Training Diffusion Models with Noisy Data via SFBD Flow

Haoye Lu [1 2] **Darren Lo** [1] **Yaoliang Yu** [1 2]

Abstract

Diffusion models achieve strong generative performance but often rely on large datasets that may include sensitive content. This challenge is compounded by the models' tendency to memorize training data, raising privacy concerns. SFBD (Lu et al., 2025) addresses this by training on corrupted data and using limited clean samples to capture local structure and improve convergence. However, its iterative denoising and fine-tuning loop requires manual coordination, making it burdensome to implement. We reinterpret SFBD as an alternating projection algorithm and introduce a continuous variant, SFBD flow, that removes the need for alternating steps. We further show its connection to consistency constraint-based methods, and demonstrate that its practical instantiation, Online SFBD, consistently outperforms strong baselines across benchmarks.

1. Introduction

Diffusion-based generative models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021a;b; 2023) have attracted growing interest and are now regarded as one of the most powerful frameworks for modelling high-dimensional distributions. They have enabled remarkable progress across various domains (Croitoru et al., 2023), including image (Ho et al., 2020; Song et al., 2021a;b; Rombach et al., 2022), audio (Kong et al., 2021; Yang et al., 2023), and video generation (Ho et al., 2022).

Diffusion models can be efficiently trained using the conditional score-matching loss, making them relatively easy to scale. This scalability enables the training of very large models on web-scale datasets – a crucial factor in achieving high performance. This approach has driven recent breakthroughs in image generation, exemplified by models such as Stable Diffusion (-XL) (Rombach et al., 2022; Podell et al., 2024) and DALL-E (Betker et al., 2023). However, this success comes with challenges: large-scale datasets often include copyrighted material, and diffusion models are more prone than earlier generative methods like GANs (Goodfellow et al., 2014; 2020) to memorizing training data, potentially reproducing entire samples (Carlini et al., 2023; Somepalli et al., 2023).

A recently proposed strategy to address memorization and copyright concerns involves training or fine-tuning diffusion models on corrupted data (Daras et al., 2023b; Somepalli et al., 2023; Daras and Dimakis, 2023; Daras et al., 2024). In this setting, the model never has direct access to the original data. Instead, each sample is transformed via a known, non-invertible corruption process, such as pixel-wise additive Gaussian noise in image datasets, ensuring that the original content cannot be reconstructed or memorized at the individual sample level. Remarkably, under mild conditions, such corruptions - although irreversible at the sample level – can induce a bijection between the original and corrupted data distributions (Bora et al., 2018). Specifically, the corrupted data distribution has a density equal to the convolution of the true data density with the corruption noise distribution (Meister, 2009; Lu et al., 2025). As a result, it is theoretically possible to recover the original data distribution by first estimating the corrupted (noisy) density from samples, and then performing density deconvolution to approximate the underlying true data density.

We refer to this task – recovering the true data distribution from noisy observations – as the *deconvolution problem*. Motivated by this formulation, several works (Daras et al., 2024; 2025; Lu et al., 2025) have shown that diffusion models can effectively address the deconvolution problem either by applying iterative denoising and fine-tuning, as in SFBD (Lu et al., 2025), or by enforcing consistency constraints (CCs) during training (Daras et al., 2023a). Specifically, when paired with a small set of copyright-free clean data, both SFBD and CC-based methods have been shown to guide diffusion models toward generating high-quality images. However, SFBD relies on costly iterative denoising and fine-tuning, while CC-based methods require solving backward stochastic differential equations (SDEs) at each training step, making both approaches computationally expensive in different ways.

*Equal contribution [1]Cheriton School of Computer Science, University of Waterloo, Canada [2]Vector Institute, Canada. Correspondence to: Haoye Lu <haoye.lu@uwaterloo.ca>.

Published at Data in Generative Models Workshop: The Bad, the Ugly, and the Greats (DIG-BUGS) at ICML 2025, Vancouver, Canada.

In this paper, we eliminate the need for iterative denoising and fine-tuning in SFBD by introducing a continuous variant, *SFBD flow*. We reinterpret SFBD as an alternating projection between two sets of stochastic processes, framing it as a stochastic process optimization problem. Inspired by Sinkhorn flow (Reza Karimi et al., 2024) and Schrödinger bridge flow (Bortoli et al., 2024), this view leads to a generalized family of diffusion-based deconvolution methods, termed γ-SFBD for $\gamma \in (0, 1]$, which guide the model toward the clean data distribution. When $\gamma = 1$, the method recovers the original SFBD; as $\gamma \to 0$, the discrete sequence of stochastic processes transitions into a continuous evolution, naturally yielding the SFBD flow.

We further show that SFBD flow arises as a steepest gradient descent in function space, with γ-SFBD as its discrete approximation. This perspective motivates Online SFBD, a practical diffusion-based deconvolution method avoiding repeated fine-tuning (see Sec 5). We also reveal a close connection to CC-based methods, offering a unified view of both approaches. Empirical results validate our analysis, with Online SFBD consistently outperforming strong baselines across benchmarks.

2. Preliminaries

In this section, we review diffusion models, the deconvolution problem, and two typical methods for training diffusion models on data corrupted by Gaussian noise.

Diffusion models. Diffusion models generate data by progressively adding Gaussian noise to input samples and then learning to reverse this process through a sequence of denoising steps. Formally, given an initial data distribution p_0 over $\mathbb{R}^d$, the forward process is governed by the SDE

$$\mathrm{d}\mathbf{x}_t = \mathrm{d}\mathbf{w}_t, \;\; \mathbf{x}_0 \sim p_0, \;\; t \in [0, T], \tag{1}$$

where T is a fixed positive constant. $\{\mathbf{w}_t\}_{t\in[0,T]}$ is the standard Brownian motion. Eq (1) induces a transition kernel $p_{t|s}(\mathbf{x}_t|\mathbf{x}_s)$ for $0 \le s \le t \le T$. For $s = 0$,

$$p_{t|0}(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0, t\,\mathbf{I}), \quad t \in [0, T]. \tag{2}$$

When T is large, the terminal state $\mathbf{x}_T$ closely approximates a sample from the isotropic Gaussian distribution $\mathcal{N}(\mathbf{0}, T\,\mathbf{I})$. Let $p_t(\mathbf{x}_t) = \int p_{t|0}(\mathbf{x}_t|\mathbf{x}_0)\, p_0(\mathbf{x}_0)\, \mathrm{d}\mathbf{x}_0$ denote the marginal distribution of $\mathbf{x}_t$, where $p_T \approx \mathcal{N}(\mathbf{0}, T\,\mathbf{I})$. Anderson (1982) showed that the time-reversed process corresponding to the forward SDE can be described by the backward SDE:

$$\mathrm{d}\mathbf{x}_t = -\mathbf{s}_t(\mathbf{x}_t)\, \mathrm{d}t + \mathrm{d}\bar{\mathbf{w}}_t, \;\; \mathbf{x}_T \sim p_T, \tag{3}$$

where $\bar{\mathbf{w}}_t$ is standard Brownian motion in reverse time and $\mathbf{s}_t = \mathbf{s}_t^* := \nabla \log p_t$ is the score function. Crucially, this reverse SDE induces transition kernels that match the posterior of the forward process: $p_{s|t}(\mathbf{x}_s|\mathbf{x}_t) = \frac{p_{t|s}(\mathbf{x}_t|\mathbf{x}_s)p_s(\mathbf{x}_s)}{p_t(\mathbf{x}_t)}$ for $s \le t$ in $[0, T]$. It is known that $\mathbf{s}_t^*(\mathbf{x}_t) = \frac{1}{t}(\mathbb{E}_{p_{0|t}}[\mathbf{x}_0|\mathbf{x}_t] - \mathbf{x}_t)$, where the conditional expectation $\mathbb{E}_{p_{0|t}}[\mathbf{x}_0 \mid \mathbf{x}_t]$ is typically approximated in practice by a neural network-denoiser $D_{\boldsymbol{\phi}}(\mathbf{x}_t)$ (Karras et al., 2022), trained by minimizing

$$\mathcal{L}_d(\boldsymbol{\phi}) = \mathbb{E}_{t\sim\mathcal{T}}\, \mathbb{E}_{p_0}\, \mathbb{E}_{p_{t|0}} \left[w(t) \| D_{\boldsymbol{\phi}}(\mathbf{x}_t, t) - \mathbf{x}_0 \|^2 \right], \tag{4}$$

where $w(t)$ is a time-dependent weighting function and $\mathcal{T}$ denotes a sampling distribution over $[0, T]$. With a well-trained denoiser $D_{\boldsymbol{\phi}}$, $\mathbf{s}_t^*$ can be approximated by

$$\mathbf{s}_t^{\boldsymbol{\phi}}(\mathbf{x}_t) := \tfrac{1}{t}\big(D_{\boldsymbol{\phi}}(\mathbf{x}_t, t) - \mathbf{x}_t\big). \tag{5}$$

Substituting this estimate into Eq (3), one can simulate the reverse-time SDE starting from $\tilde{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}, T\,\mathbf{I})$, yielding a sample $\tilde{\mathbf{x}}_0$ that serves as an approximation sampled from p_0.

Deconvolution problem. We follow the setup of Daras et al. (2024); Lu et al. (2025), where corrupted samples $\mathcal{Y} = \{\mathbf{y}^{(i)}\}_{i=1}^n$ are generated as $\mathbf{y}^{(i)} = \mathbf{x}^{(i)} + \boldsymbol{\epsilon}^{(i)}$, with $\mathbf{x}^{(i)} \sim p_{\text{data}}$ and $\boldsymbol{\epsilon}^{(i)} \sim h = \mathcal{N}(\mathbf{0}, \tau\mathbf{I})$ drawn independently, where $\tau \in (0, T)$ is known and fixed. The resulting samples $\mathbf{y}^{(i)}$ follow a distribution with density $p_\tau^* = p_{\text{data}} * h$, where $*$ denotes the convolution operator (Lu et al., 2025). In addition, we assume access to a small set of clean samples $\mathcal{D}_{\text{clean}} = \{\mathbf{x}^{(i)}\}_{i=1}^M$ with $\mathbf{x}^{(i)} \sim p_{\text{data}}$.

While deconvolution theory (Meister, 2009; Lu et al., 2025) and related empirical results in the context of GANs (Bora et al., 2018) have demonstrated the theoretical and practical feasibility of learning the true data distribution from noisy samples, a key challenge remains: how to effectively train a diffusion model on corrupted data to generate clean samples.

Consistency constraint-based method. Daras et al. (2024) first addressed this problem using CCs (Daras et al., 2023a). With noisy samples $\mathbf{x}_\tau \sim p_\tau^*$, they trained a network $\mathbf{s}_t^{\boldsymbol{\phi}}$ to approximate score $\mathbf{s}_t^*$ for $t > \tau$ via a modified loss called ambient score matching (ASM). Specifically, $\mathbf{s}_t^{\boldsymbol{\phi}}$ is implemented through Eq (5), where $D_{\boldsymbol{\phi}}(\mathbf{x}_t, t)$ approximates $\mathbb{E}_{p_{0|t}}[\mathbf{x}_0 \mid \mathbf{x}_t]$. For $t \le \tau$, score matching is inapplicable, and instead $D_{\boldsymbol{\phi}}(\mathbf{x}_t, t)$ is trained to satisfy the CCs:

$$\mathbb{E}_{p_{0|s}}[\mathbf{x}_0|\mathbf{x}_s] = \mathbb{E}_{p_{r|s}}\big[\mathbb{E}_{p_{0|r}}[\mathbf{x}_0|\mathbf{x}_r]\big], \text{ for } 0 \le r \le s \le T$$

by jointly minimizing the *consistency loss*:

$$\mathcal{L}_{\text{con}}(\boldsymbol{\phi}, r, s) = \mathbb{E}_{p_s} \big\| D_{\boldsymbol{\phi}}(\mathbf{x}_s, s) - \mathbb{E}_{p_{r|s}}[D_{\boldsymbol{\phi}}(\mathbf{x}_r, r)] \big\|^2 \tag{6}$$

where r and s are sampled from predefined distributions. Sampling from $p_{r|s}$ is performed by solving Eq (3) backward from $\mathbf{x}_s$, using the network-estimated drift $\mathbf{s}_t^{\boldsymbol{\phi}}$ from Eq (5). To sample from p_s, one first draws $\mathbf{x}_\tau$ for $\tau > s$, then samples $\mathbf{x}_s \sim p_{s|\tau}$ analogously. If $D_{\boldsymbol{\phi}}$ minimizes the consistency loss for all r, s and satisfies $\mathbf{s}_t^{\boldsymbol{\phi}} = \mathbf{s}_t^*$ for $t > \tau$, then $\mathbf{s}_t^{\boldsymbol{\phi}}$ exactly recovers $\mathbf{s}_t^*$ for all $t \in [0, T]$, allowing $p_0 = p_{\text{data}}$ to be sampled via Eq (3) (Daras et al., 2024).

However, both Daras et al. and Lu et al. showed that using CCs alone is insufficient to recover the drift below τ due to poor sample complexity (Lu et al., 2025; Daras et al., 2025). To address this, Daras et al. propose jointly training the model with the standard denoising loss Eq (4) on $\mathcal{D}_{\text{clean}}$ and demonstrate strong empirical performance.

Stochastic forward-backward deconvolution (SFBD). Instead of relying on CCs to recover the distribution for $t \le \tau$, Lu et al. proposed an iterative scheme, SFBD, that alternates between finetuning and denoising steps (Lu et al., 2025). Given a sample set $\mathcal{E}$, let $p_{\mathcal{E}}$ denote the empirical distribution induced by $\mathcal{E}$. Starting from a pretrained model D_{ϕ_0} trained on $\mathcal{D}_{\text{clean}}$, the algorithm proceeds as follows for $k = 1, 2, \dots, K$:

(Denoise) $\mathcal{E}_k \leftarrow \{\mathbf{y}_0^{(i)}$: solve Eq (3) from $t = \tau$ to 0 with $\mathbf{s}_t(\mathbf{x}_t) = \frac{D_{\phi_k}(\mathbf{x}_t, t) - \mathbf{x}_t}{t}, \mathbf{x}_\tau = \mathbf{y}_\tau^{(i)} \in \mathcal{E}_{\text{noisy}}\}$.

(Finetune) Update D_{ϕ_k} to obtain $D_{\phi_{k+1}}$ by miminizing Eq (4) with $p_0 = p_{\mathcal{E}_k}$.

Lu et al. (2025) showed that as $K \to \infty$, $p_{\mathcal{E}_K}$ converges to the true distribution p_{data}. While SFBD outperforms DDIM (Song et al., 2021a) trained solely on clean data (e.g., on CelebA (Liu et al., 2015)), its iterative nature makes implementation challenging. In Sec 3, we show that the Denoise and Finetune steps can be viewed as alternating projections in the space of stochastic processes, leading to a continuous formulation, SFBD flow, that removes the need for iterative finetuning.

3. SFBD as alternative projections

In this section, we show that SFBD can be interpreted as an alternating projection algorithm. We begin by introducing notation to facilitate the discussion.

Notation. Let $\mathcal{M}$ denote the set of path measures over $t \in [0, \tau]$ induced by the backward process Eq (3), with arbitrary drift $\mathbf{s} : [0, \tau] \times \mathbb{R}^d \to \mathbb{R}^d$ and fixed initial distribution p_τ^* at $t = \tau$. We write $M(\mathbf{s}) \in \mathcal{M}$ to denote the path measure corresponding to drift $\mathbf{s}$. Similarly, let $\mathcal{D}$ denote the set of path measures over $t \in [0, \tau]$ induced by the forward process Eq (1), with arbitrary initial distribution p_0, and let $D(q) \in \mathcal{D}$ denote the measure induced by $p_0 = q$.

Alternative projections. SFBD then can be formulated as an algorithm alternating between two projections: the Markov projection (M-Proj) and the diffusion projection (D-Proj), defined as follows:

$$\text{(M-Proj)}\ M^k = \text{proj}_{\mathcal{M}} P^k := \underset{M \in \mathcal{M}}{\text{argmin}}\, D_{\text{KL}}(P^k \,\|\, M) \quad (7)$$

$$\text{(D-Proj)}\ P^{k+1} = \text{proj}_{\mathcal{D}} M^k := \underset{P \in \mathcal{D}}{\text{argmin}}\, D_{\text{KL}}(M^k \,\|\, P) \quad (8)$$

for $k = 0, 1, 2, \dots, K$, with initial path measure $P^0 = \mathcal{D}(p_{\mathcal{E}_{\text{clean}}})$. Since each $M \in \mathcal{M}$ is fully determined by a backward drift $\mathbf{s}$, we denote the drift of M^k by $\mathbf{s}^k$, i.e., $M^k = M(\mathbf{s}^k)$. Thus, the M-Proj can be equivalently written as $\text{argmin}_{\mathbf{s}} D_{\text{KL}}(P^k \,\|\, M(\mathbf{s}))$.

The M-Proj corresponds to the finetuning step in SFBD. To see this, by Lem 1 in Appx A.6,

$$D_{\text{KL}}(P^k \,\|\, M^k) = D_{\text{KL}}(p_\tau^k \,\|\, p_\tau^*) + \mathbb{E}_{P^k}\left[\tfrac{1}{2}\int_0^\tau \|\nabla \log p_t^k(\mathbf{x}_t) - \mathbf{s}_t^k(\mathbf{x}_t)\|^2\right]$$

where p_t^k denotes the marginal density of P^k at time t. Since the first term is independent of M^k, minimizing the KL reduces to setting $\mathbf{s}_t^k(\mathbf{x}_t) = \nabla \log p_t^k(\mathbf{x}_t)$, i.e., performing score matching. This corresponds to the fine-tuning step that minimizes Eq (4) with $p_0 = p_0^k$ (Karras et al., 2022).

Likewise, D-Proj corresponds to the denoising step. By the disintegration theorem (Vargas et al., 2021),

$$D_{\text{KL}}(M^k \,\|\, P) = D_{\text{KL}}(m_0^k \,\|\, p_0) + \mathbb{E}_{M^k}\Big[\log \tfrac{\mathrm{d}M^k(\cdot|\mathbf{x}_0)}{\mathrm{d}P(\cdot|\mathbf{x}_0)}\Big] \quad (9)$$

where m_0^k is the marginal of M^k at $t = 0$. Since $P \in \mathcal{D}$ is determined by the forward SDE in Eq (1), its conditional path measure given $\mathbf{x}_0$ is fixed, making the second term constant. Therefore, minimizing the KL divergence reduces to matching the marginals, i.e., $p_0 = m_0^k$ and thus $P^{k+1} = D(m_0^k)$. In other words, D-Proj sets p_0 to the distribution of the denoised samples in the denoising step.

Evolution of $\mathbf{s}^k$. In practice, the only component in SFBD requiring estimation is $\mathbf{s}^k$, parameterized by a neural network to approximate $\nabla \log p_t^k(\mathbf{x}_t)$. As $k \to \infty$, $\mathbf{s}_t^k$ converges to the true score function $\mathbf{s}_t^* = \nabla \log p_t$ associated with the forward diffusion process Eq (1) initialized with $p_0 = p_{\text{data}}$ (Lu et al., 2025).

The updates in Eq (7) and Eq (8) can be compactly written as $M^{k+1} = \text{argmin}_{M \in \mathcal{M}} D_{\text{KL}}(\text{proj}_{\mathcal{D}} M^k \,\|\, M)$. Since each $M^k = M(\mathbf{s}^k)$ is fully determined by its drift $\mathbf{s}^k$, this is equivalent to

$$\mathbf{s}^{k+1} = \text{argmin}_{\mathbf{s}}\, \tilde{\mathcal{L}}(\mathbf{s}, M^k), \quad (10)$$

with $\tilde{\mathcal{L}}(\mathbf{s}, M^k) = D_{\text{KL}}(\text{proj}_{\mathcal{D}} M^k \,\|\, M(\mathbf{s}))$. It can be shown (in Appx A.1) that minimizing $\tilde{\mathcal{L}}$ is equivalent to minimizing

$$\mathcal{L}(\mathbf{s}, M^k) = \int_0^\tau \mathcal{L}_t \,\mathrm{d}t := \int_0^\tau \mathbb{E}_{D(m_0^k)}\, \tfrac{1}{2}\left\|\tfrac{\mathbf{x}_0 - \mathbf{x}_t}{t} - \mathbf{s}_t(\mathbf{x}_t)\right\|^2 \mathrm{d}t, \quad (11)$$

where m_0^k is the marginal of M^k at $t = 0$. Thus, SFBD can be interpreted as the iterative update:

$$\mathbf{s}^{k+1} = \text{argmin}_{\mathbf{s}} \mathcal{L}(\mathbf{s}, m_0(\mathbf{s}^k)), \quad (12)$$

with $\mathbf{s}^0 = \operatorname{argmin}_{\mathbf{s}} D_{\mathrm{KL}}(\mathcal{D}(p_{\mathcal{E}_{\text{clean}}}) \,\|\, M(\mathbf{s}))$. In practice, estimating each $\mathbf{s}^k$ requires training a separate neural network, making the process computationally expensive and difficult to implement due to manual intervention and unclear stopping criteria. In Sec 4, we show that this update can be made continuous by following the steepest descent direction of $\mathcal{L}$, enabling end-to-end training of a single network.

4. SFBD flow

In this section, we extend SFBD to a family of iterative deconvolution procedures, γ-SFBD for $\gamma \in (0, 1]$. When $\gamma = 1$, it recovers the original SFBD; as $\gamma \to 0$, the discrete sequence $\{M^k\}_{k\in\mathbb{N}}$ and drift $\mathbf{s}^k$ converge to continuous flows $\{M^\kappa\}_{\kappa\geq 0}$ and $\{\mathbf{s}^\kappa\}_{\kappa\geq 0}$.

We show that γ-SFBD admits two natural derivations: a generalized D-Proj, which intuitively explains how smaller γ yields smoother trajectories; and a discretized functional gradient descent on $\mathcal{L}(\mathbf{s}, M_0(\mathbf{s}))$, formally establishing the convergence of the discrete sequence to a continuous flow.

Derive γ-SFBD through a generalized D-Proj. For $\gamma \in (0, 1]$, consider a generalized D-Proj:

$$(\gamma\text{-D-Proj}) \quad \begin{aligned} P^{k+1,\gamma} = \operatorname{argmin}_{P\in\mathcal{D}} &(1-\gamma)\, D_{\mathrm{KL}}(P^{k,\gamma} \,\|\, P) \\ &+ \gamma\, D_{\mathrm{KL}}(M^k \,\|\, P) \end{aligned} \tag{13}$$

We refer to SFBD with D-Proj replaced by γ-D-Proj as γ-SFBD. When $\gamma = 1$, it recovers the original SFBD. (Although M^k does depend on γ, we keep the original notation for simplicity.) To see how γ-D-Proj smooths the update, note that the denoised samples at iteration k follow a distribution with density (see Appx A.2):

$$p_0^{k+1,\gamma} = (1-\gamma)\, p_0^{k,\gamma} + \gamma\, m_0^k \tag{14}$$

where $p_0^{0,\gamma} = p_{\mathcal{E}_{\text{clean}}}$ and $P^{k+1,\gamma} = D(p_0^{k+1,\gamma})$. Basically, the parameter γ controls how much of the denoised set is updated using the latest model. When $\gamma = 1$, all samples are replaced, recovering standard SFBD. As $\gamma \to 0$, the updates become infinitesimal, leaving M^{k+1} – obtained by projecting $P^{k+1,\gamma}$ onto $\mathcal{M}$ – and its corresponding $\mathbf{s}^{k+1}$ nearly unchanged. Despite the smoothing effect, γ-SFBD guarantees convergence for all $\gamma \in (0, 1]$. Let $\Phi_p(\mathbf{u}) = \mathbb{E}_p[\exp(i\,\mathbf{u}^\top \mathbf{x})]$ denote the characteristic function of p for $\mathbf{u} \in \mathbb{R}^d$. Under mild assumptions,

Proposition 1. *For $k \geq 0$, $D_{\mathrm{KL}}(p_{\text{data}} \,\|\, p_0^{k+1,\gamma}) - D_{\mathrm{KL}}(p_{\text{data}} \,\|\, p_0^{k,\gamma}) \leq -\gamma D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^{k,\gamma})$. In addition,*

$$\min_{k=1,\ldots,K} \left|\Phi_{p_{\text{data}}}(\mathbf{u}) - \Phi_{p_0^{k,\gamma}}(\mathbf{u})\right| \leq \exp\left(\tfrac{\tau}{2}\|\mathbf{u}\|^2\right)\left(\tfrac{2M}{\gamma K}\right)^{1/2}$$

for $K \geq 1$, $\mathbf{u} \in \mathbb{R}^d$, and $M = D_{\mathrm{KL}}(p_{\text{data}} \,\|\, p_{\mathcal{E}_{\text{clean}}})$.

(All proofs are deferred to the appendix.) Prop 1 shows that for all $\gamma \in (0, 1]$, $p_0^{k,\gamma}$ progressively approaches p_{data} as k increases, and the convergence of characteristic functions implies convergence of the underlying distributions.

γ-SFBD as functional gradient descent. In Sec 3, we showed that SFBD updates the backward drift $\mathbf{s}^k$ by solving $\operatorname{argmin}_{\mathbf{s}} \mathcal{L}(\mathbf{s}, M_0(\mathbf{s}^k))$. We now consider a relaxed version, where $\mathbf{s}$ is updated via a single gradient descent step in function space with step size $\gamma \in (0, 1]$. This update rule exactly recovers the γ-SFBD algorithm.

Recall that for a functional $\ell : \mathcal{F} \to \mathbb{R}$ defined over a function space $\mathcal{F}$, its functional derivative at $\mathbf{u} \in \mathcal{F}$ with respect to a reference measure P is a function $\nabla_P \ell(\mathbf{u}) \in \mathcal{F}$ (when it exists) satisfying (Courant and Hilbert, 1989):

$$\int\langle\nabla_P \ell(\mathbf{u})(\mathbf{x}), \boldsymbol{\nu}(\mathbf{x})\rangle \mathrm{d}\mu(\mathbf{x}) = \lim_{\lambda\to 0}\tfrac{1}{\lambda}\big(\ell(\mathbf{u}+\lambda\boldsymbol{\nu}) - \ell(\mathbf{u})\big)$$

for all $\boldsymbol{\nu} \in \mathcal{F}$. Building on this, we have:

Proposition 2. *Let $\gamma \in (0, 1]$ and $k \in \mathbb{N}$. Let $P^{k,\gamma}$ and M^k denote the stochastic process sequences generated by γ-SFBD via the update rules in Eq (7) and Eq (13). Then the update of $M(\mathbf{s}^k) = M^k$ satisfies*

$$\mathbf{s}_t^{k+1}(\mathbf{x}) = \mathbf{s}_t^k(\mathbf{x}) - \gamma \nabla_{P_t^{k+1,\gamma}} \mathcal{L}_t(\mathbf{s}_t^k, m_0(\mathbf{s}^k))(\mathbf{x}) \tag{15}$$

for all $\mathbf{x} \in \mathbb{R}^d$ and $t \in [0, \tau]$.

As a result, γ-SFBD basically performs a discretized functional gradient descent on $\mathcal{L}(\mathbf{s}, M_0(\mathbf{s}))$ with step size γ, following the steepest descent under the reference distribution $P^{k,\gamma}$, updated via (14). Remarkably, Prop 1 shows that value γ does not affect convergence of $p_0^{k,\gamma}$ to p_{data}. Thus, *for any $\gamma \in (0, 1]$, $\mathbf{s}_t^k$ converges to the true score function $\mathbf{s}_t^*$ learned by a diffusion model trained on clean data, with $\gamma = 1$ recovering the original SFBD result (Lu et al., 2025).*

SFBD flow. The functional gradient descent perspective shows that as $\gamma \to 0$, the discrete sequence $\{\mathbf{s}^k\}_{k\in\mathbb{N}}$ and the associated distributions $p_0^{k,\gamma}$ converge to continuous flows $\{\mathbf{s}^\kappa\}_{\kappa\geq 0}$ and p_0^κ, governed by the gradient flow of $\mathcal{L}(\mathbf{s}, M_0(\mathbf{s}))$. We refer to this continuous formulation as *SFBD flow*.

To characterize the evolution of p_0^κ, fix $\kappa > 0$ and let $\{\gamma_i\} \to 0$ with $k_i = \kappa/\gamma_i \in \mathbb{N}$. Then $p_0^{k_i,\gamma_i} \to p_0^\kappa$ and $m_0^{k_i} \to m_0(\mathbf{s}^\kappa)$ via Euler approximation. Taking the limit,

$$\frac{\mathrm{d}}{\mathrm{d}\kappa} p_0^\kappa = \lim_{i\to\infty} \tfrac{1}{\gamma_i}\big(p_0^{k_i+1,\gamma_i} - p_0^{k_i,\gamma_i}\big) \overset{(14)}{=} m_0(\mathbf{s}^\kappa) - p_0^\kappa,$$

where $p_0^0 = p_{\mathcal{E}_{\text{clean}}}$. Thus, p_0^κ evolves according to an ODE driven by the mismatch between the model's denoised output $m_0(\mathbf{s}^\kappa)$ and the current estimate p_0^κ. Under this flow formulation, the convergence of γ-SFBD reduces to:

Corollary 1. *For $\kappa > 0$, we have $\frac{\mathrm{d}}{\mathrm{d}\kappa} D_{\mathrm{KL}}(p_{\text{data}} \,\|\, p_0^\kappa) \leq -D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^\kappa)$. Additionally,*

$$\inf_{\kappa\in[0,\mathcal{K}]} \left|\Phi_{p_{data}}(\mathbf{u}) - \Phi_{p_0^\kappa}(\mathbf{u})\right| \leq \exp\left(\tfrac{\tau}{2}\|\mathbf{u}\|^2\right)\left(\tfrac{2M}{\mathcal{K}}\right)^{1/2}$$

for $\mathcal{K} > 0$, $\mathbf{u} \in \mathbb{R}^d$ and $M = D_{\mathrm{KL}}(p_{\text{data}} \,\|\, p_{\mathcal{E}_{\text{clean}}})$.

5. Online SFBD optimization

As discussed in Sec 4, when γ is small, Prop 2 shows that the sequence $\mathbf{s}^k$ closely tracks its continuous limit $\mathbf{s}^\kappa$. Since $\mathbf{s}^k$ is parameterized by neural networks, this continuity motivates replacing iterative fine-tuning in SFBD with a single network $\mathbf{s}^\phi$ that continuously approximates the evolving $\mathbf{s}^k$. The optimization of $\mathbf{s}^\phi$ follows M-Proj Eq (7), implemented by minimizing the loss of matching score Eq (4) with $p_0 = p_0^{k,\gamma}$. Unlike standard SFBD, γ-SFBD refreshes only a fraction γ of denoised samples in each γ-D-Proj step, inducing small changes to $p_0^{k,\gamma}$ - so a few gradient steps suffice for $\mathbf{s}^\phi$ to track the new minimizer. Building on this insight, we propose *Online SFBD* in Alg 1, which eliminates the need to fine-tune a sequence of networks.

Combining denoised and clean samples. Since the copyright-free clean samples are drawn from the true data distribution, we follow the original SFBD framework (Lu et al., 2025) and set $p_0 = p_{\mathcal{E}\cup\mathcal{E}_{\text{clean}}}$ in the M-Proj step. This choice helps accelerate optimization by aligning the target distribution for updating ϕ more closely with the true data distribution. As detailed in Appx A.4, this corresponds to a variant of γ-Diff Proj, and we provide additional justification there for the observed performance gains.

Denoising and sampling. While Alg 1 uses a naive backward sampler by solving Eq (3), the algorithm allows any backward SDE and solver that yield the same marginals. We adopt the 2nd-order Heun method from EDM (Karras et al., 2022) for better error control and efficiency. To improve sample quality (Nichol and Dhariwal, 2021; Karras et al., 2022), we maintain an EMA version of the model for denoising and use it to update $\mathcal{E}$; all reported results in Sec 6 are based on this EMA model. In practice, γ is typically small (e.g., $\gamma < 0.02$), so the mild asynchrony between γ-D-Proj and M-Proj has negligible effect, as suggested by preliminary exploration during framework implementation. This motivates a practical strategy we call *asynchronous denoising*: denoising runs independently on a separate, low-performance GPU, updating $\mathcal{E}$ in the background, while the main training loop minimizes Eq (4) on high-performance hardware using the latest $p_0 = p_{\mathcal{E}}$. We adopt this strategy throughout our experiments in Sec 6.

Relationship to consistency constraint-based methods. Consistency constraint-based (CC-based) methods such as TweedieDiff (Daras et al., 2024) and TweedieDiff+ (Daras et al., 2025), which enforce consistency only between time zero and positive time steps, can be seen as special cases of Online SFBD with a single gradient step ($m = 1$). (See Appx A.5 for details and an extension to arbitrary time pairs.) These methods approximate $p^{k,\gamma}$ using m_0^k rather than the EMA over $\{m_0^j\}_{j\le k}$ as defined in Eq (14), which is not exact unless $\gamma = 1$. Since $\mathbf{s}$ is updated just once per iteration, m_0^j for j close to k tends to be similar, making m_0^k a reasonable proxy of $p^{k,\gamma}$ when γ is not too small.

Algorithm 1 Online SFBD

Input: clean data: $\mathcal{E}_{\text{clean}} = \{\mathbf{x}^{(i)}\}_{i=1}^M$, noisy data: $\mathcal{E}_{\text{noisy}} = \{\mathbf{y}_\tau^{(i)}\}_{i=1}^N$, number of gradient steps: m

```
// Initialize Denoiser
```

$\phi \leftarrow$ Pretrain D_ϕ using Eq (4) with $p_0 = p_{\mathcal{E}_{\text{clean}}}$

$\mathcal{E} \leftarrow \{\mathbf{y}_0^{(i)}$: solve Eq (3) from $t = \tau$ to 0 with $\mathbf{s}_t(\mathbf{x}_t) = \frac{D_\phi(\mathbf{x}_t,t)-\mathbf{x}_t}{t}$, $\mathbf{x}_\tau = \mathbf{y}_\tau^{(i)} \in \mathcal{E}_{\text{noisy}}\}$

repeat

  Update ϕ with m graident steps on Eq (4) with $p_0 = p_{\mathcal{E}}$. `// M-Proj`

  $\mathcal{E} \leftarrow$ {Replace ratio γ of denoised samples in $\mathcal{E}$ with the new ones by solving Eq (3) from $t = \tau$ to 0 with $\mathbf{s}_t(\mathbf{x}_t) = \frac{D_\phi(\mathbf{x}_t,t)-\mathbf{x}_t}{t}$, $\mathbf{x}_\tau = \mathbf{y}_\tau^{(i)}$ randomly picked from $\mathcal{E}_{\text{noisy}}$} `// γ-D-Proj`

until *reach the maximum number of iterations*

Output: Final denoiser D_ϕ

In Sec 6, we show that avoiding this approximation enables Online SFBD to consistently outperform CC-based methods. Remarkably, it also achieves significantly lower computational cost. This is because Online SFBD reuses cached denoised samples throughout training, whereas CC-based methods generate them on demand – requiring more samples per step for stability and making asynchronous denoising impractical. Moreover, CC-based methods typically enforce consistency between arbitrary time pairs, requiring multiple neural network forward passes per update. In contrast, Online SFBD matches the compute cost of a standard diffusion model, apart from denoised sample updates – which can be performed asynchronously on separate GPUs.

6. Empirical study

In this section, we show the effectiveness of the Online SFBD algorithm. We begin by exploring its behaviour under various configurations to identify optimal settings. Our ablation studies support the theoretical analysis and provide practical guidance for applying Online SFBD. Leveraging these findings, we benchmark Online SFBD and show that it consistently outperforms models trained on noisy data. Compared to standard SFBD, the online variant typically yields better or comparable results while *avoiding the costly iterative finetuning and denoising steps* – except in cases with very limited clean samples and severely corrupted sensitive data.

Datasets and evaluation metrics. We conduct experiments on CIFAR-10 (Krizhevsky and Hinton, 2009) and CelebA (Liu et al., 2022), using image resolutions of 32×32 and 64×64, respectively. CIFAR-10 contains 50,000 training and 10,000 test images across 10 classes. CelebA includes 162,770 training, 19,867 validation, and 19,962 test

Table 1: Comparison. For $\sigma > 0$, models are trained on images corrupted with Gaussian noise $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$.

Method	CIFAR-10 (32×32)			CelebA (64×64)		
	σ	Pretrain (50 imgs)	FID	σ	Pretrain (50 imgs)	FID
DDPM (Ho et al., 2020)	0.0	No	4.04	0.0	No	**3.26**
DDIM (Song et al., 2021a)	0.0	No	4.16	0.0	No	6.53
EDM (Karras et al., 2022)	0.0	No	**1.97**	–	–	–
EMDiff (Bai et al., 2024)	0.2	Yes	86.47	–	–	–
TweedieDiff (Daras et al., 2024)	0.2	Yes	65.21	0.2	Yes	58.52
TweedieDiff+ (Daras et al., 2025)	0.2	Yes	8.05	0.2	Yes	6.81
SFBD (Lu et al., 2025)	0.2	Yes	13.53	0.2	Yes	6.49
OSFBD (ours)	0.2	Yes	**3.22**	0.2	Yes	**3.23**

Table 2: Additional results for competitive models under various settings. (All models are pretrained.)

Method	CIFAR-10 (32×32)			CelebA (64×64)		
	σ	clean samples	FID	σ	clean samples	FID
TweedieDiff+ (Daras et al., 2025)	0.2	10%	2.81	1.38	50	35.65
SFBD (Lu et al., 2025)	0.2	10%	**2.58**	1.38	50	**23.63**
OSFBD (ours)	0.2	10%	2.73	1.38	50	27.09
TweedieDiff+ (Daras et al., 2025)	0.59	4%	6.75	1.38	1,500	6.81
SFBD (Lu et al., 2025)	0.59	4%	**6.31**	1.38	1,500	5.91
OSFBD (ours)	0.59	4%	6.56	1.38	1,500	**5.72**

images; we use the preprocessed version from the official DDIM repository (Song et al., 2021a). Corrupted images are generated by adding independent Gaussian noise with standard deviation σ to each pixel after rescaling to $[-1, 1]$. Only one noisy counterpart is generated per clean image.

Image quality is evaluated using Fréchet Inception Distance (FID), computed between the reference dataset and 50,000 model-generated samples. Generated image samples are shown in Appx B.

Models and configurations. Online SFBD (OSFBD) adopts the backbone and hyperparameters from EDM (Karras et al., 2022) in the unconditional setting, with non-leaky augmentation to mitigate overfitting. Backward sampling is performed using the 2$^{\text{nd}}$-order Heun method (Karras et al., 2022). See Appx C for details. We pretrain models on clean samples using the standard denoising score matching loss, combined with the ASM loss on corrupted data to guide score estimation for $t > \tau$. All experiments set $m = 20$.

6.1. Performance comparison

We compare OSFBD with representative models for training on noisy images, as summarized in Table 1. EMDiff (Bai et al., 2024) uses a diffusion-based EM algorithm for inverse problems. TweedieDiff (Daras et al., 2024) applies the original consistency loss from Eq (6) and is pretrained on clean data. TweedieDiff+ (Daras et al., 2025) adopts the same pretraining as OSFBD, followed by joint training with a simplified consistency objective. SFBD (Lu et al., 2025) is the original algorithm requring iteratively finetuning.

Benchmark. Following the setup of Bai et al. (2024); Lu et al. (2025), we use 50 clean samples along with data corrupted Gaussian noise ($\sigma = 0.2$), with the same clean set across all experiments. For reference, we also report results for models trained on fully clean data ($\sigma = 0$). As shown in Table 1, OSFBD consistently outperforms all baselines, producing significantly higher-quality iOSFBDmages. Notably, it even surpasses DDPM and DDIM trained exclusively on clean samples on both datasets.

To further assess the capacity of OSFBD, we evaluate it under additional dataset configurations in Table 2, alongside the two strongest baselines: TweedieDiff+ and SFBD. OSFBD consistently outperforms TweedieDiff+ and matches SFBD in most settings, except for one challenging CelebA case with very limited clean data and high noise (σ). We discuss this further below.

OSFBD vs SFBD. The results in Table 1 and Table 2 suggest that SFBD performs better in settings with very limited clean data and high noise. Specifically, both methods perform comparably when a moderate amount of clean data is available. Under low noise and limited clean data, OSFBD outperforms SFBD – likely due to ASM loss during pretraining and smoother updates of the target distribution $p^{k,\gamma}$, which help mitigate overfitting, a known issue in SFBD on small datasets (Lu et al., 2025). However, at high noise levels, denoising requires more backward SDE steps, amplifying errors from imperfect training. In such cases, more accurate score estimation is needed and would require a prohibitively large number of gradient steps m, making SFBD a more stable and effective choice. Notably, even SFBD yields high FID in these settings, indicating that the task remains difficult and potentially unsuitable for practical deployment.

OSFBD vs TweedieDiff+. We observe that OSFBD consistently outperforms TweedieDiff+ across all settings in Table 1 and Table 2, consistent with our discussion on their relationship in Sec 5. Both methods share the same pretraining procedure and differ only in how they learn the score function for $t < \tau$. By updating the denoised sample set in an EMA-like manner, OSFBD presents a significantly more accurate target distribution $p^{k,\gamma}$, leading to improved performance. Notably, this improvement also reduces computational cost – though at the expense of additional memory to cache denoised samples.

7. Discussion

This paper extends the original SFBD algorithm to a family of variants, γ-SFBD. When $\gamma = 1$, it recovers SFBD; as $\gamma \to 0$, it yields SFBD flow and its practical counterpart – Online SFBD – which eliminates the need for alternating between denoising and fine-tuning. We also highlight its close connection to CC-based methods, another class of leading diffusion-based deconvolution techniques. Empirical results corroborate our analysis, showing that Online SFBD consistently outperforms strong baselines across most benchmarks.

Acknowledgement

We gratefully acknowledge funding support from NSERC, the Canada CIFAR AI Chairs program and the Ontario Early Researcher program. Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute.

Impact Statement

Online SFBD enables training diffusion models on noisy data, supporting privacy-preserving data sharing without direct access to the originals. While it offers a mathematically grounded solution to address privacy and copyright concerns, responsible implementation is essential to mitigate risks of information leakage and overstated security.

References

B D O Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12 (3):313–326, 1982. URL https://doi.org/10.1016/0304-4149(82)90051-5.

Weimin Bai, Yifei Wang, Wenzheng Chen, and He Sun. An expectation-maximization algorithm for training clean diffusion models from corrupted observations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=jURBh4V9N4.

James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. OpenAI, 2023. URL https://cdn.openai.com/papers/dall-e-3.pdf.

Ashish Bora, Eric Price, and Alexandros G. Dimakis. AmbientGAN: Generative models from lossy measurements. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Hy7fDog0b.

Valentin De Bortoli, Iryna Korshunova, Andriy Mnih, and Arnaud Doucet. Schrodinger bridge flow for unpaired data translation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=1F32iCJFfa.

Nicolas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramer, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. In *32nd USENIX Security Symposium*, pages 5253–5270, 2023. URL https://www.usenix.org/system/files/usenixsecurity23-carlini.pdf.

R. Courant and D. Hilbert. *Methods of Mathematical Physics*. WILEY-VCH Verlag GmbH & Co. KGaA, 1989. ISBN 9783527414475. doi: 10.1002/9783527617210.

Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(9):10850–10869, 2023. URL https://doi.org/10.1109/TPAMI.2023.3261988.

Giannis Daras and Alex Dimakis. Solving inverse problems with ambient diffusion. In *NeurIPS 2023 Workshop on Deep Learning and Inverse Problems*, 2023. URL https://openreview.net/forum?id=mGwg10bgHk.

Giannis Daras, Yuval Dagan, Alex Dimakis, and Constantinos Daskalakis. Consistent diffusion models: Mitigating sampling drift by learning to be consistent. In *Advances in Neural Information Processing Systems*, pages 42038–42063, 2023a. URL https://openreview.net/forum?id=GfZGdJHj27.

Giannis Daras, Kulin Shah, Yuval Dagan, Aravind Gollakota, Alex Dimakis, and Adam Klivans. Ambient diffusion: Learning clean distributions from corrupted data. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b. URL https://openreview.net/forum?id=wBJBLy9kBY.

Giannis Daras, Alex Dimakis, and Constantinos Costis Daskalakis. Consistent diffusion meets tweedie: Training exact ambient diffusion models with noisy data. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=PlVjIGaFdH.

Giannis Daras, Yeshwanth Cherapanamjeri, and Constantinos Costis Daskalakis. How much is a noisy image worth? data scaling laws for ambient diffusion. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=qZwtPEw2qN.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. URL https://doi.org/10.1145/3422622.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Jonathan Ho, Tim Salimans, Alexey A. Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=f3zNgKga_ep.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=k7FuTOWMOc7.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference for Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=a-xFK8Ymz5J.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *Proceedings of the Eighth International Conference on Learning Representations (ICLR 2020)*, April 2020.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=XVjTT1nw5z.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. *Proceedings of International Conference on Computer Vision (ICCV)*, 2015. URL http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html.

Haoye Lu, Qifan Wu, and Yaoliang Yu. Stochastic forward-backward deconvolution: Training diffusion models with finite noisy datasets, 2025. arXiv:2502.05446.

Alexander Meister. *Deconvolution Problems in Nonparametric Statistics*. Springer, 2009. URL https://doi.org/10.1007/978-3-540-87557-4.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=-NEXDKk8gZ.

Michele Pavon and Anton Wakolbinger. On free energy, stochastic control, and Schrödinger processes. In *Modeling, Estimation and Control of Systems with Uncertainty: Proceedings of a Conference held in Sopron, Hungary, September 1990*, pages 334–348. Birkhäuser Boston, 1991. URL https://doi.org/10.1007/978-1-4612-0443-5_22.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=di52zR8xgf.

Mohammad Reza Karimi, Ya-Ping Hsieh, and Andreas Krause. Sinkhorn flow as mirror flow: A continuous-time framework for generalizing the Sinkhorn algorithm. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li, editors, *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 4186–4194. PMLR, 02–04 May 2024. URL https://proceedings.mlr.press/v238/reza-karimi24a.html.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022. URL https://doi.org/10.1109/CVPR52688.2022.01042.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference* *on Machine Learning*, pages 2256–2265, 2015. URL https://proceedings.mlr.press/v37/sohl-dickstein15.html.

Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Diffusion art or digital forgery? investigating data replication in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6048–6058, 2023. URL https://doi.org/10.1109/CVPR52729.2023.00586.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

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

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, pages 32211–32252, 2023. URL https://proceedings.mlr.press/v202/song23a.html.

Francisco Vargas, Pierre Thodoroff, Austen Lamacraft, and Neil Lawrence. Solving schrodinger bridges via maximum likelihood. *Entropy*, 23(9), 2021. URL https://www.mdpi.com/1099-4300/23/9/1134.

Dongchao Yang, Jianwei Yu, Helin Wang, Wen Wang, Chao Weng, Yuexian Zou, and Dong Yu. Diffsound: Discrete diffusion model for text-to-sound generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:1720–1733, 2023. URL https://doi.org/10.1109/TASLP.2023.3268730.

A. Theoretical results

A.1. Minimizing KL divergence is equivalent to conditional drift matching

In Sec 3, we claimed that minimizing $\tilde{\mathcal{L}}$ defined in Eq (10) is equivalent to minimizing

$$\mathcal{L}(\mathbf{s}, M^k) = \int_0^\tau \mathcal{L}_t \, \mathrm{d}t = \int_0^\tau \mathbb{E}_{D(m_0^k)} \tfrac{1}{2} \left\| \frac{\mathbf{x}_0 - \mathbf{x}_t}{t} - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \mathrm{d}t. \tag{16}$$

To see this, note that according to Eq (9), D-Proj sets $P^{k+1} = \mathrm{proj}_{\mathcal{D}} M^k = D(m_0^k)$. As a result,

$$\tilde{\mathcal{L}}(\mathbf{s}, M^k) = D_{\mathrm{KL}}(\mathrm{proj}_{\mathcal{D}} M^k \,\|\, M(\mathbf{s})) = D_{\mathrm{KL}}(D(m_0^k) \,\|\, M(\mathbf{s})).$$

By Lem 1, the KL divergence

$$D_{\mathrm{KL}}(D(m_0^k) \,\|\, M(\mathbf{s})) = \underbrace{D_{\mathrm{KL}}(m_0^k * \mathcal{N}(\mathbf{0}, \tau\mathbf{I}) \,\|\, p_\tau^*)}_{\text{const.}} + \mathbb{E}_{D(m_0^k)} \int_0^\tau \frac{1}{2} \left\| \mathbf{b}(\mathbf{x}_t, t) - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \mathrm{d}t,$$

where $\mathbf{b}^k(\mathbf{x}_t, t)$ is the drift of the backward SDE starting from τ with the initial distribution $m_0^k * \mathcal{N}(\mathbf{0}, \tau\mathbf{I})$. Anderson (1982) showed that $\mathbf{b}^k(\mathbf{x}_t, t) = \nabla \log m_t^k(\mathbf{x}_t)$, where $m_t^k(\mathbf{x}_t)$ denotes the density of the marginal distribution of M^k. It can be shown that (e.g., see (Song et al., 2023, Lemma 1)):

$$\nabla \log m_t^k(\mathbf{x}_t) = \mathbb{E}_{m_{0|t}^k}[\nabla_{\mathbf{x}_t} \log m_t^k(\mathbf{x}_t | \mathbf{x}_0) | \mathbf{x}_t] = \mathbb{E}_{m_{0|t}^k}\left[\frac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right]. \tag{17}$$

As a result,

$$\mathbb{E}_{D(m_0^k)} \int_0^\tau \frac{1}{2} \left\| \mathbf{b}(\mathbf{x}_t, t) - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \mathrm{d}t = \mathbb{E}_{D(m_0^k)} \int_0^\tau \frac{1}{2} \left\| \mathbb{E}_{m_{0|t}^k}\left[\frac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \mathrm{d}t.$$

Therefore,

$$\begin{aligned} \mathrm{argmin}_{\mathbf{s}} \tilde{\mathcal{L}}(\mathbf{s}, M^k) &= \mathrm{argmin}_{\mathbf{s}} \; \mathbb{E}_{D(m_0^k)} \int_0^\tau \frac{1}{2} \left\| \mathbf{b}(\mathbf{x}_t, t) - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \mathrm{d}t \\ &= \mathrm{argmin}_{\mathbf{s}} \mathbb{E}_{D(m_0^k)} \int_0^\tau \left\| \mathbb{E}_{m_{0|t}^k}\left[\frac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \mathrm{d}t. \end{aligned}$$

In addition, for any $t \in [0, \tau]$,

$$\begin{aligned} &\mathbb{E}_{D(m_0^k)} \left\| \tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \\ =& \mathbb{E}_{D(m_0^k)} \left\| \tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) - \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] + \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \\ =& \mathbb{E}_{D(m_0^k)} \left\| \tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) - \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] \right\|^2 + \mathbb{E}_{D(m_0^k)} \left\| \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \\ &\quad + \mathbb{E}_{D(m_0^k)} \left\langle \tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) - \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right], \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] - \mathbf{s}_t(\mathbf{x}_t) \right\rangle. \end{aligned}$$

For the last term,

$$\begin{aligned} &\mathbb{E}_{D(m_0^k)} \left\langle \tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) - \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right], \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] - \mathbf{s}_t(\mathbf{x}_t) \right\rangle \\ =& \mathbb{E}_{m_t^k} \left\langle \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] - \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right], \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] - \mathbf{s}_t(\mathbf{x}_t) \right\rangle \\ =& \mathbb{E}_{m_t^k} \left\langle \mathbf{0}, \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] - \mathbf{s}_t(\mathbf{x}_t) \right\rangle = 0. \end{aligned}$$

As a result,

$$\begin{aligned}
&\mathop{\mathbb{E}}_{D(m_0^k)} \left\| \tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \\
&= \underbrace{\mathop{\mathbb{E}}_{D(m_0^k)} \left\| \tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) - \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] \right\|^2}_{\text{Independent of } \mathbf{s} \Rightarrow \text{Const.}} + \mathop{\mathbb{E}}_{D(m_0^k)} \left\| \mathbb{E}_{m_{0|t}^k}\left[\tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] - \mathbf{s}_t(\mathbf{x}_t) \right\|^2.
\end{aligned}$$

Thus,

$$\begin{aligned}
\operatorname{argmin}_{\mathbf{s}} \tilde{\mathcal{L}}(\mathbf{s}, M^k) &= \operatorname{argmin}_{\mathbf{s}} \mathop{\mathbb{E}}_{D(m_0^k)} \int_0^\tau \left\| \mathbb{E}_{m_{0|t}^k}\left[\frac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) \middle| \mathbf{x}_t \right] - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \mathrm{d}t \\
&= \operatorname{argmin}_{\mathbf{s}} \int_0^\tau \mathop{\mathbb{E}}_{D(m_0^k)} \left\| \tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \mathrm{d}t + \text{Const.} \\
&= \operatorname{argmin}_{\mathbf{s}} \int_0^\tau \mathop{\mathbb{E}}_{D(m_0^k)} \left\| \tfrac{1}{t}(\mathbf{x}_0 - \mathbf{x}_t) - \mathbf{s}_t(\mathbf{x}_t) \right\|^2 \mathrm{d}t.
\end{aligned}$$

A.2. Optimal Solution to Eq (13)

Note that, by the disintegration theorem (e.g., see Vargas et al. 2021, Appx B),

$$\begin{aligned}
&\mathop{\operatorname{argmin}}_{P \in \mathcal{D}} (1-\gamma) D_{\mathrm{KL}}(P^{k,\gamma} \,\|\, P) + \gamma D_{\mathrm{KL}}(M^k \,\|\, P) \\
&= \mathop{\operatorname{argmin}}_{P \in \mathcal{D}} (1-\gamma) \left[D_{\mathrm{KL}}(p_0^{k,\gamma} \,\|\, p_0) + \underbrace{\mathop{\mathbb{E}}_{P^{k,\gamma}} \left[\log \frac{\mathrm{d}P^{k,\gamma}(\cdot|\mathbf{x}_0)}{\mathrm{d}P(\cdot|\mathbf{x}_0)} \right]}_{\text{Const.}} \right] \\
&\qquad + \gamma \left[D_{\mathrm{KL}}(m_0^k \,\|\, p_0) + \underbrace{\mathop{\mathbb{E}}_{M^k} \left[\log \frac{\mathrm{d}M^k(\cdot|\mathbf{x}_0)}{\mathrm{d}P(\cdot|\mathbf{x}_0)} \right]}_{\text{Const.}} \right] \\
&= \mathop{\operatorname{argmin}}_{P \in \mathcal{D}} (1-\gamma) D_{\mathrm{KL}}(p_0^{k,\gamma} \,\|\, p_0) + \gamma D_{\mathrm{KL}}(m_0^k \,\|\, p_0) \\
&= \mathop{\operatorname{argmin}}_{P \in \mathcal{D}} - \int_{\mathbb{R}^d} \left[(1-\gamma)\, p_0^{k,\gamma}(\mathbf{x}_0) + \gamma\, m_0^k(\mathbf{x}_0) \right] \log p_0(\mathbf{x}_0) \; \mathrm{d}\mathbf{x}_0 + \text{Const.} \\
&= \mathop{\operatorname{argmin}}_{P \in \mathcal{D}} D_{\mathrm{KL}}((1-\gamma) p_0^{k,\gamma} + \gamma m_0^k \,\|\, p_0).
\end{aligned}$$

As a result,

$$p_0^{k+1,\gamma} = (1-\gamma)\, p_0^{k,\gamma} + \gamma\, m_0^k. \tag{18}$$

A.3. Results related to SFBD flow

Proposition 1. *For $k \geq 0$, $D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^{k+1,\gamma}) - D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^{k,\gamma}) \leq -\gamma D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^{k,\gamma})$. In addition,*

$$\min_{k=1,\dots,K} \left| \Phi_{p_{\mathrm{data}}}(\mathbf{u}) - \Phi_{p_0^{k,\gamma}}(\mathbf{u}) \right| \leq \exp\left(\tfrac{\tau}{2} \|\mathbf{u}\|^2 \right) \left(\tfrac{2M}{\gamma K} \right)^{1/2}$$

for $K \geq 1$, $\mathbf{u} \in \mathbb{R}^d$, and $M = D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_{\mathcal{E}_{\mathrm{clean}}})$.

Proof. Let P^* denote the path measure induced by the forward process Eq (1) with $p_0 = p_{\mathrm{data}}$. In addition, let $\mathcal{F}(q) = D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, q)$. For brevity, we drop the γ in $P^{k,\gamma}$ and its marginal distributions $p_0^{k,\gamma}$ and $p_\tau^{k,\gamma}$.

Note that,

$$D_{\mathrm{KL}}(P^* \,\|\, M^k) = \mathcal{F}(m_0^k) + \underbrace{\mathbb{E}_{P^*}\left[\frac{1}{2}\int_0^\tau \|\mathbf{b}^k(\mathbf{x}_t,t)\|^2 \,\mathrm{d}t\right]}_{:=\mathcal{B}_k}, \tag{19}$$

where $\mathbf{b}^k(\mathbf{x}_t, t)$ is the drift of the forward process inducing M^k with $\mathbf{x}_0 \sim m_0^k$.

In addition, through the convexity of the KL divergence,

$$\mathcal{F}\big(p_0^{k+1}\big) = \mathcal{F}\big((1-\gamma)p_0^k + \gamma m_0^k\big) \le (1-\gamma)\mathcal{F}(p_0^k) + \gamma\mathcal{F}(m_0^k),$$

which implies,

$$\mathcal{F}(m_0^k) \ge \mathcal{F}(p_0^k) + \tfrac{1}{\gamma}\big(\mathcal{F}(p_0^{k+1}) - \mathcal{F}(p_0^k)\big). \tag{20}$$

As a result,

$$\begin{aligned}
\mathcal{F}(p_0^k) = D_{\mathrm{KL}}(P^* \,\|\, P^k) &= D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^k) + \mathbb{E}_{p^*}\left[\int_0^\tau \frac{1}{2}\big\|\nabla \log p_t(\mathbf{x}_t) - \mathbf{s}_t^k(\mathbf{x}_t)\big\|\right] \\
&= D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^k) + D_{\mathrm{KL}}(P^* \,\|\, M^k) \overset{(19)}{=} D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^k) + \mathcal{F}(m_0^k) + \mathcal{B}_k \\
&\overset{(20)}{\ge} D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^k) + \mathcal{B}_k + \tfrac{1}{\gamma}\Big(\mathcal{F}(p_0^{k+1}) - \mathcal{F}(p_0^k)\Big) + \mathcal{F}(p_0^k) \\
&\ge D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^k) + \tfrac{1}{\gamma}\Big(\mathcal{F}(p_0^{k+1}) - \mathcal{F}(p_0^k)\Big) + \mathcal{F}(p_0^k).
\end{aligned}$$

Rearrangement yields

$$D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^{k+1,\gamma}) - D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^{k,\gamma}) \le -\gamma D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^{k,\gamma}), \tag{21}$$

the monotonicity of $p_0^{k,\gamma}$ in k in the proposition. Equivalently,

$$\mathcal{F}(p_0^{k+1,\gamma}) - \mathcal{F}(p_0^{k,\gamma}) \le -\gamma D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^{k,\gamma}). \tag{22}$$

Telescoping it yields:

$$\mathcal{F}(p_0^{0,\gamma}) = \sum_{k=0}^{K} \mathcal{F}(p_0^{k,\gamma}) - \mathcal{F}(p_0^{k+1,\gamma}) \ge \gamma \sum_{k=1}^{K} D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^{k,\gamma}). \tag{23}$$

Thus,

$$\min_{k\in\{1,2,\dots,K\}} D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^{k,\gamma}) \le \frac{\mathcal{F}(p_0^{0,\gamma})}{\gamma K} = \frac{\mathcal{F}(p_{\mathcal{E}_{\mathrm{clean}}})}{\gamma K}. \tag{24}$$

Applying Prop 3, we get

$$\min_{k\in\{1,2,\dots,K\}} \Big|\Phi_{p_{\mathrm{data}}}(\mathbf{u}) - \Phi_{p_0^{k,\gamma}}(\mathbf{u})\Big| \le \exp\big(\frac{\tau}{2}\|\mathbf{u}\|^2\big)\left(\frac{2D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_{\mathcal{E}_{\mathrm{clean}}})}{\gamma K}\right)^{1/2}. \tag{25}$$

□

Proposition 2. *Let $\gamma \in (0,1]$ and $k \in \mathbb{N}$. Let $P^{k,\gamma}$ and M^k denote the stochastic process sequences generated by γ-SFBD via the update rules in Eq (7) and Eq (13). Then the update of $M(\mathbf{s}^k) = M^k$ satisfies*

$$\mathbf{s}_t^{k+1}(\mathbf{x}) = \mathbf{s}_t^k(\mathbf{x}) - \gamma\nabla_{P_t^{k+1,\gamma}}\mathcal{L}_t(\mathbf{s}_t^k, m_0(\mathbf{s}^k))(\mathbf{x}) \tag{15}$$

for all $\mathbf{x} \in \mathbb{R}^d$ and $t \in [0,\tau]$.

Proof. For $t \in [0, \tau]$, let $\boldsymbol{\phi}$ be a function of the same function space as $\mathbf{s}_t^k$ and p_0 the density of a distribution defined on $\mathbb{R}^d$. Then for $\epsilon \in (0, 1]$, we have

$$\begin{aligned}
\mathcal{L}_t(\mathbf{s}_t + \epsilon\boldsymbol{\phi}, p_0) &= \mathop{\mathbb{E}}_{D(p_0)}\left[\frac{1}{2}\left\|\frac{\mathbf{x}_0 - \mathbf{x}_t}{t} - (\mathbf{s}_t + \epsilon\boldsymbol{\phi})(\mathbf{x}_t)\right\|^2\right] \\
&= \mathcal{L}_t(\mathbf{x}_t, p_0) + \epsilon \mathop{\mathbb{E}}_{D(p_0)}\left[\left\langle \mathbf{s}_t(\mathbf{x}_t) - \frac{\mathbf{x}_0 - \mathbf{x}_t}{t}, \boldsymbol{\phi}(\mathbf{x}_t)\right\rangle\right] + o(\epsilon) \\
&= \mathcal{L}_t(\mathbf{x}_t, p_0) + \epsilon\left\langle \boldsymbol{\phi}(\mathbf{x}_t), \left(\mathbf{s}_t(\mathbf{x}_t) - \frac{\mathbf{x}_0 - \mathbf{x}_t}{t}\right) \mathrm{d}P_{0t}(\mathbf{x}_0, \mathbf{x}_t)\right\rangle \\
&= \mathcal{L}_t(\mathbf{x}_t, p_0) + \epsilon\left\langle \boldsymbol{\phi}(\mathbf{x}_t), \left(\mathbf{s}_t(\mathbf{x}_t) - \frac{\mathbf{x}_0 - \mathbf{x}_t}{t}\right) \mathrm{d}P_{0t}(\mathbf{x}_0, \mathbf{x}_t)\right\rangle \\
&= \mathcal{L}_t(\mathbf{x}_t, p_0) + \epsilon\left\langle \boldsymbol{\phi}(\mathbf{x}_t), \left(\mathbf{s}_t(\mathbf{x}_t) - \frac{\mathbb{E}_{P_{0|t}}[\mathbf{x}_0|\mathbf{x}_t] - \mathbf{x}_t}{t}\right) dP_t(\mathbf{x}_t)\right\rangle.
\end{aligned}$$

As a result,

$$\nabla_\mu \mathcal{L}(\mathbf{s}_t, p_0)(\mathbf{x}_t) = \left(\mathbf{s}_t(\mathbf{x}_t) - \frac{\mathbb{E}_{P_{0|t}}[\mathbf{x}_0|\mathbf{x}_t] - \mathbf{x}_t}{t}\right)\frac{\mathrm{d}P_t}{\mathrm{d}\mu}(\mathbf{x}_t). \tag{26}$$

We note that when $k = 0$, $\mathbf{s}_t^0(\mathbf{x}_t)$ is pretrained on $P_{\mathcal{E}_{\text{clean}}}$. As a result, by e.g., (Song et al., 2023, Lemma 1),

$$\mathbf{s}_t^0(\mathbf{x}_t) = \frac{\mathbb{E}_{(P_{\mathcal{E}_{\text{clean}}})_{0|t}}(\mathbf{x}_0|\mathbf{x}_t) - \mathbf{x}_t}{t} = \frac{\mathbb{E}_{P_{0|t}^{0,\gamma}}(\mathbf{x}_0|\mathbf{x}_t) - \mathbf{x}_t}{t} \tag{27}$$

for any $t \in [0, \tau]$ and $\mathbf{x}_t \in \mathbb{R}^d$, which is the negative backward drift of M^0 in γ-SFBD.

Then assume that for $k \in \mathbb{N}$, for any $t \in [0, \tau]$ and $\mathbf{x}_t \in \mathbb{R}^d$, we have

$$\mathbf{s}_t^k(\mathbf{x}_t) = \frac{\mathbb{E}_{P_{0|t}^{k,\gamma}}(\mathbf{x}_0|\mathbf{x}_t) - \mathbf{x}_t}{t}, \tag{28}$$

correpsonding to the negative backward drift of M^k in γ-SFBD.

Then for $k + 1$, Eq (15) gives

$$\begin{aligned}
\mathbf{s}_t^{k+1}(\mathbf{x}_t) &= \mathbf{s}_t^k(\mathbf{x}_t) - \gamma\, \nabla_{P_t^{k+1,\gamma}} \mathcal{L}_t(\mathbf{s}_t^k, m_0(\mathbf{s}^k))(\mathbf{x}_t) \\
&\overset{(26)}{=} \mathbf{s}_t^k(\mathbf{x}_t) - \gamma\left(\mathbf{s}_t^k(\mathbf{x}_t) - \frac{\mathbb{E}_{D(m_0^k)_{0|t}}[\mathbf{x}_0|\mathbf{x}_t] - \mathbf{x}_t}{t}\right)\frac{\mathrm{d}D(m_0^k)_t}{\mathrm{d}P_t^{k+1,\gamma}}(\mathbf{x}_t) \\
&= \big(1 - \delta(\mathbf{x}_t)\big)\, \mathbf{s}_t^k(\mathbf{x}_t) + \delta(\mathbf{x}_t)\, \frac{\mathbb{E}_{D(m_0^k)_{0|t}}[\mathbf{x}_0|\mathbf{x}_t] - \mathbf{x}_t}{t},
\end{aligned} \tag{29}$$

where $\delta(\mathbf{x}_t) = \gamma \frac{\mathrm{d}D(m_0^k)_t}{\mathrm{d}P_t^{k+1,\gamma}}(\mathbf{x}_t)$. We note that, by Eq (14),

$$p_0^{k+1,\gamma} = (1 - \gamma)\, p_0^{k,\gamma} + \gamma\, m_0^k. \tag{30}$$

As a result,

$$P_t^{k+1,\gamma} = (1 - \gamma)\, P_t^{k,\gamma} + \gamma\, D(m_0^k)_t \tag{31}$$

and

$$\delta(\mathbf{x}_t) = \frac{\gamma\, \mathrm{d}D(m_0^k)_t}{\mathrm{d}(1 - \gamma)\, P_t^{k,\gamma} + \gamma\, D(m_0^k)_t}(\mathbf{x}_t),\ 1 - \delta(\mathbf{x}_t) = \frac{(1 - \gamma)\, \mathrm{d}P_t^{k,\gamma}}{\mathrm{d}(1 - \gamma)\, P_t^{k,\gamma} + \gamma\, D(m_0^k)_t}(\mathbf{x}_t). \tag{32}$$

Thus,

$$\begin{aligned}
\mathbf{s}_t^{k+1}(\mathbf{x}_t) &\overset{(29)}{=} \mathbf{s}_t^k(\mathbf{x}_t)\, \frac{(1-\gamma)\,\mathrm{d}P_t^{k,\gamma}}{\mathrm{d}(1-\gamma)\,P_t^{k,\gamma} + \gamma\, D(m_0^k)_t}(\mathbf{x}_t) \\
&\quad + \frac{\mathbb{E}_{D(m_0^k)_{0|t}}[\mathbf{x}_0|\mathbf{x}_t] - \mathbf{x}_t}{t}\, \frac{\gamma\,\mathrm{d}D(m_0^k)_t}{\mathrm{d}(1-\gamma)\,P_t^{k,\gamma} + \gamma\, D(m_0^k)_t}(\mathbf{x}_t) \\
&\overset{(28)}{=} \frac{\mathbb{E}_{P_{0|t}^{k,\gamma}}(\mathbf{x}_0|\mathbf{x}_t) - \mathbf{x}_t}{t}\, \frac{(1-\gamma)\,\mathrm{d}P_t^{k,\gamma}}{\mathrm{d}(1-\gamma)\,P_t^{k,\gamma} + \gamma\, D(m_0^k)_t}(\mathbf{x}_t) \\
&\quad + \frac{\mathbb{E}_{D(m_0^k)_{0|t}}[\mathbf{x}_0|\mathbf{x}_t] - \mathbf{x}_t}{t}\, \frac{\gamma\,\mathrm{d}D(m_0^k)_t}{\mathrm{d}(1-\gamma)\,P_t^{k,\gamma} + \gamma\, D(m_0^k)_t}(\mathbf{x}_t) \\
&= -\frac{1}{t}\mathbf{x}_t + \frac{1}{t}\int_{\mathbf{x}_0'\in\mathbb{R}^d} \mathbf{x}_0'\, \frac{\mathrm{d}\,(1-\gamma)P_{0t}^{k,\gamma} + \gamma D(m_0^k)_{0t}}{\mathrm{d}\,(1-\gamma)\,P_t^{k,\gamma} + \gamma\, D(m_0^k)_t}(\mathbf{x}_0', \mathbf{x}_t) \\
&= -\frac{1}{t}\mathbf{x}_t + \frac{1}{t}\int_{\mathbf{x}_0'\in\mathbb{R}^d} \mathbf{x}_0'\, \frac{\mathrm{d}\,P_{0t}^{k+1,\gamma}}{\mathrm{d}\,P_t^{k+1,\gamma}}(\mathbf{x}_0', \mathbf{x}_t) \\
&= \frac{\mathbb{E}_{P_{0|t}^{k+1,\gamma}}(\mathbf{x}_0|\mathbf{x}_t) - \mathbf{x}_t}{t},
\end{aligned}$$

which is the negative backward drift of M^{k+1}. □

Corollary 1. *For $\kappa > 0$, we have $\frac{\mathrm{d}}{\mathrm{d}\kappa} D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^\kappa) \leq -D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^\kappa)$. Additionally,*

$$\inf_{\kappa\in[0,\mathcal{K}]} \left|\Phi_{p_{data}}(\mathbf{u}) - \Phi_{p_0^\kappa}(\mathbf{u})\right| \leq \exp\left(\tfrac{\tau}{2}\|\mathbf{u}\|^2\right) \left(\tfrac{2M}{\mathcal{K}}\right)^{1/2}$$

for $\mathcal{K} > 0$, $\mathbf{u} \in \mathbb{R}^d$ and $M = D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_{\mathcal{E}_{\mathrm{clean}}})$.

Proof. According to Eq (21), we have

$$\frac{1}{\gamma}\left(D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^{k+1,\gamma}) - D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^{k,\gamma})\right) \leq -D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^{k,\gamma}), \tag{33}$$

for all $\gamma > 0$ and $k \in \mathbb{N}$.

Fix $\kappa > 0$ and let $\{\gamma_i\} \to 0$ with $k_i = \kappa/\gamma_i \in \mathbb{N}$. Then $p_0^{k_i,\gamma_i} \to p_0^\kappa$ via Euler approximation. Taking the limit yields:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\kappa} D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^\kappa) &= \lim_{i\to\infty} \frac{1}{\gamma_i}\left(D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^{k_i+1,\gamma_i}) - D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^{k_i,\gamma_i})\right) \\
&\overset{(33)}{\leq} \lim_{i\to\infty} -D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^{k_i,\gamma_i}) = -D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^\kappa),
\end{aligned} \tag{34}$$

establishing the monotonicity claim.

In addition, integrating both sides of Eq (34) over $[0, \mathcal{K}]$ gives:

$$D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^{\mathcal{K}}) - D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^0) \leq -\int_0^{\mathcal{K}} D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^\kappa)\,\mathrm{d}\kappa. \tag{35}$$

As a result,

$$\inf_{\kappa\in[0,\mathcal{K}]} D_{\mathrm{KL}}(p_\tau^* \,\|\, p_\tau^\kappa) \leq \frac{1}{\mathcal{K}} D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_0^0) = \frac{1}{\mathcal{K}} D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_{\mathcal{E}_{\mathrm{clean}}}).$$

Applying Prop 3 concludes the convergence argument in the corollary. □

A.4. A variant of γ-SFBD

Since the copyright-free clean samples are drawn from the true data distribution, it is practical to mix them with the denoised samples during denoiser updates to enhance overall sample quality. In particular, we generally believe that

$$\mathcal{L}_{\text{vis}}(\alpha\, p_{\text{clean}} + (1-\alpha)\, p_{\text{denoise}}) \leq \mathcal{L}_{\text{vis}}(p_{\text{denoise}}), \tag{36}$$

where $\mathcal{L}_{\text{vis}}(p)$ denotes an aggregate loss that measures the visual quality of samples drawn from distribution p, and p_{clean} and p_{denoise} represent the distributions of clean and denoised samples, respectively. α depends on the ratios between the numbers of clean samples and the denoised samples. In practice, we have observed that this is always true when $\mathcal{L}_{\text{vis}}$ is instantiated as the FID.

We note that this heuristic technique can be naturally covered in our framework with little work. In particular, we can replace the original γ-Diff Proj with

$$(\gamma\text{-Diff Proj-v2}) \quad P^{k+1,\gamma} = \operatorname{argmin}_{P\in\mathcal{D}}\ \alpha\, D_{\text{KL}}(P_{\text{clean}} \,\|\, P) + (1-\alpha)\left[(1-\gamma)\, D_{\text{KL}}(P^{k,\gamma} \,\|\, P) + \gamma\, D_{\text{KL}}(M^k \,\|\, P)\right]$$

where $P_{\text{clean}} = D(p_{\text{clean}})$ is a fixed diffusion process in $\mathcal{D}$ with the initial distribution having density p_{clean}.

Applying a derivation similar to the one in Appx A.2, again through the disintegration theorem, we have

$$\begin{aligned}
&\arg\min_{P\in\mathcal{D}}\ \alpha\, D_{\text{KL}}(P_{\text{clean}} \,\|\, P) + (1-\alpha)\Big[(1-\gamma)\, D_{\text{KL}}(P^{k,\gamma} \,\|\, P) + \gamma\, D_{\text{KL}}(M^k \,\|\, P)\Big] \\
=&\arg\min_{P\in\mathcal{D}}\ \alpha\, D_{\text{KL}}(p_{\text{clean}} \,\|\, p_0) + (1-\alpha)\Big[(1-\gamma)\, D_{\text{KL}}(p_0^{k,\gamma} \,\|\, p_0) + \gamma\, D_{\text{KL}}(m_0^k \,\|\, p_0)\Big] + \text{Const.} \\
=&\operatorname{argmin}_{P\in\mathcal{D}}\ -\int_{\mathbb{R}^d} \alpha p_{\text{clean}}(\mathbf{x}_0) + (1-\alpha)\ \Big[(1-\gamma)\, p_0^{k,\gamma}(\mathbf{x}_0) + \gamma\, m_0^k(\mathbf{x}_0)\Big] \log p_0(\mathbf{x}_0)\ \mathrm{d}\mathbf{x}_0 + \text{Const.} \\
=&\arg\min_{P\in\mathcal{D}}\ D_{\text{KL}}(\alpha p_{\text{clean}} + (1-\alpha)\big[(1-\gamma)\, p_0^{k,\gamma} + \gamma\, m_0^k\big] \,\|\, p_0).
\end{aligned}$$

As a result,

$$\begin{aligned}
\tilde{p}_0^{k+1,\gamma} &= \alpha\, p_{\text{clean}} + (1-\alpha)\ \Big[(1-\gamma)\, p_0^{k,\gamma} + \gamma\, m_0^k\Big] \\
&= \alpha\, p_{\text{clean}} + (1-\alpha)\, p_0^{k+1,\gamma},
\end{aligned}$$

where $p_0^{k+1,\gamma}$ is obtained via the standard γ-D-Proj defined in Eq (13), and corresponds to p_{denoise} in Eq (36).

This variant of γ-D-Proj therefore recovers the exact update rule underlying the heuristic practice of mixing clean and denoised samples prior to fine-tuning the diffusion model.

Notably, when $\gamma = 1$, this variant reduces to a form of the original SFBD algorithm, which was heuristically employed in the initial SFBD paper (Lu et al., 2025) to boost model performance—despite lacking theoretical justification at the time.

A.5. Relationship to consistency constraint-based methods

In Sec 5, we argued that consistency-constraint-based (CC-based) methods enforcing consistency only between $r = 0$ and a positive time s can be viewed as a special case of Online SFBD with $m = 1$ with $p^{k,\gamma}$ approximated by m_0^k. In this section, we elaborate on this connection and extend the discussion to more general CC-based methods that enforce consistency between arbitrary time steps $r < s$.

Enforcing consistency between $r = 0$ and $s > 0$. We assume the denoiser network satisfies $D_{\boldsymbol{\phi}}(\cdot, 0) = \text{Id}(\cdot)$, a condition explicitly enforced in EDM-based implementations. This design is both natural and intuitive, as $D_{\boldsymbol{\phi}}(\mathbf{x}_0, 0)$ approximates $\mathbb{E}_{p_{0|t}}[\mathbf{x}_0 \mid \mathbf{x}_0]$ at $t = 0$, which is $\mathbf{x}_0$ for any p_{0t} induced by some distribution p_0 argumented by the forward transition kernel $p_{t|0}$ in Eq (2). It has been adopted in all CC-based methods (Daras et al., 2024; 2025), the SFBD framework (Lu et al., 2025), and our work.

Lu et al. (2025) showed that, under this assumption, the denoising loss in Eq (4) is equivalent to the consistency loss Eq (6).

For completeness, we include their derivation as follows:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{p_0}\mathop{\mathbb{E}}_{p_{s|0}}\left[\|D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbf{x}_0\|^2\right]=\mathbb{E}_{p_s}\mathbb{E}_{p_{0|s}}\left[\|D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbf{x}_0\|^2\right]\\
=&\,\mathbb{E}_{p_s}\mathbb{E}_{p_{0|s}}\left[\|D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbb{E}_{p_{0|s}}[\mathbf{x}_0|\mathbf{x}_s]+\mathbb{E}_{p_{0|s}}[\mathbf{x}_0|\mathbf{x}_s]-\mathbf{x}_0\|^2\right]\\
=&\,\mathbb{E}_{p_s}\mathbb{E}_{p_{0|s}}\left[\|D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbb{E}_{p_{0|s}}[\mathbf{x}_0|\mathbf{x}_s]\|^2\right]+\underbrace{\mathbb{E}_{p_s}\mathbb{E}_{p_{0|s}}\left[\|\mathbb{E}_{p_{0|s}}[\mathbf{x}_0|\mathbf{x}_s]-\mathbf{x}_0\|^2\right]}_{\text{Const.}}\\
&\qquad\qquad\qquad+2\underbrace{\mathbb{E}_{p_s}\mathbb{E}_{p_{0|s}}\left[\left\langle D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbb{E}_{p_{0|s}}[\mathbf{x}_0|\mathbf{x}_s],\mathbb{E}_{p_{0|s}}[\mathbf{x}_0|\mathbf{x}_s]-\mathbf{x}_0\right\rangle\right]}_{=0}\\
=&\,\mathbb{E}_{p_s}\left[\|D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbb{E}_{p_{0|s}}[\mathbf{x}_0|\mathbf{x}_s]\|^2\right]+\text{Const.} &(37)\\
\overset{\text{(arch ass)}}{=}&\,\mathbb{E}_{p_s}\left[\|D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbb{E}_{p_{0|s}}[D_{\boldsymbol{\phi}}(\mathbf{x}_0,0)]\|^2\right]+\text{Const.},
\end{aligned}$$

which is the consistency loss in Eq (6) when $r=0$. Therefore,

$$\mathop{\mathbb{E}}_{p_0}\mathop{\mathbb{E}}_{p_{s|0}}\left[\|D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbf{x}_0\|^2\right]\equiv\mathbb{E}_{p_s}\left[\|D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbb{E}_{p_{0|s}}[D_{\boldsymbol{\phi}}(\mathbf{x}_0,0)]\|^2\right] \tag{38}$$

up to a constant, establishing the equivalence between the denoising loss used in Alg 1 (M-proj) and the consistency loss in CC-based methods.

For Online SFBD, at the k-th iteration, we have

$$p_0=p_0^{k+1,\gamma}=(1-\gamma)\,p_0^{k,\gamma}+\gamma\,m_0^k, \tag{39}$$

as presented in Eq (14), and p_s is

$$p_s=p_0*\mathcal{N}(0,s\mathbf{I})=p_0^{k,\gamma}*\mathcal{N}(0,s\mathbf{I}). \tag{40}$$

Instead, in CC-based methods,

$$p_0\approx m_0^k. \tag{41}$$

To see this, note that practical implementations of CC-based methods typically rely on two approximations:

(a) p_s is approximated using samples generated by adding Gaussian noise to corrupted data, where s is chosen no less than the corruption level τ (Daras et al., 2025);

(b) $p_{0|s}$ is estimated via the backward SDE Eq (3), with the drift term approximated by the current network (i.e., $\mathbf{s}^k$).

For simplicity, we restrict the discussion to the case $s=\tau$. For the cases $s>\tau$, they reduce to the case $s=\tau$ under the assumption that the score function above τ is accurately estimated, which can be achieved by training the model through the ASM loss combined with the noisy samples (Daras et al., 2024; 2025). These approximations essentially define the backward SDE process M^k, whose marginal at $t=0$ is m_0^k, serving as the effective p_0 in CC-based training.

Note that CC-based methods form $(\mathbf{x}_0,\mathbf{x}_s)$ pairs using the backward SDE, whereas Online SFBD uses the forward process. As CC-based methods assume that corrupted samples can be approximated as drawn from p_s, the two pairing schemes are equivalent: both the forward and backward SDE yield the same joint distribution $p_{0s}(\mathbf{x}_0,\mathbf{x}_s)$, as discussed following Eq (3).

This approximation is reasonable when m_0^k evolves slowly and γ is not too small, as discussed in the main text. This condition typically holds in practice, as CC-based methods only take one gradient step per iteration. Moreover, CC-based methods often adopt the same pretraining strategy as OSFBD, allowing the network to learn global structure early on. As a result, drift updates during subsequent training are small, and m_0^k changes slightly across iterations.

Enforcing consistency between $r<s$. For any pair $r<s$, we note that

$$\begin{aligned}
&\mathbb{E}_{p_s}\left[\|D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbb{E}_{p_{0|s}}[D_{\boldsymbol{\phi}}(\mathbf{x}_0,0)]\|^2\right]\\
\overset{\text{(arch ass)}}{=}&\,\mathbb{E}_{p_s}\left[\|D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbb{E}_{p_{0|s}}[\mathbf{x}_0|\mathbf{x}_s]\|^2\right]\\
=&\,\mathbb{E}_{p_s}\left[\left\|D_{\boldsymbol{\phi}}(\mathbf{x}_s,s)-\mathbb{E}_{p_{r|s}}\left[\mathbb{E}_{p_{0|r}}[\mathbf{x}_0|\mathbf{x}_r]\middle|\mathbf{x}_s\right]\right\|^2\right], &(42)
\end{aligned}$$

where the final equality uses the fact that the backward process is Markovian. In more detail, since the forward process is Brownian and thus Markovian, its time reversal (the backward process described by Eq (3)) is also Markovian. Consequently, we can justify:

$$\begin{aligned}\mathbb{E}_{p_{r|s}}\left[\mathbb{E}_{p_{0|r}}[\mathbf{x}_0|\mathbf{x}_r]\big|\mathbf{x}_s\right] &= \int \mathbf{x}_0 \left(\int p_{0|r}(\mathbf{x}_0|\mathbf{x}_r)\, p_{r|s}(\mathbf{x}_r|\mathbf{x}_s)\ \mathrm{d}\mathbf{x}_r\right) \mathrm{d}\mathbf{x}_0 \\ &= \int \mathbf{x}_0 \left(\int p_{0r|s}(\mathbf{x}_0, \mathbf{x}_r|\mathbf{x}_s)\ \mathrm{d}\mathbf{x}_r\right) \mathrm{d}\mathbf{x}_0 \\ &= \int \mathbf{x}_0\, p_{0|s}(\mathbf{x}_0|\mathbf{x}_s)\ \mathrm{d}\mathbf{x}_0 \\ &= \mathbb{E}_{p_{0|s}}[\mathbf{x}_0|\mathbf{x}_s].\end{aligned}$$

As a result, by Eq (42), we have

$$\begin{aligned}\mathcal{L}_{\text{con}}(\boldsymbol{\phi}, 0, s) &= \mathbb{E}_{p_s}\left[\|D_{\boldsymbol{\phi}}(\mathbf{x}_s, s) - \mathbb{E}_{p_{0|s}}[D_{\boldsymbol{\phi}}(\mathbf{x}_0, 0)]\|^2\right] \\ &= \mathbb{E}_{p_s}\left[\left\|D_{\boldsymbol{\phi}}(\mathbf{x}_s, s) - \mathbb{E}_{p_{r|s}}\left[\mathbb{E}_{p_{0|r}}[\mathbf{x}_0|\mathbf{x}_r]\big|\mathbf{x}_s\right]\right\|^2\right] \\ &\approx \mathbb{E}_{p_s}\left[\left\|D_{\boldsymbol{\phi}}(\mathbf{x}_s, s) - \mathbb{E}_{p_{r|s}}\left[D_{\mathsf{stopgrad}(\boldsymbol{\phi})}(\mathbf{x}_r, r)\big|\mathbf{x}_s\right]\right\|^2\right] \\ &= \mathcal{L}_{\text{con}}(\boldsymbol{\phi}, r, s)\end{aligned}$$

where $\mathbb{E}_{p_{0|r}}[\mathbf{x}_0|\mathbf{x}_r]$ is approximated using the current network, and $\mathsf{stopgrad}$ indicates a stop-gradient operation.

This suggests that enforcing consistency between arbitrary time pairs $r < s$ is effectively equivalent to enforcing it between 0 and s, so the same argument for $r = 0$ applies.

A.6. Auxiliary results for references

Proposition 3 (Lu et al. 2025, Prop 1)**.** *Let p and q be two distributions defined on $\mathbb{R}^d$. For all $\mathbf{u} \in \mathbb{R}^d$,*

$$|\Phi_p(\mathbf{u}) - \Phi_q(\mathbf{u})| \le \exp\left(\frac{\tau}{2}\|\mathbf{u}\|^2\right)\sqrt{2\, D_{\mathrm{KL}}(p * h \| q * h)},$$

where $h \sim \mathcal{N}(\mathbf{0}, \tau\mathbf{I})$.

Lemma 1 (Pavon and Wakolbinger 1991, Vargas et al. 2021)**.** *Given two SDEs:*

$$\mathrm{d}\mathbf{x}_t = \mathbf{f}_i(\mathbf{x}_t, t)\,\mathrm{d}t + \delta\,\mathrm{d}\mathbf{w}_t, \quad \mathbf{x}_0 \sim p_0^{(i)}(\mathbf{x}) \quad t \in [0, T] \tag{43}$$

for $i = 1, 2$. Let $P_{0:T}^{(i)}$, for $i = 1, 2$, be the path measure induced by them, respectively. Then we have,

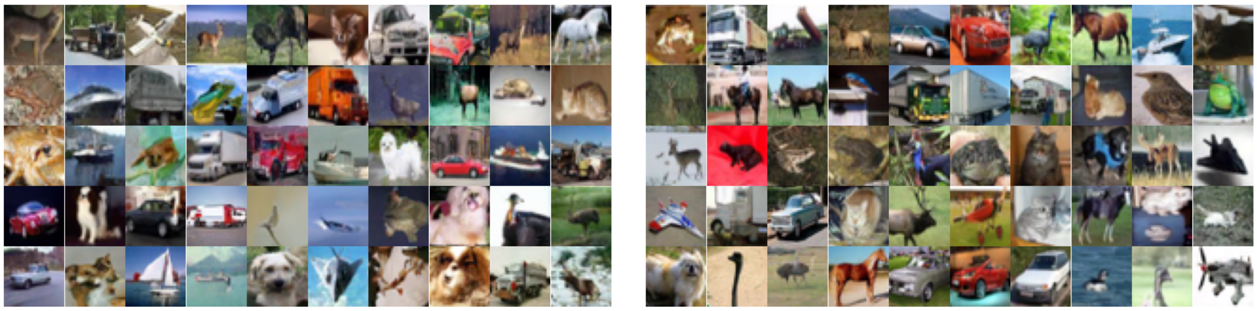
$$D_{\mathrm{KL}}(P_{0:T}^{(1)} \,\|\, P_{0:T}^{(2)}) = D_{\mathrm{KL}}(p_0^{(1)} \,\|\, p_0^{(2)}) \ + \ \mathbb{E}_{P_{0:T}^{(1)}}\Big[\int_0^T \frac{1}{2}\,\|\mathbf{f}_1(\mathbf{x}_t, t) - \mathbf{f}_2(\mathbf{x}_t, t)\|^2\, dt\Big]. \tag{44}$$

A similar result also applies to a pair of backward SDEs as well, where $p_0^{(i)}$ is replaced with $p_T^{(i)}$.

B. Sampling results

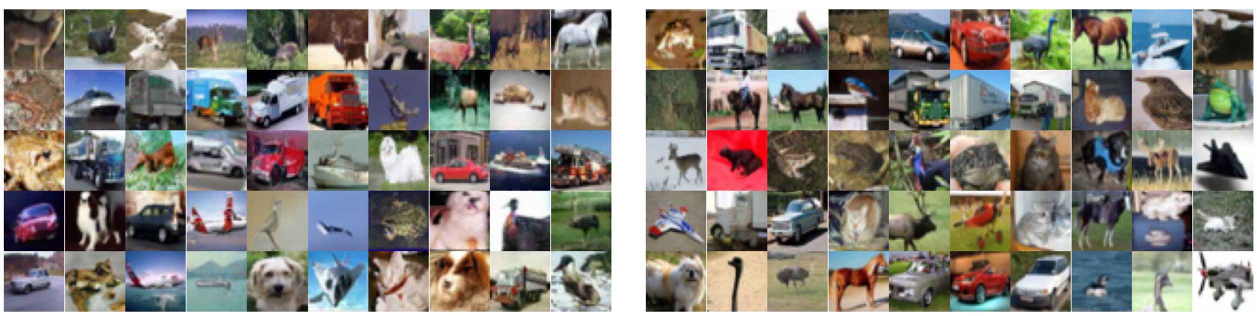
In this section, we present model-generated samples used to compute FID scores in Sec 6, corresponding to the benchmark results in Table 1 and Table 2. We also include denoised samples at the final training step.

B.1. CIFAR-10



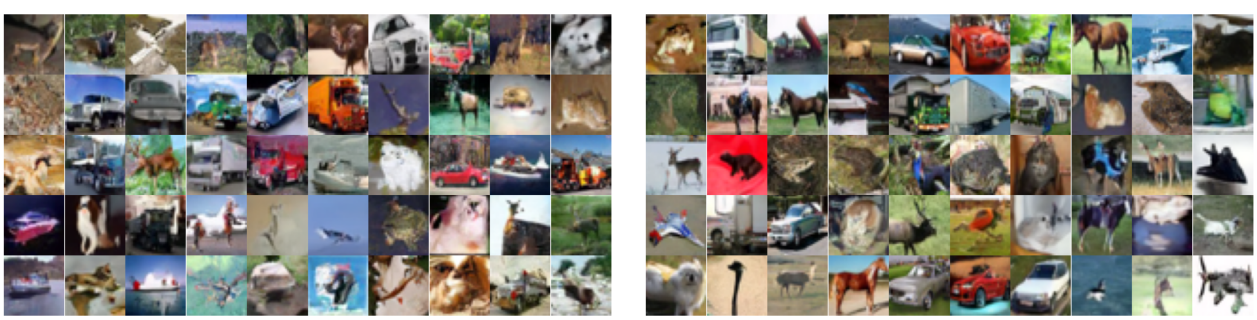
(a) Generated (FID: 3.22) (b) Denoised (FID: 1.11)

Figure 1: 50 clean samples, noise level $\sigma = 0.2$



(a) Generated (FID: 2.73) (b) Denoised (FID: 1.02)

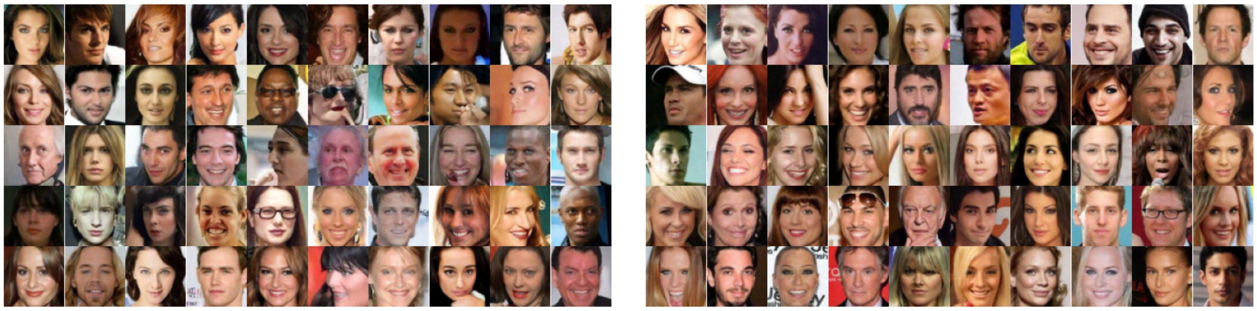
Figure 2: 5,000 clean samples (10%), noise level $\sigma = 0.2$.



(a) Generated (FID: 6.56) (b) Denoised (FID: 4.84)

Figure 3: 2,000 clean samples (4%), noise level $\sigma = 0.59$.

B.2. CelebA



(a) Generated (FID: 3.23)

(b) Denoised (FID: 1.07)

Figure 4: 50 clean samples, noise level $\sigma = 0.2$.

(a) Generated (FID: 27.09)

(b) Denoised (FID: 24.31)

Figure 5: 50 clean samples, noise level $\sigma = 1.38$.

(a) Generated (FID: 5.72)

(b) Denoised (FID: 4.28)

Figure 6: 1,500 clean samples, noise level $\sigma = 0.2$.

C. Experiment configurations

C.1. Hardware configurations

All diffusion models were trained on the main process using four NVIDIA A40 or RTX 6000 GPUs, managed by a SLURM scheduling system. The asynchronous denoising process ran concurrently in the background on a separate RTX 6000 GPU, taking less than 2.5 minutes to update 640 images on CIFAR-10 and under 5 minutes on CelebA.

Training on CIFAR-10 completes in under 5 days, and CelebA experiments in under 8 days.

C.2. Model architectures

We implement the proposed Online SFBD algorithm using the EDM backbone (Karras et al., 2022), following the configuration described below throughout our empirical studies.

Table 3: Experimental Configuration for CIFAR-10 and CelebA

Parameter	CIFAR-10	CelebA
General		
Batch Size	512	256
Loss Function	`EDMLoss` (Karras et al., 2022)	`EDMLoss` (Karras et al., 2022)
Denoising Method	2^{nd} order Heun method (EDM) (Karras et al., 2022)	2^{nd} order Heun method (EDM) (Karras et al., 2022)
Sampling Method	2^{nd} order Heun method (EDM) (Karras et al., 2022)	2^{nd} order Heun method (EDM) (Karras et al., 2022)
Sampling steps	18	40
Network Configuration		
Dropout	0.13	0.05
Channel Multipliers	$\{2, 2, 2\}$	$\{1, 2, 2, 2\}$
Model Channels	128	128
Resample Filter	$\{1, 1\}$	$\{1, 3, 3, 1\}$
Channel Mult Noise	1	2
Optimizer Configuration		
Optimizer Class	`RAdam` (Kingma and Ba, 2015; Liu et al., 2020)	`RAdam` (Kingma and Ba, 2015; Liu et al., 2020)
Learning Rate	0.001	0.0002
Epsilon	1×10^{-8}	1×10^{-8}
Betas	(0.9, 0.999)	(0.9, 0.999)

C.3. Datasets

All experiments on CIFAR-10 (Krizhevsky and Hinton, 2009) and CelebA (Liu et al., 2015) are conducted using only the training set. For FID evaluation, the model generates 50,000 samples, and FID is computed against the full training set, which includes both copyright-free and sensitive samples.