HIGH-ORDER MATCHING FOR ONE-STEP SHORTCUT DIFFUSION MODELS

Bo Chen* **Chengyue Gong†** **Xiaoyu Li‡** **Yingyu Liang§**
Zhizhou Sha¶ **Zhenmei Shi‖** **Zhao Song**** **Mingda Wan††**

ABSTRACT

One-step shortcut diffusion models [Frans, Hafner, Levine and Abbeel, ICLR 2025] have shown potential in vision generation, but their reliance on first-order trajectory supervision is fundamentally limited. The Shortcut model's simplistic velocity-only approach fails to capture intrinsic manifold geometry, leading to erratic trajectories, poor geometric alignment, and instability-especially in high-curvature regions. These shortcomings stem from its inability to model mid-horizon dependencies or complex distributional features, leaving it ill-equipped for robust generative modeling. In this work, we introduce *HOMO* (**H**igh-**O**rder **M**atching for **O**ne-Step Shortcut Diffusion), a game-changing framework that leverages high-order supervision to revolutionize distribution transportation. By incorporating acceleration, jerk, and beyond, HOMO not only fixes the flaws of the Shortcut model but also achieves unprecedented smoothness, stability, and geometric precision. Theoretically, we prove that HOMO's high-order supervision ensures superior approximation accuracy, outperforming first-order methods. Empirically, HOMO dominates in complex settings, particularly in high-curvature regions where the Shortcut model struggles. Our experiments show that HOMO delivers smoother trajectories and better distributional alignment, setting a new standard for one-step generative models.

1 INTRODUCTION

In recent years, deep generative models have exhibited extraordinary promise across various types of data modalities. Techniques such as Generative Adversarial Networks (GANs) Goodfellow et al. (2014), autoregressive models Vaswani (2017), normalizing flows Lipman et al. (2022), and diffusion models Ho et al. (2020) have achieved outstanding results in tasks related to image, audio, and video generation Kalchbrenner et al. (2018); Blattmann et al. (2023). These models have attracted considerable interest owing to their capacity to create invertible and highly expressive mappings, transforming simple prior distributions into complex target data distributions. This fundamental characteristic is the key reason they are capable of modeling any data distribution. Particularly, Lipman et al. (2022); Liu et al. (2022a) have effectively unified conventional normalizing flows with score-based diffusion methods. These techniques produce a continuous trajectory, often referred to as a "flow", which transitions samples from the prior distribution to the target data distribution. By adjusting parameterized velocity fields to align with the time derivatives of the transformation, flow matching achieves significant experimental gains and retains a strong theoretical foundation.

Despite the remarkable progress in flow-based generative models, such as the Shortcut model Frans et al. (2025), these approaches still face challenges in accurately modeling complex data distribu-

* bc7b@mtmail.mtsu.edu. Middle Tennessee State University.
† cygong17@utexas.edu. The University of Texas at Austin.
‡ xiaoyu.li2@student.unsw.edu.au. University of New South Wales.
§ yingyul@hku.hk. The University of Hong Kong. yliang@cs.wisc.edu. University of Wisconsin-Madison.
¶ shazz20@mails.tsinghua.edu.cn. Tsinghua University.
‖ zhmeishi@cs.wisc.edu. University of Wisconsin-Madison.
** magic.linuxkde@gmail.com. The Simons Institute for the Theory of Computing at UC Berkeley.
†† dylan.r.mathison@gmail.com. Anhui University.

tions, particularly in regions of high curvature or intricate geometric structure Wang et al. (2024a); Hu et al. (2024d). This limitation stems from the reliance on first-order techniques, which primarily focus on aligning instantaneous velocities while neglecting the influence of higher-order dynamics on the overall flow geometry. Recent research in diffusion-based modeling Chen (2023); Hang & Gu (2024); Lin et al. (2024) has highlighted the importance of capturing higher-order information to improve the fidelity of learned trajectories. However, a systematic framework for incorporating high-order dynamics into flow matching, especially in Shortcut models, remains an open problem.

In this work, we propose HOMO (High-Order Matching for One-Step Shortcut Diffusion), a revolutionary leap beyond the limitations of the original Shortcut model Frans et al. (2025). While Shortcut models rely on simplistic first-order dynamics, often empirically struggling to capture complex data distributions and producing erratic trajectories in high-curvature regions, HOMO shatters these barriers by introducing high-order supervision. By incorporating acceleration, jerk, and beyond, HOMO not only addresses the empirical shortcomings of the Shortcut model but also achieves unparalleled geometric precision and stability. Where the Shortcut model falters—yielding suboptimal trajectories and poor distributional alignment—HOMO thrives, delivering smoother, more accurate, and fundamentally superior results.

Our primary contribution is a rigorous theoretical and empirical framework that showcases the dominance of HOMO. We prove that HOMO's high-order supervision drastically reduces approximation errors, ensuring precise trajectory alignment from the earliest stages to long-term evolution. Empirically, we demonstrate that the Shortcut model's first-order dynamics fall short in complex settings, while HOMO consistently outperforms it, achieving faster convergence, better sample quality, and unmatched robustness. The contributions of our work is summarized as follows: (i) We introduce high-order supervision into the Shortcut model, resulting in the HOMO framework, which includes novel training and sampling algorithms. (ii) We provide rigorous theoretical guarantees for the approximation error of high-order flow matching, demonstrating its effectiveness in both the early and late stages of the generative process. (iii) We demonstrate that HOMO achieves superior empirical performance in complex settings, especially in intricate distributional landscapes, beyond the capabilities of the original Shortcut model Frans et al. (2025).

2 PRELIMINARY

We begin with establishing the notations and theoretical foundations for the subsequent analysis in this section.

2.1 NOTATIONS

We use $\Pr[\cdot]$ to denote the probability. We use $\mathbb{E}[\cdot]$ to denote the expectation. We use $\mathrm{Var}[\cdot]$ to denote the variance. We use $\|x\|_p$ to denote the ℓ_p norm of a vector $x \in \mathbb{R}^n$, i.e. $\|x\|_1 := \sum_{i=1}^n |x_i|$, $\|x\|_2 := (\sum_{i=1}^n x_i^2)^{1/2}$, and $\|x\|_\infty := \max_{i\in[n]} |x_i|$. We use $f(x) = O(g(x))$ or $f(x) \lesssim g(x)$ to denote that $f(x) \leq C \cdot g(x)$ for some constant $C > 0$. We use $\mathcal{N}(0, I)$ to denote the standard Gaussian distribution.

2.2 SHORTCUT MODEL

Next, we describe the general framework of flow matching and its second-order rectification. These concepts form the basis for our proposed method, as they integrate first and second-order information for trajectory estimation.

Fact 2.1. *Let a field x_t be defined as $x_t = \alpha_t x_0 + \beta_t x_1$, where α_t and β_t are functions of t, and x_0, x_1 are constants. Then, the first-order gradient $\dot{x}_t$ and the second-order gradient $\ddot{x}_t$ can be manually calculated as $\dot{x}_t = \dot{\alpha}_t x_0 + \dot{\beta}_t x_1$ and $\ddot{x}_t = \ddot{\alpha}_t x_0 + \ddot{\beta}_t x_1$.*

In practice, one often samples (x_0, x_1) from (μ_0, π_0) and parameterizes x_t (e.g., interpolation) at intermediate times to build a training objective that matches the velocity field to the true time derivative $\dot{x}_t$.

Definition 2.2 (Shortcut models, implicit definition from page 3 on Frans et al. (2025))**.** *Let $\Delta t = 1/128$. Let x_t be current field. Let $t \in \mathbb{N}$ denote time step. Let $u_1(x_t, t, d)$ be the network to* *be trained. Let $d \in (1/128, 1/64, \ldots, 1/2, 1)$ denote step size. Then, we define Shortcut model compute next field x_{t+d} as follow:*

$$x_{t+d} = \begin{cases} x_t + u_1(x_t, t, d)d & \text{if} d \geq 1/128, \\ x_t + u_1(x_t, t, 0)\Delta t & \text{if} d < 1/128. \end{cases}$$

3 Methodology

Training a flow-based model like the Shortcut model using only first-order terms has limitations compared to incorporating high-order terms. (1) First-order terms provide a less accurate approximation of the true dynamics, capturing only linear components and missing important nonlinearities, which can lead to slower convergence. (2) While reducing complexity and overfitting, first-order terms may limit generalization, especially in highly nonlinear systems. (3) In contrast, higher-order terms improve accuracy and generalization by capturing complex patterns, though they increase computational complexity and overfitting risks.

We introduce HOMO (High-Order Matching for One-step Shortcut Diffusion Model) to address these issues. By leveraging high-order dynamics, HOMO improves the accuracy and stability of field evolution approximations, capturing nonlinearities and enhancing generalization across various scenarios.

Definition 3.1 (HOMO Inference)**.** *Let $\Delta t = 1/128$. Let x_t be the current field. Let $t \in \mathbb{N}$ denote the time step. Let $u_{1,\theta_1}(\cdot)$ and $u_{2,\theta_2}(\cdot)$ denote the HOMO models to be trained. Let $d \in (0, 1/128, 1/64, \ldots, 1/2, 1)$ denote the step size. Then, we define the HOMO computation of the next field x_{t+d} as follows:*

$$x_{t+d} = \begin{cases} x_t + d \cdot u_1(x_t, t, d) + \frac{d^2}{2} \cdot u_2(u_1(x_t, t, d), x_t, t, d) & \textit{if } d \geq 1/128, \\ x_t + \Delta t \cdot u_1(x_t, t, 0) + \frac{(\Delta t)^2}{2} \cdot u_2(u_1(x_t, t, 0), x_t, t, 0) & \textit{if } d < 1/128. \end{cases}$$

The self-consistency target is to ensure that the model's predictions are consistent across different time steps. This is crucial for maintaining the stability and accuracy of the model over long-term predictions.

Definition 3.2 (HOMO Self-Consistency Target)**.** *Let u_{1,θ_1} be the networks to be trained. Let x_t be the current field and x_{t+d} be defined in Definition 3.1. Let $t \in \mathbb{N}$ denote the time step. Let $d \in (0, 1/128, 1/64, \ldots, 1/2, 1)$ denote the step size. Then, we define the Self-Consistency target as follows:*

$$\dot{x}_t^{\text{target}} = u_{1,\theta_1}(x_t, t, d)/2 + u_{1,\theta_1}(x_{t+d}, t, d)/2$$

The second-order HOMO loss is designed to optimize the model by minimizing the discrepancy between the predicted and true velocities and accelerations. This loss function ensures that the model not only captures the immediate dynamics but also the underlying trends and changes in the system.

Definition 3.3 (Second-order HOMO Loss)**.** *Let x_t be the current field. Let $t \in \mathbb{N}$ denote the time step. Let $\dot{x}_t^{\text{target}}$ be defined by Definition 3.2. Let $u_{1,\theta_1}(\cdot)$ and $u_{2,\theta_2}(\cdot)$ denote the HOMO models to be trained. Let $d \in (0, 1/128, 1/64, \ldots, 1/2, 1)$ denote the step size. Let $\dot{x}_t^{\text{true}}$ and $\ddot{x}_t^{\text{true}}$ be the observed (or numerically approximated) true velocity and acceleration. Let $\dot{x}_t^{\text{pred}} := u_{1,\theta_1}(x_t, t, 2d)$ denote the model prediction of the first-order term. Then, we define the HOMO Loss as follows:*

$$L_{(\theta_1,\theta_2)} = \mathbb{E}[\ell_{2,1,\theta_1}(x_t, \dot{x}_t^{\text{true}})] + \mathbb{E}[\ell_{2,2,\theta_2,\theta_1}(x_t, \ddot{x}_t^{\text{true}})] + \mathbb{E}[\|u_{1,\theta_1}(x_t, t, 2d) - \dot{x}_t^{\text{target}}\|^2]$$

We define

$$\ell_{2,1,\theta_1}(x_t, \dot{x}_t^{\text{true}}) := \|u_{1,\theta_1}(x_t, t, 2d) - \dot{x}_t^{\text{true}}\|^2,$$
$$\ell_{2,2,\theta_2,\theta_1}(x_t, \ddot{x}_t^{\text{true}}) := \|u_{2,\theta_2}(\dot{x}_t^{\text{pred}}, x_t, t, 2d) - \ddot{x}_t^{\text{true}}\|^2$$
$$\ell_{\text{selfc}}(x_t, \dot{x}_t^{\text{target}}) := \|u_{1,\theta_1}(x_t, t, 2d) - \dot{x}_t^{\text{target}}\|^2$$

and

$$\ell_{(\theta_1,\theta_2)}(x_t, x_t^{\text{true}}) := \ell_{2,1,\theta_1}(x_t, \dot{x}_t^{\text{true}}) + \ell_{2,2,\theta_2,\theta_1}(x_t, \ddot{x}_t^{\text{true}}) + \ell_{\text{selfc}}(x_t, \dot{x}_t^{\text{target}}).$$

Remark 3.4 (Simple notations)**.** *For simplicity, we denote first-order matching as M1, which implies that HOMO is optimized solely by the first-order loss* $\ell_{2,1,\theta_1}(x_t, \dot{x}_t^{\mathrm{true}})$*. Second-order matching is denoted as M2, where HOMO is optimized only by the second-order loss* $\ell_{2,2,\theta_2,\theta_1}(x_t, \ddot{x}_t^{\mathrm{true}})$*. We refer to HOMO optimized solely by the self-consistency loss as SC, denoted by* $\ell_{\mathrm{selfc}}(x_t, \dot{x}_t^{\mathrm{target}})$*. Combinations of M1, M2, and SC are used to indicate HOMO optimized by corresponding combinations of loss terms. For example, (M1 + M2) denotes HOMO optimized by both first-order and second-order terms, while (M1 + M2 + SC) represents HOMO optimized by the first-order, second-order, and self-consistency terms.*

4 Theoretical Analysis

In this section, we will introduce our main result, the approximation error of the second order flow matching. The theory for higher order flow matching is deferred to Section D.

Algorithm 1 HOMO Training

procedure HOMOTraining(θ, D, p, k)
 ▷ Parameter θ for HOMO model u_1 and u_2.
 ▷ Training dataset D
 ▷ Stepsize and time index distribution p
 ▷ Batch size k
  while not converged **do**
   $x_0 \sim \mathcal{N}(0, I), x_1 \sim D, (d, t) \sim p$
   $\beta_t \leftarrow \sqrt{1 - \alpha_t^2}$
   $x_t \leftarrow \alpha_t \cdot x_0 + \beta_t \cdot x_1$ ▷ Noise data point
   for first k batch elements **do**
    $\dot{s}_t^{\mathrm{true}} \leftarrow \dot{\alpha}_t x_0 + \dot{\beta}_t x_1$ ▷ First-order target
    $\ddot{s}_t^{\mathrm{true}} \leftarrow \ddot{\alpha}_t x_0 + \ddot{\beta}_t x_1$ ▷ Second-order target
    $d \leftarrow 0$
   end for
   for other batch elements **do**
    $s_t \leftarrow u_1(x_t, t, d)$ ▷ First small step of first order
    $\dot{s}_t \leftarrow u_2(u_1(x_t, t, d), x_t, t, d)$ ▷ First small step of second order
    $x_{t+d} \leftarrow x_t + d \cdot s_t + \frac{d^2}{2} \dot{s}_t$ ▷ Follow ODE
    $s_{t+d} \leftarrow u_1(x_{t+d}, t + d, d)$ ▷ Second small step of first order
    $\dot{s}_t^{\mathrm{target}} \leftarrow$ stopgrad $(s_t + s_{t+d})/2$ ▷ Self-consistency target of first order
   end for
   $\theta \leftarrow \nabla_\theta(\|u_1(x_t, t, 2d) - \dot{s}_t^{\mathrm{true}}\|^2 + \|u_2(u_1(x_t, t, 2d), x_t, t, 2d) - \ddot{s}_t^{\mathrm{true}}\|^2 + \|u_1(x_t, t, 2d) - \dot{s}_t^{\mathrm{target}}\|^2)$
  end while
  return θ
end procedure

We first present the approximation error result for the early stage of the diffusion process. This result establishes theoretical guarantees on how well a neural network can approximate the first and second order flows during the initial phases of the trajectory evolution.

Theorem 4.1 (Approximation error of second order flow matching for small t, informal version of Theorem D.1)**.** *Let* N *be a value associated with sample size* n*. Let* $T_0 := N^{-R_0}$ *and* $T_* := N^{-\frac{\kappa^{-1}-\delta}{d}}$ *where* R_0, κ, δ *are some parameters. Let* s *be the order of smoothness of the Besov space that the target distribution belongs to. Under some mild assumptions, there exist neural networks* ϕ_1, ϕ_2 *from a class of neural networks such that, for sufficiently large* N*, we have*

$$\int (\|\phi_1(x,t) - \dot{x}_t^{\mathrm{true}}\|_2^2 + \|\phi_2(x,t) - \ddot{x}_t^{\mathrm{true}}\|_2^2) p_t(x) \mathrm{d}x$$
$$\lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}} + \mathop{\mathbb{E}}_{x \sim P_t}[\|\dot{x}_t^{\mathrm{true}} - \ddot{x}_t^{\mathrm{true}}\|_2^2]$$

Algorithm 2 HOMO Sampling

```
procedure HOMOSAMPLING(θ, M)
                                   ▷ Parameter θ for the HOMO model u_1 and u_2
                                   ▷ The number of sampling steps M
    x ∼ N(0, I)
    d ← 1/M
    t ← 0
    for n ∈ [0, ..., M − 1] do
        x ← x + d · u_1(x, t, d) + (d^2/2) · u_2(u_1(x, t, d), x, t, d)
        t ← t + d
    end for
    return x
end procedure
```

holds for any $t \in [T_0, 3T_*]$*. In addition,* ϕ_1, ϕ_2 *can be taken so we have*

$$\|\phi_1(\cdot, t)\|_\infty = O(|\dot{\alpha}_t|\sqrt{\log n} + |\dot{\beta}_t|),$$
$$\|\phi_2(\cdot, t)\|_\infty = O(|\dot{\alpha}_t|\sqrt{\log n} + |\dot{\beta}_t|).$$

Next, we present the approximation error result for the later stages, confirming that the second-order flow matching remains effective throughout the generative process.

Theorem 4.2 (Approximation error of second order flow matching for large t, informal version of Theorem D.3)**.** *Let* N *be a value associated with sample size* n*. Let* $T_0 := N^{-R_0}$ *and* $T_* := N - \frac{\kappa^{-1}-\delta}{d}$ *where* R_0, κ, δ *are some parameters. Let* s *be the order of smoothness of the Besov space that the target distribution belongs to. Fix* $t_* \in [T_*, 1]$ *and let* $\eta > 0$ *be arbitrary. Under some mild assumptions, there exists neural networks* ϕ_1, ϕ_2 *from a class of neural networks such that*

$$\int (\|\phi_1(x,t) - \dot{x}_t^{\mathrm{true}}\|_2^2 + \|\phi_2(x,t) - \ddot{x}_t^{\mathrm{true}}\|_2^2) p_t(x) \mathrm{d}x$$
$$\lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\eta} + \mathop{\mathbb{E}}_{x \sim P_t}[\|\dot{x}_t^{\mathrm{true}} - \ddot{x}_t^{\mathrm{true}}\|_2^2]$$

holds for any $t \in [2t_*, 1]$*. In addition,* ϕ_1, ϕ_2 *can be taken so we have*

$$\|\phi_1(\cdot, t)\|_\infty = O(|\dot{\alpha}_t| \log N + |\dot{\beta}_t|),$$
$$\|\phi_2(\cdot, t)\|_\infty = O(|\dot{\alpha}_t| \log N + |\dot{\beta}_t|).$$

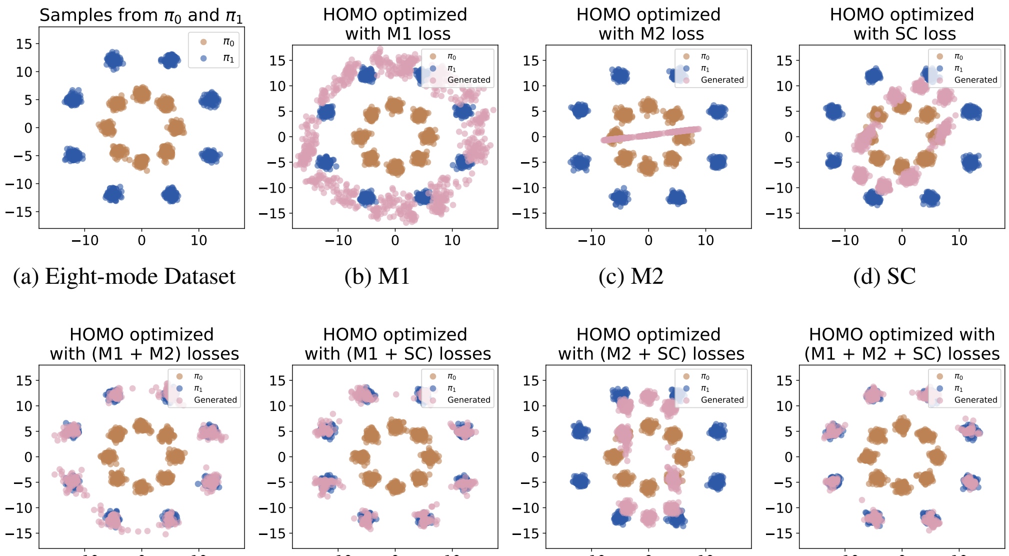
Overall, these two results demonstrate the effectiveness across different phases.

5 Experiments

This section presents a series of experiments to evaluate the effectiveness of our HOMO method and assess the impact of each loss component. Our results demonstrate that HOMO significantly improves distribution generation.

5.1 Experiment setup

We evaluate HOMO on various data distributions and loss combinations. HOMO with first-order and self-consistency losses is equivalent to the original One-step Shortcut model Frans et al. (2025), i.e., M1+SC. The methods M1+M2+SC and M1+M2+M3+SC are our proposed approaches. We implement HOMO with losses defined in Definition 3.3 and we follow Remark 3.4, first-order matching is denoted as M1, second-order as M2, and self-consistency as SC. For target transport, we follow the VP ODE framework Liu et al. (2022b) with $x_t = \alpha_t x_0 + \beta_t x_1$, where $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$, $\beta_t = \sqrt{1-\alpha_t^2}$, and hyperparameters $a = 19.9$, $b = 0.1$.



(a) Eight-mode Dataset (b) M1 (c) M2 (d) SC

(e) (M1 + M2) (f) (M1 + SC) Frans et al. (2025) (g) (M2 + SC) (h) (M1+M2+SC) (Ours)

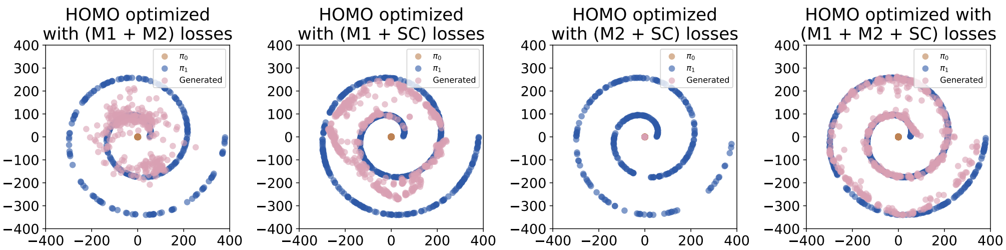
Figure 1: **HOMO on a mixture of Gaussian datasets.** The first row shows results for the initial eight-mode dataset (a) and HOMO optimized with first-order loss (M1), second-order loss (M2), and self-consistency loss (SC) Figures (b-d). The second row presents combinations of losses: M1+M2 (e), M1+SC Frans et al. (2025) (f), M2+SC (g), and M1+M2+SC (Ours) (h). Quantitative results are shown in Table 1.

Table 1: **Euclidean distance loss on Gaussian datasets.** Lower values indicate more accurate distribution matching. Optimal values are in **Bold**, with Underlined numbers representing second-best results. For qualitative results, please refer to Figure 1.

Losses	**Four mode**	**Five mode**	**Eight mode**
M1	2.759	3.281	3.321
M2	11.089	6.554	10.830
SC	6.761	10.893	7.646
M1 + M2	0.941	1.097	<u>0.977</u>
M2 + SC	8.708	9.212	4.801
M1 + SC Frans et al. (2025)	<u>0.820</u>	<u>1.067</u>	1.084
M1 + M2 + SC (Ours)	**0.809**	**0.917**	**0.778**

Table 2: **Euclidean distance loss on complex datasets.** Lower values indicate better distribution matching. Optimal results are in **Bold**, with the second-best marked in Underlined. For qualitative results of complex distribution experiments, please refer to Figure 2 and Figure 13, 14, 15, 16.

Losses	**Circle**	**Irregular**	**Spiral**	**Spin**
M1 + M2	<u>0.642</u>	<u>0.731</u>	7.233	31.009
M1 + SC	0.736	0.743	<u>3.289</u>	<u>12.055</u>
M2 + SC	7.233	0.975	10.096	50.499
M1 + M2 + SC	**0.579**	**0.678**	**1.840**	**10.066**



(a) (M1 + M2) / spin (b) (M1 + SC) / spin (c) (M2 + SC) / spin (d) (M1 + M2 + SC) / spin

Figure 2: **HOMO on complex datasets (Spin).** Results show HOMO optimized with various loss combinations: M1+M2 (a), M1+SC Frans et al. (2025) (b), M2+SC (c), and M1+M2+SC (Ours) (d). Quantitative results are in Table 2.

5.2 Mixture of Gaussian experiments

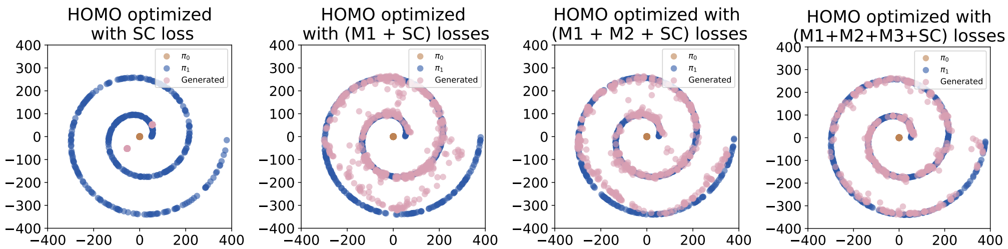
We evaluate HOMO on Gaussian mixture datasets Liang et al. (2024e) with varying modes (four, five, and eight). The eight-mode distribution is the most challenging, where HOMO with all three losses (M1+M2+SC) yields the best performance, achieving the lowest Euclidean distance. HOMO with first-order, second-order, and self-consistency losses is the only model that accurately learns the target distribution, achieving the lowest Euclidean distance among all configurations. The second-order loss is crucial—without it, the model fails to capture finer details (Figure 1 (f)), but with it, the model matches the target distribution more closely (Figure 1 (h)). We analyze the contributions of each loss: (i) The first-order loss captures the general structure but misses finer details (Figures 1 (b) and (g)). (ii) The second-order loss can lead to overfitting, focusing on details at the expense of the broader distribution (Figure 1 (c)). (iii) The self-consistency loss helps concentrate the learned distribution (Figure 1 (d)), whereas without it, the distribution becomes sparse (Figure 1 (e)).

Table 3: **Euclidean distance loss of three complex distribution datasets under original trajectory setting.** Lower values indicate more accurate distribution transfer results. Optimal values are highlighted in **Bold**. And Underlined numbers represent the second best (second lowest) loss value for each dataset (row). For the qualitative results of a mixture of Gaussian experiments, please refer to Figure 3.

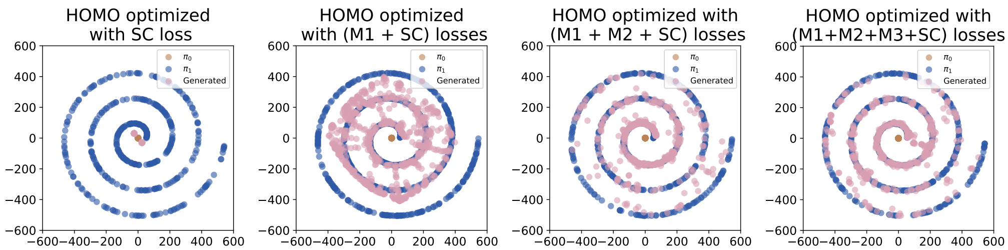
Loss terms	2 Round spin	3 Round spin	Dot-Circle
SC	59.490	50.981	89.974
M1 + SC Frans et al. (2025)	17.866	23.606	37.550
M1 + M2 + SC (Ours)	<u>9.417</u>	<u>13.085</u>	<u>30.679</u>
M1 + M2 + SC + M3 (Ours)	**7.440**	**10.679**	**26.819**

5.3 Complex distribution experiments

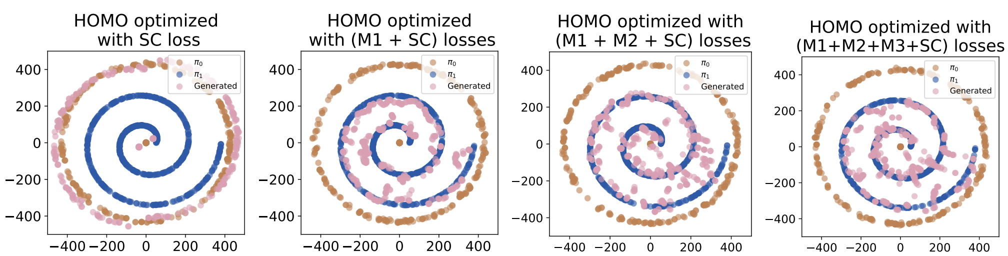
In this section, we test HOMO on datasets with complex distributions. We begin with the spin dataset used in Figure 2, where we sample 600 points from a Gaussian distribution with variance 0.3 for both the source and target distributions. The second-order loss is critical for accurate fitting, particularly for irregular and spiral distributions. As shown in Figure 2 (b) and (d), the second-order loss enables the model to better align with the outer boundaries of the target distribution. The second-order loss is key to HOMO's success in learning complex distributions. Figure 2 (b) shows that the original shortcut model, using only first-order and self-consistency losses, fails to capture the outer circle distribution. However, as shown in Figure 2 (d), adding the second-order loss allows HOMO to accurately model the target distribution, demonstrating its importance in learning more complex structures. We also conducted experiments with HOMO optimized using each loss individually and on other datasets. Further details can be found in Sections E.2, E.3, and E.4.



(a) SC / 2 Round (b) (M1 + SC)) / 2 Round (c) (M1 + M2 + SC)) / 2 Round (d) (M1+M2+M3+SC))/2 Round



(e) SC / 3 Round (f) (M1 + SC)) / 3 Round (g) (M1 + M2 + SC)) / 3 Round (h) (M1+M2+M3+SC))/3 Round



(i) SC / DC (j) (M1 + SC)) / DC (k) (M1 + M2 + SC)) / DC (l) (M1+M2+M3+SC)) / DC

Figure 3: We present the third-order HOMO results in three kinds of complex datasets: 2-round spiral (2 Round), 3-round spiral (3 Round), and dot-circle (DC) datasets. **Left most, Figure (a), (e), (i), (m):** (SC) HOMO optimized with self-consistency loss; **Middle left, Figure (b), (f), (j), (n):** (M1+SC Frans et al. (2025)) HOMO optimized with first-order and self-consistency losses; **Middle right, Figure (c), (g), (k), (o):** (M1+M2+SC (Ours)) HOMO optimized with first-order, second-order and self-consistency losses; **Right most, Figure (d), (h), (l), (p):** (M1+M2+M3+SC (Ours)) HOMO optimized with first-order, second-order, third-order and self-consistency losses. A quantitative evaluation of the complex distribution experiments is presented in Table 3.

5.4 THIRD-ORDER HOMO

In this section, we investigate the impact of adding a third-order loss to HOMO. We use three datasets: 2 Round spin, 3 Round spin, and Dot-Circle. In both the 2 Round spin and 3 Round spin datasets, we sample 600 points from a Gaussian distribution with a variance of 0.3 for both the source and target distributions. In the Dot-Circle dataset, we combine 300 points from the center dot and 300 points from the outermost circle as the source distribution, and sample 600 points from the 2 Round spin distribution as the target. The qualitative results (Figure 3) show that the third-order loss helps HOMO better capture more complex target distributions. Comparisons between Figures 3 (c) and (d), and (g) and (h) highlight how the third-order loss improves the model's fit to intricate distributions. These results are consistent with the quantitative findings in Table 3. The addition of higher-order loss terms demonstrates the value of higher-order supervision in modeling complex distribution transformations.

REFERENCES

Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 22669–22679, 2023.

Song Bian, Zhao Song, and Junze Yin. Federated empirical risk minimization via second-order method. *arXiv preprint arXiv:2305.17482*, 2023.

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22563–22575, 2023.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

Yang Cao, Xiaoyu Li, and Zhao Song. Grams: Gradient descent with adaptive momentum scaling. *arXiv preprint arXiv:2412.17107*, 2024.

Yuefan Cao, Xiaoyu Li, Yingyu Liang, Zhizhou Sha, Zhenmei Shi, Zhao Song, and Jiahao Zhang. Dissecting submission limit in desk-rejections: A mathematical analysis of fairness in ai conference policies. *manuscript*, 2025.

YF Chang and George Corliss. Atomft: solving odes and daes using taylor series. *Computers & Mathematics with Applications*, 28(10-12):209–233, 1994.

Bo Chen, Xiaoyu Li, Yingyu Liang, Jiangxuan Long, Zhenmei Shi, and Zhao Song. Circuit complexity bounds for rope-based transformer architecture. *arXiv preprint arXiv:2411.07602*, 2024a.

Bo Chen, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. Bypassing the exponential dependency: Looped transformers efficiently learn in-context by multi-step gradient descent. *arXiv preprint arXiv:2410.11268*, 2024b.

Bo Chen, Yingyu Liang, Zhizhou Sha, Zhenmei Shi, and Zhao Song. Hsr-enhanced sparse attention acceleration. *arXiv preprint arXiv:2410.10165*, 2024c.

Bo Chen, Xiaoyu Li, Yingyu Liang, Zhao Song, and Zhizhou Sha. Nrflow: Towards noise-robust generative modeling via second-order flow matching. *manuscript*, 2025.

Ricky TQ Chen and Yaron Lipman. Flow matching on general geometries. In *The Twelfth International Conference on Learning Representations*, 2024.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

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

Yifang Chen, Jiayan Huo, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. Fast gradient computation for rope attention in almost linear time. *arXiv preprint arXiv:2412.17316*, 2024d.

Yifang Chen, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. The computational limits of state-space models and mamba via the lens of circuit complexity. *arXiv preprint arXiv:2412.06148*, 2024e.

Zirui Cheng, Jingfei Xu, and Haojian Jin. Treequestion: Assessing conceptual learning outcomes with llm-generated multiple-choice questions. *Proceedings of the ACM on Human-Computer Interaction*, 8(CSCW2):1–29, 2024.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

George Corliss and YF Chang. Solving ordinary differential equations using Taylor series. *ACM Transactions on Mathematical Software (TOMS)*, 8(2):114–144, 1982.

Maximilian Dax, Jonas Wildberger, Simon Buchholz, Stephen R Green, Jakob H Macke, and Bernhard Scholkopf. Flow matching for scalable simulation-based inference. *arXiv preprint arXiv:2305.17161*, 2023.

Yichuan Deng, Zhao Song, Yitan Wang, and Yuanyuan Yang. A nearly optimal size coreset algorithm with nearly linear time. *arXiv preprint arXiv:2210.08361*, 2022.

Yichuan Deng, Sridhar Mahadevan, and Zhao Song. Randomized and deterministic attention sparsification algorithms for over-parameterized feature dimension. *arXiv preprint arXiv:2304.04397*, 2023.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2019.

Franck Djeumou, Cyrus Neary, Eric Goubault, Sylvie Putot, and Ufuk Topcu. Taylor-Lagrange Neural Ordinary Differential Equations: Toward Fast Training and Evaluation of Neural ODEs. *arXiv:2201.05715*, 2022.

Shibo Feng, Chunyan Miao, Zhong Zhang, and Peilin Zhao. Latent diffusion transformer for probabilistic time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 11979–11987, 2024a.

Tao Feng, Chuanyang Jin, Jingyu Liu, Kunlun Zhu, Haoqin Tu, Zirui Cheng, Guanyu Lin, and Jiaxuan You. How far are we from agi. *arXiv preprint arXiv:2405.10313*, 2024b.

Chris Finlay, Jörn-Henrik Jacobsen, Levon Nurbekyan, and Adam Oberman. How to train your neural ODE: the world of Jacobian and kinetic regularization. In *International conference on machine learning*, pp. 3154–3164. PMLR, 2020.

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

Kenji Fukumizu, Taiji Suzuki, Noboru Isobe, Kazusato Oko, and Masanori Koyama. Flow matching achieves minimax optimal convergence. *arXiv preprint arXiv:2405.20879*, 2024.

Peng Gao, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, et al. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*, 2023a.

Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021a.

Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021b.

Yeqi Gao, Sridhar Mahadevan, and Zhao Song. An over-parameterized exponential regression. *arXiv preprint arXiv:2303.16504*, 2023b.

Yeqi Gao, Zhao Song, Weixin Wang, and Junze Yin. A fast optimization view: Reformulating single layer attention in llm based on tensor and svm trick, and solving it in matrix multiplication time. *arXiv preprint arXiv:2309.07418*, 2023c.

Yeqi Gao, Zhao Song, and Junze Yin. Gradientcoin: A peer-to-peer decentralized large language models. *arXiv preprint arXiv:2308.10502*, 2023d.

Yeqi Gao, Zhao Song, and Junze Yin. An iterative algorithm for rescaled hyperbolic functions regression. *arXiv preprint arXiv:2305.00660*, 2023e.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Tiankai Hang and Shuyang Gu. Improved noise schedule for diffusion training. *arXiv preprint arXiv:2407.03297*, 2024.

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

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Jerry Yao-Chieh Hu, Wei-Po Wang, Ammar Gilani, Chenyang Li, Zhao Song, and Han Liu. Fundamental limits of prompt tuning transformers: Universality, capacity and efficiency. *arXiv preprint arXiv:2411.16525*, 2024a.

Jerry Yao-Chieh Hu, Weimin Wu, Yi-Chen Lee, Yu-Chao Huang, Minshuo Chen, and Han Liu. On statistical rates of conditional diffusion transformers: Approximation, estimation and minimax optimality. *arXiv preprint arXiv:2411.17522*, 2024b.

Jerry Yao-Chieh Hu, Weimin Wu, Zhao Song, and Han Liu. On statistical rates and provably efficient criteria of latent diffusion transformers (dits). *arXiv preprint arXiv:2407.01079*, 2024c.

Vincent Hu, Di Wu, Yuki Asano, Pascal Mettes, Basura Fernando, Björn Ommer, and Cees Snoek. Flow matching for conditional text generation in a few sampling steps. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 380–392, 2024d.

Yitong Jiang, Zhaoyang Zhang, Tianfan Xue, and Jinwei Gu. Autodir: Automatic all-in-one image restoration with latent diffusion. In *European Conference on Computer Vision*, pp. 340–359. Springer, 2025.

Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient neural audio synthesis. In *International Conference on Machine Learning*, pp. 2410–2419. PMLR, 2018.

Yekun Ke, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. Advancing the understanding of fixed point iterations in deep neural networks: A detailed analytical study. *arXiv preprint arXiv:2410.11279*, 2024.

Yekun Ke, Xiaoyu Li, Yingyu Liang, Zhizhou Sha, Zhenmei Shi, and Zhao Song. On computational limits and provably efficient criteria of visual autoregressive models: A fine-grained complexity analysis. *arXiv preprint arXiv:2501.04377*, 2025a.

Yekun Ke, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. Circuit complexity bounds for visual autoregressive model. *arXiv preprint arXiv:2501.04299*, 2025b.

Jacob Kelly, Jesse Bettencourt, Matthew J Johnson, and David K Duvenaud. Learning differential equations that are easy to solve. *Advances in Neural Information Processing Systems*, 33:4370–4380, 2020.

Leon Klein, Andreas Kramer, and Frank Noe. Equivariant flow matching. *Advances in Neural Information Processing Systems*, 36, 2024.

Peter E Kloeden and Eckhard Platen. *Numerical Solution of Stochastic Differential Equations*. Springer, 1992.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2021.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, 2021.

Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. A tighter complexity analysis of sparsegpt. *arXiv preprint arXiv:2408.12151*, 2024a.

Xiaoyu Li, Yingyu Liang, Zhenmei Shi, Zhao Song, and Yufa Zhou. Fine-grained attention i/o complexity: Comprehensive analysis for backward passes. *arXiv preprint arXiv:2410.09397*, 2024b.

Xiaoyu Li, Jiangxuan Long, Zhao Song, and Tianyi Zhou. Fast second-order method for neural network under small treewidth setting. In *2024 IEEE International Conference on Big Data (BigData)*. IEEE, 2024c.

Xiaoyu Li, Zhao Song, and Junwei Yu. Quantum speedups for approximating the john ellipsoid. *arXiv preprint arXiv:2408.14018*, 2024d.

Xiaoyu Li, Yingyu Liang, Zhenmei Shi, Zhao Song, Wei Wang, and Jiahao Zhang. On the computational capability of graph neural networks: A circuit complexity bound perspective. *arXiv preprint arXiv:2501.06444*, 2025.

Yingyu Liang, Jiangxuan Long, Zhenmei Shi, Zhao Song, and Yufa Zhou. Beyond linear approximations: A novel pruning approach for attention matrix, 2024a.

Yingyu Liang, Zhizhou Sha, Zhenmei Shi, and Zhao Song. Differential privacy mechanisms in neural tangent kernel regression. *arXiv preprint arXiv:2407.13621*, 2024b.

Yingyu Liang, Zhizhou Sha, Zhenmei Shi, Zhao Song, and Yufa Zhou. Multi-layer transformers gradient can be approximated in almost linear time. *arXiv preprint arXiv:2408.13233*, 2024c.

Yingyu Liang, Zhizhou Sha, Zhenmei Shi, Zhao Song, and Yufa Zhou. Looped relu mlps may be all you need as practical programmable computers. *arXiv preprint arXiv:2410.09375*, 2024d.

Yingyu Liang, Zhenmei Shi, Zhao Song, and Yufa Zhou. Unraveling the smoothness properties of diffusion models: A gaussian mixture perspective. *arXiv preprint arXiv:2405.16418*, 2024e.

Shanchuan Lin, Bingchen Liu, Jiashi Li, and Xiao Yang. Common diffusion noise schedules and sample steps are flawed. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 5404–5411, 2024.

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

Chengyi Liu, Jiahao Zhang, Shijie Wang, Wenqi Fan, and Qing Li. Score-based generative diffusion models for social recommendations. *arXiv preprint arXiv:2412.15579*, 2024.

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

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

Justin Lovelace, Varsha Kishore, Chao Wan, Eliot Shekhtman, and Kilian Q Weinberger. Latent diffusion for language generation. *Advances in Neural Information Processing Systems*, 36, 2024.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

Simian Luo, Chuanhao Yan, Chenxu Hu, and Hang Zhao. Diff-foley: Synchronized video-to-audio synthesis with latent diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 2022.

Maxwell Nye, Anders Johan Andreassen, Gur AriGuy, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.

Kazusato Oko, Shunta Akiyama, and Taiji Suzuki. Diffusion models are minimax optimal distribution estimators. In *International Conference on Machine Learning*, pp. 26517–26582. PMLR, 2023.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

OpenAI. Introducing ChatGPT, 2024.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 2022.

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

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, and Ricky TQ Chen. Multisample flow matching: Straightening flows with minibatch couplings. *arXiv preprint arXiv:2304.14772*, 2023.

Lianke Qin, Zhao Song, and Baocheng Sun. Is solving graph neural tangent kernel equivalent to training graph neural network? *arXiv preprint arXiv:2309.07452*, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Litu Rout, Yujia Chen, Abhishek Kumar, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. Beyond first-order tweedie: Solving inverse problems using latent diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9472–9481, 2024.

Xuan Shen, Zhao Song, Yufa Zhou, Bo Chen, Yanyu Li, Yifan Gong, Kai Zhang, Hao Tan, Jason Kuen, Henghui Ding, et al. Lazydit: Lazy learning for the acceleration of diffusion transformers. *arXiv preprint arXiv:2412.12444*, 2024a.

Xuan Shen, Zhao Song, Yufa Zhou, Bo Chen, Yanyu Li, Yifan Gong, Kai Zhang, Hao Tan, Jason Kuen, Henghui Ding, et al. Lazydit: Lazy learning for the acceleration of diffusion transformers. *arXiv preprint arXiv:2412.12444*, 2024b.

Xuan Shen, Zhao Song, Yufa Zhou, Bo Chen, Jing Liu, Ruiyi Zhang, Ryan A Rossi, Hao Tan, Tong Yu, Xiang Chen, et al. Numerical pruning for efficient autoregressive models. *arXiv preprint arXiv:2412.12441*, 2024c.

Xuan Shen, Zhao Song, Yufa Zhou, Bo Chen, Jing Liu, Ruiyi Zhang, Ryan A Rossi, Hao Tan, Tong Yu, Xiang Chen, et al. Numerical pruning for efficient autoregressive models. *arXiv preprint arXiv:2412.12441*, 2024d.

Zhenmei Shi, Jiefeng Chen, Kunyang Li, Jayaram Raghuram, Xi Wu, Yingyu Liang, and Somesh Jha. The trade-off between universality and label efficiency of representations from contrastive learning. In *The Eleventh International Conference on Learning Representations*, 2023.

Anshumali Shrivastava, Zhao Song, and Zhaozhuo Xu. A theoretical analysis of nearest neighbor search on approximate near neighbor graph. *arXiv preprint arXiv:2303.06210*, 2023.

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

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

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

Zhao Song, Weixin Wang, and Junze Yin. A unified scheme of resnet and softmax. *arXiv preprint arXiv:2309.13482*, 2023.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. Black-box tuning for language-model-as-a-service. In *International Conference on Machine Learning*. PMLR, 2022.

Taiji Suzuki. Adaptivity of deep relu network for learning in besov and mixed smooth besov spaces: optimal rate and curse of dimensionality. In *International Conference on Learning Representations*, 2019.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Hans Triebel. *Theory of Function Spaces II*, volume 84 of *Monographs in Mathematics*. Birkhäuser, 1992.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 2017.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*. PMLR, 2023.

Xiaofei Wang, Sefik Emre Eskimez, Manthan Thakker, Hemin Yang, Zirun Zhu, Min Tang, Yufei Xia, Jinzhu Li, Sheng Zhao, Jinyu Li, et al. An investigation of noise robustness for flow-matching-based zero-shot tts. *arXiv preprint arXiv:2406.05699*, 2024a.

Yilin Wang, Zeyuan Chen, Liangjun Zhong, Zheng Ding, Zhizhou Sha, and Zhuowen Tu. Dolfin: Diffusion layout transformers without autoencoder. *arXiv preprint arXiv:2310.16305*, 2023a.

Yilin Wang, Haiyang Xu, Xiang Zhang, Zeyuan Chen, Zhizhou Sha, Zirui Wang, and Zhuowen Tu. Omnicontrolnet: Dual-stage integration for conditional image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7436–7448, 2024b.

Yuntao Wang, Zirui Cheng, Xin Yi, Yan Kong, Xueyang Wang, Xuhai Xu, Yukang Yan, Chun Yu, Shwetak Patel, and Yuanchun Shi. Modeling the trade-off of privacy preservation and activity recognition on low-resolution images. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pp. 1–15, 2023b.

Zirui Wang, Zhizhou Sha, Zheng Ding, Yilin Wang, and Zhuowen Tu. Tokencompose: Grounding diffusion with token-level supervision. *arXiv preprint arXiv:2312.03626*, 2023c.

Jerry Wei, Le Hou, Andrew Kyle Lampinen, Xiangning Chen, Da Huang, Yi Tay, Xinyun Chen, Yifeng Lu, Denny Zhou, Tengyu Ma, and Quoc V Le. Symbol tuning improves in-context learning in language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8121–8130, 2022.

Zhuoyan Xu, Zhenmei Shi, Junyi Wei, Yin Li, and Yingyu Liang. Improving foundation models for few-shot learning via multitask finetuning. In *ICLR 2023 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2023.

Zhuoyan Xu, Zhenmei Shi, Junyi Wei, Fangzhou Mu, Yin Li, and Yingyu Liang. Towards few-shot adaptation of foundation models via multitask finetuning. In *The Twelfth International Conference on Learning Representations*, 2024.

Renrui Zhang, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*. PMLR, 2021.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, LILI YU, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: Less is more for alignment. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Appendix

Roadmap. In Section A, we introduce the Shortcut Model Training and Sampling Algorithm. Section B discusses related works that inspire our approach. Section C states the tools from Fukumizu et al. (2024) used in our analysis. Section D explores the theory behind Higher-Order Flow Matching. Section E investigates the impact of different optimization terms through empirical ablation studies. Section F examines model performance on complex distribution experiments. Section G extends HOMO to third-order dynamics and evaluates its effectiveness on complex tasks. Section H quantifies the computational and optimization costs associated with different configurations. In Section I, we conclude our paper.

A Original Algorithm

Here we introduce Shortcut Model Training and Sampling Algorithm from Page 5 of Frans et al. (2025)

Algorithm 3 Shortcut Model Training from page 5 of Frans et al. (2025)

while not converged **do**
  $x_0 \sim \mathcal{N}(0, I), x_1 \sim D, (d, t) \sim p(d, t)$
  $x_t \leftarrow (1-t)x_0 + tx_1$ ▷ Noise data point
  for first k batch elements **do**
    $s_{\text{target}} \leftarrow x_1 - x_0$ ▷ Flow-matching target
    $d \leftarrow 0$
  end for
  for other batch elements **do**
    $s_t \leftarrow u_1(x_t, t, d)$ ▷ Fitst small step
    $x_{t+d} \leftarrow x_t + s_t d$ ▷ Follow ODE
    $s_{t+d} \leftarrow u_1(x_{t+d}, t+d, d)$ ▷ Second small step
    $s_{\text{target}} \leftarrow$ stopgrad $(s_t + s_{t+d})/2$ ▷ Self-consistency target
  end for
  $\theta \leftarrow \nabla_\theta \|u_1(x_t, t, 2d) - s_{\text{target}}\|^2$
end while

Algorithm 4 Shortcut model. Sampling from page 5 of Frans et al. (2025)

$x \sim \mathcal{N}(0, I)$
$d \leftarrow 1/M$
$t \leftarrow 0$
for $n \in [0, \dots, M-1]$ **do**
  $x \leftarrow x + d \cdot u_1(x, t, d)$
  $t \leftarrow t + d$
end for
return x

B Related work

In this section, we discuss more related work which inspire our work.

Diffusion Models. Diffusion models have garnered significant attention for their capability to generate high-fidelity images by incrementally refining noisy samples, as exemplified by DiT Peebles & Xie (2023) and U-ViT Bao et al. (2023). These approaches typically involve a forward process that systematically adds noise to an initial clean image and a corresponding reverse process that learns to remove noise step by step, thereby recovering the underlying data distribution in a probabilistic manner. Early works Song & Ermon (2019); Song et al. (2020) established the theoretical foundations of this denoising strategy, introducing score-matching and continuous-time diffusion frameworks that significantly improved sample quality and diversity. Subsequent research has focused on more efficient training and sampling procedures Lu et al. (2022); Shen et al. (2024b;d), aiming to reduce computational overhead and converge faster without sacrificing image fidelity. Other lines of work leverage latent spaces to learn compressed representations, thereby streamlining both training and inference Rombach et al. (2022); Hu et al. (2024c). This latent learning approach integrates naturally with modern neural architectures and can be extended to various modalities beyond images, showcasing the versatility of diffusion processes in modeling complex data distributions. In parallel, recent researchers have also explored multi-scale noise scheduling and adaptive step-size strategies to enhance convergence stability and maintain high-resolution detail in generated content in Lovelace et al. (2024); Feng et al. (2024a); Rout et al. (2024); Jiang et al. (2025); Luo et al. (2024). There are more other works also inspire our work Xu et al. (2022); Dax et al. (2023); Pooladian et al. (2023); Wang et al. (2023c;a); Shen et al. (2024a;c); Wang et al. (2024b); Chen & Lipman (2024); Klein et al. (2024); Chen et al. (2025); Cao et al. (2025); Cheng et al. (2024); Wang et al. (2023b); Feng et al. (2024b); Liu et al. (2024); Hu et al. (2024b).

Flow Matching. Generative models like diffusion (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2020) and flow-matching (Lipman et al., 2022; Liu et al., 2022a) operate by learning ordinary differential equations (ODEs) that map noise to data. To simplify, this study leverages the optimal transport flow-matching formulation (Liu et al., 2022a). A linear combination of a noise sample $x_0 \sim \mathcal{N}(0, \mathbb{I})$ and a data point $x_1 \sim \mathcal{D}$ defines x_t:

$$x_t = (1-t)x_0 + tx_1, \qquad v_t = x_1 - x_0,$$

with v_t representing the velocity vector directed from x_0 to x_1. While v_t is uniquely derived from (x_0, x_1), knowledge of only x_t renders it a random variable due to the ambiguity in selecting (x_0, x_1). Neural networks in flow models approximate the expected velocity $\bar{v}_t = \mathbb{E}[v_t \mid x_t]$, calculated as an average over all valid pairings. Training involves minimizing the deviation between predicted and empirical velocities:

$$\begin{aligned} &\bar{v}_\theta(x_t, t) \sim \mathbb{E}_{x_0, x_1 \sim \mathcal{D}}\left[v_t \mid x_t\right] \\ &\mathcal{L}^{\mathrm{F}}(\theta) = \mathbb{E}_{x_0, x_1 \sim \mathcal{D}}\left[\|\bar{v}_\theta(x_t, t) - (x_1 - x_0)\|^2\right]. \end{aligned} \tag{1}$$

Sampling involves first drawing a noise point $x_0 \sim \mathcal{N}(0, I)$ and iteratively transforming it into a data point x_1. The denoising ODEs, parameterized by $\bar{v}_\theta(x_t, t)$, governs this transformation, and Euler's method approximates it over small, discrete time steps.

High-order ODE Gradient in Diffusion Models. Higher-order gradient-based methods like TTMs Kloeden & Platen (1992) have applications far exceeding DDMs. For instance, solvers Djeumou et al. (2022) and regularization frameworks Kelly et al. (2020); Finlay et al. (2020) for neural ODEs Chen et al. (2018); Grathwohl et al. (2018) frequently utilize higher-order derivatives. Beyond machine learning contexts, the study of higher-order TTMs has been extensively directed toward solving stiff Chang & Corliss (1994) and non-stiff Chang & Corliss (1994); Corliss & Chang (1982) systems.

Large Language Models. Neural networks built upon the Transformer architecture Vaswani et al. (2017) have swiftly risen to dominate modern machine learning approaches in natural language processing. Extensive Transformer models, trained on wide-ranging and voluminous datasets while encompassing billions of parameters, are often termed large language models (LLM) or foundation models Bommasani et al. (2021). Representative instances include BERT Devlin et al. (2019), PaLM Chowdhery et al. (2022), Llama Touvron et al. (2023), ChatGPT OpenAI (2024), GPT4 OpenAI (2023), among others. These LLMs have showcased striking general intelligence abilities Bubeck et al. (2023) in various downstream tasks. Numerous adaptation methods have been developed to tailor LLMs for specific applications, such as adapters Hu et al. (2022); Zhang et al. (2023); Gao et al. (2023a); Shi et al. (2023), calibration schemes Zhao et al. (2021); Zhou et al. (2023), multitask fine-tuning Gao et al. (2021a); Xu et al. (2023); Von Oswald et al. (2023); Xu et al. (2024), prompt optimization Gao et al. (2021b); Lester et al. (2021), scratchpad approaches Nye et al. (2021), instruction tuning Li & Liang (2021); Chung et al. (2022); Mishra et al. (2022), symbol tuning Wei et al. (2023), black-box tuning Sun et al. (2022), and reinforcement learning from human feedback (RLHF) Ouyang et al. (2022). Additional lines of research endeavor to boost model efficiency without sacrificing performance across diverse domains, for example in Deng et al.

(2022); Song et al. (2023); Gao et al. (2023c;e;d); Bian et al. (2023); Deng et al. (2023); Gao et al. (2023b); Shrivastava et al. (2023); Qin et al. (2023); Chen et al. (2024c); Li et al. (2024d); Chen et al. (2024b); Liang et al. (2024c); Chen et al. (2024a); Liang et al. (2024b;d;a); Li et al. (2024a;c); Cao et al. (2024); Li et al. (2024b); Chen et al. (2024e;d); Ke et al. (2024; 2025a;b); Li et al. (2025); Hu et al. (2024a).

C TOOLS FROM PREVIOUS WORKS

We state the tools in Fukumizu et al. (2024) that we will use to prove our main results.

C.1 DEFINITIONS OF BESOV SPACE

Definition C.1 (Modulus of Smoothness)**.** *Let Ω be a domain in $\mathbb{R}^d$. For a function $f \in L^{p'}(\Omega)$ with $p' \in (0, \infty]$, the r-th modulus of smoothness of f is defined by*

$$w_{r,p'}(f,t) = \sup_{\|h\|_2 \le t} \|\Delta_h^r(f)\|_{p'},$$

where the finite difference operator $\Delta_h^r(f)(x)$ is given by

$$\Delta_h^r(f)(x) = \begin{cases} \sum_{j=0}^r \binom{r}{j}(-1)^{r-j} f(x+jh), & \mathrm{if} x+jh \in \Omega \mathrm{forall} j, \\ 0, & \mathrm{otherwise}. \end{cases}$$

Definition C.2 (Besov Seminorm)**.** *Let $0 < p', q' \le \infty$, $s > 0$, and set $r := |s| + 1$. The Besov seminorm of $f \in L^{p'}(\Omega)$ is defined as*

$$|f|_{B^s_{p',q'}} := \begin{cases} \left(\int_0^\infty (t^{-s} w_{r,p'}(f,t))^{q'} \frac{dt}{t}\right)^{\frac{1}{q'}}, & q' < \infty, \\ \sup_{t>0} t^{-s} w_{r,p'}(f,t), & q' = \infty. \end{cases}$$

Definition C.3 (Besov Space)**.** *The Besov space $B^s_{p',q'}(\Omega)$ is the function space equipped with the norm*

$$\|f\|_{B^s_{p',q'}} := \|f\|_{p'} + |f|_{B^s_{p',q'}},$$

It consists of all functions $f \in L^{p'}(\Omega)$ such that

$$B^s_{p',q'}(\Omega) := \{f \in L^{p'}(\Omega) \mid \|f\|_{B^s_{p',q'}} < \infty\}.$$

Remark C.4. *The parameter s governs the degree of smoothness of functions in $B^s_{p',q'}(\Omega)$. In particular, when $p' = q'$ and s is an integer, the Besov space $B^s_{p',q'}(\Omega)$ coincides with the standard Sobolev space of order s. For further details on the properties and applications of Besov spaces, see Triebel (1992).*

C.2 B-SPLINE

Definition C.5 (Indicator Function)**.** *Let $\mathcal{N}(x)$ be the characteristic function defined by*

$$\mathcal{N}(x) = \begin{cases} 1, & x \in [0,1], \\ 0, & \mathrm{otherwise}. \end{cases}$$

Definition C.6 (Cardinal B-Spline)**.** *For $\ell \in \mathcal{N}$, the cardinal B-spline of order ℓ is defined by*

$$\mathcal{N}_\ell(x) := \underbrace{\mathcal{N} * \mathcal{N} * \cdots * \mathcal{N}}_{\ell+1 \mathrm{times}}(x),$$

where $*$ denotes the convolution operation. Explicitly, the convolution of two functions $f, g : \mathbb{R} \to \mathbb{R}$ is given by

$$(f * g)(x) = \int_{\mathbb{R}} f(x-y) g(y) \mathrm{d}y.$$

Thus, $\mathcal{N}_\ell(x)$ is obtained by convolving $\mathcal{N}$ with itself $(\ell + 1)$ times.

Definition C.7 (Tensor Product B-Spline Basis). *For a multi-index $k \in \mathcal{N}^d$ and $j \in \mathbb{Z}^d$, the tensor product B-spline basis in $\mathbb{R}^d$ of order ℓ is defined as*

$$M^d_{k,j}(x) := \prod_{i=1}^{d} \mathcal{N}_\ell(2^{k_i} x_i - j_i).$$

This basis is constructed as the product of univariate B-splines, scaled and translated according to the parameters k and j.

Definition C.8 (B-Spline Approximation in Besov Spaces in Suzuki (2019); Oko et al. (2023)). *A function f in the Besov space can be approximated using a superposition of tensor product B-splines as*

$$f_N(x) = \sum_{(k,j)} \alpha_{k,j} M^d_{k,j}(x),$$

where the summation is taken over appropriate index sets (k, j), and the coefficients $\alpha_{k,j}$ are real numbers that determine the contribution of each basis function.

C.3 Class of Neural Networks

Definition C.9 (Neural Network Class in Fukumizu et al. (2024)). *Let $L \in \mathbb{N}$ denote the depth (number of layers), $W = (W_1, W_2, \ldots, W_{L+1}) \in \mathbb{N}^{L+1}$ the width configuration of the network, $S \in \mathbb{N}$ a sparsity constraint, and $B > 0$ a norm bound. The class of neural networks $\mathcal{M}(L, W, S, B)$ is defined as*

$$\mathcal{M}(L, W, S, B) := \{ \; \psi_{A^{(L)},b^{(L)}} \circ \cdots \circ \psi_{A^{(2)},b^{(2)}}(A^{(1)}x + b^{(1)})m | A^{(i)} \in \mathbb{R}^{W_{i+1}\times W_i}, b^{(i)} \in \mathbb{R}^{W_{i+1}},$$
$$\sum_{i=1}^{L} (\|A^{(i)}\|_0 + \|b^{(i)}\|_0) \le S, \quad \max_{1\le i\le L} \{\|A^{(i)}\|_\infty \vee \|b^{(i)}\|_\infty\} \le B \}.$$

Here, the function $\psi_{A,b} : \mathbb{R}^{W_i} \to \mathbb{R}^{W_{i+1}}$ represents the affine transformation with ReLU activation, given by

$$\psi_{A,b}(z) = A \cdot \mathsf{ReLU}(z) + b, \quad \text{where} \quad \mathsf{ReLU}(z) = \max\{0, z\}.$$

The sparsity constraint ensures that the total number of nonzero entries in all weight matrices and bias vectors does not exceed S, while the norm constraint limits their maximum absolute values to B.

C.4 Assumptions

Remark C.10. *We introduce a small positive constant $\delta > 0$ and denote by N the number of basis functions in the B-spline used to approximate $p_t(x)$. The value of N is determined by the sample size n, specifically following the relation $N = n^{\frac{d}{2s+d}}$, which balances the approximation error and the complexity of both the B-spline and the neural network.*

Definition C.11 (Stopping Time). *As we introduce in Remark C.10, we define the stopping time as $T_0 = N^{-R_0}$, where R_0 is a parameter to be specified later, and consider solving the ODE backward in time from $t = 1$ down to $t = T_0$.*

Definition C.12 (Reduced Cube). *Let $I^d = [-1, 1]^d$ denote the d-dimensional cube. To mitigate boundary effects when N is large, we define the reduced cube as*

$$I^d_N := [-1 + N^{-(1-\kappa\delta)}, 1 - N^{-(1-\kappa\delta)}]^d,$$

where the parameter $\kappa > 0$ will be specified later in Assumption C.15.

Assumption C.13 (Smoothness and support of p_0). *The target probability P_0 has support contained in I^d, and its probability density function p_0 satisfies*

$$p_0 \in B^s_{p',q'}(I^d) \quad \text{and} \quad p_0 \in B^{\widetilde{s}}_{p',q'}(I^d \setminus I^d_N) \quad \text{with} \quad \widetilde{s} \ge \max\{6s - 1, 1\}.$$

Assumption C.14 (Boundedness away from 0 and above). *There exists a constant $C_0 > 0$ such that*

$$C_0^{-1} \le p_0(x) \le C_0 \quad \text{forall} \quad x \in I^d.$$

Assumption C.15 (Form of (α_t, β_t) and their bounds)**.** *There are constants $\kappa \geq \frac{1}{2}$, $b_0 > 0$, $\widetilde{\kappa} > 0$, and $\widetilde{b}_0 > 0$ such that, for sufficiently small $t \geq T_0$,*

$$\alpha_t = b_0, t^{\kappa}, \quad \text{and} \quad 1 - \beta_t = \widetilde{b}_0, t^{\widetilde{\kappa}}.$$

Moreover, there exist $D_0 > 0$ and $K_0 > 0$ such that $\forall t \in [T_0, 1]$, we have

$$D_0^{-1} \leq \alpha_t^2 + \beta_t^2 \leq D_0, \quad |\dot{\alpha}_t| + |\dot{\beta}_t| \leq N^{K_0}.$$

Assumption C.16 (Additional bound in the critical case $\kappa = \frac{1}{2}$)**.** *If $\kappa = \frac{1}{2}$, then there exist $b_1 > 0$ and $D_1 > 0$ such that, for all $0 \leq \gamma < R_0$,*

$$\int_{T_0}^{N^{-\gamma}} \{(\dot{\alpha}_t)^2 + (\dot{\beta}_t)^2\} \mathrm{d}t \leq D_1 (\log N)^{b_1}.$$

Assumption C.17 (Lipschitz bound on the first moment)**.** *There is a constant $C_L > 0$ such that, for all $t \in [T_0, 1]$,*

$$\| \frac{\partial}{\partial x} \int y p_t(y|x) \mathrm{d}y \|_{\mathrm{op}} \leq C_L.$$

C.5 Approximation error for small t

Lemma C.18 (Theorem 7 in Fukumizu et al. (2024))**.** *Under Assumptions C.13 C.14 C.15 C.16 and C.17, and if the following holds*

- $L = O(\log^4 N)$.
- $\|W\|_\infty = O(N \log^6 N)$
- $S = O(N \log^8 N)$
- $B = \exp(O(\log N \log \log N))$.

Then there exists a neural network $\phi \in \mathcal{M}(L, W, S, B)$ such that, for sufficiently large N, we have

$$\int \|\phi(x,t) - \dot{x}_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}},$$

holds for any $t \in [T_0, 3T_*]$. In addition, ϕ can be taken so we have

$$\|\phi(\cdot, t)\|_\infty = O(|\dot{\alpha}_t| \sqrt{\log n} + |\dot{\beta}_t|).$$

C.6 Approximation error for large t

Lemma C.19 (Theorem 7 in Fukumizu et al. (2024))**.** *Fix $t_* \in [T_*, 1]$ and let $\eta > 0$ be arbitrary, under Assumptions C.13 C.14 C.15 C.16 and C.17, and if the following holds*

- $L = O(\log^4 N)$.
- $\|W\|_\infty = O(N)$
- $S = O(t_*^{-d\kappa} N^{\delta\kappa})$
- $B = \exp(O(\log N \log \log N))$.

Then there exist a neural network $\phi \in \mathcal{M}(L, W, S, B)$ such that

$$\int \|\phi(x,t) - \dot{x}_t^{\mathrm{true}}\|^2 p_t(x) \mathrm{d}x \lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\eta}.$$

holds for any $t \in [2t_*, 1]$. In addition, ϕ can be taken so we have

$$\|\phi(\cdot, t)\|_\infty = O(|\dot{\alpha}_t| \log N + |\dot{\beta}_t|).$$

D THEORY OF HIGHER ORDER FLOW MATCHING

We use $\frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}}$ to denote the k-th order derivative of x_t^{true} with respect to t. Note that $\dot{x}_t^{\mathrm{true}} := \frac{\mathrm{d}}{\mathrm{d}t} x_t^{\mathrm{true}}$, and $\ddot{x}_t^{\mathrm{true}} := \frac{\mathrm{d}^2}{\mathrm{d}t^2} x_t^{\mathrm{true}}$.

D.1 APPROXIMATION ERROR OF SECOND ORDER FLOW MATCHING FOR SMALL t

Theorem D.1 (Approximation error of second order flow matching for small t, formal version of Theorem 4.1)**.** *Under Assumptions C.13 C.14 C.15 C.16 and C.17, and if the following holds*

- $L = O(\log^4 N)$.
- $\|W\|_\infty = O(N \log^6 N)$
- $S = O(N \log^8 N)$
- $B = \exp(O(\log N \log\log N))$.

Then there exists neural networks $\phi_1, \phi_2 \in \mathcal{M}(L, W, S, B)$ such that, for sufficiently large N, we have

$$\int (\|\phi_1(x,t) - \dot{x}_t^{\mathrm{true}}\|_2^2 + \|\phi_2(x,t) - \ddot{x}_t^{\mathrm{true}}\|_2^2) p_t(x) \mathrm{d}x$$
$$\lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}} + \mathop{\mathbb{E}}_{x \sim P_t} [\|\dot{x}_t^{\mathrm{true}} - \ddot{x}_t^{\mathrm{true}}\|_2^2]$$

holds for any $t \in [T_0, 3T_*]$. In addition, ϕ_1, ϕ_2 can be taken so we have

$$\|\phi_1(\cdot,t)\|_\infty = O(|\dot{\alpha}_t|\sqrt{\log n} + |\dot{\beta}_t|) \quad \text{and} \quad \|\phi_2(\cdot,t)\|_\infty = O(|\dot{\alpha}_t|\sqrt{\log n} + |\dot{\beta}_t|).$$

Proof. Suppose that $t \in [T_0, 3T_*]$. By Lemma C.18, there is $\phi_1 \in \mathcal{M}(L, W, S, B)$ such that

$$\int (\|\phi_1(x,t) - \dot{x}_t^{\mathrm{true}}\|_2^2 p_t(x) \lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}}. \tag{2}$$

Next, we can show that there exists some $\phi_2 \in \mathcal{M}(L, W, S, B)$ such that

$$\begin{aligned}
\int \|\phi_2(x,t) - \ddot{x}_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x = & \int \|\phi_2(x,t) - \dot{x}_t^{\mathrm{true}} + \dot{x}_t^{\mathrm{true}} - \ddot{x}_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
\leq & \int (\|\phi_2(x,t) - \dot{x}_t^{\mathrm{true}}\|_2 + \|\dot{x}_t^{\mathrm{true}} - \ddot{x}_t^{\mathrm{true}}\|_2)^2 p_t(x) \mathrm{d}x \\
\leq & \int 2(\|\phi_2(x,t) - \dot{x}_t^{\mathrm{true}}\|_2^2 + \|\dot{x}_t^{\mathrm{true}} - \ddot{x}_t^{\mathrm{true}}\|_2^2) p_t(x) \mathrm{d}x \\
= & 2 \int \|\phi_2(x,t) - \dot{x}_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x + 2 \int \|\dot{x}_t^{\mathrm{true}} - \ddot{x}_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
= & 2 \int \|\phi_2(x,t) - \dot{x}_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x + 2 \mathop{\mathbb{E}}_{x \sim P_t} [\|\dot{x}_t^{\mathrm{true}} - \ddot{x}_t^{\mathrm{true}}\|_2^2 \\
\lesssim & (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}} + \mathop{\mathbb{E}}_{x \sim P_t} [\|\dot{x}_t^{\mathrm{true}} - \ddot{x}_t^{\mathrm{true}}\|_2^2]
\end{aligned} \tag{3}$$

where the first step follows from the basic algebra, the second step follows from the triangle inequality, the third step follows from $(a+b)^2 \leq 2a^2 + 2b^2$, the fourth step follows from basic algebra, the fifth step follows from the definition of expectation, and the last step follows from Lemma C.18.

Finally, by Eq. (2) and Eq. (3), for any $t \in [T_0, 3T_*]$, we have

$$\int (\|\phi_1(x,t) - \dot{x}_t^{\mathrm{true}}\|_2^2 + \|\phi_2(x,t) - \ddot{x}_t^{\mathrm{true}}\|_2^2) p_t(x) \mathrm{d}x$$
$$\lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}} + \mathop{\mathbb{E}}_{x \sim P_t} [\|\dot{x}_t^{\mathrm{true}} - \ddot{x}_t^{\mathrm{true}}\|_2^2].$$

Moreover, by Lemma C.18, ϕ_1, ϕ_2 can be taken so we have

$$\|\phi_1(\cdot,t)\|_\infty = O(|\dot{\alpha}_t|\sqrt{\log n} + |\dot{\beta}_t|) \quad \text{and} \quad \|\phi_2(\cdot,t)\|_\infty = O(|\dot{\alpha}_t|\sqrt{\log n} + |\dot{\beta}_t|).$$

Thus, the proof is complete. $\square$

D.2 Approximation Error of Higher Order Flow Matching for Small t

Theorem D.2 (Approximation error of higher order flow matching for small t)**.** *Under Assumptions C.13 C.14 C.15 C.16 and C.17, and if the following holds*

- $L = O(\log^4 N)$.
- $\|W\|_\infty = O(N \log^6 N)$
- $S = O(N \log^8 N)$
- $B = \exp(O(\log N \log\log N))$
- $K = O(1)$

Then there exists neural networks $\phi_1, \phi_2, \ldots, \phi_K \in \mathcal{M}(L, W, S, B)$ such that, for sufficiently large N, we have

$$\begin{aligned}
&\int (\sum_{k=1}^{K} \|\phi_k(x,t) - \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}}\|_2^2) p_t(x) \mathrm{d}x \\
&\lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}} + \sum_{k=1}^{K-1} \mathop{\mathbb{E}}_{x\sim P_t}[\|\frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2]
\end{aligned}$$

holds for any $t \in [T_0, 3T_*]$. In addition, for any $k \in [K]$, ϕ_k can be taken so we have

$$\|\phi_k(\cdot,t)\|_\infty = O(|\dot{\alpha}_t|\sqrt{\log n} + |\dot{\beta}_t|).$$

Proof. We first show that for any $k \geq 2$, for any $t \in [T_0, 3T_*]$, there exists $\phi \in \mathcal{M}(L, W, S, B)$ such that

$$\begin{aligned}
&\int \|\phi(x,t) - \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
&\lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}} + \sum_{j=1}^{k} \mathop{\mathbb{E}}_{x\sim P_t}[\|\frac{\mathrm{d}^j}{\mathrm{d}t^j} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|_2^2]. \qquad (4)
\end{aligned}$$

We prove this by mathematical induction.

Base case. The statements hold when $k = 2$ because of Lemma D.1.

Induction step. We assume that the statement hold for $k \geq 2$. We would like to show that it holds for $k+1$. We can show that, for any $t \in [T_0, 3T_*]$, there exists $\phi \in \mathcal{M}(L, S, W, B)$ such that

$$\begin{aligned}
&\int \|\phi(x,t) - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
= &\int \|\phi(x,t) - \frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}} + \frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
\leq &\int (\|\phi(x,t) - \frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}}\|_2 + \|\frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2)^2 p_t(x) \mathrm{d}x \\
\leq &\int 2(\|\phi(x,t) - \frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}}\|_2^2 + \|\frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2) p_t(x) \mathrm{d}x \\
= &\, 2\int \|\phi(x,t) - \frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x + 2\int \|\frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
= &\, 2\int \|\phi(x,t) - \frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x + 2 \mathop{\mathbb{E}}_{x \sim P_t}[\|\frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2] \\
\lesssim &\, (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}} + \sum_{j=1}^{k} \mathop{\mathbb{E}}_{x \sim P_t}[\|\frac{\mathrm{d}^{j}}{\mathrm{d}t^{j}} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|_2^2] + \mathop{\mathbb{E}}_{x \sim P_t}[\|\frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2] \\
= &\, (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}} + \sum_{j=1}^{k+1} \mathop{\mathbb{E}}_{x \sim P_t}[\|\frac{\mathrm{d}^{j}}{\mathrm{d}t^{j}} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|_2^2], \qquad (5)
\end{aligned}$$

where the first step follows from basic algebra, the second step follows from triangle inequality, the third step follows from the Cauchy-Schwarz inequality, the fourth step follows from basic algebra, the fifth step follows from the definition of expectation, the six step follows from Eq. (4).

Hence, there exists $\phi_1, \phi_2, \ldots, \phi_K \in \mathcal{M}(L, W, S, B)$ such that for $k \in [K]$, for any $t \in [T_0, 3T_*]$, we have

$$\begin{aligned}
&\int \|\phi_k(x,t) - \frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
\lesssim &\, (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}} + \sum_{j=1}^{k} \mathop{\mathbb{E}}_{x \sim P_t}[\|\frac{\mathrm{d}^{j}}{\mathrm{d}t^{j}} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|_2^2]. \qquad (6)
\end{aligned}$$

Taking the summation over $k \in [K]$, we have for any $t \in [T_0, 3T_*]$,

$$\begin{aligned}
&\int \sum_{k=1}^{K} \|\phi_k(x,t) - \frac{\mathrm{d}^{k}}{\mathrm{d}t^{k}} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
\lesssim &\, K \cdot (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\frac{2s}{d}} + \sum_{k=1}^{K} (k \cdot \mathop{\mathbb{E}}_{x \sim P_t}[\|\frac{\mathrm{d}^{j}}{\mathrm{d}t^{j}} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|_2^2]) \\
\lesssim &\, ((\dot{\alpha}_t)^2 \log N + (\dot{\beta}_t)^2) N^{-\frac{2s}{d}} + \sum_{k=1}^{K} \mathop{\mathbb{E}}_{x \sim P_t}[\|\frac{\mathrm{d}^{j}}{\mathrm{d}t^{j}} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|_2^2]
\end{aligned}$$

where the first step follows from Eq. (6), and the second step uses $K = O(1)$.

Moreover, by Lemma C.19, $\phi_1, \phi_2, \ldots, \phi_K$ can be taken so we have for $k \in [K]$,

$$\|\phi_k(\cdot, t)\|_\infty = O(|\dot{\alpha}_t| \log \sqrt{n} + |\dot{\beta}_t|).$$

Thus, the proof is complete. □

D.3 Approximation Error of Second Order Flow Matching for Large t

Theorem D.3 (Approximation error of second order flow matching for large t, formal version of Theorem 4.2)**.** *Fix $t_* \in [T_*, 1]$ and let $\eta > 0$ be arbitrary, under Assumptions C.13 C.14 C.15 C.16 and C.17, and if the following holds*

- $L = O(\log^4 N)$.
- $\|W\|_\infty = O(N)$
- $S = O(t_*^{-d\kappa} N^{\delta\kappa})$
- $B = \exp(O(\log N \log\log N))$.

Then there exist neural networks $\phi_1, \phi_2 \in \mathcal{M}(L, W, S, B)$ such that

$$\int (\|\phi_1(x,t) - \dot{x}_t^{\rm true}\|_2^2 + \|\phi_2(x,t) - \ddot{x}_t^{\rm true}\|_2^2) p_t(x) \mathrm{d}x$$
$$\lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\eta} + \mathop{\mathbb{E}}_{x\sim P_t}[\|\dot{x}_t^{\rm true} - \ddot{x}_t^{\rm true}\|_2^2]$$

holds for any $t \in [2t_*, 1]$. In addition, ϕ_1, ϕ_2 can be taken so we have

$$\|\phi_1(\cdot, t)\|_\infty = O(|\dot{\alpha}_t| \log N + |\dot{\beta}_t|) \quad \text{and} \quad \|\phi_2(\cdot, t)\|_\infty = O(|\dot{\alpha}_t| \log N + |\dot{\beta}_t|).$$

Proof. Suppose that $t \in [2t_*, 1]$. By Lemma C.19, there is $\phi_1 \in \mathcal{M}(L, W, S, B)$ such that

$$\int (\|\phi_1(x,t) - \dot{x}_t^{\rm true}\|_2^2 p_t(x) \mathrm{d}x \lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\eta}. \tag{7}$$

Next, we can show that there exists some $\phi_2 \in \mathcal{M}(L, W, S, B)$ such that

$$\begin{aligned}
\int \|\phi_2(x,t) - \ddot{x}_t^{\rm true}\|_2^2 p_t(x) \mathrm{d}x = & \int \|\phi_2(x,t) - \dot{x}_t^{\rm true} + \dot{x}_t^{\rm true} - \ddot{x}_t^{\rm true}\|_2^2 p_t(x) \mathrm{d}x \\
\leq & \int (\|\phi_2(x,t) - \dot{x}_t^{\rm true}\|_2 + \|\dot{x}_t^{\rm true} - \ddot{x}_t^{\rm true}\|_2)^2 p_t(x) \mathrm{d}x \\
\leq & \int 2(\|\phi_2(x,t) - \dot{x}_t^{\rm true}\|_2^2 + \|\dot{x}_t^{\rm true} - \ddot{x}_t^{\rm true}\|_2^2) p_t(x) \mathrm{d}x \\
= & 2\int \|\phi_2(x,t) - \dot{x}_t^{\rm true}\|_2^2 p_t(x) \mathrm{d}x + 2\int \|_2 \dot{x}_t^{\rm true} - \ddot{x}_t^{\rm true}\|_2^2 p_t(x) \mathrm{d}x \\
= & 2\int \|\phi_2(x,t) - \dot{x}_t^{\rm true}\|_2^2 p_t(x) \mathrm{d}x + 2 \mathop{\mathbb{E}}_{x\sim P_t}[\|\dot{x}_t^{\rm true} - \ddot{x}_t^{\rm true}\|_2^2 \\
\lesssim & (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\eta} + \mathop{\mathbb{E}}_{x\sim P_t}[\|\dot{x}_t^{\rm true} - \ddot{x}_t^{\rm true}\|_2^2]
\end{aligned} \tag{8}$$

where the first step follows from the basic algebra, the second step follows from the triangle inequality, the third step follows from $(a+b)^2 \leq 2a^2 + 2b^2$, the fourth step follows from basic algebra, the fifth step follows from the definition of expectation, and the last step follows from Lemma C.19.

Finally, by Eq. (7) and Eq. (8), we have

$$\int (\|\phi_1(x,t) - \dot{x}_t^{\rm true}\|^2 + \|\phi_2(x,t) - \ddot{x}_t^{\rm true}\|^2) p_t(x) \mathrm{d}x$$
$$\lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\eta} + \mathop{\mathbb{E}}_{x\sim P_t}[\|\dot{x}_t^{\rm true} - \ddot{x}_t^{\rm true}\|^2].$$

Moreover, by Lemma C.19, ϕ_1, ϕ_2 can be taken so we have

$$\|\phi_1(\cdot, t)\|_\infty = O(|\dot{\alpha}_t| \log N + |\dot{\beta}_t|) \quad \text{and} \quad \|\phi_2(\cdot, t)\|_\infty = O(|\dot{\alpha}_t| \log N + |\dot{\beta}_t|).$$

Thus, the proof is complete. ☐

D.4 Approximation Error of Higher Order Flow Matching for Large t

Theorem D.4 (Approximation error of higher order flow matching for large t)**.** *Fix $t_* \in [T_*, 1]$ and let $\eta > 0$ be arbitrary, under Assumptions C.13 C.14 C.15 C.16 and C.17, and if the following holds*

- $L = O(\log^4 N)$.

- $\|W\|_\infty = O(N)$
- $S = O(t_*^{-d\kappa} N^{\delta\kappa})$
- $B = \exp(O(\log N \log\log N))$
- $K = O(1)$

Then there exist neural networks $\phi_1, \phi_2, \ldots, \phi_K \in \mathcal{M}(L, W, S, B)$ such that,

$$\begin{aligned}
&\int (\sum_{k=1}^{K} \|\phi_k(x,t) - \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}}\|^2) p_t(x) \mathrm{d}x \\
&\lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\eta} + \sum_{k=1}^{K-1} \mathbb{E}_{x \sim P_t}[\| \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|^2]
\end{aligned}$$

holds for any $t \in [2t_*, 1]$. In addition, for any $k \in [K]$, ϕ_k can be taken so we have

$$\|\phi_k(\cdot, t)\|_\infty = O(|\dot{\alpha}_t| \log N + |\dot{\beta}_t|).$$

Proof. We first show that for any $k \geq 2$, for any $t \in [2t_*, 1]$, there exists $\phi \in \mathcal{M}(L, W, S, B)$ such that

$$\begin{aligned}
&\int \|\phi(x,t) - \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
&\lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\eta} + \sum_{j=1}^{k} \mathbb{E}_{x \sim P_t}[\| \frac{\mathrm{d}^j}{\mathrm{d}t^j} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|^2].
\end{aligned} \tag{9}$$

We prove this by mathematical induction.

Base case. The statements hold when $k = 2$ because of Lemma D.3.

Induction step. We assume that the statement hold for $k \geq 2$. We would like to show that it holds for $k+1$. We can show that, for any $t \in [2t_*, 1]$, there exists $\phi \in \mathcal{M}(L, S, W, B)$ such that

$$\begin{aligned}
&\int \|\phi(x,t) - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
&= \int \|\phi(x,t) - \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}} + \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
&\leq \int (\|\phi(x,t) - \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}}\|_2 + \|\frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2)^2 p_t(x) \mathrm{d}x \\
&\leq \int 2(\|\phi(x,t) - \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}}\|_2^2 + \|\frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2) p_t(x) \mathrm{d}x \\
&= 2\int \|\phi(x,t) - \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x + 2\int \|\frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x \\
&= 2\int \|\phi(x,t) - \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}}\|_2^2 p_t(x) \mathrm{d}x + 2 \mathbb{E}_{x \sim P_t}[\|\frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2] \\
&\lesssim (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\eta} + \sum_{j=1}^{k} \mathbb{E}_{x \sim P_t}[\|\frac{\mathrm{d}^j}{\mathrm{d}t^j} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|_2^2] + \mathbb{E}_{x \sim P_t}[\|\frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{k+1}}{\mathrm{d}t^{k+1}} x_t^{\mathrm{true}}\|_2^2] \\
&= (\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2) N^{-\eta} + \sum_{j=1}^{k+1} \mathbb{E}_{x \sim P_t}[\|\frac{\mathrm{d}^j}{\mathrm{d}t^j} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|_2^2],
\end{aligned} \tag{10}$$

where the first step follows from basic algebra, the second step follows from triangle inequality, the third step follows from the Cauchy-Schwarz inequality, the fourth step follows from basic algebra, the fifth step follows from the definition of expectation, the six step follows from Eq. (9).

Hence, there exists $\phi_1, \phi_2, \ldots, \phi_K \in \mathcal{M}(L, W, S, B)$ such that for $k \in [K]$, for any $t \in [2t_*, 1]$, we have

$$\begin{aligned}
&\int \|\phi_k(x,t) - \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}}\|_2^2 p_t(x)\mathrm{d}x \\
\lesssim\ &(\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2)N^{-\eta} + \sum_{j=1}^{k} \mathop{\mathbb{E}}_{x\sim P_t}[\|\frac{\mathrm{d}^j}{\mathrm{d}t^j} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|_2^2]. \qquad (11)
\end{aligned}$$

Taking the summation over $k \in [K]$, we have for any $t \in [2t_*, 1]$,

$$\begin{aligned}
&\int \sum_{k=1}^{K} \|\phi_k(x,t) - \frac{\mathrm{d}^k}{\mathrm{d}t^k} x_t^{\mathrm{true}}\|_2^2 p_t(x)\mathrm{d}x \\
\lesssim\ &((\dot{\alpha}_t)^2 \log N + (\dot{\beta}_t)^2)N^{-\eta} + \sum_{k=1}^{K} (k \cdot \mathop{\mathbb{E}}_{x\sim P_t}[\|\frac{\mathrm{d}^j}{\mathrm{d}t^j} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|_2^2]) \\
\lesssim\ &(\dot{\alpha}_t^2 \log N + \dot{\beta}_t^2)N^{-\eta} + \sum_{k=1}^{K} \mathop{\mathbb{E}}_{x\sim P_t}[\|\frac{\mathrm{d}^j}{\mathrm{d}t^j} x_t^{\mathrm{true}} - \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}} x_t^{\mathrm{true}}\|_2^2]
\end{aligned}$$

where the first step follows from Eq. (11), and the second step uses $K = O(1)$. Moreover, by Lemma C.19, $\phi_1, \phi_2, \ldots, \phi_K$ can be taken so we have for $k \in [K]$,

$$\|\phi_k(\cdot, t)\|_\infty = O(|\dot{\alpha}_t| \log N + |\dot{\beta}_t|).$$

Thus, the proof is complete. □

E Empirical Ablation Study

In Section E.1, we introduce the three Gaussian mixture distribution datasets—four-mode, five-mode, and eight-mode—used in our empirical ablation study, along with their configurations for source and target modes. The subsequent subsections analyze the impact of different optimization terms. Section E.2 evaluates the performance of HOMO optimized solely with the first-order term. Section E.3 examines the effect of using only the second-order term. Section E.4 assesses results when optimization is guided by the self-consistency term. Section E.5 explores the combined effect of first- and second-order terms, while Section E.6 investigates the combination of second-order and self-consistency terms. Through these analyses, we aim to dissect the contributions of individual and combined loss terms in achieving effective transport trajectories.

E.1 Dataset

Here we introduce three datasets we use: four-mode, five-mode, and eight-mode Gaussian mixture distribution datasets; each Gaussian component has a variance of 0.3. In the four-mode Gaussian mixture distribution, four source mode(**brown**) positioned at a distance $D_0 = 5$ from the origin, and four target mode(**indigo**) positioned at a distance $D_0 = 14$ from the origin, each mode sample 200 points. In five-mode Gaussian mixture distribution, five source mode(**brown**) positioned at a distance $D_0 = 6$ from the origin, and five target mode(**indigo**) positioned at a distance $D_0 = 13$ from the origin, each mode sample 200 points. And in eight-mode Gaussian mixture distribution, eight source mode(**brown**) positioned at a distance $D_0 = 6$ from the origin, and eight target mode(**indigo**) positioned at a distance $D_0 = 13$ from the origin, each mode sample 100 points.

E.2 Only First Order Term

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the target transport trajectory setting, we follow the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In the four-mode dataset, five-mode dataset, and eight-mode dataset, we all sample 100

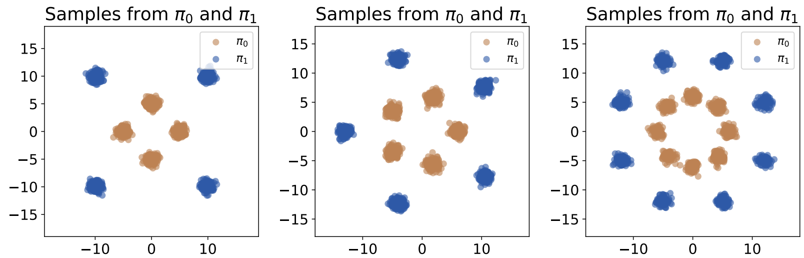


Figure 4: The four-mode Gaussian mixture distribution (**Left**), five-mode Gaussian mixture distribution (**Middle**), and eight-mode Gaussian mixture distribution (**Right**). Our goal is to make HOMO learn a transport trajectory from distribution π_0 (**brown**) to distribution π_1 (**indigo**).

points in each source mode and target mode. And in four-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 1000 training steps. In five-mode dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 1000 training steps. And in eight-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 1000 training steps.

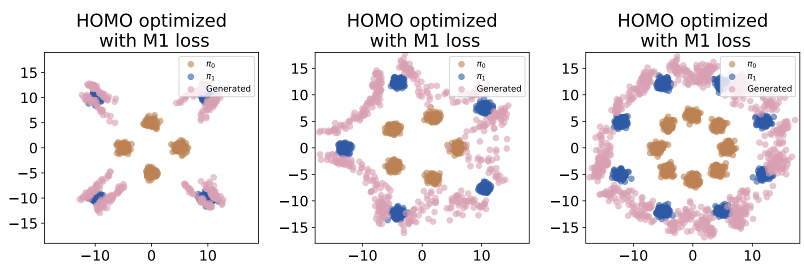


Figure 5: (A) The distributions generated by HOMO are only optimized by first-order term in four-mode dataset (**Left**), five-mode dataset (**Middle**), and eight-mode dataset (**Right**). The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

E.3 Only Second Order Term

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the target transport trajectory setting, we follow the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In the four-mode dataset, five-mode dataset, and eight-mode dataset, we all sample 100 points in each source mode and target mode. And in four-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 100 training steps. In five-mode dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 100 training steps. And in eight-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 100 training steps.

E.4 Only Self-Consistency Term

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the target transport trajectory setting, we follow the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose

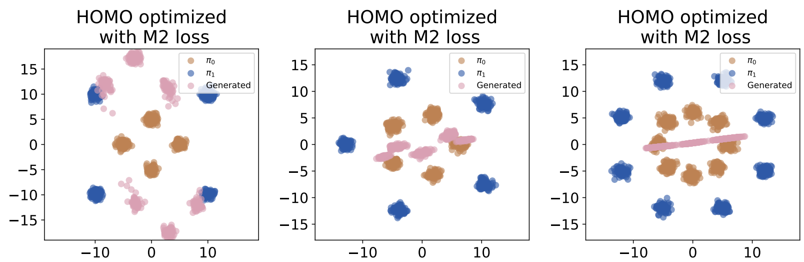


Figure 6: (B) The distributions generated by HOMO are only optimized by second-order term in the four-mode dataset (**Left**), five-mode dataset (**Middle**), and eight-mode dataset (**Right**). The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

$\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In the four-mode dataset, five-mode dataset, and eight-mode dataset, we all sample 100 points in each source mode and target mode. And in four-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 50 training steps. In five-mode dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 50 training steps. And in eight-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 50 training steps.

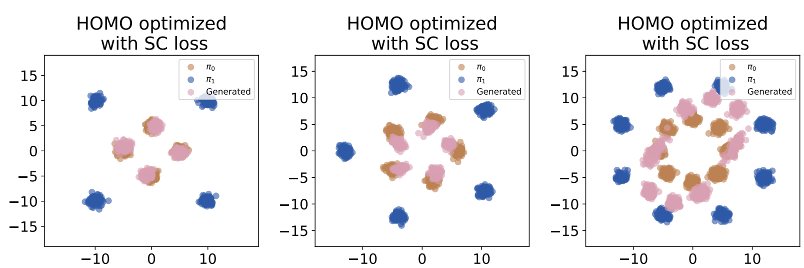


Figure 7: (C) The distributions generated by HOMO are only optimized by self-consistency term in the four-mode dataset (**Left**), five-mode dataset (**Middle**), and eight-mode dataset (**Right**). The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

E.5 First Order Plus Second Order

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the target transport trajectory setting, we follow the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In the four-mode dataset, five-mode dataset, and eight-mode dataset, we all sample 100 points in each source mode and target mode. And in four-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 1000 training steps. In five-mode dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 2000 training steps. And in eight-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 2000 training steps.

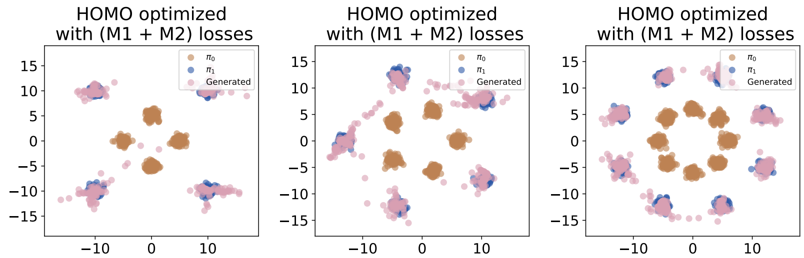


Figure 8: (A + B) The distributions generated by HOMO, optimized by first-order term and second-order term in four-mode dataset (**Left**), five-mode dataset (**Middle**), and eight-mode dataset (**Right**). The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

E.6 Second Order Plus Self-Target

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the target transport trajectory setting, we follow the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In the four-mode dataset, five-mode dataset, and eight-mode dataset, we all sample 100 points in each source mode and target mode. And in four-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 100 training steps. In five-mode dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 100 training steps. And in eight-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 100 training steps.

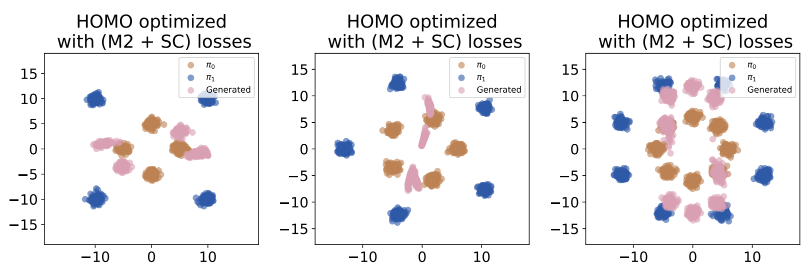


Figure 9: (B + C) The distributions generated by HOMO, optimized by second-order term and self-consistency term in four-mode dataset (**Left**), five-mode dataset (**Middle**), and eight-mode dataset (**Right**). The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

E.7 First Order Plus Self-Target

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the target transport trajectory setting, we follow the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In the four-mode dataset, five-mode dataset, and eight-mode dataset, we all sample 100 points in each source mode and target mode. And in four-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 1000 training steps. In five-mode dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 1000 training steps. And in eight-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 1000 training steps.

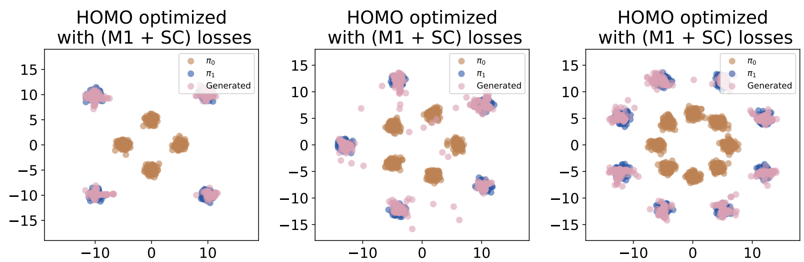


Figure 10: (A + C) The distributions generated by HOMO, optimized by first-order term and self-consistency term in four-mode dataset (**Left**), five-mode dataset (**Middle**), and eight-mode dataset (**Right**). The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

E.8 HOMO

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the target transport trajectory setting, we follow the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In the four-mode dataset, five-mode dataset, and eight-mode dataset, we all sample 100 points in each source mode and target mode. And in four-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 1000 training steps. In five-mode dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 1000 training steps. And in eight-mode dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 1000 training steps.

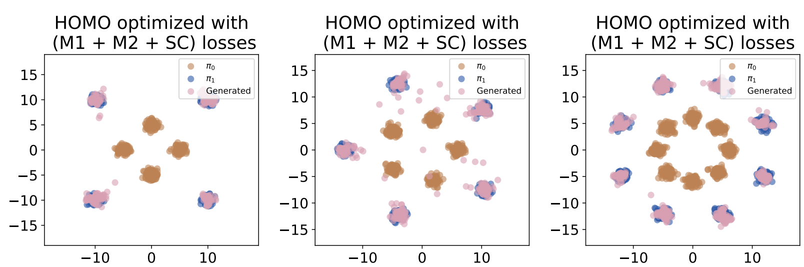


Figure 11: (A + B + C) The distributions generated by HOMO in four-mode dataset (**Left**), five-mode dataset (**Middle**), and eight-mode dataset (**Right**). The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

F COMPLEX DISTRIBUTION EXPERIMENT

In Section F.1, we introduce the datasets used in our experiments. The analysis of results with first-order and second-order terms in Section F.2, and we evaluate the performance with first-order and self-consistency terms in Section F.3, assess the impact of second-order and self-consistency terms in Section F.4. Finally, we present the overall results of HOMO with all loss terms combined in Section F.5.

F.1 DATASETS

Here, we introduce four datasets we proposed: circle dataset, irregular ring dataset, spiral line dataset, and spin dataset. In the circle dataset, we sample 600 points from Gaussian distribution with 0.3 variance for both source distribution and target distribution. In the irregular ring dataset, we sample 600 points from Gaussian distribution with 0.3 variance for both source distribution and target distribution. In the spiral line dataset, we sample 600 points from Gaussian distribution with 0.3 variance for both source distribution and target distribution. In the spin dataset, we sample 600 points from the Gaussian distribution with 0.3 variance for both source distribution and target distribution.

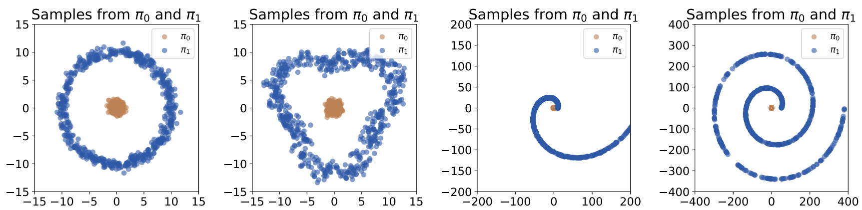


Figure 12: The circle dataset(**Left most**), irregular ring dataset (**Middle left**), spiral line dataset (**Middle right**), and spin dataset (**Right most**). Our goal is to make HOMO to learn a transport trajectory from distribution π_0 (**brown**) to distribution π_1 (**indigo**).

F.2 FIRST ORDER PLUS SECOND ORDER

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the target transport trajectory setting, we follow the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In the circle dataset, we all sample 400 points, both source distribution and target distribution. In the irregular ring dataset, we all sample 600 points, both source distribution and target distribution. In the spiral line dataset, we all sample 300 points, both source distribution and target distribution. In circle dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 1000 training steps. In irregular ring dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 1000 training steps. In spiral line dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 1000 training steps. In spiral line dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 1000 training steps.

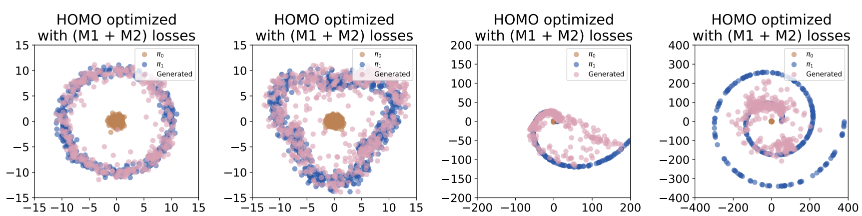


Figure 13: (M1+M2) **HOMO results on complex datasets with two kinds of loss: first-order and second-order terms.** The distributions generated by HOMO, in circle dataset(**Left most**), irregular ring dataset (**Middle left**), spiral line dataset (**Middle right**) and spin dataset (**Right most**). The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

F.3 First Order Plus Self-Consistency Term

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the target transport trajectory setting, we follow the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In the circle dataset, we all sample 400 points, both source distribution and target distribution. In the irregular ring dataset, we all sample 600 points, both source distribution and target distribution. In the spiral line dataset, we all sample 300 points, both source distribution and target distribution. In circle dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 1000 training steps. In irregular ring dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 1000 training steps. In spiral line dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 1000 training steps. In spiral line dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 1000 training steps.

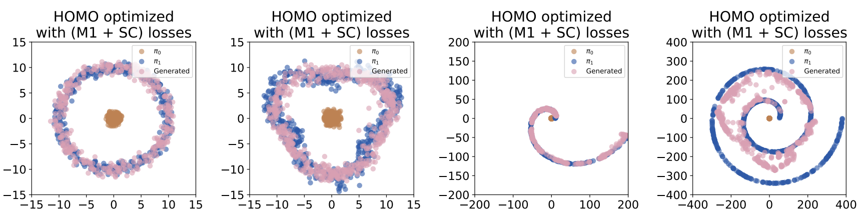


Figure 14: (M1+SC) **HOMO results on complex datasets with two kinds of loss: first-order and self-consistency terms.** The distributions generated by HOMO, in circle dataset(**Left most**), irregular ring dataset (**Middle left**), spiral line dataset (**Middle right**) and spin dataset (**Right most**). The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

F.4 Second Order Plus Self-Consistency Term

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the target transport trajectory setting, we follow the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In the circle dataset, we all sample 400 points, both source distribution and target distribution. In the irregular ring dataset, we all sample 600 points, both source distribution and target distribution. In the spiral line dataset, we all sample 300 points, both source distribution and target distribution. And in circle dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 100 training steps. In irregular ring dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 100 batch size, 0.005 learning rate, and 1000 training steps. In spiral line dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 100 training steps. In spiral line dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 1000 training steps.

F.5 HOMO

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the target transport trajectory setting, we follow the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In the circle dataset, we all sample 400 points, both source distribution and target distri-

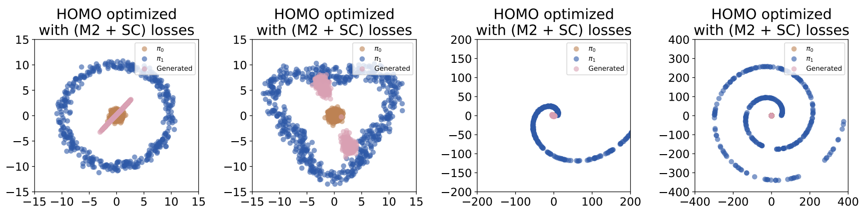


Figure 15: (M2+SC) **HOMO results on complex datasets with two kinds of loss: second-order and self-consistency terms.** The distributions generated by HOMO, in circle dataset(**Left most**), irregular ring dataset (**Middle left**), spiral line dataset (**Middle right**) and spin dataset (**Right most**). The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

bution. In the irregular ring dataset, we all sample 600 points, both source distribution and target distribution. In the spiral line dataset, we all sample 300 points, both source distribution and target distribution. And in circle dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 1000 training steps. In irregular ring dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 1000 training steps. In spiral line dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 1000 training steps. In spiral line dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 1000 training steps.

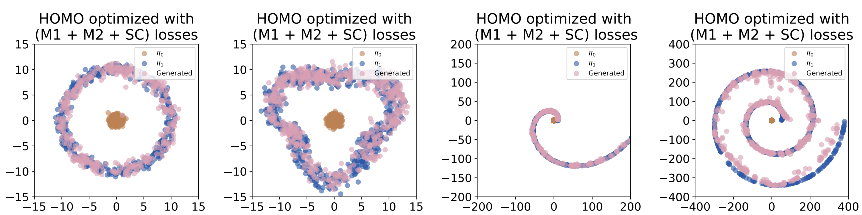


Figure 16: (M1+M2+SC) **HOMO results on complex datasets with three kinds of loss: first-order, second-order, and self-consistency terms.** The distributions generated by HOMO in circle dataset(**Left most**), irregular ring dataset (**Middle left**), spiral line dataset (**Middle right**), and spin dataset (**Right most**). The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

G Third-Order HOMO

This section extends HOMO to third-order dynamics and analyzes its performance on complex synthetic tasks. Section G.1 introduces the training and sampling algorithms incorporating third-order dynamics. Section G.2 compares two trajectory parameterization strategies for high-order systems. Section G.3 describes the 2 Round Spin, 3 Round Spin, and Dot-Circle datasets designed to test complex mode transitions. Section G.4 provides quantitative analysis through Euclidean distance metrics between generated and target distributions. Section G.5 evaluates the isolated impact of self-consistency constraints. Section G.6 examines first-order dynamics coupled with self-consistency regularization. Section G.7 studies the combined effect of first-, second-order dynamics and self-consistency. Finally, Section G.8 demonstrates full third-order HOMO with all optimization terms, analyzing trajectory linearity and mode fidelity under different trajectory settings.

G.1 Algorithm

Here we first introduce the training algorithm of our third-order HOMO:

Then we will discuss the sampling algorithm in third-order HOMO:

Algorithm 5 Third-Order HOMO Training

while not converged **do**
  $x_0 \sim \mathcal{N}(0, I), x_1 \sim D, (d, t) \sim p(d, t)$
  $\beta_t \leftarrow \sqrt{1-\alpha_t^2}$
  $x_t \leftarrow \alpha_t \cdot x_0 + \beta_t \cdot x_1$ ▷ Noise data point
  for first k batch elements **do**
    $\dot{s}_t^{\text{true}} \leftarrow \dot{\alpha}_t x_0 + \dot{\beta}_t x_1$ ▷ First-order target
    $\ddot{s}_t^{\text{true}} \leftarrow \ddot{\alpha}_t x_0 + \ddot{\beta}_t x_1$ ▷ Second-order target
    $\dddot{s}_t^{\text{true}} \leftarrow \dddot{\alpha}_t x_0 + \dddot{\beta}_t x_1$ ▷ Third-order target
    $d \leftarrow 0$
  end for
  for other batch elements **do**
    $s_t \leftarrow u_1(x_t, t, d)$ ▷ First small step of first order
    $\dot{s}_t \leftarrow u_2(u_1(x_t, t, d), x_t, t, d)$ ▷ First small step of second order
    $\ddot{s}_t \leftarrow u_3(u_2(u_1(x_t, t, d), x_t, t, d), u_1(x_t, t, d), x_t, t, d)$ ▷ First small step of third order
    $x_{t+d} \leftarrow x_t + d \cdot s_t + \frac{d^2}{2}\dot{s}_t + \frac{d^6}{3}\ddot{s}_t$ ▷ Follow ODE
    $s_{t+d} \leftarrow u_1(x_{t+d}, t+d, d)$ ▷ Second small step of first order
    $\dot{s}_t^{\text{target}} \leftarrow$ stopgrad $(s_t + s_{t+d})/2$ ▷ Self-consistency target of first order
  end for
  $\theta \leftarrow \nabla_\theta(\|u_1(x_t, t, 2d) - \dot{s}_t^{\text{true}}\|^2$
    $+\|u_2(u_1(x_t, t, 2d), x_t, t, 2d) - \ddot{s}_t^{\text{true}}\|^2$
    $+\|u_3(u_2(u_1(x, t, d), x, t, d), u_1(x, t, d), x, t, d) - \dddot{s}_t^{\text{true}}\|^2$
    $+\|u_1(x_t, t, 2d) - \dot{s}_t^{\text{target}}\|^2$
end while

Algorithm 6 Third-Order HOMO Sampling

$x \sim \mathcal{N}(0, I)$
$d \leftarrow 1/M$
$t \leftarrow 0$
for $n \in [0, \dots, M-1]$ **do**
  $x \leftarrow x + d \cdot u_1(x, t, d) + \frac{d^2}{2} \cdot u_2(u_1(x, t, d), x, t, d) + \frac{d^3}{6} \cdot u_3(u_2(u_1(x, t, d), x, t, d), u_1(x, t, d), x, t, d)$
  $t \leftarrow t + d$
end for
return x

G.2 Trajectory setting

We have trajectory as:

$$z_t = \alpha_t z_0 + \beta_t z_1$$

In original trajectory, we choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. And new trajectory as $\alpha_t = 1 - (3t^2 - 2t^3)$ and $\beta_t = 3t^2 - 2t^3$.

G.3 Dataset

Here, we introduce three datasets we use: 2 Round spin, 3 Round spin, and Dot-Circle datasets. In 2 Round spin dataset and 3 Round spin dataset, we both sample 600 points from Gaussian distribution with 0.3 variance for both source distribution and target distribution. In Dot-Circle datasets, we sample 300 points from the center dot and 300 points from the outermost circle, combine them as source distribution, and then sample 600 points from 2 round spin distribution.

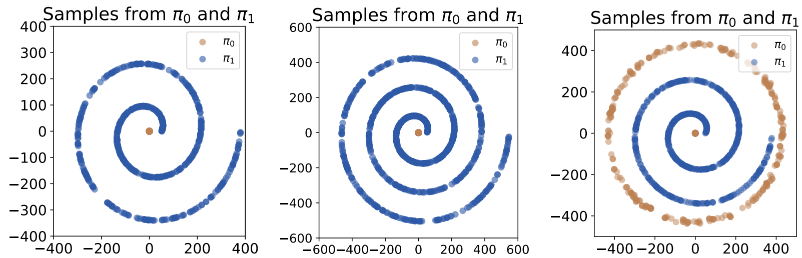


Figure 17: The 2 Round spin dataset(**Left**), 3 Round spin dataset(**Middle**), and Dot-Circle datasets(**Right**). Our goal is to make HOMO learn a transport trajectory from distribution π_0 (**brown**) to distribution π_1 (**indigo**).

G.4 Euclidean distance loss

Here, we present the Euclidean distance loss performance of four different loss terms combined under the original trajectory setting and the new trajectory setting.

Table 4: **Euclidean distance loss of three complex distribution datasets under new trajectory setting.** Lower values indicate more accurate distribution transfer results. Optimal values are highlighted in **Bold**. And Underlined numbers represent the second best (second lowest) loss value for each dataset (row). For the qualitative results of a mixture of Gaussian experiments, please refer to Figure 1.

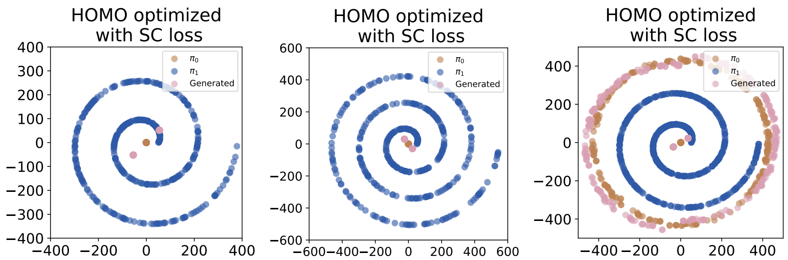
Loss terms	**2 Round spin**	**3 Round spin**	**Dot-Circle**
SC	41.265	48.201	87.407
M1 + SC	14.926	18.376	30.027
M1 + M2 + SC	<u>11.435</u>	<u>12.422</u>	<u>24.712</u>
M1 + M2 + SC + M3	**4.701**	**9.261**	**21.968**

G.5 Only Self-Consistency Term

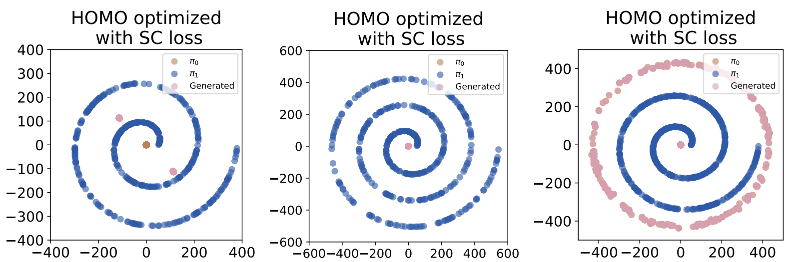
We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the first line, we use the original transport trajectory setting, followed by the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In 2 Round spin datasets and 3 Round spin datasets, we sample 400 points, both source distribution and target distribution. In the Dot-Circle dataset, we sample 600 points from both source distribution and target distribution, 300 points of source points from the circle, and another 300 from the center dot. In 2-round dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 180 training steps. In 3-round spin dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 180 training steps. In Dot-Circle dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 180 training steps.

G.6 First Order Plus Self-Consistency

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the first line, we use the original transport trajectory setting, followed by the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In 2 Round spin datasets and 3 Round spin datasets, we sample 400 points in both source distribution and target distribution. And in the Dot-Circle dataset, we sample 600 points from



(a) (original)SC / 2 Round (b) (original)SC / 3 Round (c) (original)SC / DotPlus-Circle



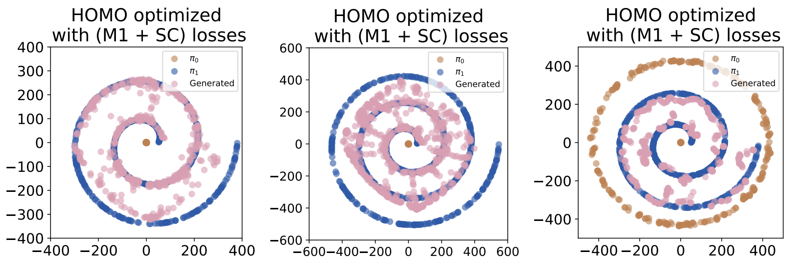
(d) (new)SC / 2 Round (e) (new)SC / 3 Round (f) (new)SC / DotPlusCircle

Figure 18: (SC) The distributions generated by HOMO are only optimized by self-consistency loss. **Upper row(original trajectory setting):** Figure (a), in 2 Round spin dataset. Figure (b), in 3 Round spin dataset. Figure (c), in Dot-Circle dataset. **Lower row(new trajectory setting):** Figure (d), in 2 Round spin dataset. Figure (e), in 3 Round spin dataset. Figure (f), in Dot-Circle dataset. The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

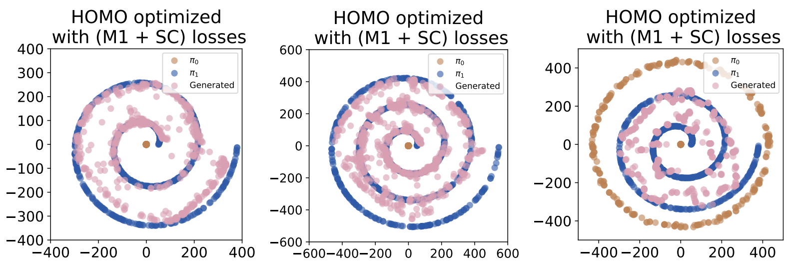
both source distribution and target distribution, 300 points of sources points from the circle, and another 300 from the center dot. In 2 Round dataset training, we use ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimension, 800 batch size, 0.005 learning rate, and 1000 training steps. In 3 Round spin dataset training, we also use ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimension, 1000 batch size, 0.005 learning rate, and 2000 training steps. And in Dot-Circle dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 10000 training steps.

G.7 First Order Plus Second Order Plus Self-Consistency

We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the first line, we use the original transport trajectory setting, followed by the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In 2 Round spin datasets and 3 Round spin datasets, we sample 400 points, both source distribution and target distribution. And in the Dot-Circle dataset, we sample 600 points from both source distribution and target distribution, 300 points of source points from the circle, and another 300 from the center dot. In 2 Round dataset training, we use ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimension, 800 batch size, 0.005 learning rate, and 1000 training steps. In 3 Round spin dataset training, we also use ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimension, 1000 batch size, 0.005 learning rate, and 2000 training steps. And in Dot-Circle dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 10000 training steps.



(a) (original)(M1+SC) / 2 Round (b) (original)(M1+SC) / 3 Round (c) (original)(M1+SC) / Dot-Circle



(d) (new)(M1 + SC) / 2 Round (e) (new)(M1 + SC) / 3 Round (f) (new)(M1+SC) / Dot-Circle

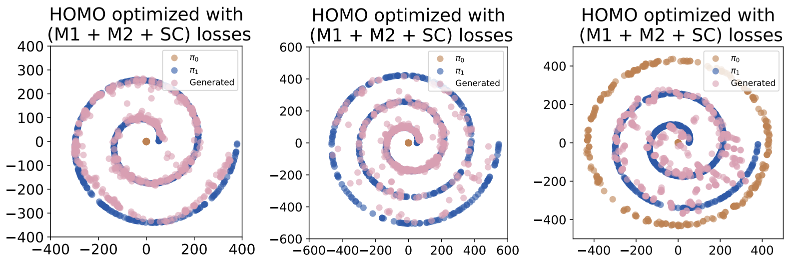
Figure 19: (M1+SC) The distributions generated by HOMO are only optimized by first-order loss and self-consistency loss. **Upper row(original trajectory setting):** Figure (a), in 2 Round spin dataset. Figure (b), in 3 Round spin dataset. Figure (c), in Dot-Circle dataset. **Lower row(new trajectory setting):** Figure (d), in 2 Round spin dataset. Figure (e), in 3 Round spin dataset. Figure (f), in Dot-Circle dataset. The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

G.8 Third-Order HOMO

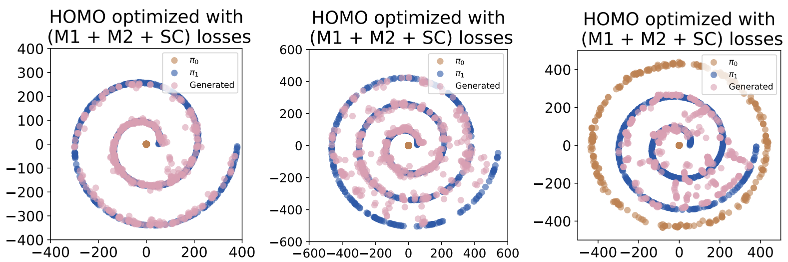
We optimize models by the sum of squared error(SSE). The source distribution and target distribution are all Gaussian distributions. For the first line, we use the original transport trajectory setting, followed by the VP ODE framework from Liu et al. (2022b), which is $x_t = \alpha_t x_0 + \beta_t x_1$. We choose $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\beta_t = \sqrt{1-\alpha_t^2}$, with hyperparameters $a = 19.9$ and $b = 0.1$. In 2 Round spin datasets and 3 Round spin datasets, we sample 400 points, both source distribution and target distribution. In the Dot-Circle dataset, we sample 600 points, both source distribution and target distribution, 300 points of source points from the circle, and another 300 from the center dot. In 2-round dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 800 batch size, 0.005 learning rate, and 1000 training steps. In 3-round spin dataset training, we also use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1000 batch size, 0.005 learning rate, and 2000 training steps. In Dot-Circle dataset training, we use an ODE solver and Adam optimizer, with 2 hidden layer MLP, 100 hidden dimensions, 1600 batch size, 0.005 learning rate, and 10000 training steps.

H Computational Cost and Optimization Cost

We profile computational efficiency on the Apple MacBook Air (M1 8GB) with an 8-core CPU. Through systematic analysis, we observe three critical tradeoffs: (1) The M2 configuration demonstrates an 8.15× FLOPs increase over M1 while achieving 4.07× parameter expansion, revealing the fundamental FLOPs-parameters scaling relationship. (2) The self-consistency (SC) term introduces minimal computational overhead, with the M2+SC configuration maintaining 144.73 it/s versus vanilla M2's 146.34 it/s (1.1% throughput reduction). (3) Architectural innovations yield



(a) (original) (M1+M2+SC) / 2 Round (b) (original) (M1+M2+SC) / 3 Round (c) (original) (M1+M2+SC) / Dot-Circle



(d) (new) (M1+M2+SC) / 2 Round (e) (new) (M1+M2+SC) / 3 Round (f) (new) (M1+M2+SC) / Dot-Circle

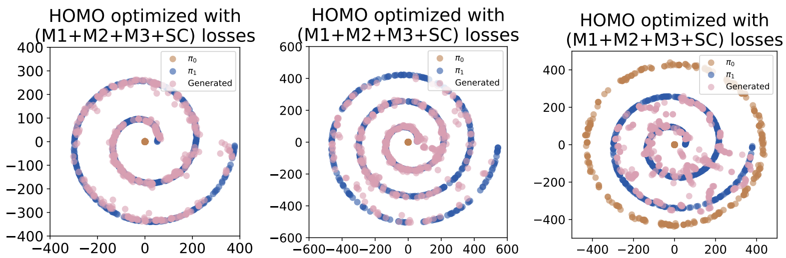
Figure 20: (M1+M2+SC) The distributions generated by HOMO are only optimized by first-order loss and second order loss, and self-consistency loss. **Upper row(original trajectory setting):** Figure (a), in 2 Round spin dataset. Figure (b), in 3 Round spin dataset. Figure (c), in Dot-Circle dataset. **Lower row(new trajectory setting):** Figure (d), in 2 Round spin dataset. Figure (e), in 3 Round spin dataset. Figure (f), in Dot-Circle dataset. The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

substantial gains - the Shortcut Model (M1+SC) achieves 33.6% faster iterations than vanilla M1 (283.20 vs 477.03 it/s) with comparable parameter counts. Table 5 quantifies these effects through comprehensive benchmarking:

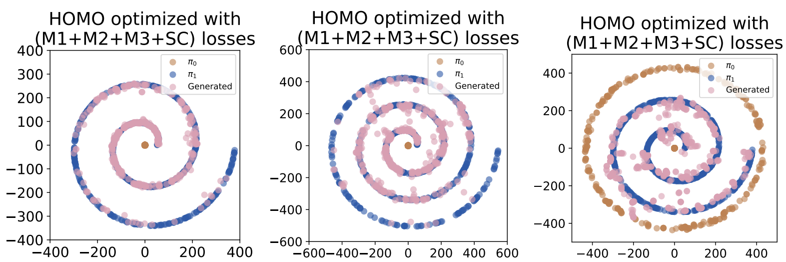
Table 5: Computational Cost Analysis of Different Configurations

Configuration	**FLOPs (M)**	**Params (K)**	**Training Speed (it/s)**
M1	8.400	10.702	477.03
M2	68.480	43.608	146.34
M3	8.400	10.702	357.45
M1 + M2	16.960	21.604	248.15
M2 + SC	68.480	43.608	144.73
(Shortcut Model) M1 + SC	8.480	10.802	283.20
M1 + M2 + SC	68.480	43.608	136.46
M1 + M2 + M3 + SC	103.680	66.012	122.18

Notably, our architecture maintains practical viability even for high-order extensions - the third-order HOMO configuration (M1+M2+M3+SC) sustains 122.18 it/s despite requiring 12.34× more FLOPs than the base M1 model. This demonstrates our method's ability to balance computational complexity with real-time performance requirements.



(a) (original)(M1+M2+M3+SC) / 2 Round
(b) (original)(M1+M2+M3+SC) / 3 Round
(c) (original)(M1+M2+M3+SC) / Dot-Circle



(d) (new)(M1+M2+M3+SC) / 2 Round
(e) (new)(M1+M2+M3+SC) / 3 Round
(f) (new)(M1+M2+M3+SC) / Dot-Circle

Figure 21: (M1+M2+M3+SC) The distributions generated by Third-Order HOMO, optimized by first-order loss and second-order loss, third-order loss and self-consistency loss. **Upper row(original trajectory setting):** Figure (a), in 2 Round spin dataset. Figure (b), in 3 Round spin dataset. Figure (c), in Dot-Circle dataset. **Lower row(new trajectory setting):** Figure (d), in 2 Round spin dataset. Figure (e), in 3 Round spin dataset. Figure (f), in Dot-Circle dataset. The source distribution, π_0 (**brown**), and the target distribution, π_1 (**indigo**), are shown, along with the generated distribution (**pink**).

I CONCLUSION

In this work, we introduced HOMO (High-Order Matching for One-Step Shortcut Diffusion), a framework that incorporates high-order dynamics into Shortcut models. By leveraging high-order supervision, HOMO improves the geometric consistency and precision of learned trajectories.

Theoretical analyses show that high-order supervision ensures stability and generalization across different stages of the generative process. Experiments demonstrate that HOMO outperforms the original Shortcut models Frans et al. (2025), achieving better distributional alignment and fewer suboptimal trajectories.

The integration of high-order terms sets a new standard for geometrically-aware generative modeling, emphasizing the importance of capturing higher-order dynamics for accurate transport learning. Our results highlight the potential of high-order supervision to enhance the fidelity and robustness of flow-based generative models.