One Step Diffusion via Flow Fitting

Anonymous Author(s)
Affiliation
Address
email

Abstract

Diffusion and flow-matching models have demonstrated impressive performance in generating diverse, high-fidelity images by learning transformations from noise to data. However, their reliance on multi-step sampling requires repeated neural network evaluations, leading to high computational cost. We propose FlowFit, a family of generative models that enables high-quality sample generation through both single-phase training and single-step inference. FlowFit learns to approximate the continuous flow trajectory between latent noise x_0 and data x_1 by fitting a basis of functions parameterized over time $t \in [0, 1]$ during training. At inference time, sampling is performed by simply evaluating the flow only at the terminal time $t = 1$, avoiding iterative denoising or numerical integration. Empirically, FlowFit outperforms prior diffusion-based single-phase training methods achieving superior sample quality.

1 Introduction

In recent years, iterative denoising methods such as diffusion models [16, 5, 17] and flow matching [7, 8] have achieved remarkable success across a wide range of generative modeling tasks, including image synthesis, molecular generation, and audio modeling. These methods define a generative process as the solution to a learned differential equation that progressively transforms simple noise into complex data through a series of small, structured updates. Their strong empirical performance stems from their ability to model complex distributions with stable training dynamics and flexible architectures.

However, a key limitation of these approaches lies in their sampling efficiency. Since generation is performed by solving a differential equation, typically through numerical integration, these methods often require hundreds of sequential function evaluations at inference time. This iterative sampling procedure can be computationally expensive, slow, and memory-intensive, limiting their practicality in real-time or resource-constrained settings.

In this work, we aim to retain the modeling flexibility and training benefits of diffusion-based methods while enabling efficient **single-step generation**.

While recent work has explored accelerating inference in diffusion models through distillation, these approaches typically follow a two-stage training paradigm. In such methods, a pre-trained diffusion model is first learned through standard iterative training, and then a separate model is trained to mimic its behavior in fewer steps, usually by generating a large synthetic dataset of intermediate trajectories [10, 7] or propagating through a series of teacher and student networks [11, 14]. This introduces additional complexity, increases memory requirements, and may limit generalization due to reliance on a fixed teacher.

In contrast, we propose FlowFit, a unified, **single-phase training** approach that learns to generate samples in one step from the outset. Our method directly parameterizes the entire flow using a

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

Figure 1: **Comparison Between Multi-Step Flow Matching and Single-Step FlowFit Generation.** The top row shows images generated using a standard flow-matching model with 128 denoising steps, while the bottom row displays outputs from our single-step FlowFit model. Each column uses the same initial noise vector for a fair comparison. FlowFit produces high-fidelity samples even with a single forward pass, offering up to 128× faster sampling than traditional diffusion and flow-matching approaches while maintaining high image quality.

basis function expansion, enabling the model to learn global transport trajectories in a compact and structured manner. Once trained, sampling from the model requires only evaluating the learned flow at terminal time $t = 1$, given a source point $x_0 \sim \mu_0$ (see examples in Figure 1).

A key insight motivating our approach is that a smooth transformation, such as a flow trajectory, can be directly fitted using the initial value and all its time derivatives. By fitting the trajectory using a set of basis functions anchored at the initial point, we capture the flow's global behavior and bypass the need for iterative integration. This allows us to retain the expressive modeling capacity of continuous-time methods while achieving orders-of-magnitude faster sampling.

We train the model to ensure that the trajectory is consistent with a learned velocity field through a combination of Conditional Flow Matching and a trajectory-velocity consistency loss. Additionally, we introduce a progressive training strategy that improves stability and alignment between the flow and velocity in early training stages.

Our main contributions can be summarized as follows

- We propose a novel formulation for flow modeling via basis function fitting. FlowFit, a new generative modeling framework that directly parameterizes continuous-time flows using a residual expansion over fixed basis functions. This allows the model to represent entire flow trajectories with a compact, structured parameterization. To the best of our knowledge, FlowFit is the first to propose such a formulation.
- We demonstrate that FlowFit achieves sample quality better or on par with existing single step diffusion-based approaches .

2 Related Work

We review prior research efforts aimed at accelerating diffusion-based generative models through single-step sampling. Existing approaches generally fall into two categories: those based on multi-stage distillation and those relying on direct, single-phase training.

2.1 Two-Phase Training Approaches

A common strategy for improving inference efficiency involves distilling multi-step diffusion models into simpler one-step samplers. These techniques typically follow a two-phase process: a full diffusion model is trained first, and then a lightweight student model is optimized to mimic its behavior over fewer denoising steps.

Several works adopt this paradigm by simulating the full denoising trajectory to generate supervision pairs, as seen in knowledge distillation [10] and rectified flows [8]. While effective, these methods are computationally intensive due to the need for full reverse-time ODE evaluations. To mitigate this, more recent efforts introduce bootstrapping mechanisms that shorten the ODE simulation path [4, 19]. Additionally, researchers have explored a variety of loss functions beyond the traditional L2 objective, including adversarial criteria [15] and distributional matching techniques [21, 20].

Progressive distillation [14, 1, 11] offers a multi-stage solution, wherein a sequence of student models is trained with progressively larger time steps. This hierarchical approach reduces the dependency on costly long-path bootstrap samples.

Our proposed method diverges from these frameworks by eliminating the need for both pretraining and distillation. Instead, it adopts a unified, end-to-end training scheme that learns a single-step generator directly, simplifying both the implementation and training pipeline.

2.2 Single-Phase Training Approaches

Only a limited number of methods have been developed for one-step generation via single-stage training. Among the first of its kind, Consistency Models [18] learn to map noisy inputs directly to their clean counterparts in a single forward pass. Although originally designed for distillation, they have also been extended to an end-to-end training setup.

Shortcut models [3] introduce a flexible generative approach that allows conditioning on both the input noise level and the desired step size, enabling inference under various computational constraints.

While FlowFit shares the general concept of consistency with [18] and [3], it is significantly different. To the best of our knowledge, FlowFit is the first method to directly aim for a single-step diffusion model using a basis of functions.

3 Preliminary: Diffusion and Flow Matching

Recent advances in generative modeling have led to the development of methods such as diffusion models [16, 5, 17] and flow-matching approaches [7, 8], which learn a continuous-time transformation from a simple noise distribution to a complex data distribution. These models typically define the generative process through an ordinary differential equation (ODE), where the time-dependent dynamics are learned to guide samples from noise toward data.

In this work, we adopt the flow-matching formulation based on the optimal transport objective introduced by [8], as it offers a simple and effective framework for learning such dynamics. While diffusion models and flow-matching approaches are often studied separately, recent perspectives, such as that of [6], highlight that flow matching can be interpreted as a deterministic special case of diffusion modeling. Accordingly, we treat the two paradigms as closely related and use the terminology interchangeably where appropriate.

Flow Matching provides a supervised learning framework for modeling deterministic, continuous-time flows that transport a base distribution μ_0 (e.g., standard Gaussian) into a target distribution μ_1 (e.g., data distribution). Drawing from optimal transport and neural ODEs, it directly learns a velocity field that defines a transport trajectory between paired samples.

Let $x_0 \sim \mu_0$ be a sample from the source distribution and $x_1 \sim \mu_1$ its corresponding target. The model learns a time-dependent velocity field $\mathbf{v}_\theta(x, t)$ such that solving the associated ODE transforms μ_0 into μ_1. Formally, the flow is described by

$$\begin{cases} \dfrac{d}{dt}\psi_t(x) = \mathbf{u}_t(\psi_t(x)), \\ \quad \psi_0(x) = x, \end{cases} \tag{1}$$

where $\mathbf{u}_t : [0, 1] \times \mathbb{R}^d \to \mathbb{R}^d$ is a neural network parameterizing the velocity field, and $\psi_t(x)$ is the flow map at time t.

A common training strategy is to supervise the model using velocity information along linear paths between x_0 and x_1, evaluated at intermediate points $(1-t)x_0 + tx_1$. The corresponding ground-truth velocity at such a point is simply $x_1 - x_0$.

The model is then trained to match this known velocity at intermediate points. The Conditional Flow Matching (CFM) objective function is defined as

$$\mathcal{L}(\theta) = \mathbb{E}_{x_0 \sim \mu_0,\, x_1 \sim \mu_1,\, t \sim \mathcal{U}[0,1]} \left[\left\| \mathbf{v}_\theta \left((1-t)x_0 + tx_1, t \right) - (x_1 - x_0) \right\|^2 \right]. \tag{2}$$

This loss guides the model to predict the instantaneous velocity field that aligns with the linear flow between samples. Notably, this avoids the need for computing density functions or score gradients, distinguishing it from traditional diffusion models.

Once training is complete, sample generation begins by drawing an initial point $x_0 \sim \mu_0$, typically from a standard Gaussian distribution. This point is then transformed toward the data distribution by solving the learned ODE defined by the velocity field $\mathbf{v}_\theta(x, t)$. In practice, this continuous-time flow is discretized and approximated using numerical integration methods such as Euler's method, where the sample is updated iteratively over a sequence of small time steps from $t = 0$ to $t = 1$.

4 FlowFit: Direct Flow Parameterization via Basis Function Fitting

In this section, we introduce FlowFit, a novel approach for modeling continuous-time flows via basis function fitting. Specifically, we aim to directly parameterize the flow $\psi_t(x)$ using a basis of functions that are conditioned on both the initial point x_0 and the time parameter t. Thus, the goal is to model the mapping $(x_0, t) \mapsto \psi_t(x) = x_t$, where $\psi_t(x)$ is the flow at time t, and x_t represents the transformed point at time t.

To this end, we approximate the true flow $\psi_t(x)$ with a learnable function $\psi_\theta(x_0, t)$[1] , parameterized via neural networks and basis functions. For simplicity, we use the notation x_t interchangeably with both $\psi_t(x)$ and $\psi_\theta(x_0, t)$, with the dependence on x_0 and t understood implicitly. This allows us to define the transformation in terms of a set of basis functions, which we leverage to approximate the flow dynamics. Importantly, this representation enables efficient single-step generation by evaluating the learned trajectory at $t = 1$, and requires only the initial sample $x_0 \sim \mu_0$ at inference time.

4.1 Trajectory Parameterization

We approximate the continuous flow trajectory using the time-dependent formulation

$$\begin{cases} \psi_\theta(x_0, t) & = x_t = \displaystyle\sum_{k=1}^{K} f_{\theta,k}(x_0) \gamma_k(t), \\ \psi_\theta(x_0, 0) & = x_0, \end{cases} \tag{3}$$

where

- $\{\gamma_k(t)\}_{k=1}^K$ are fixed scalar basis functions (e.g., polynomial or Fourier),
- $f_{\theta,k} : \mathbb{R}^d \to \mathbb{R}^d$ are neural networks that produce the coefficients,
- θ denotes all learnable parameters.

A first question is whether one can approximate arbitrarily well the trajectory $\psi(x, t)$ with the above basis expansion. This is a rather straightforward result that we report in the Appendix for completeness.

4.2 FlowFit Training

We jointly train two models: ψ_θ, which parameterizes the flow trajectory, and $v_{\theta'}$, which models the time-dependent velocity field. To ensure that the learned trajectory $\psi_\theta(x_0, t)$ evolves in alignment

[1] In the general case, we aim to learn $\psi_\theta(x_0, c, t)$, where c denotes any form of conditioning (e.g., class labels). For simplicity of derivation, we omit c.

with the correct transport dynamics, we supervise it by matching its time derivative to a known target velocity field. Taking the derivative of the basis function parameterization gives

$$\frac{d\psi_\theta}{dt}(x_0, t) = \sum_{k=1}^{K} f_{\theta,k}(x_0) \frac{d\gamma_k}{dt}(t). \tag{4}$$

Simultaneously, the velocity model $v_{\theta'}(x, t)$ is trained using the standard Conditional Flow Matching (CFM) objective. In this setup, a source point x_0 is randomly sampled from the base distribution μ_0 and a target point x_1 is independently sampled from the data distribution μ_1. The model is then trained to match the ground-truth velocity $\tilde{\mathbf{v}}(x_0, x_1, t) = x_1 - x_0$ at intermediate points along the linear interpolation between x_0 and x_1. The loss is given by

$$\mathcal{L}_{\text{CFM}}(\theta') = \mathbb{E}_{x_0 \sim \mu_0,\, x_1 \sim \mu_1,\, t \sim \mathcal{U}[0,1]} \left[\| v_{\theta'}((1-t)x_0 + t x_1, t) - (x_1 - x_0) \|^2 \right]. \tag{5}$$

To align the basis-induced flow with the learned velocity field, we require that the velocity induced by the basis-function trajectory matches the prediction of the velocity model at the corresponding location along the flow

$$\frac{d\psi_\theta}{dt}(x_0, t) \approx v_{\theta'}(\psi_\theta(x_0, t), t). \tag{6}$$

We formalize this requirement with a loss that enforces matching the flow derivative to the velocity

$$\mathcal{L}_{\text{derivative}}(\theta) = \mathbb{E}_{x_0 \sim \mu_0,\, t \sim \mathcal{U}[0,1]} \left[\left\| \frac{d\psi_\theta}{dt}(x_0, t) - v_{\theta'}(\text{sg}[\psi_\theta(x_0, t)], t) \right\|^2 \right], \tag{7}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient. This loss encourages the parameterized trajectory to follow a velocity field that is internally consistent with the learned dynamics, improving the alignment between the path and the underlying transport vector field.

Despite the fact that $\psi_\theta(x_0, t)$ may initially provide limited information early in training, we observe that the propagation loss remains robust and effective. This eliminates the need for manually scheduling t, thereby simplifying the training process and enhancing both stability and usability, all without sacrificing performance. The full training algorithm and the corresponding sampling procedure are outlined in Algorithm 1 and 2.

4.3 Single-Step Generation

At inference time, the model generates samples from the target distribution μ_1 by drawing $x_0 \sim \mu_0$ and evaluating the fitted trajectory at terminal time:

$$x_1 = \psi_\theta(x_0, t = 1). \tag{8}$$

This enables fast and deterministic generation without iterative integration, in contrast to diffusion or traditional flow-based models.

5 Experiments

5.1 Experimental Setup

We evaluate our method alongside a range of established baselines under consistent training conditions. To ensure fairness, all models are trained from scratch using an identical implementation and share the same backbone architecture, the DiT-B diffusion transformer [12]. Our evaluation includes two tasks: unconditional image generation on the CelebAHQ-256 dataset [9]. In our supplementary, we also provide a comparison with class-conditional generation on ImageNet-256 [2] . For the experiments reported in Table 1, we use the AdamW optimizer with a constant learning rate of 4×10^{-5}, no weight decay, and train each model for 500K iterations. We use a Polynomial basis with order 8. All models are trained and sampled in the latent space provided by the `sd-vae-ft-mse` autoencoder [13]. Additional experimental details and our codebase are available in the supplementary material.

Algorithm 1 Training FlowFit

Initialize θ, θ', time window $\alpha_t \leftarrow 0$
Initialize $\psi_\theta(., t)$ as in Equation 3
for each training step **do**
  Sample $x_0 \sim \mu_0, x_1 \sim \mu_1, t \sim \mathcal{U}[0, 1]$
  Train $v_{\theta'}$ with CFM:
    $\tilde{x}_t = (1-t)x_0 + tx_1$
    $\min_{\theta'} \mathcal{L}_{\text{CFM}}(\theta') = \|v_{\theta'}(\tilde{x}_t, t) - (x_1 - x_0)\|^2$
    Update θ'
  Train ψ_θ with consistency loss:
    Compute $\psi_\theta(x_0, t)$, $\frac{d\psi_\theta}{dt}(x_0, t)$ using Equation 4
    $\min_\theta \mathcal{L}_{\text{derivative}}(\theta) = \|\frac{d\psi_\theta}{dt}(x_0, t) - v_{\theta'}(\psi_\theta(x_0, t), t)\|^2$
    Update θ
  Increase α_t toward 1
end for

Algorithm 2 Sampling

Sample $x_0 \sim \mu_0$
Return $x_1 = \psi_\theta(x_0, t = 1)$

5.2 Baseline Approaches

We benchmark our approach against a diverse set of generative models, closely following the evaluation protocol introduced in [3]. The results for the competing methods, shown in Table 1, are taken from [3]. For clarity, we briefly summarize the training procedures of the baseline models as described in that work. The baselines fall into two main categories: *two-phase training models*, which distill fast samplers from pretrained diffusion models, and *end-to-end models*, which are trained from scratch to perform generation in a single step.

In the distillation category, we include the standard diffusion model, following the setup from [12], and Flow Matching [8], which replaces the diffusion objective with a transport-based loss. Reflow [8] is a representative distillation method that generates 50k synthetic (x_0, x_1) pairs on CelebAHQ using a teacher model requiring 128 forward passes per sample. The resulting student is then trained over the full time interval $t \in (0, 1)$. Progressive Distillation [14] iteratively distills a sequence of student models, each trained with a step size double that of its predecessor, starting from a classifier-free guided teacher. Consistency Distillation [18] trains the student model to predict final samples from intermediate pairs generated by the teacher.

In contrast, end-to-end methods eliminate the need for a teacher model. Consistency Training [18] trains a single-step model directly by enforcing consistency between empirical sample pairs $(x_t, x_{t+\delta})$, with progressively increasing time intervals during training. Shortcut models [3] introduce step-size conditioning, allowing for flexible generation across varying inference budgets. Finally, Live Reflow [3] jointly trains on both flow-matching and Reflow-distilled targets, regenerating new targets at each training step via full denoising from a flow-matching teacher. While this yields strong performance, it incurs significant computational cost.

5.3 Evaluation Protocol

We follow the evaluation framework established in [3]. Each model generates 50k samples for computing the FID-50k score. Our method is evaluated using a single-step sampler, while the baseline models are assessed under 128-step, 4-step, and 1-step variants. FID-50k is calculated using statistics from the full dataset, with no compression applied to the generated samples. All images are resized to 299×299 via bilinear interpolation and normalized to the $[-1, 1]$ range. During inference, we apply the Exponential Moving Average (EMA) of the model parameters to improve stability and performance.

5.4 Comparison

Table 1 highlights that FlowFit delivers high-quality generations in the single sampling step. Notably, it surpasses all other single-phase training methods in one-step generation performance and performs on par with progressive distillation techniques that require multiple training stages. In contrast

Table 1: **Comparison of various training objectives applied to the same architecture (DiT-B)**. We report FID-50k scores (lower is better) for 128, 4, and 1-step denoising. FlowFit achieves high-quality samples using a single training phase and a one-step inference process, significantly reducing the performance gap with distillation-based methods. Results in parentheses indicate settings beyond the intended use of the corresponding objective.

	CelebAHQ-256		
	128-Step	**4-Step**	**1-Step**
Two phase training			
Progressive Distillation	(302.9)	(251.3)	**14.8**
Consistency Distillation	59.5	39.6	38.2
Reflow	**16.1**	**18.4**	23.2
End-to-end (single training run)			
Diffusion	23.0	(123.4)	(132.2)
Flow Matching	7.3	(63.3)	(280.5)
Consistency Training	53.7	19.0	33.2
Live Reflow	**6.3**	27.2	43.3
Shortcut Models	6.9	**13.8**	20.5
FlowFit (ours)	-	-	**14.1**

Figure 2: **Latent Space Interpolation.** All images shown are generated by the model. Each row illustrates the result of applying one-step denoising to intermediate samples obtained by variance-preserving interpolation between two independent Gaussian noise vectors.

to these multi-stage approaches, FlowFit achieves its results within a single training phase. As anticipated, conventional diffusion and flow-matching models experience a notable decline in sample quality when constrained to just 4 or 1 inference steps. We include further qualitative examples in the supplementary material.

5.5 Semantic Structure in the Latent Space of FlowFit

To assess whether FlowFit gives rise to a semantically meaningful and smooth latent space, we perform an interpolation experiment in the input noise domain. We begin by selecting pairs of Gaussian noise vectors x_0^0 and x_0^1, and interpolate between them using a variance-preserving scheme $x_0^n = nx_0^1 + \sqrt{1-n^2}\, x_0^0$ with $n \in [0, 1]$. Each interpolated point x_0^n is then processed through the trained model to generate the corresponding output. Figure 2 shows representative results from this interpolation. Even though no explicit smoothness constraints or regularization terms are imposed during training, the outputs exhibit continuous and visually coherent changes. The interpolated generations preserve high-level semantics while gradually morphing between endpoints, indicating that FlowFit captures an underlying latent structure that supports semantically consistent transitions.

6 Ablations

6.1 Effect of Basis Order

We investigate how the order of the polynomial basis affects the resulting image quality. In Table 2, we present results for two polynomial basis orders under the same training conditions: order 2 with

Table 2: Impact of the chosen polynomial basis order on image quality.

Basis Order	$\{t^k\}_{k=1}^2$	$\{t^k\}_{k=1}^8$
FID (↓)	18.2	14.1

Table 3: Impact of the basis nature using the same expansion order.

Basis	Polynomial	Trigonometric
FID ↓	14.1	16.3

basis functions $\{t^k\}_{k=1}^2$ and order 8 with basis functions $\{t^k\}_{k=1}^8$. Our findings show that using a higher-order basis leads to improved image quality.

6.2 Effect of the Basis Type

We evaluate the influence of the basis function type on generation quality while keeping the expansion order fixed. Specifically, we use $\{t^k\}_{k=1}^8$ for the polynomial basis and $\{\cos(2k\pi t)\}_{k=1}^4 \cup \{\sin(2k\pi t)\}_{k=1}^4$ for the trigonometric basis. As reported in Table 3, the polynomial basis yields a better FID score of 14.1 compared to the trigonometric basis (16.3).

Limitations and Future Work

While FlowFit demonstrates strong performance, it also has certain limitations. One notable drawback is the requirement to train two separate networks, which may lead to increased computational overhead. Moreover, the current design is restricted to single-step inference. A promising direction for future research is to generalize the framework to support a unified model architecture that enables flexible sampling with a variable number of inference steps.

Due to computational constraints, we explore the method up to order 8 in this work. It would be interesting to investigate higher-order expansions and assess their impact. Another potential direction is to explore alternative basis functions in the formulation.

7 Conclusion

We introduced FlowFit, a novel diffusion-based generative model that enables both single-step sampling. The core idea behind our approach is a new formulation for flow modeling based on basis function fitting. FlowFit delivers fast, high-quality generation while substantially simplifying the training process, making it a practical and efficient diffusion-based method.

References

[1] David Berthelot, Arnaud Autef, Jierui Lin, Dian Ang Yap, Shuangfei Zhai, Siyuan Hu, Daniel Zheng, Walter Talbott, and Eric Gu. Tract: Denoising diffusion models with transitive closure time-distillation. *arXiv preprint arXiv:2303.04248*, 2023.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[3] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024.

[4] Jiatao Gu, Shuangfei Zhai, Yizhe Zhang, Lingjie Liu, and Joshua M Susskind. Boot: Data-free distillation of denoising diffusion models with bootstrapping. In *ICML 2023 Workshop on Structured Probabilistic Inference $\{\backslash\&\}$ Generative Modeling*, 2023.

[5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[6] Diederik Kingma and Ruiqi Gao. Understanding diffusion objectives as the elbo with simple data augmentation. *Advances in Neural Information Processing Systems*, 36:65484–65516, 2023.

[7] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[8] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[9] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15(2018):11, 2018.

[10] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021.

[11] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14297–14306, 2023.

[12] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[13] 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*, pages 10684–10695, 2022.

[14] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[15] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pages 87–103. Springer, 2024.

[16] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015.

[17] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*.

[18] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. 2023.

[19] Sirui Xie, Zhisheng Xiao, Diederik P Kingma, Tingbo Hou, Ying Nian Wu, Kevin Patrick Murphy, Tim Salimans, Ben Poole, and Ruiqi Gao. Em distillation for one-step diffusion models, 2024. *URL https://arxiv. org/abs/2405.16852*.

[20] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and Bill Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in neural information processing systems*, 37:47455–47487, 2024.

[21] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6613–6623, 2024.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: the claims made reflect the paper contributions.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We included a Limitations section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We included the assumptions and the full derivations.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We will share our code in the supplementary.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: we will share the code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We will share all the details in the supplementary.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We follow prior works that do not .

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: We will provide this information if the reviewers ask for it.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We follow the code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: We believe there is no major societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: There is no high risk.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cited the used assets.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer:[NA]

Justification: No new asset.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: No crowd-sourcing.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: No risks.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: No major use of LLMs.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.