Rectified Noise: A Generative Model Using Positive-incentive Noise

Zhenyu Gu[1,2*], **Yanchen Xu**[1,3*], **Sida Huang**[1,3*], **Yubin Guo**[1,4], **Hongyuan Zhang**[1,5†],

[1]Institute of Artificial Intelligence (TeleAI), China Telecom
[2]Research Institute of Intelligent Complex Systems, Fudan University
[3]School of Artificial Intelligence, OPtics and ElectroNics (iOPEN), Northwestern Polytechnical University
[4]University of Science and Technology of China
[5]The University of Hong Kong

Abstract

Rectified Flow (RF) has been widely used as an effective generative model. Although RF is primarily based on probability flow Ordinary Differential Equations (ODE), recent studies have shown that injecting noise through reverse-time Stochastic Differential Equations (SDE) for sampling can achieve superior generative performance. Inspired by Positive-incentive Noise (π-noise), we propose an innovative generative algorithm to train π-noise generators, namely Rectified Noise (ΔRN), which improves the generative performance by injecting π-noise into the velocity field of pre-trained RF models. After introducing the Rectified Noise pipeline, pre-trained RF models can be efficiently transformed into π-noise generators. We validate Rectified Noise by conducting extensive experiments across various model architectures on different datasets. Notably, we find that: (1) RF models using Rectified Noise reduce FID from **10.16 to 9.05** on ImageNet-1k. (2) The models of π-noise generators achieve improved performance with only **0.39%** additional training parameters.

1 Introduction

Flow Matching (FM) (Lipman et al. 2022; Albergo and Vanden-Eijnden 2022; Liu, Gong, and Liu 2022) for generative models trains continuous normalizing flows (Papamakarios et al. 2021) by regressing ideal probability flow fields that connect a base distribution to the data distribution. FM models show superior performance and has seen widespread adoption in modern generative modeling (Esser et al. 2024; Polyak et al. 2024; Fu et al. 2025). Rectified Flow (RF) (Liu, Gong, and Liu 2022) is a specific kind of FM that simplifies the training objective by prescribing a straight-line path between the source and target distributions. Different from diffusion models relying on reverse-time Stochastic Differential Equation (SDE) (Song et al. 2020) , RF directly learns the velocity field that transforms an analytic distribution into the target data distribution without introducing additional stochasticity. Training RF models through a simple regression-based objective enables more stable and efficient training.

However, recent studies have demonstrated that introducing stochastic noise to a pretrained RF model during sampling, specifically via a reverse-time SDE (Ma et al. 2024) can improve performance metrics like Fréchet Inception Distance (FID) (Heusel et al. 2017). This phenomenon inspires us to investigate:

(1) What kind of stochasitic noise can lead to performance gain for RF?

(2) How to introduce the beneficial noise to RF?

Inspired by Positive-incentive Noise (π-noise) (Li 2022; Zhang et al. 2025; Huang, Zhang, and Li 2025), it may be a reliable scheme to apply π-noise framework to learn the beneficial noise for improving the performance of RF models. Specifically, the goal of π-noise is to find the beneficial noise by maximizing the mutual information (Shannon 1948) between the task and noise. The existing works have shown that π-noise can be used to effectively enhance both the classical neural networks (Zhang et al. 2025, 2024; Huang et al. 2025a; Jiang et al. 2025) and vision-language models (Huang, Zhang, and Li 2025; Huang et al. 2025b; Wang, Zhang, and Yuan 2025). The success of applying π-noise to enhance model performance adds to the rationality to apply π-noise to generative models as stated above. In this paper, we propose Rectified Noise (Δ RN), a novel framework that leverages learned π-noise to enhance the performance of RF models. The contributions can be briefly summarized as follows:

- Under the π-noise framework, we measure the complexity of RF by designing an auxiliary Gaussian distribution related to RF loss. The auxiliary Gaussian variable connects RF and information entropy.
- Motivated by the connection between π-noise and RF, we propose a π-noise generator to automatically learn the additional noise component when solving the velocity in RF. We further design a framework to convert pre-trained RF models into π-noise generators.
- Experiments on multiple datasets including ImageNet, AFHQ and CelebA-HQ validate the effectiveness of our proposed ΔRN. Our experiments show this framework achieves performance improvements while maintaining computational efficiency. ΔRN enhances RF performance across all datasets, reducing FID by up to **1.11**

*These authors contributed equally.
†Corresponding author.

Figure 1: **Image results of RF models using ΔRN**. Sampling with ΔRN improves natural image generation. The images without red highlight show the generation of the standard RF model. The images outlined in red present the result using ΔRN. Here we show comparisons between images generated by SiT models trained on ImageNet-1k (256 × 256) and SiT models using ΔRN.

on ImageNet, **1.89** on AFHQ and **3.52** on CelebA-HQ.

2 Related Work

In this section, we will discuss the related work about generative models, Scalable Interpolant Transformers (SiT)(Ma et al. 2024), and π-noise, respectively.

Generative Model

Diffusion models (Sohl-Dickstein et al. 2015; Song and Ermon 2019; Ho, Jain, and Abbeel 2020) have been developed into a highly successful framework for generative modeling. These models progressively add noise to clean data and train a neural network to reverse this process. Flow Matching (FM) (Lipman et al. 2022; Albergo and Vanden-Eijnden 2022; Liu, Gong, and Liu 2022) methods extend this framework trains continuous normalizing flows (Papamakarios et al. 2021) by regressing ideal probability flow fields that connect a base distribution to the data distribution.

Rectified Flow (RF) offers an efficient alternative in generative modeling. It directly parameterizes continuous-time transport, greatly reducing sampling steps. Unlike diffusion models using separate score estimation (Song and Ermon 2019; Vahdat, Kreis, and Kautz 2021) and noise, RF learns a clear, deterministic map between data and latent distributions with probability flow ODEs (Chen et al. 2018; Papamakarios et al. 2021; Zheng et al. 2023). This direct method simplifies generation by avoiding noisy and iterative steps and usually improves training. RF uses a straight-line sample pairing strategy to define a simple and consistent path between two distributions and uses reflow to cut the high computational cost of diffusion sampling while keeping high-quality image generation with fewer steps.

Scalable Interpolant Transformers

SiT represents a novel family of generative models, building upon the foundation of Diffusion Transformers (DiT) (Peebles and Xie 2023). SiT is an extension of Vision Transformer (ViT) (Dosovitskiy et al. 2020) that operates within the stochastic interpolant framework (Albergo, Boffi, and Vanden-Eijnden 2023). Its primary contributions include a systematic exploration of the design space for generative models, encompassing aspects like time discretization, model prediction, interpolants and samplers. This systematic approach has not only facilitated a modular study of each component but also led to the discovery of optimal practices for enhancing generation performance. SiT also explores the performance gains brought by the interpolant framework under Classifier-Free Guidance (CFG) (Ho and Salimans 2022).

Furthermore, a key finding of SiT's research pertains to the use of reverse-time SDE for sampling of flow matching models. Using reverse-time SDE sampling (Song et al. 2020) often leads to better performance compared to probability flow ODE sampling (Song and Ermon 2019).

Positive-incentive Noise

π-noise introduces an information-theoretic framework to formally claim that noise may not always be harmful. π-noise can be seen as a type of information gain brought by noise. This approach proposes learning the π-noise by maximizing the mutual information between the task and the noise. To optimize the intractable loss of π-noise, VPN (Zhang et al. 2025) proposed to optimize its variational bound and PiNI (Huang, Zhang, and Li 2025) extended it to vision-language models. With the variational inference, a VPN generator is designed for enhancing base models and simplifying the inference without changing the architecture

of base models.

3 Preliminaries

In this section, we provide a brief overview of RF model from the perspective of stochastic interpolants (Albergo, Boffi, and Vanden-Eijnden 2023) and revisit the π-noise framework.

Rectified Flow

RF aims to learn a transport map from a standard Gaussian noise distribution $\mathcal{N}(0, I)$ to an arbitrary distribution $q(\mathbf{x}_*)$ defined on the reals. Specifically, the goal is to gradually transform an initial noise sample $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$ over time into data $\mathbf{x}_* \sim q(\mathbf{x}_*)$ for the generating task. Stochastic interpolants define this transformation as a time-dependent stochastic process, which can be summarized as

$$\mathbf{x_t} = t\mathbf{x}_* + (1-t)\boldsymbol{\epsilon}, \tag{1}$$

RF models interpolate between noise and data over a finite time interval defined on $t \in [0, 1]$. Sampling from these models can be achieved via a probability flow ordinary differential equation (**Probability Flow ODE**) with a velocity field

$$\dot{\mathbf{x}}_t = \mathbf{v}(\mathbf{x}_t, t), \tag{2}$$

where the velocity field $\mathbf{v}(\mathbf{x}_t, t)$ is given by the conditional expectation

$$\mathbf{v}(\mathbf{x}, t) = \mathbb{E}[\dot{\mathbf{x}}_t | \mathbf{x}_t = \mathbf{x}], \tag{3}$$

where $\mathbf{v}(\mathbf{x}_t, t)$ signifies the expected direction of all transport paths between the noise and $p(\mathbf{x})$ that cross through $\mathbf{x}_t$ at time t. We can estimate $\mathbf{v}(\mathbf{x}_t, t)$ via the loss

$$\mathcal{L}_{\mathbf{velocity}}(\theta) := \mathbb{E}_{\mathbf{x}_*, \boldsymbol{\epsilon}, t}\left[\|\mathbf{v}_\theta(\mathbf{x}_t, t) - \mathbf{x}_* + \boldsymbol{\epsilon}\|^2\right]. \tag{4}$$

Reverse-time SDE (Tzen and Raginsky 2019; Vargas et al. 2021; De Bortoli et al. 2021; Song et al. 2020) offers an alternative sampling method for flow matching, which can be expressed as

$$\mathrm{d}\mathbf{x}_t = \mathbf{v}(\mathbf{x}_t, t)\mathrm{d}t - \frac{1}{2} w_t \mathbf{s}(\mathbf{x}_t, t)\mathrm{d}t + \sqrt{w_t}\mathrm{d}\bar{\mathbf{W}}_t, \tag{5}$$

where $\bar{W}_t$ is a reverse-time Wiener process, $w_t > 0$ denotes an arbitrary time-dependent diffusion coefficient, $\mathbf{v}(\mathbf{x}_t, t)$ refers to the velocity and $\mathbf{s}(\mathbf{x}_t, t) = \nabla \log q_t(\mathbf{x}_t)$ is identified as the score, which is also determined by a conditional expectation

$$\begin{aligned} \mathbf{s}(\mathbf{x}_t, t) &= -\frac{\mathbb{E}[\boldsymbol{\epsilon} \mid \mathbf{x}_t = \mathbf{x}]}{1-t} \\ &= \frac{\mathbf{x}_* - \mathbf{x}_t}{(1-t)t^2}. \end{aligned} \tag{6}$$

Originally, reverse-time SDE were used in score-based diffusion models, where the diffusion coefficient w_t typically depended on the forward SDE (Song et al. 2020; Chen 2023; Singhal, Goldstein, and Ranganath 2023). The stochastic interpolant framework provides greater flexibility by decoupling the formulation of $\mathbf{x}_t$ from the forward SDE, which allows for a wider choice of w_t: any $w_t \geq 0$ can be used. It's noteworthy that the choice of w_t can be made after training, as it does not impact the velocity $\mathbf{v}(\mathbf{x}, t)$ or the score $\mathbf{s}(\mathbf{x}, t)$.

Formulation of π-Noise

π-noise is primarily studied from an information-theoretic perspective. The goal of π-noise is to find the beneficial noise by maximizing the mutual information $\max_{\mathcal{E}} \mathrm{MI}(\mathcal{T}, \mathcal{E})$ between the task and noise. The principle of learning π-noise is formulated as

$$\max_{\mathcal{E}} \mathrm{MI}(\mathcal{T}, \mathcal{E}) = H(\mathcal{T}) - H(\mathcal{T}|\mathcal{E}) \Leftrightarrow \max_{\mathcal{E}} -H(\mathcal{T}|\mathcal{E}), \tag{7}$$

where $H(\cdot)$ represents the information entropy.

The task entropy $H(\mathcal{T})$ is the core of this framework. To measure the difficulty of a RF learning task for a given dataset $\mathcal{D}$ sampled from $p(\mathbf{x})$, it is essential to properly define a random variable for the task. Building on this, we can further derive the expression for $H(\mathcal{T}|\mathcal{E})$.

4 Rectified Noise

In this section, we elaborate on the proposed Rectified Noise, a novel approach for injecting π-noise into the velocity of pre-trained RF models. We first demonstrate how to define the task entropy of RF models. Subsequently, we demonstrate how to learn π-noise distribution under this definition. Finally, we propose two optimization strategies for the objective of learning π-noise and design Rectified Noise pipeline to transform RF models into π-noise generators.

Formulate Task Entropy via RF Loss

Measuring the learning complexity of RF models across diverse datasets is a challenge problem. Therefore, we concentrate on measuring this complexity by defining the task entropy on a given dataset.

Considering a given distribution $q(\mathbf{x}_*)$, the value of loss $\mathcal{L}_{\mathbf{velocity}}(\psi^*)$ can serve as a measure of generation task difficulty for RF, where ψ^* represents the optimal parameters of the neural network model. To simplify the derivation, let $\mathbf{x} = (\mathbf{x}_*, \mathbf{x_0}) \sim p(\mathbf{x})$ where $\mathbf{x}_* \sim q(\mathbf{x}_*)$ and $\mathbf{x_0} \sim \mathcal{N}(0, I)$ and let

$$\begin{aligned} \mathcal{L}_{\mathbf{velocity}}(\psi^*) &= \mathbb{E}_{\mathbf{x},t}\mathcal{L}_(\mathbf{x}, t; \psi^*) \\ &= \mathbb{E}_{\mathbf{x},t}\left[\|\mathbf{v}_{\psi^*}(\mathbf{x}_t, t) - \mathbf{x}_* + \mathbf{x_0}\|^2\right]. \end{aligned} \tag{8}$$

The smaller the value of $\mathcal{L}(\mathbf{x})$ is, the easier it is for the neural network to fit the velocity field generated by interpolating the data pair $\mathbf{x} = (\mathbf{x}_*, \mathbf{x_0})$ and vice versa.

To bridge the framework of π-noise and the complexity metric $\mathcal{L}(\mathbf{x}, t; \psi^*)$, we introduce an auxiliary random variable α , satisfying

$$\alpha|\mathbf{x}, t \sim \mathcal{N}(0, \exp(\mathcal{L}(\mathbf{x}, t; \psi^*))). \tag{9}$$

The information entropy of the auxiliary distribution $p(\alpha|\mathbf{x})$ reflects the difficulty for the corresponding generative model parameterized by ψ^*. Consequently, for a given distribution $q(\mathbf{x}_*)$, the task entropy of the generation task $\mathcal{T}$ can be written as

$$\begin{aligned} H(\mathcal{T}) &= \mathbb{E}_{\mathbf{x},t} H(p(\alpha|\mathbf{x}, t)) \\ &= \frac{1}{2}\mathbb{E}_{\mathbf{x},t}\mathcal{L}(\mathbf{x}, t; \psi^*) + \frac{1}{2}\ln(2\pi e). \end{aligned} \tag{10}$$

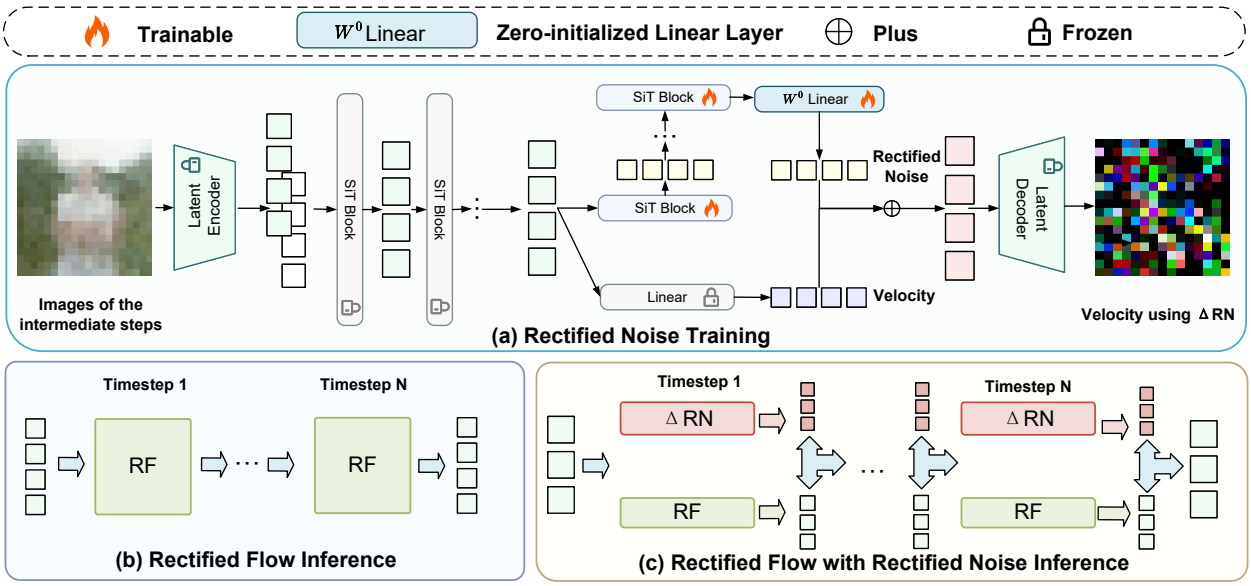

Figure 2: **Overview of Rectified Noise pipeline. (a)** The Rectified Noise model inherits pre-trained knowledge from a foundation model (RF) through parameter freezing. Additional and trainable SiT blocks are integrated to predict π-noise. **(b)** Inference of traditional RF models. **(c)** Inference with the Rectified Noise involves adding π-noise to the predicted velocity field.

Inject π-Noise to RF Model

Now, we show how we learn π-noise for a RF model by uncovering the connection between π-noise and RF. With the definition of $H(\mathcal{T})$ given in the previous subsection, the mutual information between the generation task $\mathcal{T}$ and the noise $\mathcal{E}$ can be calculated as

$$\begin{aligned}\mathrm{MI}(\mathcal{T}|\mathcal{E}) &= \mathbb{E}_{\mathbf{x},t}\int p(\alpha,\boldsymbol{\epsilon}|\mathbf{x},t)\log\frac{p(\alpha,\boldsymbol{\epsilon}|\mathbf{x},t)}{p(\alpha|\mathbf{x},t)p(\boldsymbol{\epsilon}|\mathbf{x},t)}d\alpha d\boldsymbol{\epsilon}\\ &= \int p(\alpha,\boldsymbol{\epsilon},\mathbf{x},t)\log\frac{p(\alpha,\boldsymbol{\epsilon}|\mathbf{x},t)}{p(\alpha|\mathbf{x},t)p(\boldsymbol{\epsilon}|\mathbf{x},t)}d\mathbf{x}d\boldsymbol{\epsilon}d\alpha dt.\end{aligned} \tag{11}$$

Similarly, the formulation of the conditional entropy can be written as $H(\mathcal{T}|\mathcal{E}) =$

$$-\int p(\alpha|\mathbf{x},\boldsymbol{\epsilon},t)p(\boldsymbol{\epsilon}|\mathbf{x},t)p(\mathbf{x},t)\log p(\alpha|\mathbf{x},\boldsymbol{\epsilon},t)d\mathbf{x}d\boldsymbol{\epsilon}d\alpha dt. \tag{12}$$

As shown in Eq. (7), maximizing mutual information can be achieved by minimizing conditional entropy. We consider the dataset $\mathcal{D}$ sampling from the joint distribution of $(\mathbf{x},t)$. Using Monte Carlo method, $H(\mathcal{T}|\mathcal{E})$ can be approximately written as $H(\mathcal{T}|\mathcal{E}) \approx$

$$-\frac{1}{|\mathcal{D}|}\sum_{(\mathbf{x},t)\in\mathcal{D}}\int p(\alpha|\mathbf{x},\boldsymbol{\epsilon},t)p(\boldsymbol{\epsilon}|\mathbf{x},t)\log p(\alpha|\mathbf{x},\boldsymbol{\epsilon},t)d\alpha d\boldsymbol{\epsilon}. \tag{13}$$

The formulation on the right involves two probabilities: $p(\alpha|\mathbf{x},\boldsymbol{\epsilon},t)$ and $p(\boldsymbol{\epsilon}|\mathbf{x},t)$. We learn $p(\boldsymbol{\epsilon}|\mathbf{x},t)$ as the distribution of π-noise with learnable parameters, making it essential to accurately model $p(\alpha|\mathbf{x},\boldsymbol{\epsilon},t)$. Then, we can define the auxiliary distribution with $\boldsymbol{\epsilon}$ as

$$\alpha|\mathbf{x},\boldsymbol{\epsilon},t \sim \mathcal{N}(0,\exp(\mathcal{L}(\mathbf{x},\boldsymbol{\epsilon},t;\psi^*))), \tag{14}$$

where

$$\mathcal{L}(\mathbf{x},\boldsymbol{\epsilon},t,\psi^*) = \|\mathbf{v}_{\psi^*}+\boldsymbol{\epsilon}(\mathbf{x}_t,t)-\mathbf{x}_*+\mathbf{x_0}\|^2. \tag{15}$$

The above formula is equivalent to injecting noise into the velocity field of a pre-trained RF model.

It should be pointed out that the optimization objective will be completely equivalent to RF model if we employ a point estimation of $p(\boldsymbol{\epsilon}|\mathbf{x},t)$ for a given $(\mathbf{x},t)$, i.e.,

$$p(\boldsymbol{\epsilon}|\mathbf{x},t) \to \delta(\boldsymbol{\epsilon}), \tag{16}$$

where $\delta(\boldsymbol{\epsilon})$ denotes the Dirac delta function, which satisfies

$$\delta(\boldsymbol{\epsilon}) = \begin{cases}\infty & \text{if } \boldsymbol{\epsilon}=0\\ 0 & \text{if } \boldsymbol{\epsilon}\neq 0\end{cases} \quad \text{and} \quad \int_{-\infty}^{\infty}\delta(\boldsymbol{\epsilon})\mathrm{d}\boldsymbol{\epsilon}=1. \tag{17}$$

With the point estimation, $H(\mathcal{T}|\mathcal{E})$ can be simplified as

$$\begin{aligned}&H(\mathcal{T}|\mathcal{E})\\ &\approx -\frac{1}{|\mathcal{D}|}\sum_{(\mathbf{x},t)\in\mathcal{D}}\int p(\alpha|\mathbf{x},\boldsymbol{\epsilon}=\mathbf{0},t)\log p(\alpha|\mathbf{x},\boldsymbol{\epsilon}=\mathbf{0},t)d\alpha\\ &= -\frac{1}{2}\ln(2\pi e)-\mathbb{E}_{\mathbf{x},t}\mathcal{L}(\mathbf{x},\boldsymbol{\epsilon}=\mathbf{0},t;\psi^*),\end{aligned} \tag{18}$$

which is equivalent to the loss of RF models. The estimation indicates that π-noise always keeps 0 in RF models. Accordingly, we can **learn the π-noise, instead of simply estimating it**, and get ΔRN as

$$\begin{aligned}&\max -\frac{1}{|\mathcal{D}|}\sum_{(\mathbf{x},t)\in\mathcal{D}}\int p(\alpha,\boldsymbol{\epsilon}|\mathbf{x},t)\log p(\alpha|\mathbf{x},\boldsymbol{\epsilon},t)d\alpha\\ &\Leftrightarrow \max -\frac{1}{|\mathcal{D}|}\mathbb{E}_{\boldsymbol{\epsilon}}\sum_{(\mathbf{x},t)\in\mathcal{D}}\int p(\alpha|\mathbf{x},\boldsymbol{\epsilon},t)\log p(\alpha|\mathbf{x},\boldsymbol{\epsilon},t)d\alpha\\ &\Leftrightarrow \max \mathbb{E}_{\boldsymbol{\epsilon},\mathbf{x},t}\mathcal{L}(\mathbf{x},\boldsymbol{\epsilon},t;\psi^*).\end{aligned} \tag{19}$$

Given a specific RF model (with ψ^* being optimal parameters), $\boldsymbol{\epsilon}$ is determined by $\mathbf{x}$ and t. We use a neural network parameterized by θ to represent $\boldsymbol{\epsilon}$, denoted as $\boldsymbol{\epsilon}_\theta$. Leveraging the aforementioned derivation, the optimization objective for the π-noise can be equivalently formulated as

$$\max_{\mathcal{E}} \mathrm{MI}(\mathcal{T}, \mathcal{E}) \Leftrightarrow \max_{\theta} \mathbb{E}_{\mathbf{x},t,\boldsymbol{\epsilon}\sim\boldsymbol{\epsilon}_\theta} \mathcal{L}(\mathbf{x}, \boldsymbol{\epsilon}, t; \psi^*). \tag{20}$$

Optimization Strategies for π-Noise

To optimize the objective $\max_\theta \mathbb{E}_{\mathbf{x},t,\boldsymbol{\epsilon}\sim\boldsymbol{\epsilon}_\theta} \mathcal{L}(\mathbf{x}, \boldsymbol{\epsilon}_\theta, t; \psi^*)$ proposed in the previous section, we discuss the two cases in applications: (1) Train ΔRN and RF simultaneously (i.e., learning θ and ψ); (2) Only train ΔRN for a pre-trained RF model (i.e., learn θ with frozen ψ^*).

4.3.1 Optimize both θ and ψ

θ and ψ are optimized simultaneously. This can be achieved by adjusting the assumption on the distribution of $\boldsymbol{\epsilon}$, thereby unifying parameters θ and ψ into a single neural network.

To facilitate predictions by the neural network model, we select three common reparameterizable distributions as the assumed distributions for π-noise:

- **Gaussian Distribution**

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I) \tag{21}$$

- **Gumbel Distribution**

$$\mathbf{z} = \boldsymbol{\mu} - \boldsymbol{\beta} \odot \log(-\log(\boldsymbol{\epsilon})), \quad \epsilon_i \sim U(0,1) \tag{22}$$

- **Uniform Distribution**

$$\mathbf{z} = \mathbf{a} + (\mathbf{b} - \mathbf{a}) \odot \boldsymbol{\epsilon}, \quad \epsilon_i \sim U(0,1) \tag{23}$$

where $\odot$ is Hadamard product operator and $U(0,1)$ represents a uniform distribution over the interval [0,1].

Taking the Gaussian distribution as an example, we will explain how to achieve the unification of θ and ψ^*. Leveraging the reparameterization trick, the initial optimization objective is reformulated as

$$\begin{aligned}
&\mathbb{E}_{\mathbf{x},t,\boldsymbol{\epsilon}\sim\boldsymbol{\epsilon}_\theta} \mathcal{L}(\mathbf{x}, \boldsymbol{\epsilon}_\theta, t; \psi^*) \\
=&\mathbb{E}_{\mathbf{x},t,\boldsymbol{\epsilon}\sim\boldsymbol{\epsilon}_\theta} \left\| \mathbf{v}_{\psi^*} + \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) - \mathbf{x}_* + \mathbf{x_0} \right\|^2 \\
=&\mathbb{E}_{\mathbf{x},t,\boldsymbol{\epsilon}\sim\boldsymbol{\epsilon}_\theta} \left\| \mathbf{v}_{\psi^*} + \boldsymbol{\mu}_\theta(\mathbf{x}_t, t) + \boldsymbol{\sigma}_\theta(\mathbf{x}_t, t) \odot \boldsymbol{\epsilon} - \mathbf{x}_* + \mathbf{x_0} \right\|^2,
\end{aligned} \tag{24}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$. Since $\mathbf{v}_{\psi^*}$ can essentially be regarded as a constant determined by $\mathbf{x}_t$ and t, $\hat{\boldsymbol{\mu}}_\theta = \mathbf{v}_{\psi^*}(\mathbf{x}_t, t) + \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)$ can be treated as a single entity and predicted by a single neural network. Algorithm 1 illustrates the implementation of an arbitrary batch step. Optimization of Gumbel distribution and uniform distribution is similar to the optimization of Gaussian distribution.

4.3.2 Optimize θ with frozen ψ^*

Initially, we learn the RF parameters ψ^*. Following this, θ is optimized while ψ^* is frozen. By fine-tuning the pre-trained RF neural network, we can more efficiently learn the parameter θ.

We fine-tune the pre-trained RF model using the strategy shown in Figure 2 (a). We extract the pre-trained RF model's features before the linear layer and use them as input for the π-noise generator. We stack additional SiT blocks, which then connect to a final linear layer to predict the π-noise. The linear layer is initialized with zeros to ensure that the initial prediction matches the original RF model output. Algorithm 2 illustrates the implementation of an arbitrary batch step.

Algorithm 1: Pseudo code for batch step of optimizing θ without pre-trained RF model

Input: A model $\boldsymbol{\epsilon}_\theta = \boldsymbol{\mu}_\theta + \boldsymbol{\sigma}_\theta \odot \boldsymbol{\epsilon}$, batch of N flow examples $F = \{(\mathbf{x_i}, \mathbf{y_i})\}$ where $(\mathbf{x_i}, \mathbf{y_i}) \sim p(\mathbf{x})$, learning rate β
Output: Updated parameters θ

```
Let L(θ) = 0.
for i in range(N) do
    t ~ U(0,1), x_t = t x_i + (1 − t) y_i
    Sample ε ~ N(0, I)
    # Depending on the noise assumption,
    # the distribution of ε can be adjusted.
    v̂ = μ_θ(x_t, t) + σ_θ(x_t, t) ⊙ ε, v = x_i − y_i
    L(θ) += ||v̂ − v||^2
end for
θ ← θ − (β/N) ∇_θ L(θ)
```

Algorithm 2: Pseudo code for batch step of optimizing θ with pre-trained RF model

Input:A pre-trained RF model $\mathbf{v}_{\psi^*} = \boldsymbol{f}_{\psi^*} \circ \mathbf{s}_{\psi^*}$ ($\boldsymbol{f}_{\psi^*}$ is linear layer and $\mathbf{s}_{\psi^*}$ are sit blocks function), a model $\boldsymbol{\epsilon}_\theta(\mathbf{s}_{\psi^*}(\mathbf{x}_t, t)) = \boldsymbol{\mu}_\theta + \boldsymbol{\sigma}_\theta \odot \boldsymbol{\epsilon}$, batch of N flow examples $F = \{(\mathbf{x}_i, \mathbf{y_i})\}$ where $(\mathbf{x_i}, \mathbf{y_i}) \sim p(\mathbf{x})$, learning rate β
Output: Updated parameters θ

```
Let L(θ) = 0
for i in range(N) do
    t ~ U(0,1), x_t = t x_i + (1 − t) y_i
    ŝ = s_ψ*(x_t, t)
    Sample ε ~ N(0, I)
    # Depending on the noise assumption,
    # the distribution of ε can be adjusted.
    v̂ = f_ψ*(ŝ) + μ_θ(ŝ) + σ_θ(ŝ) ⊙ ε, v = x_i − y_i
    L(θ) += ||v̂ − v||^2
end for
θ ← θ − (β/N) ∇_θ L(θ)
```

5 Experiments

In this section, we design experiments to investigate the following questions:

Q1 Does employing ΔRN lead to an improvement in RF model performance?

Q2 Which reparameterizable distribution is most suitable for modeling the π-noise distribution?

Q3 Which optimization strategy is more suitable: simultaneous optimization of θ and ψ, or optimizing ψ first and then θ?

Experimental Setup

Implementation Details

We strictly follow the setup in SiT (Ma et al., 2024a)

Dataset	Setting	Rectified Noise Setting	Extra SiT Block	Ratio of Added Parameters	Metrics				
					FID ↓	IS ↑	sFID ↓	Prec. ↑	Rec. ↑
ImageNet-1k	SiT-XL/2	-	-	-	10.16	123.86	12.02	0.50	**0.62**
	+ Δ RN	$\mathcal{N}(\mathbf{0}, \sum)$	0	0.39%	9.72	122.21	12.02	0.51	0.61
			1	3.93%	9.85	124.40	11.63	0.51	0.61
			2	7.48%	9.75	130.21	11.28	0.52	0.61
			4	14.56%	9.60	131.19	11.18	**0.53**	0.62
		$\mathcal{N}(\boldsymbol{\mu}, \sum)$	0	0.39%	9.06	130.21	**11.18**	0.52	0.61
			1	3.93%	**9.05**	**132.10**	11.23	0.52	**0.62**
			2	7.48%	9.08	129.58	11.31	0.52	**0.62**
			4	14.56%	9.15	131.43	11.26	0.52	**0.62**
AFHQ	SiT-B/2	-	-	-	12.33	9.99	28.14	0.55	0.53
	+ Δ RN	$\mathcal{N}(\mathbf{0}, \sum)$	0	0.93%	12.20	**10.13**	28.19	0.56	**0.54**
			1	9.17%	11.98	10.06	27.99	0.55	**0.54**
			2	17.41%	12.03	10.01	28.10	0.55	**0.54**
		$\mathcal{N}(\boldsymbol{\mu}, \sum)$	0	0.93%	10.62	**10.13**	26.68	0.57	**0.54**
			1	9.17%	10.52	9.88	**26.32**	0.57	0.52
			2	17.41%	**10.44**	9.80	26.41	**0.58**	0.52
CelebA-HQ	SiT-B/2	-	-	-	11.25	3.55	18.31	0.62	0.47
	+ Δ RN	$\mathcal{N}(\mathbf{0}, \sum)$	0	0.93%	11.18	**3.55**	18.27	0.62	**0.48**
			1	9.17%	11.16	3.54	18.20	0.62	**0.48**
			2	17.41%	11.15	3.54	18.26	0.62	**0.48**
		$\mathcal{N}(\boldsymbol{\mu}, \sum)$	0	0.93%	**7.73**	3.37	**14.73**	0.70	0.45
			1	9.17%	7.75	3.39	14.78	**0.71**	0.45
			2	17.41%	7.74	3.38	14.74	0.70	0.45

Table 1: **Evaluation of Rectified Noise.** The performance of generative models using Rectified Noise on the different dataset at a resolution of 256x256 without Classifier-Free Guidance (CFG), evaluated under different rectified noise settings. ↑ indicates that higher values are better, with ↓ denoting the opposite.

unless otherwise specified. We use linear interpolation to align with the RF optimization objective. We use ImageNet (1.28 million images, 1,000 categories) (Deng et al. 2009), AFHQ (16,130 images of animal faces, 3 categories) (Kim et al. 2019), and CelebA-HQ (30,000 images of celebrity facial images, 2 categories) (Karras et al. 2017) as training datasets. The model's input for all datasets is 256x256. Each image is then encoded into a compressed vector $z \in R^{32\times32\times4}$ using the Stable Diffusion VAE (Rombach et al. 2022). For model configurations, we use the B/2 and XL/2 architectures introduced in the DiT papers, which process inputs with a patch size of 2.

Evaluation Protocol

To comprehensively evaluate image generation quality across multiple dimensions, we employ a rigorous set of quantitative metrics, all computed on a standardized set of generated samples to ensure statistical reliability. For ImageNet, we use 50k generated samples to compute FID for assessing realism, structural FID (sFID) (Nash et al. 2021) for evaluating spatial coherence and Inception Score (IS) (Salimans et al. 2016) for measuring class-conditional diversity, as well as precision (Prec.) for quantifying sample fidelity and recall (Rec.) (Kynkäänniemi et al. 2019) for evaluating coverage of the target distribution. For AFHQ and CelebA-HQ, we generated 15k images and 30k images for evaluation. All evaluations are performed using the SDE Euler–Maruyama solve with 100 steps. The generated images from the standard SiT model and the model using ΔRN are shown in Figure 1. We also visualized the generated π-noise over time, as shown in Figure 3.

Rectified Noise Improves SiT

For the ImageNet dataset, a pre-trained SIT model iterated for 6 million steps was utilized to train π-noise genartor. The AFHQ dataset used a SIT model pre-trained for 100k steps and the CelebA-HQ dataset used one pre-trained for 200k steps. For both AFHQ and CelebA-HQ datasets, the optimization steps were set to 10k. The results are summarized in Table 1. Overall, ΔRN improves RN in nearly all metrics and all datasets.On the ImageNet, employing ΔRN with SiT-XL/2 lowers FID by up to 1.11. Furthermore, it achieved FID improvements of 1.89 and 3.52 on the AFHQ and CelebA-HQ datasets respectively.

Notably, the number of SiT blocks provides limited performance gains, as a small parameter count is sufficient to achieve good results.

Different Noise Analysis

We employ the fine-tuning strategy to train the π-noise generator. This is done by building upon a RF model that had been pre-trained for 6 million iterations on ImageNet. We explored three different noise assumptions—Gaussian

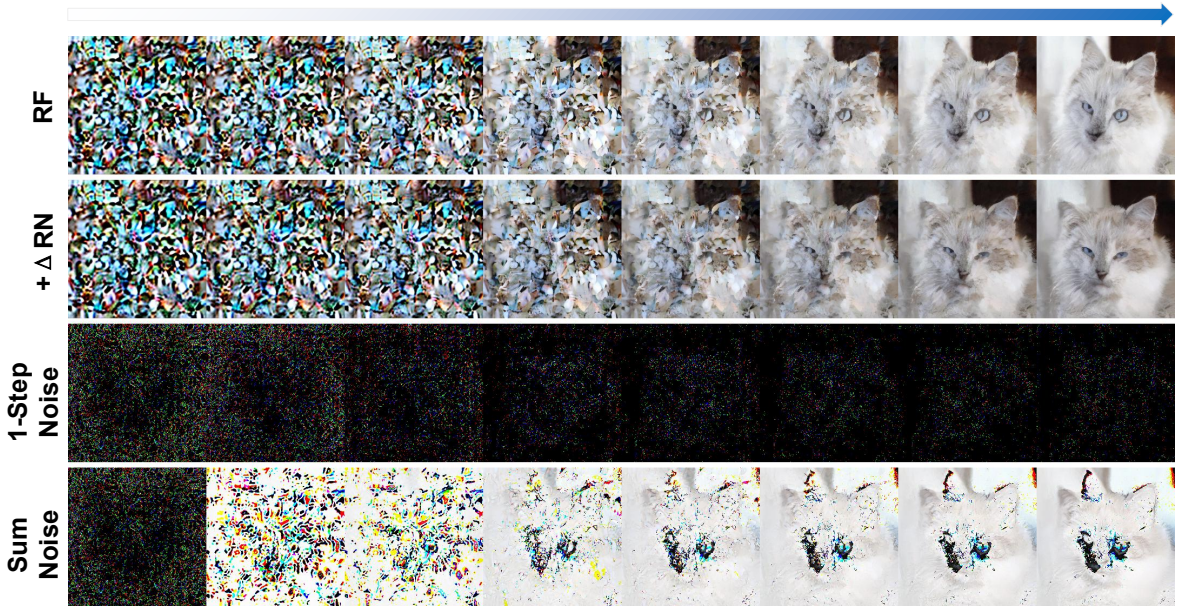

Figure 3: **Visualization of the π-noise by ΔRN.** The first line shows the original image generation with RF model, the second line shows the results of RF models using ΔRN in one step, the third line shows the generated noise for one step and the fourth line shows the cumulative noise for each time step. We use 180 steps for visualization.

distribution, Gumbel distribution, and Uniform distribution—training each for 100k iterations. The final results for each metric are summarized in Table 2. Uniform, Gumbel, and Gaussian distributions all improve model performance. Among these, Gaussian distribution is the most effective to enhance model performance.

	Metrics				
Model	FID ↓	IS ↑	sFID ↓	Prec. ↑	Rec. ↑
SiT-XL/2	10.16	123.86	12.02	0.50	0.62
Gumbel	9.42	129.73	11.42	0.52	0.61
Gaussian	**9.05**	**132.10**	**11.23**	**0.52**	**0.62**
Uniform	10.02	124.40	11.63	0.51	0.62

Table 2: **ImageNet-1k (256x256) results of different noise assumptions**. Gaussian noise is the most effective in enhancing the performance of the SiT model.

Different Training Strategies

We train the ΔRN model using a strategy that simultaneously optimizes the parameters θ and ψ . On the AFHQ and CelebA-HQ datasets, the trends in FID scores for SiT-B/2 and SiT-B/2 + ΔRN are shown in the Figure 4.

From the FID trend graph, we can see that training the π-noise with the traditional RF method during the training process does not yield a notable FID improvement. While theoretically there are two strategies to optimize the θ of π-noise generators, introducing random noise during training leads to instability and difficulty converging to an optimal solution. It is more advisable to employ a fine-tuning strategy to train the ΔRN model.

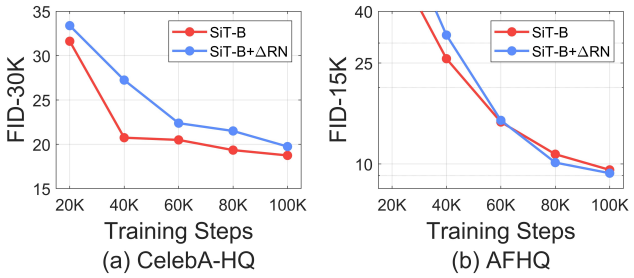

Figure 4: Training FID comparison for SiT-B/2 and SiT-B/2+ΔRN. The SiT B/2 + ΔRN model converges slower than the SiT B/2 model.

6 Conclusion

In this work, we introduced Rectified Noise, a novel generative model that enhances Rectified Flow models by injecting π-noise into their velocity fields. We introduce an auxiliary Gaussian distribution related to the flow matching loss to define the task entropy, the core of the π-noise framework. With the definition of Rectified Flow task entropy, we derive the optimization objective for Rectified Noise. We achieve efficient π-noise generator training through a fine-tuning approach and validated the effectiveness of this method. Furthermore, we compare the impact of different optimization strategies and noise assumptions on the model. We can further explore the potential of combining flow matching with π-noise in the future work.

References

Albergo, M. S.; Boffi, N. M.; and Vanden-Eijnden, E. 2023. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*.

Albergo, M. S.; and Vanden-Eijnden, E. 2022. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*.

Chen, R. T.; Rubanova, Y.; Bettencourt, J.; and Duvenaud, D. K. 2018. Neural ordinary differential equations. *Advances in neural information processing systems*, 31.

Chen, T. 2023. On the importance of noise scheduling for diffusion models. *arXiv preprint arXiv:2301.10972*.

De Bortoli, V.; Thornton, J.; Heng, J.; and Doucet, A. 2021. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in neural information processing systems*, 34: 17695–17709.

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

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Esser, P.; Kulal, S.; Blattmann, A.; Entezari, R.; Müller, J.; Saini, H.; Levi, Y.; Lorenz, D.; Sauer, A.; Boesel, F.; et al. 2024. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*.

Fu, Y.; Si, R.; Wang, H.; Zhou, D.; Sun, J.; Luo, P.; Hu, D.; Zhang, H.; and Li, X. 2025. Object-AVEdit: An Object-level Audio-Visual Editing Model. arXiv:2510.00050.

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30.

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

Ho, J.; and Salimans, T. 2022. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*.

Huang, S.; Xu, Y.; Zhang, H.; and Li, X. 2025a. Learn Beneficial Noise as Graph Augmentation. arXiv:2505.19024.

Huang, S.; Zhang, H.; and Li, X. 2025. Enhance vision-language alignment with noise. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 17449–17457.

Huang, Z.; Qiu, X.; Ma, Y.; Zhou, Y.; Chen, J.; Zhang, H.; Zhang, C.; and Li, X. 2025b. NFIG: Multi-Scale Autoregressive Image Generation via Frequency Ordering. arXiv:2503.07076.

Jiang, K.; Shi, Z.; Zhang, D.; Zhang, H.; and Li, X. 2025. Mixture of Noise for Pre-Trained Model-Based Class-Incremental Learning. arXiv:2509.16738.

Karras, T.; Aila, T.; Laine, S.; and Lehtinen, J. 2017. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*.

Kim, J.; Kim, M.; Kang, H.; and Lee, K. 2019. U-gat-it: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. *arXiv preprint arXiv:1907.10830*.

Kynkäänniemi, T.; Karras, T.; Laine, S.; Lehtinen, J.; and Aila, T. 2019. Improved precision and recall metric for assessing generative models. *Advances in neural information processing systems*, 32.

Li, X. 2022. Positive-incentive noise. *IEEE Transactions on Neural Networks and Learning Systems*, 35(6): 8708–8714.

Lipman, Y.; Chen, R. T.; Ben-Hamu, H.; Nickel, M.; and Le, M. 2022. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*.

Liu, X.; Gong, C.; and Liu, Q. 2022. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*.

Ma, N.; Goldstein, M.; Albergo, M. S.; Boffi, N. M.; Vanden-Eijnden, E.; and Xie, S. 2024. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, 23–40. Springer.

Nash, C.; Menick, J.; Dieleman, S.; and Battaglia, P. W. 2021. Generating images with sparse representations. *arXiv preprint arXiv:2103.03841*.

Papamakarios, G.; Nalisnick, E.; Rezende, D. J.; Mohamed, S.; and Lakshminarayanan, B. 2021. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57): 1–64.

Peebles, W.; and Xie, S. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, 4195–4205.

Polyak, A.; Zohar, A.; Brown, A.; Tjandra, A.; Sinha, A.; Lee, A.; Vyas, A.; Shi, B.; Ma, C.-Y.; Chuang, C.-Y.; et al. 2024. Movie gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10684–10695.

Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; and Chen, X. 2016. Improved techniques for training gans. *Advances in neural information processing systems*, 29.

Shannon, C. E. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3): 379–423.

Singhal, R.; Goldstein, M.; and Ranganath, R. 2023. Where to diffuse, how to diffuse, and how to get back: Automated learning for multivariate diffusions. *arXiv preprint arXiv:2302.07261*.

Sohl-Dickstein, J.; Weiss, E.; Maheswaranathan, N.; and Ganguli, S. 2015. Deep unsupervised learning using

nonequilibrium thermodynamics. In *International conference on machine learning*, 2256–2265. pmlr.

Song, Y.; and Ermon, S. 2019. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32.

Song, Y.; Sohl-Dickstein, J.; Kingma, D. P.; Kumar, A.; Ermon, S.; and Poole, B. 2020. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

Tzen, B.; and Raginsky, M. 2019. Theoretical guarantees for sampling and inference in generative models with latent diffusions. In *Conference on Learning Theory*, 3084–3114. PMLR.

Vahdat, A.; Kreis, K.; and Kautz, J. 2021. Score-based generative modeling in latent space, 2021. *URL https://arxiv.org/abs/2106.05931*.

Vargas, F.; Thodoroff, P.; Lamacraft, A.; and Lawrence, N. 2021. Solving schrödinger bridges via maximum likelihood. *Entropy*, 23(9): 1134.

Wang, J.; Zhang, H.; and Yuan, Y. 2025. Adv-CPG: A Customized Portrait Generation Framework with Facial Adversarial Attacks. arXiv:2503.08269.

Zhang, H.; Huang, S.; Guo, Y.; and Li, X. 2025. Variational positive-incentive noise: How noise benefits models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhang, H.; Xu, Y.; Huang, S.; and Li, X. 2024. Data Augmentation of Contrastive Learning is Estimating Positive-incentive Noise. arXiv:2408.09929.

Zheng, K.; Lu, C.; Chen, J.; and Zhu, J. 2023. Improved techniques for maximum likelihood estimation for diffusion odes. In *International Conference on Machine Learning*, 42363–42389. PMLR.