FLOWPINN: PHYSICS-INFORMED NEURAL FLOW MAPS FOR ONE-STEP GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Conditional flow matching (CFM) learns a velocity field and then generates every sample with a numerical ODE solve. We replace that pair with a single object: a neural *flow map* $\phi_t(x;\theta)$ whose velocity is recovered by automatic differentiation in t and which is trained by minimizing the residual of the flow ODE once, in the manner of a physics-informed neural network (PINN). Generation becomes a single network evaluation $\phi_1(x_0;\theta)$, $x_0 \sim p_0$. The integration work moves from sampling time, where it recurs for every sample, to training time, where it is paid once. The construction turns on one distinction that is easy to get wrong — flow matching supplies an unbiased velocity target at the network's *input*, whereas the flow ODE constrains the velocity at its *output*. Two naive substitutions of a flow map into the flow-matching loss that have exactly computable and degenerate minimizers turn to be wrong. We resolve the issue by incorporating the flow ODE residual at network output into the loss, penalized by the terminal conditional at distribution level, and hence propose a physics-informed flow matching based on PINN style solving (FlowPINN). Using benchmark examples of checkerboard and eight-Gaussian, and heavy-tailed funnel distributions, we demonstrate that one-step generation by FLOWPINN reaches the accuracy (measured by energy distance) that is worth multiple steps of Euler solver or Runge-Kutta method in the classical CFM. In high-dimensional image generation problems (MNIST and CIFAR-10), FLOWPINN matches a hundred-step solve to within the hardware noise band while sampling two orders of magnitude faster, and outperforms the state-of-the-art (SOTA) models including shortcut and meanflow.

1 INTRODUCTION

Continuous-time generative models including diffusion models, flow matching, stochastic interpolants (Ho et al., 2020; Song et al., 2021; Lipman et al., 2023; Albergo & Vanden-Eijnden, 2023) learn a time-dependent velocity field whose flow transports a simple reference law onto the data distribution. Training is simulation-free and remarkably stable; however, sampling is neither. Each sample is produced by numerically integrating an ODE, the number of function evaluations (NFE) spent there dominates the cost of deployment, and the same integration is repeated from scratch for every sample ever drawn. There has been effort in the literature to reduce NFE by better solvers (Lu et al., 2022; Zhang & Chen, 2023), distillation of a teacher's trajectory into a student taking larger steps (Salimans & Ho, 2022; Song et al., 2023), and a two-time map $\Phi_{s\to t}$ under a self-consistency or average-velocity identity (Boffi et al., 2024; 2025; Frans et al., 2025; Geng et al., 2025). All keep the velocity field as the primary learned object: a *flow-map* is obtained from it by integration, by distilling somebody else's integration, or by a consistency condition that encodes integration implicitly.

We start from the other end in this paper. The flow map is the solution to a differential equation which can be *solved* rather than integrated. We let a network represent the map itself,

$$\phi_t(y;\theta) \;=\; y + t\, f_\theta(t,y), \tag{1}$$

recover its velocity $\dot{\phi}_t = \frac{d}{dt}\phi_t$ with one forward-mode automatic-differentiation pass, and train by minimizing the residual of the flow ODE at collocation points — the recipe of a physics-informed neural network (PINN) (Lagaris et al., 1998; Raissi et al., 2019). Because (1) implies $\phi_0(y;\theta) = y$ identically for every θ, the PINN initial condition is by construction satisfied rather than penalized.

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.

Generation is $\phi_1(x_0;\theta)$ with $x_0 \sim p_0$ in one evaluation with no numerical solver or schedule. The ODE is solved once, by minimization, and that solve is amortized over every sample drawn thereafter.

One subtlety emerges when developing a method: the conditional flow-matching (CFM Lipman et al., 2023) regression supplies an unbiased velocity target *at the point fed to the network*, whereas the flow ODE constrains the velocity *at the point the network returns*. Substituting $\dot{\phi}_t(x_0;\theta)$ for v_t in the CFM loss makes the resulting flow map collapse in closed form to a blob of radius $\sigma_{\min}$ at the data mean, equivalent to a single explicit Euler step. Replacing it with $\dot{\phi}_t(x_t;\theta)$ does not define the flow map either – its minimizer integrates the marginal velocity at a frozen input that is overshooting. We fix the issue by imposing a balanced residual at the output in PINN-style loss. Therefore, we propose the phyiscs-informed neural flow matching methods termed FLOWPINN.

Relation to few-step and flow-map models The few-step literature is a sequence of increasingly direct attacks on the same integral of flow ODE. *Distillation* (Salimans & Ho, 2022; Song et al., 2023; Song & Dhariwal, 2024; Lu & Song, 2025) trains a student to reproduce a teacher's multi-step output, so the integration is performed expensively and then compressed. *Two-time flow maps* (Boffi et al., 2024; 2025; Kim et al., 2024) learn $\Phi_{s\to t}$ under $\Phi_{s\to t} = \Phi_{r\to t} \circ \Phi_{s\to r}$, encoding integration as an algebraic constraint on the map; *shortcut models* (Frans et al., 2025) condition on step size and enforce that one big step equals two small ones; *MeanFlow* (Geng et al., 2025) regresses an average velocity under the differential identity relating it to the instantaneous field.

While the works above constrain a relation between the map at **two times**, we constrain the map's own defining differential equation at **a single time** argument. Unlike distillation that requires a teacher or consistency that involves a composition, a residual only needs a right-hand side, which we take from a denoiser trained by exactly the objective that trains the baseline. Nothing is integrated anywhere in the pipeline. This framework nicely connects to MeanFlow by a two-time relaxation which includes our single-time formula as its diagonal, and hence we regard the single-time residual as the more primitive object. Parametrizing a solution rather than a derivative also has a precedent in *neural flows* (Biloš et al., 2021), which replace the neural ODE (Chen et al., 2018) by a network representing its solution operator. However, fitting that operator requires observed trajectories, whereas our fitting to a residual demands no pre-existing trajectory data. Our FLOWPINN methods make multifold contributions as follows.

- **A one-time residual formulation of the flow map** (§3), with three working variants that embed the initial condition in the architecture and impose the residual at the map's *output* with its right-hand side coming from a concurrently learned denoiser.
- **A diagnosis of the two failure modes** (§2.2) that "model the flow map and substitute it into the flow-matching loss" most naturally indicates, with their exact minimizers.
- **Two constructions that make the method practical** (§3.2, §3.3): a separable time basis representation of flow that removes the forward-mode pass and halves the peak memory consumption and a combined network for the map and its denoiser as two heads of one trunk at half the parameters.
- **Numerical advantages with evidence at three scales** (§4): from two-dimensional benchmarks where one evaluation is worth about eight of a solver's, to Neal's funnel at $d = 10/100$ dimensions with increasing complexity, and to MNIST ($d = 784$), where one evaluation matches a hundred-step solve on a UNet at $117\times$ less sampling time, and further CIFAR-10 ($d = 3072$) where FLOWPINN clearly outperforms SOTA models.

The remainder of the paper is organized as follows. CFM and PINN are reviewed in §1.1 and §1.2. §2 motivates the migration from two-stage solving to one-time minimization and explains two natural failures. We introduce FLOWPINN and its variants in §3. Then we present numerical results in §4 and conclude with a discussion on limitations and future work in §5.

1.1 CONDITIONAL FLOW MATCHING

Let $p_0 = \mathcal{N}(0, I)$ on $\mathbb{R}^d$ and q be the data distribution. FM models generate synthetic samples by transporting p_0 onto q along the flow of a time-dependent velocity field u_t, that is, by the *flow map* $\phi : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ solving the **flow ODE**

$$\dot{\phi}_t(x) \;=\; u_t\big(\phi_t(x)\big), \qquad \phi_0(x) = x, \tag{2}$$

and returning $\phi_1(x_0)$ for $x_0 \sim p_0$. CFM learns its right-hand side and then integrates it numerically, at k network evaluations per sample; §3 learns its *solution* and never integrates.

Let $\sigma_{\min} > 0$ and $c := 1 - \sigma_{\min}$. The conditional optimal-transport path of Lipman et al. (2023) is

$$\psi_t(x_0 \mid x_1) = \sigma_t\, x_0 + t\, x_1, \qquad \sigma_t = 1 - ct, \qquad u_t(x \mid x_1) = \frac{x_1 - c\, x}{\sigma_t}, \tag{3}$$

so $\sigma_0 = 1$, $\sigma_1 = \sigma_{\min}$, and the conditional target along the path is constant in t: $u_t(\psi_t(x_0 \mid x_1) \mid x_1) = x_1 - c\, x_0$. Because $u_t(x \mid x_1)$ is affine in x_1, the marginal field has a denoiser form we use throughout, where $p_t(\cdot|x_1) = \mathcal{N}(\cdot; tx_1, \sigma_t^2 I)$:

$$u_t(x) \;=\; \frac{D_t(x) - c\, x}{\sigma_t}, \qquad D_t(x) := \mathbb{E}[x_1 \mid x_t = x] = \frac{\int x_1\, p_t(x \mid x_1) q(x_1)\, dx_1}{p_t(x)}. \tag{4}$$

Note that $\|u_t\| \sim \sigma_t^{-1}$ diverges as $t \to 1$, while D_t always lies in the convex hull of $\mathrm{supp}(q)$. Conditioning the path target on the state it produced and using (4) gives the following **CFM identity**:

$$\mathbb{E}\big[\, x_1 - c\, x_0 \;\big|\; x_t = x \,\big] \;=\; u_t(x), \qquad x_t = \psi_t(x_0 \mid x_1). \tag{5}$$

There are two implications that are fundamental: (i) the conditional target is an *unbiased* estimate of the marginal field; and (ii) it holds *at the state that produced it* and nowhere else. The conditioning event is $x_t = x$, so the substitution is **licensed** (permitted by the CFM identity) at x on the conditional path and at no other point. Every failure in §2.2 and every design choice in §3 is a consequence of that second clause. Equation (5) is also why the CFM objective

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,x_0,x_1} \big\| v_t(x_t;\theta) - (x_1 - c\, x_0) \big\|^2, \qquad x_t = \psi_t(x_0 \mid x_1), \tag{6}$$

has minimizer $v_t(\cdot;\theta^\star) = u_t$ a.e., which is then substituted into (2) and integrated. Given N data points, one may also replace q by $\widehat{q} = \sum_{n=1}^N \delta_{x_1^{(n)}}$ in (4), giving the plug-in (Nadaraya–Watson) estimator

$$\widehat{D}_t(x) = \sum_{n=1}^N w_n(x,t)\, x_1^{(n)}, \qquad w_n(x,t) \propto \exp\Big(- \tfrac{\|x - t x_1^{(n)}\|^2}{2\sigma_t^2} \Big), \qquad \widehat{u}_t(x) = \frac{\widehat{D}_t(x) - c\, x}{\sigma_t}, \tag{7}$$

which is the exact marginal denoiser of $\widehat{q}$ and therefore defines a fully specified ODE. This serves as a useful control, as exploited in §4.

1.2 Physics-informed neural networks

PINNs (Lagaris et al., 1998; Raissi et al., 2019; Karniadakis et al., 2021) represent the solution of a differential equation by a network, form the equation's residual by automatic differentiation, and fit by minimizing that residual on collocation points, with initial and boundary conditions imposed either as penalties or built into the architecture as in (1). The solve is one global optimization rather than a time-stepping recursion, so the solution is available at any t without integrating up to it.

Written for the equation (2), the generic PINN objective for a trial flow map $\phi_t(\cdot;\theta)$ is

$$\mathcal{L}_{\mathrm{PINN}}(\theta) = \underbrace{\mathbb{E}_{(t,x)\sim\mu}\; w(t) \big\| \dot{\phi}_t(x;\theta) - u_t\big(\phi_t(x;\theta)\big) \big\|^2}_{\text{ODE residual}} + \lambda_{\mathrm{IC}}\, \mathbb{E}_{x\sim p_0} \big\| \phi_0(x;\theta) - x \big\|^2, \tag{8}$$

where μ is the collocation law on $[0,1] \times \mathbb{R}^d$ and $w(t) > 0$ a residual weight. Choices inside (8) need to be made including which μ (§2.1), what w (§3.1), how u_t is estimated *at the network's output* $\phi_t(x;\theta)$ (§3.1), and whether the second term is needed (soft constraint) or not (hard constraint) (§1).

Two known pathologies are directly relevant: (i) residual terms of very different magnitude compete and the loss is dominated by the stiffest region (Wang et al., 2021; 2022), and (ii) soft boundary penalties can create landscapes the network fails to traverse (Krishnapriyan et al., 2021). Both appear below in unusually explicit form as the correct w is known in closed form rather than estimated adaptively, and the initial condition is exact by construction. However, there is one caveat that differs from the classical setting of PINN and constitutes the source of all the difficulties in §3: *the coefficients of* (2) *are not given, since u_t is itself estimated from data, and $\phi_t(x;\theta)$ is in general not a point at which that estimate is cheap or unbiased.*

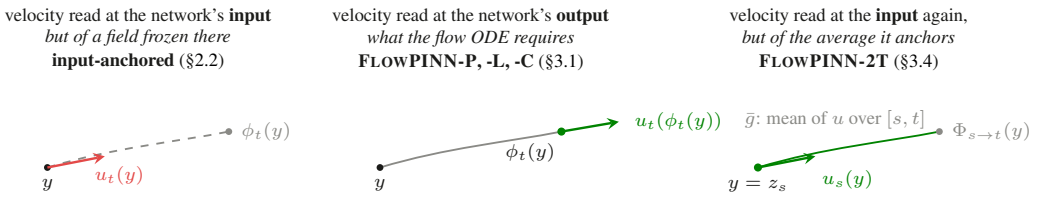

Figure 1: *Where is the velocity read?* The answer separates every formulation in this paper. The CFM identity (5) gives an unbiased target for u *at the state that produced it*, and nowhere else, so each design is fixed by which state it reads at. **Left:** at the point handed to the network, licensed but wrong because it pins down the integral of a field frozen there rather than the flow map (**input-anchored**, Proposition 2). **Middle:** at the point the network *returns*, which is what the flow ODE (2) constrains. Correct but unlicensed: $\phi_t(y)$ is off the conditional path, so u_t there must be *estimated*, which is what FLOWPINN-P, FLOWPINN-L and FLOWPINN-C exist to do (§3.1). **Right:** back at the input, but legitimate as the input now anchors an *average* rather than a frozen field; FLOWPINN-2T (§3.4) buys back the evaluation the middle panel pays a denoiser for, at the price of a constraint no longer pointwise in time. The first formulation of §2.2, **source-indexed**, is not a reading of this figure: it feeds the network x_0 against a target with no t-dependence, and fails for the separate reason in Proposition 1.

2 SOLVE THE FLOW ODE ONCE INSTEAD OF INTEGRATING IT PER SAMPLE

2.1 FROM TWO-STAGE SOLVING TO ONE-TIME MINIMIZATION

The standard pipeline has two stages with very different cost. Learning v_t is paid once and amortized over every sample; integrating (2) is paid *per sample*, with accuracy controlled only by the step count k. Nothing about the second stage is reused because the solver rediscovers the same trajectory family on every call, and its discretization error is a property of the sampler instead of anything learned.

Making the map the learned object merges the two stages into one. Because ϕ obeys (2) at every (t, y), the residual $\mathcal{R}_t(y;\theta) = \dot{\phi}_t(y;\theta) - u_t(\phi_t(y;\theta))$ is a training signal available at any collocation point, without any integration or any teacher trajectory to distil from; minimizing it once performs the solve, and sampling reads the solution off at $t = 1$. There are three features worth highlighting. *The differentiation is cheap and d-independent*: $\dot{\phi}_t$ is one forward-mode pass cost by the hidden width, not the input dimension (§D measures a $1.5\times$ wall-clock spread over a $32\times$ range of d). *The collocation law is the sampling law*: the residual is imposed at (t, x_0) with $x_0 \sim p_0$, precisely the inputs the sampler will present, so there is no train/test mismatch, the defect that ruins the second failure mode of §2.2. *The initial condition is free*: it is an identity in θ, so the term Krishnapriyan et al. (2021) identify as a common source of PINN optimization failure simply disappears.

2.2 TWO NATURAL FORMULATIONS THAT DO NOT WORK

The prescription "model the flow map ϕ directly and substitute $\dot{\phi}$ into the flow-matching loss" admits two possible constructions, depending on what to feed the network. Both fail in a closed form rather than in practice, based on which the diagnosis paves the way for the design of §3. The proofs and two additional consequences are deferred to Appendix A. Both minimizers are measured against their closed forms in §4.1.

Source-indexed regression. The sampling rule is $\phi_1(x_0;\theta)$, so the first thing to try is to feed the network the initial point and regress its time derivative on the baseline's conditional target:

$$\mathcal{L}_{\mathrm{src}}(\theta) = \mathbb{E}_{t,\,(x_0,x_1)\sim\pi}\big\|\dot{\phi}_t(x_0;\theta) - (x_1 - c\,x_0)\big\|^2. \tag{9}$$

Proposition 1 (indexing collapse)**.** *Let π be any coupling of p_0 and q. Under the hard initial condition* (1)*, any minimizer of* (9) *satisfies $\dot{\phi}_t(x_0;\theta^\star) = \mathbb{E}_\pi[x_1 \mid x_0] - c\,x_0$ for a.e. t, hence*

$$\phi_1(x_0;\theta^\star) \;=\; \sigma_{\min}\,x_0 + \mathbb{E}_\pi[x_1 \mid x_0]. \tag{10}$$

Under the independent coupling this is $\sigma_{\min}x_0 + m$ with $m = \mathbb{E}_q[x_1]$, so $\phi_{1\sharp}\,p_0 = \mathcal{N}(m, \sigma_{\min}^2 I)$: a blob of radius $\sigma_{\min}$ at the data mean, for every architecture, dataset and training budget.

Input-anchored regression. A natural repair feeds the network with the path point, which is the literal substitution of $\dot{\phi}_t(x_t;\theta)$ for v_t in (6):

$$\mathcal{L}_{\text{in}}(\theta) = \mathbb{E}_{t,x_0,x_1}\left\|\dot{\phi}_t(x_t;\theta) - (x_1 - c\,x_0)\right\|^2, \qquad x_t = \psi_t(x_0 \mid x_1). \tag{11}$$

Now the regression is sound because by (5) its minimizer is $\dot{\phi}_t(y;\theta^\star) = u_t(y)$, exactly as for the baseline field. What goes wrong is the map that pins down.

Proposition 2 (the exact minimizer of (11) is the frozen-state map)**.** *Suppose $\dot{\phi}_t(\cdot;\theta)$ under* (1) *can realize any square-integrable field. Then any minimizer $\theta^\star$ of* (11) *satisfies, p_t-almost everywhere,*

$$\phi_t(y;\theta^\star) \;=\; y + \int_0^t u_s(y)\,\mathrm{d}s \;=:\; \Psi_t(y), \tag{12}$$

the marginal velocity integrated at the frozen *input point y. Ψ_t solves $\dot{\Psi}_t(y) = u_t(y)$ and coincides with the flow map of* (2) *iff $u_s(\phi_s(y)) = u_s(y)$ for a.e. s — that is, only where the flow is stationary.*

Therefore, neither formulation addresses the actual defect, which is the one Figure 1 draws: both constrain $\dot{\phi}$ **at the point handed to the network**, and the flow ODE constrains it **at the point the network returns**. The lesson is that where the marginal velocity u_t **anchors** (defined in a certain state) matters. The CFM identity (5) requires that any candidate velocity u_t (read at a state the conditional path produced) must have $x_1 - c\,x_0$ as an unbiased estimator to be *licensed* (permitted) in the CFM loss (6). The input-anchored $\dot{\phi}_t(x_t;\theta)$ is licensed but wrong. §3 fixes the failure with the output-anchor (FLOWPINN-P/-L/-C) or the input-anchor with average-velocity (FLOWPINN-2T).

3 FLOWPINN

3.1 RESIDUAL AT THE OUTPUT, WITH A LEARNED RIGHT-HAND SIDE

Take the parametrization (1) and define the residual and its balanced form,

$$\mathcal{R}_t(x;\theta) = \dot{\phi}_t(x;\theta) - u_t\big(\phi_t(x;\theta)\big), \qquad \widetilde{\mathcal{R}}_t(x;\theta) = \sigma_t\,\dot{\phi}_t(x;\theta) + c\,\phi_t(x;\theta) - D_t\big(\phi_t(x;\theta)\big), \tag{13}$$

so that $\widetilde{\mathcal{R}} = \sigma_t\mathcal{R}$ by (4) and the two share the same support. Training minimizes

$$\mathcal{L}(\theta) \;=\; \underbrace{\mathbb{E}_{t\sim U[0,1],\,x_0\sim p_0}\left\|\widetilde{\mathcal{R}}_t(x_0;\theta)\right\|^2}_{\text{ODE residual}} + \lambda_{\text{term}}\,\mathcal{E}\big(\phi_1(\cdot\,;\theta)_\sharp\,p_0,\; q\big), \tag{14}$$

where $\phi_1(\cdot\,;\theta)_\sharp\,p_0$ is the law of $\phi_1(x_0;\theta)$ for $x_0 \sim p_0$ and $\mathcal{E}$ is the energy distance (Székely & Rizzo, 2013), which vanishes if and only if the two laws agree; in practice its V-statistic on the two minibatches (line 7 of Algorithm 1; Appendix C). No initial-condition term appears, since (1) makes it an identity. The equation is solved once, by minimization, and never integrated. Note that the choice of $\widetilde{\mathcal{R}}_t$ over $\mathcal{R}_t$ is because $\|\mathcal{R}_t\| \sim \sigma_t^{-1}$ outweights the collocation points near $t = 1$ more than $t = 0$ and $\widetilde{\mathcal{R}}_t$ balances their weights. The adoption of terminal condition in distribution as opposed to pointwise is because the latter, $\phi_1(x_0) = x_1$, is the source-indexed regression of (10) and collapses. What remains in the recipe is to supply the right-hand side of Equation (13). It needs D_t at $\phi_t(x_0;\theta)$, which generally lies off the conditional path, where flow matching offers no unbiased estimate.

FLOWPINN-P: excellent control, not a generative model The *plug-in* choice $D_t = \widehat{D}_t$ of (7) is the exact marginal denoiser of $\widehat{q}$, so the ODE being solved is fully specified and its solution known as the flow of $\widehat{u}_t$ pushes p_0 onto $\mathrm{KDE}_{\sigma_{\min}}(\widehat{q})$. We denote it as FLOWPINN-P, which makes an excellent *control*, since accuracy can then be measured against a reference integration of the same field rather than only by sample quality (§4.1). However, it is not a generative proposal as it memorizes by construction and costs $O(Nd)$ per collocation point. §E shows that it does not survive dimension.

3.2 FLOWPINN-L: LEARNING A DESNOISER

The *learned* choice trains a denoiser D_ψ concurrently by $\mathbb{E}\|D_\psi(t,x_t) - x_1\|^2$, which is (6) up to (4), the baseline's own objective, and uses $\mathrm{sg}[D_\psi]$ in (13), touching no training point at residual time, at $O(1)$ per collocation point. We denote this as **FLOWPINN-L, our proposed method**. Algorithm 1 describes one training step that the two branches, the regression batch on the conditional path and the collocation batch on the source law queried by the sampler, draw *independently*.

Separable time: making $\dot{\phi}$ free. Forward-mode automatic differentiation gives $\dot{\phi}$ for one extra forward pass that costs $O(1)$ due to the *single scalar* time input and never scales with d. However, the price paid is the memory demanded high when running the network on dual numbers that calls a doubled graph to back-propagate through. The cost is an artifact of the parametrization, not of the objective, and can be reduced by factoring the time dependence:

$$\phi_t(x) \;=\; x + \sum_{k=1}^{K} b_k(t)\, g_k(x), \qquad \dot{\phi}_t(x) \;=\; \sum_{k=1}^{K} b_k'(t)\, g_k(x), \tag{15}$$

with $g = (g_1, \ldots, g_K)$ the backbone carrying *no time input*, K output channels per state channel, and $b : [0,1] \to \mathbb{R}^K$ a small network of t alone whose derivative is written out rather than differentiated (Appendix C). With such time separation representation, we have: (i) exact and free derivative; (ii) structural initial condition ($b_k(0) = 0$ by construction); and (iii) collocation free in time. All vector experiments use $K = 8$ and the image runs $K = 16$ in §4.

3.3 FLOWPINN-C: ONE NETWORK, TWO HEADS

FLOWPINN-L carries two networks of identical shape, neither evaluated together nor sharing weights. However, their targets are close relatives that the map's free part must represent $x_1 - c\,x_0$ and the denoiser $\mathbb{E}[x_1 \mid x_t]$, both answering "given this point, where in the data did it come from" at different arguments. FLOWPINN-C is the same two functions off one body (as illustrated in Figure 5) with the objective of §3.1 otherwise unchanged and D_ψ replaced by the denoiser head. Under (15) the trunk emits $K + J$ time-free fields, K for the map and J for the denoiser, each contracted against its own time basis. We take $J = K$. The difference between FLOWPINN-L and FLOWPINN-C is a consequence of the sharing, and the control $n_{\text{shared}} = 0$ reproduces FLOWPINN-L bit-identically.

3.4 FLOWPINN-2T: THE TWO-TIME RELAXATION, AND HOT IT CONNECTS TO MEANFLOW

The learned denoiser is one way to supply u_t at the network's output; a second time argument removes the need entirely. Let $\bar{g}(y, s, t; \theta)$ denote the average velocity over $[s, t]$ along the trajectory passing through y at time s,

$$\bar{g}(y, s, t) \;=\; \frac{1}{t-s} \int_s^t u_\tau(z_\tau)\, \mathrm{d}\tau, \qquad \dot{z}_\tau = u_\tau(z_\tau), \quad z_s = y, \tag{16}$$

and set $\Phi_{s\to t}(y; \theta) = y + (t-s)\, \bar{g}(y, s, t; \theta)$, so that $\Phi_{s\to t}(y) = z_t$ with $\bar{g}$ in (16).

For a vector $v \in \mathbb{R}^d$, the following the derivative of $\bar{g}$ along the direction $(v, 1, 0)$ of (y, s, t) can be computed by one forward-mode pass, at the cost of one extra evaluation of $\bar{g}$:

$$(\mathcal{D}_v\, \bar{g})(y, s, t) \;:=\; \partial_s \bar{g}(y, s, t) \;+\; \big(v \cdot \nabla_y\big)\, \bar{g}(y, s, t) \;\in\; \mathbb{R}^d. \tag{17}$$

Proposition 3 (average-velocity identity, anchored at earlier time)**.** *For $s < t$, u the marginal field,*

$$\bar{g}(y, s, t) \;=\; u_s(y) \;+\; (t-s)\, \big(\mathcal{D}_{u_s(y)}\, \bar{g}\big)(y, s, t). \tag{18}$$

On the diagonal $s = t$ the second term vanishes and (18) *reduces to $\bar{g}(y, t, t) = u_t(y)$, i.e. the marginal vector field.*

Why anchoring at earlier time? The identity comes from differentiating $(t-s)\, \bar{g}(z_s, s, t) = z_t - z_s$ in one of its two times, and the choice decides where u is evaluated: in t it is read at z_t, the endpoint the map *returns*; in s, at $y = z_s$, the endpoint the map is *handed*. Only the second is usable because by (5), $y = z_s = \psi_s(x_0 \mid x_1)$ is a genuine path state during training, and the CFM identity licenses replacing $u_s(y)$ by the unbiased target $x_1 - c\,x_0$, which does not hold at z_t. The rule is therefore not "earlier" as such but **anchor at whichever endpoint carries the network's spatial input**, which in the convention of (3) is the source end s. That is precisely the ingredient the two failed formulations of §2.2 lack, and the one §3.1 has to buy a denoiser to obtain. Training minimizes

$$\mathcal{L}_{2\mathrm{T}}(\theta) = \mathbb{E}_{s\le t,\, x_0, x_1} \Big\| \bar{g}_\theta(x_s, s, t) \;-\; \mathrm{sg}\Big[\, \tilde{v} \;+\; (t-s)\, \big(\mathcal{D}_{\tilde{v}}\, \bar{g}_\theta\big)(x_s, s, t)\Big]\Big\|^2, \quad \tilde{v} := x_1 - c\,x_0, \tag{19}$$

with $x_s = \psi_s(x_0 \mid x_1)$ and a fixed fraction of each batch on the diagonal $s = t$. Sampling is $\Phi_{0\to 1}(x_0; \theta) = x_0 + \bar{g}_\theta(x_0, 0, 1)$ in again one evaluation, and now the semigroup condition is expressible and can be added as a further residual. We denote Algorithm 2 as FLOWPINN-2T.

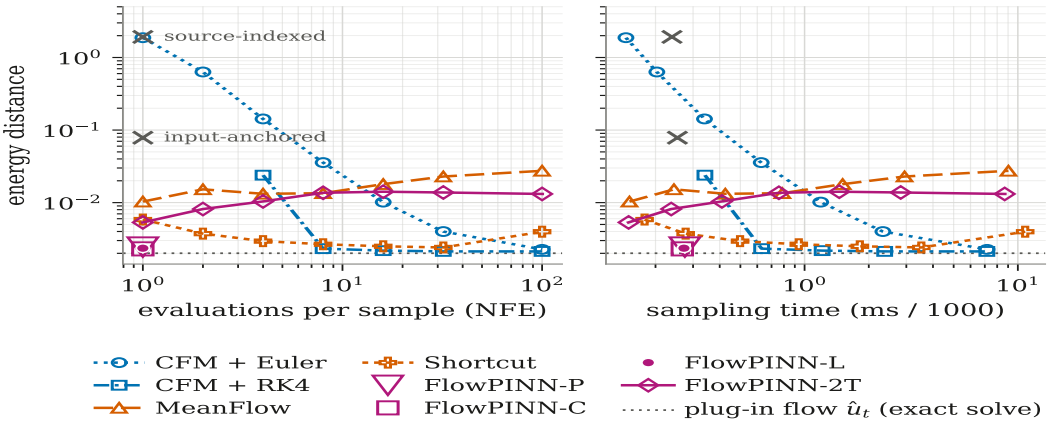

Figure 2: Cost against quality on checkerboard, seed-averaged over ten seeds; the numbers are Table 3's, over the whole sampling grid. **Left:** network evaluations per sample. **Right:** measured sampling wall-clock, the quantity actually paid. Color carries the family and marker the member. The value of a flow map is the left-hand edge of either panel, where the solver cannot reach; the three single-time maps agree to the fourth decimal, so their markers are nested rather than displaced.

One can refer to Appendix B on the connection to and distinction from MEANFLOW. Finally, Table 2 summarizes all the aforementioned methods including the baseline, the two failure formulations of §2.2, and the four variants of this section, each with the objective, model size and computational cost.

4 EXPERIMENTS

In this section, we compare our proposed FLOWPINN methods with the baseline CFM (Lipman et al., 2023), SOTA models including MEANFLOW (Geng et al., 2025) and SHORTCUT (Frans et al., 2025) using 2d benchmarks, medium dimensional ($d = 10, 100$) funnel distribution, and higher dimensional image generation such as MNIST ($d = 784$) and CIFAR-10 ($d = 3072$). For evaluation metrics, we adopt the energy distance $\mathcal{E}$, and the memorization index MEMO in the first two vector applications and the Fréchet Inception Distance (FID), and the Inception Score (IS) for image generation problems. See Appendix §C for more details of setups. The computer codes will be released upon publication.

4.1 THE 2D BENCHMARKS: IS ONE EVALUATION WORTH WHAT A SOLVER COSTS?

We first consider two benchmarks: checkerboard and eight-Gaussians (Appendix D.1). In Figure 2, all three single-time FMs, FLOWPINN-P, FLOWPINN-L, FLOWPINN-C, achieve the lowest energy distance at NFE 1, the same precision that CFM requires a hundred solver steps to reach. Both MEANFLOW and SHORTCUT stay above this level with MEANFLOW getting close at NFE ≥ 10. FLOWPINN-2T does not match them in this example. Table 3 supports these findings – FLOWPINN methods attain $\mathcal{E} = 0.0024, 0.0024, 0.0023$ at NFE 1, against 0.0021 at the best point by CFM + RK4 at NFE 100 on checkerboard, and $0.0040, 0.0039, 0.0040$ against 0.0036 on eight-Gaussians.

Both source-indexed and input-anchored maps fail with their closed forms – the former reaches 1.9217 and 2.8650 versus the one-step Euler baseline's 1.8744 and 2.7461, as predicted by Corollary 1; the latter is $34\times$ and $8.5\times$ worse than the best one-evaluation map. Figures 7 and 8 indicate that both failures transport the mass to the wrong place rather than being overfitted. Appendix §E conducts an ablation study on dimensionality and explains why FLOWPINN-P fails with increasing dimensions.

4.2 NEAL'S FUNNEL: A TARGET WITH NO SINGLE SCALE

Neal's funnel distribution (Neal, 2003), $f(x, \nu) = \mathcal{N}_d(x; 0, e^{\nu/2} I)\mathcal{N}(\nu; 0, 3)$, features a cusp on one end and a fat tail on the other, which poses a challenge for generative algorithms. The funnel's conditional scale varies over $e^{\pm 6}$, so a fixed-step solver is mismatched somewhere *by construction*. This is exactly the regime where solving once by residual differs from integrating.

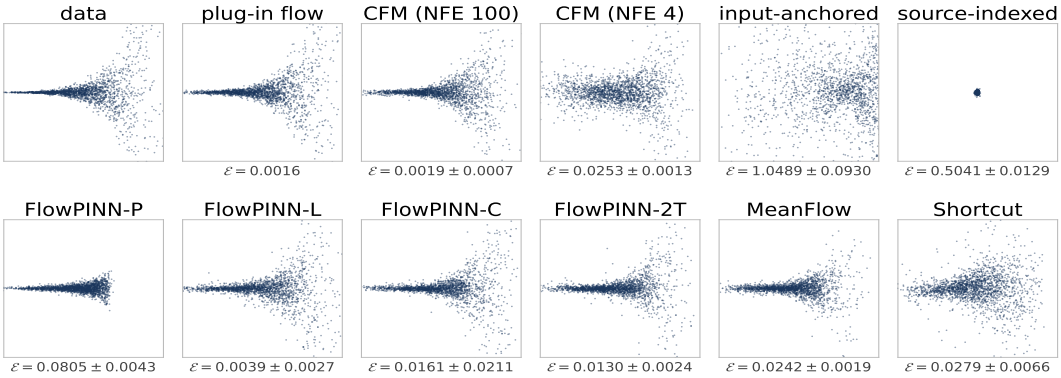

Figure 3: Neal's funnel at $d = 10$, whitened and projected on two coordinates (Appendix C). Data, an exact solve of the empirical field and CFM at NFE=100 recovers the cusp-funnel pattern, while at NFE=4 has neither features, only a smear between them. The two failure modes lose the target altogether: input-anchored fills the box, source-indexed is a point. FLOWPINN-P keeps the cusp and truncates the funnel; FLOWPINN-L, FLOWPINN-C, FLOWPINN-2T, MEANFLOW and SHORTCUT recover a funnel at one evaluation, in the order Table 4 gives.

Table 1: MNIST on the UNet backbone: mean ± s.d. over five seeds at 100k steps, every method on the same backbone, budget and optimiser. MEMO is the memorisation ratio, healthy at ≈ 1; *ms/1k* is sampling wall-clock on an A100, and clf-FID uses one LeNet for every row of both tables (Appendix C). The last row is held-out real data through the identical pipeline: a floor, not a method.

method	NFE	FID↓	clf-FID↓	IS↑	MEMO	ms/1k↓	params
CFM + Euler	1	299.7 ±4.9	570.0 ±1.0	1.159 ±0.009	1.013 ±0.001	127	7.28M
CFM + Euler	4	59.50 ±1.03	103.4 ±1.2	1.937 ±0.022	0.999 ±0.001	508	7.28M
CFM + midpoint	100	**24.23** ±0.12	**0.79** ±0.14	2.041 ±0.015	0.995 ±0.004	12692	7.28M
FLOWPINN-P	1	209.6 ±3.0	732.5 ±8.2	1.250 ±0.083	0.993 ±0.022	108	7.21M
FLOWPINN-L	1	**24.29** ±0.75	1.60 ±0.21	2.034 ±0.009	0.997 ±0.004	109	14.49M
FLOWPINN-C	1	31.05 ±1.30	**1.34** ±0.16	2.001 ±0.016	0.997 ±0.003	109	7.22M
FLOWPINN-2T	1	26.14 ±0.30	6.40 ±0.57	2.014 ±0.009	0.998 ±0.004	108	7.29M
FLOWPINN-2T	4	24.49 ±0.59	5.05 ±0.64	2.008 ±0.007	0.997 ±0.005	432	7.29M
MEANFLOW	1	39.49 ±3.15	9.80 ±0.83	2.016 ±0.015	0.996 ±0.004	127	7.29M
MEANFLOW	4	26.29 ±0.58	9.17 ±1.14	2.021 ±0.013	0.998 ±0.004	508	7.29M
SHORTCUT	1	44.33 ±3.58	11.00 ±1.30	2.180 ±0.013	0.996 ±0.004	108	7.29M
SHORTCUT	4	33.93 ±3.19	7.51 ±0.94	2.147 ±0.018	0.996 ±0.004	433	7.29M
real data (floor)	–	1.61	0.41	2.096	0.996	–	–

Figure 3 compares the samples generated by various generative models against direct sampling ($v \sim \mathcal{N}(0, 3^2)$ and $x_i \mid v \sim \mathcal{N}(0, e^v)$) for $d = 10$. Most models recover the cusp-funnel feature except two failure modes. CFM at NFE 4, FLOWPINN-P, and SHORTCUT perform poorly with unfaithful recovery of the distribution. Table 4 demonstrates the advantage of FLOWPINN methods that FLOWPINN-L reaches the lowest $\mathcal{E} = 0.0039$, as does CFM under RK4 at *eight* evaluations (0.0019 at 100), with MEANFLOW 6.2× and SHORTCUT 7.2× the record at $d = 10$. At $d = 100$, FLOWPINN-2T's best $\mathcal{E} = 0.1430$ is slightly better than CFM's best 0.1436 for NFE=2. Refer to Table 4 and Appendix D.2 for more results.

4.3 MNIST

In this $d = 784$ dimensional hand written digits problem, Table 1 shows that FLOWPINN-L at one evaluation reaches FID 24.29 ± 0.75 comparable to CFM's 24.23 ± 0.12 at a hundred, while sampling 117× faster (109 against 12,692 ms per thousand images)! Against CFM at a matched *four* evaluations (59.50 ± 1.03) it is 2.45× better, and Figure 9 illustrates that at four evaluations the solver has not yet resolved all the digits. This strongly supports the notion that *the flow map*

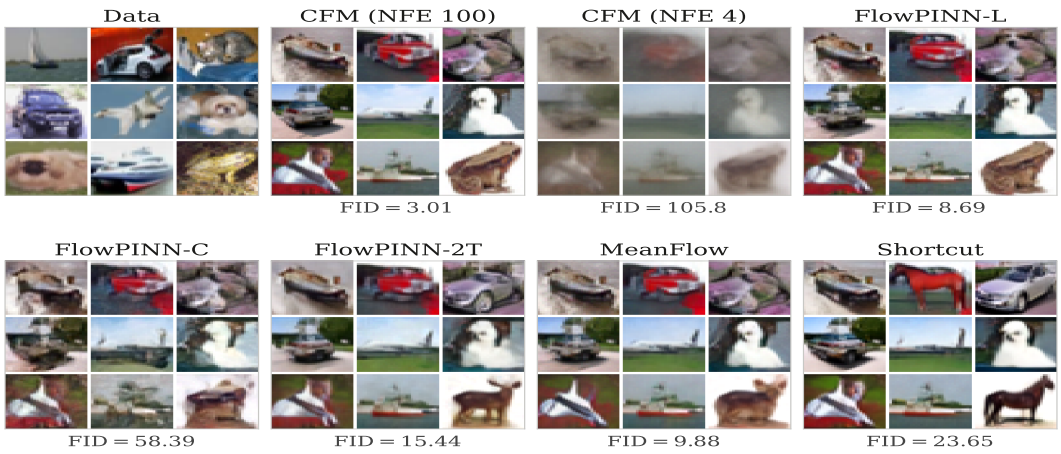

Figure 4: CIFAR-10 samples on **nine shared draws**: every generated panel decodes the same nine source vectors. *Data* are real images, whose nine cells are class-matched to others (Appendix C).

buys the solver's quality at the solver's cheapest possible budget. Note that FLOWPINN-L beats MEANFLOW (39.49 ± 3.15) and SHORTCUT (44.33 ± 3.58) at NFE=1 by factors of 1.6 and 1.8 respectively. FLOWPINN-P (209.58 ± 3.02) fails in this high dimensional generation problem, as predicted by Proposition 4. See Appendix D.3 for extended results.

4.4 CIFAR-10

Lastly, we consider CIFAR-10 dataset at $d = 3072$ dimensions. As illustrated in Figure 4, with nine shared random noises, FLOWPINN-L at NFE=1 generates sample images that are the most consistent with CFM at NFE=100, the gold-standard. Corresponding to the frog picture generated by CFM and FLOWPINN-L, FLOWPINN-2T has a deer, MEANFLOW has a fox, and SHORTCUT has a horse. CFM has not fully resolved the images at NFE=4 and FLOWPINN-C does not survive this scale, probably due to loss of capacity by sharing trunks of the network. Table 6 shows that at $d = 3072$ no one-evaluation map matches the solver (best FID 3.01 at NFE=100), but against the solver at a budget it can be run at the margin grew. FLOWPINN-L reaches 8.69 against MEANFLOW's 9.88 and SHORTCUT's 23.65, at 29 A100-hours of training against MEANFLOW's 33.

5 DISCUSSION

In this paper, we propose a novel FM framework that learns the flow map as the solution of its own differential equation, by minimizing a balanced residual once rather than recovering that solution per sample with a numerical solver. The resulting sampler costs one network evaluation. Numerical evidences by benchmarks and image generation have demonstrated that the proposed FLOWPINN models match the best a numerical solver reaches at higher budget, and beats two few-step SOTA models from the literature trained on the same backbone and budget.

Limitations. *The parity result depends on the backbone.* On the UNet, one evaluation matches a hundred; on a DiT of the same size every one-evaluation map loses between 41% and 68%, while the solver baseline moves by at most 8%. We hypothesize that adaLN-Zero makes time conditioning load-bearing in a way the separable trunk disrupts and will test a sweep of K on the DiT in future. *The map returns no uncertainty*, unlike a probabilistic ODE solver (Hennig et al., 2015; Tronarp et al., 2021), which is a real gap for a method whose premise is that the solve happened correctly.

Extensions. A last-layer Bayesian treatment would return a posterior over $\phi_1(x_0)$, porting the probabilistic-solver literature's calibration results (Bosch et al., 2021) to a trial function whose resolution follows the data. Because the residual needs only a denoiser, a flow map can also be fitted against a *frozen pretrained* D_ψ, a post-hoc one-step sampler for existing checkpoints, and the cheapest route to scale, since it reuses a model that has already been trained.

AI USE STATEMENT

In this work, we used generative AI tools for polishing presentation of the paper (refining sentences and paragraphs). We have not used generative AI tools for writing the whole paper. In addition, we used generative AI tools for assistance in coding and debuging. We have reviewed all AI-assisted work. For example, the LLM-generated code was verified and tested for correctness by all co-authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Appendix C specifies architectures, optimizer settings, benchmarks, metrics and seeds for every number reported; Appendix A proves the four propositions of the main text and states the corollaries they imply; Appendix D gives the full tables and figures from which the main-text selections are drawn, organized by target. All results are reproducible using `python` codes that will be released after publication. We include a `jupyter` notebook of a demonstrative example in the submitted supplementary materials.

REFERENCES

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

Marin Biloš, Johanna Sommer, Syama Sundar Rangapuram, Tim Januschowski, and Stephan Günnemann. Neural flows: Efficient alternative to neural ODEs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

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.

Nicholas M. Boffi, Michael S. Albergo, and Eric Vanden-Eijnden. How to build a consistency model: Learning flow maps via self-distillation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Nathanael Bosch, Philipp Hennig, and Filip Tronarp. Calibrated adaptive probabilistic ODE solvers. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

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

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Philipp Hennig, Michael A. Osborne, and Mark Girolami. Probabilistic numerics and uncertainty in computations. *Proceedings of the Royal Society A*, 471(2179), 2015.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

George E. Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ODE trajectory of diffusion. In *International Conference on Learning Representations (ICLR)*, 2024.

Aditi S. Krishnapriyan, Amir Gholami, Shandian Zhe, Robert M. Kirby, and Michael W. Mahoney. Characterizing possible failure modes in physics-informed neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Isaac E. Lagaris, Aristidis Likas, and Dimitrios I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. In *International Conference on Learning Representations (ICLR)*, 2025.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Radford M. Neal. Slice sampling. *The Annals of Statistics*, 31(3), jun 2003.

Maziar Raissi, Paris Perdikaris, and George E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations (ICLR)*, 2022.

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *International Conference on Learning Representations (ICLR)*, 2024.

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 (ICLR)*, 2021.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning (ICML)*, 2023.

Gábor J. Székely and Maria L. Rizzo. Energy statistics: A class of statistics based on distances. *Journal of Statistical Planning and Inference*, 143(8):1249–1272, 2013.

Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research (TMLR)*, 2024.

Filip Tronarp, Simo Särkkä, and Philipp Hennig. Bayesian ODE solvers: The maximum a posteriori estimate. *Statistics and Computing*, 31(3), 2021.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.

Sifan Wang, Xinling Yu, and Paris Perdikaris. When and why PINNs fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, 449:110768, 2022.

Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. In *International Conference on Learning Representations (ICLR)*, 2023.

A PROOFS AND COROLLARIES

Throughout, $c = 1 - \sigma_{\min}$, $\sigma_t = 1 - ct$, $\psi_t(x_0 \mid x_1) = \sigma_t x_0 + t x_1$ and $D_t(x) = \mathbb{E}[x_1 \mid x_t = x]$, so that $u_t(x) = (D_t(x) - cx)/\sigma_t$ by (4). The four propositions are stated where they are used: Propositions 1 and 2 in §2.2, Proposition 3 in §3.4, and Proposition 4 in §E. This appendix proves them and records the two corollaries they imply.

A.1 PROPOSITION 1: THE SOURCE-INDEXED MINIMIZER

Proof of Proposition 1. The regression target $x_1 - c\,x_0$ in (9) has no t-dependence, so the pointwise minimizer in t is the t-independent conditional mean $\mathbb{E}_\pi[x_1 \mid x_0] - c\,x_0$. Integrating from $\phi_0(x_0) = x_0$ and using $1 - c = \sigma_{\min}$ gives (10). Under $\pi = p_0 \otimes q$ the conditional mean is the unconditional mean m, and $x_0 \mapsto \sigma_{\min} x_0 + m$ pushes $\mathcal{N}(0, I)$ to $\mathcal{N}(m, \sigma_{\min}^2 I)$. □

Corollary 1 (the collapsed map is one-step Euler)**.** *Under the independent coupling, the map of* (10) *is exactly the result of a single explicit Euler step on* (2) *from $t = 0$ to $t = 1$. Hence the source-indexed neural flow map, at its optimum, is the NFE-1 Euler baseline.*

Proof. At $t = 0$ the conditional law of x_1 given x_0 is q itself, so $D_0 \equiv m$ and $u_0(x) = m - c\,x$ by (4). One explicit Euler step returns $x_0 + u_0(x_0) = \sigma_{\min} x_0 + m$, which is (10). □

A.2 PROPOSITION 2: THE INPUT-ANCHORED MINIMIZER

Proof of Proposition 2. Pointwise in (t, y) the integrand of (11) is minimized by the conditional mean $\mathbb{E}[\,x_1 - c\,x_0 \mid x_t = y,\, t\,]$. On the path $x_0 = (y - t x_1)/\sigma_t$, so the x_0-dependence is eliminated and this conditional mean equals $(D_t(y) - c\,y)/\sigma_t = u_t(y)$ by (4). Integrating $\dot\phi_t(y) = u_t(y)$ subject to the hard initial condition $\phi_0(y) = y$ gives (12). The flow ODE (2) instead requires $\dot\phi_t(y) = u_t(\phi_t(y))$; the two right-hand sides agree almost everywhere exactly under the stated stationarity condition. □

Corollary 2 (logarithmic overshoot)**.** *If $D_s(y) \equiv \bar D$ is constant in s, then*

$$\Psi_1(y) \;=\; y + (\bar D - c\,y)\,\frac{-\log \sigma_{\min}}{1 - \sigma_{\min}},$$

an affine map with gain $\log(1/\sigma_{\min}) \gg 1$ that diverges as $\sigma_{\min} \to 0$, whereas the true flow map displaces y by $O(\operatorname{diam}\operatorname{supp} q)$ independently of $\sigma_{\min}$. At $\sigma_{\min} = 0.02$ the gain is 3.99.

Proof. Substitute the constant $\bar D$ into (4) and integrate (12). □

Two consequences of (10)**.** First, the ceiling of *any* source-indexed regression is the barycentric projection $x_0 \mapsto \mathbb{E}_\pi[x_1 \mid x_0]$ of whatever coupling is used, never the flow of u_t. This predicts, correctly, that replacing the independent coupling by a minibatch optimal-transport coupling (Tong et al., 2024) repairs the collapse without ever producing a flow map. The OT plan is generically deterministic, so $\mathbb{E}_\pi[x_1 \mid x_0]$ becomes a Monge map. An earlier, shorter run of ours behaved that way, improving the distance by orders of magnitude while leaving the support statistics essentially unrepaired; that ablation has not been re-run at the present budget, so we state the consequence as what (10) predicts rather than as something this paper measures. Second, it is why a *pointwise* terminal condition $\phi_1(x_0; \theta) = x_1$, one of the extra PINN terms one might naturally reach for, must not be imposed under an independent coupling: it is precisely the regression that collapses. §3.1 uses a distributional terminal condition instead.

Why the input-anchored network does not look as bad as (12)**.** Training visits (t, x_t) with $x_t \sim p_t$; sampling evaluates (t, x_0) for all $t \in [0, 1]$ with x_0 held *fixed*. For t near 1 the pair (t, x_0) is exponentially unlikely under p_t, so $\dot\phi_t(x_0; \theta)$ there is pure extrapolation, governed by an implicit smoothness bias of the architecture, which is an uncontrolled quantity that improves or degrades with capacity for reasons unrelated to the target. Figure 7 shows what it produces: a diffuse cloud, not a blurred version of the target.

A.3 Proposition 3: the average-velocity identity

Proof of Proposition 3. By definition $(t-s)\,\bar{g}(z_s, s, t) = z_t - z_s$ along the trajectory. Fix t and differentiate in s, using $\dot{z}_s = u_s(z_s)$ and the fact that z_t does not depend on s:

$$-\bar{g} + (t-s)\big[\partial_s \bar{g} + u_s(z_s) \cdot \nabla_y \bar{g}\big] \;=\; -u_s(z_s),$$

which rearranges to (18). Setting $s = t$ kills the bracket and leaves $\bar{g}(y, t, t) = u_t(y)$. □

This is the MeanFlow identity of Geng et al. (2025) written in the forward-time convention of (3); their derivation differentiates at the later time, which is the correct anchor for the noise-to-data direction, and the two are related by time reversal.

A.4 Proposition 4: softmax collapse in d

Proof of Proposition 4. Expand

$$\|x_t - t\,x_1^{(n)}\|^2 = \|\sigma_t x_0 + t\Delta_n\|^2 = \sigma_t^2\|x_0\|^2 + 2t\sigma_t\langle x_0, \Delta_n\rangle + t^2\|\Delta_n\|^2,$$

subtract the same expression at $n = \star$, where $\Delta_\star = 0$, and divide by $2\sigma_t^2$; this gives (23). The scaling claims follow from the block-independence of the product benchmark, which makes $\|\Delta_n\|^2$ a sum of $\Theta(d)$ i.i.d. terms, together with a standard concentration bound on the inner product $\langle x_0, \Delta_n\rangle$. The softmax over N candidates concentrates once the smallest gap exceeds $\log N$, which by (23) happens at (24). □

Measured. The value of t/σ_t at which the effective neighbor count $N_{\text{eff}} = 1/\sum_n w_n^2$ first falls below 2, evaluated at genuine path states $x_t \sim p_t$, is 1.410, 0.412 and 0.279 at $d = 8, 32, 64$. Doubling d from 32 to 64 shrinks it by a factor 1.48, against the predicted $\sqrt{2} \approx 1.41$. At $d = 2$ it never collapses within the sampled range. One consequence is that in high dimension the sharpening of D_t is not a $t \to 1$ phenomenon. At $d = 64$ the plug-in denoiser is already a nearest-neighbor selector by $t \approx 0.219$.

Algorithm 1 FLOWPINN-L: one training step. Here $\sigma_t = 1 - (1 - \sigma_{\min})t$, $c = 1 - \sigma_{\min}$, and $\mathrm{sg}[\cdot]$ is the stop-gradient. The two branches draw *independently*: $i = 1, \dots, B$ indexes the regression minibatch, which lives on the conditional path, and $j = 1, \dots, M$ the collocation batch, which lives on the source law the sampler will query. Nothing pairs i with j, and M need not equal B.

Require: flow map f_θ, denoiser D_ψ, regression batch B, collocation batch M, weight $\lambda_{\mathrm{term}} \geq 0$

Denoiser branch — the CFM objective, reparametrized by (4).

1: $x_1^i \sim \widehat{q}, \quad x_0^i \sim p_0, \quad t^i \sim U[0,1], \quad x_t^i \leftarrow \sigma_{t^i} x_0^i + t^i x_1^i$ $\qquad i = 1, \dots, B$

2: $\mathcal{L}_D \leftarrow \frac{1}{B} \sum_{i=1}^{B} \| D_\psi(t^i, x_t^i) - x_1^i \|^2$

Flow-map branch — collocation on $U[0,1] \otimes p_0$.

3: $z^j \sim p_0, \quad s^j \sim U[0,1]$ $\qquad j = 1, \dots, M$

4: $\phi^j \leftarrow \phi_{s^j}(z^j; \theta) = z^j + s^j f_\theta(s^j, z^j), \quad \dot{\phi}^j \leftarrow \frac{d}{dt}\big|_{t=s^j} \phi_t(z^j; \theta)$ $\qquad$ *one forward-mode pass*

5: $\widetilde{\mathcal{R}}^j \leftarrow \sigma_{s^j} \dot{\phi}^j + c\, \phi^j - \mathrm{sg}\big[D_\psi(s^j, \phi^j)\big]$ $\qquad$ *residual read at the* output ϕ^j

6: $\mathcal{L}_R \leftarrow \frac{1}{M} \sum_{j=1}^{M} \| \widetilde{\mathcal{R}}^j \|^2$

7: **if** $\lambda_{\mathrm{term}} > 0$ **then** $\mathcal{L}_R \leftarrow \mathcal{L}_R + \lambda_{\mathrm{term}}\, \mathcal{E}\big(\{z^j + f_\theta(1, z^j)\}_{j=1}^{M},\ \{x_1^i\}_{i=1}^{B}\big)$ $\qquad$ *distributional, never pointwise*

8: $(\theta, \psi) \leftarrow$ Adam step on $\mathcal{L}_R + \mathcal{L}_D$

Sampling: $\widehat{x}_1 = \phi_1(x_0; \theta) = x_0 + f_\theta(1, x_0), \ x_0 \sim p_0$ $\qquad$ *NFE 1*

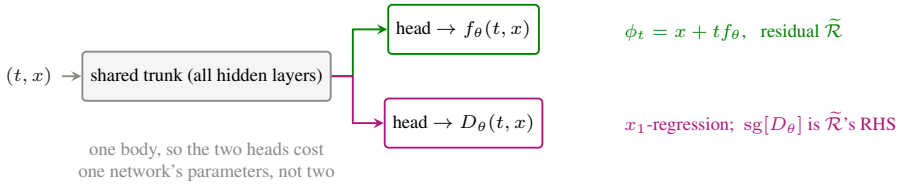

Figure 5: FLOWPINN-C. The flow map's free part and the denoiser are two heads on one trunk, so the method holds one network's worth of weights against FLOWPINN-L's two.

B ALGORITHMS

One training step of each of the two objectives the paper introduces. They differ in where the field is read, which is the whole argument of §3: Algorithm 1 needs D at the map's *output* and therefore carries a denoiser and a collocation batch, while Algorithm 2 reads u at the network's own *input* and needs neither. FLOWPINN-C is Algorithm 1 with f_θ and D_ψ as two heads of one trunk; FLOWPINN-P is Algorithm 1 with line 5's D_ψ replaced by (7) and the denoiser branch deleted.

Connection and distinction with MeanFlow. Identity (18) is MeanFlow's (Geng et al., 2025), rewritten in the forward-time convention of (3). Their derivation differentiates at the *later* time index, t, and ours at the earlier one, s : their axis runs noise-to-data, so their source end carries the larger index. Objective (19) is likewise theirs, and two-time maps with self-consistency conditions are due to Boffi et al. (2024; 2025), with related constructions in Song et al. (2023); Frans et al. (2025).

What differs is the role the identity plays. For MeanFlow the identity *is* the method. Here it is a *relaxation* of §3.1, reached by one concession: the residual (13) must evaluate u_t off the conditional path and pays a denoiser to estimate it there, whereas widening the map to two time arguments reads u_s at the network's own input, so the unbiased CFM target replaces it and no field estimate is needed anywhere. The price is that the constraint is no longer pointwise in time: (18) ties $\bar{g}$ together on the whole triangle $s \leq t$, and the trajectory it averages over grows longer and more curved with d. Setting $s = t$ returns the flow ODE, so §3.1 is exactly the restriction of this objective to the diagonal.

B.1 WHAT EACH VARIANT OPTIMIZES, AND WHAT IT COSTS

Table 2 puts every method in the paper on one page: the baseline, the two formulations of §2.2 that do not work, and the four variants of this section, each with what it optimizes and what that costs. Costs are in the unit

$$C \ = \ \Theta(P) \tag{20}$$

Algorithm 2 FLOWPINN-2T: one training step. There is no denoiser and no collocation batch — the whole objective lives on the conditional path, which is the trade §3.4 describes. $\mathrm{sg}[\cdot]$ is the stop-gradient and $c = 1 - \sigma_{\min}$.

Require: two-time map $\bar{g}_\theta$, batch B, diagonal fraction p_{diag}, semigroup weight $\lambda_{\mathrm{semi}} \geq 0$

$x_1^i \sim \widehat{q}, \;\; x_0^i \sim p_0$ $\qquad i = 1, \dots, B$

$a^i, b^i \sim U[0,1], \;\; s^i \leftarrow \min(a^i, b^i), \;\; t^i \leftarrow \max(a^i, b^i)$ $\qquad$ *a point of the triangle $s \leq t$*

with probability p_{diag}: $\;\; t^i \leftarrow s^i$ $\qquad$ *anchor the diagonal, where* (18) *is the flow ODE*

$x_s^i \leftarrow \sigma_{s^i} x_0^i + s^i x_1^i, \quad \tilde{v}^i \leftarrow x_1^i - c\, x_0^i$ $\qquad$ *$\tilde{v}^i$ is unbiased for $u_{s^i}(x_s^i)$ by* (5)

$\tau^i \leftarrow \big(\mathcal{D}_{\tilde{v}^i} \bar{g}_\theta\big)(x_s^i, s^i, t^i)$ $\qquad$ *one forward-mode pass in direction $(\tilde{v}^i, 1, 0)$*

$\mathrm{err}^i \leftarrow \bar{g}_\theta(x_s^i, s^i, t^i) - \mathrm{sg}\big[\, \tilde{v}^i + (t^i - s^i)\, \tau^i \,\big]$ $\qquad$ *the residual of* (18)

$w^i \leftarrow \mathrm{sg}\big[(\|\mathrm{err}^i\|^2 + \epsilon)^{-1/2}\big]$, normalised to mean 1 $\qquad$ *MeanFlow's adaptive weighting*

$\mathcal{L} \leftarrow \frac{1}{B} \sum_i w^i \|\mathrm{err}^i\|^2$

if $\lambda_{\mathrm{semi}} > 0$ **then** add $\lambda_{\mathrm{semi}}\, \mathbb{E}\big\|\Phi_{s\to t}(y) - \Phi_{r\to t}(\Phi_{s\to r}(y))\big\|^2, \;\; s \leq r \leq t, \;\; y \sim p_0$ $\qquad$ *expressible only here*

$\theta \leftarrow$ Adam step on $\mathcal{L}$

Sampling: $\widehat{x}_1 = \Phi_{0\to 1}(x_0; \theta) = x_0 + \bar{g}_\theta(x_0, 0, 1), \;\; x_0 \sim p_0$ $\qquad$ *NFE 1*

Table 2: Every method in the paper on one page: what it optimises, and what that costs. Costs are asymptotic, in the units of (20). One backbone has P parameters and one forward pass of it costs $C = \Theta(P)$, so a *per step* entry reads directly as a multiple of the baseline's model size. The $O(Kdw)$ in *params* is the separable readout (15) widening from d to Kd outputs: subleading for an image backbone, and more than paid for by the time-conditioning path it removes. The two formulations of §2.2 are priced here too, and the point of their rows is that they cost what the baseline costs: what separates them from the working rows is where the constraint is imposed, not what it is spent on. FLOWPINN-C's "one trunk" is FLOWPINN-L's map and denoiser carried as two heads of a single backbone (§3.3).

			training		sampling	
method	object	params	objective	per step	evaluation	per sample
CFM baseline	$v_t(x;\theta)$	P	$\mathbb{E}\|v_t(x_t) - \tilde{v}\|^2$	$O(BC)$	numerical ODE	$O(kC)$
input-anchored	$\phi_t(y;\theta)$	$P{+}O(Kdw)$	$\mathbb{E}\|\dot{\phi}_t(x_t) - \tilde{v}\|^2$	$O(BC)$	$\phi_1(x_0)$	$O(C)$
source-indexed	$\phi_t(y;\theta)$	$P{+}O(Kdw)$	$\mathbb{E}\|\dot{\phi}_t(x_0) - \tilde{v}\|^2$	$O(BC)$	$\phi_1(x_0)$	$O(C)$
FLOWPINN-P	$\phi_t(y;\theta)$	$P{+}O(Kdw)$	$\mathbb{E}\|\widehat{\mathcal{R}}_t(z)\|^2$, $D_t = \widehat{D}_t$	$O(MC{+}MNd)$	$\phi_1(x_0)$	$O(C)$
FLOWPINN-L	ϕ_t, D_ψ	$2P{+}O(Kdw)$	$\mathbb{E}\|\widetilde{\mathcal{R}}_t(z)\|^2$, $D_t = \mathrm{sg}[D_\psi]$ $+ \mathbb{E}\|D_\psi(x_t) - x_1\|^2$	$O((B{+}M)C)$	$\phi_1(x_0)$	$O(C)$
FLOWPINN-C	(f_θ, D_θ)	$P{+}O((K{+}J)dw)$	as FLOWPINN-L, one trunk	$O((B{+}M)C)$	$\phi_1(x_0)$	$O(C)$
FLOWPINN-2T	$\Phi_{s\to t}(y;\theta)$	P	$\mathbb{E}\|\bar{g}_\theta - \mathrm{sg}[\tau]\|^2$ $\tau = \tilde{v} + (t{-}s)\mathcal{D}_{\tilde{v}}\bar{g}_\theta$	$O(BC)$	$\Phi_{0\to 1}(x_0)$	$O(C)^\dagger$

Every row but the baseline samples at **NFE 1**; the baseline integrates, at NFE $k \geq 1$. †FLOWPINN-2T samples in one evaluation like the others, but may optionally be iterated as a genuine two-time map — §4.1 and §4.2 report what that does, which is not what iterating a solver does.

of one forward pass of a backbone with P parameters, each parameter used once, in a multiply and an add, so a pass is $\approx 2P$ operations. B is the regression minibatch and M the collocation batch of Algorithm 1, N the training-set size, d the state dimension, w the width of the last hidden layer, K and J the two basis ranks, and k the number of solver steps.

Three entries in it are worth mentioning; §4.3 and Appendix D.3 measure them.

Only the plug-in variant pays for the dataset. FLOWPINN-P's right-hand side is (7) at the map's output, $O(Nd)$ per collocation point and so $O(MNd)$ per step on top of the network, which is not a constant to be tuned away, and why FLOWPINN-P is a control rather than a proposal. Appendix E shows the same estimator failing for a statistical reason at the same time.

Everything else is a constant factor over regression, and the parametrization sets the constant. A forward-mode JVP leaves a doubled tape; the separable basis (15) returns $\dot{\phi}$ inside the pass it already makes and lets the M collocation points be drawn as M_x states $\times$ M_t times, so refining the time grid is nearly free. That option is open to the one-time maps only: $\Phi_{s\to t}$ is not of the form (15) and

its directional derivative (17) is not with respect to a single scalar, so FLOWPINN-2T keeps the JVP. The merge behind FLOWPINN-C moves parameters and not arithmetic, so its *per step* entry is FLOWPINN-L's exactly.

Sampling is $O(C)$ for every map and $O(kC)$ for the solver. This is the one column that depends on none of the choices above.

C EXPERIMENTAL PROTOCOL

Protocol. Four targets, in increasing order of what they can rule out: checkerboard (2×2 cells tiling $[-4, 4)^2$) and 8-Gaussians (radius 4, $\sigma = 0.4$), where every method can be held against an exact solve of the very field it is trying to represent; Neal's funnel at $d = 10$ and $d = 100$, whose geometry has no single scale and whose log-density is closed-form; and two image targets, MNIST at $d = 784$ and CIFAR-10 at $d = 3072$. The vector grids share $\sigma_{\min} = 0.02$, $8{,}192$ training points, width-192 depth-3 SiLU MLPs, $\mathbf{20{,}000}$ steps at batch 512, $K = 8$ and **ten seeds**; tables give mean $\pm$ s.d. Two comparisons run throughout: the same trained CFM field under Euler and RK4, and MEANFLOW (Geng et al., 2025) and SHORTCUT (Frans et al., 2025) trained here on the same data, backbone, budget and seeds. Those two are the closest relatives of what this paper proposes. All three learn an object evaluated once and differ only in what constrains it, so a difference between them is evidence about the constraint. We report the energy distance $\mathcal{E}$ to held-out data, one structural measure per target, and MEMO, the nearest-neighbor distance from samples to the training set over the same statistic for held-out real data (≈ 1 healthy, $\ll 1$ memorization, $\gg 1$ misplaced). The figures' reference line is an exact solve of the plug-in field (7), drawn as a floor rather than a competitor because its memorization ratio of 0.62–0.71 shows it buys its distance by reproducing the training set. Appendix C has the rest.

Targets. *checkerboard*: 2×2 cells tiling $[-4, 4)^2$. *8-Gaussians*: eight isotropic components of standard deviation 0.4 on a circle of radius 4. *Neal's funnel*: $v \sim \mathcal{N}(0, 3^2)$ and $x_i \mid v \sim \mathcal{N}(0, e^v)$ for $i = 1, \dots, d-1$, at $d = 10$ and $d = 100$; coordinates are whitened by the true marginal moments before training, so that the difficulty left in the target is its geometry and not its scale. *MNIST* at 28×28, $d = 784$. *CIFAR10* at $32 \times 32 \times 3$, $d = 3,072$. *Product checkerboard* (§E): $d/2$ independent copies of the $d = 2$ benchmark, so the marginal field factorizes over blocks and the true flow map is representable at every d.

Vector architectures and optimization. Width-192 depth-3 SiLU MLPs, 16-frequency Fourier time features (Tancik et al., 2020), Adam with a cosine schedule and warm-up, EMA decay 0.999, batch size 512, $20{,}000$ steps, $\sigma_{\min} = 0.02$. The learning rate is 2×10^{-3} on the two-dimensional benchmarks and 5×10^{-4} on the funnel, chosen once on the baseline and then held fixed for every method, so that no row is quietly running a different optimizer from the rows it is tabled against. The flow map uses the hard initial condition (1) and the separable parametrization (15) at $K = 8$, which supplies $\dot{\phi}$ in closed form; where the separable form is unavailable (FLOWPINN-2T), $\dot{\phi}$ comes from a single forward-mode JVP in t. For FLOWPINN-L the denoiser D_ψ has the same architecture and is trained concurrently, entering the residual through a stop-gradient; FLOWPINN-C replaces the pair by one trunk with two heads at full sharing (§3.3).

The separable time basis in full. Equation (15) uses

$$b(t) = t\,u(t), \quad u(t) = W_2 \tanh\big(W_1 \psi(t) + c_1\big) + c_2, \quad b'(t) = u(t) + t\,u'(t), \tag{21}$$

with $\psi(t) = [\,t,\ \sin(\omega t),\ \cos(\omega t)\,]$ a fixed feature map of known derivative, so that $u'(t) = W_2\big[(1 - \tanh^2) \odot (W_1 \psi'(t))\big]$. The explicit factor t is what makes $\phi_0(y) = y$ an identity. Against a forward-mode JVP the closed form agrees to 1.9×10^{-6}, i.e. to float32 round-off. What the JVP costs and the basis does not is memory and time: on our image backbones the dual-number pass holds $2.0\times$ (DiT) to $4.7\times$ (UNet) the activations saved for backward and runs at about $3.3\times$ the step time of a plain regression step.

FLOWPINN-C: why $\lambda_{\text{res}} = 1$ and full sharing are fixed. A probe at an earlier budget located the boundary at which the shared trunk's drift matters. At $d = 64$ the two knobs that raise the residual's

influence relative to the regression, $\lambda_{\text{res}} = 3$ and an $8\times$ larger learning rate, both diverge with the *same* signature, a map under-transporting onto a shell some 40% short of the data's, while $\lambda_{\text{res}} = 1$ is at its optimum in both directions. Separately, a sweep over how much of the trunk to share measured sharing *every* hidden layer to be the right default, on a surface that is not monotone in the number of shared layers. Neither has been repeated at the present budget, so both are reported as the evidence behind a fixed choice rather than as results of this paper.

FLOWPINN-C: what the merge buys, and what it risks. It buys parameters, and only parameters: $0.50\times$ FLOWPINN-L's at full sharing, changing $162{,}820 \to 81{,}796$ on the two-dimensional benchmarks, $14.49\text{M} \to 7.22\text{M}$ on the MNIST UNet. It buys nothing at sampling time, since the denoiser is discarded once training ends, and nothing per training step either: the two functions are needed at different points, so the trunk is evaluated twice whether or not it is shared. What it risks is the stop-gradient. No gradient reaches the denoiser head through the residual term, but nothing prevents the residual moving the *trunk* through the flow head, with the denoiser changing as a side effect — the denoiser sliding toward whatever field the current map already satisfies, so that the residual reports success while the map transports nothing. The regression term is the only thing holding D against that drift. Appendix C reports the probe that located the boundary where it matters, and is the reason $\lambda_{\text{res}} = 1$ and full sharing are fixed throughout rather than swept here.

Image architectures. A 7.2M-parameter UNet and a 7.3M-parameter DiT, both at 28×28, trained for $100{,}000$ steps. Flow-map methods use the separable parametrization at $K = 16$; FLOWPINN-2T cannot and runs with the JVP. Because the two backbones are matched to within 2.5% on parameter count for the same method, a difference between them is not a capacity difference.

Evaluation. Vector metrics use $4{,}096$ generated samples against $4{,}096$ held-out draws from the true q, over a fixed training set of $8{,}192$ points per seed, so that memorization is measurable for every method and not only for the plug-in variant. MEMO is the median nearest-neighbor distance from generated samples to the training set, divided by the same statistic computed for held-out real data. On the funnel, $|\Delta \log p|$ is the absolute difference in mean exact log-density between generated and real samples, and *neck KS* is the Kolmogorov–Smirnov distance of the v coordinate, for which real held-out data attains 0.013. The terminal term of (14) uses the energy-distance V-statistic on the two minibatches $\{x^{(i)}\}_{i\le n}$ and $\{y^{(j)}\}_{j\le m}$,

$$\widehat{\mathcal{E}} \;=\; \frac{2}{nm}\sum_{i,j}\|x^{(i)} - y^{(j)}\| \;-\; \frac{1}{n^2}\sum_{i,i'}\|x^{(i)} - x^{(i')}\| \;-\; \frac{1}{m^2}\sum_{j,j'}\|y^{(j)} - y^{(j')}\|, \tag{22}$$

where the $1/n^2$ and $1/m^2$ keep the diagonal terms — that is what makes it a V- rather than a U-statistic, and why $\widehat{\mathcal{E}} \ge 0$ for every batch, at the price of an $O(1/n)$ bias that vanishes with batch size.

Image metrics. On MNIST, $10{,}000$ generated images against the $10{,}000$-image test set: Inception-v3 FID and IS, plus a clf-FID in the 84-dimensional features of a LeNet trained on the same split — a **single** LeNet for all 230 evaluations of both MNIST tables, real-data accuracy 0.972, so that column is comparable across every row and both backbones. On CIFAR-10, $50{,}000$ generated images against the $50{,}000$ training originals, with clf-FID in the features of a ResNet-20 trained on the same split (test accuracy 0.94). Training-time model selection there uses a cheaper $10{,}000$-sample score at the method's own sampling tag; the *selected checkpoint* is then re-scored at $50{,}000$ across every budget, and that re-scored record is what Table 6 prints. CIFAR-10 carries no real-data floor row, because held-out real data was never scored through that pipeline.

Sample figures. Figures 3, 7 and 8 share one set of conventions. *Selection*: each panel is that method's best of ten seeds by energy distance and the number under a panel is the ten-seed mean of Tables 3 and 4. *Thinning*: every panel draws **every second** of its $4{,}096$ points, the same subsample throughout, so a position is still one draw carried across methods; the printed distance is computed on all $4{,}096$. *Provenance*: the two CFM panels of each figure are a CPU re-run of that one cell.

The two image figures share the other half of that. Every generated sheet comes from **one fixed noise batch**, so a column of Figure 9 and a position in Figure 4 are a single draw carried across every generated row or panel. Real data is the exception because no real image is the target of a particular

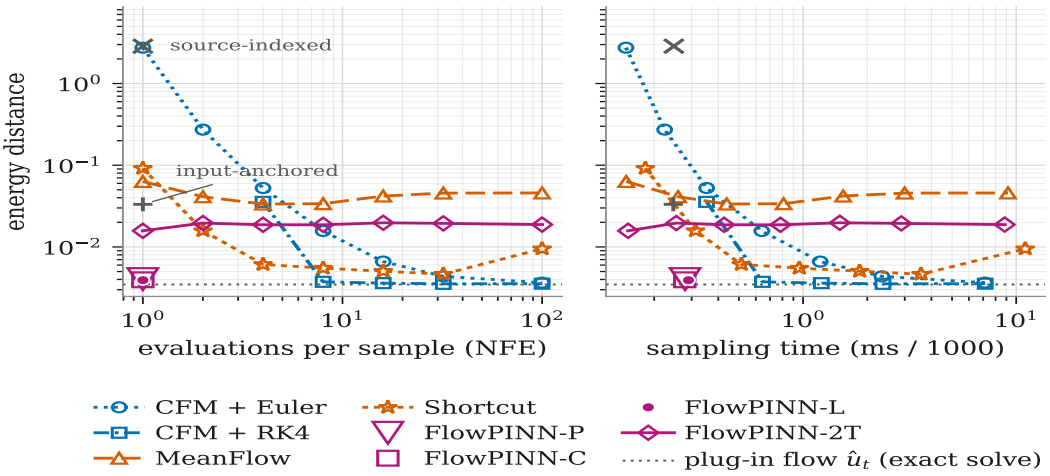

Figure 6: Cost against quality on 8-Gaussians — the companion to Figure 2, drawn from the same ten seeds and on the same axes. The reading is the same, and the separation between families is wider: at one evaluation the residual maps sit at 0.0039–0.0040 where MEANFLOW is at 0.0633 and SHORTCUT at 0.0918, and the solver needs eight RK4 evaluations to reach the same place.

noise vector. So the real-data row and the *Data* panel choose their own cells: on MNIST to put the same digit in each column, on CIFAR-10 to put the class the hundred-evaluation solve produced at that position, taken at the ResNet-20's highest confidence. Both are matches on a reference sheet and nothing stronger.

Seeds and error bars. Ten seeds for every vector experiment, **five** for every MNIST run and **one** for CIFAR-10; tables give the mean and the standard deviation across seeds where there is more than one. Both MNIST tables therefore carry measured error bars and no longer appeal to a hardware-noise band, as earlier single-seed versions had to; the CIFAR-10 table has no spread to report, and Table 1's five-seed spreads are the nearest available guide to how much of a gap there is noise. Sampling and step times come from one card per backbone across all five MNIST seeds and are reproducible to about 0.1%, so the wall-clock columns are within-table comparisons of the same measurement rather than of separate waves. The dimension sweep of §E runs at the same budget of ten seeds and 20,000 steps, with its own exact-solve reference computed once per dimension rather than once per seed.

D ADDITIONAL NUMERICAL RESULTS

Everything the main text compresses, in the order §4 takes the three targets: the two-dimensional benchmarks, then Neal's funnel, then MNIST. Every table here is emitted directly from the experiment records by the scripts shipped with the paper, so a row cannot disagree with a summary of itself, and rows at the sampling budgets the main text does not discuss are present in the source as comments, one deletion away from being restored.

D.1 THE TWO-DIMENSIONAL BENCHMARKS

Table 3 is the grid §4.1 reads from: nine methods at every sampling budget, ten seeds, mean and seed-to-seed standard deviation.

The readings §4.1. *Composition is not integration.* Iterating the two-time map raises the energy distance monotonically (0.0053 → 0.0132 on checkerboard and 0.0158 → 0.0188 on 8-Gaussians over NFE 1 → 100) while *improving* the structural measures (off-support 0.121 → 0.054, in-mode 0.951 → 0.995). MEANFLOW does the same on checkerboard (0.0103 → 0.0274) and the opposite on 8-Gaussians (0.0633 → 0.0458); SHORTCUT, whose objective is a self-consistency condition between step sizes, improves on both until it does not, reaching 0.0029 on checkerboard at four evaluations. Composition sharpens a learned map onto the data manifold and simultaneously distorts

Table 3: The two-dimensional benchmarks in full: every method at every sampling budget, mean ± s.d. over **10 seeds** at 20,000 training steps. $\mathcal{E}$ is the energy distance to held-out data; MEMO is the nearest-neighbour memorisation ratio, healthy at ≈ 1 and recall below it; *off-supp.* (lower better) and *in-mode* (higher better) are the structural measures. The last row is an exact RK4-200 solve of the empirical field (7), a floor and not a competing method: it buys its distance by reproducing the training set, which is what its memorisation ratio says.

		checkerboard			8-Gaussians		
method	NFE	$\mathcal{E}\downarrow$	MEMO	off-supp.↓	$\mathcal{E}\downarrow$	MEMO	in-mode↑
CFM + Euler	1	1.8744 ±0.0145	1.55	0.430	2.7461 ±0.0637	111.6	0.000
CFM + Euler	2	0.6315 ±0.0117	1.45	0.360	0.2726 ±0.0126	8.08	0.423
CFM + Euler	4	0.1423 ±0.0064	1.13	0.167	0.0526 ±0.0024	1.31	0.863
CFM + RK4	4	0.0239 ±0.0024	1.22	0.243	0.0356 ±0.0069	1.11	0.911
CFM + RK4	8	0.0023 ±0.0010	1.05	0.086	0.0037 ±0.0017	1.00	0.973
CFM + RK4	100	**0.0021** ±0.0009	1.05	0.085	**0.0036** ±0.0018	0.99	0.983
source-indexed	1	1.9217 ±0.0147	1.44	0.361	2.8650 ±0.0035	112.3	0.000
input-anchored	1	0.0785 ±0.0060	1.38	0.329	0.0332 ±0.0057	2.63	0.719
FLOWPINN-P	1	0.0024 ±0.0011	1.02	0.054	0.0040 ±0.0016	0.95	0.970
FLOWPINN-L	1	0.0024 ±0.0008	1.12	0.162	**0.0039** ±0.0021	1.01	0.934
FLOWPINN-C	1	**0.0023** ±0.0008	1.12	0.159	0.0040 ±0.0020	0.99	0.929
FLOWPINN-2T	1	0.0053 ±0.0018	1.08	0.121	0.0158 ±0.0085	0.93	0.951
FLOWPINN-2T	2	0.0082 ±0.0015	1.04	0.072	0.0196 ±0.0076	0.88	0.989
FLOWPINN-2T	4	0.0104 ±0.0023	1.03	0.067	0.0187 ±0.0045	0.86	0.994
MEANFLOW	1	0.0103 ±0.0023	1.06	0.117	0.0633 ±0.0339	0.98	0.932
MEANFLOW	2	0.0152 ±0.0062	1.04	0.074	0.0412 ±0.0244	0.92	0.970
MEANFLOW	4	0.0132 ±0.0046	1.01	0.044	0.0335 ±0.0200	0.87	0.986
SHORTCUT	1	0.0058 ±0.0013	1.38	0.328	0.0918 ±0.2528	4.76	0.759
SHORTCUT	2	0.0037 ±0.0012	1.20	0.227	0.0158 ±0.0275	1.17	0.893
SHORTCUT	4	0.0029 ±0.0012	1.12	0.159	0.0061 ±0.0018	1.07	0.942
plug-in flow $\widehat{u}_t$ (ref.)	200	0.0020	0.62	0.016	0.0035	0.71	0.989

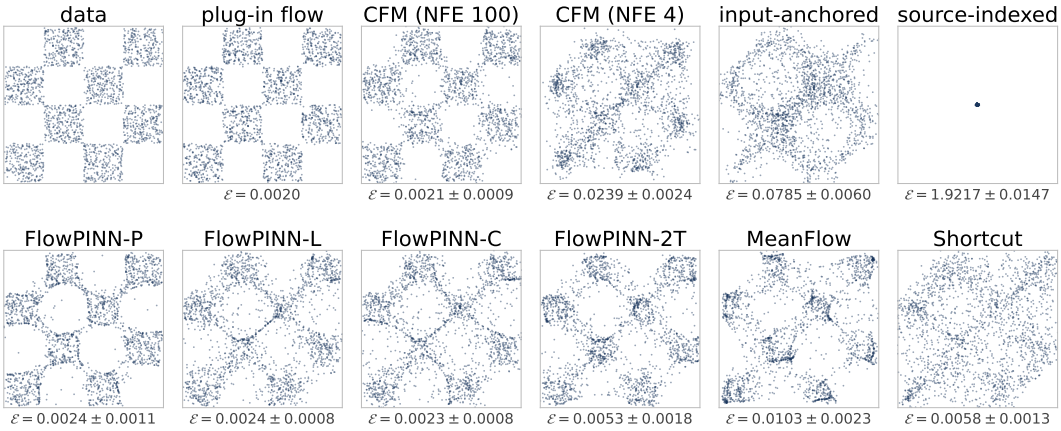

Figure 7: Generated samples on checkerboard (Appendix C). Three things are visible here that no table shows. The source-indexed map is a single dot at the data mean by Corollary 1, as what one Euler step produces. The input-anchored map is a diffuse cloud that ignores the support, the signature of Proposition 2: not a blurred target but mass carried somewhere else. And FLOWPINN-L and FLOWPINN-C, at one evaluation each, are hard to tell apart from the hundred-step solve in the row above and easy to tell apart from the same solver at four, the panel beside it — which is the claim of §4.1 in a form a reader can check directly. MEANFLOW and SHORTCUT at the same budget are visibly thinner on the cells, and SHORTCUT has lost them.

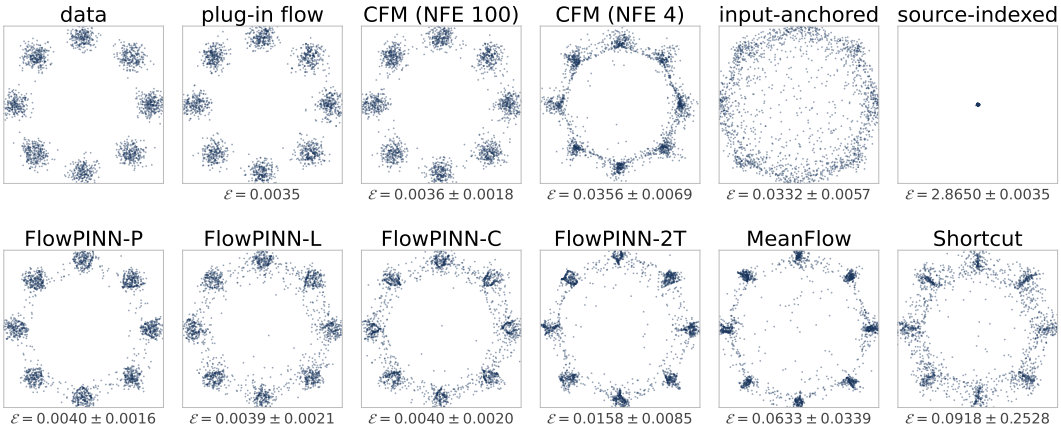

Figure 8: The same twelve panels on 8-Gaussians, in the same order and under the same conventions. The failure of the source-indexed map is starker here because the data mean lies in a hole in the support: its memorization ratio of 112 in Table 3 is this picture as a number. SHORTCUT at one evaluation places mass between the modes, which is what an energy distance of 0.0918 against the residual maps' 0.0039 buys, and CFM at four evaluations puts a whole ring there.

the mass it carries there, because self-consistency error compounds where a solver's discretization error contracts. This is the honest cost of that parametrization and the empirical content of §3.4.

The failure modes agree with their closed forms to more than one digit. The source-indexed memorization ratios track the one-step Euler baseline's more tightly than the distances do (1.44 against 1.55 on checkerboard; 112.3 against 111.6 on 8-Gaussians), both being the signature of a point mass sitting where the data is not.

The wall-clock saving is not uniform across the family. The single-time maps sit 25–27$\times$ to the left of CFM's hundred-evaluation solve at equal or better quality; FLOWPINN-2T and MEANFLOW, which carry no denoiser at sampling time, sit 47–48$\times$ to the left. The gap widens with the size of the backbone — on MNIST it is $117\times$ (§4.3) — because at width 192 a large part of a call is overhead that a solver amortizes over its steps and a one-evaluation map cannot.

Two $d = 2$ entries that invert later. The plug-in right-hand side is not merely adequate at this dimension but the best of the three residual maps on both structural measures: 0.054 off-support against 0.086 for a hundred-step solve. FLOWPINN-C matches FLOWPINN-L to within 0.0001 on both benchmarks at half the trainable parameters. Both readings are reversed by dimension: §4.2 and §4.3 are where each stops being true, and Appendix E gives the plug-in mechanism in closed form – the kernel's softmax becomes a hard nearest-neighbor selector at a rate $t/\sigma_t \gtrsim \Theta(d^{-1/2})$.

D.2 NEAL'S FUNNEL

Table 4 is the grid §4.2 reads from, and the paragraphs below are the detail it compresses.

Why this target. Three properties, and the paper needs all three. Its conditional scale varies over $e^{\pm 6}$, so a fixed-step solver is mismatched somewhere *by construction*, precisely the regime in which solving once by residual should differ from integrating, and the reason a target with a single characteristic scale cannot separate the two. Its log-density is exact, so the geometry can be scored directly rather than through a distributional proxy, which matters here because the three kinds of score disagree. And it scales adversarially: every added coordinate is another draw at the conditional scale, so $d = 10 \to 100$ makes the target harder rather than merely wider. This is unlike the product benchmark of §E, whose blocks are independent and whose true map stays representable at every d.

The ordering at $d = 10$. FLOWPINN-L at one evaluation reaches 0.0039, which is exactly what CFM reaches under an eight-evaluation RK4 step; the solver needs a hundred evaluations to halve it, to 0.0019. Among one-evaluation methods the order is FLOWPINN-L 0.0039, FLOWPINN-2T 0.0130, FLOWPINN-C 0.0161, MEANFLOW 0.0242, SHORTCUT 0.0279, FLOWPINN-P 0.0805, and the two failure modes an order of magnitude beyond that, at 0.50 and 1.05. FLOWPINN-P's position here, twentyfold behind a learned right-hand side at only five times the dimension of the benchmark it won, is the first sign of the collapse §E predicts in closed form. Figure 3 shows what each of them draws.

What the neck statistic adds. Energy distance, log-density gap and neck fraction rank the methods three different ways, which is the argument for reporting all three. At $d = 10$ they now agree that the target is solved: FLOWPINN-L puts 0.0224 of its mass in the thin end against a true 0.0228, and so does FLOWPINN-2T at two evaluations. At $d = 100$ they separate. FLOWPINN-C leads on distance and puts 0.0000 there; FLOWPINN-L is second on distance, best on log-density by an order of magnitude, and puts 0.0031; CFM at a hundred evaluations puts 0.0104 while scoring worst of the three on log-density. SHORTCUT is the limiting case, which is the best neck KS of any one-evaluation row, at an energy distance of 2.64, and the reason Table 4 ranks distance and nothing else.

The readings §4.2. *The metrics disagree, and we report all three.* FLOWPINN-2T and FLOWPINN-C lead on energy distance at $d = 100$ while FLOWPINN-L is the only method that recovers the funnel's *geometry*: a log-density gap of 17 against 164 for FLOWPINN-C, 375 for CFM, 594 for FLOWPINN-2T and 870 for MEANFLOW, and the best neck statistic of any one-evaluation row that is not also a failure. SHORTCUT is the cautionary case, scoring the lowest neck KS at $d = 100$ while sitting at $\mathcal{E} = 2.64$: a marginal can be right while the joint is not, which is why the table ranks only distance.

The neck. The thin end, $v < -2\sigma_v$, should hold 0.0228 of the mass. At $d = 10$ every decent method now finds it: 0.0224 for FLOWPINN-L and for FLOWPINN-2T at two evaluations, against 0.0214 for a hundred Euler steps and 0.0222 for the exact solve, which is new at this budget and was not true at $8{,}000$ steps. At $d = 100$ it is still missed: 0.0031 for FLOWPINN-L, 0.0000 for FLOWPINN-C, 0.0104 for CFM at a hundred, against 0.0224 for an exact solve of the same empirical field. More training moved this once already, which is why we read it as a learning failure with an address.

How the two $d = 2$ winners come apart. FLOWPINN-P fails on both counts at $d = 100$: $\mathcal{E} = 0.517$, the worst of the residual maps, with a memorization ratio of 0.43 against 0.69 for FLOWPINN-L, so the variant that was best on the two-dimensional structural measures is both the least accurate and the one reproducing its training set. FLOWPINN-C's inversion is narrower than it looks: better than FLOWPINN-L on distance at $d = 100$ (0.1774 against 0.2774), worse on everything else, needing a smaller learning rate to converge there at all, and carrying an outlier seed at $d = 10$.

Table 4: Neal's funnel, whitened, mean ± s.d. over **10 seeds** at 20,000 training steps. $|\Delta \log p|$ is the absolute gap in mean exact log-density between generated and real samples, and *neck KS* the Kolmogorov–Smirnov distance of the neck coordinate v, for which held-out real data attains 0.013. Lower is better in every column. The last row is an exact solve of the empirical field, which reaches its numbers by memorizing (MEMO 0.33 and 0.18). FLOWPINN-C at $d = 100$ is the one row trained at a different learning rate (2×10^{-4} against the grid's 5×10^{-4}, which did not converge there); every other row shares the grid's. Intermediate solver budgets are present in the source as comments.

		$d = 10$			$d = 100$		
method	NFE	$\mathcal{E}$	$\lvert\Delta \log p\rvert$	neck KS	$\mathcal{E}$	$\lvert\Delta \log p\rvert$	neck KS
CFM + Euler	1	0.3334 ±0.0326	2.7	0.407	0.4325 ±0.1346	879	0.156
CFM + Euler	2	0.1294 ±0.0113	3.1	0.195	**0.1436** ±0.1408	193	0.083
CFM + Euler	4	0.0499 ±0.0064	2.4	0.105	0.1574 ±0.1514	269	0.067
CFM + RK4	4	0.0253 ±0.0013	20.4	0.036	0.4096 ±0.2213	374	0.082
CFM + RK4	8	0.0039 ±0.0012	9.3	0.021	0.2836 ±0.1892	391	0.033
CFM + RK4	100	**0.0019** ±0.0007	4.7	0.019	0.2672 ±0.1899	375	0.031
source-indexed	1	0.5041 ±0.0129	4.2	0.475	0.5224 ±0.0163	30.7	0.438
input-anchored	1	1.0489 ±0.0930	27.9	0.542	1.5614 ±0.3527	219	0.598
FLOWPINN-P	1	0.0805 ±0.0043	2.1	0.230	0.5170 ±0.0149	73.5	0.535
FLOWPINN-L	1	**0.0039** ±0.0027	6.6	0.027	0.2774 ±0.0384	17.2	0.180
FLOWPINN-C	1	0.0161 ±0.0211	8.8	0.041	0.1774 ±0.0159	164	0.254
FLOWPINN-2T	1	0.0130 ±0.0024	2.8	0.081	0.2354 ±0.0277	1020	0.239
FLOWPINN-2T	2	0.0170 ±0.0039	0.9	0.091	**0.1430** ±0.0083	594	0.254
FLOWPINN-2T	4	0.0240 ±0.0051	0.9	0.108	0.1898 ±0.0081	790	0.271
MEANFLOW	1	0.0242 ±0.0019	2.7	0.120	0.2029 ±0.0080	870	0.204
MEANFLOW	2	0.0352 ±0.0060	0.5	0.142	0.1790 ±0.0141	981	0.224
MEANFLOW	4	0.0536 ±0.0116	1.3	0.161	0.1730 ±0.0164	1197	0.271
SHORTCUT	1	0.0279 ±0.0066	25.4	0.071	2.6440 ±0.0888	2228	0.124
SHORTCUT	2	0.0228 ±0.0087	18.5	0.035	2.6691 ±0.1120	2419	0.104
SHORTCUT	4	0.0206 ±0.0089	18.0	0.045	2.2797 ±0.3800	1967	0.108
plug-in flow $\widehat{u}_t$ (ref.)	200	0.0016	2.0	0.017	0.0042	20.9	0.017

Table 5: The DiT half of the MNIST matrix: the same protocol, row set and conventions as Table 1, mean ± s.d. over five seeds at 100k steps. Parameter counts agree with the UNet's to within 2.5% for the same method, which is what makes the comparison in §4.3 a controlled one. Best NFE-1 FID in bold; the last row is the same real-data floor, which does not depend on the backbone.

method	NFE	FID↓	clf-FID↓	IS↑	MEMO	ms/1k↓	params
CFM + Euler	1	284.7 ±4.0	569.0 ±0.6	1.197 ±0.010	1.014 ±0.000	55	7.34M
CFM + Euler	4	64.36 ±1.08	116.0 ±2.0	1.816 ±0.010	1.001 ±0.001	220	7.34M
CFM + midpoint	100	**26.16** ±0.16	**2.03** ±0.36	2.023 ±0.013	0.997 ±0.004	5513	7.34M
FLOWPINN-P	1	213.2 ±2.9	754.9 ±1.0	1.087 ±0.003	1.023 ±0.036	57	7.32M
FLOWPINN-L	1	40.88 ±1.23	7.19 ±1.38	2.046 ±0.018	1.002 ±0.004	57	14.66M
FLOWPINN-C	1	47.16 ±1.59	**5.08** ±0.58	2.027 ±0.015	1.001 ±0.003	57	7.39M
FLOWPINN-2T	1	39.70 ±2.25	11.67 ±0.87	2.108 ±0.009	0.998 ±0.003	54	7.36M
FLOWPINN-2T	4	**30.18** ±0.76	6.44 ±0.57	1.995 ±0.009	0.998 ±0.004	216	7.36M
MEANFLOW	1	55.56 ±2.37	14.94 ±1.08	2.130 ±0.016	1.001 ±0.004	55	7.36M
MEANFLOW	4	31.11 ±1.15	17.04 ±1.28	2.018 ±0.013	1.000 ±0.003	219	7.36M
SHORTCUT	1	72.36 ±1.26	23.04 ±1.20	2.187 ±0.020	1.001 ±0.003	62	7.36M
SHORTCUT	4	48.94 ±0.39	12.27 ±0.69	2.087 ±0.018	0.999 ±0.003	248	7.36M
real data (floor)	–	1.61	0.41	2.096	0.996	–	–

D.3 MNIST

The DiT is the faster backbone at every budget with 57 against 109 ms per thousand samples for FLOWPINN-C, and 5,513 against 12,692 for the hundred-step solver — so the one-evaluation saving is still about 100× there. What it is not is a backbone on which one evaluation matches a hundred: the best one-evaluation map, FLOWPINN-2T, gives 39.70 ± 2.25 against DiT-CFM's 26.16 ± 0.16, a 52% gap the UNet does not show, and FLOWPINN-L is 56% behind. It does still beat CFM at a matched four evaluations by 38%.

Training step time is measured over five seeds. As a multiple of each backbone's own CFM step, the UNet gives 0.86× (FLOWPINN-P), 1.10× (SHORTCUT), 1.88× (FLOWPINN-L), 1.90× (FLOWPINN-C), 2.17× (FLOWPINN-2T) and 2.55× (MEANFLOW); the DiT gives 1.02×, 1.47×, 2.19×, 2.21×, 2.14× and 2.17× in the same order. The residual maps and MEANFLOW cost about twice a regression step on either backbone and SHORTCUT rather less, which is the constant factor §B.1 predicts; nothing here carries the ±10% cross-wave caveat the earlier version of this paragraph had to attach.

The readings §4.3. *Why the DiT penalizes one evaluation.* Two facts point at a cause. The penalty falls as a map is given more evaluations (FLOWPINN-2T: +52% at NFE 1, +23% at NFE 4), and the separable restriction of §3.2 costs on the DiT while costing nothing on the UNet. Both are what one would expect if the difficulty is that adaLN-Zero modulates every DiT block from the time embedding, which the separable trunk removes, where the UNet's FiLM conditioning is a lighter touch. A sweep of K on the DiT is the experiment this calls for, and we have not run it.

Three smaller observations. FLOWPINN-2T is the one place in this paper where composition helps monotonically ($26.14 \to 25.91 \to 24.49$ over NFE $1 \to 4$ on the UNet, and $39.70 \to 35.79 \to 30.18$ on the DiT), in direct contradiction to the funnel, where it does not; we have no account of the difference. Every method's memorization ratio sits in $[0.99, 1.03]$ on both backbones, so nothing here is reproducing the training set. And the Inception score is worth ignoring on this data: SHORTCUT posts the highest IS in both tables (2.18 and 2.19) while placing last on FID, and both exceed the 2.096 that real held-out digits score — a metric a method can beat the data on is not measuring the thing the table is about, which is why the last row of each table is that data.

How Figure 9 picks its ten cells and its seed. The cells are those on which all six generated rows agree, measured as the worst pairwise pixel correlation among them; the ten shown range from 0.91 (the 4) to 0.99 (the 1 and the 6). Of the five seeds it is the one whose draws admit a full 0–9 sweep that all six rows agree on, and the choice does not flatter us: it is simultaneously the *best* of the five

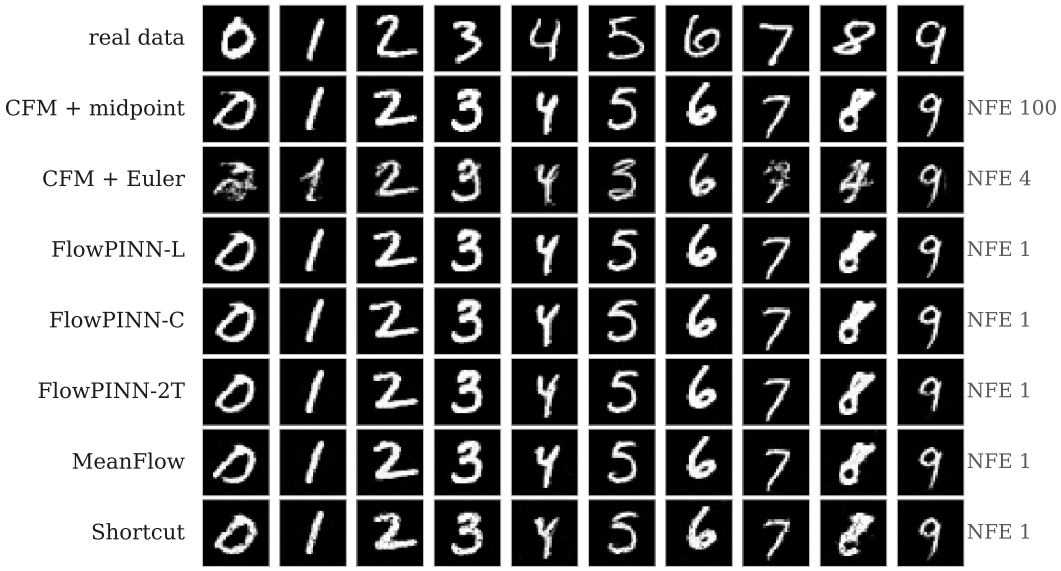

Figure 9: Ten source draws, one per digit, decoded by each method on the UNet backbone; conventions in Appendix C and the cell and seed selection in Appendix D.3. The row that matters is *CFM + Euler at four evaluations* against the five one-evaluation maps below it: four times a map's budget, and the solver still has not resolved half of these digits while every map has resolved all ten. SHORTCUT, last on FID, is the one map whose strokes break up. FLOWPINN-P is absent because it emits the same stroke for every draw (FID 209.6, class entropy 0.0000).

for the hundred-step solver (24.13 against a five-seed mean of 24.23) and the *worst* for SHORTCUT (48.39 against 44.33).

What the merge does not buy. FLOWPINN-C's step time is within 1.3% of FLOWPINN-L's on the MNIST UNet (371.0 against 366.3 ms) and its peak memory within 0.11 GiB: the separable parametrization had already removed the memory pressure the merge was designed to relieve, and the two savings are not additive.

The separable basis is what makes the MNIST matrix affordable, and it is not free on every backbone. On the UNet, writing the map in the basis of §3.2 at $K = 16$ takes FLOWPINN-L from 19.02 to 7.59 GiB of peak memory and from $5.79\times$ to $1.88\times$ CFM's step time at an unchanged parameter count, and FLOWPINN-P from $5.06\times$ to $0.86\times$, cheaper per step than the regression baseline it is compared against — while costing nothing measurable in FID ($26.63 \to 24.63$). On the DiT it costs 13.2% ($35.58 \to 40.28$). Those four pairs are single-seed measurements from an earlier wave, made under identical conditions within each pair, which is what an ablation needs; the five-seed values Tables 1 and 5 give for the $K = 16$ endpoints are 24.29 ± 0.75 and 40.88 ± 1.23, so neither endpoint moves outside its spread.

What FLOWPINN-C *gives up, in the other FID.* clf-FID moves the *opposite* way to Inception FID on both backbones, with 1.34 against FLOWPINN-L's 1.60 on the UNet, 5.08 against 7.19 on the DiT, and that column is comparable across every row, one LeNet having scored all 230 evaluations, so the disagreement between the two variants is real about what the merge costs rather than a scoring artifact.

D.4 CIFAR-10

Table 6 is one seed per method, so this appendix has no spread to add to it. What it can add is the accounting §4.4 compresses.

What the frozen right-hand side costs, and what it does not. The frozen-field FLOWPINN-L row and its jointly trained counterpart differ in *what has to be trained*, not in what is sampled: both sample one 55.6M-parameter map in one evaluation, and the frozen variant additionally requires a 55.7M-

Table 6: CIFAR-10 at 32×32: 200k steps and one 55.7M-parameter UNet for every row, each row scored at its own best checkpoint on 50,000 generated samples against the 50,000 training originals. *ms/1k* is sampling wall-clock rescaled to an A100 (Appendix C); MEMO is the memorisation ratio, healthy at ≈ 1. Appendix D.4 has the checkpoints, the compute accounting and the rows at intermediate budgets, which are in the source as comments.

method	NFE	FID↓	clf-FID↓	IS↑	MEMO	ms/1k↓	params
CFM + Euler	1	291.6	29.54	1.908	0.945	528	55.68M
CFM + Euler	4	105.8	16.41	5.092	1.005	2111	55.68M
CFM + midpoint	100	**3.01**	**0.08**	9.616	0.997	52713	55.68M
+FLOWPINN-L	1	**8.69**	0.38	8.497	0.997	518	55.57M
FLOWPINN-C	1	58.39	5.37	5.083	0.999	525	55.79M
FLOWPINN-2T	1	15.44	0.41	7.958	0.999	528	55.74M
FLOWPINN-2T	4	5.28	**0.22**	9.160	1.000	2111	55.74M
MEANFLOW	1	9.88	0.34	8.343	0.999	548	55.74M
MEANFLOW	4	6.09	0.26	9.313	0.999	2177	55.74M
SHORTCUT	1	23.65	4.16	6.568	0.997	522	55.74M
SHORTCUT	4	18.41	4.35	7.190	0.997	2085	55.74M

parameter CFM field to have been trained to convergence first. In wall-clock on one A100 that is 29 hours for the map and 11 for the field, against 33 for MEANFLOW and 15 for SHORTCUT: cheaper than MEANFLOW if the field is taken as given, 25% more expensive if it is not. Which framing is right depends on the question. Against *how good can one evaluation of a flow map be* the field is given, and 9.38 against a 3.25 teacher is a one-evaluation approximation to a hundred-evaluation solve at $96\times$ less sampling time. Against *how good is this objective as a way to learn a map from data* it is not, and the number is the jointly trained 33.76.

The rest of the table. SHORTCUT trails at 23.65 and 18.41, $2.5\times$ behind FLOWPINN-L at one evaluation; memorization ratios stay in $[0.945, 1.005]$ throughout; and with the three cards rescaled onto one, every method costs the same per evaluation, so NFE is the entire cost story at this scale as well. clf-FID does *not* reproduce the Inception ordering at the top of the table, with 0.34 for MEANFLOW, 0.41 for FLOWPINN-2T and 0.52 for FLOWPINN-L, against 4.16 for SHORTCUT and 5.37 for FLOWPINN-C. So the three leading rows are separated by less than the choice of feature space, which is the same caution Table 1 earns for FLOWPINN-C.

E THE PLUG-IN RIGHT-HAND SIDE AND DIMENSION

§4.1 states that the plug-in field is a low-dimensional convenience. This appendix gives the mechanism in closed form and the sweep that measures it, at the same budget as everything else in the paper: ten seeds, 20,000 steps, $d \in \{2, 8, 32, 64, 128\}$, every method. Figure 10 is the energy distance against d; Table 7 adds the structural and memorization measures.

The benchmark. A product of $d/2$ independent two-dimensional checkerboards, which at $d = 2$ is exactly the benchmark of §4.1. Two properties make it the right test rather than merely a bigger one: the marginal field factorizes over blocks, so the true flow map stays exactly representable at every d and nothing about the target degrades; and with $8^{d/2}$ modes ($\approx 6 \times 10^{57}$ at $d = 128$) against $N = 8{,}192$ training points, a sample's *cell code*, which cell each block falls in, is a direct memorization probe.

The plug-in right-hand side collapses; the learned one does not. This is the result the appendix exists for. FLOWPINN-P gives $\mathcal{E} = 0.0031$ at $d = 2$ and then 0.304, 2.67, 3.01 and 3.97 at $d = 8, 32, 64, 128$, resulting a total failure, while the same residual with a learned denoiser gives 0.00295, 0.00584, 0.0130, 0.0162 and 0.0699, which is within a factor 1.5 of the exact-solve reference at every d up to 64 and a factor 3.2 at 128. The memorization ratio says the same thing from the other side:

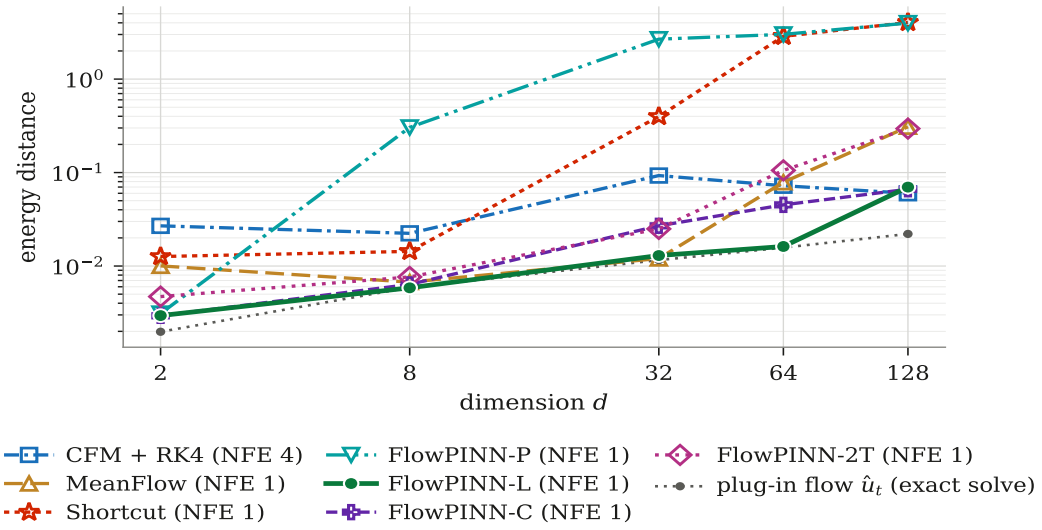

Figure 10: Energy distance against dimension on the product checkerboard, ten seeds at 20,000 steps, each method at the budget its legend entry names. Both FLOWPINN-P and SHORTCUT, at one evaluation, end the sweep near $\mathcal{E} = 4$ where every other method is between 0.06 and 0.31. They *cross* the pack rather than starting above it: at $d = 2$ FLOWPINN-P is the equal of the best map in the table and SHORTCUT is within a factor of five of it, which is what makes the collapse a statement about dimension rather than about the methods being weak. FLOWPINN-L tracks the exact solve within a factor 1.5 out to $d = 64$. Unlike Figures 2 and 6, color here identifies the method rather than its family: there the same-family markers are nested on a single point, while seven curves crossing over five dimensions cannot be read by marker alone.

FLOWPINN-P sits at 0.77–0.78 from $d = 32$ on, against ≈ 1.0 for FLOWPINN-L. The mechanism is softmax concentration, and it is predictable in advance.

Proposition 4 (softmax collapse in d)**.** *Let $x_t = \sigma_t x_0 + t\, x_1^\star$ with $x_1^\star$ the source datum, and for a candidate $x_1^{(n)}$ put $\Delta_n = x_1^\star - x_1^{(n)}$. The log-weights of* (7) *satisfy*

$$\ell^\star - \ell_n = \frac{t^2\|\Delta_n\|^2 + 2t\sigma_t\langle x_0, \Delta_n\rangle}{2\sigma_t^2}. \tag{23}$$

On a product benchmark the coordinates of Δ_n are i.i.d. across blocks, so $\|\Delta_n\|^2 = \Theta(d)$ while $\langle x_0, \Delta_n\rangle = O(\sqrt{d})$. The gap is therefore dominated by $t^2\Theta(d)/2\sigma_t^2$, and the softmax concentrates on a single neighbor once it exceeds $\log N$*, i.e. once*

$$t/\sigma_t \gtrsim \Theta\big(d^{-1/2}\big). \tag{24}$$

Past that threshold $\widehat{D}_t$ is a hard nearest-neighbor selector and the ODE it defines is a stiff switching system that no smooth trial function can satisfy. The rate itself is measured directly in Appendix A, on genuine path states and independently of any training run, and the sweep is what it predicts: at $d = 2$ the threshold is never crossed within the sampled range of t/σ_t and FLOWPINN-P is within 6% of the best learned map in the table, while by $d = 8$ it has been crossed and FLOWPINN-P is already a factor 52 worse than the same residual with a learned denoiser, with no recovery at any larger d. **The $O(Nd)$ plug-in field is a two-dimensional convenience, not a method**. The prediction is not confined to this benchmark: §4.2 finds the same collapse at $d = 100$ on a target with no product structure at all, and §4.3 finds it at $d = 784$ in its most legible form, a single digit reproduced for every draw.

A published one-evaluation map collapses the same way, and we cannot prove why. SHORTCUT at one evaluation follows the same trajectory as FLOWPINN-P one dimension later: 0.0126 and 0.0143 at $d = 2, 8$, then 0.398, 2.87 and 4.04 at $d = 32, 64, 128$, against FLOWPINN-P's 3.97 at $d = 128$. The signature is indistinguishable — per-block accuracy at exactly chance (0.500 and 0.502

at $d = 64$) and a memorization ratio of 0.78–0.79 for both, which is what an under-dispersed cloud scores: a cloud that has contracted towards the data's centre sits nearer the training set than a correct one does. What differs is the account we can give. For FLOWPINN-P the cause is Proposition 4: a supervision signal that provably degenerates. For SHORTCUT we can only report the measurement, and note that iterating does not rescue it: four evaluations still leave it at 0.353 at $d = 128$, a factor 16 above the reference, where MEANFLOW, which degrades at one evaluation just as sharply (0.309), recovers to 0.0453 with four. We read this as evidence that the failure is specific to the self-consistency recursion rather than to one-evaluation sampling as such, but that is an inference from one benchmark, not a result.

The two-time map needs a second evaluation as d grows. At $d = 2$ it was best at NFE 1 and worse when iterated; that reverses with dimension. $\mathcal{E}$ at NFE 1 degrades to 0.0250 ($d = 32$), 0.106 ($d = 64$) and 0.295 ($d = 128$), while iterating undoes it: four evaluations give 0.0150, 0.0320 and 0.0293, within a factor 1.3 to 2.0 of the exact-solve reference at all three. This is the cheap cost predicted in §3.4, and it buys back the accuracy, but a real asymmetry in favor of the single-time residual, whose one evaluation needs no such correction. MEANFLOW, the other two-time objective here, degrades and recovers on the same schedule which suggests the asymmetry belongs to the relaxation rather than to our instantiation of it. The funnel reproduces the sign of this at $d = 100$ (Table 4) and MNIST reverses it (§4.3); we have no account of the difference.

The learned maps generalize; an exact solve of the empirical field recalls. At $d \geq 32$ every learned method places every sample in a cell code absent from the training set, matching real held-out data at 1.000, while the plug-in flow reaches 0.228, 0.406 and 0.642 at $d = 32, 64, 128$: solving the empirical ODE exactly returns the training set, and it does so at every dimension we can measure. The memorization ratio has the reference at 0.010, 0.0084 and 0.0076 at those three dimensions against ≈ 1 for the learned maps. This is the same statement as the reference row of Table 3, at a dimension where it is no longer subtle, and it is why the reference is drawn as a floor and never as a competitor.

What breaks first is not the parametrization. Between $d = 8$ and $d = 32$ the per-block on-support rate falls to within a few points of chance for *every* trained method. The range over the whole table is 0.38 to 0.52 at $d \geq 32$, and that includes CFM with a hundred-evaluation RK4 solve ($0.675 \to 0.405$), which shares no part of our parametrization. The exact solve of the empirical field holds 0.984 at every dimension, so the benchmark has not become harder to represent; what has run out is $N = 8{,}192$ samples against $8^{d/2}$ modes. The consequence runs both ways, and it is the reason this sweep is an appendix and Neal's funnel is the main text: nothing here indicts the flow-map idea at $d \geq 32$, and nothing vindicates it either, because at that point the estimation problem, not the solver and not the map, is what every method is losing to.

Table 7: The dimension sweep in full, on a product of $d/2$ independent two-dimensional checkerboards: mean $\pm$ s.d. over **10 seeds** at $20{,}000$ training steps, at every dimension and for every method. The benchmark is built so that nothing about the target degrades with d — the marginal field factorizes over blocks, so the true flow map stays exactly representable — which is what makes a decline here a statement about estimation rather than about expressiveness. The last row is one exact RK4 solve of the empirical field per dimension, in a single run with no spread, and its memorization ratio is why it is a floor and not a competitor.

method	NFE	$d = 2$	$d = 8$	$d = 32$	$d = 64$	$d = 128$
energy distance $\downarrow$						
CFM + Euler	8	0.0366 ± 0.0033	0.0503 ± 0.0063	0.0206 ± 0.0024	0.0314 ± 0.0019	0.0570 ± 0.0026
CFM + RK4	4	0.0269 ± 0.0033	0.0224 ± 0.0064	0.0929 ± 0.0052	0.0723 ± 0.0024	0.0601 ± 0.0025
CFM + RK4	100	0.00233 ± 0.00093	0.00574 ± 0.0015	0.0260 ± 0.0022	0.0321 ± 0.0014	0.0378 ± 0.0021
FLOWPINN-P	1	0.00310 ± 0.0010	0.304 ± 0.062	2.67 ± 0.22	3.01 ± 0.084	3.97 ± 0.075
FLOWPINN-L	1	0.00295 ± 0.00078	0.00584 ± 0.0014	0.0130 ± 0.0015	0.0162 ± 0.00082	0.0699 ± 0.0030
FLOWPINN-C	1	0.00293 ± 0.00084	0.00636 ± 0.0016	0.0269 ± 0.0026	0.0451 ± 0.0031	0.0663 ± 0.0034
FLOWPINN-2T	1	0.00471 ± 0.0017	0.00764 ± 0.0022	0.0250 ± 0.0069	0.106 ± 0.0085	0.295 ± 0.0072
FLOWPINN-2T	4	0.00967 ± 0.0018	0.00648 ± 0.0015	0.0150 ± 0.0046	0.0320 ± 0.0028	0.0293 ± 0.0019
MEANFLOW	1	0.0100 ± 0.0035	0.00673 ± 0.0022	0.0120 ± 0.0012	0.0787 ± 0.0075	0.309 ± 0.0092
MEANFLOW	4	0.0178 ± 0.0063	0.00980 ± 0.0029	0.0144 ± 0.0030	0.0178 ± 0.0013	0.0452 ± 0.0041
SHORTCUT	1	0.0126 ± 0.0023	0.0143 ± 0.0040	0.398 ± 0.57	2.87 ± 0.012	4.04 ± 0.015
SHORTCUT	4	0.00534 ± 0.0016	0.0110 ± 0.0028	0.0191 ± 0.0056	0.119 ± 0.013	0.353 ± 0.026
plug-in flow $\widehat{u}_t$	200	0.00198	0.00591	0.0116	0.0157	0.0221
memorization ratio (≈ 1 healthy, $\ll 1$ recall)						
CFM + Euler	8	1.16 ± 0.023	1.04 ± 0.0088	0.983 ± 0.0034	0.975 ± 0.0017	0.969 ± 0.0010
CFM + RK4	4	1.24 ± 0.027	1.05 ± 0.0091	1.09 ± 0.0038	1.06 ± 0.0018	1.04 ± 0.0011
CFM + RK4	100	1.11 ± 0.023	1.05 ± 0.0097	1.04 ± 0.0029	1.03 ± 0.0017	1.02 ± 0.0012
FLOWPINN-P	1	1.05 ± 0.024	1.08 ± 0.0064	0.767 ± 0.032	0.781 ± 0.010	0.780 ± 0.0025
FLOWPINN-L	1	1.20 ± 0.027	1.10 ± 0.0053	0.996 ± 0.0026	1.01 ± 0.0023	1.04 ± 0.0014
FLOWPINN-C	1	1.18 ± 0.028	1.08 ± 0.0039	0.975 ± 0.0029	0.964 ± 0.0019	0.963 ± 0.0012
FLOWPINN-2T	1	1.15 ± 0.017	1.04 ± 0.0056	0.974 ± 0.0082	0.941 ± 0.0034	0.918 ± 0.0010
FLOWPINN-2T	4	1.06 ± 0.018	1.00 ± 0.0086	1.02 ± 0.0085	1.03 ± 0.0024	1.02 ± 0.0012
MEANFLOW	1	1.12 ± 0.030	1.05 ± 0.0046	1.00 ± 0.0020	0.951 ± 0.0027	0.918 ± 0.0012
MEANFLOW	4	1.04 ± 0.016	1.04 ± 0.0074	1.01 ± 0.014	0.992 ± 0.0020	0.974 ± 0.0016
SHORTCUT	1	1.84 ± 0.13	1.20 ± 0.013	0.920 ± 0.056	0.793 ± 0.0030	0.783 ± 0.0013
SHORTCUT	4	1.26 ± 0.035	1.11 ± 0.0073	0.987 ± 0.0091	0.940 ± 0.0029	0.913 ± 0.0029
plug-in flow $\widehat{u}_t$	200	0.623	0.0278	0.0100	0.00841	0.00757