
Soft Truncation: A Universal Training Technique of Score-based Diffusion Model for High Precision Score Estimation

Dongjun Kim¹ Seungjae Shin¹ Kyungwoo Song² Wanmo Kang¹ Il-Chul Moon^{1,3}

Abstract

Recent advances in diffusion models bring state-of-the-art performance on image generation tasks. However, empirical results from previous research in diffusion models imply an inverse correlation between density estimation and sample generation performances. This paper investigates with sufficient empirical evidence that such inverse correlation happens because density estimation is significantly contributed by small diffusion time, whereas sample generation mainly depends on large diffusion time. However, training a score network well across the entire diffusion time is demanding because the loss scale is significantly imbalanced at each diffusion time. For successful training, therefore, we introduce Soft Truncation, a universally applicable training technique for diffusion models, that softens the fixed and static truncation hyperparameter into a random variable. In experiments, Soft Truncation achieves state-of-the-art performance on CIFAR-10, CelebA, CelebA-HQ 256 × 256, and STL-10 datasets.

1. Introduction

Recent advances in generative models enable the creation of highly realistic images. One direction of such modeling is *likelihood-free models* (Karras et al., 2019) based on minimax training. The other direction is *likelihood-based models*, including VAE (Vahdat & Kautz, 2020), autoregressive models (Parmar et al., 2018), and flow models (Grcić et al., 2021). Diffusion models (Ho et al., 2020) are one of the most successful *likelihood-based models*, where the reverse diffusion models the generative process. The success of diffusion models achieves state-of-the-art performance in image generation (Dhariwal & Nichol, 2021).

¹KAIST, South Korea ²University of Seoul, South Korea ³Summary.AI. Correspondence to: Dongjun Kim <dongjoun57@kaist.ac.kr>.

Previously, a model with the emphasis on Fréchet Inception Distance (FID), such as DDPM (Ho et al., 2020) and ADM (Dhariwal & Nichol, 2021), trains the score network with the variance weighting; whereas a model with the emphasis on Negative Log-Likelihood (NLL), such as ScoreFlow (Song et al., 2021a) and VDM (Kingma et al., 2021), trains the score network with the likelihood weighting. Such models, however, have the trade-off between NLL and FID: models with the emphasis on FID perform poorly on NLL, and vice versa. Instead of widely investigating the trade-off, they limit their work by separately training the score network on FID-favorable and NLL-favorable settings. This paper introduces Soft Truncation that significantly resolves the trade-off, with the NLL-favorable setting as the default training configuration. Soft Truncation reports a comparable FID against FID-favorable diffusion models while keeping NLL at the equivalent level of NLL-favorable models.

For that, we observe that the truncation hyperparameter is a significant hyperparameter that determines the overall scale of NLL and FID. This hyperparameter, ϵ , is the smallest diffusion time to estimate the score function, and the score function beneath ϵ is not estimated. A model with small enough ϵ favors NLL at the sacrifice on FID, and a model with relatively large ϵ is preferable to FID but has poor NLL. Therefore, we introduce Soft Truncation, which softens the fixed and static truncation hyperparameter (ϵ) into a random variable (τ) that randomly selects its smallest diffusion time at every optimization step. In every mini-batch update, we sample a new smallest diffusion time, τ , randomly, and the batch optimization endeavors to estimate the score function only on $[\tau, T]$, rather than $[\epsilon, T]$, by ignoring beneath τ . As τ varies by mini-batch updates, the score network successfully estimates the score function on the entire range of diffusion time on $[\epsilon, T]$, which brings an improved FID.

There are two interesting properties of Soft Truncation. First, though Soft Truncation is nothing to do with the weighting function in its algorithmic design, surprisingly, Soft Truncation turns out to be equivalent to a diffusion model with a *general weight* in the expectation sense (Eq. (10)). The random variable of τ determines the weight function (Theorem 1), and this gives a partial reason why Soft Truncation is successful in FID as much as the FID-favorable train-

ing (Table 4), even though Soft Truncation only considers the truncation threshold in its implementation (Section 4.2). Second, once τ is sampled in a mini-batch optimization, Soft Truncation optimizes the log-likelihood *perturbed* by τ (Lemma 1). Thus, Soft Truncation could be framed by Maximum Perturbed Likelihood Estimation (MPLE), a generalized concept of MLE that is specifically defined only in diffusion models (Section 4.4).

2. Preliminary

Throughout this paper, we focus on continuous-time diffusion models (Song et al., 2021b). A continuous diffusion model slowly and systematically perturbs a data random variable, \mathbf{x}_0 , into a noise variable, \mathbf{x}_T , as time flows. The diffusion mechanism is represented as a Stochastic Differential Equation (SDE), written by

$$d\mathbf{x}_t = \mathbf{f}(\mathbf{x}_t, t) dt + g(t) d\mathbf{w}_t, \quad (1)$$

where \mathbf{w}_t is a standard Wiener process. The drift (\mathbf{f}) and the diffusion (g) terms are fixed, so the data variable is diffused in a fixed manner. We denote $\{\mathbf{x}_t\}_{t=0}^T$ as the solution of the given SDE of Eq. (1), and we omit the subscript and superscript to denote $\{\mathbf{x}_t\}$, if no confusion is arised.

The theory of stochastic calculus indicates that there exists a corresponding reverse SDE given by

$$d\bar{\mathbf{x}}_t = [\mathbf{f}(\mathbf{x}_t, t) - g^2(t) \nabla \log p_t(\mathbf{x}_t)] d\bar{t} + g(t) d\bar{\mathbf{w}}_t, \quad (2)$$

where the solution of this reverse SDE exactly coincides to the solution of the forward SDE of Eq. (1). Here, $d\bar{t}$ is the backward time differential; $d\bar{\mathbf{w}}_t$ is a standard Wiener process flowing backward in time (Anderson, 1982); and $p_t(\mathbf{x}_t)$ is the probability distribution of \mathbf{x}_t . Henceforth, we represent $\{\mathbf{x}_t\}$ as the solution of SDEs of Eqs. (1) and (2).

The diffusion model’s objective is to *learn* the stochastic process, $\{\mathbf{x}_t\}$, as a parametrized stochastic process, $\{\mathbf{x}_t^\theta\}$. A diffusion model builds the parametrized stochastic process as a solution of a generative SDE,

$$d\mathbf{x}_t^\theta = [\mathbf{f}(\mathbf{x}_t^\theta, t) - g^2(t) \mathbf{s}_\theta(\mathbf{x}_t^\theta, t)] d\bar{t} + g(t) d\bar{\mathbf{w}}_t. \quad (3)$$

We construct the parametrized stochastic process by solving the generative SDE of Eq. (3) backward in time with a starting variable of $\mathbf{x}_T^\theta \sim \pi$, where π is an noise distribution. Throughout the paper, we denote p_t^θ as the probability distribution of \mathbf{x}_t^θ .

A diffusion model learns the generative stochastic process by minimizing the score loss (Song et al., 2021a) of

$$\mathcal{L}(\theta; \lambda) = \frac{1}{2} \int_0^T \lambda(t) \mathbb{E}_{\mathbf{x}_t} [\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_t(\mathbf{x}_t)\|_2^2] dt,$$

where $\lambda(t)$ is a weighting function that counts the contribution of each diffusion time on the loss function. This

score loss is infeasible to optimize because the data score, $\nabla \log p_t(\mathbf{x}_t)$, is intractable in general. Fortunately, $\mathcal{L}(\theta; \lambda)$ is known to be equivalent to the (continuous) denoising NCSN loss (Song et al., 2021b; Song & Ermon, 2019),

$$\begin{aligned} & \mathcal{L}_{NCSN}(\theta; \lambda) \\ &= \frac{1}{2} \int_0^T \lambda(t) \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t} [\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] dt, \end{aligned}$$

up to a constant that is irrelevant to θ -optimization.

Two important SDEs are known to attain analytic transition probabilities, $\log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)$: Variance Exploding SDE (VESDE) and Variance Preserving SDE (VPSDE) (Song et al., 2021b). First, VESDE assumes $\mathbf{f}(\mathbf{x}_t, t) = 0$ and $g(t) = \sigma_{min} (\frac{\sigma_{max}}{\sigma_{min}})^t \sqrt{2 \log \frac{\sigma_{max}}{\sigma_{min}}}$. With such specific forms of \mathbf{f} and g , the transition probability of VESDE turns out to follow a Gaussian distribution of $p_{0t}(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \mu_{VE}(t)\mathbf{x}_0, \sigma_{VE}^2(t)\mathbf{I})$ with $\mu_{VE}(t) \equiv 1$ and $\sigma_{VE}^2(t) = \sigma_{min}^2 [(\frac{\sigma_{max}}{\sigma_{min}})^{2t} - 1]$. Similarly, VPSDE takes $\mathbf{f}(\mathbf{x}_t, t) = -\frac{1}{2}\beta(t)\mathbf{x}_t$ and $g(t) = \sqrt{\beta(t)}$, where $\beta(t) = \beta_{min} + t(\beta_{max} - \beta_{min})$; and its transition probability falls into a Gaussian distribution of $p_{0t}(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \mu_{VP}(t)\mathbf{x}_0, \sigma_{VP}^2\mathbf{I})$ with $\mu_{VP}(t) = e^{-\frac{1}{2} \int_0^t \beta(s) ds}$ and $\sigma_{VP}^2(t) = 1 - e^{-\int_0^t \beta(s) ds}$.

Recently, Kim et al. (2022) categorize VESDE and VPSDE as a family of linear diffusions that has the SDE of

$$d\mathbf{x}_t = -\frac{1}{2}\beta(t)\mathbf{x}_t dt + g(t) d\mathbf{w}_t, \quad (4)$$

where $\beta(t)$ and $g(t)$ are generic t -functions. Under the linear diffusions, we derive the transition probability to follow a Gaussian distribution $p_{0t}(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \mu(t)\mathbf{x}_0, \sigma^2(t)\mathbf{I})$ for certain $\mu(t)$ and $\sigma(t)$ depending on $\beta(t)$ and $g(t)$, respectively (see Eq. (16) of Appendix A.1). We emphasize that the suggested Soft Truncation is applicable for any SDE of Eq. (1), but we limit our focus to the family of linear SDEs of Eq. (4), particularly VESDE and VPSDE among linear SDEs, to maintain the simplicity. With such a Gaussian transition probability, the denoising NCSN loss with a linear SDE is equivalent to

$$\frac{1}{2} \int_0^T \frac{\lambda(t)}{\sigma^2(t)} \mathbb{E}_{\mathbf{x}_0, \epsilon} [\|\epsilon_\theta(\mu(t)\mathbf{x}_0 + \sigma(t)\epsilon, t) - \epsilon\|_2^2] dt,$$

if $\epsilon_\theta(\mu(t)\mathbf{x}_0 + \sigma(t)\epsilon, t) = -\sigma(t)\mathbf{s}_\theta(\mu(t)\mathbf{x}_0 + \sigma(t)\epsilon, t)$, where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ is a random perturbation, and ϵ_θ is the neural network that predicts ϵ . This is the (continuous) DDPM loss (Song et al., 2021b), and the equivalence of the two losses provides a unified view of NCSN and DDPM. Hence, NCSN and DDPM are exchangeable for each other, and we take the NCSN loss as a default form of a diffusion loss throughout the paper.

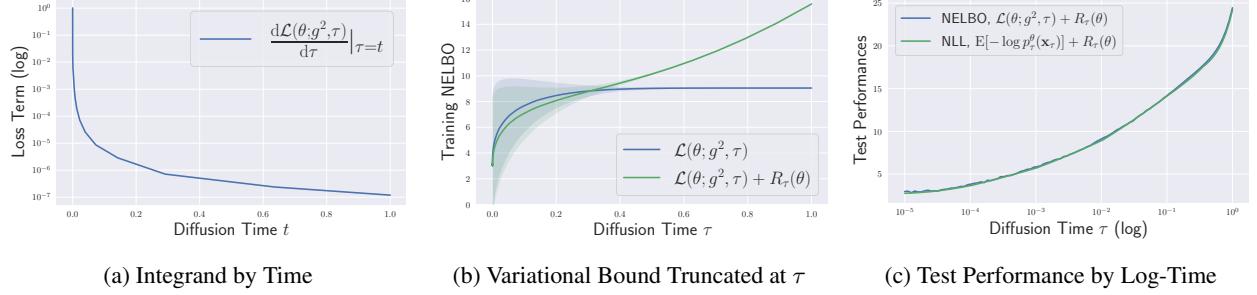


Figure 1: The contribution of diffusion time on the variational bound experimented on CIFAR-10 with DDPM++ (VP, NLL) (Song et al., 2021a). (a) The integrand of the variational bound is extremely imbalanced on $[\epsilon, T]$. (b) The truncated variational bound only changes near $\tau \approx 0$. (c) The truncation hyperparameter (ϵ) is a significant factor for performances.

The NCSN loss training is connected to the likelihood training in Song et al. (2021a) by

$$\mathbb{E}_{\mathbf{x}_0}[-\log p_0^\theta(\mathbf{x}_0)] \leq \mathcal{L}_{NCSN}(\theta; g^2), \quad (5)$$

when the weighting function is the square of the diffusion term as $\lambda(t) = g^2(t)$, called the likelihood weighting.

3. Training and Evaluation of Diffusion Models in Practice

3.1. The Need of Truncation

In the family of linear SDEs, the gradient of the log transition probability satisfies $\nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0) = -\frac{\mathbf{x}_t - \mu(t)\mathbf{x}_0}{\sigma^2(t)} = -\frac{\mathbf{z}}{\sigma(t)}$, where \mathbf{x}_t is given to $\mu(t)\mathbf{x}_0 + \sigma(t)\mathbf{z}$ with $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$. The denominator of $\sigma(t)$ converges to zero as $t \rightarrow 0$, which leads $\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2$ to diverge as $t \rightarrow 0$, as illustrated in Figure 1-(a), see Appendix A.2 for details. Therefore, the Monte-Carlo estimation of the NCSN loss is under high variance, which prevents stable training of the score network. In practice, therefore, previous research truncates the diffusion time range to $[\tau, T]$, with a positive truncation hyperparameter, $\tau = \epsilon > 0$.

3.2. Variational Bound With Positive Truncation

For the analysis for density estimation in Section 3.3, this section derives the variational bound of the log-likelihood when a diffusion model has a positive truncation because Inequality (5) holds only with zero truncation ($\tau = 0$). Lemma 1 provides a generalization of Inequality (5), proved by applying the data processing inequality (Gerchinovitz et al., 2020) and the Girsanov theorem (Pavon & Wakolbinger, 1991; Vargas et al., 2021; Song et al., 2021a).

Lemma 1. For any $\tau \in [0, T]$,

$$\mathbb{E}_{\mathbf{x}_\tau}[-\log p_\tau^\theta(\mathbf{x}_\tau)] \leq \mathcal{L}(\theta; g^2, \tau) \quad (6)$$

holds, where $\mathcal{L}(\theta; g^2, \tau) = \frac{1}{2} \int_\tau^T g^2(t) \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t} [\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] dt$, up to a constant, see Eq. (17).

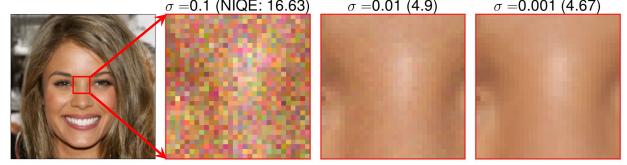


Figure 2: The truncation time is key to enhance the microscopic sample quality.

Lemma 1 is a generalization of Inequality (5) in that Inequality (6) collapses to Inequality (5) under the zero truncation: $\mathcal{L}_{NCSN}(\theta; \lambda) = \mathcal{L}(\theta; \lambda, \tau = 0)$. If the time range is truncated to $[\tau, T]$ for $\tau \in [0, T]$, then from the variational inference, the log-likelihood becomes

$$\mathbb{E}_{\mathbf{x}_0}[-\log p_0^\theta(\mathbf{x}_0)] \leq \mathbb{E}_{\mathbf{x}_\tau}[-\log p_\tau^\theta(\mathbf{x}_\tau)] + R_\tau(\theta) \quad (7)$$

where

$$R_\tau(\theta) = \mathbb{E}_{\mathbf{x}_0} \left[\int p_{0\tau}(\mathbf{x}_\tau | \mathbf{x}_0) \log \frac{p_{0\tau}(\mathbf{x}_\tau | \mathbf{x}_0)}{p_\theta(\mathbf{x}_0 | \mathbf{x}_\tau)} d\mathbf{x}_\tau \right],$$

with $p_\theta(\mathbf{x}_0 | \mathbf{x}_\tau)$ being the probability distribution of \mathbf{x}_0 given \mathbf{x}_τ and the score estimation with \mathbf{s}_θ at τ . For any τ , we apply Lemma 1 to the right-hand-side of Inequality (7) to obtain the variational bound of the log-likelihood as

$$\mathbb{E}_{\mathbf{x}_0}[-\log p_0^\theta(\mathbf{x}_0)] \leq \mathcal{L}(\theta; g^2, \tau) + R_\tau(\theta). \quad (8)$$

3.3. A Universal Phenomenon in Diffusion Training: Extremely Imbalanced Loss

To avoid the diverging issue introduced in Section 3.1, previous works in VPSDE (Song et al., 2021a; Vahdat et al., 2021) modify the loss by truncating the integration on $[\tau, T]$ with a fixed hyperparameter $\tau = \epsilon > 0$ so that the score network does not estimate the score function on $[0, \epsilon]$. Analogously, previous works in VESDE (Song et al., 2021b; Chen et al., 2022) approximate $\sigma_{VE}^2(t) \approx \sigma_{min}^2 (\frac{\sigma_{max}}{\sigma_{min}})^{2t}$ to truncate the minimum variance of the transition probability to be σ_{min}^2 . Truncating diffusion time at ϵ in VPSDE is equivalent to

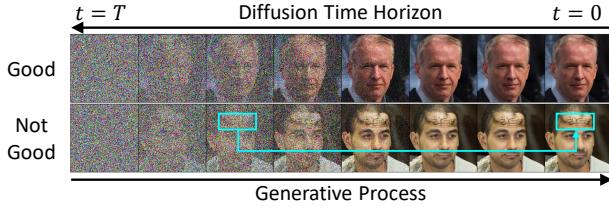


Figure 3: Illustration of the generative process trained on CelebA-HQ 256×256 with NCSN++ (VE) (Song et al., 2021b). The score precision on large diffusion time is key to construct the realistic overall sample quality.

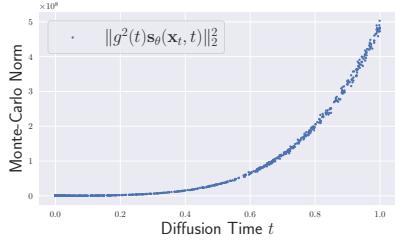


Figure 4: Norm of reverse drift of generative process, trained on CIFAR-10 with DDPM++ (VP, FID) (Song et al., 2021b).

truncating diffusion variance (σ_{min}^2) in VESDE, so these two truncations on VE/VP SDEs have the identical effect on bounding the diffusion loss. Henceforth, this paper discusses the argument of truncating diffusion time (VPSDE) and diffusion variance (VESDE) exchangeably.

Figure 1 illustrates the significance of truncation in the training of diffusion models. With the truncation of strictly positive $\epsilon = 10^{-5}$, Figure 1-(a) shows that the integrand of $\mathcal{L}(\theta; g^2, \tau)$ in the Bits-Per-Dimension (BPD) scale is still extremely imbalanced. It turns out that such extreme imbalance appears to be a universal phenomenon in training a diffusion model, and this phenomenon lasts from the beginning to the end of training.

Figure 1-(b) with the green line presents the variational bound of the log-likelihood (right-hand-side of Inequality (8)) on the y -axis, and it indicates that the variational bound is sharply decreasing near the small diffusion time. Therefore, if ϵ is insufficiently small, the variational bound is not tight to the log-likelihood, and a diffusion model fails at MLE training. In addition, Figure 2 indicates that insufficiently small ϵ (or σ_{min}) would also harm the microscopic sample quality. From these observations, ϵ becomes a significant hyperparameter that needs to be selected carefully.

3.4. Effect of Truncation on Model Evaluation

Figure 1-(c) reports test performances on density estimation. Figure 1-(c) illustrates that both Negative Evidence Lower Bound (NELBO) and NLL monotonically decrease

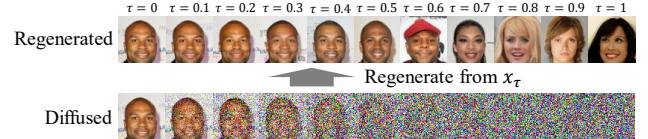


Figure 5: Regenerated samples synthesized by solving the probability flow ODE on $[\epsilon, \tau]$ backwards with the initial point of $\mathbf{x}_\tau = \mu(\tau)\mathbf{x}_0 + \sigma(\tau)\mathbf{z}$ for $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$, trained on CelebA with DDPM++ (VP, FID) (Song et al., 2021b).

by lowering ϵ because NELBO is largely contributed by small diffusion time at test time as well as training time. Therefore, it could be a common strategy to reduce ϵ as much as possible to reduce test NELBO/NLL.

On the contrary, there is a counter effect on FID for ϵ .

Table 1, trained on CIFAR-10 (Krizhevsky et al., 2009) with NCSN++ (Song et al., 2021b), presents that FID is worsened as we take smaller hyperparameter σ_{min} for the training.

It is the range of small diffusion time that significantly contributes to the variational bound in the blue line of Figure 1-(b), so the score network with a small truncation hyperparameter, σ_{min} or ϵ , remains unoptimized on large diffusion time. In the lens of Figure 2, therefore, the inconsistent result of Table 1 is attributed to the inaccurate score on large diffusion time.

We design an experiment to validate the above argument in Table 2. This experiment utilizes two types of score networks: 1)

three alternative networks (As) with diverse $\sigma_{min} \in \{10^{-3}, 10^{-4}, 10^{-5}\}$ trained in Table 1 experiment; 2) a network (B) with $\sigma_{min} = 10^{-5}$ (the last row of Table 1). With these score networks, we denoise the noises by either one of the first-typed As from σ_{max} to a common and fixed $\sigma_{tr} (= 1)$, and we use B to further denoise from σ_{tr} to $\sigma_{min} = 10^{-5}$. This further denoising step with model B enables us to compare the score accuracy on large diffusion time for models with diverse truncation hyperparameters in a fair resolution setting. Table 2 presents that the model with $\sigma_{min} = 10^{-3}$ has the best FID, implying that the training with too small truncation would harm the sample fidelity.

Specifically, Figure 4 shows the Euclidean norm of $g^2(t)\mathbf{s}_\theta(\mathbf{x}_t, t)$, where each dot represents for a Monte-Carlo sample from $p_t(\mathbf{x}_t)$. Here, $g^2(t)\mathbf{s}_\theta(\mathbf{x}_t, t)$ is in the reverse drift term of the generative process, $d\mathbf{x}_t^\theta = [\mathbf{f}(\mathbf{x}_t^\theta, t) - g^2(t)\mathbf{s}_\theta(\mathbf{x}_t^\theta, t)] dt + g(t) d\bar{\mathbf{w}}_t$. Figure 4 illustrates that it is

Table 1: Ablation on σ_{min} .

σ_{min}	CIFAR-10	
	NLL (\downarrow)	FID-10k (\downarrow)
10^{-2}	4.95	6.95
10^{-3}	3.04	7.04
10^{-4}	2.99	8.17
10^{-5}	2.97	8.29

Table 2: FID-10k scores.

σ_{min}	10^{-3}	10^{-4}	10^{-5}
$\sigma_{tr} = 1$	6.84	8.04	8.29

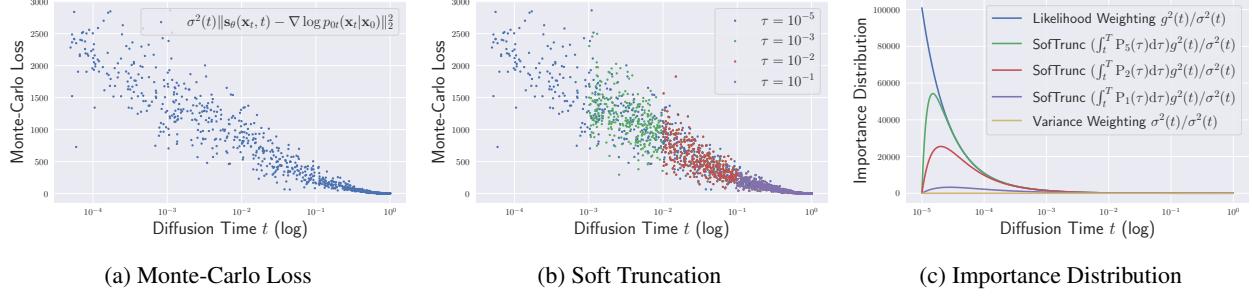


Figure 6: The experimental result trained on CIFAR-10 with DDPM++ (VP, NLL) (Song et al., 2021a). (a) The Monte-Carlo loss for each diffusion time, $\sigma^2(t)\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2$. (b) The Monte-Carlo loss for each diffusion time on various truncation time. (c) The importance distribution for various truncation distributions.

the large diffusion time that dominates the sampling process. Therefore, a precise score network on large diffusion time is particularly important in sample generation.

The imprecise score mainly affects the global sample context, as the denoising on small diffusion time only crafts the image in its microscopic details, illustrated in Figures 3 and 5. Figure 3 shows how the global fidelity is damaged: a man synthesized in the second row has unrealistic curly hair on his forehead, constructed on the large diffusion time. Figure 5 deepens the importance of learning a good score estimation on large diffusion time. It shows the regenerated samples by solving the generative process time reversely, starting from \mathbf{x}_τ (Meng et al., 2021).

4. Soft Truncation: A Training Technique for a Diffusion Model

As in Section 3, the choice of ϵ is crucial for training and evaluation, but it is computationally infeasible to search for the optimal ϵ . Therefore, we introduce a training technique that predominantly mediates the need for ϵ -search by softening the fixed truncation hyperparameter into a truncation random variable so that the truncation time varies in every optimization step. Our approach successfully trains the score network on large diffusion time without sacrificing NLL. We explain the Monte-Carlo estimation of the variational bound in Section 4.1, which is the common practice of previous research but explained to emphasize how simple (though effective) Soft Truncation is, and we subsequently introduce Soft Truncation in Section 4.2.

4.1. Monte-Carlo Estimation of Truncated Variational Bound with Importance Sampling

In this section, we fix a truncation hyperparameter to be $\tau = \epsilon$. For every batch $\{\mathbf{x}_0^{(b)}\}_{b=1}^B$, the Monte-Carlo estimation of the variational bound in Inequality (6) is $\mathcal{L}(\boldsymbol{\theta}; g^2, \epsilon) \approx \hat{\mathcal{L}}(\boldsymbol{\theta}; g^2, \epsilon) = \frac{1}{2B} \sum_{b=1}^B g^2(t^{(b)}) \|\mathbf{s}_\theta(\mathbf{x}_{t^{(b)}}, t^{(b)}) - \nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2$

$\nabla \log p_{0t^{(b)}}(\mathbf{x}_{t^{(b)}} | \mathbf{x}_0)\|_2^2$, up to a constant irrelevant to $\boldsymbol{\theta}$, where $\mathbf{x}_{t^{(b)}} = \mu(t^{(b)})\mathbf{x}_0 + \sigma(t^{(b)})\epsilon^{(b)}$ with $\{\mathbf{x}_t^{(b)}\}_{b=1}^B$ and $\{\epsilon^{(b)}\}_{b=1}^B$ be the corresponding Monte-Carlo samples from $t^{(b)} \sim [\epsilon, T]$ and $\epsilon^{(b)} \sim \mathcal{N}(0, \mathbf{I})$, respectively. Note that this Monte-Carlo estimation is tractably computed from the analytic form of the transition probability as $\nabla \log p_{0t^{(b)}}(\mathbf{x}_{t^{(b)}} | \mathbf{x}_0) = \frac{\epsilon^{(b)}}{\sigma(t^{(b)})}$ under linear SDEs.

Previous works (Song et al., 2021a; Huang et al., 2021) apply the importance sampling with the importance distribution of $p_{iw}(t) = \frac{g^2(t)/\sigma^2(t)}{Z_\epsilon} 1_{[\epsilon, T]}(t)$, where $Z_\epsilon = \int_\epsilon^T \frac{g^2(t)}{\sigma^2(t)} dt$. It is well known (Goodfellow et al., 2016) that the Monte-Carlo variance of $\hat{\mathcal{L}}$ is minimum if the importance distribution is $p_{iw}^*(t) \propto g^2(t)L(t)$ with $L(t) = \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t} [\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2]$, but sampling of Monte-Carlo diffusion time from $p_{iw}^*(t)$ at every training iteration would incur $2\times$ slower training speed, at least, because the importance sampling requires the score evaluation. Therefore, previous research approximates $L(t)$ by $\hat{L}(t) = \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t} [\|\nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] \propto 1/\sigma^2(t)$, and $p_{iw}(t)$ becomes the approximate importance weight. This approximation, at the expense of bias, is cheap because the closed-form of the inverse Cumulative Distribution Function (CDF) is known. Unless we train the variance directly as in Kingma et al. (2021), we believe $p_{iw}(t)$ is the maximally efficient sampler as long as the training speed matters. The importance weighted Monte-Carlo estimation becomes

$$\begin{aligned} & \mathcal{L}(\boldsymbol{\theta}; g^2, \epsilon) \\ &= \frac{Z_\epsilon}{2} \int_\epsilon^T p_{iw}(t) \sigma^2(t) \mathbb{E} [\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] dt \\ &\approx \frac{Z_\epsilon}{2B} \sum_{b=1}^B \sigma^2(t_{iw}^{(b)}) \left\| \mathbf{s}_\theta(\mathbf{x}_{t_{iw}^{(b)}}, t_{iw}^{(b)}) - \frac{\epsilon^{(b)}}{\sigma(t_{iw}^{(b)})} \right\|_2^2 \\ &:= \hat{\mathcal{L}}_{iw}(\boldsymbol{\theta}; g^2, \epsilon), \end{aligned} \quad (9)$$

where $\{t_{iw}^{(b)}\}_{b=1}^B$ is the Monte-Carlo sample from the importance distribution, i.e., $t_{iw}^{(b)} \sim p_{iw}(t) \propto \frac{g^2(t)}{\sigma^2(t)}$.

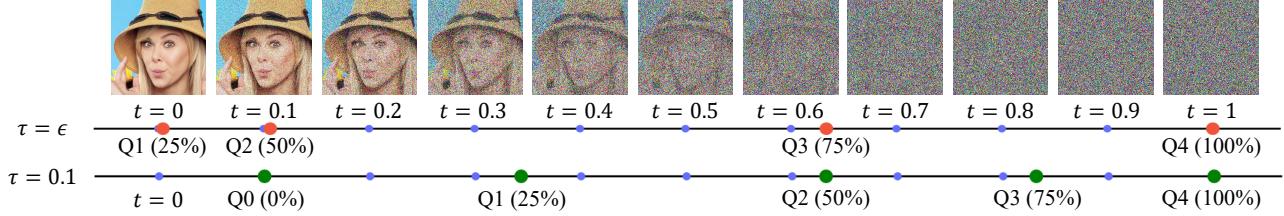


Figure 7: Quartile of importance weighted Monte-Carlo time of VPSDE. Red dots represent Q1/Q2/Q3/Q4 quantiles when truncated at $\tau = \epsilon = 10^{-5}$. About 25% and 50% of Monte-Carlo time are located in $[\epsilon, 5 \times 10^{-3}]$ and $[\epsilon, 0.106]$, respectively. Green dots represent Q0-Q5 quantiles when truncated at $\tau = 0.1$. Importance weighted Monte-Carlo time with $\tau = 0.1$ is distributed much more balanced compared to the truncation at $\tau = \epsilon$.

The importance sampling is advantageous in both NLL and FID (Song et al., 2021a) over the uniform sampling, as the importance sampling significantly reduces the estimation variance. Figure 6-(a) illustrates the sample-by-sample loss, and the importance sampling significantly mitigates the loss scale by diffusion time compared to the scale in Figure 1-(a). However, the importance distribution satisfies $p_{iw}(t) \rightarrow \infty$ as $t \rightarrow 0$ in Figure 6-(c) blue line, and most of the importance weighted Monte-Carlo time is concentrated at $t \approx \epsilon$ in Figure 7. Hence, the use of the importance sampling has a trade-off between the reduced variance (Figure 6-(a)) versus the over-sampled diffusion time near $t \approx \epsilon$ (Figure 7). Regardless of whether to use the importance sampling or not, therefore, the inaccurate score estimation on large diffusion time appears sampling-strategic-independently, and solving this pre-matured score estimation becomes a nontrivial task.

Instead of the likelihood weighting, previous works (Ho et al., 2020; Nichol & Dhariwal, 2021; Dhariwal & Nichol, 2021) train the denoising score loss with the variance weighting, $\lambda(t) = \sigma^2(t)$. With this weighting, the importance distribution becomes the uniform distribution, $p_{iw}(t) = \frac{\lambda(t)}{\sigma^2(t)} \equiv 1$, so it significantly alleviates the trade-off of using the likelihood weighting. However, the variance weighting favors FID at the sacrifice in NLL because the loss is no longer the variational bound of the log-likelihood. In contrast, the training with the likelihood weighting is leaning towards NLL than FID, so Soft Truncation is for the *balanced* NLL and FID, using the likelihood weighting.

4.2. Soft Truncation

Soft Truncation releases the truncation hyperparameter from a static variable to a random variable with a probability distribution of $\mathbb{P}(\tau)$. In every mini-batch update, Soft Truncation optimizes the diffusion model with $\hat{\mathcal{L}}_{iw}(\theta; g^2, \tau)$ in Eq. (9) for a sampled $\tau \sim \mathbb{P}(\tau)$. In other words, for every batch $\{\mathbf{x}_0^{(b)}\}_{b=1}^B$, Soft Truncation optimizes the Monte-Carlo loss

$$\hat{\mathcal{L}}_{iw}(\theta; \lambda, \tau) = \frac{Z_\tau}{2B} \sum_{b=1}^B \sigma^2(t_{iw}^{(b)}) \left\| \mathbf{s}_\theta(\mathbf{x}_{t_{iw}^{(b)}}, t_{iw}^{(b)}) - \frac{\epsilon^{(b)}}{\sigma(t_{iw}^{(b)})} \right\|_2^2$$

with $\{t_{iw}^{(b)}\}_{b=1}^B$ sampled from the importance distribution of $p_{iw, \tau}(t) = \frac{g^2(t)/\sigma^2(t)}{Z_\tau} \mathbf{1}_{[\tau, T]}(t)$, where $Z_\tau := \int_\tau^T \frac{g^2(t)}{\sigma^2(t)} dt$.

Soft Truncation resolves the oversampling issue of diffusion time near $t \approx \epsilon$, meaning that Monte-Carlo time is not concentrated on ϵ anymore. Figure 7 illustrates the quantiles of importance weighted Monte-Carlo time with Soft Truncation under $\tau = \epsilon$ and $\tau = 0.1$. The score network is trained more equally on diffusion time when $\tau = 0.1$, and as a consequence, the loss imbalance issue in each training step is also alleviated as in Figure 6-(b) with purple dots. This limited range of $[\tau, T]$ provides a chance to learn a score network more balanced on diffusion time. As τ is softened, such truncation level will vary by mini-batch updates: see the loss scales change by blue, green, red, and purple dots according to various τ s in Figure 6-(b). Eventually, the softened τ will provide a fair chance to learn the score network from small as well as large diffusion time.

4.3. Soft Truncation Equals to A Diffusion Model With A General Weight

In the original diffusion model, the loss estimation, $\hat{\mathcal{L}}(\theta; g^2, \epsilon)$, is just a batch-wise approximation of a population loss, $\mathcal{L}(\theta; g^2, \epsilon)$. However, the target population loss of Soft Truncation, $\mathcal{L}(\theta; g^2, \tau)$, is depending on a random variable τ , so the target population loss itself becomes a random variable. Therefore, we derive the *expected* Soft Truncation loss to reveal the connection to the original diffusion model:

$$\begin{aligned} \mathcal{L}_{ST}(\theta; g^2, \mathbb{P}) &:= \mathbb{E}_{\mathbb{P}(\tau)} [\mathcal{L}(\theta; g^2, \tau)] \\ &= \frac{1}{2} \int_\epsilon^T \mathbb{P}(\tau) \int_\tau^T g^2(t) \mathbb{E} [\|\mathbf{s}_\theta - \nabla \log p_{0t}\|_2^2] dt d\tau \\ &= \frac{1}{2} \int_\epsilon^T g_{\mathbb{P}}^2(t) \mathbb{E} [\|\mathbf{s}_\theta - \nabla \log p_{0t}\|_2^2] dt, \end{aligned}$$

up to a constant, where $g_{\mathbb{P}}^2(t) = (\int_0^t \mathbb{P}(\tau) d\tau) g^2(t)$, by exchanging the orders of the integrations. Therefore, we conclude that Soft Truncation reduces to a diffusion model with a general weight of $g_{\mathbb{P}}^2(t)$, see Appendix A.3:

$$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}) = \mathcal{L}(\theta; g_{\mathbb{P}}^2, \epsilon). \quad (10)$$

4.4. Soft Truncation is Maximum Perturbed Likelihood Estimation

As explained in Section 4.3, Soft Truncation is a diffusion model with a general weight, in the expected sense. Reversely, this section analyzes a diffusion model with a general weight in view of Soft Truncation. Suppose we have a general weight λ . Theorem 1 implies that this general weighted diffusion loss, $\mathcal{L}(\boldsymbol{\theta}; \lambda, \epsilon)$, is the variational bound of the perturbed KL divergence expected by $\mathbb{P}_\lambda(\tau)$. Theorem 1 collapses to Lemma 1 if $\lambda(t) = cg^2(t)$ for any $c > 0$ ¹. See Appendix B for the detailed statement and proof.

Theorem 1. Suppose $\frac{\lambda(t)}{g^2(t)}$ is a nondecreasing and nonnegative absolutely continuous function on $[\epsilon, T]$ and zero on $[0, \epsilon)$. For the probability defined by

$$\mathbb{P}_\lambda([a, b]) = \left[\int_{\max(a, \epsilon)}^b \left(\frac{\lambda(s)}{g^2(s)} \right)' ds + \frac{\lambda(\epsilon)}{g^2(\epsilon)} 1_{[a, b]}(\epsilon) \right] / Z,$$

where $Z = \frac{\lambda(T)}{g^2(T)}$: up to a constant, the variational bound of the general weighted diffusion loss becomes

$$\begin{aligned} & \mathbb{E}_{\mathbb{P}_\lambda(\tau)} [D_{KL}(p_\tau \| p_\tau^\theta)] \\ & \leq \frac{1}{2Z} \int_\epsilon^T \lambda(t) \mathbb{E}_{\mathbf{x}_t} [\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_t(\mathbf{x}_t)\|_2^2] dt \\ & = \frac{1}{Z} \mathcal{L}(\boldsymbol{\theta}; \lambda, \epsilon) = \mathbb{E}_{\mathbb{P}_\lambda(\tau)} [\mathcal{L}(\boldsymbol{\theta}; g^2, \tau)]. \end{aligned}$$

The meaning of Soft Truncation becomes clearer in view of Theorem 1. Instead of training the general weighted diffusion loss, $\mathcal{L}(\boldsymbol{\theta}; \lambda, \epsilon)$, we optimize the *truncated* variational bound, $\mathcal{L}(\boldsymbol{\theta}; g^2, \tau)$. This truncated loss upper bounds the *perturbed* KL divergence, $D_{KL}(p_\tau \| p_\tau^\theta)$ by Lemma 1, and Figure 1-(c) indicates that the Inequality (6) is nearly tight. Therefore, Soft Truncation could be interpreted as the Maximum Perturbed Likelihood Estimation (MPLE), where the perturbation level is a random variable. Soft Truncation is not MLE training because the Inequality 8 is not tight as demonstrated in Figure 1-(b) unless τ is sufficiently small.

Old wisdom is to minimize the loss variance if available for stable training. However, some optimization methods in the deep learning era (e.g., stochastic gradient descent) deliberately add noises to a loss function that eventually helps escape from a local optimum. Soft Truncation is categorized in such optimization methods that *inflate* the loss variance by intentionally imposing auxiliary randomness on loss estimation. This randomness is represented by the outmost expectation of $\mathbb{E}_{\mathbb{P}_\lambda(\tau)}$, which controls the diffusion time range batch-wisely. Additionally, the loss with a sampled τ is the proxy of the perturbed KL divergence by τ , so the auxiliary randomness on loss estimation is theoretically tamed, meaning that it is not a random perturbation.

¹If $\lambda(t) = cg^2(t)$, the probability satisfies $\mathbb{P}([a, b]) = 1_{[a, b]}(\epsilon)$, which is a probability distribution of one mass at ϵ .

4.5. Choice of Truncation Probability Distribution

We parametrize the probability distribution of τ by

$$\mathbb{P}_k(\tau) = \frac{1/\tau^k}{Z_k} 1_{[\epsilon, T]}(\tau) \propto \frac{1}{\tau^k}, \quad (11)$$

where $Z_k = \int_\epsilon^T \frac{1}{\tau^k} d\tau$ with sufficiently small enough truncation hyperparameter. Note that it is still beneficial to remain ϵ strictly positive because a batch update with $\tau \approx 0 < \epsilon$ would drift the score network away from the optimal point. Figure 6-(c) illustrates the importance distribution of $\lambda_{\mathbb{P}_k}$ for varying k . From the definition of Eq. (11), $\mathbb{P}_k(\tau) \rightarrow \delta_\epsilon(\tau)$ as $k \rightarrow \infty$, and this limiting delta distribution corresponds to the original diffusion model with the likelihood weighting. Figure 6-(c) shows that the importance distribution of \mathbb{P}_k with finite k interpolates the likelihood weighting and the variance weighting.

With the current simple form, we experimentally find that the sweet spot is $k \approx 1.0$ in VPSDE and $k = 2.0$ in VESDE with the emphasis on the sample quality. For VPSDE, the importance distribution in Figure 6-(c) is nearly equal to that of the variance weighting if $k \approx 1.0$, so Soft Truncation with $k \approx 1.0$ improves the sample fidelity, while maintaining low NLL. On the other hand, if k is too small, no τ will be sampled near ϵ , so it hurts both sample generation and density estimation. We leave further study on searching for the optimal distribution of τ as future work.

5. Experiments

This section empirically studies our suggestions on benchmark datasets, including CIFAR-10 (Krizhevsky et al., 2009), ImageNet 32 × 32 (Van Oord et al., 2016), STL-10 (Coates et al., 2011)² CelebA (Liu et al., 2015) 64 × 64 and CelebA-HQ (Karras et al., 2018) 256 × 256.

Soft Truncation is a universal training technique independent to model architectures and diffusion strategies. In the experiments, we test Soft Truncation on various architectures, including vanilla NCSN++, DDPM++, Unbounded NCSN++ (UNCSN++), and Unbounded DDPM++ (UD-DDPM++). Also, Soft Truncation is applied to various diffusion SDEs, such as VESDE, VPSDE, and Reverse VESDE (RVESDE). Although we use continuous SDEs for the diffusion strategies, Soft Truncation with the discrete model, such as DDPM (Ho et al., 2020), is a straightforward application of continuous models. Appendix D enumerates the specifications of score architectures and SDEs.

From Figure 1-(c), a sweet spot of the hard threshold is $\epsilon = 10^{-5}$, in which NLL/NELBO are no longer improved under this threshold. As the diffusion model has no informa-

²We downsize the dataset from 96 × 96 to 48 × 48 following Jiang et al. (2021); Park & Kim (2022).

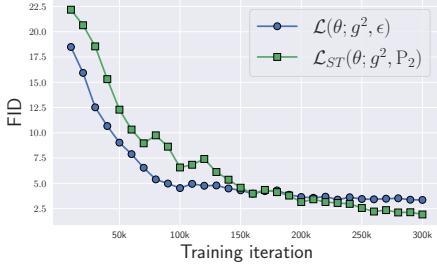


Figure 8: Soft Truncation improves FID on CelebA trained with UNCSN++ (RVE).

Table 4: Ablation study of Soft Truncation for various weightings on CIFAR-10 and ImageNet32 with DDPM++ (VP).

	Loss	Soft Truncation	NLL	NELBO	FID ODE
CIFAR-10	$\mathcal{L}(\theta; g^2, \epsilon)$	X	3.03	3.13	6.70
	$\mathcal{L}(\theta; \sigma^2, \epsilon)$	X	3.21	3.34	3.90
	$\mathcal{L}(\theta; g_{\mathbb{P}_1}^2, \epsilon)$	X	3.06	3.18	6.11
	$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$	✓	3.01	3.08	3.96
	$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_{0.9})$	✓	3.03	3.13	3.45
ImageNet32	$\mathcal{L}(\theta; g^2, \epsilon)$	X	3.92	3.94	12.68
	$\mathcal{L}(\theta; \sigma^2, \epsilon)$	X	3.95	4.00	9.22
	$\mathcal{L}(\theta; g_{\mathbb{P}_1}^2, \epsilon)$	X	3.93	3.97	11.89
	$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_{0.9})$	✓	3.90	3.91	8.42

Table 5: Ablation study of Soft Truncation for various model architectures and diffusion SDEs on CelebA.

SDE	Model	Loss	NLL	NELBO	FID PC ODE
VE	NCSN++	$\mathcal{L}(\theta; \sigma^2, \epsilon)$	3.41	3.42	3.95 -
		$\mathcal{L}_{ST}(\theta; \sigma^2, \mathbb{P}_2)$	3.44	3.44	2.68 -
RVE	UNCSN++	$\mathcal{L}(\theta; g^2, \epsilon)$	2.01	2.01	3.36 -
		$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_2)$	1.97	2.02	1.92 -
DDPM++		$\mathcal{L}(\theta; \sigma^2, \epsilon)$	2.14	2.21	3.03 2.32
		$\mathcal{L}_{ST}(\theta; \sigma^2, \mathbb{P}_1)$	2.17	2.29	2.88 1.90
UDDPM++		$\mathcal{L}(\theta; \sigma^2, \epsilon)$	2.11	2.20	3.23 4.72
		$\mathcal{L}_{ST}(\theta; \sigma^2, \mathbb{P}_1)$	2.16	2.28	2.22 1.94
DDPM++		$\mathcal{L}(\theta; g^2, \epsilon)$	2.00	2.09	5.31 3.95
		$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$	2.00	2.11	4.50 2.90
UDDPM++		$\mathcal{L}(\theta; g^2, \epsilon)$	1.98	2.12	4.65 3.98
		$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$	2.00	2.10	4.45 2.97

tion on $[0, \epsilon]$, we comply Kim et al. (2022) to use Inequality (7) for NLL computation and Inequality (8) for NELBO computation. Following Kim et al. (2022), we compute $\log p_\epsilon^\theta(\mathbf{x}_\epsilon)$, rather than $\log p_\epsilon^\theta(\mathbf{x}_0)$. It is the common practice of continuous diffusion models (Song et al., 2021b;a; Dockhorn et al., 2022) to report their performances with $\log p_\epsilon^\theta(\mathbf{x}_0)$, but Kim et al. (2022) show that $\log p_\epsilon^\theta(\mathbf{x}_\epsilon)$ differs to $\log p_\epsilon^\theta(\mathbf{x}_0)$ by 0.05 in BPD scale when $\epsilon = 10^{-5}$, which is quite significant. We use the uniform dequantization (Theis et al., 2016) as default, otherwise noted. For sample generation, we use either of Predictor-Corrector (PC) sampler or Ordinary Differential Equation (ODE) sampler (Song et al., 2021b). We denote $\mathcal{L}(\theta; \lambda, \epsilon)$ as the vanilla training with λ -weighting, and $\mathcal{L}_{ST}(\theta; g^2, \mathbb{P})$ as the training by Soft Truncation with the truncation probability of

Table 6: Ablation study of Soft Truncation for various ϵ on CIFAR-10 with DDPM++ (VP).

Loss	ϵ	NLL	NELBO	FID (ODE)
$\mathcal{L}(\theta; g^2, \epsilon)$	10^{-2}	4.64	4.69	38.82
	10^{-3}	3.51	3.52	6.21
	10^{-4}	3.05	3.08	6.33
	10^{-5}	3.03	3.13	6.70
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$	10^{-2}	4.65	4.69	39.83
	10^{-3}	3.51	3.52	5.14
	10^{-4}	3.05	3.08	4.16
	10^{-5}	3.01	3.08	3.96

Table 7: Ablation study of Soft Truncation for various \mathbb{P}_k on CIFAR-10 trained with DDPM++ (VP).

Loss	NLL	NELBO	FID (ODE)
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_0)$	3.24	3.39	6.27
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_{0.8})$	3.03	3.05	3.61
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_{0.9})$	3.03	3.13	3.45
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$	3.01	3.08	3.96
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_{1.1})$	3.02	3.09	3.98
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_{1.2})$	3.03	3.09	3.98
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_2)$	3.01	3.10	6.31
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_3)$	3.02	3.09	6.54
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_\infty)$ $= \mathcal{L}(\theta; g^2, \epsilon)$	3.01	3.09	6.70

Table 8: Ablation study of Soft Truncation for CIFAR-10 trained with DDPM++ when a diffusion is combined with a normalizing flow in INDM (Kim et al., 2022).

Loss	NLL	NELBO	FID (ODE)
INDM (VP, NLL)	2.98	2.98	6.01
INDM (VP, FID)	3.17	3.23	3.61
INDM (VP, NLL) + ST	3.01	3.02	3.88

\mathbb{P} . We additionally denote $\mathcal{L}_{ST}(\theta; \sigma^2, \mathbb{P})$ for updating the network by the variance weighted loss per batch-wise update. We release our code at <https://github.com/Kim-Dongjun/Soft-Truncation>.

FID by Iteration Figure 8 illustrates the FID score (Heusel et al., 2017) in y -axis by training steps in x -axis. Figure 8 shows that Soft Truncation beats the vanilla training after 150k of training iterations.

Ablation Studies Tables 4, 5, 6, and 7 show ablation studies on various weighting functions, model architectures, SDEs, ϵ s, and probability distributions of τ , respectively. See Appendix E.2. Table 4 shows that Soft Truncation beats or equals to the vanilla training in all performances. We highlight that Soft Truncation with $\mathbb{P}_{0.9}$ outperforms the FID-favorable model with the variance weighting with respect to FID on both CIFAR-10 and ImageNet32.

Not only comparing with the pre-existing weighting functions, such as $\lambda = g^2$ or $\lambda = \sigma^2$, Table 4 additionally reports the experimental result of a general weighting function of $\lambda = g_{\mathbb{P}_1}^2$. From Eq. (10), Soft Truncation with \mathbb{P}_1 and the vanilla training with $\lambda = g_{\mathbb{P}_1}^2$ coincide in their loss functions in average, i.e., $\mathcal{L}(\theta; g_{\mathbb{P}_1}^2, \epsilon) = \mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$. Thus, when

Table 9: Performance comparisons on benchmark datasets. The boldfaced numbers present the best performance, and the underlined numbers present the second-best performance. We report NLL of DDPM++ on CIFAR-10, ImageNet32, and CelebA with the variational dequantization (Song et al., 2021a) to compare with the baselines in a fair setting.

Model	CIFAR10				ImageNet32				CelebA		CelebA-HQ		STL-10	
	32 × 32		NLL (↓)	FID (↓)	32 × 32		NLL	FID	IS	NLL	FID	256 × 256	FID	FID
Likelihood-free Models														
StyleGAN2-ADA+Tuning (Karras et al., 2020)	-	2.92	<u>10.02</u>	-	-	-	-	-	-	-	-	-	-	-
Styleformer (Park & Kim, 2022)	-	2.82	9.94	-	-	-	-	-	-	3.66	-	<u>15.17</u>	<u>11.01</u>	-
Likelihood-based Models														
ARDM-Upscale 4 (Hoogeboom et al., 2021)	2.64	-	-	-	-	-	-	-	-	-	-	-	-	-
VDM (Kingma et al., 2021)	<u>2.65</u>	7.41	-	<u>3.72</u>	-	-	-	-	-	-	-	-	-	-
LSGM (FID) (Vahdat et al., 2021)	3.43	2.10	-	-	-	-	-	-	-	-	-	-	-	-
NCSN++ cont. (deep, VE) (Song et al., 2021b)	3.45	<u>2.20</u>	9.89	-	-	-	-	2.39	3.95	<u>7.23</u>	-	-	-	-
DDPM++ cont. (deep, sub-VP) (Song et al., 2021b)	2.99	2.41	9.57	-	-	-	-	-	-	-	-	-	-	-
DenseFlow-74-10 (Grčić et al., 2021)	2.98	34.90	-	3.63	-	-	-	1.99	-	-	-	-	-	-
ScoreFlow (VP, FID) (Song et al., 2021a)	3.04	3.98	-	3.84	8.34	-	-	-	-	-	-	-	-	-
Efficient-VDVAE (Hazami et al., 2022)	2.87	-	-	-	-	-	-	1.83	-	-	-	-	-	-
PNDM (Liu et al., 2022)	-	3.26	-	-	-	-	-	-	2.71	-	-	-	-	-
ScoreFlow (deep, sub-VP, NLL) (Song et al., 2021a)	2.81	5.40	-	3.76	10.18	-	-	-	-	-	-	-	-	-
Improved DDPM (L_{simple}) (Nichol & Dhariwal, 2021)	3.37	2.90	-	-	-	-	-	-	-	-	-	-	-	-
UNCSN++ (RVE) + ST	3.04	2.33	10.11	-	-	-	-	1.97	<u>1.92</u>	7.16	7.71	13.43	-	-
DDPM++ (VP, FID) + ST	2.91	2.47	9.78	-	-	-	-	2.10	1.90	-	-	-	-	-
DDPM++ (VP, NLL) + ST	2.88	3.45	9.19	3.85	<u>8.42</u>	11.82	<u>1.96</u>	2.90	-	-	-	-	-	-

comparing the paired experiments, Soft Truncation could be considered as an alternative way of estimating the same loss, and Table 4 implies that Soft Truncation gives better optimization than the vanilla method. This strongly implies that Soft Truncation could be a default training method for a general weighted denoising diffusion loss.

Table 5 provides two implications. First, Soft Truncation particularly boosts FID while maintaining density estimation performances under the variation of score networks and diffusion strategies. Second, Table 5 shows that Soft Truncation is effective on CelebA even when we apply Soft Truncation on the variance weighting, i.e., $\mathcal{L}_{ST}(\theta; \sigma^2, \mathbb{P})$, but we find that this does not hold on CIFAR-10 and ImageNet32. We leave it as a future work on this extent.

Table 6 shows a contrastive trend of the vanilla training and Soft Truncation. The inverse correlation appears between NLL and FID in the vanilla training, but Soft Truncation monotonically reduces both NLL and FID by ϵ . This implies that Soft Truncation significantly reduces the effort of the ϵ search. Table 7 studies the effect of the probability distribution of τ in VPSDE. It shows that Soft Truncation significantly improves FID upon the experiment of $\mathcal{L}(\theta; g^2, \epsilon)$ on the range of $0.8 \leq k \leq 1.2$. Finally, Table 8 shows that Soft Truncation also works with a nonlinear forward SDE (Kim et al., 2022), so the scope of Soft Truncation is not limited to a family of linear SDEs.

Quantitative Comparison to SOTA Table 9 compares Soft Truncation (ST) against the current best generative models. It shows that Soft Truncation achieves the state-of-the-art sample generation performances on CIFAR-10, CelebA,

CelebA-HQ, and STL-10, while keeping NLL intact. In particular, we have experimented thoroughly on the CelebA dataset, and we find that Soft Truncation largely exceeds the previous best FID scores by far. In FID, Soft Truncation with DDPM++ performs 1.90, which exceeds the previous best FID of 2.92 by DDGM. Also, Soft Truncation significantly improves FID on STL-10.

6. Conclusion

This paper proposes a generally applicable training method for diffusion models. The suggested training method, Soft Truncation, is motivated from the observation that the density estimation is mostly counted on small diffusion time, while the sample generation is mostly constructed on large diffusion time. However, small diffusion time dominates the Monte-Carlo estimation of the loss function, so this imbalance contribution prevents accurate score learning on large diffusion time. Soft Truncation softens the truncation level at each mini-batch update, and this simple modification is connected to the general weighted diffusion loss and the concept of Maximum Perturbed Likelihood Estimation.

Acknowledgements

This research was supported by AI Technology Development for Commonsense Extraction, Reasoning, and Inference from Heterogeneous Data(IITP) funded by the Ministry of Science and ICT(2022-0-00077). We thank Jaeyoung Byeon and Daehan Park for their fruitful mathematical advice, and Byeonghu Na for his support of the experiments.

References

- Anderson, B. D. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.
- Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.
- Chen, T., Liu, G.-H., and Theodorou, E. Likelihood training of schrödinger bridge using forward-backward SDEs theory. In *International Conference on Learning Representations*, 2022.
- Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223. JMLR Workshop and Conference Proceedings, 2011.
- Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34, 2021.
- Dockhorn, T., Vahdat, A., and Kreis, K. Score-based generative modeling with critically-damped langevin diffusion. *International Conference on Learning Representations*, 2022.
- Evans, L. C. Partial differential equations. *Graduate studies in mathematics*, 19(2), 1998.
- Gerchinovitz, S., Ménard, P., and Stoltz, G. Fano’s inequality for random variables. *Statistical Science*, 35(2): 178–201, 2020.
- Goodfellow, I., Bengio, Y., and Courville, A. *Deep learning*. MIT press, 2016.
- Grcić, M., Grubišić, I., and Šegvić, S. Densely connected normalizing flows. *Advances in Neural Information Processing Systems*, 34, 2021.
- Hazami, L., Mama, R., and Thurairatnam, R. Efficient-vdvae: Less is more. *arXiv preprint arXiv:2203.13751*, 2022.
- Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.
- Ho, J., Chen, X., Srinivas, A., Duan, Y., and Abbeel, P. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pp. 2722–2730. PMLR, 2019.
- Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.
- Hoogeboom, E., Gritsenko, A. A., Bastings, J., Poole, B., Berg, R. v. d., and Salimans, T. Autoregressive diffusion models. *arXiv preprint arXiv:2110.02037*, 2021.
- Huang, C.-W., Lim, J. H., and Courville, A. C. A variational perspective on diffusion-based generative models and score matching. *Advances in Neural Information Processing Systems*, 34, 2021.
- Jiang, Y., Chang, S., and Wang, Z. Transgan: Two pure transformers can make one strong gan, and that can scale up. *Advances in Neural Information Processing Systems*, 34, 2021.
- Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.
- Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4401–4410, 2019.
- Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., and Aila, T. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems*, 33:12104–12114, 2020.
- Kim, D., Na, B., Kwon, S. J., Lee, D., Kang, W., and Moon, I.-C. Maximum likelihood training of implicit nonlinear diffusion models. *arXiv preprint arXiv:2205.13699*, 2022.
- Kingma, D. P., Salimans, T., Poole, B., and Ho, J. Variational diffusion models. In *Advances in Neural Information Processing Systems*, 2021.
- Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.
- Liu, L., Ren, Y., Lin, Z., and Zhao, Z. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.
- Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pp. 3730–3738, 2015.
- Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.-Y., and Ermon, S. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2021.

- Nichol, A. Q. and Dhariwal, P. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pp. 8162–8171. PMLR, 2021.
- Oksendal, B. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.
- Park, J. and Kim, Y. Styleformer: Transformer based generative adversarial networks with style vector. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2022.
- Parmar, G., Zhang, R., and Zhu, J.-Y. On buggy resizing libraries and surprising subtleties in fid calculation. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2022.
- Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., and Tran, D. Image transformer. In *International Conference on Machine Learning*, pp. 4055–4064. PMLR, 2018.
- Pavon, M. and Wakolbinger, A. On free energy, stochastic control, and schrödinger processes. In *Modeling, Estimation and Control of Systems with Uncertainty*, pp. 334–348. Springer, 1991.
- Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.
- Song, Y. and Ermon, S. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.
- Song, Y., Durkan, C., Murray, I., and Ermon, S. Maximum likelihood training of score-based diffusion models. *Advances in Neural Information Processing Systems*, 34, 2021a.
- Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b.
- Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.
- Theis, L., van den Oord, A., and Bethge, M. A note on the evaluation of generative models. In *International Conference on Learning Representations (ICLR 2016)*, pp. 1–10, 2016.
- Vahdat, A. and Kautz, J. Nvae: A deep hierarchical variational autoencoder. *Advances in Neural Information Processing Systems*, 33:19667–19679, 2020.
- Vahdat, A., Kreis, K., and Kautz, J. Score-based generative modeling in latent space. *Advances in Neural Information Processing Systems*, 34, 2021.
- Van Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. Pixel recurrent neural networks. In *International Conference on Machine Learning*, pp. 1747–1756. PMLR, 2016.
- Vargas, F., Thodoroff, P., Lamacraft, A., and Lawrence, N. Solving schrödinger bridges via maximum likelihood. *Entropy*, 23(9):1134, 2021.
- Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688. Citeseer, 2011.

A. Derivation

A.1. Transition Probability for Linear SDEs

Kim et al. (2022) has classified linear SDEs as

$$d\mathbf{x}_t = -\frac{1}{2}\beta(t)\mathbf{x}_t dt + g(t) d\mathbf{w}_t, \quad (12)$$

where $\beta : \mathbb{R} \rightarrow \mathbb{R}_{\geq 0}$ and $g : \mathbb{R} \rightarrow \mathbb{R}_{\geq 0}$ are real-valued functions. VESDE has $\beta(t) \equiv 0$ and $g(t) = \sqrt{d\sigma^2(t)/dt} = \sigma_{min}(\frac{\sigma_{max}}{\sigma_{min}})^t \sqrt{2 \log \frac{\sigma_{max}}{\sigma_{min}}}$, where σ_{min} and σ_{max} are the minimum/maximum perturbation variances, respectively. It has the transition probability of

$$p_{0t}(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \mu_{VE}(t)\mathbf{x}_0, \sigma_{VE}^2(t)\mathbf{I}),$$

where $\mu_{VE}(t) \equiv 1$ and $\sigma_{VE}^2(t) := \sigma_{min}^2[(\frac{\sigma_{max}}{\sigma_{min}})^{2t} - 1]$. VPSDE has $\beta(t) = \beta_{min} + (\beta_{max} - \beta_{min})t$ and $g(t) = \sqrt{\beta(t)}$ with the transition probability of

$$p_{0t}(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \mu_{VP}(t)\mathbf{x}_0, \sigma_{VP}^2(t)\mathbf{I}),$$

where $\mu_{VP}(t) = e^{-\frac{1}{2} \int_0^t \beta(s) ds}$ and $\sigma^2(t) = 1 - e^{-\int_0^t \beta(s) ds}$.

Analogous to VE/VP SDEs, the transition probability of the generic linear SDE of Eq. (12) is a Gaussian distribution of $p_{0t}(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t | \mu(t)\mathbf{x}_0, \sigma^2(t)\mathbf{I})$, where its mean and covariance functions are characterized as a system of ODEs of

$$\frac{d\mu(t)}{dt} = -\frac{1}{2}\beta(t)\mu(t), \quad (13)$$

$$\frac{d\sigma^2(t)}{dt} = -\beta(t)\sigma^2(t) + g^2(t), \quad (14)$$

with initial conditions to be $\mu(0) = 1$ and $\sigma^2(0) = 0$.

Eq. (13) has its solution by

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

If we multiply $e^{\int_0^t \beta(s) ds}$ to Eq. (14), then Eq. (14) equals to

$$\begin{aligned} & e^{\int_0^t \beta(s) ds} \frac{d\sigma^2(t)}{dt} + e^{\int_0^t \beta(s) ds} \beta(t)\sigma^2(t) = e^{\int_0^t \beta(s) ds} g^2(t) \\ \iff & \frac{d[e^{\int_0^t \beta(s) ds} \sigma^2(t)]}{dt} = e^{\int_0^t \beta(s) ds} g^2(t) \\ \iff & e^{\int_0^t \beta(s) ds} \sigma^2(t) = \int_0^t e^{\int_0^\tau \beta(s) ds} g^2(\tau) d\tau + C \\ \iff & \sigma^2(t) = e^{-\int_0^t \beta(s) ds} \int_0^t e^{\int_0^\tau \beta(s) ds} g^2(\tau) d\tau + C e^{-\int_0^t \beta(s) ds}. \end{aligned} \quad (15)$$

If we impose $\sigma^2(0) = 0$ to Eq. (15), then the constant C satisfies $C = 0$, and the variance formula becomes

$$\sigma^2(t) = e^{-\int_0^t \beta(s) ds} \int_0^t e^{\int_0^\tau \beta(s) ds} g^2(\tau) d\tau.$$

To sum up, the family of linear SDEs of $d\mathbf{x}_t = -\frac{1}{2}\beta(t)\mathbf{x}_t dt + g(t) d\mathbf{w}_t$ gets the transition probability to be

$$p_{0t}(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t \middle| e^{-\frac{1}{2} \int_0^t \beta(s) ds} \mathbf{x}_0, e^{-\int_0^t \beta(s) ds} \left(\int_0^t e^{\int_0^\tau \beta(s) ds} g^2(\tau) d\tau \right) \mathbf{I}\right). \quad (16)$$

A.2. Diverging Denoising Loss

The gradient of the log transition probability, $\nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0) = -\frac{\mathbf{x}_t - \mu(t)\mathbf{x}_0}{\sigma^2(t)} = -\frac{\mathbf{z}}{\sigma(t)}$, is diverging at $\mu(t)\mathbf{x}_0$, where $\mathbf{x}_t = \mu(t)\mathbf{x}_0 + \sigma(t)\mathbf{z}$. Below Lemma 2 indicates that $\|\mathbf{s}(\mathbf{x}_t, t) - \nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2 \rightarrow \infty$ for any continuous score function, \mathbf{s} . This leads that the denoising score loss diverges as $t \rightarrow 0$ as illustrated in Figure 1-(a).

Lemma 2. Let $\mathcal{H}_{[0,T]} = \{\mathbf{s} : \mathbb{R}^d \times [0, T] \rightarrow \mathbb{R}^d, \mathbf{s} \text{ is locally Lipschitz}\}$. Suppose a continuous vector field \mathbf{v} defined on a subset U of a compact manifold M (i.e., $\mathbf{v} : U \subset M \rightarrow \mathbb{R}^d$) is unbounded, then there exists no $\mathbf{s} \in \mathcal{H}_{[0,T]}$ such that $\lim_{t \rightarrow 0} \mathbf{s}(\mathbf{x}, t) = \mathbf{v}(\mathbf{x})$ a.e. on U .

Proof of Lemma 2. Since U is an open subset of a compact manifold M , $\|\mathbf{x}_1 - \mathbf{x}_2\| \leq \text{diam}(M)$ for all $\mathbf{x}_1, \mathbf{x}_2 \in U$. Also, if $t_1, t_2 \in [0, T]$, $|t_1 - t_2|$ is bounded. Hence, the local Lipschitzness of \mathbf{s} implies that there exists a positive $K > 0$ such that $\|\mathbf{s}(\mathbf{x}_1, t_1) - \mathbf{s}(\mathbf{x}_2, t_2)\| \leq K(\|\mathbf{x}_1 - \mathbf{x}_2\| + |t_1 - t_2|)$ for any $\mathbf{x}_1, \mathbf{x}_2 \in U$ and $t_1, t_2 \in [0, T]$. Therefore, for any $\mathbf{s} \in \mathcal{H}_{[0,T]}$, there exists $C > 0$ such that $\|\mathbf{s}(\mathbf{x}, t)\| < C$ for all $\mathbf{x} \in U$ and $t \in [0, T]$, which leads no \mathbf{s} that satisfies $\mathbf{s}(\mathbf{x}, t) \rightarrow \mathbf{v}(\mathbf{x})$ a.e. on U as $t \rightarrow 0$. \square

A.3. General Weighted Diffusion Loss

The denoising score loss is

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}; g^2, \tau) &= \frac{1}{2} \int_\tau^T g^2(t) \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t} [\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2 - \|\log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] dt \\ &\quad - \int_\tau^T \mathbb{E}_{\mathbf{x}_t} [\text{div}(\mathbf{f}(\mathbf{x}_t, t))] dt - \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)], \end{aligned} \tag{17}$$

for any $\tau \in [0, T]$. For an appropriate class of function $A(t)$,

$$\begin{aligned} \int_0^T \mathbb{P}(\tau) \left(\int_\tau^T A(t) dt \right) d\tau &= \int_0^T \int_0^T \mathbb{P}(\tau) A(t) 1_{[\tau, T]}(t) dt d\tau \\ &= \int_0^T \int_0^T \mathbb{P}(\tau) A(t) 1_{[\tau, T]}(t) dt d\tau \\ &= \int_0^T \int_0^t \mathbb{P}(\tau) A(t) d\tau dt \\ &= \int_0^T \left(\int_0^t \mathbb{P}(\tau) d\tau \right) A(t) dt \end{aligned}$$

holds by changing the order of integration. Therefore, we get

$$\begin{aligned} \mathcal{L}_{ST}(\boldsymbol{\theta}; g^2, \mathbb{P}) &:= \mathbb{E}_{\mathbb{P}(\tau)} [\mathcal{L}(\boldsymbol{\theta}; g^2, \tau)] \\ &= \int_0^T \mathbb{P}(\tau) \left[\frac{1}{2} \int_\tau^T g^2(t) \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t} [\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2 - \|\log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] dt \right. \\ &\quad \left. - \int_\tau^T \mathbb{E}_{\mathbf{x}_t} [\text{div}(\mathbf{f}(\mathbf{x}_t, t))] dt - \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)] \right] d\tau \\ &= \int_0^T \left(\int_0^t \mathbb{P}(\tau) d\tau \right) \left[\frac{1}{2} g^2(t) \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t} [\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2 - \|\log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] \right. \\ &\quad \left. - \mathbb{E}_{\mathbf{x}_t} [\text{div}(\mathbf{f}(\mathbf{x}_t, t))] \right] dt - \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)] \\ &= \frac{1}{2} \int_0^T g_{\mathbb{P}}^2(t) \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t} [\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] dt + C, \end{aligned}$$

where

$$C = -\frac{1}{2} \int_0^T g_{\mathbb{P}}^2(t) \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t} [\|\log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] dt - \int_0^T \left(\int_0^t \mathbb{P}(\tau) d\tau \right) \mathbb{E}_{\mathbf{x}_t} [\text{div}(\mathbf{f}(\mathbf{x}_t, t))] dt - \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)].$$

If $\mathbf{f}(\mathbf{x}_t, t) = -\frac{1}{2}\beta(t)\mathbf{x}_t$, then we have

$$C = -\frac{d}{2} \int_0^T \left(\int_0^t \mathbb{P}(\tau) d\tau \right) \frac{g^2(t)}{\sigma^2(t)} dt + \frac{d}{2} \int_0^T \left(\int_0^t \mathbb{P}(\tau) d\tau \right) \beta(t) dt - \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)].$$

B. Theorems and Proofs

Lemma 1. For any $\tau \in [0, T]$,

$$\begin{aligned} \mathbb{E}_{\mathbf{x}_\tau} [-\log p_\tau^\theta(\mathbf{x}_\tau)] &\leq \mathcal{L}(\boldsymbol{\theta}; g^2, \tau) = \frac{1}{2} \int_\tau^T g^2(t) \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t} [\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2 \\ &\quad - \|\nabla_{\mathbf{x}_t} \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] dt - \int_\tau^T \mathbb{E}_{\mathbf{x}_t} [\text{div}(\mathbf{f}(\mathbf{x}_t, t))] dt - \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)]. \end{aligned}$$

Proof. Suppose $\boldsymbol{\mu}$ is the path measure of the forward SDE, and $\boldsymbol{\nu}_\theta$ is the path measure of the generative SDE. The restricted measure is defined by $\boldsymbol{\mu}|_{[\tau, T]}(\{F_t\}_{t=\tau}^T) := \boldsymbol{\mu}(\{F_t\}_{t=0}^T)$, where $F_t = \mathbb{R}^d$ if $t \in [0, \tau)$ and F_t is a measurable set in \mathbb{R}^d otherwise. The restricted measure of $\boldsymbol{\nu}_\theta$ is defined analogously. Then, by the data processing inequality, we get

$$D_{KL}(p_\tau \| p_\tau^\theta) \leq D_{KL}(\boldsymbol{\mu}|_{[\tau, T]} \| \boldsymbol{\nu}_\theta|_{[\tau, T]}). \quad (18)$$

Now, from the chain rule of KL divergences, we have

$$D_{KL}(\boldsymbol{\mu}|_{[\tau, T]} \| \boldsymbol{\nu}_\theta|_{[\tau, T]}) = D_{KL}(p_T \| \pi) + \mathbb{E}_{\mathbf{z} \sim p_T} \left[D_{KL}(\boldsymbol{\mu}|_{[\tau, T]}(\cdot | \mathbf{x}_T = \mathbf{z}) \| \boldsymbol{\nu}_\theta|_{[\tau, T]}(\cdot | \mathbf{x}_T = \mathbf{z})) \right]. \quad (19)$$

From the Girsanov theorem and the Martingale property, we get

$$D_{KL}(\boldsymbol{\mu}|_{[\tau, T]}(\cdot | \mathbf{x}_T = \mathbf{z}) \| \boldsymbol{\nu}_\theta|_{[\tau, T]}(\cdot | \mathbf{x}_T = \mathbf{z})) = \frac{1}{2} \int_\tau^T \mathbb{E}_{p_t(\mathbf{x}_t)} [g^2(t) \|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_t(\mathbf{x}_t)\|_2^2] dt, \quad (20)$$

and combining Eq. (18), (19) and (20), we have

$$D_{KL}(p_\tau \| p_\tau^\theta) \leq D_{KL}(p_T \| \pi) + \frac{1}{2} \int_\tau^T \mathbb{E}_{p_t(\mathbf{x}_t)} [g^2(t) \|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_t(\mathbf{x}_t)\|_2^2] dt. \quad (21)$$

Now, from

$$\begin{aligned} &\frac{1}{2} \int_\tau^T \mathbb{E}_{p_t(\mathbf{x}_t)} [g^2(t) [\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2 - \|\log p_t(\mathbf{x}_t)\|_2^2]] dt \\ &= \frac{1}{2} \int_\tau^T \mathbb{E}_{p_t(\mathbf{x}_t)} [g^2(t) \|\mathbf{s}_\theta(\mathbf{x}_t, t)\|_2^2 - 2g^2(t) \mathbf{s}_\theta(\mathbf{x}_t, t) \cdot \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)] dt \\ &= \frac{1}{2} \int_\tau^T \mathbb{E}_{p_t(\mathbf{x}_t)} [g^2(t) \|\mathbf{s}_\theta(\mathbf{x}_t, t)\|_2^2] dt - \int_\tau^T \int g^2(t) \mathbf{s}_\theta(\mathbf{x}_t, t) \cdot \nabla_{\mathbf{x}_t} p_t(\mathbf{x}_t) d\mathbf{x}_t dt \\ &= \frac{1}{2} \int_\tau^T \mathbb{E}_{p_t(\mathbf{x}_t)} [g^2(t) \|\mathbf{s}_\theta(\mathbf{x}_t, t)\|_2^2] dt - \int_\tau^T \int g^2(t) \mathbf{s}_\theta(\mathbf{x}_t, t) \cdot \nabla_{\mathbf{x}_t} \int p_r(\mathbf{x}_0) p_{0t}(\mathbf{x}_t | \mathbf{x}_0) d\mathbf{x}_0 d\mathbf{x}_t dt \\ &= \frac{1}{2} \int_\tau^T \mathbb{E}_{p_t(\mathbf{x}_t)} [g^2(t) \|\mathbf{s}_\theta(\mathbf{x}_t, t)\|_2^2] dt - \int_\tau^T \int g^2(t) \mathbf{s}_\theta(\mathbf{x}_t, t) \cdot \int p_r(\mathbf{x}_0) \nabla_{\mathbf{x}_t} p_{0t}(\mathbf{x}_t | \mathbf{x}_0) d\mathbf{x}_0 d\mathbf{x}_t dt \\ &= \frac{1}{2} \int_\tau^T \mathbb{E}_{p_r(\mathbf{x}_0) p_{0t}(\mathbf{x}_t | \mathbf{x}_0)} [g^2(t) [\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2 - \|\nabla_{\mathbf{x}_t} \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2]] dt, \end{aligned}$$

we can transform $\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_t(\mathbf{x}_t)\|_2^2$ into $\|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2$, Eq. (21) is equivalent to

$$\mathbb{E}_{p_\tau(\mathbf{x}_\tau)} [-\log p_\tau^\theta(\mathbf{x}_\tau)] \leq D_{KL}(p_T \| \pi) + \frac{1}{2} \int_\tau^T \mathbb{E}_{p_t(\mathbf{x}_t)} [g^2(t) \|\mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla \log p_t(\mathbf{x}_t)\|_2^2] dt + \mathcal{H}(p_\tau) \quad (22)$$

$$= D_{KL}(p_T \| \pi) + \frac{1}{2} \int_{\tau}^T \mathbb{E}_{p_t(\mathbf{x}_t)} [g^2(t) \|\mathbf{s}_{\theta}(\mathbf{x}_t, t) - \nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2 - \|\nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] dt \quad (23)$$

$$+ \frac{1}{2} \int_{\tau}^T \mathbb{E}_{p_t(\mathbf{x}_t)} [g^2(t) \nabla \log p_t(\mathbf{x}_t) \|_2^2] dt + \mathcal{H}(p_{\tau}). \quad (24)$$

Now, directly applying Theorem 4 of [Song et al. \(2021a\)](#), the entropy of $\mathcal{H}(p_{\tau})$ becomes

$$\mathcal{H}(p_{\tau}) = \mathcal{H}(p_T) - \frac{1}{2} \int_{\tau}^T \mathbb{E}_{p_t(\mathbf{x}_t)} [2\text{div}(\mathbf{f}(\mathbf{x}_t, t)) + g^2(t) \|\nabla \log p_t(\mathbf{x}_t)\|_2^2] dt. \quad (25)$$

Therefore, from Eq. (22) and (25), we get

$$\begin{aligned} \mathbb{E}_{p_{\tau}(\mathbf{x}_{\tau})} [-\log p_{\tau}^{\theta}(\mathbf{x}_{\tau})] &\leq \frac{1}{2} \int_{\tau}^T \mathbb{E}_{p_t(\mathbf{x}_t)} [g^2(t) \|\mathbf{s}_{\theta}(\mathbf{x}_t, t) - \nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2 - \|\nabla \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2] dt \\ &\quad - \int_{\tau}^T \mathbb{E}_{\mathbf{x}_t} [\text{div}(\mathbf{f}(\mathbf{x}_t, t))] dt - \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)]. \end{aligned}$$

□

Theorem 1. Suppose $\lambda(t)$ is a weighting function of the NCSN loss. If $\frac{\lambda(t)}{g^2(t)}$ is a nondecreasing and nonnegative absolutely continuous function on $[\epsilon, T]$ and zero on $[0, \epsilon]$, then

$$\begin{aligned} \mathcal{L}(\theta; \lambda, \epsilon) &\geq \int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' \mathbb{E}_{\mathbf{x}_{\tau}} [-\log p_{\tau}^{\theta}(\mathbf{x}_{\tau})] d\tau + \frac{\lambda(\epsilon)}{g^2(\epsilon)} \mathbb{E}_{\mathbf{x}_{\epsilon}} [-\log p_{\epsilon}^{\theta}(\mathbf{x}_{\epsilon})] \\ &\quad + \int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} - 1 \right) \mathbb{E}_{\mathbf{x}_{\tau}} [\text{div}(\mathbf{f}(\mathbf{x}_{\tau}, \tau))] d\tau + \left[\frac{\lambda(T)}{g^2(T)} - 1 \right] \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)]. \end{aligned}$$

Proof. We prove the theorem by using

$$\begin{aligned} \int_{\epsilon}^T \lambda(t) A(t) dt &= \int_{\epsilon}^T \left[\int_{\epsilon}^t \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' d\tau + \frac{\lambda(\epsilon)}{g^2(\epsilon)} \right] g^2(t) A(t) dt \\ &= \int_{\epsilon}^T \int_{\epsilon}^T 1_{[\epsilon, t]}(\tau) \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' g^2(t) A(t) d\tau dt + \frac{\lambda(\epsilon)}{g^2(\epsilon)} \int_{\epsilon}^T g^2(t) A(t) dt \\ &= \int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' \int_{\tau}^T g^2(t) A(t) dt d\tau + \frac{\lambda(\epsilon)}{g^2(\epsilon)} \int_{\epsilon}^T g^2(t) A(t) dt. \end{aligned} \quad (26)$$

By plugging $A(t) = \frac{1}{2} \mathbb{E}_{\mathbf{x}_t} [\|\mathbf{s}_{\theta}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2 - \|\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2]$ in Eq. (26), we have

$$\begin{aligned} \mathcal{L}(\theta; \lambda, \epsilon) &:= \frac{1}{2} \int_{\epsilon}^T \lambda(t) \mathbb{E}_{\mathbf{x}_t} [\|\mathbf{s}_{\theta}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2 - \|\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2] dt \\ &\quad - \int_{\epsilon}^T \mathbb{E}_{\mathbf{x}_t} [\text{div}(\mathbf{f}(\mathbf{x}_t, t))] dt - \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)] \\ &= \int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' \left[\frac{1}{2} \int_{\tau}^T g^2(t) \mathbb{E}_{\mathbf{x}_t} [\|\mathbf{s}_{\theta}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2 - \|\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2] dt \right. \\ &\quad \left. - \int_{\tau}^T \mathbb{E}_{\mathbf{x}_t} [\text{div}(\mathbf{f}(\mathbf{x}_t, t))] dt - \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)] \right] d\tau \\ &\quad + \frac{\lambda(\epsilon)}{g^2(\epsilon)} \left[\frac{1}{2} \int_{\epsilon}^T g^2(t) \mathbb{E}_{\mathbf{x}_t} [\|\mathbf{s}_{\theta}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2 - \|\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2] dt \right. \\ &\quad \left. - \int_{\epsilon}^T \mathbb{E}_{\mathbf{x}_t} [\text{div}(\mathbf{f}(\mathbf{x}_t, t))] dt - \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)] \right] \\ &\quad + \int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' \int_{\tau}^T \mathbb{E}_{\mathbf{x}_t} [\text{div}(\mathbf{f}(\mathbf{x}_t, t))] dt d\tau + \left(\frac{\lambda(\epsilon)}{g^2(\epsilon)} \right) \int_{\epsilon}^T \mathbb{E}_{\mathbf{x}_t} [\text{div}(\mathbf{f}(\mathbf{x}_t, t))] dt \end{aligned} \quad (27)$$

$$-\int_{\epsilon}^T \mathbb{E}_{\mathbf{x}_t} [\operatorname{div}(\mathbf{f}(\mathbf{x}_t, t))] dt + \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)] \left[\int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' d\tau + \frac{\lambda(\epsilon)}{g^2(\epsilon)} - 1 \right].$$

Also, plugging $A(t) = \frac{1}{g^2(t)} \mathbb{E}_{\mathbf{x}_t} [\operatorname{div}(\mathbf{f}(\mathbf{x}_t, t))]$ into Eq. (26), we have

$$\int_{\epsilon}^T \frac{\lambda(t)}{g^2(t)} \mathbb{E}_{\mathbf{x}_t} [\operatorname{div}(\mathbf{f}(\mathbf{x}_t, t))] = \int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' \int_{\tau}^T \mathbb{E}_{\mathbf{x}_t} [\operatorname{div}(\mathbf{f}(\mathbf{x}_t, t))] dt d\tau + \left(\frac{\lambda(\epsilon)}{g^2(\epsilon)} \right) \int_{\epsilon}^T \mathbb{E}_{\mathbf{x}_t} [\operatorname{div}(\mathbf{f}(\mathbf{x}_t, t))] dt. \quad (28)$$

Using Eq. (27) and (28), we get

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}; \lambda, \epsilon) &= \int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' \mathcal{L}(\boldsymbol{\theta}; g^2, \tau) d\tau + \frac{\lambda(\epsilon)}{g^2(\epsilon)} \mathcal{L}(\boldsymbol{\theta}; g^2, \epsilon) \\ &\quad + \int_{\epsilon}^T \left(\frac{\lambda(t)}{g^2(t)} - 1 \right) \mathbb{E}_{\mathbf{x}_t} [\operatorname{div}(\mathbf{f}(\mathbf{x}_t, t))] dt + \left[\frac{\lambda(T)}{g^2(T)} - 1 \right] \mathbb{E}_{\mathbf{x}_T} [\log \pi(\mathbf{x}_T)]. \end{aligned} \quad (29)$$

Then, applying Lemma 1 to Eq. (29) yields the desired result. \square

Corollary 1. Suppose $\lambda(t)$ is a weighting function of the NCSN loss. If $\frac{\lambda(t)}{g^2(t)}$ is a nondecreasing and nonnegative continuous function on $[\epsilon, T]$ and zero on $[0, \epsilon)$, then

$$\begin{aligned} \frac{1}{2} \int_{\epsilon}^T \lambda(t) \mathbb{E}_{\mathbf{x}_t} [\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2] dt + \frac{\lambda(T)}{g^2(T)} D_{KL}(p_T \| \pi) \\ \geq \int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' D_{KL}(p_{\tau} \| p_{\tau}^{\boldsymbol{\theta}}) d\tau + \frac{\lambda(\epsilon)}{g^2(\epsilon)} D_{KL}(p_{\epsilon} \| p_{\epsilon}^{\boldsymbol{\theta}}). \end{aligned}$$

Remark 1. A direct extension of the proof indicates that Theorem 1 still holds when $\frac{\lambda(t)}{g^2(t)}$ has finite jump on $[0, T]$.

Remark 2. The weight of $\frac{\lambda(T)}{g^2(T)}$ is the normalizing constant of the unnormalized truncation probability, \mathbb{P} .

Proof. By plugging $A(t) = \frac{1}{2} \mathbb{E}_{\mathbf{x}_t} [\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2]$ in Eq. (26) and using Lemma 1, we have

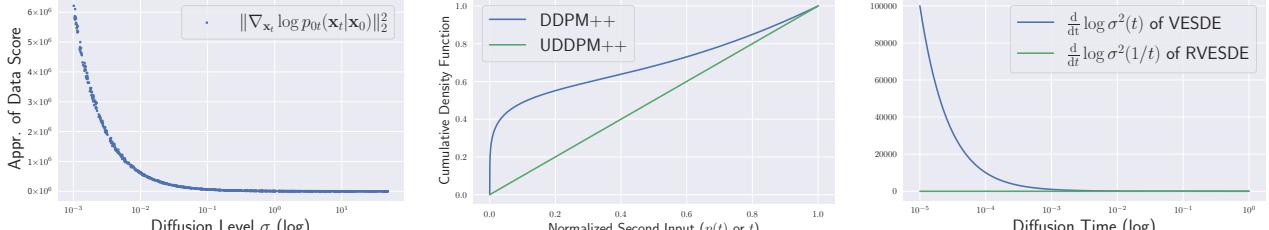
$$\begin{aligned} \frac{1}{2} \int_{\epsilon}^T \lambda(t) \mathbb{E}_{\mathbf{x}_t} [\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2] dt + \frac{\lambda(T)}{g^2(T)} D_{KL}(p_T \| \pi) \\ = \int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' \frac{1}{2} \int_{\tau}^T g^2(t) \mathbb{E}_{\mathbf{x}_t} [\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2] dt d\tau \\ + \left(\frac{\lambda(\epsilon)}{g^2(\epsilon)} \right) \frac{1}{2} \int_{\epsilon}^T g^2(t) \mathbb{E}_{\mathbf{x}_t} [\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2] dt + \frac{\lambda(T)}{g^2(T)} D_{KL}(p_T \| \pi) \\ \geq \int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' [D_{KL}(p_{\tau} \| p_{\tau}^{\boldsymbol{\theta}}) - D_{KL}(p_T \| \pi)] d\tau + \frac{\lambda(\epsilon)}{g^2(\epsilon)} [D_{KL}(p_{\epsilon} \| p_{\epsilon}^{\boldsymbol{\theta}}) - D_{KL}(p_T \| \pi)] + \frac{\lambda(T)}{g^2(T)} D_{KL}(p_T \| \pi) \\ = \int_{\epsilon}^T \left(\frac{\lambda(\tau)}{g^2(\tau)} \right)' D_{KL}(p_{\tau} \| p_{\tau}^{\boldsymbol{\theta}}) d\tau + \frac{\lambda(\epsilon)}{g^2(\epsilon)} D_{KL}(p_{\epsilon} \| p_{\epsilon}^{\boldsymbol{\theta}}). \end{aligned}$$

\square

C. Additional Score Architectures and SDEs

C.1. Additional Score Architectures: Unbounded Parametrization

From the released code of Song et al. (2021b), the NCSN++ network is modeled by $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \log \sigma(t))$, where the second argument is $\log \sigma(t)$ instead of t . Experiments with $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ or $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \sigma(t))$ were not as good as the parametrization of $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \log \sigma(t))$, and we analyze this experimental results from Lemma 2 and Proposition 1.



(a) Approximate data score diverges. (b) Cumulative density function of t and η . (c) VESDE violates geometric progression.

Figure 9: (a) The approximate data score, $\|\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2 = \|\nabla_{\mathbf{x}_t} \log \int p_r(\mathbf{x}_0) p_{0t}(\mathbf{x}_t | \mathbf{x}_0) d\mathbf{x}_0\|_2^2 \approx \|\nabla_{\mathbf{x}_t} \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2$, diverges as $t \rightarrow 0$. (b) Comparison of DDPM++ and UDDPM++ in terms of the cumulative density function of the second input. (c) VESDE violates geometric progression.

Proposition 1. Let $\mathcal{H}_{[1,\infty)} = \{\mathbf{s} : \mathbb{R}^d \times [1, \infty) \rightarrow \mathbb{R}^d, \mathbf{s} \text{ is locally Lipschitz}\}$. Suppose a continuous vector field \mathbf{v} defined on a d -dimensional open subset U of a compact manifold M is unbounded, and the projection of \mathbf{v} on each axis is locally integrable. Then, there exists $\mathbf{s} \in \mathcal{H}_{[1,\infty)}$ such that $\lim_{\eta \rightarrow \infty} \mathbf{s}(\mathbf{x}, \eta) = \mathbf{v}(\mathbf{x})$ a.e. on U .

The gradient of the log transition probability diverges at $t \approx 0$ theoretically (Section A.2) and empirically (Figure 9-(a)). Here, in high-dimensional space, $p_{0t}(\mathbf{x}_t | \mathbf{x}_0) / p_{0t}(\mathbf{x}_t | \mathbf{x}_0)$ with $\mathbf{x}_0 \neq \mathbf{x}'_0$ is either zero or infinity. Thus, the data score is nearly identical to the gradient of the log transition probability, $\|\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2 = \|\nabla_{\mathbf{x}_t} \log \int p_r(\mathbf{x}_0) p_{0t}(\mathbf{x}_t | \mathbf{x}_0) d\mathbf{x}_0\|_2^2 \approx \|\nabla_{\mathbf{x}_t} \log p_{0t}(\mathbf{x}_t | \mathbf{x}_0)\|_2^2$, and the observation of Figure 9-(a) is valid for the exact data score, as well.

Although Lemma 2 is based on $\mathbf{s}_\theta(\mathbf{x}_t, t)$, the identical result also holds for the parametrization of $\mathbf{s}_\theta(\mathbf{x}_t, \sigma(t))$, so it indicates that both $\mathbf{s}_\theta(\mathbf{x}_t, t)$ and $\mathbf{s}_\theta(\mathbf{x}_t, \sigma(t))$ cannot estimate the data score as $t \rightarrow 0$. On the other hand, Proposition 1 implies that there exists a score function that estimates the unbounded data score asymptotically, and Proposition 1 explains the reason why the parametrization of Song et al. (2021b), i.e., $\mathbf{s}_\theta(\mathbf{x}_t, \log \sigma(t))$, is successful on score estimation.

On top of that, we introduce another parametrization that particularly focuses on the score estimation near $t \approx 0$. We name Unbounded NCSN++ (UNCSN++) as the network of $\mathbf{s}_\theta(\mathbf{x}_t, \eta(t))$ with $\eta(t) = \begin{cases} \log \sigma(t) & \text{if } \sigma(t) \geq \sigma_0 \\ -\frac{c_1}{\sigma(t)} + c_2 & \text{if } \sigma(t) < \sigma_0 \end{cases}$ and Unbounded DDPM++ (UDDPM++) as the network of $\mathbf{s}_\theta(\mathbf{x}_t, \eta(t))$ with $\eta(t) := \int \frac{g^2(t)}{\sigma^2(t)} dt$.

In UNCSN++, c_1, c_2 and σ_0 are the hyperparameters. By acknowledging the parametrization of $\log \sigma(t)$, we choose σ_0 as 0.01. Also, to satisfy the continuously differentiability of $\eta(t)$, two hyperparameters c_1 and c_2 satisfy a system of equations with degree 2, so c_1 and c_2 are fully determined with this system of equations.

The choice of such $\eta(t)$ for UDDPM++ is expected to enhance the score estimation near $t \approx 0$ because the input of $\eta(t)$ is distributed uniformly when we draw samples from the importance weight. Concretely, when the sampling distribution on the diffusion time is given by $p_{iw}(t) \propto \frac{g^2(t)}{\sigma^2(t)}$, the η -distribution from the importance sampling becomes $p(\eta) \propto 1$, which is depicted in Figure 9-(b).

Proof of Proposition 1. Let h be a standard mollifier function. If $h_t(x) = t^{-n} h(x/t)$, then $v_t := h_t * v$ converges to v a.e. on U as $t \rightarrow 0$ (Theorem 7-(ii) of Appendix C in (Evans, 1998)). Therefore, if we define $s(\mathbf{x}, \eta) := v_{1/\eta}(\mathbf{x})$ on the domain of $v_{1/\eta}(\mathbf{x})$ and $s(\mathbf{x}, \eta) := 0$ elsewhere, then $s(\mathbf{x}, \eta) = v_{1/\eta}(\mathbf{x}) \rightarrow v(\mathbf{x})$ a.e. on U as $\eta \rightarrow \infty$.

Now, to show that $\mathbf{s}(\mathbf{x}, \eta)$ is locally Lipschitz, let $\tilde{M} \times [\eta, \bar{\eta}]$ be a compact subset of $\mathbb{R}^n \times [1, \infty)$. From $\|\mathbf{s}(\mathbf{x}_1, \eta_1) - \mathbf{s}(\mathbf{x}_2, \eta_2)\| = \|v_{1/\eta_1}(\mathbf{x}_1) - v_{1/\eta_2}(\mathbf{x}_2)\| \leq \|v_{1/\eta_1}(\mathbf{x}_1) - v_{1/\eta_1}(\mathbf{x}_2)\| + \|v_{1/\eta_1}(\mathbf{x}_2) - v_{1/\eta_2}(\mathbf{x}_2)\|$, if there exists $K_1, K_2 > 0$ such that $\|v_{1/\eta_1}(\mathbf{x}_1) - v_{1/\eta_1}(\mathbf{x}_2)\| \leq K_1 \|\mathbf{x}_1 - \mathbf{x}_2\|$ and $\|v_{1/\eta_1}(\mathbf{x}_1) - v_{1/\eta_2}(\mathbf{x}_1)\| \leq K_2 |\eta_1 - \eta_2|$ for all $\mathbf{x}_1, \mathbf{x}_2 \in \tilde{M}$ and $\eta_1, \eta_2 \in [\eta, \bar{\eta}]$, then $\mathbf{s}(\mathbf{x}, \eta) = v_{1/\eta}(\mathbf{x})$ is Lipschitz on $\tilde{M} \times [\eta, \bar{\eta}]$.

First, since $v_{1/\eta}$ is infinitely differentiable on its domain (Theorem 7-(i) of Appendix C in (Evans, 1998)) and $\eta \in [\eta, \bar{\eta}]$, there exists $K_1 > 0$ such that $\|v_{1/\eta}(\mathbf{x}_1) - v_{1/\eta}(\mathbf{x}_2)\| \leq K_1 \|\mathbf{x}_1 - \mathbf{x}_2\|$. Second, the mollifier satisfies the uniform convergence on any compact subset of U (Theorem 7-(iii) of Appendix C in (Evans, 1998)), which leads that $\|v_{1/\eta_1}(\mathbf{x}) - v_{1/\eta_2}(\mathbf{x})\| \leq K_2 \left| \frac{1}{\eta_1} - \frac{1}{\eta_2} \right| = K_2 \frac{|\eta_1 - \eta_2|}{\eta_1 \eta_2} \leq K_3 |\eta_1 - \eta_2|$ for some $K_2, K_3 > 0$. Therefore, \mathbf{s} becomes an element of $\mathcal{H}_{[1,\infty)}$. \square

C.2. Additional SDE: Reciprocal VESDE

VESDE assumes $g(t) = \sigma_{\min}(\frac{\sigma_{\max}}{\sigma_{\min}})^t \sqrt{2 \log \frac{\sigma_{\max}}{\sigma_{\min}}}$. Then, the variance of the transition probability $p_{0t}(\mathbf{x}_t | \mu_{VE}(t)\mathbf{x}_0, \sigma_{VE}^2(t))$ becomes $\sigma_{VE}^2(t) = \int_0^t g^2(s) ds = \sigma_{\min}^2[(\frac{\sigma_{\max}}{\sigma_{\min}})^{2t} - 1]$ if the diffusion starts from $t = 0$ with the initial condition of $\mathbf{x}_0 \sim p_r$. VESDE was originally introduced in [Song & Ermon \(2020\)](#) in order to satisfy the geometric property for its smooth transition of the distributional shift. Mathematically, the variance is geometric if $\frac{d}{dt} \log \sigma_{VE}^2(t)$ is a constant, but VESDE loses the geometric property as illustrated in Figure 9-(c).

To attain the geometric property in VESDE, VESDE approximates the variance to be $\tilde{\sigma}_{VE}^2(t) = \sigma_{\min}^2(\frac{\sigma_{\max}}{\sigma_{\min}})^{2t}$ by omitting 1 from $\sigma_{VE}^2(t)$. However, this approximation leads that \mathbf{x}_t is not converging to \mathbf{x}_0 in distribution because $\sigma_{\min}^2(\frac{\sigma_{\max}}{\sigma_{\min}})^{2t} \rightarrow \sigma_{\min}^2 \neq 0$ as $t \rightarrow 0$. Indeed, a bit stronger claim is possible:

Proposition 2. *There is no SDE that has the stochastic process $\{\mathbf{x}_t\}_{t \in [0, T]}$, defined by a transition probability $p_{0t}(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \mathbf{x}_0, \sigma_{\min}^2(\frac{\sigma_{\max}}{\sigma_{\min}})^{2t} \mathbf{I})$, as the solution.*

Proposition 2 indicates that if we approximate the variance by $\sigma_{VE}^2(t)$, then the reverse diffusion process cannot be modeled by a generative process.

Rigorously, however, if the diffusion process starts from $t = -\infty$, rather than $t = 0$, then the variance of the transition probability becomes $\sigma_{VE, -\infty}^2(t) = \int_{-\infty}^t g^2(s) ds = \sigma_{\min}^2(\frac{\sigma_{\max}}{\sigma_{\min}})^{2t}$, which is exactly the variance $\tilde{\sigma}_{VE}^2(t)$. Therefore, VESDE can be considered as a diffusion process starting from $t = -\infty$.

From this point of view, we introduce a SDE that satisfies the geometric progression property starting from $t = 0$. We name a new SDE as the Reciprocal VE SDE (RVESDE). RVESDE has the identical form of SDE, $d\mathbf{x}_t = g_{RVE}(t) d\mathbf{w}_t$, with

$$g_{RVE}(t) := \begin{cases} \sigma_{\max} (\frac{\sigma_{\min}}{\sigma_{\max}})^{\frac{\epsilon}{t}} \frac{\sqrt{2\epsilon \log(\frac{\sigma_{\max}}{\sigma_{\min}})}}{t} & \text{if } t > 0, \\ 0 & \text{if } t = 0. \end{cases}$$

Then, the transition probability of RVESDE becomes

$$p_{0t}(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t; \mathbf{x}_0, \sigma_{\max}^2 \left(\frac{\sigma_{\min}}{\sigma_{\max}}\right)^{\frac{2\epsilon}{t}} \mathbf{I}\right).$$

As illustrated in Figure 9-(c), RVESDE attains the geometric property at the expense of having reciprocated time, $1/t$. Also, RVESDE satisfies $\sigma_{RVE}^2(\epsilon) = \sigma_{\min}^2$ and $\sigma_{RVE}^2(T) \approx \sigma_{\max}^2$. The existence and uniqueness of solution for RVESDE is guaranteed by Theorem 5.2.1 in [\(Oksendal, 2013\)](#).

D. Implementation Details

D.1. Experimental Details

Training Throughout the experiments, we train our model with a learning rate of 0.0002, warmup of 5000 iterations, and gradient clipping by 1. For UNCSN++, we take $\sigma_{\min} = 10^{-3}$, and for NCSN++, we take $\sigma_{\min} = 10^{-2}$. On ImageNet32 training of the likelihood weighting and the variance weighting without Soft Truncation, we take $\epsilon = 5 \times 10^{-5}$, following the setting of [Song et al. \(2021a\)](#). Otherwise, we take $\epsilon = 10^{-5}$. For other hyperparameters, we run our experiments according to [Song et al. \(2021b;a\)](#).

On datasets of resolution 32×32 , we use the batch size of 128, which consumes about 48Gb GPU memory. On STL-10 with resolution 48×48 , we use the batch size of 192, and on datasets of resolution 64×64 , we experiment with 128 batch size. The batch size for the datasets of resolution 256×256 is 40, which takes nearly 120Gb of GPU memory. On the dataset of 1024×1024 resolution, we use the batch size of 16, which takes around 120Gb of GPU memory. We use five NVIDIA RTX-3090 GPU machines to train the model exceeding 48Gb, and we use a pair of NVIDIA RTX-3090 GPU machines to train the model that consumes less than 48Gb.

Evaluation We apply the EMA with rate of 0.999 on NCSN++/UNCSN++ and 0.9999 on DDPM++/UDDPM++. For the density estimation, we obtain the NLL performance by the Instantaneous Change of Variable ([Song et al., 2021b; Chen et al., 2018](#)). We choose $[\epsilon = 10^{-5}, T = 1]$ to integrate the instantaneous change-of-variable of the probability flow as default, even for the ImageNet32 dataset. In spite that [Song et al. \(2021b;a\)](#) integrates the change-of-variable formula with the

Table 10: Ablation study of Soft Truncation with/without the reconstruction term when training on CIFAR-10 trained with DDPM++ (VP).

Loss	Soft Truncation	Reconstruction Term for Training	NLL		NELBO		ODE
			$\mathbb{E}_{\mathbf{x}_0}[-\log p_{\epsilon}^{\theta}(\mathbf{x}_0)]$	$\mathbb{E}_{\mathbf{x}_{\epsilon}}[-\log p_{\epsilon}^{\theta}(\mathbf{x}_{\epsilon})] + R_{\epsilon}(\theta)$	$\mathcal{L}(\theta; g^2, \epsilon)$ (without residual)	$\mathcal{L}(\theta; g^2, \epsilon) + R_{\epsilon}(\theta)$ (with residual)	
$\mathcal{L}(\theta; g^2, \epsilon)$	\times	\times	2.97	3.03	3.11	3.13	6.70
$\mathcal{L}(\theta; g^2, \epsilon) + \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_{\epsilon}}[-\log p(\mathbf{x}_0 \mathbf{x}_{\epsilon})]$	\times	\checkmark	3.01	2.99	3.07	3.09	6.93
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1) = \mathbb{E}_{\mathbb{P}_1(\tau)}[\mathcal{L}(\theta; g^2, \tau)] = \mathbb{E}_{\mathbb{P}_1(\tau)}[\mathcal{L}(\theta; g^2, \tau)]$	\checkmark	\times	2.98	3.01	3.08	3.08	3.96
$\mathbb{E}_{\mathbb{P}_1(\tau)}[\mathcal{L}(\theta; g^2, \tau) + R_{\tau}(\theta)]$	\checkmark	\checkmark	2.95	2.98	3.04	3.04	4.23

starting variable to be \mathbf{x}_0 , Table 5 of Kim et al. (2022) analyzes that there are significant difference between starting from \mathbf{x}_{ϵ} and \mathbf{x}_0 , if ϵ is not small enough. Therefore, we follow Kim et al. (2022) to compute $\mathbb{E}_{\mathbf{x}_{\epsilon}}[-\log p_{\epsilon}^{\theta}(\mathbf{x}_{\epsilon})]$. However, to compare with the baseline models, we also evaluate the way Song et al. (2021b;a) and Vahdat et al. (2021) compute NLL. We denote the way of Kim et al. (2022) as *after correction* and Song et al. (2021a) as *before correction*, throughout the appendix. We dequantize the data variable by the uniform dequantization (Ho et al., 2019) for both after-and-before corrections. In the main paper, we only report the *after correction* performances.

For the sampling, we apply the Predictor-Corrector (PC) algorithm introduced in Song et al. (2021b). We set the signal-to-noise ratio as 0.16 on 32×32 datasets, 0.17 on 48×48 and 64×64 datasets, 0.075 on 256×256 sized datasets, and 0.15 on 1024×1024 . On datasets less than 256×256 resolution, we iterate 1,000 steps for the PC sampler, while we apply 2,000 steps on the other high-dimensional datasets. Throughout the experiments for VESDE, we use the reverse diffusion (Song et al., 2021b) for the predictor algorithm and the annealed Langevin dynamics (Welling & Teh, 2011) for the corrector algorithm. For VPSDE, we use the Euler-Maruyama for the predictor algorithm, and we do not use any corrector algorithm.

We compute the FID score (Song et al., 2021b) based on the modified Inception V1 network³ using the tensorflow-gan package for CIFAR-10 dataset, and we use the clean-FID (Parmar et al., 2022) based on the Inception V3 network (Szegedy et al., 2016) for the remaining datasets. We note that FID computed by (Parmar et al., 2022) reports a higher FID score compared to the original FID calculation⁴.

E. Additional Experimental Results

E.1. Ablation Study on Reconstruction Term

Table 10 presents that the training with the reconstruction term outperforms the training without the reconstruction term on NLL/NELBO with the sacrifice on sample generation. If τ is fixed as ϵ , then the bound

$$\mathbb{E}_{\mathbf{x}_0}[-\log p_0^{\theta}(\mathbf{x}_0)] \leq \mathcal{L}(\theta; g^2, \tau) + \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_{\tau}}[-\log p(\mathbf{x}_0 | \mathbf{x}_{\tau})]$$

is tight enough to estimate the negative log-likelihood. However, if τ is a subject of random variable, then the bound is not tight to the negative log-likelihood, as evidenced in Figure 1-(b). On the other hand, if we do not count the reconstruction, then the bound becomes

$$\mathbb{E}_{\mathbf{x}_0}[-\log p_{\tau}^{\theta}(\mathbf{x}_{\tau})] \leq \mathcal{L}(\theta; g^2, \tau),$$

up to a constant, and this bound becomes tight regardless of τ , which is evidenced in Figure 1-(c). This is why we call Soft Truncation as Maximum Perturbed Likelihood Estimation (MPLE).

E.2. Full Tables

Tables 11, 12, 13, 14, and 15 present the full list of performances for Soft Truncation.

³<https://tfhub.dev/tensorflow/tfgan/eval/inception/1>

⁴See <https://github.com/GaParmar/clean-fid> for the detailed experimental results.

Table 11: Ablation study of Soft Truncation for various weightings on CIFAR-10 and ImageNet32 with DDPM++ (VP).

Dataset	Loss	Soft Truncation	NLL		NELBO		FID
			after correction	before correction	with residual	without residual	ODE
CIFAR-10	$\mathcal{L}(\theta; g^2, \epsilon)$	✗	3.03	2.97	3.13	3.11	6.70
	$\mathcal{L}(\theta; \sigma^2, \epsilon)$	✗	3.21	3.16	3.34	3.32	3.90
	$\mathcal{L}(\theta; g_{\mathbb{P}_1}^2, \epsilon)$	✗	3.06	3.02	3.18	3.14	6.11
	$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$	✓	3.01	2.98	3.08	3.08	3.96
ImageNet32	$\mathcal{L}(\theta; g^2, \epsilon)$	✗	3.92	3.90	3.94	3.95	12.68
	$\mathcal{L}(\theta; \sigma^2, \epsilon)$	✗	3.95	3.96	4.00	4.01	9.22
	$\mathcal{L}(\theta; g_{\mathbb{P}_1}^2, \epsilon)$	✗	3.93	3.92	3.97	3.98	11.89
	$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$	✓	3.90	3.87	3.92	3.92	9.52
	$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_{0.9})$	✓	3.90	3.88	3.91	3.91	8.42

Table 12: Ablation study of Soft Truncation for various model architectures and diffusion SDEs on CelebA.

SDE	Model	Loss	NLL		NELBO		FID	
			after correction	before correction	with residual	without residual	PC	ODE
VE	NCSN++	$\mathcal{L}(\theta; \sigma^2, \epsilon)$	3.41	2.37	3.42	3.96	3.95	-
		$\mathcal{L}_{ST}(\theta; \sigma^2, \mathbb{P}_2)$	3.44	2.42	3.44	3.97	2.68	-
RVE	UNCSN++	$\mathcal{L}(\theta; g^2, \epsilon)$	2.01	1.96	2.01	2.17	3.36	-
		$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_2)$	1.97	1.91	2.02	2.18	1.92	-
DDPM++		$\mathcal{L}(\theta; \sigma^2, \epsilon)$	2.14	2.07	2.21	2.22	3.03	2.32
		$\mathcal{L}_{ST}(\theta; \sigma^2, \mathbb{P}_1)$	2.17	2.08	2.29	2.26	2.88	1.90
UDDPM++		$\mathcal{L}(\theta; \sigma^2, \epsilon)$	2.11	2.07	2.20	2.21	3.23	4.72
		$\mathcal{L}_{ST}(\theta; \sigma^2, \mathbb{P}_1)$	2.16	2.08	2.28	2.25	2.22	1.94
VP	DDPM++	$\mathcal{L}(\theta; g^2, \epsilon)$	2.00	1.93	2.09	2.09	5.31	3.95
		$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$	2.00	1.94	2.11	2.11	4.50	2.90
UDDPM++		$\mathcal{L}(\theta; g^2, \epsilon)$	1.98	1.95	2.12	2.15	4.65	3.98
		$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$	2.00	1.94	2.10	2.10	4.45	2.97

E.3. Generated Samples

Figure 10 shows how images are created from the trained model, and Figures from 11 to 16 present non-cherry picked generated samples of the trained model.

Table 13: Ablation study of Soft Truncation for various σ_{min} (equivalently, ϵ) on CIFAR-10 with UNCSN++ (RVE).

Loss	ϵ	NLL		NELBO		FID
		after correction	before correction	with residual	without residual	
$\mathcal{L}(\theta; g^2, \epsilon)$	10^{-2}	4.64	4.02	4.69	5.20	38.82
	10^{-3}	3.51	3.20	3.52	3.90	6.21
	10^{-4}	3.05	2.98	3.08	3.24	6.33
	10^{-5}	3.03	2.97	3.13	3.11	6.70
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$	10^{-2}	4.65	4.03	4.69	5.20	39.83
	10^{-3}	3.51	3.21	3.52	3.88	5.14
	10^{-4}	3.05	2.98	3.08	3.24	4.16
	10^{-5}	3.01	2.98	3.08	3.08	3.96

 Table 14: Ablation study of Soft Truncation for various \mathbb{P}_k on CIFAR-10 trained with DDPM++ (VP).

Loss	NLL		NELBO		FID
	after correction	before correction	with residual	without residual	
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_0)$	3.24	3.16	3.39	3.34	6.27
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_{0.8})$	3.03	3.00	3.05	3.05	3.61
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_{0.9})$	3.03	2.99	3.13	3.13	3.45
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_1)$	3.01	2.98	3.08	3.08	3.96
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_{1.1})$	3.02	2.99	3.09	3.10	3.98
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_{1.2})$	3.03	2.99	3.09	3.09	3.98
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_2)$	3.01	2.97	3.10	3.09	6.31
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_3)$	3.02	2.96	3.09	3.09	6.54
$\mathcal{L}_{ST}(\theta; g^2, \mathbb{P}_\infty)$ $= \mathcal{L}(\theta; g^2, \epsilon)$	3.01	2.95	3.09	3.07	6.70

 Table 15: Ablation study of Soft Truncation for CIFAR-10 trained with DDPM++ when a diffusion is combined with a normalizing flow (Kim et al., 2022). We use $\mathbb{P}([a, b]) = \frac{1}{2}1_{[a,b]}(\epsilon) + \frac{1}{2}\mathbb{P}_{0.9}([a, b])$.

Loss	NLL		NELBO		FID
	after correction	before correction	with residual	without residual	
$\mathcal{L}(\theta; g^2, \epsilon)$	2.97	2.94	2.97	2.96	6.06
$\mathcal{L}(\theta; \sigma^2, \epsilon)$	3.17	3.11	3.23	3.18	3.61
$\mathcal{L}(\theta; g^2, \mathbb{P})$	3.01	2.98	3.02	3.01	3.89



Figure 10: Image generation by denoising via predictor-corrector sampler.

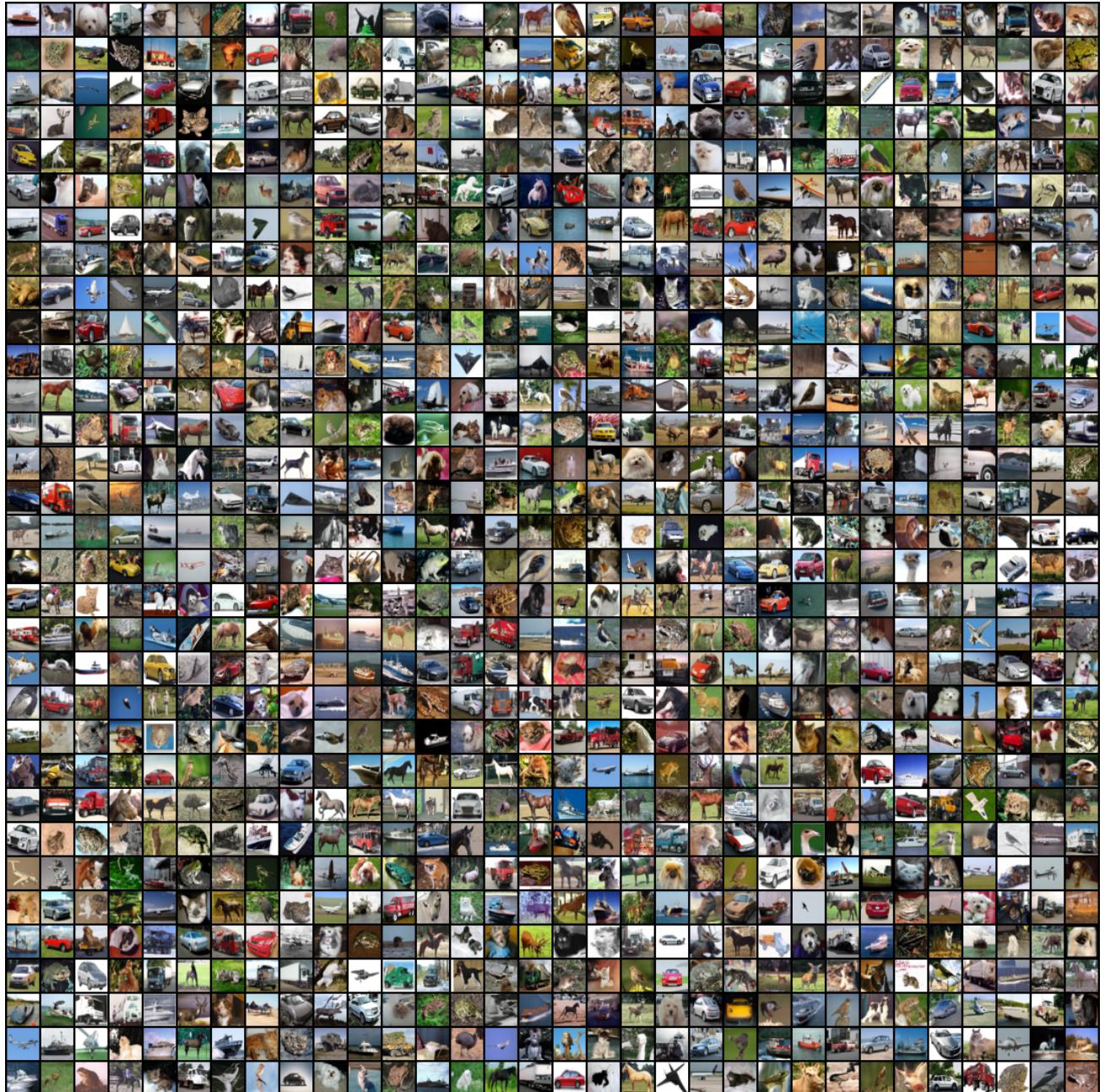


Figure 11: Random samples of CIFAR-10.



Figure 12: Random samples on CelebA.



Figure 13: Random samples on LSUN Bedroom.



Figure 14: Random samples on LSUN Church.



Figure 15: Random samples on FFHQ 256.



Figure 16: Random samples on CelebA-HQ 256.