ADAPTING NOISE TO DATA BY QUANTILE LEARNING

Jannis Chemseddine , Gregor Kornhardt , Richard Duong & Gabriele Steidl
Technische Universität Berlin
Straße des 17. Juni 136, 10623 Berlin, Germany
{chemseddine, kornhardt, duong, steidl}@math.tu-berlin.de

ABSTRACT

The common assumption of a Gaussian latent space in flow-based generative models can be restrictive, especially when modeling heavy-tailed data distributions. To better accommodate complex data geometries, we propose learning data-adaptive latent distributions via one-dimensional quantile functions. These are trained by minimizing the Wasserstein distance between noise and data. Thereby, the latent adapts to both heavy-tailed and compactly supported distributions while shortening transport paths. Numerical results confirm the method's flexibility and effectiveness achieved with negligible computational overhead.

1 INTRODUCTION

Flow-based generative models have become a dominant paradigm in modern generative modeling. In particular, score-based diffusion models Sohl-Dickstein et al. (2015); Song & Ermon (2019); Song et al. (2021), flow matching (FM) methods Albergo et al. (2023); Lipman et al. (2023); Liu (2022), and more recently few-step approaches such as consistency-style models Song et al. (2023); Geng et al. (2025) achieve state-of-the-art performance across a wide range of domains, including image synthesis and molecular generation Hoogeboom et al. (2022), as well as discrete modalities such as text Austin et al. (2023).

In flow-based models, the default choice of noise distribution is Gaussian, which can cause difficulties when learning, for example, multimodal or heavy-tailed target distributions; we refer to the extensive literature Wiese et al. (2019); Hagemann & Neumayer (2021); Salmona et al. (2022); Pandey et al. (2024); Ghane et al. (2025); Tam & Dunson (2025). As a typical example, see the Neal's funnel in Figure 1. To enable sampling from heavy-tailed target distributions, recent state-of-the-art approaches typically replace the standard Gaussian reference with a heavy-tailed noise distribution. However, these methods still require manually tuning the tail behavior to match the data. For instance, Pandey et al. (2024) proposes a Student-t noise distribution with a tunable degrees-of-freedom parameter ν. Similarly, Shariatian et al. (2025a) extends the SDE framework to heavy-tailed settings by driving the dynamics with α-stable noise.

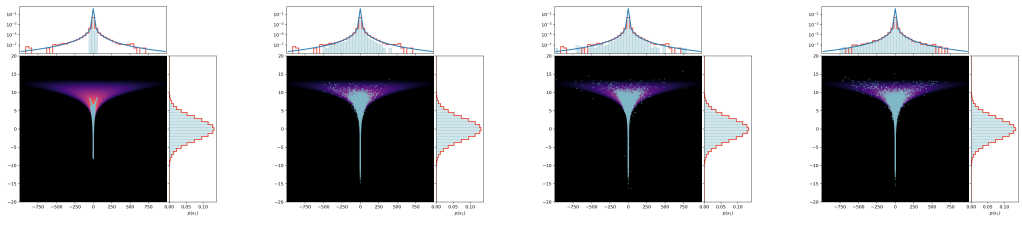


Figure 1: Sampling of Neal's funnel with different latent distributions.[1] Left to Right: *uniform* on $[-1, 1]$, *standard Gaussian, Student-T* (with parameters $(20, 4)$ inspired by the choice in Pandey et al. (2024)) and our *learned distribution*.

In this paper, we propose a novel *lightweight* approach to design a noise distribution tailored to the data by *learning* it directly from samples. In standard FM and diffusion models, isotropic Gaussian

noise is gradually added to the data, i.e.

$$\mathbf{X}_t = \alpha_t \mathbf{X}_0 + \mathbf{N}_t,$$

where $\mathbf{N}_t = (N_t^1, \dots, N_t^d)$ with i.i.d. $N_t^i \sim \mathcal{N}(0, \sigma_t^2)$.

Motivated by this componentwise independent structure of the Gaussian and the Brownian motion, we construct *multidimensional* flows from *one-dimensional* ones. This enables us to address the latter ones by their *quantile functions*. While quantile functions can represent any 1D probability distribution, their monotonicity makes them particularly simple to parameterize, for example using rational quadratic splines Gregory & Delbourgo (1982). More precisely, for quantile functions Q^i, $i = 1, \dots, d$, we consider the *quantile processes*

$$X_t^i = (1-t)\, X_0^i + t\, Q^i(U^i), \qquad t \in [0,1],$$

with i.i.d. $U^1, \dots, U^d \sim \mathcal{U}[0,1]$. We learn individual quantile functions Q_ϕ^i such that their componentwise concatenation $\mathbf{Q}_\phi(\mathbf{U}) := (Q_\phi^1(U^1), ..., Q_\phi^d(U^d))$ is *"close"* to the data by minimizing a (regularized) Wasserstein distance between the data distribution and the noise. Note that this is a constrained optimization problem, since the noise is learned as a product of 1D measures.

The simplicity of quantile functions gives us a flexible tool, which enables us to simultaneously learn the noising process and apply the FM framework.

Our approach allows us to i) avoid the limitations of Gaussian base noise, ii) eliminate manual fine-tuning of the noise distribution, and iii) drastically shorten the resulting transport paths as illustrated in Figure 2.

Contributions. 1. We introduce a *lightweight* framework for adapting the noise distribution to the data distribution, capturing e.g. the tail behavior, while still producing high quality samples.

2. We propose to *learn the 1D components of the noise distributions* themselves within the FM framework in a data adapted way, by parameterizing them through quantile functions and minimizing the Wasserstein distance between the data samples and the noise.

3. Numerical experiments demonstrate that our method efficiently handles diverse marginal structures including heavy-tailed, compact, and multi-modal distributions. Learned quantiles shorten transport paths by capturing per-coordinate structure while *cross-dimensional dependencies* are modeled by the velocity field in FM.

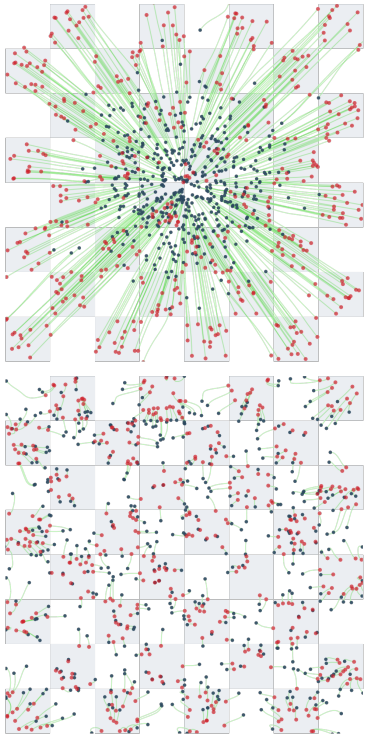


Figure 2: FM via optimal coupling with Gaussian noise (top) and our learned noise (bottom). Latent samples are shown in black, generated in red, and transportation paths in green. Starting from the learned latent drastically shortens the paths.

2 Flow Matching

In general, flow models can be described mathematically as curves in Wasserstein space; for background we refer to Ambrosio et al. (2008) and Lipman et al. (2024); Wald & Steidl (2025). In the following, we denote the target distribution by $\mu_0 \in \mathcal{P}(\mathbb{R}^d)$ and the noise distribution by $\mu_1 \in \mathcal{P}(\mathbb{R}^d)$.

Curves in Wasserstein Spaces. Let $(\mathcal{P}_2(\mathbb{R}^d), W_2)$ denote the complete metric space of probability measures with finite second moments equipped with the Wasserstein distance

$$W_2^2(\mu,\nu) := \min_{\pi \in \Pi(\mu,\nu)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x-y\|^2 \, \mathrm{d}\pi(x,y).$$

Here $\Pi(\mu,\nu)$ denotes the set of all probability measures on $\mathbb{R}^d \times \mathbb{R}^d$ having marginals μ and ν. By $\pi_o \in \Pi(\mu,\nu)$ we denote the minimizer of the right-hand side. The push-forward measure of

$\mu \in \mathcal{P}_2(\mathbb{R}^d)$ by a measurable map $\mathcal{T} : \mathbb{R}^d \to \mathbb{R}^d$ is defined by $\mathcal{T}_\sharp \mu := \mu \circ \mathcal{T}^{-1}$. A narrowly continuous curve $\mu_t : [0,1] \to \mathcal{P}_2(\mathbb{R}^d)$ is absolutely continuous, iff there exists a Borel measurable vector field $v : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ with $\|v_t\|_{L_2(\mathbb{R}^d, \mathbb{R}^d, \mu_t)} \in L_2([0,1])$ such that (μ_t, v_t) satisfies the continuity equation

$$\partial_t \mu_t + \nabla_x \cdot (\mu_t v_t) = 0$$

in the sense of distributions. For a probability space $(\Omega, \Sigma, \mathbb{P})$ and a differentiable stochastic processes $(X_t)_{t\in[0,1]}$ with $X_t \in L_2(\Omega, \mathbb{R}^d, \mathbb{P})$, we have that (μ_t, v_t) with

$$X_t \sim \mu_t \coloneqq X_{t,\sharp}\mathbb{P} \quad \text{and} \quad v_t \coloneqq \mathbb{E}[\dot{X}_t | X_t = \cdot] \tag{1}$$

satisfies the continuity equation.

From the Continuity Equation to the ODE. If the vector field fulfills $\int_0^1 \sup_{x\in B} \|v_t(x)\| + \mathrm{Lip}(v_t, B)\, \mathrm{d}t < \infty$ for all compact $B \subset \mathbb{R}^d$, then the ODE

$$\partial_t \varphi(t,x) = v_t(\varphi(t,x)), \qquad \varphi(0,x) = x, \tag{2}$$

has a solution $\varphi : I \times \mathbb{R}^d \to \mathbb{R}^d$ and $\mu_t = \varphi(t,\cdot)_\sharp \mu_0$.

We can reverse the flow from the latent to the target distribution using just the opposite velocity field $-v_{1-t}$ in the ODE (2). If the velocity field v_t is known, we can sample from the target distribution by starting in a sample from the latent one and then applying any ODE solver in (2).

Flow matching. FM aims to learn the velocity field v_t with a neural network v_t^θ by minimizing the loss

$$\mathcal{L}(\theta) \coloneqq \mathbb{E}_{t,\, x\sim\mu_t}\left[\left\|v_t^\theta(x) - v_t(x)\right\|^2\right]$$

with uniformly sampled time $t \sim \mathcal{U}([0,1])$. Indeed, this untractable loss function can be rewritten as $\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{CFM}}(\theta) + \text{const}$ with the conditional flow matching (CFM) loss

$$\mathcal{L}_{\mathrm{CFM}}(\theta) \coloneqq \mathbb{E}_{t,\, x_0\sim\mu_0,\, x\sim\mu_t(\cdot|x_0)}\left[\left\|v_t^\theta(x) - v_t(x|x_0)\right\|^2\right].$$

For a linear interpolating process

$$X_t = (1-t)X_0 + tX_1$$

and a coupling $\pi \in \Pi(\mu_0, \mu_1)$, where $X_0 \sim \mu_0$ and $X_1 \sim \mu_1$, we have by (1) that $v_t(x|x_0) = x_1 - x_0$, so that $\mathcal{L}_{\mathrm{CFM}}(\theta)$ can be rewritten as

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,\,(x,y)\sim\pi}\left[\left\|v_t^\theta\big((1-t)x + ty\big) - (y-x)\right\|_2^2\right].$$

Taking the optimal coupling π_o instead of the usually applied independent coupling $\pi = \mu_0 \otimes \mu_1$, leads to OT-FM with reduced variance in training and both shorter and straighter paths, see Tong et al. (2024); Pooladian et al. (2023).

3 Adapting Noise to Data

The choice of the latent distribution has a crucial influence on the FM sample quality as shown in Figure 1. Therefore, we propose to *learn* the noising process. First, we have to revisit the connection between 1D distributions and their quantile functions.

3.1 Quantile Functions

The *cumulative distribution function* (CDF) R_μ of $\mu \in \mathcal{P}_2(\mathbb{R})$ and its *quantile function* Q_μ are given by

$$R_\mu(x) \coloneqq \mu\big((-\infty, x]\big), \quad x \in \mathbb{R},$$
$$Q_\mu(u) \coloneqq \min\{x \in \mathbb{R} : R_\mu(x) \geq u\}, \quad u \in (0,1).$$

In Figure 3, we exemplify the CDF and quantile function of a standard Gaussian.

[1] Note that we used the independent coupling for training of these models. We also used z-score normalization.

The quantile functions form a closed, convex cone

$$\mathcal{C} \coloneqq \{f \in L_2(0,1) : f \text{ increasing } a.e.\}$$

in $L_2(0,1)$. The mapping $\mu \mapsto Q_\mu$ is an isometric embedding of $(\mathcal{P}_2(\mathbb{R}), W_2)$ into $(L_2(0,1), \|\cdot\|_{L_2})$, meaning that

$$W_2^2(\mu,\nu) = \int_0^1 \left|Q_\mu(s) - Q_\nu(s)\right|^2 \, \mathrm{d}s \tag{3}$$

and $\mu = Q_{\mu,\sharp}\mathcal{U}_{(0,1)}$. Let $U \sim \mathcal{U}[0,1]$ be uniformly distributed on $[0,1]$. Now, any probability measure flow μ_t can be described by their respective quantile flow $Q_t \coloneqq Q_{\mu_t}$ via $\mu_t = Q_{t,\sharp}\mathcal{U}_{(0,1)}$ and $Q_t \circ U$ is a stochastic process with marginals μ_t.

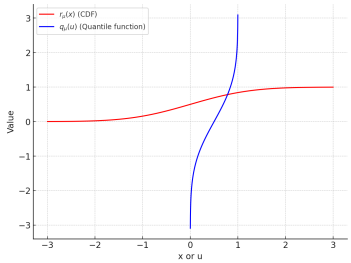


Figure 3: The CDF R_μ and quantile function Q_μ of a standard normal distribution μ.

The quantile function can be modeled using rational quadratic splines Gregory & Delbourgo (1982); Durkan et al. (2019), as described in Section 4.1.

3.2 Learning a Data-Adaptive Noise Distribution

Rather than manually selecting a noise family and tuning its parameters, we propose to *learn* the noise directly from the data. We restrict to noise $\mathbf{Q}$ with independent components having law from the set $S \coloneqq \left\{\nu \in \mathcal{P}_2(\mathbb{R}^d) : \nu(x) = \prod_{i=1}^d \nu^i(x^i)\right\}$. In other words, the ν^i correspond to quantile processes of the form

$$X_t^i = (1-t)\, X_0^i + t\, Q^i(U^i), \quad t \in [0,1],$$

and we set $\mathbf{Q}(\mathbf{u}) \coloneqq (Q^1(u^1), \ldots, Q^d(u^d))$. The quantile functions Q^i determine the scale, support, and tail behavior of each marginal of $\mathbf{Q}(\mathbf{U})$. Note that per the factorization of the noise, the corresponding conditional velocity field decomposes into univariate velocity components.

Learning Objective. We learn the noise $\mathbf{Q}_\phi$ by minimizing the Wasserstein distance between μ_0 and $\nu_\phi \coloneqq (\mathbf{Q}_\phi)_\# \mathcal{U}([0,1]^d)$,

$$\mathcal{L}_{\mathrm{AN}}(\phi) = W_2^2(\mu_0, \nu_\phi). \tag{4}$$

Note that due to the restriction of our quantiles to the class S, the minimizer of (4) is in general *not* μ_0. Crucially, the independence constraint restricts $(\mathbf{Q}_\phi)_\#\ \mathcal{U}([0,1]^d)$ to per-coordinate adaptation and prevents encoding *cross-dimensional* correlations. The latter are introduced via the OT coupling (x, y) and modeled by the velocity field through the target $y - x$. By this separation the latent remains simple and computationally efficient, while delegating cross-dimensional dependencies to the flow. As seen in the next remark, the minimizer is not necessarily the measure with the correct margins.

Remark 3.1 (Counterexample: Marginal Product). For the measure

$$\mu = \tfrac{1}{2}\delta_{(1,1)} + \tfrac{1}{2}\delta_{(-1,-1)} \in \mathcal{P}_2(\mathbb{R}^2), \qquad \mu_{\mathrm{marg}} = \left(\tfrac{1}{2}\delta_{-1} + \tfrac{1}{2}\delta_1\right) \otimes \left(\tfrac{1}{2}\delta_{-1} + \tfrac{1}{2}\delta_1\right),$$

one has $W_2^2(\mu, \mu_{\mathrm{marg}}) = 2$, whereas for

$$\nu_\alpha = \left(\tfrac{1}{2}\delta_{-\alpha} + \tfrac{1}{2}\delta_\alpha\right) \otimes \left(\tfrac{1}{2}\delta_{-\alpha} + \tfrac{1}{2}\delta_\alpha\right)$$

it holds $W_2^2(\mu, \nu_\alpha) = 2\big(1 - \alpha + \alpha^2\big) = 1.5$ for $\alpha = 0.5$. Thus the W_2–closest independent latent may contract or expand the marginals to partially account for correlations it cannot represent.

Although our quantiles can be trained independently, in order to provide an aligned training signal for the velocity field, we propose to train $\mathbf{Q}_\phi$ *jointly* with velocity v^θ. Hence, we aim to minimize the loss

$$\mathcal{L}(\theta,\phi) = \mathcal{L}_{\mathsf{CFM}}(\theta,\phi) + \lambda\, \mathcal{L}_{\mathrm{AN}}(\phi), \quad \lambda > 0,$$

with $\mathcal{L}_{\mathsf{CFM}}(\theta,\phi)$ given by

$$\mathbb{E}_{t,(x,y_\phi)\sim\pi_\phi}\Big[\big\|v_t^\theta\big((1-t)x + t y_\phi\big) - (\mathrm{sg}(y_\phi) - x)\big\|_2^2\Big],$$

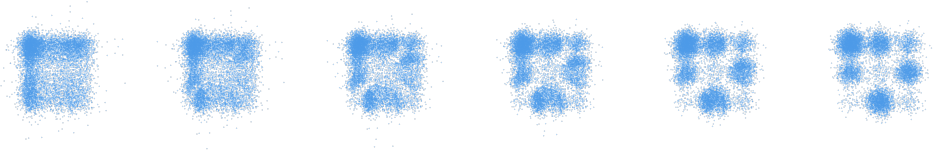


Figure 4: A generated trajectory from the learned quantile latent (left) to the unevenly weighted Gaussian mixture target (right). The *learned latent* is already close to the target distribution.

where $\pi_\phi \in \Pi_o(\mu_0, \nu_\phi)$ and ν_ϕ and $\mathrm{sg}(\cdot)$ denotes the stop-gradient operator.

Implementation Details. In practice, we optimize via minibatches; see Appendix B.2 for details. We compute a minibatch OT map T minimizing $\sum_{j=1}^{B} \|\mathbf{x}_0^{(j)} - \mathbf{y}^{(T(j))}\|_2^2$ for batches from $\mathbf{X}_0$ and $\mathbf{Q}_\phi(\mathbf{U})$. Crucially, this coupling is reused both for optimizing the quantile and for OT-FM. We train the quantile jointly for a fixed number of iterations before freezing its parameters; thereafter only the velocity field is optimized.

Algorithm 1 Joint learning of 1D quantiles and FM velocity

Input: dataset $\mathcal{D}$, batch size B, weight λ, iterations K; quantile model $\mathbf{Q}_\phi$, velocity model v^θ
for $k = 1$ **to** K **do**
  Sample $\{\mathbf{x}_i\}_{i=1}^B \sim \mathcal{D}$, $\{\mathbf{u}_j\}_{j=1}^B \sim \mathcal{U}([0,1]^d)$, $\{t_j\}_{j=1}^B \sim \mathcal{U}(0,1)$
  $C_{ij} \leftarrow \|\mathbf{x}_i - \mathbf{Q}_\phi(\mathbf{u}_j)\|_2^2$
  $T \leftarrow \arg\min_T \sum_{i=1}^B C_{i,T(i)}$
  Define P such that $P(j) = i$ iff $T(i) = j$
  for $j = 1$ **to** B **do**
    $\mathbf{z}_j \leftarrow (1 - t_j)\mathbf{x}_{P(j)} + t_j\, \mathbf{Q}_\phi(\mathbf{u}_j)$
    $\mathbf{v}_{\mathrm{target},j} \leftarrow \mathrm{sg}\big(\mathbf{Q}_\phi(\mathbf{u}_j) - \mathbf{x}_{P(j)}\big)$
  end for
  $\widehat{\mathcal{L}}_{\mathrm{AN}} \leftarrow \frac{1}{B}\sum_{j=1}^B \|\mathbf{x}_{P(j)} - \mathbf{Q}_\phi(\mathbf{u}_j)\|_2^2$
  $\widehat{\mathcal{L}}_{\mathsf{CFM}} \leftarrow \frac{1}{B}\sum_{j=1}^B \|v^\theta(\mathbf{z}_j, t_j) - \mathbf{v}_{\mathrm{target},j}\|_2^2$
  $\widehat{\mathcal{L}} \leftarrow \widehat{\mathcal{L}}_{\mathsf{CFM}} + \lambda\, \widehat{\mathcal{L}}_{\mathrm{AN}}$
  Update (θ, ϕ) by a gradient step on $\widehat{\mathcal{L}}$
end for
Output: learned parameters (θ, ϕ) =0

4 Experiments

To provide intuition and validate our proposed method, we conduct experiments on both synthetic and image datasets. First we briefly outline how to efficiently parametrize quantile functions.

4.1 Rational Quadratic Splines

We parameterize each quantile function Q^i using a rational quadratic spline (RQS) Gregory & Delbourgo (1982); Durkan et al. (2019). Unlike normalizing flows that compose many RQS layers within coupling architectures, we use a single RQS per coordinate with directly optimized parameters.

For coordinate i, the spline $S_\phi^i : [-B, B] \to [-B, B]$ is defined by K knots with associated bin widths $\{w_k\}_{k=1}^K$, bin heights $\{h_k\}_{k=1}^K$, and slopes $\{s_k\}_{k=0}^K$. We enforce positivity via softplus and normalize widths and heights to span $2B$. This guarantees strict monotonicity. Outside $[-B, B]$, we extend S_ϕ^i linearly with C^1 continuity at the boundaries.

To map from $(0, 1)$ to the spline domain, we apply $\psi(u) = \mathrm{logit}(u)$ or $\psi(u) = B(2u - 1)$, yielding the quantile

$$Q_\phi^i(u) = a_i \cdot S_\phi^i(\psi(u)) + b_i,$$

where the scale $a_i > 0$ and bias b_i are learned per coordinate.

The total parameter count is $\mathcal{O}(Kd)$: for $K = 32$ bins and $d = 3072$ (CIFAR-10), approximately 300k parameters. This lightweight parametrization, combined with reusing the minibatch OT coupling and freezing the quantile after 55k iterations, introduces minimal overhead compared to standard Gaussian OT-FM. On CIFAR-10 (NVIDIA RTX 5090), we measure approximately 2.7% overhead during joint training and 0.5% after freezing.

4.2 Synthetic Datasets

We begin by qualitatively analyzing our algorithm on several synthetic 2D distributions, see also Appendix B.1, each designed to highlight a specific aspect of our approach. We provide intuition about the learned latent distribution and demonstrate that it is closer to the data in the Wasserstein sense, yields shorter transport paths, and successfully captures the tail behavior.

Gaussian Mixture Model (GMM). We first consider a 2D GMM with nine unevenly weighted modes, as visualized in Figure 4. Due to the independence assumption inherent in our factorized quantile function, the learned latent cannot perfectly replicate the target's joint distribution and is *not the product of the correct marginals*, see also Example 3.1. Instead, it approximates a distribution where the components cannot further independently improve the transport cost to the target.

Funnel Distribution. The funnel distribution (Figure 1) presents a challenge due to its heavy-tailed, conditional structure. Several methods have been proposed in the diffusion context Pandey et al. (2024); Shariatian et al. (2025a;b). We compare to Pandey et al. (2024), where Student-t parameters were hand-selected per dimension. Using a capacity-constrained network (three layers, width 64, no positional embeddings), we observe that a compact latent performs worst, followed by the Gaussian, while our learned latent successfully *adapts to the target's heavy tails*, see Figure 5.[2]

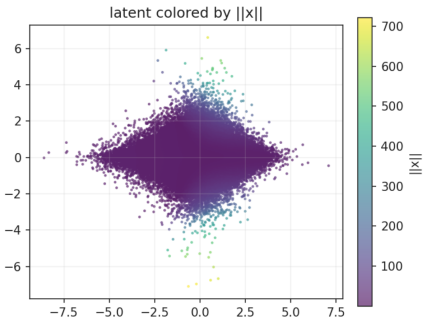


Figure 5: Samples (1M) from our learned latent of the funnel distribution. Color shows norm of the endpoint sample after solving the reverse ODE.

Checkerboard Distribution. The checkerboard in Figure 10 features compact support. Our method learns a latent approximating a uniform distribution over the target's support. Combined with OT coupling, the resulting *transport paths are substantially shorter* than starting from a Gaussian as visualized in Figure 2. In addition, in training, the vector field converges faster, see Figure 6. This underscores our central claim: combining a data-dependent latent with a data-dependent coupling can significantly improve performance.

4.3 Image Datasets

Next, we analyze our method on standard image generation benchmarks. Our quantile is extremely lightweight compared to the U-Net architecture used for the flow model. In high-dimensional settings and given fixed batch sizes, the signal for the quantile function can be noisy, potentially leading to degenerate solutions. To mitigate this, we add a regularization term to the loss that penalizes the expected negative log-determinant of the Jacobian of the quantile. Since the quantile maps from a uniform distribution $U \sim \mathcal{U}[0, 1]$, this term equals the negative differential entropy h of the learned latent

$$h(Q(U)) = h(U) + \mathbb{E}[\log |\det J_Q(U)|] = \mathbb{E}[\log |\det J_Q(U)|].$$

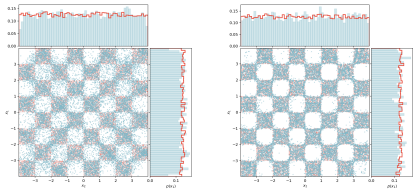


Figure 6: FM with OT coupling using Gaussian noise (top) and our learned noise (bottom) after **20k** steps (same parameters). Starting from learned noise yields much better samples. Blue: generated; red: ground truth.

[2]Due to the high variance when sampling minibatch from the funnel, we pre-train the quantile and use the independent coupling for all models.

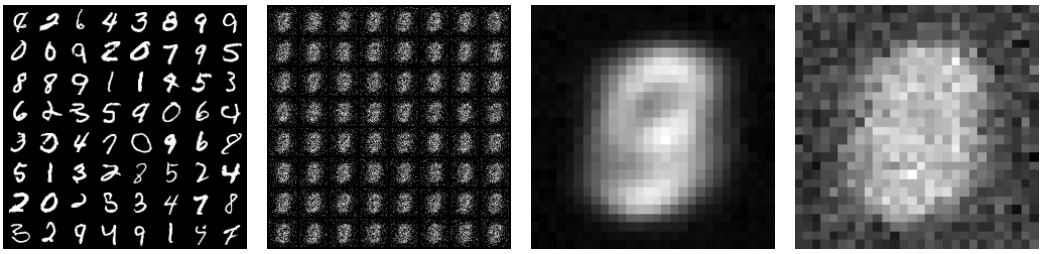


Figure 7: MNIST (left to right): generated samples, samples from the learned latent, and the learned latent's mean and standard deviation.

Access to analytic derivatives makes this efficient, for more details see Appendix A. In total, this yields the loss $\mathcal{L}(\theta, \phi) = \mathcal{L}_{\mathsf{CFM}}(\theta, \phi) + \lambda \mathcal{L}_{\mathrm{AN}}(\phi) + \beta \mathbb{E}[\log |\det J_Q(U)|]$.

MNIST. MNIST exhibits strong marginal structure: center pixels are frequently active while border pixels are nearly always zero. Our learned quantile captures these statistics, concentrating mass in regions corresponding to active pixels as shown in Figure 7. Where a pixel is essentially always black, the learned quantile concentrates around that value; in center regions with higher uncertainty, the quantiles remain spread around zero (gray), accurately reflecting the data variability. We compare learned and empirical quantiles at different pixel locations in Figure 11.

This dataset is well-suited for our method since the marginal structure provides a strong learning signal. In Figure 8, we compare performance under capacity constraints using channels $8, 16, 32$ ($\lambda = 1$, $\beta = 0.1$). Our learned latent achieves significantly lower FIDs across all capacity levels. See Figure 8 in the Appendix for the total amount of parameters used for each model. By removing redundant information in the noise distribution, the network can allocate its parameters more efficiently toward modeling the relevant structure. The independence assumption prevents capturing spatial correlations (e.g., digit shapes).

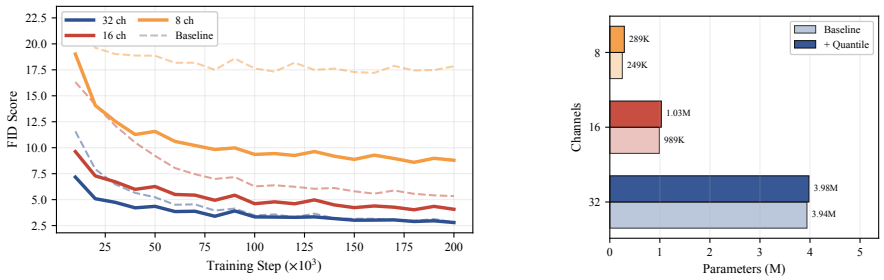
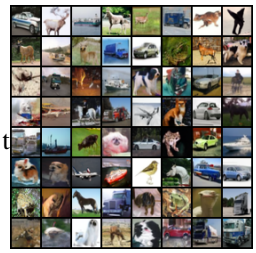


Figure 8: Ablation study over capacity of the U-Net for sampling from the MNIST dataset. The FID curves show that our method achieves significantly lower FIDs for lower capacities. Note the difference in parameters is approximately 40k.

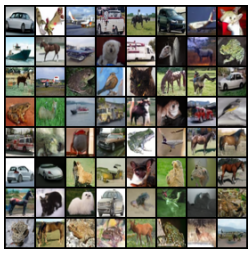
CIFAR-10. We evaluate on CIFAR-10 to demonstrate that our method scales to natural images and integrates seamlessly with standard architectures. We follow the setup of Tong et al. (2024), using the U-Net architecture from Nichol & Dhariwal (2021), to enable a fair comparison. Figure 9 reports results for varying β at fixed $\lambda = 1$. Our method successfully learns a data-adapted latent while maintaining competitive FID scores, confirming that the quantile training remains stable at scale. Improvements are marginal as expected, since CIFAR-10 features strong spatial and inter-channel correlations that a product measure cannot fully capture.

5 Related Work

Current state-of-the-art methods largely rely on corrupting images with Gaussian noise. Diffusion models Song et al. (2021); Ho et al. (2020); Song et al. (2022) and the closely related flow matching



Baseline



$\beta = 0.8$

Euler steps →	**20**	**100**
β-logdet ↓	**FID**	**FID**
0.2	7.81	4.75
0.3	**7.48**	4.53
0.5	7.66	4.49
0.8	7.77	**4.42**
1.0	8.35	4.66
Baseline	8.42	4.63

Figure 9: CIFAR results for a selection of regularization parameters and for the baseline, for complete results see Figure 12. Our method reached the best validation FID after 320k steps, while the baseline took 340k. We evaluated the FID using 5 seeds and report the mean. We used those checkpoints for the evaluation. The visualized samples were generated using 100 Euler steps.

models Liu et al. (2023); Lipman et al. (2023); Albergo et al. (2023) form the basis of most approaches. To further improve inference speed, few-step methods have achieved remarkable results Song et al. (2023); Geng et al. (2025); Zhou et al. (2025).

The following papers keep the Gaussian distribution as a starting point, but updated the interpolation path. There exist only few approaches to learn the noising process, Bartosh et al. (2025) fit the forward diffusion process to the backward via a learned invertible map that is trained end-to-end, Kapusniak et al. (2024) use metric flow matching, i.e., a neural network to adapt the path to a underlying Riemannian metric. In a related approach Sahoo et al. (2024) learns a input-conditioned componentwise Gaussian noise schedule. In the setting of sampling from unnormalized target densities, Blessing et al. (2025a) learn the latent noise by optimizing the mean and covariance of a Gaussian prior. A Gaussian mixture prior has been used both for sampling Blessing et al. (2025b) and for generative modeling Issachar et al. (2025).
Along the lines of normalizing flows and VAEs, there is a large body of work on learning more flexible noise distributions and priors Stimper et al. (2022); Bauer & Mnih (2019); Hickling & Prangle (2025); Amiri et al. (2024).

From a complementary perspective, Wiese et al. (2019) propose to separate marginal modeling from dependence structure using copula and marginal flows, recognizing that standard architectures struggle with tail asymptotics, a motivation conceptually aligned with our componentwise quantile approach. On the other hand Pandey et al. (2024); Zhang et al. (2024) design heavy-tailed diffusions using Student-t latent distributions, and Shariatian et al. (2025b) extend the framework to the family of α-stable distributions.
Recent work by Ghane et al. (2025) explains why heavy-tailed sampling can be challenging: diffusion models driven by Gaussian noise satisfy a concentration-of-measure property. In particular, when initialized from a Gaussian distribution, the generated distribution inherits many Gaussian-like properties. Moreover, by Tam & Dunson (2025), GANs, VAEs and diffusion models with Gaussian or log-concave noise distributions can only generate light-tailed samples and are not universal generators.

6 Conclusions

We introduce a quantile approach for generative modeling—a unifying theoretical framework and practical toolkit that elevates noise selection to a data-driven design choice which integrates seamlessly into FM. Experimental results show that our method can learn highly efficient, freely parameterized latent distributions that go well beyond smooth transformations of Gaussian priors. This work opens several promising directions for future research, including the development of tailored objectives for learning time-dependent quantile functions to optimize entire path distributions, as well as more expressive conditional quantile functions for applications such as class-conditional and text-to-image generation.

Acknowledgments GS and RD acknowledge funding by the German Research Foundation (DFG) within the Excellence Cluster MATH+ and JC by project STE 571/17- 2 within the The Mathematics of Deep Learning. GK acknowledges funding by the BMBF VIScreenPRO (ID: 100715327).

REFERENCES

M. S. Albergo, N. M. Boffi, and E. Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

L. Ambrosio, N. Gigli, and G. Savaré. *Gradient Flows*. Lectures in Mathematics ETH Zürich. Birkhäuser, Basel, 2nd edition, 2008. doi: 10.1007/978-3-7643-8722-8.

Saba Amiri, Eric Nalisnick, Adam Belloum, Sander Klous, and Leon Gommans. Practical synthesis of mixed-tailed data with normalizing flows. *Transactions on Machine Learning Research*, 2024.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces, 2023. URL https://arxiv.org/abs/2107.03006.

G. Bartosh, D. Vetrov, and C. A. Naesseth. Neural flow diffusion models: Learnable forward process for improved diffusion modelling, 2025. URL https://arxiv.org/abs/2404.12940.

Matthias Bauer and Andriy Mnih. Resampled priors for variational autoencoders, 2019. URL https://arxiv.org/abs/1810.11428.

Denis Blessing, Julius Berner, Lorenz Richter, and Gerhard Neumann. Underdamped diffusion bridges with applications to sampling. In *International Conference on Learning Representations (ICLR)*, 2025a. URL https://arxiv.org/abs/2503.01006.

Denis Blessing, Xiaogang Jia, and Gerhard Neumann. End-to-end learning of gaussian mixture priors for diffusion sampler, 2025b. URL https://arxiv.org/abs/2503.00524.

R. T. Q. Chen. torchdiffeq, 2018. URL https://github.com/rtqichen/torchdiffeq.

C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios. Neural spline flows. *Advances in Neural Information Processing Systems*, 32, 2019.

R. Flamary, N. Courty, A. Gramfort, M. Z. Alaya, A. Boisbunon, S. Chambon, L. Chapel, A. Corenflos, K. Fatras, N. Fournier, et al. Pot: Python Optimal Transport. *Journal of Machine Learning Research (JMLR)*, 22(1):3571–3578, 2021.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling, 2025. URL https://arxiv.org/abs/2505.13447.

Reza Ghane, Anthony Bao, Danil Akhtiamov, and Babak Hassibi. Concentration of measure for distributions generated via diffusion models. *arXiv preprint arXiv:2501.07741*, 2025.

J. Gregory and R. Delbourgo. Piecewise rational quadratic interpolation to monotonic data. *IMA Journal of Numerical Analysis*, 2(2):123–130, 1982.

P. L. Hagemann and S. Neumayer. Stabilizing invertible neural networks using mixture models. *Inverse Problems*, 37(7):085002, 2021.

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Tennessee Hickling and Dennis Prangle. Flexible tails for normalizing flows. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Z6RsbHAJk5.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. URL https://arxiv.org/abs/2006.11239.

Emiel Hoogeboom, Victor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d, 2022. URL https://arxiv.org/abs/2203.17003.

Noam Issachar, Mohammad Salama, Raanan Fattal, and Sagie Benaim. Designing a conditional prior distribution for flow-based generative models, 2025. URL https://arxiv.org/abs/2502.09611.

K. Kapusniak, P. Potaptchik, T. Reu, L. Zhang, A. Tong, M. Bronstein, A. J. Bose, and F. Di Giovanni. Metric flow matching for smooth interpolations on the data manifold. *arXiv preprint arXiv:2405.14780*, 2024.

D. P. Kingma and J. Ba. Adam: a method for stochastic optimization. In *Proceedings of the ICLR '15*, 2015.

Y. Lipman, R. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. *ICLR*, 2023.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le1, Brian Karrer, Ricky T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu3, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

Q. Liu. Rectified flow: A marginal preserving approach to optimal transport. *arXiv preprint arXiv:2209.14577*, 2022.

X. Liu, Ch. Gong, and Q. Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *ICLR*, 2023.

R. M. Neal. Slice sampling. *The Annals of Statistics*, 31(3):705–767, 2003. doi: 10.1214/aos/1056562461.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models, 2021. URL https://openreview.net/forum?id=-NEXDKk8gZ.

K. Pandey, J. Pathak, Y. Xu, S. Mandt, M. Pritchard, A. Vahdat, and M. Mardani. Heavy-tailed diffusion models, 2024. URL https://arxiv.org/abs/2410.14171.

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, and Ricky T. Q. Chen. Multisample flow matching: Straightening flows with batch couplings, 2023. URL https://arxiv.org/abs/2304.14772.

Subham Sekhar Sahoo, Aaron Gokaslan, Christopher De Sa, and Volodymyr Kuleshov. Diffusion models with learned adaptive noise. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

A. Salmona, V. De Bortoli, J. Delon, and A. Desolneux. Can push-forward generative models fit multimodal distributions? *Advances in Neural Information Processing Systems*, 35:0766–10779, 2022.

D. Shariatian, U. Simsekli, and A. Durmus. Heavy-tailed diffusion with denoising Lévy probabilistic models, 2025a. URL https://arxiv.org/abs/2407.18609.

D. Shariatian, U. Simsekli, and A. O. Durmus. Heavy-tailed diffusion with denoising levy probabilistic models. *ICLR 2025*, 2025b.

J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/sohl-dickstein15.html.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022. URL https://arxiv.org/abs/2010.02502.

Y. Song and St. Ermon. Generative modeling by estimating gradients of the data distribution. *ArXiv 1907.05600*, 2019.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *ICLR*, 2021.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, 2023.

Vincent Stimper, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Resampling base distributions of normalizing flows, 2022. URL https://arxiv.org/abs/2110.15828.

Edric Tam and David B. Dunson. On the statistical capacity of deep generative models. *arXiv preprint arXiv:2501.07763*, 2025.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport, 2024. URL https://arxiv.org/abs/2302.00482.

C. Wald and G. Steidl. Flow Matching: Markov kernels, stochastic processes and transport plans. In *Variational and Information Flows in Machine Learning and Optimal Transport, Oberwolfach Seminars. Vol. 56*, pp. 185–254. Birkhäuser, 2025.

Magnus Wiese, Robert Knobloch, and Ralf Korn. Copula & marginal flows: Disentangling the marginal from its joint. *arXiv preprint arXiv:1907.03361*, 2019.

T. Zhang, H. Zheng, J. Yao, X. Wang, M. Zhou, Y. Zhang, and Y. Wang. Long-tailed diffusion models with oriented calibration. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NW2s5XXwXU.

L. Zhou, S. Ermon, and J. Song. Inductive moment matching. *ICML 2025*, 2025.

A Details on the Architecture of the Learned Quantiles

We implement each one–dimensional quantile function with rational–quadratic splines (RQS) Gregory & Delbourgo (1982); Durkan et al. (2019). We explored several ways to map $u \in (0,1)$ into the spline input; the two variants below consistently performed well and are used in our experiments. For every coordinate i, we write

$$Q_\phi^i(u) \;=\; S_\phi^i\big(\psi(u)\big), \qquad u \in (0,1),$$

where $S_\phi^i : \mathbb{R} \to \mathbb{R}$ is a strictly increasing RQS with an interior knot interval $(-B, B)$ (with K bins) and linear tails outside $\pm B$ that are C^1-matched at the boundaries. The two settings differ only in the "activation" ψ:

$$\text{(A) Logit: } \psi(u) = \operatorname{logit}(u), \qquad \text{(B) Affine: } \psi(u) = \alpha_B(u) = B(2u-1).$$

Thus, both (A) and (B) share exactly the same spline S_ϕ^i architecture—including the bounded interior $(-B, B)$ and slope-matched linear tails—and differ only in how $(0,1)$ is mapped into the spline's input. In (A), $\psi(u) \in \mathbb{R}$ and the linear tails of S_ϕ^i are used whenever $|\operatorname{logit}(u)| > B$; in (B), $\psi(u) \in (-B, B)$ so the forward pass never touches the tails (they remain important for invertibility and out-of-range evaluation).

Parameterization and Constraints. Each spline S_ϕ^i is parameterized by raw bin widths, heights, and knot slopes. We pass these raw parameters through $\mathrm{softplus}$, normalize widths and heights to sum to one (scaled to the domain span $2B$ and the learned range span, respectively), and add a small constant $s_{\min} > 0$ to each slope to enforce a positive lower bound. The linear tail slopes (left/right) are learned in the same way and are chosen so that both function value and slope agree at $\pm B$. These constraints guarantee strict monotonicity, hence Q_ϕ^i is strictly increasing on $(0,1)$ under both (A) and (B). Closed-form formulas for the spline pieces and their (log-)derivatives are available; by the chain rule,

$$\frac{d}{du} Q_\phi^i(u) = S_\phi^{i\prime}\big(\psi(u)\big)\, \psi'(u), \quad \text{with} \quad \psi'(u) = \begin{cases} \frac{1}{u(1-u)} & \text{for (A),} \\ 2B & \text{for (B).} \end{cases}$$

Per-Component Affine Wrapper (Scale/Bias). After computing $Q_\phi^i(u)$, we add a tiny affine head per coordinate:

$$\tilde{Q}_\phi^i(u) \;=\; s_i\, Q_\phi^i(u) + b_i, \qquad s_i \;=\; \mathrm{softplus}\big(\log \alpha_i\big), \quad b_i \;=\; \beta_i,$$

where $\alpha_i > 0$ and $\beta_i \in \mathbb{R}$ are learned per component. Using $\mathrm{softplus}(\log \alpha_i)$ keeps $s_i > 0$ with a convenient dynamic range; this preserves monotonicity and adds only one scale and one bias parameter per component.

Regularization via Expected Negative Log–Jacobian Let $Q_\phi : (0,1)^d \to \mathbb{R}^d$ be the componentwise map with affine heads, $Q_\phi(u) = \big(\tilde{Q}_\phi^1(u_1), \ldots, \tilde{Q}_\phi^d(u_d)\big)$. Since the construction is per–coordinate, the Jacobian is diagonal with entries $\partial_{u_i} \tilde{Q}_\phi^i(u_i) > 0$. We regularize with the expected negative log–determinant of the Jacobian:

$$\begin{aligned} \mathcal{L}_{\mathrm{reg}}(\phi) &= \lambda_{\mathrm{reg}}\, \mathbb{E}_{u \sim p_U}\big[-\operatorname{logdet} J_{Q_\phi}(u)\big] \\ &= \lambda_{\mathrm{reg}}\, \mathbb{E}_{u \sim p_U}\Big[-\sum_{i=1}^{d} \log\big(\partial_{u_i} \tilde{Q}_\phi^i(u_i)\big)\Big]. \end{aligned}$$

Here $p_U = \mathrm{Unif}\big((0,1)^d\big)$. In practice, we evaluate the log–derivatives in closed form.

Computational Efficiency and Scalability. The quantile architecture is highly efficient in both computation and memory. Each component i requires only $\mathcal{O}(K)$ parameters for the RQS (where K is the number of bins) plus two affine parameters, totaling roughly $4K + 2$ parameters per dimension for typical implementations. For a d-dimensional problem, this yields $\mathcal{O}(d \cdot K)$ total parameters—negligible compared to modern UNet architectures which often contain millions of

parameters. Forward evaluation of $Q_\phi(u)$ involves d independent spline evaluations operating in parallel. The diagonal Jacobian structure means that both the determinant and its gradient reduce to d independent scalar derivatives with analytical closed-form expressions which are fully parallelizable, avoiding expensive automatic differentiation of matrix operations.

In practice, as noted in Section 4, the computational overhead (on CIFAR10) during joint training is approximately 2.7% and drops to 0.5% after freezing the quantile. Furthermore we only used 300k parameters for the quantile in contrast to 35M for the U-Net, making the approach highly scalable to high-dimensional problems. The strict monotonicity constraints and bounded parameterization (via softplus and normalization) ensure numerical stability throughout training, and we observed no instabilities across our experiments spanning dimensions from $d = 2$ to $d = 3072$ (CIFAR-10).

B Experiment Details

B.1 Toy Target Distributions

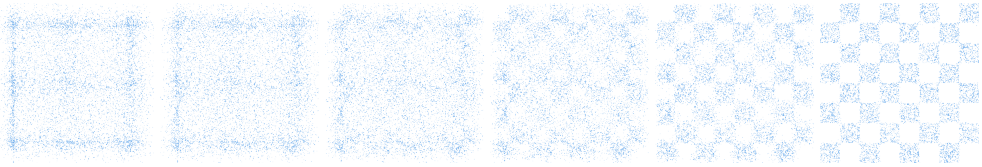


Figure 10: A generated sample path from the learned quantile latent to the checkerboard. The adapted latent (left) is already close to the target distribution.

We use three standard challenging low-dimensional distributions: Neal's funnel, a 3×3 Gaussian mixture, and a checkerboard.

Funnel. For the toy illustration in Figure 1, we work with the dataset known as Neals Funnel Neal (2003). The distribution of Neal's funnel is defined as follows:

$$p(x_1, x_2) \;=\; \mathcal{N}\big(x_1; 0, 3\big)\, \mathcal{N}\big(x_2; 0, \exp(x_1/2)\big).$$

Grid Gaussian Mixture. We give more details about the mixture of Gaussian we consider in our experiment. It is designed in a grid pattern in $[-1, 1]^2$, as follows:

$$\sum_{i=1}^{9} w_i \cdot \mathcal{N}(\mu_i, \sigma^2 I_2)\,,$$

where $(w_i)_{i=1}^9 = (0.01,\, 0.1,\, 0.3,\, 0.2,\, 0.02,\, 0.15,\, 0.02,\, 0.15,\, 0.05)$, $\mu_i = (\mu_1, \mu_2)$ with $\mu_1 = (i \bmod 3) - 1$, $\mu_2 = \left\lfloor \frac{i}{3} \right\rfloor - 1$, and $\sigma = 0.025$.

Checkerboard. Fix $\ell < h$ and domain $\Omega = [\ell, h]^2$. Define the support

$$\mathcal{S} = \big\{(x, y) \in \Omega :\; \lfloor x \rfloor + \lfloor y \rfloor \text{ is even}\big\}.$$

The checkerboard distribution is uniform on $\mathcal{S}$ and zero elsewhere:

$$p_{\text{Checker}}(x, y) = \begin{cases} \dfrac{1}{\text{area}(\mathcal{S})}, & (x, y) \in \mathcal{S}, \\ 0, & \text{otherwise.} \end{cases}$$

For integer ℓ, h with even side length (e.g. $\ell = -4, h = 4$), exactly half of Ω is active, hence

$$p_{\text{Checker}}(x, y) = \frac{2}{(h - \ell)^2}\, \mathbf{1}_{\mathcal{S}}(x, y).$$

B.2 Loss Implementation

For training, the minibatch OT is computed empirically as follows: draw a minibatch $\{\mathbf{x}_0^{(i)}\}_{i=1}^B \sim \mu_0$ and $\{\mathbf{u}^{(j)}\}_{j=1}^B \sim \mathcal{U}([0,1]^d)$, set $\mathbf{y}^{(j)} = \mathbf{Q}_\phi(\mathbf{u}^{(j)})$, and define the empirical measures

$$\hat{\mu}_0^B = \tfrac{1}{B}\sum_{i=1}^B \delta_{\mathbf{x}_0^{(i)}}, \qquad \hat{\nu}_\phi^B = \tfrac{1}{B}\sum_{j=1}^B \delta_{\mathbf{y}^{(j)}}.$$

The minibatch quantile alignment objective is

$$\widehat{\mathcal{L}}_{\mathrm{AN}}(\phi) \;=\; W_2^2\big(\hat{\mu}_0^B, \hat{\nu}_\phi^B\big),$$

and gradients backpropagate through $\mathbf{y}^{(j)} = \mathbf{Q}_\phi(\mathbf{u}^{(j)})$. Let $T : \{1,\dots,B\} \to \{1,\dots,B\}$ denote the optimal assignment that minimizes $\sum_{i=1}^B \|\mathbf{x}_0^{(i)} - \mathbf{y}^{(T(i))}\|_2^2$, and define its inverse $P(j) = i$ such that $T(i) = j$. We use the conditional flow path $\mathbf{x}_t^{(j)} = (1-t_j)\mathbf{x}_0^{(P(j))} + t_j\,\mathbf{y}^{(j)}$, $j = 1,\dots,B$, with $t_j \sim \mathcal{U}(0,1)$. The target velocity is $\mathbf{y}^{(j)} - \mathbf{x}_0^{(P(j))}$, we apply a stop-gradient operator $\mathrm{sg}(\cdot)$ to this target in the flow matching loss. This prevents gradients from the velocity model from flowing back through the quantile function in this term, ensuring that $\mathbf{Q}_\phi$ is updated primarily through $\widehat{\mathcal{L}}_{\mathrm{AN}}$, while v^θ learns to match the transport directions defined by the current quantile map. Note however the stop gradient operation only slightly stabilizes training, we can train with full gradients as well. We optimize the empirical version

$$\widehat{\mathcal{L}}_{\mathsf{CFM}}(\theta,\phi) = \frac{1}{B}\sum_{j=1}^B \big\| v^\theta\big(\mathbf{x}_t^{(j)}, t_j\big) - \mathrm{sg}\,\big(\mathbf{y}^{(j)} - \mathbf{x}_0^{(P(j))}\big)\big\|_2^2, \quad \widehat{\mathcal{L}}(\theta,\phi) = \widehat{\mathcal{L}}_{\mathsf{CFM}}(\theta,\phi) + \lambda\,\widehat{\mathcal{L}}_{\mathrm{AN}}(\phi).$$

B.3 Implementation Details

We support baseline flow matching, optional quantile pretraining, and joint quantile+velocity optimisation. Pretraining fits the RQS transport before optionally freezing it; joint training updates both modules simultaneously. Once the quantile learning rate decays to zero we freeze its weights and continue optimising the velocity field only.

The coupling plans are calculated using the Python Optimal Transport package Flamary et al. (2021). For inference simulate the corresponding ODEs using the torchdiffeq Chen (2018) package. For all models we only used the batch size 128 and learning rate $2e-4$ for the velocities. We use Adam Kingma & Ba (2015) as the optimizer. The quantiles are parameterised by rational-quadratic splines as described in A, we set the minimum bin width and height to $1e-3$ and the minimum slope to $1e-5$. We could in principle stack multiple RQS layers, however for all of our experiments we use one layer.

All models include a sinusoidal time embedding and SiLU activation functions. For both image datasets, we adapt the U-Net from Nichol & Dhariwal (2021) to parametrize our velocity field.

Funnel. For all models we used 3 hidden layers with width 64. We used a batch size of 128, a learning rate of $2e-4$ and exponential moving average on the network weights of 0.999. The baselines were trained for 200,000 iterations. Since there is a very high variance when sampling from the funnel, we pretrain our quantiles and use the frozen quantiles during flow matching. We trained our quantile for 50,000 steps and to compensate we trained our velocity for only 100,000 steps. For the RQS we chose logit activation, 32 bins and a bound of 500.

Grid Gaussian Mixture and Checker. The quantiles were trained for the first 20,000 steps, after which the learning rate was linearly decayed to 0 by step 25,000. For both datasets, we trained the velocity model with 3 layers and a hidden width of 256 for 100,000 steps. Furthermore we used sinusoidal positional embeddings for the checkerboard. For the RQS we chose logit activation, 32 bins with a bound of 50.

MNIST. For the MNIST dataset we use the U-Net with channel multipliers $(1, 2, 4)$, two residual blocks per resolution, attention at 7×7, and 1 attention head. We clip the gradient norm to 1 and

use exponential moving averaging with a decay of 0.99. We test three configurations with base widths of 8, 16, and 32 channels. For these ablation runs, we use quantile loss weight $\lambda = 1.0$, regularization parameter $\beta = 0.1$, and rational quadratic spline with bounded activation, 16 bins and bound 3.0. The quantiles were trained for the first 20,000 steps, after which the learning rate was linearly decayed to 0 over the next 10,000 steps. The images in Figure 7 were generated using our 32 channel configuration.

CIFAR. We use exactly the same U-Net setup from Tong et al. (2024). We clip the gradient norm to 1 and use exponential moving averaging with a decay of 0.9999. To evaluate our results, we use the Fréchet inception distance (FID) Heusel et al. (2017). The quantiles were trained for the first 50,000 steps, after which the learning rate was linearly decayed to 0 by step 55,000. We used $\lambda = 1$ and varied β. For the RQS we used logit activation, 32 bins and a bound of 25.

CIFAR-10 inputs are normalized to $[-1, 1]$ with random horizontal flips.

C Further Experimental Results

For **MNIST**, Figure 11 visualizes the learned latent distribution $\mathbf{Q}_\phi$ at different pixel locations (e.g., the top-left corner, the center, and the mid-right region). The learned latent matches the correct mean, while the increased variance in the top-right region is induced by the differential-entropy regularization, see Section 4.3. We again plot FID over training steps for the capacity-constrained networks, and additionally report the corresponding parameter counts in Figure 8.

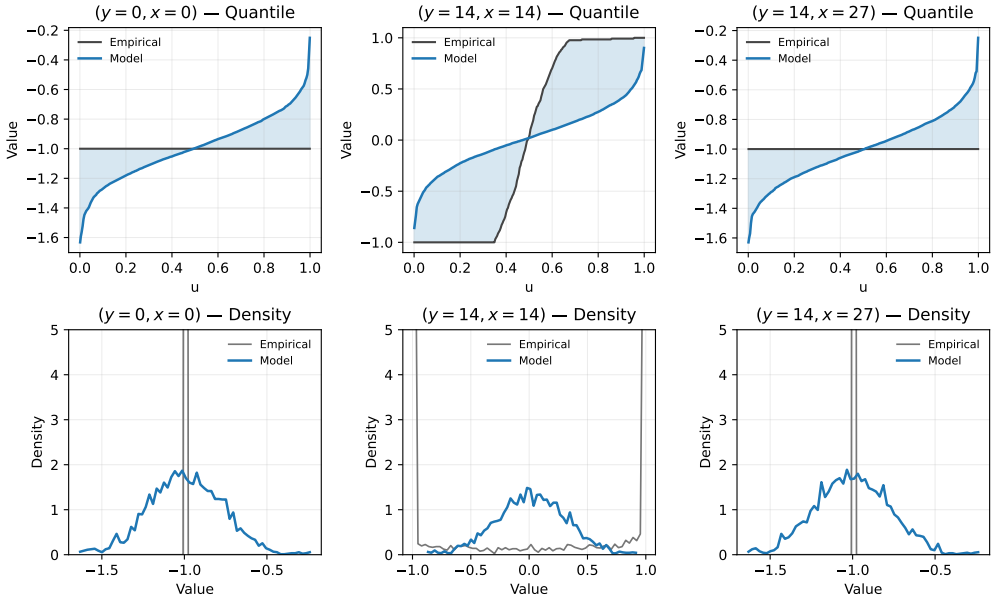


Figure 11: Comparison of the empirical and learned probability density functions and their quantile functions at different pixel locations (y, x), averaged over images from the MNIST dataset. The blue area illustrates the difference between the quantiles, corresponding to the one-dimensional Wasserstein distance; see (3).

For **CIFAR10**, Figure 12, we present the full ablation over the regularization parameter (see Section 4.3). Most regularization settings outperform the standard Gaussian noise distribution.

β	FID (20 steps)	FID (100 steps)
0.1	$8.93_{\pm 0.04}$	$5.22_{\pm 0.02}$
0.2	$7.81_{\pm 0.04}$	$4.75_{\pm 0.02}$
0.3	$\mathbf{7.48}_{\pm 0.05}$	$4.53_{\pm 0.05}$
0.4	$7.60_{\pm 0.05}$	$4.54_{\pm 0.01}$
0.5	$7.66_{\pm 0.03}$	$4.49_{\pm 0.02}$
0.6	$7.70_{\pm 0.05}$	$4.47_{\pm 0.03}$
0.7	$7.93_{\pm 0.05}$	$4.59_{\pm 0.02}$
0.8	$7.77_{\pm 0.05}$	$\mathbf{4.42}_{\pm 0.02}$
0.9	$8.09_{\pm 0.04}$	$4.56_{\pm 0.02}$
1.0	$8.35_{\pm 0.03}$	$4.66_{\pm 0.04}$
1.1	$8.60_{\pm 0.04}$	$4.80_{\pm 0.03}$
Baseline	$8.42_{\pm 0.07}$	$4.63_{\pm 0.05}$

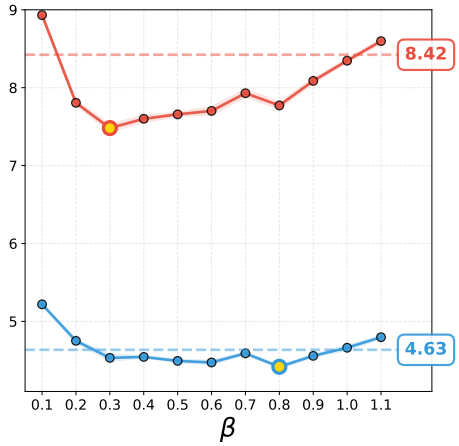


Figure 12: Complete FID scores on CIFAR-10 for all β values. Our method reached the best validation FID after 320k steps, while the baseline took 340k. We used those checkpoints for the evaluation. We evaluated the FID using 5 seeds and report the mean as well as the standard deviation. Red denotes 20 step FID, blue 100 step FID, dotted line refers to baseline.