Improved Techniques for Maximum Likelihood Estimation for Diffusion ODEs

Kaiwen Zheng* 1 Cheng Lu* 1 Jianfei Chen 1 Jun Zhu 1 2

Abstract

Diffusion models have exhibited excellent performance in various domains. The probability flow ordinary differential equation (ODE) of diffusion models (i.e., diffusion ODEs) is a particular case of continuous normalizing flows (CNFs), which enables deterministic inference and exact likelihood evaluation. However, the likelihood estimation results by diffusion ODEs are still far from those of the state-of-the-art likelihood-based generative models. In this work, we propose several improved techniques for maximum likelihood estimation for diffusion ODEs, including both training and evaluation perspectives. For training, we propose velocity parameterization and explore variance reduction techniques for faster convergence. We also derive an error-bounded high-order flow matching objective for finetuning, which improves the ODE likelihood and smooths its trajectory. For evaluation, we propose a novel training-free truncated-normal dequantization to fill the training-evaluation gap commonly existing in diffusion ODEs. Building upon these techniques, we achieve state-of-the-art likelihood estimation results on image datasets (2.56 on CIFAR-10, 3.43/3.69 on ImageNet-32) without variational dequantization or data augmentation.

1. Introduction

Likelihood is an important metric to evaluate density estimation models, and accurate likelihood estimation is the key for many applications such as data compression (Ho et al., 2021; Helminger et al., 2020; Kingma et al., 2021; Yang & Mandt, 2022), anomaly detection (Chen et al., 2018c; Dias et al., 2020) and out-of-distribution detection (Serrà et al., 2020; Xioa et al., 2020). Many deep generative models can compute tractable likelihood, including autoregressive models (Oord et al., 2016; Salimans et al., 2017; Chen et al., 2018b), variational auto-encoders (VAE) (Kingma & Welling, 2014; Vahdat & Kautz, 2020), normalizing flows (Dinh et al., 2017; Kingma & Dhariwal, 2018; Ho et al., 2019) and diffusion models (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020; Song et al., 2021c;a; Karras et al., 2022). Among these models, recent work named variational diffusion models (VDM) (Kingma et al., 2021) achieves state-of-the-art likelihood estimation performance on standard image density estimation benchmarks, which is a variant of diffusion models.

There are two types of diffusion models, one is based on the reverse stochastic differential equation (SDE) (Song et al., 2021c), named as diffusion SDE; the other is based on the probability flow ordinary differential equation (ODE) (Song et al., 2021c), named as diffusion ODE. These two types of diffusion models define and evaluate the likelihood in different manners: diffusion SDE can be understood as an infinitely-deep VAE (Huang et al., 2021) and can only compute a variational lower bound of the likelihood (Song et al., 2021c; Kingma et al., 2021); while diffusion ODE is a variant of continuous normalizing flows (Chen et al., 2018a) and can compute the exact likelihood by ODE solvers. Thus, it is natural to hypothesize that the likelihood performance of diffusion ODEs may be better than that of diffusion SDEs. However, all existing methods for training diffusion ODEs (Song et al., 2021b; Lu et al., 2022a; Lipman et al., 2022; Albergo & Vanden-Eijnden, 2022; Liu et al., 2022b) cannot even achieve a comparable likelihood performance with VDM, which belongs to diffusion SDEs. It still remains largely open whether diffusion ODEs are also great likelihood estimators.

Real-world data is usually discrete, and evaluating the likelihood of discrete data by diffusion ODEs needs to first perform a dequantization process (Dinh et al., 2017; Salimans et al., 2017) to make sure the input data of diffusion ODEs is continuous. In this work, we observe that previous likelihood evaluation of diffusion ODEs has flaws in the dequantization process: the uniform dequantization (Song et al., 2021b) causes a large training-evaluation gap, and the variational dequantization (Ho et al., 2019; Song et al., 2021b) requires additional training overhead and is hard to train to the optimal.
In this work, we propose several improved techniques, including both the evaluation perspective and training perspective, to allow the likelihood estimation by diffusion ODEs to outperform the existing state-of-the-art likelihood estimators. In the aspect of evaluation, we propose a training-free dequantization method dedicated to diffusion models by a carefully-designed truncated-normal distribution, which can fit diffusion ODEs well and improve the likelihood evaluation by a large margin compared to uniform dequantization. We also introduce an importance weighted likelihood estimator to get a tighter bound. In the aspect of training, we split our training into pretraining and finetuning phases. For pretraining, we propose a new model parameterization method including velocity parameterization, which is an extended version of flow matching (Lipman et al., 2022) with practical modifications, and log-signal-to-noise-ratio (2.56 on CIFAR-10, 3.43/3.69 on ImageNet-32), surpassing the previous best ODEs of 2.90 and 3.48/3.82, with the superiority that we use no data augmentation and throw away the need for training variational dequantization models.

2. Diffusion Models

2.1. Diffusion ODEs and Maximum Likelihood Training

Suppose we have a \(d\)-dimensional data distribution \(q_0(x_0)\). Diffusion models (Ho et al., 2020; Song et al., 2021c) gradually diffuse the data by a forward stochastic differential equation (SDE) starting from \(x_0 \sim q_0(x_0)\):

\[
\frac{dx_t}{dt} = f(t)x_t + q(t) \xi, \quad x_0 \sim q_0(x_0), \tag{1}
\]

where \(f(t), q(t) \in \mathbb{R}\) are manually designed noise schedules and \(\xi \in \mathbb{R}^d\) is a standard Wiener process. The forward process \(\{x_t\}_{t \in [0,T]}\) is accompanied with a series of marginal distributions \(\{q_t\}_{t \in [0,T]}\), so that \(q_T(x_T) \approx N(x_T; 0, \sigma_T^2 I)\) with some constant \(\sigma_T > 0\). Since this is a simple linear SDE, the transition kernel is an analytical Gaussian (Song et al., 2021c): \(q_T(x_T | x_0) = N(x_T; x_0, \sigma_T^2 I)\), where the coefficients satisfy \(f(t) = \frac{d}{dt} \log \alpha_t, \quad g^2(t) = \frac{d^2}{dt^2} \log \alpha_t - 2 \frac{d}{dt} \frac{\alpha_t}{\sigma_t^2} \) (Kingma et al., 2021). Under some regularity conditions (Anderson, 1982), the forward process has an equivalent probability flow ODE (Song et al., 2021c):

\[
\frac{dx_t}{dt} = f(t)x_t - \frac{1}{2}g^2(t)\nabla_x \log q_t(x_t), \tag{2}
\]

which produces the same marginal distribution \(q_t\) at each time \(t\) as that in Eqn. (1). The only unknown term \(\nabla_x \log q_t(x_t)\) is the score function of \(q_t\). By parameterizing a score network \(s_0(x_t, t)\) to predict the time-dependent \(\nabla_x \log q_t(x_t)\), we can replace the true score function, resulting in the diffusion ODE (Song et al., 2021c):

\[
\frac{dx_t}{dt} = f(t)x_t - \frac{1}{2}g^2(t)s_0(x_t, t), \tag{3}
\]

with the associated marginal distributions \(\{p_t\}_{t \in [0,T]}\). Diffusion ODEs are special cases of continuous normalizing flows (CNFs) (Chen et al., 2018a), thus can perform exact inference of the latents and exact likelihood evaluation.

Though traditional maximum likelihood training methods for CNFs (Grathwohl et al., 2019) are feasible for diffusion ODEs, the training costs of these methods are quite expensive and hard to scale up because of the requirement of solving ODEs at each iteration. Instead, a more practical way is to match the generative probability flow \(\{p_t\}_{t \in [0,T]}\) with \(\{q_t\}_{t \in [0,T]}\) by a simulation-free approach. Specifically, Lu et al. (2022a) proves that \(D_{KL}(q_0 \parallel p_0^{ODE})\) can be formulated by \(D_{KL}(q_0 \parallel p_T^{ODE}) = D_{KL}(q_T \parallel p_T^{ODE}) + J_{ODE}(\theta)\), where

\[
J_{ODE}(\theta) := \frac{1}{2} \int_0^T g(t)^2 \mathbb{E}_{q_t(x_t)}[(s_0(x_t, t) - \nabla_x \log q_t(x_t))^\top (\nabla_x \log p_t(x_t) - \nabla_x \log q_t(x_t))] \, dt \tag{4}
\]

However, computing \(\nabla_x \log p_t(x_t)\) requires solving another ODE and is also expensive (Lu et al., 2022a). To minimize \(J_{ODE}(\theta)\) in a simulation-free manner, Lu et al. (2022a) also proposes a combination of \(g^2(t)\) weighted first-order and high-order score matching objectives. Particularly, the first-order score matching objective is

\[
J_{SM}(\theta) := \int_0^T \frac{g^2(t)}{2\sigma_t^2} \mathbb{E}_{x_0, \xi}[\|\sigma_t s_0(x_t, t) + \xi\|^2_2] \, dt, \tag{5}
\]

where \(x_t = \alpha_t x_0 + \sigma_t \xi, \quad x_0 \sim q_0(x_0)\) and \(\xi \sim N(\epsilon \mathbf{I})\).

2.2. Log-SNR Timed Diffusion Models

Diffusion models have manually designed noise schedule \(\alpha_t, \sigma_t\), which has high freedom and affects the performance. Even for restricted design space such as Variance Preserving (VP) (Song et al., 2021c), which constrains the noise schedule by \(\alpha_t^2 + \sigma_t^2 = 1\), we could still have various choices about how fast \(\alpha_t, \sigma_t\) changes w.r.t time \(t\). To decouple the specific schedule form, variational diffusion
models (VDM) (Kingma et al., 2021) use a negative log-signal-to-noise-ratio (log-SNR) for the time variable and can greatly simplify both noise schedules and training objectives. Specifically, denote $\gamma_t = -\log\text{-SNR}(t) = -\log \frac{\sigma_t^2}{\sigma_0^2}$, the change-of-variable relation from $\gamma$ to $t$ is

$$\frac{d\gamma}{dt} = \frac{q^2(t)}{\sigma_t^2},$$

(6)

and replace the time subscript with $\gamma$, we get the simplified score matching objective with likelihood weighting:

$$J_{\text{SM}}(\theta) = \frac{1}{2} \int_{\gamma_0}^{\gamma_T} \mathbb{E}_{x_0, \epsilon} \left[ \| \sigma_1 s_\theta(x, \gamma) + \epsilon \|_2^2 \right] d\gamma$$

(7)

This result is in accordance with the continuous diffusion loss in Kingma et al. (2021).

### 2.3. Dequantization for Density Estimation

Many real-world datasets usually contain discrete data, such as images or texts. In such cases, learning a continuous density model to these discrete data points will cause degenerate results (Uria et al., 2013) and cannot provide meaningful density estimations. A common solution is dequantization (Dinh et al., 2017; Salimans et al., 2017; Ho et al., 2019). Specifically, suppose $x_0$ is 8-bit discrete data scaled to $[-1, 1]$. Dequantization methods assume that we have trained a continuous model distribution $p_{\text{model}}$ for $x_0$, and define the discrete model distribution by

$$p_{\text{model}}(x_0) := \int_{[-\frac{1}{2^8}, \frac{1}{2^8})^d} p_{\text{model}}(x_0 + u) du.$$

To train $p_{\text{model}}(x_0)$ by maximum likelihood estimation, variational dequantization (Ho et al., 2019) introduces a dequantization distribution $q(u|x_0)$ and jointly train $p_{\text{model}}$ and $q(u|x_0)$ by a variational lower bound:

$$\log p_{\text{model}}(x_0) \geq \mathbb{E}_{q(u|x_0)} \left[ \log p_{\text{model}}(x_0 + u) - \log q(u|x_0) \right].$$

A simple way for $q(u|x_0)$ is uniform dequantization, where we set $q(u|x_0) = U(-\frac{1}{2^8}, \frac{1}{2^8})$.

### 3. Diffusion ODEs with Truncated-Normal Dequantization

In this section, we discuss the challenges of training diffusion ODEs with dequantization and propose a training-free dequantization method for diffusion ODEs.

#### 3.1. Challenges for Diffusion ODEs with Dequantization

We first discuss the challenges for diffusion ODEs with dequantization in this section.

**Truncation introduces an additional gap.** Theoretically, we want to train diffusion ODEs by minimizing $D_{\text{KL}}(q_\epsilon \| p_\epsilon)$ and use $p_\epsilon(x_0)$ for the continuous model distribution. However, as $\sigma_0 = 0$, we have $\gamma_0 = -\infty$. Due to this, it is shown in previous work (Song et al., 2021c; Kim et al., 2022) that there are numerical issues near $t = 0$ for both training and sampling, so we cannot directly compute the model distribution $p_\epsilon$ at time 0. In practice, a common solution is to choose a small starting time $\epsilon > 0$ for improving numerical stability. The training objective then becomes minimizing $D_{\text{KL}}(q_\epsilon \| p_\epsilon)$, which is equivalent to

$$\max_{\theta} \mathbb{E}_{q_\epsilon(x_0), p_\epsilon(x_0)} \left[ \log p_\epsilon(x_0) \right],$$

(8)

and $\mathbb{E}_{q_\epsilon(x_0)} \log p_\epsilon(x_0)$ is directly used to evaluate the data likelihood. However, as $p_\epsilon \neq p_0$, such a method will introduce an additional gap due to the mismatch between training ($\mathbb{E}_{q_\epsilon(x_0)} \log p_\epsilon(x_0)$) and testing ($\mathbb{E}_{q_\epsilon(x_0)} \log p_\epsilon(x_0)$), which may degrade the likelihood evaluation performance.

**Uniform dequantization causes a train-test mismatch.** After choosing $\epsilon$, the continuous model distribution is defined by $p_{\text{model}}(x) := p_\epsilon(x)$. Let $q(u|x_0)$ be a dequantization distribution with support over $u \in [-\frac{1}{2^8}, \frac{1}{2^8})^d$. The variational lower bound for the discrete model density $P_0(x_0)$ is:

$$\mathbb{E}_{q_\epsilon(x_0)} \left[ \log P_0(x_0) \right] \geq \mathbb{E}_{q_\epsilon(x_0)} \left[ \log p_\epsilon(x_0 + u) \right] - \mathbb{E}_{q_\epsilon(x_0)} \left[ \log q(u|x_0) \right].$$

One widely-used choice for $q(u|x_0)$ is uniform distribution (uniform dequantization). However, this leads to a training-evaluation gap: for training, we fit $p_\epsilon$ to the distribution $q_\epsilon(x_0)$, which is a Gaussian distribution near each discrete data point $x_0$ because $x_0 = x_0 + \epsilon e$ for $\epsilon \sim \mathcal{N}(0, I)$; while for evaluation, we test $p_\epsilon$ on uniform dequantized data $x_0 + \epsilon$. Such a gap will also degrade the likelihood evaluation performance and is not well-studied.

In addition, another way for dequantization is to train a variational dequantization model $q_\theta(u|x_0)$ (Ho et al., 2019; Song et al., 2021b) but it will need additional costs and is hard to train (Kim et al., 2022).

#### 3.2. Training-Free Dequantization by Truncated Normal

In this section, we show that there exists a training-free dequantization distribution that fits diffusion ODEs well. As discussed in Sec. 3.1, the gap between training and testing of diffusion ODEs is due to the difference between the training input $x_0 = \sigma_0 x_0 + \epsilon$ (where $\epsilon \sim \mathcal{N}(0, I)$) and the testing input $x_0 + \epsilon$. To fill such a gap, we can choose
where the truncated-normal dequantization in Eqn. (10), then the discrete model distribution has the following importance weighted variational bound:

$$
\text{log} P_0(x_0) \geq E_{q(\varepsilon)} \left[ \log p_\theta(x) \right] + \frac{d}{2} \left( 1 + \log(2\pi\sigma_x^2) \right) + d \log \sigma_e
$$

where

$$
\hat{\varepsilon}_e = \alpha_e x_0 + \sigma_e \hat{\varepsilon}, \quad \hat{\varepsilon} \sim \mathcal{N}(\hat{\varepsilon} | 0, I, -\tau, \tau)
$$

and

$$
\varepsilon = \text{erf} \left( \frac{\tau}{\sqrt{2}} \right)
$$

Besides, we also have the following importance weighted likelihood estimator by using \( K \) i.i.d. samples by using Jensen’s inequality as in Burda et al. (2015). As \( K \) increases, the estimator gives a tighter bound, which enables more precise likelihood estimation.

**Corollary 3.2** (Importance Weighted Variational Bound under Truncated-Normal Dequantization). Suppose we use

\[
\begin{align*}
\mathcal{N}(x | \mu, \sigma^2 I, a, b)
\end{align*}
\]

for each of \( \alpha_e \approx 1 \), the discrete model distribution has the following importance weighted variational bound:

$$
\text{log} P_0(x_0) \geq E_{\mathcal{N}(\varepsilon)} \left[ \log p_\theta(x) \right] + \frac{d}{2} \left( 1 + \log(2\pi\sigma_x^2) \right) + d \log \sigma_e
$$

where

$$
\hat{\varepsilon}_e = \alpha_e x_0 + \sigma_e \varepsilon, \quad \varepsilon \sim \mathcal{N}(\varepsilon | 0, I, -\tau, \tau)
$$

and

$$
\varepsilon = \text{erf} \left( \frac{\tau}{\sqrt{2}} \right)
$$

Besides, we also have the following importance weighted likelihood estimator by using \( K \) i.i.d. samples by using Jensen’s inequality as in Burda et al. (2015). As \( K \) increases, the estimator gives a tighter bound, which enables more precise likelihood estimation.
which is the drift of probability flow ODE in Eqn. (2).

We give an intuitive explanation for $J_\text{FM}$ in Appendix D that the prediction target $\mathbf{v}$ is the tangent (velocity) of the diffusion path, and we name $\mathbf{v}_0$ as velocity parameterization. Besides, we show it empirically alleviates the imbalance problem in noise prediction.

In addition, we prove the equivalence between different predictors and different matching objectives for general noise schedules in Appendix B. We also show in Appendix E that the flow matching method Lipman et al. (2022); Albergo & Vanden-Eijnden (2022); Liu et al. (2022b) and related techniques for improving the sample quality of diffusion models in Karras et al. (2022); Salimans & Ho (2022); Ho et al. (2022) can all be reformulated in velocity parameterization. To be consistent, we still call $J_\text{FM}$ as flow matching. It’s an extended version of Lipman et al. (2022) with likelihood weighting and several practical modifications as detailed in Section 4.3.

4.2. Error-bounded Second-Order Flow Matching

According to Chen et al. (2018a), the ODE likelihood of Eqn. (11) can be evaluated by solving the following differential equation from $\epsilon$ to $T$:

$$\frac{d \log p_t(x_t)}{d \epsilon} = - \text{tr}(\nabla_x \mathbf{v}_0(x_t, \epsilon)).$$

As $J_\text{FM}$ in Eqn. (12) can only restrict the distance between $\mathbf{v}_0$ and $\mathbf{v}^*$, but not the divergence $\text{tr}(\nabla_x \mathbf{v}_0)$ and $\text{tr}(\nabla_x \mathbf{v}^*)$. The precision and smoothness of the trace $\text{tr}(\nabla_x \mathbf{v}_0(x_t, \epsilon))$ affects the likelihood performance and the number of function evaluations for sampling. For simulation-free training of $\text{tr}(\nabla_x \mathbf{v}(x_t, \epsilon))$, we propose an error-bounded trace of second-order flow matching, where the second-order error is bounded by the proposed objective and first-order error.

**Theorem 4.1.** (Error-Bounded Trace of Second-Order Flow Matching) Suppose we have a first-order velocity estimator $\hat{\mathbf{v}}_0(x_t, \epsilon)$, we can learn a second-order trace velocity model $v_2^{\text{trace}}(\cdot, \epsilon) : \mathbb{R}^d \rightarrow \mathbb{R}$ which minimizes

$$E_{q_t(x_t)} \left[ v_2^{\text{trace}}(x_t, \epsilon) - \text{tr}(\nabla_x v^*(x_t, \epsilon)) \right]^2,$$

by optimizing

$$\theta^* = \arg\min_{\theta} E_{\mathbf{v}_0, \epsilon} \left[ v_2^{\text{trace}}(x_t, \epsilon, \theta) - \text{tr}(\nabla_x v^*(x_t, \epsilon)) \right]^2.$$  

where

$$\ell_1(\epsilon, x_0, t) := \frac{2}{g^2(t)} \| \hat{\mathbf{v}}_0(x_t, \epsilon) - \mathbf{v} \|^2,$$

$$x_t = \alpha_t x_0 + \sigma_t \epsilon, \quad \mathbf{v} = \hat{\alpha}_t x_0 + \hat{\sigma}_t \epsilon, \quad \epsilon \sim \mathcal{N}(0, I).$$

Moreover, denote the first-order flow matching error as

$$\delta_1(x_t, \epsilon) := \| \hat{\mathbf{v}}_1(x_t, \epsilon) - \mathbf{v}^*(x_t, \epsilon) \|_2,$$

then $\forall x_t, \theta$, the estimation error for $v_2^{\text{trace}}(x_t, \epsilon; \theta)$ can be bounded by:

$$|v_2^{\text{trace}}(x_t, \epsilon; \theta) - \text{tr}(\nabla_x \mathbf{v}^*(x_t, \epsilon))| < |v_2^{\text{trace}}(x_t, \epsilon; \theta) - v_2^{\text{trace}}(x_t, \epsilon; \theta^*)| + \frac{2}{g^2(t)} \delta_1^2(x_t, \epsilon).$$

The proof is provided in Appendix F. In practice, we choose $v_2^{\text{trace}}(x_t, \epsilon; \theta) = \text{tr}(\nabla_x \mathbf{v}_0(x_t, \epsilon))$ for self-regularizing. As for scalability, we use Hutchinson’s trace estimator (Hutchinson, 1990) to unbiasedly estimate the trace, and use forward-mode automatic differentiation to compute Jacobian-vector product (Lu et al., 2022a).

4.3. Timing by Log-SNR and Normalizing Velocity

In practice, we make two modifications to improve the performance. First, we use negative log-SNR $\gamma_t$ to time the diffusion process. Still, we parameterize $\mathbf{v}_0(x, \gamma)$ to predict the drift of the $\gamma$ timed diffusion ODE i.e. $d\alpha_t/d\gamma = \mathbf{v}_0(x, \gamma)$, so the corresponding predictor $\mathbf{v}_0(x, \epsilon) = \mathbf{v}_0(x, \gamma) d\gamma/d\epsilon$. Second, the velocity of the diffusion path $\mathbf{v} = \hat{\alpha_t} x_0 + \hat{\sigma_t} \epsilon$ may have different scales at different $\epsilon$, so we propose to predict the normalized velocity $\tilde{\mathbf{v}} = \mathbf{v}/\sqrt{\hat{\alpha}_t^2 + \hat{\sigma}_t^2}$, with the parameterized network $\tilde{\mathbf{v}}_0(x_t, \epsilon) = \mathbf{v}_0(x_t, \epsilon)/\sqrt{\hat{\alpha}_t^2 + \hat{\sigma}_t^2}$, which is equal to $\tilde{\mathbf{v}}_0(x_t, \epsilon) = \mathbf{v}_0(x_t, \epsilon)/\sqrt{\hat{\alpha}_t^2 + \hat{\sigma}_t^2}$. The objective in Eqn. (12) reduces to

$$J_{\text{FM}}(\theta) = \int_{\gamma_0}^{\gamma_T} \frac{\hat{\alpha}_t^2 + \hat{\sigma}_t^2}{\sigma_\gamma^2} E_{x_0, \epsilon} \| \tilde{\mathbf{v}}_0(x_t, \epsilon, \gamma) - \tilde{\mathbf{v}}_0^* \|_2^2 d\gamma.$$

And the corresponding second-order objective:

$$J_{\text{FM}, \text{tt}} = \int_{\gamma_0}^{\gamma_T} \frac{\hat{\alpha}_t^2 + \hat{\sigma}_t^2}{\sigma_\gamma^2} E_{x_0, \epsilon} \left( \sigma_\gamma \text{tr}(\nabla \tilde{\mathbf{v}}_0) - \frac{\sigma_\gamma}{\sqrt{\hat{\alpha}_t^2 + \hat{\sigma}_t^2}} d\gamma \right. + \frac{\sigma_\gamma}{\hat{\alpha}_t^2 + \hat{\sigma}_t^2} \| \tilde{\mathbf{v}}_0^*(x_t, \epsilon, \gamma) - \tilde{\mathbf{v}}_0^* \|_2^2 d\gamma,$$

where $\tilde{\mathbf{v}}_0^*$ is the stop-gradient version of $\tilde{\mathbf{v}}_0$, since we only use the parameterized first-order velocity predictor as an estimator. Our final formulation of parameterized diffusion ODE is

$$\frac{d\alpha_t}{d\gamma} = \sqrt{\hat{\alpha}_t^2 + \hat{\sigma}_t^2} \tilde{\mathbf{v}}_0(x_t, \gamma).$$

4.4. Variance Reduction with Importance Sampling

The flow matching is conducted for all $\gamma$ in $[\gamma_0, \gamma_T]$ through an integral. In practice, the evaluation of the integral is time-consuming, and Monte-Carlo methods are used to unbiasedly estimate the objective by uniformly sampling $\gamma$. In
We propose to use two types of importance sampling (IS), \( \parallel \)

This way, the coefficients of \( ||\hat{v}_\theta(x, \gamma) - \hat{v}||^2_2 \), then

\[
\mathcal{J}_{FM}(\theta) = \mathbb{E}_{\gamma \sim p(\gamma)} \mathbb{E}_{x, \epsilon} \left[ \frac{\mathcal{L}_\theta(x_0, \epsilon, \gamma)}{p(\gamma)} \right]
\]

(18)

We propose to use two types of importance sampling (IS), and empirically compare them for faster convergence.

**Designed IS** Intuitively, we can choose \( p(\gamma) \propto \frac{\alpha^2 + \sigma^2}{(\sigma^2 + \gamma^2)} \).

This way, the coefficients of \( ||\hat{v}_\theta(x, \gamma) - \hat{v}||^2_2 \) is a time-invariant constant, and the velocity matching error is not amplified or shrinked at any \( \gamma \). This is similar to the IS in Song et al. (2021b), where the \( g(t)/\sigma^2 \) weighting before the noise matching error \( ||\epsilon_{\theta}(x, t) - \epsilon||^2_2 \) is cancelled, and it corresponds to uniform \( \gamma \) under our parameterization.

For noise schedules used in this paper, we can obtain closed-form sampling procedures using inverse transform sampling, see Appendix C.

**Learned IS** The variance of the Monte-Carlo estimator depends on the learned network \( \hat{v}_\tilde{\theta} \). To minimize the variance, we can parameterize the IS with another network and treat the variance as an objective. Actually, learning \( p(\gamma) \) is equivalent to learning a monotone mapping \( \gamma(t) : [0, 1] \rightarrow \{\gamma_0, \gamma_T\} \), which is inverse cumulative distribution function of \( p(\gamma) \). We can uniformly sample \( t \), and regard the IS as change-of-variable from \( \gamma \) to \( t \).

\[
\mathcal{J}_{FM}(\theta) = \mathbb{E}_{t \sim U[0, 1]} \mathbb{E}_{x_0, \epsilon} \left[ \gamma'(t) \mathcal{L}_\theta(x_0, \epsilon, \gamma(t)) \right]
\]

(19)

Suppose we parameterize \( \gamma(t) \) with \( \eta \). Denote \( \mathcal{L}_{\hat{v}_\theta}(x_0, \epsilon, t) = \gamma'(t)\mathcal{L}_\theta(x_0, \epsilon, \gamma(t)) \), which is a Monte-Carlo estimator of \( \mathcal{J}_{FM}(\theta) \). Since its variance \( \text{Var}_{\epsilon, x_0} \left[ L_{\hat{v}_\theta}(x_0, \epsilon, t) \right] = \mathbb{E}_{\epsilon, x_0} \left[ L_{\hat{v}_\theta}(x_0, \epsilon, t) \right] - \mathcal{J}_{FM}(\theta) \) and \( \mathcal{J}_{FM}(\theta) \) is invariant to \( \gamma(0) \), we can minimize \( \mathbb{E}_{\epsilon, x_0} \left[ L_{\hat{v}_\theta}(x_0, \epsilon, t) \right] \) for variance reduction.

While this approach seeks the optimal IS, they cause extra overhead by introducing an IS network, requiring complex gradient operation or additional training steps. Thus, we only use it as a reference to test the optimality of our designed IS. We simplify the variance reduction in Kingma et al. (2021), and propose an adaptive IS algorithm, which is detailed in Appendix H. Empirically, we show that designed IS is a more preferred approach since it is training-free and achieves a similar convergence speed to learned IS.

5. Related Work

Diffusion models, also known as score-based generative models (SGMs), have achieved state-of-the-art sample quality and likelihood (Dhariwal & Nichol, 2021; Karras et al., 2022; Kingma et al., 2021) among deep generative models, yielding extensive downstream applications such as speech and singing synthesis (Chen et al., 2021; Liu et al., 2022a), conditional image generation (Ramesh et al., 2022; Rombach et al., 2022), guided image editing (Meng et al., 2022; Nichol et al., 2022), unpaired image-to-image translation (Zhao et al., 2022) and inverse problem solving (Chung et al., 2022; Kawar et al., 2022).

Diffusion ODEs are special formulations of neural ODEs and can be viewed as continuous normalizing flows (Chen et al., 2018a). Training of diffusion ODEs can be categorized into simulation-based and simulation-free methods. The former utilizes the exact likelihood evaluation formula of ODE (Chen et al., 2018a), which leads to a maximum likelihood training procedure (Grathwohl et al., 2019). However, it involves expensive ODE simulations for forward and backward propagation and may result in unnecessary complex dynamics (Finlay et al., 2020) since it only cares about the model distribution at \( t = 0 \). The latter trains neural ODEs by matching their trajectories to a predefined path, such as the diffusion process. This approach is proposed in Song et al. (2021c), and extended in Lu et al. (2022a); Lipman et al. (2022); Albergo & Vanden-Eijnden (2022); Liu et al. (2022b). We propose velocity parameterization which is an extension of Lipman et al. (2022) with practical modifications and claim that the paths used in Lipman et al. (2022); Albergo & Vanden-Eijnden (2022); Liu et al. (2022b) are special cases of noise schedule. Aiming at maximum likelihood training, we also get inspiration from Lu et al. (2022a). We additionally apply likelihood weighting and propose to finetune the model with high-order flow matching.

Variance reduction techniques are commonly used for training diffusion models. Nichol & Dhariwal (2021) proposes an importance sampling (IS) for discrete-time diffusion models by maintaining the historical losses at each time step and building the proposal distribution based on them. Song et al. (2021b) designs an IS to cancel out the weighting before the noise matching loss. Kingma et al. (2021) proposes a variance reduction method that is equivalent to learning a parameterized IS. We simply their procedure and propose an adaptive IS scheme for ablation. By empirically comparing different IS methods, we find a designed and analytical IS distribution that achieves a good performance-efficiency trade-off.

6. Experiments

In this section, we present our training procedure and experiment settings, and our ablation studies to demonstrate how our techniques improve the likelihood of diffusion ODEs. We implement our methods based on the open-source code-
base of Kingma et al. (2021) implemented with JAX Bradbury et al. (2018), and use similar network and hyperparameter settings. We first train the model by optimizing our first-order flow matching objective \( \min_\theta J_{FM}(\theta) \) for enough iterations, so that the first-order velocity prediction has little error. Then, we finetune the pretrained first-order model using a mixture of first-order and second-order flow matching objectives \( \min_\theta J_{FM}(\theta) + \lambda J_{FM,sc}(\theta) \). The finetune process converges in much fewer iterations than pretraining. Finally, we evaluate the likelihood on the test set using the variational bound under our proposed truncated-normal dequantization. The detailed training configurations are provided in Appendix I.

Our training and evaluation procedure is feasible for any noise schedule \( \alpha, \sigma \). We choose two special noise schedules:

**Variance Preserving (VP)** \( \alpha^2 + \sigma^2 = 1 \). This schedule is widely used in diffusion models, which yields a process with a fixed variance of one when the initial distribution has unit variance.

**Straight Path (SP)** \( \alpha = \sigma = 1 \). This schedule is used in Lipman et al. (2022); Albergo & Vanden-Eijnden (2022); Liu et al. (2022b), where they call it OT path and claim it leads to better dynamics since the pairwise diffusion paths are straight lines. We simply regard it as a special kind of noise schedule.

Under these two schedules, \( \alpha, \sigma \) are uniquely determined by \( \gamma \), and we do not have any extra hyperparameters. They also have corresponding objectives and designed IS, which can be expressed in closed form (see Appendix C for details). We train our 1-DODE on CIFAR-10 (Krizhevsky et al., 2009) and ImageNet-32\(^1\) (Deng et al., 2009), which are two popular benchmarks for generative modeling and density estimation.

### 6.1. Likelihood and Samples

Table 1 shows our experiment results on CIFAR-10 and ImageNet-32 datasets. Our models are pretrained with velocity parameterization, designed IS, and finetuned with second-order flow matching. We report the likelihood values using our truncated-normal dequantization with the importance weighted estimator under \( K = 20 \). To compute the FID values, we apply an adaptive-step ODE solver to draw samples from the diffusion ODEs. We also report the NFE during the sampling process, which reflects the smoothness of the dynamics.

Combining our training techniques and dequantization, we exceed the likelihood of previous ODEs, especially by a large margin on CIFAR-10. In Figure 1, we compare our pretraining phase to VDM (Kingma et al., 2021), which indicates that our techniques achieve 2x–3x times of previous convergence speed. We do not observe the superiority of SP to VP such as lower FID and NFE as in Lipman et al. (2022). We suspect it may result from maximum likelihood training, which put more emphasis on the high log-SNR region. More theoretical comparisons with Lipman et al. (2022) are given in Appendix E.2.

Randomly generated samples from our models are provided in Appendix J. Since we use network architecture and techniques targeted at the likelihood, our FID is worse than the state-of-the-art, which can be improved by designing time weighting to emphasize the training at small log-SNR levels (Kingma et al., 2021) or using high-quality sampling algorithms such as PC sampler (Song et al., 2021c).

### 6.2. Ablations

Due to the expensive time cost of pretraining, we only conduct ablation studies on CIFAR-10 under the VP schedule. First, we test our techniques for pretraining when training from scratch. We plot the training curves with noise predictor (Kingma et al., 2021) and velocity predictor, then further implement our IS strategies (Figure 2). We find that velocity parameterization and IS both accelerate the training process, while designed IS performs slightly worse than adaptive IS. Considering the extra time cost for learning the IS network, we conclude that designed IS is a better choice for large-scale pretraining. Then we visualize different IS by plotting the mapping from uniform \( t \) to importance sampled

---

\(^1\)There are two different versions of ImageNet32 and ImageNet64 datasets. For fair comparisons, we use both versions of ImageNet32, one is downloaded from https://image-net.org/data/donsample/imagenet32_train.zip, following Lipman et al. (2022), and the other is downloaded from http://image-net.org/small/train_32x32.tar (old version, no longer available), following Song et al. (2021b) and Kingma et al. (2021). The former dataset applies anti-aliasing and is easier for maximum likelihood training.
Table 1. Negative log-likelihood (NLL) in bits/dim (BPD), sample quality (FID scores) and number of function evaluations (NFE) on CIFAR-10 and ImageNet-32x32. For fair comparisons, we list NLL results of previous ODEs without variational dequantization or data augmentation, and FID/NFE results obtained by adaptive-step ODE solver. Results with “/” means they are not reported in the original papers. * For VDM, since they have no ODE formulation, the FID score is obtained by 1000 step discretization of their SDE. We report their corresponding ODE result in the ablation study. * Corresponding to the old version ImageNet-32 dataset.

<table>
<thead>
<tr>
<th>Model</th>
<th>CIFAR-10</th>
<th>ImageNet-32</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NLL ↓</td>
<td>FID ↓</td>
</tr>
<tr>
<td>VDM (Kingma et al., 2021)</td>
<td>2.65</td>
<td>7.60*</td>
</tr>
<tr>
<td>(Previous ODE)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FFJORD (Grathwohl et al., 2019)</td>
<td>3.40</td>
<td>/</td>
</tr>
<tr>
<td>ScoreSDE (Song et al., 2021c)</td>
<td>2.99</td>
<td>2.92</td>
</tr>
<tr>
<td>ScoreFlow (Song et al., 2021b)</td>
<td>2.90</td>
<td>5.40</td>
</tr>
<tr>
<td>Soft Truncation (Kim et al., 2022)</td>
<td>3.01</td>
<td>3.96</td>
</tr>
<tr>
<td>Flow Matching (Lipman et al., 2022)</td>
<td>2.99</td>
<td>6.35</td>
</tr>
<tr>
<td>Stochastic Interp. (Albergo &amp; Vanden-Eijnden, 2022)</td>
<td>2.99</td>
<td>10.27</td>
</tr>
<tr>
<td>i-DODE (SP) (ours)</td>
<td>2.56</td>
<td>11.20</td>
</tr>
<tr>
<td>i-DODE (VP) (ours)</td>
<td>2.57</td>
<td>10.74</td>
</tr>
</tbody>
</table>

Figure 2. Training curve from scratch for ablation. We compute the loss on the training set by the SDE likelihood bound in Kingma et al. (2021).

γ, as well as the variance of noise levels on the pretrained model (Figure 3). We show that the IS reduces the variance by sampling more in high log-SNR regions.

Table 2. Ablation study when converged. We report negative log-likelihood (NLL) in bits/dim (BPD), sample quality (FID scores), and number of function evaluations (NFE) after our pretraining and finetuning phase. We evaluate NLL by uniform (U) and truncated-normal (TN) dequantization without importance weight. We retrain VDM and evaluate its ODE form.

<table>
<thead>
<tr>
<th>Model</th>
<th>NLL (U)</th>
<th>NLL (TN)</th>
<th>FID</th>
<th>NFE</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDM (Kingma et al., 2021)</td>
<td>2.78</td>
<td>2.64</td>
<td>8.65</td>
<td>213</td>
</tr>
<tr>
<td>Pretrain (ours)</td>
<td>2.75</td>
<td>2.61</td>
<td>10.66</td>
<td>248</td>
</tr>
<tr>
<td>+ Finetune (ours)</td>
<td>2.74</td>
<td>2.60</td>
<td>10.74</td>
<td>126</td>
</tr>
</tbody>
</table>

Next, we test our pretraining, finetuning and evaluation on the converged model (Table 2). As stated before, our pretraining has faster loss descent and converges to a higher likelihood than VDM. Based on it, our finetuning slightly improves the ODE likelihood and smooths the flow, leading to much less NFE when sampling. Our truncated-normal dequantization is also a key factor for precise likelihood computing, which surpasses previous uniform dequantization by a large margin.

In agreement with Song et al. (2021b), our improvements in likelihood lead to slightly worse FIDs. We also argue that the degeneration is small in terms of visual quality. We provide additional samples in the Appendix J for comparison.

7. Conclusion

We propose improved techniques for simulation-free maximum likelihood training and likelihood evaluation of diffusion ODEs. Our training stage involves improved pretraining and additional finetuning, which results in fast convergence, high likelihood and smooth trajectory. We improve the likelihood evaluation with novel truncated-normal dequantization, which is training-free and tailor-made for diffusion ODEs. Empirically, we achieve state-of-the-art likelihood on image datasets without variational dequantification.
zation or data augmentation and make a breakthrough on CIFAR-10 compared to previous ODEs. Due to resource limitations, we didn’t explore tuning of hyperparameters and network architectures, which are left for future work.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (2020AAA0106302); NSF of China Projects (Nos. 62061136001, 61620106010, 62076145, U19B2034, U1811461, U19A2081, 6197222, 62106120, 62076145); a grant from Tsinghua Institute for Guo Qiang; the High Performance Computing Center, Tsinghua University. J.Z. was also supported by the New Cornerstone Science Foundation through the XPLORER PRIZE.

References


Ho, J., Chan, W., Saharia, C., Whang, J., Gao, R., Gritsenko, A., Kingma, D. P., Poole, B., Norouzi, M., Fleet, D. J.,


Improved Techniques for Maximum Likelihood Estimation for Diffusion ODEs


A. Different perspective of diffusion ODEs for bridging the gap between discrete and continuous data

Suppose the discrete data $X_0$ to be modelled are 8-bit integers $\{0, 1, \ldots, 255\}$. Following the common transform in diffusion models, we first normalize it to range $[-1,1]$ by the mapping $x_0 = \frac{X_0 + 1 - 128}{128}$. In the following discussions, we consider the model distribution $P_0(x_0)$ on transformed discrete data $x_0$, which is equal to $P_0(X_0)$ since the scaling does not alter the discrete probability.

A.1. Dequantization perspective

The discrete data $x_0$ has a uniform gap $\frac{1}{128}$ between two consecutive values on each dimension. We can define the discrete model distribution as

$$P_0(x_0) = \int_{u \in [-\frac{1}{256}, \frac{1}{256}]} p_e(x + u) du$$

(20)

where $p_e$ is the diffusion ODE defined at time $\epsilon$. Then, we can introduce a dequantization distribution $q(u|x_0)$ with support over $[-\frac{1}{256}, \frac{1}{256}]^d$. Treating $q$ as an approximate posterior, we obtain the following variational bound (Ho et al., 2019):

$$\log P_0(x_0) \geq \mathbb{E}_{q(u|x_0)}[\log p_e(x_0 + u) - \log q(u|x_0)]$$

(21)

The ODE term $\log p_e(x_0 + u)$ can be evaluated exactly by solving another ODE called “Instantaneous Change of Variables” (Chen et al., 2018a). As for the posterior $\log q(u|x_0)$, we can derive closed-form solutions for predefined posterior formulation. We provide the details for uniform dequantization and our proposed truncated-normal dequantization.

Uniform dequantization. We simply use uniform posterior $q(u|x_0) = \mathcal{U}(-\frac{1}{256}, \frac{1}{256})$. In this case, $\log q(u|x_0) = d \log 128$ is a constant, and the bound becomes

$$\log P_0(x_0) \geq \mathbb{E}_{u \sim \mathcal{U}(-\frac{1}{256}, \frac{1}{256})}[\log p_e(x_0 + u)] - d \log 128$$

(22)

Similar to Burda et al. (2015), we can also sample multiple $u$ to derive a tighter bound, which is called importance weighted estimator:

$$\log P_0(x_0) \geq \mathbb{E}_{u^{(1)}, \ldots, u^{(K)} \sim \mathcal{U}(-\frac{1}{256}, \frac{1}{256})} \left[ \log \left( \frac{1}{K} \sum_{i=1}^{K} p_e(x_0 + u^{(i)}) \right) \right] - d \log 128$$

(23)

However, this dequantization will cause a training-evaluation gap. For training, we fit $p_e$ to the distribution of $\hat{x}_e = \alpha_e x_0 + \sigma_e \epsilon, \epsilon \sim \mathcal{N}(0, I)$. For evaluation, we test $p_e$ on uniform dequantized $x_0 + u, u \sim \mathcal{U}(-\frac{1}{256}, \frac{1}{256})$. This gap will degenerate the likelihood performance, as we will show later.

Truncated-normal dequantization. To bridge the training-evaluation gap, we test $p_e$ on $\hat{x}_e = \alpha_e x_0 + \sigma_e \epsilon$, where $\epsilon$ obeys a truncated-normal distribution to make sure the range of $u$ on each dimension does not exceed $[-\frac{1}{256}, \frac{1}{256}]$. Specifically, denote $\tau := \frac{\sigma_e}{\alpha_e \epsilon}$, we define the truncated-normal distribution as

$$\epsilon \sim \mathcal{T}N(\epsilon|0, I, -\tau, \tau)$$

(24)

Let

$$u := \frac{\sigma_e}{\alpha_e \epsilon} \epsilon \in \left[ -\frac{1}{256}, \frac{1}{256} \right]$$

(25)

By the change of variables for probability density, we have

$$\log p_e(x_0 + u) = \log p_e \left( x_0 + \frac{\sigma_e}{\alpha_e} \epsilon \right) = \log p_e(\hat{x}_e) + d \log \alpha_e$$

(26)

$$\log q(u|x_0) = \log q \left( \frac{\sigma_e}{\alpha_e} \epsilon \right) = \log q(\hat{\epsilon}) + d \log \frac{\alpha_e}{\sigma_e}$$

(27)
where \( q(\hat{e}) \) is the probability distribution function of truncated-normal distributions

\[
q(\hat{e}) = \frac{1}{(2\pi Z^2)^{\frac{d}{2}}} \exp\left(-\frac{1}{2} ||\hat{e}||_2^2\right), \quad Z := \Phi(\tau) - \Phi(-\tau) = \text{erf}\left(\frac{\tau}{\sqrt{2}}\right)
\]  

(28)

Here \( \Phi(\cdot) \) is the cumulative distribution function of standard normal distribution, and \( \text{erf}(\cdot) \) is the error function. Combining the equations above, the bound is reduced to

\[
\log P_0(x_0) \geq \mathbb{E}_{q(\hat{e})} [\log p_\epsilon(\hat{x}_\epsilon) - \log q(\hat{e})] + d \log \sigma_\epsilon
\]

(29)

Further, we can derive closed-form solutions for the entropy term of truncated-normal distribution:

\[
-\mathbb{E}_{q(\hat{e})}[\log q(\hat{e})] = H(q(\hat{e})) = d \log(\sqrt{2\pi}\epsilon) + d \log Z - d - \frac{\tau}{\sqrt{2\pi}Z} \exp\left(-\frac{1}{2} \tau^2\right)
\]

(30)

and we finally obtain the exact form of the bound:

\[
\log P_0(x_0) \geq \mathbb{E}_{\hat{e}_0,\ldots,\hat{e}_K \sim T_N(0, I, -\tau, \tau)} [\log \left(\frac{1}{K} \sum_{i=1}^{K} \frac{p_\epsilon(\hat{x}_\epsilon^{(i)})}{q(\hat{e}^{(i)})}\right)] + d \log \sigma_\epsilon
\]

(32)

where \( \hat{x}_\epsilon^{(i)} := \alpha_\epsilon x_\epsilon + \sigma_\epsilon \hat{e}^{(i)} \), and \( q(\hat{e}) \) is expressed in Eqn. (28).

In our experiments, we choose the start time \( \gamma_\epsilon = -13.3 \). Under this setting, we have \( \tau \approx 3 \), and the truncated-normal distribution \( T_N(0, I, -\tau, \tau) \) is almost the same as the standard normal distribution \( N(0, I) \) due to the 3-\sigma principle. Thus, \( x_\epsilon \) used in training and \( \hat{x}_\epsilon \) used in testing are virtually identically distributed, resulting in a negligible training-evaluation gap.

### A.2. Variational perspective

From the variational perspective, we can view the transition from discrete \( x_0 \) to continuous \( x_\epsilon \) as a variational autoencoder, where the prior \( p_\epsilon(x_\epsilon) \) is modeled by diffusion ODE, and the approximate posterior \( q_0(\epsilon, x_\epsilon | x_0) \) is the analytical Gaussian transition kernel in the forward diffusion process at the start. We have the variational bound:

\[
\log P_0(x_0) \geq \mathbb{E}_{q_0(x_\epsilon | x_0)} [\log p_\epsilon(x_\epsilon | x_\epsilon) + \log p_\epsilon(x_\epsilon) - \log q_0(x\epsilon | x_0)]
\]

(33)

where \( q_0(x_\epsilon | x_0) = N(x_\epsilon | \alpha_\epsilon, \sigma_\epsilon^2 I) \). We want to use the reconstruction term \( p_\epsilon(x_\epsilon | x_\epsilon) \) to approximate \( q_0(x\epsilon | x_0) \). Note that

\[
q_\epsilon(x_\epsilon | x_0) = \frac{q_0(x_\epsilon | x_0) q_\epsilon(x_\epsilon)}{q_\epsilon(x_\epsilon)}
\]

(34)

for small enough \( \epsilon \), we have \( q_\epsilon(x_\epsilon) \approx q_\epsilon(x_\epsilon) \), so \( q_\epsilon(x_\epsilon | x_0) \propto q_\epsilon(x_\epsilon | x_0) = \prod_i q_\epsilon(x_{\epsilon,i} | x_{0,i}) \), where \( i \) represents the \( i \)-th dimension. Thus, we also choose \( p_\epsilon(x_\epsilon | x_\epsilon) \) as a factorized distribution, following Kingma et al. (2021):

\[
p_\epsilon(x_\epsilon | x_\epsilon) = \prod_i p_\epsilon(x_{\epsilon,i} | x_{0,i})
\]

(35)

where each

\[
p_\epsilon(x_{\epsilon,i} | x_{0,i}) \propto q_\epsilon(x_{\epsilon,i} | x_{0,i}) \propto \exp\left(-\frac{(x_{\epsilon,i} - \alpha_\epsilon x_{0,i})^2}{2\sigma_\epsilon^2}\right)
\]

(36)

As \( x_0 \) is a discrete variable, the probability can be computed by softmax, so we have

\[
\log p_\epsilon(x_\epsilon | x_\epsilon) = \sum_{i=1}^{d} \log \text{softmax}_{j=0}^{255} \left(-\frac{(x_{\epsilon,i} - \alpha_\epsilon x_{0,i})^2}{2\sigma_\epsilon^2}\right) [x_{0,i}]
\]

(37)
We test the two types of dequantization and the variational perspective on our final models, using different numbers of

\[ \hat{\alpha}_x \]

We note the only difference is that our proposed truncated-normal dequantization uses

\[ q_{x|0}(x_0|x_0) \]

where \( q \) is the data distribution. Since BPD averages the log-likelihood on each dimension, scaling dimensionality has no effect on the final result.

**Remark A.1.** For high-dimensional data such as images, directly comparing log-likelihood may suffer from scaling issues by the dimension. In practice, we usually compare the BPD (bits/dim) by

\[ \text{BPD} = \mathbb{E}_{x_0 \sim q_0} \left[ -\frac{\log P_0(x_0)}{d \log 2} \right] \]

where \( q_0 \) is the data distribution. Since BPD averages the log-likelihood on each dimension, scaling dimensionality has no effect on the final result.

We test the two types of dequantization and the variational perspective on our final models, using different numbers of importance samples \( K \). The results are listed in Table 3. Empirically, truncated-normal dequantization performs slightly better than variational, while uniform dequantization gives a bad likelihood due to the large training-evaluation gap. We also observe that increasing \( K \) further improves the results by giving a tighter bound.

**Remark A.2.** Since uniform dequantized data has a larger noise level than truncated-normal dequantized data, we find evaluating \( \log p_x(x_0 + u) \) at start time \( \gamma = -13.3 \) leads to bad likelihood. Thus, we tune \( \gamma \) for uniform dequantization (Figure 4), and eventually choose \( \gamma = -12.0, -11.9, -11.7, -11.6 \) for CIFAR-10 (VP), CIFAR-10 (SP), ImageNet-32 (VP), ImageNet-32 (SP) respectively.

### A.3. Practical connections and results

Let us consider the bound without importance weighted estimator. By observing the bound in Eqn. (31) for truncated-normal dequantization and the bound in Eqn. (39) for variational perspective, we can find that they have similar formulations. Suppose we use \( \gamma = -13.3 \), we have \( \tau \approx 3.01809, Z \approx 0.9974613 \), and the bound in Eqn. (31) is approximately

\[ \log P_0(x_0) \geq \mathbb{E}_{e \sim N(0, I)} [\log p_e(x_e)] + \frac{d}{2} (1 + \log(2\pi \sigma_e^2)) - 0.01522 \times d \]

Next, consider the variational perspective. Though the reconstruction term \( \log p_{0\|}(x_0|x_e) \) in Eqn. (39) depends on the data distribution, empirically it is nearly a constant \( \log p_{0\|}(x_0|x_e) \approx -0.01 \times d \). So we have the approximate bound

\[ \log P_0(x_0) \geq \mathbb{E}_{e \sim N(0, I)} [\log p_e(x_e)] + \frac{d}{2} (1 + \log(2\pi \sigma_e^2)) - 0.01 \times d \]

We note the only difference is that our proposed truncated-normal dequantization uses \( \hat{x}_e \) rather than \( x_e \) for ODE likelihood evaluation, and there is a small constant difference in the bound.

**Remark A.1.** For high-dimensional data such as images, directly comparing log-likelihood may suffer from scaling issues by the dimension. In practice, we usually compare the BPD (bits/dim) by

\[ \text{BPD} = \mathbb{E}_{x_0 \sim q_0} \left[ -\frac{\log P_0(x_0)}{d \log 2} \right] \]

where \( q_0 \) is the data distribution. Since BPD averages the log-likelihood on each dimension, scaling dimensionality has no effect on the final result.

We test the two types of dequantization and the variational perspective on our final models, using different numbers of importance samples \( K \). The results are listed in Table 3. Empirically, truncated-normal dequantization performs slightly better than variational, while uniform dequantization gives a bad likelihood due to the large training-evaluation gap. We also observe that increasing \( K \) further improves the results by giving a tighter bound.

**Remark A.2.** Since uniform dequantized data has a larger noise level than truncated-normal dequantized data, we find evaluating \( \log p_x(x_0 + u) \) at start time \( \gamma = -13.3 \) leads to bad likelihood. Thus, we tune \( \gamma \) for uniform dequantization (Figure 4), and eventually choose \( \gamma = -12.0, -11.9, -11.7, -11.6 \) for CIFAR-10 (VP), CIFAR-10 (SP), ImageNet-32 (VP), ImageNet-32 (SP) respectively.
Figure 4. The likelihood evaluation results under uniform dequantization for different start times $\gamma_t$. To plot the curve, we estimate the likelihood using the first 1024 test samples for CIFAR-10, and the first 512 test samples for ImageNet-32.

**B. Equivalence of different predictors and matching objectives**

We have the following theorem which demonstrates that different predictors are mutually transformable by a time-dependent skip connection, and they can be trained in a simulation-free approach by equivalent matching objectives.

**Theorem B.1.** Let $x_0$ be the sample from data distribution, and $\epsilon$ be the sample from $\mathcal{N}(0, I)$. Denote $x_t = \alpha_t x_0 + \sigma_t \epsilon$, $v = \alpha_t x_0 + \sigma_t \epsilon$. Suppose we have four kinds of predictors parameterized by $\theta$ and corresponding matching objectives with positive time weighting function $w(t)$:

- **score predictor** $s_\theta(x_t, t)$ and score matching loss $J_{SM}(\theta, w(t)) = \mathbb{E}_t \left[ w(t) \mathbb{E}_{x_0, \epsilon} \left[ ||s_\theta(x_t, t) - \nabla_x \log q_t(x_t)||^2 \right] \right]$
- **noise predictor** $e_\theta(x_t, t)$ and noise matching loss $J_{NM}(\theta, w(t)) = \mathbb{E}_t \left[ w(t) \mathbb{E}_{x_0, \epsilon} \left[ ||e_\theta(x_t, t) - \epsilon||^2 \right] \right]$
- **data predictor** $x_\theta(x_t, t)$ and data matching loss $J_{DM}(\theta, w(t)) = \mathbb{E}_t \left[ w(t) \mathbb{E}_{x_0, \epsilon} \left[ ||x_\theta(x_t, t) - x_0||^2 \right] \right]$
- **velocity predictor** $v_\theta(x_t, t)$ and flow matching loss $J_{FM}(\theta, w(t)) = \mathbb{E}_t \left[ w(t) \mathbb{E}_{x_0, \epsilon} \left[ ||v_\theta(x_t, t) - v||^2 \right] \right]$

For any $w(t)$, if we denote the optimal (ground-truth) predictors that minimize the corresponding matching losses as $s^*(x_t, t), e^*(x_t, t), x^*(x_t, t), v^*(x_t, t)$ respectively, then they are equivalent by the following relations:

$$
\begin{align*}
    e^*(x_t, t) &= -\sigma_t s^*(x_t, t) \\
    x^*(x_t, t) &= \frac{1}{\alpha_t} x_t + \frac{\sigma_t^2}{\alpha_t} s^*(x_t, t) \\
    v^*(x_t, t) &= f(t)x_t - \frac{1}{2} g^2(t)s^*(x_t, t)
\end{align*}
$$

where $s^*(x_t, t) = \nabla_x \log q_t(x_t)$ is the ground-truth score.

**Proof.** For any positive weighting $w(t)$, the overall optimum of the matching loss $\mathbb{E}_t \left[ w(t) \mathbb{E}_{x_0, \epsilon} \left[ || \cdot ||^2 \right] \right]$ is achieved when the optimum of the inner expectation $\mathbb{E}_{x_0, \epsilon} \left[ || \cdot ||^2 \right]$ is achieved for any $t$. For fixed $t$, by denoising score matching (Vincent, 2011), we know minimizing $\mathbb{E}_{x_0, \epsilon} \left[ ||s_\theta(x_t, t) - \nabla_x \log q_t(x_t)||^2 \right]$ is equivalent to minimizing $\mathbb{E}_{x_0, \epsilon} \left[ ||s_\theta(x_t, t) - \nabla_x \log q_{00}(x_t|x_0)||^2 \right] = \mathbb{E}_{q(x_t)} \mathbb{E}_{q_{00}(x_0|x_t)} ||s_\theta(x_t, t) - \nabla_x \log q_{00}(x_t|x_0)||^2$, where $\log q_{00}(x_t|x_0) = -\frac{\epsilon}{\sigma_t}$. The inner expectation is a minimum mean square error problem, so the optimal score predictor satisfies

$$
    s^*(x_t, t) = \mathbb{E}_{q_{00}(x_0|x_t)} [\nabla_x \log q_{00}(x_t|x_0)] = -\frac{1}{\sigma_t} \mathbb{E}_{q_{00}(x_0|x_t)} [\epsilon]  
$$

Similarly, for $J_{SM}(\theta, w(t))$, the optimal noise predictor satisfies

$$
    e^*(x_t, t) = \mathbb{E}_{q_{00}(x_0|x_t)} [\epsilon] = -\sigma_t s^*(x_t, t)  
$$
For $J_{\text{DM}}(\theta, w(t))$, the optimal data predictor satisfies

\[
x^*(x_t, t) = \mathbb{E}_{q_0(x_0|x_t)}[x_0] = \mathbb{E}_{q_0(x_0|x_t)}[x_t - \sigma_t \epsilon]\frac{x_t - \sigma_t \mathbb{E}_{q_0(x_0|x_t)}[\epsilon]}{\alpha_t} = \frac{1}{\alpha_t} x_t - \frac{\sigma_t^2}{\alpha_t} \mathbb{E}_{q_0(x_0|x_t)}[\epsilon] = \frac{1}{\alpha_t} x_t + \frac{\sigma_t^2}{\alpha_t} s^*(x_t, t)
\]

(47)

For $J_{\text{FM}}(\theta, w(t))$, the optimal velocity predictor satisfies

\[
v^*(x_t, t) = \mathbb{E}_{q_0(x_0|x_t)}[\dot{x}_0 + \dot{\sigma}_t \epsilon] = \dot{\alpha}_t \mathbb{E}_{q_0(x_0|x_t)}[x_0] + \dot{\sigma}_t \mathbb{E}_{q_0(x_0|x_t)}[\epsilon] = \frac{\dot{\alpha}_t}{\alpha_t} x_t + \frac{\dot{\alpha}_t \sigma_t^2 - \sigma_t \dot{\sigma}_t}{\alpha_t} s^*(x_t, t) = f(t)x_t - \frac{1}{2} g^2(t) s^*(x_t, t)
\]

(48)

The equivalence of optimal predictors also implies the equivalence of parameterized predictors. From the above theorem, we know $v_0(x_t, t)$ and $\epsilon_0(x_t, t)$ are related by $v_0(x_t, t) = f(t)x_t + \frac{\sigma_t^2}{2\alpha_t} \epsilon_0(x_t, t)$. In practice, we use $\gamma$ timing. From the relationship $v_\theta(x_t, t) = v_\theta(x_{\gamma}, \gamma) \frac{d\gamma}{dt}$, $\epsilon_\theta(x_t, t) = \epsilon_\theta(x_{\gamma}, \gamma)$, we obtain the noise predictor expressed by $v_\theta(x_{\gamma}, \gamma)$

\[
\epsilon_\theta(x_{\gamma}, \gamma) = 2 \frac{v_\theta(x_{\gamma}, \gamma) - \frac{\dot{\alpha}_t}{\alpha_t} x_{\gamma}}{\sigma_{\gamma}}
\]

(49)

Further, we can replace $v_\theta(x_{\gamma}, \gamma)$ with the normalized velocity predictor $v_\theta(x_{\gamma}, \gamma) = v_\theta(x_{\gamma}, \gamma)/\sqrt{\dot{\alpha}_t^2 + \dot{\sigma}_t^2}$.

Moreover, we can derive the equivalent training objectives under different parameterizations by employing the relations discussed above freely. For example, when we replace the normalized velocity predictor $v_\theta$ with the score predictor $s_\theta$ in the second-order objective Eqn. (16), we can obtain the second-order denoising score matching similar to Lu et al. (2022a). However, though theoretically equivalent, the actual performance of these objectives highly depends on the specific model architecture, hyperparameters and parameterization, and the authors of Lu et al. (2022a) find that their high-order denoising score matching objectives only work for VE schedule, but degenerate the performance of pretrained models with VP schedule.

C. Specifications under VP and SP schedule

As stated in Section 4.3, using $\gamma$ timing and normalized velocity predictor $v_\theta$, the likelihood weighted first-order and second-order flow matching objectives are reformulated as:

\[
J_{\text{FM}} = \int_{\gamma_0}^{\gamma_T} 2 \frac{\dot{\alpha}_t^2 + \dot{\sigma}_t^2}{\sigma_{\gamma}^2} \mathbb{E}_{x_0, \epsilon} \| \tilde{v}_\theta(x_{\gamma}, \gamma) - \tilde{v} \|^2 d\gamma
\]

\[
J_{\text{FM, tr}} = \int_{\gamma_0}^{\gamma_T} 2 \frac{\dot{\alpha}_t^2 + \dot{\sigma}_t^2}{\sigma_{\gamma}^2} \mathbb{E}_{x_0, \epsilon} \left( \sigma_{\gamma} \text{tr}(\nabla \tilde{v}_\theta) - \frac{\dot{\sigma}_t}{\sqrt{\dot{\alpha}_t^2 + \dot{\sigma}_t^2}} d + \frac{2 \sqrt{\dot{\alpha}_t^2 + \dot{\sigma}_t^2}}{\sigma_{\gamma}} \| \tilde{v}_\theta(x_{\gamma}, \gamma) - \tilde{v} \|^2 \right)^2 d\gamma
\]

(50)

(51)

where $v = \dot{\alpha}_t x_0 + \dot{\sigma}_t \epsilon, \tilde{v} = v/\sqrt{\dot{\alpha}_t^2 + \dot{\sigma}_t^2}$. For VP and SP schedule, since $\gamma = \log(\sigma_{\gamma}^2/\alpha_{\gamma}^2)$, using their schedule properties, $\alpha_{\gamma}, \sigma_{\gamma}$ are deterministic functions of $\gamma$ without any hyperparameters. Thus, we can derive their specific
The solution has a closed-form expression, which gives the inverse transformation from \( t \) to \( \gamma \):

\[
\frac{1}{Z} \int_{\gamma_0}^{\gamma_t} \alpha_{\gamma}^2 d\gamma = t, \quad Z = \int_{\gamma_0}^{\gamma_t} \sigma_{\gamma}^2 d\gamma
\] (52)

Here we assume maximum time \( T = 1 \), and \( Z \) is a normalizing constant.

**VP**  We have (omit the constant of the indefinite integral)

\[
\int \alpha_{\gamma}^2 d\gamma = \log \frac{1}{1 + \exp(-\gamma)} = \log \alpha_{\gamma}
\] (53)

Then the equation for inverse transform sampling is

\[
\log \frac{1}{1 + \exp(-\gamma)} - \log \alpha_{\gamma_0}^2 = Zt, \quad Z = \log \frac{\sigma_{\gamma}^2}{\sigma_{\gamma_0}^2}
\] (54)

The solution has a closed-form expression, which gives the inverse transformation from \( t \) to \( \gamma \):

\[
\gamma_t = \log \frac{1}{\exp(-Zt)/\sigma_{\gamma_0}^2 - 1}, \quad t \sim U(0, 1)
\] (55)

**SP**  We have (omit the constant of the indefinite integral)

\[
\int \sigma_{\gamma}^2 d\gamma = -2 \left( \log(1 + \exp(-\gamma/2)) + \frac{1}{1 + \exp(-\gamma/2)} \right)
\] (56)

Denote \( F(\gamma) = -\log(1 + \exp(-\gamma/2)) - (1 + \exp(-\gamma/2))^{-1} \), then the equation for inverse transform sampling is

\[
\frac{F(\gamma_t) - F(\gamma_0)}{F(\gamma_1) - F(\gamma_0)} = t
\] (57)

The solution has no closed-form expressions. Similar to the implementation in Song et al. (2021b), we use the bisection method to find the root.

<table>
<thead>
<tr>
<th>Formula</th>
<th>VP</th>
<th>SP</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \alpha_{\gamma} )</td>
<td>( \sqrt{\frac{1}{1 + \exp(\gamma)}} )</td>
<td>( \sqrt{\frac{1}{1 + \exp(\gamma/2)}} )</td>
</tr>
<tr>
<td>( \sigma_{\gamma} )</td>
<td>( \sqrt{\frac{1}{1 + \exp(-\gamma)}} )</td>
<td>( \sqrt{\frac{1}{1 + \exp(-\gamma/2)}} )</td>
</tr>
<tr>
<td>( \hat{\alpha}_{\gamma} )</td>
<td>(-\frac{1}{2} \alpha_{\gamma} \sigma_{\gamma}^2 )</td>
<td>(-\frac{1}{2} \alpha_{\gamma} \sigma_{\gamma} )</td>
</tr>
<tr>
<td>( \hat{\sigma}_{\gamma} )</td>
<td>( \frac{1}{2} \alpha_{\gamma} \sigma_{\gamma} )</td>
<td>( \frac{1}{2} \alpha_{\gamma} \sigma_{\gamma} )</td>
</tr>
<tr>
<td>( \hat{v} )</td>
<td>( \alpha_{\gamma} \epsilon - \sigma_{\gamma} x_0 )</td>
<td>( \frac{\alpha_{\gamma}}{\epsilon - x_0} )</td>
</tr>
<tr>
<td>( J_{FM} )</td>
<td>( \frac{1}{2} \int_{\gamma_0}^{\gamma_T} \alpha_{\gamma}^2 E_{\nu, 0, \epsilon}(\theta, \gamma) - \hat{\theta} \parallel d\gamma )</td>
<td>( \int_{\gamma_0}^{\gamma_T} \alpha_{\gamma}^2 E_{\nu, 0, \epsilon}(\theta, \gamma) - \hat{\theta} \parallel d\gamma )</td>
</tr>
<tr>
<td>( \hat{J}_{FM, tr} )</td>
<td>( \frac{1}{2} \int_{\gamma_0}^{\gamma_T} \alpha_{\gamma}^2 E_{\nu, 0, \epsilon}(\sigma, \nu, \epsilon) (\hat{\theta})^2 d\gamma )</td>
<td>( \int_{\gamma_0}^{\gamma_T} \alpha_{\gamma}^2 E_{\nu, 0, \epsilon}(\sigma, \nu, \epsilon) (\hat{\theta})^2 d\gamma )</td>
</tr>
<tr>
<td>( \epsilon_\theta(x, \gamma, \gamma') )</td>
<td>( \sigma_{\gamma} x_{\gamma} + \alpha_{\gamma} \hat{\theta}(x, \gamma, \gamma') )</td>
<td>( \hat{\theta}_\nu(x, \gamma, \gamma') )</td>
</tr>
</tbody>
</table>

Table 4. Specification of related values and objectives under VP and SP schedule.
D. Illustration of velocity prediction and imbalance problem

First, we give an illustrative intuition of our velocity parameterization and corresponding flow matching objective in Section 4.1. As shown in Figure 5(a), for each pair \((x_0, \epsilon)\) where \(x_0 \sim q_0(x_0)\) and \(\epsilon \sim \mathcal{N}(0, I)\), let \(x_t = \alpha_t x_0 + \sigma_t \epsilon\). As \(t\) increases, \(x_t\) moves from \(x_0\) to \(\epsilon\) gradually, forming a diffusion path in the sample space, and \(\nu\) is the velocity \(\frac{\partial x_t(x_0, \epsilon)}{\partial \epsilon}\) across the path. Thus, minimizing \(J_{FM}\) is to predict the expected velocity for all possible \((x_0, \epsilon)\) pairs.

Next, we interpret the superiority of velocity prediction from the perspective of balanced prediction difficulty. Intuitively, the noise prediction model suffers from an imbalance problem: at small \(t\), \(x_t\) is similar to data, and extracting the insignificant noise component is hard; at large \(t\), \(x_t\) is similar to noise, so the noise prediction is easy and has a small error. Velocity prediction, on the other hand, has a property that the prediction target \(\nu\) is less relevant to input \(x_t\). In Fig. 5(b) we empirically confirm it on our pretrained model. We plot the mean square prediction error (MSE) w.r.t. time \(t\), which shows that velocity prediction alleviates the imbalance problem by enlarging the training at large \(t\). Since the overall error is a weighted combination of the MSE at different \(t\) and is invariant to the parameterization, we can conclude that under noise prediction, the MSE is lower near \(t = 1\), but is imposed a larger weight, so it has a larger gradient variance.

E. Relationship between velocity parameterization and other works

In this section, we demonstrate how the techniques in related works (Karras et al., 2022; Lipman et al., 2022; Salimans & Ho, 2022; Ho et al., 2022) can be reformulated as velocity parameterization.

E.1. Interpretation by preconditioning

Works that aim at improving the sample quality of diffusion models also consider the network parameterizations that adaptively mix signal and noise. Karras et al. (2022) proposes to precondition the neural network with a time-dependent skip connection that allows it to estimate either data \(x_0\) or noise \(\epsilon\), or something in between. Similarly, we write the noise predictor \(\epsilon_\theta(\cdot)\) in the following formulation:

\[
\epsilon_\theta(x, \gamma) = c_{\text{skip}}(\gamma)x + c_{\text{out}}(\gamma)F_\theta(c_{\text{in}}(\gamma)x, \gamma)
\]  

(58)

where \(F_\theta(\cdot)\) is the pure network, \(F_\theta(\cdot) = \alpha_\gamma x_0 + \sigma_\gamma \epsilon\). The flow matching loss can be rewritten as

\[
J_{FM}(\theta) = \frac{1}{2} \int_{\gamma_0}^{\gamma_T} \mathbb{E}_{x_0, \epsilon} \left[ \| \epsilon_\theta(x, \gamma) - \epsilon \|^2 \right]
\]

\[
= \frac{1}{2} \int_{\gamma_0}^{\gamma_T} \mathbb{E}_{x_0, \epsilon} \left[ \| c_{\text{skip}}(\gamma)x_\gamma + c_{\text{out}}(\gamma)F_\theta(c_{\text{in}}(\gamma)x_\gamma, \gamma) - \epsilon \|^2 \right]
\]

(59)

\[
= \frac{1}{2} \int_{\gamma_0}^{\gamma_T} \mathbb{E}_{x_0, \epsilon} \left[ c_{\text{out}}(\gamma)^2 \| F_\theta(c_{\text{in}}(\gamma)x_\gamma, \gamma) - F_{\text{target}}(x_0, \epsilon, \gamma) \|^2 \right]
\]
where

$$F_{target}(x_0, \epsilon, \gamma) = \frac{\epsilon - c_{skip}(\gamma)x_\gamma}{c_{out}(\gamma)}$$ (60)

Following first principles in EDM, we derive formulas for $c_{in}(\gamma)$, $c_{out}(\gamma)$, $c_{skip}(\gamma)$ to ensure:

1. The training inputs of $F_\theta(\cdot)$ have unit variance.
2. The effective training target $F_{target}$ has unit variance.
3. We select $c_{skip}(\gamma)$ to minimize $c_{out}(\gamma)$, so that the errors of $F_\theta$ are amplified as little as possible.

From principle 1, we have

$$1 = \text{Var}[c_{in}(\gamma)x_{\gamma}]$$
$$1 = \text{Var}[c_{in}(\gamma)(\alpha_{\gamma}x_0 + \sigma_{\gamma}\epsilon)]$$
$$1 = c_{in}^2(\gamma)(\alpha_{\gamma}^2\sigma_{\text{data}}^2 + \sigma_{\gamma}^2)$$
$$c_{in}(\gamma) = \frac{1}{\sqrt{\sigma_{\gamma}^2 + \sigma_{\text{data}}^2\alpha_{\gamma}^2}}$$ (61)

From principle 2, we have

$$1 = \text{Var}[F_{target}(x_0, \epsilon, \gamma)]$$
$$1 = \text{Var}\left[\frac{\epsilon - c_{skip}(\gamma)x_{\gamma}}{c_{out}(\gamma)}\right]$$
$$c_{out}^2(\gamma) = \text{Var}[\epsilon - c_{skip}(\gamma)x_{\gamma}]$$
$$c_{in}^2(\gamma) = \text{Var}[\epsilon - c_{skip}(\gamma)(\alpha_{\gamma}x_0 + \sigma_{\gamma}\epsilon)]$$
$$c_{out}^2(\gamma) = \text{Var}[(1 - c_{skip}(\gamma)\sigma_{\gamma})\epsilon - c_{skip}(\gamma)\alpha_{\gamma}x_0]$$
$$c_{out}^2(\gamma) = (1 - c_{skip}(\gamma)\sigma_{\gamma})^2 + c_{skip}^2(\gamma)\alpha_{\gamma}^2\sigma_{\text{data}}^2$$ (62)

From principle 3, we have

$$0 = \frac{dc_{out}^2(\gamma)}{dc_{skip}(\gamma)}$$
$$0 = -2\sigma_{\gamma}(1 - \sigma_{\gamma}c_{skip}(\gamma)) + 2\alpha_{\gamma}^2\sigma_{\text{data}}^2c_{skip}(\gamma)$$
$$c_{skip}(\gamma) = \frac{\sigma_{\gamma}}{\sigma_{\gamma}^2 + \sigma_{\text{data}}^2\alpha_{\gamma}^2}$$ (63)

We now substitute Eqn. (63) into Eqn. (62) to obtain the formula for $c_{out}(\gamma)$:

$$c_{out}(\gamma) = \frac{\sigma_{\text{data}}\alpha_{\gamma}}{\sqrt{\sigma_{\gamma}^2 + \sigma_{\text{data}}^2\alpha_{\gamma}^2}}$$ (64)

If we assume $\sigma_{\text{data}} = 1$ and consider VP schedule, we have $\alpha_{\gamma}^2 + \sigma_{\gamma}^2 = 1$, and the coefficients are reduced to

$$c_{in}(\gamma) = 1, \quad c_{skip}(\gamma) = \sigma_{\gamma}, \quad c_{out}(\gamma) = \alpha_{\gamma}$$ (65)

In this case, the preconditioning is in agreement with our velocity parameterization by $\tilde{v}_\theta(x_{\gamma}, \gamma) = F_\theta(x_{\gamma}, \gamma)$. In practice, we find setting $\sigma_{\text{data}} = 0.5$ as in Karras et al. (2022) leads to faster descent of the loss at the start, but slower convergence as the training proceeds.
E.2. Connection to flow matching in Lipman et al. (2022)

Lipman et al. (2022) defines a conditional probability path \( p_t(x|x_0) \) that gradually moves the data \( x_0 \) to a target distribution \( p_1(x) \). Note that they use \( t = 1 \) to represent data distribution and \( t = 0 \) to represent target distribution. To be consistent, we reverse their time representation. They obtain the marginal probability path by marginalizing over \( q(x_0) \):

\[
p_t(x) = \int p_t(x|x_0)q(x_0)dx_0
\]  

(66)

They want to learn a vector field \( v_t(x) \), which defines a flow \( \phi : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}^d \) by

\[
\frac{d}{dt}\phi_t(x) = v_t(\phi_t(x)), \quad \phi_1(x) = x
\]  

(67)

so that the marginal \( p_t \) can be generated by the push-forward \( p_t = [\phi_t]_* p_1 \). In practice, they consider the Gaussian conditional probability paths

\[
p_t(x|x_0) = \mathcal{N}(x|\mu_t(x_0), \sigma_t^2(x_0)I)
\]  

(68)

and propose a conditional flow matching (CFM) objective for simulation-free training of \( v_t(x) \)

\[
\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t,q(x_0), p_t(x|x_0)} \|v_t(x) - u_t(x|x_0)\|_2^2
\]  

(69)

where

\[
u_t(x|x_0) = \frac{\sigma_t^2(x_0)}{\sigma_t(x_0)}(x - \mu_t(x_0)) + \mu_t(x_0)
\]  

(70)

Suppose the mean \( \mu_t(x_0) \) is linear to \( x_0 \), and the standard deviation \( \sigma_t(x_0) \) is invariant to \( x_0 \), as the two experimented cases in flow matching. By setting \( \mu_t(x_0) = \alpha_t x_0, \sigma_t(x_0) = \sigma_t \), we have

\[
u_t(x|x_0) = \frac{\gamma_t}{\sigma_t}(x - \alpha_t x_0) + \hat{\alpha}_t x_0 + \hat{\sigma}_t \epsilon
\]  

(71)

where we use \( x = \alpha_t x_0 + \sigma_t \epsilon, \epsilon \sim \mathcal{N}(0, I) \) since \( p_t(x|x_0) = \mathcal{N}(x|\alpha_t x_0, \sigma_t^2 I) \). Then we can observe that they are corresponding to our notations: the conditional probability path \( p_t(x|x_0) \) corresponds to the Gaussian transition kernel \( q_0(x_t|x_0) \) of the forward diffusion process; the marginal probability path \( p_t(x) \) corresponds to the ground-truth marginals \( q_t(x_t) \) associated with the forward diffusion process; the matching target \( u_t(x|x_0) \) in CFM corresponds to the velocity of the diffusion path \( v = \hat{\alpha}_t x_0 + \hat{\sigma}_t \epsilon \) in our formulation.

Therefore, the CFM objective in Lipman et al. (2022) is actually velocity parameterization when specific to Gaussian diffusion processes, which is similar to our first-order objective of the pretraining phase. We can express CFM in a simpler form, which is easier to analyze and generalize to any noise schedule. Then by the equivalence of different predictors (Theorem B.1) and the relationship between \( f(t) \), \( g(t) \) and \( \alpha_t, \sigma_t \), we have

\[
\mathcal{L}_{\text{CFM}}(\theta) = \int_0^T \mathbb{E}_{x_0, \epsilon} \|v_\theta(x_t, t) - v\|_2^2 dt
\]

\[
= \int_0^T \mathbb{E}_{x_0, \epsilon} \left\| f(t)x_t - \frac{1}{2} g^2(t)s_\theta(x_t, t) - (\hat{\alpha}_t x_0 + \hat{\sigma}_t \epsilon) \right\|_2^2 dt
\]

\[
= \int_0^T \frac{1}{4} \bar{g}^2(t)\mathbb{E}_{x_0, \epsilon} \left\| s_\theta(x_t, t) + \frac{\epsilon}{\sigma_t} \right\|_2^2 dt
\]  

(72)

which demonstrates that the CFM objective not only changes the parameterization but also imposes a different time weighting \( w(t) = \frac{1}{4} \bar{g}^2(t) \) on the original denoising score matching objective. When the training aims for improving the sample quality (e.g., FID), the optimal choice for \( w(t) \) is still an open problem.

Comparing the CFM objective to our first-order objective Eqn. (50), the practical differences are that we use normalized predictor \( \hat{v}_\theta \), \( \gamma \) timing, and apply likelihood weighting. The likelihood weighting refers to time weighting \( w(t) = \frac{\bar{g}^2(t)}{2\sigma_t^4} \) in Eqn. (5) and \( w(t) = \frac{\bar{g}^2(t)}{g^2(t)} \) in Eqn. (12), which is consistent under different parameterizations and is the theoretically optimal
choice for maximum likelihood training (Song et al., 2021c). Also, changing the time domain from \( t \) to \( \gamma \) will not alter the value of the objective, but will affect the variance of Monte-Carlo estimation and the convergence speed, as we have discussed. For example, the OT path in Lipman et al. (2022) is \( \alpha_t = 1 - t, \sigma_t = 1 - (1 - \sigma_{\text{min}})(1 - t) \approx t \), and the relation between \( \gamma \) and \( t \) is \( \gamma = \log(\sigma_t^2/\alpha_t^2) = 2 \log(t/(1 - t)) \). Under \( \gamma \) timing, we can decouple the choice of noise schedules to the greatest extent, and regard the change of variable from \( \gamma \) to \( t \) as a tunable importance sampling procedure.

Besides, normalizing the field is necessary for stable training of the velocity predictor and is the key to unifying \( v \) prediction and preconditioning. Such strategies have also been adopted in more general physics-inspired generative models. For example, Xu et al. (2022; 2023) propose to normalize the Poisson field when training Poisson flow generative models. In Salimans & Ho (2022); Ho et al. (2022), a technique called “\( v \) prediction” is used, which parameterizes a network to predict \( v = \alpha_t \epsilon - \sigma_t x_0 \). Assuming a VP schedule following their choice, we have \( \alpha_t^2 + \sigma_t^2 = 1 \), so by taking the derivative w.r.t. \( t \) we have \( \alpha_t \dot{\alpha}_t + \sigma_t \dot{\sigma}_t = 0 \), then

\[
\dot{\alpha}_t = -\frac{\sigma_t \dot{\sigma}_t}{\alpha_t}, \quad \frac{d \log \alpha_t}{dt} = \frac{\dot{\alpha}_t}{\alpha_t} = -\frac{\sigma_t \dot{\sigma}_t}{\alpha_t^2}
\]

so

\[
g^2(t) = \frac{d \sigma_t^2}{dt} - 2 \frac{d \log \alpha_t}{dt} \sigma_t^2
\]

\[
= 2 \sigma_t \dot{\sigma}_t + 2 \frac{\sigma_t \dot{\sigma}_t}{\alpha_t} \sigma_t^2
\]

\[
= 2 \frac{\sigma_t \dot{\sigma}_t}{\alpha_t^2} (\alpha_t^2 + \sigma_t^2)
\]

\[
= 2 \frac{\sigma_t \dot{\sigma}_t}{\alpha_t^2}
\]

and the velocity is

\[
v = \dot{\alpha}_t x_0 + \dot{\sigma}_t \epsilon
\]

\[
= \frac{\dot{\sigma}_t}{\alpha_t} (\alpha_t \epsilon - \sigma_t x_0)
\]

\[
= \frac{\alpha_t \dot{\sigma}_t}{\alpha_t^2} (\alpha_t \epsilon - \sigma_t x_0)
\]

Besides, we can compute the normalizing factor as

\[
\sqrt{\dot{\alpha}_t^2 + \dot{\sigma}_t^2} = \sqrt{\frac{\sigma_t^2 \dot{\sigma}_t^2}{\alpha_t^2} + \dot{\sigma}_t^2} = \frac{\dot{\sigma}_t}{\alpha_t} = \frac{\alpha_t}{2 \sigma_t} g^2(t)
\]

so we have the normalized velocity

\[
\ddot{v} = \frac{v}{\sqrt{\dot{\alpha}_t^2 + \dot{\sigma}_t^2}} = \alpha_t \epsilon - \sigma_t x_0
\]

Therefore, \( v = \ddot{v} \), which means that \( v \) prediction is a special case of velocity parameterization when the noise schedule is VP.

**F. Error-bounded trace of second-order flow matching**

Here we provide the proofs for the error-bounded trace of second-order flow matching. First, we provide a lemma that gives the Jacobian of the ground-truth velocity predictor \( v^*(x_t, t) \).

**Lemma F.1.** Suppose \((x_0, x_t) \sim q(x_0, x_t)\), denote \( x_t = \alpha_t x_0 + \sigma_t \epsilon, v = \dot{\alpha}_t x_0 + \dot{\sigma}_t \epsilon, \nabla(\cdot) = \nabla_{x_t}(\cdot) \), we have

\[
\nabla v^*(x_t, t) = \frac{\dot{\alpha}_t}{\sigma_t} I - \frac{2}{g^2(t)} \mathbb{E}_{q(x_0|x_t)} \left[ (v^*(x_t, t) - v)(v^*(x_t, t) - v)^\top \right]
\]

We have

\[
\frac{\dot{\alpha}_t}{\sigma_t} I - \frac{2}{g^2(t)} \mathbb{E}_{q(x_0|x_t)} \left[ (v^*(x_t, t) - v)(v^*(x_t, t) - v)^\top \right]
\]
and
\[
\text{tr}(\nabla v^*(x_t, t)) = \frac{\hat{\sigma}_t}{\sigma_t} d - \frac{2}{g^2(t)} E_{q_0(x_0|x_t)} \left[ ||v^*(x_t, t) - v||^2 \right] \tag{79}
\]

**Proof.** First, the gradient of \( q_0 \) can be calculated as

\[
\nabla q_0(x_0|x_t) = \nabla \frac{q_0(x_0)q_0_t(x_t|x_0)}{q_t(x_t)} = \frac{q_0(x_0)q_0_t(x_t|x_0) - q_0_t(x_t|x_0) \nabla q_t(x_t)}{q_t(x_t)^2} = \frac{q_0(x_0)q_0_t(x_t|x_0)}{q_t(x_t)} \left( \nabla \log q_0_t(x_t|x_0) - \nabla \log q_t(x_t) \right)
\]

where we use the relation between \( v^*(x_t, t) \) and \( \nabla \log q_t(x_t) \) in Theorem B.1. From Eqn. (48), we know \( v^*(x_t, t) = E_{q_0(x_0|x_t)}[v] \), and for given \( x_0 \), we have

\[
\nabla v = \nabla \left( \hat{\sigma}_t x_0 + \frac{x_t - \alpha_t x_0}{\sigma_t} \right) = \frac{\hat{\sigma}_t}{\sigma_t} I \tag{81}
\]

and

\[
E_{q_0(x_0|x_t)} \left[ (v^*(x_t, t) - v)v^*(x_t, t)^\top \right] = E_{q_0(x_0|x_t)} [v^*(x_t, t) - v] v^*(x_t, t)^\top = 0 \tag{82}
\]

So

\[
\nabla v^*(x_t, t) = \nabla \int q_0(x_0|x_t)vdx_0
\]

\[
= \int \nabla q_0(x_0|x_t)v^\top + q_0(x_0|x_t)\nabla vdx_0
\]

\[
= \int q_0(x_0|x_t) \left( \frac{2}{g^2(t)} (v^*(x_t, t) - v)v^\top + \frac{\hat{\sigma}_t}{\sigma_t} I \right) dx_0 \tag{83}
\]

\[
= \frac{\hat{\sigma}_t}{\sigma_t} I + \frac{2}{g^2(t)} E_{q_0(x_0|x_t)} [(v^*(x_t, t) - v)v^\top]
\]

\[
= \frac{\hat{\sigma}_t}{\sigma_t} I - \frac{2}{g^2(t)} E_{q_0(x_0|x_t)} [(v^*(x_t, t) - v)(v^*(x_t, t) - v)^\top]
\]

The expression for \( \text{tr}(\nabla v^*(x_t, t)) \) can be easily derived from the above equation. \( \square \)

Then we prove Theorem 4.1 as follows.

**Proof.** The optimization in Eqn. (15) can be rewritten as

\[
\theta^* = \arg\min_{\theta} \frac{2\sigma_t^2}{g^2(t)} E_{q_1(x_t)} E_{q_0(x_0|x_t)} \left[ v^\text{trace}_2(x_t, t; \theta) - \frac{\hat{\sigma}_t}{\sigma_t} d + \frac{2}{g^2(t)} ||\hat{v}_1(x_t, t) - v||^2_2 \right]^2 . \tag{84}
\]

For fixed \( t \) and \( x_t \), minimizing the inner expectation is a minimum mean square error problem for \( v^\text{trace}_2(x_t, t; \theta) \), so the optimal \( \theta^* \) satisfies

\[
v^\text{trace}_2(x_t, t; \theta^*) = \frac{\hat{\sigma}_t}{\sigma_t} d - \frac{2}{g^2(t)} E_{q_0(x_0|x_t)}[||\hat{v}_1(x_t, t) - v||^2_2] \tag{85}
\]

22
Using Lemma F.1 and $v^*(x_t, t) = \mathbb{E}_{q_{t0}(x_t|x_1)}[v]$, we have

\[
\text{tr}(\nabla v^*(x_t, t)) - v_2^{\text{trace}}(x_t, t; \theta^*) \\
= \frac{2}{g^2(t)} \mathbb{E}_{q_{t0}(x_t|x_1)} \left[ \|\hat{v}_1(x_t, t)\|^2_2 - 2v^T \hat{v}_1(x_t, t) - \|v^*(x_t, t)\|^2_2 + 2v^T v^*(x_t, t) \right] \\
= \frac{2}{g^2(t)} \|\hat{v}_1(x_t, t) - v^*(x_t, t)\|^2_2
\]  

(86)

Therefore, we can obtain the error bound by

\[
|v_2^{\text{trace}}(x_t, t; \theta) - \text{tr}(\nabla x v^*(x_t, t))| \leq |v_2^{\text{trace}}(x_t, t; \theta) - v_2^{\text{trace}}(x_t, t; \theta^*)| + |v_2^{\text{trace}}(x_t, t; \theta^*) - \text{tr}(\nabla v^*(x_t, t))| \\
= |v_2^{\text{trace}}(x_t, t; \theta) - v_2^{\text{trace}}(x_t, t; \theta^*)| + \frac{2}{g^2(t)} \delta_1(x_t, t)
\]

(87)

where $\delta_1(x_t, t) = \|\hat{v}_1(x_t, t) - v^*(x_t, t)\|_2$ is the first-order estimation error.

G. Difference between our second-order flow matching and the previous time score matching in Choi et al. (2022)

We propose the error-bounded second-order flow matching objective to regularize $-\text{tr}(\nabla x v_0(x_t, t))$, which is equal to $\frac{d \log p_t(x_t)}{dt}$ by the “Instantaneous Change of Variables” formula of CNFs (Chen et al., 2018a). Choi et al. (2022) proposes a joint score matching method to estimate the data score as well as the time score $(\nabla x \log p_t(x), \partial_t \log p_t(x))$, which seems related. However, they are essentially different.

Firstly, the change-of-variable for CNFs describes the total derivative of $\log p_t(x_t)$ w.r.t. $x_t$, which evolves by the ODE flow trajectory, not each fixed data point $x \in \mathbb{R}^d$. However, $\partial_t \log p_t(x)$ in Choi et al. (2022) describes the partial derivative of $\log p_t(x)$ for $x \in \mathbb{R}^d$, i.e., any fixed data point in the whole space. Specifically, according to the Fokker-Planck equation, we have

\[
\partial_t p_t(x) = -\nabla x \cdot (p_t(x) v_0(x, t))
\]

(88)

It follows that

\[
\partial_t \log p_t(x) = -\nabla x \cdot v_0(x, t) - v_0(x, t)^\top \nabla x \log p_t(x)
\]

(89)

Therefore, the total derivative $\frac{d \log p_t(x_t)}{dt}$ we care about is different from the partial derivative $\partial_t \log p_t(x)$ in Choi et al. (2022), and their training objectives are also different (with different optimal solutions).

Moreover, there is another difference: Choi et al. (2022) trains another model to estimate the partial derivative $\partial_t \log p_t(x)$, which is independent of the ODE velocity $v_0(x, t)$ (in the form of the parameterized score function $s_\theta(x, t)$). However, our method restricts the parameterized velocity $v_\theta(x, t)$ itself, and does not employ another model.

Finally, the techniques used in Choi et al. (2022) and our work are also different. Choi et al. (2022) estimates the score matching loss for the partial derivative $\partial_t \log p_t(x)$ by the well-known integral-by-parts, which is used to derive the famous sliced score matching (Song et al., 2020), to avoid the computation of the score function $\nabla x \log p_t(x)$; However, our method leverages the property of mean square error (that its minimum is conditional mean), which is used to derive the famous denoising score matching (Vincent, 2011), to estimate the divergence of $v_\theta(x, t)$. In the score matching literature, sliced score matching and denoising score matching are two rather different techniques. As first-order denoising score matching is widely used in training diffusion models (such as Song et al. (2021c)), our proposed second-order flow matching is also suitable for training diffusion ODEs.
In this section, we give details of our adaptive IS stated in Section 4.4. First, we parameterize $\gamma_\eta(t)$ similar to Kingma et al. (2021):

$$
\gamma_\eta(t) = \gamma_0 + (\gamma_T - \gamma_0) \frac{\tilde{\gamma}_\eta(t) - \tilde{\gamma}_\eta(0)}{\tilde{\gamma}_\eta(1) - \tilde{\gamma}_\eta(0)}
$$

(90)

where $\tilde{\gamma}_\eta(t)$ is a dense monotone increasing network. Concretely, we use a two-layer fully-connected network $\tilde{\gamma}_\eta(t) = l_2(\phi(l_1(t)))$ where $\phi$ is the sigmoid activation function, $l_1, l_2$ are linear layers with positive weight and output units of 1024 and 1.

Algorithm 1 Adaptive importance sampling (single iteration)

\textbf{Require:} velocity network $v_\theta$, IS network $\tilde{\gamma}_\eta$, noise schedule $\alpha_\gamma, \sigma_\gamma$, batch size $N$

\begin{itemize}
  \item Sample $x^{(1)}_0, \ldots, x^{(N)}_0$ from data distribution
  \item Sample $\epsilon^{(1)}, \ldots, \epsilon^{(N)}_N$ from standard Gaussian distribution $N(0, I)$
  \item Sample $t^{(1)}, \ldots, t^{(N)}$ from uniform distribution $U(0, 1)$
  \item Calculate $\gamma_\eta(t^{(i)}): i = 1, \ldots, N$ by Eqn. (90)
  \item Fix $\eta$, optimize $\theta$ to minimize $\frac{1}{N} \sum_{i=1}^{N} L_{\theta, \eta}(x^{(i)}_0, \epsilon^{(i)}, t^{(i)})$
  \item Fix $\theta$, optimize $\eta$ to minimize $\frac{1}{N} \sum_{i=1}^{N} L_{\theta, \eta}(x^{(i)}_0, \epsilon^{(i)}, t^{(i)})$
\end{itemize}

Then we present our adaptive IS procedure in Algorithm 1. Kingma et al. (2021) proposes to reuse the gradient $\nabla_{\eta} \mathcal{L}_{\theta, \eta}(x_0, \epsilon, t)$ to optimize $\eta$ and avoid a second backpropagation by decomposing the gradient $\nabla_{\eta} \mathcal{L}_{\theta, \eta}^2(x_0, \epsilon, t)$ using chain-rule. We simply their learning of $\tilde{\gamma}_\eta$ by removing the complex gradient operation in one iteration and propose to alternatively optimize $\theta$ and $\eta$. It may take extra overhead, but also seeks the optimal IS and is enough for ablation.

I. Experiment details

In this section, we provide details of our experiment settings. Our network, hyperparameters and training are the same for different noise schedules on the same dataset.

Model architectures Our diffusion ODEs are parameterized in terms of the $\gamma$-timed normalized velocity predictor $v_\theta(x, \gamma)$, based on the U-Net structure of Kingma et al. (2021). This architecture is tailor-made for maximum likelihood training, employing special designs such as removing the internal downsampling/upsampling and adding Fourier features for fine-scale prediction. Our configuration for each dataset also follows Kingma et al. (2021): For CIFAR-10, we use U-Net of depth 32 with 128 channels; for ImageNet-32, we still use U-Net of depth 32, but double the number of channels to 256. All our models use a dropout rate of 0.1 in the intermediate layers.

Hyperparameters and training We follow the same default training settings as Kingma et al. (2021). For all our experiments, we use the Adam (Kingma & Ba, 2014) optimizer with learning rate $2 \times 10^{-4}$, exponential decay rates of $\beta_1 = 0.9, \beta_2 = 0.99$ and decoupled weight decay (Loshchilov & Hutter, 2019) coefficient of 0.01. We also maintain an exponential moving average (EMA) of model parameters with an EMA rate of 0.9999 for evaluation.

For other hyperparameters, we use fixed start and end times which satisfy $\gamma_e = -13.3, \gamma_T = 5.0$, which is the default setting in Kingma et al. (2021). In the finetuning stage, we simply set the coefficient $\lambda$ in the mixed loss $\mathcal{J}_{\text{FM}}(\theta) + \lambda \mathcal{J}_{\text{FM}, \text{tr}}(\theta)$ as 0.1 without no further tuning, so that the magnitude of the second-order loss is negligible w.r.t the first-order loss. Since the first-order matching accuracy is critical to the second-order matching, a large $\lambda$ will make the training unstable or even degenerate the likelihood performance.

All our training processes are conducted on 8 GPU cards of NVIDIA A40 expect for ImageNet-32 (old version). For CIFAR-10, we pretrain the model for 6 million iterations, which takes around 3 weeks. Then we finetune the model for 200k iterations, which takes around 1 day. For ImageNet-32 (new version), we pretrain the model for 2 million iterations, which takes around 2 weeks. Then we finetune the model for 250k iterations, which takes around 3 days. We use a batch size of 128 for both training stages and both datasets.

Note that in related works (Lipman et al., 2022; Albergo & Vanden-Eijnden, 2022), experiments on ImageNet-32 (new
version) are conducted at a larger batch size (512 or 1024), which may improve the results. We did not use a larger batch size or train longer due to resource limitations.

For ImageNet-32 (old version), the training processes are conducted on 8 GPU cards of NVIDIA A100 (40GB). We pretrain the model for 2 million iterations using a batch size of 512, which takes around 2 weeks. Then we finetune the model for 500k iterations using a batch size of 128 and accumulate the gradient for every 4 batches, which takes around 2.5 days.

**Likelihood and sample quality** For likelihood, we use our truncated-normal dequantization. When the number of importance samples $\hat{K} = 1$, we report the BPD on the test dataset with 5 times repeating to reduce the variance of the trace estimator. When $\hat{K} = 5$ or $\hat{K} = 20$, we do not repeat the dataset since the log-likelihood of a data sample is already evaluated multiple times. For sampling, since we are concentrated on ODE, we simply use an adaptive-step ODE solver with RK45 method (Dormand & Prince, 1980) (relative tolerance $10^{-5}$ and absolute tolerance $10^{-5}$). We generate 50k samples and report the FIDs on them. Utilizing high-quality sampling procedures such as PC sampler (Song et al., 2021c) or fast sampling algorithms such as DPM-Solver (Lu et al., 2022b) may improve the results, which are left for future work.

### J. Additional samples

![Additional samples](image)

*Figure 6. Random samples for ablation by ODE sampler. Our pretraining and finetuning lead to a better likelihood with small visual quality degeneration.*
Figure 7. Random samples by ODE sampler (CIFAR-10, VP).

Figure 8. Random samples by ODE sampler (CIFAR-10, SP).
Figure 9. Random samples by ODE sampler (ImageNet-32, VP).

Figure 10. Random samples by ODE sampler (ImageNet-32, SP).