Nearest Neighbour Score Estimators for Diffusion Generative Models

Matthew Niedoba [1 2] **Dylan Green** [1 2] **Saeid Naderiparizi** [1 2] **Vasileios Lioutas** [1 2] **Jonathan Wilder Lavington** [1 2] **Xiaoxuan Liang** [1 2] **Yunpeng Liu** [1 2] **Ke Zhang** [1 2] **Setareh Dabiri** [2] **Adam Ścibior** [1 2] **Berend Zwartsenberg** [2] **Frank Wood** [1 2]

Abstract

Score function estimation is the cornerstone of both training and sampling from diffusion generative models. Despite this fact, the most commonly used estimators are either biased neural network approximations or high variance Monte Carlo estimators based on the conditional score. We introduce a novel nearest neighbour score function estimator which utilizes multiple samples from the training set to dramatically decrease estimator variance. We leverage our low variance estimator in two compelling applications. Training consistency models with our estimator, we report a significant increase in both convergence speed and sample quality. In diffusion models, we show that our estimator can replace a learned network for probability-flow ODE integration, opening promising new avenues of future research.

1. Introduction

Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021) have emerged as a powerful class of generative models. They have seen broad adoption across a variety of tasks such as image generation (Rombach et al., 2022), video generation (Harvey et al., 2022), and 3D object synthesis (Poole et al., 2023). Although these models achieve state of the art performance, their sampling procedure requires integration of the probability flow ODE (PF-ODE) or diffusion SDE (Song et al., 2021). This procedure hampers sampling speed as each integration step requires a neural network evaluation. Typically, many integration steps are required per sample (Ho et al., 2020; Karras et al., 2022).

Motivated by this shortcoming, an array of methods have been proposed which learn few-step approximations of the PF-ODE solution. Amongst these, some attempt to distill a pre-trained diffusion network such as progressive distillation (Salimans & Ho, 2022), consistency distillation (Song et al., 2023) or adversarial diffusion distillation (Sauer et al., 2023). Alternatively, consistency training (Song et al., 2023; Song & Dhariwal, 2024) proposes a method to train consistency models without a pre-trained diffusion model. We refer to the general class of models including diffusion models and related few-step methods as diffusion generative models.

As diffusion generative models are grounded in diffusion processes, they utilize the *score function*. This critical quantity can be estimated by network approximation or Monte Carlo methods. Diffusion and consistency training utilize a one-sample Monte Carlo estimator of the score in their training procedure, (excepting (Xu et al., 2023)). Alternatively, both diffusion sampling and diffusion distillation methods rely upon neural network estimators. However, each estimator has drawbacks. The single-sample Monte Carlo estimator has high variance (Xu et al., 2023) while learned neural network approximators are imperfect, leading to bias (Karras et al., 2022).

Our work puts forward a new method for score function estimation with lower variance than the one-sample estimator and less bias than neural network approximation. Utilizing self-normalized importance sampling (Hesterberg, 1995), we estimate the score through a weighted average over a batch of training set examples. We draw these elements from a proposal which prioritizes examples which are the most likely to have generated the current noisy value (Figure 1). We show that because diffusion processes are Gaussian, these examples are equivalent to the ℓ_2 nearest neighbours to the noisy data. With this finding, we can rapidly identify and sample important examples using a fast KNN search (Johnson et al., 2019). We analyze our method and derive bounds on the variance of our estimator.

Empirically, we measure our performance in three settings. First, we compare our score estimate against the analytic score on CIFAR-10 (Krizhevsky et al., 2009). We find that our method has near-zero variance and bias – substantially outperforming both STF (Xu et al., 2023) and EDM (Karras

[1]Department of Computer Science, University of British Columbia, Vancouver, Canada [2]Inverted AI, Vancouver, Canada. Correspondence to: Matthew Niedoba <mniedoba@cs.ubc.ca>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

et al., 2022). Applying our method to consistency models, we find that replacing one-sample estimators with our method improves consistency training – resulting in faster convergence and higher sample quality. Finally, we show that our method can replace neural networks in PF-ODE integration, opening interesting future research avenues. We release our code[1] to encourage use of our estimator by the community.

2. Background

Diffusion processes form the foundation of both diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) and consistency models (Song et al., 2023). In the variance exploding formulation (Song et al., 2021), diffusion processes convolve a data distribution $p(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^d$ with zero-mean Gaussian noise $\mathcal{N}\left(\mathbf{0}, \sigma(t)^2\mathbf{I}\right)$, resulting in a continuous marginal distribution path $p_t(\mathbf{z}) = \int p_{\text{data}}(\mathbf{x})\,\mathcal{N}\left(\mathbf{z}; \mathbf{x}, \sigma(t)^2\mathbf{I}\right) d\mathbf{x}$, $t \in [t_{\min}, T], \mathbf{z} \in \mathbb{R}^d$. The noise schedule of the diffusion process $\sigma(t)$ and diffusion length T are selected to ensure $p_T(\mathbf{z})$ is indistinguishable from a normal prior $\pi(\mathbf{z}) = \mathcal{N}\left(\mathbf{z}; \mathbf{0}, \sigma(T)^2\mathbf{I}\right)$.

Since the density $p(\mathbf{x})$ is usually unknown, it is approximated by $p_{\text{data}}(\mathbf{x})$, a finite mixture of Dirac measures $p_{data}(\mathbf{x}) = \frac{1}{N}\sum_{\mathbf{x}^{(i)}\in\mathcal{D}} \delta\left(\mathbf{x} - \mathbf{x}^{(i)}\right)$ where $\mathcal{D} = \left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\right\}$ is a dataset of samples drawn from the true data density. With this approximation, the time-varying marginals $p_t(\mathbf{z})$ and posteriors $p_t(\mathbf{x}^{(i)}|\mathbf{z})$ take the forms

$$p_t(\mathbf{z}) = \sum_{i=1}^{N} p_t(\mathbf{z}|\mathbf{x}^{(i)}) p_{\text{data}}(\mathbf{x}^{(i)}) = \sum_{i=1}^{N} \frac{p_t(\mathbf{z}|\mathbf{x}^{(i)})}{N} \quad (1)$$

$$p_t(\mathbf{x}^{(i)}|\mathbf{z}) = \frac{p_t(\mathbf{z}|\mathbf{x}^{(i)}) p_{\text{data}}\left(\mathbf{x}^{(i)}\right)}{p_t(\mathbf{z})} = \frac{p_t(\mathbf{z}|\mathbf{x}^{(i)})}{N \cdot p_t(\mathbf{z})} \quad (2)$$

Under the finite data approximation, each posterior distribution $p_t(\mathbf{x}^{(i)}|\mathbf{z})$ is categorical over the elements of $\mathcal{D}$, ie. $\sum_{i=1}^{N} p(\mathbf{x}^{(i)}|\mathbf{z}) = 1$. In addition, we refer to $p_t(\mathbf{z}|\mathbf{x}^{(i)}) = \mathcal{N}\left(\mathbf{z}; \mathbf{x}, \sigma(t)^2\mathbf{I}\right)$ as the forward likelihood.

Diffusion processes can be described using stochastic or ordinary differential equations (Song et al., 2021). In this work, we focus on the probability flow ODE (PF-ODE) formulation, introduced by Song et al. (2021). Following the parameterization of EDM (Karras et al., 2022), we select $\sigma(t) = t$. However, other choices are possible, as discussed in Appendix B. With our chosen diffusion schedule, the PF-ODE is given by

$$d\mathbf{z} = -t\,\nabla_{\mathbf{z}} \log p_t(\mathbf{z}) dt. \quad (3)$$

Numerical integration of the PF-ODE requires repeated evaluation of $\nabla_{\mathbf{z}} \log p_t(\mathbf{z})$, known as the *score function*. Because the forward likelihood is Gaussian, the *conditional* score has the straightforward form $\nabla_{\mathbf{z}} \log p_t(\mathbf{z}|\mathbf{x}^{(i)}) = \frac{\mathbf{x}^{(i)} - \mathbf{z}}{t^2}$. However, calculating the *marginal* score for fixed $\mathbf{z}$ and t requires an expectation over the posterior $p_t(\mathbf{x}^{(i)}|\mathbf{z})$

$$\nabla_{\mathbf{z}} \log p_t(\mathbf{z}) = \mathop{\mathbb{E}}_{\mathbf{x}^{(i)}\sim p_t(\mathbf{x}^{(i)}|\mathbf{z})}\left[\nabla_{\mathbf{z}} \log p_t(\mathbf{z}|\mathbf{x}^{(i)})\right] \quad (4)$$

$$= \frac{\mathbb{E}[\mathbf{x}|\mathbf{z}, t] - \mathbf{z}}{t^2} \quad (5)$$

where $\mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]$ is the mean of $p_t(\mathbf{x}^{(i)}|\mathbf{z})$.

Since calculating posterior probabilities via Equation (2) involves summing over $\mathcal{D}$ to find $p_t(\mathbf{z})$, exact evaluation of Equation (5) is computationally prohibitive outside of small data regimes. Instead, most methods which require the score function must use estimators of various kinds. In the following sections, we highlight the use of score estimators in diffusion and consistency models.

2.1. Diffusion Models

Diffusion models are a class of generative models which can generate samples through numerical integration of Equation (3) from T to $t_{\min} \approx 0$ with initial value $\mathbf{z} \sim \pi(\mathbf{z})$.

In place of the exact score function discussed previously, diffusion models use learned neural network estimators of the marginal score $\mathbf{s}_\theta(\mathbf{z}, t)$ to generate samples. Following Equation (5), $\mathbf{s}_\theta$ can be parameterized as $\mathbf{s}_\theta(\mathbf{z}, t) = \frac{D_\theta(\mathbf{z}, t) - \mathbf{z}}{t^2}$ where D_θ is a learned estimator of the posterior mean $\mathbb{E}[\mathbf{x}|\mathbf{z}, t]$. Training is performed via stochastic gradient descent on a denoising score matching objective $\mathcal{L}_{DSM} = \mathbb{E}_t\left[\lambda(t)\mathcal{L}_t\right]$ with weights $\lambda(t)$. $\mathcal{L}_t$ is given by

$$\mathcal{L}_t = \mathbb{E}_{\mathbf{z}, \mathbf{x}^{(i)}\sim p_t(\mathbf{z}, \mathbf{x}^{(i)})}\left[\|\mathbf{D}_\theta\left(\mathbf{z}, t\right) - \mathbf{x}^{(i)}\|_2^2\right]. \quad (6)$$

The minimizer of Equation (6) is $D_\theta^\star\left(\mathbf{z}, t\right) = \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]$ (Karras et al., 2022). However, during stochastic gradient descent optimization, per-batch loss is calculated using $(\mathbf{x}^{(i)}, \mathbf{z})$ tuples drawn from the joint distribution instead of against the true posterior mean $\mathbb{E}[\mathbf{x}|\mathbf{z}, t]$. Minimizing the loss with individual $\mathbf{x}^{(i)}$ converges to $\mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]$ because the joint distribution from which the tuples are drawn $p_t(\mathbf{z}, \mathbf{x}^{(i)})$ is proportional to $p_t(\mathbf{x}^{(i)}|\mathbf{z})$. Due to this relation, each $\mathbf{x}^{(i)}$ can be seen as a single-sample Monte Carlo estimator of the posterior mean (Xu et al., 2023).

2.2. Consistency Models

Sampling from diffusion models requires repeated evaluation of the score estimator $s_\theta(\mathbf{z}, t)$. To address this shortcoming, consistency models (Song et al., 2023) learn a one step mapping between $\pi(\mathbf{z})$ and the data distribution.

Equation (3) defines trajectories which map between any marginal distribution $p_t(\mathbf{z})$ and $p_{t_{\min}}(\mathbf{z}) \approx p_{data}(\mathbf{x})$, where

[1] https://github.com/plai-group/knn-score

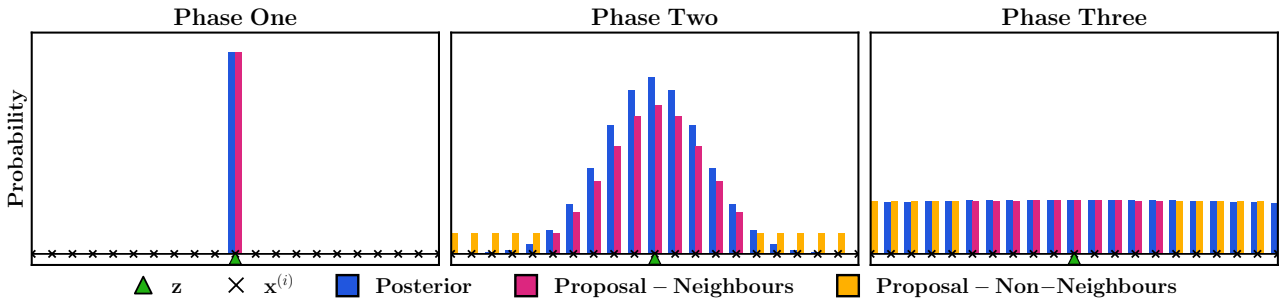

Figure 1: Illustration of our proposal and the posterior across three phases of a toy 1D diffusion process. **Left:** For small t, the posterior probability is concentrated on the single, closest element to $\mathbf{z}$. **Middle:** For intermediate t, we upper bound the posterior probability for non-neighbour elements, resulting in under weighting neighbours. **Right:** As t becomes large, the posterior approaches a uniform distribution and the proposal matches the posterior well.

$t_{\min} \approx 0$ is used for numerical stability. These mappings $\boldsymbol{f} : (\mathbf{z}, t) \to \mathbf{x}$ are known as *consistency functions* because the value $\boldsymbol{f}(\mathbf{z}, t) = \mathbf{x}^\star$ is consistent for all $(\mathbf{z}, t)$ on a common PF-ODE trajectory. Consistency models (Song et al., 2023) exploit this self-consistency property to learn $\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{z}, t)$, a neural network parameterized approximation to the true consistency function $\boldsymbol{f}$. In practice, consistency models are trained by optimizing the consistency matching objective $\mathcal{L}_{CM} = \mathbb{E}_{t_i}\left[\lambda(t_i)\mathcal{L}_{t_i}\right]$ where

$$\mathcal{L}_{t_i} = \mathbb{E}\left[\boldsymbol{d}\left(\boldsymbol{f}_\theta\left(\mathbf{z}, t_i\right), \boldsymbol{f}_{\theta^-}\left(\hat{\mathbf{z}}, t_{i-1}\right)\right)\right]. \tag{7}$$

The consistency matching objective is optimized with respect to θ over N discrete time steps $t_{\min} = t_1 < \ldots < t_N = T$, with an expectation over $\mathbf{x}^{(i)} \sim p_{data}\left(\mathbf{x}\right)$, $i \sim p(i)$ and $\mathbf{z} \sim p_{t_i}(\mathbf{z}|\mathbf{x})$. Here $p(i)$ is a categorical distribution over integers $i \in [2, N]$, $\lambda(t_i)$ is a weighting function, and $d(\mathbf{x}, \mathbf{y})$ is a metric function such as ℓ_2, LPIPS (Zhang et al., 2018), or Pseudo-Huber (Charbonnier et al., 1997).

The loss aims to ensure consistency between the student network $\boldsymbol{f}_\theta$ and the teacher network $\boldsymbol{f}_{\theta^-}$, evaluated at points $(\mathbf{z}, t_i)$ and $(\hat{\mathbf{z}}, t_{i-1})$ which lie on the same PF-ODE trajectory. To obtain $\hat{\mathbf{z}}$, consistency model training performs a one step integration of Equation (3) from from t_i to t_{i-1}, setting $\hat{\mathbf{z}} = \texttt{solver}\left(\mathbf{z}, t_i, t_{i-1}, \nabla_{\mathbf{z}} \log p_{t_i}(\mathbf{z})\right)$. An accurate score function estimate is critical to performing this integration, ensuring $\mathbf{z}$ and $\hat{\mathbf{z}}$ lie on the same PF-ODE path.

Since $\nabla_{\mathbf{z}} \log p_{t_i}(\mathbf{z})$ is infeasible to calculate exactly outside the small data regime, consistency models are trained with score estimators. Song et al. (2023) propose training consistency models with two such estimators. Consistency distillation utilizes a neural network approximator $\nabla_{\mathbf{z}} \log p_{t_i}(\mathbf{z}) \approx s_\theta(\mathbf{z}, t_i)$ from a pre-trained diffusion "teacher" model. To train consistency models without a pre-trained score estimator, consistency training utilizes the estimator $\nabla_{\mathbf{z}} \log p_{t_i}(\mathbf{z}) \approx \frac{\mathbf{x}^{(i)} - \mathbf{z}}{t^2}$ with $\mathbf{x}^{(i)}$ and $\mathbf{z}$ sampled from $p_t(\mathbf{z}, \mathbf{x}^{(i)})$. Like diffusion training, a single-sample Monte Carlo estimate is used in place of the marginal score.

3. Nearest Neighbour Score Estimators

3.1. SNIS Score Estimation

Although the score is necessary for both training and sampling from diffusion generative models, there are only two widely used estimators of the score function – learned networks or single-sample Monte Carlo estimators. Since the latter is required to produce the former, we consider methods to reduce variance in Monte Carlo score estimation.

When calculating the score via Equation (5), the unknown quantity is $\mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]$ which we will refer to as μ hereafter

$$\mu = \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right] = \sum_{i=1}^{N} p_t(\mathbf{x}^{(i)}|\mathbf{z})\, \mathbf{x}^{(i)}. \tag{8}$$

If $p_t(\mathbf{x}^{(i)}|\mathbf{z})$ is available, Monte Carlo estimation can be used to convert the sum over $\mathcal{D}$ in Equation (8) into a sum over a batch $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \sim p_t(\mathbf{x}^{(i)}|\mathbf{z})$ of size n

$$\hat{\mu}_{\text{MC}} = \frac{1}{n} \sum_{\mathbf{x}^{(i)} \in S} \mathbf{x}^{(i)}. \tag{9}$$

The previously discussed single-sample Monte Carlo estimator for diffusion and consistency training is an example of Equation (9) with $n = 1$ and $\mathbf{x}_1 \sim p_t(\mathbf{x}^{(i)}, \mathbf{z}) \propto p_t(\mathbf{x}^{(i)}|\mathbf{z})$. This estimator is unbiased but the variance per-dimension scales with $1/n$ (Owen, 2013). Unfortunately, drawing $n > 1$ samples to reduce estimator variance is challenging because $p_t(\mathbf{x}^{(i)}|\mathbf{z})$ is expensive to compute. The normalizing constant $p_t(\mathbf{z})$ can only be found by summing over $\mathcal{D}$, so naive multi-sample Monte Carlo estimation is impractical.

To increase n and thereby decrease estimator variance, we appeal to importance sampling (IS)(Kahn & Marshall, 1953) taking inspiration from Xu et al. (2023). Instead of sampling from $p_t(\mathbf{x}^{(i)}|\mathbf{z})$, we draw samples from a proposal distribution $q(\mathbf{x}^{(i)})$. We can formulate a new Monte Carlo estimator using a batch of n samples drawn from the pro-

posal $S_q = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \sim q(\mathbf{x}^{(i)})$ as

$$\hat{\mu}_{\text{IS}} = \frac{1}{n} \sum_{\mathbf{x}^{(i)} \in S_q} \frac{p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right) p_{\text{data}}\left(\mathbf{x}^{(i)}\right)}{p_t\left(\mathbf{z}\right) q(\mathbf{x}^{(i)})} \mathbf{x}^{(i)}. \tag{10}$$

Equation (10) no longer requires samples from the posterior and with the finite data approximation, we know $p_{\text{data}}(\mathbf{x}^{(i)}) = 1/N$. However, finding $p_t(\mathbf{z})$ still requires a sum over $\mathcal{D}$. We resolve this issue with self-normalized importance sampling (Hesterberg, 1995). By also estimating $p_t(\mathbf{z})$ with an IS estimator of Equation (1) using the same S_q, we arrive at a self-normalized importance sampling (SNIS) estimator

$$\hat{\mu}_{\text{SNIS}} = \frac{\sum_{\mathbf{x}^{(i)} \in S_q} w_i \mathbf{x}^{(i)}}{\sum_{\mathbf{x}^{(j)} \in S_q} w_j}, \; w_i = \frac{p_t(\mathbf{z}|\mathbf{x}^{(i)})}{q(\mathbf{x}^{(i)})}. \tag{11}$$

The constant factor $p_{\text{data}}(\mathbf{x}^{(i)})$ cancels because it is present in the numerator of Equation (10) and in Equation (1).

Combining Equation (11) with Equation (5) yields a SNIS estimator of the marginal score. Although the SNIS estimator has some bias, by drawing $n > 1$ samples we can greatly reduce the variance of score estimation.

3.2. Nearest Neighbour Proposals

With a defined a framework for reducing score estimator variance, we must next determine a suitable proposal distribution $q(\mathbf{x}^{(i)})$. To identify a good proposal, we start by analyzing the variance of Equation (11) (Owen, 2013). The diagonal of the SNIS score estimator covariance is

$$\begin{aligned} &\text{Diag}\left(\text{Cov}\left(\frac{\hat{\mu}_{\text{SNIS}} - \mathbf{z}}{t^2}\right)\right) \\ &\quad = \frac{1}{nt^4} \sum_{i=1}^{N} \frac{p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)^2}{q\left(\mathbf{x}^{(i)}\right)} \left(\mathbf{x}^{(i)} - \mu\right)^{\circ 2} \end{aligned} \tag{12}$$

Where $^{\circ 2}$ indicates the Hadamard power operator.

Examining Equation (12), we can infer that an appropriate proposal distribution $q(\mathbf{x}^{(i)})$ is important for estimating $\mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]$ with low per-dimension variance. Specifically, high importance ratios $p_t(\mathbf{x}^{(i)}|\mathbf{z})/q(\mathbf{x}^{(i)})$ will increase variance if $(\mathbf{x}^{(i)} - \mu)^{\circ 2}$ is non-zero.

For univariate SNIS estimators, the optimal proposal $q^\star(x)$ for a target distribution $p(x)$ with true mean $\mu = \mathbb{E}_p[x]$ is $q^\star(x) \propto p(x)\,|x - \mu|$ (Kahn & Marshall, 1953). However, $q^\star$ is not directly applicable to our problem because $\mathbf{x}^{(i)}$ is multivariate and the true mean $\mu = \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]$ is unknown.

Although we cannot make immediate use of the univariate optimal proposal, we note that $q^\star$ is proportional to the target distribution $p(x)$. We therefore hypothesize that a suitable choice for our proposal is to match the true posterior $p_t(\mathbf{x}^{(i)}|\mathbf{z})$ as closely as possible. Because $p_t(\mathbf{x}^{(i)}|\mathbf{z})$ varies with t and $\mathbf{z}$, we propose also varying our proposal with these variables, denoting our proposal as $q_t(\mathbf{x}^{(i)}|\mathbf{z})$.

Algorithm 1 Nearest Neighbour Score Estimator

Input: $\mathbf{z}$, t, $\mathcal{D}$, $\mathcal{I}$,k,n
$\mathcal{K}, d \leftarrow \texttt{search}(\mathcal{I}, \mathbf{z}, k)$
$p_t(\mathbf{z}|\mathbf{x}_i) \leftarrow \exp(\frac{-d_i^2}{2t^2}) \; \forall \; \mathbf{x}_i \in \mathcal{K}$
$\mathcal{Z}_q \leftarrow \sum_{\mathbf{x}_i} p_t(\mathbf{z}|\mathbf{x}_i) + (N-k) p_t(\mathbf{z}|\mathbf{x}_k)$
$q_t(\mathbf{x}^{(i)}|\mathbf{z}) \leftarrow \frac{1}{\mathcal{Z}_q} \begin{cases} p_t(\mathbf{z}|\mathbf{x}_i) \; \forall \; \mathbf{x}^{(i)} \in \mathcal{K} \\ p_t(\mathbf{z}|\mathbf{x}_k) \; \forall \; \mathbf{x}^{(i)} \notin \mathcal{K} \end{cases}$
Sample $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \sim q_t(\mathbf{x}^{(i)}|\mathbf{z})$
$w_i \leftarrow p(\mathbf{z}|\mathbf{x}_i)/q_t(\mathbf{x}_i|\mathbf{z})$
$\bar{w}_i \leftarrow \frac{w_i}{\sum_{j=1}^n w_j}$
$\hat{\mu}_{\text{KNN}} \leftarrow \sum_{i=1}^n \bar{w}_i \mathbf{x}_i$
return $\frac{\hat{\mu}_{\text{KNN}} - \mathbf{z}}{t^2}$

Determining the posterior exactly requires summing over $\mathcal{D}$ to compute $p_t(\mathbf{z})$. To more efficiently construct our proposal, we structure $q_t(\mathbf{x}^{(i)}|\mathbf{z})$ to approximate $p_t(\mathbf{x}^{(i)}|\mathbf{z})$ by using only k most probable elements of $p_t(\mathbf{x}^{(i)}|\mathbf{z})$. Let $\mathcal{K}(\mathbf{z}, t) = (\mathbf{x}_1, \ldots \mathbf{x}_k | \mathbf{x}_i \in \mathcal{D}) \; s.t. \; p_t(\mathbf{x}_1|\mathbf{z}) \geq \ldots \geq p_t(\mathbf{x}_k|\mathbf{z})$ be the ordered set of the k most probable dataset elements under the posterior distribution $p_t(\mathbf{x}^{(i)}|\mathbf{z})$. The subscript of each $\mathbf{x}_j \in \mathcal{K}(\mathbf{z}, t)$ indicates its ordering, where $\mathbf{x}_k$ is the element of $\mathcal{K}(\mathbf{z}, t)$ with the lowest probability. For brevity, we refer to $\mathcal{K}(\mathbf{z}, t)$ as $\mathcal{K}$ hereafter and use it to define our nearest neighbours proposal

$$q_t(\mathbf{x}^{(i)}|\,\mathbf{z}) = \frac{1}{\mathcal{Z}_q} \begin{cases} p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right) & \mathbf{x}^{(i)} \in \mathcal{K} \\ p_t\left(\mathbf{z}|\mathbf{x}_k\right) & \mathbf{x}^{(i)} \notin \mathcal{K} \end{cases}. \tag{13}$$

There are several beneficial properties of our nearest neighbours proposal. Most importantly, Equation (13), approximates the posterior using only the elements of $\mathcal{K}$, instead requiring a full summation over $\mathcal{D}$. In addition, since $p_t(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\mathbf{x}, t^2\mathbf{I})$ is Gaussian, and the size of the $\mathcal{D}$ is known, the normalizing constant $\mathcal{Z}_q = \sum_{i=1}^k p_t(\mathbf{z}|\mathbf{x}_i) + (N-k)p_t(\mathbf{z}|\mathbf{x}_k)$ is found easily. Finally, the shape of $q_t(\mathbf{x}^{(i)}|\mathbf{z})$ matches the shape of $p_t(\mathbf{x}^{(i)}|\mathbf{z})$, as demonstrated in Figure 1.

Examining Figure 1, for $\mathbf{x}^{(i)} \in \mathcal{K}$, the proposal is proportional to the posterior with $\mathcal{Z}_q q_t(\mathbf{x}^{(i)}|\mathbf{z}) = p_t(\mathbf{z}) p_t(\mathbf{x}^{(i)}|\mathbf{z})$. In general, $p_t(\mathbf{z}) \leq \mathcal{Z}_q$ with equality in two cases. When $p_t(\mathbf{z}|\mathbf{x}_k) = 0$ or $k = N$, the posterior is fully concentrated on $\mathcal{K}$ and $p_t(\mathbf{z}) = \sum_{i=1}^N p_t(\mathbf{z}|\mathbf{x}^{(i)}) = \sum_{i=1}^k p_t(\mathbf{z}|\mathbf{x}_i)$. In these cases, our nearest neighbour proposal exactly matches the posterior distribution.

When $p_t(\mathbf{z}|\mathbf{x}_k) > 0$ and $k < N$, the likelihood $p_t(\mathbf{z}|\mathbf{x}_k)$ upper bounds the likelihood for all $\mathbf{x}^{(i)} \notin \mathcal{K}$. In this case,

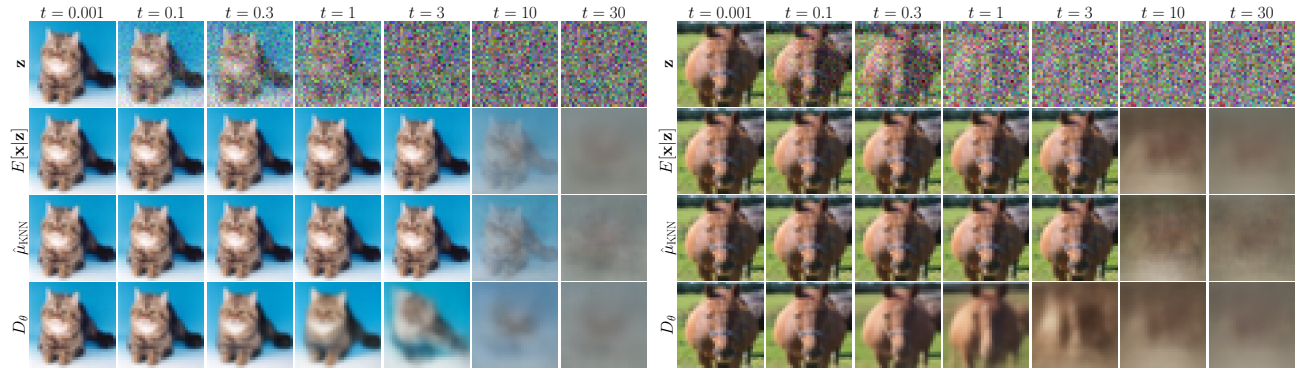

Figure 2: Visualization of posterior mean estimators on CIFAR-10 images. **Top:** Noisy $\mathbf{z}$ samples from $p_t(\mathbf{z}|\mathbf{x}^{(i)})$ for increasing noise levels. **Second Row:** True posterior mean. **Third Row:** Posterior mean estimates from our estimator. Our estimator nearly perfectly matches the true mean levels. **Bottom:** Estimated posterior mean from a trained diffusion model. The diffusion model does not match the high frequency features for low noise levels, but has fewer artifacts at the highest noise level than our method.

we underweight the probability for $\mathbf{x}^{(i)} \in \mathcal{K}$ and assign more mass to the tails of the proposal distribution, as seen in Figure 1b.

Equation (13) requires access to the k most probable elements of the posterior. To quickly identify $\mathcal{K}$, we rely on the Gaussian nature of diffusion processes. Since $p_t(\mathbf{x}^{(i)}|\mathbf{z}) \propto p_t(\mathbf{z}|\mathbf{x}^{(i)})$ and $p_t(\mathbf{z}|\mathbf{x}^{(i)})$ is Gaussian, the k most probable elements of the posterior are equivalent to the k elements of $\mathcal{D}$ with the smallest ℓ_2 distance to $\mathbf{z}$. Therefore, we transform the difficult problem of finding the k most likely elements of the posterior into the easier problem of identifying the k nearest neighbours of $\mathbf{z}$ in Euclidean space.

To determine $\mathcal{K}$ in practice, we utilize FAISS (Johnson et al., 2019) to perform fast k nearest neighbour search over $\mathcal{D}$. To construct $q_t(\mathbf{x}^{(i)}|\mathbf{z})$ for a given $\mathbf{z}$, we first use FAISS to find the nearest neighbours of $\mathbf{z}$. Helpfully, FAISS returns both the nearest neighbour set $\mathcal{K}$ and their ℓ_2 distances $d \in \mathbb{R}^k$ to $\mathbf{z}$. Using d, we compute the forward likelihood for each element of $\mathcal{K}$, and use Equation (13) to compute the proposal. We can then use the proposal to estimate the score via Equation (11). Algorithm 1 outlines our full nearest neighbour score estimation algorithm.

4. Estimator Performance

4.1. Analysis

Motivated by our goal of reducing the variance of the score estimate, we derive two theoretical bounds on the trace of the covariance of $\hat{\mu}_{\text{KNN}}$ – our posterior mean estimator. We form our bounds using the trace of the covariance of a Monte Carlo estimator $\hat{\mu}_{MC}$ and another SNIS estimator with $q_t(\mathbf{x}^{(i)}|\mathbf{z}) = \frac{1}{N}$ which we denote with $\hat{\mu}_U$. We note that $\hat{\mu}_U$ is is only asymptotically equivalent to STF (Xu et al., 2023) because in addition to a uniform $q_t(\mathbf{x}^{(i)}|\mathbf{z})$, STF also deterministically includes the source $\mathbf{x}^{(i)}$ in their sample batch S.

Theorem 4.1. *Let $t \in (0, \infty)$, $\hat{\mu}_{MC}$ be the Monte Carlo estimator defined by Equation (9), and $\hat{\mu}_{KNN}$ be the estimator described by Equation (11) with proposal given by Equation (13). Then for a fixed $n \geq 1$,*

$$Tr\left(Cov\left(\hat{\mu}_{KNN}\right)\right) \leq \frac{\mathcal{Z}_q}{p_t(\mathbf{z})} Tr\left(Cov\left(\hat{\mu}_{MC}\right)\right). \tag{14}$$

Proof. We defer the proof to Appendix C.1. □

In general, our estimator's trace-of-covariance is larger than the simple Monte Carlo estimator's trace-of-covariance. The ratio of the two is upper bounded by the factor $\frac{\mathcal{Z}_q}{p_t(\mathbf{z})}$ – the ratio of the normalizing constants of the proposal and the posterior distributions. This ratio will be exactly unity when $p_t(\mathbf{x}^{(i)}|\mathbf{z})$ is fully concentrated on $\mathcal{K}$ (as is true when t is small or $k = N$). The ratio will also approach unity as $p_t(\mathbf{x}^{(i)}|\mathbf{z})$ approaches a uniform distribution over $\mathcal{D}$. In any of these cases, the trace-of-covariance of our estimator will approach the trace-of-covariance of multi-sample posterior Monte Carlo.

Theorem 4.2. *Let $\hat{\mu}_{MC}$ be defined by Equation (9). Let $\hat{\mu}_{KNN}$ and $\hat{\mu}_U$ be two estimators defined by Equation (11) with proposals given by Equation (13) and $q_t(\mathbf{x}^{(i)}|\mathbf{z}) = \frac{1}{N}$ respectively. Then, for $t \in (0, \infty)$ and a fixed $n \geq 1$,*

$$\begin{aligned} Tr\left(Cov\left(\hat{\mu}_{KNN}\right)\right) \leq & \\ & \left(1 - \frac{\sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z})}\right) Tr\left(Cov\left(\hat{\mu}_{MC}\right)\right) \\ & + \frac{(N-k)}{N} Tr\left(Cov\left(\hat{\mu}_U\right)\right). \end{aligned}$$

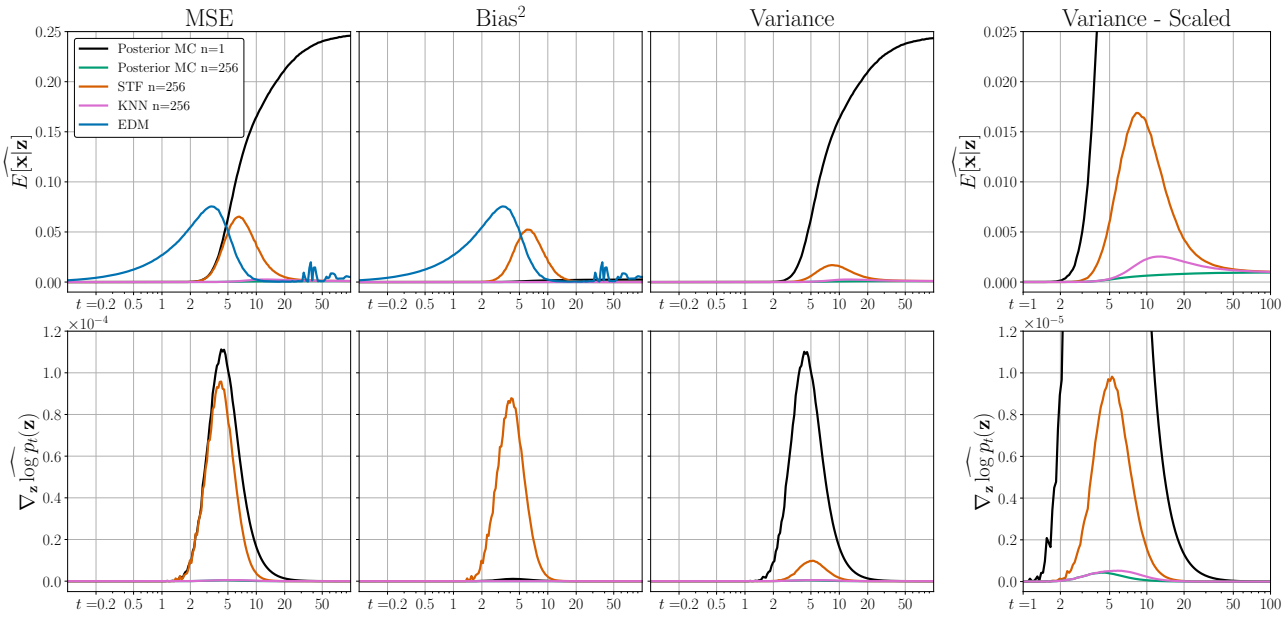

Figure 3: Estimator performance on CIFAR-10. Our estimator reduces bias and variance to near zero, significantly outperforming even a network score estimator. In contrast, STF reduces variance but has significant bias for intermediate t.

Proof. We defer the proof to Appendix C.3. □

Our estimator's trace-of-covariance is upper bounded by the sum of a fixed multiple of the uniform SNIS estimator's trace-of-covariance and a time-varying multiple of the posterior Monte Carlo trace-of-covariance. We note that when $p_t(\mathbf{x}^{(i)}|\mathbf{z}) = \frac{1}{N}\ \forall\ \mathbf{x}^{(i)} \in \mathcal{D}$, then $\mathrm{Tr}\left(\mathrm{Cov}\left(\hat{\mu}_{\mathrm{MC}}\right)\right) = \mathrm{Tr}\left(\mathrm{Cov}\left(\hat{\mu}_{\mathrm{U}}\right)\right)$, the frst term coefficient becomes $\frac{k}{N}$, and the trace-of-covariance of $\hat{\mu}_{\mathrm{KNN}}$ and $\hat{\mu}_{\mathrm{U}}$ are equal. For intermediate t we expect the trace-of-covariance of $\hat{\mu}_{\mathrm{KNN}}$ to be lower than that of $\hat{\mu}_{\mathrm{U}}$ because we expect $\mathrm{Tr}(\mathrm{Cov}(\hat{\mu}_{\mathrm{MC}})) << \mathrm{Tr}(\mathrm{Cov}(\hat{\mu}_{\mathrm{U}}))$ in this region.

4.2. Empirical Performance

To gain insight into the performance of our estimator beyond our theoretical analysis, we empirically compare our estimator against various other score estimators on the CIFAR-10 dataset (Krizhevsky et al., 2009). We measure estimator performance with average mean squared error

$$\mathrm{MSE}\left(\hat{\mu}\right) = \frac{1}{d}\,\mathbb{E}\left\|\hat{\mu} - \mu\right\|_2^2 = \frac{1}{d}\left(\mathrm{Bias}\left(\hat{\mu}\right)^2 + \mathrm{Var}\left(\hat{\mu}\right)\right). \tag{15}$$

Equation (15) compares the estimated posterior mean $\hat{\mu}$ against the true posterior mean μ, calculated via Equation (8). MSE is equivalent to the trace-of-covariance metric introduced in (Xu et al., 2023) and discussed in Section 4.1 scaled by $1/d$. The outer expectation of Equation (15) is calculated over a large set of $\mathbf{x}^{(i)} \sim \mathcal{D}$, with one $\mathbf{z} \sim p_t(\mathbf{z}|\mathbf{x}^{(i)})$ for each $\mathbf{x}^{(i)}$. For each $\mathbf{z}$, we average over repeated estimator evaluations. This allows us to decompose MSE into squared bias and variance terms to further investigate estimator performance. When calculating variance and squared bias, we compute the sample mean from the repeated estimator evaluations per-$\mathbf{z}$.

In Figure 3, we plot score and posterior mean estimator performance across a range of noise levels. At each t, we evaluate the estimators using ten thousand $\mathbf{z}$ samples, and one hundred estimator evaluations per $\mathbf{z}$. We compare our method against four others: the single-sample posterior estimator typically used in diffusion and consistency training, a $n = 256$ multi-sample posterior Monte Carlo estimator, an STF (Xu et al., 2023) estimator with $n = 256$, and EDM – a pre-trained near-SoTA diffusion model (Karras et al., 2022). As EDM is deterministic, we do not report its variance. Further, we do not plot its score metrics since errors in the posterior mean estimation for small t are amplified in the score by a factor of $1/t^4$. For the multi-sample Monte Carlo estimator, we calculate the posterior by calculating Equation (1).

Examining Figure 3, we note several important results. Firstly, our estimator outperforms the STF estimator in both posterior mean and score estimation across every metric. The difference is especially stark for score estimation, where the peak MSE of our estimator is approximately 100 times better than that of STF. We hypothesize that our excellent performance is because for an intermediate range of t, $q_t(\mathbf{x}^{(i)}|\mathbf{z})$ matches the posterior nearly

perfectly. When $p_t(\mathbf{x}^{(i)}|\mathbf{z})$ is fully concentrated on a single element, the variance of the SNIS estimators is zero because $p_t(\mathbf{x}^{(i)}|\mathbf{z}) = 1 \implies (\mathbf{x}^{(i)} - \mathbb{E}[\mathbf{x}|\mathbf{z}, t])^2 = 0$ and $(\mathbf{x}^{(i)} - \mathbb{E}[\mathbf{x}|\mathbf{z}, t])^2 > 0 \implies p_t(\mathbf{x}^{(i)}|\mathbf{z}) = 0$. However, as t increases, the number of elements of $\mathcal{D}$ with non-zero $p_t(\mathbf{x}^{(i)}|\mathbf{z})$ also increases, resulting in a non-zero estimator variance. In these cases, we believe the variance is dominated by the importance ratio p_t/q_t. Compared to STF, our proposal has lower importance ratios when the posterior mass is concentrated on less than k elements.

Our second finding is that the STF estimator has a significant bias for intermediate t. We believe this bias is because STF deterministically includes the generating $\mathbf{x}^{(i)}$ in its sample batch, but does not account for this when computing the SNIS weights. Although Xu et al. (2023) prove that as $n \to \infty$, the bias of STF converges to 0, we see that the bias of the estimator is not negligible for a practical choice of n.

Finally, we find that our estimator significantly outperforms even a near-SoTA diffusion model for most t. Surprisingly, we find that peak MSE for the diffusion model does not coincide with the Monte Carlo methods, and instead occurs at substantially lower t than the peak for both SNIS estimators.

We further compare our method and EDM qualitatively in Figure 2. We find our method better estimates the posterior mean for most t. However, we find some evidence of artifacts at high t which may be mitigated by increasing n.

5. Consistency Training

5.1. Consistency Models

We use our estimator to train consistency models on CIFAR-10 (Krizhevsky et al., 2009). Specifically, we replace the single-sample score estimator used to produce $\hat{\mathbf{z}}$ in Equation (7) with SNIS score estimators. We use the improved consistency training (iCT) techniques proposed by Song & Dhariwal (2024), comparing a baseline iCT model with models trained with an STF (Xu et al., 2023) and a KNN score estimator. For both SNIS score estimators, we use $n = 256$, with a $k = 2056$ for the KNN proposal. We measure performance using Frechet Inception Distance (Heusel et al., 2017) and Inception Score (Salimans et al., 2016).

From Table 1, we observe consistency training with KNN score estimators results in lower FID and higher Inception Score compared to our re-implementation of iCT (Song & Dhariwal, 2024). While we are unable to reproduce the results of (Song & Dhariwal, 2024) outright (training code was not available at the time of writing) we do improve over our re-implementation of iCT. Surprisingly, we find that using the STF score estimator (Xu et al., 2023) results in worse FID than both our method and the iCT baseline. We hypothesize STF's bias may contribute to this degradation.

Table 1: Single step unconditional image generation performance on CIFAR-10. Bold indicates the best result within a section, underline indicates the best result overall.

Method	FID↓	IS↑
CD (Song et al., 2023)	3.55	9.48
CT (Song et al., 2023)	8.70	8.49
iCT (Song & Dhariwal, 2024)	**<u>2.83</u>**	**<u>9.54</u>**
iCT (Reimplemented)	3.67	9.45
iCT + STF (Xu et al., 2023)	3.79	**9.50**
iCT + KNN (ours)	**3.59**	9.49

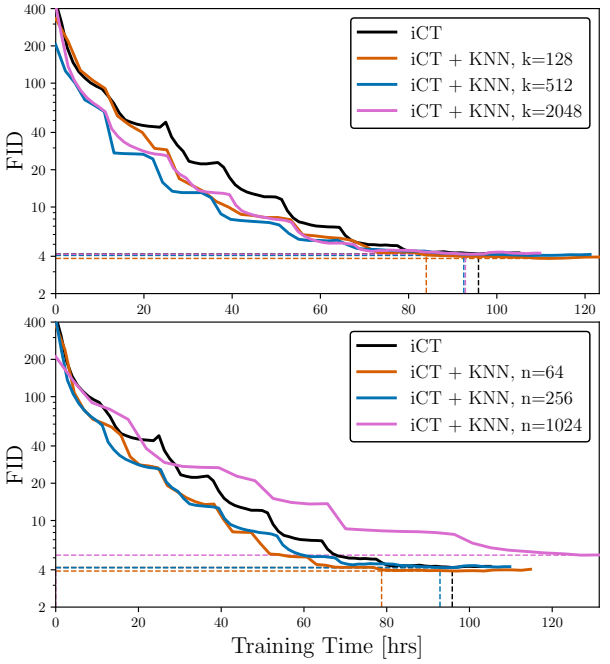

Figure 4: Impact of KNN score estimator performance on Consistency Training. Horizontal lines indicate minimum FID per model, vertical lines indicate when FID improves on baseline iCT. **Top:** Effect of varying KNN search size. **Bottom:** Effect of varying estimator sample size.

5.2. Ablations

We sweep over k and n to investigate their impact on downstream model performance. For our ablation, we train consistency models with the same number of iterations but halve the batch size to reduce computational cost.

Figure 4 highlights that for most hyperparameter choices, our method improves the convergence rate of consistency training. KNN models generally reach lower FID in less time than the iCT baseline. However, the training time required to achieve optimal FID is similar across all methods. Notably, we find reducing n and k improves sample FID, even though this increases estimator variance (Ap-

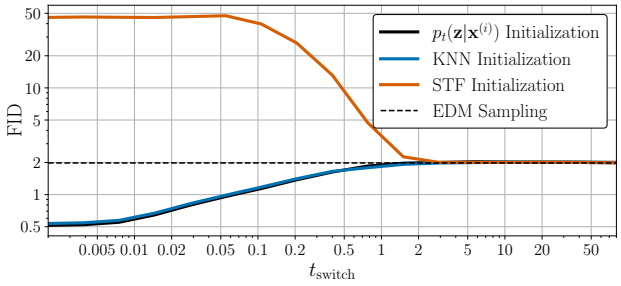

Figure 5: Comparison of sample FID versus t_{switch} in our hybrid sampling approach. Initializing diffusion sampling with KNN PF-ODE integration ith our score estimator yields identical performance to forward process initialization. For $t < 2$, STF is unsuitable for PF-ODE integration.

pendix D.1). We suspect score variance may have a regularizing effect, and suggest future work investigate further hyperparameter tuning to achieve optimal performance.

6. Diffusion Sampling

Motivated by the low error of our estimator compared to network approximators (Figure 3) we investigate the effect of replacing a diffusion model with our estimator for diffusion sampling. To this end, we generate samples via PF-ODE integration using a hybrid sampling approach. For $T < t < t_{\text{switch}}$, we integrate Equation (3) with our score estimate using the Huen solver proposed by Karras et al. (2022). After t_{switch}, we finish sampling with a pre-trained EDM score estimator.

Figure 5 plots the quality of the resulting hybrid samples against t_{switch}. We compare our estimator, STF, and a baseline which initializes EDM sampling using samples from the forward likelihood. Naively, our estimator is able to drive FID to an unprecedented minimum of 0.5. However, this performance is matched by forward process initialization, indicating that integrating the PF-ODE with our estimator is equivalent to drawing samples from the training dataset. This means our method cannot be used on its own as a generative model, as it does not produce generalization.

As shown by Section 5 and by Xu et al. (2023), training diffusion generative models with improved score estimates does not degrade generalization. However, when sampling using an estimator which closely matches the *exact minimizer* of the diffusion training objective (Equation (6)), we see that generalization disappears. This reinforces the findings of (Yi et al., 2023): that the lauded generalization properties of diffusion models can be attributed to bias in the network score estimator caused by architecture, optimization procedure or other factors.

Although integrating the PF-ODE using our estimator does not produce generalization, we still closely match the forward process initialization, indicating that our method is capable of general PF-ODE integration. This finding suggests that using our estimator, it may be possible to convert distillation methods (Salimans & Ho, 2022; Sauer et al., 2023; Lu et al., 2023) into from-scratch training procedures, opening promising avenues of future research.

In contrast, when $t_{\text{switch}} < 2$, the FID of STF increases, indicating it is unsuitable for general purpose PF-ODE integration. We hypothesize that this increase is because for small t, the STF estimator's performance is reliant on including $\mathbf{x}^{(i)}$ used to generate $\mathbf{z}$ in the SNIS batch. For general PF-ODE integration where there is no such $\mathbf{x}^{(i)}$, the uniform proposal of STF is unable to produce a suitable score estimate.

7. Related Work

Generative Modelling with Diffusion Score Estimators Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021) have become a widely utilized class of generative models which use learned score estimators to generate data across a variety of fields. Learned score estimators are also used to train downstream models with distillation (Salimans & Ho, 2022), consistency matching (Song et al., 2023; Kim et al., 2023), or to regularize the training of generative adversarial models (Sauer et al., 2023; Lu et al., 2023).

Importance Sampling for Generative Model Training Multiple methods have used importance sampling to reduce variance in generative model training. Reweighted Wake-Sleep (Bornschein & Bengio, 2015) uses importance sampling to improve variance in the sleep-phase of the wake-sleep algorithm (Hinton et al., 1995). IWAE (Burda et al., 2016) uses importance sampling to tighten the ELBO used to train variational autoencoders (Kingma & Welling, 2014). Importance sampling has also been utilized to minimize bias in diffusion training (Kim et al., 2024). Similar to this work, Stable Target Fields (Xu et al., 2023) utilizes an SNIS estimator to estimate the marginal score of diffusion processes. However, their work uses a uniform proposal and is focused on improving diffusion models training instead of generally estimating the score function.

Retrieval Augmented Methods Many recent methods have proposed using memory systems to improve generative models. In natural language processing, methods propose combining language model outputs (Khandelwal et al., 2020; 2021) or to augment transformer contexts with results from a similarity search based over an external database (Guu et al., 2020; Borgeaud et al., 2022). In computer vision, RetrieveGAN (Tseng et al., 2020) conditions generated images on patches queried using text from a dataset while Retrieval-

Fuse (Siddiqui et al., 2021) searches over a database of 3D structures to augment scene reconstruction. In diffusion models, KNN-Diffusion (Sheynin et al., 2023) and Retrieval-Augmented Diffusion Models (Blattmann et al., 2022) both condition diffusion models on the k nearest CLIP (Radford et al., 2021) embeddings of clean training images. Although our method also uses a KNN search for diffusion generative models, our approach is orthogonal to these methods as we do not condition our network on the retrieved images.

8. Conclusions

Our work introduces a nearest neighbour estimator of the score function for diffusion generative models. We analytically bound the variance of our estimator, and show empirically that it has low bias and variance compared to other estimation techniques. Training consistency models, we find that our estimator both increases training speed and sample quality. We highlight how our estimator can be used for general purpose probability flow ODE traversal. We believe our work opens intriguing opportunities for future research. One possible research direction is substituting our estimator for pre-trained diffusion networks in distillation approaches, to produces new from-scratch training methods. Another area of work is exploring alternate metric spaces. The ℓ_2 distance which forms the basis of our method does not capture high level similarity between images well. By transforming our data into a latent space as described by Rombach et al. (2022), we may be able leverage more informative neighbours for score estimation. Finally, we believe more research is warrented to investigate the cause of observed differences between peak score network estimator error and the peak error of SNIS estimators.

Impact Statement

Diffusion models require large amounts of computation and energy to train. Arguably, generating from such models is even more energy intensive. Consistency models address this inference energy problem by limiting the number of integration steps required to generate data. Directly training consistency models, rather than learning from a pre-trained diffusion model further lessons the energy burden. Our work directly addresses all these energy problems by making it possible to train better consistency models more energy efficiently, without first training a diffusion model. Powerful generative AI models are dual-use technology. We generally believe that democratizing access to the technology by lowering energy requirements for training and inference helps rather than harms.

Acknowledgements

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC), the Canada CIFAR AI Chairs Program, Inverted AI, MITACS, the Department of Energy through Lawrence Berkeley National Laboratory, and Google. This research was enabled in part by technical support and computational resources provided by the Digital Research Alliance of Canada Compute Canada (alliancecan.ca), the Advanced Research Computing at the University of British Columbia (arc.ubc.ca), Amazon, and Oracle.

References

Blattmann, A., Rombach, R., Oktay, K., Müller, J., and Ommer, B. Retrieval-augmented diffusion models. *Advances in Neural Information Processing Systems*, 35: 15309–15324, 2022.

Borgeaud, S., Mensch, A., Hoffmann, J., Cai, T., Rutherford, E., Millican, K., van den Driessche, G., Lespiau, J., Damoc, B., Clark, A., de Las Casas, D., Guy, A., Menick, J., Ring, R., Hennigan, T., Huang, S., Maggiore, L., Jones, C., Cassirer, A., Brock, A., Paganini, M., Irving, G., Vinyals, O., Osindero, S., Simonyan, K., Rae, J. W., Elsen, E., and Sifre, L. Improving language models by retrieving from trillions of tokens. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvári, C., Niu, G., and Sabato, S. (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 2206–2240. PMLR, 2022.

Bornschein, J. and Bengio, Y. Reweighted wake-sleep. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Burda, Y., Grosse, R. B., and Salakhutdinov, R. Importance weighted autoencoders. In Bengio, Y. and LeCun, Y. (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

Charbonnier, P., Blanc-Féraud, L., Aubert, G., and Barlaud, M. Deterministic edge-preserving regularization in computed imaging. *IEEE Trans. Image Process.*, 6(2): 298–311, 1997.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Guu, K., Lee, K., Tung, Z., Pasupat, P., and Chang, M. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3929–3938. PMLR, 2020.

Harvey, W., Naderiparizi, S., Masrani, V., Weilbach, C., and Wood, F. Flexible diffusion modeling of long videos. *Advances in Neural Information Processing Systems*, 35: 27953–27965, 2022.

Hesterberg, T. Weighted average importance sampling and defensive mixture distributions. *Technometrics*, 37(2): 185–194, 1995.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 6626–6637, 2017.

Hinton, G. E., Dayan, P., Frey, B. J., and Neal, R. M. The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7 (3):535–547, 2019.

Kahn, H. and Marshall, A. W. Methods of reducing sample size in monte carlo computations. *Journal of the Operations Research Society of America*, 1(5):263–278, 1953.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Khandelwal, U., Levy, O., Jurafsky, D., Zettlemoyer, L., and Lewis, M. Generalization through memorization: Nearest neighbor language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020.

Khandelwal, U., Fan, A., Jurafsky, D., Zettlemoyer, L., and Lewis, M. Nearest neighbor machine translation. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

Kim, D., Lai, C.-H., Liao, W.-H., Murata, N., Takida, Y., Uesaka, T., He, Y., Mitsufuji, Y., and Ermon, S. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. *arXiv preprint arXiv:2310.02279*, 2023.

Kim, Y., Na, B., Park, M., Jang, J., Kim, D., Kang, W., and Moon, I.-C. Training unbiased diffusion models from biased dataset. *arXiv preprint arXiv:2403.01189*, 2024.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In Bengio, Y. and LeCun, Y. (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Lu, H., Lu, Y., Jiang, D., Szabados, S. R., Sun, S., and Yu, Y. Cm-gan: Stabilizing gan training with consistency models. In *ICML 2023 Workshop on Structured Probabilistic Inference & Generative Modeling*, 2023.

Obukhov, A., Seitzer, M., Wu, P.-W., Zhydenko, S., Kyl, J., and Lin, E. Y.-J. High-fidelity performance metrics for generative models in pytorch, 2020. URL https://github.com/toshas/torch-fidelity. Version: 0.3.0, DOI: 10.5281/zenodo.4957738.

Owen, A. B. *Monte Carlo theory, methods and examples*. https://artowen.su.domains/mc/, 2013.

Poole, B., Jain, A., Barron, J. T., and Mildenhall, B. Dreamfusion: Text-to-3d using 2d diffusion. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 10674–10685. IEEE, 2022.

Salimans, T. and Ho, J. Progressive distillation for fast sampling of diffusion models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

Salimans, T., Goodfellow, I. J., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. In Lee, D. D., Sugiyama, M., von Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 2226–2234, 2016.

Sauer, A., Lorenz, D., Blattmann, A., and Rombach, R. Adversarial diffusion distillation. *arXiv preprint arXiv:2311.17042*, 2023.

Sheynin, S., Ashual, O., Polyak, A., Singer, U., Gafni, O., Nachmani, E., and Taigman, Y. knn-diffusion: Image generation via large-scale retrieval. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023.

Siddiqui, Y., Thies, J., Ma, F., Shan, Q., Nießner, M., and Dai, A. Retrievalfuse: Neural 3d scene reconstruction with a database. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 12548–12557. IEEE, 2021.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Song, Y. and Dhariwal, P. Improved techniques for training consistency models. In *The Twelfth International Conference on Learning Representations*, 2024.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. Consistency models. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252. PMLR, 2023.

Tseng, H., Lee, H., Jiang, L., Yang, M., and Yang, W. Retrievegan: Image synthesis via differentiable patch retrieval. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J. (eds.), *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part VIII*, volume 12353 of *Lecture Notes in Computer Science*, pp. 242–257. Springer, 2020.

Xu, Y., Tong, S., and Jaakkola, T. S. Stable target field for reduced variance score estimation in diffusion models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023.

Yi, M., Sun, J., and Li, Z. On the generalization of diffusion model. *arXiv preprint arXiv:2305.14712*, 2023.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Appendices

A. Derivations

A.1. Marginal Score as a Posterior Expectation of Conditional Scores

Below, we derive Equation (23).

$$\nabla_{\mathbf{z}} \log p_t(\mathbf{z}) = \frac{\nabla_{\mathbf{z}} p_t(\mathbf{z})}{p_t(\mathbf{z})} \tag{16}$$

$$= \frac{\nabla_{\mathbf{z}} \int p_t(\mathbf{z}|\mathbf{x}) p(\mathbf{x}) d\mathbf{x}}{p_t(\mathbf{z})} \tag{17}$$

$$= \frac{\int \nabla_{\mathbf{z}} p_t(\mathbf{z}|\mathbf{x}) p(\mathbf{x}) d\mathbf{x}}{p_t(\mathbf{z})} \tag{18}$$

$$= \frac{\int \frac{p_t(\mathbf{z}|\mathbf{x})}{p_t(\mathbf{z}|\mathbf{x})} \nabla_{\mathbf{z}} p_t(\mathbf{z}|\mathbf{x}) p(\mathbf{x}) d\mathbf{x}}{p_t(\mathbf{z})} \tag{19}$$

$$= \frac{\int p_t(\mathbf{z}|\mathbf{x}) \nabla_{\mathbf{z}} \log p_t(\mathbf{z}|\mathbf{x}) p(\mathbf{x}) d\mathbf{x}}{p_t(\mathbf{z})} \tag{20}$$

$$= \int \frac{p_t(\mathbf{z}|\mathbf{x}) p(\mathbf{x})}{p_t(\mathbf{z})} \nabla_{\mathbf{z}} \log p_t(\mathbf{z}|\mathbf{x}) d\mathbf{x} \tag{21}$$

$$= \int p_t(\mathbf{x}|\mathbf{z}) \nabla_{\mathbf{z}} \log p_t(\mathbf{z}|\mathbf{x}) d\mathbf{x} \tag{22}$$

$$= \mathop{\mathbb{E}}_{\mathbf{x} \sim p_t(\mathbf{x}|\mathbf{z})} \left[\nabla_{\mathbf{z}} \log p_t(\mathbf{z}|\mathbf{x}) \right]. \tag{23}$$

Equation (18) is due to the Leibniz integral rule, while Equation (22) is an application of Bayes Rule.

A.2. Conditional Score

We define $p_t(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \mathbf{x}, t^2\mathbf{I})$. Then the conditional score of $p_t(\mathbf{z}|\mathbf{x})$ is

$$\nabla_{\mathbf{z}} \log p_t(\mathbf{z}|\mathbf{x}) = \frac{\nabla_{\mathbf{z}} p_t(\mathbf{z}|\mathbf{x})}{p_t(\mathbf{z}|\mathbf{x})} \tag{24}$$

$$= \frac{1}{p_t(\mathbf{z}|\mathbf{x})} \cdot \nabla_{\mathbf{z}} \frac{1}{\sqrt{(2\pi)^d \det(t^2\mathbf{I})}} \exp\left(\frac{-1}{2t^2} (\mathbf{z} - \mathbf{x})^\top (\mathbf{z} - \mathbf{x}) \right) \tag{25}$$

$$= \frac{p_t(\mathbf{z}|\mathbf{x})}{p_t(\mathbf{z}|\mathbf{x})} \nabla_{\mathbf{z}} \frac{-1}{2t^2} (\mathbf{z} - \mathbf{x})^\top (\mathbf{z} - \mathbf{x}) \tag{26}$$

$$= \frac{\mathbf{x} - \mathbf{z}}{t^2}. \tag{27}$$

A.3. Marginal Score via Posterior Mean

Substituting Equation (27) into Equation (23) we get

$$\nabla_{\mathbf{z}} \log p_t(\mathbf{z}) = \mathop{\mathbb{E}}_{\mathbf{x} \sim p_t(\mathbf{x}|\mathbf{z})} \left[\frac{\mathbf{x} - \mathbf{z}}{t^2} \right]$$

$$= \frac{\mathbb{E}_{\mathbf{x} \sim p_t(\mathbf{x}|\mathbf{z})} [\mathbf{x}] - \mathbf{z}}{t^2}$$

$$= \frac{\mathbb{E}[\mathbf{x}|\mathbf{z}] - \mathbf{z}}{t^2},$$

where $\mathbb{E}[\mathbf{x}|\mathbf{z}] = \mathbb{E}_{\mathbf{x}\sim p_t(\mathbf{x}|\mathbf{z})}[\mathbf{x}]$.

A.4. Score Matching Optimal Denoiser

We consider the diffusion objective from Equation (6),

$$\mathcal{L}_t = \mathbb{E}_{\mathbf{z},\mathbf{x}^{(i)}\sim p_t(\mathbf{z},\mathbf{x}^{(i)})}\left[\|\mathbf{D}_\theta(\mathbf{z},t) - \nabla_{\mathbf{z}}\log p_t(\mathbf{z}|\mathbf{x})\|_2^2\right] \tag{28}$$

$$= \mathbb{E}_{\mathbf{z},\mathbf{x}\sim p_t(\mathbf{z},\mathbf{x})}\left[\mathbf{D}_\theta(\mathbf{z},t)^2 - 2\mathbf{D}_\theta(\mathbf{z},t)\mathbf{x}^{(i)} + (\mathbf{x}^{(i)})^2\right]. \tag{29}$$

$$\tag{30}$$

For a fixed $\mathbf{z}$, we can write

$$\mathcal{L}_{t,\mathbf{z}} = \mathbb{E}_{\mathbf{x}^{(i)}\sim p_t(\mathbf{x}^{(i)}|\mathbf{z})}\left[\mathbf{D}_\theta(\mathbf{z},t)^2 - 2\mathbf{D}_\theta(\mathbf{z},t)\mathbf{x}^{(i)} + (\mathbf{x}^{(i)})^2\right] \tag{31}$$

$$= \mathbf{D}_\theta(\mathbf{z},t)^2 - 2\mathbf{D}_\theta(\mathbf{z},t)\mathbb{E}_{\mathbf{x}^{(i)}\sim p_t(\mathbf{x}^{(i)}|\mathbf{z})}\left[\mathbf{x}^{(i)}\right] + \mathbb{E}_{\mathbf{x}^{(i)}\sim p_t(\mathbf{x}^{(i)}|\mathbf{z})}\left[(\mathbf{x}^{(i)})^2\right]. \tag{32}$$

Equation (32) is quadratic with respect to $\mathbf{D}_\theta(\mathbf{z},t)$, so a unique minimizer $\mathbf{D}^\star_\theta(\mathbf{z},t)$ can be found by solving $\frac{\partial\mathcal{L}}{\partial\mathbf{D}_\theta(\mathbf{z},t)} = \mathbf{0}$.

$$\frac{\partial\mathcal{L}_{t,z}}{\partial\mathbf{D}_\theta(\mathbf{z},t)} = \mathbf{0} = 2\mathbf{D}_\theta(\mathbf{z},t) - 2\mathbb{E}_{\mathbf{x}^{(i)}\sim p_t(\mathbf{x}^{(i)}|\mathbf{z})}\left[\mathbf{x}^{(i)}\right] \tag{33}$$

$$\mathbf{D}^\star_\theta(\mathbf{z},t) = \mathbb{E}_{\mathbf{x}^{(i)}\sim p_t(\mathbf{x}^{(i)}|\mathbf{z})}\left[\mathbf{x}^{(i)}\right] \tag{34}$$

$$= \mathbb{E}\left[\mathbf{x}|\mathbf{z},t\right]. \tag{35}$$

A.5. Consistency Training as Consistency Matching with Euler Solver and One Sample Score Estimator

The original Consistency Training objective from (Song et al., 2023) is formulated as

$$\mathcal{L}^N_{CT} = \mathbb{E}\left[\lambda(t_n)d\left(\boldsymbol{f}_\theta\left(\mathbf{x}+t_n\cdot\boldsymbol{\epsilon},t\right),\boldsymbol{f}_{\theta^-}\left(\mathbf{x}+t_{n-1}\cdot\boldsymbol{\epsilon},t_{n-1}\right)\right)\right], \tag{36}$$

where $\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})$. Via the reparameterization trick, we define $\mathbf{z}\sim p_{t_n}(\mathbf{z}|\mathbf{x})$ as

$$\mathbf{z} = \mathbf{x} + t_n\boldsymbol{\epsilon}. \tag{37}$$

Substituting Equation (37) into the Euler update function $\texttt{solver}(\mathbf{z},t_n,t_{n-1},\nabla_{\mathbf{z}}p_{t_n}(\mathbf{z},t_n)) = \mathbf{z} + (t_{n-1}-t_n)\frac{d\mathbf{z}}{dt}$ for Equation (3) with the single sample score estimate $\nabla_{\mathbf{z}}\log p_t(\mathbf{z}) \approx \frac{\mathbf{x}-\mathbf{z}}{t^2}$ yields

$$\mathbf{z}' = \mathbf{z} - (t_{n-1}-t_n)\frac{\mathbf{x}-\mathbf{z}}{t_n} \tag{38}$$

$$= \mathbf{x} + t_n\boldsymbol{\epsilon} - (t_{n-1}-t_n)\frac{\mathbf{x}-\mathbf{x}+t_n\epsilon}{t_n} \tag{39}$$

$$= \mathbf{x} + t_n\boldsymbol{\epsilon} - t_n\boldsymbol{\epsilon} + t_{n-1}\boldsymbol{\epsilon} \tag{40}$$

$$= \mathbf{x}t_{n-1}\boldsymbol{\epsilon}. \tag{41}$$

Substituting Equation (37) and Equation (41) into Equation (36) yields

$$\mathcal{L}^N_{CT} = \mathbb{E}\left[\lambda(t_n)d\left(\boldsymbol{f}_\theta\left(\mathbf{z},t\right),\boldsymbol{f}_{\theta^-}\left(\mathbf{z}',t_{n-1}\right)\right)\right] \tag{42}$$

which matches the form in Equation (7).

B. Extension to Generalized Diffusion Processes

For simplicity, this paper utilizes the EDM diffusion process given by Equation (3). However, other diffusion processes are widely used which vary the rate at which noise is added to the data and the scaling of the data through the diffusion process. Karras et al. (2022) introduce a general purpose PF ODE which captures these choices

$$d\mathbf{x} = \left[\frac{\dot{s}(t)}{s(t)}\mathbf{x} - s(t)^2\dot{\sigma}(t)\sigma(t)\nabla_{\mathbf{z}}\log p_t\left(\frac{\mathbf{z}}{s(t)}\right)\right]dt. \tag{43}$$

In our work, we select $\sigma(t) = t$, $s(t) = 1$, however other choices are possible. For example, one popular choice is the variance preserving diffusion process (Ho et al., 2020; Song et al., 2021), corresponding to

$$\sigma(t) = \sqrt{e^{\frac{1}{2}\beta_d t^2 + \beta_{min} t} - 1} \tag{44}$$

$$s(t) = 1/\sqrt{e^{\frac{1}{2}\beta_d t^2 + \beta_{min} t}}. \tag{45}$$

The derivations of $s(t)$ and $\sigma(t)$ is covered in depth in (Karras et al., 2022). We now derive a general KNN score estimator for a generic diffusion processes defined by $s(t)$ and $\sigma(t)$. First, we define the forward likelihood for the generalized diffusion process.

$$p_t(\mathbf{z}|\mathbf{x}^{(i)}) = \mathcal{N}\left(\mathbf{z}; s(t)\mathbf{x}^{(i)}, s(t)^2\sigma(t)^2\mathbf{I}\right). \tag{46}$$

Since the score function which is used for evaluation of Equation (43) is a function of $\mathbf{z}/s(t)$, we express Equation (46) in that manner

$$p_t(\mathbf{z}|\mathbf{x}^{(i)}) = \mathcal{N}\left(\mathbf{z}; s(t)\mathbf{x}^{(i)}, s(t)^2\sigma(t)^2\mathbf{I}\right) \tag{47}$$

$$= (2\pi)^{\frac{-d}{2}} s(t)^{-d}\sigma(t)^{-d}\exp\left(\frac{-\left\|\mathbf{z} - s(t)\mathbf{x}^{(i)}\right\|_2^2}{2s(t)^2\sigma(t)^2}\right) \tag{48}$$

$$= (2\pi)^{\frac{-d}{2}} s(t)^{-d}\sigma(t)^{-d}\exp\left(\frac{-s(t)^2\left\|\frac{\mathbf{z}}{s(t)} - \mathbf{x}^{(i)}\right\|_2^2}{2s(t)^2\sigma(t)^2}\right) \tag{49}$$

$$= s(t)^{-d}\cdot(2\pi)^{\frac{-d}{2}}\sigma(t)^{-d}\exp\left(\frac{-\left\|\frac{\mathbf{z}}{s(t)} - \mathbf{x}^{(i)}\right\|_2^2}{2\sigma(t)^2}\right) \tag{50}$$

$$= s(t)^{-d}\mathcal{N}\left(\frac{\mathbf{z}}{s(t)}; \mathbf{x}^{(i)}, \sigma(t)^2\boldsymbol{I}\right). \tag{51}$$

We define

$$p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right) = \mathcal{N}\left(\frac{\mathbf{z}}{s(t)}; \mathbf{x}^{(i)}, \sigma(t)^2\boldsymbol{I}\right). \tag{52}$$

The score of Equation (52) is

$$\nabla_{\mathbf{z}}\log p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right) = p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right)^{-1}\nabla_{\mathbf{z}}p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right) \tag{53}$$

$$= p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right)^{-1}\cdot p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right)\cdot\nabla_{\mathbf{z}}\frac{-\left\|\frac{\mathbf{z}}{s(t)} - \mathbf{x}^{(i)}\right\|_2^2}{2\sigma(t)^2} \tag{54}$$

$$= \frac{\mathbf{x}^{(i)} - \frac{\mathbf{z}}{s(t)}}{s(t)\sigma(t)^2}. \tag{55}$$

Using Equation (4) derived in Appendix A.1, we write

$$\nabla_{\mathbf{z}} \log p_t\left(\frac{\mathbf{z}}{\sigma(t)}\right) = \mathop{\mathbb{E}}_{\mathbf{x}^{(i)} \sim p_t(\mathbf{x}^{(i)}|\mathbf{z}/s(t))}\left[\nabla_{\mathbf{z}} \log p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right)\right] \tag{56}$$

$$= \mathop{\mathbb{E}}_{\mathbf{x}^{(i)} \sim p_t(\mathbf{x}^{(i)}|\mathbf{z}/s(t))}\left[\frac{\mathbf{x}^{(i)} - \frac{\mathbf{z}}{s(t)}}{s(t)\sigma(t)^2}\right] \tag{57}$$

$$= \frac{\mathbb{E}\left[\mathbf{x}\middle|\frac{\mathbf{z}}{s(t)}, t\right] - \frac{\mathbf{z}}{s(t)}}{s(t)\sigma(t)^2}. \tag{58}$$

Therefore, estimating the score function in Equation (43) is equivalent to estimating the posterior mean $\mathbb{E}\left[\mathbf{x}\middle|\frac{\mathbf{z}}{s(t)}, t\right]$. The posterior distribution $p_t\left(\mathbf{x}^{(i)}\middle|\frac{\mathbf{z}}{s(t)}\right)$ can be expressed using Bayes Rule as

$$p_t\left(\mathbf{x}^{(i)}\middle|\frac{\mathbf{z}}{s(t)}\right) = \frac{p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right) p\left(\mathbf{x}^{(i)}\right)}{p_t\left(\frac{\mathbf{z}}{s(t)}\right)} \tag{59}$$

$$= \frac{p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right)}{N p_t\left(\frac{\mathbf{z}}{s(t)}\right)}. \tag{60}$$

We define $\mathcal{K}'$ to be the k most probable elements of $p_t\left(\mathbf{x}^{(i)}\middle|\frac{\mathbf{z}}{s(t)}\right)$ let $\mathbf{x}_k$ be the element of $\mathcal{K}'$ with the smallest posterior probability. Then, using $\mathcal{K}'$, we define a general proposal

$$\mathcal{Z}_q \cdot q_t\left(\mathbf{x}^{(i)}\middle|\frac{\mathbf{z}}{s(t)}\right) = \begin{cases} p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right) \ \forall\, \mathbf{x}^{(i)} \in \mathcal{K}' \\ p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}_k\right) \ \forall\, \mathbf{x}^{(i)} \notin \mathcal{K}'. \end{cases} \tag{61}$$

Because $p_t\left(\mathbf{x}^{(i)}\middle|\frac{\mathbf{z}}{s(t)}\right) \propto p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right)$ and $p_t\left(\frac{\mathbf{z}}{s(t)}\middle|\mathbf{x}^{(i)}\right)$ is Gaussian, the most probable elements of $p_t\left(\mathbf{x}^{(i)}\middle|\frac{\mathbf{z}}{s(t)}\right)$ are the elements with the smallest distance $\left\|\frac{\mathbf{z}}{s(t)} - \mathbf{x}^{(i)}\right\|_2^2$. We can therefore use a fast KNN search with query $\frac{\mathbf{z}}{s(t)}$ over $\mathcal{D}$ to identify $\mathcal{K}'$. Together, an algorithm for estimating the score of a generic diffusion process is outlined in Algorithm 2

Algorithm 2 General Nearest Neighbour Score Estimator

Input: $\mathbf{z}$, dataset $\mathcal{D}$, index $\mathcal{I}$, neighbour size k, sample size n, scale $s(t)$, noise level $\sigma(t)$
$\mathcal{K}', d \leftarrow \texttt{search}(I, \mathbf{z}/s(t), k)$
$p_t(\frac{\mathbf{z}}{s(t)}|\mathbf{x}_i) \leftarrow \exp(\frac{-d_i^2}{2\sigma(t)^2}) \ \forall\, \mathbf{x}_i \in \mathcal{K}'$
$\mathcal{Z}_q \leftarrow \sum_{i=1}^k p_t(\frac{\mathbf{z}}{s(t)}|\mathbf{x}_i) + (N-k)p_t(\frac{\mathbf{z}}{s(t)}|\mathbf{x}_k)$
$q_t(\mathbf{x}^{(i)}|\frac{\mathbf{z}}{s(t)}) \leftarrow \frac{1}{\mathcal{Z}_q}\begin{cases} p_t(\frac{\mathbf{z}}{s(t)}|\mathbf{x}_i) \ \forall\, \mathbf{x}^{(i)} \in \mathcal{K}' \\ p_t(\frac{\mathbf{z}}{s(t)}|\mathbf{x}_k) \ \forall\, \mathbf{x}^{(i)} \notin \mathcal{K}' \end{cases}$
Sample $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \sim q_t(\mathbf{x}^{(i)}|\frac{\mathbf{z}}{s(t)})$
$w_i = p(\frac{\mathbf{z}}{s(t)}|\mathbf{x}_i)/q_t(\mathbf{x}_i|\frac{\mathbf{z}}{s(t)})$
$\bar{w}_i = \frac{w_i}{\sum_{j=1}^n w_j}$
$\hat{\mathbf{x}} = \sum_{i=1}^n \bar{w}_i \mathbf{x}_i$ {Equation (11)}
return $\frac{\hat{\mathbf{x}} - \frac{\mathbf{z}}{s(t)}}{s(t)\sigma(t)^2}$

C. Proofs

C.1. Proof of Theorem 4.1

Theorem 4.1 states

Let $t \in (0, \infty)$, then for a fixed n

$$\operatorname{Tr}\left(\operatorname{Cov}\left(\hat{\mu}_{\mathrm{KNN}}\right)\right) \leq \frac{\mathcal{Z}_q}{p_t(\mathbf{z})} \operatorname{Tr}\left(\operatorname{Cov}\left(\hat{\mu}_{\mathrm{MC}}\right)\right). \tag{62}$$

Proof. Starting with the diagonal of the covariance of the KNN estimator, we have

$$n \cdot \operatorname{Diag}\left(\operatorname{Cov}\left(\widehat{\mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]}_{KNN}\right)\right) = \sum_{i=1}^{N} \frac{p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)^2}{q_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)} \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \tag{63}$$

$$= \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)^2}{q_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)} \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} + \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} \frac{p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)^2}{q_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)} \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \tag{64}$$

$$\begin{aligned} = &\frac{\mathcal{Z}_q}{p_t\left(\mathbf{z}\right)} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right) p\left(\mathbf{x}^{(i)}\right)}{p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right) p\left(\mathbf{x}^{(i)}\right)} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\ &+ \frac{\mathcal{Z}_q}{p_t\left(\mathbf{z}\right)} \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} \frac{p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right) p\left(\mathbf{x}^{(i)}\right)}{p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right) p\left(\mathbf{x}^{(i)}\right)} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \end{aligned} \tag{65}$$

$$\begin{aligned} = &\frac{\mathcal{Z}_q}{p_t\left(\mathbf{z}\right)} \Bigg(\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\ &+ \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} \frac{p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}{p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right)} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\ &\pm \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \Bigg) \end{aligned} \tag{66}$$

$$\begin{aligned} = &\frac{\mathcal{Z}_q}{p_t\left(\mathbf{z}\right)} \Bigg(\sum_{i=1}^{N} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\ &+ \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} \frac{p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}{p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right)} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\ &- \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \Bigg) \end{aligned} \tag{67}$$

$$\begin{aligned} = &\frac{\mathcal{Z}_q}{p_t\left(\mathbf{z}\right)} \Bigg(\sum_{i=1}^{N} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\ &+ \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} \frac{p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}{p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right)} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\ &- \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} \frac{p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right)}{p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right)} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \Bigg) \end{aligned} \tag{68}$$

$$= \frac{\mathcal{Z}_q}{p_t(\mathbf{z})}\left(\sum_{i=1}^{N} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z},t\right]\right)^{\circ 2} + \sum_{\mathbf{x}^{(i)}\notin\mathcal{K}} \frac{p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right) - p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right)}{p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right)} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z},t\right]\right)^{\circ 2}\right) \tag{69}$$

$$= \frac{\mathcal{Z}_q}{p_t(\mathbf{z})}\left(\sum_{i=1}^{N} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z},t\right]\right)^{\circ 2} + \sum_{\mathbf{x}^{(i)}\notin\mathcal{K}} \frac{p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right) - p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right)}{p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right)} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z},t\right]\right)^{\circ 2}\right). \tag{70}$$

For $\mathbf{x}^{(i)} \notin \mathcal{K},\ p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right) \leq p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right)$ so therefore $\left(p_t\left(\mathbf{z}|\mathbf{x}^{(i)}\right) - p_t\left(\mathbf{z}|\mathbf{x}^{(k)}\right)\right) \leq 0$. Since all other factors in the second term are positive, The term as a whole is negative. We upper bound this term with 0 and continue with an elementwise inequality on the elements of the vector

$$n \cdot \text{Diag}\left(\text{Cov}\left(\widehat{\mathbb{E}\left[\mathbf{x}|\mathbf{z},t\right]}_{KNN}\right)\right) \leq \frac{\mathcal{Z}_q}{p_t(\mathbf{z})}\left(\sum_{i=1}^{N} p_t\left(\mathbf{x}^{(i)}|\mathbf{z}\right)\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z},t\right]\right)^{\circ 2}\right) \tag{71}$$

$$\leq \frac{\mathcal{Z}_q}{p_t(\mathbf{z})} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right) \tag{72}$$

$$\text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right) \leq \frac{\mathcal{Z}_q}{p_t(\mathbf{z})}\text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right). \tag{73}$$

Let $\text{Diag}\left(\text{Cov}\left(\cdot\right)\right)^j$ denote the j^{th} component of the diagonal vector. Since the inequaltiy is expressed element wise, $\forall j \in [0,\ldots,d]$ we have

$$\text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right)^j \leq \frac{\mathcal{Z}_q}{p_t(\mathbf{z})}\text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right)^j. \tag{74}$$

Summing over j, we write

$$\sum_{j=1}^{d}\text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right)^j \leq \sum_{j=1}^{d}\frac{\mathcal{Z}_q}{p_t(\mathbf{z})}\text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right)^j \tag{75}$$

$$\sum_{j=1}^{d}\text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right)^j \leq \frac{\mathcal{Z}_q}{p_t(\mathbf{z})}\sum_{j=1}^{d}\text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right)^j \tag{76}$$

$$\text{Tr}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right) \leq \frac{\mathcal{Z}_q}{p_t(\mathbf{z})}\text{Tr}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right), \tag{77}$$

concluding the proof. □

C.2. Uniform SNIS Trace of Covariance

For utility in future proofs, we derive an expression for the trace of covariance for a self-normalized importance sampling estimator with a uniform proposal

Lemma C.1. *Let $\hat{\mu}_U$ be a SNIS posterior mean estimator with $q(\mathbf{x}^{(i)}) = \frac{1}{N}\ \forall \mathbf{x}^{(i)} \in D$ then,*

$$n \cdot Tr\left(Cov\left(\hat{\mu}_U\right)\right) = N \cdot \sum_{i=1}^{N} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z},t\right])^{\circ 2}. \tag{78}$$

Proof.

$$n \cdot \text{Diag}\left(\text{Cov}(\hat{\mu}_{\text{SNIS}})\right) = \sum_{i=1}^{N} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q(\mathbf{x}^{(i)})}(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right])^{\circ 2} \tag{79}$$

$$n \cdot \text{Diag}\left(\text{Cov}(\hat{\mu}_{\text{SNIS}})\right) = \sum_{i=1}^{N} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{1/N}(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right])^{\circ 2} \tag{80}$$

$$n \cdot \text{Diag}\left(\text{Cov}(\hat{\mu}_{\text{SNIS}})\right) = N \cdot \sum_{i=1}^{N} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right])^{\circ 2}. \tag{81}$$

□

C.3. Proof of Theorem 4.2

Let $t \in (0, \infty)$, then for a fixed n

$$\begin{aligned}&\text{Tr}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right) \leq \\ &\left(1 - \frac{\sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z})}\right) \text{Tr}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right) + \frac{(N-k)}{N}\text{Tr}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right).\end{aligned} \tag{82}$$

Proof. We start from the expression for the Diagonal of the covariance of an SNIS posterior mean estimator

$$n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{SNIS}}\right)\right) = \sum_{i=1}^{N} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})}\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \tag{83}$$

$$= \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})}\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} + \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})}\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2}. \tag{84}$$

$$\tag{85}$$

For the KNN based proposal described in Equation (13), we can write

$$\begin{aligned}n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right) = &\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})}\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\ &+ \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} \frac{\mathcal{Z}_q}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2}\end{aligned} \tag{86}$$

$$\begin{aligned}= &\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})}\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\ &+ \frac{\mathcal{Z}_q}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2}\end{aligned} \tag{87}$$

$$\begin{aligned}= &\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})}\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\ &+ \frac{\mathcal{Z}_q}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\ &\pm \frac{\mathcal{Z}_q}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2\left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2}\end{aligned} \tag{88}$$

$$
\begin{aligned}
&= \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})} \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{\mathcal{Z}_q}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{\mathcal{Z}_q}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \sum_{i=1}^{N} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2}
\end{aligned} \tag{89}
$$

$$
\begin{aligned}
&= \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})} \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)}) + (N-k) p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)}) + (N-k) p_t(\mathbf{z}|\mathbf{x}^{(k)})}{N p_t(\mathbf{z}|\mathbf{x}^{(k)})} \cdot \sum_{i=1}^{N} N \cdot p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2}
\end{aligned} \tag{90}
$$

$$
\begin{aligned}
&= \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})} \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{(N-k) p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{N p_t(\mathbf{z}|\mathbf{x}^{(k)})} \cdot \sum_{i=1}^{N} N \cdot p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{(N-k) p_t(\mathbf{z}|\mathbf{x}^{(k)})}{N p_t(\mathbf{z}|\mathbf{x}^{(k)})} \cdot \sum_{i=1}^{N} N \cdot p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2}.
\end{aligned} \tag{91}
$$

We substitute Equation (78) into last term of Equation (91). Continuing, we have

$$
\begin{aligned}
n \cdot \mathrm{Diag}\left(\mathrm{Cov}\left(\hat{\mu}_{\mathrm{KNN}}\right)\right) &= \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})} \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{(N-k) p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{N p_t(\mathbf{z}|\mathbf{x}^{(k)})} \cdot \sum_{i=1}^{N} N \cdot p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{(N-k)}{N} \cdot n \cdot \mathrm{Diag}\left(\mathrm{Cov}\left(\hat{\mu}_{\mathrm{U}}\right)\right)
\end{aligned} \tag{92}
$$

$$
\begin{aligned}
&= \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})} \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{(N-k)p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \cdot \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z})^2 \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{(N-k)}{N} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right) \\
&= \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})} \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{(N-k)p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z})} \cdot \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} \frac{p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{(N-k)}{N} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right).
\end{aligned}
\tag{93, 94}
$$

We note that for $\mathbf{x}^{(i)} \notin \mathcal{K}$, $\frac{p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \leq 1$. Similarly for $\mathbf{x}^{(i)} \in \mathcal{K}$, $\frac{p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} \geq 1$ Swapping these fractions with 1 in the second and third term upper bounds each dimension of the trace of covariance. We substitute $\frac{p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z}|\mathbf{x}^{(k)})} = 1$ in the second and third terms of Equation (94), and swap the equality with a element-wise inequality

$$
\begin{aligned}
n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right) &\leq \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})} \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{(N-k)p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z})} \cdot \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{(N-k)}{N} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right) \\
&\leq \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{x}^{(i)}|\mathbf{z})^2}{q_t(\mathbf{x}^{(i)}|\mathbf{z})} \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{(N-k)p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{\mathcal{Z}_q - (N-k)p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z})} \cdot \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{(N-k)}{N} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right)
\end{aligned}
\tag{95, 96}
$$

$$
\begin{aligned}
&\leq \frac{\mathcal{Z}_q}{p_t(\mathbf{z})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} \frac{p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z}|\mathbf{x}^{(i)})} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{(N-k)p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z})} \sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{\mathcal{Z}_q - (N-k)p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z})} \cdot \sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{(N-k)}{N} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right)
\end{aligned} \tag{97}
$$

$$
\begin{aligned}
&\leq \frac{\mathcal{Z}_q}{p_t(\mathbf{z})} \sum_{i=1}^{N} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad - \frac{(N-k)p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z})} \sum_{i=1}^{N} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2} \\
&\quad + \frac{(N-k)}{N} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right).
\end{aligned} \tag{98}
$$

We substitute $n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right) = \sum_{i=1}^{N} p_t(\mathbf{x}^{(i)}|\mathbf{z}) \left(\mathbf{x}^{(i)} - \mathbb{E}\left[\mathbf{x}|\mathbf{z}, t\right]\right)^{\circ 2}$ into the second and third terms of Equation (98)

$$
\begin{aligned}
n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right) &\leq \frac{\mathcal{Z}_q}{p_t(\mathbf{z})} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right) \\
&\quad - \frac{(N-k)p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z})} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right) \\
&\quad + \frac{(N-k)}{N} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right)
\end{aligned} \tag{99}
$$

$$
\begin{aligned}
&\leq \frac{\mathcal{Z}_q - (N-k)p_t(\mathbf{z}|\mathbf{x}^{(k)})}{p_t(\mathbf{z})} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right) \\
&\quad + \frac{(N-k)}{N} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right)
\end{aligned} \tag{100}
$$

$$
\begin{aligned}
&\leq \frac{\sum_{\mathbf{x}^{(i)} \in \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z})} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right) \\
&\quad + \frac{(N-k)}{N} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right)
\end{aligned} \tag{101}
$$

$$
\begin{aligned}
&\leq \left(1 - \frac{\sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z})}\right) \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right) \\
&\quad + \frac{(N-k)}{N} \cdot n \cdot \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right).\right)
\end{aligned} \tag{102}
$$

Dividing both sides by n yields

$$
\begin{aligned}
&\text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right) \leq \\
&\left(1 - \frac{\sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z})}\right) \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right) + \frac{(N-k)}{N} \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right).
\end{aligned} \tag{103}
$$

Let $\text{Diag}\left(\text{Cov}\left(\cdot\right)\right)^j$ denote the j^{th} component of the diagonal vector. Since the inequaltiy in Equation (103) is expressed element wise, $\forall j \in [0, \ldots, d]$ we have

$$\text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right)^j \leq \left(1 - \frac{\sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z})}\right) \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right)^j + \frac{(N-k)}{N} \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right)^j . \tag{104}$$

Summing Equation (104) from $j = 1$ to $j = d$, we get

$$\sum_{j=1}^{d} \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right)^j \leq \sum_{j=1}^{d} \left(\left(1 - \frac{\sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z})}\right) \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right)^j + \frac{(N-k)}{N} \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right)^j \right) \tag{105}$$

$$\text{Tr}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right) \leq \left(1 - \frac{\sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z})}\right) \sum_{j=1}^{d} \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right)^j + \frac{(N-k)}{N} \sum_{j=1}^{d} \text{Diag}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right)^j \tag{106}$$

$$\text{Tr}\left(\text{Cov}\left(\hat{\mu}_{\text{KNN}}\right)\right) \leq \left(1 - \frac{\sum_{\mathbf{x}^{(i)} \notin \mathcal{K}} p_t(\mathbf{z}|\mathbf{x}^{(i)})}{p_t(\mathbf{z})}\right) \text{Tr}\left(\text{Cov}\left(\hat{\mu}_{\text{MC}}\right)\right) + \frac{(N-k)}{N} \text{Tr}\left(\text{Cov}\left(\hat{\mu}_{\text{U}}\right)\right), \tag{107}$$

concluding the proof. □

D. Additional Results

D.1. Score Estimator Hyperparameter Ablation

We include additional results from the investigation of estimator errors from Section 4.2. Figures 6 and 7 show the performance of the posterior Monte Carlo, STF, and KNN estimators for varying n and k. In both cases, we see that the KNN estimator improves with k with low bias for all estimators, and lower MSE than STF. Increasing the sample size n reduces MSE substantially for our method, but not STF due to the prevalent bias of their estimator. We see that for small k and n, the KNN estimator exhibits similar variance to STF. We hypothesize that becuase STF deterministically includes only one image from the posterior, the weights of this image dominates that of the other images drawn from the data distribution, in turn leading to lower sample variance.

D.2. Additional Denoiser Images

Figure 8 shows additional KNN posterior mean estimates for a variety of source images and noise levels.

D.3. CIFAR-10 Consistency Model Samples

Figure 9 shows examples from our trained consistency models for the models reported in Table 1. The samples are drawn with consistent seeds for each of the three models.

D.4. Diffusion Models

Although we demonstrate our method's effectiveness for consistency model training in Section 5, our method is not specifically tailored to consistency models, and may be used during model training whenever a empirical score estimator is required. In particular, our estimator can be directly applied to train diffusion models.

To evaluate the effect of our estimator on diffusion model training, we train and evaluate unconditional EDM (Karras et al., 2022) models on CIFAR-10 (Krizhevsky et al., 2009) using varying score estimators. Specifically, we train models using the standard single-sample Monte Carlo estimator, the STF estimator (Xu et al., 2023), and our nearest neighbour estimator. We retrain EDM and STF from their official repositories, utilizing the DDPM++ architecture for all models. We report the best FID and IS on 50,000 samples across all model checkpoints in Table 2

From Table 2, we see that both our method and STF improve the quality of generated samples over EDM. Compared to STF, our method has similar FID and better Inception Score.

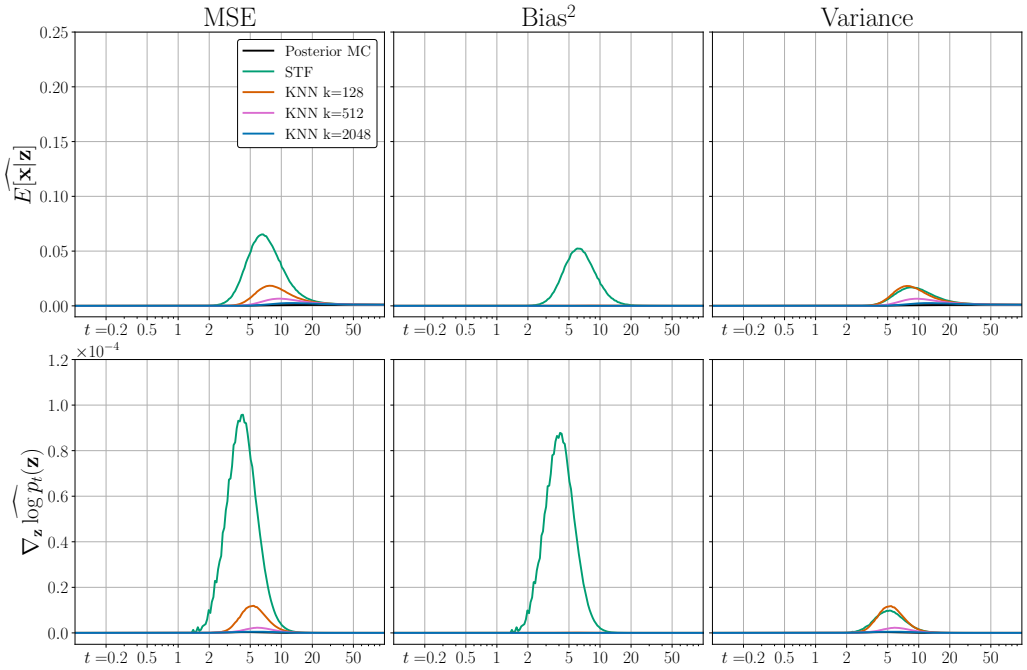

Figure 6: Estimator performance for $n = 256$. KNN estimator performance generally improves with k.

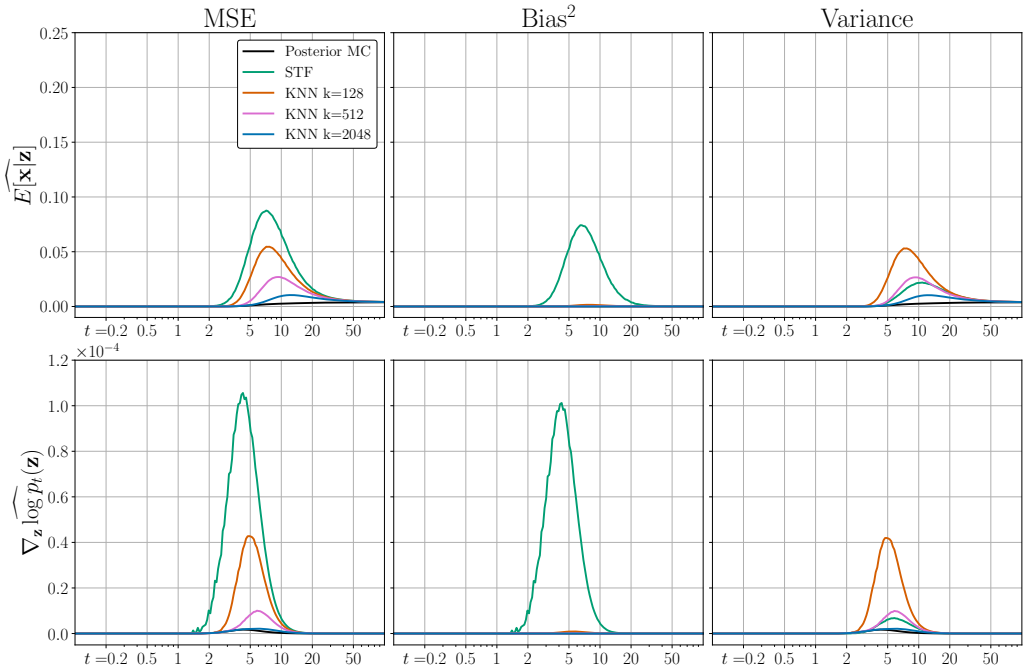

Figure 7: Estimator performance for $n = 64$. KNN estimator performance generally improves with k.

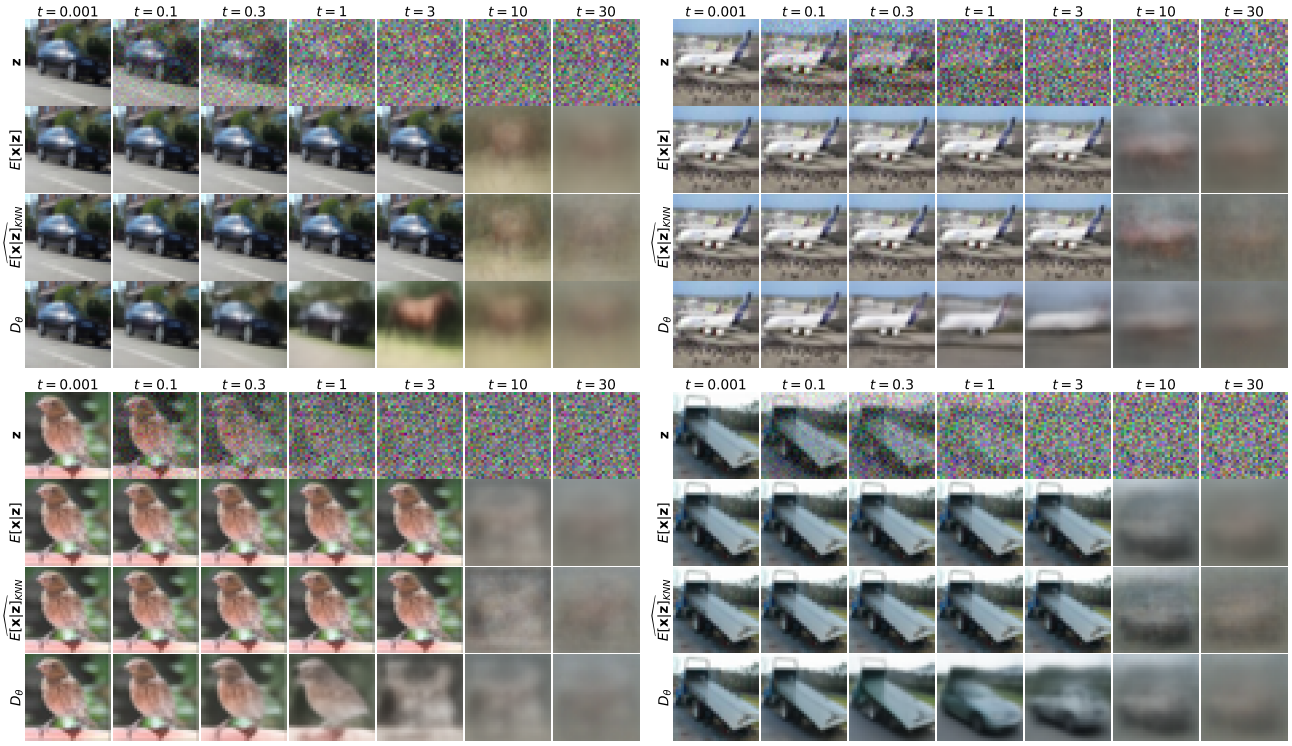

Figure 8: Additional qualitative comparisons of our estimator versus the ground truth posterior mean and a trained EDM denoiser.

Table 2: Unconditional EDM CIFAR-10 sample quality with varying score estimators.

Method	FID↓	IS↑
EDM (Karras et al., 2022)	2.01	9.76
STF (Xu et al., 2023)	**1.94**	9.81
EDM + KNN (ours)	1.96	**9.87**

D.5. Score Estimation on Additional Datasets

We apply our approach to two additional datasets to evaluate the efficacy of our method on larger and higher dimensionality data. Figure 10 shows the performance of our score estimator on CelebA 64x64 (Liu et al., 2015), while Figure 11 demonstrates the score estimation error of out method on Imagenet 64x64 (Deng et al., 2009).

Applying our method to higher dimensional and larger datasets presents two challenges. Firstly, to perform fast nearest neighbours search, we generate an index of the entire dataset which is stored in memory. Larger images and larger datasets therefore present memory concerns. Secondly, larger datasets increase the runtime of nearest neighbour search, which especially apparent on large datasets such as Imagenet.

To address these challenges, we apply two strategies. To reduce the memory overhead of our index, we quantize the data to 8 bit precision, reducing memory overhead by a factor of four. This has a minimal affect on our estimator, as pixel intensities are generally represented as 8 bit values. To improve the runtime of nearest neighbour search, we utilize an inverted file index (IVF) which partitions the data into c clusters and performs nearest neighbour search over only the p closest partitions. Both c and p can be tuned to trade off runtime versus accuracy. For CelebA 64x64 we utilized $c = 1788$ and $p = 25$ while for Imagenet, we used $c = 4527$ and $p = 100$.

In our work, we assume that the nearest neighbour search returns the k most probable elements of the posterior distribution. However, by switching to an approximate nearest neighbour search, this may no longer be true. We evaluate the impact of using approximate nearest neighbour search on estimator performance in Figure 12.

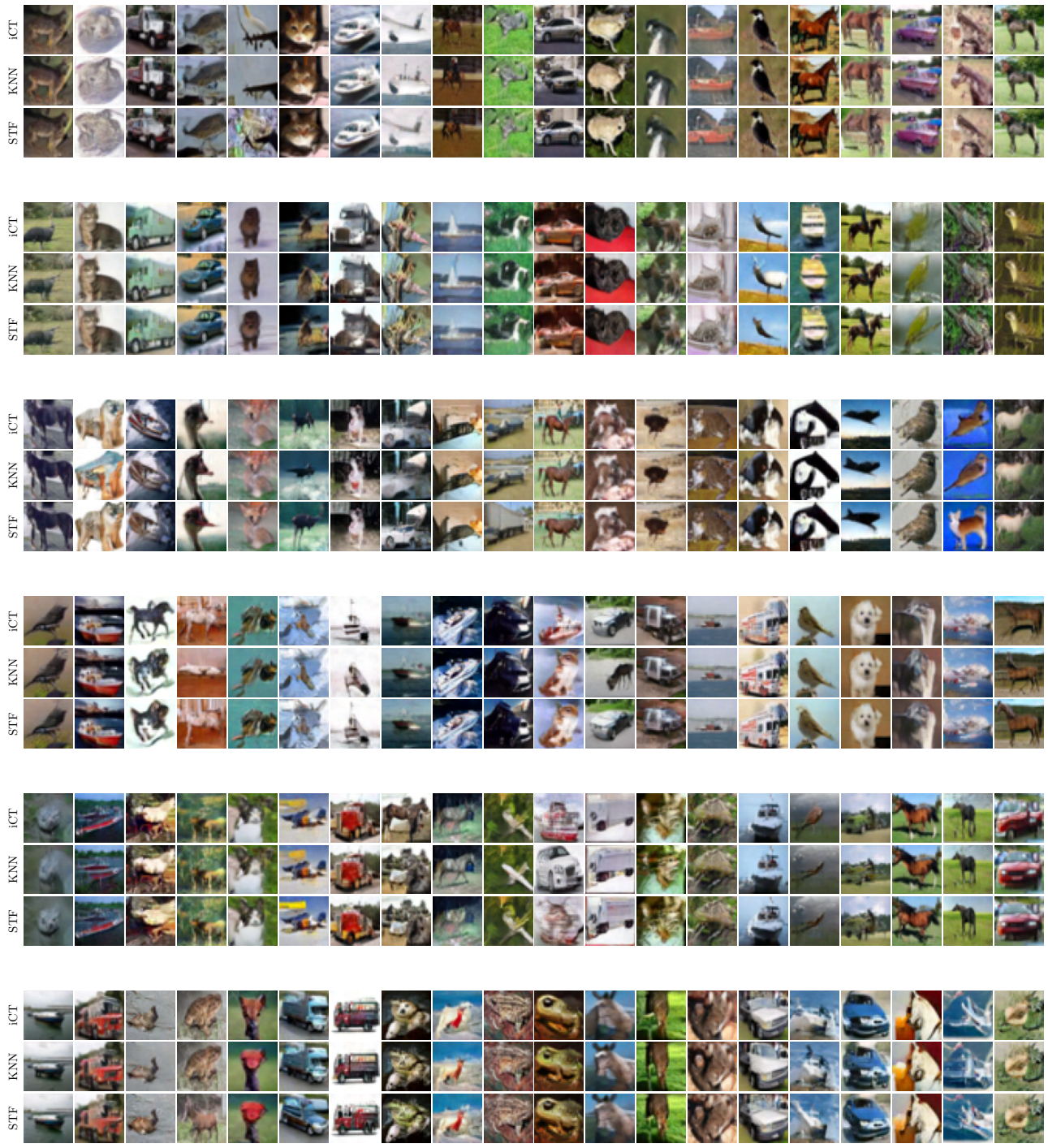

Figure 9: CIFAR-10 samples from trained consistency models with varying score estimators.

From Figure 12, we can see that approximate nearest neighbour introduces some additional bias to the posterior mean estimation, however the bias is still less that STF for the majority of noise levels. We expect bias could be reduced by increasing the number of samples n or the number of search partitions p.

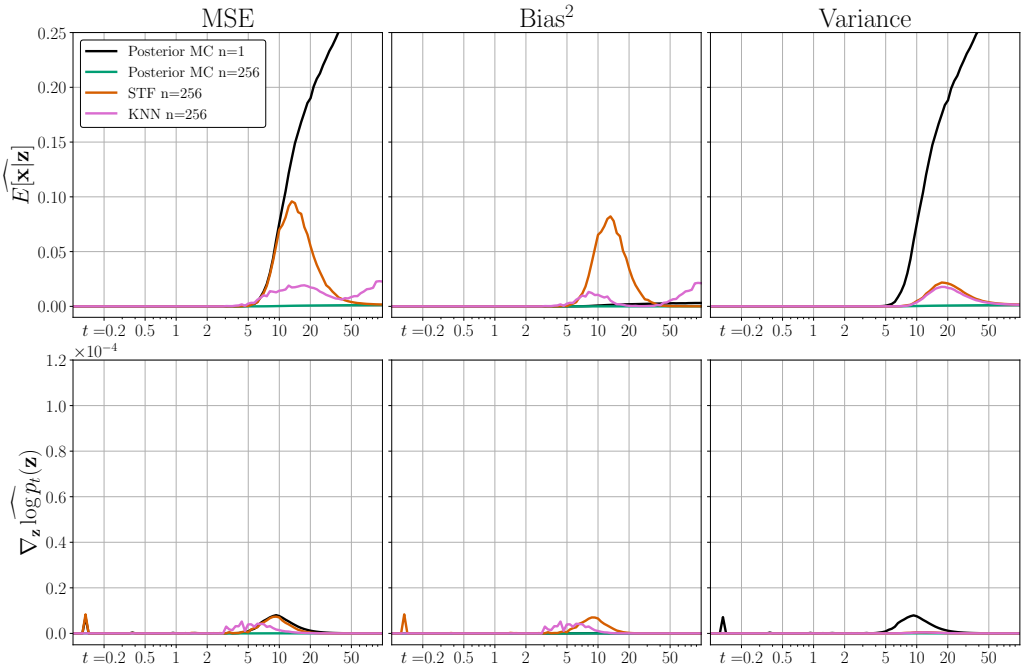

Figure 10: Comparison of various score estimators on unconditional CelebA 64x64

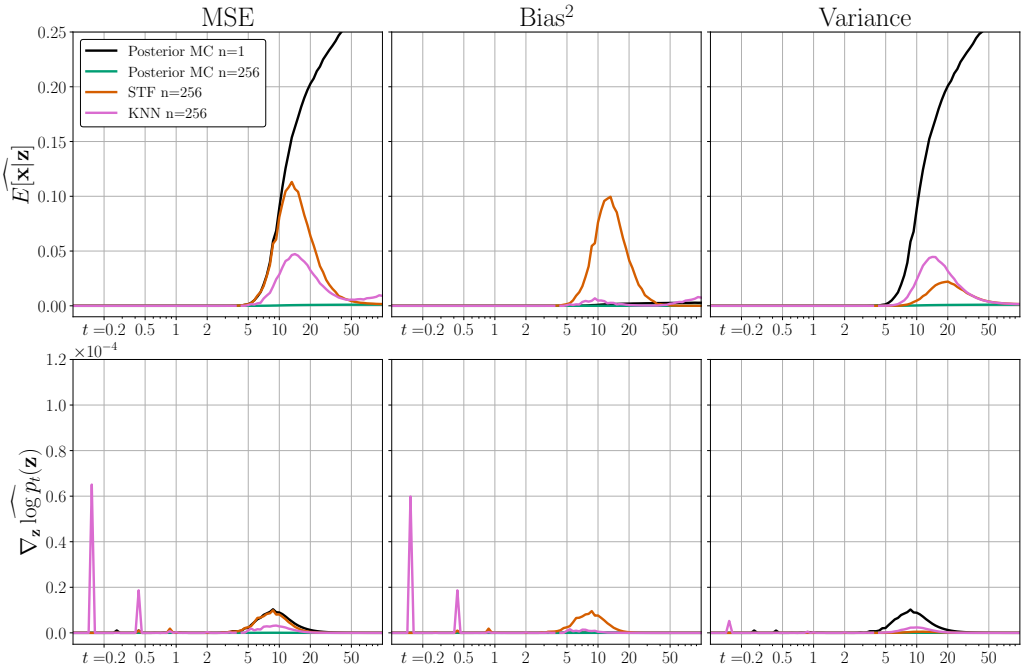

Figure 11: Comparison of various score estimators on unconditional Imagenet 64x64

E. Experimental Details

E.1. Consistency Model Training Configurations

To train the consistency models reported in Table 1, we follow the configuration outlined by Song & Dhariwal (2024). For the reader's convenience we reiterate that configuration here. We use the NCSN++ architecture (Song et al., 2021)

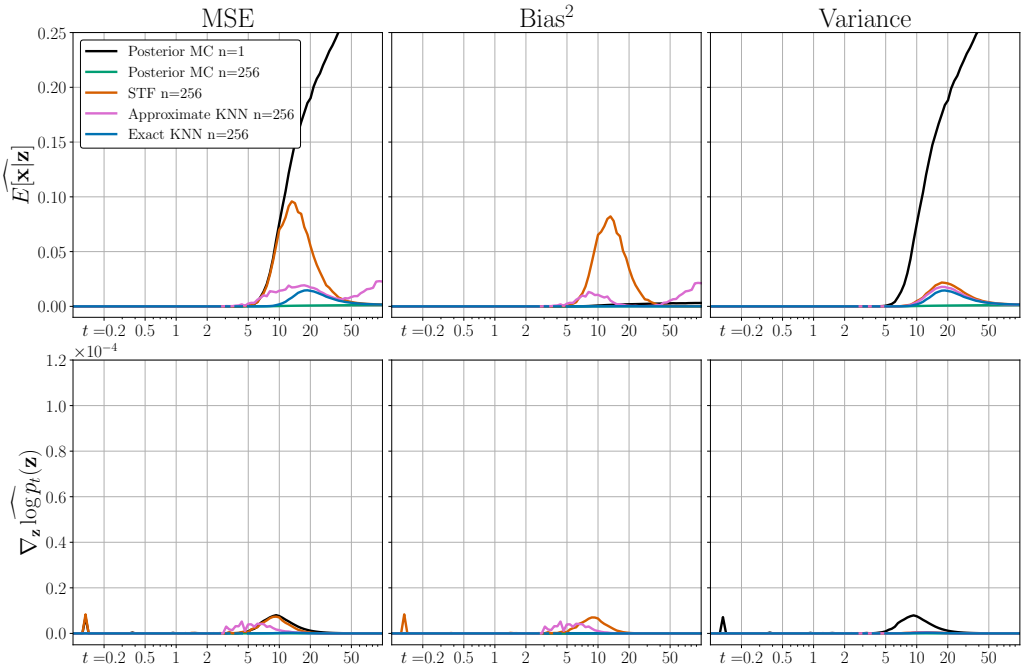

Figure 12: Comparison of approximate and exact nearest neighbour score estimators on CelebA 64x64

Table 3: Hyperparameter choices for CIFAR-10 consistency model training

Hyperparameter	iCT	iCT + KNN	iCT + STF
Training iterations	400,000	400,000	400,000
Batch Size	1024	1024	1024
Learning Rate	0.0001	0.0001	0.0001
EMA Decay Rate	0.99993	0.99993	0.99993
Dropout	0.3	0.3	0.3
ODE Solver	Euler	Huen	Huen
Score Estimator	Posterior MC	KNN	STF
Estimator batch size	1	256	256
KNN search size	N/A	2048	N/A

for all models. Each model was trained using the improved consistency training algorithm, with the Pseudo-Huber loss function (Charbonnier et al., 1997) with $c = 0.03$. We weight the consistency matching loss with the weighting function $\lambda(t) = \frac{1}{\sigma_t - \sigma_{t-1}}$. Noise levels are sampled from a log-normal distribution with $P_{mean} = -1.1$ and $P_{std} = 2.0$. We follow an exponential discretzation schedule, starting with $s_0 = 10$ discretization steps, doubling every 50,000 training iterations to a final value of $s_1 = 1280$. We do not use exponential averaging on the teacher network, instead just applying a stopgrad to the student network weights. Hyperparameters for all runs are reported in Table 3

We checkpoint each model every 2.5 million training images. We evaluate each checkpoint by generated 50,000 images and comparing against the training dataset using the torch-fidelity package (Obukhov et al., 2020). We share random seeds across all evaluations. For each method, we select the checkpoint with the minimum FID to report in Table 1.

Table 4: Total training times for consistency model training

Model	**Training time**	**Training Time vs iCT**
iCT	188.9 hrs	100%
iCT + KNN	220.5 hrs	117%
iCT + STF	185.8 hrs	98.3%

The total training time for each of the above configurations is reported in Table 4. We note that each model was trained on a different node of A100 GPUs which may explain how the iCT+STF model was able to train faster than the iCT model despite extra score estimation overhead.