

# Gaussian Mixture Flow Matching Models

Hansheng Chen<sup>1</sup> Kai Zhang<sup>2</sup> Hao Tan<sup>2</sup> Zexiang Xu<sup>3</sup> Fujun Luan<sup>2</sup>  
Leonidas Guibas<sup>1</sup> Gordon Wetzstein<sup>1</sup> Sai Bi<sup>2</sup>

<https://github.com/Lakonik/GMFlow>

## Abstract

Diffusion models approximate the denoising distribution as a Gaussian and predict its mean, whereas flow matching models reparameterize the Gaussian mean as flow velocity. However, they underperform in few-step sampling due to discretization error and tend to produce over-saturated colors under classifier-free guidance (CFG). To address these limitations, we propose a novel Gaussian mixture flow matching (GMFlow) model: instead of predicting the mean, GMFlow predicts dynamic Gaussian mixture (GM) parameters to capture a multi-modal flow velocity distribution, which can be learned with a KL divergence loss. We demonstrate that GMFlow generalizes previous diffusion and flow matching models where a single Gaussian is learned with an  $L_2$  denoising loss. For inference, we derive GM-SDE/ODE solvers that leverage analytic denoising distributions and velocity fields for precise few-step sampling. Furthermore, we introduce a novel probabilistic guidance scheme that mitigates the over-saturation issues of CFG and improves image generation quality. Extensive experiments demonstrate that GMFlow consistently outperforms flow matching baselines in generation quality, achieving a Precision of 0.942 with only 6 sampling steps on ImageNet 256×256.

## 1. Introduction

Diffusion probabilistic models (Sohl-Dickstein et al., 2015; Ho et al., 2020), score-based models (Song & Ermon, 2019; Song et al., 2021b), and flow matching models (Lipman et al., 2023; Liu et al., 2022) form a family of generative

<sup>1</sup>Stanford University, CA 94305, USA <sup>2</sup>Adobe Research, CA 95110, USA <sup>3</sup>Hillbot. Correspondence to: Hansheng Chen <hanshengchen@stanford.edu>.

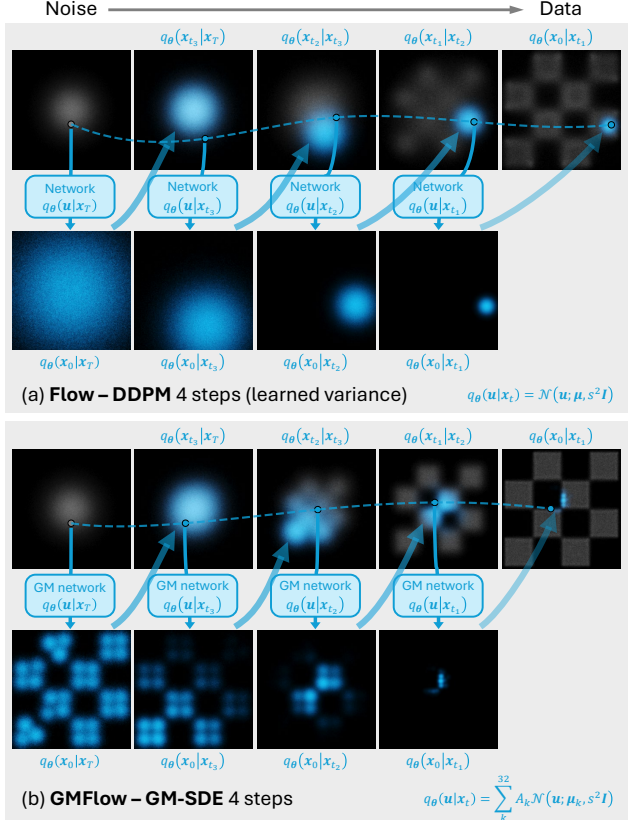


Figure 1. Comparison between vanilla diffusion (with flow velocity parameterization) and GMFlow on a 2D checkerboard distribution. (a) The vanilla diffusion model predicts mean velocity, modeling the denoising distribution  $q_\theta(x_0|x_t)$  as a single Gaussian, and then samples from a Gaussian transition distribution  $q_\theta(x_{t-\Delta t}|x_t)$ . (b) GMFlow predicts a GM for velocity and yields a multi-modal GM denoising distribution, from which the transition distribution can be analytically derived for precise next-step sampling, allowing more accurate few-step sampling (4 steps in this case).

models that share an underlying theoretical framework and have made significant advances in image and video generation (Yang et al., 2023; Po et al., 2024; Rombach et al., 2022; Saharia et al., 2022b; Podell et al., 2024; Chen et al., 2024; Esser et al., 2024; Blattmann et al., 2023; Hong et al., 2023; HaCohen et al., 2024; Kong et al., 2025). Standard

diffusion models approximate the denoising distribution as a Gaussian and train neural networks to predict its mean, and optionally its variance (Nichol & Dhariwal, 2021; Bao et al., 2022b;a). Flow matching models reparameterize the Gaussian mean as flow velocity, formulating an ordinary differential equation (ODE) that maps noise to data. These formulations remain dominant in image generation as per user studies (Artificial Analysis, 2025; Jiang et al., 2024).

However, vanilla diffusion and flow models require tens of sampling steps for high-quality generation, since Gaussian approximations hold only for small step sizes, and numeric ODE integration introduces discretization errors. In addition, high quality generation requires a higher classifier-free guidance (CFG) scale (Ho & Salimans, 2021), yet stronger CFG often leads to over-saturated colors (Saharia et al., 2022a; Kynkäänniemi et al., 2024) due to out-of-distribution (OOD) extrapolation (Bradley & Nakkiran, 2024), thus limiting overall image quality even with hundreds of steps.

To address these limitations, we deviate from previous single-Gaussian assumption and introduce Gaussian mixture flow matching (GMFlow). Unlike vanilla flow models, which predict the mean of flow velocity  $\mathbf{u}$ , GMFlow predicts the parameters of a Gaussian mixture (GM) distribution, representing the probability density function (PDF) of  $\mathbf{u}$ . This provides two key benefits: (a) our GM formulation captures more intricate denoising distributions, enabling more accurate transition estimates at larger step sizes and thus requiring fewer steps for high-quality generation (Fig. 1); (b) CFG can be reformulated by reweighting the GM probabilities rather than extrapolation, thus bounding the samples within the conditional distribution and avoiding over-saturation, thereby improving overall image quality.

We train GMFlow by minimizing the KL divergence between the predicted velocity distribution and the ground truth distribution, which we show is a generalization of previous diffusion and flow matching models (§ 3.1). For inference, we introduce novel SDE and ODE solvers that analytically derive the reverse transition distribution and flow velocity field from the predicted GM, enabling fast and precise few-step sampling (§ 3.3). Meanwhile, we develop a probabilistic guidance approach for conditional generation, which reweights the GM PDF using a Gaussian mask to enhance condition alignment (§ 3.2).

For evaluation, we compare GMFlow against vanilla flow matching baselines on both 2D toy dataset and ImageNet (Deng et al., 2009). Extensive experiments reveal that GMFlow consistently outperforms baselines equipped with advanced solvers (Lu et al., 2022; 2023; Zhao et al., 2023; Karras et al., 2022). Notably, on ImageNet 256×256, GMFlow excels in both Precision and FID metrics with fewer than 8 sampling steps; with 32 sampling steps, GMFlow achieves a state-of-the-art Precision of 0.950 (Fig. 6).

The main contributions of this paper are as follows:

- We propose GMFlow, a generalized formulation of diffusion models based on GM denoising distributions.
- We introduce a GM-based sampling framework consisting of novel SDE/ODE solvers and probabilistic guidance.
- We empirically validate that GMFlow outperforms flow matching baselines in both few- and many-step settings.

## 2. Diffusion and Flow Matching Models

In this section, we provide background on diffusion and flow matching models as the basis for GMFlow. Note that we introduce flow matching as a special diffusion parameterization since they largely overlap in practice (Albergo & Vanden-Eijnden, 2023; Gao et al., 2024).

**Forward diffusion process.** Let  $\mathbf{x} \in \mathbb{R}^D$  be a data point sampled from a distribution with its PDF denoted by  $p(\mathbf{x})$ . A typical diffusion model defines a time-dependent interpolation between the data point and a random Gaussian noise  $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ , yielding the noisy data  $\mathbf{x}_t = \alpha_t \mathbf{x} + \sigma_t \epsilon$ , where  $t \in [0, T]$  denotes the diffusion time, and  $\alpha_t, \sigma_t$  are the predefined time-dependent monotonic coefficients (noise schedule) that satisfy the boundary condition  $\mathbf{x}_0 = \mathbf{x}$ ,  $\mathbf{x}_T \approx \epsilon$ . Apparently, the marginal PDF of the noisy data  $p(\mathbf{x}_t)$  can be written as the data distribution  $p(\mathbf{x})$  convolved by a Gaussian kernel  $p(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \alpha_t \mathbf{x}_0, \sigma_t^2 \mathbf{I})$ , i.e.,  $p(\mathbf{x}_t) = \int_{\mathbb{R}^D} p(\mathbf{x}_t|\mathbf{x}_0)p(\mathbf{x}_0) d\mathbf{x}_0$ . Alternatively,  $p(\mathbf{x}_t)$  can be expressed as a previous PDF  $p(\mathbf{x}_{t-\Delta t})$  convolved by a transition Gaussian kernel  $p(\mathbf{x}_t|\mathbf{x}_{t-\Delta t})$ , given by:

$$p(\mathbf{x}_t|\mathbf{x}_{t-\Delta t}) = \mathcal{N}\left(\mathbf{x}_t; \frac{\alpha_t}{\alpha_{t-\Delta t}} \mathbf{x}_{t-\Delta t}, \left(\overbrace{\sigma_t^2 - \frac{\alpha_t^2}{\alpha_{t-\Delta t}^2} \sigma_{t-\Delta t}^2}^{\beta_{t,\Delta t}}\right) \mathbf{I}\right), \quad (1)$$

where the transition variance is denoted by  $\beta_{t,\Delta t}$  for brevity. A series of convolutions from 0 to  $T$  constructs a diffusion process over time. With infinitesimal step size  $\Delta t$ , this process can be continuously modeled by the forward-time SDE (Song et al., 2021b).

**Reverse denoising process.** Diffusion models generate samples by first sampling the noise  $\mathbf{x}_T \leftarrow \epsilon$  and then reversing the diffusion process to obtain  $\mathbf{x}_0$ . This can be achieved by recursively sampling from the reverse transition distribution  $p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t) = \frac{p(\mathbf{x}_{t-\Delta t})p(\mathbf{x}_t|\mathbf{x}_{t-\Delta t})}{p(\mathbf{x}_t)}$ . DDPM (Ho et al., 2020) approximate  $p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t)$  with a Gaussian and reparameterize its mean in terms of residual noise, which is then learned by a neural network. Its sampling process is equivalent to a first-order solver of a reverse-time SDE. Song et al. (2021b) reveal that the time evolution of the marginal distribution  $p(\mathbf{x}_t)$  described by the SDE is the same as that described by a flow ODE, which maps the noise  $\mathbf{x}_T$  to data  $\mathbf{x}_0$  deterministically. In particular, flow matching models

train a neural network to directly predict the ODE velocity field  $\frac{d\mathbf{x}_t}{dt}$ . They typically adopt a linear noise schedule, which defines  $T := 1, \alpha_t := 1 - t, \sigma_t := t$ , thus yielding a simplified velocity formulation  $\frac{d\mathbf{x}_t}{dt} = \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0|\mathbf{x}_t)}[\mathbf{u}]$ , with the random flow velocity  $\mathbf{u}$  defined as:

$$\mathbf{u} := \frac{\mathbf{x}_t - \mathbf{x}_0}{\sigma_t}. \quad (2)$$

This reveals that the velocity field  $\frac{d\mathbf{x}_t}{dt}$  represents the mean of the random velocity  $\mathbf{u}$  over the denoising distribution  $p(\mathbf{x}_0|\mathbf{x}_t) = \frac{p(\mathbf{x}_0)p(\mathbf{x}_t|\mathbf{x}_0)}{p(\mathbf{x}_t)}$ .

**Flow matching loss.** Flow models stochastically regress  $\frac{d\mathbf{x}_t}{dt}$  using randomly paired samples of  $\mathbf{x}_0$  and  $\mathbf{x}_t$ . Let  $\boldsymbol{\mu}_\theta(\mathbf{x}_t)$  denote a flow velocity neural network with learnable parameters  $\theta$ , the  $L_2$  flow matching loss is given by:

$$\mathcal{L} = \mathbb{E}_{t, \mathbf{x}_0, \mathbf{x}_t} \left[ \frac{1}{2} \|\mathbf{u} - \boldsymbol{\mu}_\theta(\mathbf{x}_t)\|^2 \right]. \quad (3)$$

In practice, additional condition signals  $\mathbf{c}$  (e.g., class label or text prompt) can be fed to the network  $\boldsymbol{\mu}_\theta(\mathbf{x}_t, \mathbf{c})$ , making it a conditioned diffusion model that learns  $p(\mathbf{x}_0|\mathbf{c})$ .

**Limitations.** Sampling errors in flow models can arise from two sources: (a) discretization errors in SDE/ODE solvers, and (b) inaccurate flow prediction  $\boldsymbol{\mu}_\theta(\mathbf{x}_t, \mathbf{c})$  due to underfitting (Karras et al., 2024). While discretization errors can be reduced by reducing step size, it increases the number of function evaluations (NFE), causing significant computational overhead. To mitigate prediction inaccuracies, CFG (Ho & Salimans, 2021) performs an extrapolation of conditional and unconditional predictions, given by  $w\boldsymbol{\mu}_\theta(\mathbf{x}_t, \mathbf{c}) + (1 - w)\boldsymbol{\mu}_\theta(\mathbf{x}_t)$ , where  $w \in [1, +\infty)$  is the guidance scale. Such a method improves image quality and condition alignment at the expense of diversity. However, higher  $w$  may lead to OOD samples and cause image over-saturation, which is often mitigated with heuristics such as thresholding (Saharia et al., 2022a; Sadat et al., 2025).

### 3. Gaussian Mixture Flow Matching Models

In this section, we introduce our GMFlow models, covering its parameterization and loss function (§ 3.1), probabilistic guidance mechanism (§ 3.2), GM-SDE/ODE solvers (§ 3.3), and other practical designs for image generation (§ 3.4). Table 1 summarizes the key differences between GMFlow and vanilla flow models. Algorithms 1 and 2 present the outlines of training and sampling schemes, respectively.

#### 3.1. Parameterization and Loss Function

Different from vanilla flow models that regress mean velocity, we model the velocity distribution  $p(\mathbf{u}|\mathbf{x}_t)$  as a Gaussian

Table 1. Comparison between vanilla flow models and GMFlow.

	Vanilla diffusion flow	GMFlow
Transition assumption	$\Delta t$ is small $\rightarrow p(\mathbf{x}_{t-\Delta t} \mathbf{x}_t)$ is approximately a Gaussian	Derive $p(\mathbf{x}_{t-\Delta t} \mathbf{x}_t)$ from the predicted GM
Network output	Mean of flow velocity $\boldsymbol{\mu}_\theta(\mathbf{x}_t)$	GM params in $q_\theta(\mathbf{u} \mathbf{x}_t) = \sum_{k=1}^K A_k \mathcal{N}(\mathbf{u}; \boldsymbol{\mu}_k, \Sigma_k)$
Training loss	$\mathbb{E}_{t, \mathbf{x}_0, \mathbf{x}_t} [\frac{1}{2} \ \mathbf{u} - \boldsymbol{\mu}_\theta(\mathbf{x}_t)\ ^2]$	$\mathbb{E}_{t, \mathbf{x}_0, \mathbf{x}_t} [-\log q_\theta(\mathbf{u} \mathbf{x}_t)]$
Sampling methods	1st-ord: Euler, DDPM... 2nd-ord: DPM++, UniPC...	1st-ord: GM-SDE/ODE 2nd-ord: GM-SDE/ODE 2
Guidance	Mean extrapolation $w\boldsymbol{\mu}_\theta(\mathbf{x}_t, \mathbf{c}) + (1 - w)\boldsymbol{\mu}_\theta(\mathbf{x}_t)$	GM reweighting $\frac{w(\mathbf{u})}{Z} q_\theta(\mathbf{u} \mathbf{x}_t, \mathbf{c})$

mixture (GM):

$$q_\theta(\mathbf{u}|\mathbf{x}_t) = \sum_{k=1}^K A_k \mathcal{N}(\mathbf{u}; \boldsymbol{\mu}_k, \Sigma_k), \quad (4)$$

where  $\{A_k, \boldsymbol{\mu}_k, \Sigma_k\}$  are dynamic GM parameters predicted by a network with parameters  $\theta$ , and  $K$  is a hyperparameter specifying the number of mixture components. To enforce  $\sum_k A_k = 1$  with  $A_k \geq 0$ , we employ a softmax activation  $A_k = \frac{\exp a_k}{\sum_k \exp a_k}$ , where  $\{a_k \in \mathbb{R}\}$  are pre-activation logits.

We train GMFlow by matching the predicted distribution  $q_\theta(\mathbf{u}|\mathbf{x}_t)$  with the ground truth velocity distribution  $p(\mathbf{u}|\mathbf{x}_t)$ . Specifically, we minimize the Kullback–Leibler (KL) divergence (Kullback & Leibler, 1951)  $\mathbb{E}_{\mathbf{u} \sim p(\mathbf{u}|\mathbf{x}_t)} [\log p(\mathbf{u}|\mathbf{x}_t) - \log q_\theta(\mathbf{u}|\mathbf{x}_t)]$ , where  $\log p(\mathbf{u}|\mathbf{x}_t)$  does not affect backpropagation and can be omitted. During training, we sample  $\mathbf{u} \sim p(\mathbf{u}|\mathbf{x}_t)$  by drawing a pair of  $\mathbf{x}_0, \mathbf{x}_t$ , and then calculate the velocity using Eq. (2). The resulting loss function is therefore reformulated as:

$$\mathcal{L} = \mathbb{E}_{t, \mathbf{x}_0, \mathbf{x}_t} [-\log q_\theta(\mathbf{u}|\mathbf{x}_t)]. \quad (5)$$

In Eq. (6), we present an expanded form of this loss function.

**Why choosing Gaussian mixture?** While there are a large family of parameterized distributions, we choose Gaussian mixture for its following desired properties:

- **Mean alignment.** The mean of the ground truth distribution  $p(\mathbf{u}|\mathbf{x}_t)$  is crucial since it decides the velocity term in the flow ODE and the drift term in the reverse-time SDE (§ C.1). When the loss in Eq. (5) is minimized (assuming sufficient network capacity), the GM distribution  $q_\theta(\mathbf{u}|\mathbf{x}_t)$  guarantees that its mean aligns with that of  $p(\mathbf{u}|\mathbf{x}_t)$ .

**Theorem 3.1.** Given any distribution  $p(\mathbf{u})$  and any symmetric positive definite matrices  $\{\Sigma_k\}$ , if  $\{a_k^*, \boldsymbol{\mu}_k^*\}$  are the optimal GM parameters w.r.t. with the objective  $\min_{\{a_k, \boldsymbol{\mu}_k\}} \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} [-\log \sum_k A_k \mathcal{N}(\mathbf{u}; \boldsymbol{\mu}_k, \Sigma_k)]$ , then the GM mean satisfies  $\sum_k A_k^* \boldsymbol{\mu}_k^* = \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} [\mathbf{u}]$ .

We provide the proof of the theorem in § C.2. It’s worth pointing out that not all distributions satisfy this property

(e.g. the mean of Laplace distribution aligns to the median instead of the mean).

- **Analytic calculation.** GM enables necessary calculations to be approached analytically (e.g., for deriving the mean and the transition distribution), as detailed in § D.
- **Expressiveness.** GMs can approximate intricate multimodal distributions. With sufficient number of components, GMs are theoretically universal approximators (Huix et al., 2024).

**Simplified covariances.** The expansion of Eq. (5) involves the inverse covariance matrix  $\Sigma_k^{-1}$ , which can lead to training instability. To mitigate this, we simplify each covariance to a scaled identity matrix, i.e.,  $\Sigma_k = s^2 \mathbf{I}$ , where  $s \in \mathbb{R}_+$  is the predicted standard deviation shared by all mixture components. It’s worth pointing out that this simplification does not limit the GM’s expressiveness, which is mainly dominated by  $K$ . Moreover, Theorem 3.1 implies that the structure of the covariance matrix is irrelevant to the accuracy of the mean velocity, mitigating the need for intricate covariances. Under this simplification, the expanded GM KL loss is reduced to:

$$\mathcal{L} = \mathbb{E}_{t, x_0, x_t} \left[ -\log \sum_{k=1}^K \exp \left( -\frac{1}{2s^2} \|\mathbf{u} - \boldsymbol{\mu}_k\|^2 - D \log s + \log A_k \right) \right], \quad (6)$$

which can be interpreted as a hybrid of regression loss to the centroids and classification loss to the components.

**Special cases of GMFlow.** GMFlow generalizes several formulations of previous diffusion and flow models. In a special case where  $K = 1, s = 1$ , Eq. (6) is identical to the  $L_2$  loss in Eq. (3). Therefore, GMFlow is a generalization of vanilla diffusion and flow models. In another case where  $\{\boldsymbol{\mu}_k\}$  are velocities towards predefined tokens and  $s \approx 0$ , then  $\{A_k\}$  represent token probabilities, making Eq. (6) analogous to categorical diffusion objectives (Gu et al., 2022; Dieleman et al., 2022; Campbell et al., 2022).

**$\mathbf{u}$ -to- $\mathbf{x}_0$  reparameterization.** While the neural network directly outputs the  $\mathbf{u}$  distribution, we can flexibly reparameterize it into an  $\mathbf{x}_0$  distribution by substituting  $\mathbf{u} = \frac{\mathbf{x}_t - \mathbf{x}_0}{\sigma_t}$  into Eq. (4), yielding  $q_\theta(\mathbf{x}_0|\mathbf{x}_t) = \sum_k A_k \mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_{xk}, s_x^2 \mathbf{I})$ , with the new parameters  $\boldsymbol{\mu}_{xk} = \mathbf{x}_t - \sigma_t \boldsymbol{\mu}_k$  and  $s_x = \sigma_t s$ . The velocity KL loss is thus equivalent to an  $\mathbf{x}_0$  likelihood loss  $\mathcal{L} = \mathbb{E}_{t, x_0, x_t} [-\log q_\theta(\mathbf{x}_0|\mathbf{x}_t)]$ .

### 3.2. Probabilistic Guidance via GM Reweighting

Vanilla CFG suffers from over-saturation due to unbounded extrapolation, which overshoots samples beyond the valid data distribution. In contrast, GMFlow can provide a well-

defined conditional distribution  $q_\theta(\mathbf{u}|\mathbf{x}_t, \mathbf{c})$ . This allows us to formulate probabilistic guidance, a principled approach that reweights the predicted distribution while preserving its intrinsic bounds and structure.

To reweight the GM PDF  $q_\theta(\mathbf{u}|\mathbf{x}_t, \mathbf{c})$  analytically, we multiply it by a Gaussian mask  $w(\mathbf{u})$ :

$$q_w(\mathbf{u}|\mathbf{x}_t, \mathbf{c}) := \frac{w(\mathbf{u})}{Z} q_\theta(\mathbf{u}|\mathbf{x}_t, \mathbf{c}), \quad (7)$$

where  $Z$  is a normalization factor. The reweighted  $q_w(\mathbf{u}|\mathbf{x}_t, \mathbf{c})$  remains a GM with analytically derived parameters (see § D.2 for derivation), and is treated as the model output for sampling.

Then, our goal is to design  $w(\mathbf{u})$  so that it enhances condition alignment without inducing OOD samples. To this end, we approximate the conditional and unconditional GM predictions  $q_\theta(\mathbf{u}|\mathbf{x}_t, \mathbf{c})$  and  $q_\theta(\mathbf{u}|\mathbf{x}_t)$  as isotropic Gaussian surrogates  $\mathcal{N}(\mathbf{u}; \boldsymbol{\mu}_c, s_c^2 \mathbf{I})$  and  $\mathcal{N}(\mathbf{u}; \boldsymbol{\mu}_u, s_u^2 \mathbf{I})$  by matching the mean and total variance of the GM (see § A.1 for details). Using these approximations, we define the unnormalized Gaussian mask as:

$$w(\mathbf{u}) := \frac{\mathcal{N}(\mathbf{u}; \boldsymbol{\mu}_c + \tilde{w} s_c \Delta \boldsymbol{\mu}_n, (1 - \tilde{w}^2) s_c^2 \mathbf{I})}{\mathcal{N}(\mathbf{u}; \boldsymbol{\mu}_c, s_c^2 \mathbf{I})}, \quad (8)$$

where  $\tilde{w} \in [0, 1)$  is the probabilistic guidance scale, and  $\Delta \boldsymbol{\mu}_n := \frac{\boldsymbol{\mu}_c - \boldsymbol{\mu}_u}{\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_u\|/\sqrt{D}}$  is the normalized mean difference.

Intuitively, the numerator in Eq. (8) shifts the conditional mean by the bias  $\tilde{w} s_c \Delta \boldsymbol{\mu}_n$  to enhance conditioning, while reducing the conditional variance according to bias-variance decomposition. Notably, for any  $\tilde{w}$ , a sample  $\mathbf{u}$  from the numerator Gaussian satisfies  $\mathbb{E}_{\mathbf{u}} [\|\mathbf{u} - \boldsymbol{\mu}_c\|^2 / D] \equiv s_c^2$ . This ensures that the original bounds of the conditional distribution are preserved.

Meanwhile, the denominator in Eq. (8) cancels out the original mean and variance of the conditional GM, so that the reweighted GM  $q_w(\mathbf{u}|\mathbf{x}_t, \mathbf{c})$  approximately inherits the adjusted mean and variance of the numerator. Since the “masking” operation retains the original GM components, the fine-grained structure of the conditional GM is preserved.

With preserved bounds and structure, probabilistic guidance reduces the risk of OOD samples compared to vanilla CFG, effectively preventing over-saturation and enhancing overall image quality.

### 3.3. GM-SDE and GM-ODE Solvers

In this subsection, we show that GMFlow enables unique SDE and ODE solvers that greatly reduce discretization errors by analytically deriving the reverse transition distribution and flow velocity field.

**GM-SDE solver.** Given the predicted  $\mathbf{x}_0$ -based GM



$q_\theta(\mathbf{x}_0|\mathbf{x}_t) = \sum_k A_k \mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_{\mathbf{x}_k}, s_{\mathbf{x}}^2 \mathbf{I})$ , the reverse transition distribution  $q_\theta(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t)$  can be analytically derived (see § C.3 for derivation):

$$q_\theta(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t) = \sum_{k=1}^K A_k \mathcal{N}(\mathbf{x}_{t-\Delta t}; c_1 \mathbf{x}_t + c_2 \boldsymbol{\mu}_{\mathbf{x}_k}, (c_3 + c_2^2 s_{\mathbf{x}}^2) \mathbf{I}), \quad (9)$$

with the coefficients  $c_1 = \frac{\sigma_{t-\Delta t}^2}{\sigma_t^2} \frac{\alpha_t}{\alpha_{t-\Delta t}}$ ,  $c_2 = \frac{\beta_{t,\Delta t}}{\sigma_t^2} \alpha_{t-\Delta t}$ ,  $c_3 = \frac{\beta_{t,\Delta t}}{\sigma_t^2} \sigma_{t-\Delta t}^2$ . By recursively sampling from  $q_\theta(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t)$ , we obtain a GM approximation to the reverse-time SDE solution. Alternatively, we can implement the solver by first sampling  $\hat{\mathbf{x}}_0 \sim q_\theta(\mathbf{x}_0|\mathbf{x}_t)$  and then sampling  $p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t, \hat{\mathbf{x}}_0) = \mathcal{N}(\mathbf{x}_{t-\Delta t}; c_1 \mathbf{x}_t + c_2 \hat{\mathbf{x}}_0, c_3 \mathbf{I})$  (Song et al., 2021a) (see § C.3 for details). Theoretically, if  $q_\theta(\mathbf{x}_0|\mathbf{x}_t)$  is accurate, the GM-SDE solution incurs no error even in a single step. In practice, a smaller step size  $\Delta t$  increases reliance on the mean (see § C.4 for details) over the shape of the distribution, which can be more accurate as per Theorem 3.1.

**GM-ODE solver.** While GMFlow supports standard ODE integration by converting its GM prediction into the current mean velocity, it also enables a unique sampling scheme with reduced discretization errors by analytically deriving the flow velocity field for any time  $\tau < t$ , thereby facilitating sub-step integration without additional neural network evaluations. Specifically, given the  $\mathbf{x}_0$ -based GM  $q_\theta(\mathbf{x}_0|\mathbf{x}_t)$  at  $\mathbf{x}_t$ , we show in § C.5 that the denoising distribution at  $\mathbf{x}_\tau$  can be derived as:

$$\hat{q}(\mathbf{x}_0|\mathbf{x}_\tau) = \frac{p(\mathbf{x}_\tau|\mathbf{x}_0)}{Z \cdot p(\mathbf{x}_t|\mathbf{x}_0)} q_\theta(\mathbf{x}_0|\mathbf{x}_t), \quad (10)$$

where  $Z$  is a normalization factor, and  $\hat{q}(\mathbf{x}_0|\mathbf{x}_\tau)$  is also a GM with analytically derived parameters. This allows us to instantly estimate GMFlow’s next-step prediction at  $\mathbf{x}_\tau$  from its current prediction at  $\mathbf{x}_t$ , without re-evaluating the neural network. The velocity can therefore be derived by reparameterizing  $\hat{q}(\mathbf{x}_0|\mathbf{x}_\tau)$  in terms of  $\mathbf{u}$  and computing its mean. This enables the GM-ODE solver, which integrates a curved trajectory along the analytic velocity field via multiple Euler sub-steps between  $t$  and  $t - \Delta t$  (Algorithm 2 line 18–22). Theoretically, if  $q_\theta(\mathbf{x}_0|\mathbf{x}_t)$  is accurate, the GM-ODE solution also incurs no error. In practice, the velocity field is only locally accurate as  $\tau \rightarrow t$  and  $\mathbf{x}_\tau \rightarrow \mathbf{x}_t$ , thus multiple network evaluations are still required.

**Second-order multistep GM solvers.** Vanilla diffusion models often use Adams–Bashforth-like second-order multistep solvers (Lu et al., 2023), which extrapolate the mean  $\mathbf{x}_0$  predictions of the last two steps to estimate the next midpoint, and then apply an Euler update. Analogously, we extend this approach to GMs. Given GM predictions at times  $t$  and  $t + \Delta t$ , we first convert the latter to time  $t$  via

Eq. (10), yielding  $\hat{q}(\mathbf{x}_0|\mathbf{x}_t)$ . In an ideal scenario where both GMs are accurate,  $\hat{q}(\mathbf{x}_0|\mathbf{x}_t)$  perfectly matches  $q_\theta(\mathbf{x}_0|\mathbf{x}_t)$ ; otherwise, their discrepancy is extrapolated following the Adams–Bashforth scheme. To perform GM extrapolation, we adopt a GM reweighting scheme similar to that in § 3.2. More details are provided in § A.2.

### 3.4. Practical Designs

**Pixel-wise factorization.** For high-dimensional data such as images, Gaussian mixture models often suffer from mode collapse. To address this, we treat each pixel (in the latent grid for latent diffusion (Rombach et al., 2022)) as an independent low-dimensional GM, making the training loss the sum of per-pixel GM KL terms.

**Spectral sampling.** Due to the factorization, image generation under GM-SDE solvers performs the sampling step  $\hat{\mathbf{x}}_0 \sim q_\theta(\mathbf{x}_0|\mathbf{x}_t)$  independently for each pixel, neglecting spatial correlations. To address this, we adopt a spectral sampling technique that imposes correlations through the frequency domain. Specifically, we generate a spatially correlated Gaussian random field from a learnable power spectrum. The per-pixel Gaussian samples are then mapped to GM samples via Knothe–Rosenblatt transport (Knothe, 1957; Rosenblatt, 1952). More details are provided in § A.3.

**Transition loss.** Empirically, we observe that the GM KL loss may still induce minor gradient spikes when  $s$  becomes small. To stabilize training, we adopt a modified loss based on the transition distributions (Eq. (9)), formulated as  $\mathcal{L}_{\text{trans}} = \mathbb{E}_{t, \mathbf{x}_{t-\Delta t}, \mathbf{x}_t} [-\log q_\theta(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t)]$ , where we define  $\Delta t := \lambda t$  with the transition ratio hyperparameter  $\lambda \in (0, 1]$ . When  $\lambda = 1$ , the loss is equivalent to the original; when  $\lambda < 1$ , the transition distribution has a lower bound of variance  $c_3$ , which improves training stability. The modified training scheme is described in Algorithm 1.

## 4. Experiments

To evaluate the effectiveness of GMFlow, we compare it against vanilla flow matching baselines on two datasets: (a) a simple 2D checkerboard distribution, which facilitates visualization of sample histograms and analysis of underlying mechanisms; (b) the class-conditioned ImageNet dataset (Deng et al., 2009), a challenging benchmark that demonstrates the practical advantages of GMFlow for large-scale image generation.

### 4.1. Sampling a 2D Checkerboard Distribution

In this subsection, we compare GMFlow with the vanilla flow model on a simple 2D checkerboard distribution, following the experimental settings of Lipman et al. (2023). All configurations adopt the same 5-layer MLP architecture for denoising 2D coordinates, differing only in their output

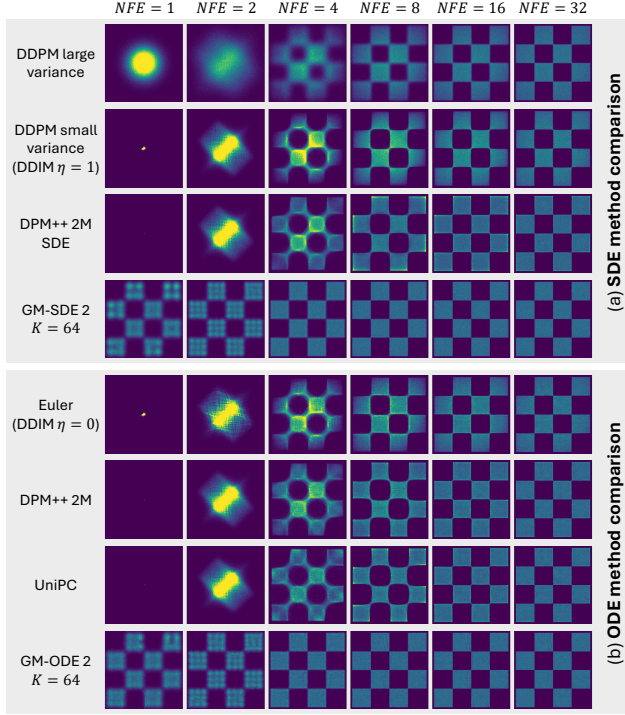


Figure 2. Comparison among vanilla flow models with different solvers and GMFlow. For both SDE and ODE, our method achieves higher quality in few-step sampling.

channels. For GMFlow, we train multiple models with different numbers of GM components  $K$  (with  $\lambda = 0.9$ ); for GM-ODE sampling, we use  $n = \lceil 128/NFE \rceil$  sub-steps.

**Comparison against flow model baseline.** In Fig. 2, we compare the 2D sample histograms of GMFlow using second-order GM solvers (GM-SDE/ODE 2) against the vanilla flow matching baseline using established SDE and ODE solvers (Lu et al., 2023; Zhao et al., 2023; Ho et al., 2020; Song et al., 2021a). Notably, vanilla flow models require approximately 8 steps to achieve a reasonable histogram and 16–32 steps for high-quality sampling. In contrast, GMFlow ( $K = 64$ ) can approximate the checkerboard in a single step and achieve high-quality sampling in 4 steps. Moreover, for vanilla flow models, samples generated by first-order solvers (DDPM, Euler) tend to concentrate toward the center, whereas those from second-order solvers (DPM++, UniPC) concentrate near the outer edges. In contrast, GMFlow samples are highly uniform. This validates GMFlow’s advantage in few-step sampling and multi-modal distribution modeling.

**Impact of the GM component count  $K$ .** Fig. 3 (a) demonstrates that increasing  $K$  significantly improves few-step sampling results, leading to sharper histograms. Notably, GM-SDE sampling with  $K = 1$  is theoretically equivalent to DDPM with learned variance (Nichol & Dhariwal, 2021), which produces a blurrier histogram. This further highlights

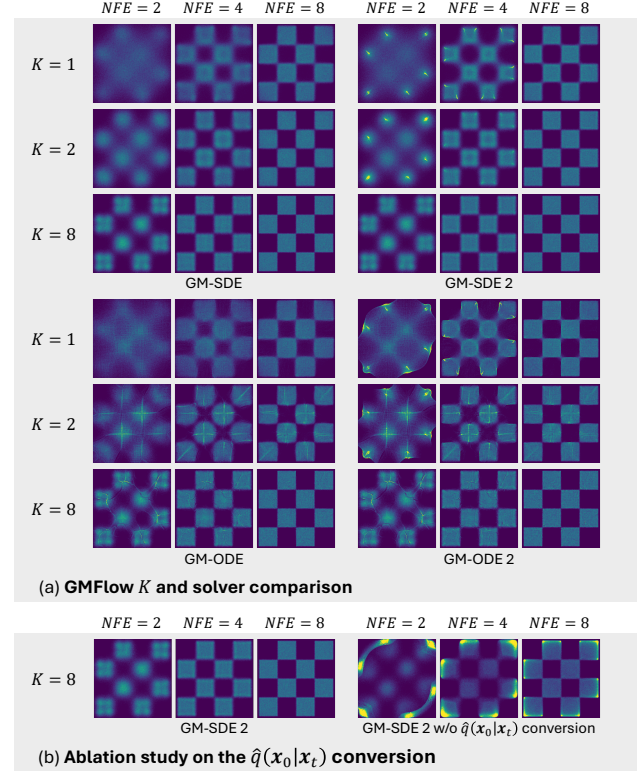


Figure 3. (a) Comparison of first- and second-order GM-SDE and GM-ODE solvers with varying  $NFE$  and GM components  $K$ . Increasing  $K$  improves few-step sampling results. Second-order solvers produce sharper histograms with fewer outliers than first-order solvers. (b) Ablation study on the  $\hat{q}(x_0|x_t)$  conversion in second-order solvers. Removing this conversion causes samples to overshoot and concentrate at the edges.

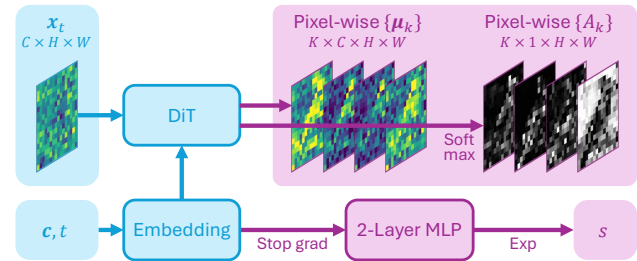


Figure 4. Architecture of GMFlow-DiT. The original DiT (Peebles & Xie, 2023) is shown in blue, and the modified output layers are shown in purple.

the expressiveness of GMFlow.

**Second-order GM solvers.** Fig. 3 (a) also compares our first- and second-order GM solvers across various NFEs and GM component counts. Comparing the left (first-order) and right (second-order) columns, we observe that second-order solvers produce sharper histograms and better suppress outliers, particularly when  $NFE$  and  $K$  are small. With  $K = 8$ , the GM probabilities are sufficiently accu-

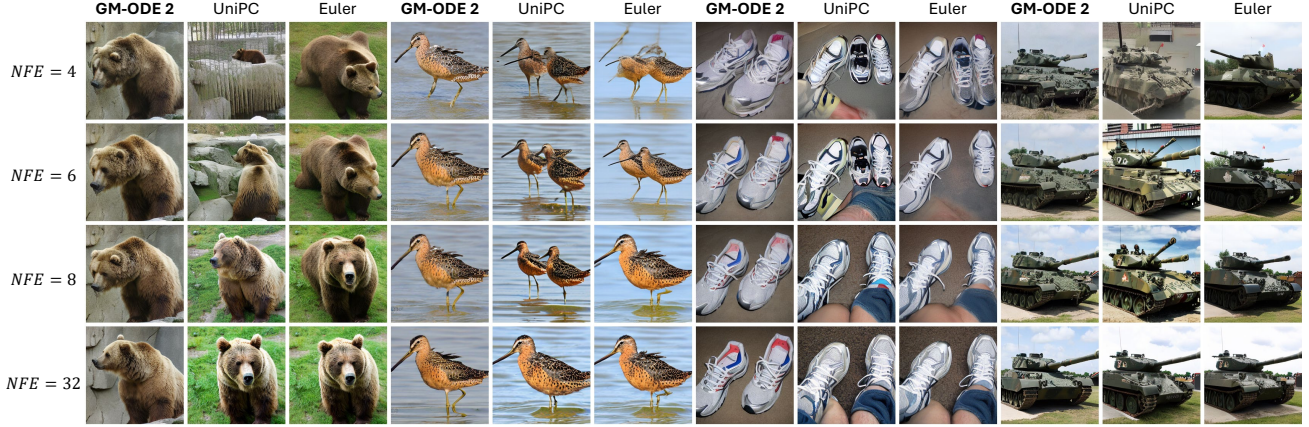


Figure 5. Qualitative comparisons (at best Precision) among GMFlow (GM-ODE 2) and vanilla flow model baselines (UniPC and Euler). GMFlow produces consistent results across various NFEs, whereas baselines struggle in few-step sampling, exhibiting distorted structures.

rate that the second-order solvers do not make a difference, aligning with our theoretical analysis.

**Ablation studies.** To validate the importance of the  $\hat{q}(x_0|x_t)$  conversion in second-order GM solvers, we conduct an ablation study by removing the conversion and directly extrapolating the  $x_0$  distributions of the last two steps, similar to DPM++ (Lu et al., 2023). As shown in Fig. 3 (b), removing this conversion introduces an overshooting bias, similar to DPM++ 2M SDE ( $NFE = 8$ ), underscoring its importance in maintaining sample uniformity.

## 4.2. ImageNet Generation

For image generation evaluation, we benchmark GMFlow against vanilla flow baselines on class-conditioned ImageNet 256×256. We train a Diffusion Transformer (DiT-XL/2) (Peebles & Xie, 2023; Vaswani et al., 2017) using the flow matching objective (Eq. (3)) as the baseline, and then adapt it into GMFlow-DiT by expanding its output channels and training it with the transition loss ( $\lambda = 0.5$ ). As illustrated in Fig. 4, GMFlow-DiT produces  $K$  weight maps and mean maps as the pixel-wise parameters  $\{A_k, \mu_k\}$ . Meanwhile, a tiny two-layer MLP separately predicts  $s$  using only the time  $t$  and condition  $c$  as inputs. Empirically, we find this design to be more stable than predicting  $s$  using the main DiT. The adaptation results in only a 0.2% increase in network parameters for  $K = 8$ . During inference, we apply the orthogonal projection technique by Sadat et al. (2025) to both the baseline and GMFlow since it universally improves Precision. Additional training and inference details are provided in § A.4.

**Evaluation protocol.** For quantitative evaluation, we adopt the standard metrics used in ADM (Dhariwal & Nichol, 2021), including Fréchet Inception Distance (FID) (Heusel et al., 2017), Inception Score (IS), and Precision–Recall (Kynkäänniemi et al., 2019). These metrics are highly sen-

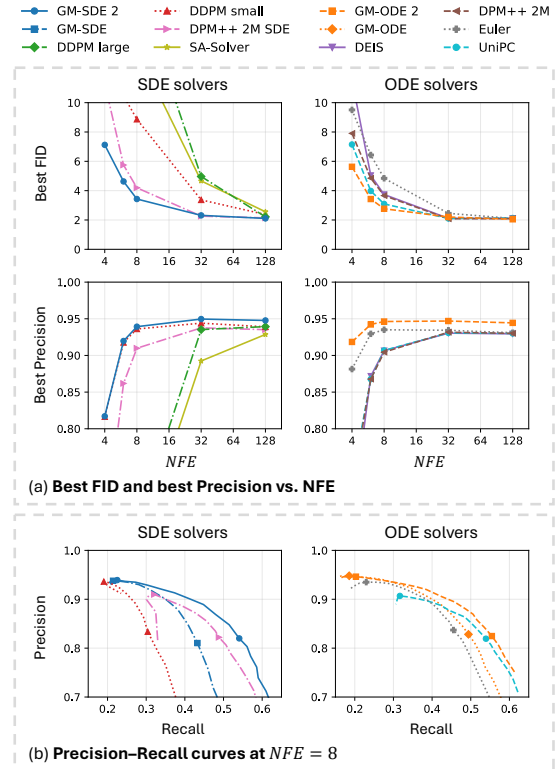


Figure 6. (a) Comparison of the best Precision and best FID among GMFlow and vanilla flow model baselines using different solvers across varying NFEs on ImageNet. For best FID, GMFlow significantly outperforms the baselines in few-step sampling; for best Precision, GMFlow consistently excels across different NFEs. (b) Precision–Recall curves of different methods at  $NFE = 8$ . Points corresponding to the best FID and best Precision are marked on the curves. GMFlow achieves superior Precision and Recall.

sitive to the classifier-free guidance (CFG) scale: while smaller CFG scales ( $w \approx 1.4$ ) often yield the best FID by balancing diversity and quality, human users generally favor



higher CFG scales ( $w > 3.0$ ) for the best perceptual quality, which also leads to the best Precision. The best Recall and IS values typically occur outside these CFG ranges and are less representative of typical usage. Therefore, for a fair and complete evaluation, we sweep over the CFG scale  $w$  or the probabilistic guidance scale  $\tilde{w}$  for each model, and report the best FID and best Precision. We also present Precision–Recall curves, illustrating the quality–diversity trade-off more comprehensively. Additionally, following Sadat et al. (2025), we report the Saturation metric at the best Precision setting to assess over-saturation.

**Comparison against flow model baselines.** For baselines, we test the vanilla flow model using various first-order solvers (DDPM (Ho et al., 2020), Euler (DDIM) (Song et al., 2021a)) and advanced second-order solvers (DPM++ (Lu et al., 2023), DEIS (Zhang & Chen, 2023), UniPC (Zhao et al., 2023), SA-Solver (Xue et al., 2023)). Details on adapting these solvers for flow matching are presented in § A.5. In Fig. 6 (a), we compare these baselines with our GMFlow ( $K = 8$ ) model equipped with second-order GM solvers (GM-SDE/ODE 2). GMFlow consistently achieves superior Precision in both SDE and ODE sampling across various NFEs. Notably, GM-ODE 2 reaches a Precision of 0.942 in just 6 steps, while GM-SDE 2 attains a state-of-the-art Precision of 0.950 in 32 steps. For FID, GMFlow outperforms baselines in few-step settings ( $NFE \leq 8$ ) and remains competitive in many-step settings ( $NFE \geq 32$ ). Fig. 6 (b) further illustrates that the Precision–Recall curves of first- and second-order GM solvers consistently outperform those of their baseline counterparts. Qualitative comparisons are presented in Fig. 5.

**Saturation assessment.** Table 2 compares the Saturation metrics of different methods at their best Precision. GMFlow ( $K = 8$ ) effectively reduces over-saturation, achieving Saturation levels closest to real data. Visual comparisons in Fig. 5 further support this finding. Additionally, ablating the orthogonal projection technique results in the same trend: GMFlow consistently achieves the best Saturation, while baseline methods perform even worse without orthogonal projection.

**Impact of the GM component count  $K$ .** In Fig. 7, we analyze the impact of the number of GM components on the metrics. Few-step sampling ( $NFE = 8$ ) benefits the most from increasing GM components, particularly with the GM-SDE 2 solver. The metrics generally saturate at  $K = 8$ . Beyond this point, the GM-SDE 2 Precisions decline because spectral sampling introduces larger numerical errors when more Gaussian components are used. Additionally, Table 3 presents the time-averaged negative log-likelihood (NLL) values of data under the predicted distribution  $q_\theta(x_0|x_t, c)$ , showing that increasing the number of Gaussians significantly reduces NLL, suggesting its potential benefits for

Table 2. ImageNet evaluation results at best Precision ( $NFE = 32$ ). The reported Saturation values (Sadat et al., 2025) are relative to the real data statistics (Saturation=0.318).

Method	Orthogonal projection	Guidance	Precision↑	Saturation
DDPM small	✓	$w = 3.3$	0.944	+0.032
DPM++ 2M SDE	✓	$w = 2.9$	0.938	+0.032
GM-SDE	✓	$\tilde{w} = 0.47$	0.950	−0.019
GM-SDE 2	✓	$\tilde{w} = 0.39$	0.950	−0.019
Euler	✓	$w = 3.3$	0.934	+0.052
UniPC	✓	$w = 3.3$	0.931	+0.048
GM-ODE	✓	$\tilde{w} = 0.47$	0.947	−0.024
GM-ODE 2	✓	$\tilde{w} = 0.47$	0.947	−0.024
DDPM small		$w = 3.3$	0.939	+0.056
DPM++ 2M SDE		$w = 2.9$	0.933	+0.054
GM-SDE		$\tilde{w} = 0.27$	0.943	+0.023
Euler		$w = 3.3$	0.931	+0.064
UniPC		$w = 3.3$	0.929	+0.060
GM-ODE		$\tilde{w} = 0.27$	0.933	+0.016

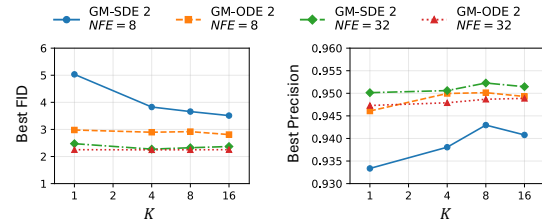


Figure 7. Best FIDs and best Precisions of GMFlow models with varying numbers of GM components  $K$ .

Table 3. Validation NLL of ImageNet training images (latents) under different GMFlow configurations.

Method	$K = 1$	$K = 4$	$K = 8$	$K = 16$	$K = 16$ +spectral
NLL (bits/dim)	0.346	0.263	0.242	0.224	0.173

posterior sampling applications.

**Ablation studies.** Table 4 presents ablation study results evaluating the impact of various design choices in our method. For GM-SDE, removing spectrum sampling (A2) leads to a noticeable degradation in FID. Replacing our probabilistic guidance with vanilla CFG (A3)—i.e., naively shifting the mean of the predicted GM—results in severe quality degradation (lower Precision) and over-saturation, which is also evident in Fig. 8. Replacing GM-SDE 2 with the second-order DPM++ SDE solver (A4) worsens both FID and Precision. Reducing the transition loss to the original KL loss (A5) degrades FID. Finally, for GM-ODE 2, directly applying ODE integration without taking sub-steps (B1) leads to significantly worse FID.

**Inference time.** GMFlow only alters the output layer and uses solvers based on simple arithmetic operations. As a result, it adds only 0.005 sec of overhead per step (batch size 125, A100 GPU) compared to its flow-matching counterpart,



which is minimal compared to the total inference time of 0.39 sec per step—most of which is spent on DiT.

## 5. Related Work

Prior works (Nichol & Dhariwal, 2021; Bao et al., 2022b;a) extend standard diffusion models by learning the variance of denoising distributions, which is effectively a special case of GMFlow with  $K = 1$  and learnable  $s$ . GMS (Guo et al., 2023) further extends this approach to third-order moments and fits a bimodal GM to the moments during inference. In § B.3, we present a comparison between GMFlow ( $K = 2$ ) and GMS. Conversely, Xiao et al. (2022) employ a generative adversarial network (Goodfellow et al., 2014) to capture a more expressive distribution, though adversarial diffusion typically relies on additional objectives for better quality and stability (Jolicœur-Martineau et al., 2021; Kim et al., 2024). However, none of these methods address deterministic ODE sampling.

A growing line of work trains networks to directly predict ODE integrals or denoised samples, enabling extremely few-step sampling (Salimans & Ho, 2022; Song et al., 2023; Yin et al., 2024b; Sauer et al., 2024). These approaches fall under the category of distillation methods, whose quality typically remains bounded by the many-step performance of standard diffusion models, unless supplemented by additional adversarial training (Kim et al., 2024; Yin et al., 2024a).

Efforts to refine CFG beyond thresholding (Saharia et al., 2022a) and orthogonal projection (Sadat et al., 2025) include disabling CFG in early steps (Kynkäänniemi et al., 2024), which compromises Precision, and adding Langevin corrections (Bradley & Nakkiran, 2024), which reduces efficiency.

Beyond diffusion models, Gaussian mixtures have also been employed in other generative models. DeLiGAN (Gurumurthy et al., 2017) and GM-GAN (Ben-Yosef & Weinshall, 2018) enhance the latent expressiveness of GANs through the introduction of mixture priors. GIVT (Tschannen et al., 2024) models the output of an autoregressive Transformer as a Gaussian mixture distribution, enabling continuous data sampling and outperforming its quantization-based counterpart.

## 6. Conclusion

In this work, we introduced GMFlow, a generalization of diffusion and flow models that represents flow velocity as a Gaussian mixture, offering greater expressiveness for capturing complex multi-modal structures. We derived principled SDE/ODE solvers unique to this approach and proposed a novel probabilistic guidance technique to eliminate

Table 4. Ablation studies on GMFlow ( $K = 8$ ,  $NFE = 8$ ) using ImageNet evaluation.

ID	Method	Best FID↓	Best Precision↑	Saturation @ Best Prec.
<b>A0</b>	<b>Full model (GM-SDE 2)</b>	3.43	0.939	−0.003
A1	A0 → GM-SDE	6.96 (+3.53)	0.938 (−0.001)	+0.009
A2	A1 w/o spec. sampling	8.98 (+2.02)	0.940 (+0.002)	−0.022
A3	A2 → Vanilla CFG	9.02 (+0.04)	0.917 (−0.023)	+0.049
A4	A0 → DPM++ 2M SDE	4.59 (+1.16)	0.912 (−0.027)	−0.062
A5	A0 → $\lambda = 1.0$	4.49 (+1.06)	0.941 (+0.002)	−0.019
<b>B0</b>	<b>Full model (GM-ODE 2)</b>	2.77	0.946	−0.028
B1	B0 w/o sub-steps	7.47 (+4.70)	0.947 (+0.001)	−0.031

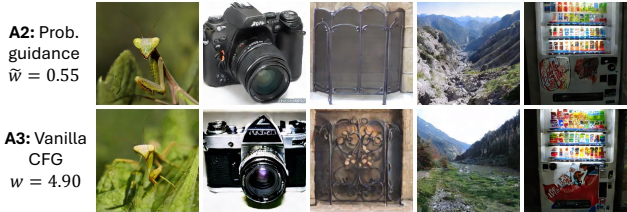


Figure 8. Ablation study on probabilistic guidance, comparing samples from Table 4 A2 and A3.

over-saturation. Extensive experiments demonstrated that GMFlow significantly improves few-step generation while enhancing overall sample quality. This framework lays the foundation for potential future research in both theoretical and practical directions, including applications such as posterior sampling with GMFlow priors.

## Limitations

To apply Gaussian mixture to high-dimensional data, we adopted pixel-wise factorization for image generation, which may not fully exploit the potential of GMFlow, leaving rooms for further development.

## Acknowledgments

We thank Liwen Wu, Lvmin Zhang, and Guandao Yang for discussions and feedback. Part of this work was done while Hansheng Chen was an intern at Adobe Research. This project was partially supported by Qualcomm Innovation Fellowship, ARL grant W911NF-21-2-0104, and Vannevar Bush Faculty Fellowship.

## Impact Statement

This paper presents work whose goal is to advance the field of generation models. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

## References

- Albergo, M. S. and Vanden-Eijnden, E. Building normalizing flows with stochastic interpolants. In *ICLR*, 2023. URL <https://arxiv.org/abs/2209.15571>.
- Artificial Analysis. Artificial analysis text to image model leaderboard, 2025. URL <https://huggingface.co/spaces/ArtificialAnalysis/Text-to-Image-Leaderboard>. Accessed: 2025-01-25.
- Bao, F., Li, C., Sun, J., Zhu, J., and Zhang, B. Estimating the optimal covariance with imperfect mean in diffusion probabilistic models. In *ICML*, 2022a.
- Bao, F., Li, C., Zhu, J., and Zhang, B. Analytic-DPM: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. In *ICLR*, 2022b. URL <https://openreview.net/forum?id=0xiJLKH-ufZ>.
- Ben-Yosef, M. and Weinshall, D. Gaussian mixture generative adversarial networks for diverse datasets, and the unsupervised clustering of images, 2018. URL <https://arxiv.org/abs/1808.10356>.
- Blattmann, A., Dockhorn, T., Kulal, S., Mendelevitch, D., Kilian, M., Lorenz, D., Levi, Y., English, Z., Voleti, V., Letts, A., Jampani, V., and Rombach, R. Stable video diffusion: Scaling latent video diffusion models to large datasets, 2023. URL <https://arxiv.org/abs/2311.15127>.
- Bradley, A. and Nakkiran, P. Classifier-free guidance is a predictor-corrector, 2024. URL <https://arxiv.org/abs/2408.09000>.
- Campbell, A., Benton, J., Bortoli, V. D., Rainforth, T., Deligiannidis, G., and Doucet, A. A continuous time framework for discrete denoising models. In *NeurIPS*, 2022.
- Chen, J., YU, J., GE, C., Yao, L., Xie, E., Wang, Z., Kwok, J., Luo, P., Lu, H., and Li, Z. Pixart- $\alpha$ : Fast training of diffusion transformer for photorealistic text-to-image synthesis. In *ICLR*, 2024. URL <https://openreview.net/forum?id=eAKmQPe3m1>.
- Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *CVPR*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.
- Dettmers, T., Lewis, M., Shleifer, S., and Zettlemoyer, L. 8-bit optimizers via block-wise quantization. In *ICLR*, 2022.
- Dhariwal, P. and Nichol, A. Q. Diffusion models beat GANs on image synthesis. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *NeurIPS*, 2021. URL <https://openreview.net/forum?id=AAWuCVzaVt>.
- Dieleman, S., Sartran, L., Roshannai, A., Savinov, N., Ganin, Y., Richemond, P. H., Doucet, A., Strudel, R., Dyer, C., Durkan, C., Hawthorne, C., Leblond, R., Grathwohl, W., and Adler, J. Continuous diffusion for categorical data, 2022. URL <https://arxiv.org/abs/2211.15089>.
- Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., Podell, D., Dockhorn, T., English, Z., Lacey, K., Goodwin, A., Marek, Y., and Rombach, R. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*, 2024.
- Gao, R., Hoogeboom, E., Heek, J., Bortoli, V. D., Murphy, K. P., and Salimans, T. Diffusion meets flow matching: Two sides of the same coin, 2024. URL <https://diffusionflow.github.io/>.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. (eds.), *NeurIPS*, volume 27. Curran Associates, Inc., 2014. URL [https://proceedings.neurips.cc/paper\\_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf).
- Gu, S., Chen, D., Bao, J., Wen, F., Zhang, B., Chen, D., Yuan, L., and Guo, B. Vector quantized diffusion model for text-to-image synthesis. In *CVPR*, 2022.
- Guo, H. A., Lu, C., Bao, F., Pang, T., YAN, S., Du, C., and Li, C. Gaussian mixture solvers for diffusion models. In *NeurIPS*, 2023. URL <https://openreview.net/forum?id=0NuseeBuB4>.
- Gurumurthy, S., Sarvadevabhatla, R. K., and Babu, R. V. Deligan : Generative adversarial networks for diverse and limited data. In *CVPR*, 2017.
- HaCohen, Y., Chiprut, N., Brazowski, B., Shalem, D., Moshe, D., Richardson, E., Levin, E., Shiran, G., Zabari, N., Gordon, O., Panet, P., Weissbuch, S., Kulikov, V., Bitterman, Y., Melumian, Z., and Bibi, O. Ltx-video: Realtime video latent diffusion, 2024. URL <https://arxiv.org/abs/2501.00103>.
- Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017.

- Ho, J. and Salimans, T. Classifier-free diffusion guidance. In *NeurIPS Workshop*, 2021.
- Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.
- Hong, W., Ding, M., Zheng, W., Liu, X., and Tang, J. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. In *ICLR*, 2023. URL <https://openreview.net/forum?id=rB6TpjAuSRy>.
- Huix, T., Korba, A., Durmus, A., and Moulines, E. Theoretical guarantees for variational inference with fixed-variance mixture of gaussians. In *ICML*, 2024.
- Jiang, D., Ku, M., Li, T., Ni, Y., Sun, S., Fan, R., and Chen, W. GenAI arena: An open evaluation platform for generative models. In *NeurIPS Datasets and Benchmarks Track*, 2024. URL <https://openreview.net/forum?id=0Gmi8TkUC7>.
- Jolicœur-Martineau, A., Piché-Taillefer, R., Mitliagkas, I., and des Combes, R. T. Adversarial score matching and improved sampling for image generation. In *ICLR*, 2021. URL <https://openreview.net/forum?id=eLfqMl3z3lq>.
- Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022.
- Karras, T., Aittala, M., Kynkäänniemi, T., Lehtinen, J., Aila, T., and Laine, S. Guiding a diffusion model with a bad version of itself. In *NeurIPS*, 2024. URL <https://openreview.net/forum?id=bq6fVPVs3s>.
- Kim, D., Lai, C.-H., Liao, W.-H., Murata, N., Takida, Y., Uesaka, T., He, Y., Mitsufuji, Y., and Ermon, S. Consistency trajectory models: Learning probability flow ODE trajectory of diffusion. In *ICLR*, 2024. URL <https://openreview.net/forum?id=ymjI8feDTD>.
- Knothe, H. Contributions to the theory of convex bodies. *Michigan Mathematical Journal*, 4(1):39–52, 1957.
- Kong, W., Tian, Q., Zhang, Z., Min, R., Dai, Z., Zhou, J., Xiong, J., Li, X., Wu, B., Zhang, J., Wu, K., Lin, Q., Yuan, J., Long, Y., Wang, A., Wang, A., Li, C., Huang, D., Yang, F., Tan, H., Wang, H., Song, J., Bai, J., Wu, J., Xue, J., Wang, J., Wang, K., Liu, M., Li, P., Li, S., Wang, W., Yu, W., Deng, X., Li, Y., Chen, Y., Cui, Y., Peng, Y., Yu, Z., He, Z., Xu, Z., Zhou, Z., Xu, Z., Tao, Y., Lu, Q., Liu, S., Zhou, D., Wang, H., Yang, Y., Wang, D., Liu, Y., Jiang, J., and Zhong, C. Hunyuanvideo: A systematic framework for large video generative models, 2025. URL <https://arxiv.org/abs/2412.03603>.
- Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images, 2009.
- Kullback, S. and Leibler, R. A. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1): 79–86, 1951.
- Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., and Aila, T. Improved precision and recall metric for assessing generative models. In *NeurIPS*, 2019.
- Kynkäänniemi, T., Aittala, M., Karras, T., Laine, S., Aila, T., and Lehtinen, J. Applying guidance in a limited interval improves sample and distribution quality in diffusion models. In *NeurIPS*, 2024.
- Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. In *ICLR*, 2023. URL <https://openreview.net/forum?id=PqvMRDCJT9t>.
- Liu, X., Gong, C., and Liu, Q. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.
- Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *ICLR*, 2019.
- Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. DPM-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *NeurIPS*, 2022. URL [https://openreview.net/forum?id=2uAaGwlp\\_V](https://openreview.net/forum?id=2uAaGwlp_V).
- Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models, 2023. URL <https://arxiv.org/abs/2211.01095>.
- Nichol, A. and Dhariwal, P. Improved denoising diffusion probabilistic models. In *ICML*, 2021.
- Peebles, W. and Xie, S. Scalable diffusion models with transformers. In *ICCV*, 2023.
- Po, R., Yifan, W., Golyanik, V., Aberman, K., Barron, J. T., Bermano, A. H., Chan, E. R., Dekel, T., Holynski, A., Kanazawa, A., Liu, C. K., Liu, L., Mildenhall, B., Nießner, M., Ommer, B., Theobalt, C., Wonka, P., and Wetzstein, G. State of the art on diffusion models for visual computing. In *Eurographics STAR*, 2024.
- Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., and Rombach, R. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *ICLR*, 2024. URL <https://openreview.net/forum?id=di52zR8xgf>.



- Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In Bach, F. and Blei, D. (eds.), *ICML*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1530–1538, Lille, France, 07–09 Jul 2015. PMLR. URL <https://proceedings.mlr.press/v37/rezende15.html>.
- Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.
- Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 234–241, 2015.
- Rosenblatt, M. Remarks on a multivariate transformation. *The Annals of Mathematical Statistics*, 23(3):470–472, 1952.
- Sadat, S., Hilliges, O., and Weber, R. M. Eliminating oversaturation and artifacts of high guidance scales in diffusion models. In *ICLR*, 2025.
- Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., Ho, J., Fleet, D. J., and Norouzi, M. Photorealistic text-to-image diffusion models with deep language understanding. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *NeurIPS*, volume 35, 2022a.
- Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al. Photorealistic text-to-image diffusion models with deep language understanding. In *NeurIPS*, 2022b.
- Salimans, T. and Ho, J. Progressive distillation for fast sampling of diffusion models. In *ICLR*, 2022.
- Sauer, A., Lorenz, D., Blattmann, A., and Rombach, R. Adversarial diffusion distillation. In *ECCV*, pp. 87–103, 2024.
- Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, pp. 2256–2265, 2015.
- Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *ICLR*, 2021a.
- Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, 2019.
- Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021b.
- Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. Consistency models. In *ICML*, 2023.
- Tschannen, M., Eastwood, C., and Mentzer, F. Givt: Generative infinite-vocabulary transformers. In *ECCV*, pp. 292–309, 2024.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *NeurIPS*, volume 30. Curran Associates, Inc., 2017.
- von Platen, P., Patil, S., Lozhkov, A., Cuenca, P., Lambert, N., Rasul, K., Davaadorj, M., Nair, D., Paul, S., Berman, W., Xu, Y., Liu, S., and Wolf, T. Diffusers: State-of-the-art diffusion models. <https://github.com/huggingface/diffusers>, 2022.
- Xiao, Z., Kreis, K., and Vahdat, A. Tackling the generative learning trilemma with denoising diffusion GANs. In *ICLR*, 2022. URL <https://openreview.net/forum?id=JprM0p-q0Co>.
- Xue, S., Yi, M., Luo, W., Zhang, S., Sun, J., Li, Z., and Ma, Z.-M. SA-solver: Stochastic adams solver for fast sampling of diffusion models. In *NeurIPS*, 2023. URL <https://openreview.net/forum?id=f6a9XVFYIo>.
- Yang, L., Zhang, Z., Song, Y., Hong, S., Xu, R., Zhao, Y., Zhang, W., Cui, B., and Yang, M.-H. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56(4), November 2023. ISSN 0360-0300. doi: 10.1145/3626235. URL <https://doi.org/10.1145/3626235>.
- Yin, T., Gharbi, M., Park, T., Zhang, R., Shechtman, E., Durand, F., and Freeman, W. T. Improved distribution matching distillation for fast image synthesis. In *NeurIPS*, 2024a. URL <https://openreview.net/forum?id=tQukGCDaNT>.
- Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W. T., and Park, T. One-step diffusion with distribution matching distillation. In *CVPR*, 2024b.
- Zhang, Q. and Chen, Y. Fast sampling of diffusion models with exponential integrator. In *ICLR*, 2023. URL <https://openreview.net/forum?id=Loek7hfb46P>.
- Zhao, W., Bai, L., Rao, Y., Zhou, J., and Lu, J. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. In *NeurIPS*, 2023.

## A. Additional Technical Details

### A.1. Details on Gaussian Surrogates

For a GM distribution of the form  $\sum_{k=1}^K A_k \mathcal{N}(\boldsymbol{\mu}_k, s^2 \mathbf{I})$ , we approximate it with an isotropic Gaussian surrogate  $\mathcal{N}(\boldsymbol{\mu}', s'^2 \mathbf{I})$  by matching the first two moments. The surrogate mean is computed as:

$$\boldsymbol{\mu}' = \sum_{k=1}^K A_k \boldsymbol{\mu}_k. \quad (11)$$

To determine the surrogate variance, we equate the trace of the GM's covariance matrix (i.e., the GM's total variance) with the trace of the surrogate covariance, which yields:

$$s'^2 = \frac{1}{D} \sum_{k=1}^K A_k \|\boldsymbol{\mu}_k - \boldsymbol{\mu}'\|^2 + s^2. \quad (12)$$

### A.2. Details on Second-Order GM Solvers

For GM extrapolation, we follow the method in § 3.2 and fit the isotropic Gaussian surrogates  $\mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_+, s_+^2 \mathbf{I}) \approx q_\theta(\mathbf{x}_0|\mathbf{x}_t)$ ,  $\mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_-, s_-^2 \mathbf{I}) \approx \hat{q}(\mathbf{x}_0|\mathbf{x}_t)$ . We then define a Gaussian mask:

$$\rho(\mathbf{x}_0) = \frac{\mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_+ + \Delta\boldsymbol{\mu}, (s_+^2 - \frac{\|\Delta\boldsymbol{\mu}\|^2}{D}) \mathbf{I})}{\mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_+, s_+^2 \mathbf{I})}, \quad (13)$$

where  $\Delta\boldsymbol{\mu} = \frac{\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-}{2}$ , so that  $\boldsymbol{\mu}_+ + \Delta\boldsymbol{\mu}$  represents the extrapolated mean at the next midpoint. The final extrapolated GM is obtained via the reweighting formulation:

$$q_{\text{ext}}(\mathbf{x}_0|\mathbf{x}_t) = \frac{\rho(\mathbf{x}_0)}{Z} q_\theta(\mathbf{x}_0|\mathbf{x}_t). \quad (14)$$

We then feed  $q_{\text{ext}}(\mathbf{x}_0|\mathbf{x}_t)$  to the first-order GM-SDE or ODE solver as a substitution for  $q_\theta(\mathbf{x}_0|\mathbf{x}_t)$ .

Empirically, we observe that second-order GM solvers using the above formulation slightly underperform with higher guidance scales, which is a common issue with multistep solvers (Lu et al., 2023; Zhao et al., 2023). To address this, we rescale the mean difference  $\Delta\boldsymbol{\mu}$  by an empirical factor  $\sqrt{\max(0, 1 - \frac{(\tilde{w}^2 + c_a)s_c^2}{\Delta t^2})}$  (with the hyperparameter  $c_a = 0.005$ ), so that a high  $\tilde{w}$  practically disables multistep extrapolation.

### A.3. Details on Spectral Sampling

Due to pixel-wise factorization, image generation under GM-SDE solvers performs the sampling step  $\hat{\mathbf{x}}_0 \sim q_\theta(\mathbf{x}_0|\mathbf{x}_t)$  independently for each pixel, neglecting spatial correlations. Spectral sampling addresses this by establishing an invertible mapping between a frequency-space

---

#### Algorithm 1: Complete GMFlow training scheme.

---

**Input:** Data distribution  $p(\mathbf{x}_0, \mathbf{c})$ , transition ratio  $\lambda$   
**Output:** Network params  $\boldsymbol{\theta}$

- 1 Initialize network params  $\boldsymbol{\theta}$
- 2 **for** sample  $\{\mathbf{x}_0, \mathbf{c}\} \sim p(\mathbf{x}_0, \mathbf{c})$  **do**
- 3   **if** use logit-normal **then**
- 4     Sample  $t \sim \text{LogitNormal}(0, 1)$
- 5   **else**
- 6     Sample  $t \sim U(0, 1)$
- 7    $\Delta t \leftarrow \lambda t$
- 8   Sample  $\mathbf{x}_{t-\Delta t} \sim p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_0)$
- 9   Sample  $\mathbf{x}_t \sim p(\mathbf{x}_t|\mathbf{x}_{t-\Delta t})$
- 10   Predict GM params in  $q_\theta(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t, \mathbf{c})$  // Eq. (9)
- 11    $\mathcal{L}_{\text{trans}} \leftarrow -\log q_\theta(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t, \mathbf{c})$
- 12   Predict magnitude spectrum  $\mathbf{s}_F$
- 13    $\mathcal{L}_{\text{spec}} \leftarrow -\log \mathcal{N}(\text{vec}(\mathbf{z}_r); \mathbf{0}, \text{diag}(\mathbf{s}_F)^2)$  // Eq. (20)
- 14   Backpropagate and update  $\boldsymbol{\theta}$

---



---

#### Algorithm 2: Complete GMFlow sampling scheme.

---

**Input:** Steps  $NFE$ , sub-steps  $n$ , guidance scale  $\tilde{w}$ , condition  $\mathbf{c}$ , network params  $\boldsymbol{\theta}$   
**Output:**  $\mathbf{x}_0$

- 1  $t \leftarrow 1, \Delta t = \frac{1}{NFE}, \mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \text{Cache} \leftarrow \{\}$
- 2 **while**  $t > 0$  **do**
- 3   Predict GM params in  $q_\theta(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c})$
- 4   **if**  $\tilde{w} > 0$  **then**
- 5     Predict GM params in  $q_\theta(\mathbf{x}_0|\mathbf{x}_t)$
- 6     Compute  $q_w(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c})$  // Eq. (7)
- 7      $q_\theta(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c}) \xleftarrow{\text{param}} q_w(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c})$
- 8   **if** use 2nd-order **then**
- 9     **if**  $\text{Cache} \neq \{\}$  **then**
- 10       Compute  $\hat{q}(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c})$  from  $\text{Cache}$  // Eq. (10)
- 11        $\text{Cache} \leftarrow \{\mathbf{x}_t, q_\theta(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c})\}$
- 12       Compute  $q_{\text{ext}}(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c})$  // Eq. (14)
- 13        $q_\theta(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c}) \xleftarrow{\text{param}} q_{\text{ext}}(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c})$
- 14     **else**
- 15        $\text{Cache} \leftarrow \{\mathbf{x}_t, q_\theta(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c})\}$
- 16   **if** use GM-SDE **then**
- 17     **if** use spectral sampling **then**
- 18       Predict magnitude spectrum  $\mathbf{s}_F$
- 19       Sample  $\hat{\mathbf{x}}_0 \sim q_{\hat{\theta}}(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c})$  // Eq. (19)
- 20     **else**
- 21       Sample  $\hat{\mathbf{x}}_0 \sim q_\theta(\mathbf{x}_0|\mathbf{x}_t, \mathbf{c})$
- 22     Sample  $\mathbf{x}_{t-\Delta t} \sim \mathcal{N}(\mathbf{x}_{t-\Delta t}; c_1 \mathbf{x}_t + c_2 \hat{\mathbf{x}}_0, c_3 \mathbf{I})$
- 23   **else if** use GM-ODE **then**
- 24      $\tau \leftarrow t, h = \Delta t/n$
- 25     **while**  $\tau > t - \Delta t$  **do**
- 26       Compute  $\hat{q}(\mathbf{x}_0|\mathbf{x}_\tau, \mathbf{c})$  // Eq. (10)
- 27        $\mathbf{x}_{\tau-h} = \mathbf{x}_\tau - h \mathbb{E}_{\mathbf{x}_0 \sim \hat{q}(\mathbf{x}_0|\mathbf{x}_\tau, \mathbf{c})} [\frac{\mathbf{x}_\tau - \mathbf{x}_0}{\sigma_\tau}]$
- 28        $\tau \leftarrow \tau - h$
- 29    $t \leftarrow t - \Delta t$

---

distribution (power spectrum) and the pixel-space GM distribution. During training, the power spectrum is optimized via

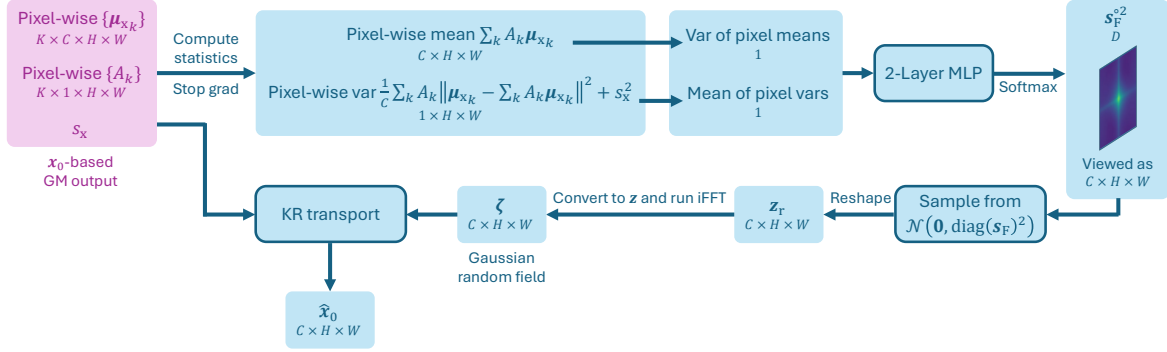


Figure 9. The optional spectral sampling pipeline used during inference. The spectrum MLP takes the statistics of pixel-wise GMs as input, and predicts the power spectrum  $s_F^2$ . Alternatively, if probabilistic guidance or second-order GM solvers are employed, one can re-use the mean and variance from the numerator in the Gaussian mask to compute the input statistics, which serves as a good approximation.

a likelihood loss, while during inference, frequency-space samples are transformed back into pixel space to introduce spatial correlation.

We establish an invertible mapping using two building blocks: the fast Fourier transform (FFT) and Knothe–Rosenblatt (KR) transport (Knothe, 1957; Rosenblatt, 1952). The FFT bridges a frequency-space Gaussian distribution and a pixel-space Gaussian distribution, while KR transport bridges each per-pixel Gaussian to its corresponding Gaussian mixture (GM) distribution.

During training, given a real image  $\mathbf{x}_0 \in \mathbb{R}^{C \times H \times W}$  and the factorized denoising distribution  $q_\theta(\mathbf{x}_0|\mathbf{x}_t) = \prod_{i=1}^H \prod_{j=1}^W \sum_{k=1}^K A_{k,ij} \mathcal{N}(\mathbf{x}_{0,ij}|\mu_{k,ij}, s^2 \mathbf{I})$ , KR transport defines an invertible mapping for each pixel:

$$T_{ij} : \mathbf{x}_{0,ij} \mapsto \zeta_{ij}, \quad (15)$$

such that each  $\zeta_{ij}$  can be viewed as a standard Gaussian sample. We then assemble all  $\zeta_{ij}$  into a tensor  $\zeta \in \mathbb{R}^{C \times H \times W}$  and apply a forward 2D FFT with orthogonal normalization, yielding the complex frequency representation  $\mathbf{z} = \text{FFT}(\zeta) \in \mathbb{C}^{C \times H \times W}$ . Since  $\mathbf{z}$  is Hermitian symmetric, we can derive a real-valued representation  $\mathbf{z}_r$  while preserving invertibility:

$$\mathbf{z}_r = \text{Re}[\mathbf{z}] + \text{Im}[\mathbf{z}]. \quad (16)$$

Finally, we impose a zero-mean Gaussian prior on  $\mathbf{z}_r$ , given by  $\mathcal{N}(\text{vec}(\mathbf{z}_r); \mathbf{0}, \text{diag}(s_F)^2)$ , where  $s_F \in \mathbb{R}_+^D$  represents the magnitude spectrum. To dynamically model  $s_F \in \mathbb{R}_+^D$ , we use a tiny two-layer MLP, which takes the mean of per-pixel GM variances and the variance of per-pixel GM means as input, and outputs the power spectrum  $s_F^2$  with a softmax activation, where  $(\cdot)^{\circ 2}$  stands for element-wise square. With this invertible mapping and the spectral Gaussian prior, we can derive the model’s spectrum-enhanced denoising PDF using the change of variables technique in a similar way to normalizing flow models (Rezende & Mohamed, 2015),

which can be expressed as:

$$q_\theta^S(\mathbf{x}_0|\mathbf{x}_t) = \left| \det \left( \frac{\partial \text{vec}(\mathbf{z}_r)}{\partial \mathbf{x}_0} \right) \right| \mathcal{N}(\text{vec}(\mathbf{z}_r); \mathbf{0}, \text{diag}(s_F)^2), \quad (17)$$

where the absolute determinant can be easily derived using the properties of FFT and KR transport:

$$\begin{aligned} \left| \det \left( \frac{\partial \text{vec}(\mathbf{z}_r)}{\partial \mathbf{x}_0} \right) \right| &= \left| \det \left( \frac{\partial \text{vec}(\mathbf{z}_r)}{\partial \text{vec}(\zeta)} \right) \right| \cdot \left| \det \left( \frac{\partial \text{vec}(\zeta)}{\partial \mathbf{x}_0} \right) \right| \\ &= 1 \cdot \frac{q_\theta(\mathbf{x}_0|\mathbf{x}_t)}{\mathcal{N}(\text{vec}(\zeta); \mathbf{0}, \mathbf{I})} \\ &= \frac{q_\theta(\mathbf{x}_0|\mathbf{x}_t)}{\mathcal{N}(\text{vec}(\mathbf{z}_r); \mathbf{0}, \mathbf{I})}. \end{aligned} \quad (18)$$

Substituting Eq. (18) into Eq. (17), we obtain:

$$q_\theta^S(\mathbf{x}_0|\mathbf{x}_t) = q_\theta(\mathbf{x}_0|\mathbf{x}_t) \frac{\mathcal{N}(\text{vec}(\mathbf{z}_r); \mathbf{0}, \text{diag}(s_F)^2)}{\mathcal{N}(\text{vec}(\mathbf{z}_r); \mathbf{0}, \mathbf{I})}. \quad (19)$$

With the derived PDF, the entire model can be trained by minimizing the negative log-likelihood (NLL) of data samples under the PDF, which inherently includes the GM KL loss (Eq. (5) and (6)) as one of its terms. Therefore, the total loss consists of the GM KL loss (replaced with the transition loss in practice) and an additional spectral loss, defined as:

$$\begin{aligned} \mathcal{L}_{\text{spec}} &= \mathbb{E}_{t, \mathbf{x}_0, \mathbf{x}_t} \left[ -\log \frac{\mathcal{N}(\text{vec}(\mathbf{z}_r); \mathbf{0}, \text{diag}(s_F)^2)}{\mathcal{N}(\text{vec}(\mathbf{z}_r); \mathbf{0}, \mathbf{I})} \right] \\ &= \mathbb{E}_{t, \mathbf{x}_0, \mathbf{x}_t} \left[ \frac{1}{2} \|(\text{diag}(s_F)^{-1} - \mathbf{I}) \text{vec}(\mathbf{z}_r)\|^2 + \log \det(\text{diag}(s_F)) \right]. \end{aligned} \quad (20)$$

In practice, we stop the gradient flow through KR transport to prevent spectral learning from influencing the main GMFlow model.

During inference, we employ the spectral sampling pipeline illustrated in Fig. 9.



#### A.4. ImageNet Experiment Details.

We train both the baseline and GMFlow-DiT on ImageNet 256×256 with a batch size of 4096 images across 16 A100 GPUs, using a total training schedule of 200K iterations. We adopt the 8-bit AdamW (Dettmers et al., 2022; Loshchilov & Hutter, 2019) optimizer with a fixed learning rate of 0.0002. Following Stable Diffusion 3 (Esser et al., 2024), both models sample  $t$  from a logit-normal distribution during training (Algorithm 1), which accelerates convergence.

While we set the transition ratio to  $\lambda = 0.5$  in the main experiments, the results in Fig. 7 and Table 3 are based on an earlier design iteration that uses randomly sampled transition ratios  $\lambda \sim \text{LogitNormal}(0, 1)$ . This setting slightly increases Precision and decreases Recall, though the overall differences remain minor.

We densely evaluate the models across different guidance scales to identify the optimal FID and Precision. For vanilla CFG, we use guidance scales  $w$  from the set  $\{1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.1, 2.3, 2.6, 2.9, 3.3, 3.7, 4.3, 4.9, 5.7, 6.5\}$ ; for probabilistic guidance, we use probabilistic guidance scales  $\tilde{w}$  from the set  $\{0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.11, 0.13, 0.16, 0.19, 0.23, 0.27, 0.33, 0.39, 0.47, 0.55, 0.65, 0.75\}$ . In practice (Algorithm 2), we implement probabilistic guidance on  $x_0$ -based GMs, which is equivalent to guidance on  $u$ .

For inference with GM-ODE solvers, we generally set the number of sub-steps to  $n = \max(\frac{128}{NFE}, 2)$ , which performs well when  $NFE \geq 8$ . For the exception when  $NFE = 4$  or 6, we observe that reducing  $n$  to 8 yields better performance.

In Table 3, the time-averaged NLL values are computed on 50K samples from the training dataset using the following equation:

$$NLL = \frac{1}{D} \mathbb{E}_{t, x_0, x_t} [-\log_2 q_\theta(x_0|x_t)], \quad (21)$$

where  $t \sim U(0, 1)$  and  $D = C \times H \times W$ . This is equivalent to the original training loss ( $\lambda = 1$ ) with a uniform time sampling scheme. When spectral prior is enabled, we replace  $q_\theta(x_0|x_t)$  with  $q_\theta^S(x_0|x_t)$  (Eq. (19)).

#### A.5. Adapting Diffusion Solvers for Flow Matching

For DDPM solvers (Ho et al., 2020), we implement them as special cases of GM-SDE solver with  $K = 1$ . The original DDPM solvers include a large variance variant ( $\beta$ ) and a small variance variant ( $\tilde{\beta}$ ). These are equivalent to setting  $s = \frac{1}{\sqrt{\alpha_t^2 + \sigma_t^2}}$  and  $s = 0$ , respectively.

For DDIM solvers (Song et al., 2021a), its stochastic variant ( $\eta = 1$ ) is equivalent to DDPM with small variance, whereas its deterministic variant ( $\eta = 0$ ) is equivalent to Euler solver in flow matching models.

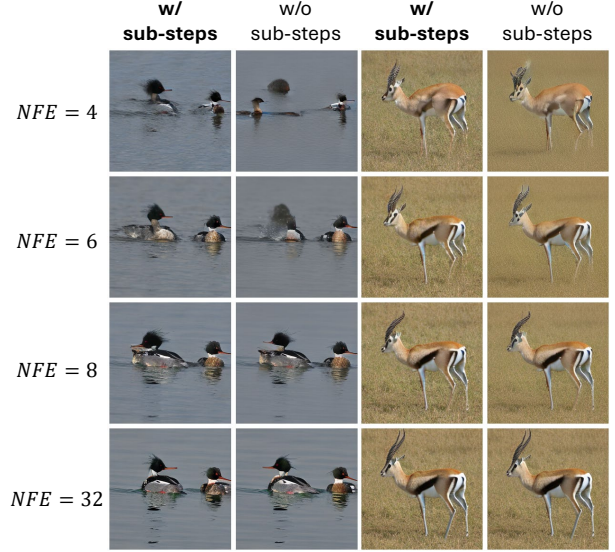


Figure 10. Qualitative comparison (at best Precision) from the ablation study on GM-ODE sub-steps. Without sub-steps (forcing  $n = 1$ ), few-step sampling tends to produce over-smoothed textures and poor detail.

For DPM++ (Lu et al., 2023), DEIS (Zhang & Chen, 2023), UniPC (Zhao et al., 2023), and SA-Solver (Xue et al., 2023), we use their Diffusers implementations (von Platen et al., 2022) and rescale their noise schedules to match the flow matching noise schedule. This approach is similar to how the EDM Euler solver (Karras et al., 2022) rescales a variance-preserving diffusion model into a variance-exploding one at test time.

## B. Additional Experiment Results

### B.1. Ablation Study on GM-ODE Sub-Steps

In addition to the results in Table 4 (B0 and B1), Fig. 10 presents additional qualitative results from the ablation study on GM-ODE sub-steps, using  $K = 8$  and the GM-ODE 2 solver. As shown in the figure, sub-steps are essential for producing detailed textures when  $NFE < 8$ .

### B.2. Additional Qualitative Comparison

Fig. 11 presents a comparison among uncensored random samples (conditioned on random class labels) from GMFlow and vanilla flow matching baselines, showcasing their respective best results under the many-step setting. Overall, the images generated by GMFlow exhibit more natural color saturation and, in some cases, improved structural coherence. These observations are consistent with the quantitative results shown in Fig. 6 and Table 2.

Table 5. FIDs on CIFAR-10 unconditional image generation. Competitor results are sourced from Guo et al. (2023).

NFE	10	25	50	100
DDPM large	205.31	84.71	37.35	14.81
DDPM small	34.76	16.18	11.11	8.38
SN-DDPM	16.33	6.05	4.19	3.83
GMS	13.80	5.48	4.00	3.46
<b>GM-SDE 2</b>	<b>9.11</b>	<b>4.16</b>	<b>3.79</b>	<b>3.76</b>

### B.3. Comparison with Moment Matching Methods

Table 5 presents a quantitative comparison among GM-Flow ( $K = 2$ ), GMS (Guo et al., 2023), SN-DDPM (Bao et al., 2022a), and DDPM (Ho et al., 2020) for CIFAR-10 (Krizhevsky et al., 2009) unconditional image generation using SDE sampling. The GMFlow model is trained from scratch using the same U-Net architecture (Ronneberger et al., 2015; Ho et al., 2020) as its competitors, but with modified output layers as in GM-DiT. The results demonstrate that GMFlow significantly outperforms its competitors in few-step sampling.

## C. Additional Theoretical Analysis

### C.1. Details on the SDE and ODE Background

For notation references, this subsection briefly recaps the SDE and ODE formulation of diffusion models by Song et al. (2021b).

The forward-time SDE of the diffusion process can be derived by taking an infinitesimal step size  $\Delta t$  when applying the transition Gaussian kernel in Eq. (1), which yields:

$$d\mathbf{x} = f_t \mathbf{x}_t dt + g_t d\mathbf{w}_t \quad (22)$$

where  $f_t = \frac{1}{\alpha_t} \frac{d\alpha_t}{dt}$ ,  $g_t = \sqrt{\frac{d\beta_t}{d\Delta t}}|_{\Delta t=0}$ , and  $\mathbf{w}_t$  is the standard Wiener process. The corresponding reverse-time SDE is:

$$d\mathbf{x}_t = (f_t \mathbf{x}_t - g_t^2 \mathbf{s}_t(\mathbf{x}_t)) dt + g_t d\bar{\mathbf{w}}_t, \quad (23)$$

with the score function  $\mathbf{s}_t(\mathbf{x}_t) := \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) = \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0|\mathbf{x}_t)} \left[ -\frac{\alpha_t \mathbf{x}_0 - \mathbf{x}_t}{\sigma_t^2} \right]$  and the reverse-time standard Wiener process  $\bar{\mathbf{w}}$ . Using the Fokker-Planck equation, we can prove that the time evolution of the PDF  $p(\mathbf{x}_t)$  described by the SDEs is equivalent to that described by an ODE:

$$d\mathbf{x}_t = \left( f_t \mathbf{x}_t - \frac{1}{2} g_t^2 \mathbf{s}_t(\mathbf{x}_t) \right) dt, \quad (24)$$

which maps the noise  $\mathbf{x}_t$  to a deterministic data point  $\mathbf{x}_0$ .

### C.2. Proof of Theorem 3.1

*Proof.* Let  $p(\mathbf{u})$  be the PDF of an arbitrary distribution on  $\mathbb{R}^D$ , and  $\{\Sigma_k\}_{k=1}^K$  be a set of arbitrary  $D \times D$  symmetric

positive definite matrices. We aim to show that if

$$\{a_k^*, \mu_k^*\} = \arg \min_{\{a_k, \mu_k\}} \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ -\log \sum_{k=1}^K A_k \mathcal{N}(\mathbf{u}; \mu_k, \Sigma_k) \right], \quad (25)$$

with  $a_k \in \mathbb{R}$ ,  $\mu_k \in \mathbb{R}^D$ ,  $A_k = \frac{\exp a_k}{\sum_{k=1}^K \exp a_k}$ , then the mean alignment property holds:

$$\sum_{k=1}^K A_k^* \mu_k^* = \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} [\mathbf{u}], \quad (26)$$

with  $A_k^* = \frac{\exp a_k^*}{\sum_{k=1}^K \exp a_k^*}$ .

For brevity, we define

$$q_k(\mathbf{u}) := A_k^* \mathcal{N}(\mathbf{u}; \mu_k^*, \Sigma_k), \quad (27)$$

$$q(\mathbf{u}) := \sum_{k=1}^K q_k(\mathbf{u}). \quad (28)$$

Since  $\{a_k^*, \mu_k^*\}$  minimizes the objective, the first-order optimality conditions imply

$$\begin{aligned} & \frac{\partial \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} [-\log q(\mathbf{u})]}{\partial a_k^*} \\ &= \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ \frac{\partial -\log q(\mathbf{u})}{\partial a_k^*} \right] \\ &= \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ A_k^* - \frac{q_k(\mathbf{u})}{q(\mathbf{u})} \right] \\ &= A_k^* - \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ \frac{q_k(\mathbf{u})}{q(\mathbf{u})} \right] \\ &= 0, \end{aligned} \quad (29)$$

and

$$\begin{aligned} & \frac{\partial \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} [-\log q(\mathbf{u})]}{\partial \mu_k^*} \\ &= \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ \frac{\partial -\log q(\mathbf{u})}{\partial \mu_k^*} \right] \\ &= \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ \frac{q_k(\mathbf{u})}{q(\mathbf{u})} \Sigma_k^{-\frac{1}{2}} (\mu_k^* - \mathbf{u}) \right] \\ &= \Sigma_k^{-\frac{1}{2}} \left( \mu_k^* \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ \frac{q_k(\mathbf{u})}{q(\mathbf{u})} \right] - \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ \frac{q_k(\mathbf{u})}{q(\mathbf{u})} \mathbf{u} \right] \right) \\ &= \mathbf{0}. \end{aligned} \quad (30)$$

From the above, it follows that

$$A_k^* = \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ \frac{q_k(\mathbf{u})}{q(\mathbf{u})} \right], \quad (31)$$

$$\mu_k^* \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ \frac{q_k(\mathbf{u})}{q(\mathbf{u})} \right] = \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ \frac{q_k(\mathbf{u})}{q(\mathbf{u})} \mathbf{u} \right]. \quad (32)$$

Substituting Eq. (31) into Eq. (32), we have

$$A_k^* \mu_k^* = \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ \frac{q_k(\mathbf{u})}{q(\mathbf{u})} \mathbf{u} \right]. \quad (33)$$

Summing over all  $k$ , we conclude that

$$\sum_{k=1}^K A_k^* \mu_k^* = \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} \left[ \frac{\sum_{k=1}^K q_k(\mathbf{u})}{q(\mathbf{u})} \mathbf{u} \right] = \mathbb{E}_{\mathbf{u} \sim p(\mathbf{u})} [\mathbf{u}]. \quad (34)$$

□

### C.3. Derivation of the Reverse Transition Distribution

Given the ground truth data distribution  $p(\mathbf{x}_0)$  and the forward diffusion Gaussian  $p(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \alpha_t \mathbf{x}_0, \sigma_t^2 \mathbf{I})$ , the ground truth denoising distribution  $p(\mathbf{x}_0|\mathbf{x}_t)$  can be derived using Bayes' theorem:

$$p(\mathbf{x}_0|\mathbf{x}_t) = \frac{p(\mathbf{x}_0)p(\mathbf{x}_t|\mathbf{x}_0)}{p(\mathbf{x}_t)}. \quad (35)$$

Conversely, let us assume that we know the ground truth denoising distribution  $p(\mathbf{x}_0|\mathbf{x}_t)$ . By rearranging Eq. (35), we can derive the data distribution as:

$$p(\mathbf{x}_0) = \frac{p(\mathbf{x}_t)p(\mathbf{x}_0|\mathbf{x}_t)}{p(\mathbf{x}_t|\mathbf{x}_0)}. \quad (36)$$

With the data distribution, we can apply the forward diffusion process and derive the noisy data distribution at  $t - \Delta t$ :

$$\begin{aligned} p(\mathbf{x}_{t-\Delta t}) &= \int_{\mathbb{R}^D} p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_0)p(\mathbf{x}_0) d\mathbf{x}_0 \\ &= \int_{\mathbb{R}^D} p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_0) \frac{p(\mathbf{x}_t)p(\mathbf{x}_0|\mathbf{x}_t)}{p(\mathbf{x}_t|\mathbf{x}_0)} d\mathbf{x}_0. \end{aligned} \quad (37)$$

Finally, the reverse transition distribution  $p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t)$  can be derived using Bayes' theorem:

$$\begin{aligned} &p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t) \\ &= \frac{p(\mathbf{x}_{t-\Delta t})p(\mathbf{x}_t|\mathbf{x}_{t-\Delta t})}{p(\mathbf{x}_t)} \\ &= \int_{\mathbb{R}^D} \frac{p(\mathbf{x}_t|\mathbf{x}_{t-\Delta t})p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_0)}{p(\mathbf{x}_t|\mathbf{x}_0)} p(\mathbf{x}_0|\mathbf{x}_t) d\mathbf{x}_0, \end{aligned} \quad (38)$$

where  $p(\mathbf{x}_t|\mathbf{x}_{t-\Delta t})$  is the forward transition Gaussian defined in Eq. (1). The term  $\frac{p(\mathbf{x}_t|\mathbf{x}_{t-\Delta t})p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_0)}{p(\mathbf{x}_t|\mathbf{x}_0)}$  can be fused into one Gaussian PDF using the conflation operation described in § D.1, which yields:

$$p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-\Delta t}; c_1 \mathbf{x}_t + c_2 \mathbf{x}_0, c_3 \mathbf{I}), \quad (39)$$

with the coefficients  $c_1, c_2, c_3$  defined in § 3.3. Therefore, Eq. (38) can be simplified into:

$$p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t) = \int_{\mathbb{R}^D} p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t, \mathbf{x}_0)p(\mathbf{x}_0|\mathbf{x}_t) d\mathbf{x}_0, \quad (40)$$

which is a convolution between  $p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t, \mathbf{x}_0)$  and  $p(\mathbf{x}_0|\mathbf{x}_t)$ . Sampling from  $p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t)$  can therefore be simulated by first sampling  $\mathbf{x}_0 \sim p(\mathbf{x}_0|\mathbf{x}_t)$  and then sampling  $\mathbf{x}_{t-\Delta t} \sim p(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t, \mathbf{x}_0)$ .

In general, given any empirical denoising distribution  $q_\theta(\mathbf{x}_0|\mathbf{x}_t)$ , we can perform stochastic denoising sampling by substituting  $p(\mathbf{x}_0|\mathbf{x}_t) \approx q_\theta(\mathbf{x}_0|\mathbf{x}_t)$  into the above sampling process. Specifically for GMFlow,  $q_\theta(\mathbf{x}_0|\mathbf{x}_t)$  is a GM PDF. § D.3 shows that the convolution of a GM and a Gaussian is also a GM with analytically derived parameters. Therefore, GMFlow models the reverse transition distribution as a GM  $q_\theta(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t)$ , given by Eq. (9).

### C.4. Additional Analysis of the Reverse Transition GM

By computing the first-order Taylor approximation of Eq. (9) w.r.t.  $\Delta t$ , we can re-write the GM reverse transition distribution as:

$$\begin{aligned} &q_\theta(\mathbf{x}_{t-\Delta t}|\mathbf{x}_t) \\ &= \sum_{k=1}^K A_k \mathcal{N}(\mathbf{x}_{t-\Delta t}; \mathbf{x}_t - (f_t \mathbf{x}_t - g_t^2 \mu_{s_k}) \Delta t, g_t^2 \Delta t \mathbf{I}) \\ &\quad + O(\Delta t) \\ &= \mathcal{N}\left(\mathbf{x}_{t-\Delta t}; \mathbf{x}_t - \left(f_t \mathbf{x}_t - g_t^2 \sum_{k=1}^K A_k \mu_{s_k}\right) \Delta t, g_t^2 \Delta t \mathbf{I}\right) \\ &\quad + O(\Delta t), \end{aligned} \quad (41)$$

where  $\mu_{s_k} := -\frac{1}{\sigma_t}(\mathbf{x}_t + \alpha_t \mu_k)$ . The term  $\sum_{k=1}^K A_k \mu_{s_k}$  represents the model's prediction of the score function  $s_t(\mathbf{x}_t)$ , and is a linear transformation of the predicted mean velocity  $\sum_{k=1}^K A_k \mu_k$ . Recursively sampling  $\mathbf{x}_{t-\Delta t}$  using the first-order approximation in Eq. (41) is equivalent to solving the reverse-time SDE in Eq. (23) (with predicted score) using the Euler–Maruyama method, a simple first-order SDE solver.

Eq. (41) is consistent with the standard diffusion model assumption that the reverse transition distribution is approximately Gaussian for small  $\Delta t$ . Even with GM parameterization, SDE sampling is primarily influenced by the mean prediction when  $\Delta t$  is sufficiently small. This underscores the significance of mean alignment not only for ODE solving but also for SDE solving.

### C.5. Derivation of $\hat{q}(\mathbf{x}_0|\mathbf{x}_\tau)$

Let us assume that we know the ground truth denoising distribution at  $\mathbf{x}_t$ , i.e.,  $p(\mathbf{x}_0|\mathbf{x}_t)$ . Eq. (36) shows us how to derive the data distribution  $p(\mathbf{x}_0)$  from  $p(\mathbf{x}_0|\mathbf{x}_t)$ . From  $p(\mathbf{x}_0)$ , the denoising distribution at  $\mathbf{x}_\tau$  can be derived using



Bayes' theorem:

$$\begin{aligned} p(\mathbf{x}_0|\mathbf{x}_\tau) &= \frac{p(\mathbf{x}_0)p(\mathbf{x}_\tau|\mathbf{x}_0)}{p(\mathbf{x}_\tau)} \\ &= \frac{p(\mathbf{x}_t)p(\mathbf{x}_0|\mathbf{x}_t)p(\mathbf{x}_\tau|\mathbf{x}_0)}{p(\mathbf{x}_t|\mathbf{x}_0)p(\mathbf{x}_\tau)} \\ &= \frac{p(\mathbf{x}_\tau|\mathbf{x}_0)}{Z \cdot p(\mathbf{x}_t|\mathbf{x}_0)} p(\mathbf{x}_0|\mathbf{x}_t), \end{aligned} \quad (42)$$

where  $Z = \frac{p(\mathbf{x}_\tau)}{p(\mathbf{x}_t)}$  is a normalization factor. Substituting  $p(\mathbf{x}_0|\mathbf{x}_t) \approx q_\theta(\mathbf{x}_0|\mathbf{x}_t)$  into the above equation yields the denoising distribution conversion rule in Eq. (10). The conversion involves conflation of Gaussians and a GM, which can be approached analytically as shown in § D.

## D. Gaussian Mixture Math References

### D.1. Conflation of Two Gaussians

Let  $p_1(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ ,  $p_2(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$  be two multivariate Gaussian PDFs, their powered conflation is defined as:

$$p'(\mathbf{x}) = \frac{p_1^{\gamma_1}(\mathbf{x})p_2^{\gamma_2}(\mathbf{x})}{Z}, \quad (43)$$

where  $\gamma_1, \gamma_2 \in \mathbb{R}$ , assuming  $\gamma_1 \boldsymbol{\Sigma}_1^{-1} + \gamma_2 \boldsymbol{\Sigma}_2^{-1}$  is positive definite, and  $Z = \int_{\mathbb{R}^D} p_1^{\gamma_1}(\mathbf{x})p_2^{\gamma_2}(\mathbf{x}) d\mathbf{x}$  is a normalization factor. It's easy to prove that,  $p'(\mathbf{x})$  can also be expressed as a Gaussian  $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}', \boldsymbol{\Sigma}')$ , with the new parameters:

$$\boldsymbol{\Sigma}' = (\gamma_1 \boldsymbol{\Sigma}_1^{-1} + \gamma_2 \boldsymbol{\Sigma}_2^{-1})^{-1}, \quad (44)$$

$$\boldsymbol{\mu}' = \boldsymbol{\Sigma}'(\gamma_1 \boldsymbol{\Sigma}_1^{-1} \boldsymbol{\mu}_1 + \gamma_2 \boldsymbol{\Sigma}_2^{-1} \boldsymbol{\mu}_2). \quad (45)$$

### D.2. Conflation of a Gaussian and a GM

Let  $p_1(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$  be a Gaussian PDF, and  $p_2(\mathbf{x}) = \sum_{k=1}^K A_k \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$  be a GM PDF, where  $A_k = \frac{\exp a_k}{\sum_{k=1}^K \exp a_k}$  with logit  $a_k$ . Their conflation is defined as:

$$p'(\mathbf{x}) = \frac{p_1(\mathbf{x})p_2(\mathbf{x})}{Z}, \quad (46)$$

where  $Z = \int_{\mathbb{R}^D} p_1(\mathbf{x})p_2(\mathbf{x}) d\mathbf{x}$  is a normalization factor. Since the GM PDF is a sum of Gaussians, the conflation of a Gaussian and a GM expands to a sum of conflation of Gaussians, which simplifies to a sum of Gaussians. Therefore,  $p'(\mathbf{x})$  can also be expressed as a GM  $\sum_{k=1}^K A'_k \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}'_k, \boldsymbol{\Sigma}'_k)$ , with the new parameters:

$$\boldsymbol{\Sigma}'_k = (\boldsymbol{\Sigma}^{-1} + \boldsymbol{\Sigma}_k^{-1})^{-1}, \quad (47)$$

$$\boldsymbol{\mu}'_k = \boldsymbol{\Sigma}'_k(\boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} + \boldsymbol{\Sigma}_k^{-1} \boldsymbol{\mu}_k), \quad (48)$$

$$A'_k = \frac{\exp a'_k}{\sum_{k=1}^K \exp a'_k}, \quad (49)$$

where the new logit  $a'_k$  is given by:

$$a'_k = a_k - \frac{1}{2}(\boldsymbol{\mu} - \boldsymbol{\mu}_k)^\top (\boldsymbol{\Sigma} + \boldsymbol{\Sigma}_k)^{-1} (\boldsymbol{\mu} - \boldsymbol{\mu}_k). \quad (50)$$

### D.3. Convolution of a Gaussian and a GM

Let  $p(\mathbf{x}_1|\mathbf{x}_2) = \mathcal{N}(\mathbf{x}_1; \boldsymbol{\mu} + c\mathbf{x}_2, \boldsymbol{\Sigma})$  be a conditional Gaussian PDF, where  $c \in \mathbb{R}$  is a linear coefficient, and  $p(\mathbf{x}_2) = \sum_{k=1}^K A_k \mathcal{N}(\mathbf{x}_2; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$  be a GM PDF, where  $A_k = \frac{\exp a_k}{\sum_{k=1}^K \exp a_k}$  with logit  $a_k$ . Their convolution yields the marginal PDF of  $\mathbf{x}_1$ :

$$p(\mathbf{x}_1) = \int_{\mathbb{R}^D} p(\mathbf{x}_1|\mathbf{x}_2)p(\mathbf{x}_2) d\mathbf{x}_2. \quad (51)$$

Since the GM PDF is a sum of Gaussians, the convolution of a Gaussian and a GM expands to a sum of convolution of Gaussians, which simplifies to a sum of Gaussians. Therefore  $p(\mathbf{x}_1)$  can also be expressed as a GM  $\sum_{k=1}^K A_k \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}'_k, \boldsymbol{\Sigma}'_k)$ , with the new parameters:

$$\boldsymbol{\Sigma}'_k = \boldsymbol{\Sigma} + c^2 \boldsymbol{\Sigma}_k, \quad (52)$$

$$\boldsymbol{\mu}'_k = \boldsymbol{\mu} + c\boldsymbol{\mu}_k. \quad (53)$$

## E. Notation

Table 6. A summary of frequently used notations.

Notation	Description
$D$	Data dimension.
$t \in [0, T]$	Diffusion time. Flow matching models define $T := 1$ .
$\alpha_t$	Noise schedule coefficient. Flow matching models define $\alpha_t := 1 - t$ .
$\sigma_t$	Noise schedule coefficient. Flow matching models define $\sigma_t := t$ .
$\epsilon$	Standard Gaussian noise.
$\mathbf{x}, \mathbf{x}_0 \in \mathbb{R}^D$	Data.
$\mathbf{x}_t := \alpha_t \mathbf{x} + \sigma_t \epsilon$	Noisy data.
$p(\mathbf{x}), p(\mathbf{x}_0)$	PDF of data (ground truth).
$p(\mathbf{x}_t)$	PDF of noisy data (ground truth).
$p(\mathbf{x}_t   \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \alpha_t \mathbf{x}_0, \sigma_t^2 \mathbf{I})$	PDF of forward diffusion distribution.
$p(\mathbf{x}_t   \mathbf{x}_{t-\Delta t}) = \mathcal{N}(\mathbf{x}_t; \frac{\alpha_t}{\alpha_{t-\Delta t}} \mathbf{x}_0, \beta_{t, \Delta t} \mathbf{I})$	PDF of forward transition distribution.
$\beta_{t, \Delta t} = \sigma_t^2 - \frac{\alpha_t^2}{\alpha_{t-\Delta t}^2} \sigma_{t-\Delta t}^2$	Variance of forward transition distribution.
$p(\mathbf{x}_0   \mathbf{x}_t) = \frac{p(\mathbf{x}_0)p(\mathbf{x}_t   \mathbf{x}_0)}{p(\mathbf{x}_t)}$	PDF of reverse denoising distribution (ground truth).
$p(\mathbf{x}_{t-\Delta t}   \mathbf{x}_t) = \frac{p(\mathbf{x}_{t-\Delta t})p(\mathbf{x}_t   \mathbf{x}_{t-\Delta t})}{p(\mathbf{x}_t)}$	PDF of reverse transition distribution (ground truth).
$\theta$	Neural network parameters.
$q_\theta(\mathbf{x}_0   \mathbf{x}_t)$	PDF of reverse denoising distribution, predicted by a network.
$q_\theta(\mathbf{x}_{t-\Delta t}   \mathbf{x}_t)$	PDF of reverse transition distribution, predicted by a network.
$\mathbf{u} := \frac{\mathbf{x}_t - \mathbf{x}_0}{\sigma_t}$	Random flow velocity.
$\mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0   \mathbf{x}_t)}[\mathbf{u}]$	Mean flow velocity at $\mathbf{x}_t$ (ground truth).
$p(\mathbf{u}   \mathbf{x}_t)$	PDF of velocity distribution at $\mathbf{x}_t$ (ground truth), derived from $p(\mathbf{x}_0   \mathbf{x}_t)$ .
$q_\theta(\mathbf{u}   \mathbf{x}_t)$	PDF of velocity distribution at $\mathbf{x}_t$ , predicted by a network.
$\mu_\theta(\mathbf{x}_t)$	Mean flow velocity at $\mathbf{x}_t$ , predicted by a network.
$K$	Number of Gaussian components.
$k$	Index of Gaussian components.
$A_k := \frac{\exp a_k}{\sum_{k=1}^K \exp a_k}$	Mixture weight of the $k$ -th Gaussian component.
$\mu_k$	Mean of the $k$ -th Gaussian component.
$\Sigma_k$	Covariance of the $k$ -th Gaussian component. GMFlow defines $\Sigma_k := s^2 \mathbf{I}$ .
$a_k$	Pre-activation logit.
$s$	Shared standard deviation.
$\mu_{xk} := \mathbf{x}_t - \sigma_t \mu_k$	Mean of the $k$ -th Gaussian component after $\mathbf{u}$ -to- $\mathbf{x}_0$ conversion.
$s_x := \sigma_t s$	Shared standard deviation after $\mathbf{u}$ -to- $\mathbf{x}_0$ conversion.
$\mathbf{c}$	Condition.
$w \in [1, +\infty)$	CFG scale.
$\tilde{w} \in [0, 1)$	Probabilistic guidance scale.
$\hat{\mathbf{x}}$	Intermediate sample of denoised data.
$NFE$	Number of function (network) evaluations, a.k.a. sampling steps.
$n$	Number of sub-steps in GM-ODE solvers.
$\tau$	Sub-step diffusion time in GM-ODE solvers.
$\hat{q}(\mathbf{x}_0   \mathbf{x}_\tau)$	PDF of reverse denoising distribution, derived from $q_\theta(\mathbf{x}_0   \mathbf{x}_t)$ (Eq. (10)).
$\hat{q}(\mathbf{x}_0   \mathbf{x}_t)$	PDF of reverse denoising distribution, derived from $q_\theta(\mathbf{x}_0   \mathbf{x}_{t+\Delta t})$ .
$\lambda \in (0, 1]$	Transition ratio.



# Gaussian Mixture Flow Matching Models

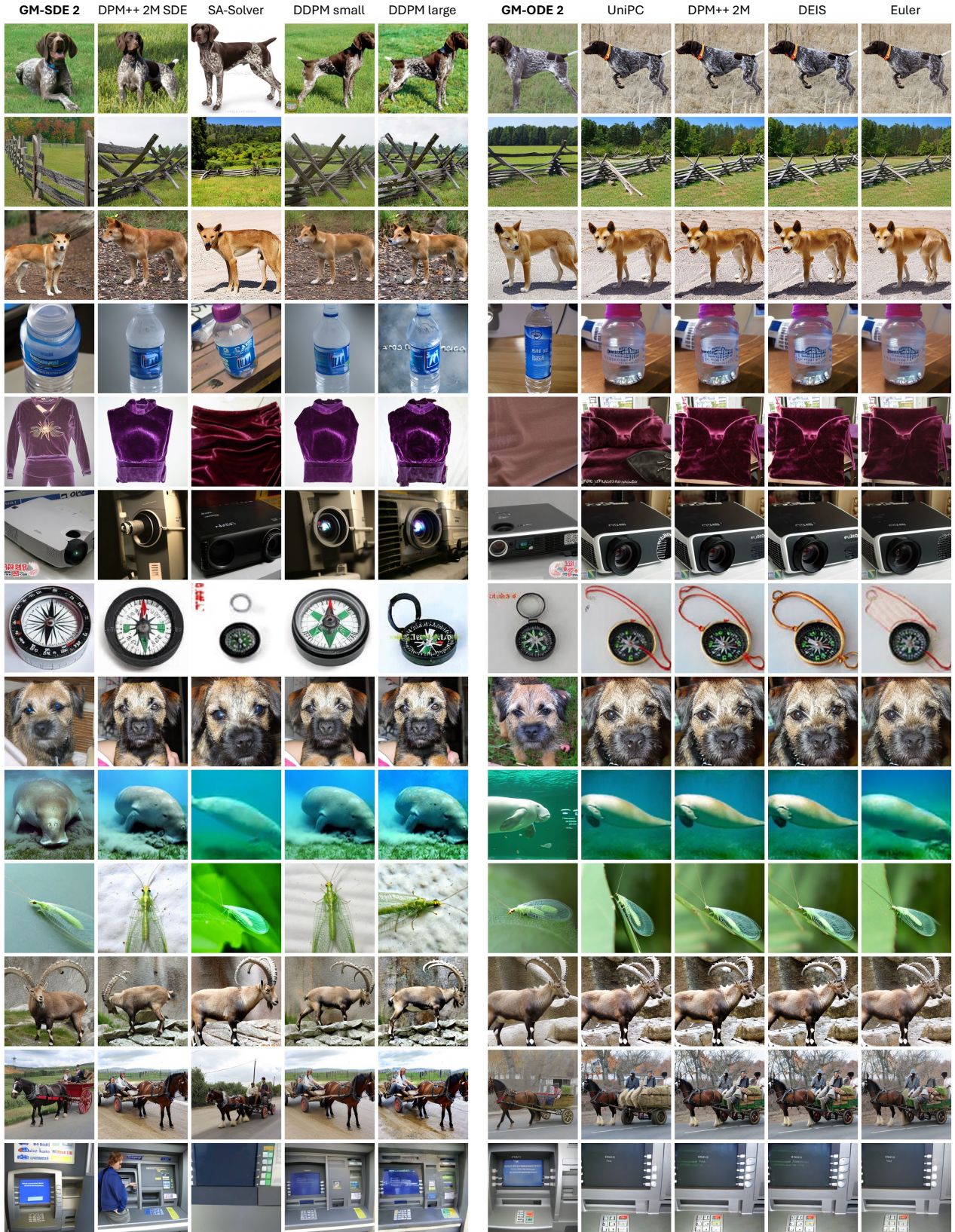


Figure 11. Uncurated samples (at best Precision,  $NFE = 32$ ) from GMFlow and vanilla flow matching baselines using different solvers. The images generated by GMFlow exhibit more natural color saturation and, in some cases, improved structural coherence.