HIGH-ORDER DRIFTING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Drifting models enable efficient one-step generation by evolving the model distribution through a kernel-induced transport field during training. Existing drifting methods, however, typically use only the local field value at sampled anchors, ignoring the local response structure of the same field. This leaves local curvature and higher-order structural mismatch unused, limiting convergence and generation fidelity. To address this problem, we propose Higher-Order Drifting Models, which exploit the local response hierarchy of the original kernel transport field. We show that the response derivatives of the field form a hierarchy of local cumulants; for Gaussian kernels, these cumulants correspond to spatial derivatives of the smoothed log-density ratio, extending the score-matching interpretation of drifting to higher orders. We further show that, under a local linearization, the higher-order responses act as a spectral reweighting that enhances the effective damping of transition-region modes without altering the asymptotic high-frequency behavior of the kernel. The experiments show that these responses provide complementary optimization signals while preserving one-step inference, accelerating convergence and improving generation accuracy, with diminishing returns as the order increases.

1 INTRODUCTION

Modern generative models, including diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2020) and flow-matching methods (Lipman et al., 2022; Albergo et al., 2025; Liu et al., 2022), achieve high-quality generation by learning continuous transformations from simple priors to data distributions. However, these approaches typically require iterative numerical integration or multiple network evaluations during inference, creating substantial computational costs. This limitation has motivated the development of one-step generative models, including consistency models (Song et al., 2023; Song & Dhariwal, 2024), flow maps (Boffi et al., 2024; 2026), and MeanFlows (Geng et al., 2026; Huang et al., 2026). Drifting models (Deng et al., 2026) take a different perspective: instead of learning an inference-time trajectory, they evolve the model distribution during training through a sample-based kernel transport field and train the generator toward the resulting targets.

The kernel transport field offers an interpretable view of distribution evolution in drifting models. Recent theoretical studies have established its connection to score-based learning and optimal transport, providing a principled understanding of drifting dynamics (Lai et al., 2026; Turan et al., 2026). However, vanilla drifting models still suffer from slow convergence and limited generation fidelity. Existing improvements mainly focus on external optimization strategies, including second-order dynamics (Brown et al., 2026), friction scheduling (Kazanskii et al., 2026), kernel-based transport refinement (He et al., 2026; Esteban-Casadevall et al., 2026), bias correction (Zhang et al., 2026), and computational acceleration (Falahati et al., 2026; Darouich et al., 2026). Despite these advances, the intrinsic structure of the transport field remains underexplored. Specifically, existing drifting methods rely primarily on first-order field values at sampled anchor points, while neglecting higher-order local responses that characterize fine-grained distribution mismatch. This limitation results in incomplete transport guidance and restricts optimization efficiency and generation accuracy.

To address this limitation, we propose Higher-Order Drifting Models, an intrinsic extension of drifting frameworks that exploits the local response hierarchy of the kernel transport field. An overview is provided in Figure 1. Instead of introducing external optimization mechanisms, our approach enriches the drifting update by incorporating additional structural information naturally contained in the field. We show that the response derivatives of the field form a hierarchy of local cumulants, and that for Gaussian kernels these cumulants correspond to spatial derivatives of the smoothed log-density ratio,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

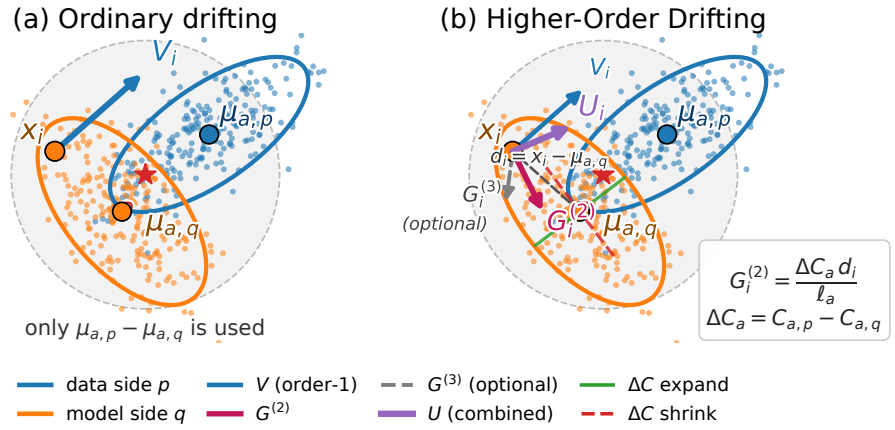


Figure 1: Overview of Higher-Order Drifting. (a) Ordinary drifting uses only the local mean difference between the data and model neighborhoods. (b) Higher-Order Drifting keeps the same kernel, anchors, and regression interface, but additionally corrects particles using the local covariance discrepancy. The green and red axes indicate the expansion and contraction directions of this discrepancy. The optional third-order correction is shown dashed and is disabled by default. The combined update is used as the stop-gradient regression target. All higher-order computation is confined to training; inference remains a single generator evaluation.

extending the score-matching interpretation of drifting to higher orders. We further show that, under a local linearization, the higher-order responses act as a spectral reweighting that enhances the effective damping of transition-region modes without altering the asymptotic high-frequency behavior of the kernel. This provides a principled account of why higher-order responses help, rather than treating them as an external shape constraint.

We validate the framework through theory and extensive experiments. The results show that higher-order local responses provide complementary optimization signals that accelerate training convergence and improve final generation precision, while preserving the one-step inference property of standard drifting. We also analyze the trade-off across response orders: the optimization gain brought by high-order responses gradually diminishes as the order increases, providing practical guidance for selecting the number of terms in downstream tasks.

In summary, our main contributions are as follows:

- We propose Higher-Order Drifting, an intrinsic extension of drifting that exploits local response derivatives of the kernel transport field beyond its mean. These responses are used only as stop-gradient regression targets, preserving the generator architecture, regression interface, and one-step inference.
- We introduce a local response-cumulant hierarchy. For Gaussian kernels, it recovers spatial derivatives of the smoothed log-density ratio, extending the score-matching view of drifting and yielding a polynomial spectral reweighting of transition-region modes under local linearization.
- We derive practical higher-order particle corrections that preserve one-step inference, and validate their convergence and fidelity benefits through theory and experiments.

2 Related Work

One-step generative modeling. Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2020) and flow-based methods (Lipman et al., 2022; Albergo et al., 2025; Liu et al., 2022) achieve strong generation quality by learning continuous transformations, but typically require iterative inference. This has motivated a growing body of one-step generative models, including consistency models (Song et al., 2023; Song & Dhariwal, 2024), flow maps (Boffi et al., 2024; 2026), and MeanFlows (Geng et al., 2026; Huang et al., 2026). More recent works further diversify this landscape: simple distillation for one-step diffusion models (Zhu et al., 2026), continuous-time

flow map distillation (Sabour et al., 2026), and inductive moment matching (Zhou et al., 2025). A frequency-domain analysis of how one-step generation emerges from diffusion pre-training is provided by (Zheng & Yang, 2025).

Theory and extensions of drifting models. Recent works connect drifting models to score-based generative models. Lai et al. (2026) constructed a unified theoretical framework via kernel decomposition and error bounds, while Turan et al. (2026) interpreted drifting dynamics as a Wasserstein gradient flow of smoothed KL divergence and analyzed its frequency-domain convergence. These results mainly characterize the first-order transport field. Since Deng et al. (2026), follow-up studies have extended drifting from several perspectives: He et al. (2026) proposed Sinkhorn-Drifting based on Sinkhorn divergence gradient flow, and Esteban-Casadevall et al. (2026) generalized it via kernel-gradient drifting for manifold-structured data; Kazanskii et al. (2026) introduced friction-augmented DMF to suppress spurious fixed points, while Brown et al. (2026) constructed second-order drifting via phase-space lifting; unlike this line of work, our method stays in the original sample space and augments only the local response of the existing transport field. Zhang et al. (2026) proposed an analytical correction for mini-batch estimation, and DriftXpress (Falahati et al., 2026) reduced field-evaluation cost via low-rank RKHS approximation; SymDrift Darouich et al. (2026) realized symmetry-aware field construction. Collectively, these works make drifting more rigorous, stable, efficient, and geometry-adaptive, but they mainly introduce external mechanisms or optimize the same first-order field. The intrinsic local structure of the transport field remains underexplored.

3 Background and Preliminary

We first summarize the population objects from ordinary drifting that will be reused throughout the paper. Let p_{data} be the data distribution on $\mathbb{R}^d$, let $z \sim \pi$ be a latent variable, and let

$$f_\theta : \mathbb{R}^{d_z} \to \mathbb{R}^d, \qquad q_\theta = (f_\theta)_{\#}\pi \tag{1}$$

be a one-step generator. At a training step, generated samples are $x_i = f_\theta(z_i)$ and data samples are $y_j \sim p_{\mathrm{data}}$. For compactness, we write p and q for p_{data} and q_θ when they appear as subscripts.

Kernel-local neighborhoods. For an anchor $a \in \mathbb{R}^d$ and Gaussian bandwidth $h > 0$, define the normalized Gaussian kernel

$$G_h(u) = (2\pi h^2)^{-d/2} \exp\left(-\frac{\|u\|_2^2}{2h^2}\right), \qquad k_h(a, y) = G_h(a - y). \tag{2}$$

Rather than introducing a separate measure for each kernel neighborhood, for any distribution $r \in \{p_{\mathrm{data}}, q_\theta\}$ and integrable function φ we define the normalized kernel-local expectation

$$\mathbb{E}_{a,r}[\varphi(Y)] := \frac{\mathbb{E}_{Y\sim r}[k_h(a, Y)\varphi(Y)]}{\mathbb{E}_{Y\sim r}[k_h(a, Y)]}. \tag{3}$$

The corresponding local mean is

$$\mu_{a,r} := \mathbb{E}_{a,r}[Y]. \tag{4}$$

Suppressing implementation-specific mass weights, the population drifting field is

$$F_a^{(1)} = \mu_{a,p} - \mu_{a,q}. \tag{5}$$

For Gaussian smoothing, Tweedie's identity gives (Lai et al., 2026; Turan et al., 2026)

$$F_a^{(1)} = h^2 \nabla_a \log \frac{p_h(a)}{q_h(a)}, \tag{6}$$

where p_h and q_h denote the Gaussian-smoothed densities. Eqn. (6) is the key first-order reference point: ordinary drifting already represents a smoothed score mismatch.

Stop-gradient regression. As in ordinary drifting, a sample-space direction U_i is converted into a generator update through

$$t_i = \mathrm{sg}(x_i + U_i), \qquad \mathcal{L}_{\mathrm{reg}}(\theta) = \frac{1}{nd}\sum_{i=1}^{n} \|f_\theta(z_i) - t_i\|_2^2. \tag{7}$$

Higher-Order Drifting changes how U_i is constructed during training. The generator architecture, regression interface, and one-step inference procedure remain unchanged.

4 HIGHER-ORDER DRIFTING

Ordinary drifting summarizes the kernel transport field by its local value $F_a(0) = \mu_{a,p} - \mu_{a,q}$ at each anchor. This value is only the first local response of the field. It does not describe how the same kernel neighborhood changes when its mass is infinitesimally reweighted. Our method keeps the same kernel, and the same regression interface, but additionally exploits this local response structure. We first define the population response hierarchy, then derive practical second- and third-order particle corrections, and finally discuss the Gaussian specialization and its spectral consequences.

4.1 FROM FIELD VALUES TO LOCAL RESPONSES

Fix an anchor a and its kernel-induced neighborhood, we perturb this neighborhood by a small exponential tilt and observe how its local mean changes. For $r \in \{p, q\}$ and $s \in \mathbb{R}^d$, define the tilted kernel-local expectation

$$\mathbb{E}_{a,r,s}[\varphi(Y)] := \frac{\mathbb{E}_{Y\sim r}\big[k_h(a,Y)\exp(s^\top(Y-a))\varphi(Y)\big]}{\mathbb{E}_{Y\sim r}[k_h(a,Y)\exp(s^\top(Y-a))]}. \tag{8}$$

At $s = 0$, this reduces to the kernel-local expectation in eq. (3). Let $\mu_{a,r}(s) = \mathbb{E}_{a,r,s}[Y]$ and $F_a(s) = \mu_{a,p}(s) - \mu_{a,q}(s)$. The unperturbed value $F_a(0)$ is ordinary drifting. The derivatives with respect to s describe how the same local field changes when mass inside the neighborhood is infinitesimally reweighted.

To make this relation precise, define the local log-partition function

$$\Psi_{a,r}(s) = \log \frac{\mathbb{E}_{Y\sim r}\big[k_h(a,Y)\exp(s^\top(Y-a))\big]}{\mathbb{E}_{Y\sim r}[k_h(a,Y)]}.$$

Because $\mu_{a,r}(s) = a + \nabla_s \Psi_{a,r}(s)$, the response derivatives are derivatives of a cumulant-generating function for the kernel-local neighborhood.

Proposition 1 (Local response–cumulant hierarchy) *Assume $\Psi_{a,r}(s)$ is finite in a neighborhood of $s = 0$. Then*

$$D_s^j \mu_{a,r}(0) = \kappa_{j+1,a,r}, \qquad j \geq 1, \tag{9}$$

where $\kappa_{j+1,a,r}$ is the $(j+1)$st cumulant tensor associated with the normalized kernel-local expectation $\mathbb{E}_{a,r}$. In particular, $D_s\mu_{a,r}(0) = C_{a,r}$ and $D_s^2\mu_{a,r}(0) = \kappa_{3,a,r}$.

The proposition shows that the hierarchy is generated by one response operator rather than by a collection of independently chosen statistics. Expanding the field around $s = 0$ gives

$$\begin{aligned} F_a(s) &= F_a(0) + \Delta C_a\, s + \frac{1}{2}\Delta\kappa_{3,a}[s,s] + O(\|s\|^3), \\ \Delta C_a &= C_{a,p} - C_{a,q}, \Delta\kappa_{3,a} = \ \kappa_{3,a,p} - \kappa_{3,a,q}. \end{aligned} \tag{10}$$

We index the hierarchy by the corresponding distributional/spatial order: ordinary drifting is order one, covariance gives order two, and the third cumulant gives order three. This indexing differs by one from the number of derivatives with respect to the tilt parameter s.

4.2 HIGHER-ORDER OBJECTIVES AND THE SECOND-ORDER CORRECTION

The propositions above define the conceptual hierarchy in terms of local cumulants. For $m \geq 2$, let $\Delta\kappa_a^{(m)} = \kappa_{m,a,p} - \kappa_{m,a,q}$. We use the dimensionally normalized population discrepancy

$$\mathcal{E}_m = \frac{1}{4}\sum_a \frac{\|\Delta\kappa_a^{(m)}\|_{\mathrm{F}}^2}{\ell_a^{m-1}}, \tag{11}$$

where $\ell_a > 0$ is a local variance scale. The normalization gives every order the units of a squared displacement, so the negative particle gradient has the units of a displacement. The corresponding order-m correction is

$$G^{(m)} = -\nabla_X \mathcal{E}_m, \tag{12}$$

with anchors and kernel weights frozen while forming the local surrogate, as in the stop-gradient construction of ordinary drifting.

Our default practical instantiation uses $m = 2$. We also develop an optional $m = 3$ extension in section 4.3. At these orders, the cumulants coincide with the covariance and the third central moment, respectively. We do not rely on an interchangeability of moments and cumulants at higher orders.

At order two, $C_{a,r} = \mathbb{E}_{a,r}[(Y - \mu_{a,r})(Y - \mu_{a,r})^\top]$, and

$$\mathcal{E}_2 = \frac{1}{4} \sum_a \frac{\|C_{a,p} - C_{a,q}\|_{\mathrm{F}}^2}{\ell_a}. \tag{13}$$

Thus, the second-order term penalizes a mismatch in the local shape of the data and model neighborhoods. Here w_{ai}^q denotes the normalized generated-side kernel weight for anchor a and particle i, and in practice we use the mean local variance as ℓ_a (see Appendix for details). With normalized generated-side weights w_{ai}^q held fixed, differentiating eq. (13) gives

$$G_i^{(2)} = \sum_a w_{ai}^q \frac{C_{a,p} - C_{a,q}}{\ell_a}(x_i - \mu_{a,q}). \tag{14}$$

Geometrically, the covariance discrepancy expands or contracts generated particles along directions in which the local model shape differs from the data shape.

4.3 Optional Third-Order Extension

A full third-order tensor requires $O(d^3)$ storage and computation. To retain the third-order response without materializing this tensor, we use independent isotropic probe vectors $\xi, \zeta \in \mathbb{R}^d$ satisfying $\mathbb{E}[\xi\xi^\top] = \mathbb{E}[\zeta\zeta^\top] = I$ and define

$$\mathcal{T}_{a,r}(\xi, \zeta) = \mathbb{E}_{a,r}\left[(Y - \mu_{a,r})(\xi^\top(Y - \mu_{a,r}))(\zeta^\top(Y - \mu_{a,r}))\right]. \tag{15}$$

For the third cumulant tensor, $\mathcal{T}_{a,r}(\xi, \zeta) = \kappa_{3,a,r}[\xi, \zeta]$. Independent isotropic probes satisfy $\mathbb{E}_{\xi,\zeta}\|\Delta\kappa_{3,a}[\xi, \zeta]\|_2^2 = \|\Delta\kappa_{3,a}\|_{\mathrm{F}}^2$, so a Monte Carlo approximation to the third-order objective is

$$\mathcal{E}_3 \approx \frac{1}{4K} \sum_{a=1}^{n} \sum_{k=1}^{K} \frac{\|\mathcal{T}_{a,p}(\xi_k, \zeta_k) - \mathcal{T}_{a,q}(\xi_k, \zeta_k)\|_2^2}{\ell_a^2}. \tag{16}$$

The probes reduce the tensor cost; they do not change the underlying third-order population quantity. Differentiating eq. (16) with respect to x_i yields the projected particle correction $G_i^{(3)}$, analogous to eq. (14). In the main experiments we set $\alpha_3 = 0$; this optional extension is activated only when explicitly enabled.

4.4 Training Objective

The final sample-space direction combines ordinary drifting with whichever response orders are enabled,

$$U_i = V_i + \alpha_2 G_i^{(2)} + \alpha_3 G_i^{(3)}, \tag{17}$$

with absent orders set to zero.

To control the second-order correction, we define the effective coefficient α_2 as

$$\alpha_2 = w_2(t) r_{\max}^{(2)} \max\left(0, \cos(\langle V, G^{(2)} \rangle)\right) \frac{\mathrm{RMS}(V)}{\mathrm{RMS}(G^{(2)})}, \tag{18}$$

where V is the first-order drift and $G^{(2)}$ is the covariance-based second-order direction. $w_2(t) = \min(1, t/T_{\mathrm{warm}}^{(2)})$ is a linear warm-up over the first $T_{\mathrm{warm}}^{(2)}$ steps. The factor $\max\left(0, \cos(\langle V, G^{(2)} \rangle)\right)$ is a cosine gate that suppresses updates opposing the first-order drift, and $r_{\max}^{(2)} \frac{\mathrm{RMS}(V)}{\mathrm{RMS}(G^{(2)})}$ is a trust-region bound that controls the scale of second-order term. α_3 is constructed in the similar way. In the default configuration we set $\alpha_3 = 0$; the third-order term is enabled only in the optional extension of. The stop-gradient operator is denoted by. The generator is then updated with the same stop-gradient regression objective as ordinary drifting, eq. (7).

4.5 GAUSSIAN INTERPRETATION: SCORE, CURVATURE, AND BEYOND

The population hierarchy above is kernel-agnostic. For Gaussian kernels, it admits a direct interpretation in terms of spatial derivatives of the same smoothed log-density ratio that appears in ordinary drifting, which gives a unified reading of the added terms rather than treating them as external shape constraints.

Proposition 2 (Gaussian spatial hierarchy) *For Gaussian kernel bandwidth h,*

$$\mu_{a,p} - \mu_{a,q} = h^2 \nabla_a \log \frac{p_h(a)}{q_h(a)}, \tag{19}$$

$$C_{a,p} - C_{a,q} = h^4 \nabla_a^2 \log \frac{p_h(a)}{q_h(a)}, \tag{20}$$

$$\kappa_{3,a,p} - \kappa_{3,a,q} = h^6 \nabla_a^3 \log \frac{p_h(a)}{q_h(a)}. \tag{21}$$

More generally, for $m \geq 2$,

$$\Delta\kappa_{m,a} = h^{2m} \nabla_a^m \log \frac{p_h(a)}{q_h(a)}. \tag{22}$$

The proof is given in the appendix. The additive $h^2 I$ term in each individual covariance cancels in the data–model difference. Consequently, order two measures local curvature mismatch and order three measures the next spatial variation of that curvature. This interpretation is specific to Gaussian smoothing; the response hierarchy itself remains well defined for more general kernel neighborhoods.

4.6 SPECTRAL INTERPRETATION

We connect the higher-order hierarchy to the spectral analysis of drifting dynamics by Turan et al. (2026). For a Gaussian kernel, they show that the drift operator equals a score difference on smoothed distributions, $V_{p,q}^{(\sigma)}(x) = \sigma^2 \nabla_x \log(p_\sigma(x)/q_\sigma(x))$, with bandwidth σ. Linearizing the McKean–Vlasov dynamics around equilibrium $q = p + \delta$ under a local-homogeneity approximation yields the mode-resolved decay rate

$$\lambda_{\text{Gauss}}(\xi) = h^2 |\xi|^2 \, e^{-h^2|\xi|^2/2}, \tag{23}$$

where we have translated their bandwidth to our h. The rate peaks at $h|\xi| = \sqrt{2}$ and decays exponentially beyond, producing a high-frequency bottleneck: modes with $h|\xi| \gg 1$ are frozen out, and the corresponding convergence time scales as $\exp(h^2|\xi|^2/2)/(h^2|\xi|^2)$.

The same linearization can be applied to the higher-order corrections. Expanding $q = p + \delta$ and keeping only first-order terms in δ, the second-order correction becomes a linear operator acting on δ. Because the covariance discrepancy ΔC_a is a second spatial derivative of $\log(p_h/q_h)$ under Gaussian kernels by eq. (20), and because the factor $(x_i - \mu_{a,q})$ contributes a displacement that is integrated against the kernel, the Fourier symbol of $G^{(2)}$ carries an extra factor of $|\xi|^2$ relative to the ordinary drift. More generally, the m-th order correction acquires an extra polynomial weight of order m. This yields the following.

Proposition 3 (Polynomial spectral reweighting) *Under the local-homogeneity linearization of Turan et al. (2026), enabling response orders up to M gives a mode-resolved effective decay rate*

$$\lambda_{\text{eff}}(\xi) = h^2|\xi|^2 e^{-h^2|\xi|^2/2} \, W_M(h|\xi|), \qquad W_M(u) = 1 + \sum_{m=2}^{M} c_m \, u^m, \tag{24}$$

for dimensionless constants c_m determined by α_m, the local variance scale ℓ_a, and the kernel bandwidth h. Consequently:

- *(i) In the transition region $h|\xi| \sim 1$ where the Gaussian rate is near its peak, $W_M(h|\xi|) > 1$ for $c_m > 0$, so the higher-order terms enhance the effective damping of the modes that carry the bulk of the information.*

(ii) In the high-frequency region $h|\xi| \to \infty$, the factor $e^{-h^2|\xi|^2/2}$ dominates over any polynomial W_M, so the exponential bottleneck is unchanged.

The proof is deferred to Appendix A.4. Proposition 3 formalizes the sense in which the higher-order responses act as a *static spectral whitening* at fixed bandwidth: they enhance the effective damping of modes in the transition region where the Gaussian rate is already informative, without altering the asymptotic high-frequency behavior of the kernel. Although proposition 3 is stated for general M, our default instantiation uses $M = 2$; the $M = 3$ case is the optional extension.

5 Experiments

We evaluate Higher-Order Drifting on three targets of increasing scale and complexity: a 16-dimensional Gaussian mixture, MNIST, and ImageNet at 256×256 resolution. The Gaussian mixture supports controlled mechanism studies, isolating the second-order correction from update-magnitude changes and comparing directional components. MNIST verifies the gain under a neural generator and an image-based metric. ImageNet measures large-scale benefit by fine-tuning a pre-trained Drifting checkpoint under a fixed evaluation protocol. In all settings, the kernel, anchors, regression interface, and single-step inference remain identical to ordinary drifting.

5.1 16-dimensional Gaussian mixture

We evaluate the method on a 16-dimensional Gaussian mixture with eight modes at the rotated corners of a three-dimensional cube, each at radius 2.5, and component standard deviation 0.25. Unless stated otherwise, training uses batch size 128, bandwidth ratio 0.4 (i.e., $h = 0.4b$ where b is the mean anchor-to-data distance), Adam, and 1000 optimization steps. We report sliced Wasserstein distance (SWD) with 256 random projections and an unbiased RBF MMD^2 with bandwidth 0.4. Additional experiments on sensitivity sweeps are given in Appendix C.

Table 1: Neural GMM optimization after 1000 steps (seed 0). Top: baseline and the controlled second-order correction. Bottom: directional controls at $r_{\max} = 20$.

Method	Final SWD ↓	Final MMD^2 ↓
Baseline and main method		
Drift	0.075433	0.000614
Controlled second-order ($r_{\max} = 5.0$)	0.060276	0.000554
Controlled second-order ($r_{\max} = 20.0$)	0.051145	0.000490
Directional controls ($r_{\max} = 20$)		
Rescaled-drift	0.070023	0.000732
Parallel-only	0.069391	0.000716
Orthogonal-only	0.056309	0.000578
Full controlled second-order	**0.051145**	**0.000490**

Neural optimization and trust-region sensitivity. Table 1 reports the final distances after 1000 steps. The top block compares ordinary drifting with the controlled second-order correction: the correction lowers both SWD and MMD^2, and $r_{\max} = 20$ gives the best final values. Figure 2 shows the corresponding 1000-step convergence curves for ordinary drifting and $r_{\max} \in \{1, 5, 10, 20, 50, 100\}$. Performance improves up to $r_{\max} = 20$, after which larger ratios begin to degrade the result, indicating that the correction is useful but should not be allowed to dominate the first-order drift. The wide sweep and the warmup ablation are reported in Appendix C.

Directional decomposition. The bottom block of Table 1 compares the controlled update with magnitude and directional controls at $r_{\max} = 20$. Rescaled drift and the parallel-only variant lower SWD relative to ordinary drift, but both increase MMD^2. The orthogonal component is substantially stronger than the parallel component, while the full correction is best on both metrics. This supports a directional contribution beyond scalar update amplification.

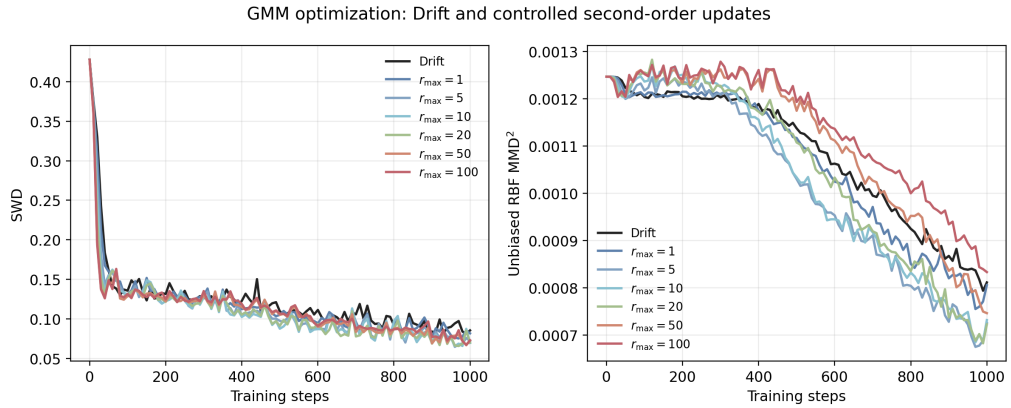


Figure 2: Neural GMM optimization. The two panels show SWD and unbiased RBF MMD^2 over 1000 steps for ordinary Drifting and controlled second-order updates with different $r_{\max}$. Metrics are recorded every 10 steps.

Third-order extension. We also investigated an optional projected third-order correction. We swept the third-order coefficient, the trust-region ratio, and the number of probes for third-order tensor estimation. Adding the third-order term on top of the second-order correction yields only marginal gains: the best setting improves SWD from 0.051 to about 0.049 and MMD^2 from 0.00049 to about 0.00048. However, computing the third-order term increases the per-step training time by about 30% relative to the second-order model. Given this limited benefit and the additional cost, we set $\alpha_3 = 0$ by default and use only the second-order correction in all experiments below.

5.2 MNIST

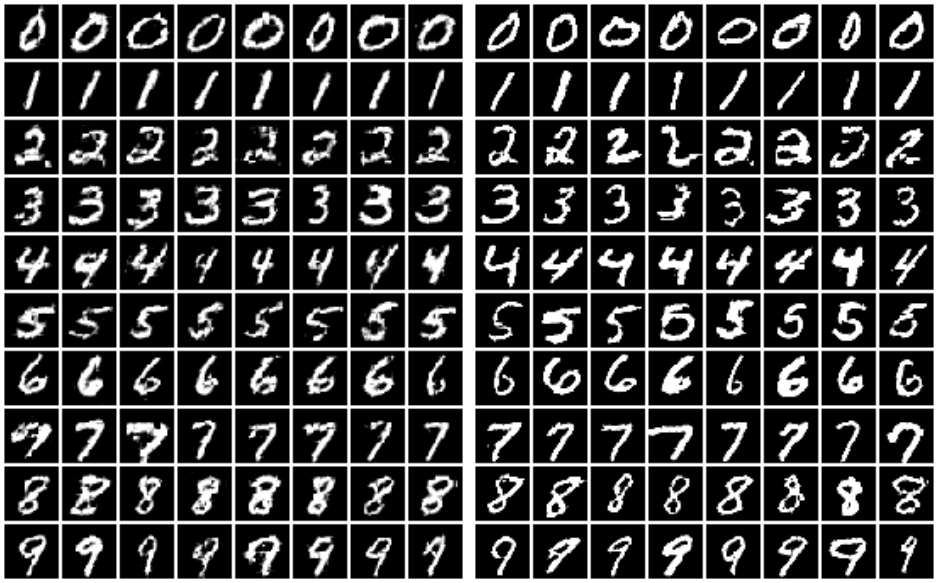
We evaluate ordinary Drifting and Higher-Order Drifting on MNIST image generation. MNIST (Lecun et al., 1998) consists of 28×28 grayscale handwritten digit images and is a standard benchmark for low-resolution image distributions. Both models use the same Diffusion Transformer (DiT) backbone (Peebles & Xie, 2023), the same training budget, and one-step inference (NFE $= 1$). At inference time, both models generate samples from Gaussian noise, and we evaluate generation quality using Fréchet Inception Distance (FID) (Heusel et al., 2017), where lower is better. Under this protocol, ordinary Drifting obtains FID 61.87, while Higher-Order Drifting reduces it to 52.89, corresponding to a 14.5% relative improvement. Because the number of runs is limited, we use FID as the primary metric and do not report Inception Score. Figure 3 shows uncurated one-step samples from both models using the same noise seed.

5.3 ImageNet

We further evaluate Higher-Order Drifting on ImageNet at 256×256 resolution. Because training a large ImageNet generator from scratch is outside our compute budget, we adopt a continuation protocol: we start from publicly available pretrained Drifting checkpoints and continue training under controlled settings. This measures the benefit of the higher-order correction during continued training and avoids confounding it with differences in pretraining.

We consider both pixel-space and latent-space generation with the B/16 and B/2 architecture. For the latent-space model, we follow the original Drifting evaluation protocol and use the SD-VAE latent space with a latent-MAE feature encoder, reporting FID on 50K generated images (Deng et al., 2026). For the pixel-space model, we use the standard ImageNet evaluation protocol. In both cases, all compared methods keep the pretrained generator, decoder (if any), feature encoder, data pipeline, and evaluation protocol fixed, and use one-step generation (NFE $= 1$).

For a fair comparison, we compare three settings: (i) the pretrained Drifting checkpoint without additional updates; (ii) the same checkpoint fine-tuned with the original Drifting loss under the official training configuration; (iii) the same checkpoint fine-tuned with our controlled second-order



(a) Ordinary Drifting. (b) Higher-Order Drifting.

Figure 3: Uncurated one-step MNIST samples from ordinary Drifting and Higher-Order Drifting. Both models use the same noise seed and a single generator evaluation (NFE $= 1$).

Table 2: ImageNet continuation from pretrained Drifting checkpoints. All methods use one-step generation and the same evaluation protocol. We run the evaluation five times and report the mean $\pm$ standard deviation.

Space	Method	FID ↓	IS ↑
Pixel	Pretrained Drifting B/16	1.73 ± 0.02	300.1 ± 1.0
	Official Drifting continuation	1.74 ± 0.03	303.2 ± 1.2
	Higher-Order Drifting continuation	1.64 ± 0.02	295.4 ± 1.2
Latent	Pretrained Drifting B/2	1.74 ± 0.03	263.4 ± 0.5
	Official Drifting continuation	1.73 ± 0.02	263.4 ± 0.4
	Higher-Order Drifting continuation	1.59 ± 0.03	287.9 ± 0.6

loss. All continuation runs use the same number of fine-tuning steps, optimizer, batch size, and data pipeline. The measured values are reported in Table 2.

Table 2 shows that continuing with the original Drifting loss leaves both pixel and latent FID nearly unchanged, suggesting that the first-order field is already saturated at the pretrained checkpoint. In contrast, Higher-Order Drifting improves FID from 1.74 to 1.64 in pixel space and from 1.73 to 1.59 in latent space. The latent model also improves IS from 263.4 to 287.9, indicating better sample quality and diversity. In pixel space, IS decreases slightly from 303.2 to 295.4 despite the FID gain, which we attribute to a trade-off between distribution matching and classifier confidence; qualitatively, the samples remain sharp and diverse (Figure 11). Overall, these results confirm that the second-order local covariance correction provides a complementary training signal at scale.

6 Conclusion

We proposed Higher-Order Drifting, which extends drifting by exploiting local response derivatives of the kernel transport field. These responses form a cumulant hierarchy, recover spatial derivatives of the smoothed log-density ratio for Gaussian kernels, and act as spectral reweighting under local linearization. Practical second-order and optional third-order corrections preserve one-step inference; the second-order term improves convergence and fidelity, while third-order gains are marginal at higher cost.

AI USE STATEMENT

In this work, we used generative AI tools for [”provide critical ingredients for proving mathematical claims”, ”assist in the writing of proofs”, ”design or provide feedback on research methodology or experiments”, ”implement methods”]. We have not used generative AI tools for [”help develop theoretical models or conceptual frameworks”, ”formulate mathematical claims”, ”propose or refine hypotheses”, ”assist with translation”, ”clean and reformat dataset”, ”support qualitative and thematic data analysis”, ”interpret results”.], and [”Generate synthetic data sets”] are not applicable to this work. Additionally, we used generative AI tools for [”create or edit software code”, ”draft parts of a research paper”, ”summarize or analyse existing literature”, ”sourcing/searching for information”, ”edit a research paper to improve readability”, ”identify relevant literature”]. We have reviewed all AI-assisted work.

REFERENCES

Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025.

Nicholas Boffi, Michael Albergo, and Eric Vanden-Eijnden. How to build a consistency model: Learning flow maps via self-distillation. *Advances in Neural Information Processing Systems*, 38: 33346–33382, 2026.

Nicholas M Boffi, Michael S Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *arXiv preprint arXiv:2406.07507*, 2024.

Drake Brown, Yuhao Huang, Shih-Hsin Wang, and Bao Wang. Second order drifting models. *arXiv preprint arXiv:2608.07924*, 2026.

Samir Darouich, Vinh Tong, Lluís Pastor-Pérez, Tanja Bien, Loay Mualem, and Mathias Niepert. Symdrift: One-shot generative modeling under symmetries. *arXiv preprint arXiv:2605.06140*, 2026.

Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. Generative modeling via drifting. *arXiv preprint arXiv:2602.04770*, 2026.

Maria Esteban-Casadevall, Jorge Carrasco-Pollo, Max Welling, Jan-Willem van de Meent, Erik J Bekkers, and Floor Eijkelboom. Kernel-gradient drifting models. *arXiv preprint arXiv:2605.10727*, 2026.

Ali Falahati, Elliot Creager, Gautam Kamath, and Shubhankar Mohapatra. Driftxpress: Faster drifting models via projected RKHS fields. *arXiv preprint arXiv:2605.12183*, 2026.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. volume 38, pp. 75460–75482, 2026.

Ping He, Om Khangaonkar, Hamed Pirsiavash, Yikun Bai, and Soheil Kolouri. Sinkhorn-drifting generative models. *arXiv preprint arXiv:2603.12366*, 2026.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6629–6640, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. volume 33, pp. 6840–6851, 2020.

Yuhao Huang, Shih-Hsin Wang, Andrea Bertozzi, and Bao Wang. Rmflow: Refined mean flow by a noise-injection step for multimodal generation. 2026:76267–76286, 2026.

Arkadii Kazanskii, Tatiana Petrova, Konstantin Bagrianskii, Aleksandr Puzikov, and Radu State. Attraction, repulsion, and friction: Introducing dmf, a friction-augmented drifting model. *arXiv preprint arXiv:2604.18194*, 2026.

Chieh-Hsin Lai, Bac Nguyen, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yuki Mitsufuji, Stefano Ermon, and Molei Tao. A unified view of drifting and score-based models. *arXiv e-prints*, pp. arXiv–2603, 2026.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

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

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

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182, 2023. doi: 10.1109/ICCV51070.2023.00387.

Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. Align your flow: Scaling continuous-time flow map distillation. *Advances in Neural Information Processing Systems*, 38:146459–146512, 2026.

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.

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *International Conference on Learning Representations*, volume 2024, pp. 15078–15097, 2024.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. 2020.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/song23a.html.

Erkan Turan, Nicolas Dufour, and Maks Ovsjanikov. Generative drifting is secretly score matching: a spectral and variational perspective. *arXiv preprint arXiv:2603.09936*, 2026.

Jiaru Zhang, Zeyun Deng, Juanwu Lu, Ziran Wang, and Ruqi Zhang. Analytical correction for subsampling bias in drifting models. *arXiv preprint arXiv:2604.27239*, 2026.

Bowen Zheng and Tianming Yang. Revisiting diffusion models: From generative pre-training to one-step generation. *arXiv preprint arXiv:2506.09376*, 2025.

Linqi Zhou, Stefano Ermon, and Jiaming Song. Inductive moment matching. *arXiv preprint arXiv:2503.07565*, 2025.

Huaisheng Zhu, Teng Xiao, Shijie Zhou, Zhimeng Guo, Hangfan Zhang, Siyuan Xu, and Vasant Honavar. Simple distillation for one-step diffusion models. *Advances in Neural Information Processing Systems*, 38:92071–92094, 2026.

A Proofs

A.1 Proof of the response–cumulant hierarchy

Fix an anchor a and a distribution r, and let $\mathbb{E}$ denote the normalized kernel-local expectation $\mathbb{E}_{a,r}$ from eq. (3). The log moment generating function of the centered local variable $Y - a$ is

$$\Psi(s) = \log \mathbb{E}[\exp(s^\top (Y - a))]. \tag{25}$$

The tilted local mean satisfies $\mu(s) - a = \nabla\Psi(s)$. By the defining property of a cumulant-generating function, $D^m\Psi(0) = \kappa_m$. Hence

$$D^j\mu(0) = D^{j+1}\Psi(0) = \kappa_{j+1}. \tag{26}$$

This proves eq. (9).

A.2 Proof of the Gaussian spatial hierarchy

Fix an anchor a and a distribution $r \in \{p, q\}$. Let

$$r_h(a) = \int r(y)\, G_h(a-y)\, dy = \mathbb{E}_{Y\sim r}[G_h(a-Y)]$$

denote the Gaussian-smoothed density. The tilted local log-partition function is

$$\Psi_{a,r}(s) = \log \frac{\mathbb{E}_{Y\sim r}\big[G_h(a-Y)\exp\big(s^\top(Y-a)\big)\big]}{\mathbb{E}_{Y\sim r}[G_h(a-Y)]}.$$

Using the Gaussian kernel identity

$$G_h(a-y)\exp\big(s^\top(y-a)\big) = \exp\left(\frac{h^2\|s\|_2^2}{2}\right) G_h(a+h^2s-y),$$

which follows by completing the square in the exponent, we obtain

$$\Psi_{a,r}(s) = \frac{h^2\|s\|_2^2}{2} + \log r_h(a+h^2s) - \log r_h(a).$$

Differentiating with respect to s gives

$$\nabla_s \Psi_{a,r}(s) = h^2 s + h^2 \nabla_a \log r_h(a+h^2s).$$

At $s = 0$, this recovers Tweedie's identity

$$\mu_{a,r} - a = h^2 \nabla_a \log r_h(a).$$

By definition of the cumulant-generating function, $D_s^m \Psi_{a,r}(0) = \kappa_{m,a,r}$. Since the first term $\frac{h^2\|s\|_2^2}{2}$ contributes only to the second derivative, we have

$$\kappa_{2,a,r} = h^2 I + h^4 \nabla_a^2 \log r_h(a),$$

and for $m \geq 3$,

$$\kappa_{m,a,r} = h^{2m} \nabla_a^m \log r_h(a).$$

Taking the difference between p and q, the additive $h^2 I$ term in κ_2 cancels, and for every $m \geq 2$ we obtain

$$\Delta\kappa_{m,a} = \kappa_{m,a,p} - \kappa_{m,a,q} = h^{2m} \nabla_a^m \log \frac{p_h(a)}{q_h(a)}.$$

This proves the claim.

A.3 Derivation of the second-order particle correction

For fixed normalized generated-side weights w_{ai}^q, let $e_{ai} = x_i - \mu_{a,q}$. Since $\sum_i w_{ai}^q e_{ai} = 0$, moving particle x_i while freezing the weights gives

$$dC_{a,q} = w_{ai}^q (dx_i e_{ai}^\top + e_{ai} dx_i^\top). \tag{27}$$

Let $\Delta C_a = C_{a,p} - C_{a,q}$. Using symmetry of ΔC_a in eq. (13),

$$\begin{aligned} d\mathcal{E}_2 &= -\frac{1}{2} \sum_a \frac{\langle \Delta C_a, dC_{a,q} \rangle}{\ell_a} \\ &= -\sum_a w_{ai}^q\, dx_i^\top \frac{\Delta C_a}{\ell_a} e_{ai}. \end{aligned} \tag{28}$$

Therefore $-\nabla_{x_i} \mathcal{E}_2$ is exactly the correction in eq. (14). The weights w_{ai}^q are frozen and stop-gradient during this differentiation.

A.4 Proof of the spectral reweighting

We record the full derivation of the linearized spectral reweighting. Throughout, we work in the same local-homogeneity approximation as Turan et al. (2026): the smoothed background density p_h is treated as constant on the scale of the perturbation, and $q = p + \delta$ with $|\delta| \ll p$.

Step 1: Linearized drift operator. For the Gaussian kernel, eq. (19) gives

$$V^{(1)}(x) = \mu_{a,p} - \mu_{a,q} = h^2 \nabla \log \frac{p_h}{q_h}(x). \tag{29}$$

Substituting $q = p + \delta$ and expanding to first order in δ,

$$V^{(1)}(x) = -h^2 \nabla \left(\frac{\delta_h}{p_h} \right)(x) + O(\delta^2), \tag{30}$$

where $\delta_h = \phi_h * \delta$. In Fourier space,

$$\widehat{V^{(1)}}(\xi) = -h^2 \, i\xi \, \frac{\hat{\delta}(\xi) e^{-h^2|\xi|^2/2}}{\hat{p}_h(\xi)}. \tag{31}$$

Under the local-homogeneity approximation $\hat{p}_h(\xi) \approx$ const, the continuity equation $\partial_t \delta = -\nabla \cdot (pV^{(1)})$ yields

$$\partial_t \hat{\delta}(\xi) = -h^2 |\xi|^2 e^{-h^2|\xi|^2/2} \hat{\delta}(\xi), \tag{32}$$

recovering eq. (23).

Step 2: Second-order correction. The continuous form of the second-order action is

$$G^{(2)}(x) = \int_a w(a,x) \, \frac{C_{a,p} - C_{a,q}}{\ell_a} \, (x - \mu_{a,q}) \, da, \qquad w(a,x) = \frac{k_h(a,x) q(x)}{\int_y k_h(a,y) q(y) \, dy}. \tag{33}$$

Using eq. (20) and the linearization $\log(p_h/q_h) \approx -\delta_h/p_h$,

$$C_{a,p} - C_{a,q} = -h^4 \nabla^2 \left(\frac{\delta_h}{p_h} \right)(a) + O(\delta^2). \tag{34}$$

Similarly, $\mu_{a,q} - \mu_{a,p} = V^{(1)}(a) \approx -h^2 \nabla (\delta_h/p_h)(a)$, and in the local-homogeneity approximation $\mu_{a,p} \approx a$, so

$$x - \mu_{a,q} = x - a + h^2 \nabla \left(\frac{\delta_h}{p_h} \right)(a) + O(h^4). \tag{35}$$

Substituting and keeping only first-order terms in δ,

$$G^{(2),\mathrm{lin}}(x) = -\frac{h^4}{\ell} \int_u k_h(u) \, f(x+u) \, (x-a) \, du \Big|_{a=x+u}, \qquad f = \nabla^2 \left(\frac{\delta_h}{p_h} \right), \tag{36}$$

where we have used $p(x)/p_h(x) \approx 1$ and $\ell_a \approx \ell$ in the local-homogeneity approximation.

Taylor expanding $f(x+u) = f(x) + \nabla f(x) u + O(|u|^2)$ and using

$$\int_u k_h(u) \, du = 1, \qquad \int_u k_h(u) \, u \, du = 0, \qquad \int_u k_h(u) \, u^\alpha u^\beta \, du = h^2 \delta^{\alpha\beta}, \tag{37}$$

we obtain

$$\int_u k_h(u) \, u \, f(x+u) \, du = h^2 \, \nabla f(x) + O(h^4). \tag{38}$$

Hence

$$G^{(2),\mathrm{lin}}(x) = \frac{h^6}{\ell} \, \nabla \left(\nabla^2 \left(\frac{\delta_h}{p_h} \right) \right)(x) + O(h^8). \tag{39}$$

Contracting the matrix indices yields $\nabla \Delta (\delta_h/p_h)$, where Δ is the Laplacian. In Fourier space,

$$\widehat{G^{(2),\mathrm{lin}}}(\xi) = -\frac{h^6}{\ell} \, i\xi \, |\xi|^2 \, \frac{\hat{\delta}(\xi) e^{-h^2|\xi|^2/2}}{\hat{p}_h(\xi)}. \tag{40}$$

The continuity equation contributes one further factor of $i\xi$, so the effective contribution to the decay rate is

$$\lambda^{(2)}(\xi) \propto \frac{h^6}{\ell} \, |\xi|^4 \, e^{-h^2|\xi|^2/2}. \tag{41}$$

Comparing with $\lambda^{(1)}$,

$$\frac{\lambda^{(2)}(\xi)}{\lambda^{(1)}(\xi)} \propto \frac{h^4}{\ell} \, |\xi|^2 = c_2 \, (h|\xi|)^2, \qquad c_2 = \frac{h^2}{\ell}, \tag{42}$$

which is the $m = 2$ term in eq. (24).

Step 3: Higher orders. For general m, the Gaussian hierarchy gives $\Delta\kappa^{(m)} = h^{2m}\nabla^m \log(p_h/q_h)$, which contributes a factor $|\xi|^m$ in Fourier space. The displacement factor $(x - \mu_{a,q})$ contributes one factor of $|\xi|$ after integration against the kernel, and the continuity equation contributes a final factor of $|\xi|$. Combined with the kernel factor $e^{-h^2|\xi|^2/2}$, the m-th order contributes

$$\lambda^{(m)}(\xi) \propto c_m\, h^2|\xi|^2\, e^{-h^2|\xi|^2/2}\, (h|\xi|)^m, \tag{43}$$

i.e. a relative weight u^m with $u = h|\xi|$ with respect to $\lambda^{(1)}$. Summing over the enabled orders yields eq. (24).

Step 4: Qualitative consequences. In the transition region $u \sim 1$, $W_M(u) = 1 + \sum_{m\geq 2} c_m u^m > 1$ when $c_m > 0$, so $\lambda_{\text{eff}}(\xi) > \lambda_{\text{Gauss}}(\xi)$. As $u \to \infty$, the factor $e^{-u^2/2}$ dominates any polynomial $W_M(u)$, so $\lambda_{\text{eff}}/\lambda_{\text{Gauss}} \to 1$ and the high-frequency bottleneck of Turan et al. (2026) is unchanged. This proves proposition 3(i)–(ii).

B Finite-Sample Estimation

B.1 Self-contraction and independent folds

Let $\widehat{\Delta}_A = \Delta + b + \varepsilon_A$ with $\mathbb{E}\varepsilon_A = 0$. Expanding the square gives

$$\mathbb{E}\|\widehat{\Delta}_A\|^2 = \|\Delta + b\|^2 + \mathbb{E}\|\varepsilon_A\|^2. \tag{44}$$

For an independent copy B,

$$\begin{aligned}\mathbb{E}\langle\widehat{\Delta}_A, \widehat{\Delta}_B\rangle &= \|\Delta + b\|^2 + \mathbb{E}\langle\varepsilon_A, \varepsilon_B\rangle \\ &= \|\Delta + b\|^2,\end{aligned} \tag{45}$$

because conditional independence and zero means imply $\mathbb{E}\langle\varepsilon_A, \varepsilon_B\rangle = 0$.

B.2 Unbiased Local Moment Estimators

The independent-fold construction removes the self-contraction variance term. Exact finite-sample unbiasedness additionally requires unbiased estimators of the local statistics themselves. At the second and third orders used in this paper, the relevant cumulants coincide with the corresponding central moments. To illustrate an unbiased construction for these central moments, suppose we can draw iid samples $X_0, \ldots, X_m$ directly from a fixed normalized kernel-local neighborhood. Define

$$H_m(X_0, \ldots, X_m) = \bigotimes_{j=1}^{m}(X_0 - X_j). \tag{46}$$

Writing $Y_j = X_j - \mu$, every expansion term except $Y_0^{\otimes m}$ contains at least one independent centered variable Y_j, $j \geq 1$, appearing once; its expectation therefore vanishes. Hence

$$\mathbb{E}H_m(X_0, \ldots, X_m) = \mathbb{E}[(X_0 - \mu)^{\otimes m}] = M^{(m)}. \tag{47}$$

Averaging H_m over tuples of distinct sample indices gives a U-statistic estimator of the local central moment. In the present implementation, local samples are instead obtained through self-normalized kernel weights, so a ratio-estimation bias can remain. We therefore describe our practical estimator as independent-fold or variance-debiased rather than claiming exact finite-sample unbiasedness of the complete kernel pipeline.

B.3 Finite-Sample Estimation for Higher-Order Drifting

The population objectives in the main text do not specify how their discrepancies should be estimated from a minibatch. This distinction matters increasingly with order. If $\widehat{\Delta}_A^{(m)}$ is a noisy estimate of a population discrepancy, then the naive square $\|\widehat{\Delta}_A^{(m)}\|^2$ contains a contraction of the estimator noise with itself. It can therefore remain positive even when the population discrepancy is zero. Equation (44) formalizes this effect and shows why it can influence the stochastic training update.

To remove the dominant self-contraction term, we use two independent support folds, $\mathcal{A}$ and $\mathcal{B}$. The population objective in eq. (11) is unchanged; only its minibatch estimator is replaced by the cross-inner-product

$$\widehat{\mathcal{E}}_m^{\mathrm{ind}} = \frac{1}{4}\sum_a \frac{\langle \widehat{\Delta}_{a,\mathcal{A}}^{(m)}, \widehat{\Delta}_{a,\mathcal{B}}^{(m)} \rangle}{\ell_a^{m-1}}. \tag{48}$$

During training, fold $\mathcal{A}$ contains the action particles. We therefore use a one-sided form in which the discrepancy estimated from fold $\mathcal{B}$ is treated as a stop-gradient target,

$$\widetilde{\mathcal{E}}_m^{\mathcal{A}\leftarrow\mathcal{B}} = \frac{1}{2}\sum_a \frac{\langle \widehat{\Delta}_{a,\mathcal{A}}^{(m)}, \mathrm{sg}(\widehat{\Delta}_{a,\mathcal{B}}^{(m)}) \rangle}{\ell_a^{m-1}}. \tag{49}$$

When the fold estimators are unbiased, the factor $1/2$ makes the expected one-sided gradient match the gradient of the $1/4$ population squared discrepancy. With self-normalized kernel weights, a separate finite-sample ratio bias can remain; the independent-fold construction is intended to remove the self-contraction variance term rather than to make the complete estimator exactly unbiased.

For second order, the action-fold correction has the same geometry as eq. (14), but the covariance discrepancy is estimated from the independent support fold $\mathcal{B}$ around the same anchors:

$$\boxed{G_{i,\mathrm{ind}}^{(2)} = \sum_a w_{ai}^{q,\mathcal{A}} \frac{\widehat{C}_a^{p,\mathcal{B}} - \widehat{C}_a^{q,\mathcal{B}}}{\ell_a^{\mathcal{B}}} (x_i^{\mathcal{A}} - \mu_{a,q}^{\mathcal{A}}).} \tag{50}$$

Third order uses the analogous one-sided projected discrepancy from eqs. (15) and (49). Thus, independent folds are an estimator for Higher-Order Drifting, not its defining population object.

B.4 Implementation Details

Finite-batch Gaussian neighborhoods. The experiments use a Gaussian kernel with bandwidth $h = 0.4\,b$, where b is the mean anchor-to-data distance of the current batch. As in our drifting baseline, affinities are normalized on both anchor and support axes before each side is row-normalized to obtain conditional local statistics. Generated self-pairs are masked when the action samples also serve as support points.

Local scale. For order normalization we use

$$\ell_a = \max\left\{ \frac{\mathrm{tr}(C_{a,p} + C_{a,q})}{2d}, 10^{-5} \right\}. \tag{51}$$

This is an implementation scale normalization, not a separate theoretical source of higher-order information.

Base-field normalization. All compared methods share the same drifting implementation and normalization. The higher-order coefficients are calibrated once on an excluded seed so that the second- and third-order corrections have comparable small RMS relative to the base field. No order-specific schedule is used in the main experiment.

Third-order probes. We use independent Rademacher probe pairs (ξ, ζ) rather than a repeated probe (ξ, ξ). The independence ensures that the probe expectation of the squared contracted tensor equals the tensor Frobenius norm without extra fourth-moment cross terms.

C GMM Supplementary Experiments

C.1 Controlled correction without warmup

We additionally remove the warmup while retaining the controlled correction with $r_{\max} = 20$. Thus the cosine-gated second-order term is active at its full gated magnitude from the first update: $T_{\mathrm{warm}} = 0$. Table 3 compares this setting with ordinary Drifting.

Table 3: Controlled correction with $r_{\max} = 20$ and no warmup after 1000 steps.

Method	Final SWD ↓	Final MMD^2 ↓
Drift	0.075433	0.000614
Controlled, $r_{\max} = 20$, $T_{\text{warm}} = 0$	**0.057606**	**0.000511**

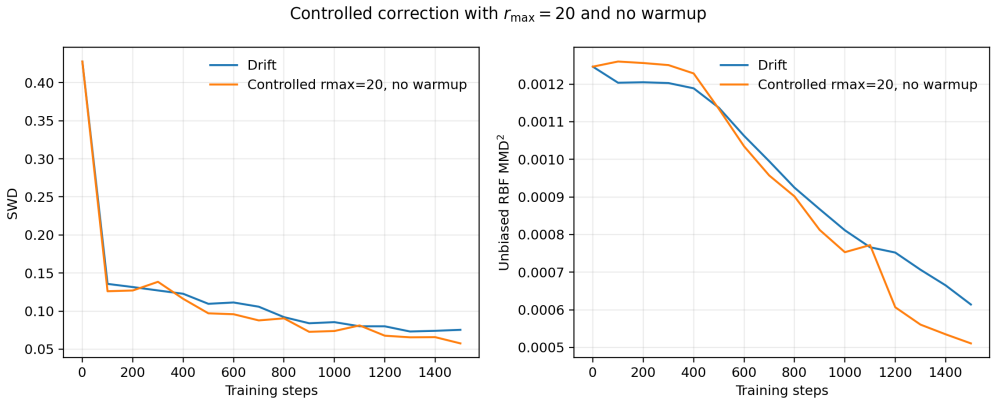


Figure 4: Drift versus controlled correction with $r_{\max} = 20$ and no warmup.

C.2 Fixed-weight second-order sensitivity

We also evaluate the unnormalized fixed-weight update $V = V + \alpha_2 G^{(2)}$ for $\alpha_2 \in \{0.1, 0.2, 0.5, 1.0\}$. These runs use no RMS normalization, cosine gate, warmup, or $r_{\max}$ clipping. The final values are shown in Table 4; complete trajectories are stored with the experiment artifacts.

Table 4: Fixed-weight second-order sensitivity after 1000 steps.

α_2	Final SWD ↓	Final MMD^2 ↓
0.1	0.058554	0.000507
0.2	0.055904	0.000511
0.5	**0.054083**	0.000446
1.0	0.055748	**0.000408**

C.3 Trust-region sensitivity and warmup ablation

Wide $r_{\max}$ sweep with warmup. We sweep $r_{\max} \in \{0, 0.01, 0.05, 0.1, 0.2, 0.5, 1, 2, 4, 5, 10, 20, 50, 100\}$ with the default 100-step warmup. The $r_{\max} = 0$ trajectory is the ordinary Drifting baseline and is labeled "Drift". Final values are reported in Table 5; the best final SWD is obtained at $r_{\max} = 20$ (0.051145), while very large ratios begin to degrade the result. Full trajectories are shown in Figure 6, and the final metric versus $r_{\max}$ is shown in Figure 7.

Warmup ablation. We repeat the wide sweep with no warmup, $T_{\text{warm}} = 0$, so that the gated correction is active from the first update. Final values are reported in Table 6. The best final SWD is obtained at $r_{\max} = 50$ (0.056838), and the no-warmup setting is consistently slightly worse than the default warmup schedule at the same $r_{\max}$. This supports keeping the short warmup.

No-warmup compact comparison. For completeness, Figure 10 shows the 1000-step compact comparison without warmup, matching the protocol of Figure 2 in the main text.

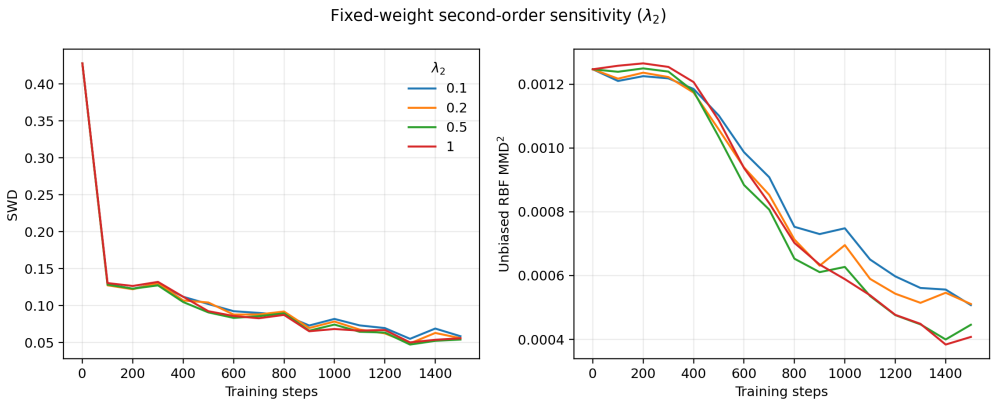


Figure 5: Fixed-weight second-order sensitivity to α_2.

Table 5: Wide $r_{\max}$ sweep with the default 100-step warmup after 1000 steps.

$r_{\max}$	SWD	MMD^2	$r_{\max}$	SWD	MMD^2
0 (Drift)	0.075433	0.000614	0.01	0.069124	0.000646
0.05	0.068976	0.000643	0.1	0.068807	0.000639
0.2	0.068522	0.000633	0.5	0.067759	0.000614
1	0.066800	0.000593	2	0.066086	0.000587
4	0.060276	0.000554	5	0.056687	0.000521
10	0.052080	0.000494	20	**0.051145**	**0.000490**
50	0.054445	0.000538	100	0.059959	0.000606

C.4 Preliminary third-order study

C.5 Third-order correction at the second-order optimum

We fix the second-order correction to the best setting identified above, `second_order_mode=safe` with $r_{\max} = 20$, and evaluate the independent-fold third-order update (`cross23`). All runs use 1000 steps, and evaluation every 10 steps.

Third-order coefficient. With two independent Rademacher probe pairs and the fixed third-order update, we sweep λ_3. The best final SWD is obtained at $\lambda_3 = 0.005$; the lowest final MMD^2 is obtained at $\lambda_3 = 0.01$.

Third-order trust-region ratio. We next use the safe third-order normalization and sweep its trust-region ratio. The second-order ratio remains fixed at 20, and the third-order term uses two probes. Ratios 10 and 20 give the strongest results in this sweep.

Number of probes. Finally, fixing $\lambda_3 = 0.001942$, we vary the number K of independent Rademacher probe pairs used by the projected third-order estimator. One probe has the best final SWD, while eight probes have the best final MMD^2 among the tested values.

D MNIST Supplementary Details and Experiments

D.1 Hyperparameters of Drifting Models

The MNIST configuration uses the `DriftDiT-Tiny` generator and operates directly in pixel space. The input and output resolution is 32×32 with one grayscale channel, and the number of classes is 10. The drifting field uses three squared bandwidth h^2,

$$\mathcal{T} = \{0.02, 0.05, 0.2\},$$

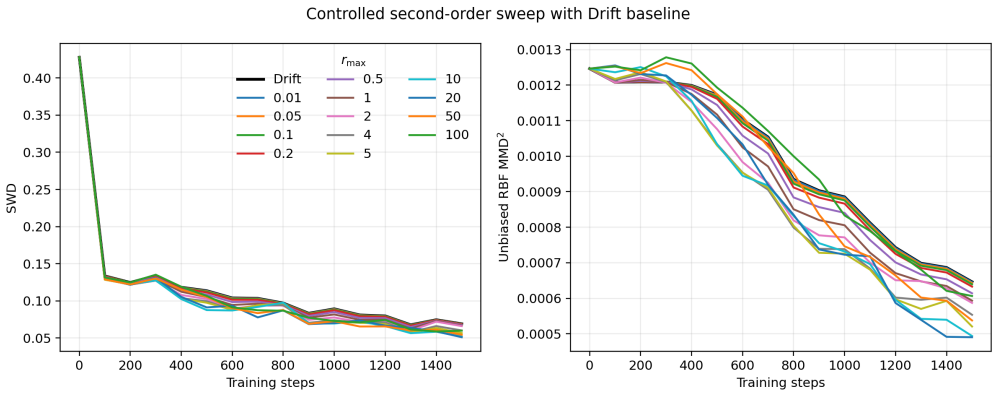


Figure 6: Wide controlled second-order sweep with the default 100-step warmup.

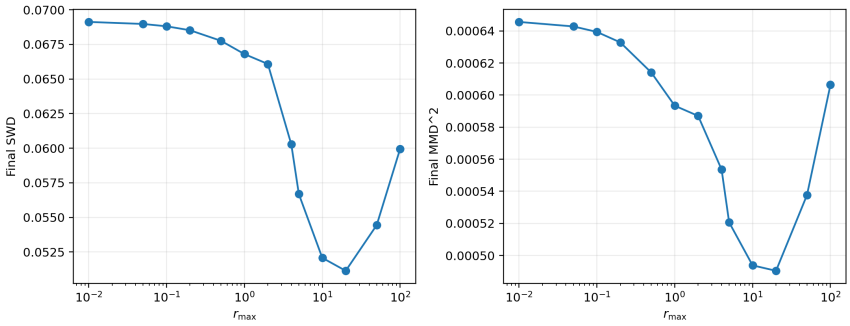


Figure 7: Final SWD and MMD^2 versus $r_{\max}$ with warmup.

whose normalized contributions are summed before forming the loss. For each optimization step, 32 generated samples and 32 positive samples are drawn for every class. Thus, the generated batch contains $10 \times 32 = 320$ samples, and the positive queue contributes $10 \times 32 = 320$ samples. The class-wise FIFO queue has capacity 128 samples per class.

The classifier-free guidance scale α is sampled independently from $\mathcal{U}(1.0, 3.0)$ during training. Class labels are dropped with probability 0.1. The learning rate is 2×10^{-4}, with linear warm-up for the first 1,000 optimization steps and a constant learning rate thereafter. AdamW uses $\beta_1 = 0.9$, $\beta_2 = 0.95$, and weight decay 0.01. The global gradient norm is clipped to 2.0. Model parameters are tracked with an exponential moving average (EMA) with decay 0.999. Training lasts for 100 epochs.

D.2 Model and Training Setup

D.2.1 Model Architecture

`DriftDiT-Tiny` is a one-step DiT-style generator. It receives Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$, a class label, and a guidance scale, and produces a $1 \times 32 \times 32$ image in one forward pass. Images are divided into 4×4 patches by a convolutional patch embedder, giving $8 \times 8 = 64$ patch tokens. The token width is 256, and the transformer contains 6 blocks with 4 attention heads and an MLP ratio of 4. Each block uses RMSNorm, QK-normalized self-attention, rotary position embeddings, and a SwiGLU MLP. Eight learned register tokens are prepended to the patch tokens and removed before unpatchification.

Conditioning is obtained by summing a class embedding, a Fourier-feature embedding of α, and a learned style embedding. The class embedding has an additional null class for classifier-free guidance. The style embedding sums 32 randomly selected tokens from a learnable codebook of size 64. Adaptive layer normalization with zero-initialized modulation provides the conditioning in every

Table 6: Wide $r_{\max}$ sweep without warmup after 1000 steps.

$r_{\max}$	SWD	MMD^2	$r_{\max}$	SWD	MMD^2
0.01	0.069366	0.000649	0.5	0.068937	0.000610
0.05	0.069350	0.000651	1	0.067138	0.000569
0.1	0.069155	0.000636	2	0.065368	0.000545
0.2	0.068217	0.000627	4	0.064187	0.000544
5	0.061957	0.000527	10	0.057795	0.000491
20	0.057606	0.000511	50	0.056838	0.000510
100	0.057621	0.000520			

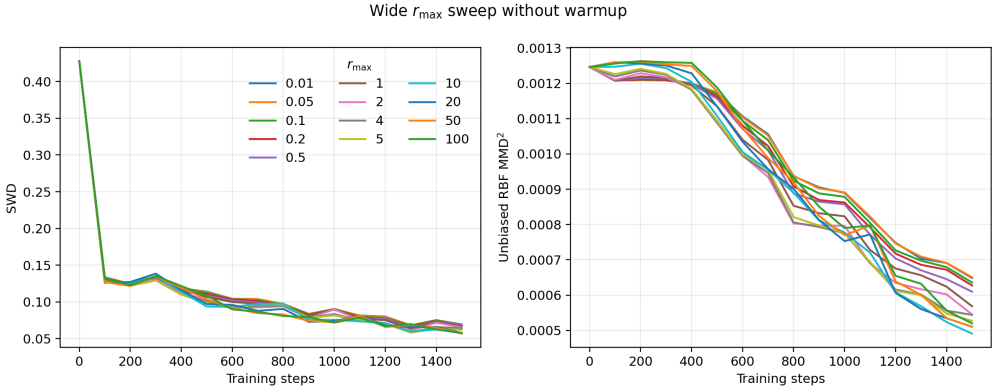


Figure 8: Wide $r_{\max}$ sweep with no warmup ($T_{\text{warm}} = 0$).

transformer block and in the final projection layer. The resulting model has approximately 5 million trainable parameters.

Local covariance estimation. In high-dimensional settings, empirical covariance estimates are unstable. To stabilize training, we do not use the full $D \times D$ covariance matrix. Instead, for image pixels (and spatial latents), we compute covariances only within a local $k \times k$ neighborhood of each spatial position, analogous to a convolution kernel, and set all remaining entries to zero. This induces a sparse, spatially local covariance and reduces the time and memory complexity from $O(D^2)$ to $O(Dk^2)$, where D is the feature dimension and k is the kernel size.

D.2.2 Training Setup

MNIST images are resized to 32×32, converted to tensors, and normalized channelwise with mean 0.5 and standard deviation 0.5, mapping pixel values to $[-1, 1]$. Training uses a batch size of 256, shuffles the data, and drops the final incomplete batch. With $60{,}000$ training examples, this gives $\lfloor 60{,}000/256 \rfloor = 234$ batches per epoch, or approximately $23{,}400$ optimization steps over 100 epochs. The kernel size k for local covariance estimation is 5. The second-order $r_{\max} = 1.0$.

At the beginning of each epoch, real examples are inserted into the class-conditional FIFO queue. Each training step samples class-balanced labels, Gaussian noise, and a random α, generates images, and samples positive examples from the queue. Because MNIST uses pixel space, the generated and positive images are flattened and ℓ_2-normalized; the multi-temperature drifting field is then computed within each class. The objective is the mean squared error between the normalized generated feature and a stop-gradient target formed by adding the drifting field. Gradients update the online model, followed by the EMA and learning-rate-scheduler updates.

D.2.3 Resource Usage and Time

We run all experiments on a single NVIDIA A100-40GB GPU. Each training run requires approximately 16 GiB of GPU memory and takes around 2 hours.

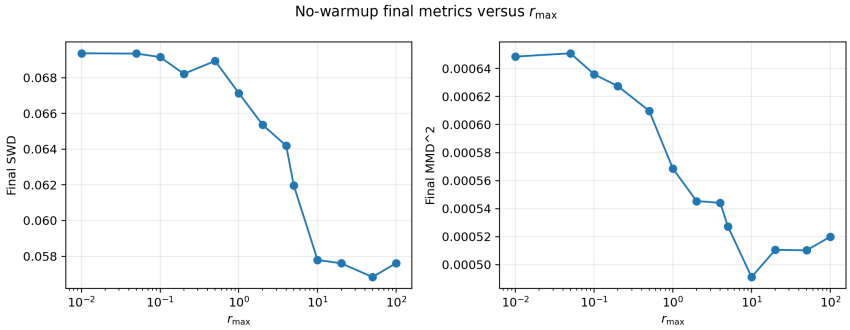


Figure 9: Final distances versus $r_{\max}$ without warmup.

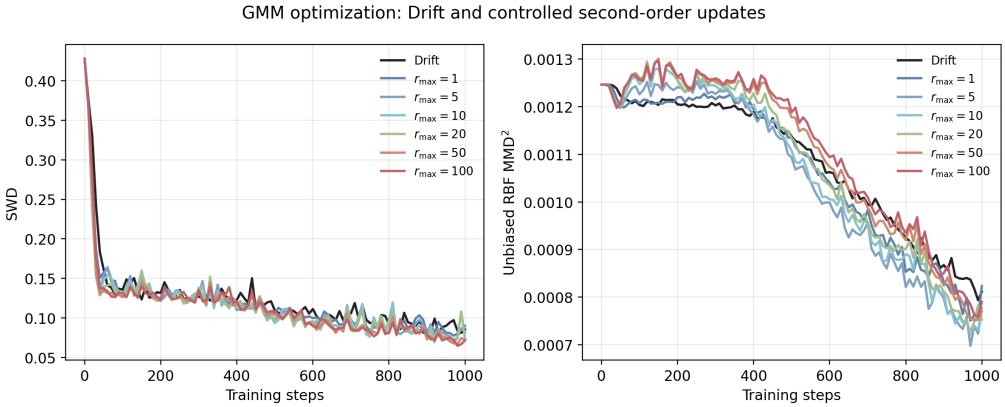


Figure 10: Compact 1000-step comparison without warmup ($T_{\text{warm}} = 0$), with measurements recorded every 10 steps.

E IMAGENET SUPPLEMENTARY DETAILS AND EXPERIMENTS

We fine-tune the official pretrained Latent-B generator released with the Drifting model on ImageNet rather than training a new network from scratch. The model uses the same DiT generator architecture and latent-space data flow as the original Drifting model. In particular, the number of layers, hidden size, attention configuration, class conditioning, noise conditioning, and feature model are unchanged. ImageNet images are precomputed in a latent cache at resolution 256 (four latent channels with spatial size 32×32), with 1000 class labels. The kernel size k for local covariance estimation is 7. The second-order $r_{\max} = 1.0$.

LoRA fine-tuning. To preserve the official model while adapting it to the new training run, we insert parameter-efficient LoRA adapters into the pretrained generator. The original weights are loaded and kept frozen; only the low-rank adapter parameters are optimized. We use rank $r = 16$, scaling factor $\alpha = 32$, and dropout 0. Adapters are added to the attention `qkv` and projection layers and to the three SwiGLU MLP projections `w1`, `w2`, and `w3`. Thus, for a linear layer with weight W, the adapted operation is

$$y = Wx + \frac{\alpha}{r} BAx,$$

where A and B are the trainable low-rank matrices and W remains fixed. At the end of training, the adapter update is merged into the base weights and the EMA version is exported as an inference-ready checkpoint.

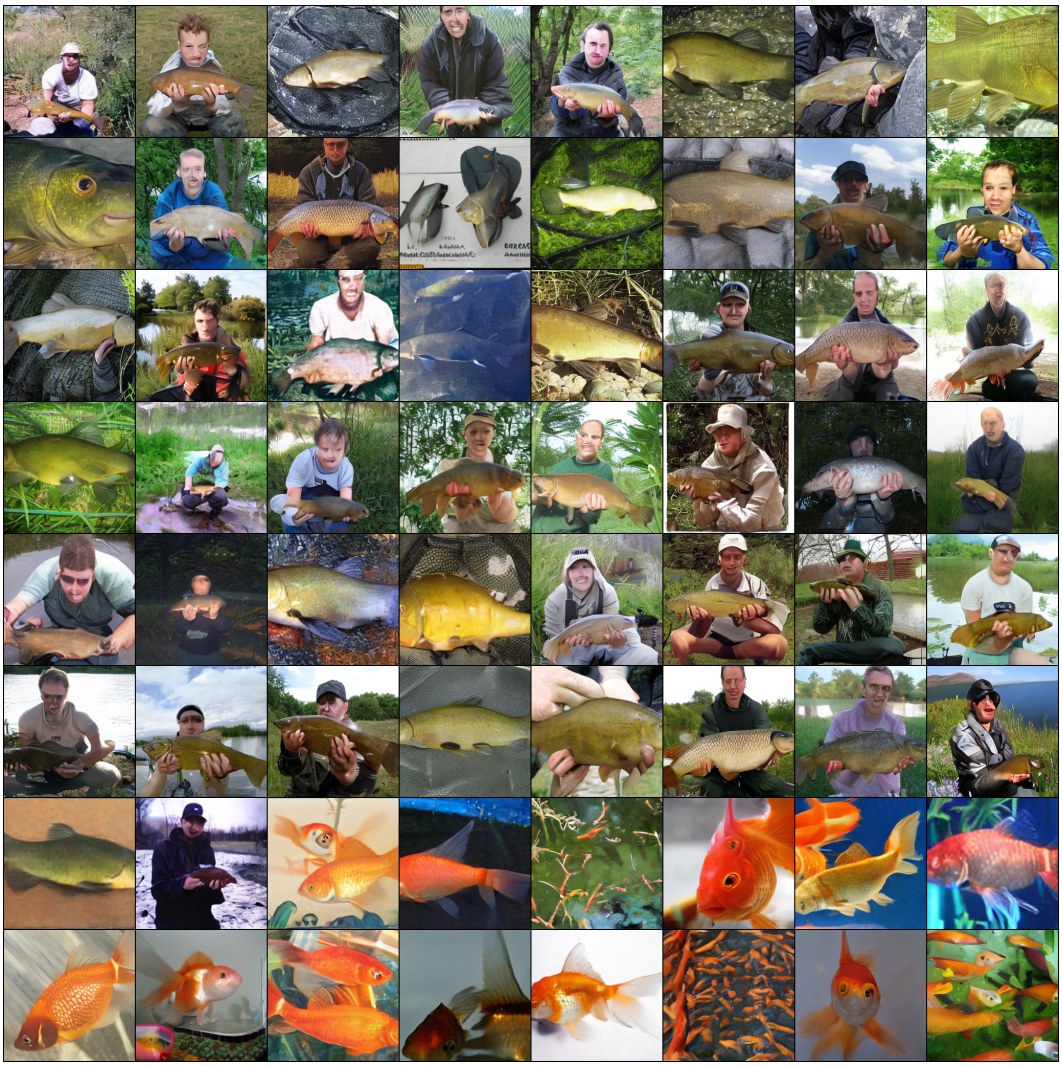


Figure 11: Samples from our latent-B/2 model with CFG = 1.0



Figure 12: Samples from our Pixel-B/16 model with CFG = 1.0

Table 7: Fixed third-order coefficient with second-order $r_{\max} = 20$.

λ_3	Final SWD ↓	Final MMD^2 ↓
0	0.052557	0.000508
0.0001	0.051395	0.000510
0.0002	0.050220	0.000511
0.0005	0.052693	0.000506
0.001	0.052768	0.000488
0.001942	0.053344	0.000487
0.003	0.050492	0.000451
0.005	**0.049135**	0.000436
0.01	0.054217	**0.000429**
0.02	0.055691	0.000494

Table 8: Safe third-order trust-region sweep with second-order $r_{\max} = 20$.

Third-order ratio	Final SWD ↓	Final MMD^2 ↓
0.01	0.052556	0.000508
0.05	0.052553	0.000508
0.1	0.052549	0.000507
0.2	0.052542	0.000507
0.5	0.052530	0.000506
1	0.052531	0.000504
2	0.052559	0.000500
5	0.051572	0.000478
10	**0.049462**	**0.000478**
20	0.050367	0.000452
50	0.052300	0.000513

Table 9: Probe-count sweep with second-order $r_{\max} = 20$.

K	Final SWD ↓	Final MMD^2 ↓
1	**0.051830**	0.000470
2	0.053344	0.000487
4	0.050929	0.000488
8	0.050715	**0.000458**
16	0.053648	0.000492
32	0.053961	0.000489
64	0.052018	0.000474