Preconditioned Flow Matching

Shadab Ahamed
University of British Columbia,
Vancouver, Canada
shadab.ahamed@hotmail.com

Eshed Gal
University of British Columbia,
Vancouver, Canada
eshedg@cs.ubc.ca

Md Shahriar Rahim Siddiqui
University of British Columbia,
Vancouver, Canada
shahriar.siddiqui@ubc.ca

Simon Ghyselincks
University of British Columbia,
Vancouver, Canada
sghyseli@cs.ubc.ca

Moshe Eliasof
Ben-Gurion University of the Negev,
Beersheba, Israel
eliasof@bgu.ac.il

Eldad Haber
University of British Columbia,
Vancouver, Canada
ehaber@eoas.ubc.ca

Project page: https://ahxmeds.github.io/preconditioned-flow-matching/

Abstract

Flow matching (FM) learns vector fields by regressing stochastic velocity targets along intermediate distributions p_t. We identify a geometric optimization bottleneck in this regression problem: when the covariance Σ_t of p_t is ill-conditioned, gradient-based training rapidly fits high-variance directions while making slow progress along low-variance ones. In an exactly solvable Gaussian setting, we prove that the excess risk is weighted by Σ_t, and that both gradient descent and stochastic gradient descent inherit condition-number-dependent convergence. We then extend the analysis to Gaussian mixtures, showing that multimodality does not average away this effect; instead, the slowest and worst-conditioned component can control optimization. Motivated by this analysis, we propose *preconditioned flow matching*, a precondition-then-match framework that transforms the target distribution into a more isotropic representation, trains the main flow in the transformed space, and maps generated samples back through the inverse transformation. We show theoretically that preconditioning reshapes the intermediate FM path and improves its conditioning. Across controlled Gaussian and Gaussian-mixture experiments, latent MNIST and other high resolution image datasets up to 512×512 resolution, preconditioning improves path-conditioning diagnostics, low-eigenvalue recovery, FID, MMD, precision, and recall. Compute-matched baselines and preconditioner-quality ablations further show that the gains are not explained merely by additional preconditioner parameters, but by improved geometry of the downstream flow matching problem. GitHub: https://github.com/ahxmeds/preconditioned-flow-matching.git.

1 Introduction

Flow matching [Lipman et al., 2023] and score-based diffusion models [Song et al., 2021, Ho et al., 2020] have become central tools for continuous-time generative modeling, achieving state-of-the-art performance across image [Caetano et al., 2026], audio [Guan et al., 2024], and 3D synthesis [Wang et al., 2025, Danese et al., 2026]. In both paradigms, generation is framed as the process of transporting samples from a simple reference distribution to a complex data distribution by learning a

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

time-dependent dynamical system. Despite their empirical success, these models exhibit a persistent and increasingly well-documented optimization phenomenon: *training loss often plateaus long before sample quality saturates*, suggesting that convergence of the aggregate objective can conceal errors in aspects of the learned dynamics that are important for generation [Lin and Yang, 2025, Xu et al., 2026].

Existing mitigation strategies, such as carefully tuned noise schedules [Aranguri et al., 2025], time-dependent loss reweighting [Billera et al., 2026, Diefenbacher et al., 2020], or probability flow distillation [Zhou et al., 2025], can improve sample quality in practice. However, these methods primarily modify the training schedule, loss weighting, or sampling procedure, rather than directly addressing the conditioning of the regression problems induced along the flow matching path. This distinction motivates our focus on the geometry of the intermediate distributions themselves. We discuss additional related work in Appendix A.

In this work, we show that a key contributor to slow convergence lies in the *conditioning* of the regression problems induced by flow matching objectives. At each time step, training reduces to the estimation of a vector field from samples, drawn from an intermediate distribution along the transport path. When these distributions exhibit strong anisotropy, the resulting optimization problem becomes ill-conditioned: *gradient-based methods rapidly fit high-variance directions while making only marginal progress along low-variance ones*.

To isolate and study this effect, we analyze flow matching in controlled settings where the exact transport dynamics are analytically tractable. We begin with a linear-Gaussian model problem, followed by a Gaussian-mixture extension that captures multimodality. In both cases, we show that optimization dynamics closely mirror those of gradient descent and stochastic gradient descent on ill-conditioned least-squares objectives, as commonly used in flow matching [Dao et al., 2023, Lipman et al., 2023, 2024]. Crucially, these difficulties arise even when the model class is expressive enough to represent the exact velocity field, demonstrating that optimization, rather than approximation, is the fundamental bottleneck in convergence.

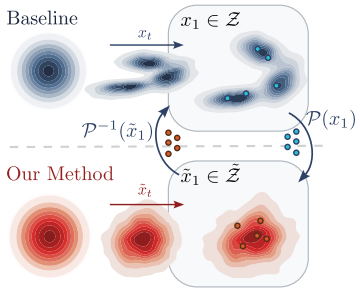


Figure 1: **Precondition-then-match. Top:** Standard flow matching learns through anisotropic intermediate distributions, which stagnates optimization. **Bottom:** A reversible preconditioner $\mathcal{P}$ maps the target to a more Gaussian-like space, improving the conditioning of the transport path. Samples are mapped back with $\mathcal{P}^{-1}$.

Motivated by this understanding, we propose a mitigation approach for flow matching: *preconditioning*. In its broader form, preconditioning refers to strategies that reshape the geometry of the learning problem without modifying the underlying generative process. This should be distinguished from standard latent generative modeling, where an autoencoder or VAE is used primarily to obtain a compact representation for efficient generation. Nevertheless, such latent spaces need not be isotropic or well-conditioned for the subsequent flow matching regression problem. Preconditioning is therefore complementary to latent flow matching and latent diffusion: it can be applied either in data space or on top of an existing latent representation to further improve the conditioning of the intermediate FM path. In our setting, this requires an additional preconditioning stage and associated design choices, *but the overhead is targeted at improving the geometry of the downstream flow matching problem rather than merely increasing model capacity*.

Contributions. Our main contributions are: (i) A theoretical analysis of the optimization process in flow matching, revealing how data anisotropy governs the optimization speed. (ii) A principled precondition-then-match framework that improves the conditioning of the downstream FM regression problem while leaving the main flow architecture and sampling ODE unchanged. (iii) Identification and derivation of FM-specific consequences of this regression geometry: small FM loss can hide large velocity error in low-variance directions; scalar time-reweighting cannot remove fixed-time directional stiffness, and preconditioning changes the covariance of the FM path itself rather than merely rescaling the objective. (iv) We experiment with a number of datasets, from 2D points to MNIST and other image datasets, and demonstrate the novelty of our method.

2 Flow Matching as Stochastic Regression

Continuous-time generative models construct a transport from a simple reference distribution p_0 (typically Gaussian) to a target data distribution p_1. Rather than learning a direct map $x_0 \mapsto x_1$, flow matching introduces a family of intermediate states

$$x_t = s(t)x_1 + c(t)x_0, \tag{1}$$

where $x_0 \sim p_0$, $x_1 \sim p_1$, and $s(0) = 0$, $s(1) = 1$, $c(0) = 1$, $c(1) = 0$. The goal is to learn a velocity field governing the evolution of the intermediate distributions p_t over time. Flow matching (FM) learns the right-hand side of the transport ODE [Lipman et al., 2023] $dx_t/dt = v_\theta(x_t, t)$. Given the interpolation path in Equation (1), the target velocity is $v_t^\star(x_0, x_1) = s'(t)x_1 + c'(t)x_0$. The standard conditional FM objective can be written as

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,x_0,x_1}\Big[\|v_\theta(x_t, t) - v_t^\star(x_0, x_1)\|^2\Big], \tag{2}$$

where $t \sim \mathrm{Unif}[0, 1]$, $x_0 \sim p_0$, and $x_1 \sim p_1$.

Proposition 2.1 (Conditional-mean characterization of CFM)**.** *For any measurable vector field $g : \mathbb{R}^d \times [0, 1] \to \mathbb{R}^d$, define*

$$\mathcal{L}(g) = \mathbb{E}\big[\|g(x_t, t) - v_t^\star(x_0, x_1)\|^2\big],$$

where the expectation is over $t \sim \mathrm{Unif}[0, 1]$, $x_0 \sim p_0$, and $x_1 \sim p_1$. Then the population minimizer is

$$g^\star(x, t) = \underbrace{\mathbb{E}\big[v_t^\star(x_0, x_1) \mid x_t = x,\, t\big]}_{\textit{conditional mean target}}. \tag{3}$$

Moreover,

$$\mathcal{L}(g) = \underbrace{\mathbb{E}\big[\|g(x_t, t) - g^\star(x_t, t)\|^2\big]}_{\textit{optimization / approximation error}} + \underbrace{\mathbb{E}[\mathrm{Tr}(\mathrm{Cov}(v_t^\star(x_0, x_1) \mid x_t, t))]}_{\textit{irreducible stochastic-target error}}. \tag{4}$$

The trace-of-covariance term in Equation (4) is the *irreducible stochastic-target error (ISTE)*. In particular, $\mathcal{L}(g) - \mathcal{L}(g^\star) = \mathbb{E}\big[\|g(x_t, t) - g^\star(x_t, t)\|^2\big]$ represents the *excess risk* (proof in Appendix D.1). Proposition 2.1 shows that conditional FM is a stochastic-target regression problem: the target velocity $v_t^\star(x_0, x_1)$ is generally not a deterministic function of (x_t, t), so the trace-of-covariance term in Equation (4) is an irreducible error floor, while the population minimizer remains the conditional mean velocity field. This establishes conditional FM as regression under the intermediate distributions p_t. In Section 3, we make the role of data geometry explicit in analytically tractable Gaussian and Gaussian-mixture flow matching settings.

3 Exact Conditioning Theory in Gaussian and Gaussian-Mixture FM

We now make the role of data geometry explicit in analytically tractable FM settings. Our goal in this section is not to prove a global convergence theorem for arbitrary nonlinear networks, but rather to isolate the optimization mechanism induced by the intermediate distributions p_t in settings where the population conditional FM target can be characterized exactly. We begin with a Gaussian model, where the target velocity is exactly linear, and then extend the analysis to a Gaussian mixture model, where multimodality sharpens the same conditioning bottleneck.

3.1 Gaussian FM as an exactly solvable regression problem

We consider the linear interpolation $x_t = (1 - t)x_0 + tx_1$ with $t \in [0, 1]$, $x_0 \sim \mathcal{N}(0, I), x_1 \sim \mathcal{N}(0, H)$, where $H \in \mathbb{R}^{d\times d}$ is symmetric positive definite and x_0 and x_1 are independent. Along the interpolation, the intermediate covariance transitions from I to H as t goes from 0 to 1, so the regression geometry gradually inherits the anisotropy of the target distribution.

We write the eigendecomposition, $H = U\Lambda U^\top$ with $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$ and $\lambda_i > 0$. We say that H is ill-conditioned when the condition number

$$\kappa(H) = \frac{\lambda_{\max}(H)}{\lambda_{\min}(H)} \tag{5}$$

is large.

Proposition 3.1 (Exact Gaussian CFM target and intermediate covariance)**.** *Under the linear interpolation, the intermediate state x_t is Gaussian with covariance*

$$\Sigma_t := \mathbb{E}[x_t x_t^\top] = (1-t)^2 I + t^2 H. \tag{6}$$

Moreover, for the linear interpolation, the conditional FM population target is

$$g_t^\star(x_t) := \mathbb{E}[x_1 - x_0 \mid x_t] = A^\star(t) x_t, \tag{7}$$

where

$$A^\star(t) = \mathrm{Cov}(x_1 - x_0, x_t)\, \Sigma_t^{-1} = \big(tH - (1-t)I\big)\big((1-t)^2 I + t^2 H\big)^{-1}. \tag{8}$$

Proposition 3.1 shows that in the Gaussian case, the population conditional FM target is exactly linear in x_t (proof in Appendix D.2). We now quantify the regression geometry induced by this exact target.

Lemma 3.2 (Exact excess-risk decomposition for linear predictors)**.** *Fix $t \in [0,1]$ and define*

$$\mathcal{L}_t(A) = \mathbb{E}\big[\|Ax_t - (x_1 - x_0)\|^2\big]. \tag{9}$$

Then the unique minimizer is $A^\star(t)$ from Equation (8)*, and for every $A \in \mathbb{R}^{d\times d}$,*

$$\mathcal{L}_t(A) = \underbrace{\mathcal{L}_t(A^\star(t))}_{\text{minimum achievable risk}} + \underbrace{\mathrm{Tr}\Big((A - A^\star(t))\Sigma_t(A - A^\star(t))^\top\Big)}_{\text{geometry-controlled excess risk}}. \tag{10}$$

Equivalently,

$$\mathcal{L}_t(A) = \underbrace{\mathbb{E}[\mathrm{Tr}(\mathrm{Cov}(x_1 - x_0 \mid x_t))]}_{\text{irreducible stochastic-label error}} + \underbrace{\mathrm{Tr}\Big((A - A^\star(t))\Sigma_t(A - A^\star(t))^\top\Big)}_{\text{geometry-controlled excess risk}}. \tag{11}$$

In particular,

$$\mathcal{L}_t(A) - \mathcal{L}_t(A^\star(t)) = \underbrace{\mathrm{Tr}\Big((A - A^\star(t))\Sigma_t(A - A^\star(t))^\top\Big)}_{\text{excess risk weighted by } \Sigma_t}. \tag{12}$$

Consequence. Lemma 3.2 identifies the exact population geometry of Gaussian flow matching: after removing the irreducible stochastic-target variance, the optimization term is a quadratic form governed by Σ_t (proof in Appendix D.3). The role of data geometry therefore enters directly through the spectrum of Σ_t.

Key observation: Even when the population conditional FM target is exactly representable, optimization can be arbitrarily slow due solely to the ill-conditioning of the intermediate covariance Σ_t, which suppresses learning along low-variance directions.

3.2 Conditioning controls optimization in Gaussian FM

Since Σ_t and H share the same eigenvectors, the eigenvalues of Σ_t are $\sigma_i(t) = (1-t)^2 + t^2\lambda_i$ with $i = 1, \ldots, d$. Hence,

$$\kappa(\Sigma_t) = \frac{\max_i \sigma_i(t)}{\min_i \sigma_i(t)} = \frac{(1-t)^2 + t^2\lambda_{\max}}{(1-t)^2 + t^2\lambda_{\min}}. \tag{13}$$

For $t = 0$, $\Sigma_t = I$ and the problem is perfectly conditioned. For $t = 1$, $\Sigma_t = H$, so the regression inherits the full anisotropy of the target distribution. For intermediate times, $\kappa(\Sigma_t)$ interpolates between these two regimes.

Theorem 3.3 (Gradient-descent convergence is controlled by Σ_t)**.** *Fix $t \in [0,1]$ and consider the quadratic population loss $\mathcal{L}_t(A)$ where $A \in \mathbb{R}^{d\times d}$ is the optimization variable, $x_t = (1-t)x_0 + tx_1$, and $\Sigma_t = \mathbb{E}[x_t x_t^\top]$. Let the cross-covariance be*

$$C_t := \mathrm{Cov}(x_1 - x_0, x_t) = tH - (1-t)I, \tag{14}$$

and let $A^\star(t)$ denote the unique minimizer of $\mathcal{L}_t$, given by $A^\star(t)\Sigma_t = C_t$. Consider full-batch gradient descent applied to $\mathcal{L}_t(A)$ with step size $\eta > 0$:

$$A_{k+1} = A_k - 2\eta\big(A_k\Sigma_t - C_t\big). \tag{15}$$

Define the error as $E_k := A_k - A^\star(t)$. Then the error evolves according to

$$E_{k+1} = E_k\big(I - 2\eta\Sigma_t\big). \tag{16}$$

Consequently, if $0 < \eta < 1/\sigma_{\max}(t)$, where $\sigma_{\max}(t)$ and $\sigma_{\min}(t)$ denote the largest and smallest eigenvalues of Σ_t, then gradient descent converges linearly to $A^\star(t)$. With the optimal fixed step size $\eta^\star = 1/(\sigma_{\max}(t) + \sigma_{\min}(t))$, the number of iterations required to reduce the error by a factor $\varepsilon \in (0, 1)$ satisfies

$$k = \mathcal{O}\Big(\kappa(\Sigma_t)\log(1/\varepsilon)\Big). \tag{17}$$

Proof of Theorem 3.3 is presented in Appendix D.4.

Corollary 3.4 (Low-variance directions are the Gaussian FM bottleneck)**.** *Under the setting of Theorem 3.3, directions corresponding to small eigenvalues $\sigma_i(t)$ converge most slowly under gradient descent. In particular, if H is ill-conditioned and t is not close to zero, then optimization becomes dominated by the low-variance directions of the intermediate distribution (proof in Appendix D.5).*

We next summarize the corresponding stochastic-gradient behavior.

Proposition 3.5 (Mode-wise SGD mean dynamics)**.** *Consider unbiased single-sample stochastic gradient descent applied to Equation* (9) *at a fixed time t. When the error is projected onto the eigenbasis of Σ_t, each scalar mode satisfies, in conditional expectation,*

$$\mathbb{E}[e_{i,m+1} \mid e_{i,m}] = (1 - 2\eta\sigma_i(t))e_{i,m}.$$

Thus the mean SGD dynamics inherit the same mode-dependent contraction as full-batch GD: directions with smaller $\sigma_i(t)$ contract more slowly. The constant-step stationary variance depends on the covariance of the stochastic gradient noise and does not, in general, scale universally as $1/\sigma_i(t)$ (proof in Appendix D.6).

Remark 3.6 (Scope of the optimizer analysis). Theorem 3.3 and Proposition 3.5 are exact population-level results for Gaussian flow matching under linear predictors. They should be interpreted as optimizer-level diagnostics for first-order training rather than as a global theorem for every adaptive optimizer or arbitrary nonlinear network. Their role is to isolate the conditioning mechanism that later motivates preconditioning.

3.3 Gaussian-mixture FM: multimodality worsens the bottleneck

We now extend the same analysis to a Gaussian mixture target,

$$x_1 \sim \sum_{k=1}^{K} \pi_k \mathcal{N}(0, H_k), \qquad \pi_k > 0, \qquad \sum_{k=1}^{K} \pi_k = 1, \tag{18}$$

with $x_0 \sim \mathcal{N}(0, I), x_t = (1-t)x_0 + tx_1$. The Gaussian-mixture setting is useful because it introduces multimodality while remaining analytically structured. It therefore provides a stylized model for datasets with multiple semantic clusters or modes.

Assumption 3.7 (Perfect component assignment)**.** For the purpose of isolating conditioning effects, we assume that the mixture responsibilities are known exactly, so that optimization decomposes componentwise.

Proposition 3.8 (Exact conditional-mean target in Gaussian-mixture FM)**.** *Under Equation* (18)*, let*

$$w_k(x_t, t) := \mathbb{P}(x_1 \textit{ belongs to component } k \mid x_t, t) \tag{19}$$

denote the posterior component weights. Then the conditional FM population target is

$$g_t^\star(x_t) = \mathbb{E}[x_1 - x_0 \mid x_t] = \sum_{k=1}^{K} w_k(x_t, t)\, A_k^\star(t)\, x_t, \tag{20}$$

where $A_k^\star(t) = \big(tH_k - (1-t)I\big)\big((1-t)^2 I + t^2 H_k\big)^{-1}$ (proof in Appendix E.1). Under Assumption 3.7, the problem decomposes into K componentwise regression subproblems with covariances $\Sigma_{t,k} = (1-t)^2 I + t^2 H_k$. If $\lambda_{k,i}$ denotes the ith eigenvalue of H_k, then the corresponding eigenvalues of $\Sigma_{t,k}$ are $\sigma_{k,i}(t) = (1-t)^2 + t^2\lambda_{k,i}$.

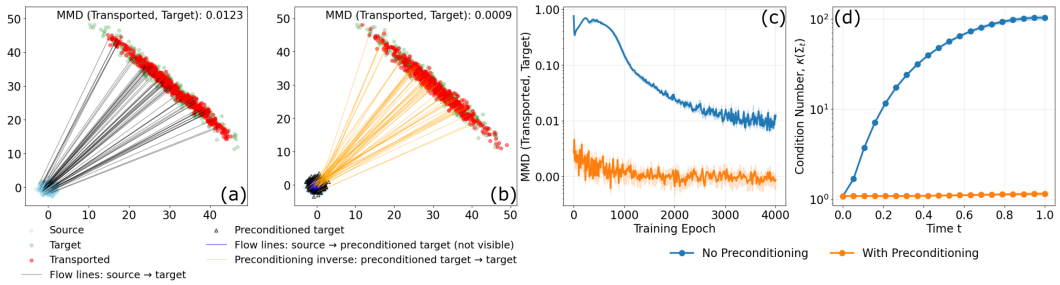


Figure 2: **2D Gaussian transport with and without preconditioning. (a)** Standard FM transports samples from an isotropic source to an elongated target, but might struggle along the narrow direction, resulting in a higher MMD. **(b)** Preconditioned FM transports samples to a whitened target and maps them back using the inverse preconditioner. **(c)** Preconditioning reduces MMD and avoids early optimization stagnation. **(d)** It also lowers the condition number of intermediate covariance Σ_t along the transport path, improving sample alignment and stability. Curves show the mean over 10 independent runs with different random seeds; shaded regions denote one standard deviation.

Theorem 3.9 (Worst-component bottleneck in Gaussian-mixture FM)**.** *Under Assumption 3.7, gradient-based optimization of the componentwise Gaussian-mixture FM problem is governed by the slowest componentwise subproblem. In particular, both full-batch GD and the mean dynamics of SGD are controlled by the slowest componentwise eigendirection,* $\min_{k,i} \sigma_{k,i}(t)$ *that is, by the smallest eigenvalue across all mixture components and directions. Consequently, even if most components are well-conditioned, a single poorly conditioned component can dominate the overall convergence time scale (proof in Appendix E.2).*

In particular, if one seeks comparable accuracy across all components under the decomposition in Assumption 3.7, the overall rate is governed by the slowest subproblem.

Corollary 3.10 (Multimodality amplifies conditioning effects)**.** *Under Theorem 3.9, multimodality does not average away conditioning effects: the overall optimization bottleneck is governed by the hardest mixture component (proof in Appendix E.3).*

Key observation: In Gaussian-mixture FM, optimization is controlled by the worst-conditioned component rather than by average data geometry. Even if most components are well-behaved, a single anisotropic component can dominate convergence, so multimodality sharpens rather than averages away the conditioning bottleneck.

Scope of the theory. The results above are exact for Gaussian and Gaussian-mixture flow matching at the population level. They show that, even when the target velocity is exactly representable, optimization can be slowed dramatically by anisotropy in the intermediate distributions. The zero-mean Gaussian assumption is only for exposition: Appendix E.4 extends the analysis to nonzero means, for which the conditional target becomes affine while the covariance-controlled excess-risk geometry and linear gradient dynamics remain unchanged after centering. This mechanism motivates preconditioning in the next section. Here, we do not claim a closed-form global convergence theorem for arbitrary nonlinear flow networks; rather, the purpose of this section is to isolate the conditioning effect in solvable settings and derive testable predictions for the practical models studied later.

4 Preconditioning for Flow Matching

The results of Section 3 show that optimization in FM is bottlenecked by the conditioning of the intermediate covariances Σ_t (for Gaussian) and $\Sigma_{t,k}$ (for Gaussian mixtures). This suggests a simple remedy: *before training the main FM model, apply an invertible map $\mathcal{P}$ to the target data so that the pushforward target distribution is more isotropic*. The main flow is then trained on the transformed transport $\mathcal{N}(0, I) \to \tilde{x}_1$ where $\tilde{x}_1 = \mathcal{P}x_1$, and samples are mapped back to the original space using $\mathcal{P}^{-1}$. In our setting, a preconditioner is any invertible or approximately invertible transformation (learned or analytically specified) that maps the target data to a representation with more isotropic geometry.

This is analogous to classical preconditioning in numerical linear algebra, where one improves optimization by transforming an ill-conditioned problem into a better-conditioned one (see Benzi [2002], Saad [1992], Golub and Ye [1999]). This design introduces an additional preconditioner-training stage and associated hyperparameters, but its role is targeted: it aims to improve the geometry of the downstream FM path rather than to simply increase the capacity of the generative model.

Theorem 4.1 (Preconditioning reshapes the Gaussian FM path)**.** *Consider the Gaussian FM setting of Section 3, where $x_0 \sim \mathcal{N}(0, I)$, $x_1 \sim \mathcal{N}(0, H)$, $x_t = (1-t)x_0 + tx_1$. Let $\mathcal{P} \in \mathbb{R}^{d\times d}$ be any invertible linear map, and define the transformed target and interpolation*

$$\tilde{x}_1 = \mathcal{P}x_1, \qquad \tilde{x}_t = (1-t)x_0 + t\tilde{x}_1.$$

Then the covariance of the transformed intermediate distribution is $\tilde{\Sigma}_t = (1-t)^2 I + t^2 \mathcal{P}H\mathcal{P}^\top$. Moreover:

(a) ***Exact whitening.*** *If $\mathcal{P} = H^{-1/2}$, then*

$$\tilde{\Sigma}_t = \big((1-t)^2 + t^2\big) I, \qquad \kappa(\tilde{\Sigma}_t) = 1 \quad \textit{for all } t \in [0,1]. \tag{21}$$

(b) ***Approximate isotropization.*** *If there exist constants $0 < m \le M$ such that $mI \preceq \mathcal{P}H\mathcal{P}^\top \preceq MI$, then $(1-t)^2 + t^2 m \;\le\; \lambda_i(\tilde{\Sigma}_t) \;\le\; (1-t)^2 + t^2 M$, and hence*

$$\kappa(\tilde{\Sigma}_t) \le \frac{(1-t)^2 + t^2 M}{(1-t)^2 + t^2 m}. \tag{22}$$

The proof and the corresponding componentwise GMM extension are provided in Appendices F.1 and F.2. Applying Theorem 3.3 to the transformed Gaussian FM problem shows that gradient-based optimization now scales with $\kappa(\tilde{\Sigma}_t)$ rather than $\kappa(\Sigma_t)$. In particular, exact whitening removes the conditioning bottleneck entirely, while approximate isotropization improves it in proportion to how well $\mathcal{P}H\mathcal{P}^\top$ is conditioned. The transformed regression problem is therefore better conditioned, mitigating early optimization stagnation that would otherwise lead to suboptimal solutions.

Algorithm 1 Precondition-then-Match

Input: Target samples $\{x_1^{(i)}\}_{i=1}^N \sim p_1$, reference $p_0 = \mathcal{N}(0, I)$, preconditioner class $\mathcal{P}_\phi$, main FM model v_θ.
Output: Generator for samples from p_1.

1. **Learn** an invertible or approximately invertible preconditioner $\mathcal{P}_\phi$ on target data.
2. **Transform** targets: $\tilde{x}_1^{(i)} = \mathcal{P}_\phi(x_1^{(i)})$.
3. **Train** v_θ by FM on the preconditioned path, $\tilde{x}_t = (1-t)x_0 + t\tilde{x}_1, x_0 \sim \mathcal{N}(0, I), t \sim \mathrm{Unif}[0,1]$.
4. **Minimize** $\mathbb{E}_{t,x_0,\tilde{x}_1}\left[\|v_\theta(\tilde{x}_t, t) - (\tilde{x}_1 - x_0)\|^2\right]$.
5. At inference, **sample** $z \sim \mathcal{N}(0, I)$ and solve the learned ODE in preconditioned space to obtain $\hat{\tilde{x}}_1$.
6. **Map back**: $\hat{x}_1 = \mathcal{P}_\phi^{-1}(\hat{\tilde{x}}_1)$.
7. **return** $\hat{x}_1$.

Interpretation. Theorem 4.1 makes the role of preconditioning explicit in the linear-Gaussian case: the intermediate covariance changes from Σ_t to $\tilde{\Sigma}_t$, with exact whitening as the analytically tractable ideal. For practical nonlinear preconditioners, the analogous goal is to make the transformed target distribution $\tilde{x}_1 = \mathcal{P}_\phi(x_1)$ more isotropic, so that the empirical intermediate distributions along the downstream FM path are better conditioned. Thus, the goal is not exact Gaussianization, but a transformed representation whose transport path is substantially better conditioned.

FM-specific implications. The preceding results have implications that are specific to flow matching, beyond the standard observation that ill-conditioned least-squares problems are slow to optimize.

With $H = U\Lambda U^\top$, let u_i denote the ith column of U. Since Σ_t and H share eigenvectors, Equation (12) gives

$$\mathcal{L}_t(A) - \mathcal{L}_t(A^\star(t)) = \sum_{i=1}^{d} \sigma_i(t) \left\|(A - A^\star(t))u_i\right\|^2, \qquad \sigma_i(t) = (1-t)^2 + t^2\lambda_i. \tag{23}$$

Thus the FM objective does not certify velocity-field accuracy uniformly across directions. In particular, if $\mathcal{L}_t(A) - \mathcal{L}_t(A^\star(t)) \le \varepsilon$, then

$$\left\|(A - A^\star(t))u_i\right\|^2 \le \frac{\varepsilon}{\sigma_i(t)}. \tag{24}$$

Consequently, when $\sigma_i(t)$ is small, *a small fixed-time FM excess loss may still be consistent with large velocity error in the corresponding low-variance direction*. This gives a mechanism for why the FM training loss can appear nearly saturated while sample quality continues to improve: *the remaining error may be concentrated in directions that are weakly weighted by p_t*. This complements recent work emphasizing that squared-error training objectives can be imperfect proxies for perceptual generation quality [Lin and Yang, 2025, Xu et al., 2026].

This also separates preconditioning from scalar time reweighting and schedule-based corrections, which are common tools for improving diffusion and FM models [Aranguri et al., 2025, Billera et al., 2026]. If the fixed-time objective is rescaled by a scalar weight $\alpha(t) > 0$, then the Hessian with respect to A is simply scaled by $\alpha(t)$, and its condition number remains governed by $\kappa(\Sigma_t)$. *Scalar time reweighting can change the relative importance of different times, but it cannot remove the directional stiffness of the regression problem at a fixed t. By contrast, Theorem 4.1 changes the covariance of the intermediate FM path itself, replacing Σ_t by $\tilde{\Sigma}_t$. Preconditioning therefore acts on the directional geometry of the FM regression problem, rather than merely rescaling the objective* [Benzi, 2002, Saad, 1992]. Full formal statements and proofs of these FM-specific consequences are provided in Corollaries F.3.1 and F.3.2 in Appendix F.3.

Normalizing flow preconditioner. A natural preconditioner is an invertible normalizing flow $\mathcal{P}_\theta$ trained by maximum likelihood to map data toward a standard Gaussian, $\tilde{x}_1 = \mathcal{P}_\theta(x_1)$, using the standard change-of-variables objective with log-determinant correction [Kobyzev et al., 2020]. Although normalizing flows are often less expressive than modern FM backbones, they are well suited for preconditioning: their invertibility allows us to transform the target data to a more isotropic representation, train the main FM model in that space, and map samples back with $\mathcal{P}_\theta^{-1}$.

Low-capacity/accuracy flow preconditioner. A second option is to use a low-capacity/accuracy FM model itself as a preconditioner. We first train a velocity field $v_\eta(x_t, t)$ using the standard FM objective. To map data samples backward through the learned flow, we solve the reverse-time ODE, $dx_\tau/d\tau = -v_\eta(x_\tau, 1-\tau), x_{\tau=0} \sim p_{t=1}, \tau \in [0,1]$. The preconditioned representation is the terminal state, $\tilde{x}_1 := \mathcal{P}_\eta(x_{\tau=0})$. Because the preconditioning flow has limited capacity, it captures only coarse structure, often producing a target representation that is substantially more Gaussian-like than the original data. We then train the main FM model on $\tilde{x}_1$ using the standard Gaussian as the reference distribution, and map generated samples back through $\mathcal{P}_\eta^{-1}$.

5 Experiments

We evaluate whether preconditioning improves flow matching in three regimes: (i) controlled low-dimensional problems where transport geometry can be inspected directly, (ii) latent-space MNIST generation where we can compare several preconditioning strategies and compute-matched baselines, and (iii) image synthesis across resolutions using flow-based preconditioning. Across these settings, our goal is not to introduce a new image backbone or claim state-of-the-art image generation, but to test the central prediction of the theory: *reshaping the target geometry should improve the conditioning of the intermediate flow matching regression problems and lead to better sample quality*.

Metrics. We report Fréchet Inception Distance (FID) and Maximum Mean Discrepancy (MMD) to measure distributional discrepancy, and Precision and Recall to measure complementary aspects of sample fidelity and distributional coverage. Lower is better for FID/MMD, and higher is better for Precision/Recall. Unless otherwise stated, all comparisons use the same downstream flow matching generator on a specific dataset.

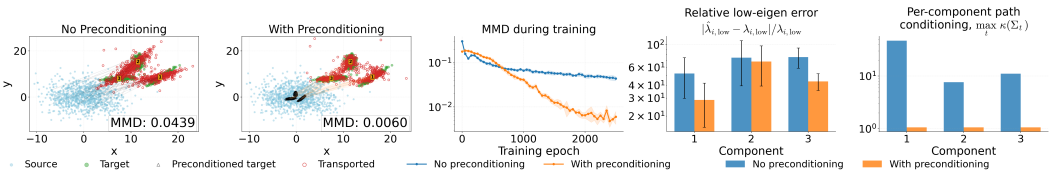


Figure 3: **Preconditioning improves Gaussian-mixture flow matching.** On a representative three-component anisotropic GMM, standard FM captures the dominant high-variance directions but poorly recovers the narrow components, leading to distorted transported samples and higher MMD. Preconditioning reshapes the target into a better-conditioned space, improves sample alignment, lowers MMD throughout training, reduces low-eigenvalue recovery error, and decreases the per-component path condition numbers. Plots show the mean over 10 independent runs with different random seeds; error bars / shaded regions denote one standard deviation. Experiments on additional GMM configurations are provided in Appendix G.2.

Controlled diagnostics. The theory predicts that anisotropic intermediate distributions slow optimization by suppressing progress along low-variance directions. The Gaussian experiment in Figure 2 visualizes this effect: standard FM fits the high-variance direction rapidly but stagnates along the narrow direction, while preconditioning improves path conditioning and reduces MMD. We further test the same mechanism on controlled Gaussian mixtures in Figure 3, with additional configurations in Appendix G.2. These experiments isolate the worst-component bottleneck predicted by Theorem 3.9: across mixtures with varying component numbers and condition numbers, preconditioning reduces the maximum path condition number, improves low-eigenvalue recovery, and lowers final MMD relative to standard FM. Additional Swiss-roll visualizations are provided in Appendix G.1.

Experiments on MNIST. We evaluate preconditioning on MNIST in the latent space of a trained VAE. The downstream generator is a class-conditional CNN-based latent FM model, kept fixed across the main variants to isolate the effect of preconditioning. We compare standard latent FM with no preconditioning, NF and FM preconditioning, while also reporting ZCA whitening [Kessy et al., 2018] and compute-matched unpreconditioned baselines in Table 1. Preconditioning consistently improves over the unpreconditioned baseline: NF gives the best FID/MMD, while FM preconditioning gives the best precision/recall. The compute-matched baseline improves only modestly, supporting that the gains come primarily from reshaping the latent geometry rather than simply adding parameters.

Table 1: MNIST comparison for preconditioning strategies. Compute-matched baselines enlarge the unpreconditioned FM generator to match the total parameters of learned preconditioner + FM. Lower is better for FID/MMD; higher is better for Precision/Recall.

Preconditioning method	**Compute matched?**	**# Params**	**FID** ↓	**MMD** ↓ ($\times 10^{-3}$)	**Precision** ↑	**Recall** ↑
No preconditioning	No	2.32M	5.97	77.10	0.5774	0.2856
No preconditioning	Yes	3.37M	5.42	75.46	0.5869	0.2618
ZCA whitening *(Ours)*	No	2.32M	4.20	6.25	0.6022	0.3107
ZCA whitening *(Ours)*	Yes	3.37M	4.02	7.02	0.6117	0.2885
NF preconditioner *(Ours)*	No	3.37M	**2.74**	**2.95**	0.7271	0.3561
FM preconditioner *(Ours)*	No	3.37M	3.98	3.02	**0.8440**	**0.5486**

Image synthesis across resolutions. We next evaluate whether flow-based preconditioning remains useful beyond MNIST. We use a UNet or DiT-based latent flow matching generator and compare standard flow matching against flow matching with a learned flow preconditioner (UNet or DiT-based). Experiments are conducted on LSUN Churches and Oxford Flowers-102 at 256×256, AFHQ Cats at 512×512, CIFAR-10 at 32×32 and ImageNet-1k at 64×64. Both methods operate in the latent space of a pretrained VAE for all datasets except CIFAR-10, where they operate in pixel space. For these datasets, we use flow-based preconditioning rather than normalizing flow preconditioning, since normalizing flows impose architectural and invertibility constraints that are less suitable for complex image data [Papamakarios et al., 2021]. As shown in Table 2, flow-based preconditioning improves FID on all five datasets and generally improves precision and recall.

Table 2: FID, Precision, and Recall across diverse image datasets. We compare standard flow matching with flow matching using a learned flow preconditioner. Experiments are conducted in the latent space of a pretrained VAE for all datasets except CIFAR-10, which is modeled directly in pixel space. Lower FID and higher Precision/Recall indicate better performance.

Dataset	Method	FID ↓	Precision ↑	Recall ↑
LSUN Churches (256×256)	No preconditioner	19.53	0.4540	0.3239
	FM preconditioner *(Ours)*	**14.47**	**0.5313**	**0.3255**
Oxford Flowers-102 (256×256)	No preconditioner	23.82	0.5213	0.2761
	FM preconditioner *(Ours)*	**22.64**	**0.5428**	**0.2854**
AFHQ Cats (512×512)	No preconditioner	7.11	0.7764	**0.5396**
	FM preconditioner *(Ours)*	**4.86**	**0.8320**	0.5297
CIFAR-10 (32×32)	No preconditioner	4.15	**0.8097**	0.5878
	FM preconditioner *(Ours)*	**4.02**	0.7856	**0.5892**
ImageNet-1k (64×64)	No preconditioner	8.66	0.7800	0.6339
	FM preconditioner *(Ours)*	**4.98**	**0.8172**	**0.6432**

Additional analyses. Optimal flow matching [Kornilov et al., 2024] and preconditioning address complementary sources of difficulty: the former reduces stochastic-target ambiguity, whereas the latter improves excess-risk optimization (Appendix G.3). This distinction highlights that improving the conditioning of the regression problem and reducing the intrinsic uncertainty of its training targets are separate, potentially complementary objectives. On latent MNIST, reduced path conditioning is accompanied by more balanced learning across high- and low-variance eigendirections (Appendices G.4 and G.5). These observations provide further empirical support for the proposed mechanism, connecting improvements in intermediate distribution geometry to more uniform optimization across directions. Capacity ablations show that the gains do not merely arise from additional model capacity (Appendix G.6), while NF preconditioning achieves lower FID than one-round Reflow at the same main-flow NFE (Appendix G.7). We further provide practical selection criteria and inverse-Jacobian sensitivity analyses (Appendices G.8 and G.9), and extend preconditioning to conditional deblurring and super-resolution tasks (Appendix G.10). These extensions illustrate the broader applicability of the preconditioning framework beyond image generation, while also examining practical considerations associated with selecting and inverting the preconditioner. Qualitative results, implementation details, and computational resources are reported in Appendices H to J; limitations and broader impact of our work are discussed in Appendices B and C.

6 Discussion and Conclusion

We analyzed flow matching as a time-indexed stochastic regression problem and showed that, in Gaussian and Gaussian-mixture settings, ill-conditioned intermediate distributions slow optimization along low-variance directions even when the target velocity is exactly representable. More broadly, this identifies the geometry of the intermediate transport distributions as an important factor governing how effectively flow matching models can be optimized, beyond considerations of model expressiveness alone. This perspective gives FM-specific implications: small FM loss may hide directional velocity errors, scalar time reweighting cannot remove fixed-time directional stiffness, and preconditioning helps by reshaping the path covariance from Σ_t to a better-conditioned $\tilde{\Sigma}_t$. Motivated by this analysis, we proposed a precondition-then-match framework that improves the geometry of the learning problem through a reversible transformation of the target distribution while preserving the standard downstream FM objective. Controlled and latent-space experiments support this mechanism through improved path conditioning, more balanced directional learning, and better sample quality across datasets. Taken together, our theoretical and empirical findings suggest that improving the representation in which transport is learned can offer meaningful advantages beyond simply increasing model capacity, highlighting preconditioning as a complementary direction for advancing flow-based generative modeling. Extending the analysis to the local geometry of nonlinear networks and developing adaptive preconditioners remain important directions for future work.

Acknowledgments and Disclosure of Funding

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) through its Discovery Grants program and Canada Graduate Scholarships – Master's (CGS-M) program. ME acknowledges support from the the Israel Science Foundation through a personal grant (ISF 3062/26) and an equipment grant (ISF 3066/26), as well as support from the Israeli Ministry of Innovation, Science & Technology. For this work, the authors declare no relevant competing interests.

References

Shadab Ahamed and Eldad Haber. Dawn-fm: Data-aware and noise-informed flow matching for solving inverse problems. *arXiv preprint arXiv:2412.04766*, 2024.

Shadab Ahamed and Arman Rahmim. Igconda-pet: Weakly-supervised pet anomaly detection using implicitly-guided attention-conditional counterfactual diffusion modeling–a multi-center, multi-cancer, and multi-tracer study. *Computerized Medical Imaging and Graphics*, page 102615, 2025.

Santiago Aranguri, Giulio Biroli, Marc Mezard, and Eric Vanden-Eijnden. Optimizing noise schedules of generative models in high dimensions. *arXiv preprint arXiv:2501.00988*, 2025.

Michele Benzi. Preconditioning techniques for large linear systems: A survey. *Journal of Computational Physics*, 182(2):418–477, 2002. ISSN 0021-9991. doi: https://doi.org/10.1006/jcph.2002.7176.

Lukas Billera, Hedwig Nora Nordlinder, and Ben Murrell. Time dependent loss reweighting for flow matching and diffusion models is theoretically justified. In *ICLR 2026 2nd Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*, 2026.

Francisco Caetano, Christiaan Viviers, Peter H.N. De With, and Fons Van der Sommen. Symmetrical flow matching: Unified image generation, segmentation, and classification with score-based generative models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40:2498–2506, 03 2026.

Wenbo Cao and Weiwei Zhang. An analysis and solution of ill-conditioning in physics-informed neural networks. *Journal of Computational Physics*, 520:113494, 2025. ISSN 0021-9991.

Shoufa Chen, Peize Sun, Yibing Song, and Ping Luo. Diffusiondet: Diffusion model for object detection. In *2023 IEEE/CVF international conference on computer vision (ICCV)*, pages 19773–19786. IEEE, 2023.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.

Hyungjin Chung and Jong Chul Ye. Score-based diffusion models for accelerated mri. *Medical image analysis*, 80:102479, 2022.

Danilo Danese, Angela Lombardi, Matteo Attimonelli, Giuseppe Fasano, and Tommaso Di Noia. Flowlet: Conditional 3d brain mri synthesis using wavelet flow matching. *arXiv preprint arXiv:2601.05212*, 2026.

Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. Flow matching in latent space. *arXiv preprint arXiv:2307.08698*, 2023.

Yann N. Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems*, 2014.

Sascha Diefenbacher, Engin Eren, Gregor Kasieczka, Anatolii Korol, Benjamin Nachman, and David Shih. Dctrgan: Improving the precision of generative models with reweighting. *Journal of Instrumentation*, 15(11):P11004–P11004, 2020.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. In *ICLR*, 2017.

Eshed Gal, Md Shahriar Rahim Siddiqui, Moshe Eliasof, and Eldad Haber. Conservative flows: A new paradigm of generative models. *arXiv preprint arXiv:2605.06905*, 2026.

Gene H Golub and Qiang Ye. Inexact preconditioned conjugate gradient method with inner-outer iteration. *SIAM Journal on Scientific Computing*, 21(4):1305–1320, 1999.

Wenhao Guan, Kaidi Wang, Wangjin Zhou, Yang Wang, Feng Deng, Hui Wang, Lin Li, Qingyang Hong, and Yong Qin. Lafma: A latent flow matching model for text-to-audio generation. *Interspeech*, 2024.

Eldad Haber, Shadab Ahamed, Md. Shahriar Rahim Siddiqui, Niloufar Zakariaei, and Moshe Eliasof. Iterative flow matching: Path correction and gradual refinement for enhanced generative modeling. *SIAM Journal on Scientific Computing*, 48(4):C814–C831, 2026. doi: 10.1137/25M1736633. URL `https://doi.org/10.1137/25M1736633`.

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

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

Agnan Kessy, Alex Lewin, and Korbinian Strimmer. Optimal whitening and decorrelation. *The American Statistician*, 72(4):309–314, 2018.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *International Conference on Learning Representations*, 2014.

Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43 (11):3964–3979, 2020.

Nikita Kornilov, Petr Mokrov, Alexander Gasnikov, and Alexander Korotin. Optimal flow matching: Learning straight trajectories in just one step. *Advances in Neural Information Processing Systems*, 37:104180–104204, 2024.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Lorenzo Lazzarino, Yuji Nakatsukasa, and Umberto Zerbinati. Preconditioned normal equations for solving discretised partial differential equations. *arXiv preprint arXiv:2502.17626*, 2025.

Shanchuan Lin and Xiao Yang. Diffusion model with perceptual loss. *arXiv preprint arXiv:2401.00110v7*, 2025.

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

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow, 2022. URL `https://arxiv.org/abs/2209.03003`.

Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 11451–11461. IEEE, 2022.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP)*, 2008.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 2021.

Rien Quirynen and Stefano Di Cairano. Presas: Block-structured preconditioning of iterative solvers within a primal active-set method for fast model predictive control. *Optimal Control Applications and Methods*, 41(6):2282–2307, 2020.

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

Yousef Saad. Analysis of some krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*, 29(1):209–228, 1992.

Yousef Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, 2003.

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.

Lloyd N. Trefethen and David Bau. *Numerical Linear Algebra*. SIAM, 1997.

Jiayi Wang, Hadrien Reynaud, Franciskus Xaverius Erick, and Bernhard Kainz. Ctflow: Video-inspired latent flow matching for 3d ct synthesis. *arXiv preprint arXiv:2508.12900*, 2025.

Yixian Xu, Shengjie Luo, Liwei Wang, Di He, and Chang Liu. Diagnosing and improving diffusion models by estimating the optimal loss value. In *International Conference on Learning Representations (ICLR)*, 2026.

Qiang Ye. Preconditioning for accelerated gradient descent optimization and regularization. *arXiv preprint arXiv:2410.00232v2*, 2024.

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Shuangfei Zhai, Ruixiang Zhang, Preetum Nakkiran, David Berthelot, Jiatao Gu, Huangjie Zheng, Tianrong Chen, Miguel Angel Bautista, Navdeep Jaitly, and Joshua M. Susskind. Normalizing flows are capable generative models. In *International Conference on Machine Learning (ICML)*, 2025.

Mingyuan Zhou, Yi Gu, Huangjie Zheng, Liangchen Song, Guande He, Yizhe Zhang, Wenze Hu, and Yinfei Yang. Score distillation of flow matching models. *arXiv preprint arXiv:2509.25127*, 2025.

Appendix Contents

A Additional Related Work **14**

B Limitation **15**

C Broader Impact **15**

D Proofs for Flow Matching as Stochastic Regression and Gaussian FM **15**

D.1 Proof of Proposition 2.1 15
D.2 Proof of Proposition 3.1 16
D.3 Proof of Lemma 3.2 17
D.4 Proof of Theorem 3.3 17
D.5 Proof of Corollary 3.4 19
D.6 Proof of Proposition 3.5 19

E Proofs for Gaussian-Mixture Flow Matching **21**

E.1 Proof of Proposition 3.8 21
E.2 Proof of Theorem 3.9 22
E.3 Proof of Corollary 3.10 22
E.4 Extension to nonzero-mean Gaussian and Gaussian-mixture FM 22

F Proofs for Preconditioning in Flow Matching **23**

F.1 Proof of Theorem 4.1 23
F.2 Additional note on Gaussian-mixture preconditioning 24
F.3 FM-specific consequences of the conditioning analysis 24

G Additional Experiments and Results **26**

G.1 Visualizing preconditioned transport for the Swiss roll data using a normalizing flow and a flow-based preconditioner 26
G.2 Controlled Gaussian-Mixture experiments: conditioning and low-eigenvalue recovery 28
G.3 Optimal flow matching reduces stochastic-target error; preconditioning improves excess-risk optimization 29
G.4 Condition number dynamics for MNIST and ImageNet-1k training 31
G.5 Mechanistic analysis of directional learning on latent MNIST 31
G.6 Preconditioner quality ablation: capacity and training iterations 34
G.7 Comparison with Rectified flow and Reflow 35
G.8 Operational preconditioner selection and limited-benefit regimes 36
G.9 Inverse-preconditioner Jacobian sensitivity 36
G.10 Preconditioning for conditional image restoration tasks 37

H Qualitative Comparison for Flow-Based Preconditioning **39**

H.1 MNIST 39
H.2 Image samples across resolutions 39

I Experimental Details **43**

I.1 MNIST 43
I.2 Image synthesis across resolutions 43

J Computational Resources Used **44**

K Existing Assets and Licenses **44**

A Additional Related Work

Generative models. Generative models learn complex data distributions to synthesize realistic samples. Early explicit density models, such as variational autoencoders and autoregressive models, use likelihood-based objectives [Kingma and Welling, 2014]. Normalizing flows and diffusion models map simple priors to complex targets via invertible maps and stochastic dynamics, respectively [Dinh et al., 2017, Ho et al., 2020]. TarFlow shows that normalizing flows are competitive for image synthesis [Zhai et al., 2025]. Flow matching learns a time-dependent velocity field for transport between distributions [Lipman et al., 2023]. To reduce errors from trajectory drift in regions poorly covered during training, Iterative Flow Matching learns endpoint corrections or gradually refines the transport [Haber et al., 2026]. Conservative Flows reuse pretrained flow models to generate variations of existing samples via stochastic updates designed to preserve the data distribution or a smoothed version of it [Gal et al., 2026]. Diffusion also supports conditional generation and domain-specific tasks in computer vision and medical imaging. Lugmayr et al. [2022] used pretrained diffusion for image inpainting, while Chen et al. [2023] cast object detection as iterative bounding-box denoising. Chung and Ye [2022] used score-based diffusion for accelerated MRI reconstruction, an ill-posed inverse problem, while Ahamed and Rahmim [2025] used class-conditional counterfactual diffusion for weakly supervised anomaly detection in PET. Together, these applications highlight the versatility of diffusion-based generative modeling beyond conventional image synthesis, spanning restoration, recognition, reconstruction, and anomaly detection.

Preconditioning. Preconditioning is a classical technique in numerical analysis aimed at improving the conditioning of optimization problems [Ye, 2024] and differential equations [Lazzarino et al.,

2025], thereby accelerating convergence and enhancing numerical stability [Saad, 2003, Trefethen and Bau, 1997]. By rescaling variables or operators to reduce anisotropy and curvature disparities, preconditioning enables more efficient iterative solvers and more stable time integration [Quirynen and Di Cairano, 2020]. Similar issues arise in machine learning, where poorly conditioned objectives can hinder optimization and lead to slow or unstable training dynamics [Cao and Zhang, 2025]. Consequently, preconditioning has been widely adopted in large-scale learning systems, both implicitly through adaptive optimization methods and explicitly through problem-dependent transformations, yielding substantial improvements in convergence speed and robustness [Dauphin et al., 2014]. In the context of diffusion models, recent work has shown that appropriate reparameterization and noise-dependent preconditioning can substantially improve optimization behavior and sample quality, highlighting the critical role of conditioning in generative model training [Karras et al., 2022].

B Limitation

Our analysis isolates the conditioning mechanism in Gaussian and Gaussian-mixture flow matching settings, where the population targets and optimization dynamics can be characterized exactly. These models are intentionally simplified: they expose how anisotropic intermediate distributions can slow first-order optimization, but they do not provide a global convergence theory for arbitrary nonlinear neural networks, adaptive optimizers, or large-scale generative backbones. In practical image models, the relevant geometry is local, high-dimensional, and changes during training, so our condition-number diagnostics should be viewed as empirical evidence for the proposed mechanism rather than a complete spectral characterization of the full training problem.

Preconditioning also introduces an additional training stage, with choices such as preconditioner architecture, capacity, training duration, and inverse accuracy. Our compute-matched experiments and ablations suggest that the gains are not explained merely by extra parameters, but the best cost–quality trade-off for the preconditioner remains task-dependent. Both standard and preconditioned flow matching use numerical ODE integration at sampling time, where each solver step introduces discretization error that can propagate along the trajectory. Flow-based preconditioning requires an additional inverse ODE solve, creating a further source of solver-dependent error. Finally, our experiments cover controlled distributions, latent MNIST, and several image datasets, but broader evaluation on larger-scale conditional generation, stronger production-level backbones, and standardized compute budgets would further clarify when preconditioning is most beneficial. Future work could also study time-dependent or locally adaptive preconditioners, which may better match the changing geometry of the flow matching path.

C Broader Impact

This work is primarily theoretical and methodological, focusing on the optimization geometry of flow matching. Potential positive impacts include more stable and efficient generative modeling methods, which may benefit scientific modeling, image synthesis, and representation learning. As with other advances in generative modeling, possible negative impacts include misuse for generating misleading or deceptive synthetic content. The paper does not target such applications, does not introduce a deployed system, and uses standard public datasets and controlled synthetic experiments.

D Proofs for Flow Matching as Stochastic Regression and Gaussian FM

D.1 Proof of Proposition 2.1

We prove the standard conditional-mean characterization of the conditional flow matching objective. Define

$$Y := v_t^\star(x_0, x_1), \qquad X := (x_t, t), \tag{25}$$

so that the population objective can be written as

$$\mathcal{L}(g) = \mathbb{E}\big[\|g(X) - Y\|^2\big]. \tag{26}$$

Let

$$g^\star(X) := \mathbb{E}[Y \mid X]. \tag{27}$$

Then

$$Y = g^\star(X) + \big(Y - g^\star(X)\big), \tag{28}$$

where by construction

$$\mathbb{E}\big[Y - g^\star(X) \mid X\big] = 0. \tag{29}$$

Hence,

$$\mathcal{L}(g) = \mathbb{E}\big[\|g(X) - g^\star(X) + g^\star(X) - Y\|^2\big] \tag{30}$$

$$= \mathbb{E}\big[\|g(X) - g^\star(X)\|^2\big] + \mathbb{E}\big[\|Y - g^\star(X)\|^2\big] + 2\,\mathbb{E}\big[\langle g(X) - g^\star(X),\, g^\star(X) - Y\rangle\big]. \tag{31}$$

The cross term vanishes because conditioning on X gives

$$\mathbb{E}\big[\langle g(X) - g^\star(X),\, g^\star(X) - Y\rangle\big] \tag{32}$$

$$= \mathbb{E}\Big[\big\langle g(X) - g^\star(X),\, \mathbb{E}[g^\star(X) - Y \mid X]\big\rangle\Big] = 0. \tag{33}$$

Therefore,

$$\mathcal{L}(g) = \mathbb{E}\big[\|g(X) - g^\star(X)\|^2\big] + \mathbb{E}\big[\|Y - g^\star(X)\|^2\big]. \tag{34}$$

Since

$$\mathbb{E}\big[\|Y - g^\star(X)\|^2\big] = \mathbb{E}[\mathrm{Tr}(\mathrm{Cov}(Y \mid X))]\,, \tag{35}$$

we obtain

$$\mathcal{L}(g) = \mathbb{E}\big[\|g(x_t, t) - g^\star(x_t, t)\|^2\big] + \mathbb{E}[\mathrm{Tr}(\mathrm{Cov}(v_t^\star(x_0, x_1) \mid x_t, t))]\,, \tag{36}$$

which proves Equation (4). In particular,

$$\mathcal{L}(g) - \mathcal{L}(g^\star) = \mathbb{E}\big[\|g(x_t, t) - g^\star(x_t, t)\|^2\big], \tag{37}$$

and therefore $g^\star$ is the unique population minimizer up to almost-sure equivalence.

D.2 Proof of Proposition 3.1

Assume $x_0 \sim \mathcal{N}(0, I), x_1 \sim \mathcal{N}(0, H), x_t = (1-t)x_0 + tx_1$, with x_0 and x_1 independent, so $\mathbb{E}[x_0 x_1^\top] = \mathbb{E}[x_1 x_0^\top] = 0$.

Step 1: covariance of the intermediate state. Since x_0 and x_1 are independent and zero mean,

$$\Sigma_t := \mathbb{E}[x_t x_t^\top] \tag{38}$$

$$= \mathbb{E}\Big[\big((1-t)x_0 + tx_1\big)\big((1-t)x_0 + tx_1\big)^\top\Big] \tag{39}$$

$$= (1-t)^2\mathbb{E}[x_0 x_0^\top] + t^2\mathbb{E}[x_1 x_1^\top] + t(1-t)\mathbb{E}[x_0 x_1^\top] + t(1-t)\mathbb{E}[x_1 x_0^\top] \tag{40}$$

$$= (1-t)^2 I + t^2 H. \tag{41}$$

Step 2: conditional mean target. Let $y := x_1 - x_0$. Because (y, x_t) is jointly Gaussian and zero mean, the conditional expectation is linear:

$$\mathbb{E}[y \mid x_t] = \mathrm{Cov}(y, x_t)\,\Sigma_t^{-1} x_t. \tag{42}$$

It remains to compute the cross-covariance:

$$\mathrm{Cov}(x_1 - x_0, x_t) = \mathbb{E}\big[(x_1 - x_0)\big((1-t)x_0 + tx_1\big)^\top\big] \tag{43}$$

$$= (1-t)\mathbb{E}[(x_1 - x_0)x_0^\top] + t\mathbb{E}[(x_1 - x_0)x_1^\top]. \tag{44}$$

Using independence and zero mean,

$$\mathbb{E}[(x_1 - x_0)x_0^\top] = -I, \qquad \mathbb{E}[(x_1 - x_0)x_1^\top] = H. \tag{45}$$

Hence,

$$\mathrm{Cov}(x_1 - x_0, x_t) = tH - (1-t)I. \tag{46}$$

Therefore,

$$g_t^\star(x_t) = \mathbb{E}[x_1 - x_0 \mid x_t] = A^\star(t)x_t, \tag{47}$$

with

$$A^\star(t) = \big(tH - (1-t)I\big)\big((1-t)^2 I + t^2 H\big)^{-1}. \tag{48}$$

This proves Proposition 3.1.

D.3 Proof of Lemma 3.2

Fix $t \in [0,1]$ and define

$$\mathcal{L}_t(A) = \mathbb{E}\big[\|Ax_t - (x_1 - x_0)\|^2\big]. \tag{49}$$

Let

$$y := x_1 - x_0, \qquad g_t^\star(x_t) = A^\star(t)x_t = \mathbb{E}[y \mid x_t]. \tag{50}$$

By Proposition 2.1, since $g_A(x_t) = Ax_t$ is a measurable predictor,

$$\mathcal{L}_t(A) = \mathbb{E}\big[\|Ax_t - A^\star(t)x_t\|^2\big] + \mathbb{E}[\mathrm{Tr}(\mathrm{Cov}(y \mid x_t))]\,. \tag{51}$$

Because

$$Ax_t - A^\star(t)x_t = (A - A^\star(t))x_t, \tag{52}$$

we have

$$\begin{aligned}
\mathbb{E}\big[\|(A - A^\star(t))x_t\|^2\big] &= \mathbb{E}\big[x_t^\top (A - A^\star(t))^\top (A - A^\star(t))x_t\big] && (53)\\
&= \mathbb{E}\big[\,\mathrm{Tr}\left(x_t^\top (A - A^\star(t))^\top (A - A^\star(t))x_t\right)\big] && (54)\\
&= \mathrm{Tr}\Big((A - A^\star(t))^\top (A - A^\star(t))\,\mathbb{E}[x_t x_t^\top]\Big) && (55)\\
&= \mathrm{Tr}\Big((A - A^\star(t))^\top (A - A^\star(t))\,\Sigma_t\Big) && (56)\\
&= \mathrm{Tr}\Big((A - A^\star(t))\Sigma_t (A - A^\star(t))^\top\Big), && (57)
\end{aligned}$$

where the last equality uses cyclic invariance of the trace. Substituting this into Equation (51) gives

$$\mathcal{L}_t(A) = \mathbb{E}[\mathrm{Tr}(\mathrm{Cov}(x_1 - x_0 \mid x_t))] + \mathrm{Tr}\Big((A - A^\star(t))\Sigma_t (A - A^\star(t))^\top\Big), \tag{58}$$

which proves Equation (11).

Since $\Sigma_t \succ 0$, the quadratic trace term is nonnegative and vanishes only when $A = A^\star(t)$. Therefore $A^\star(t)$ is the unique minimizer. Evaluating the previous identity at $A = A^\star(t)$ gives

$$\mathcal{L}_t(A^\star(t)) = \mathbb{E}[\mathrm{Tr}(\mathrm{Cov}(x_1 - x_0 \mid x_t))]\,. \tag{59}$$

Subtracting this identity from the expression for $\mathcal{L}_t(A)$ yields

$$\mathcal{L}_t(A) - \mathcal{L}_t(A^\star(t)) = \mathrm{Tr}\Big((A - A^\star(t))\Sigma_t (A - A^\star(t))^\top\Big), \tag{60}$$

which proves Equation (10) and Equation (12).

D.4 Proof of Theorem 3.3

Recall the population loss

$$\mathcal{L}_t(A) = \mathbb{E}\big[\|Ax_t - (x_1 - x_0)\|^2\big]. \tag{61}$$

Let

$$C_t := \mathrm{Cov}(x_1 - x_0, x_t) = tH - (1-t)I. \tag{62}$$

Using the standard matrix derivative identity,

$$\nabla_A \mathcal{L}_t(A) = 2(A\Sigma_t - C_t). \tag{63}$$

Therefore full-batch gradient descent with step size η is

$$A_{k+1} = A_k - 2\eta(A_k\Sigma_t - C_t). \tag{64}$$

Let

$$E_k := A_k - A^\star(t), \tag{65}$$

where $A^\star(t)\Sigma_t = C_t$. Then

$$\begin{aligned}
E_{k+1} &= A_{k+1} - A^\star(t) && (66)\\
&= A_k - 2\eta(A_k\Sigma_t - C_t) - A^\star(t) && (67)\\
&= A_k - A^\star(t) - 2\eta(A_k - A^\star(t))\Sigma_t && (68)\\
&= E_k(I - 2\eta\Sigma_t). && (69)
\end{aligned}$$

This proves Equation (16). Hence, for a fixed step size η, the error recursion is

$$E_{k+1} = E_k(I - 2\eta\Sigma_t). \tag{70}$$

Since Σ_t is symmetric positive definite, let

$$\Sigma_t = U \operatorname{diag}(\sigma_1(t), \ldots, \sigma_d(t))U^\top, \tag{71}$$

where

$$0 < \sigma_{\min}(t) \le \sigma_i(t) \le \sigma_{\max}(t). \tag{72}$$

Then

$$I - 2\eta\Sigma_t = U \operatorname{diag}\big(1 - 2\eta\sigma_1(t), \ldots, 1 - 2\eta\sigma_d(t)\big)U^\top. \tag{73}$$

Thus each eigendirection contracts independently by a factor

$$|1 - 2\eta\sigma_i(t)|. \tag{74}$$

The worst-case contraction factor for a given η is therefore

$$\rho(\eta) = \max_i |1 - 2\eta\sigma_i(t)| = \max\Big\{|1 - 2\eta\sigma_{\min}(t)|,\ |1 - 2\eta\sigma_{\max}(t)|\Big\}. \tag{75}$$

Convergence requires $\rho(\eta) < 1$, which is equivalent to

$$0 < \eta < \frac{1}{\sigma_{\max}(t)}. \tag{76}$$

The optimal fixed step size minimizes the worst-case contraction factor

$$\rho(\eta) = \max\Big\{|1 - 2\eta\sigma_{\min}(t)|,\ |1 - 2\eta\sigma_{\max}(t)|\Big\}. \tag{77}$$

At the optimum, the two endpoint contractions are balanced:

$$1 - 2\eta^\star\sigma_{\min}(t) = -\big(1 - 2\eta^\star\sigma_{\max}(t)\big). \tag{78}$$

Solving gives

$$1 - 2\eta^\star\sigma_{\min}(t) = -1 + 2\eta^\star\sigma_{\max}(t), \tag{79}$$

and hence

$$2 = 2\eta^\star\big(\sigma_{\max}(t) + \sigma_{\min}(t)\big). \tag{80}$$

Therefore,

$$\eta^\star = \frac{1}{\sigma_{\max}(t) + \sigma_{\min}(t)}. \tag{81}$$

Substituting this step size into the contraction factor gives

$$\rho^\star = 1 - 2\eta^\star\sigma_{\min}(t) = 1 - \frac{2\sigma_{\min}(t)}{\sigma_{\max}(t) + \sigma_{\min}(t)}. \tag{82}$$

Thus

$$\rho^\star = \frac{\sigma_{\max}(t) - \sigma_{\min}(t)}{\sigma_{\max}(t) + \sigma_{\min}(t)}. \tag{83}$$

Equivalently, using

$$\kappa(\Sigma_t) = \frac{\sigma_{\max}(t)}{\sigma_{\min}(t)}, \tag{84}$$

we obtain

$$\rho^\star = \frac{\frac{\sigma_{\max}(t)}{\sigma_{\min}(t)} - 1}{\frac{\sigma_{\max}(t)}{\sigma_{\min}(t)} + 1} = \frac{\kappa(\Sigma_t) - 1}{\kappa(\Sigma_t) + 1}. \tag{85}$$

After k gradient-descent steps,

$$\|E_k\| \le (\rho^\star)^k \|E_0\|. \tag{86}$$

To reduce the error by a factor $\varepsilon \in (0, 1)$, it is sufficient that

$$(\rho^\star)^k \le \varepsilon. \tag{87}$$

Taking logarithms gives

$$k \log(\rho^\star) \leq \log(\varepsilon). \tag{88}$$

Since $\rho^\star \in (0,1)$, we have $\log(\rho^\star) < 0$, and therefore

$$k \geq \frac{\log(1/\varepsilon)}{-\log(\rho^\star)}. \tag{89}$$

Now

$$\rho^\star = \frac{\kappa(\Sigma_t) - 1}{\kappa(\Sigma_t) + 1} = 1 - \frac{2}{\kappa(\Sigma_t) + 1}. \tag{90}$$

Using the standard bound $-\log(1-a) \geq a$ for $a \in (0,1)$, with

$$a = \frac{2}{\kappa(\Sigma_t) + 1}, \tag{91}$$

we obtain

$$-\log(\rho^\star) = -\log\left(1 - \frac{2}{\kappa(\Sigma_t) + 1}\right) \geq \frac{2}{\kappa(\Sigma_t) + 1}. \tag{92}$$

Hence

$$\frac{\log(1/\varepsilon)}{-\log(\rho^\star)} \leq \frac{\kappa(\Sigma_t) + 1}{2} \log(1/\varepsilon). \tag{93}$$

Therefore, it is sufficient to choose

$$k \geq \frac{\kappa(\Sigma_t) + 1}{2} \log(1/\varepsilon). \tag{94}$$

Thus the number of iterations required to reduce the error by a factor ε scales as

$$k = \mathcal{O}(\kappa(\Sigma_t) \log(1/\varepsilon)). \tag{95}$$

This proves Theorem 3.3.

D.5 Proof of Corollary 3.4

By Theorem 3.3, the GD error evolves modewise according to the spectrum of Σ_t. The slowest mode corresponds to the smallest eigenvalue $\sigma_{\min}(t)$, so directions associated with small $\sigma_i(t)$ converge most slowly. If H is ill-conditioned and t is not close to zero, then

$$\sigma_i(t) = (1-t)^2 + t^2 \lambda_i \tag{96}$$

inherits the disparity in the eigenvalues of H, and therefore the optimization becomes dominated by the low-variance directions of the intermediate distribution. This proves the corollary.

D.6 Proof of Proposition 3.5

We derive the stochastic-gradient dynamics for the Gaussian FM regression problem at a fixed time t. For a single sample $(x_0^{(m)}, x_1^{(m)})$, define

$$x_t^{(m)} = (1-t)x_0^{(m)} + t x_1^{(m)}, \qquad y^{(m)} = x_1^{(m)} - x_0^{(m)}. \tag{97}$$

The single-sample loss is

$$\ell_m(A) = \|A x_t^{(m)} - y^{(m)}\|^2, \tag{98}$$

with gradient

$$\nabla_A \ell_m(A) = 2(A x_t^{(m)} - y^{(m)})(x_t^{(m)})^\top. \tag{99}$$

Thus SGD with step size η gives

$$A_{m+1} = A_m - 2\eta(A_m x_t^{(m)} - y^{(m)})(x_t^{(m)})^\top. \tag{100}$$

Let $A^\star = A^\star(t)$ denote the population minimizer and define

$$E_m = A_m - A^\star. \tag{101}$$

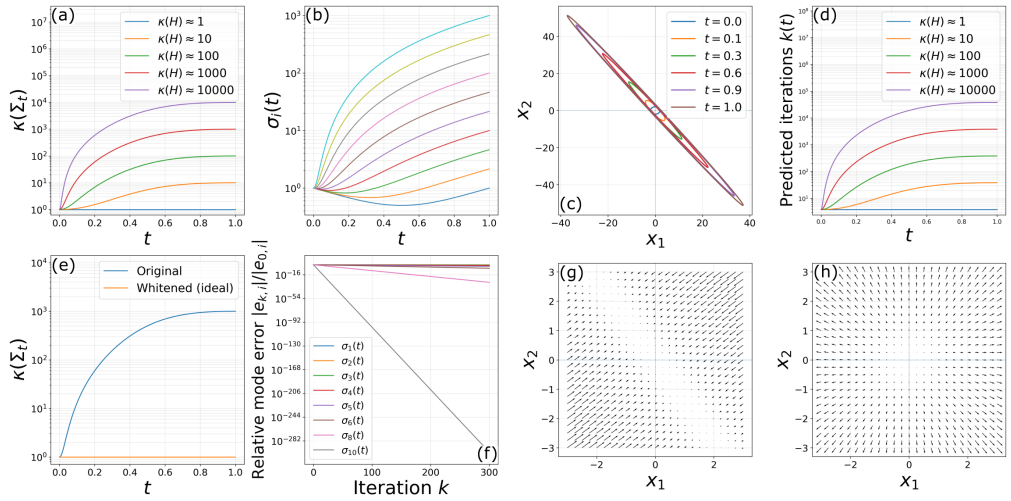


Figure 4: **Gaussian model illustrating conditioning and optimization effects in flow matching: (a)** Growth of the condition number $\kappa(\Sigma_t)$ along the linear interpolation $\Sigma_t = (1-t)^2 I + t^2 H$ showing how intermediate distributions p_t gradually inherit the ill-conditioning of H; **(b)** evolution of the eigenvalues $\sigma_i(t)$ of Σ_t for $\kappa(H) = 1000$, highlighting the increasing spectral disparity that creates slow and fast directions; **(c)** Covariance ellipses of $p_t = \mathcal{N}(0, \Sigma_t)$ at selected times, visualizing the transition from an isotropic distribution to a highly anisotropic one; **(d)** Predicted number of gradient-descent iterations $k(t)$ required to reach a fixed error tolerance, demonstrating the sharp slowdown at large t due to ill-conditioning; **(e)** Effect of ideal preconditioning (whitening), which removes conditioning growth and keeps $\kappa(\Sigma_t) \approx 1$ for all t; **(f)** Per-mode gradient-descent error decay at $t = 0.9$, showing that large-variance modes converge rapidly while small-variance modes decay extremely slowly; **(g)** Optimal score field $s^*(x,t) = -\Sigma_t^{-1} x$, exhibiting strong anisotropy and stiffness along low-variance directions; **(h)** Optimal flow matching velocity field $v^\star(x,t) = A^\star(t)x$, which is smoother than the score but still inherits anisotropy from Σ_t.

Then

$$E_{m+1} = A_{m+1} - A^\star \tag{102}$$

$$= E_m - 2\eta E_m x_t^{(m)} (x_t^{(m)})^\top - 2\eta (A^\star x_t^{(m)} - y^{(m)})(x_t^{(m)})^\top. \tag{103}$$

Taking conditional expectation given E_m gives

$$\mathbb{E}[E_{m+1} \mid E_m] = E_m - 2\eta E_m \mathbb{E}[x_t x_t^\top] - 2\eta \mathbb{E}\big[(A^\star x_t - y) x_t^\top\big]. \tag{104}$$

Using

$$\mathbb{E}[x_t x_t^\top] = \Sigma_t \tag{105}$$

and the population normal equation

$$A^\star \Sigma_t = \mathrm{Cov}(y, x_t) = \mathbb{E}[y x_t^\top], \tag{106}$$

we obtain

$$\mathbb{E}\big[(A^\star x_t - y) x_t^\top\big] = A^\star \Sigma_t - \mathbb{E}[y x_t^\top] = 0. \tag{107}$$

Therefore,

$$\mathbb{E}[E_{m+1} \mid E_m] = E_m (I - 2\eta \Sigma_t). \tag{108}$$

Thus the conditional mean SGD error follows the same contraction matrix as full-batch gradient descent.

Now diagonalize

$$\Sigma_t = U \,\mathrm{diag}(\sigma_1(t), \ldots, \sigma_d(t)) U^\top. \tag{109}$$

Define the error in the eigenbasis of Σ_t by

$$\widetilde{E}_m := E_m U. \tag{110}$$

Multiplying the mean recursion on the right by U gives

$$\mathbb{E}[\widetilde{E}_{m+1} \mid E_m] = \widetilde{E}_m \,\mathrm{diag}\big(1 - 2\eta\sigma_1(t), \ldots, 1 - 2\eta\sigma_d(t)\big). \tag{111}$$

Hence, for any output coordinate k and eigendirection i, the scalar component

$$e_{ki,m} := (\widetilde{E}_m)_{ki} \tag{112}$$

satisfies

$$\mathbb{E}[e_{ki,m+1} \mid E_m] = (1 - 2\eta\sigma_i(t))e_{ki,m}. \tag{113}$$

Equivalently, the stochastic recursion can be written as

$$e_{ki,m+1} = (1 - 2\eta\sigma_i(t))e_{ki,m} + \zeta_{ki,m}, \tag{114}$$

where $\zeta_{ki,m}$ is a zero-mean stochastic-gradient noise term satisfying

$$\mathbb{E}[\zeta_{ki,m} \mid E_m] = 0. \tag{115}$$

Therefore, directions with smaller $\sigma_i(t)$ contract more slowly in the mean SGD dynamics, matching the full-batch GD behavior. In particular, when Σ_t is ill-conditioned, the mean dynamics of SGD also make slow progress along low-variance eigendirections. This proves Proposition 3.5.

The preceding Gaussian proofs isolate the same spectral mechanism from several angles: the covariance Σ_t becomes increasingly anisotropic with t, the GD contraction factors become mode-dependent, and low-variance eigendirections converge slowly. Figure 4 visualizes these consequences, together with the corresponding score and flow fields, and the effect of ideal whitening.

E Proofs for Gaussian-Mixture Flow Matching

E.1 Proof of Proposition 3.8

Assume

$$x_1 \sim \sum_{k=1}^{K} \pi_k \mathcal{N}(0, H_k), \qquad x_0 \sim \mathcal{N}(0, I), \qquad x_t = (1-t)x_0 + tx_1. \tag{116}$$

Let $Z \in \{1, \dots, K\}$ denote the latent mixture component of x_1, with

$$\mathbb{P}(Z = k) = \pi_k. \tag{117}$$

Then by the law of total expectation,

$$g_t^\star(x_t) = \mathbb{E}[x_1 - x_0 \mid x_t] \tag{118}$$

$$= \sum_{k=1}^{K} \mathbb{P}(Z = k \mid x_t, t)\, \mathbb{E}[x_1 - x_0 \mid x_t, Z = k, t]. \tag{119}$$

Define

$$w_k(x_t, t) := \mathbb{P}(Z = k \mid x_t, t). \tag{120}$$

Conditioned on $Z = k$, the pair (x_0, x_1) is Gaussian with

$$x_0 \sim \mathcal{N}(0, I), \qquad x_1 \sim \mathcal{N}(0, H_k), \qquad x_t = (1-t)x_0 + tx_1. \tag{121}$$

Therefore, by the single-Gaussian result already proved in Appendix D.2,

$$\mathbb{E}[x_1 - x_0 \mid x_t, Z = k, t] = A_k^\star(t)x_t, \tag{122}$$

where

$$A_k^\star(t) = \big(tH_k - (1-t)I\big)\big((1-t)^2 I + t^2 H_k\big)^{-1}. \tag{123}$$

Substituting into the law-of-total-expectation formula yields

$$g_t^\star(x_t) = \sum_{k=1}^{K} w_k(x_t, t)\, A_k^\star(t)\, x_t, \tag{124}$$

which proves Proposition 3.8.

E.2 Proof of Theorem 3.9

Under Assumption 3.7, the mixture responsibilities are known exactly, so the Gaussian-mixture FM regression problem decomposes into independent componentwise Gaussian regression subproblems. For component k, the intermediate covariance is

$$\Sigma_{t,k} = (1-t)^2 I + t^2 H_k. \tag{125}$$

If $\lambda_{k,i}$ is the ith eigenvalue of H_k, then the corresponding eigenvalues of $\Sigma_{t,k}$ are

$$\sigma_{k,i}(t) = (1-t)^2 + t^2 \lambda_{k,i}. \tag{126}$$

For each component k, the same argument as in the Gaussian case applies. In particular, full-batch GD applied to the componentwise quadratic loss has error recursion

$$E_{k,m+1} = E_{k,m}(I - 2\eta\Sigma_{t,k}), \tag{127}$$

so along eigendirection i,

$$e_{k,i,m+1} = (1 - 2\eta\sigma_{k,i}(t))e_{k,i,m}. \tag{128}$$

Thus directions with smaller $\sigma_{k,i}(t)$ contract more slowly.

Similarly, for single-sample SGD, the conditional mean error obeys

$$\mathbb{E}[E_{k,m+1} \mid E_{k,m}] = E_{k,m}(I - 2\eta\Sigma_{t,k}), \tag{129}$$

and therefore each scalar mode satisfies

$$\mathbb{E}[e_{k,i,m+1} \mid E_{k,m}] = (1 - 2\eta\sigma_{k,i}(t))e_{k,i,m}. \tag{130}$$

Thus the mean SGD dynamics inherit the same componentwise mode-dependent contraction as full-batch GD.

Consequently, if one seeks comparable convergence across all mixture components and eigendirections, the overall rate is limited by the slowest componentwise mode. This mode is determined by

$$\min_{k,i} \sigma_{k,i}(t). \tag{131}$$

Therefore, even if most mixture components are well-conditioned, a single component with a small eigenvalue can dominate the global optimization time scale. This proves Theorem 3.9.

E.3 Proof of Corollary 3.10

The corollary follows immediately from Theorem 3.9. Since the optimization rate is controlled by the smallest eigenvalue across all components, multimodality does not average away the bottleneck; instead, the hardest component determines the overall difficulty.

E.4 Extension to nonzero-mean Gaussian and Gaussian-mixture FM

Corollary E.1 (Extension to nonzero-mean Gaussian and Gaussian-mixture FM case)**.** *Let $x_0 \sim \mathcal{N}(0, I)$ and $x_1 \sim \mathcal{N}(\mu, H)$ be independent, with $H \succ 0$, and let $x_t = (1-t)x_0 + tx_1$. Define*

$$m_t = t\mu, \qquad \Sigma_t = (1-t)^2 I + t^2 H, \qquad C_t = tH - (1-t)I.$$

Then the population conditional FM target is affine:

$$g_t^\star(x) = \mathbb{E}[x_1 - x_0 \mid x_t = x] = A^\star(t)x + b^\star(t),$$

where

$$A^\star(t) = C_t\Sigma_t^{-1}, \qquad b^\star(t) = \mu - A^\star(t)m_t = (1-t)\Sigma_t^{-1}\mu.$$

For

$$L_t(A, b) = \mathbb{E}\left[\|Ax_t + b - (x_1 - x_0)\|_2^2\right],$$

the excess risk is

$$\begin{aligned} L_t(A, b) - L_t(A^\star, b^\star) = \operatorname{Tr}\big((A - A^\star)\Sigma_t(A - A^\star)^\top\big) \\ + \|(A - A^\star)m_t + (b - b^\star)\|_2^2. \end{aligned}$$

Equivalently, parameterizing the centered predictor as $A(x_t - m_t) + \beta$, with $\beta^\star = \mu$, gives

$$\widetilde{L}_t(A, \beta) - \widetilde{L}_t(A^\star, \mu) = \mathrm{Tr}\big((A - A^\star)\Sigma_t(A - A^\star)^\top\big) + \|\beta - \mu\|_2^2.$$

Thus the slope and intercept decouple. Gradient descent on the slope satisfies

$$E_{\ell+1} = E_\ell(I - 2\eta\Sigma_t), \qquad E_\ell = A_\ell - A^\star(t),$$

while the intercept error contracts by $1 - 2\eta$. Hence, after centering or profiling out the intercept, nonzero means do not alter the covariance-controlled directional bottleneck.

More generally, if

$$x_1 \mid Z = k \sim \mathcal{N}(\mu_k, H_k),$$

then

$$\Sigma_{t,k} = (1 - t)^2 I + t^2 H_k$$

is independent of μ_k, and

$$\mathbb{E}[x_1 - x_0 \mid x_t = x, Z = k] = A_k^\star(t)x + b_k^\star(t),$$

where

$$A_k^\star(t) = \big(tH_k - (1 - t)I\big)\Sigma_{t,k}^{-1}, \qquad b_k^\star(t) = (1 - t)\Sigma_{t,k}^{-1}\mu_k.$$

Therefore, under Assumption 3.7, the componentwise rates and the worst-component result of Theorem 3.9 remain unchanged with nonzero component means.

Remark E.2 (Effect of mean shifts). In the uncentered (A, b) parameterization, the Hessian is governed by

$$M_t = \begin{pmatrix} \Sigma_t + m_t m_t^\top & m_t \\ m_t^\top & 1 \end{pmatrix},$$

so a large mean can worsen raw optimization conditioning. This is a removable parameterization effect: explicit centering eliminates it exactly, whereas the directional stiffness caused by $\kappa(\Sigma_t)$ remains. For mixtures, distinct means affect the posterior responsibilities and mode locations, but not the within-component covariances. Moreover, increasing separation makes the perfect-assignment approximation in Assumption 3.7 more accurate.

Proof. Let $y = x_1 - x_0$. The pair (y, x_t) is jointly Gaussian, with

$$\mathbb{E}[y] = \mu, \qquad \mathbb{E}[x_t] = m_t, \qquad \mathrm{Cov}(x_t) = \Sigma_t, \qquad \mathrm{Cov}(y, x_t) = C_t.$$

Therefore,

$$\mathbb{E}[y \mid x_t = x] = \mu + C_t\Sigma_t^{-1}(x - m_t),$$

which gives $A^\star = C_t\Sigma_t^{-1}$ and $b^\star = \mu - A^\star m_t$. Since

$$\Sigma_t - tC_t = (1 - t)I,$$

we also obtain $b^\star = (1 - t)\Sigma_t^{-1}\mu$.

Let $\Delta A = A - A^\star$, $\Delta b = b - b^\star$, and $z_t = x_t - m_t$. By the conditional-mean decomposition,

$$L_t(A, b) - L_t(A^\star, b^\star) = \mathbb{E}\big[\|\Delta A x_t + \Delta b\|_2^2\big].$$

Using $\mathbb{E}[z_t] = 0$ and $\mathrm{Cov}(z_t) = \Sigma_t$ yields

$$L_t(A, b) - L_t(A^\star, b^\star) = \mathrm{Tr}(\Delta A\Sigma_t\Delta A^\top) + \|\Delta A m_t + \Delta b\|_2^2.$$

The centered form and gradient-descent recursion follow immediately. Conditioning on $Z = k$ gives the same identities with (μ, H, Σ_t) replaced by $(\mu_k, H_k, \Sigma_{t,k})$. Since $\Sigma_{t,k}$ is independent of μ_k, Theorem 3.9 continues to apply. □

F Proofs for Preconditioning in Flow Matching

F.1 Proof of Theorem 4.1

$$x_0 \sim \mathcal{N}(0, I), \qquad x_1 \sim \mathcal{N}(0, H), \qquad x_t = (1 - t)x_0 + tx_1. \tag{132}$$

Let $\mathcal{P} \in \mathbb{R}^{d\times d}$ be invertible and define

$$\tilde{x}_1 = \mathcal{P}x_1, \qquad \tilde{x}_t = (1 - t)x_0 + t\tilde{x}_1. \tag{133}$$

Step 1: transformed covariance. Since x_0 and x_1 are independent and zero mean,

$$\tilde{\Sigma}_t := \mathbb{E}[\tilde{x}_t \tilde{x}_t^\top] \tag{134}$$

$$= \mathbb{E}\Big[\big((1-t)x_0 + t\mathcal{P}x_1\big)\big((1-t)x_0 + t\mathcal{P}x_1\big)^\top\Big] \tag{135}$$

$$= (1-t)^2 \mathbb{E}[x_0 x_0^\top] + t^2 \mathbb{E}[\mathcal{P}x_1 x_1^\top \mathcal{P}^\top] \tag{136}$$

$$+ t(1-t)\mathbb{E}[x_0 x_1^\top]\mathcal{P}^\top + t(1-t)\mathcal{P}\mathbb{E}[x_1 x_0^\top] \tag{137}$$

$$= (1-t)^2 I + t^2 \mathcal{P} H \mathcal{P}^\top. \tag{138}$$

Step 2: exact whitening. If $\mathcal{P} = H^{-1/2}$, then

$$\mathcal{P} H \mathcal{P}^\top = I, \tag{139}$$

and therefore

$$\tilde{\Sigma}_t = (1-t)^2 I + t^2 I = \big((1-t)^2 + t^2\big) I. \tag{140}$$

Hence every eigenvalue of $\tilde{\Sigma}_t$ is equal to $(1-t)^2 + t^2$, so

$$\kappa(\tilde{\Sigma}_t) = 1 \qquad \text{for all } t \in [0,1]. \tag{141}$$

Step 3: approximate isotropization. Assume

$$mI \preceq \mathcal{P} H \mathcal{P}^\top \preceq MI \tag{142}$$

for some $0 < m \leq M$. Then

$$(1-t)^2 I + t^2 mI \preceq \tilde{\Sigma}_t \preceq (1-t)^2 I + t^2 MI. \tag{143}$$

Therefore every eigenvalue of $\tilde{\Sigma}_t$ satisfies

$$(1-t)^2 + t^2 m \leq \lambda_i(\tilde{\Sigma}_t) \leq (1-t)^2 + t^2 M, \tag{144}$$

and so

$$\kappa(\tilde{\Sigma}_t) \leq \frac{(1-t)^2 + t^2 M}{(1-t)^2 + t^2 m}. \tag{145}$$

This proves Theorem 4.1.

F.2 Additional note on Gaussian-mixture preconditioning

The same idea extends componentwise to the Gaussian-mixture setting. If one could apply a componentwise whitening map

$$T_k = H_k^{-1/2} \tag{146}$$

to each mixture component, then the transformed component covariances would satisfy

$$\tilde{\Sigma}_{t,k} = \big((1-t)^2 + t^2\big) I, \tag{147}$$

so all components would become perfectly conditioned and no single mode could dominate optimization. In the main paper, however, Section 4 focuses on the single-Gaussian FM theorem because it is the cleanest exact statement. The practical learned preconditioners in the experiments should be viewed as approximate nonlinear analogues of this isotropization principle.

F.3 FM-specific consequences of the conditioning analysis

We derive several consequences of the Gaussian FM analysis that are specific to the flow matching objective and its time-indexed regression geometry.

Corollary F.3.1 (Small FM loss can hide directional velocity error)**.** *Under the setting of Lemma 3.2, let $H = U\Lambda U^\top$, and let u_i denote the ith column of U. Since Σ_t and H share eigenvectors, with eigenvalues $\sigma_i(t) = (1-t)^2 + t^2\lambda_i$, the fixed-time excess risk decomposes as*

$$\mathcal{L}_t(A) - \mathcal{L}_t(A^\star(t)) = \sum_{i=1}^{d} \sigma_i(t) \big\|(A - A^\star(t))u_i\big\|^2. \tag{148}$$

Consequently, if $\mathcal{L}_t(A) - \mathcal{L}_t(A^\star(t)) \le \varepsilon$, then for every eigendirection u_i,

$$\big\|(A - A^\star(t))u_i\big\|^2 \le \frac{\varepsilon}{\sigma_i(t)}. \tag{149}$$

Thus, in low-variance directions where $\sigma_i(t)$ is small, a small FM excess loss can still permit large velocity-field error.

Proof. By Equation (12), the fixed-time excess risk is

$$\mathcal{L}_t(A) - \mathcal{L}_t(A^\star(t)) = \mathrm{Tr}\Big((A - A^\star(t))\Sigma_t(A - A^\star(t))^\top\Big). \tag{150}$$

Let

$$\Delta_t := A - A^\star(t). \tag{151}$$

Since $H = U\Lambda U^\top$ and $\Sigma_t = (1-t)^2 I + t^2 H$, the matrices H and Σ_t share the same eigenvectors. Hence

$$\Sigma_t = U\,\mathrm{diag}(\sigma_1(t), \dots, \sigma_d(t))U^\top, \qquad \sigma_i(t) = (1-t)^2 + t^2\lambda_i. \tag{152}$$

Substituting this eigendecomposition into the excess-risk expression gives

$$\mathrm{Tr}\big(\Delta_t\Sigma_t\Delta_t^\top\big) = \mathrm{Tr}\Big(\Delta_t U\,\mathrm{diag}(\sigma_1(t), \dots, \sigma_d(t))U^\top\Delta_t^\top\Big). \tag{153}$$

Let u_i denote the ith column of U. Since

$$U\,\mathrm{diag}(\sigma_1(t), \dots, \sigma_d(t))U^\top = \sum_{i=1}^d \sigma_i(t) u_i u_i^\top, \tag{154}$$

we obtain

$$\mathrm{Tr}\big(\Delta_t\Sigma_t\Delta_t^\top\big) = \sum_{i=1}^d \sigma_i(t)\,\mathrm{Tr}\big(\Delta_t u_i u_i^\top\Delta_t^\top\big). \tag{155}$$

For each i,

$$\mathrm{Tr}\big(\Delta_t u_i u_i^\top\Delta_t^\top\big) = \mathrm{Tr}\big((\Delta_t u_i)(\Delta_t u_i)^\top\big) = \|\Delta_t u_i\|^2. \tag{156}$$

Therefore,

$$\mathcal{L}_t(A) - \mathcal{L}_t(A^\star(t)) = \sum_{i=1}^d \sigma_i(t)\big\|(A - A^\star(t))u_i\big\|^2, \tag{157}$$

which proves Equation (148).

Now suppose

$$\mathcal{L}_t(A) - \mathcal{L}_t(A^\star(t)) \le \varepsilon. \tag{158}$$

Since Σ_t is positive definite, each eigenvalue satisfies $\sigma_i(t) > 0$, and every term

$$\sigma_i(t)\big\|(A - A^\star(t))u_i\big\|^2 \tag{159}$$

in the sum is nonnegative. Hence no single term can exceed the whole sum. Therefore, for every i,

$$\sigma_i(t)\big\|(A - A^\star(t))u_i\big\|^2 \le \sum_{j=1}^d \sigma_j(t)\big\|(A - A^\star(t))u_j\big\|^2 = \mathcal{L}_t(A) - \mathcal{L}_t(A^\star(t)) \le \varepsilon. \tag{160}$$

Dividing by $\sigma_i(t) > 0$ gives

$$\big\|(A - A^\star(t))u_i\big\|^2 \le \frac{\varepsilon}{\sigma_i(t)}. \tag{161}$$

This proves Equation (149). □

Corollary F.3.2 (Scalar time reweighting does not remove fixed-time directional stiffness)**.** *Fix $t \in [0,1]$ and let $\alpha(t) > 0$. Consider the scalar-reweighted fixed-time objective*

$$\mathcal{L}_{t,\alpha}(A) = \alpha(t)\,\mathbb{E}\big[\|Ax_t - (x_1 - x_0)\|^2\big]. \tag{162}$$

Then the curvature of this objective with respect to A is scaled by $\alpha(t)$, but its directional condition number is unchanged. In particular, the convergence of full-batch gradient descent remains controlled by $\kappa(\Sigma_t)$ at fixed t.

Proof. Fix $t \in [0,1]$. Since $\alpha(t) > 0$ is a scalar that does not depend on A, the reweighted objective is simply

$$\mathcal{L}_{t,\alpha}(A) = \alpha(t)\mathcal{L}_t(A), \qquad \mathcal{L}_t(A) = \mathbb{E}\big[\|Ax_t - (x_1 - x_0)\|^2\big]. \tag{163}$$

From the Gaussian fixed-time regression problem, the gradient of the unweighted objective is

$$\nabla_A \mathcal{L}_t(A) = 2(A\Sigma_t - C_t), \tag{164}$$

where

$$\Sigma_t = \mathbb{E}[x_t x_t^\top], \qquad C_t = \mathrm{Cov}(x_1 - x_0, x_t). \tag{165}$$

Therefore the gradient of the reweighted objective is

$$\nabla_A \mathcal{L}_{t,\alpha}(A) = \alpha(t)\nabla_A \mathcal{L}_t(A) = 2\alpha(t)(A\Sigma_t - C_t). \tag{166}$$

To see the effect on curvature, consider a perturbation $\Delta \in \mathbb{R}^{d\times d}$ around A. The gradient changes as

$$\nabla_A \mathcal{L}_{t,\alpha}(A + \Delta) - \nabla_A \mathcal{L}_{t,\alpha}(A) = 2\alpha(t)\Delta\Sigma_t. \tag{167}$$

Thus the Hessian is the linear operator

$$\Delta \mapsto 2\alpha(t)\Delta\Sigma_t. \tag{168}$$

Since Σ_t is symmetric positive definite, let

$$\Sigma_t = U\,\mathrm{diag}(\sigma_1(t), \dots, \sigma_d(t))U^\top. \tag{169}$$

In the eigenbasis of Σ_t, the curvature in direction u_i is scaled by

$$2\alpha(t)\sigma_i(t). \tag{170}$$

Hence the largest and smallest directional curvatures are

$$2\alpha(t)\sigma_{\max}(t) \qquad \text{and} \qquad 2\alpha(t)\sigma_{\min}(t), \tag{171}$$

respectively. Their ratio is therefore

$$\frac{2\alpha(t)\sigma_{\max}(t)}{2\alpha(t)\sigma_{\min}(t)} = \frac{\sigma_{\max}(t)}{\sigma_{\min}(t)} = \kappa(\Sigma_t). \tag{172}$$

Thus scalar time reweighting changes the overall scale of the curvature, but not the fixed-time directional condition number.

Equivalently, full-batch gradient descent on $\mathcal{L}_{t,\alpha}$ gives

$$A_{k+1} = A_k - 2\eta\alpha(t)(A_k\Sigma_t - C_t). \tag{173}$$

Let $A^\star(t)$ be the minimizer, so that $A^\star(t)\Sigma_t = C_t$, and define

$$E_k := A_k - A^\star(t). \tag{174}$$

Then

$$E_{k+1} = E_k\big(I - 2\eta\alpha(t)\Sigma_t\big). \tag{175}$$

Projecting onto an eigendirection u_i of Σ_t gives the mode-wise contraction factor

$$1 - 2\eta\alpha(t)\sigma_i(t). \tag{176}$$

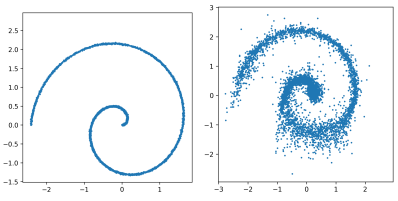
Thus $\alpha(t)$ rescales the effective step size from η to $\eta\alpha(t)$, but the relative separation between fast and slow directions is still governed by the spectrum of Σ_t. With the optimal fixed step size, the iteration complexity therefore remains controlled by $\kappa(\Sigma_t)$ at fixed t. □

G Additional Experiments and Results

G.1 Visualizing preconditioned transport for the Swiss roll data using a normalizing flow and a flow-based preconditioner

To build intuition and isolate algorithmic behavior of the proposed preconditioning strategy, we begin with 2D experiments since low-dimensional settings allow for direct visualization of densities, transport maps, and trajectories.

We study the Swiss roll, a curved, anisotropic, non-Gaussian manifold with severe scale separation, as a testbed for evaluating preconditioning methods. Even with small isotropic noise, its geometry is challenging: elongated along the manifold and tightly concentrated transversely. Preconditioning here serves as geometric isotropization, aiming to reduce anisotropy and simplify the structure to facilitate easier transport toward a Gaussian. As a baseline, we use standard flow matching to map a Gaussian to the Swiss roll, the results for which are presented in Figure 5. We then use both a normalizing flow and a flow matching-based model to precondition the problem.



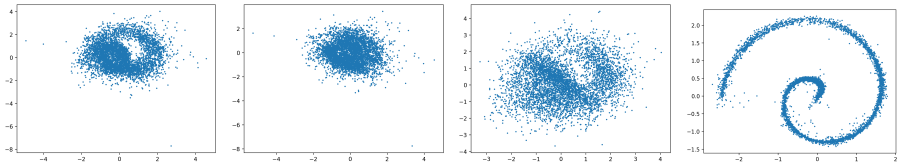
(a) $x_1 \sim p_1$ (b) $\hat{x}_1 = \text{Flow}(z)$

Figure 5: **Flow matching without preconditioning. (a)** Ground-truth Swiss roll samples. **(b)** Generated samples obtained by transporting Gaussian source z through a learned flow.

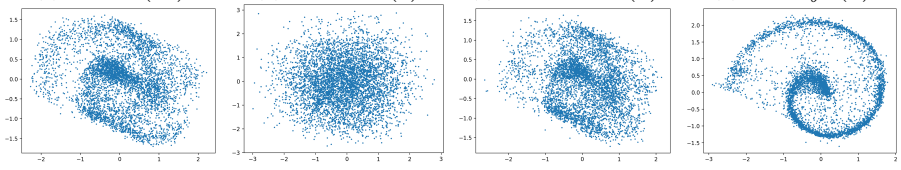
Normalizing flow preconditioner. Figure 6 (top row) shows that normalizing flow (NF) significantly improves generation quality over the standard non-preconditioned flow matching (Figure 5). While the pushforward $\tilde{x} = \mathcal{P}_\theta(x_1)$ in Figure 6 (a) isn't perfectly Gaussian either, it is far more isotropic (and closer to Gaussian), allowing the second-stage flow (b) to operate on a simpler geometry. In inference, Gaussian samples are pulled back via the learned flow (c) and the inverse NF (d), recovering the original distribution more faithfully, as compared to Figure 5 (b).

Table 3: Sliced-Wasserstein distances for the Swiss roll dataset. The top row compares generated samples with ground truth; the bottom row measures how Gaussian the pushed data is. Lower is better.

Direction	No preconditioner	NF preconditioner	FM preconditioner
$z \to x_1$	0.11	0.06	0.07
$x_1 \to z$	0.81	0.31	0.34



(a) $\tilde{x} = \mathcal{P}_\theta(x_1)$ (b) $\hat{\tilde{x}} = \text{Flow}(z)$ (c) $\tilde{z} = \text{Flow}^{-1}(\tilde{x})$ (d) $\hat{x} = \mathcal{P}_\theta^{-1}(\hat{\tilde{x}})$



(e) $\tilde{x} = \text{Flow}_{\text{lc}}(x_1)$ (f) $\hat{\tilde{x}} = \text{Flow}(z)$ (g) $\tilde{z} = \text{Flow}^{-1}(\tilde{x})$ (h) $\hat{x} = \text{Flow}_{\text{lc}}^{-1}(\hat{\tilde{x}})$

Figure 6: **Flow matching with preconditioning. Top row: Normalizing flow preconditioning. (a)** The data x_1 (Swiss roll) is pushed forward by the normalizing flow preconditioner to $\tilde{x}$. **(b)** Random Gaussian samples are mapped by the learned downstream flow to the preconditioned data space. **(c)** For visualization, data in the preconditioned space can also be mapped backward through the learned flow toward Gaussian samples. **(d)** The points in the preconditioned space are mapped back to the original space using $\mathcal{P}_\theta^{-1}$. **Bottom row: Low-capacity flow preconditioning. (e)** The data x_1 is pushed forward by a low-capacity flow preconditioner to $\tilde{x}$. **(f)** Random Gaussian samples are mapped by the learned downstream flow to the preconditioned data space. **(g)** For visualization, data in the preconditioned space can also be mapped backward through the learned flow toward Gaussian samples. **(h)** Generated samples in the preconditioned space are mapped back using the inverse low-capacity flow.

Flow matching preconditioner. In our second experiment, we use a low-capacity flow Flow_{lc} with just 168 parameters as a preconditioner. As shown in Figure 6 (bottom row), even such a lightweight model improves the performance of a downstream high-capacity flow, similar to the normalizing flow case in Figure 6 (top row).

To evaluate the effect of preconditioning quantitatively, we compute the Sliced-Wasserstein distance between point clouds in both directions: $x_1 \to z$ (data to Gaussian) and $z \to x_1$ (Gaussian to data). Results in Table 3 show that both preconditioners improve upon the baseline, with normalizing flow performing the best. Its distinct transformation is likely to complement flow matching by reducing the effective null space of the standard flow.

G.2 Controlled Gaussian-Mixture experiments: conditioning and low-eigenvalue recovery

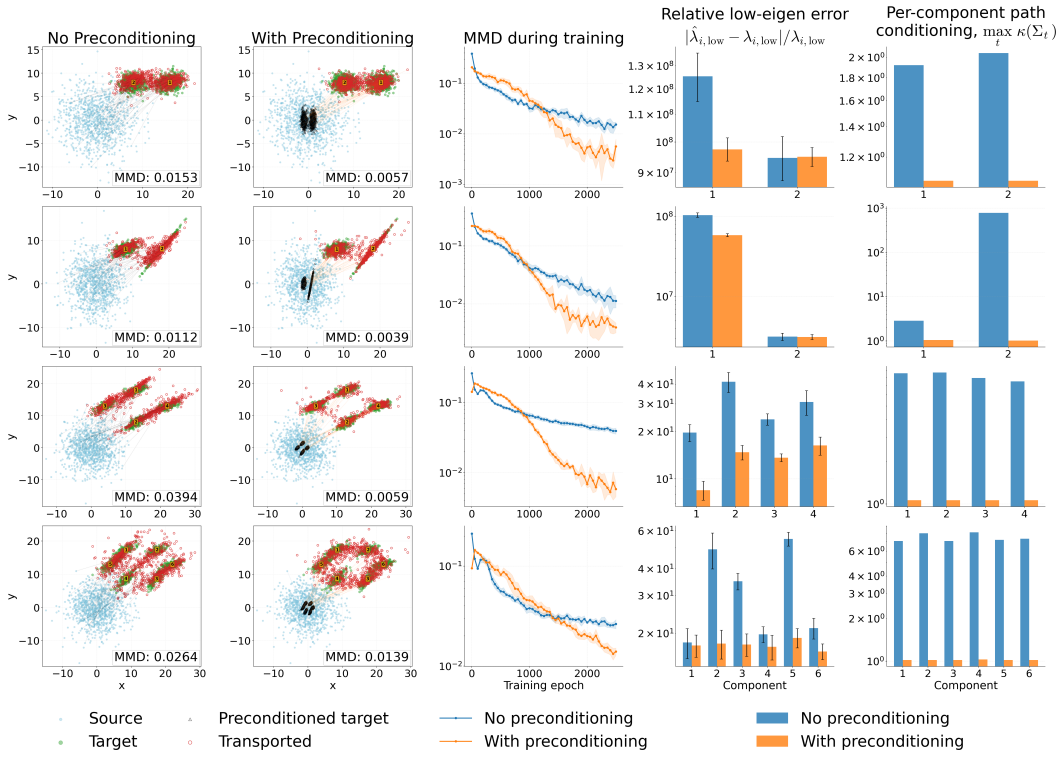


Figure 7: **Preconditioning improves Gaussian-mixture flow matching across controlled 2D configurations.** We compare standard flow matching and preconditioned flow matching on GMM targets with varying numbers of components, anisotropy levels, and transport distances. Each row shows one GMM configuration, while columns show: transported samples without preconditioning, transported samples with preconditioning, MMD during training, relative low-eigenvalue recovery error, and maximum per-component path condition number. Across all configurations, preconditioning produces better-aligned transported samples, lower final MMD, smaller low-eigenvalue error, and substantially reduced path conditioning. These diagnostics support the theoretical prediction that preconditioning improves flow matching optimization by reducing anisotropy along the interpolation path, especially for the hardest mixture components. Plots show the mean over 10 independent runs with different random seeds; error bars / shaded regions denote one standard deviation.

Across all example GMM configurations shown in Figure 7, preconditioning improves the learned transport relative to the unpreconditioned baseline. In these cases, the unpreconditioned model often captures the high-variance direction of a component but fails to accurately recover its low-variance direction, producing visibly distorted samples and larger low-eigenvalue error.

The MMD curves show the same behavior from an optimization perspective (see column 3 in Figure 7). The unpreconditioned baseline often decreases the discrepancy early in training but then plateaus at a higher value. This matches the theoretical prediction that gradient-based training can make rapid progress along high-variance directions while making much slower progress along suppressed low-variance directions. Preconditioning reduces this imbalance, allowing training to continue improving the distributional fit.

The path-conditioning diagnostic provides the most direct link to the theory (see columns 4–5 in Figure 7). In the unpreconditioned case, the interpolation path inherits the anisotropy of the

hardest mixture components, leading to large values of $\max_t \kappa(\Sigma_{t,k})$. After preconditioning, the corresponding path condition numbers are reduced across components. This explains why the preconditioned model better recovers low-eigenvalue directions and achieves lower final MMD. These results support the claim that preconditioning is not merely changing the visual shape of the generated samples, but is directly improving the geometry of the regression problems encountered during flow matching training.

G.3 Optimal flow matching reduces stochastic-target error; preconditioning improves excess-risk optimization

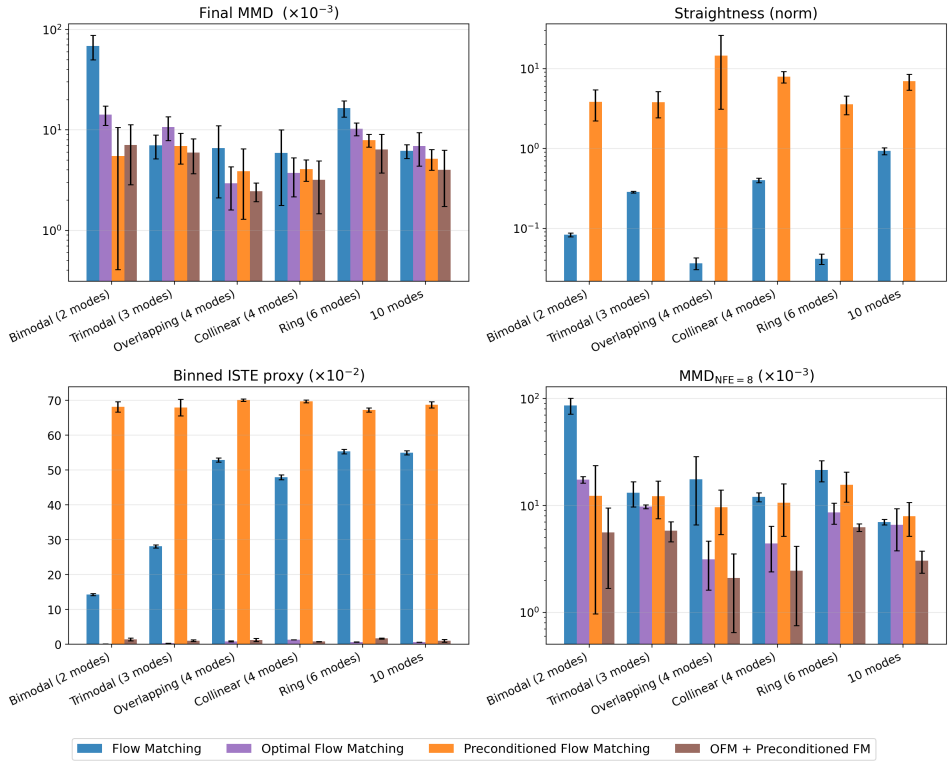


Figure 8: **OFM and preconditioning across six Gaussian-mixture configurations.** Final MMD, trajectory straightness, normalized binned ISTE proxy, and MMD at the reported NFE $= 8$ setting. Endpoint MMD is evaluated after mapping preconditioned samples back to the original target space. Error bars show variability across 10 runs. The corresponding source and target distributions, together with samples transported by each method, are shown in Figure 9.

The FM risk decomposition in Equation (4) separates irreducible stochastic-target error (ISTE), caused by ambiguity in the target velocity at an intermediate point, from excess risk, which measures error in the learned conditional-mean velocity. Optimal Flow Matching (OFM) [Kornilov et al., 2024] addresses the first term through a deterministic Monge-form coupling. Preconditioning instead aims to reduce the second by improving the conditioning of the intermediate distributions. Because transforming the target can also increase ISTE, we examine this trade-off on six Gaussian-mixture configurations by comparing standard FM, Optimal Flow Matching (OFM) [Kornilov et al., 2024], preconditioned FM, and OFM applied in the preconditioned space. Generated samples from the latter two methods are mapped back to the original space for MMD evaluation. We estimate a normalized, binned ISTE proxy by averaging the conditional covariance of the target displacement within bins of intermediate points and dividing by the overall displacement covariance. This proxy can remain nonzero under a deterministic Monge coupling because each bin contains distinct intermediate points.

Figure 8 show that preconditioning alone increases the binned ISTE proxy and the measured deviation from straight trajectories in all six configurations, even though it reduces final MMD relative to standard FM. Thus, the improvement in distributional fit does not, per say, require a reduction in stochastic-target ambiguity. OFM and its preconditioned variant retain a low binned ISTE proxy. The

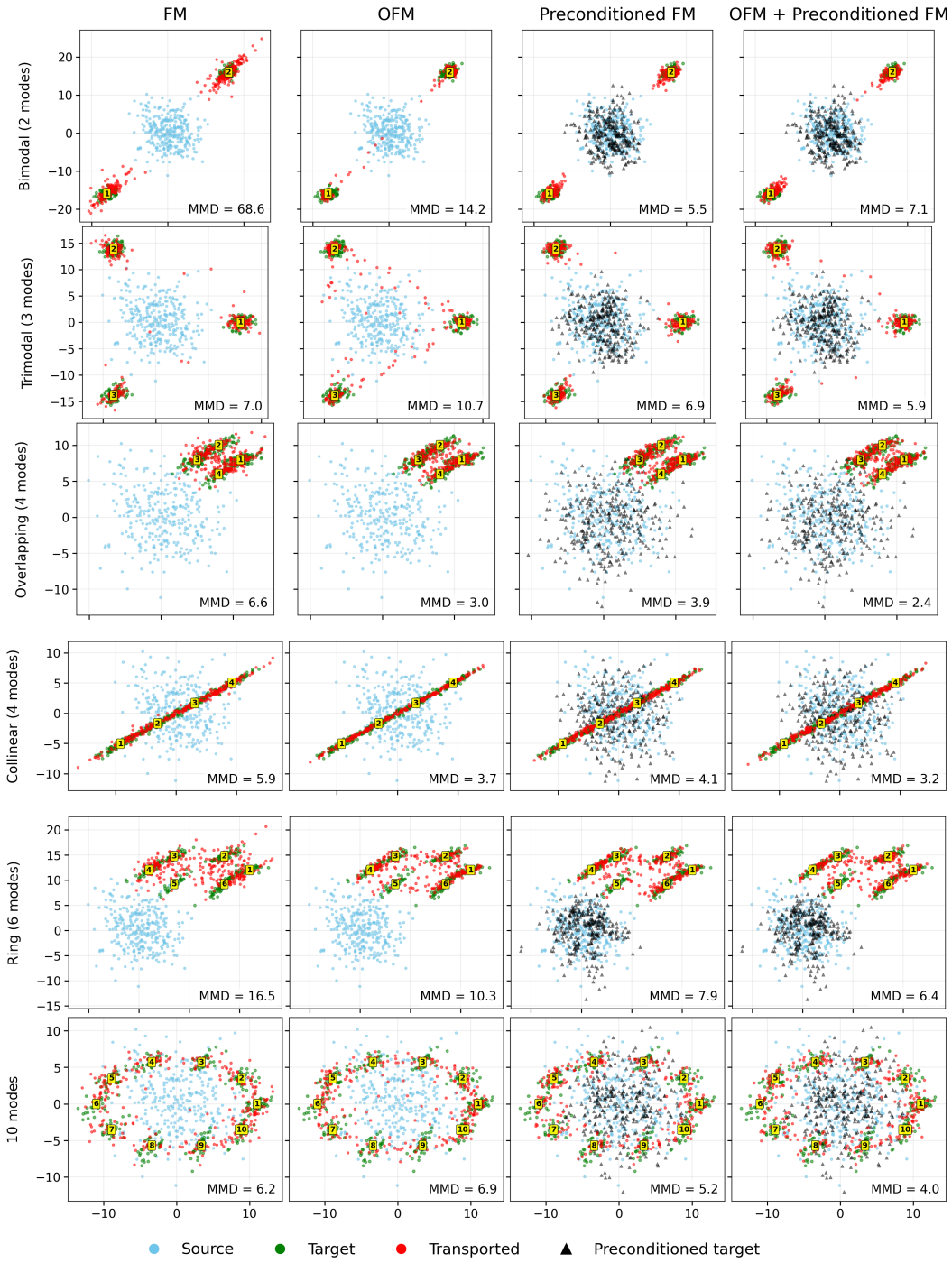


Figure 9: **Generated samples for FM, OFM, preconditioned FM, and their combination.** Rows show Gaussian-mixture configurations; columns show methods. Blue points denote source samples, green points target samples, red points transported samples, and black triangles the transformed target for preconditioned methods. The displayed MMD values are in units of 10^{-3}.

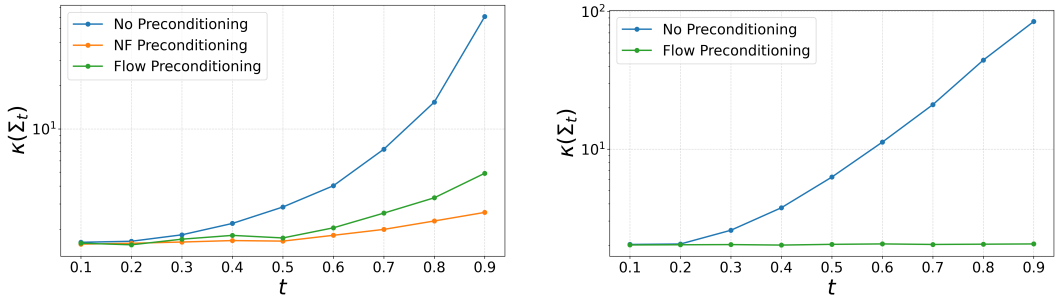
combined method has the lowest *mean* final MMD in five of six configurations and the lowest MMD at the reported NFE $= 8$ setting in all six. These results in Figures 8 and 9 suggest that target-space conditioning and Monge-form coupling can provide complementary benefits. The binned proxy does not, however, establish whether individual trajectories cross, and straightness in preconditioned coordinates need not persist after applying a nonlinear inverse preconditioner.

G.4 Condition number dynamics for MNIST and ImageNet-1k training

To understand how preconditioning impacts the geometry of the transport process, we measure the condition number $\kappa(\Sigma_t)$ of the covariance matrix of samples x_t from the intermediate distributions p_t at various time steps $t \in (0, 1)$ during the flow matching training for the MNIST and ImageNet-1k datasets. A higher condition number indicates stronger anisotropy, which typically hinders optimization during flow matching.

Condition number estimation. During training, for each time step t, we draw 8192 samples from the intermediate distribution by forming $x_t = s(t)x_1 + c(t)x_0$, where $x_0 \sim \mathcal{N}(0, I)$ and x_1 is drawn uniformly from the training set (normalized latents for the baseline; preconditioner-transformed latents $\tilde{x}_1 = \mathcal{P}(x_1)$ for the preconditioned variants). We then compute the empirical covariance $\hat{\Sigma}_t = \frac{1}{N-1}\sum_{i=1}^{N}(x_t^{(i)} - \bar{x}_t)(x_t^{(i)} - \bar{x}_t)^\top$ and report the condition number $\kappa(\hat{\Sigma}_t) = \lambda_{\max}(\hat{\Sigma}_t)/\lambda_{\min}(\hat{\Sigma}_t)$, regularized by adding ϵI with $\epsilon = 10^{-6} \cdot (\mathrm{tr}(\hat{\Sigma}_t)/D)$ to avoid numerical degeneracies. All covariances are computed on flattened latent vectors.

Figure 10 shows that the intermediate FM distributions become increasingly ill-conditioned as $t \to 1$ for both MNIST and ImageNet-1k latent representations. This indicates that the downstream regression problem inherits anisotropy from the target latent distribution. Both normalizing flow (NF) and flow matching (FM) preconditioners substantially reduce the condition numbers, especially near the data endpoint. On MNIST, the NF preconditioner gives the most isotropic path, consistent with its strong quantitative performance in Table 1. The ImageNet-1k results show that this conditioning improvement also holds in a higher-dimensional and more complex latent space, supporting the broader role of preconditioning as a geometric optimization tool for flow matching.



(a) MNIST latent space

(b) ImageNet-1k latent space

Figure 10: **Conditioning of intermediate flow matching distributions with and without preconditioning.** We plot the condition number $\kappa(\Sigma_t)$ of the intermediate covariance matrices Σ_t over time t for **(a)** MNIST and **(b)** ImageNet-1k latent representations. Without preconditioning, the intermediate distributions become increasingly ill-conditioned as $t \to 1$, indicating that the downstream flow matching regression problem inherits strong anisotropy from the target representation. Both normalizing flow (NF) and flow matching (FM) preconditioning substantially reduce $\kappa(\Sigma_t)$, especially near the data endpoint, supporting the claim that preconditioning improves the geometry of the FM path rather than merely adding model capacity.

G.5 Mechanistic analysis of directional learning on latent MNIST

Appendix G.4 showed that preconditioning reduces the condition number of the empirical flow-matching path on both MNIST and ImageNet-1k. We next investigate whether this improvement in path geometry is accompanied by the directional optimization behavior predicted by the theory. In particular, we examine whether the learned nonlinear velocity field fits high-variance directions more rapidly than low-variance directions, and whether preconditioning reduces this learning imbalance.

We perform this analysis post hoc using checkpoints saved throughout the training of the latent-MNIST models.

Empirical latent geometry. Let $z_1 \in \mathbb{R}^{64}$ denote the latent representation produced by the pretrained MNIST VAE. For the unpreconditioned model, the flow-matching target is z_1. For the NF-preconditioned model, the corresponding target representation is $\widetilde{z}_1 = P_\phi(z_1, y)$, where P_ϕ is the class-conditional normalizing-flow preconditioner and y is the MNIST class label. All quantities below are computed separately in the representation space used to train each model.

For each representation, we estimate the target mean and covariance from the MNIST training set:

$$\widehat{\mu} = \frac{1}{M}\sum_{m=1}^{M} z_1^{(m)}, \qquad \widehat{H} = \frac{1}{M-1}\sum_{m=1}^{M}\left(z_1^{(m)} - \widehat{\mu}\right)\left(z_1^{(m)} - \widehat{\mu}\right)^\top. \tag{177}$$

Here, $z_1^{(m)}$ denotes either the original or preconditioned latent, depending on the model under consideration. We eigendecompose $\widehat{H} = U\Lambda U^\top$, and order the eigenvectors u_i by increasing eigenvalue. The eight eigenvectors associated with the largest eigenvalues define the high-variance subspace, while those associated with the smallest eigenvalues define the low-variance subspace.

The MNIST models use the trigonometric interpolation, $x_t = s(t)z_1 + c(t)z_0$, with $s(t) = \sin\left(\frac{\pi t}{2}\right)$ and $c(t) = \cos\left(\frac{\pi t}{2}\right)$, where $z_0 \sim \mathcal{N}(0, I)$. The corresponding conditional flow-matching target is

$$v_t^\star = \dot{s}(t)z_1 + \dot{c}(t)z_0 = \frac{\pi}{2}\left(c(t)z_1 - s(t)z_0\right). \tag{178}$$

Under the moment-matched Gaussian approximation $z_1 \sim \mathcal{N}(\widehat{\mu}, \widehat{H})$, the intermediate covariance is

$$\widehat{\Sigma}_t = c(t)^2 I + s(t)^2 \widehat{H}. \tag{179}$$

Because $\widehat{\Sigma}_t$ is a polynomial in $\widehat{H}$, the two matrices share the same eigenvectors. The associated Gaussian-optimal linear velocity operator is

$$\widehat{A}_t^\star = \frac{\pi}{2} s(t)c(t)\left(\widehat{H} - I\right)\left(c(t)^2 I + s(t)^2\widehat{H}\right)^{-1}. \tag{180}$$

Jacobian-based directional error. The trained MNIST velocity field is nonlinear, so its local action at an intermediate sample x_t is characterized by its input Jacobian

$$J_{\theta_e}(x_t, y, t) = \frac{\partial v_{\theta_e}(x_t, y, t)}{\partial x_t}, \tag{181}$$

where e denotes the training epoch. We evaluate the action of this Jacobian along each covariance eigendirection using exact Jacobian-vector products.

For eigendirection u_i, we define the covariance-unweighted directional operator error as

$$e_i(t, e) = \frac{1}{N_{\text{probe}}}\sum_{n=1}^{N_{\text{probe}}}\left\| J_{\theta_e}\left(x_t^{(n)}, y^{(n)}, t\right)u_i - \widehat{A}_t^\star u_i\right\|_2^2. \tag{182}$$

The same fixed set of probe samples is used for every checkpoint, so changes in this quantity reflect changes in the learned velocity field rather than sampling variation. We evaluate Equation equation 182 at $t \in \{0.50, 0.75, 0.90\}$.

Let $\mathcal{I}_{\text{top-8}}$ and $\mathcal{I}_{\text{bottom-8}}$ denote the indices of the eight highest- and eight lowest-variance eigendirections, respectively. We compute

$$\text{err}_{\text{top-8}}(t, e) = \frac{1}{8}\sum_{i \in \mathcal{I}_{\text{top-8}}} e_i(t, e), \tag{183}$$

$$\text{err}_{\text{bottom-8}}(t, e) = \frac{1}{8}\sum_{i \in \mathcal{I}_{\text{bottom-8}}} e_i(t, e). \tag{184}$$

Unlike the FM excess risk, these errors are not weighted by the corresponding covariance eigenvalues. Consequently, an error is not down-weighted merely because the data variance along that direction is small.

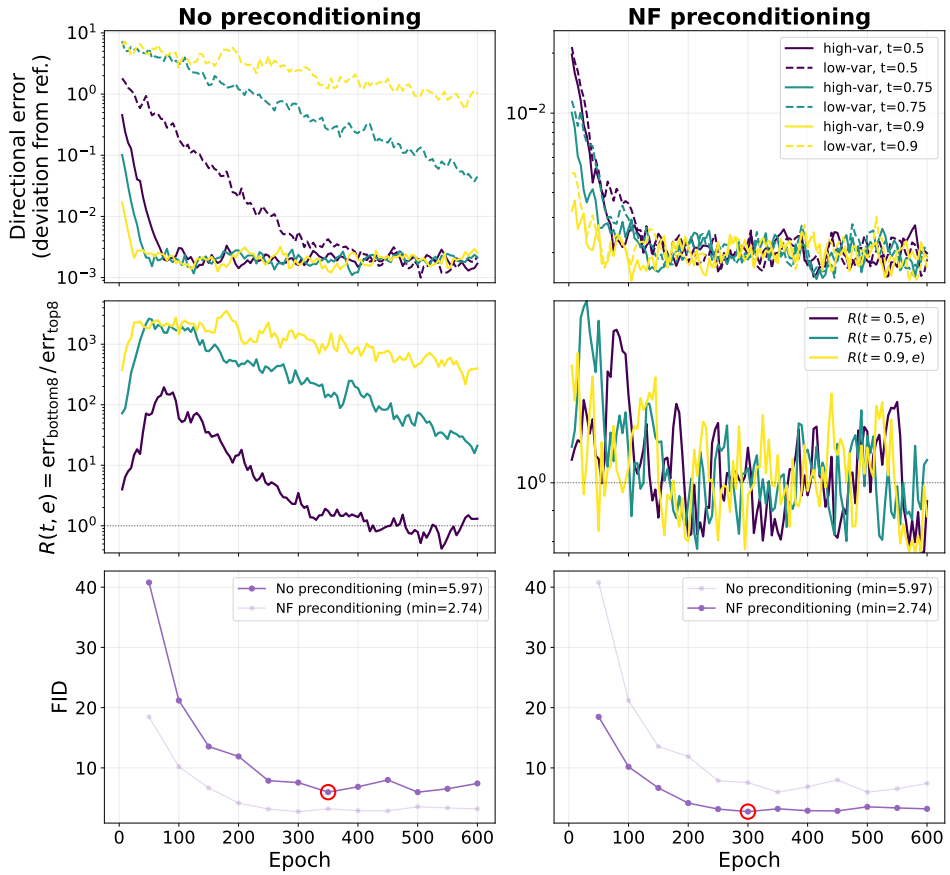


Figure 11: **Directional learning dynamics and generation quality on latent MNIST.** The left and right columns show standard flow matching and NF-preconditioned flow matching, respectively. Both models use the sine–cosine interpolation $x_t = \sin(\pi t/2)x_1 + \cos(\pi t/2)x_0$, where $x_0 \sim \mathcal{N}(0, I)$ and x_1 denotes the target in the latent representation used by each model. **Top:** Covariance-unweighted directional operator errors along the eight highest-variance (solid) and eight lowest-variance (dashed) eigendirections at $t \in \{0.50, 0.75, 0.90\}$. **Middle:** The directional imbalance ratio $R(t, e) = \mathrm{err}_{\text{bottom-8}}(t, e)/\,\mathrm{err}_{\text{top-8}}(t, e)$; the dotted line denotes $R(t, e) = 1$. **Bottom:** FID measured throughout the corresponding MNIST training trajectories. The directional diagnostics are computed within the latent representation used by each model, whereas FID is evaluated in the common decoded image space after applying the inverse preconditioner where necessary.

We summarize the directional learning imbalance using

$$R(t, e) = \frac{\mathrm{err}_{\text{bottom-8}}(t, e)}{\mathrm{err}_{\text{top-8}}(t, e)}. \tag{185}$$

A value $R(t, e) \approx 1$ indicates that the high- and low-variance subspaces are learned with comparable accuracy, whereas $R(t, e) \gg 1$ indicates preferential learning of the high-variance directions.

Preconditioning reduces directional learning imbalance. Figure 11 reveals a consistent difference between the directional learning dynamics with and without preconditioning. Without preconditioning, the errors along the high-variance eigendirections decrease rapidly and reach a low plateau, whereas the errors along the low-variance eigendirections decay much more slowly. This separation is present across the tested values of t and becomes progressively more pronounced as t approaches the data endpoint, where the intermediate distribution inherits more of the anisotropy of the target latent distribution. Consequently, the directional imbalance ratio $R(t, e)$ can remain substantially larger than one, especially at larger values of t, indicating that good recovery of the high-variance subspace does not necessarily imply equally accurate recovery of the low-variance subspace.

With NF preconditioning, the high- and low-variance directional errors rapidly approach comparable scales for all tested values of t. Accordingly, the imbalance ratios remain close to one throughout

most of training and exhibit no persistent growth as the interpolation approaches the data endpoint. Importantly, this improved balance is not obtained by degrading the high-variance directions: their errors remain low, while the errors along the low-variance directions are reduced substantially. These results are consistent with the proposed mechanism that preconditioning improves the geometry of the FM path and enables directions receiving comparatively weak gradient signal in the original representation to be learned at rates comparable to the dominant high-variance directions.

Relation to generation quality. The bottom row of Figure 11 reports FID along the same training trajectories. NF preconditioning achieves a best FID of 2.74, compared with 5.97 without preconditioning. NF preconditioning also achieves lower FID throughout training.

We do not directly compare raw FM losses between the two models because they regress velocity targets defined in different representation spaces. Instead, the directional error measures recovery relative to the appropriate moment-matched Gaussian reference within each representation, while the imbalance ratio measures the relative discrepancy between low- and high-variance directions within the same run. FID is computed after mapping generated samples back to the original latent space and decoding them, making it directly comparable across the two models.

Together with the path-conditioning measurements in Appendix G.4, these results provide an empirical connection between the linear theory and nonlinear flow matching on real data. Preconditioning reduces the anisotropy of the empirical FM path, makes learning more balanced across covariance eigendirections, and is accompanied by improved generation quality. This experiment is intended as a mechanistic diagnostic rather than a global convergence result for arbitrary nonlinear velocity fields.

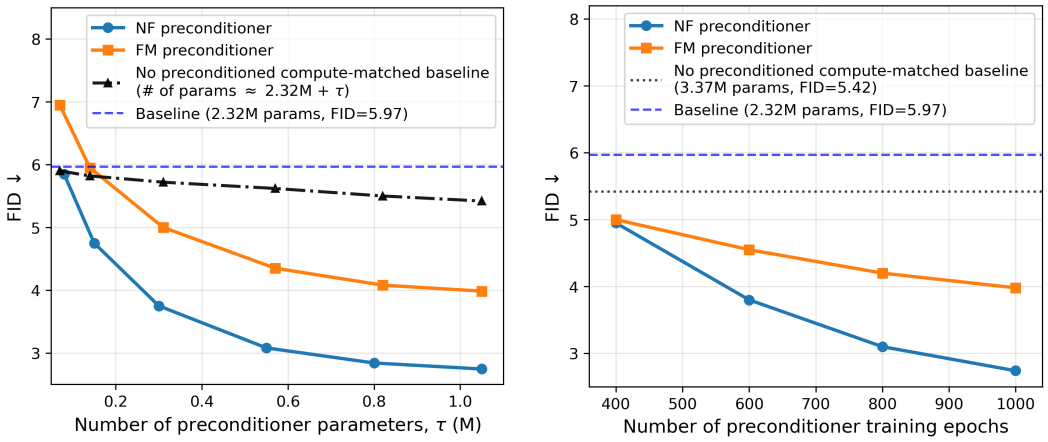
G.6 Preconditioner quality ablation: capacity and training iterations

We ablate the quality of the learned preconditioner on MNIST by varying two factors: (i) the capacity of the preconditioner, measured by its number of trainable parameters, and (ii) the amount of preconditioner training, measured by the number of training epochs. All evaluations are performed using 50k generated MNIST samples. The goal of this ablation is to test whether the gains from preconditioning require a highly accurate or highly overparameterized preconditioner, or whether a moderately trained, moderate-capacity preconditioner is already sufficient to improve downstream flow matching generation.

Figure 12 shows that the effectiveness of preconditioning depends on both the capacity and the training quality of the preconditioner. In Figure 12 (a), increasing the number of preconditioner parameters steadily improves FID for both NF and FM preconditioners. For each preconditioner-capacity point, the preconditioner was trained to convergence using periodically evaluated FID10k as the stopping/selection criterion. We used the checkpoint with the best converged FID10k for the downstream FM model, so the ablation reflects the effect of preconditioner capacity rather than differences in training duration or undertraining. This suggests that a more expressive preconditioner can better reshape the data distribution into a geometry that is easier for the downstream flow matching model to learn. However, the improvement is not simply due to adding parameters: the compute-matched unpreconditioned baseline, which receives a comparable increase in parameter count, improves only marginally relative to the original baseline. Thus, the benefit comes primarily from the geometric effect of preconditioning rather than from model capacity alone.

Figure 12 (b) shows a complementary trend when the preconditioner capacity is fixed (1.05M parameters) and the number of preconditioner training epochs is varied. Poorly trained preconditioners provide only moderate improvement, while better-trained preconditioners lead to substantially lower FID. The gains eventually begin to plateau, indicating that the downstream flow matching model does not require a perfectly converged preconditioner to benefit from the transformed geometry. This behavior is desirable in practice: preconditioning can improve generation even when the preconditioner is trained only to moderate accuracy.

Across both ablations, NF preconditioning achieves the lowest FID, while FM preconditioning still consistently improves over the unpreconditioned and compute-matched baselines. These results support the view that preconditioning is not merely an architectural enlargement of the generator. Instead, the preconditioner changes the geometry of the learning problem in a way that improves downstream flow matching optimization and sample quality.



(a) Effect of preconditioner capacity.

(b) Effect of preconditioner training iterations.

Figure 12: **Preconditioner quality ablation on MNIST.** We evaluate image generation quality as a function of preconditioner capacity and training progress using FID computed on 50k generated samples. **(a)** Increasing the number of preconditioner parameters improves generation quality for both normalizing flow (NF) and flow matching (FM) preconditioners. The blue dashed line denotes the original unpreconditioned FM baseline with 2.32M parameters, and the black dash-dotted curve shows a compute-matched unpreconditioned baseline whose parameter count is increased by approximately the same number of parameters as the preconditioner. Preconditioning improves FID more substantially than simply allocating the same additional parameter budget to the unpreconditioned generator. **(b)** Increasing the number of preconditioner training epochs also improves FID, with the gains eventually beginning to saturate. The dotted horizontal line shows the compute-matched unpreconditioned baseline with 3.37M parameters, and the dashed horizontal line shows the original 2.32M-parameter baseline. Across both ablations, NF preconditioning yields the strongest gains, while FM preconditioning also consistently improves over the unpreconditioned baselines.

G.7 Comparison with Rectified flow and Reflow

Preconditioning and Reflow [Liu et al., 2022] modify different aspects of the transport problem. Preconditioning reshapes the target distribution before training the main flow, whereas Reflow generates model-induced source – target pairs and retrains a Rectified Flow (RF) to obtain straighter trajectories [Liu et al., 2022]. We compare both approaches on MNIST using one Reflow round and 60 function evaluations (NFE) for the main flow.

Table 4: Comparison with Rectified flow and Reflow on MNIST. NFE counts ODE vector-field evaluations. †The NF inverse is a direct network evaluation and is therefore not included in the ODE NFE.

Method	Additional training	Additional inference operation	Main-flow NFE	Inverse-prec. NFE	Total NFE	FID ↓
FM	–	–	60	0	60	5.97
NF preconditioning (Ours)	Train NF preconditioner	One direct NF inverse pass	60	N/A	60†	**2.74**
FM preconditioning (Ours)	Train FM preconditioner	Integrate inverse preconditioner	60	60	120	3.98
RF	–	–	60	0	60	6.86
RF + Reflow	Pair generation and RF retraining	–	60	0	60	5.15

Reflow improves the RF baseline from an FID of 6.86 to 5.15. Nevertheless, both preconditioned variants achieve lower FID. NF preconditioning obtains the best quality–inference trade-off, reaching an FID of 2.74 with 60 ODE evaluations and one direct inverse pass. FM preconditioning reaches an FID of 3.98, but requires an additional 60 evaluations to integrate the inverse preconditioning flow.

Reflow avoids this additional inference integration by placing its extra cost in pair generation and retraining. Thus, the comparison demonstrates better sample quality for target-space preconditioning, although it does not by itself establish lower total training cost because the additional training stages were not normalized by wall-clock time or training FLOPs.

G.8 Operational preconditioner selection and limited-benefit regimes

We propose a practical preconditioner selection protocol. A practical selection procedure is to first estimate $\max_t \kappa(\Sigma_t)$ along the baseline FM path, including class- or component-specific estimates when global covariance may conceal local anisotropy. An affine or ZCA transformation is appropriate when the anisotropy is predominantly global and approximately elliptical, whereas a nonlinear invertible map is preferable when substantial local or componentwise anisotropy remains. In either case, we recommend using the weakest or lowest-capacity preconditioner that materially reduces path conditioning while keeping the inverse sensitivity $\|J_{\mathcal{P}^{-1}}\|_2$ controlled.

Table 5: Relationship between path-conditioning reduction and generation quality across all image datasets. Conditioning reduction is the ratio of the baseline maximum path condition number to its value after preconditioning.

Dataset	Training space	Baseline $\max_t \kappa(\Sigma_t) \downarrow$	Precond. $\max_t \kappa(\widetilde{\Sigma}_t) \downarrow$	Cond. reduction ↑	FID: no precond. ↓	FID: FM precond. ↓	Relative FID gain ↑	Diagnostic
MNIST	Latent	52.83	2.02	26.2×	5.97	3.98	33.3%	Preconditioning beneficial
LSUN Churches	Latent	102.63	5.54	18.53×	19.53	14.47	25.9%	Preconditioning beneficial
Oxford Flowers-102	Latent	83.47	47.92	1.74×	23.82	22.64	5.0%	Limited benefit
AFHQ Cats	Latent	64.75	2.36	27.44×	7.11	4.86	31.6%	Preconditioning beneficial
CIFAR-10	Pixel	755.92	453.68	1.67×	4.15	4.02	3.1%	Limited benefit
ImageNet-1k	Latent	84.37	1.90	44.4×	8.66	4.98	42.5%	Preconditioning beneficial

Table 5 indicates that the achieved reduction in path conditioning is more predictive of the FID benefit than the baseline condition number alone. Across our datasets, reductions of approximately $18.5\times$–$44.4\times$ correspond to relative FID improvements of 25.9%–42.5%, whereas reductions below $2\times$ yield improvements of at most 5%. These ranges are empirical diagnostics rather than universal thresholds. Preconditioning is therefore expected to provide limited benefit when the baseline path is already well-conditioned, when a pilot preconditioner produces only a small conditioning reduction, or when further reduction requires an excessively sensitive inverse map.

In either case, we recommend using the weakest or lowest-capacity preconditioner that materially reduces path conditioning while keeping the inverse sensitivity $\|J_{\mathcal{P}^{-1}}\|_2$ controlled; in Appendix G.9, we evaluate this trade-off directly.

G.9 Inverse-preconditioner Jacobian sensitivity

Let z denote an endpoint in the preconditioned space. For a sufficiently small endpoint error δz, the corresponding reconstruction error satisfies

$$\mathcal{P}^{-1}(z+\delta z) - \mathcal{P}^{-1}(z) = J_{\mathcal{P}^{-1}}(z)\,\delta z + \mathcal{O}\big(\|\delta z\|_2^2\big)\,. \tag{186}$$

Thus, the local worst-case absolute amplification is governed by $\|J_{\mathcal{P}^{-1}}(z)\|_2 = \sigma_{\max}(J_{\mathcal{P}^{-1}}(z))$. In contrast, $\kappa(J_{\mathcal{P}^{-1}}) = \sigma_{\max}(J_{\mathcal{P}^{-1}})/\sigma_{\min}(J_{\mathcal{P}^{-1}})$ measures the variation in amplification across directions and does not, by itself, determine the absolute amplification of an endpoint error.

Controlled GMM ablation. We first vary the affine preconditioning strength γ on the bimodal and anisotropic-ring GMMs. Here, $\gamma = 0$ denotes no preconditioning, whereas $\gamma = 1$ denotes full whitening. Table 6 reports the resulting path conditioning, inverse-map sensitivity, and generation quality.

Table 6: Trade-off between path conditioning, inverse sensitivity, and generation quality under partial affine preconditioning. Path conditioning and MMD are reported as mean ± standard deviation over three seeds. Since the preconditioner is affine, its inverse-Jacobian quantities are constant over space. Bold values of γ identify the lowest final MMD for each target.

Distribution	γ	Peak path $\max_t \kappa(\Sigma_t) \downarrow$	$\|J_{\mathcal{P}_\gamma^{-1}}\|_2 \downarrow$	$\kappa(J_{\mathcal{P}_\gamma^{-1}}) \downarrow$	Final MMD ($\times 10^{-3}$) $\downarrow$
Bimodal	0.00	849.75 ± 11.14	1.00	1.00	114.69 ± 4.26
	0.25	160.02 ± 2.05	2.18	2.31	202.96 ± 10.04
	0.50	30.14 ± 0.37	4.76	5.34	55.74 ± 17.32
	0.75	5.70 ± 0.06	10.39	12.35	66.88 ± 17.59
	1.00	1.21 ± 0.02	22.66	28.53	324.64 ± 5.02
Anisotropic ring	0.00	7.95 ± 0.17	1.00	1.00	20.46 ± 2.90
	0.25	4.71 ± 0.09	1.63	1.29	14.01 ± 4.78
	0.50	2.80 ± 0.03	2.66	1.67	20.26 ± 5.72
	0.75	1.68 ± 0.02	4.34	2.16	24.48 ± 10.49
	1.00	1.19 ± 0.01	7.08	2.79	53.78 ± 15.70

Full whitening minimizes the path condition number but also produces the largest inverse amplification and worse final MMD. The best performance is instead obtained with partial preconditioning. For the bimodal target, $\gamma = 0.50$ reduces peak path conditioning by $28.2\times$ and MMD by 51.4%, while limiting the inverse gain to 4.76. For the anisotropic ring, $\gamma = 0.25$ reduces MMD by 31.5% with an inverse gain of 1.63. Controlled perturbations of magnitude 10^{-3} relative to the endpoint RMS produce negligible changes in MMD.

Learned nonlinear preconditioner on MNIST. We next evaluate the learned nonlinear MNIST preconditioner at both preconditioned real samples and FM-generated endpoints. At each point, we compute the singular values of the exact inverse Jacobian and measure finite-difference amplification along random directions and along the leading right singular vector of the Jacobian.

Table 7: Local sensitivity of the learned nonlinear MNIST inverse. Amplification is measured as $\|\mathcal{P}^{-1}(z+\delta z) - \mathcal{P}^{-1}(z)\|_2 / \|\delta z\|_2$. The worst direction is the leading right singular vector of the inverse Jacobian; p95 denotes the 95th percentile.

Points evaluated	$\|J_{\mathcal{P}^{-1}}\|_2$ median / p95 / max	$\kappa(J_{\mathcal{P}^{-1}})$ median / p95	Random amplification median / p95	Worst-direction amplification median / p95
Real preconditioned samples	2.41 / 2.99 / 3.33	275 / 388	0.60 / 0.88	2.41 / 2.99
Generated endpoints	2.32 / 3.01 / 4.65	273 / 379	0.66 / 0.95	2.32 / 3.01

Across the evaluated MNIST points, the median and p95 worst-direction gains are approximately 2.3–3.0, with a maximum observed gain of 4.65. Random perturbations are typically attenuated, and the sensitivities at real and generated endpoints are similar. Notably, the large Jacobian condition numbers coexist with moderate spectral norms, illustrating that directional anisotropy does not imply equally large absolute error amplification. These results support selecting a partial preconditioner that balances improved path conditioning against controlled inverse-map sensitivity.

G.10 Preconditioning for conditional image restoration tasks

We investigate whether preconditioning extends to conditional flow-matching tasks in which the model must preserve measurement information. We consider two lightweight inverse problems on MNIST, both defined by

$$b = Ax + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, s^2 I), \tag{187}$$

where the noise level s is sampled uniformly between 0% and 10% of the measurement dynamic range. We evaluate Gaussian deblurring with blur standard deviation $\sigma = 3$ pixels and $4\times$ super-resolution, for which A is a factor-4 downsampling operator.

Conditional flow-matching formulation. The baseline is a conditional velocity field, $v_\theta(x_t, t, c)$ with $c = (A^\top b, s)$, implemented using a multiscale UNet. We used the formulation from Ahamed and Haber [2024] for solving inverse problems using flow matching. The estimated endpoint,

$$\widehat{x}_1 = x_t + (1-t) v_\theta(x_t, t, c) \tag{188}$$

is regularized using a measurement-consistency penalty $\|A\widehat{x}_1 - b\|_2^2$. The preconditioned model follows Algorithm 1 and provides the same condition c to both the preconditioner and the main FM model. The main model is trained with the analogous latent-space consistency penalty $\|\widetilde{z}_1 - \widetilde{z}_1^{\parallel}\|_2^2$, where $\widetilde{z}_1^{\parallel}$ denotes the measurement-consistent latent endpoint.

We compare against the original conditional FM baseline and unpreconditioned controls matched to either the total parameter count or the total training FLOPs of the preconditioned pipeline. Reconstruction quality is measured using per-sample SSIM and PSNR of the posterior mean. We additionally report the posterior standard deviation as a measure of sample diversity and the measurement misfit.

Table 8: Conditional image restoration on MNIST. "Params/FLOPs" reports the parameter count and training FLOPs relative to the baseline. For our method, the two parameter counts correspond to the main FM model and preconditioner. PSNR is computed from the posterior mean, while posterior standard deviation summarizes sample diversity. Higher SSIM and PSNR are better; lower measurement misfit is better.

Deblurring					
Method	**Params/FLOPs**	**SSIM ↑**	**PSNR (posterior mean) ↑**	**Posterior std.**	**Misfit** $(10^{-3})\downarrow$
Baseline FM	$1.07\mathrm{M}/1.0\times$	0.803	22.4	0.135	**5.8**
Parameter-matched	$1.50\mathrm{M}/1.0\times$	0.756	**22.6**	0.156	**5.8**
FLOPs-matched	$1.07\mathrm{M}/1.27\times$	0.817	22.5	0.128	**5.8**
Preconditioned (Ours)	$(1.07+0.44)\mathrm{M}/1.27\times$	**0.834**	21.7	0.128	6.2
$4\times$ **super-resolution**					
Method	**Params/FLOPs**	**SSIM ↑**	**PSNR (posterior mean) ↑**	**Posterior std.**	**Misfit** $(10^{-3})\downarrow$
Baseline FM	$1.07\mathrm{M}/1.0\times$	0.685	**19.8**	0.195	**9.6**
Preconditioned (Ours)	$(1.07+0.44)\mathrm{M}/1.27\times$	**0.708**	19.2	0.203	15.4

Restoration performance. As shown in Table 8, preconditioning achieves the best SSIM on deblurring, outperforming both the parameter-matched and FLOPs-matched unpreconditioned controls. The posterior standard deviation is preserved and the measurement misfit remains within approximately 7% of the baseline. The preconditioned model also reaches comparable early reconstruction quality in approximately 2–3$\times$ fewer epochs.

The super-resolution experiment reproduces the SSIM improvement at the same $1.27\times$ training-FLOPs ratio, while maintaining comparable posterior diversity. The improvement in structural similarity is accompanied by a modest reduction in posterior-mean PSNR and a larger measurement misfit, indicating that measurement consistency does not transfer as cleanly as in the deblurring experiment. Overall, the results demonstrate that preconditioning can improve conditional reconstruction quality without removing the diversity of the learned conditional distribution, while also revealing a task-dependent trade-off between perceptual fidelity and strict data consistency.

Conditioning diagnostics under conditional generation. For the unpreconditioned model, the path condition number increases from $\kappa(\Sigma_t) = 3.9$ at $t = 0.1$ to 7.0×10^3 at $t = 0.9$. Preconditioning reduces these values to 3.6 and 1.7×10^3, respectively. The marginal target condition number is also reduced from 8.0×10^7 to 7.8×10^4 for deblurring and to 6.4×10^4 for super-resolution.

Conditioning the preconditioner itself is important. In an ablation, removing the condition produces a slightly more isotropic marginal distribution (5.7×10^4 versus 7.5×10^4) but substantially poorer reconstructions. Using the digit class as a proxy for the measurement condition, the worst condition-specific condition number is also larger for the unconditional preconditioner than for the conditional one (9.5×10^5 versus 4.3×10^5). Thus, marginal isotropy alone can conceal poor geometry within individual conditional distributions.

We further measure the sensitivity of the inverse preconditioner using

$$G = \frac{\left\|P^{-1}(\widetilde{z}+\delta, c) - P^{-1}(\widetilde{z}, c)\right\|_2}{\|\delta\|_2}. \tag{189}$$

The unconditional preconditioner amplifies perturbations, with $G = 2.3$, whereas the conditional preconditioner is contractive, with $G = 0.37$, and remains stable for perturbations between 1% and 10% of $\|\widetilde{z}\|_2$. These results show that conditioning information does not undermine the benefits of preconditioning; rather, supplying the condition to the preconditioner is important for preserving condition-specific geometry and inverse stability. Consequently, conditional preconditioners should be selected using worst-case condition-specific conditioning and inverse-map sensitivity, rather than marginal isotropy alone.

H Qualitative Comparison for Flow-Based Preconditioning

H.1 MNIST

We provide qualitative comparison for the flow matching in the latent space of an MNIST-trained VAE (latent dimension 64, see Appendix I.1 for details) when the latent space was not preconditioned (left) (FID = 5.97), preconditioned using a normalizing flow (middle) (FID = 2.74), and using a flow matching model (right) (FID = 3.98) in Figure 13. The visual quality of the samples is consistent with the FID scores, with sharper and more coherent digits observed in the preconditioned cases. We also analyzed the condition number dynamics as a function of time t during the training of these flow models, which are presented in Figure 10a.

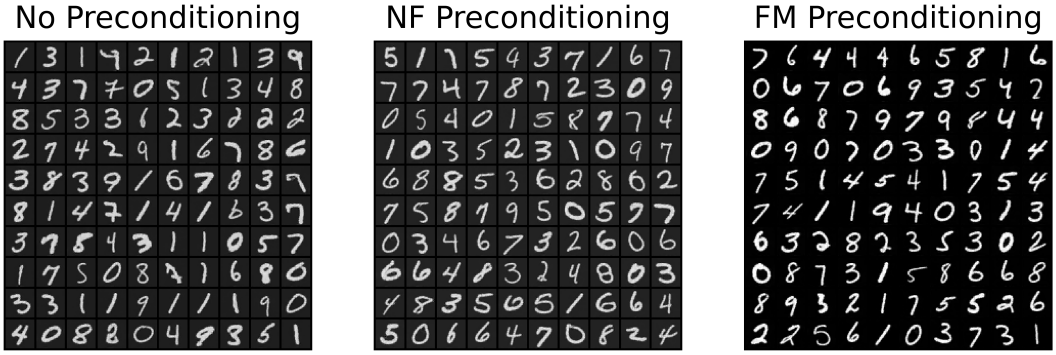


Figure 13: **MNIST qualitative comparison.** Samples generated by latent-space flow matching under different preconditioning strategies. **Left:** The no-preconditioning baseline produces recognizable digits but shows occasional shape inconsistencies and reduced sharpness (FID = 5.97). **Middle:** NF preconditioning improves sample fidelity and yields cleaner digit shapes (FID = 2.74). **Right:** FM preconditioning also improves over the baseline, producing more consistent and diverse digits (FID = 3.98). Overall, the qualitative trends are consistent with the quantitative FID improvements.

H.2 Image samples across resolutions

We provide qualitative comparisons illustrating the effect of flow-based preconditioning on sample quality for LSUN Churches [Yu et al., 2015], Oxford Flowers-102 [Nilsback and Zisserman, 2008], AFHQ Cats [Choi et al., 2020], CIFAR-10 [Krizhevsky et al., 2009] and ImageNet-1k [Chrabaszcz et al., 2017]. As shown in Figures 14 to 19, these examples complement the quantitative improvements reported in Table 2 and illustrate how the additional flow improves sample consistency in cases where standard flow matching often remains visually imperfect.



Figure 14: **LSUN Churches qualitative comparison.** Samples generated by standard flow matching (*top*) and by flow-based preconditioning using an additional flow (*bottom*). Preconditioning produces a sharper global structure and more stable spatial layouts.



Figure 15: **Oxford Flowers-102 qualitative comparison.** Samples generated by standard flow matching (*top*) and by flow-based preconditioning using an additional flow (*bottom*). Preconditioning yields more coherent structure and improved visual consistency.



Figure 16: **AFHQ Cats qualitative comparison.** Samples generated by standard flow matching (*top*) and by flow-based preconditioning using an additional flow (*bottom*).

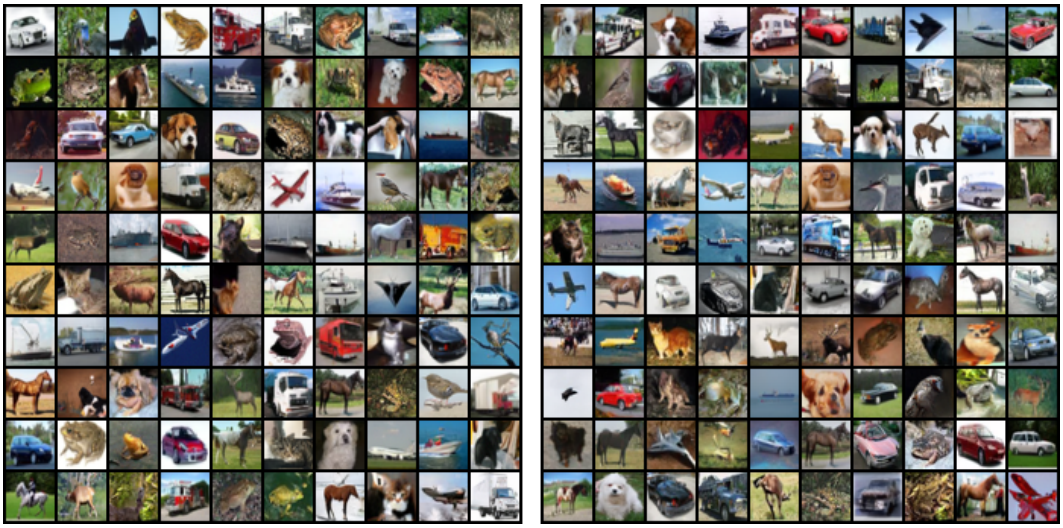


Figure 17: **CIFAR-10 qualitative results.** Samples generated by standard flow matching (*left*) and by flow-based preconditioning using an additional flow (*right*).

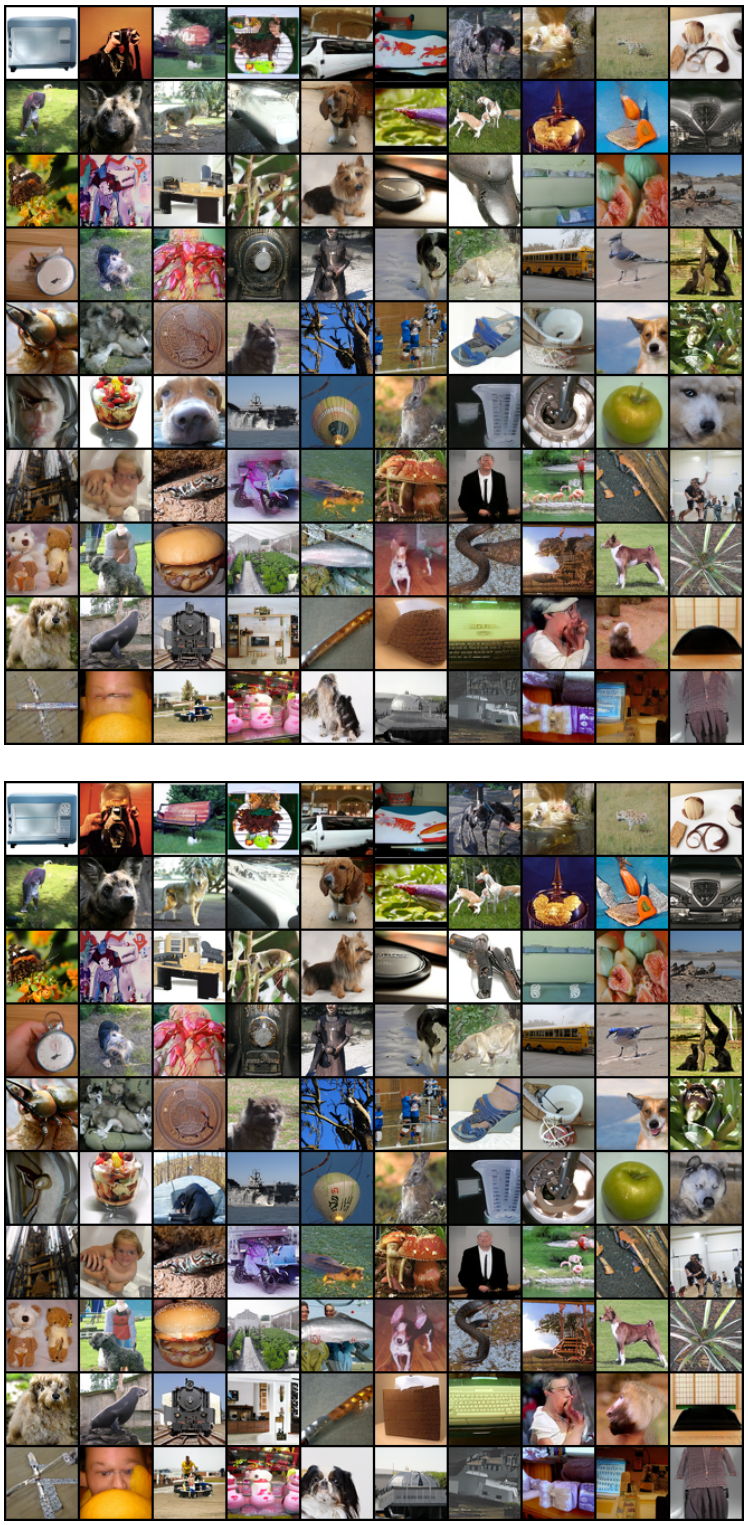


Figure 18: **ImageNet-1k qualitative comparison.** Samples generated by standard flow matching *(top)* and by flow-based preconditioning using an additional flow *(bottom)*. Preconditioning improves visual consistency and sample fidelity while using the same downstream generation setting.

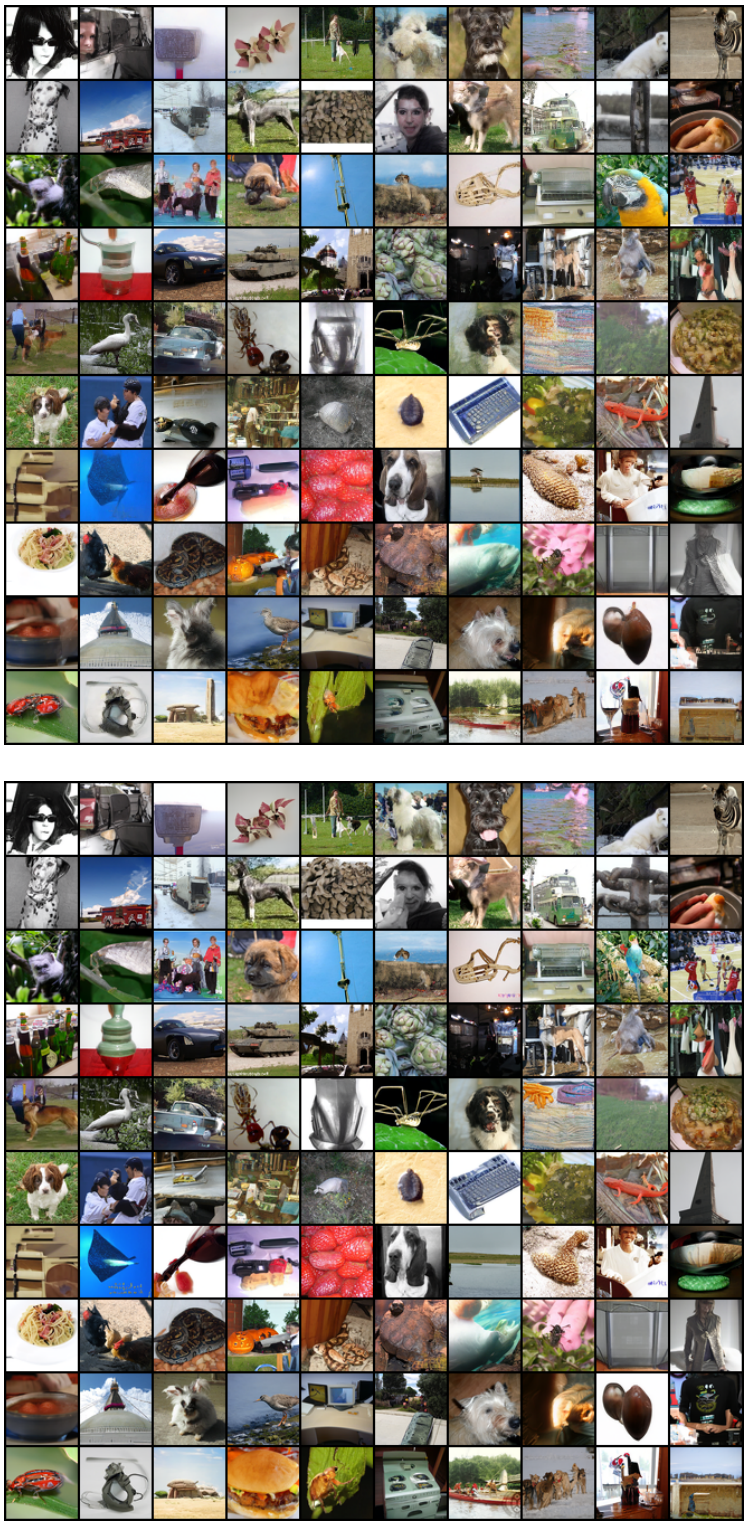


Figure 19: **ImageNet-1k qualitative comparison.** Samples generated by standard flow matching *(top)* and by flow-based preconditioning using an additional flow *(bottom)*. Preconditioning improves visual consistency and sample fidelity while using the same downstream generation setting.

I Experimental Details

I.1 MNIST

We evaluate preconditioned flow matching on MNIST dataset. All MNIST experiments are performed in the latent space of a trained VAE. This VAE consisted of a deep ResNet-based architecture with group normalization, SiLU activations, and was trained using a combination of MSE and perceptual loss. The encoder and decoder consist of multiple downsampling and upsampling ResBlocks, with a latent dimensionality of 64. VAE training used cosine learning rate decay, KL regularization with low $\beta = 0.05$. Generated latent samples are decoded back to image space using the VAE decoder for evaluation.

We compare three variants: standard latent flow matching with no preconditioning, normalizing flow (NF) preconditioning, and flow matching (FM) preconditioning. In all three cases, the main generator is a class-conditional CNN-based latent vector field with 256 hidden channels, 7 convolutional layers, and one-hot class conditioning through a label MLP. The main generator has approximately 2.32M parameters across all variants, so differences in performance are not caused by changing the main FM architecture.

The two preconditioned variants differ in how the auxiliary transformation is learned. For NF preconditioning, we use a conditional RealNVP model with affine coupling layers, hidden MLP width 256, and roughly 1.05M parameters. For FM preconditioning, we first train a smaller class-conditional CNN vector field with 169 hidden channels and roughly 1.05M parameters, then train the main FM generator on the transformed latent distribution. Thus, the preconditioning stage is substantially cheaper than the main generator while still reshaping the geometry of the downstream flow matching problem.

The exact architectures, training spaces, interpolation functions, latent dimensions, learning rates, batch sizes, training epochs, and model sizes are summarized in Table 9.

I.2 Image synthesis across resolutions

We evaluate preconditioned flow matching on five image datasets: LSUN Churches (256×256) [Yu et al., 2015], Oxford Flowers-102 (256×256) [Nilsback and Zisserman, 2008], AFHQ Cats (512×512) [Choi et al., 2020], CIFAR-10 (32×32) [Krizhevsky et al., 2009], and ImageNet-1k (64×64) [Chrabaszcz et al., 2017]. For LSUN Churches, Oxford Flowers-102, AFHQ Cats, and ImageNet-1k, we train in the latent space of a pretrained Stable Diffusion VAE encoder–decoder [Rombach et al., 2022]. This compresses images to $4 \times 32 \times 32$ latents for LSUN Churches and Oxford Flowers-102, $4 \times 64 \times 64$ latents for AFHQ Cats, and $4 \times 8 \times 8$ latents for ImageNet-1k. CIFAR-10 is trained directly in pixel space at its native 32×32 resolution.

The generator and preconditioner architectures are chosen according to the scale and structure of each dataset. For LSUN Churches and Oxford Flowers-102, both the main generator and the preconditioner use UNet backbones in the VAE latent space, with the preconditioner trained for fewer epochs than the main generator. For AFHQ Cats, both models operate in the VAE latent space: we use a UNet preconditioner and a DiT-L/2 backbone for the main flow matching generator. CIFAR-10 is modeled in pixel space using class-conditional UNets, with a smaller preconditioner obtained by reducing the base channel count. For ImageNet-1k, we use a DiT-B/2 generator and a smaller DiT-S/2 preconditioner in the VAE latent space.

For each dataset, we compare a standard flow matching generator against a preconditioned variant. The preconditioned variant first trains an auxiliary flow model, which is then used to transform the data distribution into a better-conditioned representation. The main flow matching generator is subsequently trained on this preconditioned distribution, and generated samples are mapped back through the inverse preconditioning flow at inference time. This design makes the preconditioning stage cheaper than the main generator, either by using a lower-capacity model or by training it to lower accuracy, while still improving the geometry of the downstream flow matching problem.

The exact architectures, training spaces, interpolation functions, latent dimensions, learning rates, batch sizes, training epochs, and model sizes are summarized in Table 10.

Table 9: Experimental settings and hyperparameters for MNIST latent flow matching experiments.

Setting	**No Preconditioning**	**NF Preconditioning**	**FM Preconditioning**
Training space	Latent	Latent	Latent
Latent dim	64	64	64
Interpolation	$s(t) = \sin(\pi t/2)$ $c(t) = \cos(\pi t/2)$	$s(t) = \sin(\pi t/2)$ $c(t) = \cos(\pi t/2)$	$s(t) = \sin(\pi t/2)$ $c(t) = \cos(\pi t/2)$
FM generator model			
Network	CNN	CNN	CNN
Hidden channels	256	256	256
Depth	7 conv layers	7 conv layers	7 conv layers
Class conditioning	one-hot(10)+MLP	one-hot(10)+MLP	one-hot(10)+MLP
Approx. parameters	2.32M	2.32M	2.32M
Max epochs	1000	1000	1000
Learning rate	2×10^{-4}	2×10^{-4}	1×10^{-3}
Optimizer	Adam	Adam	Adam
Batch size	4096	4096	1024
Preconditioner model			
Network	–	RealNVP	CNN
Key architecture	–	hidden MLP width = 256	hidden channels = 169
	–	coupling layers = 8	depth = 7 conv layers
Class conditioning	–	one-hot(10)+MLP	one-hot(10)+MLP
Approx. parameters	–	1.05M	1.05M
Max epochs	–	450	650
Learning rate	–	2×10^{-4}	1×10^{-3}
Optimizer	–	Adam	Adam
Batch size	–	4096	1024

J Computational Resources Used

The experiments in this paper were run on a single compute server with three NVIDIA RTX A6000 GPUs, each providing 48 GB of memory. The host machine further comprised two Intel Xeon Gold 5317 CPUs (24 physical cores / 48 logical CPUs total) and 1 TB of RAM, and ran AlmaLinux 9.6.

All model training, sampling, and evaluation were performed on this system. Where applicable, we used distributed data-parallel training across the three GPUs. This hardware configuration was sufficient for all experiments reported in the paper.

K Existing Assets and Licenses

The experiments use existing public datasets and standard model components. We cite the original sources for all datasets used in the paper, including MNIST, CIFAR-10, ImageNet-1k, LSUN Churches, Oxford Flowers-102, and AFHQ Cats. We use these assets only for research and evaluation purposes and respect their associated terms of use. We also cite the Stable Diffusion VAE (CC-BY 4.0) checkpoint which was used to encode our images to the latent space for higher-resolution images. No new dataset is introduced or redistributed as part of this work. Any released code will include attribution to external codebases or libraries used, together with their corresponding licenses.

Table 10: Experimental settings and hyperparameters for LSUN Churches, Oxford Flowers-102, AFHQ Cats, CIFAR-10, and ImageNet-1k.

Setting	**LSUN Churches**	**Oxford Flowers-102**	**AFHQ Cats**	**CIFAR-10**	**ImageNet-1k**
Image resolution	256×256	256×256	512×512	32×32	64×64
Training space	Latent	Latent	Latent	Pixel	Latent
VAE / latent encoder	Stable Diffusion VAE	Stable Diffusion VAE	Stable Diffusion VAE	–	Stable Diffusion VAE
Latent dim	$4 \times 32 \times 32$	$4 \times 32 \times 32$	$4 \times 64 \times 64$	–	$4 \times 8 \times 8$
Interpolation	$s(t) = \sin(\pi t/2)$ $c(t) = \cos(\pi t/2)$	$s(t) = \sin(\pi t/2)$ $c(t) = \cos(\pi t/2)$	$s(t) = \sin(\pi t/2)$ $c(t) = \cos(\pi t/2)$	$s(t) = t$ $c(t) = (1 - t)$	$s(t) = t$ $c(t) = (1 - t)$
FM generator model					
Network	UNet	UNet	DiT-L/2	UNet	DiT-B/2
Base / hidden channels	256	256	1024	256	768
Depth / residual blocks	3 blocks	3 blocks	24 blocks	3 blocks	12 blocks
Attention heads	4	4	16	4	12
Attention resolutions	16×16	16×16	–	$16\times16, 8\times8$	–
Patch size	–	–	2×2	–	2×2
Class embedding dim	–	–	–	128	768
CFG dropout	–	–	–	0.1	0.1
Approx. parameters	189M	189M	458M	214M	130M
Max epochs	150	300	50	1275	350
Learning rate	5×10^{-5}	5×10^{-5}	1×10^{-4}	2×10^{-4}	1×10^{-4}
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
Batch size	32	32	32 (16×2)	512	1536
Preconditioner model					
Network	UNet	UNet	UNet	UNet	DiT-S/2
Base / hidden channels	256	256	192	128	384
Depth / residual blocks	3 blocks	3 blocks	2 blocks	3 blocks	12 blocks
Attention heads	4	4	12–24	4	6
Attention resolutions	16×16	16×16	$16\times16, 8\times8$	$16\times16, 8\times8$	–
Patch size	–	–	2×2	–	2×2
Class embedding dim	–	–	–	128	384
Approx. parameters	189M	189M	148M	54M	33M
Max epochs	50	100	200	200	200
Learning rate	5×10^{-5}	5×10^{-5}	1×10^{-4}	1×10^{-4}	1×10^{-4}
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
Batch size	32	32	32 (16×2)	128	1536

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main theoretical claims: flow matching can be viewed as a stochastic regression problem, ill-conditioned intermediate distributions can slow optimization along low-variance directions, Gaussian and Gaussian-mixture settings make this mechanism analyzable, and preconditioning reshapes the intermediate path to improve conditioning. These claims are supported by the formal results in the paper and by controlled diagnostics and image-generation experiments. The introduction and discussion also state the scope of the analysis, namely that the theory is exact for Gaussian/Gaussian-mixture settings and is intended as a mechanistic diagnostic rather than a global convergence theory for arbitrary nonlinear flow networks.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations in Appendix B. It states that the theory is exact for Gaussian and Gaussian-mixture settings but does not provide a global convergence theorem for arbitrary nonlinear flow networks or adaptive optimizers. It also discusses practical limitations of preconditioning, including the additional training stage, choices of preconditioner architecture/capacity/training duration, approximate invertibility of flow-based preconditioners, and the need for broader evaluation under larger-scale conditional generation and standardized compute budgets.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.

- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The paper states the assumptions for each theoretical result, including the Gaussian setting, linear interpolation, fixed-time regression objective, population-level analysis, linear predictors, SGD/GD settings, Gaussian-mixture model, perfect component assignment assumption, and invertibility/linear assumptions for the preconditioning theorem. Complete proofs are provided in the appendix for the regression decomposition, Gaussian target characterization, excess-risk decomposition, GD and SGD dynamics, Gaussian-mixture bottleneck results, preconditioning theorem, and FM-specific consequences. The main text also states the scope of the results and clarifies that the theory is intended as a mechanistic population-level analysis rather than a global convergence theorem for arbitrary nonlinear flow networks. These are presented in Appendices D to F.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides the algorithmic description of preconditioned flow matching, specifies the preconditioner types used in the experiments, and reports the evaluation metrics and baselines used to support the main claims. The appendix includes additional implementation details for the controlled Gaussian/GMM experiments, MNIST latent experiments, image-synthesis experiments, condition-number diagnostics, and preconditioner-capacity/training-quality ablations. These details are intended to make the main experimental comparisons reproducible, including the datasets, model classes, evaluation metrics, compute-matched baselines, and preconditioning procedures. The relevant information are provided throughout the Appendix, especially in Appendix I.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.

- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The experiments use simulated or publicly available datasets and the paper provides algorithmic descriptions, evaluation metrics, baselines, and implementation details needed to understand and approximately reproduce the main results. The code is available at `https://github.com/ahxmeds/preconditioned-flow-matching.git`.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: The paper specifies the experimental settings needed to understand the reported results in Appendix I. The main text summarizes the datasets, baselines, metrics, and compute-matched comparisons, while the appendix provides detailed experimental settings, including training spaces, latent dimensions, model architectures, interpolation functions, optimizer choices, learning rates, batch sizes, training epochs, parameter counts, and evaluation procedures for the MNIST, controlled-distribution, higher-resolution image-synthesis, condition-number, and ablation experiments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The controlled Gaussian and Gaussian-mixture experiments, which directly test the conditioning mechanism predicted by the theory, report error bars as one standard deviation on mean over 10 independent runs. These error bars capture variability due to random initialization and sampling in the controlled experimental setup. For larger image experiments, we report single-run metrics because repeated full training runs are computationally expensive; this limitation is stated explicitly, and the imaging results are supported by compute-matched baselines and preconditioner-capacity/training-quality ablations for MNIST rather than statistical significance tests.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The paper reports the main hardware used for all experiments in Appendix J: a single compute server with three NVIDIA RTX A6000 GPUs with 48 GB memory each, two Intel Xeon Gold 5317 CPUs, 1 TB RAM, and AlmaLinux 9.6. It also states that all training, sampling, and evaluation were performed on this system, with distributed data-parallel training used where applicable.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The research conforms to the NeurIPS Code of Ethics. The work is theoretical and methodological, focusing on the optimization geometry of flow matching and preconditioning. The experiments use standard public benchmark datasets and controlled synthetic distributions, and do not involve human-subject studies, private or sensitive data, deceptive data collection, or deployment in high-stakes decision-making settings. Since the method applies to generative modeling, it shares the general possibility of misuse associated with image generation methods, but the paper does not target or enable any specific harmful application.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The paper discusses broader impacts. The work is primarily theoretical and methodological, aiming to improve the optimization and reliability of flow matching models through better-conditioned training geometry. Potential positive impacts include more efficient and stable generative modeling, which may benefit scientific modeling, image synthesis, and representation learning. Potential negative impacts are the general misuse risks associated with improved generative models, such as the creation of misleading synthetic images or other deceptive content. The paper does not target such applications and does not introduce a deployed system, private dataset, or high-stakes decision-making pipeline.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release new datasets, pretrained generative models, or code artifacts that would require safeguards for controlled access or misuse prevention. The experiments use standard public datasets and controlled synthetic distributions, and no scraped dataset, deployed system, or high-risk model checkpoint is released as part of the submission.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites the original sources for the existing datasets and model components used in the experiments, including MNIST, CIFAR-10, ImageNet-1k, LSUN Churches, Oxford Flowers-102, and AFHQ Cats. Appendix K includes an asset and license statement explaining that these assets are used only for research and evaluation, that no new dataset is redistributed, and that any released code will preserve attribution and license information for external codebases or libraries.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce or release new datasets, benchmarks, or other research assets as part of the submission, other than the code at `https://github.com/ahxmeds/preconditioned-flow-matching.git`. The work uses existing public datasets and controlled synthetic experiments for evaluation, and no new asset documentation is therefore required.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: Not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: Not applicable.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.