


Figure 5: Samples from DDIM with the same random x_T and different number of steps.

Song, Meng, and Ermon, "Diffusion Denoising Implicit Model", *ICLR* 2021

Miscellaneous) DDIM's Ability to Interpolate

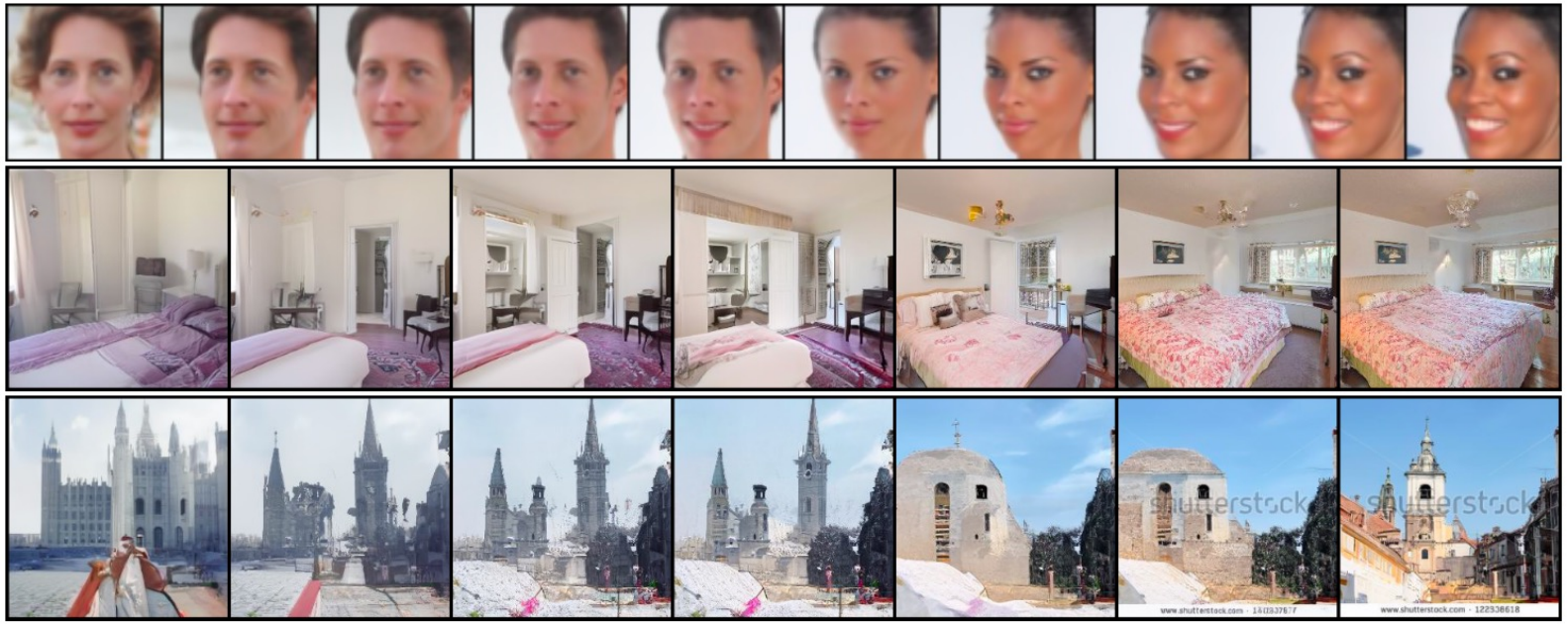


Figure 6: Interpolation of samples from DDIM with $\dim(\tau) = 50$.

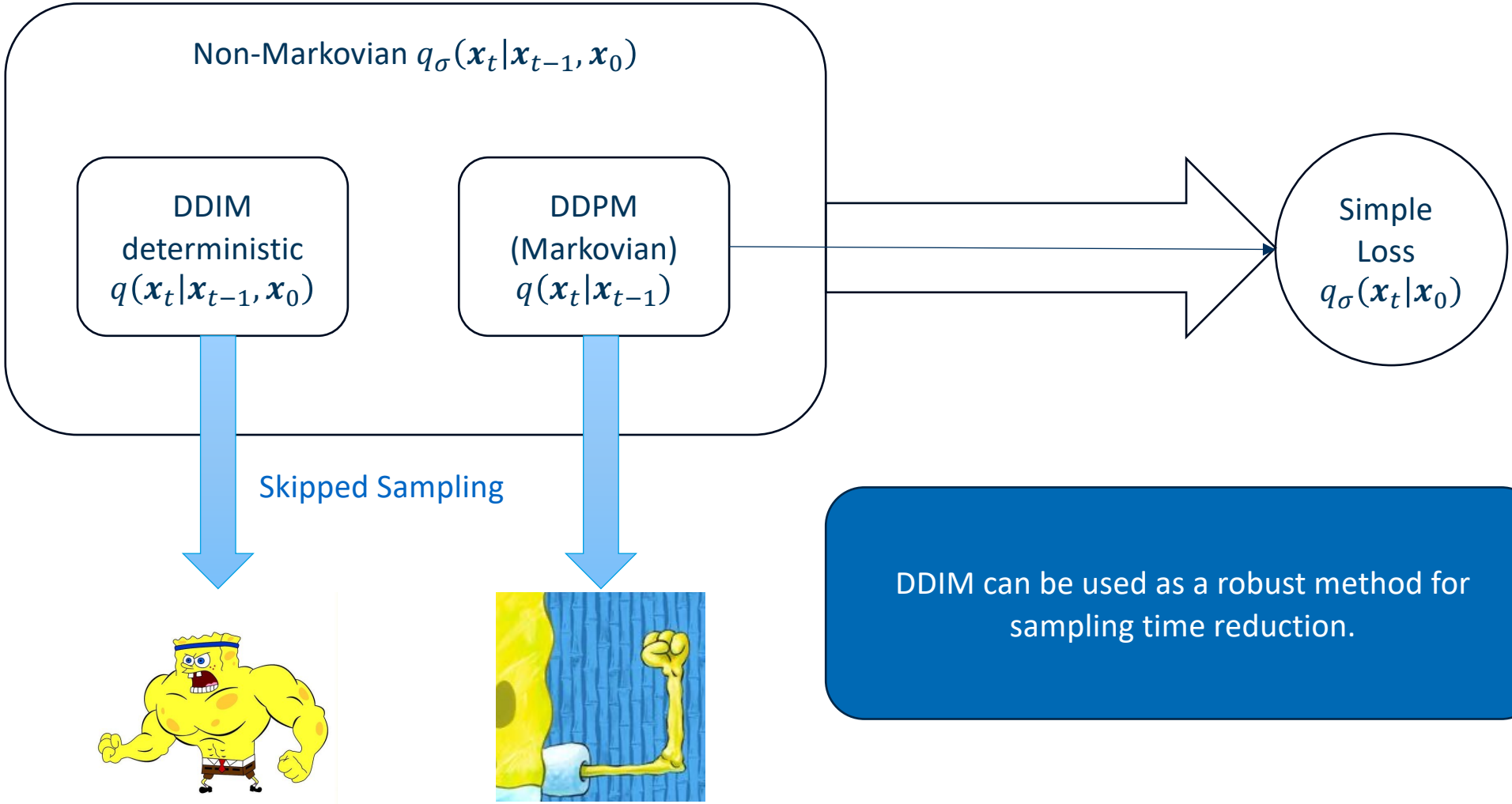
Another advantages of DDIM

- Easy to control the generated sample due to the deterministic sampling.
- Nice interpolation property.

$\Rightarrow$ In case your application needs such properties….

Song, Meng, and Ermon, "Diffusion Denoising Implicit Model", *ICLR* 2021

1. Fast Sampling by DDIM Summary



DDIM can be used as a robust method for sampling time reduction.

Jupyter Notebook Demo

Source code: https://github.com/Fritschek/MinDiffusion

2. Need for improving log-likelihood of DDPMs

- Why log-likelihood matters?

Mode coverage, sample quality and learnt feature representations, etc.

Razavi, van den Oord, and Vinyals, “Generating diverse high-fidelity images with VQ-VAE-2”, *NeurIPS* 2019.
Henighan et al., “Scaling laws for autoregressive generative modeling”, *arXiv* 2020.

- Room for improvement?

As the ELBO is only the lower bound of the NLL, likelihood itself can be improved further.

- How to improve?

Learning free: noise scheduling, non-uniform time index distribution, parametrization techniques.
Learning-based: optimization of the variance of the inferred noise, noise scheduling, weight in loss.

Nichol and Dhariwal, “Improved Denoising Diffusion Probabilistic Models”, *ICML* 2021
Kingma, Salimans, Poole, and Ho, “Variational Diffusion Models”, *NeurIPS* 2021

2. Need for improving log-likelihood of DDPMs

- Why log-likelihood matters?

Mode coverage, sample quality and learnt feature representations, etc.

Razavi, van den Oord, and Vinyals, “Generating diverse high-fidelity images with VQ-VAE-2”, *NeurIPS* 2019.
Henighan et al., “Scaling laws for autoregressive generative modeling”, *arXiv* 2020.

- Room for improvement?

As the ELBO is only the lower bound of the NLL, likelihood itself can be improved further.

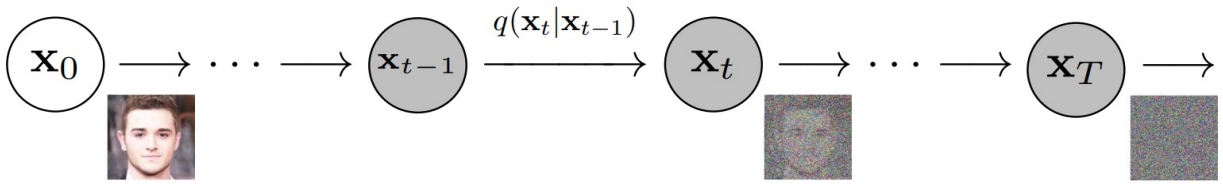
- How to improve?

Learning free: **noise scheduling**, non-uniform time index distribution, **parametrization techniques**.
Learning-based: optimization of the variance of the inferred noise, noise scheduling, weight in loss.

Nichol and Dhariwal, “Improved Denoising Diffusion Probabilistic Models”, *ICML* 2021
Kingma, Salimans, Poole, and Ho, “Variational Diffusion Models”, *NeurIPS* 2021

2. Improving Likelihood - Noise Scheduling Methods

Recall: Diffusion Process



$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = N(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}) \qquad 0 < \beta_t \ll 1$$

$$\begin{aligned}
\boldsymbol{x}_t &= \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}, && \boldsymbol{\epsilon} \sim N(\boldsymbol{0}, \boldsymbol{I}) \\
&= \sqrt{\alpha_t}\boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}, && \alpha_t = 1 - \beta_t \\
&= \text{... (solving recursion)} \\
&= \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, && \bar{\alpha}_t = \Pi_{i=1}^{t}\alpha_i
\end{aligned}$$

How much noise do you add at each step?

=> How to define $\{\beta_t\}_1^T$?

Ho, Jain, and Abbeel, "Denoising diffusion probabilistic models," *NeurIPS*, 2020.

[15] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in Proc. Adv. Neural Inf. Process. Syst., vol. 33, pp. 6840–6851, Dec. 2020.

2. Improving Likelihood - Noise Scheduling Formulas

1. Increasing β_t from β_1 to β_T
   1. Linearly: $\beta_t = \beta_1 + (t-1)\frac{\beta_T - \beta_1}{T-1}$.
   2. Quadratically: $\beta_t = \left(\sqrt{\beta_1} + \frac{t-1}{T-1}\left(\sqrt{\beta_T} - \sqrt{\beta_1}\right)\right)^2$
   3. By a sigmoid function: $\beta_t = \beta_1 + \frac{1}{1+\exp(\tau(t))}(\beta_T - \beta_1),\ where\ \tau(t) = -6 + (t-1)\frac{12}{T-1}$.
2. Cosine scheduled from a small number to 1. (improve log-likelihood)

$$\beta_t = 1 - \frac{f(t)}{f(t-1)}, \qquad where\ f(t) = \left(\cos\left(\frac{\pi}{2}\frac{t-1}{T-1} + 0.008\right)\right)^2.$$

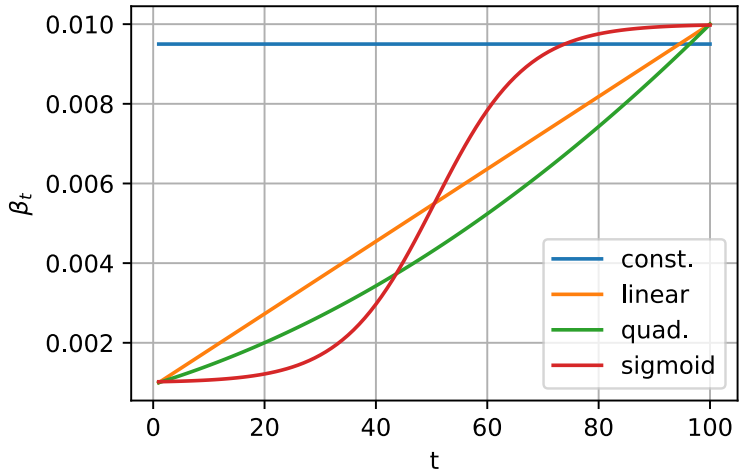
Nichol and Dhariwal, Improved Denoising Diffusion Probabilistic Models, ICML 2021

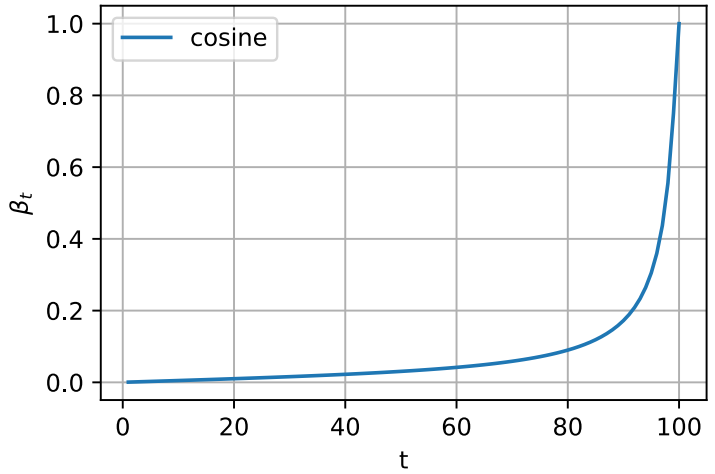
3. Learning it

$$\sqrt{1 - \prod_{i=1}^{t}(1-\beta_i)} = \text{sigmoid}\left(\gamma_{\boldsymbol{\eta}}(t)\right),$$

where $\gamma_{\boldsymbol{\eta}}(t)$ is optimized by learning

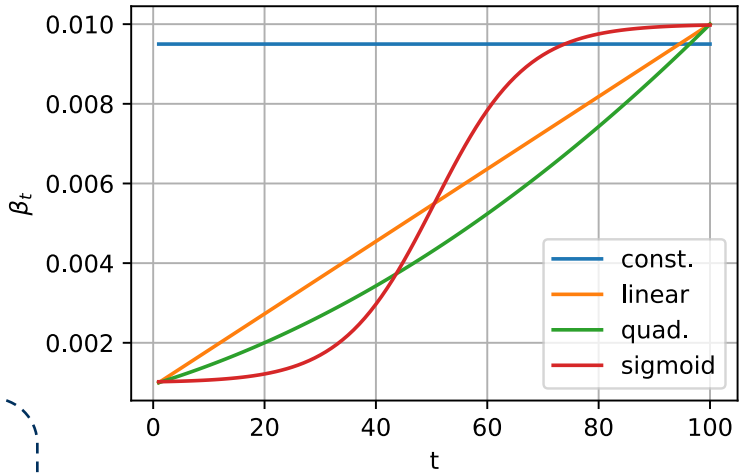
Kingma, Salimans, Poole, and Ho, Variational Diffusion Models, *NeurIPS* 2021





2. Improving Likelihood - Noise Scheduling Formulas

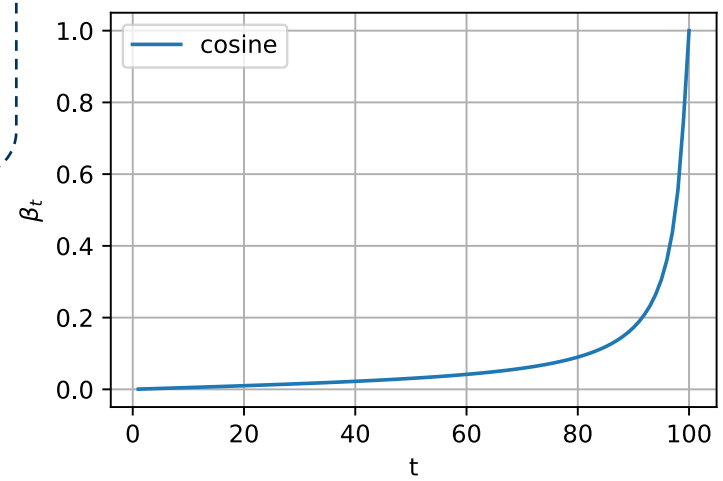
1. Increasing β_t from β_1 to β_T
   1. Linearly: $\beta_t = \beta_1 + (t-1)\frac{\beta_T - \beta_1}{T-1}$.
   2. Quadratically: $\beta_t = \left(\sqrt{\beta_1} + \frac{t-1}{T-1}\left(\sqrt{\beta_T} - \sqrt{\beta_1}\right)\right)^2$
   3. By a sigmoid function: $\beta_t = \beta_1 + \frac{1}{1+\exp(\tau(t))}(\beta_T - \beta_1),\ where\ \tau(t) = -6 + (t-1)\frac{12}{T-1}$.



2. Cosine scheduled from a small number to 1. (improve log-likelihood)

$$\beta_t = 1 - \frac{f(t)}{f(t-1)}, \qquad where\ f(t) = \left(\cos\left(\frac{\pi}{2}\frac{t-1}{T-1} + 0.008\right)\right)^2.$$

Immediate improvement without increasing complexity!



3. Learning it in a parametrized way

$$\sqrt{1 - \prod_{i=1}^{t}(1-\beta_i)} = \text{sigmoid}\left(\gamma_{\boldsymbol{\eta}}(t)\right),$$

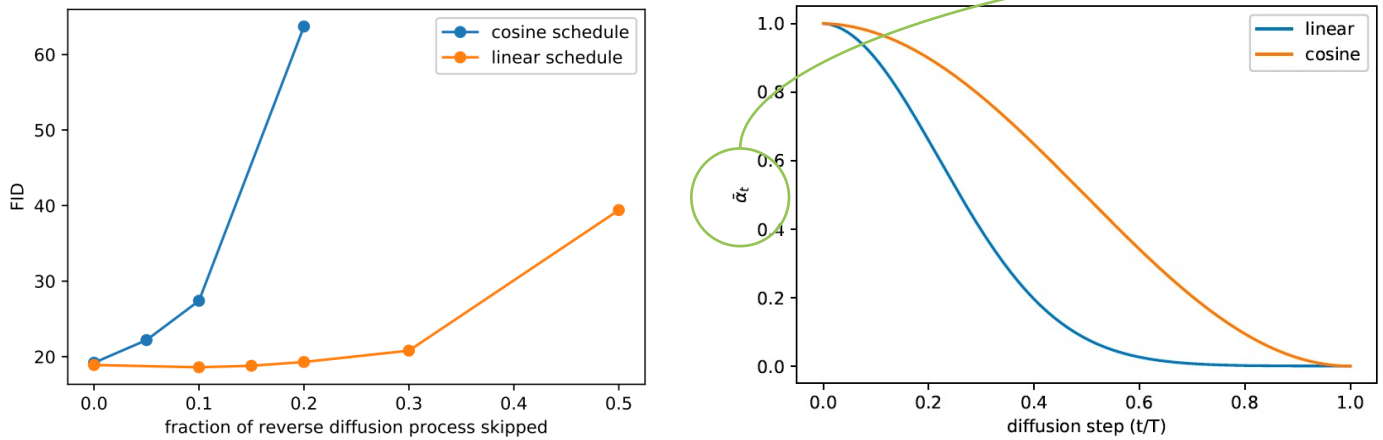
where $\gamma_{\boldsymbol{\eta}}(t)$ is the part learned by deep learning.

Kingma, Salimans, Poole, and Ho, Variational Diffusion Models, *NeurIPS* 2021

2. Improving Likelihood - Why cosine scheduled noise?

Figure 3. Latent samples from linear (top) and cosine (bottom)

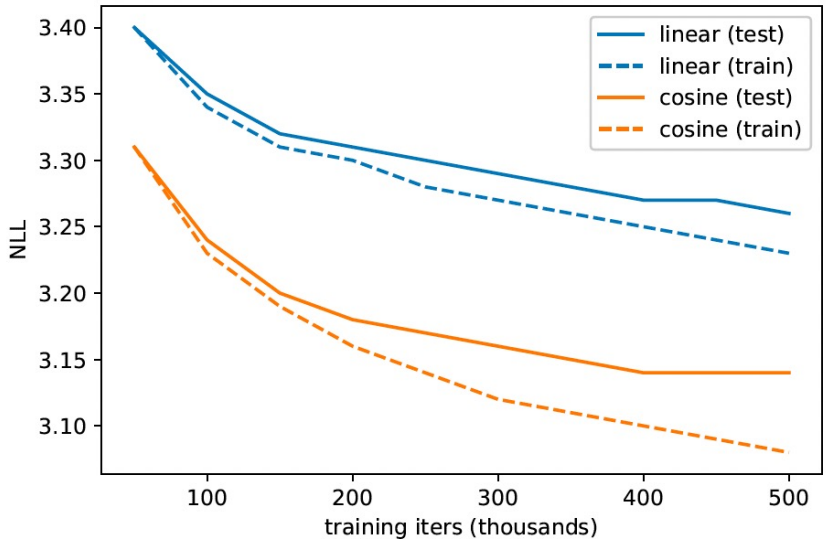
$\bar{\alpha}_t$ is the scaling factor s.t.
$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$



Relatively gradual diffusion by cosine scheduling.

Nichol and Dhariwal, Improved Denoising Diffusion Probabilistic Models, *ICML* 2021

2. Improving Likelihood - Why cosine scheduled noise?



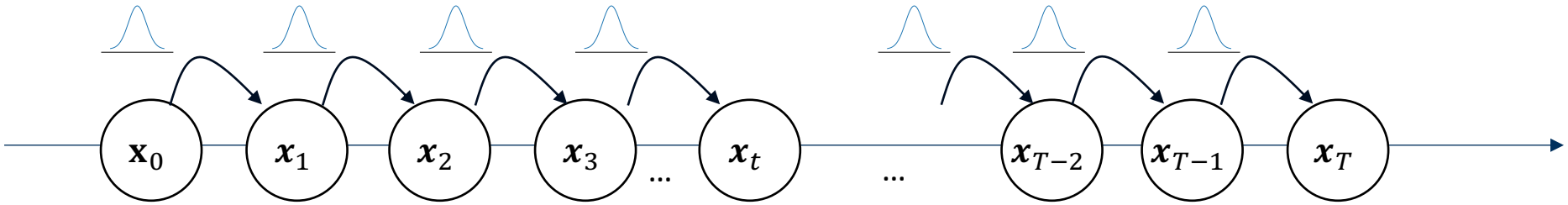
Tested with CIFAR10, cosine scheduling offered better log-likelihood, which is related to the mode coverage.

A takeaway: cosine scheduling could be useful if an improvement of mode coverage is desired.

Nichol and Dhariwal, Improved Denoising Diffusion Probabilistic Models, *ICML* 2021

3. Improving Sample Quality – SNR-based Notation

Generalization of the model with new notations



Scaling factors of the data α_t and noise variance σ_t are independent.

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = N(\alpha_t \boldsymbol{x}_0, \sigma_t^2 \boldsymbol{I})$$

Signal-to-noise ratio (SNR) is defined as $\mathrm{SNR}(t) \coloneqq \frac{\alpha_t^2}{\sigma_t^2}$ and assumed to be strictly monotonically decreasing:

$\mathrm{SNR}(s) > \mathrm{SNR}(t)$ for $s > t$. "variance-preserving diffusion process".

cf) The conventional model is a special case with $\alpha_t = \sqrt{1 - \sigma_t^2}$.

Kingma, Salimans, Poole, and Ho, "Variational Diffusion Models", *NeurIPS* 2021

3. Improving Sample Quality – Nontrivial Discrepancy

Schedule	Definition ($i = \frac{t-1}{T-1}$)	SNR(T)	$\sqrt{\bar{\alpha}_T}$
Linear [3]	$\beta_t = 0.0001 \cdot (1-i) + 0.02 \cdot i$	4.035993e-05	0.006353
Cosine [8]	$\beta_t = \min(1-\frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}, 0.999), \bar{\alpha}_t = \frac{f(t)}{f(0)}, f(t) = \cos(\frac{i+0.008}{1+0.008} \cdot \frac{\pi}{2})^2$	2.428735e-09	4.928220e-05
Stable Diffusion [10]	$\beta_t = (\sqrt{0.00085} \cdot (1-i) + \sqrt{0.012} \cdot i)^2$	0.004682	0.068265

Table 1. Common schedule definitions and their SNR and $\sqrt{\bar{\alpha}}$ on the last timestep. All schedules use total timestep $T = 1000$. None of the schedules has zero SNR on the last timestep $t = T$, causing inconsistency in train/inference behavior.

Conventionally, $\alpha_T \cong 0$ and $\sigma_T \cong 1$ were used in implementations.
This discrepancy can lead to training and inference discrepancy.

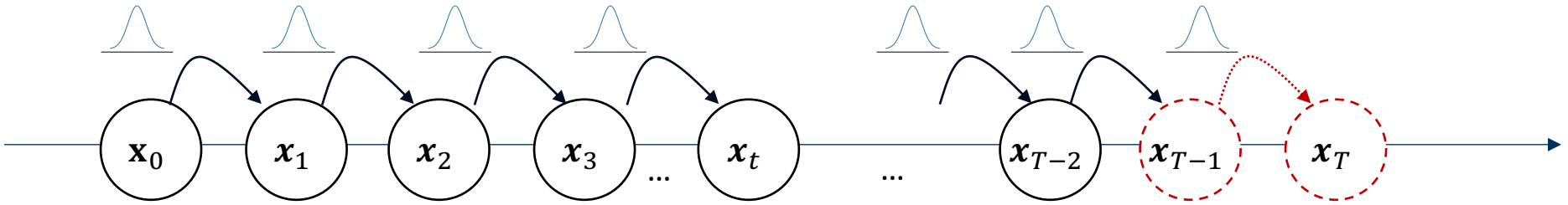
Ex.) The mean value of the diffused sample is 0.068 for
Stable Diffusion, and over-exposure problem is observed.

$\Rightarrow$ The discrepancy is not negligible. Perfect diffusion is needed.

(a) Flawed (b) Corrected

Lin, Liu, Li, and Yang, "Common Diffusion Noise Scheduling and Sample Steps Flawed", *WACV*, 2024
Rombach et al., "High-Resolution Image Synthesis with Latent Diffusion Models", *CVPR*, 2022.

3. Improving Sample Quality – 0 Terminal SNR and Broken Chain

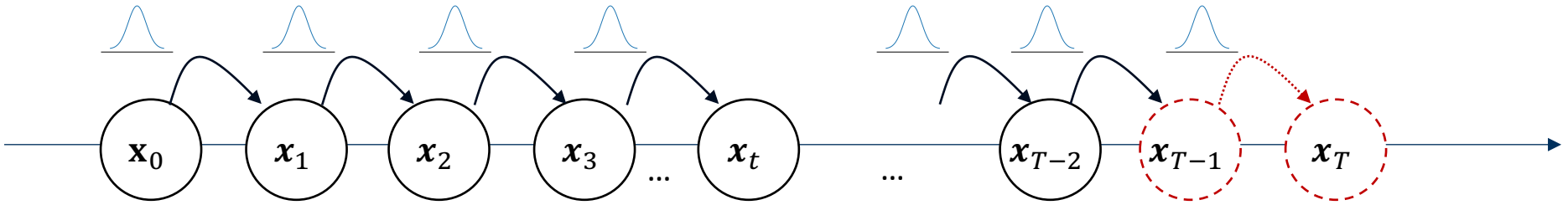


For perfect diffusion, it is assumed that $\alpha_T = 0, \sigma_T = 1$, and accordingly $\mathrm{SNR}(T) = 0$.

$$\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon} \quad \Rightarrow \quad \boldsymbol{x}_T = 0 \cdot \boldsymbol{x}_0 + 1 \cdot \boldsymbol{\epsilon} = \boldsymbol{\epsilon}$$

Salimans and Ho, "Progressive Distillation for Fast Sampling of Diffusion Models", *ICLR* 2022.

3. Improving Sample Quality – 0 Terminal SNR and Broken Chain



For perfect diffusion, it is assumed that $\alpha_T = 0, \sigma_T = 1$, and accordingly $\mathrm{SNR}(T) = 0$.

$$\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon} \quad \Rightarrow \quad \boldsymbol{x}_T = 0 \cdot \boldsymbol{x}_0 + 1 \cdot \boldsymbol{\epsilon} = \boldsymbol{\epsilon}$$

A problem: $\boldsymbol{x}_T$ is no longer informative of $\boldsymbol{x}_0$ at all. Connection lost!

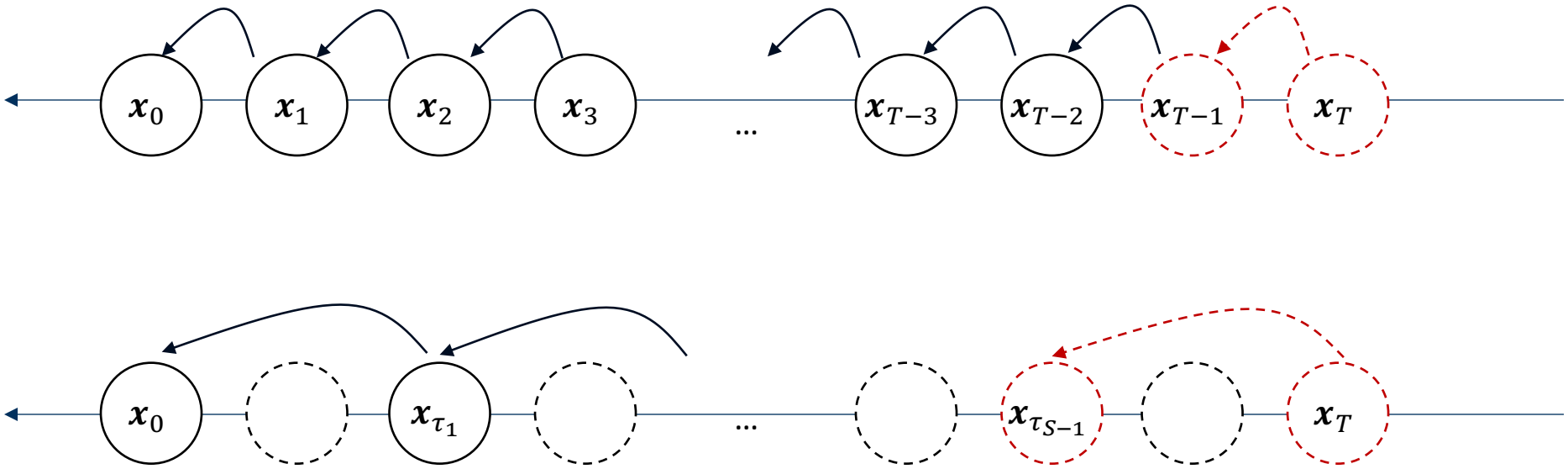
$$q(\boldsymbol{x}_T|\boldsymbol{x}_{T-1}) = q(\boldsymbol{x}_T) = N(0, \boldsymbol{I})$$
$$q(\boldsymbol{x}_{T-1}|\boldsymbol{x}_T) = q(\boldsymbol{x}_{T-1})$$

$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(\boldsymbol{x}_t)}{\sqrt{\alpha_t}} \right)}_{\text{“predicted } \boldsymbol{x}_0\text{”}} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \epsilon_\theta^{(t)}(\boldsymbol{x}_t)}_{\text{“direction pointing to } \boldsymbol{x}_t\text{”}} + \underbrace{\sigma_t \epsilon_t}_{\text{random noise}}$$

This is impossible to obtain $\boldsymbol{x}_0$ for given $\boldsymbol{\epsilon}_\theta^{(T)}$ due to the broken chain:

$$\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon} \quad \Rightarrow \quad \boldsymbol{x}_T = 0 \cdot \boldsymbol{x}_0 + 1 \cdot \boldsymbol{\epsilon} = \boldsymbol{\epsilon}.$$

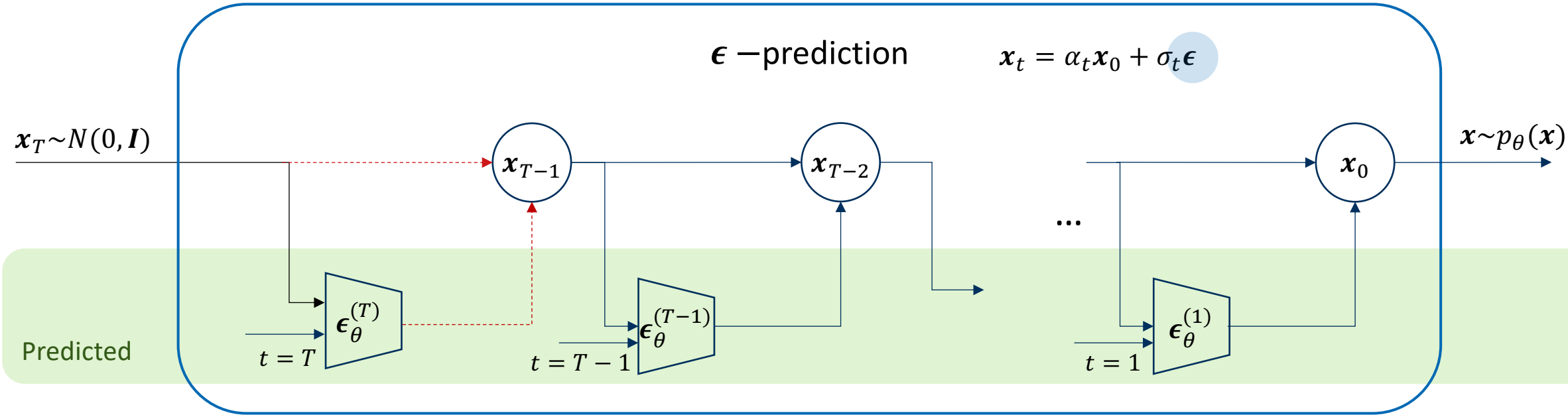
3. Improving Sample Quality – 0 Terminal SNR and Broken Chain



For skipped sampling, the impact the missteps can be fatal.
Less chances to be corrected later on.

Salimans and Ho, "Progressive Distillation for Fast Sampling of Diffusion Models", *ICLR* 2022.

3. Improving Sample Quality - Recall Parametrization

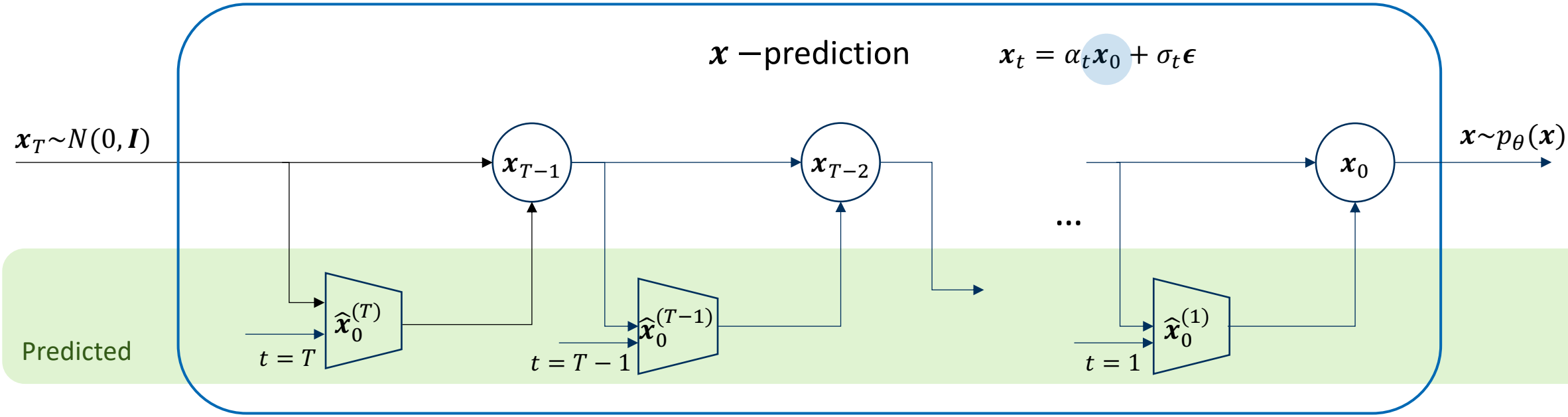


Diffusion and denoising by ϵ−prediction

$\Rightarrow$ The formula breaks when $\mathrm{SNR}(T) = 0$.

$\Rightarrow$ x_{T-1} is **generated**, not obtained by denoising.

3. Improving Sample Quality - Alternative $\boldsymbol{x}$ Parametrization



To predict $\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t, t)$ directly and use it to predict $\boldsymbol{x}_{t-1}$,

$$\boldsymbol{x}_{t-1} = \alpha_{t-1}\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t, t) + \sigma_{t-1}\boldsymbol{\epsilon}.$$

Salimans and Ho, "Progressive Distillation for Fast Sampling of Diffusion Models", *ICLR* 2022.

3. Improving Sample Quality – Hybrid v-Parametrization

3. Improving Sample Quality – Hybrid v-Parametrization

3. Improving Sample Quality – Hybrid $\boldsymbol{v}$-Prediction

A linear combination of $\boldsymbol{\epsilon}$ −prediction and direct x − prediction.

$$\boldsymbol{v}_\theta^{(t)}(\boldsymbol{x}_t) \coloneqq \alpha_t \boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_t) - \sigma_t \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)$$

Recall the definition of α_t and σ_t
$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = N(\alpha_t \boldsymbol{x}_0, \sigma_t^2 \boldsymbol{I})$

$\Rightarrow$ Coefficients compensating the proportion in diffusion process.

From $\boldsymbol{v}$, one can reconstruct $\hat{\boldsymbol{x}}_0$ and $\boldsymbol{\epsilon}_\theta^{(t)}$ by

$$\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t) = \alpha_t \boldsymbol{x}_t - \sigma_t \boldsymbol{v}_\theta^{(t)}(\boldsymbol{x}_t) \text{ and } \boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_t) = (\boldsymbol{x}_t - \alpha_t \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t))/\sigma_t.$$

Accordingly, the denoising step becomes as below by using $\boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_t)$ in the existing formula.

$$\begin{aligned}\boldsymbol{x}_s &= \alpha_s \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t) + \sigma_s \boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_t) + (1 - \alpha_s^2 - \sigma_s^2)\boldsymbol{\epsilon}, \quad \text{where } \boldsymbol{\epsilon} \sim N(0, \boldsymbol{I}) \\ &= (\alpha_s \alpha_t - \sigma_s \sigma_t)\boldsymbol{x}_t + (\sigma_s \alpha_t - \alpha_s \sigma_t)\boldsymbol{v}_\theta^{(t)}(\boldsymbol{x}_t) + (1 - \alpha_s^2 - \sigma_s^2)\boldsymbol{\epsilon}\end{aligned}$$

* s indicates the next time index.

Salimans and Ho, “Progressive Distillation for Fast Sampling of Diffusion Models”, *ICLR* 2022.

3. Improving Sample Quality – Hybrid $\boldsymbol{v}$-Prediction

A linear combination of $\boldsymbol{\epsilon}$ −prediction and direct x − prediction.

$$\boldsymbol{v}_\theta^{(t)}(\boldsymbol{x}_t) \coloneqq \alpha_t \boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_t) - \sigma_t \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)$$

Recall the definition of α_t and σ_t
$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = N(\alpha_t \boldsymbol{x}_0, \sigma_t^2 \boldsymbol{I})$

Even when $t = T, \alpha_T = 0, \sigma_T = 1,$ and $\mathrm{SNR}(T) = 0$

$$\boldsymbol{x}_s = \left[(\alpha_s\alpha_t - \sigma_s\sigma_t)\boldsymbol{x}_t + (\sigma_s\alpha_t - \alpha_s\sigma_t)\boldsymbol{v}_\theta^{(t)}(\boldsymbol{x}_t) + (1 - \alpha_s^2 - \sigma_s^2)\boldsymbol{\epsilon}\right]_{t=T}$$
$$= -\sigma_s \boldsymbol{x}_T - \alpha_s \boldsymbol{v}_\theta^{(t)}(\boldsymbol{x}_T) + (1 - \alpha_s^2 - \sigma_s^2)\boldsymbol{\epsilon}$$

$\Rightarrow$ The previous sample survives even for the zero-SNR case.

Cf) For the traditional $\boldsymbol{\epsilon}$ −prediction,

$$\boldsymbol{x}_s = \left[\alpha_s \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t) + \sigma_s \boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_t) + (1 - \alpha_s^2 - \sigma_s^2)\boldsymbol{\epsilon}\right]_{t=T}$$
$$= \alpha_s \hat{\boldsymbol{x}}_0(\boldsymbol{x}_T) + \sigma_s \boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_T) + (1 - \alpha_s^2 - \sigma_s^2)\boldsymbol{\epsilon}$$

This is impossible to obtain arithmetically due to the broken chain:

$$\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon} \quad \Rightarrow \quad \boldsymbol{x}_T = 0 \cdot \boldsymbol{x}_0 + 1 \cdot \boldsymbol{\epsilon} = \boldsymbol{\epsilon}.$$

Salimans and Ho, “Progressive Distillation for Fast Sampling of Diffusion Models”, ICLR 2022.

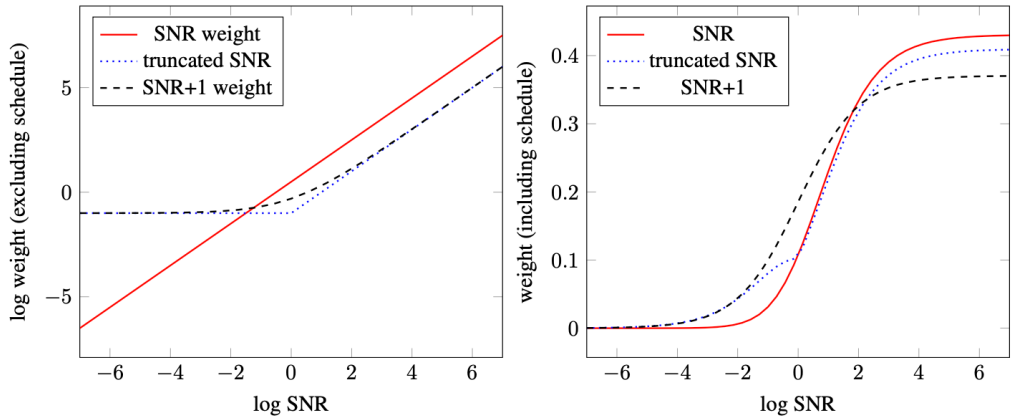
3. Improving Sample Quality – Weighting in Loss Function

The simple loss can be rewritten w.r.t. the notation with SNR:

$$L_{\text{simple}} = \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_t) \right\|^2 = \left\| \frac{1}{\sigma_t}(\boldsymbol{x}_t - \alpha_t \boldsymbol{x}_0) - \frac{1}{\sigma_t}\left(\boldsymbol{x}_t - \alpha_t \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)\right) \right\|^2 = \frac{\alpha_t^2}{\sigma_t^2} \|\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)\|^2 = \text{SNR}(t) \|\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)\|^2.$$

$\Rightarrow$ For scheduling methods with almost-0 SNR values for some $\boldsymbol{t}$, it's ignored.

$\Rightarrow$ Weighting in the Loss function considering low SNR cases are suggested.



Salimans and Ho, "Progressive Distillation for Fast Sampling of Diffusion Models", ICLR 2022.

3. Improving Sample Quality - Weighting of Loss Function

The simple loss can be rewritten w.r.t. the notation with SNR:

$$L_{\text{simple}} = \left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_t)\right\|^2 = \left\|\frac{1}{\sigma_t}(\boldsymbol{x}_t - \alpha_t \boldsymbol{x}_0) - \frac{1}{\sigma_t}\left(\boldsymbol{x}_t - \alpha_t \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)\right)\right\|^2 = \frac{\alpha_t^2}{\sigma_t^2}\|\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)\|^2 = \text{SNR}(t)\|\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)\|^2.$$

⇒ For scheduling methods with almost-0 SNR values for some $\boldsymbol{t}$, it's ignored.

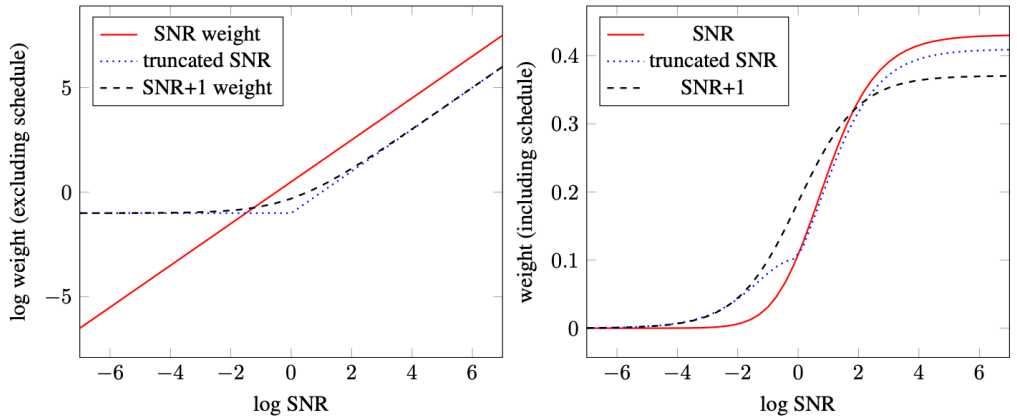
⇒ Weighting in the Loss function considering low SNR cases are suggested.

'truncated SNR' weighting:

$$L_\theta = \max\left(\|\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)\|^2, \left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_t)\right\|^2\right) = \max(\text{SNR}(t), 1)\|\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)\|^2.$$

'SNR+1' weighting:

$$\begin{aligned} L_\theta &= \left\|\boldsymbol{v} - \boldsymbol{v}_\theta^{(t)}(\boldsymbol{x}_t)\right\|^2 = \left\|\alpha_t(\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_t)) - \sigma_t(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t))\right\|^2 \\ &= \left\|\left(-\frac{\alpha_t^2}{\sigma_t} - \sigma_t\right)(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t))\right\|^2 = \frac{1}{\sigma_t^2}\|(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t))\|^2 \\ &= (\text{SNR}(t) + 1)\|\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)\|^2. \end{aligned}$$



Recall) $\boldsymbol{v}_\theta^{(t)}(\boldsymbol{x}_t) \coloneqq \alpha_t \boldsymbol{\epsilon}_\theta^{(t)}(\boldsymbol{x}_t) - \sigma_t \hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)$.

Salimans and Ho, "Progressive Distillation for Fast Sampling of Diffusion Models", ICLR 2022.

3. Improving Sample Quality - Evaluation of Parametrization and Weighting

Network Output	Loss Weighting	Stochastic sampler	DDIM sampler
$(\mathbf{x}, \epsilon)$ combined	SNR	2.54/9.88	2.78/9.56
	Truncated SNR	2.47/9.85	2.76/9.49
	SNR+1	2.52/9.79	2.87/9.45
$\mathbf{x}$	SNR	2.65/9.80	2.75/9.56
	Truncated SNR	2.53/**9.92**	**2.51/9.58**
	SNR+1	2.56/9.84	2.65/9.52
ϵ	SNR	2.59/9.84	2.91/9.52
	Truncated SNR	N/A	N/A
	SNR+1	2.56/9.77	3.27/9.41
$\mathbf{v}$	SNR	2.65/9.86	3.05/9.56
	Truncated SNR	**2.45**/9.80	2.75/9.52
	SNR+1	2.49/9.77	2.87/9.43

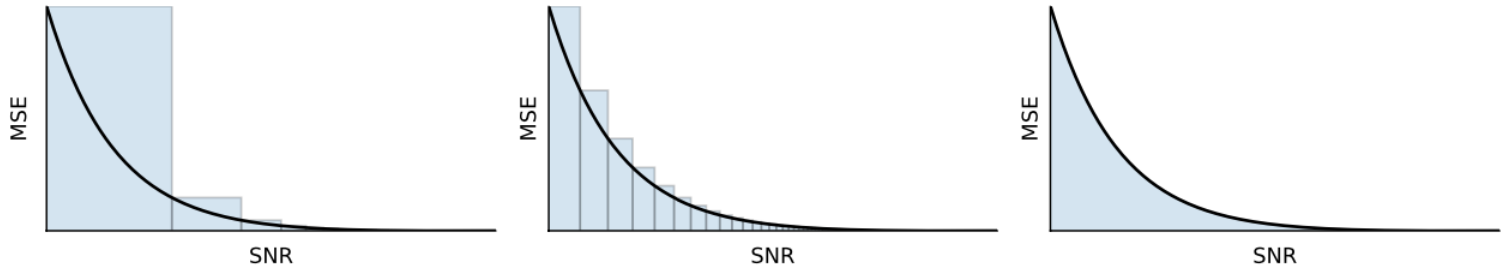
Table 1: Generated sample quality as measured by FID and Inception Score (FID/IS) on unconditional CIFAR-10, training the original model (no distillation), and comparing different parameterizations and loss weightings discussed in Section 4. All reported results are averages over 3 random

- The best cases are by the truncated SNR weighting.
- The ϵ-prediction is not optimal in any cases.

$\Rightarrow$ Including the prediction of $\boldsymbol{x}$ helps for skipped sampling and 0-SNR scheduling.

Salimans and Ho, "Progressive Distillation for Fast Sampling of Diffusion Models", ICLR 2022.

Important Learning-Based Techniques: Variational Diffusion Model

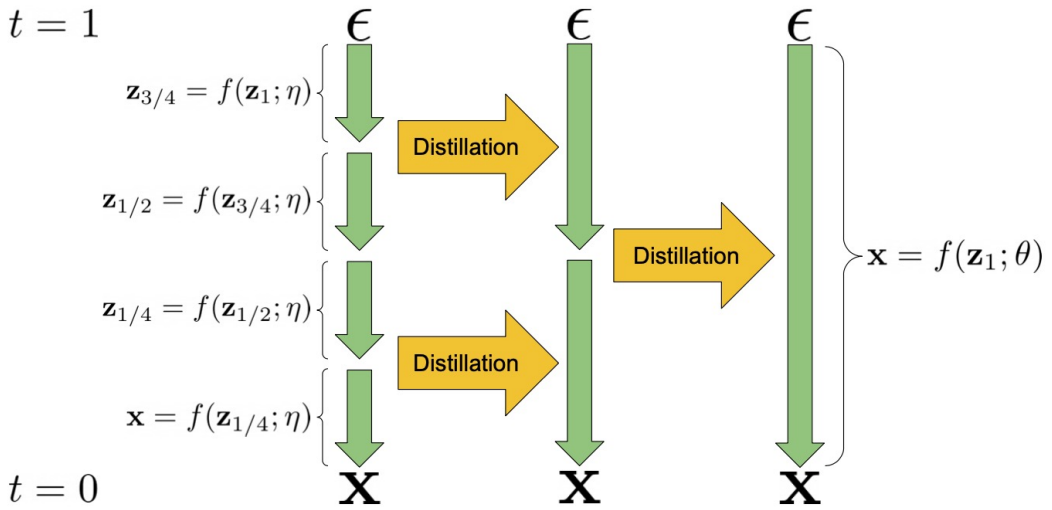


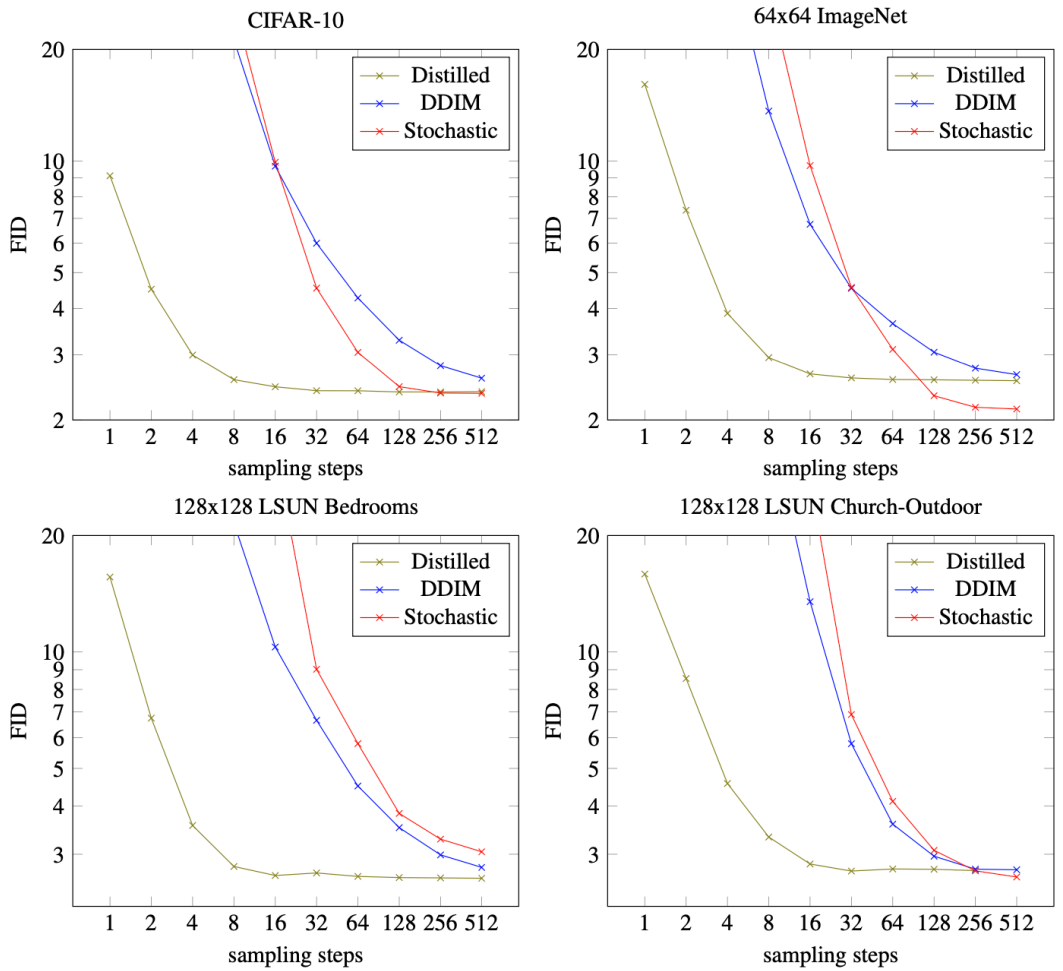
- More steps leading better loss -> continuous and infinite diffusion steps
- Learning the noise scheduling in a parametric form: $\sigma_t^2 = \text{sigmoid}(\gamma_{\boldsymbol{\eta}}(t))$ $\alpha_t^2 = \text{sigmoid}(-\gamma_{\boldsymbol{\eta}}(t))$ $\text{SNR}(t) = \exp(-\gamma_{\boldsymbol{\eta}}(t))$
- Using Fourier features of the latent variables as augmented input.

$\Rightarrow$ Lowering the variance in the variational bound, faster convergence, improved likelihood.

Kingma, Salimans, Poole, and Ho, "Variational Diffusion Models", *NeurIPS* 2021

Important Learning-Based Techniques: Progressive Distillation





- Learning distilled models for fewer denoising steps.
- Reducing # of steps with less performance degradation.

Salimans and Ho, "Progressive Distillation for Fast Sampling of Diffusion Models", *ICLR* 2022.

Part 3. Application of Diffusion Models in Communications

Applications of Diffusion Models in Communications

- Rx: Denoising undesired signal from for accuracy. Focusing on the gradual denoising idea.

Detection

Massive MIMO: Zilberstein et al., "Annealed Langevin Dynamics for Massive MIMO Detection.", *IEEE TWC* 2023.

Zilberstein, Swami and Segarra, "Joint Channel Estimation and Data Detection in Massive MIMO Systems Based on Diffusion Models," *ICASSP* 2024

Hardware Impairment, quantization error, channel distortion:

Letafati, Ali, Latva-aho, "Denoising diffusion probabilistic models for hardware-impaired communication systems: Towards wireless generative AI," *arXiv*, 2023.

Decoding

Channel coding: Choukroun, Yoni, and Lior Wolf., "Denoising Diffusion Error Correction Codes.", *ICLR* 2023.

Semantic coding: Wu et al., "CDDM: Channel Denoising Diffusion Models for Wireless Semantic Communications." *IEEE TWC* 2024

Pezone, Musa, Caire, and Barbarossa, "Semantic-Preserving Image Coding Based on Conditional Diffusion Models," *ICASSP,* 2024

Source Separation

Jayashankar et al., "Score-based Source Separation with Applications to Digital Communication Signals.", *NeurIPS* 2023.

Applications of Diffusion Models in Communications

- Channel: Learning channel distributions by using diffusion models.

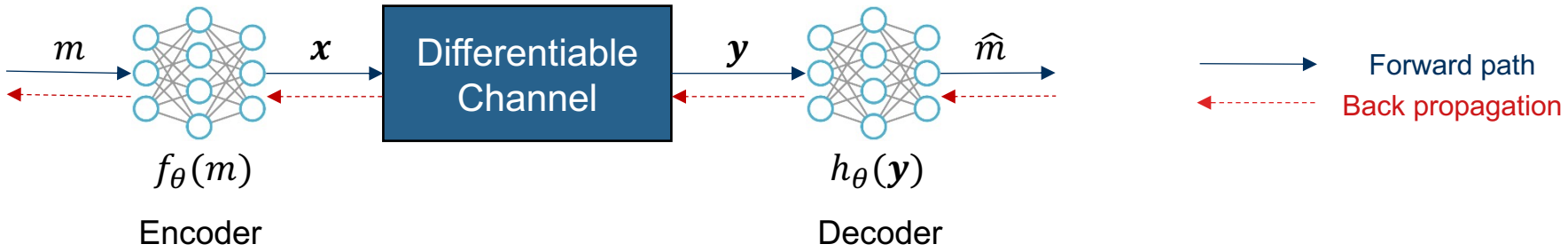
Kim, Fritschek, and Schaefer, "Learning End-to-End Channel Coding with Diffusion Models," *ITG WSA-SCC and arXiv*, 2023

- Tx: learning the probabilistic constellation design with diffusion models

Letafati, Ali, Latva-aho, “Probabilistic constellation shaping with denoising diffusion probabilistic models: A novel approach,” *arXiv*, 2023.

NN-Based End-to-End Channel Coding and Modulation

cf. classical channel coding framework



Advantage: joint optimization of channel coding and modulation, lower error probability.

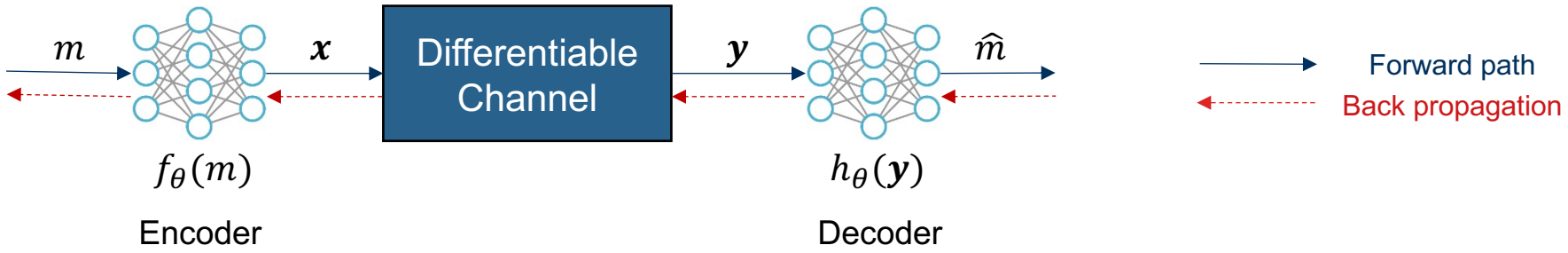
O'Shea, Karra, and Clancy, "Learning to communicate: Channel auto-encoders, domain specific regularizers, and attention," *IEEE ISSPIT*, 2016.

O'Shea and Hoydis, "An introduction to deep learning for the physical layer," *IEEE TCCN*, 2017.

Challenge: the channel should be differentiable to train encoder by backpropagation.

NN-Based End-to-End Channel Coding and Modulation

cf. classical channel coding framework



Solutions: Approximating

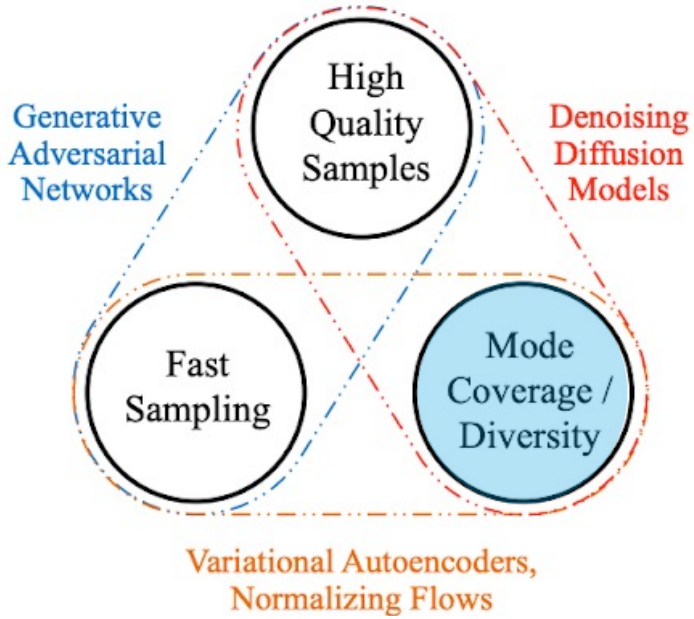
- the channel distribution by generative models e.g. GAN, WGAN, residual-aided GAN.

  Ye, Li, Juang, and Sivanesan, "Channel agnostic end-to-end learning based communication systems with conditional GAN," *IEEE Globecom*, 2018.
  O'Shea, Roy, West, and Hilburn, "Physical layer communications system design over-the-air using adversarial networks," *EUSIPCO*, 2018.
  Dörner, Henninger, Cammerer, and t. Brink, "WGAN-based autoencoder training over-the-air," *IEEE SPAWC*, 2020.
  Jiang et al., "Residual-aided end-to-end learning of communication system without known channel," *IEEE TCCN*, 2022.

- the gradient of the channel by reinforcement learning.

  Aoudia and Hoydis, "End-to-end learning of communications systems without a channel model," *ACSSC* 2018.

Motivation

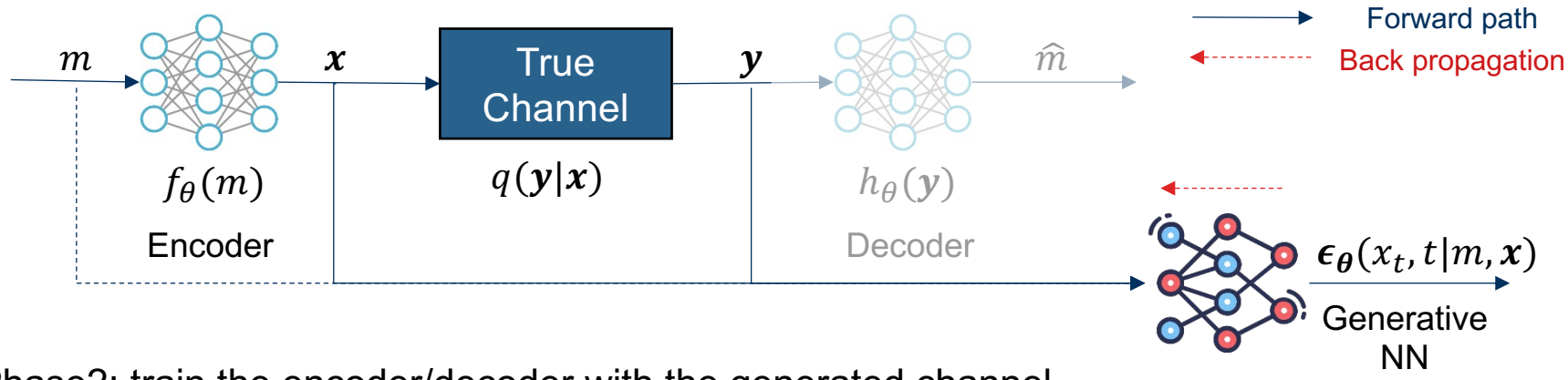
Usage	Channel Approximation	Image Generation
GAN-based models	Works well for simple channels (e.g. AWGN, Rayleigh fading channel and a static wireless channel.) ⇒ Limitation: mode collapse (e.g. mode collapse with a tapped delay line channel)	
Diffusion Models	???	

Ye, Li, Juang, and Sivanesan, "Channel agnostic end-to-end learning based communication systems with conditional GAN," *IEEE Globecom*, 2018.
O'Shea, Roy, West, and Hilburn, "Physical layer communications system design over-the-air using adversarial networks," *EUSIPCO*, 2018.
Dörner, Henninger, Cammerer, and t. Brink, "WGAN-based autoencoder training over-the-air," *IEEE SPAWC*, 2020.
Jiang et al., "Residual-aided end-to-end learning of communication system without known channel," *IEEE TCCN*, 2022.

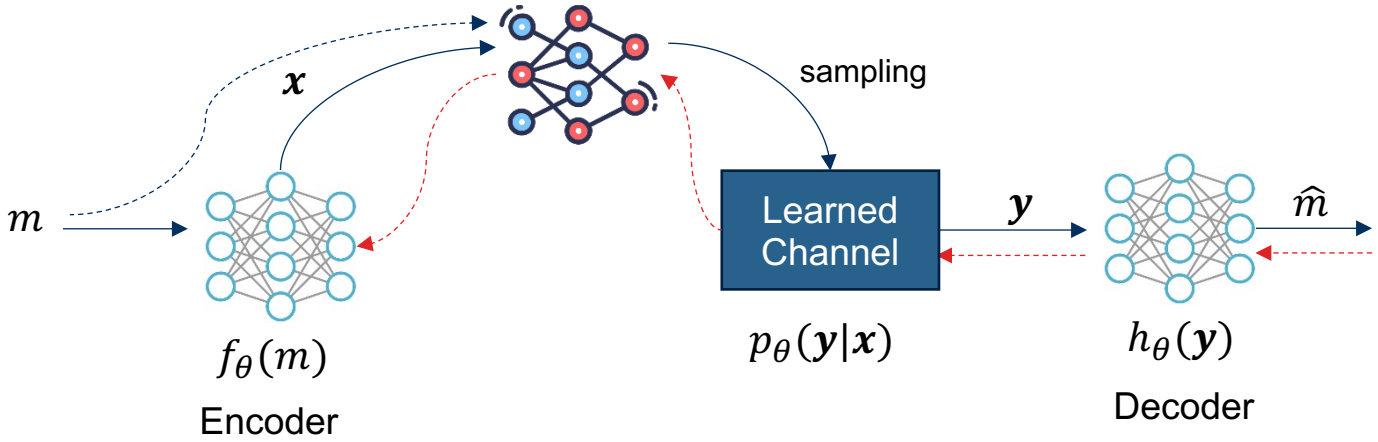
NN-Based End-to-End Channel Coding Framework

Conventional iterative training algorithm.

Phase1: train the generative NN for the codewords. The encoder is fixed.



Phase2: train the encoder/decoder with the generated channel.



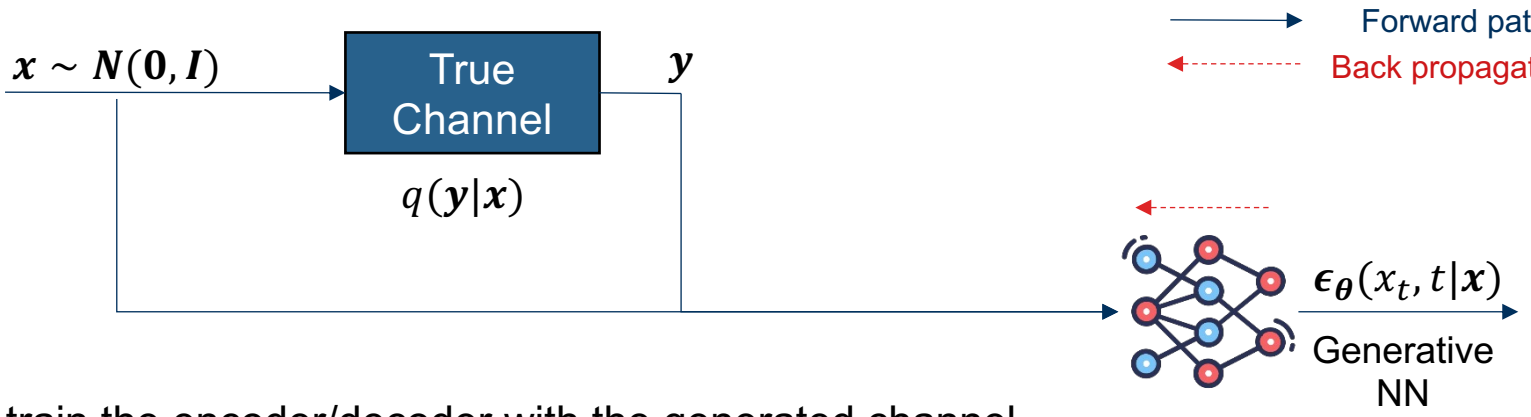
Repeat until converged:
- Phase1
- Phase 2

=> Taking indefinite time!
=> Encoder–specific generator.

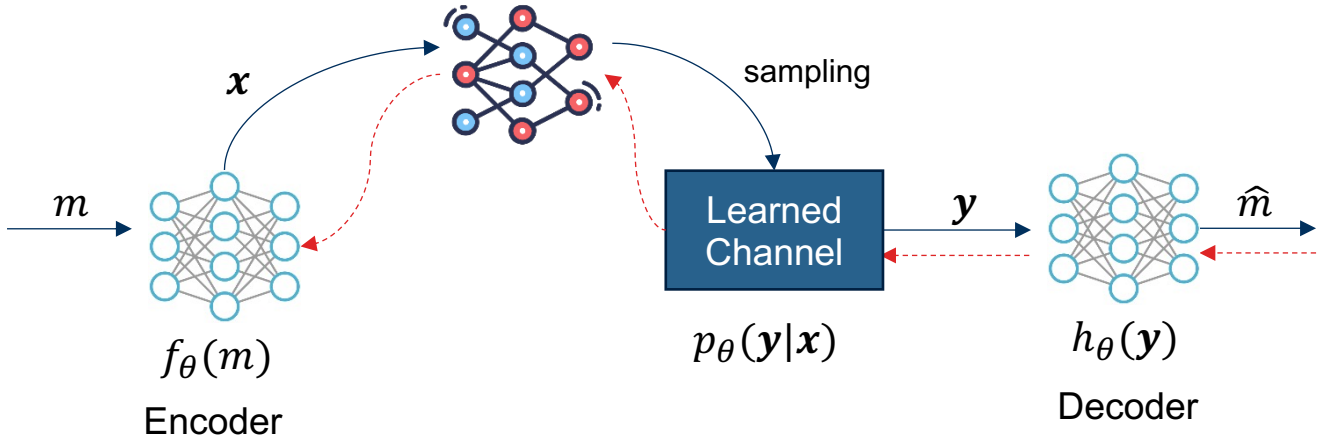
NN-Based End-to-End Channel Coding Framework

Proposed pretraining algorithm.

Phase1: train the generative NN thoroughly for random channel inputs.

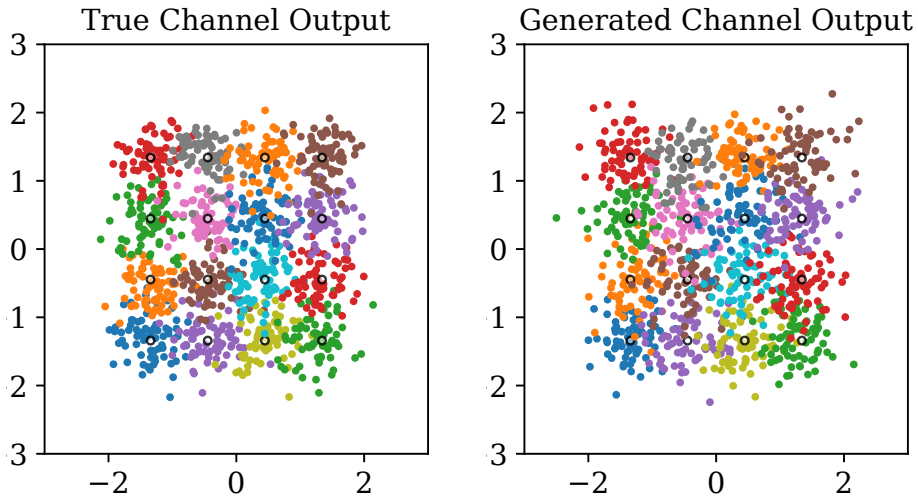


Phase2: train the encoder/decoder with the generated channel.

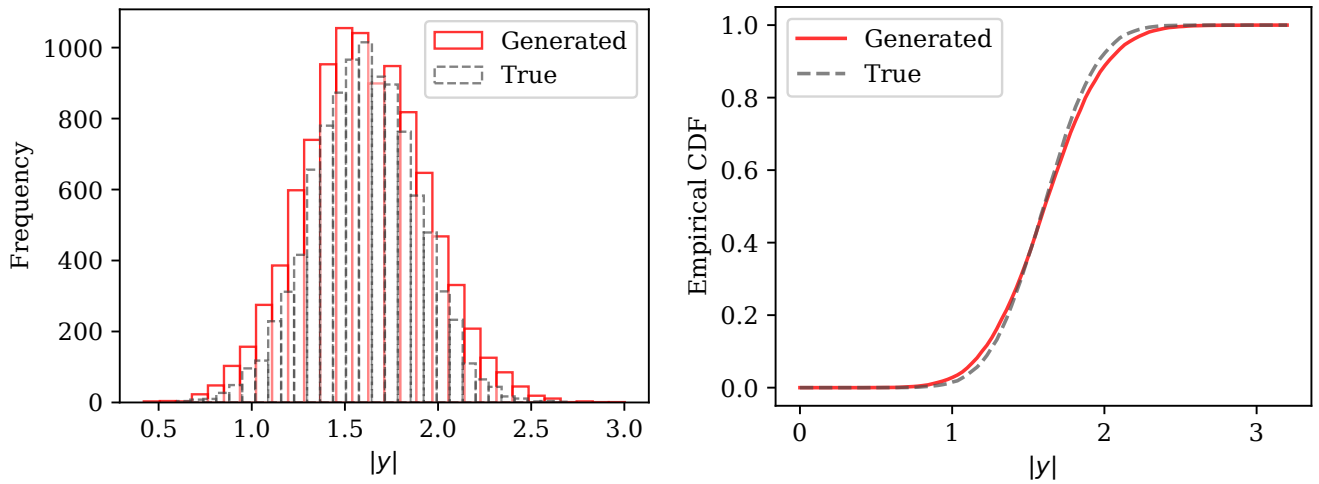


=> Iteration-free. Do Phase 1 only once.
=> Versatile.

Simulations – Channel Generation with 16-QAM and AWGN



(a) Constellation of channel output



(b) Histogram and empirical CDF of $|\boldsymbol{y}|$

Simulation Settings

Channel: AWGN with $\frac{E_b}{N_0} = 5$ dB, n $= 2$

Modulation: 16-QAM ($M = 16$)

Diffusion Model: $T = 100, \beta_t = 0.05$ ∀t, 3 hidden layers w/ 64 neurons,

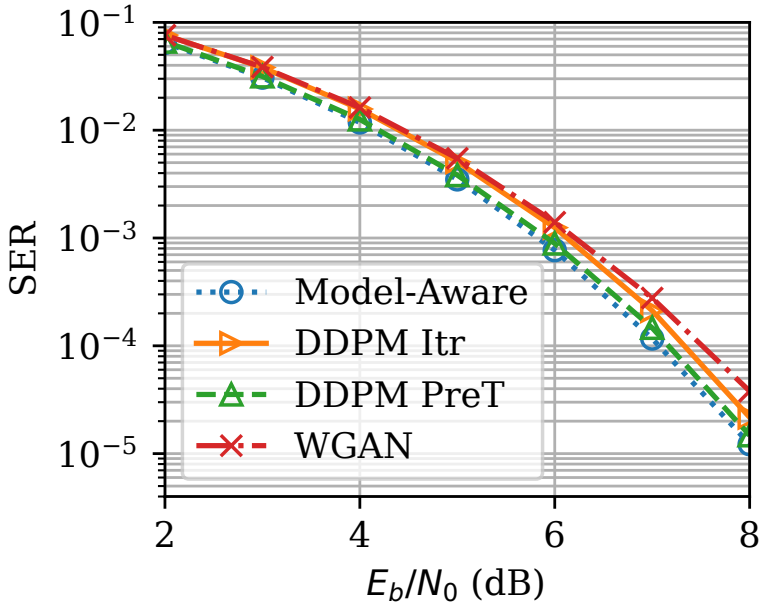
Adam optimizer,

learning rate $10^{-3} \sim 10^{-4}$

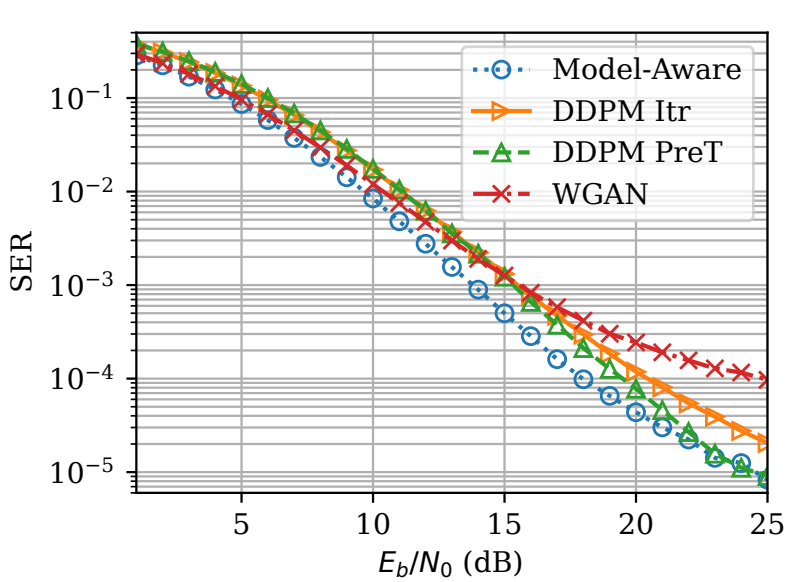
The conditional diffusion model can generate the distribution of each message accurately.

Kim, Fritschek, and Schaefer, "Learning End-to-End Channel Coding with Diffusion Models," *ITG WSA-SCC and arXiv*, 2023

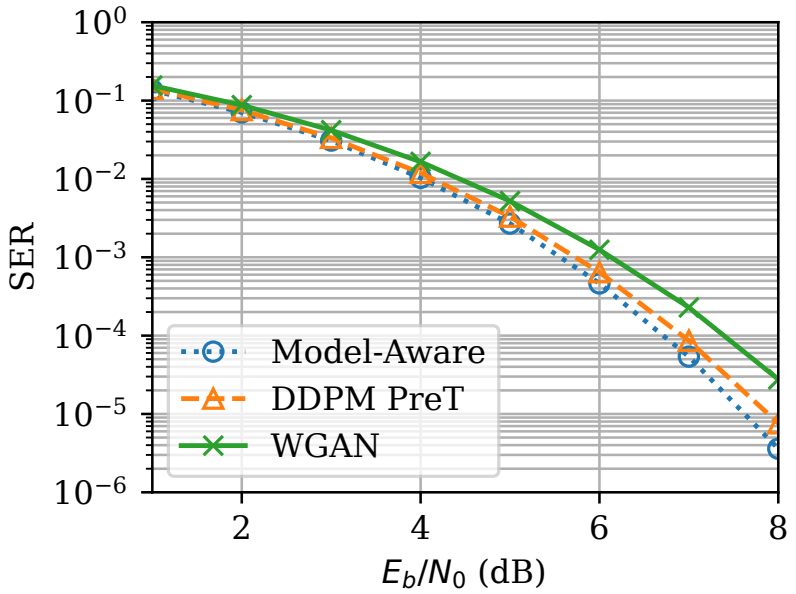
Simulations – E2E Learning for 3 Channel Models



(a) AWGN



(b) Rayleigh fading

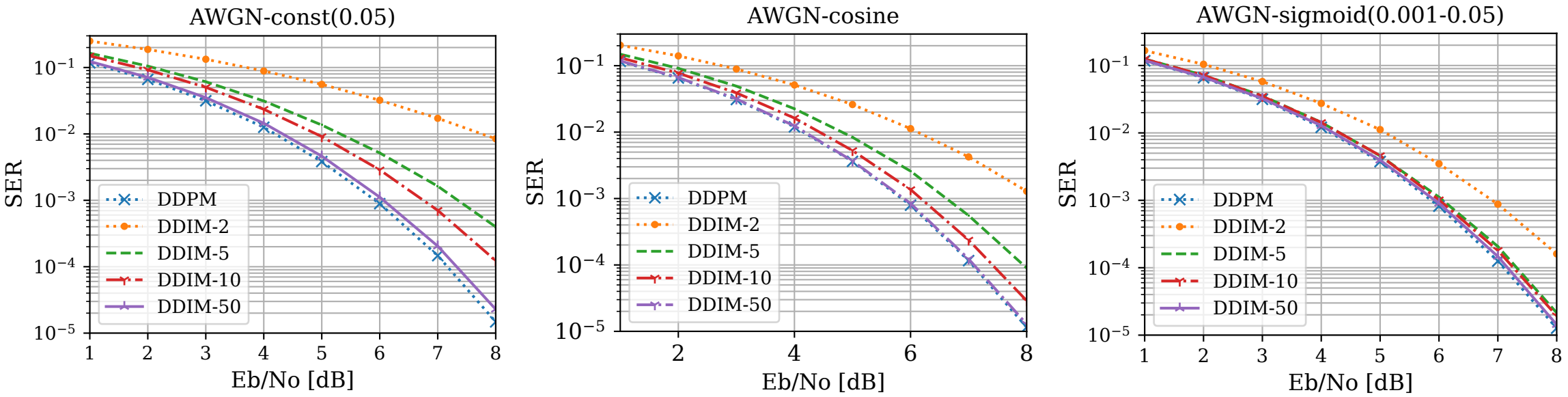


(c) Solid State Power Amplification (SSPA)

- The pre-training algorithm outperform the iterative algorithm.
- DDPM outperformed the tested WGANs.
- A simple DDPM could substitute channel models with random noise, random fading, and non-linear power amplification.

Kim, Fritschek, and Schaefer, "Learning End-to-End Channel Coding with Diffusion Models," *ITG WSA-SCC and arXiv*, 2023

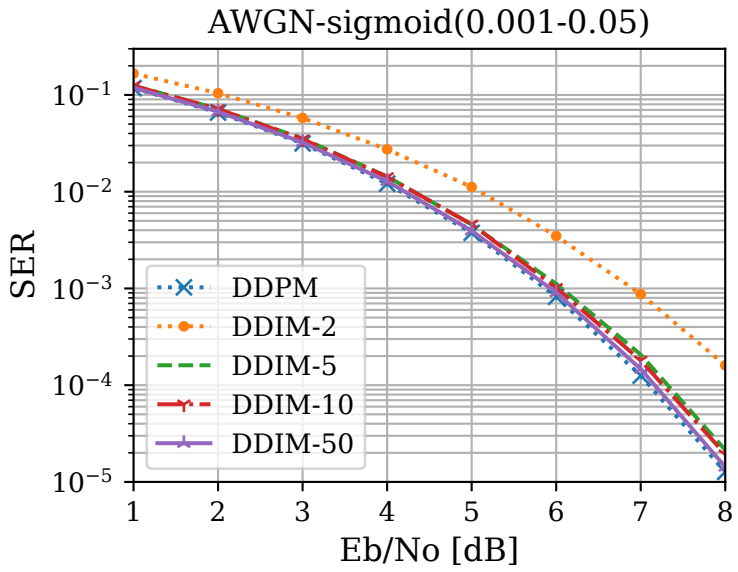
Optimization of the Model: Noise Scheduling w/ AWGN Channel

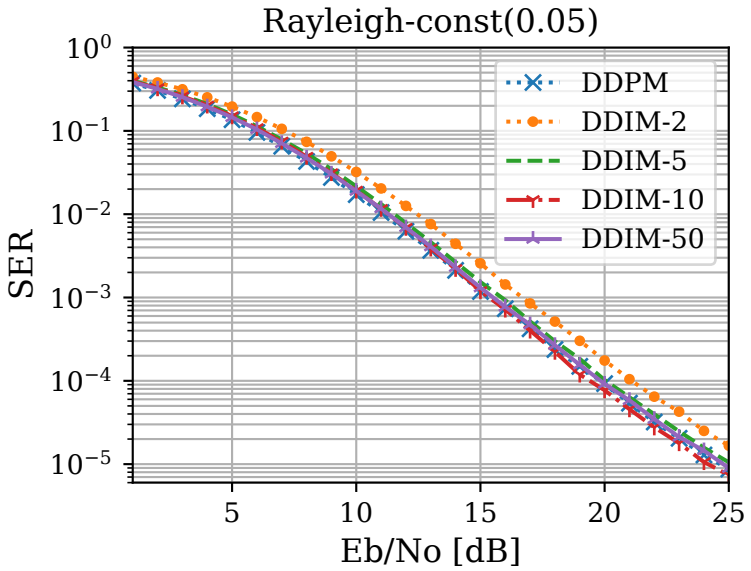


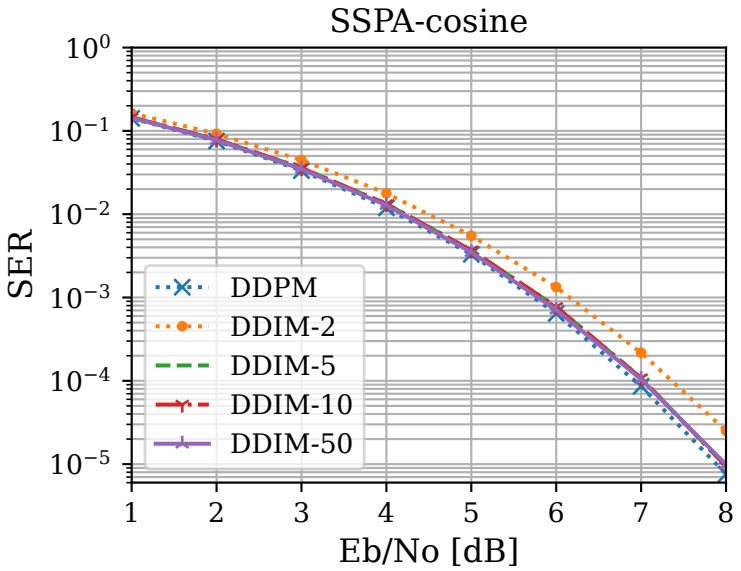
1. DDIMs with skipped sampling gives some performance degradation.
2. The degradation depends heavily on $\boldsymbol{\beta_t}$. For DDPM, the difference is slight.

Kim, Fritschek, and Schaefer, "Learning End-to-End Channel Coding with Diffusion Models," *ITG WSA-SCC and arXiv*, 2023

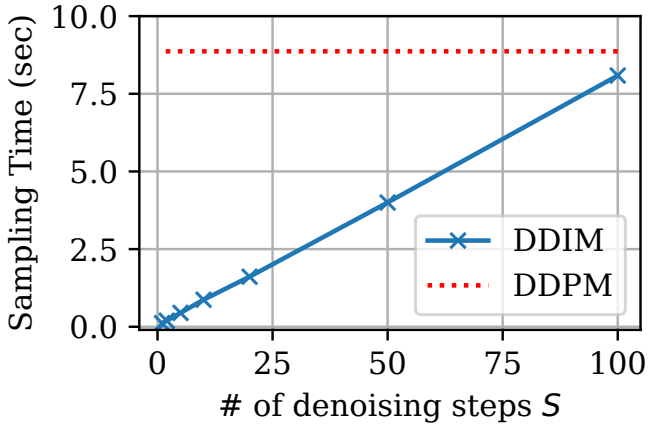
Sampling Acceleration with Skipped Sampling







With a good choice of β_t scheduling,
the sample quality drop can be insignificant, while
reducing the sampling time significantly.



Measured with the AWGN channel,
# of samples = 2^{17}

Kim, Fritschek, and Schaefer, "Learning End-to-End Channel Coding with Diffusion Models," *ITG WSA-SCC and arXiv*, 2023

Summary of the Tutorial

- Pros and Cons of Diffusion Models?

High sample quality, mode coverage, and low sampling speed.
⇒ Good for generating synthetic data, emulating distributions, but computationally expensive.

- What can be improved for diffusion models?
- Sampling speed by skipped sampling and robust techniques (DDIM, distillation, optimal trajectory)
- Likelihood and mode coverage (Noise scheduling, VDM, iDDPM)
- Sample quality (loss weighting, parametrization, …)

Remarks
- No absolute solution for all problems: experimental optimization is important for each application.
- Yet, proper techniques can be chosen smartly for target performances.
- In communications, mostly the baseline diffusion models are used. => Room for further improvement!

TECHNISCHE UNIVERSITÄT DRESDEN

Sponsored by

DFG Deutsche Forschungsgemeinschaft German Research Foundation

Grant SCHA 1944/7-1

CeTI Centre for Tactile Internet with Human-in-the-Loop

GEFÖRDERT VOM

Bundesministerium für Bildung und Forschung

6G-life

Grant 16KISK001K

Contacts: {muah.kim, rick.fritschek, rafael.schaefer}@tu-dresden.de

Coffee Break