FLOW-ANCHORED CONSISTENCY MODELS

Yansong Peng[1,2∗], **Kai Zhu**[1,2], **Yu Liu**[2],
Pingyu Wu[1,2∗], **Hebei Li**[1], **Xiaoyan Sun**[1†], **Feng Wu**[1]
[1]University of Science and Technology of China [2]Tongyi Lab
pengyansong@mail.ustc.edu.cn

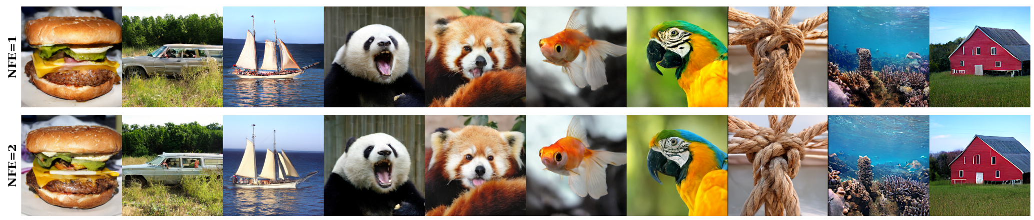


Figure 1: Generated 1-step and 2-step samples on ImageNet 256×256 by our FACM.

ABSTRACT

Continuous-time Consistency Models (CMs) promise efficient few-step generation but face significant challenges with training instability. We argue this instability stems from a fundamental conflict: by training a network to learn only a shortcut across a probability flow, the model loses its grasp on the instantaneous velocity field that defines the flow. Our solution is to explicitly anchor the model in the underlying flow during training. We introduce the Flow-Anchored Consistency Model (FACM), a simple but effective training strategy that uses a Flow Matching (FM) task as an anchor for the primary CM shortcut objective. This **Flow-Anchoring** approach requires no architectural modifications and is broadly compatible with standard model architectures. By distilling a pre-trained LightningDiT model, our method achieves a state-of-the-art FID of 1.32 with two steps (NFE=2) and 1.76 with just one step (NFE=1) on ImageNet 256×256, significantly outperforming previous methods. This provides a general and effective recipe for building high-performance, few-step generative models. Our code and pretrained models: `https://github.com/ali-vilab/FACM`.

1 INTRODUCTION

As generative models scale to unprecedented sizes and applications demand real-time synthesis, the need for efficient, few-step samplers has become paramount. Consistency Models (CMs) (Song et al., 2023) have emerged as a promising paradigm for few-step generation. Early successful works were largely based on discrete-time formulations (Song et al., 2023; Song & Dhariwal, 2023; Geng et al., 2024), which are inherently prone to discretization errors. While their continuous-time counterparts can circumvent these errors, they have been historically hindered by severe training instability. Recent approaches, notably sCM (Lu & Song, 2024), have made significant strides in stabilizing continuous-time training through a combination of regularization techniques and architectural modifications. Concurrently, another line of research has aimed to stabilize training by reformulating the shortcut objective itself. For instance, IMM (Zhou et al., 2025) introduces additional self-consistency constraints between multiple timesteps, while MeanFlow (Geng et al., 2025) models the "average velocity" to arbitrary endpoints, implicitly learning about the underlying flow. Although these methods achieve stable few-step sampling, they do not explicitly decouple the task of modeling the instantaneous velocity field from the task of predicting the shortcut. We

[∗]Work done during their internships at Tongyi Lab.
[†]Corresponding author

contend that this entanglement leads to sub-optimal performance, as the model must learn both the foundational flow and the shortcut simultaneously through a single, overly-coupled objective.

This paper addresses the root cause of instability in the continuous CM objective from a different perspective. We posit that the standard continuous CM objective, while powerful for learning a direct “shortcut” across a probability flow, is inherently unstable when trained in isolation. This is because it implicitly assumes the model has a robust understanding of the underlying flow, yet training exclusively on the shortcut objective can cause the model’s representation of this flow to degrade, leading to training collapse. Our key insight is that stability can be achieved by explicitly anchoring the model in the very flow it is shortcutting. The most direct way to achieve this **Flow-Anchoring** is to re-introduce the explicit training of the **instantaneous velocity field** that defines the flow. We propose that an objective based on Flow Matching (FM) (Lipman et al., 2022) can act as a crucial anchor, enabling the primary shortcut objective to be trained effectively. Based on this principle, we introduce the Flow-Anchored Consistency Model (FACM), which employs a simple yet effective training strategy combining two distinct objectives:

- **Flow-Anchoring Objective** that learns the flow’s velocity field to provide stability.
- **Shortcut Objective** that learns the efficient 1-step consistency mapping.

Our method is an architecturally-agnostic training strategy. We propose two effective implementations for this mixed-objective design, including an innovative Expanded Time Interval strategy that achieves optimal results without any architectural modifications. It supports both stable training from scratch and distillation from a pre-trained FM model, with the distillation pathway achieving superior performance at a significantly lower total training cost. Our approach sets new state-of-the-art FID scores of 1.76 (NFE=1) and 1.32 (NFE=2) on the class-conditional ImageNet 256×256 benchmark, providing a general recipe for building high-performance, few-step generative models.

2 PRELIMINARIES

Generative modeling aims to learn a probability flow ODE, $\mathrm{d}\boldsymbol{x}_t/\mathrm{d}t = \boldsymbol{v}_\theta(\boldsymbol{x}_t, t)$, that transforms a prior distribution p_0 (e.g., $\mathcal{N}(0, I)$) to a data distribution p_1.

2.1 FLOW MATCHING

Flow Matching (FM) (Liu et al., 2022; Albergo & Vanden-Eijnden, 2022; Lipman et al., 2022) learns the vector field $\boldsymbol{v}_\theta$ by regressing the network’s output against a conditional velocity field. This is achieved by defining a probability path between a noise sample $\boldsymbol{x}_0 \sim p_0$ and a data sample $\boldsymbol{x}_1 \sim p_1$. A simple and effective choice is OT-FM defined as $\boldsymbol{x}_t = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1$, which has a constant conditional velocity of $\boldsymbol{x}_1 - \boldsymbol{x}_0$. By training the model to predict this easily computable conditional velocity, it learns the marginal velocity field. This leads to the practical Flow Matching objective:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{x}_1} \left\| \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) - (\boldsymbol{x}_1 - \boldsymbol{x}_0) \right\|_2^2 . \tag{1}$$

2.2 CONSISTENCY MODELS

Consistency Models (CMs) (Song et al., 2023) are trained to map any point $\boldsymbol{x}_t$ on a given ODE trajectory directly to the trajectory’s endpoint in a single step. This is enforced through a self-consistency property: for any two points $\boldsymbol{x}_t$ and $\boldsymbol{x}_t'$ on the same trajectory, their outputs must be identical, i.e., $f_\theta(\boldsymbol{x}_t, t) = f_\theta(\boldsymbol{x}_t', t')$. This implies that for an ideal model, $f_\theta(\boldsymbol{x}_t, t)$ should equal the trajectory’s endpoint $\boldsymbol{x}_1$ for all $t \in [0, 1]$, which imposes a boundary condition $f_\theta(\boldsymbol{x}_1, 1) = \boldsymbol{x}_1$. CMs can be trained in both discrete and continuous time. Discrete-time CMs enforce consistency between two adjacent points with a finite distance, $f_\theta(\boldsymbol{x}_t, t) \approx f_{\theta^-}(\boldsymbol{x}_{t+\Delta t}, t + \Delta t)$, an approach that is inherently prone to discretization errors. Our work focuses on the continuous-time formulation. Here, the consistency property requires the total derivative of the shortcut function to be zero, $\frac{\mathrm{d}f_\theta(\boldsymbol{x}_t,t)}{\mathrm{d}t} = 0$. To satisfy the boundary condition, the function is parameterized as $f_\theta(\boldsymbol{x}_t, t) = \boldsymbol{x}_t + (1-t)\boldsymbol{F}_\theta(\boldsymbol{x}_t, t)$, which implies that the network $\boldsymbol{F}_\theta$ must satisfy:

$$\boldsymbol{F}_\theta(\boldsymbol{x}_t, t) = \boldsymbol{v} + (1-t)\frac{\mathrm{d}\boldsymbol{F}_\theta(\boldsymbol{x}_t, t)}{\mathrm{d}t}. \tag{2}$$

Here, $\boldsymbol{v}$ represents the conditional velocity $\boldsymbol{x}_1 - \boldsymbol{x}_0$ from the underlying flow. In the distillation paradigm, this velocity is provided by a pre-trained FM teacher, which can also incorporate classifier-free guidance (CFG) (Ho & Salimans, 2022). This objective, which relies on the Jacobian-vector product (JVP) to compute the derivative term, is known to be unstable to train (Lu & Song, 2024).

3 FLOW-ANCHORED CONSISTENCY MODELS (FACM)

This section first analyzes the core instability of continuous-time Consistency Models (CMs), identifying the "missing anchor" as the root cause. We then present our solution, the Flow-Anchored Consistency Model (FACM), detailing its mixed-objective training strategy.

3.1 DECONSTRUCTING CM INSTABILITY: THE MISSING ANCHOR

Our analysis reframes the challenge of training continuous-time Consistency Models. We argue that the instability is not an inherent flaw of the shortcut objective itself, but a consequence of training on it in isolation, which causes the model to lose its anchor in the flow's underlying velocity field.

3.1.1 THE SHORTCUT TARGET AS AN AVERAGE VELOCITY

To understand the mechanics of the generative shortcut, we first re-examine the consistency model's learning objective. The goal of a consistency function $f_\theta(\boldsymbol{x}_t, t)$ is to map any point $\boldsymbol{x}_t$ on an ODE trajectory to its endpoint $\boldsymbol{x}_1$. Using the OT-FM parameterization $f_\theta(\boldsymbol{x}_t, t) = \boldsymbol{x}_t + (1-t)\boldsymbol{F}_\theta(\boldsymbol{x}_t, t)$, the ideal shortcut $f_\theta(\boldsymbol{x}_t, t) = \boldsymbol{x}_1$ can only be achieved if the network $\boldsymbol{F}_\theta$ learns to predict a very specific quantity:

$$\boldsymbol{x}_t + (1-t)\boldsymbol{F}_\theta(\boldsymbol{x}_t, t) = \boldsymbol{x}_1 \quad \Rightarrow \quad \boldsymbol{F}_\theta(\boldsymbol{x}_t, t) = \frac{\boldsymbol{x}_1 - \boldsymbol{x}_t}{1-t}. \tag{3}$$

This term has a clear physical interpretation: it is the **average velocity** required to travel from point $\boldsymbol{x}_t$ to the endpoint $\boldsymbol{x}_1$ in the remaining time $1-t$. We denote this quantity as $\overline{\boldsymbol{v}}(\boldsymbol{x}_t, t)$. Thus, the task of learning the 1-step shortcut is equivalent to training $\boldsymbol{F}_\theta$ to predict this average velocity.

Now, we investigate the properties that this average velocity field must satisfy. From its definition in Eq. 3, we have $(1-t)\overline{\boldsymbol{v}}(\boldsymbol{x}_t, t) = \boldsymbol{x}_1 - \boldsymbol{x}_t$. Differentiating both sides with respect to t using the product rule gives:

$$\frac{d}{dt}\left((1-t)\cdot\overline{\boldsymbol{v}}(\boldsymbol{x}_t, t)\right) = -\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} \quad \Rightarrow \quad -\overline{\boldsymbol{v}}(\boldsymbol{x}_t, t) + (1-t)\frac{\mathrm{d}\overline{\boldsymbol{v}}(\boldsymbol{x}_t, t)}{\mathrm{d}t} = -\boldsymbol{v}(\boldsymbol{x}_t, t). \tag{4}$$

Rearranging the terms, we arrive at a key differential identity that the true average velocity field must satisfy:

$$\overline{\boldsymbol{v}}(\boldsymbol{x}_t, t) = \boldsymbol{v}(\boldsymbol{x}_t, t) + (1-t)\frac{\mathrm{d}\overline{\boldsymbol{v}}(\boldsymbol{x}_t, t)}{\mathrm{d}t}. \tag{5}$$

This identity is formally identical to the continuous-time CM learning objective (repeated from Eq. 2 for clarity):

$$\boldsymbol{F}_\theta(\boldsymbol{x}_t, t) = \boldsymbol{v} + (1-t)\frac{\mathrm{d}\boldsymbol{F}_\theta(\boldsymbol{x}_t, t)}{\mathrm{d}t}. \tag{6}$$

This confirms that the CM objective directly forces the network $\boldsymbol{F}_\theta$ to learn the properties of an average velocity field, thus enabling the 1-step generation shortcut.

3.1.2 THE SOURCE OF INSTABILITY: LOSING THE FLOW ANCHOR

While Eq. 6 correctly identifies the target, its practical implementation via the training objective $T = \boldsymbol{v} + (1-t)\frac{\mathrm{d}\boldsymbol{F}_{\theta^-}(\boldsymbol{x}_t, t)}{\mathrm{d}t}$ is notoriously unstable. The core of this instability lies in the target's self-referential nature. This dependency creates two fundamental, intertwined problems:

Missing Instantaneous Velocity Field Supervision The target T explicitly depends on the ground-truth instantaneous velocity $\boldsymbol{v}$. The CM objective, however, only enforces a loss on the final prediction $\boldsymbol{F}_\theta$ (the average velocity). There is no explicit mechanism to ensure that the model's learned dynamics remains faithful to the underlying instantaneous velocity field $\boldsymbol{v}$. The model is being asked to learn the integral of a function (average velocity) without being explicitly taught the function itself (instantaneous velocity).

Self-Referential Derivative Estimation This lack of direct supervision on $\boldsymbol{v}$ makes the derivative term, $\frac{\mathrm{d}\boldsymbol{F}_{\theta^-}}{\mathrm{d}t}$, highly unstable. The total derivative, expanded via the chain rule, is:

$$\frac{\mathrm{d}\boldsymbol{F}_{\theta^-}(\boldsymbol{x}_t, t)}{\mathrm{d}t} = (\nabla_{\boldsymbol{x}_t} \boldsymbol{F}_{\theta^-})\boldsymbol{v} + \frac{\partial \boldsymbol{F}_{\theta^-}}{\partial t}. \tag{7}$$

The network is required to estimate its own derivative. Even if the network is pre-trained (e.g., with Flow Matching) and initially provides a good approximation of the instantaneous velocity, the CM objective alone provides no continuous supervision to maintain this alignment. Without this anchor, the model's output $\boldsymbol{F}_\theta$ can quickly drift, becoming meaningless or unstructured. In this state, the derivative term $\frac{\mathrm{d}\boldsymbol{F}_{\theta^-}}{\mathrm{d}t}$ becomes noisy and erratic. Using this noisy derivative to construct the training target creates a vicious cycle that rapidly amplifies errors and leads to training collapse.

These two issues are deeply intertwined, stemming from the same fundamental problem: the CM objective is ungrounded. It lacks a stable foundation in the very flow it is supposed to shortcut. The antidote is to re-introduce the explicit supervision of the instantaneous velocity field $\boldsymbol{v}$ via a Flow Matching objective. This provides a stable **anchor** for the model's internal dynamics, ensuring that the model's gradient field is well-behaved, which directly stabilizes the derivative term in the CM objective and allows the primary shortcut objective to be learned effectively. We term this principle **Flow-Anchoring**.

3.2 THE FACM TRAINING STRATEGY

Based on our analysis, we introduce the Flow-Anchored Consistency Model (FACM). Instead of requiring specialized architectures, FACM employs a simple and effective training strategy that mixes two complementary objectives: one for stability (the anchor) and one for efficiency (the accelerator).

3.2.1 THE FACM OBJECTIVE: AN ANCHOR AND AN ACCELERATOR

The FACM training approach harnesses the stability of **Flow-Anchoring** (the FM task) and the efficiency of direct shortcut learning (the CM task) within a single training loop. The overall training loss, $\mathcal{L}_{\text{FACM}}$, is a sum of two complementary objectives:

$$\mathcal{L}_{\text{FACM}} = \mathcal{L}_{\text{FM}} + \mathcal{L}_{\text{CM}} \tag{8}$$

To enable the model to distinguish between the two tasks, each objective uses a distinct conditioning signal, c_{FM} and c_{CM}, which we detail in Section 3.2.2.

Flow Matching (FM) Loss (The Anchor) This loss component anchors the model by regressing its output towards the instantaneous velocity $\boldsymbol{v}$. The target $\boldsymbol{v}$ is constructed with a base velocity $\boldsymbol{v}_{\text{base}}$ and an optional classifier-free guidance (CFG) term:

$$\boldsymbol{v} = \boldsymbol{v}_{\text{base}} + w \cdot (\boldsymbol{v}_{\text{cond}} - \boldsymbol{v}_{\text{uncond}}), \tag{9}$$

where w is the guidance scale. The definitions of these components vary by training paradigm. For from-scratch training, the base is the conditional velocity, $\boldsymbol{v}_{\text{base}} = \boldsymbol{x}_1 - \boldsymbol{x}_0$, and the guidance term is derived from the online model $\boldsymbol{F}_\theta$ itself. In distillation, the model is initialized with weights from a pre-trained FM model. A non-trainable copy of these weights, denoted as the "teacher" F_δ, provides all velocity components for the target, with $\boldsymbol{v}_{\text{base}} = \boldsymbol{v}_{\text{uncond}} = F_\delta(\boldsymbol{x}_t, \emptyset)$, making the formula equivalent to standard CFG. Without CFG ($w = 1$), the target simply defaults to $\boldsymbol{v}_{\text{cond}}$. The FM loss then combines an L2 term with a cosine similarity term $L_{\cos}(\boldsymbol{a}, \boldsymbol{b}) = 1 - (\boldsymbol{a} \cdot \boldsymbol{b})/(\|\boldsymbol{a}\|_2 \|\boldsymbol{b}\|_2)$:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}\left[\|\boldsymbol{F}_\theta(\boldsymbol{x}_t, c_{\text{FM}}) - \boldsymbol{v}\|_2^2 + L_{\cos}(\boldsymbol{F}_\theta(\boldsymbol{x}_t, c_{\text{FM}}), \boldsymbol{v})\right]. \tag{10}$$

Consistency Model (CM) Loss (The Accelerator) This component acts as an accelerator, training the model to learn the generative shortcut. We interpret the consistency condition (Eq. 6) as a fixed-point problem, $\boldsymbol{F}_\theta = T(\boldsymbol{F}_\theta)$, where the operator is $T(\boldsymbol{F}) \triangleq \boldsymbol{v} + (1-t)\frac{\mathrm{d}\boldsymbol{F}}{\mathrm{d}t}$. The training objective is designed to solve this problem stably and iteratively. First, we compute the consistency residual $\boldsymbol{g}$ of the stop-gradient model $\boldsymbol{F}_{\theta^-}$ ($\boldsymbol{F}_{\theta^-} = \text{sg}(\boldsymbol{F}_\theta)$) :

$$\boldsymbol{g} = \boldsymbol{F}_{\theta^-}(\boldsymbol{x}_t, c_{\text{CM}}) - T(\boldsymbol{F}_{\theta^-}) = \boldsymbol{F}_{\theta^-}(\boldsymbol{x}_t, c_{\text{CM}}) - \left(\boldsymbol{v} + (1-t)\frac{\mathrm{d}\boldsymbol{F}_{\theta^-}(\boldsymbol{x}_t, c_{\text{CM}})}{\mathrm{d}t}\right). \tag{11}$$

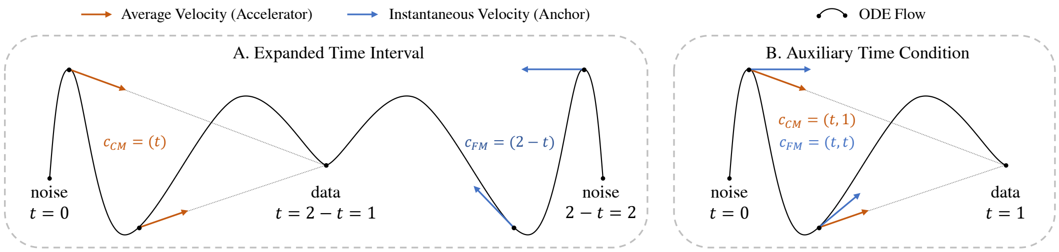


Figure 2: Two implementation strategies for the mixed-objective function in FACM. (A) **Expanded Time Interval** (default): The time domain is conceptually doubled, showing the same ODE flow on two intervals. The CM task is performed on $t \in [0, 1]$. To perform the FM task at a point t on the flow, the model is conditioned on $c_{\text{FM}} = 2 - t$, which maps the time to the alternate interval $[1, 2]$ to distinguish the two tasks. (B) **Auxiliary Time Condition**: An additional time condition r is introduced to the model. When $r = 1$, the model learns the CM task (average velocity from t to 1, orange); when $r = t$, it learns the FM task (instantaneous velocity at t, blue).

This residual $\boldsymbol{g}$ is then clamped to the range $[-1, 1]$ to prevent extreme gradients. A perturbed target is then formed as:

$$\boldsymbol{v}_{\text{tar}} = \boldsymbol{F}_{\theta^-}(\boldsymbol{x}_t, c_{\text{CM}}) - \alpha(t) \cdot \boldsymbol{g}. \tag{12}$$

Substituting the definition of $\boldsymbol{g}$ reveals the target's structure as a relaxation step for the fixed-point iteration:

$$\boldsymbol{v}_{\text{tar}} = (1 - \alpha(t))\boldsymbol{F}_{\theta^-} + \alpha(t)T(\boldsymbol{F}_{\theta^-}). \tag{13}$$

This formulation provides a stable, interpolated learning target between the current model's output and the ideal consistency target. The final CM loss component uses a norm L2 loss, L_{norm} (detailed in Appendix A.2), and is modulated by weighting functions $\alpha(t)$ and $\beta(t)$:

$$\mathcal{L}_{\text{CM}}(\theta) = \mathbb{E}\left[\beta(t) \cdot L_{\text{norm}}(\boldsymbol{F}_\theta(\boldsymbol{x}_t, c_{\text{CM}}), \boldsymbol{v}_{\text{tar}})\right]. \tag{14}$$

The combination of the interpolated target $\boldsymbol{v}_{\text{tar}}$ from the CM loss and the stabilizing flow anchor from the FM loss enables effective training. It is important to note that our specific choices for weighting and loss functions are designed to accelerate convergence, not as prerequisites for stability, which is guaranteed by the Flow-Anchoring principle.

3.2.2 IMPLEMENTATION OF THE MIXED OBJECTIVE

A key design question is how to encode the distinct conditioning signals, c_{FM} for the FM loss and c_{CM} for the CM loss, that tell the model which velocity to predict. While this conditioning can include various information like class labels, for clarity in this section, we focus only on the time-based components. We explore two effective strategies for this (Figure 2):

Expanded Time Interval We innovatively propose leveraging an expanded time domain to distinguish between the two tasks, a strategy that requires no architectural modifications. The primary CM task operates on the interval $t \in [0, 1]$, using the time directly as the condition: $c_{\text{CM}} = t$. To perform the FM task at the same point $\boldsymbol{x}_t$ (defined by t), we signal this by mapping t to the alternate interval $[1, 2]$. This is done by setting the conditioning input to $c_{\text{FM}} = 2 - t$, which makes the two conditions decoupled, symmetric, and easily distinguishable. This mapping also ensures continuity at the boundary $t = 1$, as the CM learning objective from Eq. 6 naturally converges to the FM objective's target at the boundary:

$$\lim_{t \to 1^-} \left(\boldsymbol{v} + (1 - t)\frac{\mathrm{d}\boldsymbol{F}_\theta(\boldsymbol{x}_t, t)}{\mathrm{d}t} \right) = \boldsymbol{v}. \tag{15}$$

This ensures a smooth transition between the two learning regimes.

Auxiliary Condition with a Second Timestamp r Alternatively, we can introduce a second time variable, r, to the model, making its full conditioning a tuple of (t, r). We then define $c_{\text{CM}} = (t, 1)$ and $c_{\text{FM}} = (t, t)$. This means the model signature is effectively $\boldsymbol{F}_\theta(\boldsymbol{x}_t, t, r)$. When $r = 1$, the model

Algorithm 1 FACM Training

Require: Online model $\boldsymbol{F}_\theta$, pretrained teacher F_δ, metrics d_1, d_2

Sample $\boldsymbol{x}_0, \boldsymbol{x}_1, t$

Define $c_{\text{CM}}, c_{\text{FM}}$ based on t (see Sec 3.2.2)

$\boldsymbol{x}_t \leftarrow (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1$

$\boldsymbol{v} \leftarrow \boldsymbol{x}_1 - \boldsymbol{x}_0$ ▷ For distillation, use $F_\delta(\boldsymbol{x}_t, c_{\text{FM}})$ instead

$F_{\text{FM}} \leftarrow \boldsymbol{F}_\theta(\boldsymbol{x}_t, c_{\text{FM}})$

$F_{\text{CM}}, \nabla_t \boldsymbol{F}_\theta \leftarrow \texttt{JVP}(\boldsymbol{F}_\theta, (\boldsymbol{x}_t, c_{\text{CM}}), (\boldsymbol{v}, 1))$ ▷ Simultaneous forward pass and JVP

$\overline{\boldsymbol{v}} \leftarrow \boldsymbol{v} + (1-t) \cdot \text{sg}(\nabla_t \boldsymbol{F}_\theta)$

$\boldsymbol{v}_{\text{tar}} \leftarrow (1-\alpha(t)) \cdot \text{sg}(F_{\text{CM}}) + \alpha(t) \cdot \overline{\boldsymbol{v}}$ ▷ Compute relaxation target

$\mathcal{L}_{\text{Total}} \leftarrow d_1(F_{\text{FM}}, \boldsymbol{v}) + d_2(F_{\text{CM}}, \boldsymbol{v}_{\text{tar}})$

is trained on the CM task (predicting average velocity). When $r = t$, the model is trained on the FM task (predicting instantaneous velocity). We can provide this auxiliary condition to the model through a zero-initialized time embedder, which does not alter its original structure or initial output.

As shown in our ablations (Table 3), while both methods effectively stabilize training, the **Expanded Time Interval** strategy consistently yields the best performance. We attribute this to its use of highly distinct time domains ($[0, 1]$ vs. $[1, 2]$), which provide clearer, more separable conditioning signals for the two tasks compared to the subtler differences in the **Auxiliary Time Condition** (e.g., $(t, 1)$ vs. (t, t)). The clarity of a signal appears to be more critical than the architectural change of adding a new input variable. Note that if the time schedule were $\boldsymbol{x}_t = t\boldsymbol{x}_0 + (1-t)\boldsymbol{x}_1$, the conditions for the two strategies would be t vs. $-t$ and $(t, 0)$ vs. (t, t), respectively.

3.2.3 FACM TRAINING ALGORITHM

With the objective functions and conditioning signals defined, we now present the complete FACM training strategy in Algorithm 1, which illustrates how the FM and CM losses are computed and combined in a single training step. The computation of the total derivative $\nabla_t \boldsymbol{F}_\theta$ in the CM loss is performed using a Jacobian-vector product, denoted as `JVP`. The "tangents" are given by the tuple $(\boldsymbol{v}, 1)$, where $\boldsymbol{v}$ is the tangent with respect to the input $\boldsymbol{x}_t$, and 1 is the tangent for the conditioning c_{CM}. We provide a formal proof in Appendix A.1 for why the condition tangent is always 1.

In summary, the principle of **Flow-Anchoring** offers a robust and fundamental solution. While other methods achieve stability, they do so implicitly and less effectively. For instance, the standard CM objective, when stabilized by methods like sCM (Lu & Song, 2024), can eventually converge to describe the velocity field, but this is an indirect consequence rather than a direct objective. Other approaches like MeanFlow (Geng et al., 2025) implicitly retain this capability by modeling the average velocity to arbitrary endpoints. However, this generalization creates an overly-coupled learning problem. Because learning the instantaneous velocity is just one special case within this broader objective, the two distinct tasks become entangled in a single loss, leading to optimization difficulties and sub-optimal performance. In contrast, FACM is a more direct and principled solution to ensure the model maintains a stable representation of the flow throughout training.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We empirically validate FACM on image generation benchmarks, including CIFAR-10 (Krizhevsky & Hinton, 2009) and ImageNet 256×256 (Deng et al., 2009). We evaluate models based on Fréchet Inception Distance (FID) (Heusel et al., 2017) and the Number of Function Evaluations (NFE). FACM can be trained from scratch or by distilling a pre-trained model. Our default experimental setup involves a two-stage process. We first pre-train a standard FM model, incorporating our mixed-objective conditioning as detailed in Appendix A.3 **(a)** to accelerate the subsequent distillation. We

Table 1: **Few-step generation on CIFAR-10 and ImageNet 256×256**. "×2" indicates that CFG doubles the NFE per step. Our method sets a new state-of-the-art on both datasets.

Unconditional CIFAR-10

Method	NFE	FID (↓)
Multi-NFE Baselines		
DPM-Solver++ (Lu et al., 2022)	10	2.91
EDM (Karras et al., 2022)	35	2.01
Few-NFE Methods (NFE=1)		
iCT (Song & Dhariwal, 2023)	1	<u>2.83</u>
eCT (Geng et al., 2024)	1	3.60
sCM (sCT) (Lu & Song, 2024)	1	2.85
IMM (Zhou et al., 2025)	1	3.20
MeanFlow (Geng et al., 2025)	1	2.92
FACM (Ours)	1	**2.69**
Few-NFE Methods (NFE=2)		
TRACT (Berthelot et al., 2023)	2	3.32
CD (LPIPS) (Song et al., 2023)	2	2.93
iCT-deep (Song & Dhariwal, 2023)	2	2.24
ECT (Geng et al., 2024)	2	2.11
sCM (sCT) (Lu & Song, 2024)	2	2.06
IMM (Zhou et al., 2025)	2	<u>1.98</u>
FACM (Ours)	2	**1.87**

Class-Conditional ImageNet 256×256

Method	Params	NFE	FID (↓)
Multi-NFE Baselines			
SiT-XL/2 (Ma et al., 2024)	675M	250×2	2.06
DiT-XL/2 (Peebles & Xie, 2023)	675M	250×2	2.27
REPA (Yu et al., 2025)	675M	250×2	1.42
LightningDiT (Yao et al., 2025)	675M	250×2	1.35
Few-NFE Methods (NFE=1)			
iCT (Song & Dhariwal, 2023)	675M	1	34.24
Shortcut (Frans et al., 2025)	675M	1	10.60
MeanFlow (Geng et al., 2025)	676M	1	<u>3.43</u>
FACM (Ours)	675M	1	**1.76**
Few-NFE Methods (NFE=2)			
iCT (Song & Dhariwal, 2023)	675M	2	20.30
IMM (Zhou et al., 2025)	675M	1×2	7.77
MeanFlow (Geng et al., 2025)	676M	2	<u>2.20</u>
FACM (Ours)	675M	2	**1.32**

Table 2: FID scores (NFE=2) on ImageNet 256×256 for different few-step methods applied to various multi-step FM models. † indicates our reproduction.

Method	Baseline (NFE=250×2)	sCM†	MeanFlow†	FACM (Ours)
SiT-XL/2	2.06	2.83	2.27	**2.07**
REPA	1.42	2.25	1.88	**1.52**
DiT-XL/2	2.27	2.91	2.62	**2.31**
LightningDiT	1.35	1.94	1.74	**1.32**

then distill this teacher using the FACM strategy. For few-step inference, we follow the standard multi-step sampling procedure for CMs as described in Song et al. (2023) and Appendix A.3 **(b)**. Further details on our experimental settings are provided in Appendix A.3.

4.2 MAIN RESULTS

4.2.1 COMPARISON WITH STATE-OF-THE-ART

As shown in Table 1, FACM achieves state-of-the-art results on both CIFAR-10 and ImageNet 256×256. Specifically, our method achieves FIDs of 1.76 (NFE=1) and 1.32 (NFE=2) on ImageNet 256×256 by training a LightningDiT model in latent-space, and 2.69 (NFE=1) and 1.87 (NFE=2) on CIFAR-10 by training a DDPM++ model (Ho et al., 2020) in pixel-space, significantly outperforming previous methods on both benchmarks. Remarkably, our few-step model even surpasses some multi-step baselines that require hundreds of function evaluations.

4.3 ABLATION STUDY ON THE TRAINING STRATEGY

We conduct ablation studies to validate our claims regarding the training strategy. We test on the ImageNet 256×256 dataset by distilling a pre-trained LightningDiT model. The results provide strong evidence for our central claim: the presence of the FM objective is the critical stabilizing anchor.

Different Architectures. To demonstrate the architectural agnosticism of our approach, we apply FACM, sCM, and MeanFlow to a range of state-of-the-art architectures, including SiT-XL/2, REPA, DiT-XL/2, and LightningDiT. All methods are distilled from their respective multi-step FM models. As shown in Table 2, FACM consistently achieves the lowest FID scores across all tested backbones. This highlights that Flow-Anchoring is a fundamental principle for stabilizing consistency training that is not limited to a specific model design. Notably, methods like REPA and LightningDiT, which

Table 3: Ablation on stabilization strategies. All methods are distilled from the same LightningDiT teacher (architectural modifications are only made to the student model). †: Our reproduction.

Method	**Params**	**Pre-train epochs**	**Distill epochs**	**FID (NFE=1, ↓)**	**Stable**
sCM (w/o pixel norm.)	675M	800	-	-	×
sCM (w/ pixel norm.)†	676M	800	250	3.04	✓
MeanFlow †	676M	800	250	2.75	✓
FACM (Auxiliary Condition)	676M	800	250	<u>1.93</u>	✓
FACM (Expanded Interval)	675M	800	250	**1.76**	✓
MeanFlow (from scratch)†	676M	0	1120	2.65	✓
FACM (from scratch)	675M	0	800	2.27	✓

improve performance by applying distribution constraints to the VAE, retain these advantages in the few-step generation setting.

Stabilization Strategy. To ensure a fair comparison between different stabilization methods for few-step generative models, we conduct a controlled experiment comparing sCM, MeanFlow, and our FACM. This setup isolates the effectiveness of each strategy by distilling from an identical, pre-trained LightningDiT teacher model where possible. As official code was unavailable, we reproduced the compared methods following their original descriptions (Detailed in Appendix A.3 **(c)**.). Architectural modifications integral to these methods were handled as follows: for sCM, we pre-trained the student model with the required pixel normalization layer from the start; for MeanFlow, the necessary additional time embedder was zero-initialized and incorporated during the distillation phase. Our FACM requires no architectural changes.

As shown in Table 3, while all methods achieve stable training, our FACM framework provides a more fundamental solution and superior results. The sCM distillation without pixel norm (cannot be added post-hoc via fine-tuning) collapses as predicted. While sCM incorporates various stabilization techniques, it only achieves a limited and fragile training stability; we found that reducing the tangent warmup steps or altering the time sampling schedule can lead to randomly occurring training collapse. MeanFlow achieves robust stability through its unified objective, but we diagnose that its conceptual unification of instantaneous and average velocity ($u(z,t,t) = v(z,t)$) leads to an over-coupling of the two tasks, which hinders optimization and slows convergence. In contrast, FACM explicitly separates the two tasks, allowing it to more effectively leverage the velocity field provided by the pre-trained FM teacher as a flow anchor while stably learning the shortcut objective.

Weighting Functions. We analyze the CM loss weighting functions (Table 4), which are crucial for navigating the trade-off between ensuring endpoint quality (high-SNR regions) and satisfying global consistency (low-SNR regions). We find that $1 - t^p$ and $\cos(t \cdot \pi/2)$ serve as effective general weighting solutions. In our experiments, $p = 0.5$ yielded the best results, appropriately focusing learning on the more informative low-SNR regime.

Impact of Teacher Model Convergence. To further validate our Flow-Anchoring hypothesis, we compare how sCM and our FACM perform when distilled from teacher FM models at different stages of pre-training. Figure 3(b) shows a striking divergence. The performance of sCM degrades when using a highly-converged teacher, which implies a fragile training process that requires an empirical trade-off: one must find a moderately-converged teacher, as a better flow model does not guarantee a better shortcut model. This is because the more complex flow of a highly-converged teacher is harder to model implicitly through the shortcut objective alone. In contrast, FACM's performance scales monotonically with the teacher's quality. This is a critical finding: Flow-Anchoring removes this empirical guesswork, transforming the two-stage training from a delicate balancing act into a principled optimization where a better FM model directly yields a better CM model.

4.3.1 DISCUSSION: FROM-SCRATCH TRAINING VS. DISTILLATION

While our method can be trained from scratch and achieves a competitive result (See Table 3), we identify the two-stage distillation paradigm as the more principled and practically superior approach. Attempting to learn both the anchor and the shortcut simultaneously from scratch introduces a "chicken-and-egg" problem, as the model must learn a shortcut based on a trajectory it has not

Table 4: Ablation on weighting functions for the CM loss component on ImageNet (NFE=2).

Weighting $\alpha(t)$	$\beta(t)$	FID (↓)	Weighting $\alpha(t)$	$\beta(t)$	FID (↓)
1.0	1.0	1.45	$1 - t^{0.5}$	$1 - t^{0.5}$	<u>1.37</u>
1.0	$\cos(t \cdot \pi/2)$	1.39	$\cos(t \cdot \pi/2)$	$\cos(t \cdot \pi/2)$	1.40
$1 - t^{0.5}$	1.0	1.42	$1 - t^{0.5}$	$\cos(t \cdot \pi/2)$	**1.32**

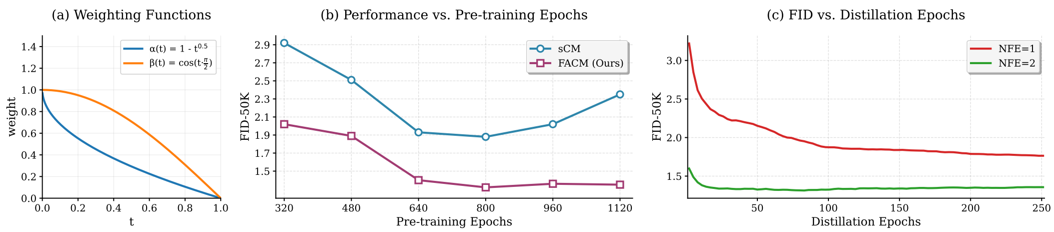


Figure 3: (a) Weighting functions used in the CM loss. (b) Performance of distilled models (NFE=2) vs. FM pre-training epochs. (c) Performance of distilled models vs. FACM distillation epochs.

yet accurately modeled. This creates an unstable "moving target" for optimization and incurs higher computational costs. In contrast, distillation from a pre-trained FM teacher provides a fixed, high-quality velocity field, offering a much more stable and well-defined learning objective. MeanFlow (Geng et al., 2025) also encounters this problem in the from-scratch setting, achieving its optimal performance only by having its objective degenerate to a Flow Matching task for a large portion of samples (e.g., 75%), which further validates our core thesis that a robust foundation in the velocity field is a prerequisite for learning stable shortcuts.

5 LIMITATIONS AND FUTURE WORK

Our primary limitation is the performance gap between 2-step and 1-step generation, indicating room to improve 1-step expressiveness. We also observe a trade-off in multi-step performance: according to Figure 3(c), after reaching the optimal 2-step result, further training continues to improve the 1-step FID, while slightly degrading the 2-step result. We hypothesize this is due to a slight deviation between the intermediate samples generated by our multi-step sampler and the true flow trajectory. Exploring sampling strategies that better align with the true flow is a promising direction for future work. Additionally, the distillation stage is computationally intensive. Each training step requires two forward passes of the teacher model for CFG, a standard forward pass of the student model, and a separate Jacobian-vector product (JVP) forward pass. This JVP computation constitutes a significant bottleneck as it is not readily accelerated by modern compilers like `torch.compile`. Finally, we plan to further validate our approach on larger-scale models and more complex conditional generation tasks, such as text-to-image synthesis.

6 CONCLUSION

We identify that the training instability of continuous-time Consistency Models stems from a "missing anchor" in the underlying velocity field. We introduce the Flow-Anchored Consistency Model (FACM), a simple, architecturally-agnostic training strategy that solves this problem by using a Flow Matching loss as an explicit anchor, and present two effective implementations for this mixed-objective design. Our method achieves new state-of-the-art FIDs on both ImageNet 256×256 (1.76 at NFE=1 and 1.32 at NFE=2) and CIFAR-10 (2.69 at NFE=1 and 1.87 at NFE=2). FACM provides a principled and effective recipe for building high-performance, few-step generative models.

REFERENCES

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

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.

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*, pp. 248–255, 2009.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations (ICLR)*, 2025.

Zhengyang Geng, Ashwini Pokle, William Luo, Justin Lin, and J Zico Kolter. Consistency models made easy. *arXiv preprint arXiv:2406.14548*, 2024.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling, 2025.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

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

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

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

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.

Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. *arXiv preprint arXiv:2410.11081*, 2024.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. *arXiv preprint arXiv:2401.08740*, 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. *arXiv preprint arXiv:2310.14189*, 2023.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *International Conference on Learning Representations*, 2025.

Linqi Zhou, Stefano Ermon, and Jiaming Song. Inductive moment matching, 2025.

A APPENDIX

A.1 ON TOTAL DERIVATIVES

In this paper, for a network $\boldsymbol{F}_\theta(\boldsymbol{x}_t, \mathcal{C}(t))$, its total derivative along the trajectory $\boldsymbol{x}_t(t) = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1$ with respect to t is given by the chain rule:

$$\frac{\mathrm{d}\boldsymbol{F}_\theta(\boldsymbol{x}_t, \mathcal{C}(t))}{\mathrm{d}t} = \frac{\partial \boldsymbol{F}_\theta}{\partial \boldsymbol{x}_t}\boldsymbol{v} + \nabla_{\mathcal{C}}\boldsymbol{F}_\theta \cdot \frac{\mathrm{d}\mathcal{C}(t)}{\mathrm{d}t}. \tag{16}$$

The term $\frac{\mathrm{d}\boldsymbol{F}_\theta(\boldsymbol{x}_t, c_{\mathrm{CM}})}{\mathrm{d}t}$ is computed for the CM task. Depending on the implementation strategy (Sec. 3.2.2), the conditioning c_{CM} can be t or a tuple $(t, 1)$. In both cases, its derivative with respect to t is effectively 1 for the time-dependent component and 0 for any constant component. Therefore, the calculation simplifies to:

$$\frac{\mathrm{d}\boldsymbol{F}_\theta(\boldsymbol{x}_t, c_{\mathrm{CM}})}{\mathrm{d}t} \approx \frac{\partial \boldsymbol{F}_\theta}{\partial \boldsymbol{x}_t}\boldsymbol{v} + \frac{\partial \boldsymbol{F}_\theta}{\partial t}, \tag{17}$$

where $\frac{\partial \boldsymbol{F}_\theta}{\partial t}$ denotes the partial derivative with respect to the explicit time argument(s) encoded in the conditioning. This term can be calculated via `JVP` using "tangents" $(\boldsymbol{v}, 1)$

A.2 NORM L2 LOSS

The CM loss component uses a norm L2 loss to improve stability against outliers. For a model prediction $\boldsymbol{p}$ and a target $\boldsymbol{y}$, let the per-sample squared error be $e = \|\boldsymbol{p} - \boldsymbol{y}\|_2^2$. The loss is then calculated as:

$$L_{\mathrm{norm}}(\boldsymbol{p}, \boldsymbol{y}) = \frac{e}{\sqrt{e + c}} \tag{18}$$

where c is a small constant. This formulation is equivalent to the adaptive L2 loss proposed in MeanFlow (Geng et al., 2025) with $p = 0.5$, and behaves similarly to a Huber loss, being robust to large errors.

A.3 EXPERIMENTAL DETAILS

(a) Pre-training Strategy. Our teacher models are standard Flow Matching models. While FACM distillation works perfectly with a standard, single-condition pre-trained teacher, we find that convergence can be accelerated by first familiarizing the teacher with our dual-task conditioning. This optional adaptation can be achieved either by pre-training from scratch with a mixed-conditioning objective (i.e., replacing the standard time conditioning with our FM-specific formats for 50% of samples) or by briefly fine-tuning a pre-trained FM model with this objective for a few epochs. Furthermore, to prevent sporadic *NaN* losses during pre-training, all our LightningDiT implementations incorporate Query-Key Normalization (QKNorm), following updates in the official repository.

(b) Sampling Strategy. Our multi-step sampling (NFE ≥ 2) follows a standard iterative refinement process. For an N-step generation, we use a simple schedule of N equally spaced timesteps $t_i = (i-1)/N$ for $i = 1, \ldots, N$. The process starts with pure noise $\boldsymbol{x}_0$. At each step i, we first compute a 1-step prediction $\hat{\boldsymbol{x}}_1$ using the model's output $\boldsymbol{F}_\theta$: $\hat{\boldsymbol{x}}_1 = \boldsymbol{x}_{t_i} + (1 - t_i)\boldsymbol{F}_\theta(\boldsymbol{x}_{t_i}, c_{\mathrm{CM}})$. If it is not the final step, we generate the input for the next step, $\boldsymbol{x}_{t_{i+1}}$, by linearly interpolating between the predicted endpoint and a new noise sample, consistent with the OT-FM framework:

$$\boldsymbol{x}_{t_{i+1}} = t_{i+1}\hat{\boldsymbol{x}}_1 + (1 - t_{i+1})z_i, \quad \text{where } z_i \sim \mathcal{N}(0, I). \tag{19}$$

The final output is the prediction from the last timestep, t_N.

(c) Reproduction Details. As the official code for MeanFlow (Geng et al., 2025) and sCM (Lu & Song, 2024) was not available, the results in our tables are from our own reproductions of their papers. Our MeanFlow implementation follows its two-time-variable conditioning and log-normal time sampling. Following their from-scratch regime, we set $t = r$ with a 75% probability for optimal performance. In the distillation paradigm, we found this setting struggled to converge and therefore did not use it. For sCM, we incorporated all necessary techniques described in their work, including pixel normalization, tangent warmup, tangent normalization, and adaptive weighting, to ensure stable training. We did not use the "TrigFlow" proposed in sCM, as we believe the specific flow construction is not key to building continuous consistency models. To ensure reproducibility, our open-source code will include our implementations of these baselines.

(e) Time Sampling Schedule. Following sCM (Lu & Song, 2024), the time $t \in [0, 1]$ is sampled according to a schedule that concentrates samples near the data endpoint ($t = 1$). We first sample a value σ from a log-normal distribution, i.e., $\ln(\sigma) \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$, and then compute t as:

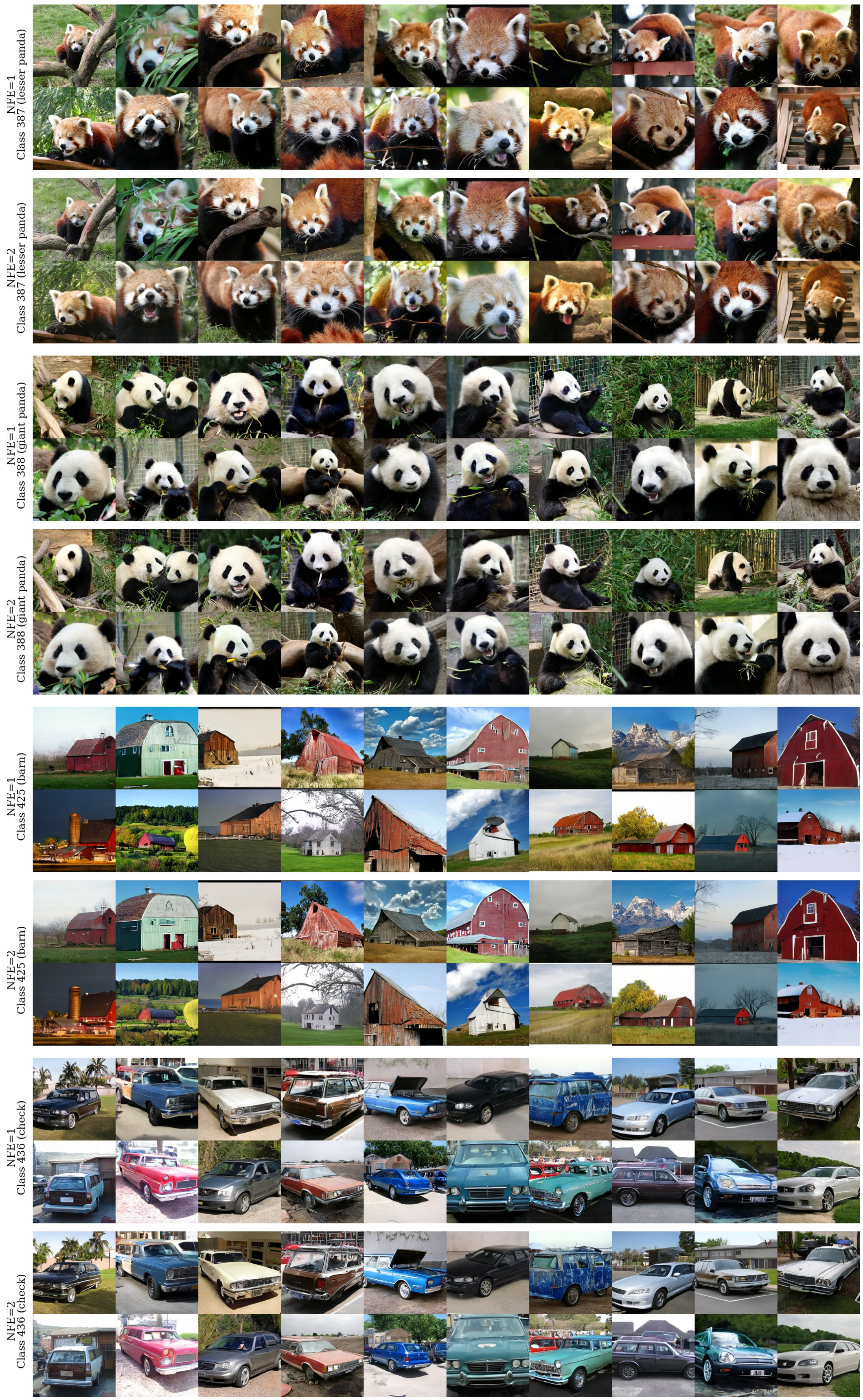
$$t = 1 - \frac{2}{\pi}\arctan(\sigma). \tag{20}$$

A.4 HYPERPARAMETERS

Table 5: Key hyperparameters for our experiments.

Hyperparameter	Value	Hyperparameter	Value	Cifar-10 Value
Optimizer	AdamW	Batch Size	1024	128
Learning Rate	1e-4	Time Sampling ($P_{\text{mean}}, P_{\text{std}}$)	(-0.8, 1.6)	(-1.0, 1.4)
Weight Decay	0	CFG Scale (w)	1.75	1.0
EMA Length (σ_{rel})	0.2	Flow Schedual	OT-FM	Simple-EDM
Norm L2 Loss c	1e-3	Dropout	0	0.1
CFG t_{low}	0.125	AdamW Betas (β_1, β_2)	(0.9, 0.999)	(0.9, 0.995)

B ADDITIONAL VISUALIZATION


NFE=1
Class 387 (lesser panda)
NFE=2
Class 387 (lesser panda)
NFE=1
Class 388 (giant panda)
NFE=2
Class 388 (giant panda)
NFE=1
Class 425 (barn)
NFE=2
Class 425 (barn)
NFE=1
Class 436 (check)
NFE=2
Class 436 (check)


NFE=1
Class 616 (knot)
NFE=2
Class 616 (knot)
NFE=1
Class 780 (schooner)
NFE=2
Class 780 (schooner)
NFE=1
Class 933 (cheeseburger)
NFE=2
Class 933 (cheeseburger)
NFE=1
Class 973 (coral reef)
NFE=2
Class 973 (coral reef)