STABLE TARGET FIELD FOR REDUCED VARIANCE SCORE ESTIMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

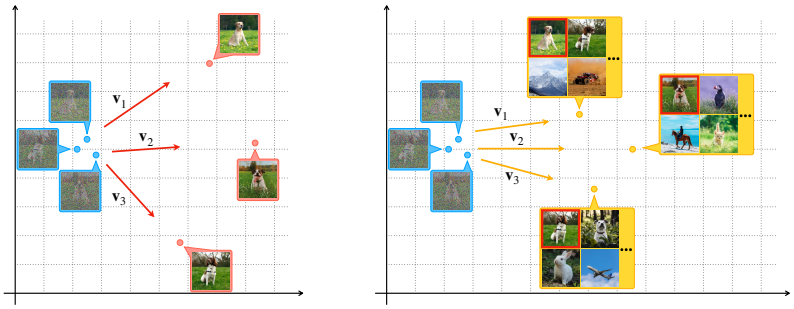
Score-based generative models (SGMs) generate samples by reversing a fixed forward diffusion process. Despite impressive empirical results, we observe that the training process leads to unstable outcomes, especially when the reverse-time solvers adopt a large step size. The performance of converged models varies significantly with different random seeds, and they produce noticeable artifacts in generated samples. We suggest that the source of such instability lies in the handling of intermediate noise-variance scales, where multiple modes in the data affect the direction of reverse paths. Thus, the score-matching objective has a large sample variance in this regime, explaining lesser quality score estimates. We propose to remedy the problem by incorporating *a reference batch* for minibatch updates where the reference batch is used to calculate weighted conditional scores as the more stable training targets. We show that the procedure indeed helps in the challenging intermediate regime by reducing (the trace of) the covariance of training targets. The new stable targets can be seen as trading bias for reduced variance where the bias vanishes with increasing reference batch size. Empirically, we show that the new objective improves the image quality of state-of-the-art SGMs across datasets with both general ODE and SDE solvers. In particular, our method improves and stabilizes the final performance of SGMs, as well as speeding up the training process.

1 INTRODUCTION

Score-based generative models (SGMs) (Song & Ermon, 2019; 2020) and diffusion models (Ho et al., 2020) have recently achieved impressive results on a wide spectrum of generative tasks, such as image generation (Nichol et al., 2022; Song et al., 2021), 3D point cloud generation (Luo & Hu, 2021) and molecular conformer generation (Shi et al., 2021; Xu et al., 2022a). These models can be subsumed under a unified framework in the form of Itô stochastic differential equations (SDE) (Song et al., 2021). The models are learned as time-dependent score fields via the score-matching objective (Vincent, 2011). The resulting score field then guides the reverse SDE during generative sampling. Among the SDE variants, the so-called variance-exploding (VE) SDE provides the best performance to date (Song et al., 2021).

Despite impressive empirical results, we observe that SGMs can be unstable, especially with samples that adopt larger step sizes. This variability can result in noticeable artifacts in generated samples. To better understand the origin of instability, we decompose the score-field into three regimes, and show that the phenomenon arises primarily due to variability of training targets in the intermediate regime. This regime is characterized by multiple modes or data points exhibiting comparable influences on the scores. In other words, in this regime, the sources of the noisy examples generated in the course of the forward process become ambiguous. As we illustrate in Figure 1(a), owing to this issue each stochastic update of the score model is based on disparate targets.

We propose a generalized version of the score-matching objective, termed the *Stable Target Field* (**STF**) objective. The idea is to include an additional *reference batch* of examples that are used to calculate weighted conditional scores as targets. We apply self-normalized importance sampling to aggregate the contribution of each example in the reference batch. Although this process can substantially reduce the variance of training targets (Figure 1(b)), especially in the intermediate



(a) vanilla (b) STF

Figure 1: Illustration of differences between the vanilla score-matching objective and our proposed STF objective. The "destroyed" images (in blue box) are close to each other while their sources (in red box) are not. Although the true score in expectation is the weighted average of $\mathbf{v}_i$, the individual training updates of vanilla objective have a high variance, which our STF objective reduces significantly by including a large reference batch (yellow box).

regime, it does introduce some bias. We show that the bias together with the trace-of-covariance of the estimated training targets shrink to zero as we increase the size of the reference batch.

Experimentally, we show that our STF objective greatly improves the stability of converged score-based models on CIFAR-10 and CelebA 64^2, while providing better average performance across random seeds with both ODE and SDE samplers. In particular, STF improves the FID scores (Heusel et al., 2017) from 8.90 ± 4.41 to 5.51 ± 0.06 with the ODE samplers for VE SDE on CIFAR-10 and from 7.56 to 5.34 on CelebA. In contrast to the vanilla objective, the generated samples of models trained by STF do not contain visually noisy ones. STF accelerates the training of score-based models ($3.6\times$ speed-up on CIFAR-10), while obtaining comparable or better FID scores. To the best of our knowledge, STF is the first technique to accelerate the training process of SGMs. We further demonstrate the performance gain when enlarging the reference batch size, validating the negative effect of large variance.

Our contributions are summarized as follows: **(1)** We detail the instability of the current SGMs training objective in a principled and quantitative manner, characterizing a region in the forward process, termed *the intermediate phase*, where the score-learning targets are most variable (Section 3). **(2)** We propose a generalized score-matching objective, *stable target field*, which provides more stable training targets (Section 4). **(3)** We analyze the behavior of the new objective and prove that it is asymptotically unbiased and reduces the trace-of-covariance of the training targets by a factor pertaining to the reference batch size in the intermediate phase under mild conditions (Section 5). **(4)** We illustrate the theoretical arguments empirically and show that the proposed STF objective improves performance, stability, and training speed of the currently state-of-the-art score-based method (Section 6).

2 Background and Related Work

Score-based generative models In SGMs, the forward process[1] is a SDE with no learned parameter, in the form of:

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w},$$

where $\mathbf{x} \in \mathbb{R}^d$ with $\mathbf{x}(0) \sim p_0$ being the data distribution, $t \in [0, 1]$, $\mathbf{f}\colon \mathbb{R}^d \times [0, 1] \to \mathbb{R}^d$, $g\colon [0, 1] \to \mathbb{R}$, and $\mathbf{w} \in \mathbb{R}^d$ is the standard Wiener process. It gradually transforms the data distribution to a known prior as time goes from 0 to 1. Sampling of SGMs is done via a corresponding reverse-time SDE (Anderson, 1982):

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] \mathrm{d}\bar{t} + g(t)\mathrm{d}\bar{\mathbf{w}},$$

[1]For simplicity, we focus on the version where the diffusion coefficient $g(t)$ is independent of $\mathbf{x}(t)$.

where $\bar{\cdot}$ denotes time traveling backwards from 1 to 0. Song et al. (2021) propose a probability flow ODE that induces the same marginal distribution $p_t(\mathbf{x})$ as the SDE:

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] \mathrm{d}\bar{t}.$$

Both formulations progressively recover p_0 from the prior p_1. We estimate the score of the transformed data distribution at time t, $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$, via a neural network, $\mathbf{s}_\theta(\mathbf{x}, t)$. Specifically, the training objective is a weighted sum of the denoising score-matching (Vincent, 2011):

$$\min_\theta \ \mathbb{E}_{t \in \mathcal{U}[0,1]} \lambda(t) \mathbb{E}_{\mathbf{x} \sim p_0} \mathbb{E}_{\mathbf{x}(t) \sim p_{t|0}(\cdot|\mathbf{x})} \left[\|\mathbf{s}_\theta(\mathbf{x}(t), t) - \nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x})\|_2^2 \right], \tag{1}$$

where $\lambda(t)$ is some positive weighting function to keep the time-dependent loss at the same magnitude (Song et al., 2021), and $p_{t|0}(\mathbf{x}(t)|\mathbf{x})$ is the transition kernel denoting the conditional distribution of $\mathbf{x}(t)$ given $\mathbf{x}$. Note that we omit "(0)" from $\mathbf{x}(0)$ when there is no ambiguity. Popular instances of this framework "destroy" data according to a diffusion process utilizing Gaussian transition kernels, which result in $p_{t|0}(\mathbf{x}(t)|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_t, \sigma_t^2 \boldsymbol{I})$. Recent works (Xu et al., 2022b; Rissanen et al., 2022) have also extended the underlying principle from the diffusion process to more general physical processes where the training objective is not necessarily score-related.

Self-normalized importance sampling We revisit the basics of importance sampling and self-normalized importance sampling (Owen, 2013), upon which we draw inspirations for our method. Importance sampling provides a Monte Carlo estimator of the expectation $\mathbb{E}_{x \sim p}[f(x)]$ by sampling from the proposal distribution q, *i.e.*, $\mathbb{E}_{x \sim p}[f(x)] \approx \frac{1}{n} \sum_{i=1}^n p(x_i)/q(x_i) f(x_i)$, $x_i \sim q$. The estimator is effective when sampling from p is infeasible, or the vanilla Monte Carlo estimator incurs high variance (Owen, 2013; Elvira & Martino, 2021). In practice though, the importance weights $w_i = p(x_i)/q(x_i)$ may have a large variance across the Monte Carlo batch. Self-normalized importance sampling (Hesterberg, 1995) remedies the issue by normalizing the importance weights in the batch: $\sum_{i=1}^m (w_i / \sum_j w_j) f(x_i)$. The technique is widely adopted in machine learning community, such as debiasing generative models (Grover et al., 2019), counterfactual learning (Swaminathan & Joachims, 2015) and reinforcement learning (Metelli et al., 2018).

3 Understanding the Training Target in Score-Matching Objective

The vanilla denoising score-matching objective at time t is:

$$\ell_{\mathrm{v}}(\theta, t) = \mathbb{E}_{p_0(\mathbf{x})} \mathbb{E}_{p_{t|0}(\mathbf{x}(t)|\mathbf{x})} [\|\mathbf{s}_\theta(\mathbf{x}(t), t) - \nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x})\|_2^2], \tag{2}$$

where the network is trained to fit the individual targets $\nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x})$ at $(\mathbf{x}(t), t)$ – the "influence" exerted by clean data $\mathbf{x}$ on $\mathbf{x}(t)$. We can swap the order of the sampling process by first sampling $\mathbf{x}(t)$ from p_t and then $\mathbf{x}$ from $p_{0|t}(\cdot|\mathbf{x}(t))$. Thus, $\mathbf{s}_\theta$ has a closed form minimizer:

$$\mathbf{s}_{\mathrm{v}}^*(\mathbf{x}(t), t) = \mathbb{E}_{p_{0|t}(\mathbf{x}|\mathbf{x}(t))} [\nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x})] = \nabla_{\mathbf{x}(t)} \log p_t(\mathbf{x}(t)). \tag{3}$$

The score field is a conditional expectation of $\nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x})$ with respect to the posterior distribution $p_{0|t}$. In practice, a Monte Carlo estimate of this target could have high variance. In particular, when multiple modes of the data distribution have comparable influences on $\mathbf{x}(t)$, $p_{0|t}(\cdot|\mathbf{x}(t))$ is a multi-mode distribution. Thus the targets $\nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x})$ vary considerably across different $\mathbf{x}$ and this can strongly affect the estimated score at $(\mathbf{x}(t), t)$, resulting in slower convergence and worse performance in practical stochastic gradient optimization (Wang et al., 2013).

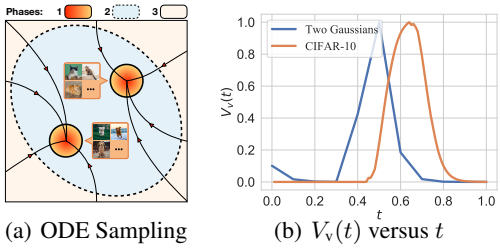
To quantitatively characterize the variations of individual targets at different time, we propose a metric – the average trace-of-covariance of training targets at time t:

$$\begin{aligned} V_{\mathrm{v}}(t) &= \mathbb{E}_{p_t(\mathbf{x}(t))} \left[\mathrm{Tr}(\mathrm{Cov}_{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}(\nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}))) \right] \\ &= \mathbb{E}_{p_t(\mathbf{x}(t))} \mathbb{E}_{p_{0|t}(\mathbf{x}|\mathbf{x}(t))} \left[\|\nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x})) - \nabla_{\mathbf{x}(t)} \log p_t(\mathbf{x}(t))\|_2^2 \right]. \end{aligned} \tag{4}$$

We use $V_{\mathrm{v}}(t)$ to define three successive phases relating to the behavior of training targets. As shown in Figure 2(a), the three phases partition the score field into near, intermediate and far regimes (Phase 1$\sim$3 respectively). Intuitively, $V_{\mathrm{v}}(t)$ peaks in the intermediate phase (Phase 2), where multiple distant

modes in the data distribution have comparable influences on the same noisy perturbations, resulting in unstable targets. In Phase 1, the posterior $p_{0|t}$ concentrates around one single mode, thus reducing the variation. In Phase 3, the targets remain similar across modes since $\lim_{t\to 1} p_{t|0}(\mathbf{x}(t)|\mathbf{x}) \approx p_1$ for commonly used transition kernels.

We validate this argument empirically in Figure 2(b). The figure shows the estimated $V_{\text{v}}(t)$ for a toy example involving a mixture of two Gaussians as well as for a more realistic scenario based on a subset of CIFAR-10 dataset (Krizhevsky et al.). We use Variance Exploding (VE) SDE, *i.e.*, $p_{t|0}(\mathbf{x}(t)|\mathbf{x}) = \mathcal{N}\left(\mathbf{x}(t); \mathbf{x}, \sigma_m^2(\frac{\sigma_M}{\sigma_m})^{2t}\boldsymbol{I}\right)$ for some σ_m and σ_M (Song et al., 2021). $V_{\text{v}}(t)$ exhibits similar phase behavior across t in both toy and realistic cases. $V_{\text{v}}(t)$ reaches its maximum value in the intermediate phase, demonstrating the large variation of individual targets. We defer more details to Appendix B.



(a) ODE Sampling (b) $V_{\text{v}}(t)$ versus t

Figure 2: **(a)**: An illustration of the three phases in a two-mode distribution. **(b)**: Estimated $V_{\text{v}}(t)$ for two distributions. We normalize the maximum value to 1 for illustration purposes.

4 Treating score as a field

The vanilla denoising score-matching approach (Equation 3) can viewed as a Monte Carlo estimator, *i.e.*, $\nabla_{\mathbf{x}(t)} \log p_t(\mathbf{x}(t)) = \mathbb{E}_{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}[\nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x})] \approx \frac{1}{n}\sum_{i=1}^n \nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_i)$ where $\mathbf{x}_i$ are sampled from $p_{0|t}(\cdot|\mathbf{x}(t))$ and $n = 1$. The variance of a Monte Carlo estimator is proportional to $\frac{1}{n}$, so we propose to use a larger batch (n) to counter the high variance problem described in Section 3. Since sampling directly from the posterior $p_{0|t}$ is not practical, we first apply importance sampling together with a large reference batch $\mathcal{B}_L = \{\mathbf{x}_i\}_{i=1}^n \sim p_0^n$. As a result, we get the following approximation:

$$\nabla_{\mathbf{x}(t)} \log p_t(\mathbf{x}(t)) \approx \frac{1}{n}\sum_{i=1}^n \frac{p_{0|t}(\mathbf{x}_i|\mathbf{x}(t))}{p_0(\mathbf{x}_i)} \nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_i).$$

The importance weights can be rewritten as $p_{0|t}(\mathbf{x}|\mathbf{x}(t))/p_0(\mathbf{x}) = p_{t|0}(\mathbf{x}(t)|\mathbf{x})/p_t(\mathbf{x}(t))$. This basic importance sampling estimator has two issues. The weights now involve an unknown normalization factor $p_t(\mathbf{x}(t))$ and the ratio between the prior and posterior distribution can be large in high dimensional spaces. To remedy these problems, we appeal to self-normalization techniques (Owen, 2013) to further stabilize the training targets:

$$\nabla_{\mathbf{x}(t)} \log p_t(\mathbf{x}(t)) \approx \sum_{i=1}^n \frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_i)}{\sum_{j=1}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j)} \nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_i). \tag{5}$$

We term this new training target in Equation 5 as *Stable Target Field (STF)*. In practice, we sample the reference batch $\mathcal{B}_L = \{\mathbf{x}_i\}_{i=1}^n$ from p_0^n and obtain $\mathbf{x}(t)$ by applying the transition kernel to the "first" training data $\mathbf{x}_1$. Taken together, the new STF objective becomes:

$$\ell_{\text{STF}}(\theta, t) = \mathbb{E}_{\{\mathbf{x}_i\}_{i=1}^n \sim p_0^n} \mathbb{E}_{\mathbf{x}(t)\sim p_{t|0}(\cdot|\mathbf{x}_1)} \left[\left\| \mathbf{s}_\theta(\mathbf{x}(t), t) - \sum_{k=1}^n \frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)}{\sum_{j=1}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j)} \nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k) \right\|_2^2\right]. \tag{6}$$

When $n = 1$, the weighted objective reduces to the vanilla score-matching objective (Equation 2). In contrast to the vanilla target, the stable target field incorporates a large reference batch to stabilize training targets. Intuitively, the stable target field assigns larger weights to clean data with higher influence on $\mathbf{x}(t)$, *i.e.*, higher transition probability $p_{t|0}(\mathbf{x}(t)|\mathbf{x})$.

Similar to our analysis in Section 3, we can again swap the sampling process in Equation 6 so that, for a perturbation $\mathbf{x}(t)$, we sample the reference batch $\mathcal{B}_L = \{\mathbf{x}_i\}_{i=1}^n$ from $p_{0|t}(\cdot|\mathbf{x}(t))p_0^{n-1}$, where

the first element involves the posterior, and the rest follow the data distribution. Thus, the minimizer of the new objective (Equation 6) is

$$\mathbf{s}^*_{\text{STF}}(\mathbf{x}(t),t) = \mathbb{E}_{\mathbf{x}_1\sim p_{0|t}(\cdot|\mathbf{x}(t))}\mathbb{E}_{\{\mathbf{x}_i\}_{i=2}^n\sim p_0^{n-1}}\left[\sum_{k=1}^n \frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)}{\sum_j p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j)}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)\right]. \tag{7}$$

Note that although the stable target field significantly reduces the variance, it introduces bias: the minimizer is no longer the true score. Nevertheless, in Section 5, we show that the bias converges to 0 as $n \to \infty$, while reducing the trace of covariance of the training targets by a factor of n when $p_{0|t} \approx p_0$. To aggregate the time-dependent STF objective over t, we uniformly sample the time variable t from $[0,1]$ during training and apply the weighting function $\lambda(t)$. Together, the final training objective for STF is $\mathbb{E}_{t\in\mathcal{U}[0,1]}\left[\lambda(t)\ell_{\text{STF}}(\theta,t)\right]$. We summarize the training process in Algorithm 1. The small batch size $|\mathcal{B}|$ is the same as the normal batch size in vanilla training process.

We further specify the STF objective (Equation 6) with VE SDE, whose transition kernel is $p_{t|0}(\mathbf{x}(t)|\mathbf{x}) \propto \exp(\frac{-\|\mathbf{x}(t)-\mathbf{x}\|^2}{2\sigma_t^2})$ with $\sigma_t = \sigma_m(\frac{\sigma_M}{\sigma_m})^t$. The STF objective for VE SDE at $\mathbf{x}(t)$ is in the following form:

$$\mathbb{E}_{\mathbf{x}_1\sim p_{0|t}(\cdot|\mathbf{x}(t))}\mathbb{E}_{\{\mathbf{x}_i\}_{i=2}^n\sim p_0^{n-1}}\left[\left\|\mathbf{s}_\theta(\mathbf{x}(t),t) - \frac{1}{\sigma_t^2}\sum_{k=1}^n \frac{\exp(-\frac{\|\mathbf{x}(t)-\mathbf{x}_k\|_2^2}{2\sigma_t^2})}{\sum_j \exp(-\frac{\|\mathbf{x}(t)-\mathbf{x}_j\|_2^2}{2\sigma_t^2})}(\mathbf{x}_k - \mathbf{x}(t))\right\|_2^2\right].$$

Algorithm 1 Learning the stable target field

Input: Training iteration T, Initial model $\mathbf{s}_\theta$, dataset $\mathcal{D}$, learning rate η.
for $t = 1 \dots T$ **do**
  Sample a large reference batch $\mathcal{B}_L$ from $\mathcal{D}$, and subsample a small batch $\mathcal{B} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{B}|}$ from $\mathcal{B}_L$
  Uniformly sample the time $\{t_i\}_{i=1}^{|\mathcal{B}|} \sim \mathcal{U}[0,1]^{|\mathcal{B}|}$
  Obtain the batch of perturbed samples $\{\mathbf{x}_i(t_i)\}_{i=1}^{|\mathcal{B}|}$ by applying the transition kernel $p_{t|0}$ on $\mathcal{B}$
  Calculate the stable target field of $\mathcal{B}_L$ for all $\mathbf{x}_i(t_i)$:
    $\mathbf{v}_{\mathcal{B}_L}(\mathbf{x}_i(t_i)) = \sum_{\mathbf{x}\in\mathcal{B}_L} \frac{p_{t_i|0}(\mathbf{x}_i(t_i)|\mathbf{x})}{\sum_{\mathbf{y}\in\mathcal{B}_L} p_{t_i|0}(\mathbf{x}_i(t_i)|\mathbf{y})}\nabla_{\mathbf{x}_i(t_i)}\log p_{t_i|0}(\mathbf{x}_i(t_i)|\mathbf{x})$
  Calculate the loss: $\mathcal{L}(\theta) = \frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\lambda(t_i)\|\mathbf{s}_\theta(\mathbf{x}_i(t_i),t_i) - \mathbf{v}_{\mathcal{B}_L}(\mathbf{x}_i(t_i))\|_2^2$
  Update the model parameter: $\theta = \theta - \eta\nabla\mathcal{L}(\theta)$
end for
return $\mathbf{s}_\theta$

5 Analysis

In this section, we are interested in the theoretical properties of our approach. In particular, we show that the new minimizer $\mathbf{s}^*_{\text{STF}}(\mathbf{x}(t),t)$ (Equation 7) converges to the true score (Section 5.1). Then, we show that the new STF targets reduce the trace of covariance by a factor of the reference batch size in the intermediate phase, with mild conditions (Section 5.2).

5.1 Asymptotic behavior

Although in general $\mathbf{s}^*_{\text{STF}}(\mathbf{x}(t),t) \neq \nabla_{\mathbf{x}(t)}\log p_t(\mathbf{x}(t))$, the bias shrinks toward 0 with a increasing n. In the following theorem we show that the minimizer of STF objective at $(\mathbf{x}(t),t)$, *i.e.*, $\mathbf{s}^*_{\text{STF}}(\mathbf{x}(t),t)$, is asymptotically normal when $n \to \infty$.

Theorem 1. *Suppose* $\forall t \in [0,1], 0 < \sigma_t < \infty$, *then*

$$\sqrt{n}\left(\mathbf{s}^*_{STF}(\mathbf{x}(t),t) - \nabla_{\mathbf{x}(t)}\log p_t(\mathbf{x}(t))\right) \xrightarrow{d} \mathcal{N}\left(\mathbf{0}, \frac{\text{Cov}(\nabla_{\mathbf{x}(t)}p_{t|0}(\mathbf{x}(t)|\mathbf{x}))}{p_t(\mathbf{x}(t))^2}\right) \tag{8}$$

We defer the proof to Appendix A.1. The theorem states that, for commonly used transition kernels, $\mathbf{s}^*_{\text{STF}}(\mathbf{x}(t),t) - \nabla_{\mathbf{x}(t)}\log p_t(\mathbf{x}(t))$ converges to a zero mean normal, and larger reference batch

size (n) will lead to smaller asymptotic variance. As can be seen in Equation 8, when $n \to \infty$, $\mathbf{s}^*_{\text{STF}}(\mathbf{x}(t), t)$ highly concentrates around the true score $\nabla_{\mathbf{x}(t)} \log p_t(\mathbf{x}(t))$.

5.2 Trace of Covariance

We now highlight the small variations of the training targets in the STF objective compared to the vanilla one. As done in Section 3, we study the trace-of-covariance of training targets in STF:

$$V_{\text{STF}}(t) = \mathbb{E}_{p_t(\mathbf{x}(t))}\left[\text{Tr}\left(\text{Cov}_{p_{0|t}(\cdot|\mathbf{x}(t))p_0^{n-1}}\left(\sum_{k=1}^{n} \frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)}{\sum_j p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j)} \nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)\right)\right)\right].$$

In the following theorem we compare V_{STF} with V_{v}. In particular, we can upper bound $V_{\text{STF}}(t)$ by

Theorem 2. *Suppose* $\forall t \in [0, 1], 0 < \sigma_t < \infty$, *then*

$$V_{STF}(t) \leq \frac{1}{n-1}\left(V_v(t) + \frac{\sqrt{3}d}{\sigma_t^2}\sqrt{\mathbb{E}_{p_t(\mathbf{x}(t))} D_f\left(p_0(\mathbf{x}) \parallel p_{0|t}(\mathbf{x}|\mathbf{x}(t))\right)}\right) + O\left(\frac{1}{n^2}\right),$$

where D_f *is an f-divergence with* $f(y) = \begin{cases}(1/y-1)^2 & (y < 1.5)\\ 8y/27 - 1/3 & (y \geq 1.5)\end{cases}$. *Further, when* $n \gg d$ *and* $p_{0|t}(\mathbf{x}|\mathbf{x}(t)) \approx p_0(\mathbf{x})$ *for all* $\mathbf{x}(t)$, $V_{STF}(t) \lessapprox \frac{V_v(t)}{n-1}$.

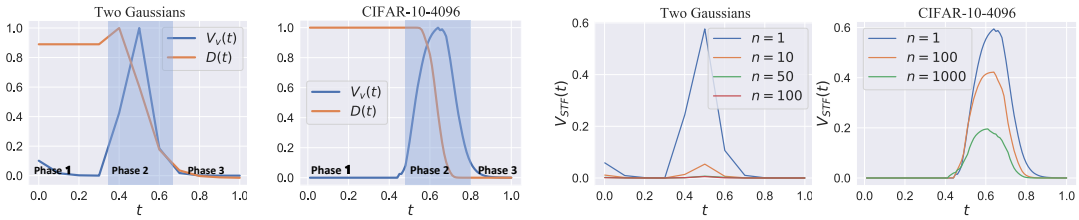
We defer the proof to Appendix A.2. Theorem 2 states that when the posterior is close to the prior, the average trace-of-covariance of the STF targets is reduced by $n-1$ times compared to the vanilla one. Since $p_t \approx p_{t|0}$ with large σ_t for commonly used transition kernels, $p_{0|t}(\cdot|\mathbf{x}(t)) \approx p_0, \forall \mathbf{x}(t)$ with large t. For example, when the data distribution is $\mathcal{N}(\boldsymbol{\mu}, \hat{\sigma}^2\boldsymbol{I}), \hat{\sigma} > 0$, the posterior in VE SDE is $p_{0|t}(\mathbf{x}|\mathbf{x}(t)) = \mathcal{N}(\frac{\mathbf{x}(t)\hat{\sigma}^2 + \boldsymbol{\mu}\sigma_t^2}{\hat{\sigma}^2+\sigma_t^2}, \frac{\hat{\sigma}^2\sigma_t^2}{\hat{\sigma}^2+\sigma_t^2}\boldsymbol{I})$. Obviously, when $\sigma_t \gg \hat{\sigma}$, $D_f(p_{0|t}(\mathbf{x}|\mathbf{x}(t)) \parallel p_0(\mathbf{x})) \approx 0$, and $V_{\text{STF}}(t) \lessapprox V_{\text{v}}(t)/n - 1$.

Together, we demonstrate that the targets have diminishing bias (Theorem 1) and are much more stable during training (Theorem 2). These properties make the STF objective more favorable for learning the score of the data distribution using stochastic gradient optimization.

6 Experiments

In this section, we first empirically validate our theoretical analysis in Section 5, especially for variance reduction in the intermediate phase (Section 6.1). Next, we show that the STF objective improves the score-based models on image generation tasks in terms of the average performance, the stability to random seeds and the robustness to large step sizes (Section 6.2). In addition, we demonstrate that STF accelerates the training of score-based models (Section 6.3), and improves the convergence speed and final performance with an increasing reference batch size (Section 6.4).

6.1 Variance reduction in the intermediate phase



(a) (b) (c) (d)

Figure 3: **(a, b)**: $V_{\text{v}}(t)$ and $D(t)$ versus t. We normalize the the maximum values to 1 for illustration purpose. **(c, d)**: $V_{\text{STF}}(t)$ with a varying reference batch size n.

The proposed algorithm 1 utilizes a large reference batch to calculate the stable target field instead of the individual target. In addition to the theoretical analysis in Section 5, we provide further empirical study to characterize the intermediate phase and verify the variance reduction effects by STF. Apart from $V(t)$, we also quantify the average divergence between the posterior $p_{0|t}(\cdot|\mathbf{x}(t))$ and the data distribution p_0 at time t (introduced in Theorem 2): $D(t) = \mathbb{E}_{p_t(\mathbf{x}(t))}\left[D_f\left(p_{0|t}(\mathbf{x}|\mathbf{x}(t)) \parallel p_0(\mathbf{x})\right)\right]$. Intuitively, the number of high-density modes in $p_{0|t}(\cdot|\mathbf{x}(t))$ grows as $D(t)$ decreases. To investigate their behaviors, we construct two synthetic datasets: (1) A 64-dimensional mixture of two Gaussian components (**Two Gaussians**); (2) A subset of 4096 images of CIFAR-10 (**CIFAR-10-4096**).

Figure 3(a) and Figure 3(b) show the behaviors of $V_{\mathrm{v}}(t)$ and $D(t)$ on Two Gaussian and CIFAR-10-4096. In both settings, $V_{\mathrm{v}}(t)$ reaches its peak in the intermediate phase (Phase 2) , while $D(t)$ gradually decreases over time. These results agree with our theoretical understanding from Section 3. In Phase 2 and 3, several modes of the data distribution have noticeable influences on the scores, but only in Phase 2 the influences are much more distinct, leading to high variations of the individual target $\nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}), \mathbf{x} \sim p_{0|t}(\cdot|\mathbf{x}(t))$.

Figure 3(c) and Figure 3(d) further show the relationship between $V_{\mathrm{STF}}(t)$ and the reference batch size n. Recall that when $n = 1$, STF degenerates to individual target and $V_{\mathrm{STF}}(t) = V_{\mathrm{v}}(t)$. We observe that $V_{\mathrm{STF}}(t)$ decreases when enlarging n. In particular, the predicted relation $V_{\mathrm{STF}}(t) \lessapprox V_{\mathrm{v}}(t)/(n-1)$ in Theorem 2 holds for the two Gaussian datasets where D_f is small. On high dimensional dataset CIFAR-10-4096, the stable target field can still largely reduce the $V_{\mathrm{STF}}(t)$ by increasing the reference batch size.

6.2 Image Generation

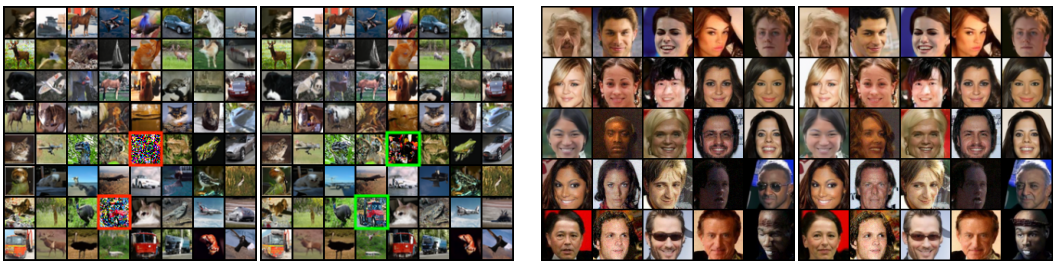
Table 1: CIFAR-10 sample quality (FID, Inception) and number of function evaluation (NFE).

Methods	Inception ↑	FID ↓	NFE ↓
StyleGAN2-ADA (Karras et al., 2020)	9.83	2.92	1
DDPM (Ho et al., 2020)	9.46	3.17	1000
NCSNv2 (Song & Ermon, 2020)	8.40	10.87	1161
RK45 method (ODE)			
VE (vanilla)	9.27 ± 0.12	8.90 ± 4.41	264 ± 52
VE (STF)	$\mathbf{9.52 \pm 0.06}$	$\mathbf{5.51 \pm 0.06}$	200 ± 8
PC sampler (SDE)			
VE (vanilla)	9.68 ± 0.12	2.75 ± 0.13	2000
VE (STF)	$\mathbf{9.86 \pm 0.05}$	$\mathbf{2.66 \pm 0.13}$	2000

We demonstrate the stability of the new objective on image generation tasks. We consider CIFAR-10 (Krizhevsky et al.) and CelebA 64×64 (Yang et al., 2015) datasets. We set the reference batch size n to 4096 (CIFAR10) and 1024 (CelebA 64^2). For comparison, we choose the current state-of-the-art score-based method VE SDE as the baseline, and replace the vanilla objective with our STF objective during training. We directly adopt the NCSN++ architecture and the hyper-parameters such as batch size, learning rate and training iterations in Song et al. (2021). To highlight the stability issue, we train three models with different seeds for all methods on CIFAR-10. We provide more experimental details in Appendix C.1.

Numerical Solver The reverse-time ODE and SDE in scored-based models are compatible with any general purpose solvers. We use the adaptive step size ODE solver in the Scipy library (Virtanen et al., 2020) with the RK45 method (Dormand & Prince, 1980) (**RK45**), unless otherwise specified. For SDEs, we apply the predictor-corrector sampler (Song et al., 2021) (**PC**), with Euler-Maruyama method as the predictor and the Langevin dynamics (MCMC) as the corrector. We defer more details to Appendix C.2.

Results For quantitative evaluation of the generated samples, we report the FID scores (Heusel et al., 2017) (lower is better) and Inception (Salimans et al., 2016) (higher is better). We measure the



(a) vanilla (b) STF (c) vanilla (d) STF

Figure 4: Uncurated samples on CIFAR-10 (a,b) and CelebA 64^2 (c,d) by ODE samplers. We mark the noisy images by the vanilla objective with red boxes and the corresponding images by the STF with green boxes.

Table 2: FID and NFE on CelebA 64^2

Methods/NFEs	FID ↓	NFE ↓
NCSNv2	10.23	2501
RK45 method (ODE)		
VE (vanilla)	7.56	260
VE (STF)	**5.34**	266
PC sampler (SDE)		
VE (vanilla)	9.13	2000
VE (STF)	**8.28**	2000

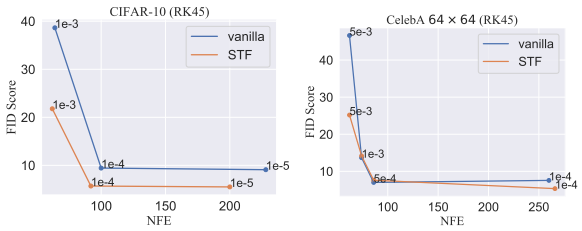


Figure 5: FID versus NFE using ODE samplers on CIFAR-10 (left) and CelebA 64×64 (right)

sampling speed by the average NFE (number of function evaluation). We omit the Inception score on CelebA 64^2 as it is not commonly reported on the dataset. We also include the results of several popular generative models (Karras et al., 2020; Ho et al., 2020; Song & Ermon, 2019) for reference.

Table 1 and Table 2 report the sample quality and the sampling speed on CIFAR-10 and CelebA 64^2. Our main findings are: **(1) The STF objective improves the (average) performance of models with different seeds.** We observe that the STF objective improves the average performance of score-based models trained on different seeds on CIFAR-10, for both ODE and SDE samplers. Remarkably, our objective obtain much better sample quality using ODE samplers, with an average FID/Inception score gain of $3.39/0.25$ on CIFAR-10, and a FID gain of 2.22 on Celeba 64^2. **(2) The STF objective stabilizes the model performance at convergence.** We also report standard deviations of performance for converged models with different seeds on CIFAR-10. We observe that the models trained with the STF objective give more consistent results, with a smaller standard deviation of most metrics. Interestingly, the STF objective provides more improvement using samplers with large step sizes, *e.g.*, RK45 method.

In Figure 5, we show the FID scores with the number of function evaluations of ODE samplers on CIFAR-10 and CelebA 64^2. To vary the NFE, we adjust the error tolerance in the RK45 method. The sample quality of STF objective degrades gracefully when decreasing the NFE. The STF objective consistently outperforms the vanilla one for all NFEs on CIFAR-10, and largely improves over the baseline when setting the tolerance to $5e-3$ on CelebA 64^2. It suggests that the STF has greater robustness to different step sizes.

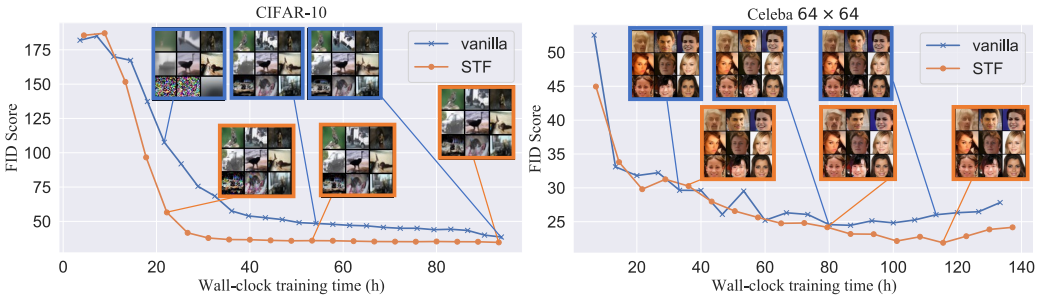
In Figure 4, we visualize the generative samples from the vanilla and STF objective. Notably, the generated samples from the STF objective do not contain noisy images, unlike the vanilla objective (marked by red boxes) on CIFAR-10. We provide more samples in Appendix D.

6.3 Accelerating training of SGMs

The variance-reduction techniques in neural network training can help to find better optima and achieve faster convergence rate (Wang et al., 2013; Defazio et al., 2014; Johnson & Zhang, 2013). In Figure 6, we demonstrate the FID scores computed on 1k samples every 50k iterations during the

course of training. We apply ODE sampler for FID evaluation, and measure the training time on two NVIDIA A100 GPUs. Here we use the forward Euler method with 50 NFE for CelebA 64^2 for faster evaluations. For fair comparison, we report the average FID scores of models trained by the vanilla and STF objective versus the wall-clock training time (h).

The STF objective achieves better FID scores with the same training time, although the calculation of target field by a large reference batch introduces slight overhead (Algorithm 1). In Figure 6(a), we show that the STF objective drastically accelerates the training of score-based models on CIFAR-10. The STF objective achieves comparable FID scores with $3.6\times$ less training time (25h versus 90h). For CelebA 64^2 datasets, the training time improvement is less significant than on CIFAR-10. Our hypothesis is that the STF objective is more effective when there are multiple well-separated modes in data distribution, *e.g.*, the ten classes in CIFAR-10, making the vanilla objective suffer from relatively larger variations in the intermediate phase. In addition, we observe that the converged models have better final performance when pairing with the STF objective on both datasets.



(a) CIFAR-10 (b) CelebA 64×64

Figure 6: FID and generated samples throughout training on (a) CIFAR-10 and (b) CelebA 64^2.

6.4 Effects of the reference batch size

According to our theory (Theorem 2), the trace-of-covariance of the training target in the intermediate phase decreases by a factor of the reference batch size. Here we study the effects of the reference batch size (n) on model performances during training. The FID scores are evaluated on $1k$ samples using ODE sampler. As shown in Figure 7, models converge faster and produce better samples when increasing n. It suggests that smaller variations of the training targets can indeed speed up training and improve the final performances of score-based models.

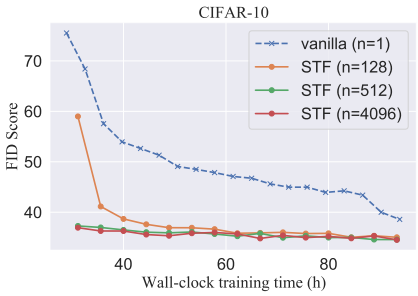


Figure 7: FID scores over the course of training when varying the reference batch size.

7 Conclusion

We highlight the instability of the current SGMs, and argue that the issue is the result of unstable individual targets during training. We identify the three phases with distinct behaviors. In particular, the high-variance targets appear in the intermediate phrase, negatively affecting the model performance. As a remedy, we present a generalized score-matching objective, *Stable Target Field (STF)*, whose formulation is analogous to the self-normalized importance sampling via a large reference batch. Albeit no longer an unbiased estimator, our proposed objective is asymptotically unbiased and reduces the trace-of-covariance of the training targets proportional to the size of the reference batch. We demonstrate the empirical effectiveness of our method on image generation tasks, and show that STF improves the performance, stability and training speed over the state-of-the-art SGMs. Future directions include a principled study on the effect of different reference batch sampling procedures. The approach presented in this paper is simply uniform sampling from the whole dataset $\{\mathbf{x}_i\}_{i=2}^n \sim p_0^{n-1}$, and we expect training SGMs with a reference batch of more samples in the neighborhood of $\mathbf{x}_1$ (the sample from which $\mathbf{x}(t)$ is perturbed) will lead to an even better estimation of the score field.

REFERENCES

Brian DO Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27, 2014.

J. R. Dormand and P. J. Prince. A family of embedded runge-kutta formulae. *Journal of Computational and Applied Mathematics*, 6:19–26, 1980.

Víctor Elvira and Luca Martino. Advances in importance sampling. *Wiley StatsRef: Statistics Reference Online*, 2021.

Aditya Grover, Jiaming Song, Ashish Kapoor, Kenneth Tran, Alekh Agarwal, Eric J Horvitz, and Stefano Ermon. Bias correction of learned generative models using likelihood-free importance weighting. *Advances in neural information processing systems*, 32, 2019.

Tim Hesterberg. Weighted average importance sampling and defensive mixture distributions. *Technometrics*, 37:185–194, 1995.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NIPS*, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26, 2013.

Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *ArXiv*, abs/2006.06676, 2020.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL http://www.cs.toronto.edu/~kriz/cifar.html.

Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2836–2844, 2021.

Alberto Maria Metelli, Matteo Papini, Francesco Faccio, and Marcello Restelli. Policy optimization via importance sampling. In *NeurIPS*, 2018.

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In *ICML*, 2022.

Art B. Owen. *Monte Carlo theory, methods and examples*. 2013.

Severi Rissanen, Markus Heinonen, and A. Solin. Generative modelling with inverse heat dissipation. *ArXiv*, abs/2206.13397, 2022.

Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *ArXiv*, abs/1606.03498, 2016.

Chence Shi, Shitong Luo, Minkai Xu, and Jian Tang. Learning gradient fields for molecular conformation generation. In *ICML*, 2021.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

Adith Swaminathan and Thorsten Joachims. The self-normalized estimator for counterfactual learning. In *NIPS*, 2015.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23:1661–1674, 2011.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, Ilhan Polat, Yu Feng, Eric W. Moore, J. Vanderplas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Daniel Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, Aditya Alessandro Pietro Alex Andreas Andreas Anthony Ant Vijaykumar Bardelli Rothberg Hilboll Kloeckner Sco, Aditya Vijaykumar, Alessandro Pietro Bardelli, Alex Rothberg, Andreas Hilboll, Andre Kloeckner, Anthony M. Scopatz, Antony Lee, Ariel S. Rokem, C. Nathan Woods, Chad Fulton, Charles Masson, Christian Häggström, Clark Fitzgerald, David A. Nicholson, David R. Hagen, Dmitrii V. Pasechnik, Emanuele Olivetti, Eric Martin, Eric Wieser, Fabrice Silva, Felix Lenders, Florian Wilhelm, Gert Young, Gavin A. Price, Gert-Ludwig Ingold, Gregory E. Allen, Gregory R. Lee, Hervé Audren, Irvin Probst, Jorg P. Dietrich, Jacob Silterra, James T. Webber, Janko Slavi, Joel Nothman, Johannes Buchner, Johannes Kulick, Johannes L. Schönberger, José Vinícius de Miranda Cardoso, Joscha Reimer, Joseph E. Harrington, Juan Luis Cano Rodríguez, Juan Nunez-Iglesias, Justin Kuczynski, Kevin Lee Tritz, Martin Dr Thoma, Matt Newville, Matthias Kümmerer, Maximilian Bolingbroke, Michael Tartre, Mikhail Pak, Nathaniel J. Smith, Nikolai Nowaczyk, Nikolay Shebanov, Oleksandr Pavlyk, Per Andreas Brodtkorb, Perry Lee, Robert T. McGibbon, Roman Feldbauer, Sam Lewis, Sam Tygier, Scott Sievert, Sebastiano Vigna, Stefan Peterson, Surhud More, Tadeusz Pudlik, Taku Oshima, Thomas J. Pingel, Thomas P. Robitaille, Thomas Spura, Thouis Raymond Jones, Tim Cera, Tim Leslie, Tiziano Zito, Tom Krauss, U. Upadhyay, Yaroslav O. Halchenko, and Y. Vázquez-Baeza. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature Methods*, 17:261 – 272, 2020.

Chong Wang, X. Chen, Alex Smola, and E. Xing. Variance reduction for stochastic gradient optimization. In *NIPS*, 2013.

Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: a geometric diffusion model for molecular conformation generation. *ArXiv*, abs/2203.02923, 2022a.

Yilun Xu, Ziming Liu, Max Tegmark, and Tommi Jaakkola. Poisson flow generative models. *arXiv preprint arXiv:2209.11178*, 2022b.

Shuo Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. From facial parts responses to face detection: A deep learning approach. *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 3676–3684, 2015.

Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. *ArXiv*, abs/2204.13902, 2022.

Appendix

A PROOFS

A.1 PROOF FOR THEOREM 1

Theorem 1. *Suppose* $\forall t \in [0,1], 0 < \sigma_t < \infty$*, then*

$$\sqrt{n}\left(\mathbf{s}^*_{STF}(\mathbf{x}(t),t) - \nabla_{\mathbf{x}(t)} \log p_t(\mathbf{x}(t))\right) \xrightarrow{d} \mathcal{N}\left(\mathbf{0}, \frac{\mathrm{Cov}(\nabla_{\mathbf{x}(t)} p_{t|0}(\mathbf{x}(t)|\mathbf{x}))}{p_t(\mathbf{x}(t))^2}\right) \tag{8}$$

Proof. Recall that $\mathbf{s}^*_{\text{STF}}(\mathbf{x}(t),t)$ is calculated via Equation 7. The mean of the denominator in the expectation $\frac{1}{n}\sum_{j=1}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j)$, for large n, approximate $\frac{1}{n-1}\sum_{j=2}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j)$, which in turn, $\frac{1}{n-1}\sum_{j=2}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j) \xrightarrow{p} p_t(\mathbf{x}(t))$ by WLLN.

Similarly, for the mean of the remaining terms in the expectation, by CLT, we have

$$\sqrt{n}\left(\frac{1}{n}\sum_{k=1}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)\nabla_{\mathbf{x}(t)} \log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k) - p_t(\mathbf{x}(t))\nabla_{\mathbf{x}(t)} \log p_t(\mathbf{x}(t))\right)$$
$$\xrightarrow{d} \mathcal{N}\left(\mathbf{0},\ \mathrm{Cov}(\nabla_{\mathbf{x}(t)} p_{t|0}(\mathbf{x}(t)|\mathbf{x}))\right)$$

Putting them together via Slutsky's theorem, we conclude the proof. □

A.2 PROOF FOR THEOREM 2

Theorem 2. *Suppose* $\forall t \in [0,1], 0 < \sigma_t < \infty$*, then*

$$V_{STF}(t) \leq \frac{1}{n-1}\left(V_v(t) + \frac{\sqrt{3}d}{\sigma_t^2}\sqrt{\mathbb{E}_{p_t(\mathbf{x}(t))} D_f\left(p_0(\mathbf{x}) \parallel p_{0|t}(\mathbf{x}|\mathbf{x}(t))\right)}\right) + O\left(\frac{1}{n^2}\right),$$

where D_f *is an f-divergence with* $f(y) = \begin{cases}(1/y-1)^2 & (y<1.5)\\ 8y/27 - 1/3 & (y \geq 1.5)\end{cases}$*. Further, when* $n \gg d$ *and* $p_{0|t}(\mathbf{x}|\mathbf{x}(t)) \approx p_0(\mathbf{x})$ *for all* $\mathbf{x}(t)$*,* $V_{STF}(t) \lessapprox \frac{V_v(t)}{n-1}$*.*

Proof. **Step 1: Make the likelihood weighting coefficients "independent"**

We first apply Hoeffding's inequality for the set $\{\mathbf{x}_i\}_{i=2}^n \sim p_0^{n-1}$ to make the summation $\sum_{j=2}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j)$ concentrate to its expectation $(n-1)p_t(\mathbf{x}(t))$. Since $p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j) \in (0, \frac{1}{(\sqrt{2\pi}\sigma_t)^d})$, we have :

$$\Pr\left[\left|\sum_{j=2}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j) - (n-1)p_t(\mathbf{x}(t))\right| \geq n^{\gamma_1}\right] \leq 2e^{\frac{-2n^{2\gamma_1}(\sqrt{2\pi}\sigma_t)^d}{(n-1)}}$$

for $\forall \gamma_1 \in (\frac{1}{2}, 1)$.

Thus the summation can be re-expressed as:

$$\begin{aligned}\sum_{j=2}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j) &= (1 - O(2e^{\frac{-2n^{2\gamma_1}(\sqrt{2\pi}\sigma_t)^d}{(n-1)}}))[(n-1)p_t(\mathbf{x}(t)) + O(n^{\gamma_1})] \\ &\quad + O(2e^{\frac{-2n^{2\gamma_1}(\sqrt{2\pi}\sigma_t)^d}{(n-1)}})O((n-1)\frac{1}{(\sqrt{2\pi}\sigma_t)^d}) \\ &= (n-1)p_t(\mathbf{x}(t)) + O(ne^{-2n^{2\gamma_1-1}(\sqrt{2\pi}\sigma_t)^d})\end{aligned}$$

The coefficient for $\mathbf{x}_1$ is then

$$\begin{aligned}\frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_1)}{\sum_{j=1}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j)} &= \frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_1)}{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_1) + \sum_{j=2}^n p_{t|0}(\mathbf{x}(t)|x_j)} \\ &= \frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_1)}{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_1) + (n-1)p_t(\mathbf{x}(t)) + O(ne^{-2n^{2\gamma_1-1}(\sqrt{2\pi}\sigma_t)^d})} \\ &= O(\frac{1}{n})\end{aligned}$$

The coefficient for $x_k, k \in \{2, \dots, n\}$ is:

$$\begin{aligned}\frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)}{\sum_{j=1}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j)} &= \frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)}{(n-1)p_t(\mathbf{x}(t)) + p_{t|0}(\mathbf{x}(t)|\mathbf{x}_1) + O(ne^{-2n^{2\gamma_1-1}(\sqrt{2\pi}\sigma_t)^d})} \\ &= \frac{1}{(n-1)}\frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)}{p_t(\mathbf{x}(t))} + O(\frac{1}{n^2}) + O(ne^{-2n^{2\gamma_1-1}(\sqrt{2\pi}\sigma_t)^d})\end{aligned}$$

Step 2: Re-express the trace-of-covariance by the "independent" weights

Plugging in the above formulation of coefficients, we can rewrite the trace-of-covariance for the new target as:

$$\begin{aligned}&V_{\text{STF}}(\mathbf{x}(t), t) \\ =& \mathbb{E}_{\mathbf{x}_1 \sim p_{0|t}(\mathbf{x}|\mathbf{x}(t))}\mathbb{E}_{\{\mathbf{x}_i\}_{i=2}^n \sim p^{n-1}(\mathbf{x})}\|\sum_{k=1}^n \frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)}{\sum_{j=1}^n p_{t|0}(\mathbf{x}(t)|x_j)}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k) \\ &\quad - \mathbb{E}_{\mathbf{x}_1 \sim p_{0|t}(\mathbf{x}|\mathbf{x}(t))}\mathbb{E}_{\{\mathbf{x}_i\}_{i=2}^n \sim p^{n-1}(\mathbf{x})}\sum_{k=1}^n \frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)}{\sum_{j=1}^n p_{t|0}(\mathbf{x}(t)|\mathbf{x}_j)}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)\|_2^2 \\ \leq& \mathbb{E}_{\mathbf{x}_1 \sim p_{0|t}(\mathbf{x}|\mathbf{x}(t))}\mathbb{E}_{\{\mathbf{x}_i\}_{i=2}^n \sim p^{n-1}(\mathbf{x})}\|\sum_{k=2}^n \frac{1}{(n-1)}\frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)}{p_t(\mathbf{x}(t))}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k) \\ &\quad - \mathbb{E}_{\mathbf{x}_1 \sim p_{0|t}(\mathbf{x}|\mathbf{x}(t))}\mathbb{E}_{\{\mathbf{x}_i\}_{i=2}^n \sim p^{n-1}(\mathbf{x})}\sum_{k=2}^n \frac{1}{(n-1)}\frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)}{p_t(\mathbf{x}(t))}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k) \\ &\quad + \sum_{k=2}^n O(\frac{1}{n^2})\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k) - O(\frac{1}{n})\mathbb{E}_{p(\mathbf{x})}[\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x})]\|_2^2 + O(\frac{1}{n^2}) \\ =& \frac{1}{(n-1)^2}\sum_{k=2}^n \text{Tr}(\text{Cov}_{\mathbf{x}_k \sim p(\mathbf{x})}(\frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k)}{p_t(\mathbf{x}(t))}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}_k))) + O(\frac{1}{n^2}) \\ =& \frac{1}{(n-1)}\text{Tr}(\text{Cov}_{\mathbf{x} \sim p(\mathbf{x})}(\frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x})}{p_t(\mathbf{x}(t))}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}))) + O(\frac{1}{n^2}) \qquad (9)\end{aligned}$$

Step 3: Upper bound the new trace-of-covariance term

Next we examine the new trace-of-covariance term:

$$
\begin{aligned}
&\mathrm{Tr}(\mathrm{Cov}_{\mathbf{x}\sim p_0(\mathbf{x})}(\frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x})}{p_t(\mathbf{x}(t))}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x})))\\
=&\sum_{i=1}^{d}\mathbb{E}_{p_0(\mathbf{x})}[(\frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x})}{p_t(\mathbf{x}(t))}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}))_i)^2]\\
&\qquad-(\mathbb{E}_{p_0(\mathbf{x})}[\frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x})}{p_t(\mathbf{x}(t))}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}))_i)])^2\\
=&\sum_{i=1}^{d}\mathbb{E}_{p_0(\mathbf{x})}[(\frac{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}{p_0(\mathbf{x})}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}))_i)^2]\\
&\qquad-(\mathbb{E}_{p_0(\mathbf{x})}[\frac{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}{p_0(\mathbf{x}(t))}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}))_i)])^2\\
=&\sum_{i=1}^{d}\mathbb{E}_{p_0(\mathbf{x})}[(\frac{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}{p_0(\mathbf{x})}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}))_i)^2]-(\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x})_i)^2\\
=&\sum_{i=1}^{d}\mathbb{E}_{p_0(\mathbf{x})}[(\frac{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}{p_0(\mathbf{x})}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x}))_i)^2]-(\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x})_i)^2\\
&\qquad+\mathbb{E}_{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}[(\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x})_i)^2]-\mathbb{E}_{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}[(\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x})_i)^2]\\
=&V_{\mathrm{v}}(\mathbf{x}(t),t)+\sum_{i=1}^{d}\mathbb{E}_{p_{0|t}}[(\frac{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}{p(\mathbf{x})}-1)\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x})_i^2]\\
\leq&V_{\mathrm{v}}(\mathbf{x}(t),t)+\sum_{i=1}^{d}\sqrt{\mathbb{E}_{p_{0|t}}[(\frac{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}{p(\mathbf{x})}-1)^2]\mathbb{E}[\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x})_i^4]}\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\text{(Cauchy's inequality)}\\
=&V_{\mathrm{v}}(\mathbf{x}(t),t)+d\sqrt{\mathbb{E}_{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}[(\frac{p_{0|t}(\mathbf{x}|\mathbf{x}(t))}{p(\mathbf{x})}-1)^2]}\sqrt{\mathbb{E}_{z\sim\mathcal{N}(0,\sigma^2)}[\frac{z^4}{\sigma_t^8}]}\\
=&V_{\mathrm{v}}(\mathbf{x}(t),t)+\sqrt{D_f\left(p_0(\mathbf{x})\parallel p_{0|t}(\mathbf{x}|\mathbf{x}(t))\right)}\frac{\sqrt{3}d}{\sigma_t^2}\qquad(f(y)\geq(\frac{1}{y}-1)^2)
\end{aligned}
\tag{10}
$$

where D_f is an f-divergence with $f(y)=\begin{cases}(1/y-1)^2 & (y<1.5)\\ 8y/27-1/3 & (y\geq 1.5)\end{cases}$. Note that we choose this particular form of $f(y)$ since it is the convex function with the tightest upper bound on $(\frac{1}{y}-1)^2$. We can further upper bound the trace-of-covariance term in Equation 9:

$$
\begin{aligned}
V_{\mathrm{STF}}(\mathbf{x}(t),t)&=\frac{1}{(n-1)}\mathrm{Tr}(\mathrm{Cov}_{\mathbf{x}\sim p(\mathbf{x})}(\frac{p_{t|0}(\mathbf{x}(t)|\mathbf{x})}{p_t(\mathbf{x}(t))}\nabla_{\mathbf{x}(t)}\log p_{t|0}(\mathbf{x}(t)|\mathbf{x})))+O(\frac{1}{n^2})\\
&\leq\frac{1}{n-1}\left(V_{\mathrm{v}}(\mathbf{x}(t),t)+\sqrt{D_f\left(p_0(\mathbf{x})\parallel p_{0|t}(\mathbf{x}|\mathbf{x}(t))\right)}\frac{\sqrt{3}d}{\sigma_t^2}\right)+O(\frac{1}{n^2})
\end{aligned}
$$

Taking the expectation w.r.t $p_t(\mathbf{x}(t))$ for both sides, we get

$$
\begin{aligned}
V_{\mathrm{STF}}(t)&\leq\frac{1}{n-1}\left(V_{\mathrm{v}}(\mathbf{x}(t),t)+\sqrt{D_f\left(p_0(\mathbf{x})\parallel p_{0|t}(\mathbf{x}|\mathbf{x}(t))\right)}\frac{\sqrt{3}d}{\sigma_t^2}\right)+O(\frac{1}{n^2})\\
&\leq\frac{1}{n-1}\left(V_{\mathrm{v}}(t)+\frac{\sqrt{3}d}{\sigma_t^2}\sqrt{\mathbb{E}_{p_t(\mathbf{x}(t))}D_f\left(p_0(\mathbf{x})\parallel p_{0|t}(\mathbf{x}|\mathbf{x}(t))\right)}\right)+O(\frac{1}{n^2})\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\text{(Concavity of } x^{\frac{1}{2}})
\end{aligned}
$$

□

B DETAILS FOR THE BEHAVIOR OF $V_{\mathrm{v}}(t)$

In Section 3, we demonstrate the behavior of $V_{\mathrm{v}}(t)$ in the three phases on Two Gaussians and a subset of CIFAR-10. Here we provide more details about the two datasets.

The distribution of the two Gaussian is $\frac{1}{2}\mathcal{N}(\boldsymbol{\mu}, \hat{\sigma}^2\boldsymbol{I}_{64\times 64}) + \frac{1}{2}\mathcal{N}(-\boldsymbol{\mu}, \hat{\sigma}^2\boldsymbol{I}_{64\times 64})$, where $\boldsymbol{\mu} = 0.1 *\mathbf{1} \in \mathbb{R}^{\mathbf{64}}$, and $\hat{\sigma} = 1e-4$. We estimate all the integrals in Equation 4 by sampling 1k points from the corresponding distributions. For the subset of CIFAR-10, we uniformly sample 4096 images from CIFAR-10 dataset, and assign uniform distribution on the discrete set. We also approximate $V_{\mathrm{v}}(t)$ by Monte Carlo estimation and sample 200 perturbations for each t. We use VE SDE for all simulations, and set $\sigma_m = 1e-2, \sigma_M = 50$.

Interestingly, $V_{\mathrm{v}}(t)$ is relatively large for the Two Gaussians distribution compared to CIFAR-10 (see Figure 2(b)) when $t \to 0$. This can be explained by their continuous and discrete natures. For the two Gaussian distribution, we can rewrite $V_{\mathrm{v}}(t)$ as

$$V_{\mathrm{v}}(t) = \mathbb{E}_{\mathbf{x}(t)\sim p_t(\mathbf{x}(t))}\left[\frac{\hat{\sigma}^2}{\sigma_t^2(\sigma_t^2+\hat{\sigma}^2)} + 4\frac{\alpha(\mathbf{x}(t))(1-\alpha(\mathbf{x}(t)))\|\boldsymbol{\mu}\|\sigma_t^4}{\sigma_t^4+\hat{\sigma}^2}\right]$$

where $\alpha(\mathbf{x}) = \frac{1}{1+\exp^{-4\mathbf{x}^T\boldsymbol{\mu}}}$ can be regarded as the probability that $\mathbf{x}$ comes from the Gaussian component $\mathcal{N}(\boldsymbol{\mu}, \hat{\sigma}^2\boldsymbol{I}_{64\times 64})$. When $t \to 0$, obviously $\alpha(\mathbf{x}(t))(1-\alpha(\mathbf{x}(t))) \to 0$, and the term $\frac{\alpha(\mathbf{x}(t))(1-\alpha(\mathbf{x}(t)))\|\boldsymbol{\mu}\|\sigma_t^4}{\sigma_t^4+\hat{\sigma}^2}$ vanishes. Hence $\lim_{t\to 0} V_{\mathrm{v}}(t) \approx \lim_{t\to 0}\frac{\hat{\sigma}^2}{\sigma_t^2(\sigma_t^2+\hat{\sigma}^2)} = \frac{\hat{\sigma}^2}{\sigma_m^2(\sigma_m^2+\hat{\sigma}^2)}$, which can not be neglected when σ_m is small. On the other hand, we can effectively view the 4069 discrete samples as a mixture of 4096 0-variance Gaussians, *i.e.*, $\hat{\sigma} = 0$. Thus by similar reasoning we could see $\lim_{t\to 0} V_{\mathrm{v}}(t) \approx \lim_{t\to 0}\frac{\hat{\sigma}^2}{\sigma_t^2(\sigma_t^2+\hat{\sigma}^2)} = 0$.

C EXPERIMENTAL DETAILS

In this section we include more details about the training and sampling of score-based models by the STF and vanilla objectives. All the experiments are run on two NVIDIA A100 GPUs.

C.1 TRAINING

We consider the CIFAR-10 and CelebA 64^2 in image geneation tasks. Following Song & Ermon (2020), we first center-crop the CelebA images and then resize them to 64×64. We use the same set of hyper-parameters and the NCSN++ backbone and the continuous-time training objectives for forward SDEs in Song et al. (2021). We set the reference batch size n to 4096 on CIFAR-10 and 1024 on CelebA 64^2. The training iteration is 1.3M on CIFAR-10 and 1M on CelebA 64^2. The small batch size $\mathcal{B}$ in Algorithm 1 is the same as the batch size in the baseline score-based method. We repeat the experiment 3 times on CIFAR-10 and 2 times on CelebA 64^2 for vanilla and STF, using different random seeds. For model selection, we pick the checkpoint with smallest FID per 50k iterations on 10k samples for computing all the scores, as in Song et al. (2021).

In Table 3, we report the wall-clock training time (s) per 50 iterations. We can see that the STF introduces additional overhead after incorporating the large reference batch. Since the calculation of the mini-batch target does not involving neural networks, the STF does not take significantly longer training time. Indeed, in Section 6.3 we show that STF achieves comparable or better performance within a shorter training time.

Table 3: Wall-clock training time (s) per 50 iterations

Dataset-Method	CIFAR-10 - vanilla	CIFAR-10 - STF	CelebA - vanilla	CelebA - STF
Wall-clock time	13	16	24	26

C.2 SAMPLING

We adopt the RK45 method and forward Euler method for reverse-time ODE. For RK45 sampler, we use the function implemented in `scipy.integrate.solve_ivp` with the tolerances `atol`=$1e-5/1e-4$, `rtol`=$1e-5/1e-4$ for CIFAR-10/CelebA 64^2. For forward Euler method, we discretize the ODE with constant step size determined by the number of function evaluations. We further adopt the exponential integrator Zhang & Chen (2022) as it empirically gives better results. As in Song et al. (2021), we set the terminal time to $1e-5$.

We use the predictor-corrector (PC) sampler for reverse-time SDE. We follow Song et al. (2021) to set the Euler-Maruyama method as the predictor and the Langevin dynamics (MCMC) as the corrector.

C.3 EVALUATIONS

For the evaluation, we compute the Fréchet distance between 50000 samples and the pre-computed statistics of CIFAR-10 dataset in Heusel et al. (2017). For CelebA 64×64, we adopt the setting in Song & Ermon (2020) where the distance is computed between 10000 samples and the test set.

D EXTENDED SAMPLES

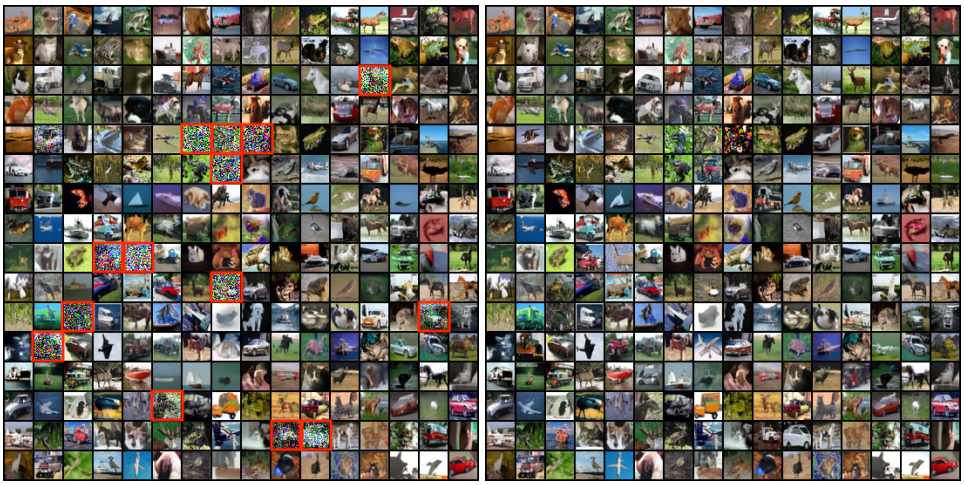
We provide extended more samples from score-based models trained by vanilla/STF objective on CIFAR-10 and CelebA 64^2 by ODE samplers. For systematic comparison, we visualize samples from models trained on different seeds.

D.1 CIFAR-10

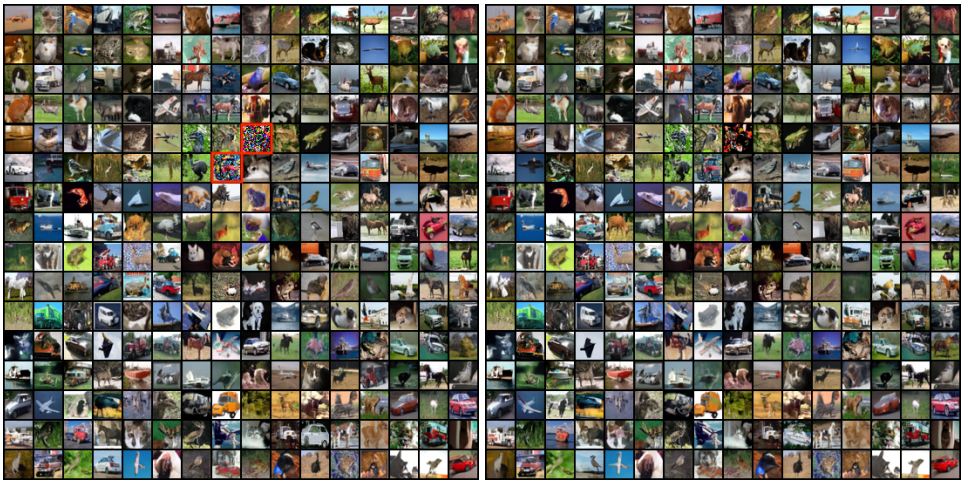
In Figure 8, we visualize the samples produced by different methods across random seeds. We observe that the model trained by the vanilla objective can produce noisy images (in red boxes), and the image quality has great variability across different random seeds. In contrast, model trained by STF objective generate clean and consistent samples with varying random seeds.

D.2 CELEBA 64×64

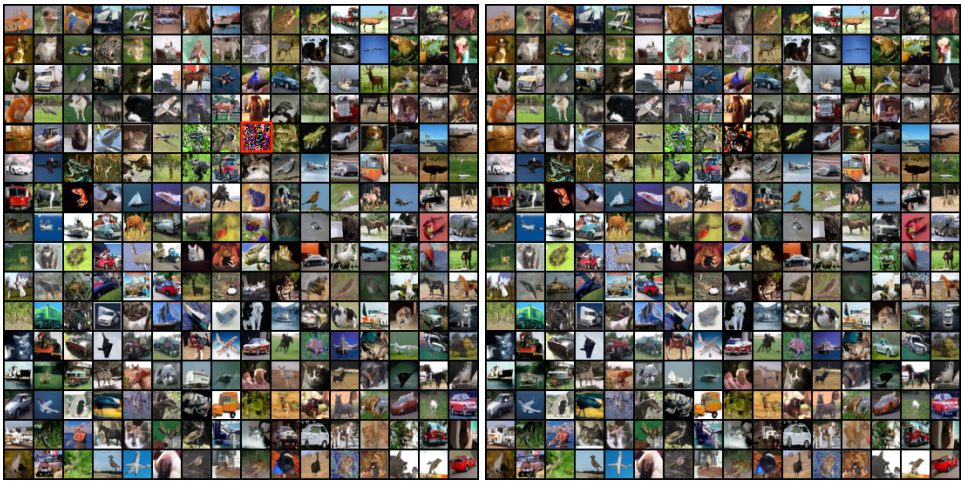
In Figure 9, we further provide samples from models trained on vanilla and STF objectives.



(a) vanilla-1 (b) STF-1



(c) vanilla-2 (d) STF-2



(e) vanilla-3 (f) STF-3

Figure 8: Samples generated by three different final models of vanilla (left column) and STF (right column) on CIFAR-10. Red boxes indicate noisy images.

(a) vanilla

(b) STF

Figure 9: Samples generated by models trained on vanilla (top) and STF (bottom) on CelebA 64^2.