Diffusion Model Journal Club

Kingma & Gao: Understanding Diffusion Objectives as the ELBO with Simple Data Augmentation

Johannes Zenn
Tübingen AI Center, University of Tübingen, IMPRS-IS

Disclaimer: These slides were largely created by ChatGPT, so I can blame all errors on it (?)

Creating Data From Noise

Last Week: Modeling the Reverse Stochastic Differential Equation

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

- forward SDE (from data to noise)

$$d\boldsymbol{z} = \boldsymbol{f}(\boldsymbol{z}, t)dt + g(t)d\boldsymbol{w} \qquad (1)$$

- the forward SDE can be reversed (Anderson, 1982) resulting in the reverse SDE

$$d\boldsymbol{z} = [\boldsymbol{f}(\boldsymbol{z}, t) - g(t)^2 \nabla_{\boldsymbol{x}} \log q(\boldsymbol{x}, t)]dt + g(t)d\boldsymbol{w} \qquad (2)$$

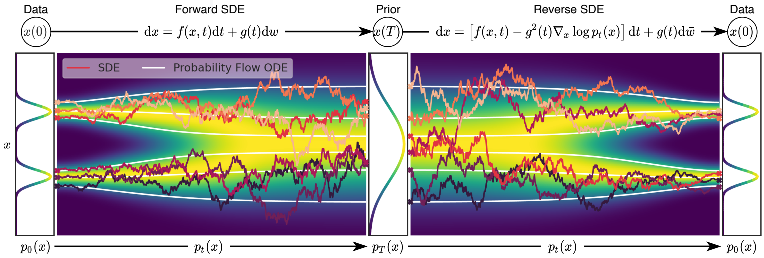


Figure taken from Song et al. (2020)

Two Stochastic Differential Equations

Last Week: Variance-Preserving and Variance-Exploding Forward SDEs

Variance-Preserving (VP) SDE

$$
\begin{aligned}
d\boldsymbol{z} = & \underbrace{-\frac{1}{2}\left(\frac{d}{dt}\log(1+e^{-\lambda_t})\right)}_{\text{drift}} \boldsymbol{z}\,dt \\
& + \underbrace{\sqrt{\frac{d}{dt}\log(1+e^{-\lambda_t})}}_{\text{diffusion}}\,d\boldsymbol{w}
\end{aligned} \tag{3}
$$

Variance-Exploding (VE) SDE

$$
\begin{aligned}
d\boldsymbol{z} = & \,0\,dt \\
& + \sqrt{\frac{d}{dt}\log(1+e^{-\lambda_t})}\,d\boldsymbol{w}
\end{aligned} \tag{4}
$$

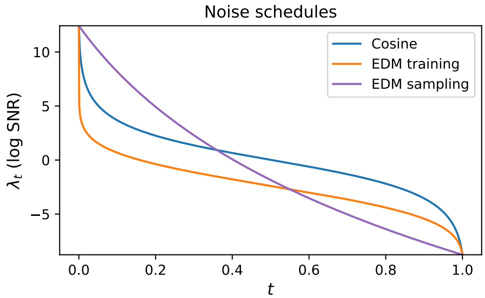


Figure taken from Kingma & Gao (2024)

Denoising Score Matching

Following Hyvärinen & Dayan (2005) and Vincent (2011)

- for generative modeling, we need to approximate the score (of the data) $\nabla_{\boldsymbol{x}} \log q(\boldsymbol{x}, \boldsymbol{t})$ in the reverse SDE

$$d\boldsymbol{z} = [\boldsymbol{f}(\boldsymbol{z}, t) - g(t)^2 \nabla_{\boldsymbol{x}} \log q(\boldsymbol{x}, t)]dt + g(t)d\boldsymbol{w} \tag{5}$$

- following TODO: Hyvärinen 2005, Vincent 2011 we want to minimize the following objective

$$\mathbb{E}_{q(\boldsymbol{x})} \left[\| \nabla_{\boldsymbol{x}} \log q(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}) \|_2^2 \right] \tag{6}$$

- we have

$$\begin{aligned} &\mathbb{E}_{q(\boldsymbol{x})} \left[\| \nabla_{\boldsymbol{x}} \log q(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}) \|_2^2 \right] \\ &= \mathbb{E}_{q(\boldsymbol{x})} \left[\left\| \frac{\partial \log q(\boldsymbol{x})}{\partial \boldsymbol{x}} \right\|^2 \right] - \mathbb{E}_{q(\boldsymbol{x})} \left[\left\langle \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}), \frac{\partial \log q(\boldsymbol{x})}{\partial \boldsymbol{x}} \right\rangle \right] \\ &+ \mathbb{E}_{q(\boldsymbol{x})} \left[\| \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}) \|^2 \right] \end{aligned} \tag{7}$$

Denoising Score Matching

Following Hyvärinen & Dayan (2005) and Vincent (2011)

- and, after some manipulations,

$$\mathbb{E}_{q(\boldsymbol{x})}\left[\left\langle \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}), \frac{\partial \log q(\boldsymbol{x})}{\partial \boldsymbol{x}} \right\rangle\right] = \mathbb{E}_{q(\boldsymbol{x},\tilde{\boldsymbol{x}})}\left[\left\langle \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}), \frac{\partial \log q(\boldsymbol{x} \mid \tilde{\boldsymbol{x}})}{\partial \boldsymbol{x}} \right\rangle\right] \tag{8}$$

  where $q(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = q(\boldsymbol{x} \mid \tilde{\boldsymbol{x}}) q_0(\tilde{\boldsymbol{x}})$ and $q(\boldsymbol{x} \mid \tilde{\boldsymbol{x}})$ is a noising distribution
- therefore,

$$\begin{aligned} &\mathbb{E}_{q(\boldsymbol{x})}\left[\|\nabla_{\boldsymbol{x}} \log q(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x})\|_2^2\right] \\ &= \mathbb{E}_{q(\boldsymbol{x},\tilde{\boldsymbol{x}})}\left[\|\nabla_{\boldsymbol{x}} \log q(\boldsymbol{x} \mid \tilde{\boldsymbol{x}}) - \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x})\|_2^2\right] \end{aligned} \tag{9}$$

- we parameterize the score of the model by

$$\begin{aligned} &\mathbb{E}_{q(\boldsymbol{x},\tilde{\boldsymbol{x}})}\left[\|\nabla_{\boldsymbol{x}} \log q(\boldsymbol{x} \mid \tilde{\boldsymbol{x}}) - \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x})\|_2^2\right] \\ &= \mathbb{E}_{q(\boldsymbol{x},\tilde{\boldsymbol{x}})}\left[\|\nabla_{\boldsymbol{x}} \log q(\boldsymbol{x} \mid \tilde{\boldsymbol{x}}) - s_{\boldsymbol{\theta}}(\boldsymbol{x})\|_2^2\right] \end{aligned} \tag{10}$$

Denoising Score Matching

Following Hyvärinen & Dayan (2005) and Vincent (2011)

- now, introduce multiple noise levels λ

$$\boldsymbol{z}_\lambda = \alpha_\lambda \boldsymbol{x} + \sigma_\lambda \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$$

- we recover the denoising score matching (DSM) objective

$$\mathcal{L}_{\text{DSM}}(\boldsymbol{x}) = \mathbb{E}_{t,\boldsymbol{\epsilon}} \left[\left\| \boldsymbol{s}_\theta(\boldsymbol{z}_\lambda; \lambda) - \frac{\boldsymbol{z}_\lambda - \alpha_\lambda \boldsymbol{x}}{\sigma_\lambda^2} \right\|_2^2 \right] \tag{11}$$

- where we used the gradient of $\log q(\boldsymbol{z}_\lambda | \boldsymbol{x})$

$$\nabla_{\boldsymbol{z}_\lambda} \log q(\boldsymbol{z}_\lambda | \boldsymbol{x}) = -\frac{\boldsymbol{z}_\lambda - \alpha_\lambda \boldsymbol{x}}{\sigma_\lambda^2} \tag{12}$$

Parameterization of the Score Network

Last Week: Following Karras et al. (2022)

- now, we can approximate the score (of the data) $\nabla_{\boldsymbol{x}} \log q(\boldsymbol{x}, \boldsymbol{t})$ in the reverse SDE

$$d\boldsymbol{z} = [\boldsymbol{f}(\boldsymbol{z}, t) - g(t)^2 \nabla_{\boldsymbol{x}} \log q(\boldsymbol{x}, t)]dt + g(t)d\boldsymbol{w} \tag{13}$$

  by $\nabla_{\boldsymbol{x}} \log q(\boldsymbol{x}, t) \approx \boldsymbol{s}_\theta(\boldsymbol{z}; \lambda)$
- the score network can be parameterized in various ways, based on the relationships between $\boldsymbol{z}_\lambda$, $\boldsymbol{x}$, and $\boldsymbol{\epsilon}$

$$\boldsymbol{z}_\lambda = \alpha_\lambda \boldsymbol{x} + \sigma_\lambda \boldsymbol{\epsilon} \tag{14}$$

$$\boldsymbol{x} = \alpha_\lambda^{-1}(\boldsymbol{z}_\lambda - \sigma_\lambda \boldsymbol{\epsilon}) \tag{15}$$

$$\boldsymbol{\epsilon} = \sigma_\lambda^{-1}(\boldsymbol{z}_\lambda - \alpha_\lambda \boldsymbol{x}) \tag{16}$$

$$\boldsymbol{s}_\theta(\boldsymbol{z}; \lambda) = -\nabla_{\boldsymbol{z}} E_\theta(\boldsymbol{z}, \lambda) \tag{17}$$

$$\boldsymbol{s}_\theta(\boldsymbol{z}; \lambda) = -\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}; \lambda)/\sigma_\lambda \tag{18}$$

$$\boldsymbol{s}_\theta(\boldsymbol{z}; \lambda) = -(\boldsymbol{z} - \alpha_\lambda \hat{\boldsymbol{x}}_\theta(\boldsymbol{z}; \lambda))/\sigma_\lambda^2 \tag{19}$$

$\boldsymbol{F}$-Prediction (Karras et al., 2022)

Last Week: Variance Exploding (VE) SDE Special Case

- $\boldsymbol{F}$-prediction model: In Karras et al., the $\boldsymbol{F}$-prediction model is used to parameterize the score network under the variance-exploding (VE) SDE
- $\boldsymbol{F}$-prediction Formula for VE SDE

$$\boldsymbol{x} = \frac{\tilde{\sigma}_{\text{data}}^2}{e^{-\lambda} + \tilde{\sigma}_{\text{data}}^2} \boldsymbol{z}_\lambda + \frac{e^{-\lambda/2} \tilde{\sigma}_{\text{data}}}{\sqrt{e^{-\lambda} + \tilde{\sigma}_{\text{data}}^2}} \boldsymbol{F} \tag{20}$$

$$\boldsymbol{F} = \frac{\sqrt{e^{-\lambda} + \tilde{\sigma}_{\text{data}}^2}}{e^{-\lambda/2} \tilde{\sigma}_{\text{data}}} \boldsymbol{x} - \frac{\tilde{\sigma}_{\text{data}} \alpha_\lambda}{e^{-\lambda/2} \sqrt{e^{-\lambda} + \tilde{\sigma}_{\text{data}}^2}} \boldsymbol{z}_\lambda \tag{21}$$

where $\tilde{\sigma}_{\text{data}} = 0.5$.

Variational Diffusion Models

Starting with Kingma et al. (2021)

Let's get probabilistic.

Evidence Lower Bound

Slowly approaching the paper for today

- the log evidence (marginal likelihood) of the data $\boldsymbol{x}$ under the model is

$$\log p(\boldsymbol{x}) = \log \int_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z} \tag{22}$$

  where $p(\boldsymbol{x}, \boldsymbol{z})$ is the joint distribution of the data $\boldsymbol{x}$ and latent variables $\boldsymbol{z}$.
- we can derive a lower bound (Jensen's inequality) on the log-evidence, introducing the variational distribution $q(\boldsymbol{z}|\boldsymbol{x})$

$$\log p(\boldsymbol{x}) = \log \int_{\boldsymbol{z}} q(\boldsymbol{z}|\boldsymbol{x}) \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q(\boldsymbol{z}|\boldsymbol{x})} d\boldsymbol{z} \geq \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q(\boldsymbol{z}|\boldsymbol{x})} \right] \tag{23}$$

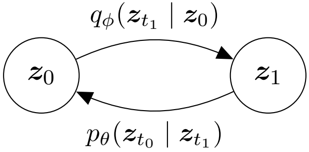
  This gives us the evidence lower bound (ELBO) (Blei et al., 2017)
- we rewrite the ELBO

$$\log p(\boldsymbol{x}) \geq \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})} \left[\log p(\boldsymbol{x}|\boldsymbol{z}) \right] - D_{KL}(q(\boldsymbol{z}|\boldsymbol{x}) || p(\boldsymbol{z})) \tag{24}$$

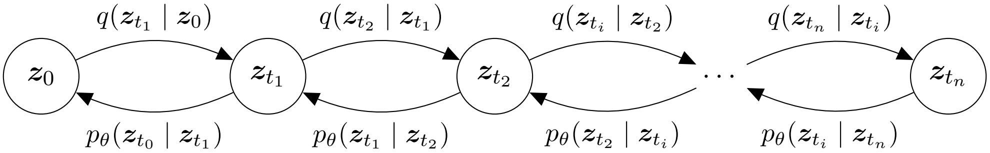
From Variational Autoencoders to Diffusion Models

Originally by Kingma & Welling (2014) and Rezende et al. (2014)

Variational Autoencoder



Diffusion Model



- We aim to learn a generative model $p_\theta(\boldsymbol{x})$ that approximates $q(\boldsymbol{x})$, where $\boldsymbol{x}$ is drawn from a dataset. (think: $\boldsymbol{x}$ image or latent representation)
- The model incorporates a sequence of latent variables $\boldsymbol{z}_t$ for $t \in [0, 1]$, where $\boldsymbol{z}_{0,\ldots,1} := \boldsymbol{z}_0, \ldots, \boldsymbol{z}_1$.
- forward process: defines a conditional distribution $q(\boldsymbol{z}_{0,\ldots,1}|\boldsymbol{x})$
- generative model: defines a joint distribution $p(\boldsymbol{z}_{0,\ldots,1})$

- forward diffusion process generates a sequence of increasingly noisy versions of the data $\boldsymbol{x}$, latent variables at time $t \in [0, 1]$ are denoted $\boldsymbol{z}_t$

$$q(\boldsymbol{z}_t|\boldsymbol{x}) = \mathcal{N}\left(\alpha_t \boldsymbol{x}, \sigma_t^2 \mathbf{I}\right) \quad \Rightarrow \quad \boldsymbol{z}_t = \alpha_t \boldsymbol{x} + \sigma_t^2 \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \tag{25}$$

where α_t and σ_t^2 are strictly positive scalar functions of time.
- Signal-to-Noise Ratio (SNR)

$$\text{SNR}(t) = \frac{\alpha_t^2}{\sigma_t^2} \tag{26}$$

- variance-preserving (VP): $\alpha_t = \sqrt{1 - \sigma_t^2}$, variance-exploding (VE): $\alpha_t^2 = 1$
- noise schedule σ_t^2 is learned via a monotonic neural network $\gamma(t)$

$$\sigma_t^2 = \text{sigmoid}(\gamma_{\boldsymbol{\eta}}(t)), \quad \alpha_t^2 = \text{sigmoid}(-\gamma_{\boldsymbol{\eta}}(t)), \quad \text{SNR}(t) = \exp(-\gamma_{\boldsymbol{\eta}}(t)) \tag{27}$$

Generative Model

Following Kingma et al. (2021)

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

- reverse diffusion process is the (hierarchical) generative model

$$p(\boldsymbol{x}) = \int_{\boldsymbol{z}} p(\boldsymbol{z}_1) p(\boldsymbol{x}|\boldsymbol{z}_0) \prod_{i=1}^{T} p(\boldsymbol{z}_{s(i)}|\boldsymbol{z}_{t(i)}), \tag{28}$$

  where $s(i) = (i-1)/T$, and $t(i) = i/T$.
- for $t = T$, with VP diffusion and sufficiently small $\mathrm{SNR}(1)$,

$$p(\boldsymbol{z}_1) = \mathcal{N}(\boldsymbol{z}_1; 0, \mathbf{I}), \tag{29}$$

- generative model approximates the (unknown) conditional distribution $q(\boldsymbol{x}|\boldsymbol{z}_0)$

$$p(\boldsymbol{x}|\boldsymbol{z}_0) = \prod_{i} p(x_i|z_{0,i}), \tag{30}$$

  where $p(x_i|z_{0,i}) \propto q(z_{0,i}|x_i)$, remember: $q(\boldsymbol{z}_t|\boldsymbol{x}) = \mathcal{N}\left(\alpha_t \boldsymbol{x}, \sigma_t^2 \mathbf{I}\right)$

Generative Model

Following Kingma et al. (2021)

- conditional model distributions

$$p(\boldsymbol{z}_s|\boldsymbol{z}_t) = q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{x} = \hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t; t)), \tag{31}$$

  where $\hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t; t)$ is the noise-prediction parameterization
- we see

$$q(\boldsymbol{z}_t|\boldsymbol{z}_s) = \mathcal{N}(\alpha_{t|s}\boldsymbol{z}_s, \sigma_{t|s}^2\mathbf{I}), \quad \alpha_{t|s} = \frac{\alpha_t}{\alpha_s}, \quad \sigma_{t|s}^2 = \sigma_t^2 - \alpha_{t|s}^2\sigma_s^2 \tag{32}$$

- posterior $q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{x}) \propto q(\boldsymbol{z}_s|\boldsymbol{x})q(\boldsymbol{z}_t|\boldsymbol{z}_s)$, we get

$$q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}_Q(\boldsymbol{z}_t, \boldsymbol{x}; s, t), \sigma_Q^2(s, t)\mathbf{I}), \tag{33}$$

  where

$$\sigma_Q^2(s, t) = \frac{\sigma_s^2\sigma_{t|s}^2}{\sigma_t^2}, \quad \boldsymbol{\mu}_Q(\boldsymbol{z}_t, \boldsymbol{x}; s, t) = \frac{\alpha_{t|s}\sigma_s^2}{\sigma_t^2}\boldsymbol{z}_t + \frac{\alpha_s\sigma_{t|s}^2}{\sigma_t^2}\boldsymbol{x}. \tag{34}$$

- for the generative model we replace $\boldsymbol{x}$ by the predicted $\hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t; t)$

$$p(\boldsymbol{z}_s|\boldsymbol{z}_t) = \mathcal{N}(\boldsymbol{\mu}_\theta(\boldsymbol{z}_t; s, t), \sigma_Q^2(s, t)\mathbf{I}), \tag{35}$$

- remember: ELBO

$$\log p(\boldsymbol{x}) \geq \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\left[\log p(\boldsymbol{x}|\boldsymbol{z})\right] - \mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})) \tag{36}$$

- minimize the negative ELBO

$$\begin{aligned} -\log p(\boldsymbol{x}) &\leq -\mathrm{VLB}(\boldsymbol{x}) \\ &= \underbrace{D_{KL}(q(\boldsymbol{z}_1|\boldsymbol{x})||p(\boldsymbol{z}_1))}_{\text{Prior loss}} + \underbrace{\mathbb{E}_{q(\boldsymbol{z}_0|\boldsymbol{x})}\left[-\log p(\boldsymbol{x}|\boldsymbol{z}_0)\right]}_{\text{Reconstruction loss}} + \underbrace{\mathcal{L}_T}_{\text{Diffusion loss}} \end{aligned} \tag{37}$$

- for finite T, the diffusion loss can be expressed as

$$\mathcal{L}_T = \sum_{i=1}^{T} \mathbb{E}_{q(\boldsymbol{z}_{t(i)}|\boldsymbol{x})} D_{KL}[q(\boldsymbol{z}_{s(i)}|\boldsymbol{z}_{t(i)}, \boldsymbol{x})||p_\theta(\boldsymbol{z}_{s(i)}|\boldsymbol{z}_{t(i)})] \tag{38}$$

- we can write

$$\mathcal{L}_T = \sum_{i=1}^{T} \mathbb{E}_{q(\boldsymbol{z}_{t(i)}|\boldsymbol{x})} D_{KL}[q(\boldsymbol{z}_{s(i)}|\boldsymbol{z}_{t(i)}, \boldsymbol{x})||p_\theta(\boldsymbol{z}_{s(i)}|\boldsymbol{z}_{t(i)})] \tag{39}$$

$$= \frac{T}{2} \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(0,\mathbf{I}), i \sim U\{1,T\}} \left[(\mathrm{SNR}(s) - \mathrm{SNR}(t)) \, ||\boldsymbol{x} - \hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t; t)||_2^2 \right] \tag{40}$$

- here, $\mathcal{L}_T$ is an unbiased Monte Carlo estimator, where
  - $U\{1,T\}$ is the uniform distribution over $\{1, \dots, T\}$,
  - $s = (i-1)/T$, $t = i/T$,
  - $\boldsymbol{z}_t = \alpha_t \boldsymbol{x} + \sigma_t \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$.

Variational Lower Bound

A Closer Look on the Connection to the SNR

- **KL Divergence between Gaussians:**

$$D_{KL}(q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{x})||p_\theta(\boldsymbol{z}_s|\boldsymbol{z}_t)) = \frac{1}{2\sigma_Q^2(s,t)}||\boldsymbol{\mu}_Q - \boldsymbol{\mu}_\theta||_2^2 \tag{41}$$

- **Mean Difference** $||\boldsymbol{\mu}_Q - \boldsymbol{\mu}_\theta||_2^2$:

$$\boldsymbol{\mu}_Q - \boldsymbol{\mu}_\theta = \frac{\alpha_s^2}{\sigma_s^2}(\boldsymbol{x} - \hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t; t)) \tag{42}$$

- **Simplification of KL Divergence**:

$$D_{KL}(q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{x})||p_\theta(\boldsymbol{z}_s|\boldsymbol{z}_t)) = \frac{1}{2}\frac{\alpha_s^2}{\sigma_s^2}\left(1 - \frac{\alpha_t^2}{\sigma_t^2}\right)||\boldsymbol{x} - \hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t; t)||_2^2 \tag{43}$$

- **Final Expression with SNR:**

$$D_{KL}(q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{x})||p_\theta(\boldsymbol{z}_s|\boldsymbol{z}_t)) = \frac{1}{2}\left(\mathsf{SNR}(s) - \mathsf{SNR}(t)\right)||\boldsymbol{x} - \hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t; t)||_2^2 \tag{44}$$

From Discrete to Continuous

The Limit $T \to \infty$

- in the limit as $T \to \infty$, the diffusion loss $\mathcal{L}_T$ can be written as a function of $\tau = 1/T$

$$\mathcal{L}_T = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),i\sim U\{1,T\}}\left[\frac{\text{SNR}(t-\tau)-\text{SNR}(t)}{\tau}||\boldsymbol{x}-\hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t;t)||_2^2\right] \tag{45}$$

- as $\tau \to 0$ and $T \to \infty$, we get

$$\mathcal{L}_\infty = -\frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),t\sim\mathcal{U}[0,1]}\left[\text{SNR}'(t)||\boldsymbol{x}-\hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t;t)||_2^2\right] \tag{46}$$

- The continuous-time formulation of the diffusion loss is then:

$$\mathcal{L}_\infty = -\frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\int_0^1 \text{SNR}'(t)||\boldsymbol{x}-\hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t;t)||_2^2\,dt \tag{47}$$

- we know: the signal-to-noise ratio (SNR) function is invertible: $t = \text{SNR}^{-1}(v)$.
- therefore, by a change of variables (t to v) we can express the diffusion loss as

$$\mathcal{L}_\infty = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})} \int_{\text{SNR}_{\text{min}}}^{\text{SNR}_{\text{max}}} \|\boldsymbol{x} - \tilde{\boldsymbol{x}}_\theta(\boldsymbol{z}_v, v)\|_2^2 \, dv \tag{48}$$

- shape of SNR function between $t = 0$ and $t = 1$ does not affect the diffusion loss; only the values at the endpoints (SNR_{min} and SNR_{max}) matter
- two different diffusion processes with the same SNR_{min} and SNR_{max} will define the same distribution $p(\boldsymbol{x})$, up to a rescaling of the latents
- therefore, variance-preserving and variance-exploding diffusion models are equivalent in continuous time

- The continuous-time diffusion loss can be generalized to a weighted loss (no equivalence)

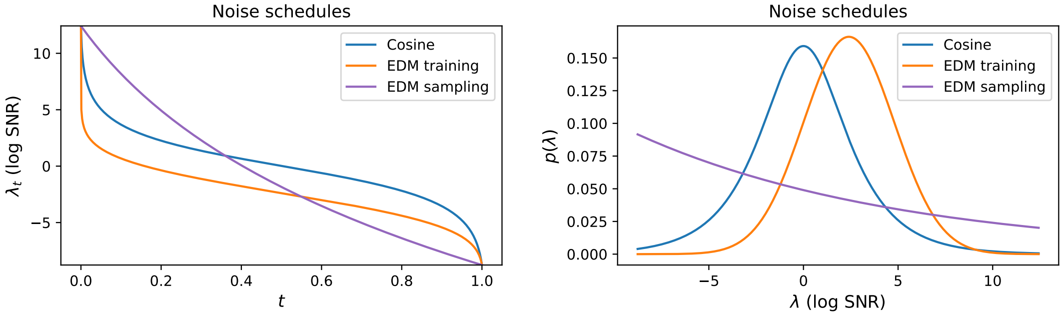
$$\mathcal{L}_\infty(\boldsymbol{x}, w) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})} \int_{\text{SNR}_{\text{min}}}^{\text{SNR}_{\text{max}}} w(v)\,\|\boldsymbol{x} - \tilde{\boldsymbol{x}}_\theta(\boldsymbol{z}_v, v)\|_2^2\, dv \tag{49}$$

- $w(v)$ is a weighting function that can emphasize different noise levels
- different objectives in diffusion models are special cases of this weighted diffusion loss

Now, let’s discuss Kingma & Gao (2024).

Noise Schedule
Gaussian Diffusion Process with Noise Scheduling

- Noise schedule is a strictly monotonically decreasing function $\lambda = f_\lambda(t)$ with endpoints $\lambda_{\text{max}} := f_\lambda(0)$ and $\lambda_{\text{min}} := f_\lambda(1)$.



Noise Schedules

Deriving the PDF from the Noise Schedule

- During training, time t is sampled uniformly: $t \sim \mathcal{U}(0, 1)$. The noise level λ is then computed via $\lambda = f_\lambda(t)$.
- This results in a distribution over noise levels, with the probability density function (PDF) given by:

$$p(\lambda_t) = -\frac{d}{d\lambda} f_\lambda^{-1}(\lambda_t) = -\frac{dt}{d\lambda} = -\frac{1}{f'_\lambda(t)}$$

- The PDF $p(\lambda_t)$ represents the likelihood of encountering a particular noise level λ during training.

Noise schedule name	$\lambda = f_\lambda(t) = \ldots$	$t = f_\lambda^{-1}(\lambda) = \ldots$	$p(\lambda) = -\frac{d}{d\lambda} f_\lambda^{-1}(\lambda)$
Cosine	$-2\log(\tan(\pi t/2))$	$(2/\pi)\arctan(e^{-\lambda/2})$	$\text{sech}(\lambda/2)/(2\pi)$
Shifted Cosine	$-2\log(\tan(\pi t/2)) + 2s$	$(2/\pi)\arctan(e^{-\lambda/2-s})$	$\text{sech}(\lambda/2 - s)/(2\pi)$
EDM (Training)	$-F_\mathcal{N}^{-1}(t; 2.4, 2.4^2)$	$F_\mathcal{N}(-\lambda; 2.4, 2.4^2)$	$\mathcal{N}(\lambda; 2.4, 2.4^2)$
EDM (Sampling)	$-2\rho\log(\sigma_{\max}^{1/\rho} + (1-t)(\sigma_{\min}^{1/\rho} - \sigma_{\max}^{1/\rho}))$	$1 - \frac{e^{-\lambda/(2\rho)} - \sigma_{\max}^{1/\rho}}{\sigma_{\min}^{1/\rho} - \sigma_{\max}^{1/\rho}}$	$\frac{e^{-\lambda/(2\rho)}}{2\rho(\sigma_{\max}^{1/\rho} - \sigma_{\min}^{1/\rho})}$
Flow Matching (OT)	$2\log((1-t)/t)$	$1/(1+e^{\lambda/2})$	$\text{sech}^2(\lambda/4)/8$

Negative ELBO and ϵ Prediction

Notation and Setup for (Kingma & Gao, 2024)

- we define the log signal-to-noise ratio (SNR) as

$$\lambda = \log(\alpha_\lambda^2 / \sigma_\lambda^2) \tag{50}$$

- therefore, we can write $\mathcal{L}_\infty$ as

$$\mathcal{L}_\infty = -\frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),t\sim\mathcal{U}[0,1]}\left[\text{SNR}'(t)||\boldsymbol{x} - \hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t;t)||_2^2\right] \tag{51}$$

$$= -\frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),t\sim\mathcal{U}[0,1]}\left[\frac{d\lambda}{dt}||\boldsymbol{x} - \hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t;t)||_2^2\right] \tag{52}$$

- we can similarly use the noise prediction parameterization

$$-\text{ELBO}(\boldsymbol{x}) = \frac{1}{2}\mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I})}\left[-\frac{d\lambda}{dt}\cdot||\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_t;\lambda_t) - \boldsymbol{\epsilon}||_2^2\right] + c \tag{53}$$

Negative ELBO and Further Connections

Reiterating Various Parameterizations

- noise prediction parameterization

$$-\text{ELBO}(\boldsymbol{x}) = \frac{1}{2}\mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I})}\left[-\frac{d\lambda}{dt}\cdot\|\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_t;\lambda_t)-\boldsymbol{\epsilon}\|_2^2\right]+c \tag{54}$$

- using

$$\boldsymbol{s}_\theta(\boldsymbol{z}_t;\lambda_t) = -\frac{\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_t;\lambda_t)}{\sigma_\lambda}$$

and $\tilde{w}(t) = \sigma_t^2$ we can connect

$$\mathcal{L}_{\text{DSM}}(\boldsymbol{x}) = \mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I})}\left[\tilde{w}(t)\cdot\|\boldsymbol{s}_\theta(\boldsymbol{z}_t,\lambda_t)-\nabla_{\boldsymbol{z}_t}\log q(\boldsymbol{z}_t|\boldsymbol{x})\|_2^2\right] \tag{55}$$

to

$$\mathcal{L}_{\boldsymbol{\epsilon}}(\boldsymbol{x}) = \frac{1}{2}\mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I})}\left[\|\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_t;\lambda_t)-\boldsymbol{\epsilon}\|_2^2\right] \tag{56}$$

- general form of the weighted loss is

$$\mathcal{L}_w(\boldsymbol{x}) = \frac{1}{2}\mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I})}\left[w(\lambda_t)\cdot\left(-\frac{d\lambda}{dt}\right)\cdot\|\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_t;\lambda_t)-\boldsymbol{\epsilon}\|_2^2\right] \tag{57}$$

- weighted loss can be rewritten as an integral

$$\mathcal{L}_w(\boldsymbol{x}) = \frac{1}{2}\int_{\lambda_{\min}}^{\lambda_{\max}} w(\lambda)\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I})}\left[\|\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_\lambda;\lambda)-\boldsymbol{\epsilon}\|_2^2\right]d\lambda \tag{58}$$

- loss does not depend on the specific noise schedule λ_t except for the endpoints $\lambda_{\min}$ and $\lambda_{\max}$, noise schedule (only!) affects the variance of the Monte Carlo estimator

- the noise schedule $p(\lambda)$ acts as an *importance sampling distribution*

$$\mathcal{L}_w(\boldsymbol{x}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),\lambda\sim p(\lambda)}\left[\frac{w(\lambda_t)}{p(\lambda)}\|\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_t;\lambda_t)-\boldsymbol{\epsilon}\|_2^2\right] \tag{59}$$

Theorem

If the weighting function $w(\lambda_t)$ is monotonic, then the weighted diffusion objective is equivalent to the ELBO with data augmentation (additive noise).

Result: Any objective with (implied) monotonic weighting, can be uunderstood as equivalent to the ELBO with simple data augmentation (additive noise)

- adaptive noise schedule: by lowering the variance of the loss estimator, this often significantly speeds up optimization

Proof of Theorem 1 (Kingma & Gao, 2024)

KL Divergence and Time Derivative

- **KL Divergence**: define $\mathcal{L}(t;\boldsymbol{x}) = D_{KL}(q(\boldsymbol{z}_{t,\dots,1}|\boldsymbol{x})||p(\boldsymbol{z}_{t,\dots,1}))$ for the KL divergence between $q(\boldsymbol{z}_{t,\dots,1}|\boldsymbol{x})$ and $p(\boldsymbol{z}_{t,\dots,1})$ for timesteps t to 1

$$\mathcal{L}(t;\boldsymbol{x}) := D_{KL}q(\boldsymbol{z}_{t,\dots,1}|\boldsymbol{x})||p(\boldsymbol{z}_{t,\dots,1})) \tag{60}$$

- **Time Derivative**: show that the time derivative of $\mathcal{L}(t;\boldsymbol{x})$ is

$$\frac{d}{dt}\mathcal{L}(t;\boldsymbol{x}) = \frac{1}{2}\frac{d\lambda}{dt}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\left[\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{z}_t;\lambda_t)\|_2^2\right] \tag{61}$$

- **Rewriting the Weighted Loss**: therefore, the weighted loss is rewritten as

$$\mathcal{L}_w(\boldsymbol{x}) = -\int_0^1 \frac{d}{dt}\mathcal{L}(t;\boldsymbol{x})\, w(\lambda_t)\, dt \tag{62}$$

Proof of Theorem 1 (Kingma & Gao, 2024)

Integration by Parts and Monotonic Weighting

- weighted loss

$$\mathcal{L}_w(\boldsymbol{x}) = -\int_0^1 \frac{d}{dt}\mathcal{L}(t;\boldsymbol{x})\, w(\lambda_t)\; dt \tag{63}$$

- **Integration by Parts**: integration by parts yields

$$\mathcal{L}_w(\boldsymbol{x}) = \int_0^1 \frac{d}{dt} w(\lambda_t)\, \mathcal{L}(t;\boldsymbol{x})\; dt + w(\lambda_{\text{max}})\, \mathcal{L}(0;\boldsymbol{x}) + \text{constant} \tag{64}$$

- **Monotonic Weighting**: assume $w(\lambda_t)$ is monotonically increasing and normalized

$$\mathcal{L}_w(\boldsymbol{x}) = \mathbb{E}_{p_w(t)}\left[\mathcal{L}(t;\boldsymbol{x})\right] + \text{constant} \tag{65}$$

- **Probability Distribution**: $p_w(t) = \frac{d}{dt} w(\lambda_t)$ is a probability distribution over $t \in [0,1]$, meaning that $\mathcal{L}_w(\boldsymbol{x})$ becomes an expected KL divergence.

- we note that

$$\mathcal{L}(t;\boldsymbol{x}) = D_{KL}(q(\boldsymbol{z}_{t,\dots,1}|\boldsymbol{x})||p(\boldsymbol{z}_{t,\dots,1})) = -\mathbb{E}_{q(\boldsymbol{z}_t|\boldsymbol{x})}[\text{ELBO}_t(\boldsymbol{z}_t)] - \mathcal{H}(q(\boldsymbol{z}_t|\boldsymbol{x})) \quad (66)$$

$$\text{ELBO}_t(\boldsymbol{z}_t) := \mathbb{E}_{q(\tilde{\boldsymbol{z}}_t|\boldsymbol{z}_t)}[\log p(\boldsymbol{z}_t, \tilde{\boldsymbol{z}}_t) - \log q(\tilde{\boldsymbol{z}}_t|\boldsymbol{z}_t)] \leq \log p(\boldsymbol{z}_t), \tilde{\boldsymbol{z}}_t := \boldsymbol{z}_{t+dt,\dots,1} \quad (67)$$

- therefore,

$$\mathcal{L}(t;\boldsymbol{x}) = D_{KL}(q(\boldsymbol{z}_{t,\dots,1}|\boldsymbol{x})||p(\boldsymbol{z}_{t,\dots,1})) \geq D_{KL}(q(\boldsymbol{z}_t|\boldsymbol{x})||p(\boldsymbol{z}_t)) \quad (68)$$

- $\mathcal{L}(t;\boldsymbol{x})$ is the expected negative ELBO of noise-perturbed data $\boldsymbol{z}_t$

$$\mathcal{L}(t;\boldsymbol{x}) = -\mathbb{E}_{q(\boldsymbol{z}_t|\boldsymbol{x})}[\text{ELBO}_t(\boldsymbol{z}_t)] + \text{c.} \geq -\mathbb{E}_{q(\boldsymbol{z}_t|\boldsymbol{x})}[\log p(\boldsymbol{z}_t)] + \text{c.} \quad (69)$$

$$\begin{aligned}\mathcal{L}_w(\boldsymbol{x}) &= \mathbb{E}_{p_w(t)}\left[\mathcal{L}(t;\boldsymbol{x})\right] + \text{c.} \\ &= -\underbrace{\mathbb{E}_{p_w(t),q(\boldsymbol{z}_t|\boldsymbol{x})}\left[\text{ELBO}_t(\boldsymbol{z}_t)\right]}_{\text{ELBO of noise-perturbed data}} + \text{c.} \geq -\underbrace{\mathbb{E}_{p_w(t),q(\boldsymbol{z}_t|\boldsymbol{x})}\left[\log p(\boldsymbol{z}_t)\right]}_{\text{ll. of noise-perturbed data}} + \text{c.}\end{aligned} \quad (70)$$

				DDPM sampler		EDM sampler	
Model parameterization	**Training noise schedule**	**Weighting function**	**Monotonic?**	**FID** $\downarrow$	**IS** $\uparrow$	**FID** $\downarrow$	**IS** $\uparrow$
$\boldsymbol{\epsilon}$-prediction	Cosine	$\mathrm{sech}(\lambda/2)$ (Baseline)		1.85	54.1 ± 0.79	1.55	59.2 ± 0.78
"	Cosine	sigmoid$(-\lambda+1)$	✓	1.75	55.3 ± 1.23		
"	Cosine	sigmoid$(-\lambda+2)$	✓	**1.68**	**56.8** ± 0.85	1.46	60.4 ± 0.86
"	Cosine	sigmoid$(-\lambda+3)$	✓	1.73	56.1 ± 1.36		
"	Cosine	sigmoid$(-\lambda+4)$	✓	1.80	55.1 ± 1.65		
"	Cosine	sigmoid$(-\lambda+5)$	✓	1.94	53.5 ± 1.12		
"	Adaptive	sigmoid$(-\lambda+2)$	✓	1.70	54.8 ± 1.20	**1.44**	60.6 ± 1.44
"	Adaptive	EDM-monotonic	✓	**1.67**	**56.8** ± 0.90	**1.44**	**61.1** ± 1.80
EDM (Karras et al., 2022)	EDM (training)	EDM (Baseline)				1.36	
EDM (our reproduction)	EDM (training)	EDM (Baseline)				1.45	60.7 ± 1.19
"	Adaptive	EDM				**1.43**	63.2 ± 1.76
"	Adaptive	sigmoid$(-\lambda+2)$	✓			1.55	**63.7** ± 1.14
"	Adaptive	EDM-monotonic	✓			**1.43**	**63.7** ± 1.48
$\boldsymbol{v}$-prediction	Adaptive	$\exp(-\lambda/2)$ (Baseline)	✓			1.62	58.0 ± 1.56
"	Adaptive	sigmoid$(-\lambda+2)$	✓			1.51	64.4 ± 1.28
"	Adaptive	EDM-monotonic	✓			**1.45**	**64.6** ± 1.35

Model parameterization	Training noise schedule	Weighting function	Monotonic?	FID ↓ train	FID ↓ eval	IS ↑
$\boldsymbol{v}$-prediction	Cosine-shifted	$\exp(-\lambda/2)$ (Baseline)	✓	1.91	3.23	171.9 ± 2.46
"	Adaptive	sigmoid$(-\lambda + 2)$-shifted	✓	1.91	3.41	**183.1** $\pm$ 2.20
"	Adaptive	EDM-monotonic-shifted	✓	**1.75**	**2.88**	171.1 ± 2.67

Results
High-Resolution ImageNet

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Method	Without guidance FID ↓ train	eval	IS ↑	With guidance FID ↓ train	eval	IS ↑
128 × 128 resolution						
ADM (Dhariwal & Nichol, 2021)	5.91	–	–	2.97	–	–
CDM (Ho et al., 2022)	3.52	3.76	128.8 ± 2.5	–	–	–
RIN (Jabri et al., 2023)	2.75	–	144.1	–	–	–
Simple Diffusion (U-Net) (Hoogeboom et al., 2023)	2.26	**2.88**	137.3 ± 2.0	–	–	–
Simple Diffusion (U-ViT, L) (Hoogeboom et al., 2023)	1.91	3.23	171.9 ± 2.5	2.05	3.57	189.9 ± 3.5
VDM++ (Ours), $w(\lambda) = \text{sigmoid}(-\lambda + 2)$	1.91	3.41	**183.1** ± 2.2	–	–	–
VDM++ (Ours), EDM-monotonic weighting	**1.75**	**2.88**	171.1 ± 2.7	**1.78**	**3.16**	**190.5** ± 2.3
256 × 256 resolution						
BigGAN-deep (no truncation) (Brock, 2018)	6.9	–	171.4 ± 2.0	–	–	–
MaskGIT (Chang et al., 2022)	6.18	–	182.1	–	–	–
ADM (Dhariwal & Nichol, 2021)	10.94	–	–	3.94	–	215.9
CDM (Ho et al., 2022)	4.88	4.63	158.7 ± 2.3	–	–	–
RIN (Jabri et al., 2023)	3.42	–	182.0	–	–	–
Simple Diffusion (U-Net) (Hoogeboom et al., 2023)	3.76	3.71	171.6 ± 3.1	–	–	–
Simple Diffusion (U-ViT, L) (Hoogeboom et al., 2023)	2.77	3.75	211.8 ± 2.9	2.44	4.08	256.3 ± 5.0
VDM++ (Ours), EDM-monotonic weighting	**2.40**	**3.36**	**225.3** ± 3.2	**2.12**	**3.69**	**267.7** ± 4.9

Results
High-Resolution ImageNet

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Method	Without guidance FID ↓ train	eval	IS ↑	With guidance FID ↓ train	eval	IS ↑
Latent diffusion with pretrained VAE:						
DiT-XL/2 (Peebles & Xie, 2023)	9.62	–	121.5	2.27	–	278.2
U-ViT (Bao et al., 2023)	–	–	–	3.40	–	–
Min-SNR-γ (Hang et al., 2023)	–	–	–	2.06	–	–
MDT (Gao et al., 2023)	6.23	–	143.0	1.79	–	283.0
512 $\times$ 512 resolution						
MaskGIT (Chang et al., 2022)	7.32	–	156.0	–	–	–
ADM (Dhariwal & Nichol, 2021)	23.24	–	–	3.85	–	221.7
RIN (Jabri et al., 2023)	–	–	–	3.95	–	216.0
Simple Diffusion (U-Net) (Hoogeboom et al., 2023)	4.30	4.28	171.0 $\pm$ 3.0	–	–	–
Simple Diffusion (U-ViT, L) (Hoogeboom et al., 2023)	3.54	4.53	205.3 $\pm$ 2.7	3.02	4.60	248.7 $\pm$ 3.4
VDM++ (Ours), EDM-monotonic weighting	**2.99**	**4.09**	**232.2** $\pm$ 4.2	**2.65**	**4.43**	**278.1** $\pm$ 5.5
Latent diffusion with pretrained VAE:						
DiT-XL/2 (Peebles & Xie, 2023)	12.03	–	105.3	3.04	–	240.8
LDM-4 (Rombach et al., 2022)	10.56	–	103.5 $\pm$ 1.2	3.60	–	247.7 $\pm$ 5.6

Conclusion and Discussion

Summary of Results:

- We have demonstrated that the **weighted diffusion loss** generalizes diffusion objectives in the literature.
- It can be interpreted as a **weighted integral of ELBO objectives**, where each ELBO corresponds to a different noise level.
- When the weighting function $w(\lambda_t)$ is monotonic, the loss has an interpretation as the ELBO objective with **data augmentation** via noise perturbation.

Implications:

- The equivalence between **monotonic weighting** and the **ELBO with data augmentation** enables a direct comparison between diffusion models and other **likelihood-based models** such as autoregressive transformers.
- This opens up avenues for optimizing other model types toward the same objective as monotonically weighted diffusion models.

Discussion