MIXED-FIELD MATCHING: TIME-CONDITIONED TRANSPORT WITH ENERGY-BASED REFINEMENT

Anonymous authors
Paper under double-blind review



Figure 1: **Mixed-Field Matching.** MFM performs time-conditioned transport when far from the data manifold, followed by time-independent refinement near the data. We show ImageNet-256 samples generated by MFM using DiT-XL/2 after 250 epochs of training.

ABSTRACT

We introduce **Mixed-Field Matching (MFM)**, which trains one network whose time input follows the noise level up to a threshold and stays fixed above it. Diffusion and flow-matching models give the network the noise level at every step of generation, while recent equilibrium models remove it entirely. We argue that the need for the noise level changes during generation. Far from the data, one noisy sample is consistent with many noise levels and many clean images, and the noise level reduces this uncertainty about where the sample should move. Close to the data, the manifold hypothesis suggests that the sample itself carries this information. MFM sampling first transports noise toward the data with the time-conditioned field and then refines each sample using a single time-independent field. A penalty that sets the field to zero on clean training examples keeps finished samples in place, so extra refinement steps do not degrade them. On a two-dimensional example where equilibrium matching (EqM) moves every sample onto the data but gives the modes the wrong proportions, MFM assigns samples to modes about as accurately as flow matching (FM). On ImageNet-256 with DiT-XL/2, at the same training and sampling cost as the baselines, MFM lowers FID by 15% relative to EqM and 28% relative to FM without guidance, and by 14% and 33% at a shared classifier-free guidance scale.

1 INTRODUCTION

Generative models aim to capture complex data distributions and synthesize new samples, with applications including image generation (Rombach et al., 2022), language modeling (Nie et al., 2025), and robot control (Chi et al., 2023). A wide range of approaches have been developed toward this goal, including autoregressive models, variational autoencoders (VAEs), generative adversarial networks (GANs), normalizing flows, and, more recently, diffusion and flow-matching models. While these

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.

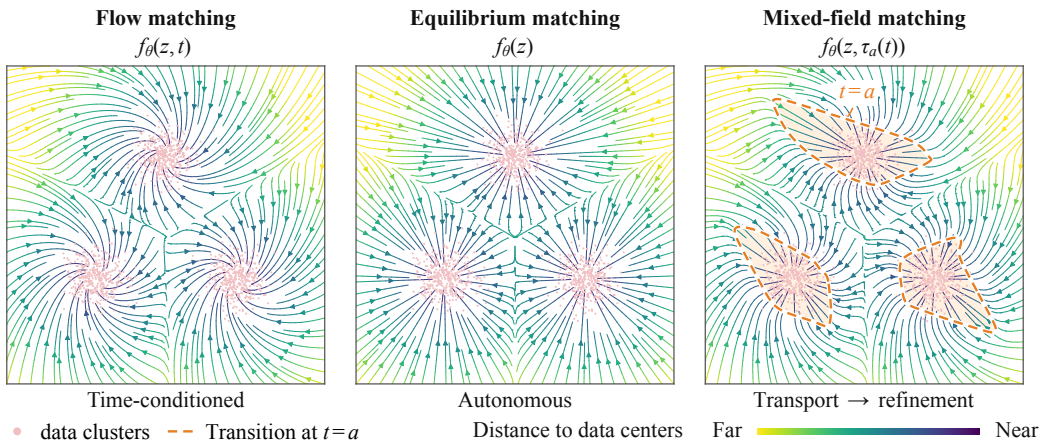


Figure 2: **Time-conditioned transport followed by autonomous refinement (schematic).** FM uses a time-conditioned transport field, whereas EqM uses one fixed field with no time input (Lipman et al., 2023; Wang & Du, 2025). Motivated by the manifold hypothesis, MFM keeps the time input during transport and switches to a fixed field for refinement near the data. Color indicates distance to the data centers. Orange contours and shading mark the transition at $t = a$. Straight local streamlines depict an ideal restoring field.

methods differ substantially in their parameterization and training objectives, they share the goal of learning a generative process that transforms a simple source distribution into the structure of the observed data.

Diffusion and flow-matching methods transform noise into data using denoisers or vector fields that are typically conditioned on the current noise level, or time t (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021b; Lipman et al., 2023). In contrast, recent research such as Equilibrium Matching (EqM) (Wang & Du, 2025) and Beckmann Transport Models (BTM) (Lee et al., 2026) shows that generation can also be achieved with time-independent or noise-unconditional fields (Sun et al., 2025; Sahraee-Ardakan et al., 2026). This raises a basic question: when does the network need t, and when can generation follow an *autonomous* field, one that depends only on the current sample?

Motivated by the changing role of noise conditioning during generation, we introduce Mixed-Field Matching (MFM). The manifold hypothesis suggests that high-dimensional data concentrate near a lower-dimensional manifold (Bengio et al., 2013; Fefferman et al., 2016). Far from that manifold, the noise level tells the network how far a sample still has to travel, which the sample alone leaves ambiguous. Near the manifold, the sample itself carries enough information for local refinement without noise conditioning, as analyses of denoising fields (Alain & Bengio, 2014) and of prediction without noise conditioning (Sun et al., 2025; Sahraee-Ardakan et al., 2026) suggest. MFM turns this transition into an explicit design choice.

Removing the time input everywhere has a cost. Far from the data, a noisy sample is consistent with many clean examples at many noise levels, and a field without a time input can only follow the average of their velocities. Such a field can still carry every sample onto the data, yet nothing ties the fraction of samples that reaches each region of the data to the probability of that region. On a simple distribution with modes of unequal probability, we show that EqM places samples on the modes but in the wrong proportions, while FM and MFM match them closely (Section 4.5). BTM restores the proportions within a fully time-independent model by changing EqM's training target (Lee et al., 2026). MFM instead keeps the time input where it resolves this ambiguity and removes it only near the data.

MFM implements this idea with one network and a threshold a on the time axis (Figure 2). For $t \leq a$ the network receives t and learns a flow-matching field that transports noise toward the data. For $t > a$ its time input stays at a, so it learns one time-independent field whose target points from each noisy sample to the clean example it came from. Sampling follows the first field up to $t = a$ and then refines the sample with repeated steps of the second. A penalty that sets the field to zero at clean

examples, the endpoints of the training paths, keeps finished samples in place under extra refinement steps. We call it the *endpoint penalty*.

Our contributions are:

- **Mixed-Field Matching.** Motivated by the manifold hypothesis, we introduce a single-network method with a controllable transition from time-conditioned transport to autonomous equilibrium refinement. We derive the exact cost of holding the time input fixed (Proposition 1) and show on an example with closed-form fields that it vanishes once a noisy sample identifies its clean example.
- **Stable refinement at no extra training cost.** Training the field to be zero on clean examples keeps finished samples in place under extra refinement steps. An in-batch estimator adds this penalty inside the ordinary batch, so training costs the same as FM or EqM.
- **Mode proportions and image quality.** MFM restores correct mode proportions in two dimensions, where EqM fails. On ImageNet-256 with DiT-XL/2, it has lower FID than FM and EqM at the same training and sampling cost, without guidance and at a shared guidance scale.

2 PRELIMINARIES

Flow matching. Diffusion and flow-based models generate samples by gradually transforming noise into data (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021b; Lipman et al., 2023; Liu et al., 2023; Albergo & Vanden-Eijnden, 2023). Flow matching learns a velocity field that determines how each sample moves during this process. The network receives the current state and its interpolation level, allowing the update to change as generation progresses. Sampling follows the predicted velocity from noise to data by solving a differential equation.

Training provides examples of these velocities through paths connecting noise to clean samples (Lipman et al., 2023). Let $x_1 \sim p_{\text{data}}$ be a clean example from the data distribution and $\epsilon \sim \mathcal{N}(0, I)$ be independent Gaussian noise. Their straight interpolation is

$$x_t = (1-t)\epsilon + tx_1, \qquad t \in [0,1]. \tag{1}$$

The path starts at noise when $t = 0$ and ends at data when $t = 1$. t is the time, or interpolation level; it sets both the signal and the noise in x_t. Its velocity is $x_1 - \epsilon$, giving the objective (Lipman et al., 2023)

$$\mathcal{L}_{\text{FM}} = \big(f(x_t, t) - (x_1 - \epsilon)\big)^2. \tag{2}$$

Here f is the network, the square of a vector is its squared Euclidean norm, and each loss is averaged over x_1, ϵ and t. Its optimal predictor $f^*(z, t)$ is the average velocity of all training paths that pass through the state z at time t.

Noise-unconditional flow matching. Removing the time input while keeping the same target gives $\mathcal{L}_{\text{NU}} = \big(f(x_t) - (x_1 - \epsilon)\big)^2$ (Sun et al., 2025; Wang & Du, 2025; Sahraee-Ardakan et al., 2026). Here "unconditional" means only that the network does not receive t. Its optimal predictor averages the velocities of all training paths that pass through the observed state, at any time, whereas the flow-matching predictor averages only those at the given time.

Equilibrium matching. EqM learns a time-independent equilibrium field with targets that vanish at the clean endpoint (Wang & Du, 2025). Writing the field in the direction of generation, its objective is $\mathcal{L}_{\text{EqM}} = \big(f(x_t) - c(t)(x_1 - \epsilon)\big)^2$, where $c(t) \geq 0$ controls the target magnitude and $c(1) = 0$. For example, linear decay uses $c(t) = 1 - t$. Starting from noise, sampling repeatedly applies $z_{k+1} = z_k + \eta f(z_k)$, where $\eta > 0$ is a step size. Like the noise-unconditional field, EqM's field is *autonomous* in the dynamical-systems sense. It depends on the current state and not on time. The two differ in the target, which EqM scales down to zero at the data.

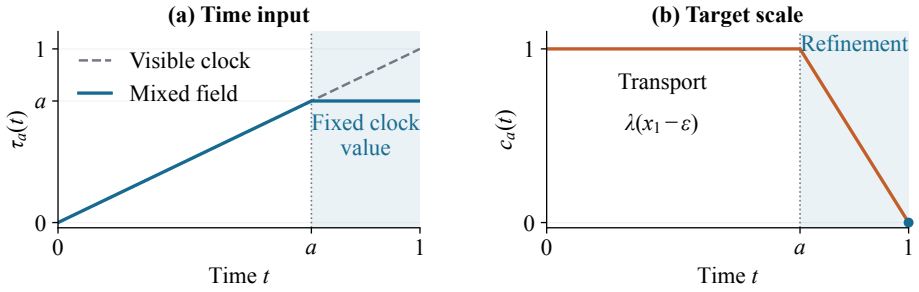


Figure 3: **MFM switches from transport to refinement at $t = a$.** (a) The time input stops at a. (b) The target scale falls to zero at the data. Shading marks the times $t > a$, which share one fixed field.

3 Mixed-field matching

The key idea of MFM is to perform time-conditioned transport when the sample is far from the data manifold, followed by autonomous equilibrium refinement once it is near a data mode.

To realize this idea, MFM learns a single vector field $f(x, \tau)$ whose time input τ is held fixed above a threshold a (Figure 3). For $t \le a$, $\tau = t$, and f is a flow-matching velocity field that transports noise toward the data. For $t > a$, $\tau = a$, which yields one time-independent field $f(\cdot, a)$ that points each noisy sample toward its clean example and vanishes at the data. Sampling therefore first integrates the time-conditioned field until $t = a$, and then repeatedly applies $f(\cdot, a)$ to refine the sample toward the data while leaving clean samples in place.

The remainder of this section defines the time input and training target (Section 3.1), the endpoint penalty that keeps clean samples in place. (Section 3.2). The sampling procedure (Section 3.3), and the conditions under which the time input can be safely held fixed (Section 3.4).

3.1 Time input and training target

Choose a threshold $0 \le a < 1$, the time at which generation switches from transport to refinement, and a target multiplier $\lambda > 0$. Define

$$\tau_a(t) = \min(t, a), \qquad c_a(t) = \begin{cases} 1, & 0 \le t \le a, \\ (1-t)/(1-a), & a < t \le 1. \end{cases} \tag{3}$$

The network sees the time input $\tau_a(t)$, which equals t up to a and stays at a afterwards. The target scale $c_a(t)$ is 1 up to a and then falls linearly to 0 at the data, so the target vanishes at clean samples; λ sets the overall target size. The MFM loss uses $\tau_a(t)$ as the input and the true t in the target,

$$\mathcal{L}_{\text{MFM}} = \big(f(x_t, \tau_a(t)) - \lambda c_a(t)(x_1 - \epsilon)\big)^2. \tag{4}$$

For $t \le a$ this is the flow-matching loss with the target scaled by λ. For $t > a$ the target becomes

$$\lambda c_a(t)(x_1 - \epsilon) = \frac{\lambda}{1-a}(x_1 - x_t), \tag{5}$$

the displacement from the noisy sample to its clean example times a constant, and the network receives the same input a for all of these examples.

Why the time input is held fixed. Above a, a network that still received the true t would learn a different field for each t, because the clean example that a noisy sample suggests can change with t. Holding the input at a pools all $t > a$ into one regression problem, whose optimum without the endpoint penalty is

$$f^*(z, a) = \frac{\lambda}{1-a}\big(\hat{x}_1(z) - z\big), \tag{6}$$

where $\hat{x}_1(z)$ is the average clean example that could have produced z at any time above a. The autonomous field therefore points from each sample toward one fixed estimate of its clean example. Section 3.4 explains what this pooling costs and when the cost vanishes.

EqM and FM as special cases. The target scale c_a is the truncated decay that EqM uses with its best cutoff (Wang & Du, 2025). EqM removes the time input at every noise level, so our EqM baseline is Equation (4) with a constant time input and no endpoint penalty. MFM differs from it in three ways. It gives the network t below a, it samples the times below a with an ODE, and it adds the endpoint penalty of Section 3.2. At $a = 1$ we set $\tau_1(t) = t$ and $c_1(t) = 1$, and with $\lambda = 1$ and no endpoint penalty Equation (4) is the FM loss (Appendix B.2).

3.2 Endpoint penalty and training cost

At $t = 1$ the noisy sample is the clean example itself and the target is zero, so the loss there is simply $(f(x_1, a))^2$. MFM adds this term with a weight w,

$$\mathcal{L} = \mathcal{L}_{\mathrm{MFM}} + w\,(f(x_1, a))^2, \tag{7}$$

and we call it the *endpoint penalty*. It trains the field to be zero at clean examples, so that refinement leaves finished samples in place. We use $w = 0.1$ for the main image runs. Training times are uniform on ImageNet and follow the log-normal noise distribution of Karras et al. (2022) on CIFAR-10 (Appendix C).

The penalty needs clean examples in every update. The simplest estimator evaluates a separate batch of clean examples next to the noisy batch, which doubles the network evaluations per update. A cheaper *in-batch* estimator replaces k of the B noisy examples in the batch by clean ones and reweights the two groups so that the penalty keeps its weight w, at the same cost as training FM or EqM. Each clean example replaces a noisy one, so k trades coverage of noise levels for supervision at the data, and we tune it per dataset. Appendix B.1 defines and compares the estimators.

3.3 Transport followed by refinement

Start from $z_0 \sim \mathcal{N}(0, I)$. Transport integrates

$$\frac{dz}{dt} = \frac{1}{\lambda} f(z, t), \qquad 0 \le t < a. \tag{8}$$

Dividing by λ restores the interpolation speed after scaling the training target. After reaching a, hold the time input at a and iterate

$$z_{k+1} = z_k + \eta f(z_k, a), \qquad \eta = \frac{r(1-a)}{\lambda}. \tag{9}$$

The parameter r is the step size relative to the target scale $\lambda/(1-a)$. For the ideal target $f(z, a) = \lambda(x_\star - z)/(1-a)$ toward a clean example $x_\star$, one step multiplies the distance to $x_\star$ by $1 - r$, so any $0 < r < 2$ converges. Our image experiments use $r = 1/2$. A learned field that is close to this ideal field near a clean example behaves the same way. Refinement started nearby stays nearby and ends within a distance set by the field's error at the clean example, which is exactly the error the endpoint penalty drives to zero (Proposition 2, Appendix A.3). Where samples converge does not decide how many converge to each mode. Section 4.5 measures this and Appendix A.4 analyzes it.

With T Euler transport steps and K refinement steps, sampling costs $T + K$ function evaluations, and Heun transport costs $2T + K$. Refinement is a fixed-point iteration on f, so it does not require f to be the gradient of an energy. Algorithm 1 in the appendix gives training and sampling pseudocode.

3.4 When the time input can be held fixed

The introduction asked when the network needs the time t. For MFM, the answer sets how low the threshold a can go. Above a, the network learns one prediction per noisy sample, the average of the best predictions for all times that could have produced it (Equation (6)). Holding the time input fixed costs nothing when these predictions agree, and Appendix A.1 makes this exact.

Proposition 1 (Cost of a fixed time input, informal)**.** *The lowest training error reachable with the time input $\tau_a(t)$ exceeds the lowest error reachable with the true time t by exactly the mean squared spread of the time-specific optimal predictions around their average over the times compatible with the noisy sample.*

Table 1: **Image generation on ImageNet-256 with DiT-XL/2.** MFM has the lowest FID with and without guidance, and its in-batch estimator reaches it at the training cost of FM and EqM. Training cost counts network evaluations per training update relative to FM and EqM. MFM computes the endpoint penalty on a separate batch or on k examples of the batch (in-batch). The last two columns use each method's best tested guidance scale. Bold marks the best value in each column. Appendix C gives the protocol and Table 4 all metrics.

		No guidance				$g = 1.5$	Best tested g	
Method	Training cost	FID ↓	sFID ↓	Prec. ↑	Rec. ↑	FID ↓	g	FID ↓
FM (Lipman et al., 2023)	1×	17.44	5.09	0.647	0.629	5.301	1.75	4.650
EqM (Wang & Du, 2025)	1×	14.92	6.70	0.651	**0.646**	4.111	1.75	3.736
MFM (ours), separate batch	2×	12.74	**4.88**	**0.672**	0.640	3.781	1.5	3.781
MFM (ours), in-batch ($k = 4$)	1×	**12.64**	4.99	0.667	0.643	**3.532**	1.5	**3.532**

Above a, the time-specific predictions differ only through the clean example they expect, so they agree in two cases. In the first, the noisy sample identifies its clean example. In the second, it identifies its time. The manifold hypothesis makes the second case likely in high dimension. When the data occupy few directions, most of the noise lies in directions orthogonal to the data, and its size there reveals the time (Sun et al., 2025). Appendix A.2 quantifies this and checks it on two moons embedded in $\mathbb{R}^d$, where the optimal fields are exact. There, changing only the time input above a changes the field by up to 100% in two dimensions and not at all once d is large enough for a noisy sample to identify its clean example, and lower thresholds need higher dimension.

The threshold should therefore sit where noisy samples already identify their clean example or their time. EqM removes the time input everywhere and pays this cost far from the data, where a sample identifies neither. Section 4.5 shows the consequence in two dimensions, where EqM reaches the modes in the wrong proportions. Because the right a depends on the data, we choose it by sample quality (Section 4.4).

4 Experiments

We ask whether MFM improves image generation, whether refinement tolerates additional steps, and how the endpoint penalty and the threshold affect these outcomes. All MFM results hold the time input fixed above a.

4.1 ImageNet-256

Protocol. We train class-conditional DiT models (Peebles & Xie, 2023) at two sizes, B/2 and XL/2, in the $4 \times 32 \times 32$ latent space of a pretrained autoencoder (Rombach et al., 2022). FM, EqM and MFM share the backbone and the 80-epoch recipe of Wang & Du (2025), and every sampler uses 250 function evaluations. Our EqM implementation matches the official code in float64 and reproduces its published B/2 FID within 0.23 (Appendix C). We report FID on 50,000 samples with the ADM evaluator (Dhariwal & Nichol, 2021; Heusel et al., 2017), and list sFID (FID on spatial features), Inception Score (Salimans et al., 2016), and precision and recall (Kynkäänniemi et al., 2019) in the tables. The guidance scale g of classifier-free guidance (Ho & Salimans, 2022) is 1 for no guidance.

Results. Table 1 reports DiT-XL/2. Without guidance and at the shared guidance scale, both MFM variants have lower FID than EqM and FM, and the in-batch variant does so at the baselines' training cost. Training the in-batch model on to 200 epochs lowers its unguided FID from 12.64 to 8.796, below the 9.62 that DiT-XL/2 reaches after 1400 epochs (Peebles & Xie, 2023) and close to the 8.3 of SiT-XL/2 at that length (Ma et al., 2024) (Appendix C).

The B/2 results in Appendix Table 5 show the same pattern. There we trained EqM and FM twice each (Table 7), and the margins of MFM without guidance and at $g = 1.5$ exceed the change between training runs, while the best-guidance differences are about as large as that change. Without the endpoint penalty, MFM differs from our EqM baseline only in the time input below a and in the

Table 2: **Image generation on CIFAR-10 at matched sampling budgets.** MFM has the lowest mean FID at both budgets, and noise-unconditional FM comes close. FID on 50,000 samples, mean $\pm$ sample standard deviation over three training seeds, or one seed where marked †, with one training recipe for every model. MFM uses $a = 0.9$ and $w = 0.1$. Bold marks the best mean in each column.

Method	Endpoint penalty	250 evaluations ↓	700 evaluations ↓
FM (Lipman et al., 2023), Euler	None	3.044 ± 0.081	2.854 ± 0.066
Noise-unconditional FM (Sun et al., 2025), Euler	None	$2.760^\dagger$	$2.738^\dagger$
EqM (Wang & Du, 2025), GD	None	3.490 ± 0.410	4.081 ± 0.938
MFM (ours), $\lambda = 4$	Separate batch	2.744 ± 0.035	2.664 ± 0.042
MFM (ours), $\lambda = 1$	Separate batch	2.678 ± 0.053	$\mathbf{2.653 \pm 0.067}$
MFM (ours), $\lambda = 1$	In-batch, $k = 32$	2.651 ± 0.051	2.733 ± 0.019
MFM (ours), $\lambda = 1$	In-batch, $k = 64$	$\mathbf{2.527 \pm 0.020}$	2.654 ± 0.106

MFM splits the 250 and 700 evaluations into $200 + 50$ and $500 + 200$ Euler transport and refinement steps. EqM uses gradient descent (GD) with step sizes 0.002 and 0.001. The in-batch estimator uses k of the 128 examples per device as clean examples.

	Inpainting, FID ↓			
Method	Random	Box	Half	Expand
FM (Lipman et al., 2023)	**1.544**	4.468	5.973	17.236
EqM (Wang & Du, 2025)	1.787	4.766	5.693	16.010
MFM (ours)	1.897	**4.417**	**5.294**	**13.757**

	Composition, FID ↓		
Method	Parent A	Parent B	Composed
EqM	62.930	**30.756**	29.625
MFM (ours)	**62.761**	30.857	**23.556**

Table 3: **Inpainting and composition on CIFAR-10.** MFM's inpainting lead grows with the hidden fraction, and adding its fields improves most over the parents. Left: FID of the filled region for masks that hide 25% to 75% of each image, each method at its best noise-ladder setting at 200 function evaluations. Right: FID of two models trained on different class subsets and of the sum of their fields, scored against the classes the subsets share. Bold marks the best value in each column. Appendix C.3 gives both protocols.

sampler, and it still lowers FID. The penalty mainly improves sFID and makes extra refinement safe (Section 4.4).

4.2 CIFAR-10

Table 2 compares methods on CIFAR-10 at exactly matched sampling budgets, using one training recipe and three training seeds per row unless marked. MFM has lower mean FID than FM and EqM at both budgets. Noise-unconditional FM (Sun et al., 2025), which has no time input and uses the same recipe, comes within 0.02 FID of MFM with $\lambda = 4$ at 250 evaluations, so at this scale keeping the time input far from the data adds little over removing it everywhere. The remaining lead comes from MFM with $\lambda = 1$ and the in-batch estimator at $k = 64$. It is best at 250 evaluations, matches the separate batch at 700, and trains in about half the time (Appendix Table 8). Unlike ImageNet, CIFAR-10 prefers $\lambda = 1$ (Appendix C).

4.3 Inpainting and composition

A time-independent field near the data has two uses beyond generation. Table 3 tests both on CIFAR-10. For inpainting, every method fills the hidden part of a test image with the same noise-ladder procedure at the same cost, and MFM's lead grows with the hidden fraction: FM is best when 25% of the pixels are missing at random, the two are tied on a 25% box, and MFM leads by 0.4 FID when half the image is hidden and by 2.3 when three quarters are (Figure 4). For composition, adding the fields of two models trained on different class subsets concentrates samples on the classes the subsets share. Both EqM and MFM beat their better parent this way, MFM by 7.3 FID and EqM by 1.1. Appendix C.3 gives the protocol and the limits of both experiments. Figure 5 shows the same operation on ImageNet-256 with the DiT-XL/2 models.

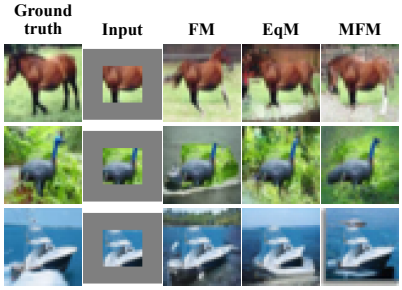


Figure 4: **Inpainting on CIFAR-10.** Given the central quarter, FM, EqM and MFM fill the rest with the same noise ladder at 200 evaluations. Selected examples where MFM's fill is cleanest.



Figure 5: **Two-class compositions on ImageNet-256.** EqM and MFM DiT-XL/2 models add the fields of two class labels, with guidance 1.5 and 250 composite-field evaluations, from the same noise in each column. Selected examples.

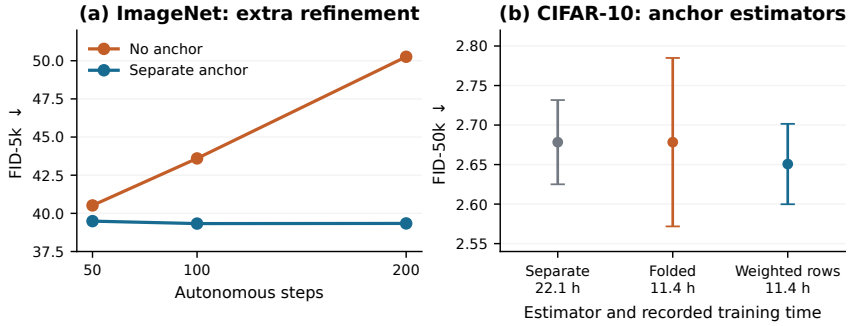


Figure 6: **Endpoint supervision and its cost.** (a) ImageNet B/2, $a = 0.8$. Transport is fixed at 230 Euler steps while refinement increases. With the endpoint penalty ($w = 0.1$), FID stays nearly constant. Values are FID-5k from a PyTorch evaluator (Appendix C). (b) CIFAR-10, $a = 0.9$, $\lambda = 1$, 250 and 700 function evaluations. Mean and sample standard deviation over three training seeds. The in-batch estimator trains in about half the time of the separate batch, and $k = 64$ has the lowest FID at 250 evaluations.

4.4 Ablations: time input, endpoint penalty, and threshold

Extra refinement. Figure 6a holds ImageNet transport fixed and raises the number of refinement steps from 50 to 200. With the endpoint penalty, FID stays flat, and without it FID rises steadily, so the result does not depend on when refinement stops, which matters for a fixed-point sampler that has no natural end time. On CIFAR-10, a model that receives the true time and has no endpoint penalty gets an order of magnitude worse as refinement continues, while MFM stays flat (Appendix Table 12). Figure 11 in the appendix shows samples at these budgets.

Choosing the threshold. A lower threshold makes the time-independent phase longer, and the best value depends on the data. On ImageNet B/2, $a = 0.7$ gives the best FID without guidance but worse sFID with guidance, so we use $a = 0.8$ (Table 10). CIFAR-10 prefers $a = 0.9$. In two dimensions, $a \geq 0.9$ matches FM while $a = 0.5$ fails on three of four targets (Table 11). This agrees with Section 3.4. A threshold set too low drops the time input before the noisy state identifies its clean example.

4.5 Mode proportions on a two-dimensional example

The target is five points on a regular pentagon in $\mathbb{R}^2$ with probabilities $(0.50, 0.20, 0.15, 0.10, 0.05)$. Two quantities matter. The total variation (TV) distance between the fraction of samples nearest to each point and its probability tells whether the modes receive the right share, and the root-mean-

square (RMS) distance to the nearest point tells whether samples reach the modes at all. At radius 0.5 and $a = 0.8$, the exact optimal EqM and MFM fields, integrated numerically, both bring samples within 2×10^{-9} of the points, yet their TV distances are 0.288 and 0.011. The gap survives training a neural network with the same endpoint penalty for both (Figure 12 and Table 14 in Appendix D.3). Every method reaches the modes, but EqM sends too many samples to the most probable mode and almost none to the least probable ones, while MFM matches the target proportions about as closely as FM and more closely than BTM. For the population fields the gap becomes small when the modes are farther apart, and MFM gives wrong proportions when it switches early ($a = 0.5$). Appendix D.3 reports the full grid over mode separation and threshold, controls without the endpoint penalty, and numerical checks.

5 Related Work

Flow matching and interpolant models learn transport fields that receive the time as input (Lipman et al., 2023; Liu et al., 2023; Albergo & Vanden-Eijnden, 2023; Ma et al., 2024). Removing the time input can still work when the noisy sample itself reveals enough about its noise level (Sun et al., 2025), and Sahraee-Ardakan et al. (2026) explain why by showing that a field without time follows the gradient of an energy averaged over noise levels. MFM takes a middle road. It keeps the time input while a noisy sample is still compatible with many noise levels and many clean examples, and removes it near the data, where Proposition 1 shows that the input no longer changes the best prediction. EqM removes the time input everywhere, scales its target to vanish at the data, and samples with gradient descent (Wang & Du, 2025). MFM uses the same target near the data, but transports with the time input, holds it fixed only for refinement, and supervises the field at clean examples directly. Beckmann Transport Models keep a fully time-independent field and recover the correct mode probabilities by changing the training target so that the field transports the noise distribution exactly onto the data (Lee et al., 2026). MFM instead recovers them by keeping the time input during transport and regresses the refinement field directly, without that constraint.

Energy-based models learn a scalar energy that is low on the data (LeCun et al., 2006). Contrastive training lowers the energy of data and raises the energy of samples drawn from the model, which requires MCMC sampling during training (Hinton, 2002; Du & Mordatch, 2019; Nijkamp et al., 2019; Du et al., 2021). Diffusion recovery likelihood makes this sampling easier by learning energies at several noise levels (Gao et al., 2021). MFM needs neither negative samples nor MCMC during training, and it does not assume an energy. Its learned image fields have clearly non-symmetric Jacobians, so they are not gradients of a scalar energy (Appendix C). Energy Matching also connects transport and near-data equilibrium behavior (Balcerak et al., 2025). Its scalar-energy formulation and training procedure differ from our single vector regression objective.

6 Conclusion

MFM gives the network the noise level only where it changes the best prediction. It transports samples with a time-conditioned field and then refines them with one time-independent field that leaves clean samples in place. In two dimensions, this assigns samples to modes about as accurately as flow matching, where a field without time does not. On ImageNet-256 with DiT-XL/2, MFM lowers FID relative to FM and EqM at the same training and sampling cost, and its samples do not degrade under extra refinement steps. On CIFAR-10, FM without any time input comes close to MFM, so the value of the time input at small scale is limited. Testing that control at ImageNet scale is the most direct next step.

AI USE STATEMENT

The research questions, hypotheses, experimental design, and conclusions are the authors' own. Under the authors' direction, AI tools assisted in implementing parts of the analysis and plotting code, provided feedback during research ideation, assisted in proving mathematical claims formulated by the authors, and drafted and polished manuscript text from the authors' outlines and results. We did not use these tools to generate synthetic data sets or for literature retrieval. The authors verified all derivations manually, checked AI-assisted code against logged experiment records, traced every reported number to its run records, and take responsibility for the final content.

REPRODUCIBILITY STATEMENT

Algorithm 1 gives the training update and the sampler. Appendix C lists the architecture, optimization, data, sampler and evaluation settings for ImageNet-256 and CIFAR-10, the inpainting and composition protocols (Appendix C.3), and the two-dimensional experiments. Appendix B defines the three estimators of the endpoint penalty and the settings that recover FM and EqM. Appendix A states the assumptions and proofs behind Propositions 1 and 2. Appendix D reports every table behind the main text with means and standard deviations over training seeds where available, including repeated training runs of the baselines and of MFM (Tables 7 and 6), and Appendix D.3 gives the mode-proportion protocol and its numerical checks.

REFERENCES

Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data-generating distribution. *Journal of Machine Learning Research*, 15(110):3743–3773, 2014. URL https://jmlr.org/papers/v15/alain14a.html.

Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations (ICLR)*, 2023.

Michal Balcerak, Tamaz Amiranashvili, Antonio Terpin, Suprosanna Shit, Lea Bogensperger, Sebastian Kaltenbach, Petros Koumoutsakos, and Bjoern Menze. Energy matching: Unifying flow matching and energy-based models for generative modeling. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2504.10612.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013. URL https://arxiv.org/abs/1206.5538.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2023. URL https://arxiv.org/abs/2303.04137.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://arxiv.org/abs/2105.05233.

Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. In *Advances in Neural Information Processing Systems*, 2019.

Yilun Du, Shuang Li, Joshua Tenenbaum, and Igor Mordatch. Improved contrastive divergence training of energy-based models. In *International Conference on Machine Learning*, 2021.

Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049, 2016.

Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P. Kingma. Learning energy-based models by diffusion recovery likelihood. In *International Conference on Learning Representations*, 2021.

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 *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html.

Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020. URL https://arxiv.org/abs/2006.11239.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, 2022.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/0234c510bc6d908b28c70ff313743079-Abstract.html.

Yann LeCun, Sumit Chopra, Raia Hadsell, Marc'Aurelio Ranzato, and Fu Jie Huang. A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press, 2006.

Cheuk Kit Lee, Florentin Coeurdoux, Peter Potaptchik, Yilun Du, Michael Albergo, and Eric Vanden-Eijnden. Beckmann transport models: From autonomous flows to one-step maps, 2026. URL https://arxiv.org/abs/2608.01692.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023.

Nanye Ma, Mark Goldstein, Michael S. Albergo, Nicholas M. Boffi, Eric Vanden-Eijnden, and Saining Xie. SiT: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, 2024. URL https://arxiv.org/abs/2401.08740.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2502.09992.

Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. Learning non-convergent non-persistent short-run MCMC toward energy-based model. In *Advances in Neural Information Processing Systems*, 2019.

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. URL https://openaccess.thecvf.com/content/ICCV2023/html/Peebles_Scalable_Diffusion_Models_with_Transformers_ICCV_2023_paper.html.

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. URL https://arxiv.org/abs/2112.10752.

Mojtaba Sahraee-Ardakan, Mauricio Delbracio, and Peyman Milanfar. The geometry of noise: Why diffusion models don't need noise conditioning. *arXiv preprint arXiv:2602.18428*, 2026.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, volume 29, 2016. URL https://arxiv.org/abs/1606.03498.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b. URL https://arxiv.org/abs/2011.13456.

Qiao Sun, Zhicheng Jiang, Hanhong Zhao, and Kaiming He. Is noise conditioning necessary for denoising generative models? In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 57469–57502. PMLR, 2025. URL https://proceedings.mlr.press/v267/sun25g.html.

Runqian Wang and Yilun Du. Equilibrium matching: Generative modeling with implicit energy-based models. *arXiv preprint arXiv:2510.02300*, 2025. URL https://arxiv.org/abs/2510.02300.

A PROOFS AND ANALYSIS

A.1 THE COST OF A FIXED TIME INPUT

This section states Proposition 1 formally, proves it, and bounds the cost it describes. Throughout, the threshold $a < 1$ is fixed, the training time $T \sim q$ has a density, $Z = x_T$ is the noisy point, and $Y = \lambda c_a(T)(x_1 - \epsilon)$ is the regression target of Equation (4), with $\mathbb{E}\|Y\|^2 < \infty$.

Formal statement. Let $\mathcal{R}_{\text{full}}$ and $\mathcal{R}_a$ be the lowest mean squared errors that any predictor of Y can reach from (Z, T) and from $(Z, \tau_a(T))$. Then

$$\mathcal{R}_a - \mathcal{R}_{\text{full}} = \mathbb{E}\big\|\mathbb{E}[Y \mid Z, T] - \mathbb{E}[Y \mid Z, \tau_a(T)]\big\|^2. \tag{10}$$

The first conditional mean is the best prediction with the true time and the second is the best prediction with the time input of MFM. They agree for $T \le a$. For $T > a$ the second averages the first over the times compatible with Z, so the right side is the spread of the time-specific predictions around their average, as Proposition 1 states.

Proof. The best predictors of Y from the true time and from the time input $\tau_a(T)$ are the conditional means

$$g(Z, T) = \mathbb{E}[Y \mid Z, T], \qquad \bar{g}\big(Z, \tau_a(T)\big) = \mathbb{E}\big[Y \mid Z, \tau_a(T)\big].$$

Since $(Z, \tau_a(T))$ is a function of (Z, T), the residual $Y - g(Z, T)$ is orthogonal to every square-integrable function of (Z, T), including $g - \bar{g}$. Hence

$$\mathbb{E}\big\|Y - \bar{g}\big(Z, \tau_a(T)\big)\big\|^2 = \mathbb{E}\|Y - g(Z, T)\|^2 + \mathbb{E}\big\|g(Z, T) - \bar{g}\big(Z, \tau_a(T)\big)\big\|^2. \tag{11}$$

Conditional means minimize the mean squared error, so the left side is $\mathcal{R}_a$ and the first term on the right is $\mathcal{R}_{\text{full}}$, which proves Equation (10). The argument uses only that the time input is a function of T. For the constant time input of our EqM baseline, it gives the cost $\mathbb{E}\|\mathbb{E}[Y \mid Z, T] - \mathbb{E}[Y \mid Z]\|^2$, which, unlike the cost of MFM, also includes the times $T \le a$.

The cost above the threshold. For $T \le a$ the time input equals T, so g and $\bar{g}$ agree. For $T > a$, Equation (5) gives $Y = \kappa(x_1 - Z)$ with $\kappa = \lambda/(1 - a)$. With the posterior means of the clean example

$$m(z, t) = \mathbb{E}[x_1 \mid Z = z,\ T = t], \qquad m_a(z) = \mathbb{E}[x_1 \mid Z = z,\ T > a],$$

the two predictors above the threshold are

$$g(z, t) = \kappa\big(m(z, t) - z\big), \qquad \bar{g}(z, a) = \kappa\big(m_a(z) - z\big), \tag{12}$$

and $\bar{g}(\cdot, a)$ is the field of Equation (6). The cost is therefore

$$\mathcal{R}_a - \mathcal{R}_{\text{full}} = \kappa^2\, \mathbb{E}\big[\mathbf{1}\{T > a\}\, \|m(Z, T) - m_a(Z)\|^2\big]. \tag{13}$$

Since $m_a(z)$ is the average of $m(z, T)$ over the posterior of T given $Z = z$ and $T > a$, the cost vanishes exactly when the posterior mean $m(z, t)$ is the same at all times above a that are compatible with the noisy point.

A bound. Suppose $m(z, t)$ is $L(z)$-Lipschitz in t over the times $t > a$ compatible with z. Given $Z = z$, let S and S' be independent draws from the posterior of T given $Z = z$ and $T > a$. Then $m_a(z) = \mathbb{E}\, m(z, S)$, and the identity $\mathbb{E}\|V - \mathbb{E}V\|^2 = \frac{1}{2}\mathbb{E}\|V - V'\|^2$ for independent copies V and V' gives

$$\begin{aligned} \mathbb{E}\|m(z, S) - m_a(z)\|^2 &= \tfrac{1}{2}\mathbb{E}\|m(z, S) - m(z, S')\|^2 \le \tfrac{1}{2} L(z)^2\, \mathbb{E}(S - S')^2 \\ &= L(z)^2\, \mathrm{Var}(T \mid Z = z,\ T > a). \end{aligned} \tag{14}$$

Combined with Equation (13), this bounds the cost of a fixed time input,

$$\mathcal{R}_a - \mathcal{R}_{\text{full}} \le \kappa^2\, \mathbb{E}\big[\mathbf{1}\{T > a\}\, L(Z)^2\, \mathrm{Var}(T \mid Z,\ T > a)\big]. \tag{15}$$

The cost is small when the noisy point identifies its time, so that the variance is small, or when its posterior mean changes slowly with time, so that $L(Z)$ is small. These are the two cases of Section 3.4.

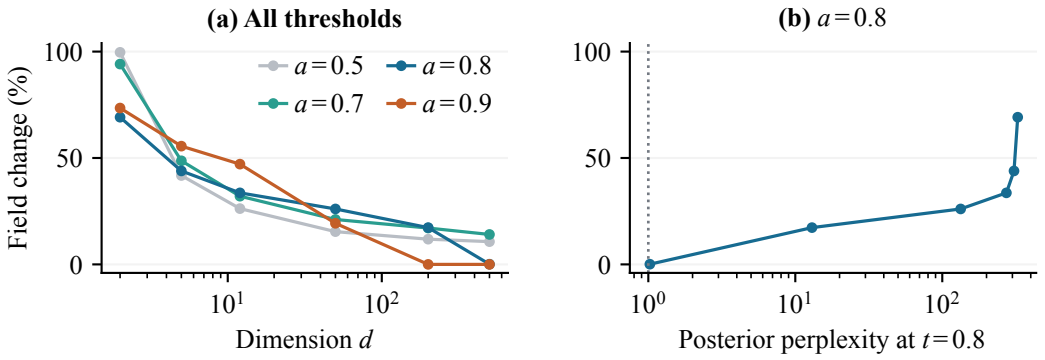


Figure 7: **The time input stops mattering once a noisy sample identifies its clean example.** Exact optimal fields for two moons in $\mathbb{R}^d$. (a) Relative change of the field when only its time input moves from a to 0.99. (b) The same change for $a = 0.8$ against the posterior perplexity, the effective number of clean examples compatible with a noisy sample.

A.2 High dimension and a closed-form example

Noise orthogonal to the data reveals the time. Consider an idealized distribution supported on a known d_0-dimensional linear subspace of $\mathbb{R}^d$, with $d > d_0$, and let $P_\perp$ be the orthogonal projection onto its complement. Since $P_\perp x_1 = 0$, for a fixed $0 \le t < 1$

$$P_\perp x_t = (1-t)P_\perp \epsilon, \qquad \frac{\|P_\perp x_t\|^2}{(1-t)^2} \sim \chi^2_{d-d_0}. \tag{16}$$

The estimate $\|P_\perp x_t\|^2/(d-d_0)$ therefore has mean $(1-t)^2$ and relative standard deviation $\sqrt{2/(d-d_0)}$, so the noisy point reveals its time more precisely as the number of directions orthogonal to the data grows. The calculation assumes a known flat subspace. For curved data the projection is local and must itself be inferred.

A closed-form example. Figure 7 checks the two cases of Section 3.4 where the optimal fields are exact. At $d = 2$, moving the time input from a to 0.99 changes the field by 69% to 100%, depending on the threshold. The change falls with dimension and reaches zero at $d = 200$ for $a = 0.9$ and at $d = 500$ for $a = 0.8$, which for $a = 0.8$ is the dimension at which the posterior perplexity reaches one. Lower thresholds need higher dimension, and at $d = 500$ the change is still 14% for $a = 0.7$ and 11% for $a = 0.5$. The data are 2,000 points on two moons in the first two coordinates of $\mathbb{R}^d$, with independent Gaussian noise of a fixed standard deviation added to every coordinate, and generation starts from $\mathcal{N}(0, I_d)$. Given t, the noisy point x_t is Gaussian around tx_1 with standard deviation $1-t$, so the posterior over the 2,000 points is a softmax of scaled squared distances, and the optimal time-conditioned field $f^*(z,t) = \mathbb{E}[Y \mid x_t = z,\, t]$ is exact. For each threshold a, we draw noisy points at $t = a$ and change only the time input of f^*, from a to 0.99. The relative change of the optimal field is the median of $\|f^*(z, 0.99) - f^*(z, a)\|$ over these points, divided by the median of $\|f^*(z, a)\|$. The perplexity in panel (b) is the exponential of the entropy of the posterior over the 2,000 points at $t = 0.8$, the effective number of clean examples that a noisy point remains compatible with. It is 2,000 when the noisy point carries no information about its clean example and 1 when it identifies it.

A control with shrinking data noise. When the noise added to the data instead shrinks as $1/\sqrt{d}$ per coordinate, so that the total spread of the data around the two moons stays fixed, the relative change also falls with dimension, to 7% for $a = 0.8$ and 8% for $a = 0.9$ at $d = 100$, while the posterior stays spread over many examples. The noisy points then lie far from the data, where the posterior mean of the clean example is close to the data mean at every time. A posterior concentrated on one example and a posterior spread over all of them both make the time input unnecessary. The time input matters in between, where the posterior is informative but has not yet singled out one example.

A.3 LOCAL CONVERGENCE OF REFINEMENT

Refinement with the ideal field $\kappa(x_\star - z)$ multiplies the distance to $x_\star$ by $1 - r$ at every step (Section 3.3). The following proposition shows that a learned field within δ of this ideal field behaves almost the same way.

Proposition 2 (Local refinement). *Let $\mathcal{B} = \{z \mid \|z - x_\star\| \le R\}$ be the ball of radius $R > 0$ around a point $x_\star$, and suppose $f_\theta(z, a) = \kappa(x_\star - z) + e(z)$ for all $z \in \mathcal{B}$, where $\kappa = \lambda/(1-a)$ and $\|e(z)\| \le \delta$. Set $\eta = r/\kappa$ with $0 < r < 2$, and $\rho = |1 - r|$. If*

$$\delta \le \kappa R \min\{1, (2-r)/r\}, \tag{17}$$

then the iterates of Equation (9) started at $z_0 \in \mathcal{B}$ remain in $\mathcal{B}$ and satisfy

$$\|z_k - x_\star\| \le \rho^k \|z_0 - x_\star\| + \frac{\eta\delta(1-\rho^k)}{1-\rho}. \tag{18}$$

Proof of Proposition 2. Whenever $z_k \in \mathcal{B}$, the update $z_{k+1} = z_k + \eta f_\theta(z_k, a)$ with $\eta\kappa = r$ gives

$$z_{k+1} - x_\star = (1-r)(z_k - x_\star) + \eta e(z_k), \qquad \|z_{k+1} - x_\star\| \le \rho\|z_k - x_\star\| + \eta\delta.$$

Condition (17) is equivalent to $\eta\delta \le (1-\rho)R$, since $(1-\rho)/r = \min\{1, (2-r)/r\}$. Thus $z_k \in \mathcal{B}$ implies $\|z_{k+1} - x_\star\| \le \rho R + \eta\delta \le R$, and by induction every iterate remains in $\mathcal{B}$. Iterating the inequality and summing the geometric series gives Equation (18), and hence $\limsup_{k\to\infty} \|z_k - x_\star\| \le \eta\delta/(1-\rho)$.

For $r \le 1$ this limit is attained. A constant error $e(z) = b$ with $\|b\| = \delta$ moves the fixed point to $x_\star + b/\kappa$, at distance $\delta/\kappa = \eta\delta/(1-\rho)$. When $x_\star$ is a clean example, $e(x_\star) = f_\theta(x_\star, a)$, which is the quantity the endpoint penalty drives to zero.

Refinement with an ideal field preserves the position on the manifold. Let $\mathcal{M}$ be a smooth manifold with a tubular neighborhood in which the nearest-point projection $\Pi_\mathcal{M}$ is unique. The ideal field

$$f_\mathcal{M}(z) = \kappa\big(\Pi_\mathcal{M}(z) - z\big) = -\nabla_z \frac{\kappa}{2} \operatorname{dist}(z, \mathcal{M})^2 \tag{19}$$

is autonomous. For $0 < r \le 1$, the update $z' = z + r(\Pi_\mathcal{M}(z) - z)$ stays on the normal segment through z. Hence $\Pi_\mathcal{M}(z') = \Pi_\mathcal{M}(z)$ and $\operatorname{dist}(z', \mathcal{M}) = (1-r)\operatorname{dist}(z, \mathcal{M})$. Refinement with this field reduces the distance to $\mathcal{M}$ and leaves the distribution of $\Pi_\mathcal{M}(z)$ unchanged, so the distribution along $\mathcal{M}$ is the one that transport delivers. The same holds for isolated fixed points. Where refinement starts determines which one it reaches, so transport sets the probability of each mode, which Section 4.5 measures.

A.4 STATIONARITY AND MODE PROBABILITIES

Section 3.3 noted that where samples converge does not decide how many converge to each mode. This section shows that the optimal field above a does not fix these probabilities by itself.

Consider independent Gaussian noise, training times drawn uniformly, and a finite data distribution $\mu_1 = \sum_j p_j \delta_{x^{(j)}}$ with atoms $x^{(j)}$ and probabilities p_j. For $t < 1$, let ρ_t be the density of x_t and $v_t(z) = \mathbb{E}[x_1 - \epsilon \mid x_t = z, t]$ the flow-matching velocity. The average density of x_t over $t \in (a, 1]$ and the unrestricted optimal field above a are

$$\nu_a(z) = \frac{1}{1-a}\int_a^1 \rho_t(z)\,dt, \qquad f^*(z,a) = \lambda \frac{\int_a^1 c_a(t)\rho_t(z)v_t(z)\,dt}{\int_a^1 \rho_t(z)\,dt}.$$

Using the continuity equation $\partial_t\rho_t + \nabla\cdot(\rho_t v_t) = 0$ and integrating by parts in t gives

$$\nabla\cdot(\nu_a f^*) = -\frac{\lambda}{1-a}\int_a^1 c_a(t)\partial_t\rho_t\,dt = \frac{\lambda}{1-a}(\rho_a - \nu_a). \tag{20}$$

This identity holds in the sense of distributions. The boundary term at $t = 1$ vanishes weakly because $c_a(t) \to 0$ while ρ_t keeps unit mass. The flux $\nu_a f^*$ therefore has the density ρ_a at the threshold as its

source and ν_a as its sink, rather than the data distribution. A consistent autonomous transport needs μ_1 in place of ν_a, which is the endpoint mass balance that BTM imposes (Lee et al., 2026).

The atoms have zero probability under $\nu_a(z)\,dz$, so a penalty on the atoms, such as the endpoint penalty, can set the field there to zero without changing the unrestricted optimum elsewhere or Equation (20). A finite network couples its predictions through shared parameters, so the endpoint penalty can still change the learned field away from the atoms. The probability carried by each basin of attraction depends on the field between the atoms.

B Objectives, estimators, and special cases

For $a < 1$, define the per-example field loss and endpoint penalty as

$$\ell_\theta(x_1, \epsilon, t) = \|f_\theta(x_t, \tau_a(t)) - \lambda c_a(t)(x_1 - \epsilon)\|^2, \tag{21}$$

$$P_\theta(x_1) = \|f_\theta(x_1, a)\|^2. \tag{22}$$

The objective of Equation (7) is $\mathbb{E}_q \ell_\theta + w\,\mathbb{E}P_\theta$. The multiplier λ sets the size of the targets, and w weights the endpoint penalty.

B.1 Three estimators of the endpoint penalty

At $t = 1$ the noisy point is x_1 and $c_a(1) = 0$, so the endpoint penalty is the field loss at the end of the path,

$$\ell_\theta(x_1, \epsilon, 1) = P_\theta(x_1). \tag{23}$$

For the mixture $q_p = (1-p)q + p\delta_1$ of the time distribution with a point mass at $t = 1$,

$$\frac{1}{1-p}\mathbb{E}_{q_p}\ell_\theta = \mathbb{E}_q \ell_\theta + \frac{p}{1-p}\mathbb{E}P_\theta, \tag{24}$$

so choosing $p = w/(1+w)$ puts the endpoint penalty into the distribution of training times. A batch can also fix the number of clean examples instead of sampling it, as in stratified sampling. For batch size B and $1 \le k < B$, the in-batch estimator used in the main text replaces k noisy examples by clean ones and reweights the two groups,

$$\widehat{\mathcal{L}}_k = \frac{1}{B-k}\sum_{i \in \mathcal{I}} \ell_i + \frac{w}{k}\sum_{j \in \mathcal{C}} P_j, \tag{25}$$

where $\mathcal{I}$ indexes the $B - k$ noisy examples and $\mathcal{C}$ the k clean ones. The three estimators are listed below.

Estimator	Examples evaluated	Batch objective
Separate batch	B noisy, B clean	$\frac{1}{B}\sum_{i=1}^{B}\ell_i + \frac{w}{B}\sum_{j=1}^{B}P_j$
In-batch, unweighted	$B-k$ noisy, k clean	$\frac{1}{B-k}\left(\sum_{i=1}^{B-k}\ell_i + \sum_{j=1}^{k}P_j\right)$
In-batch	$B-k$ noisy, k clean	$\frac{1}{B-k}\sum_{i=1}^{B-k}\ell_i + \frac{w}{k}\sum_{j=1}^{k}P_j$

The unweighted in-batch estimator gives the penalty the weight $k/(B-k)$, and choosing $k = \mathrm{round}(wB/(1+w))$ approximates the requested weight up to integer rounding. The in-batch estimator of Equation (25) preserves w exactly for any valid k. Inside an ordinary batch mean, its weights are $B/(B-k)$ for noisy examples and wB/k for clean examples. Changing k changes how many examples each term receives and the variance of the estimate. Tables 8 and 9 compare the three estimators on CIFAR-10 and ImageNet.

Algorithm 1 MFM training and sampling (in-batch estimator)

Require: $a \in [0, 1)$, $\lambda > 0$, endpoint-penalty weight $w > 0$, batch size B, clean examples $1 \le k < B$

procedure TRAINING UPDATE
  Draw B independent pairs (x_1^i, ϵ^i)
  Draw $t_i \sim q$ for $i \le B - k$; set $t_i = 1$ otherwise
  $x_{t_i}^i \leftarrow (1 - t_i)\epsilon^i + t_i x_1^i$
  $y_i \leftarrow \lambda c_a(t_i)(x_1^i - \epsilon^i)$
  $\omega_i \leftarrow B/(B - k)$ for noisy examples; $\omega_i \leftarrow wB/k$ for clean examples
  $L \leftarrow B^{-1} \sum_i \omega_i \| f_\theta(x_{t_i}^i, \min(t_i, a)) - y_i \|^2$
  Update θ using the gradient of L
end procedure
procedure SAMPLING(T, K, r)
  $z \sim \mathcal{N}(0, I)$; $\eta \leftarrow r(1 - a)/\lambda$
  if $a > 0$ **then**
    Choose $T \ge 1$; $h \leftarrow a/T$
    for $i = 0, \ldots, T - 1$ **do**
      $z \leftarrow z + (h/\lambda) f_\theta(z, ih)$
    end for
  end if
  for $j = 1, \ldots, K$ **do**
    $z \leftarrow z + \eta f_\theta(z, a)$
  end for
  return z
end procedure

Implementation details that affect equivalence. The expectation identities assume identical network behavior on clean examples. The image implementation evaluates the separate batch with dropout disabled, while both in-batch estimators share the training-mode forward pass. In class-conditional ImageNet training, label dropout can therefore apply to the clean examples of the in-batch estimators. The CIFAR U-Net also uses activation dropout (probability 0.3), disabled for the separate batch but active for in-batch clean examples. On images, the separate-batch estimator uses two vectorized forward passes and the in-batch estimators use one. B denotes the batch size per device.

B.2 SPECIAL CASES AND THRESHOLD SELECTION

Define $c_1(t) = 1$ separately. At $a = 1$, $\tau_1(t) = t$, so setting $w = 0$ and $\lambda = 1$ gives the ordinary flow-matching objective. With another positive multiplier, the transport velocity is recovered by dividing the field by λ. The identity for the endpoint penalty above applies only to $a < 1$.

At $a = 0$, the time input is constant and $c_0(t) = 1 - t$. With $w = 0$, this is the linear-decay variant of EqM. The published truncated-decay EqM baseline instead removes the time input everywhere while keeping its nonzero decay threshold. It is recovered by removing the time input independently of that threshold, as in our EqM baseline, rather than by setting the MFM threshold to zero.

Finally, we select a by sample quality (Table 10). The regression loss is unsuitable for this choice. Changing a changes both the information available to the network and the size of its targets, so a lower loss can reflect smaller targets.

C ALGORITHMS AND EXPERIMENTAL PROTOCOL

Algorithm 1 uses Euler transport and the sign convention of the main text. At $a = 0$ omit transport. For flow matching at $a = 1$, use its separate definition and omit refinement and the endpoint penalty. Setting $w = 0$ requires no clean examples. Class labels, when used, are additional network inputs in both phases.

C.1 IMAGENET-256

ImageNet (Deng et al., 2009) images are encoded with the frozen `sd-vae-ft-ema` autoencoder (Rombach et al., 2022), yielding $4 \times 32 \times 32$ latents scaled by 0.18215. The precomputed training set stores the encoder's posterior mean and variance for each image and its horizontal flip, and a new latent is sampled at each use. Main models use DiT-B/2 (approximately 130 million parameters), 400,320 optimizer updates, global batch 256, AdamW learning rate 10^{-4}, $\beta = (0.9, 0.999)$, zero weight decay, constant learning rate, and EMA decay 0.9999 without warm-up. Arithmetic is FP32 with TF32 enabled. Training times are uniform on $[0, 1]$. Class-label dropout is 0.1.

The EqM baseline has a constant time input, target cutoff 0.8, and multiplier 4. FM uses the unscaled velocity target and the true time as input. The main MFM setting uses $a = 0.8$, $\lambda = 4$, and an endpoint penalty of weight $w = 0.1$ on a separate batch. It takes 230 Euler transport steps and 20 refinement steps with $r = 0.5$, hence $\eta = 0.025$. EqM uses 250 gradient-descent (GD) updates of size 0.003. The FM SDE baseline uses 250 evaluations and the same trained FM field. Our EqM implementation matches the official code in float64, and our EqM baseline reaches an unguided B/2 FID of 32.62 against the published 32.85.

Longer training and the showcase. The DiT-XL/2 in-batch model ($k = 4$) of Table 1 was trained on from its 80-epoch checkpoint to 200 epochs (1,000,000 updates) with no other change. At 200 epochs its ADM FID-50k is 8.796 without guidance and 2.445 at $g = 1.5$, against 12.64 and 3.532 at 80 epochs. Without guidance, this 200-epoch model already has a lower FID than DiT-XL/2 after 1400 epochs (7M updates), which reports 9.62 (Peebles & Xie, 2023), and is within 0.5 of SiT-XL/2 at the same length, which reports 8.3 (Ma et al., 2024). Figure 1 shows samples from a later checkpoint of the same run, at 1,250,000 updates (about 250 epochs), for which no FID was computed. They were selected by eye from 512 samples of 16 chosen classes, drawn with $g = 1.5$, 230 Euler transport steps and 20 refinement steps. From update 1,000,000 the run uses eight devices with 32 examples each and keeps $k = 4$ per device. Figure 10 shows unselected samples.

Classifier-free guidance (Ho & Salimans, 2022) combines the class-conditioned and null-label predictions as $f_g = f_\varnothing + g(f_y - f_\varnothing)$ on the first three latent channels and keeps the class-conditioned prediction on the fourth channel, following the baseline implementation. The scale $g = 1$ means no guidance. When $g > 1$, one function evaluation includes both label conditions. The tables count function evaluations and state this doubling of example evaluations. All methods use the same rule. Guidance scales are screened with 5,000-sample evaluations, and the selected settings are evaluated on 50,000 samples.

Main ImageNet metrics use ADM's TensorFlow evaluator against the reference `VIRTUAL_imagenet256_labeled.npz`. Samples use uniform class labels and EMA weights, and decoded pixels use the baseline's $127.5x + 128$ conversion. FID, sFID, IS, precision, and recall in each table row refer to the same generated batch. PyTorch FID (TorchMetrics/torch-fidelity) appears only where labeled, such as Figure 6a.

C.2 CIFAR-10

CIFAR-10 (Krizhevsky, 2009) is modeled unconditionally in pixel space with a U-Net. The matched comparison trains every method for 293,000 updates with global batch 512, AdamW learning rate 10^{-4} decaying linearly, $\beta = (0.9, 0.95)$, weight decay 0.01, horizontal flips, and EMA decay 0.999 with the implementation's early-update warm-up. Every method draws training times from the log-normal noise distribution of Karras et al. (2022), with $\log \sigma \sim \mathcal{N}(-1.2, 1.2^2)$ and $t = 1/(1+\sigma)$. The interpolation path remains linear. Seeds are 42, 101, and 102. Evaluation uses 50,000 samples and the same in-repository FID implementation for every model.

MFM sampling uses 200 transport and 50 refinement evaluations at budget 250, or 500 transport and 200 refinement evaluations at budget 700. FM uses 250 or 700 Euler steps, and EqM uses the same number of gradient-descent updates with separately tuned step sizes 0.002 and 0.001. Estimators of the endpoint penalty are compared at batch size 128 per device, with $k = 12$ clean examples for the unweighted in-batch estimator and $k = 32$ or $k = 64$ for the in-batch estimator.

The target multiplier. On ImageNet with DiT-B/2, replacing $\lambda = 4$ by $\lambda = 1$ raises the in-repository FID-50k from 33.755 to 39.137 without guidance and from 10.047 to 11.280 at $g = 2.25$, with both samplers dividing by the correct multiplier. We therefore use $\lambda = 4$ on ImageNet. On CIFAR-10 the order reverses, and $\lambda = 1$ has the lower mean FID at both budgets (Table 2).

C.3 Inpainting and composition

The quantitative experiments use the CIFAR-10 models of Table 2 with one training seed and FID on 10,000 samples. We also show qualitative examples from the ImageNet DiT-XL/2 models.

Inpainting. Part of each of the 10,000 test images is hidden and filled in, with the same masks for every method. The masks hide 25% of the pixels at random, a centered square covering 25%, the left half, or everything but a centered quarter (expand). Every method fills the hidden region with the same procedure. A ladder of noise levels is visited from noise to data, and at each level the observed pixels are set to their noised true values while the hidden pixels are updated by the model, as in RePaint (**?**). The ladder's shape (uniform or geometric rungs, repeats per rung, and how much fresh noise is injected) is swept over the same grid for every method, and Table 3 reports each method's best setting at 200 function evaluations. MFM is the separate-batch model with $a = 0.9$ and $\lambda = 1$, FM the Euler model, and EqM the model of Table 2. Filling the hidden region with refinement alone, holding the observed pixels fixed and iterating the fixed field of MFM without any noise ladder, failed. On MFM's own model it is worse than the ladder by 3.8 FID on the box mask and 7.9 on random pixels, because the hidden pixels start at noise, far from the data, where the field above a has no training signal. The advantage of MFM in Table 3 grows with the hidden fraction. FM is best on scattered pixels, the box mask is a near tie, and MFM leads by 0.40 FID on the half mask and by 2.25 on the expand mask, where it wins all eight ladder settings that every method ran. The RMSE of the filled region against the true pixels differs by at most 0.02 between methods, because the true pixels are one sample among many plausible fills. Each entry is the best of up to 26 ladder settings from a single model seed.

ImageNet inpainting. Figures 8 and 9 use the DiT-XL/2 models of Table 1 on 256 ImageNet validation images with a centered 128×128 square hidden. The masked image is encoded once, with the hidden square set to gray, and the hidden latent cells are generated along the same linear path as training. At each of 250 uniform levels, the observed latent cells are replaced by a freshly noised copy of the encoded observation, following the observed-region substitution of the latent-diffusion DDIM sampler (Song et al., 2021a; Rombach et al., 2022), and the final latent is decoded once with the known pixels copied from the input. FM's update is the Euler step, and EqM and MFM convert their targets to a clean estimate for the same update. Every method sees the same masks, class labels and random draws. The six cases were fixed before the runs and are shown at guidance 1.5 and 1.0. No FID was computed for these runs, the completions show visible artifacts for every method, and they do not show a clear visual advantage for any one method.

Composition. Two models are trained on class subsets whose overlap is the bird and cat classes: parent A on the vehicle classes plus birds and cats, and parent B on the animal classes. A composed field adds the two parents' fields, and its samples are scored against the real bird and cat images. All samples use 250 function evaluations, with MFM's sampler for MFM and gradient descent for EqM. Both composed fields concentrate on the shared classes, beating their better parent, and MFM does so by a wider margin: 7.3 FID against 1.1 for EqM. A composed FM model, which averages the two velocity fields, reaches 25.122 under the same protocol, between MFM and EqM. One class pairing and one seed per model.

ImageNet composition. Following Wang & Du (2025), we add the fields conditioned on two class labels. EqM samples the sum with gradient descent. MFM applies its native sampler to the averaged field, with 230 transport steps and 20 refinement steps. Both models are trained for about 80 epochs and use the same labels, initial noise and guidance 1.5. Each image requires 250 composite-field evaluations, or 750 network-branch evaluations with guidance. A disjoint pilot fixes one step size per method. The summed-field gradient descent steps are 0.0017 for EqM and 0.0125 for MFM refinement. Figure 5 selects examples from 832 matched pairs across 26 label combinations to show

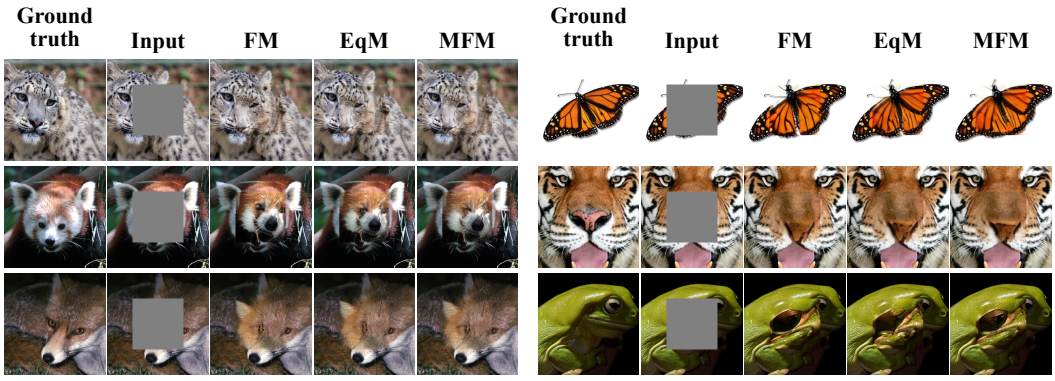


Figure 8: **ImageNet-256 inpainting with guidance** 1.5**.** No method has a visible advantage. DiT-XL/2 models of Table 1 fill a centered 128×128 square with the same observed-latent replacement procedure at 250 function evaluations, with matched masks, labels and random draws. Each group shows the ground truth, the masked input, and the fills of FM, EqM and MFM.

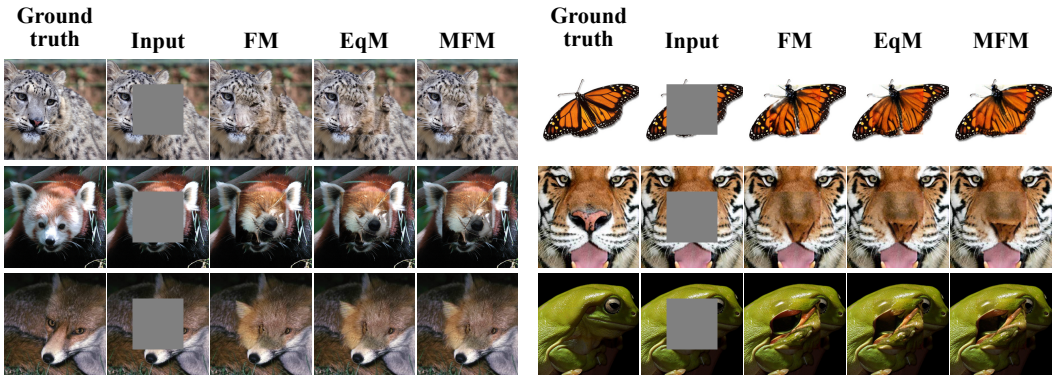


Figure 9: **ImageNet-256 inpainting without guidance.** The six cases of Figure 8 under the same protocol at guidance 1.0.

visually coherent MFM compositions. In these selected pairs, MFM produces a clearer fish profile and lighthouse tower, along with cleaner cup and spoon geometry. The flamingo comparison is closer.

C.4 Two-dimensional experiments

The threshold study in Table 11 uses four two-dimensional datasets (moons, eight Gaussians, checkerboard, and Swiss roll). The sliced Wasserstein-2 distance compares one-dimensional projections of generated and held-out data. Each entry is one training run, and the MFM and FM budgets are approximately matched at 200–206 evaluations. EqM uses a separately tuned 50-evaluation sampler.

Vector fields and energy interpretations. The image models are unconstrained vector networks. We measure the asymmetry of their Jacobian J by $\|(J - J^\top)v\|/\|(J + J^\top)v\|$ for random directions v, which is zero for the gradient of a scalar function. On DiT-B/2, at the points each sampler reaches after 20 refinement steps, its median is 0.83 for MFM ($a = 0.9$) and 0.54 for EqM, so neither field is the gradient of an energy there. The optimum in Equation (6) explains this. It mixes fixed-noise denoisers with state-dependent weights, and such a mixture of gradient fields is generally not a gradient.

Table 4: **Image generation on ImageNet-256 with DiT-XL/2, all metrics.** MFM has the lowest FID without guidance and at the shared scale $g = 1.5$, and the lowest sFID at every scale. At $g = 1.75$, the baselines' best tested scale, EqM has the lowest FID. ADM metrics on 50,000 images, 250 function evaluations, one training run per method. MFM uses $a = 0.8$, $\lambda = 4$ and $w = 0.1$, and k counts clean examples out of 64 per device. Bold marks the best value in each block.

Method	g	FID ↓	sFID ↓	IS ↑	Precision ↑	Recall ↑
FM (Lipman et al., 2023), SDE	1.00	17.44	5.09	76.8	0.647	0.629
EqM (Wang & Du, 2025), GD	1.00	14.92	6.70	86.1	0.651	**0.646**
MFM (ours), separate batch	1.00	12.74	**4.88**	92.6	**0.672**	0.640
MFM (ours), in-batch, $k = 16$	1.00	15.25	4.93	80.7	0.663	0.630
MFM (ours), in-batch, $k = 4$	1.00	**12.640**	4.985	**95.37**	0.66676	0.6428
FM, SDE	1.50	5.301	4.827	168.08	0.80718	0.5253
EqM, GD	1.50	4.111	5.664	189.91	0.80506	**0.5491**
MFM (ours), separate batch	1.50	3.781	**4.749**	198.59	0.8207	0.5350
MFM (ours), in-batch, $k = 16$	1.50	4.922	5.119	172.45	0.8114	0.5276
MFM (ours), in-batch, $k = 4$	1.50	**3.532**	4.975	**206.33**	**0.82152**	0.5409
FM, SDE	1.75	4.650	5.287	211.8	0.855	0.469
EqM, GD	1.75	**3.736**	5.685	236.0	0.852	**0.493**
MFM (ours), separate batch	1.75	3.868	**5.162**	**246.15**	**0.8635**	0.4804
MFM (ours), in-batch, $k = 16$	1.75	4.495	5.678	215.08	0.8584	0.4735

Table 5: **Image generation on ImageNet-256 with DiT-B/2.** MFM has the lowest FID in every guidance block, and the endpoint penalty ($w = 0.1$) lowers sFID relative to no penalty ($w = 0$). ADM metrics on 50,000 images, 250 function evaluations, one training run per row, MFM with $a = 0.8$. The guidance scale $g = 1$ means no guidance, and the last block uses each method's best tested scale. Bold marks the best value in each block. Tables 6 and 7 report repeated runs.

Method	g	FID ↓	sFID ↓	IS ↑	Precision ↑	Recall ↑
FM (Lipman et al., 2023), SDE	1.00	35.90	6.52	41.32	0.521	**0.645**
EqM (Wang & Du, 2025), GD	1.00	32.62	7.74	44.02	0.545	0.634
MFM (ours), $w = 0$	1.00	**30.03**	6.78	46.27	**0.561**	0.634
MFM (ours), $w = 0.1$	1.00	30.21	**5.78**	**47.17**	0.557	0.642
FM, SDE	1.50	15.49	5.29	91.00	0.690	**0.553**
EqM, GD	1.50	13.90	6.43	96.46	0.703	0.551
MFM (ours), $w = 0.1$	1.50	**12.35**	**5.16**	**104.33**	**0.718**	0.539
FM, SDE	2.50	7.70	7.12	192.88	0.858	0.364
EqM, GD	2.50	7.15	7.01	**200.11**	**0.862**	0.369
MFM (ours), $w = 0$	2.25	7.23	8.03	184.76	0.847	**0.401**
MFM (ours), $w = 0.1$	2.25	**6.92**	**6.36**	189.01	0.853	0.396

FM uses its SDE sampler, EqM gradient descent (GD) with step size 0.003, and MFM 230 Euler transport steps and 20 refinement steps. The endpoint penalty is computed on a separate clean batch. At each method's best guidance scale, the FID difference to EqM is about as large as the change between training runs.

D Additional experimental results

The following tables report DiT-XL/2 and DiT-B/2 results, repeated training runs, estimators of the endpoint penalty, threshold choices, and the two-dimensional experiments.

Table 6: **Repeated MFM training runs on ImageNet-256.** FID changes by less than 0.4 between runs at every guidance scale, below MFM's margins over the baselines without guidance and at $g = 1.5$ (Table 5). DiT-B/2, $a = 0.8$, 250 function evaluations, ADM metrics on 50,000 samples. Bold marks the best value in each column.

	FID ↓			sFID ↓		
g	Run 1	Run 2	Mean ± SD	Run 1	Run 2	Mean ± SD
1.00	30.21	30.44	30.33 ± 0.16	5.78	5.86	5.82 ± 0.06
1.50	12.35	12.56	12.46 ± 0.15	**5.16**	**5.18**	$\mathbf{5.17 \pm 0.01}$
2.00	7.27	7.19	7.23 ± 0.06	5.78	5.62	5.70 ± 0.11
2.25	**6.92**	**6.72**	$\mathbf{6.82 \pm 0.14}$	6.36	6.08	6.22 ± 0.20
2.50	7.22	6.88	7.05 ± 0.24	6.98	6.58	6.78 ± 0.28

Table 7: **Repeated DiT-B/2 baseline runs on ImageNet-256.** Baseline FID changes by up to 1.1 between runs without guidance and by less than 0.2 at $g = 2.25$ and $g = 2.5$, so MFM's unguided margins in Table 5 exceed run-to-run variation while its best-guidance margins are about as large as it. ADM FID on 50,000 samples, two training seeds per method, 250 function evaluations. Bold marks the best value in each column.

Method, run	$g = 1$	$g = 1.5$	$g = 2.25$	$g = 2.5$
EqM (Wang & Du, 2025), run 1	**32.62**	**13.90**	**7.18**	**7.15**
EqM, run 2	32.971	14.096	7.193	7.193
FM (Lipman et al., 2023), run 1	35.90	15.49	7.72	7.70
FM, run 2	34.770	14.725	7.558	7.632

Table 8: **Estimators of the endpoint penalty on CIFAR-10.** The in-batch estimator with $k = 64$ has the lowest FID at 250 evaluations in half the training time of the separate batch, and without the penalty FID degrades at 700 evaluations. All MFM runs use $a = 0.9$ and the recipe of Table 2. FID is mean ± sample standard deviation over three seeds, $B = 128$ is the batch size per device, and k counts clean examples. Bold marks the best mean in each FID column.

Estimator	λ	Examples/update	Wall time	FID, 250 eval.	FID, 700 eval.
None ($w = 0$)	4	B	11.4 h	3.700 ± 0.171	13.75 ± 6.22
Separate batch	4	$2B$	22.1 h	2.744 ± 0.035	2.664 ± 0.042
Separate batch	1	$2B$	22.1 h	2.678 ± 0.053	$\mathbf{2.653 \pm 0.067}$
In-batch, unweighted, $k = 12$	1	B	11.4 h	2.678 ± 0.107	2.724 ± 0.069
In-batch, $k = 32$	1	B	11.4 h	2.651 ± 0.051	2.733 ± 0.019
In-batch, $k = 64$	1	B	11.4 h	$\mathbf{2.527 \pm 0.020}$	2.654 ± 0.106

The separate-batch and in-batch estimators use $w = 0.1$. The unweighted in-batch estimator sets $k = \mathrm{round}(0.1B/1.1)$, giving the penalty weight $k/(B-k) = 0.1034$. Both the transport and the refinement step counts increase between the budget columns.

Table 9: **Estimators of the endpoint penalty on ImageNet-256.** Every estimator lowers sFID relative to no penalty, and the separate batch has the lowest FID at $g = 1.5$ and $g = 2.25$. DiT-B/2, $a = 0.8$, $\lambda = 4$, 250 function evaluations, ADM metrics on 50,000 samples from one training run per model. k counts clean examples per device, out of 128 for the unweighted estimator and out of 64 otherwise. Bold marks the best value in each column.

	$g = 1$		$g = 1.5$		$g = 2.25$		$g = 2.5$	
Estimator	FID ↓	sFID ↓	FID ↓	sFID ↓	FID ↓	sFID ↓	FID ↓	sFID ↓
None ($w = 0$)	**30.03**	6.78	12.520	6.318	7.23	8.03	7.548	8.866
Separate batch	30.21	**5.78**	**12.346**	**5.162**	**6.92**	**6.36**	7.216	**6.978**
In-batch, unweighted, $k = 12$	32.03	5.97	13.535	5.489	7.33	6.98	7.428	7.748
In-batch, $k = 16$	33.53	5.97	14.491	5.520	7.72	7.04	7.687	7.797
In-batch, $k = 8$	31.701	5.910	13.250	5.443	7.154	6.915	7.379	7.663
In-batch, $k = 6$	32.156	6.001	13.482	5.437	7.210	6.897	7.273	7.636
In-batch, $k = 4$	31.364	5.965	13.094	5.472	7.048	6.905	**7.212**	7.651

Table 10: **Threshold selection on ImageNet-256.** $a = 0.7$ has the best FID without guidance, and $a = 0.8$ the best FID at the best scale and the best sFID at $g = 1.5$, so we use $a = 0.8$. DiT-B/2 with the endpoint penalty ($w = 0.1$, separate batch), $\lambda = 4$, 250 function evaluations, ADM metrics on 50,000 samples, one training run per row. Bold marks the best value in each column.

	$g = 1$		$g = 1.5$		Best tested guidance		
a	FID ↓	sFID ↓	FID ↓	sFID ↓	g	FID ↓	sFID ↓
0.7	**29.73**	**5.76**	**12.23**	5.66	2.25	7.15	7.63
0.8	30.21	5.78	12.35	**5.16**	2.25	**6.92**	6.36
0.9	32.84	6.49	13.78	5.35	2.25	6.95	**5.92**

On CIFAR-10, the $a = 0.8$ three-run means are 4.016 and 3.536 at 250 and 700 evaluations, and the $a = 0.9$ three-run means are 2.744 and 2.664 (both with $\lambda = 4$ and the endpoint penalty on a separate batch).

Table 11: **Two-dimensional generation across thresholds.** MFM with $a \geq 0.9$ matches FM, and $a = 0.5$ fails on three of the four datasets. Sliced Wasserstein-2 distance to held-out samples, lower is better, one training run per entry. The MFM rows use no endpoint penalty ($w = 0$), and EqM uses its best tested sampler for each dataset. Bold marks the best value in each column.

Method	Evaluations	Moons	Eight Gaussians	Checkerboard	Swiss roll
FM (Lipman et al., 2023)	200	0.0212	0.1224	0.0898	0.0152
EqM (Wang & Du, 2025), tuned	50	0.0733	0.1211	0.1507	0.0669
MFM (ours), $a = 0.50$	206	0.1157	**0.1071**	0.4482	0.1219
MFM (ours), $a = 0.80$	204	0.0246	0.1217	0.1007	0.0225
MFM (ours), $a = 0.90$	203	**0.0177**	0.1248	0.0941	0.0169
MFM (ours), $a = 0.95$	203	**0.0177**	0.1240	**0.0895**	**0.0147**

FM and MFM use 100 Heun transport steps, costing 200 evaluations. MFM adds six refinement steps at $a = 0.5$, four at $a = 0.8$, and three at $a \geq 0.9$. EqM selects its decay threshold and step size per dataset.



Figure 10: **Selected ImageNet-256 samples at guidance** $g = 1.5$**.** MFM produces clearer subject shapes and details than FM and EqM on these examples, selected from 512 saved outputs with the same output indices for every method. DiT-B/2 models of Table 5, 250 function evaluations, MFM with $a = 0.8$ and $w = 0.1$.

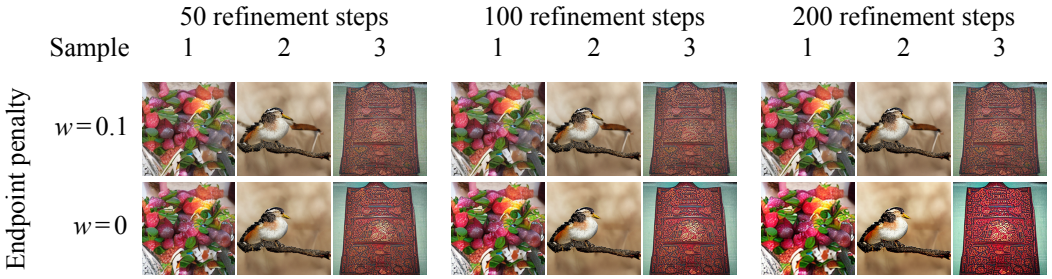


Figure 11: **Stability under additional refinement.** With the endpoint penalty ($w = 0.1$) appearance stays nearly unchanged from 50 to 200 refinement steps, and without it ($w = 0$) contrast and saturation increase. Three examples selected from 64 saved sequences, DiT-B/2, $a = 0.8$, $g = 1$, 230 Euler transport steps.

D.1 Qualitative image comparisons

These selected examples illustrate image structure and stability during refinement. Figure 10 shows cases where MFM produces clearer subject shapes and details than FM and EqM. Figure 11 shows how appearance changes with additional refinement. All evaluations use seed zero, batch size 250, EMA weights, and the same decoder. Each column uses the same output index across rows. Shared class and initial-noise ordering is inferred from the common sampling setup; FM's SDE sampler adds further noise. Each row is one trained model.

D.2 True time input versus a time input held fixed

Table 12 reports a CIFAR-10 experiment with fixed transport and varying refinement. The MFM model with the endpoint penalty keeps FID-10k near 5.03 from 10 through 50 refinement steps. The model that receives the true time and has no endpoint penalty worsens from 10.02 at three steps to 101.8 at 50. Both samplers hold the time input fixed during refinement. The two runs differ in the time input, the endpoint penalty, and the device count.

Table 12: **True time input versus a time input held fixed above a.** With the true time input and no penalty, FID rises by an order of magnitude as refinement continues, while MFM stays flat. CIFAR-10, FID on 10,000 samples after fixed transport. MFM uses $w = 0.1$, and the model with the true time no penalty. The final column gives FID on 50,000 samples at 20 refinement steps for MFM and at three for the model with the true time. Bold marks the better value in each threshold block.

Time input	Refinement steps, FID-10k ↓							FID-50k ↓
	0	1	2	3	10	20	50	
True time, t, $a = 0.9$	**82.4**	**42.1**	**16.3**	10.02	37.4	55.6	101.8	7.872
MFM (ours), $\tau_a(t)$, $a = 0.9$	83.93	43.17	17.29	**6.68**	**5.03**	**5.03**	**5.02**	**2.969**
True time, t, $a = 0.8$	153.17	86.23	45.90	25.11	67.44	102.00	132.07	23.218
MFM (ours), $\tau_a(t)$, $a = 0.8$	**152.97**	**85.39**	**44.68**	**18.39**	**6.60**	**6.52**	**6.32**	**4.476**

All four runs use $\lambda = 4$, 400,000 training steps, global batch 256, constant learning rate 10^{-4}, EMA decay 0.9999, and training seed 42. The MFM runs use four devices and the runs with the true time two, which changes the random draws at the same seed. Sampling uses 250 Heun transport steps (500 function evaluations) followed by the stated refinement steps at $r = 0.5$, so $\eta = 0.0125$ at $a = 0.9$ and $\eta = 0.025$ at $a = 0.8$. The model with the true time is best after three refinement steps at both thresholds.

Table 13: **Mode proportions of the exact fields across separation and threshold.** EqM misallocates samples when the modes are close and MFM does so only when it switches early ($a = 0.5$); with $a \geq 0.8$ MFM is close to FM and BTM. TV distance, mean $\pm$ sample standard deviation across three evaluation draws. In every cell all samples reach the points, within 0.05. Bold marks the lowest mean in each row.

		Baselines			
Radius b	a	FM (Lipman et al., 2023)	EqM (Wang & Du, 2025)	BTM (Lee et al., 2026)	MFM (ours)
0.5	0.5	$\mathbf{0.0032 \pm 0.0010}$	0.5000 ± 0.0000	0.0136 ± 0.0021	0.3396 ± 0.0023
0.5	0.8	$\mathbf{0.0032 \pm 0.0010}$	0.2943 ± 0.0014	0.0065 ± 0.0013	0.0099 ± 0.0013
0.5	0.9	0.0032 ± 0.0010	0.0777 ± 0.0020	0.0046 ± 0.0010	$\mathbf{0.0031 \pm 0.0010}$
1	0.5	$\mathbf{0.0032 \pm 0.0010}$	0.3241 ± 0.0015	0.0076 ± 0.0009	0.0524 ± 0.0022
1	0.8	$\mathbf{0.0032 \pm 0.0010}$	0.0695 ± 0.0007	0.0049 ± 0.0012	$\mathbf{0.0032 \pm 0.0011}$
1	0.9	$\mathbf{0.0032 \pm 0.0010}$	0.0126 ± 0.0014	0.0044 ± 0.0011	$\mathbf{0.0032 \pm 0.0010}$
2	0.5	$\mathbf{0.0032 \pm 0.0010}$	0.0592 ± 0.0027	0.0041 ± 0.0023	0.0060 ± 0.0013
2	0.8	$\mathbf{0.0032 \pm 0.0010}$	0.0054 ± 0.0020	0.0035 ± 0.0021	$\mathbf{0.0032 \pm 0.0010}$
2	0.9	$\mathbf{0.0032 \pm 0.0010}$	0.0037 ± 0.0025	0.0036 ± 0.0019	$\mathbf{0.0032 \pm 0.0010}$

D.3 Mode proportions under continued refinement

Distribution and measurements. The target consists of five points on a regular pentagon in $\mathbb{R}^2$, with probabilities $p = (0.50, 0.20, 0.15, 0.10, 0.05)$ and radius $b \in \{0.5, 1, 2\}$. Sampling starts from $\mathcal{N}(0, I)$. For the fractions $\widehat{p}_j$ of samples nearest to each point, the total variation (TV) distance to the target proportions is $D_{\mathrm{TV}} = \frac{1}{2}\sum_{j=1}^{5} |\widehat{p}_j - p_j|$. It equals the fraction of samples that would have to change mode to match p. We separately measure the root-mean-square distance to the nearest point and the fraction of samples farther than 0.05 from every point. The sweep over radius and threshold uses fixed optimization settings.

Population fields. We compute the optimal fields exactly over the choice of clean point and by numerical quadrature over the time t wherever a field averages over it. Composite Gauss–Legendre quadrature uses 32 nodes per panel and double precision. EqM and MFM share the truncated target, cutoff a, and multiplier $\lambda = 4$. BTM uses its corrected interpolation schedule (Lee et al., 2026), with the raw field divided by $\lambda(1 + a)/2$ for its native sampling scale. Each cell uses three independent draws of 32,768 starting points. MFM uses 128 Heun transport steps followed by 1,024 Euler refinement steps of size $\eta = (1 - a)/16$. EqM and BTM take 1,024 steps of their time-independent fields. FM uses 128 Heun steps to $1 - 10^{-4}$ and one final denoising evaluation, then holds its output. Total function evaluations are therefore 1,280 for MFM, 1,024 for EqM and BTM, and 257 for FM.

Table 13 separates the distance to the points from the mode proportions. At $b = 0.5$, EqM and MFM with an early threshold ($a = 0.5$) both reach the points with incorrect proportions. Keeping the time input until $a = 0.8$ lowers the MFM error by more than an order of magnitude, while the EqM error stays large. When the modes are farther apart ($b = 2$) and $a \geq 0.8$, all methods are close. With equal probabilities at $b = a = 0.5$, EqM stays at distance 0.5 from the points, and every sample lies farther than 0.05 from all of them.

Numerical resolution. For $b = 0.5$ and $a = 0.8$, increasing quadrature to 64 nodes per panel, transport to 256 steps, and refinement to 2,048 steps at half the step size gives $D_{\mathrm{TV}} = 0.287638 \pm 0.001315$ for EqM, 0.010842 ± 0.001366 for MFM, and 0.004584 ± 0.000962 for BTM. Their mean distances are 1.87×10^{-9}, 1.54×10^{-9}, and 1.03×10^{-4}, respectively, with zero outside fraction. The EqM discrepancy changes with numerical resolution but remains large. This control costs 2,560 evaluations for MFM and 2,048 for EqM and BTM. For comparison, 20,000 independent multinomial draws of 32,768 exact target samples give a mean TV distance of 0.00391, with central 95% range $[0.00137, 0.00741]$. The MFM error at this threshold remains above that reference range.

Learned fields with the same endpoint penalty. At $b = 0.5$, $a = 0.8$, all methods use the same 34,306-parameter network with three hidden layers of width 128, paired initialization and training draws, batch 1,024, and Adam learning rate 10^{-3}. All models evaluate $2B$ examples per update.

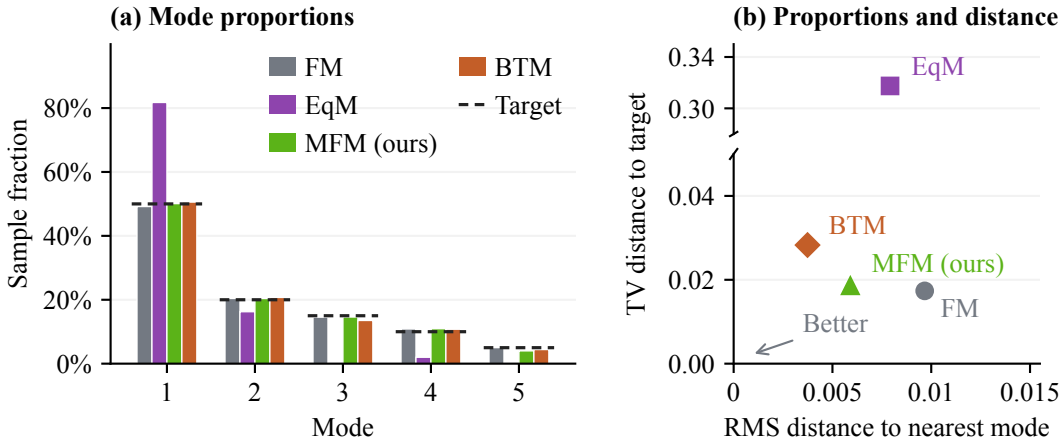


Figure 12: **Same destinations, different proportions.** Five unequal modes at radius 0.5, $a = 0.8$, 40,000 training updates, three training seeds. EqM and MFM use the same endpoint penalty ($w = 0.1$). (a) The fraction of samples nearest to each mode against its target probability. EqM sends too many samples to the most probable mode and almost none to the least probable ones, while FM, MFM and BTM match the targets. (b) Total variation distance to the target proportions against root-mean-square distance to the nearest mode, averaged over seeds. Every method reaches the modes and only the proportions differ, with EqM far above the others. Evaluations are 257 for FM, 1,024 for EqM and BTM, and 1,280 for MFM.

Table 14: **Mode proportions and distance to the points for learned fields.** EqM's TV distance stays large with and without the endpoint penalty, while MFM's is close to FM's at both training lengths. Unequal probabilities at $b = 0.5$, $a = 0.8$. Outside is the fraction of samples farther than 0.05 from every point, and w is the weight of the endpoint penalty. Mean ± sample standard deviation over three training seeds. Bold marks the lowest mean in each column of each block.

Updates	Method	TV distance ↓	RMS distance ↓	Outside ↓
20,000	FM (Lipman et al., 2023)	$\mathbf{0.0158 \pm 0.0043}$	0.0120 ± 0.0047	0.0039 ± 0.0041
	EqM (Wang & Du, 2025), $w = 0$	0.3273 ± 0.0756	0.0191 ± 0.0053	0.0392 ± 0.0679
	EqM, $w = 0.1$	0.2587 ± 0.0463	0.0073 ± 0.0030	**0**
	BTM (Lee et al., 2026)	0.0458 ± 0.0196	$\mathbf{0.0045 \pm 0.0013}$	**0**
	MFM (ours), $w = 0$	0.0256 ± 0.0149	0.0109 ± 0.0008	**0**
	MFM (ours), $w = 0.1$	0.0261 ± 0.0148	0.0081 ± 0.0025	**0**
40,000	FM	$\mathbf{0.0174 \pm 0.0028}$	0.0097 ± 0.0036	0.0022 ± 0.0026
	EqM, $w = 0.1$	0.3175 ± 0.0218	0.0079 ± 0.0029	**0**
	BTM	0.0283 ± 0.0046	$\mathbf{0.0037 \pm 0.0013}$	**0**
	MFM (ours), $w = 0.1$	0.0186 ± 0.0063	0.0059 ± 0.0004	**0**

EqM and MFM are each trained without the endpoint penalty ($w = 0$) and with the same endpoint penalty of weight $w = 0.1$. Table 14 includes all six models at 20,000 updates and four of them again at 40,000 updates. For each training seed, we average three evaluation draws of 16,384 samples, then report the mean and sample standard deviation across three training seeds. Sampling follows the 128-transport-step and 1,024-refinement-step protocol above, with the same unequal evaluation budgets. Figure 12 plots the 40,000-update rows of Table 14.

The endpoint penalty reduces the distance to the points for both EqM and MFM at 20,000 updates, while the TV distance of EqM stays large (0.2587 with the penalty). MFM has a lower TV distance with and without the penalty, and the longer training preserves the comparison. At 40,000 updates, the reported mode proportions and distances of the autonomous controls do not change between 512 and 1,024 refinement steps. BTM matches the target proportions without a time input in the

population setting (0.0065 at $b = 0.5$, $a = 0.8$), and its learned field reaches a TV distance of 0.0283 at 40,000 updates.