Structured Coupling for Flow Matching

Xavier Sumba[1], Carles Balsells-Rodas[1], Yingzhen Li[1,2]
[1] Imperial College London [2] Nanyang Technological University
{xxs22,cb221,yingzhen.li}@imperial.ac.uk

Abstract

Standard flow matching scales well but typically relies on an unstructured source distribution, limiting its ability to learn interpretable latent structure. Latent-variable models, by contrast, capture structure but often sacrifice generative quality. We bridge this gap by proposing Structured Coupling for Flow Matching (SCFM), a cooperative framework that augments flow matching with structured latent representation learning. By introducing structured latent variables and exogenous noise into the source, SCFM jointly learns a structured prior (via latent variable modeling) and a continuous transport map (via flow matching). It uses a shared time-dependent recognition network for both latent variable model variational inference and intermediate-time flow velocity estimation. This yields a structurally informed yet unconditional, simulation-free flow model, where the latent variable model can also assist flow sampling. Empirically, SCFM facilitates unsupervised latent representation learning for clustering, disentanglement and downstream tasks, while remaining competitive with flow matching in sample quality, showing that meaningful structure can be learned without sacrificing generative fidelity[1].

1 Introduction

Diffusion models [Ho et al., 2020, Nichol and Dhariwal, 2021, Song et al., 2021a,b] and flow-based models [Lipman et al., 2023, Liu et al., 2023, Gat et al., 2024, Tong et al., 2024, Isobe et al., 2025] are central paradigms in modern generative modeling. Among flow-based approaches, flow matching has emerged as an effective framework for training continuous normalizing flows [Lipman et al., 2023, Eijkelboom et al., 2024, Albergo et al., 2025], combining scalable optimization with strong sample quality [Ma et al., 2024]. However, standard flow matching typically uses a fixed, unstructured source distribution, so the learned transport does not explicitly encourage latent structure useful for interpretation, clustering, disentanglement, or downstream tasks.

Latent-variable models such as variational autoencoders (VAEs) [Kingma and Welling, 2014, Burda et al., 2015] address the complementary problem. By introducing an explicit latent space and amortized inference, they can learn structured and often interpretable representations, including representations that better separate underlying factors of variation [Dilokthanakul et al., 2016, Jiang et al., 2017, Higgins et al., 2017, Chen et al., 2018b, Locatello et al., 2019]. However, their generative quality is often limited by restrictive decoder families or posterior approximations [Burgess et al., 2018]. Our goal is to learn a latent variable that can be inferred from data, sampled from a structured prior, and used to initialize generation.

To close this gap, we propose *Structured Coupling for Flow Matching* (SCFM), a framework that combines structured latent-variable learning with simulation-free flow matching in a stochastic-interpolant formulation [Albergo and Vanden-Eijnden, 2023, Albergo et al., 2025]. SCFM replaces the standard source with an augmented variable $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$, where $\mathbf{z}$ follows a learnable structured prior and ε provides exogenous transport degrees of freedom. During training, an encoder couples

[1] Code is available at `https://github.com/cuent/scfm`.

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

Table 1: Standard flow matching (FM) versus structured coupling for flow matching (SCFM).

Aspect	**FM**	**SCFM**
Sampling source	fixed $p_0(\mathbf{x}_0)$	$\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$, $p_\psi(\mathbf{z})p(\varepsilon)$
Training source marginal	$p_0(\mathbf{x}_0)$	$q_\phi^{\text{agg}}(\mathbf{z})p(\varepsilon)$; endpoint regularization encourages prior alignment
Train coupling	$p_0(\mathbf{x}_0)\, p_{\text{data}}(\mathbf{x}_1)$	$p_{\text{data}}(\mathbf{x}_1)\, q_\phi(\mathbf{z}\,\vert\,\mathbf{x}_1)\, p(\varepsilon)$
Interpolant	$I_t(\mathbf{x}_0, \mathbf{x}_1)$	same linear I_t
Posterior model	none explicit	shared mean for $t < 1$; first d_z coordinates plus endpoint variance head define $q_\phi(\mathbf{z}\,\vert\,\mathbf{x}_1)$ at $t = 1$
Sampling	flow from fixed $p_0(\mathbf{x}_0)$	flow from $p_\psi(\mathbf{z})p(\varepsilon)$; optional decoder proposal plus short refinement
Endpoint branch	none	VAE-style prior and decoder learning
Latent structure	none explicit	structured latents for clustering, disentanglement, and downstream tasks

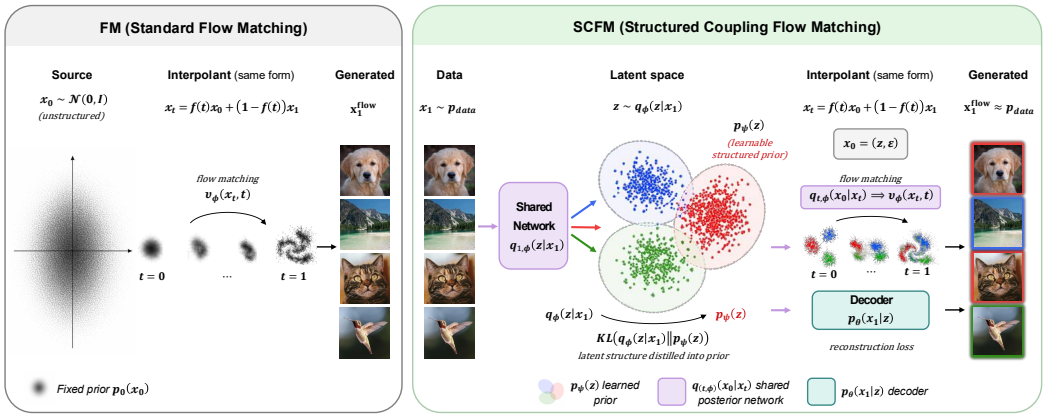

Figure 1: While standard flow matching (left) relies on a fixed, unstructured source, SCFM (right) replaces this with an augmented source $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$. An encoder induces a data-dependent coupling, distilling semantic structure into a learnable prior (indicated by colored latent clusters). A shared posterior-mean network supports endpoint variational inference and intermediate-time flow velocity estimation. This unified objective yields a structured latent space without sacrificing the continuous, simulation-free transport of standard flow models.

data and source variables. A shared time-dependent network parameterizes the encoder mean at $t = 1$ and the posterior mean that determines the flow at $t < 1$. A VAE-style endpoint objective regularizes the aggregated posterior over $\mathbf{z}$ toward the learned sampling prior, encouraging agreement between the training and sampling source marginals. The resulting model supports reconstruction, unconditional generation, and decoder-initialized refinement. We compare joint training with a matched two-stage alternative in Section 4.4.

Our contributions are summarized as follows (also see Figure 1 and Table 1):

- We introduce SCFM, a structured flow-matching framework that jointly learns a latent prior and a continuous transport map. The core idea is to use an augmented source variable $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$, together with an encoder-induced coupling that separates semantic structure from transport flexibility.
- We share a posterior-mean network between endpoint variational inference and intermediate-time flow estimation. Endpoint regularization encourages agreement between the training source marginal and sampling prior, while the decoder supports reconstruction and initializes flow refinement.

- We demonstrate empirically that SCFM learns structural latent representations that are useful for clustering, disentanglement, and downstream classification, while preserving competitive sample generation quality. The decoder-initialized refinement method also improves VAE generation quality towards the levels of flow-matching models.

2 Preliminaries

(Variational) Flow matching. Continuous normalizing flows (CNFs) [Chen et al., 2018a] transform samples from a source distribution into data by solving an ordinary differential equation:

$$\frac{d\mathbf{x}_t}{dt} = v_{\phi,t}(\mathbf{x}_t), \qquad \mathbf{x}_0 \sim p_0(\mathbf{x}_0), \quad t \in [0,1]. \tag{1}$$

Likelihood-based CNF training requires tracking the change of density along the probability path $\{p_t\}_{t\in[0,1]}$, which introduces numerical integration and Jacobian trace terms [Chen et al., 2018a, Grathwohl et al., 2019]. Flow matching [Lipman et al., 2023, Albergo and Vanden-Eijnden, 2023, Albergo et al., 2025, Ma et al., 2024] avoids these costs by learning $v_{\phi,t}$ through regression against velocities induced by a prescribed path between source and target distributions. Once the vector field is learned, sampling simply draws $\mathbf{x}_0 \sim p_0$ and integrates the ODE (1) forward to $t = 1$, producing samples that are approximately distributed as the data distribution $p_{\text{data}}(\mathbf{x}_1)$.

In detail, let $\Gamma(\mathbf{x}_0, \mathbf{x}_1)$ be a coupling between a source distribution $p_0(\mathbf{x}_0)$ and the data distribution $p_{\text{data}}(\mathbf{x}_1)$. With $f(t) = 1 - t$, the interpolant

$$\mathbf{x}_t = f(t)\mathbf{x}_0 + (1 - f(t))\mathbf{x}_1, \qquad t \in [0,1], \tag{2}$$

defines a path from source samples to data samples. The corresponding conditional velocity field is $v_t(\mathbf{x}_t \mid \mathbf{x}_0) = \frac{\partial_t f(t)}{1-f(t)}\big(\mathbf{x}_0 - \mathbf{x}_t\big)$. The marginal velocity field that transports the marginal distribution of $\mathbf{x}_t$ is obtained by averaging this conditional velocity over the posterior source distribution $\Gamma_t(\mathbf{x}_0 \mid \mathbf{x}_t)$, which is defined by the interpolant (2) and the coupling $\Gamma(\mathbf{x}_0, \mathbf{x}_1)$:

$$v_t(\mathbf{x}_t) = \mathbb{E}_{\Gamma_t(\mathbf{x}_0|\mathbf{x}_t)}[v_t(\mathbf{x}_t \mid \mathbf{x}_0)] = \frac{\partial_t f(t)}{1 - f(t)} \left(\mathbb{E}_{\Gamma_t(\mathbf{x}_0|\mathbf{x}_t)}[\mathbf{x}_0] - \mathbf{x}_t\right). \tag{3}$$

Flow matching trains a neural network-based velocity field to approximate the marginal velocity field (3) via regression. Instead, Variational Flow Matching (VFM) [Eijkelboom et al., 2024] introduces a variational posterior model $q_{t,\phi}(\mathbf{x}_0 \mid \mathbf{x}_t)$ to approximate $\Gamma_t(\mathbf{x}_0 \mid \mathbf{x}_t)$, so the induced vector field depends only on the approximate posterior mean (see Appendix A for details):

$$v_{\phi,t}(\mathbf{x}_t) = \frac{\partial_t f(t)}{1 - f(t)}\big(\mathbb{E}_{q_{t,\phi}(\mathbf{x}_0|\mathbf{x}_t)}[\mathbf{x}_0] - \mathbf{x}_t\big).$$

Role of the source. In standard (variational) flow matching, the source distribution $p_0(\mathbf{x}_0)$ is usually fixed to a simple prior, such as an isotropic Gaussian. This choice is convenient for sampling, but the source coordinates are not trained to expose semantic structure; representation structure, if present, is only implicit in the learned transport. As we shall see, SCFM instead makes the source distribution learnable and structured by tying the source posterior in Eq. (3) to VAE-style endpoint losses at the data endpoint, while retaining the simulation-free training advantages of flow matching.

Variational autoencoders. The VAE prior and posterior describe $\mathbf{z}$, while the flow source and target describe the endpoints $\mathbf{x}_0$ and $\mathbf{x}_1$. VAEs [Kingma and Welling, 2014] build latent variable models for generative modelling, where such a model consists of a prior $p_\psi(\mathbf{z})$, potentially with learnable parameters ψ for structural representation learning, and a stochastic decoder $p_\theta(\mathbf{x}_1 \mid \mathbf{z})$. Training is done via minimizing the negative Evidence Lower Bound (ELBO), computed using an approximate posterior (stochastic encoder) $q_\phi(\mathbf{z} \mid \mathbf{x}_1)$:

$$\mathcal{L}_{\text{VAE}}(\theta, \phi, \psi) = \mathbb{E}_{p_{\text{data}}(\mathbf{x}_1)}\Big[- \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_1)}[\log p_\theta(\mathbf{x}_1 \mid \mathbf{z})] + \mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}_1) \,\|\, p_\psi(\mathbf{z})\big)\Big]. \tag{4}$$

While VAEs are powerful for representation learning, they often yield lower sample quality than diffusion/flow-based models. As we shall see, SCFM exploits the VAE framework to structurally inform the flow's source distribution, unifying the representation learning of VAEs with the high-fidelity generation of flow matching.

3 Structured Coupling for Flow Matching

We introduce SCFM, whose key ideas include (1) the encoder-induced coupling (Section 3.1), (2) time-split posterior matching and practical training loss (Section 3.2), and (3) flexible sampling modes (Section 3.3). Table 1 provides a summary of SCFM innovations over standard FM. In essence, SCFM makes the source marginal structural and learnable via encoder-induced coupling, while retaining the same linear interpolant and simulation-free training target.

3.1 Encoder- and Decoder-Induced Couplings

SCFM introduces structure by changing the source endpoint used by flow matching in order to achieve structural latent representation learning. We define the source variable as a concatenation of structural representation $\mathbf{z}$ and exogenous noise ε:

$$\mathbf{x}_0 = (\mathbf{z}, \varepsilon) \in \mathbb{R}^D, \qquad \mathbf{z} \in \mathbb{R}^{d_z}, \qquad \varepsilon \in \mathbb{R}^{d_\varepsilon}, \qquad D = d_z + d_\varepsilon.$$

We train a flow-matching generative model that, at sampling time, uses a learnable source prior factorized into a structural prior over $\mathbf{z}$ and a fixed exogenous-noise prior over ε:

$$p_\psi(\mathbf{x}_0) = p_\psi(\mathbf{z})\, p(\varepsilon), \qquad p(\varepsilon) = \mathcal{N}(0, I_{d_\varepsilon}), \tag{5}$$

where $p_\psi(\mathbf{z})$ is learnable. In our experiments, $p_\psi(\mathbf{z})$ is chosen to impose structure on the latent space through a Gaussian mixture prior. Standard flow matching would typically use the independent coupling $p_\psi(\mathbf{x}_0)p_{\text{data}}(\mathbf{x}_1)$. Instead, SCFM uses the *encoder-induced coupling*

$$\Gamma_\phi^{\text{enc}}(\mathbf{x}_0, \mathbf{x}_1) := p_{\text{data}}(\mathbf{x}_1)\, q_\phi(\mathbf{z} \mid \mathbf{x}_1)\, p(\varepsilon), \qquad \mathbf{x}_0 = (\mathbf{z}, \varepsilon), \tag{6}$$

where $q_\phi(\mathbf{z} \mid \mathbf{x}_1)$ is a learnable stochastic encoder. Thus, each interpolation path starts from a source endpoint whose structured coordinate is inferred from the data endpoint. This makes the latent representation part of the transport problem itself, rather than an auxiliary representation learned beside the flow. Under this coupling, the optimal flow-matching model transports the source marginal

$$\Gamma_\phi^{\text{enc}}(\mathbf{x}_0) = q_\phi^{\text{agg}}(\mathbf{z})\, p(\varepsilon), \qquad q_\phi^{\text{agg}}(\mathbf{z}) := \int p_{\text{data}}(\mathbf{x}_1)\, q_\phi(\mathbf{z} \mid \mathbf{x}_1)\, d\mathbf{x}_1, \tag{7}$$

to the data distribution $p_{\text{data}}(\mathbf{x}_1)$. The resulting gap between the training marginal $\Gamma_\phi^{\text{enc}}(\mathbf{x}_0)$ and the sampling prior $p_\psi(\mathbf{x}_0)$ lies entirely in the structural prior over $\mathbf{z}$. The VAE-style endpoint loss regularizes this mismatch. With a stochastic decoder $p_\theta(\mathbf{x}_1 \mid \mathbf{z})$, the VAE loss (4) is equivalent to

$$\mathcal{L}_{\text{VAE}}(\theta, \phi, \psi) = \text{KL}\big(p_{\text{data}}(\mathbf{x}_1)\, q_\phi(\mathbf{z} \mid \mathbf{x}_1) \,\|\, p_\psi(\mathbf{z})\, p_\theta(\mathbf{x}_1 \mid \mathbf{z})\big) + \text{const}. \tag{8}$$

If this joint KL is zero [Hoffman and Johnson, 2016, Alemi et al., 2018], then $q_\phi^{\text{agg}}(\mathbf{z}) = p_\psi(\mathbf{z})$, and training and sampling use the same source marginal (Appendix B.2). Finite training with the practical endpoint losses need not satisfy this condition.

Jointly training the flow and the latent-variable model has two complementary benefits:

1. **Flow matching gives the latent-variable model an ODE sampler.** When this joint KL is zero, the encoder-induced coupling equals the *decoder-induced coupling*

   $$\Gamma_{\theta,\psi}^{\text{dec}}(\mathbf{x}_0, \mathbf{x}_1) := p_\psi(\mathbf{z})\, p(\varepsilon)\, p_\theta(\mathbf{x}_1 \mid \mathbf{z}), \qquad \mathbf{x}_0 = (\mathbf{z}, \varepsilon).$$

   Its $t = 1$ marginal is the latent-variable model

   $$\Gamma_{\theta,\psi}^{\text{dec}}(\mathbf{x}_1) = p_{\theta,\psi}(\mathbf{x}_1) := \int p_\psi(\mathbf{z})\, p_\theta(\mathbf{x}_1 \mid \mathbf{z})\, d\mathbf{z}.$$

   Under exact flow matching, integrating the ODE samples this marginal:

   $$\mathbf{x}_1 \sim p_{\theta,\psi}(\mathbf{x}_1) \quad \Leftrightarrow \quad \mathbf{x}_0 = (\mathbf{z}, \varepsilon) \sim p_\psi(\mathbf{z})\, p(\varepsilon), \qquad \mathbf{x}_1 = \mathbf{x}_0 + \int_0^1 v_t(\mathbf{x}_t)\, dt. \tag{9}$$

   The equality is in distribution; trained decoders and flows can differ in sample quality. This connection motivates the decoder-initialized refinement scheme in Section 3.3.

2. **VAE enriches flow-matching models via structured latent representation learning.** Beyond matching the prior to the aggregated posterior, the encoder $q_\phi(\mathbf{z} \mid \mathbf{x}_1)$ is trained to approximate the posterior of the latent-variable model $p_{\theta,\psi}(\mathbf{x}_1, \mathbf{z})$, thereby extracting compressed representations of the data into $\mathbf{z}$. Through this VAE objective, the flow itself becomes structurally informed: both the learned source prior $p_\psi(\mathbf{x}_0)$ and the encoder-induced coupling $\Gamma_\phi^{\text{enc}}(\mathbf{x}_0, \mathbf{x}_1)$ that define flow training inherit the latent structure learned in $\mathbf{z}$. Importantly, this is not external conditioning. The flow remains unconditional, while $\mathbf{z}$ still acquires semantic meaning in an unsupervised manner, which later supports disentanglement, clustering, and latent-space classification, and provides a degree of control through the learned latent variable.

3.2 Time-split Posterior Matching and Training Objectives

The remaining question is how to couple the endpoint latent-variable objective to flow matching without a second encoder. In standard FM, under the linear interpolant Eq. (2), the marginal vector field depends on the posterior only through $\mathbb{E}_{\Gamma_t(\mathbf{x}_0|\mathbf{x}_t)}[\mathbf{x}_0]$. Similarly, SCFM uses a single time-dependent posterior-mean network over the structured source endpoint $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$.

Proposition 3.1 (Structured source posterior velocity)**.** *Under the encoder-induced coupling in Eq.* (6) *and the linear interpolant in Eq.* (2)*, the marginal velocity is*

$$v_t(\mathbf{x}_t) = \frac{\partial_t f(t)}{1 - f(t)} \left(\mathbb{E}_{\Gamma_t^{enc}(\mathbf{x}_0|\mathbf{x}_t)}[\mathbf{x}_0] - \mathbf{x}_t \right) = \frac{\partial_t f(t)}{1 - f(t)} \left(\mathbb{E}_{\Gamma_t^{enc}(\mathbf{z},\varepsilon|\mathbf{x}_t)}[(\mathbf{z}, \varepsilon)] - \mathbf{x}_t \right). \tag{10}$$

Intermediate-time regime ($t < 1$). For $t < 1$, SCFM reduces to variational flow matching on the structured source endpoint, using the fixed-covariance Gaussian approximate posterior (encoder)

$$q_{t,\phi}(\mathbf{x}_0 \mid \mathbf{x}_t) = \mathcal{N}\left(\mu_\phi(\mathbf{x}_t, t), \sigma_{\mathbf{x}_0}^2 I_D\right) \quad \Rightarrow \quad v_{\phi,t}(\mathbf{x}_t) = \frac{\partial_t f(t)}{1 - f(t)} \left(\mu_\phi(\mathbf{x}_t, t) - \mathbf{x}_t\right). \tag{11}$$

The induced vector field $v_{\phi,t}(\mathbf{x}_t)$ follows Eq. (10) but with the oracle posterior mean replaced by the encoder mean. The resulting posterior-matching objective is

$$\mathcal{J}_{<1}(\phi) = \mathbb{E}_{t\sim\rho_{<1}} \mathbb{E}_{p_t(\mathbf{x}_t)} \mathrm{KL}\left(\Gamma_t^{\text{enc}}(\mathbf{x}_0 \mid \mathbf{x}_t) \,\|\, q_{t,\phi}(\mathbf{x}_0 \mid \mathbf{x}_t)\right), \tag{12}$$

where $\rho_{<1}$ is a distribution over t supported on $t < 1$, and Γ_t^{enc} is induced by the encoder coupling in Eq. (6). We use the following posterior-model loss, where sg denotes stop-gradient:

$$\mathcal{L}_{\text{VFM}}(\phi) = -\,\mathbb{E}_{t\sim\rho_{<1},(\mathbf{x}_t,\mathbf{x}_0)\sim\Gamma_t^{\text{enc}}(\mathbf{x}_0,\mathbf{x}_t)} \left[\log q_{t,\phi}(\mathrm{sg}(\mathbf{x}_0) \mid \mathrm{sg}(\mathbf{x}_t))\right] + \text{const.}, \tag{13}$$

where $(\mathbf{x}_t, \mathbf{x}_0)$ are sampled from the encoder interpolant-induced joint. In practice, we draw $\mathbf{x}_1 \sim p_{\text{data}}$, then $\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x}_1)$ and $\varepsilon \sim p(\varepsilon)$, set $\mathbf{x}_0 = (\mathrm{sg}(\mathbf{z}), \varepsilon)$, and form $\mathbf{x}_t = f(t)\mathbf{x}_0 + (1 - f(t))\mathbf{x}_1$. Under the fixed-covariance Gaussian family in Eq. (11), this KL is equivalent to posterior-mean regression, and hence to the usual time-weighted velocity regression (see Appendix B.3).

Endpoint regime ($t = 1$). The exact posterior $\Gamma_t^{\text{enc}}(\mathbf{x}_0 \mid \mathbf{x}_t)$ of the encoder-induced coupling converges to $q_\phi(\mathbf{z} \mid \mathbf{x}_1)p(\varepsilon)$ as $t \to 1$, meaning that at $t = 1$ VFM training is no longer suitable. Instead, at $t = 1$ SCFM performs structural learning by minimising the VAE loss Eq. (8), together with decoder training. In particular, the convergence result inspires us to use the *same* network for the mean $\mu_\phi(\mathbf{x}_t, t)$ for both $t < 1$ and $t = 1$. Furthermore, via splitting $\mu_\phi(\mathbf{x}_1, 1) = \left(\mu_\phi^z(\mathbf{x}_1), \mu_\phi^\varepsilon(\mathbf{x}_1)\right)$, the first d_z coordinates of the mean network, together with an endpoint-only variance head branching from the mean network, parameterize the approximate posterior in the VAE

$$q_\phi(\mathbf{z} \mid \mathbf{x}_1) = \mathcal{N}\left(\mu_\phi^z(\mathbf{x}_1), \mathrm{diag}(\sigma_\phi^2(\mathbf{x}_1))\right). \tag{14}$$

The exogenous variable ε is excluded from the decoder $p_\theta(\mathbf{x}_1 \mid \mathbf{z})$. Again due to the exact posterior convergence result, by defining $q_\phi^\varepsilon(\varepsilon \mid \mathbf{x}_1) = \mathcal{N}(\mu_\phi^\varepsilon(\mathbf{x}_1), I_{d_\varepsilon})$, we minimize the KL divergence

$$\mathcal{R}_\varepsilon(\phi) = \frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x}_1)} \left[\left\|\mu_\phi^\varepsilon(\mathbf{x}_1)\right\|^2\right] = \mathrm{KL}(q_\phi^\varepsilon(\varepsilon \mid \mathbf{x}_1) \,\|\, p(\varepsilon)) + \text{const.} \tag{15}$$

Thus at $t = 1$ the endpoint contribution to the total loss is $\mathcal{L}_{\text{end}} = \mathcal{L}_{\text{VAE}} + \mathcal{R}_\varepsilon$. Appendix B.1 gives the endpoint-objective derivation.

Flow–encoder network sharing. The same mean network $\mu_\phi(\mathbf{x}_t, t)$ serves as the intermediate-time posterior estimator and, at $t = 1$, as the encoder mean $\mu^z_\phi(\mathbf{x}_1)$; only the encoder variance head is endpoint-specific. Both losses update the shared mean network; the flow loss stops gradients through the sampled endpoint latents. The endpoint VAE term also encourages $q^{\mathrm{agg}}_\phi(\mathbf{z}) \approx p_\psi(\mathbf{z})$, making the encoder-induced training marginal compatible with the sampling prior; see Eq. (8) and Section 3.1. This is a tractable regularizer rather than a guarantee of exact marginal alignment.

Total loss objective. In practice, one SCFM training step, summarized in Algorithm 1 in appendix, optimizes the decomposed objective

$$\mathcal{L}_{\mathrm{SCFM}}(\theta, \phi, \psi) = \mathcal{L}_{\mathrm{VFM}}(\phi) + \mathcal{L}_{\mathrm{rec}}(\theta, \phi) + \mathcal{L}_{\mathrm{KL}}(\phi, \psi) + \mathcal{R}_\varepsilon(\phi). \tag{16}$$

Here $\mathcal{L}_{\mathrm{VFM}}$ trains the shared posterior model away from the endpoint, $\mathcal{L}_{\mathrm{rec}} + \mathcal{L}_{\mathrm{KL}} = \mathcal{L}_{\mathrm{VAE}}$ is the endpoint VAE term, and $\mathcal{R}_\varepsilon$ anchors the exogenous coordinates. We use β-VAE [Higgins et al., 2017] or β-TCVAE [Chen et al., 2018b] endpoint regularization under a GMM prior. Reconstruction uses dataset-specific losses, including MSE, ℓ_1, and perceptual terms; see Appendix B.4.

3.3 Sampling via Decoder-Initialized Refinement

SCFM supports two flow-based sampling modes. The standard mode (Algorithm 2 in appendix) is full ODE integration, similar to standard flow matching. For unconditional generation, we draw $\mathbf{z} \sim p_\psi(\mathbf{z})$ and $\varepsilon \sim p(\varepsilon)$, set $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$, and integrate $v_{\phi,t}(\mathbf{x}_t)$ from $t = 0$ to $t = 1$. As SCFM also trains an encoder, if an observation $\mathbf{x}_1$ is available, the same procedure yields a reconstruction by replacing the prior draw with $\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x}_1)$ while still sampling $\varepsilon \sim p(\varepsilon)$.

The decoder in SCFM provides an alternative sampling mode, as it can be used to initialize the flow near the endpoint. This *decoder-initialized refinement* mode is summarized in Algorithm 3 in appendix. Given $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$, we first sample $\widehat{\mathbf{x}}_1 \sim p_\theta(\cdot \mid \mathbf{z})$ and then initialize the interpolant at some $t_0 \in [0, 1)$ via $\mathbf{x}_{t_0} = f(t_0)\mathbf{x}_0 + (1 - f(t_0))\widehat{\mathbf{x}}_1$. We then integrate $v_{\phi,t}(\mathbf{x}_t)$ only on $[t_0, 1]$, trading fewer ODE evaluations for a stronger dependence on decoder quality. If $\Gamma^{\mathrm{dec}}_{\theta,\psi} = \Gamma^{\mathrm{enc}}_\phi$, this initialization has the training interpolant's marginal at t_0. Practical endpoint losses and finite training need not achieve this equality.

4 Experiments

We evaluate SCFM along three axes: structured representations, disentanglement, and sample quality. We report quantitative metrics and qualitative visualizations; Appendix F defines all metrics.

Experimental setups. All SCFM models use a learnable GMM prior over $\mathbf{z}$. MNIST [LeCun et al., 2002] clustering and Cars3D [Reed et al., 2015]/Shapes3D [Kim and Mnih, 2018] disentanglement use separate models with β-VAE and β-TCVAE endpoint losses; Appendices E.3 and E.5 give the corresponding setups. In contrast, CIFAR-10 [Krizhevsky et al., 2009] and ImageNet-128 [Russakovsky et al., 2015] reuse the same trained SCFM model for both latent evaluation and image generation. CIFAR-10 uses a U-Net backbone with a β-VAE endpoint objective augmented by LPIPS [Johnson et al., 2016, Zhang et al., 2018]; Appendix E.4 gives the training details. ImageNet-128 follows a latent-diffusion-style setup: a pretrained Stable-Diffusion VAE maps images to VAE latents, and SCFM is trained directly in this latent image space using a SiT-XL/2 backbone [Rombach et al., 2022, Ma et al., 2024]. Appendix E.6 gives the ImageNet-128 model, training, and sampling setup, and Appendix E.7 gives the probing setup.

4.1 Structured Latent Representations

We first evaluate the representations learning capabilities of SCFM. On MNIST we measure cluster alignment, while on CIFAR-10 and ImageNet-128 we measure how much class information remains accessible from frozen latents without label supervision by training probing classifiers post-hoc.

MNIST clustering. On MNIST, we use a learnable GMM prior with $K = 10$ and compare SCFM against VaDE [Jiang et al., 2017] and MFCVAE [Falck et al., 2021]. SCFM is trained with β-VAE [Higgins et al., 2017] and β-TCVAE [Chen et al., 2018b] endpoint losses. We report normalized mutual information (NMI) and clustering accuracy (ACC) over five runs.

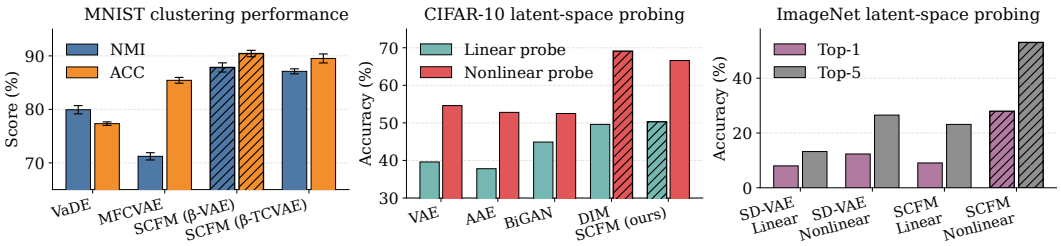

Figure 2: **Structured latent representations.** Left: MNIST clustering metrics, reported as mean with standard-deviation error bars over five runs. Middle: CIFAR-10 downstream probe accuracy from learned latent representations. Right: ImageNet latent-space probing with frozen representations.



Figure 3: **Qualitative disentanglement via factor swaps.** Left: Cars3D factor swaps. Right: Shapes3D factor swaps. Each column shows one source-target pair; each swap row transfers one target factor while preserving the others.

	Method	Cars3D		Shapes3D	
		FactorVAE ↑	DCI ↑	FactorVAE ↑	DCI ↑
VAE	β-VAE	0.887 ± 0.039	0.218 ± 0.045	0.883 ± 0.091	0.624 ± 0.122
	β-TCVAE	0.855 ± 0.082	0.140 ± 0.019	0.873 ± 0.074	0.613 ± 0.114
Diffusion	DisDiff	$\underline{0.976 \pm 0.018}$	0.232 ± 0.019	0.902 ± 0.043	0.723 ± 0.013
	FDAE	0.912 ± 0.020	0.329 ± 0.061	0.998 ± 0.003	0.762 ± 0.064
	EncDiff	0.948 ± 0.017	$\mathbf{0.357 \pm 0.072}$	$\mathbf{0.999 \pm 0.001}$	$\mathbf{0.952 \pm 0.028}$
	DyGA	0.846 ± 0.015	0.307 ± 0.032	0.958 ± 0.044	$\underline{0.833 \pm 0.054}$
Flow	SCFM (β-VAE)	0.940 ± 0.019	$\underline{0.337 \pm 0.082}$	0.836 ± 0.034	0.779 ± 0.065
	SCFM (β-TCVAE)	$\mathbf{0.977 \pm 0.027}$	$\mathbf{0.357 \pm 0.078}$	$\underline{0.957 \pm 0.083}$	0.828 ± 0.054

Table 2: **Quantitative comparison on Cars3D and Shapes3D using FactorVAE score and DCI (mean±std).** For SCFM, we report both β-VAE and β-TCVAE endpoint regularizers. Best results are shown in **bold** and second-best results are underlined.

Figure 2 (left) summarizes the results, with full values in Appendix Table 15. SCFM (β-VAE) performs best, followed by SCFM (β-TCVAE). Relative to VaDE, the best SCFM variant improves NMI by nearly 8 points and ACC by more than 13 points.

Figure 4 shows a largely label-aligned latent partition under the learned GMM, with most ambiguity between visually similar digits such as 3 and 5. Overall, $\mathbf{z}$ learns cluster-discriminative structure rather than acting only as a sampling coordinate. Appendix G.4 provides the full comparison across models. Clustering performance remains high for $K = 10$–20, even when the number of mixture components exceeds the number of digit classes (Appendix G.4.1).

CIFAR-10 representation quality. On CIFAR-10, we set $K = 10$ as a coarse structured prior, without assuming a one-to-one correspondence between mixture components and class labels. We therefore evaluate representation quality by freezing $\mathbf{z}$ and training linear and nonlinear probes without data augmentation. SCFM results are averaged over five seeds, while baseline numbers are taken from Zhang et al. [2022].

Figure 2 (middle) shows that SCFM achieves the best linear probe accuracy and the second-best nonlinear probe accuracy; full values are in Appendix Table 17. It outperforms VAE [Kingma and Welling, 2014], AAE [Makhzani et al., 2015], and BiGAN [Donahue et al., 2016] on both probes while remaining competitive with DIM [Hjelm et al., 2019]. The source therefore retains class-relevant information despite being trained without labels. Appendix G.6 shows reliable separation of vehicle classes, with most confusion among visually similar animal classes; Appendix G.5 visualizes samples across mixture-components with coherent appearance statistics.

ImageNet-128 representation quality. We next test whether these representation gains persist at ImageNet scale. SCFM is trained in the Stable-Diffusion VAE latent space with a $K = 100$ GMM prior, and we evaluate frozen $\mathbf{z}$ with linear and nonlinear probes.

Figure 2 (right) summarizes the ImageNet-128 results, with full values in Appendix Table 18. Relative to a frozen SD-VAE [Rombach et al., 2022] encoder, SCFM improves linear Top-1/Top-5 accuracy from $8.00/13.23$ to $9.07/23.14$ and nonlinear Top-1/Top-5 accuracy from $12.34/26.54$ to $27.96/53.08$. These gains indicate that the structured latent variable $\mathbf{z}$ retains substantially more class-accessible information than the pretrained VAE latent baseline.

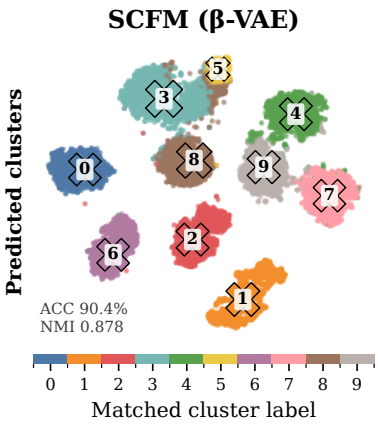

Figure 4: MNIST latents.

4.2 Disentanglement

To test whether the learned structured latent variable supports controllable generative factors, we evaluate SCFM on Cars3D and Shapes3D, using ground-truth factors only for evaluation. Following Locatello et al. [2019], models are trained without factor labels, with latent dimension 10 and a learnable GMM prior with $K = 10$ components. For each endpoint regularizer, we train 10 models while sweeping β; qualitative swaps are taken from the best run.

Table 2 reports FactorVAE and DCI disentanglement scores, with VAE and diffusion baselines from Chi et al. [2026]. SCFM is competitive on both datasets. On Cars3D, SCFM (β-TCVAE) attains the best FactorVAE score and a DCI score comparable to the strongest baselines, while SCFM (β-VAE) remains competitive on both metrics. On Shapes3D, prior diffusion models perform best, but both SCFM variants remain strong in the fully unsupervised setting. Figure 3 shows qualitative factor swaps obtained by interpolating in latent space and generated with full ODE sampling. SCFM transfers target factors while largely preserving the remaining attributes, although cylinder-to-cube transitions remain harder. Overall, SCFM learns factor-sensitive latent structure. Appendix G.3 compares joint SCFM with independently trained endpoint models; the results vary across datasets and metrics.

4.3 Image Generation

We evaluate sample quality against vanilla FM controls with similar parameter counts and the same number of optimization steps (Table 3; Appendix E.1). On CIFAR-10, SCFM obtains 2.121 FID versus 2.183 for matched FM. These results indicate comparable sample quality between the two trained models. Figure 5 shows representative full-flow SCFM samples on CIFAR-10 and ImageNet-128.

On ImageNet-128, increasing vanilla FM from 675M to 1.1B parameters improves FID from 26.349 to 23.854. SCFM reaches 17.180 with 1.1B parameters and the same 400K training steps, at higher forward cost (153G versus 94.6G FLOPs). SCFM samples without class labels; the class-conditional SiT-XL/2 reference obtains 17.243.

Short decoder-initialized refinements trade sample quality for lower sampling cost (Figure 6). At 50 NFE, refinement obtains 3.103 FID versus 2.121 for full flow (Appendix G.2). Equal NFE does not imply equal FLOPs because refinement also requires a decoder pass. Appendix G.1 reports the individual FID evaluations.

Table 3: **Image generation.** FID 50K and total parameters.

Dataset	Model	Params	FID ↓
CIFAR-10	FM	73.6M	2.137
	Matched FM	104M	2.183
	SCFM	101M	**2.121**
ImageNet-128	SiT-XL/2 (cond.)	675M	17.243
	SiT-XL/2 (uncond.)	675M	26.349
	Matched FM	1.1B	23.854
	SCFM	1.1B	**17.180**

Table 4: **Joint versus separate training.** CIFAR-10 FID 50K and probe accuracy (%).

Generator	FID ↓	Linear ↑	Nonlin. ↑
Independent β-VAE decoder	27.646	46.28	61.28
Frozen β-VAE source / FM	15.733	46.28	61.28
Joint SCFM flow	**2.121**	**50.26**	**66.58**



Figure 5: **Representative full-flow SCFM samples.** ImageNet-128 samples are shown above and CIFAR-10 samples below.

Additional uncurated samples for all datasets appear in Appendix H.

4.4 Effect of Joint Training

To compare joint and separate training, we first train the same structured β-VAE endpoint model, then freeze it and train a randomly initialized data-space FM from its structured source. This two-stage control uses the same latent dimension, GMM component count, flow backbone, flow-training steps, sampler, NFE, and FID pipeline as SCFM. We also evaluate the frozen decoder separately. Appendix E.2 details the protocol and computational costs.

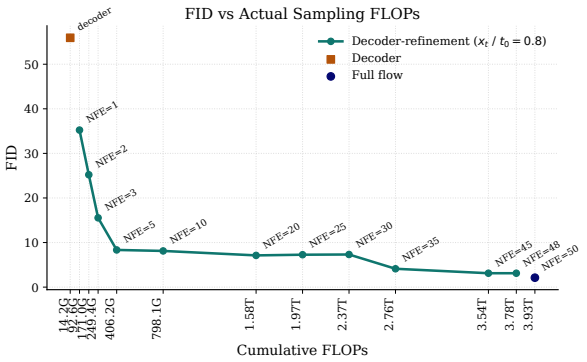

Figure 6: **CIFAR-10 FID vs. cumulative sampling FLOPs.** We compare decoder-only sampling, decoder refinement from x_t with $t_0 = 0.8$, and full-flow sampling. Short refinements reduce sampling cost; full flow has lower FID at 50 NFE.

Joint SCFM reduces FID from 15.733 to 2.121 and improves linear and nonlinear probe accuracy by 3.98 and 5.30 percentage points, respectively (Table 4). The disentanglement comparison is mixed: joint training yields higher Cars3D FactorVAE and Shapes3D DCI scores, whereas the independent endpoint model has a higher Shapes3D FactorVAE score (Appendix G.3).

5 Discussion

Related work overview Related work around SCFM falls into three main categories. First, latent-variable models such as VAEs and their structured-prior extensions learn useful representations

for clustering and disentanglement, but their generation quality is typically constrained by the decoder family and by direct sampling from the latent-variable model [Kingma and Welling, 2014, Burda et al., 2015, Dilokthanakul et al., 2016, Jiang et al., 2017, Falck et al., 2021, Higgins et al., 2017, Burgess et al., 2018, Chen et al., 2018b, Locatello et al., 2019]. Second, diffusion- and transport-based methods improve the source distribution or coupling geometry through learned priors, optimal-transport pairings, or variational couplings, but generally do not treat the source itself as a learned representation space [Sang-gil et al., 2022, Guan et al., 2023, Tong et al., 2024, Albergo et al., 2024, Wang et al., 2024, Silvestri et al., 2025]. Third, recent flow-matching approaches begin to target representation learning more directly through latent-variable transport, structured flow autoencoding, disentangled flows, or joint encoder-generator training [Guo and Schwing, 2025, Zhang et al., 2025, Xu et al., 2026, Ukita and Okita, 2026, Chi et al., 2026]. SCFM is closest to the last category, but differs in that it learns a structured *source distribution* for a stochastic interpolant: the latent variable is part of the source endpoint being transported, not only a conditioning signal or a factorization of the velocity field. Appendix D provides an extended related-work discussion.

Comparison with guidance SCFM is distinct from guidance-based generation. Classifier-free guidance and conditional flow-matching models steer sampling by injecting external conditioning signals, such as labels or prompts, into the denoising or velocity network [Ho and Salimans, 2022]. SCFM instead samples unconditionally from $p_\psi(\mathbf{z})p(\varepsilon)$, without labels, prompts, or an external guidance term. The latent variable $\mathbf{z}$ can still shape generation because it is part of the source endpoint transported by the flow. Thus, SCFM provides implicit structural control through the learned source distribution, rather than explicit conditioning at sampling time.

Summary and future work SCFM learns a structured source for flow matching, using the same latent variables for inference and generation. Compared with parameter- and step-matched FM, it achieves comparable CIFAR-10 FID and lower ImageNet-128 FID. Joint training also improves CIFAR-10 probe accuracy and FID over the two-stage alternative, while disentanglement results vary across datasets and metrics.

The endpoint model adds training overhead: SCFM uses more forward FLOPs than FM at matched parameter counts and training steps (Table 5). The practical endpoint objectives and finite optimization also leave a possible mismatch between the aggregated posterior and sampling prior. Decoder-initialized refinement depends on decoder quality and can be limited by posterior collapse or poor reconstruction. Reducing the endpoint overhead and improving decoder robustness are directions for future work.

Acknowledgments and Disclosure of Funding

X. S. and C. B.-R. acknowledge PhD scholarship support from the Department of Computing at Imperial College London.

References

Michael Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025. URL `http://jmlr.org/papers/v26/23-1605.html`.

Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=li7qeBbCR1t`.

Michael Samuel Albergo, Mark Goldstein, Nicholas Matthew Boffi, Rajesh Ranganath, and Eric Vanden-Eijnden. Stochastic interpolants with data-dependent couplings. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 921–937. PMLR, 21–27 Jul 2024. URL `https://proceedings.mlr.press/v235/albergo24a.html`.

Alexander Alemi, Ben Poole, Ian Fischer, Joshua Dillon, Rif A. Saurous, and Kevin Murphy. Fixing a broken ELBO. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages

159–168. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/alemi18a.html.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in beta-vae. *arXiv preprint arXiv:1804.03599*, 2018.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018a. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf.

Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. *Advances in neural information processing systems*, 31, 2018b.

Jinjin Chi, Taoping Liu, Mengtao Yin, Ximing Li, Yongcheng Jing, and Dacheng Tao. Disentangled representation learning via flow matching. *arXiv preprint arXiv:2602.05214*, 2026.

Nat Dilokthanakul, Pedro AM Mediano, Marta Garnelo, Matthew CH Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

Nicolas Dufour, Alexei A Efros, and Patrick Perez. The FID lottery: Quantifying hidden randomness in generative model evaluation. In *ECCV 2026 Workshop on Empirical Theory in Representation Learning*, 2026. URL https://openreview.net/forum?id=O1uayf5dX2.

Cian Eastwood and Christopher K. I. Williams. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=By-7dz-AZ.

Floor Eijkelboom, Grigory Bartosh, Christian A. Naesseth, Max Welling, and Jan-Willem van de Meent. Variational flow matching for graph generation. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 11735–11764. Curran Associates, Inc., 2024. doi: 10.52202/079017-0374. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/15b780350b302a1bf9a3bd273f5c15a4-Paper-Conference.pdf.

Fabian Falck, Haoting Zhang, Matthew Willetts, George Nicholson, Christopher Yau, and Chris C Holmes. Multi-facet clustering variational autoencoders. *Advances in Neural Information Processing Systems*, 34:8676–8690, 2021.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky TQ Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. *Advances in Neural Information Processing Systems*, 37: 133345–133385, 2024.

Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, and David Duvenaud. Scalable reversible generative models with free-form continuous dynamics. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJxgknCcK7.

Jiaqi Guan, Xiangxin Zhou, Yuwei Yang, Yu Bao, Jian Peng, Jianzhu Ma, Qiang Liu, Liang Wang, and Quanquan Gu. Decompdiff: diffusion models with decomposed priors for structure-based drug design. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Pengsheng Guo and Alex Schwing. Variational rectified flow matching. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Rk18ZikrFI.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bklr3j0cKX.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2022. URL https://openreview.net/forum?id=qw8AKxfYbI.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Matthew D Hoffman and Matthew J Johnson. Elbo surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in advances in approximate Bayesian inference, NIPS*, volume 1, 2016.

Noboru Isobe, Masanori Koyama, Jinzhe Zhang, Kenji Fukumizu, and Kohei Hayashi. Extended flow matching : a method of conditional generation with generalized continuity equation, 2025. URL https://openreview.net/forum?id=0QJPszYxpo.

Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. Variational deep embedding: an unsupervised and generative approach to clustering. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 1965–1972. AAAI Press, 2017. ISBN 9780999241103.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 694–711, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46475-6.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *International conference on machine learning*, pages 2649–2658. PMLR, 2018.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

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

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 2002.

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

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4114–4124. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/locatello19a.html.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pages 23–40. Springer, 2024.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

Scott E Reed, Yi Zhang, Yuting Zhang, and Honglak Lee. Deep visual analogy-making. *Advances in neural information processing systems*, 28, 2015.

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.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115 (3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Lee Sang-gil, Heeseung Kim, Chaehun Shin, Xu Tan, Chang Liu, Qi Meng, Tao Qin, Wei Chen, Sungroh Yoon, and Tie-Yan Liu. Priorgrad: Improving conditional denoising diffusion models with data-dependent adaptive prior. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=_BNiN4IjC5.

Gianluigi Silvestri, Luca Ambrogioni, Chieh-Hsin Lai, Yuhta Takida, and Yuki Mitsufuji. VCT: Training consistency models with variational noise coupling. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=CMoX0BEsDs.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS.

Alexander Tong, Kilian FATRAS, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=CD9Snc73AW. Expert Certification.

Kosuke Ukita and Tsuyoshi Okita. High-performance self-supervised learning by joint training of flow matching. In *The 29th International Conference on Artificial Intelligence and Statistics*, 2026. URL https://openreview.net/forum?id=yW5dvLytON.

Zhanpeng Wang, Shenghao Li, Chen Wang, Shuting Cao, Na Lei, and Zhongxuan Luo. Solving Prior Distribution Mismatch in Diffusion Models via Optimal Transport. *arXiv e-prints*, art. arXiv:2410.13431, October 2024. doi: 10.48550/arXiv.2410.13431.

Yidan Xu, Yixin Wang, and XuanLong Nguyen. Structured flow autoencoders: Learning structured probabilistic representations with flow matching. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=KYdfvF2SZN.

Mingtian Zhang, Tim Z. Xiao, Brooks Paige, and David Barber. Improving vae-based representation learning, 2022. URL https://arxiv.org/abs/2205.14539.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Yichi Zhang, Yici Yan, Alex Schwing, and Zhizhen Zhao. Towards hierarchical rectified flow. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=6F6qwdycgJ.

Appendix for "Structured Coupling for Flow Matching"

Contents

A Extended Variational Flow-Matching Background **16**

B SCFM: Extended Derivations and Implementation Details **17**

B.1 Endpoint regime: posterior identity and endpoint objective 17

B.2 Coupling consistency and prior matching 17

B.3 Intermediate-time regime: specialization to VFM 18

B.4 Practical SCFM Objective and Prior Parameterization 19

C SCFM Algorithms **19**

C.1 Training 19

C.2 Sampling 19

D Extended Related Work **19**

E Experimental Details **21**

E.1 Parameter- and Step-Matched Flow Controls 21

E.2 Independent Endpoint Model and Two-Stage Flow 22

E.3 MNIST Clustering Setup 22

E.4 CIFAR-10 Setup 22

E.5 Cars3D and Shapes3D Disentanglement Setup 23

E.6 ImageNet-128 Setup 23

E.7 CIFAR-10 and ImageNet-128 Probing Setup 26

F Evaluation Metrics **26**

F.1 Clustering Metrics 26

F.2 Disentanglement Metrics 27

F.3 Generative Quality Metric 27

F.4 Downstream Representation Metrics 28

G Additional Results and Diagnostics **28**

G.1 FID Variability Across Sampling Seeds 28

G.2 Decoder Initialization at Matched NFE 28

G.3 Independent Endpoint Disentanglement Controls 29

G.4 MNIST Clustering and Latent Diagnostics 29

G.5 CIFAR-10 Component-Level Generation Diagnostics 31

G.6 CIFAR-10 Latent Probe Diagnostics 31

G.7 ImageNet-128 Latent Probe Diagnostics 32

H Additional Qualitative Samples **33**

I Broader Impacts **38**

Appendix organization. Appendix A reviews the VFM background, and Appendix B gives the SCFM derivations and implementation details. Appendix C summarizes training and sampling, Appendix D extends the related-work discussion, and Appendices E–H collect experimental details, metrics, and additional results.

A Extended Variational Flow-Matching Background

This appendix recalls the variational interpretation of flow matching introduced by Eijkelboom et al. [2024], specialized to the source-endpoint convention used in the main text. The key point is that the marginal vector field can be written as a posterior expectation over endpoints of the interpolant. Approximating this posterior with a variational distribution yields a variational inference objective; under a Gaussian variational family and a linear interpolant, this objective reduces to the usual flow-matching regression loss.

Let $\Gamma(\mathbf{x}_0, \mathbf{x}_1)$ be a coupling between a source endpoint $\mathbf{x}_0$ and a data endpoint $\mathbf{x}_1$, and let

$$\mathbf{x}_t = f(t)\mathbf{x}_0 + (1 - f(t))\mathbf{x}_1.$$

For the source-to-data convention used in the main text, conditioning on $\mathbf{x}_0$ gives the conditional velocity

$$v_t(\mathbf{x}_t \mid \mathbf{x}_0) = \frac{\partial_t f(t)}{1 - f(t)}\big(\mathbf{x}_0 - \mathbf{x}_t\big). \tag{17}$$

The marginal vector field is the posterior average of these conditional velocities,

$$v_t(\mathbf{x}_t) = \mathbb{E}_{\Gamma_t(\mathbf{x}_0|\mathbf{x}_t)}\left[v_t(\mathbf{x}_t \mid \mathbf{x}_0)\right], \tag{18}$$

where $\Gamma_t(\mathbf{x}_0 \mid \mathbf{x}_t)$ is the posterior over source endpoints induced by the interpolant. Thus learning the marginal vector field can be viewed as approximating this endpoint posterior.

Variational flow matching introduces a recognition model $q_{t,\phi}(\mathbf{x}_0 \mid \mathbf{x}_t)$ and minimizes

$$\mathcal{L}_{\mathrm{VFM}}(\phi) = \mathbb{E}_{t,\mathbf{x}_t}\left[\mathrm{KL}(\Gamma_t(\cdot \mid \mathbf{x}_t) \,\|\, q_{t,\phi}(\cdot \mid \mathbf{x}_t))\right]. \tag{19}$$

Equivalently, up to a constant independent of ϕ,

$$\mathcal{L}_{\mathrm{VFM}}(\phi) \equiv -\mathbb{E}_{t,\mathbf{x}_t,\mathbf{x}_0}\left[\log q_{t,\phi}(\mathbf{x}_0 \mid \mathbf{x}_t)\right], \qquad \mathbf{x}_0 \sim \Gamma_t(\cdot \mid \mathbf{x}_t). \tag{20}$$

This is the variational inference view: flow matching is recast as posterior approximation over the endpoint variable that generated the intermediate state.

Now assume a fixed-covariance Gaussian recognition family,

$$q_{t,\phi}(\mathbf{x}_0 \mid \mathbf{x}_t) = \mathcal{N}\big(\mu_\phi(\mathbf{x}_t, t), \sigma_{\mathbf{x}_0}^2 I\big).$$

Then Eq. (20) is equivalent, up to constants and a positive scale, to regression of the posterior mean:

$$\mathcal{L}_{\mathrm{VFM}}(\phi) \equiv \mathbb{E}\left[\left\|\mu_\phi(\mathbf{x}_t, t) - \mathbf{x}_0\right\|^2\right]. \tag{21}$$

Since Eq. (17) is affine in $\mathbf{x}_0$, posterior-mean regression can be written as a time-weighted velocity regression with the induced vector field

$$v_{\phi,t}(\mathbf{x}_t) = \frac{\partial_t f(t)}{1 - f(t)}\big(\mu_\phi(\mathbf{x}_t, t) - \mathbf{x}_t\big), \tag{22}$$

Indeed,

$$\mathcal{L}_{\mathrm{VFM}}(\phi) \equiv \mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_1}\left[\left(\frac{1 - f(t)}{\partial_t f(t)}\right)^2 \left\|v_{\phi,t}(\mathbf{x}_t) - v_t(\mathbf{x}_t \mid \mathbf{x}_0)\right\|^2\right]. \tag{23}$$

Thus, Gaussian VFM and standard flow matching use the same conditional velocity target, differing only by a time-dependent weighting of the regression loss. Under the linear schedule $f(t) = 1 - t$, this recovers the simulation-free flow-matching objective used by SCFM. The distinction is that SCFM applies this posterior matching to the structured source endpoint $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$ rather than to an unstructured noise variable.

B SCFM: Extended Derivations and Implementation Details

This appendix complements Section 3 with extended derivations and implementation details. Specifically, it gives the endpoint derivation, the endpoint consistency argument, the intermediate-time specialization to VFM, and the practical SCFM objective.

Shared posterior-matching view. We extend the posterior-matching construction (see Appendix A), such that the source endpoint becomes structured as $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$. The remainder of this appendix focuses on the two SCFM-specific regimes: the endpoint derivation at $t = 1$ and the intermediate-time specialization for $t < 1$.

B.1 Endpoint regime: posterior identity and endpoint objective

At $t = 1$, the interpolant endpoint is $\mathbf{x}_t = \mathbf{x}_1$. Under the encoder-induced coupling in Eq. (6), the source posterior becomes

$$\Gamma_1(\mathbf{x}_0 \mid \mathbf{x}_1) = q_\phi(\mathbf{z} \mid \mathbf{x}_1)\, p(\varepsilon), \qquad \mathbf{x}_0 = (\mathbf{z}, \varepsilon).$$

Using the SCFM decoder factorization

$$p_{\theta,\psi}(\mathbf{x}_1, \mathbf{x}_0) = p_\theta(\mathbf{x}_1 \mid \mathbf{z})\, p_\psi(\mathbf{z})\, p(\varepsilon),$$

the shared posterior mean splits as

$$\mu_\phi(\mathbf{x}_1, 1) = \big(\mu^z_\phi(\mathbf{x}_1), \mu^\varepsilon_\phi(\mathbf{x}_1)\big),$$

with latent posterior

$$q_\phi(\mathbf{z} \mid \mathbf{x}_1) = \mathcal{N}\big(\mathbf{z};\, \mu^z_\phi(\mathbf{x}_1), \mathrm{diag}(\sigma^2_\phi(\mathbf{x}_1))\big).$$

The endpoint mean is recovered from the shared velocity parameterization through

$$\mu_\phi(\mathbf{x}_1, 1) = \mathbf{x}_1 + \frac{1}{\partial_t f(1)}\, v_{\phi,1}(\mathbf{x}_1). \tag{24}$$

For compactness, write

$$q_x(\mathbf{z}) := q_\phi(\mathbf{z} \mid \mathbf{x}_1), \qquad q^\varepsilon_x(\varepsilon) := \mathcal{N}\big(\varepsilon;\, \mu^\varepsilon_\phi(\mathbf{x}_1), I_{d_\varepsilon}\big).$$

Because the decoder depends only on $\mathbf{z}$, $p_\theta(\mathbf{x}_1 \mid \mathbf{x}_0) = p_\theta(\mathbf{x}_1 \mid \mathbf{z})$. The endpoint KL then decomposes as

$$\begin{aligned}
\mathrm{KL}\big(q_x\, q^\varepsilon_x \,\|\, p_\psi(\mathbf{z})\, p(\varepsilon)\big) &= \mathrm{KL}\big(q_x \,\|\, p_\psi(\mathbf{z})\big) \\
&\quad + \mathrm{KL}\big(q^\varepsilon_x \,\|\, p(\varepsilon)\big) \\
&= \mathrm{KL}\big(q_x \,\|\, p_\psi(\mathbf{z})\big) + \frac{1}{2}\|\mu^\varepsilon_\phi(\mathbf{x}_1)\|^2.
\end{aligned} \tag{25}$$

Likewise,

$$\mathbb{E}_{q_x q^\varepsilon_x}\big[\log p_\theta(\mathbf{x}_1 \mid \mathbf{x}_0)\big] = \mathbb{E}_{q_x}\big[\log p_\theta(\mathbf{x}_1 \mid \mathbf{z})\big].$$

Therefore the endpoint contribution is

$$\begin{aligned}
\mathcal{L}_{\mathrm{end}} &= \mathbb{E}_{p_{\mathrm{data}}(\mathbf{x}_1)}\Big[- \mathbb{E}_{q_x} \log p_\theta(\mathbf{x}_1 \mid \mathbf{z}) + \mathrm{KL}\big(q_x \,\|\, p_\psi(\mathbf{z})\big) + \frac{1}{2}\|\mu^\varepsilon_\phi(\mathbf{x}_1)\|^2\Big] \\
&= \mathcal{L}_{\mathrm{VAE}} + \mathcal{R}_\varepsilon.
\end{aligned} \tag{26}$$

This is exactly the endpoint decomposition stated in Section 3, with $\mathcal{R}_\varepsilon$ matching the quadratic penalty in Eq. (15).

B.2 Coupling consistency and prior matching

For compactness, define the encoder-induced and decoder-induced latent joints

$$\Gamma^{\mathrm{enc}}_\phi(\mathbf{x}_1, \mathbf{z}) = p_{\mathrm{data}}(\mathbf{x}_1)\, q_\phi(\mathbf{z} \mid \mathbf{x}_1)$$

$$\Gamma^{\mathrm{dec}}_{\theta,\psi}(\mathbf{x}_1, \mathbf{z}) = p_\psi(\mathbf{z})\, p_\theta(\mathbf{x}_1 \mid \mathbf{z}).$$

Then

$$\mathrm{KL}\big(\Gamma_\phi^{\mathrm{enc}} \,\|\, \Gamma_{\theta,\psi}^{\mathrm{dec}}\big) = \mathcal{L}_{\mathrm{VAE}} + \mathrm{const},$$

where the constant depends only on p_{data}. As in Section 3.1, the exact identity

$$q_\phi^{\mathrm{agg}}(\mathbf{z}) = p_\psi(\mathbf{z})$$

holds if the joint KL is zero [Hoffman and Johnson, 2016, Alemi et al., 2018]. More generally, marginalization bounds the prior–aggregated-posterior KL by the joint KL for the standard likelihood-based VAE objective. Let $T(\mathbf{x}_1, \mathbf{z}) = \mathbf{z}$ be the projection onto the latent coordinate. Then

$$T_\# \Gamma_\phi^{\mathrm{enc}} = q_\phi^{\mathrm{agg}}(\mathbf{z}), \qquad T_\# \Gamma_{\theta,\psi}^{\mathrm{dec}} = p_\psi(\mathbf{z}),$$

so the KL data-processing inequality under marginalization gives

$$\mathrm{KL}\big(q_\phi^{\mathrm{agg}}(\mathbf{z}) \,\|\, p_\psi(\mathbf{z})\big) \le \mathrm{KL}\big(\Gamma_\phi^{\mathrm{enc}} \,\|\, \Gamma_{\theta,\psi}^{\mathrm{dec}}\big) = \mathcal{L}_{\mathrm{VAE}} + \mathrm{const}. \tag{27}$$

Consequently, reducing the endpoint VAE objective also reduces an upper bound on the mismatch between the encoder-induced training marginal and the sampling prior. If this joint KL is zero, then

$$\Gamma_\phi^{\mathrm{enc}}(\mathbf{x}_1, \mathbf{z}) = \Gamma_{\theta,\psi}^{\mathrm{dec}}(\mathbf{x}_1, \mathbf{z}) \qquad \text{a.e.}$$

and marginalizing over $\mathbf{x}_1$ yields

$$q_\phi^{\mathrm{agg}}(\mathbf{z}) := \int p_{\text{data}}(\mathbf{x}_1)\, q_\phi(\mathbf{z} \mid \mathbf{x}_1)\, d\mathbf{x}_1 = p_\psi(\mathbf{z}).$$

This is the endpoint consistency statement used in the main text: exact endpoint alignment recovers the same source marginal that is used at sampling time.

B.3 Intermediate-time regime: specialization to VFM

For $t \in [0, 1)$, SCFM reduces to the same posterior-matching problem analyzed in Appendix A, now with structured endpoint variable $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$. We use the shared Gaussian family

$$q_{t,\phi}(\mathbf{x}_0 \mid \mathbf{x}_t) = \mathcal{N}\big(\mu_\phi(\mathbf{x}_t, t), \sigma_{\mathbf{x}_0}^2 I_D\big), \tag{28}$$

whose mean is shared with the endpoint encoder, while the endpoint covariance of $q_\phi(\mathbf{z} \mid \mathbf{x}_1)$ is predicted by a separate variance head. Under the linear interpolant,

$$\mathbf{x}_t = f(t)\mathbf{x}_0 + (1 - f(t))\mathbf{x}_1,$$

the conditional velocity is

$$v_t(\mathbf{x}_t \mid \mathbf{x}_0) = \frac{\partial_t f(t)}{1 - f(t)}\big(\mathbf{x}_0 - \mathbf{x}_t\big). \tag{29}$$

The corresponding variational objective is

$$\mathcal{L}_{\mathrm{VFM}}(\phi) = \mathbb{E}_{t\sim\mathcal{U}[0,1),\, \mathbf{x}_t\sim p_t}\Big[\mathrm{KL}\big(\Gamma_t(\cdot \mid \mathbf{x}_t) \,\|\, q_{t,\phi}(\cdot \mid \mathbf{x}_t)\big)\Big]. \tag{30}$$

With the coupling held fixed during each update of $q_{t,\phi}$, the loss has the stop-gradient form

$$\mathcal{L}_{\mathrm{VFM}}(\phi) \equiv -\,\mathbb{E}_{t,(\mathbf{x}_t,\mathbf{x}_0)\sim\Gamma_t}\Big[\log q_{t,\phi}\big(\mathrm{sg}(\mathbf{x}_0) \mid \mathrm{sg}(\mathbf{x}_t)\big)\Big] + \mathrm{const.}, \tag{31}$$

where, in practice, $\mathbf{x}_1 \sim p_{\mathrm{data}}$, $\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x}_1)$, $\varepsilon \sim p(\varepsilon)$, $\mathbf{x}_0 = (\mathrm{sg}(\mathbf{z}), \varepsilon)$, and $\mathbf{x}_t = f(t)\mathbf{x}_0 + (1 - f(t))\mathbf{x}_1$. Under the Gaussian family in Eq. (28), Appendix A implies that minimizing Eq. (30) is equivalent to posterior-mean regression and therefore to the time-weighted velocity regression

$$\mathcal{L}_{\mathrm{VFM}}(\phi) \equiv \mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_1}\left[\left(\frac{1 - f(t)}{\partial_t f(t)}\right)^2 \|v_{\phi,t}(\mathbf{x}_t) - v_t(\mathbf{x}_t \mid \mathbf{x}_0)\|^2\right],$$
$$\mathbf{x}_t = f(t)\mathbf{x}_0 + (1 - f(t))\mathbf{x}_1. \tag{32}$$

B.4 Practical SCFM Objective and Prior Parameterization

The practical SCFM objective combines the intermediate-time VFM loss with an endpoint latent-variable objective:

$$\mathcal{L}_{\mathrm{SCFM}} = \mathcal{L}_{\mathrm{VFM}} + \mathcal{L}_{\mathrm{end}}^{\mathrm{train}}.$$

The endpoint term is implemented as

$$\mathcal{L}_{\mathrm{end}}^{\mathrm{train}} = \mathcal{L}_{\mathrm{rec}} + \mathcal{R}_z + \mathcal{R}_\varepsilon, \qquad \mathcal{L}_{\mathrm{rec}} = \mathcal{L}_{\mathrm{pix}} + \alpha_{\mathrm{perc}}\mathcal{L}_{\mathrm{perc}}.$$

The perceptual term can be instantiated with LPIPS [Zhang et al., 2018], following the use of perceptual reconstruction losses in image generation [Johnson et al., 2016]. The latent regularizer $\mathcal{R}_z$ is instantiated as either a β-VAE penalty [Higgins et al., 2017],

$$\mathcal{R}_z^{\beta\text{-VAE}} = \beta\,\mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}_1) \,\|\, p_\psi(\mathbf{z})\big),$$

or a β-TCVAE-style regularizer [Chen et al., 2018b],

$$\mathcal{R}_z^{\beta\text{-TCVAE}} = \mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}_1) \,\|\, p_\psi(\mathbf{z})\big) + \beta\,\mathrm{TC}\big(q_\phi^{\mathrm{agg}}(\mathbf{z})\big),$$

where $q_\phi^{\mathrm{agg}}(\mathbf{z}) = \int p_{\mathrm{data}}(\mathbf{x}_1) q_\phi(\mathbf{z} \mid \mathbf{x})\, d\mathbf{x}$ and $\mathrm{TC}(q) = \mathrm{KL}\big(q(\mathbf{z}) \,\|\, \prod_j q(z_j)\big)$. Thus, β-VAE and β-TCVAE provide two endpoint regularizers for structured latent learning under the same prior. These choices affect only the endpoint term and leave $\mathcal{L}_{\mathrm{VFM}}$ unchanged. The joint-KL identity above applies to the standard likelihood ELBO. The practical variants change this objective and encourage prior matching without guaranteeing exact equality.

GMM prior. We parameterize the structured prior as a learnable Gaussian mixture,

$$p_\psi(\mathbf{z}) = \sum_{k=1}^{K} \pi_k\, \mathcal{N}\big(\mathbf{z}; \mu_{\psi,k}, \mathrm{diag}(\sigma_{\psi,k}^2)\big), \qquad \sum_{k=1}^{K} \pi_k = 1.$$

The KL term is estimated by sampling $\mathbf{z} = \mu_\phi^z(\mathbf{x}) + \sigma_\phi(\mathbf{x}) \odot \boldsymbol{\xi}$, with $\boldsymbol{\xi} \sim \mathcal{N}(0, I)$, and evaluating $\mathrm{KL}(q_\phi(\mathbf{z} \mid \mathbf{x}_1) \| p_\psi(\mathbf{z})) = \mathbb{E}_{q_\phi}[\log q_\phi(\mathbf{z} \mid \mathbf{x}_1) - \log p_\psi(\mathbf{z})]$. At sampling time, we draw $\mathbf{z} \sim p_\psi(\mathbf{z})$ and $\varepsilon \sim \mathcal{N}(0, I)$, forming the structured source $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$.

C SCFM Algorithms

This section collects compact pseudocode summaries of the practical SCFM procedures described in Sections 3.2 and 3.3.

C.1 Training

Algorithm 1 summarizes one SCFM training step, including the endpoint latent-variable terms and the intermediate-time flow-matching update.

C.2 Sampling

Algorithm 2 gives the full ODE sampler, while Algorithm 3 summarizes the decoder-initialized refinement sampler that initializes near the data manifold using the decoder.

D Extended Related Work

Structured priors and couplings. Several lines of work mitigate limitations of fixed, unstructured priors in diffusion- and transport-based models. PriorGrad [Sang-gil et al., 2022] learns a data-dependent Gaussian prior for conditional diffusion, while DecompDiff [Guan et al., 2023] uses decomposed priors for structured molecule generation. Other approaches leave the prior fixed but change the pairing between source and data samples, for example through minibatch optimal transport [Tong et al., 2024], data-dependent couplings in stochastic interpolants [Albergo et al., 2024], or post-hoc correction of prior mismatch with optimal transport maps [Wang et al., 2024]. These methods improve the transport problem or sampling geometry, but generally do not make the source itself a learned representation space.

Algorithm 1 One SCFM training step

Require: minibatch $\mathbf{x}_1 \sim p_{\text{data}}$, velocity model $v_{\phi,t}$, endpoint variance head $\sigma_\phi^2(\mathbf{x}_1)$, decoder $p_\theta(\mathbf{x}_1 \mid \mathbf{z})$, latent prior $p_\psi(\mathbf{z})$

Endpoint VAE regime:
  Compute $\mu_\phi(\mathbf{x}_1, 1) = \big(\mu_\phi^z(\mathbf{x}_1), \mu_\phi^\varepsilon(\mathbf{x}_1)\big)$
  Form $q_\phi(\mathbf{z} \mid \mathbf{x}_1) = \mathcal{N}\big(\mu_\phi^z(\mathbf{x}_1), \operatorname{diag}(\sigma_\phi^2(\mathbf{x}_1))\big)$
  Sample $\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x}_1)$ and compute $\mathcal{L}_{\text{rec}}$ from $p_\theta(\mathbf{x}_1 \mid \mathbf{z})$
  Compute $\mathcal{L}_{\text{KL}} = \text{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}_1) \,\|\, p_\psi(\mathbf{z})\big)$
  Compute $\mathcal{R}_\varepsilon = \frac{1}{2}\|\mu_\phi^\varepsilon(\mathbf{x}_1)\|^2$
Flow-matching regime:
  Sample $\varepsilon \sim p(\varepsilon) = \mathcal{N}(0, I_{d_\varepsilon})$ and set $\mathbf{x}_0 = \big(\text{sg}(\mathbf{z}), \varepsilon\big)$
  Sample $t \sim \mathcal{U}[0,1)$ and form $\mathbf{x}_t = f(t)\mathbf{x}_0 + (1 - f(t))\mathbf{x}_1$ with $f(t) = 1 - t$
  Compute $\mathcal{L}_{\text{VFM}} = \Big(\frac{1-f(t)}{\partial_t f(t)}\Big)^2 \Big\| v_{\phi,t}(\mathbf{x}_t) - \frac{\partial_t f(t)}{1-f(t)}\big(\mathbf{x}_0 - \mathbf{x}_t\big)\Big\|^2$
Form $\mathcal{L}_{\text{SCFM}} = \mathcal{L}_{\text{VFM}} + \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{KL}} + \mathcal{R}_\varepsilon$
Update (θ, ϕ, ψ) using $\mathcal{L}_{\text{SCFM}}$

Algorithm 2 Full ODE SCFM sampling / reconstruction

Require: velocity model $v_{\phi,t}$, endpoint variance head $\sigma_\phi^2(\mathbf{x}_1)$, latent prior $p_\psi(\mathbf{z})$, optional observation $\mathbf{x}_1$

if an observation $\mathbf{x}_1$ is available **then**
  Compute $\mu_\phi(\mathbf{x}_1, 1) = \big(\mu_\phi^z(\mathbf{x}_1), \mu_\phi^\varepsilon(\mathbf{x}_1)\big)$
  Form $q_\phi(\mathbf{z} \mid \mathbf{x}_1) = \mathcal{N}\big(\mu_\phi^z(\mathbf{x}_1), \operatorname{diag}(\sigma_\phi^2(\mathbf{x}_1))\big)$
  Sample $\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x}_1)$
else
  Sample $\mathbf{z} \sim p_\psi(\mathbf{z})$
end if
Sample $\varepsilon \sim p(\varepsilon) = \mathcal{N}(0, I_{d_\varepsilon})$ and set $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$
Integrate the ODE with velocity field $v_{\phi,t}(\mathbf{x}_t)$ on $t \in [0,1]$ with initial condition $\mathbf{x}_{t=0} = \mathbf{x}_0$
return $\mathbf{x}_{t=1}$

Algorithm 3 Decoder-initialized refinement SCFM sampling / reconstruction

Require: start time $t_0 \in (0,1)$, velocity model $v_{\phi,t}$, endpoint variance head $\sigma_\phi^2(\mathbf{x}_1)$, decoder $p_\theta(\mathbf{x}_1 \mid \mathbf{z})$, latent prior $p_\psi(\mathbf{z})$, optional observation $\mathbf{x}_1$

if an observation $\mathbf{x}_1$ is available **then**
  Compute $\mu_\phi(\mathbf{x}_1, 1) = \big(\mu_\phi^z(\mathbf{x}_1), \mu_\phi^\varepsilon(\mathbf{x}_1)\big)$
  Form $q_\phi(\mathbf{z} \mid \mathbf{x}_1) = \mathcal{N}\big(\mu_\phi^z(\mathbf{x}_1), \operatorname{diag}(\sigma_\phi^2(\mathbf{x}_1))\big)$
  Sample $\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x}_1)$
else
  Sample $\mathbf{z} \sim p_\psi(\mathbf{z})$
end if
Sample $\varepsilon \sim p(\varepsilon) = \mathcal{N}(0, I_{d_\varepsilon})$ and set $\mathbf{x}_0 = (\mathbf{z}, \varepsilon)$
Sample $\widehat{\mathbf{x}}_1 \sim p_\theta(\cdot \mid \mathbf{z})$
Form $\mathbf{x}_{t_0} = f(t_0)\mathbf{x}_0 + (1 - f(t_0))\widehat{\mathbf{x}}_1$ with $f(t) = 1 - t$
Integrate the ODE with velocity field $v_{\phi,t}(\mathbf{x}_t)$ on $t \in [t_0, 1]$ with initial condition $\mathbf{x}_{t=t_0} = \mathbf{x}_{t_0}$
return $\mathbf{x}_{t=1}$

Variational coupling. Variational Consistency Training [Silvestri et al., 2025] is closest to our use of an encoder for coupling: it learns a data-noise coupling with a VAE-style encoder and a KL penalty that keeps emitted noise close to the sampling prior, reducing variance in consistency training. SCFM uses a related variational mechanism for a different purpose. Rather than learning a coupling only to stabilize a consistency objective, we use the encoder to define a structured source posterior whose latent component is trained as a representation and whose marginal prior is learned jointly with the flow.

Flow matching for representations. Latent-variable formulations of flow and rectified-flow matching model transport ambiguity with hidden variables [Guo and Schwing, 2025, Zhang et al., 2025]. Structured Flow Autoencoders [Xu et al., 2026] are especially close in motivation: they use posterior-averaged conditional vector fields to combine flow matching with structured probabilistic representations. Recent concurrent work, FlowFM [Ukita and Okita, 2026], jointly trains a representation encoder and a conditional flow-matching generator for self-supervised learning, emphasizing downstream recognition performance and computational efficiency on wearable-sensor data. Recent work on disentangled representation learning via flow matching [Chi et al., 2026] learns factor-conditioned latent-space flows and regularizes factor-specific velocity components to reduce cross-factor leakage. SCFM differs from these lines by learning a structured *source distribution* for a stochastic interpolant: the latent variable is part of the source endpoint being transported, not only a conditioning variable for a generator or a factorization of the velocity field.

Latent-variable models. Variational autoencoders and their extensions learn latent representations through amortized inference and variational training [Kingma and Welling, 2014, Burda et al., 2015]. This family includes structured-prior models for clustering, such as Gaussian-mixture and multi-facet variants [Dilokthanakul et al., 2016, Jiang et al., 2017, Falck et al., 2021], as well as regularized objectives that encourage disentangled latent factors [Higgins et al., 2017, Burgess et al., 2018, Chen et al., 2018b, Locatello et al., 2019]. These methods show that explicit latent variables are effective for representation learning, but their sample quality is often limited by the decoder family and by the fact that generation is tied directly to the latent-variable model. SCFM inherits the representation-learning advantages of this line of work, but differs in how generation is performed: the latent model is used to learn a structured source and endpoint supervision, while the main sampler remains a flow that transports samples from the learned source prior to the data distribution.

E Experimental Details

This appendix provides the implementation details needed to reproduce the experiments in Section 4. We describe the dataset-specific architectures, training schedules, latent dimensionalities, endpoint objectives, priors, samplers, and probing protocols used for MNIST, CIFAR-10, Cars3D, Shapes3D, and ImageNet-128. Unless otherwise stated, all models are trained without using class or factor labels; labels and ground-truth factors are used only for downstream probing or disentanglement evaluation.

Experiments were run on a heterogeneous pool of GPUs: 2 NVIDIA RTX PRO 6000 Blackwell Server Edition GPUs, 2 NVIDIA A100 80GB PCIe GPUs, 2 NVIDIA RTX A6000 GPUs, and 3 Quadro RTX 6000 GPUs.

The original CIFAR-10 and ImageNet-128 flow-matching baselines retain SCFM's flow backbone and training configuration with the structured endpoint branch removed. The additional controls allocate approximately SCFM's full parameter budget to vanilla flow matching and match optimization steps, while retaining different training costs.

E.1 Parameter- and Step-Matched Flow Controls

The matched baselines devote all trainable parameters to vanilla flow matching, omitting the latent head, learnable GMM prior, auxiliary decoder, and VAE-style loss. They follow the dataset training schedules for approximately 508K steps on CIFAR-10 and 400K on ImageNet-128. Table 5 reports parameter counts and forward FLOPs; Table 3 reports generation quality.

These controls compare two capacity allocations at similar total parameter budgets: vanilla flow modeling and joint flow/representation learning. Intermediate capacity splits remain unexplored.

Table 5: **Capacity-control setup.** FLOPs are for one batch-size-one forward pass. The controls match parameters and optimization steps, while SCFM retains greater forward cost.

Dataset	Model	Parameters	FLOPs / forward	Steps
CIFAR-10	Original FM	73.6M	40.4G	≈ 508K
	Matched FM	104M	62.0G	≈ 508K
	SCFM	101M	95.4G	≈ 508K
ImageNet-128	Original FM	675M	58.1G	400K
	Matched FM	1.1B	94.6G	400K
	SCFM	1.1B	153G	400K

Vanilla FM has no endpoint encoder for frozen-$\mathbf{z}$ probing; evaluating its internal features requires a separate protocol.

E.2 Independent Endpoint Model and Two-Stage Flow

For the two-stage control in Section 4.4, we first train an independent β-VAE with SCFM's encoder/decoder architecture, latent dimension, endpoint objective, and learned-GMM prior. After freezing this checkpoint, we train a randomly initialized data-space flow-matching model directly from its structured source. The frozen decoder is evaluated separately as a generator. The mixture-component count, flow backbone, flow-training steps, sampler/NFE, and FID50K pipeline, including sample count and preprocessing, match SCFM.

Table 6: **CIFAR-10 independent and joint pipeline resources.** Parameter counts distinguish trainable and frozen components during the indicated training stage. Counts are rounded estimates; the displayed 73.6M flow and 101M frozen endpoint model sum to 174.6M. FLOPs are reported per training forward pass, not as complete training-run costs.

Training stage	Parameters			FLOPs / forward
	Trainable	Frozen	Total	
Independent β-VAE	101M	0	101M	56.2G
Flow with frozen β-VAE	73.6M	101M	174.6M	81.2G
Joint SCFM	101M	0	101M	95.4G

The independent encoder's CIFAR-10 linear and nonlinear probe accuracies, 46.28% and 61.28%, come from one endpoint checkpoint and remain fixed during flow training. SCFM's 50.26% and 66.58% are averages over five independently trained models (Table 17). These probe summaries differ from the fixed-checkpoint generation evaluations in Table 4. The independent decoder and flow FIDs, 27.646 and 15.733, are single FID50K evaluations; Appendix G.1 reports repeated sampling from one SCFM checkpoint.

E.3 MNIST Clustering Setup

The main MNIST comparison uses $K = 10$ mixture components across methods and reports means and standard deviations over five independent training runs. The component-count sweep in Appendix G.4.1 uses one seed per setting, with all other settings fixed. SCFM assigns each image to a component using one posterior draw and the learned GMM responsibilities (Appendix F).

E.4 CIFAR-10 Setup

Table 8 summarizes the encoder, decoder, training, and sampling settings used for the CIFAR-10 SCFM experiments.

Table 7: **MNIST clustering settings.**

	VaDE	**MFCVAE**	**SCFM (β-VAE)**	**SCFM (β-TCVAE)**
Latent	$z \in \mathbb{R}^{10}$	J=2, z_j=5	$z \in \mathbb{R}^{\{10,128,256\}}$	$z \in \mathbb{R}^{\{10,128,256\}}$
Components	K=10	K_j=10	K=10	K=10
Objective	β=1 + GMM	MFCVAE + diag. $p(z_j \mid c_j)$	GMM + β=$\{1,\ldots,4\}$	GMM + β=$\{1,\ldots,3\}$ (TC)
Batch size	256	256	256	256
Optimizer	AdamW (10^{-3})	Adam (5×10^{-4})	AdamW (10^{-3})	AdamW (10^{-3})

Table 8: **SCFM CIFAR-10 network and training setup.** The encoder parameterizes the shared posterior/recognition network, while the decoder is used for the endpoint reconstruction objective and decoder-initialized refinement.

Hyperparameter	**Encoder / recognition net**	**Decoder**
Role	Shared $q_{t,\phi}(x_0 \mid x_t)$ and endpoint encoder $q_\phi(z \mid x)$	Endpoint decoder $p_\theta(x \mid z)$
Backbone	U-Net	U-Net
Channels	128	128
Depth	4	4
Channel multipliers	$(2, 2, 2)$	$(2, 2, 2)$
Heads	1	–
Head channels	-1	–
Attention resolution	[2]	–
Dropout	0.3	–
Latent dimension	$\{64, 128, 256, 512\}$	
Training length	1300 epochs, $\approx$ 508K iterations	
Optimizer	Adam, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$	
Learning rate	5×10^{-4} with polynomial decay	
Warmup	45K steps, linear warmup from 10^{-8}	
Structured prior	Learnable GMM prior $p_\psi(z)$ with $K = 10$ components	
Reconstruction loss	Pixel reconstruction with LPIPS augmentation	
Endpoint objective	β-VAE with $\beta = \{0.5, 2.8, 4\}$	
EMA	Enabled, decay 0.9999	
Sampler	Heun ODE sampler	

E.5 Cars3D and Shapes3D Disentanglement Setup

Table 9 summarizes the dataset, training. We follow the standard disentanglement benchmark protocol of Locatello et al. [2019]: models are trained without access to ground-truth factor labels, while factor annotations are used only for evaluation. For each dataset and endpoint regularizer, we train models across 10 random seeds and sweep over the corresponding β values for the VAE-style objective. All models use convolutional encoder–decoder networks and a 10-dimensional latent representation.

E.6 ImageNet-128 Setup

Table 10 summarizes the ImageNet-128 latent SCFM configuration. We build on a pretrained Stable-Diffusion VAE Rombach et al. [2022], using `stabilityai/sd-vae-ft-ema` to map 128×128 images into a 4-channel latent image space. The SCFM encoder operates on these VAE latents using a SiT-XL/2 transformer backbone, while the decoder also uses a transformer architecture to map the learned 64-dimensional structured latent representation back to the VAE latent image space. The decoder architecture is depicted in Figure 7. The prior over $\mathbf{z}$ is modeled as a learnable GMM with 100 components. The model is trained with a β-VAE endpoint objective and an ℓ_1 reconstruction loss, and samples are generated with the Dormand–Prince ODE solver.

Table 9: **Cars3D and Shapes3D disentanglement setup.** Models are trained without factor labels; ground-truth factors are used only for disentanglement evaluation.

	Cars3D	**Shapes3D**
Resolution	$64\times64\times3$	$64\times64\times3$
Images	17,568	480,000
Factors	3	6
Cardinalities	$183\times4\times24$	$10\times10\times10\times8\times4\times15$
Factor names	car ID, elevation, azimuth	floor hue, wall hue, object hue, scale, shape, orientation
Train labels	None	None
Eval factors	Used for metrics only	Used for metrics only
Latent dimension	$\{10, 32, 64\}$	$\{10, 32, 64\}$
Representation	$\mu_\phi(x)$	$\mu_\phi(x)$
β-VAE sweep	$\{1, 2, 4, 6, 8, 16\}$	$\{1, 2, 4, 6, 8, 16\}$
β-TCVAE sweep	$\{1, 2, 4, 6, 8, 10\}$	$\{1, 2, 4, 6, 8, 10\}$
Training	Adam, learning rate 10^{-4}, batch size 64, 300K steps	

Table 10: **ImageNet-128 SiT-XL/2 latent model setup.** The encoder parameterizes the latent recognition pathway, while the decoder maps latent representations back to the VAE latent image space. Shared optimization and sampling settings are merged across both columns.

Hyperparameter	**Encoder / recognition net**	**Decoder**
Role	Latent encoder	Latent decoder
Backbone	SiT-XL/2	SiT-like decoder
Input	VAE latent image	$\mathbf{z} \in \mathbb{R}^{64}$
Patch size	2×2	–
In/out channels	4 input channels	4 output channels
Hidden size	1152	1152
Depth	28	14
Attention heads	16	16
MLP ratio	4.0	4.0
Dropout	Class dropout 0.1	Dropout 0.0
Conditioning	AdaLN-Zero time and null-token conditioning	AdaLN on $\mathbf{z}$
Latent dimension	$\{64, 128, 256, 512\}$	
VAE	`stabilityai/sd-vae-ft-ema`, scaling factor 0.18215	
Prior	Learnable GMM prior with 100 components	
Training steps	400K	
Batch size	256	
Optimizer	AdamW, $\beta_1 = 0.9$, $\beta_2 = 0.95$	
Learning rate	1×10^{-4} with linear decay	
Reconstruction loss	ℓ_1 reconstruction loss	
Endpoint objective	β-VAE with $\beta = \{0.5, 2.8, 4\}$	
Gradient clipping	Maximum gradient norm 5.0	
EMA	Enabled, decay 0.9999	
Sampler	Dormand–Prince method of order 5 (Dopri5)	

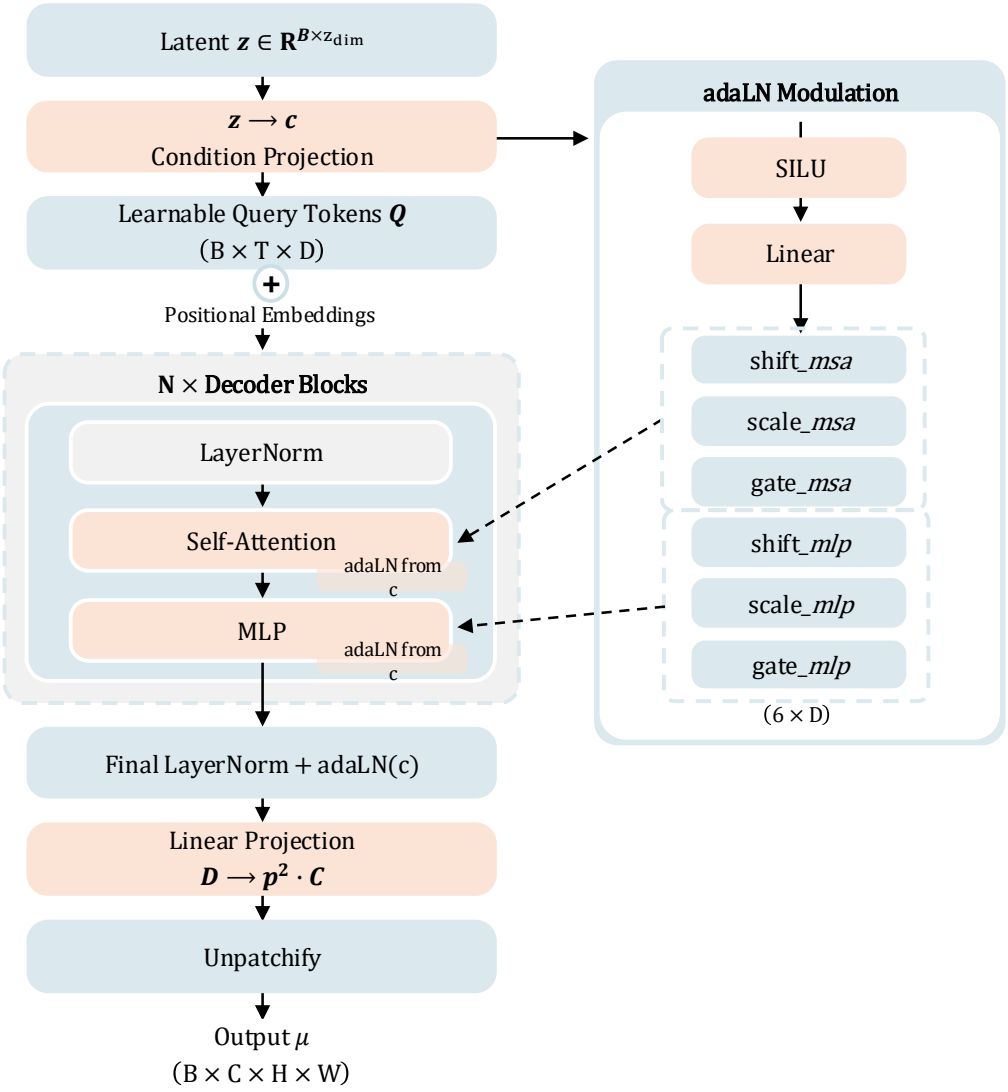

Figure 7: **SiT-like decoder architecture used in the ImageNet-128 SCFM setup.**

E.7 CIFAR-10 and ImageNet-128 Probing Setup

Table 11 summarizes the probing protocols for CIFAR-10 and ImageNet-128. For CIFAR-10, we follow the latent probing setup of Zhang et al. [2022]. For ImageNet-128, we evaluate frozen latent representations with linear and nonlinear probes and compare SCFM against a frozen SD-VAE-FT-EMA encoder from Stable Diffusion [Rombach et al., 2022]. In both cases, probes are trained without data augmentation so that performance reflects information available in the frozen representation.

Table 11: **Probe settings for CIFAR-10 and ImageNet-128.** After training, frozen latent representations are evaluated with linear and nonlinear probes.

Component	**CIFAR-10**	**ImageNet-128**
Representation	$\mu_\phi(\mathbf{x})$	$\mu_\phi(\mathbf{x})$
Target	CIFAR-10 label	ImageNet label
Data augmentation	None	None
Model-selection split	Not specified	5% stratified holdout from ImageNet train
Evaluation split	Standard test split	ImageNet validation split
Linear probe	Linear SVM	Single linear layer on standardized features
Nonlinear probe	2-layer MLP hidden size 200, ReLU	Residual MLP hidden $(4096, 2048, 1024)$ GELU activations LayerNorm
Regularization	BatchNorm dropout 0.1	dropout 0.12–0.15 weight decay 10^{-4}–2×10^{-4} label smoothing 0.05–0.1
Reporting	Accuracy	Top-1 / Top-5 accuracy

F Evaluation Metrics

F.1 Clustering Metrics

We evaluate clustering performance using two standard metrics: clustering accuracy (ACC) and normalized mutual information (NMI).

SCFM cluster prediction. For each MNIST image $\mathbf{x}$, the endpoint encoder produces posterior mean $\boldsymbol{\mu}_\phi(\mathbf{x})$ and standard deviation $\boldsymbol{\sigma}_\phi(\mathbf{x})$. We draw one posterior sample

$$\mathbf{z} = \boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\sigma}_\phi(\mathbf{x}) \odot \boldsymbol{\eta}, \qquad \boldsymbol{\eta} \sim \mathcal{N}(0, I), \tag{33}$$

compute the learned GMM responsibilities $r_k(\mathbf{z}) = p_\psi(c = k \mid \mathbf{z})$, and assign $c(\mathbf{x}) = \arg\max_k r_k(\mathbf{z})$. Both ACC and NMI use this single cluster assignment per image. Digit labels are used only for evaluation.

Clustering accuracy (ACC). ACC measures the alignment between predicted cluster assignments and ground-truth class labels, accounting for the permutation ambiguity of cluster indices. Since cluster IDs are arbitrary, we first compute an optimal mapping π from cluster IDs to class labels, and then measure the fraction of correctly assigned samples:

$$\mathrm{ACC} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\{y_i = \pi(c_i)\}, \tag{34}$$

Here y_i is the ground-truth label and c_i the predicted cluster. For SCFM, we obtain π by one-to-one Hungarian matching of the component-by-digit contingency matrix, padding it with zeros to make it square when $K \neq 10$. ACC is the matched count divided by the total number of evaluated images. When $K < 10$, at most K digit labels can be matched, limiting attainable ACC. For MFCVAE, we report facet-wise ACC, assigning each cluster to its most frequent digit label.

Normalized mutual information (NMI). NMI measures the statistical dependence between predicted clusters and ground-truth labels, and is invariant to label permutations. It is defined based on the mutual information $I(Y;C)$ between labels Y and clusters C, normalized by their entropies. Using the arithmetic normalization adopted in our implementation:

$$\mathrm{NMI}(Y,C) = \frac{2\,I(Y;C)}{H(Y)+H(C)}. \tag{35}$$

NMI ranges from 0 to 1, where 0 indicates no mutual information and 1 corresponds to perfect agreement between clusters and labels.

F.2 Disentanglement Metrics

We evaluate disentanglement using the FactorVAE score [Kim and Mnih, 2018] and the disentanglement component of the DCI framework [Eastwood and Williams, 2018].

FactorVAE score. Let $r(x) \in \mathbb{R}^d$ denote the learned representation and let the data be generated by K independent factors. For each training example of the metric, we fix one ground-truth factor k, sample a batch B_k while varying all other factors, encode the batch, and normalize each latent coordinate by its empirical standard deviation over the dataset:

$$s_j^2 = \mathrm{Var}_{p_{\mathrm{data}}}[r_j(x)]\,.$$

We then compute the within-batch variance of each normalized latent coordinate,

$$\hat{v}_j(B_k) = \mathrm{Var}_{x\in B_k}\left[\frac{r_j(x)}{s_j}\right],$$

and predict the fixed factor using the index of the least-varying coordinate,

$$j^\star(B_k) = \arg\min_j \hat{v}_j(B_k).$$

Accumulating many such pairs $(j^\star(B_k), k)$ yields a majority-vote classifier from latent dimensions to factors. The FactorVAE score is the resulting classification accuracy on held-out batches. Higher is better.

DCI disentanglement. The DCI framework measures disentanglement, completeness, and informativeness from an importance matrix relating latent coordinates to ground-truth factors. Since Table 2 reports only the disentanglement component, we denote that score by DCI. Let $R \in \mathbb{R}_{\geq 0}^{L\times K}$ be a relative-importance matrix, where R_{ij} measures how much latent coordinate c_i contributes to predicting factor z_j, and each column is normalized so that $\sum_{i=1}^{L} R_{ij} = 1$. For each latent coordinate, define the normalized row distribution

$$P_{ij} = \frac{R_{ij}}{\sum_{k=1}^{K} R_{ik}}, \qquad D_i = 1 - H_K(P_{i\cdot}),$$

where

$$H_K(P_{i\cdot}) = -\sum_{j=1}^{K} P_{ij}\log_K P_{ij}.$$

The overall disentanglement score is the importance-weighted average

$$\mathrm{DCI} = \sum_{i=1}^{L} \rho_i D_i, \qquad \rho_i = \frac{1}{K}\sum_{j=1}^{K} R_{ij}.$$

This score is highest when each latent coordinate is primarily predictive of a single generative factor.

F.3 Generative Quality Metric

Fréchet inception distance (FID). To evaluate image quality and diversity, we use FID [Heusel et al., 2017]. Let (m_r, C_r) and (m_g, C_g) denote the empirical mean and covariance of Inception features extracted from real and generated samples, respectively. FID is defined as

$$\mathrm{FID} = \|m_r - m_g\|_2^2 + \mathrm{Tr}\left(C_r + C_g - 2(C_rC_g)^{1/2}\right).$$

Lower FID indicates that generated samples are closer to real data in the feature space, reflecting both fidelity and diversity.

F.4 Downstream Representation Metrics

Linear and nonlinear probing. To evaluate whether the learned latent representation retains class-relevant semantic information, we freeze the encoder and train supervised probes on top of the latent code. Let

$$\mathbf{z}_i = r_\phi(\mathbf{x}_i)$$

denote the frozen representation of image $\mathbf{x}_i$. A linear probe trains a single linear classifier

$$h(\mathbf{z}) = W\mathbf{z} + b$$

using labeled training examples, while keeping r_ϕ fixed. A nonlinear probe uses a small multilayer classifier on top of the same frozen representation.

Probe performance is measured by Top-k classification accuracy on the held-out test set:

$$\text{Top-}k = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\left\{y_i \in \mathrm{TopK}\big(h(\mathbf{z}_i), k\big)\right\},$$

where $\mathrm{TopK}(h(\mathbf{z}_i), k)$ denotes the set of k classes with the highest predicted scores. The usual classification accuracy corresponds to the special case $k = 1$:

$$\mathrm{Acc} = \text{Top-1} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\left\{\arg\max_c h_c(\mathbf{z}_i) = y_i\right\}.$$

For CIFAR-10, we report Top-1 accuracy. For ImageNet-128, we report both Top-1 and Top-5 accuracy

G Additional Results and Diagnostics

This section collects supplementary diagnostics for the representation-learning experiments in Section 4.1 and the image-generation experiments in Section 4.3.

G.1 FID Variability Across Sampling Seeds

FID varies with both training and sampling randomness [Dufour et al., 2026]. For CIFAR-10 matched FM and SCFM, we evaluate one fixed checkpoint per method using four independent sampling seeds, generating 50K images per seed with the sampler, NFE, preprocessing, and reference statistics held fixed. The means and sample standard deviations in Table 12 therefore measure sampling variability at these checkpoints. They do not measure variation across independently trained models or establish statistical superiority across training runs. The original CIFAR-10 FM and all ImageNet-128 FIDs in Table 3 are single evaluations.

Table 12: **CIFAR-10 FID50K from repeated sampling.** Four independent generated sets per fixed checkpoint. Mean and sample standard deviation summarize sampling-seed variability.

Fixed checkpoint	**Sampling-seed FIDs**				**Mean ± sample SD**
	1	2	3	4	
Matched FM (104M)	2.175	2.203	2.202	2.151	2.183 ± 0.025
SCFM (101M)	2.117	2.133	2.110	2.124	$\mathbf{2.121} \pm 0.010$

G.2 Decoder Initialization at Matched NFE

Figure 6 plots FID against cumulative sampling FLOPs for decoder-only generation, decoder-initialized refinement from $t_0 = 0.8$ at 1–48 NFE, and full-trajectory SCFM at 50 NFE. Table 13 compares the methods at 50 NFE, where full-trajectory SCFM achieves lower FID than refinement. Equal NFE does not equalize total FLOPs: refinement includes a decoder pass, and flow backbones may have different costs per evaluation.

Table 13: **CIFAR-10 generation at matched NFE.** Refinement is one FID50K evaluation; the full-flow SCFM and matched-FM values summarize four sampling seeds from fixed checkpoints (Table 12).

Sampling method	**NFE**	**FID50K** $\downarrow$
SCFM decoder-initialized refinement	50	3.103
SCFM full trajectory	50	$\mathbf{2.121} \pm 0.010$
Parameter-matched vanilla FM (104M)	50	2.183 ± 0.025

G.3 Independent Endpoint Disentanglement Controls

Table 14 compares the independently trained β-VAE endpoint models used in the two-stage controls with joint SCFM under the same endpoint regularizer. Independent entries are single-model estimates; SCFM entries are the ten-run means and standard deviations from Table 2. The independent encoder remains frozen during flow training, so its latent scores cannot measure whether the flow preserves interventions. Joint SCFM has higher Cars3D FactorVAE and Shapes3D DCI scores, while the independent model has higher Shapes3D FactorVAE; Cars3D DCI is comparable.

Table 14: **Independent versus joint endpoint representations.** Independent β-VAE entries are single-checkpoint estimates. SCFM (β-VAE) entries are mean $\pm$ standard deviation under the paper's ten-run protocol.

Dataset	**Representation**	**FactorVAE** $\uparrow$	**DCI** $\uparrow$
Cars3D	Independent β-VAE	0.8804	$\mathbf{0.3411}$
	Joint SCFM (β-VAE)	$\mathbf{0.940} \pm 0.019$	0.337 ± 0.082
Shapes3D	Independent β-VAE	$\mathbf{0.8806}$	0.6600
	Joint SCFM (β-VAE)	0.836 ± 0.034	$\mathbf{0.779} \pm 0.065$

G.4 MNIST Clustering and Latent Diagnostics

Latent clustering. Figure 8 compares the latent spaces learned by MFCVAE, VaDE, and the two SCFM variants. Columns correspond to different models. The top row colors points by ground-truth digit label, while the bottom row colors the same embeddings by predicted cluster assignment after cluster-to-label matching. The SCFM variants produce more compact and label-consistent clusters than the VAE-based baselines, with SCFM (β-VAE) achieving the best clustering performance in Table 15.

Table 15: **Clustering performance on MNIST** (mean $\pm$ std). Baseline methods are VaDE [Jiang et al., 2017] and MFCVAE [Falck et al., 2021].

Model	**NMI** $\uparrow$	**ACC** $\uparrow$
VaDE	79.93 ± 0.77	77.33 ± 0.33
MFCVAE	71.22 ± 0.68	85.43 ± 0.54
SCFM (β-VAE)	$\mathbf{87.84 \pm 0.86}$	$\mathbf{90.44 \pm 0.60}$
SCFM (β-TCVAE)	87.11 ± 0.47	89.52 ± 0.84

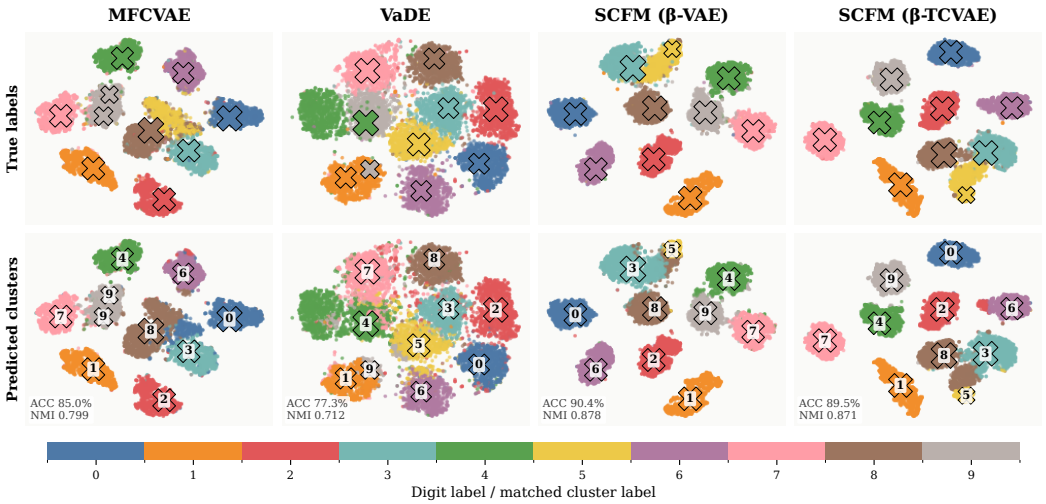

Figure 8: t-SNE visualization of the learned latent spaces on MNIST. Each column corresponds to a different model. The top row colors points by digit label, and the bottom row colors the same embeddings by matched cluster assignment.

G.4.1 Sensitivity to the Number of Mixture Components

Table 16 varies the number of learned GMM components on MNIST with all other settings fixed, using one seed per setting rather than the five-run summaries in Table 15. Clustering remains strong for $K = 10$–20, showing that digit-label alignment does not require one component per label. At $K = 5$, one-to-one Hungarian matching can match only five labels, limiting attainable ACC. At $K = 25$, digit-label alignment decreases; inspection found components grouping writing styles or shared strokes, including curved variants of 3 and 8. Digit-label ACC/NMI do not directly assess these appearance groupings.

Table 16: **MNIST component-count sensitivity.** Single-seed sweep with all other settings fixed. Metrics are on the $[0, 1]$ scale. Cluster prediction and matching follow Appendix F.

Components K	**NMI** ↑	**ACC** ↑
5	0.75216	0.52600
10	0.85748	0.88216
15	0.87751	0.89037
20	0.86581	0.89448
25	0.77964	0.79258

G.5 CIFAR-10 Component-Level Generation Diagnostics

Figure 9 analyzes how the learned GMM prior organizes CIFAR-10 samples and how decoder-initialized refinement affects generation quality. Each row corresponds to one learned component. For each component, we compare decoder-only samples from the learned latent prior, full-flow samples obtained by integrating from the structured source, and decoder-initialized refinement samples initialized from the decoder and refined by the flow. The components are not class-pure, which is expected for CIFAR-10 without label supervision. Instead, samples within a component often share coherent appearance statistics, such as color palette, background structure, object scale, or coarse geometry. This suggests that the learned prior captures appearance modes rather than directly recovering semantic classes.

The comparison across sampling modes highlights the cooperative role of the decoder and the flow. Decoder-only samples already capture coarse component-specific structure, while full-flow sampling improves realism and fine detail. Decoder-initialized refinement preserves the coarse structure proposed by the decoder while adding sharper edges and more realistic textures. In the last three columns, refinement starts from the decoder output at $t_0 = 0.8$ and uses only 5 NFE, compared with 50 NFE for full-flow sampling, yet remains visually close to the full-flow samples. This supports decoder-initialized refinement as a lower-cost sampling mode.

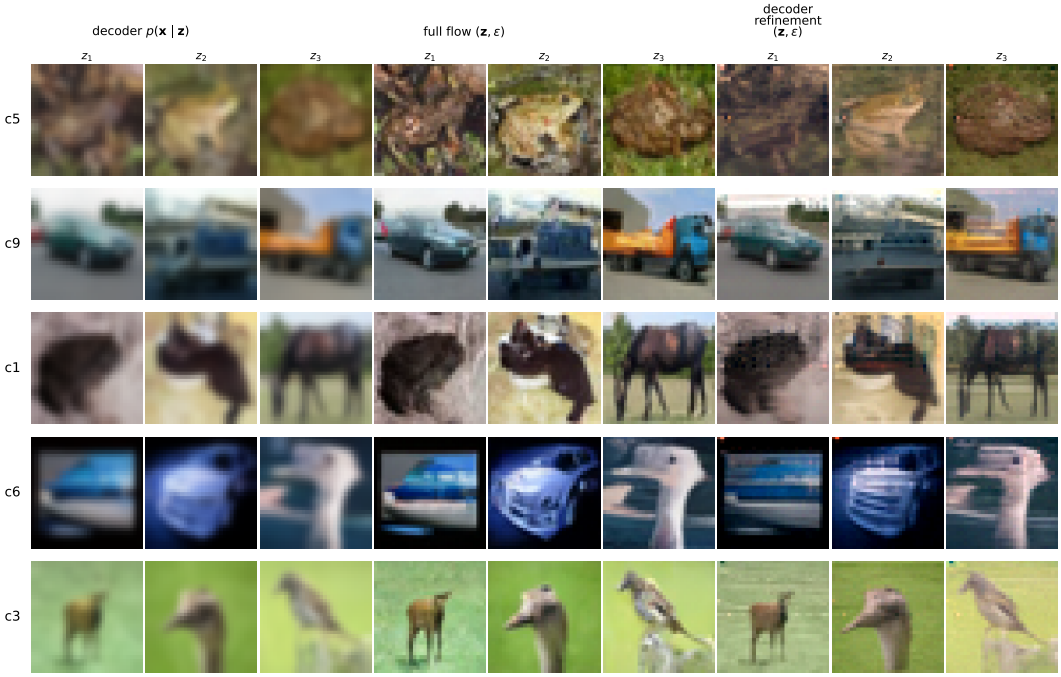

Figure 9: **Representative CIFAR-10 mixture components learned by SCFM.** Each row corresponds to one learned GMM component. The first three columns show decoder-only samples $p_\theta(\mathbf{x} \mid \mathbf{z})$ for latent draws $\mathbf{z}$ assigned to the component. The next three columns show full-flow samples obtained from the structured source $(\mathbf{z}, \varepsilon)$. The last three columns show decoder-initialized refinement samples initialized from the decoder output and refined by integrating from $t_0 = 0.8$ with 5 NFE. Although the components are not label-pure, they capture coherent appearance modes such as shared color palettes, backgrounds, object scale, and coarse geometry, while flow refinement improves visual detail without destroying component coherence.

G.6 CIFAR-10 Latent Probe Diagnostics

For CIFAR-10, we evaluate how much class-relevant information is accessible from the learned latent code. We train linear and nonlinear probes on frozen representations, without updating the SCFM encoder or using data augmentation.

For SCFM, probe results are averaged over five independently trained models with different random seeds. Non-SCFM baseline results are reported from Zhang et al. [2022]. Table 17 reports CIFAR-10

probe accuracies, while Figure 10 shows latent-classifier diagnostics for a representative nonlinear probe.

Table 17: **CIFAR-10 probe accuracy from learned latent representations**. Linear and nonlinear probe results are shown. SCFM is reported as the mean over five seeds, while the non-SCFM baseline numbers are taken from Zhang et al. [2022].

Model	**Linear Probe** ↑	**Nonlinear Probe** ↑
VAE	39.59	54.61
AAE	37.76	52.81
BiGAN	44.90	52.54
DIM	49.62	**69.13**
SCFM (ours)	**50.26**	66.58

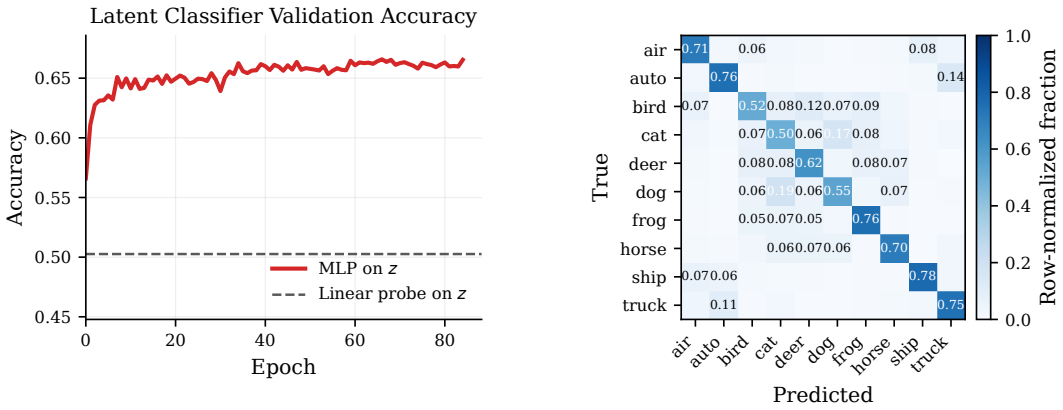

Figure 10: **CIFAR-10 latent classifier diagnostics.** Left: validation accuracy of an MLP trained on the frozen latent code $\mathbf{z}$, with the linear-probe accuracy shown as a dashed baseline. Right: row-normalized confusion matrix for the latent MLP classifier. The representation separates vehicle classes reliably, while most residual confusion occurs among animal categories with similar pose and texture statistics.

G.7 ImageNet-128 Latent Probe Diagnostics

We evaluate whether the representation gains observed on smaller datasets persist at ImageNet scale. For ImageNet-128, we freeze the latent representation and train linear and nonlinear probes without data augmentation. Probe performance is measured using Top-1 and Top-5 accuracy, which quantify how much ImageNet class information is accessible from the latent space. We compare SCFM against a frozen Stable-Diffusion VAE encoder baseline using SD-VAE-FT-EMA [Rombach et al., 2022]. Table 18 shows that SCFM improves over the SD-VAE baseline for both probe families, with the largest gains under the nonlinear probe. This suggests that the SCFM latent space retains substantially more class-relevant information than the frozen VAE encoder alone, while still being learned through the generative training objective.

Table 18: **ImageNet-128 latent-space probing.** Frozen latent representations are evaluated with linear and nonlinear probes. The VAE baseline uses the frozen SD-VAE-FT-EMA encoder from Stable Diffusion [Rombach et al., 2022].

Method	**Probe**	**Top-1 Acc.** ↑	**Top-5 Acc.** ↑
SD-VAE-FT-EMA	Linear	8.00	13.23
SD-VAE-FT-EMA	Non-linear	12.34	26.54
SCFM (ours)	Linear	**9.07**	**23.14**
SCFM (ours)	Non-linear	**27.96**	**53.08**

H Additional Qualitative Samples

MNIST Generation and reconstruction modes. Figure 11 evaluates whether the learned latent space remains useful for generation and reconstruction. Decoder-only samples provide an endpoint proposal, full-flow samples integrate from the learned source prior, and decoder-initialized refinement starts from the decoder output at $t_0 = 0.9$ and applies only 3 flow evaluations. The results show that the decoder captures the digit identity, while short flow refinement improves visual consistency. This illustrates the cooperative role of the VAE endpoint branch and the flow-matching transport in SCFM.

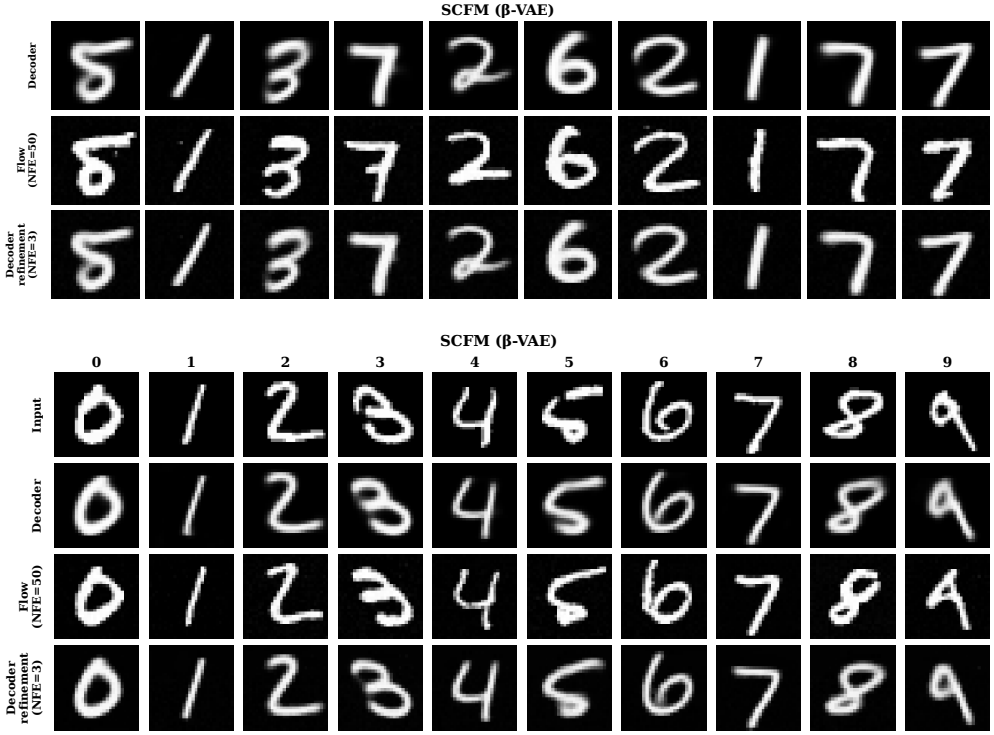

Figure 11: **MNIST generation and reconstruction with SCFM (β-VAE).** Top panel: generation with decoder-only samples from the learned prior, full-flow samples obtained by integrating from the source prior with 50 function evaluations (NFE), and decoder-initialized refinement initialized from the decoder output at $t_0 = 0.9$ and refined with 3 NFE. Bottom panel: reconstruction with input images, decoder-only reconstructions, full-flow reconstructions obtained by integrating from the encoder-induced source with 50 NFE, and decoder-initialized refinement initialized from the decoder output at $t_0 = 0.9$ and refined with 3 NFE.

The rest of this section provides additional qualitative samples for the SCFM models. Figures 12, 13, 14, and 15 show additional uncurated full-flow samples for CIFAR-10, Shapes3D, Cars3D, and ImageNet-128, respectively.

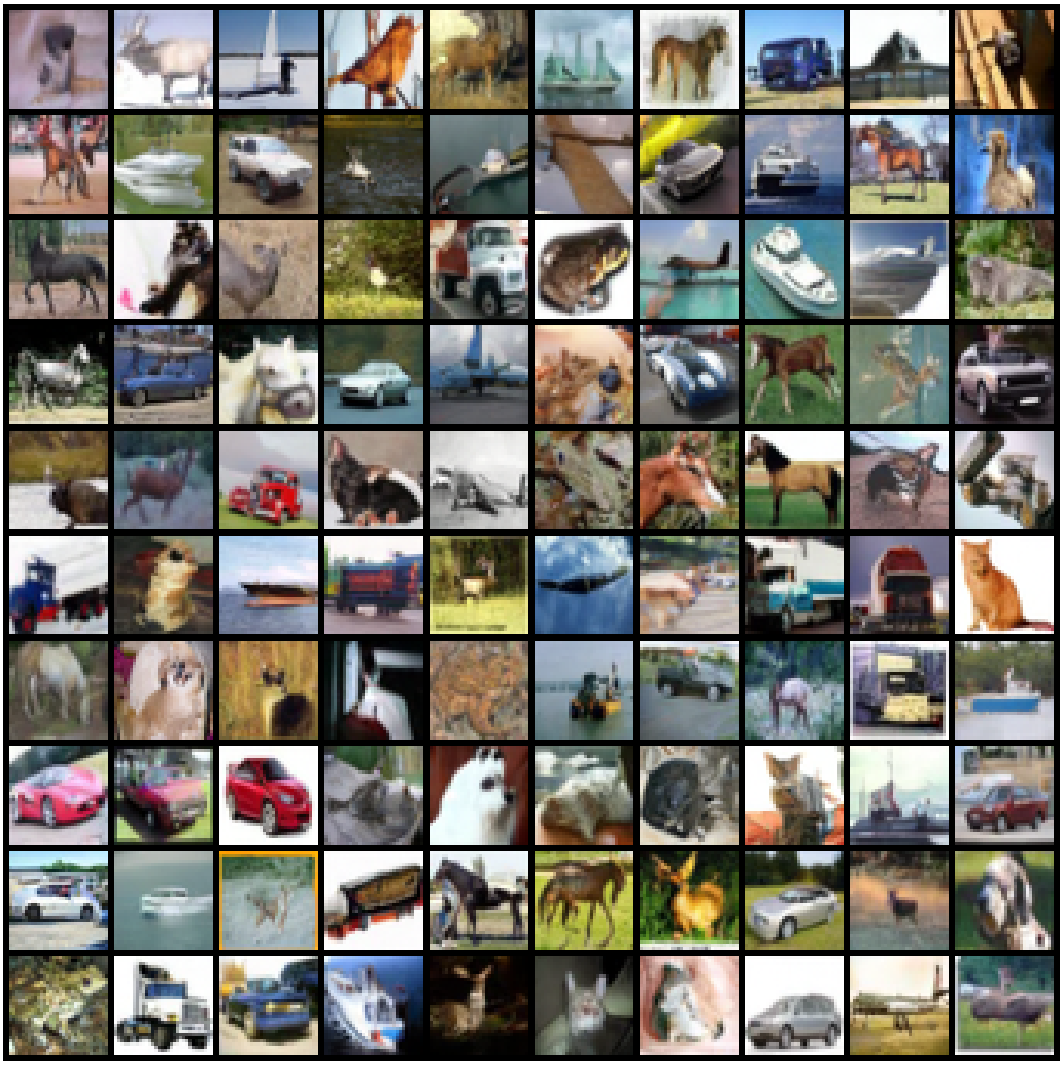

Figure 12: **Additional uncurated CIFAR-10 samples from SCFM.** Samples are generated with full-flow sampling from the learned structured source prior.

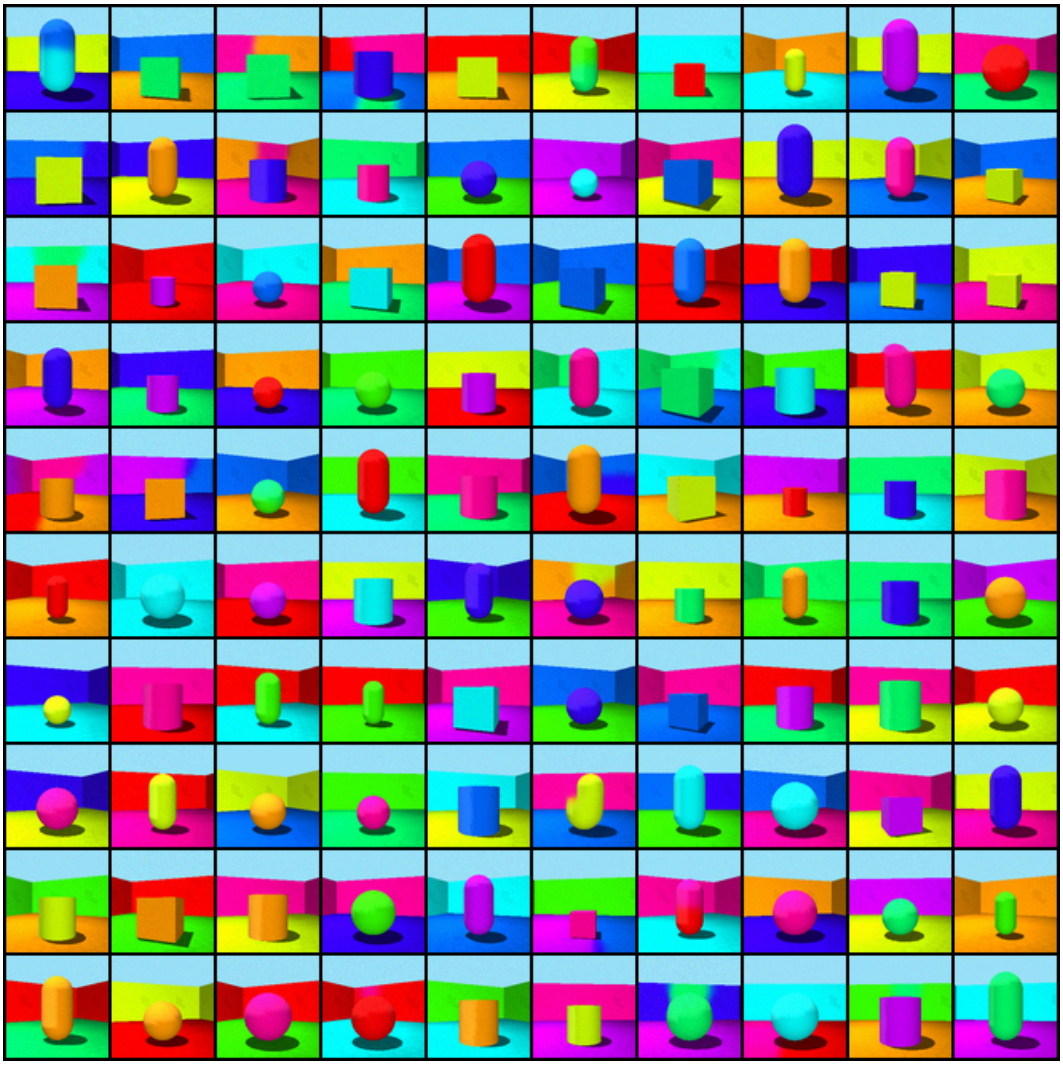

Figure 13: **Additional uncurated Shapes3D samples from SCFM.** Samples are generated with full-flow sampling from the learned structured source prior.

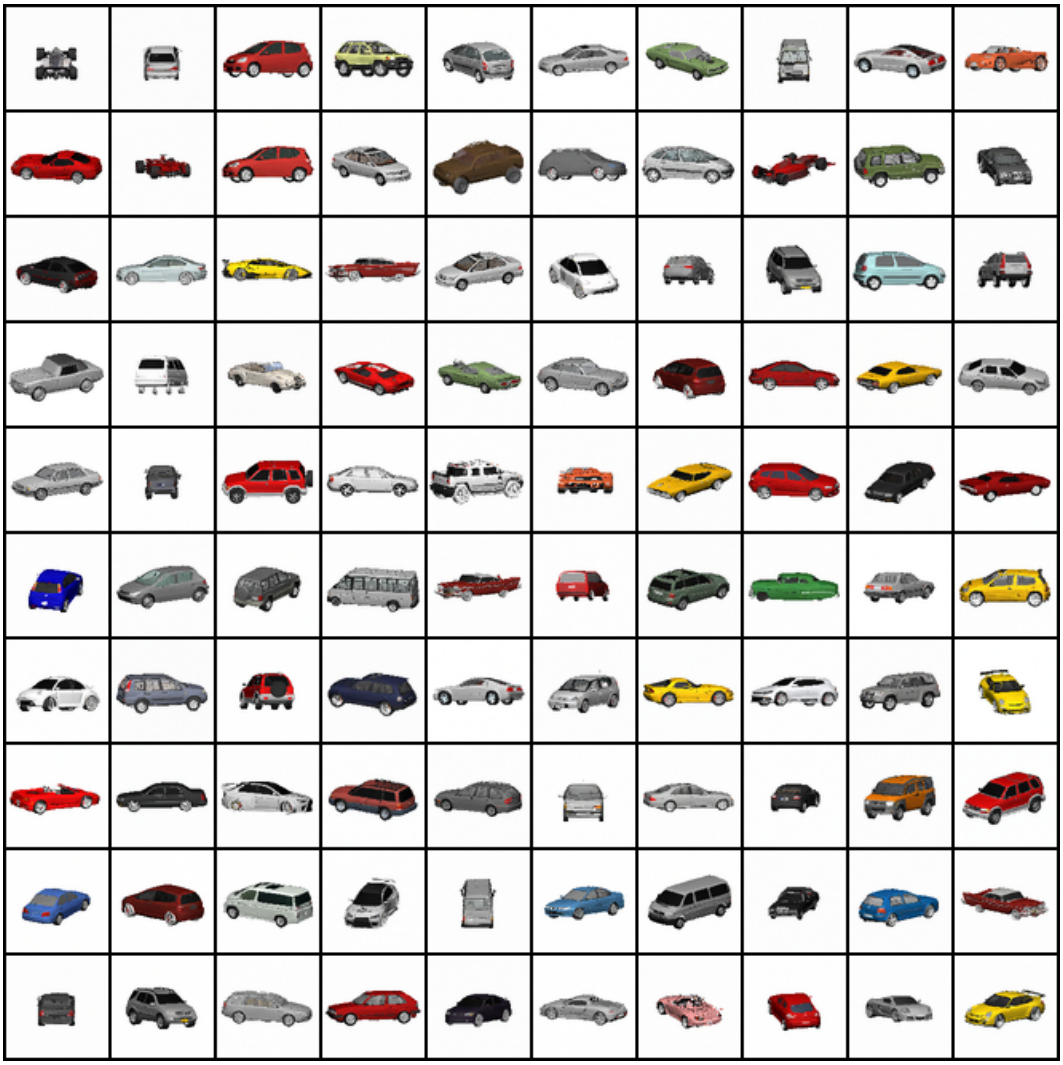

Figure 14: **Additional uncurated Cars3D samples from SCFM.** Samples are generated with full-flow sampling from the learned structured source prior.

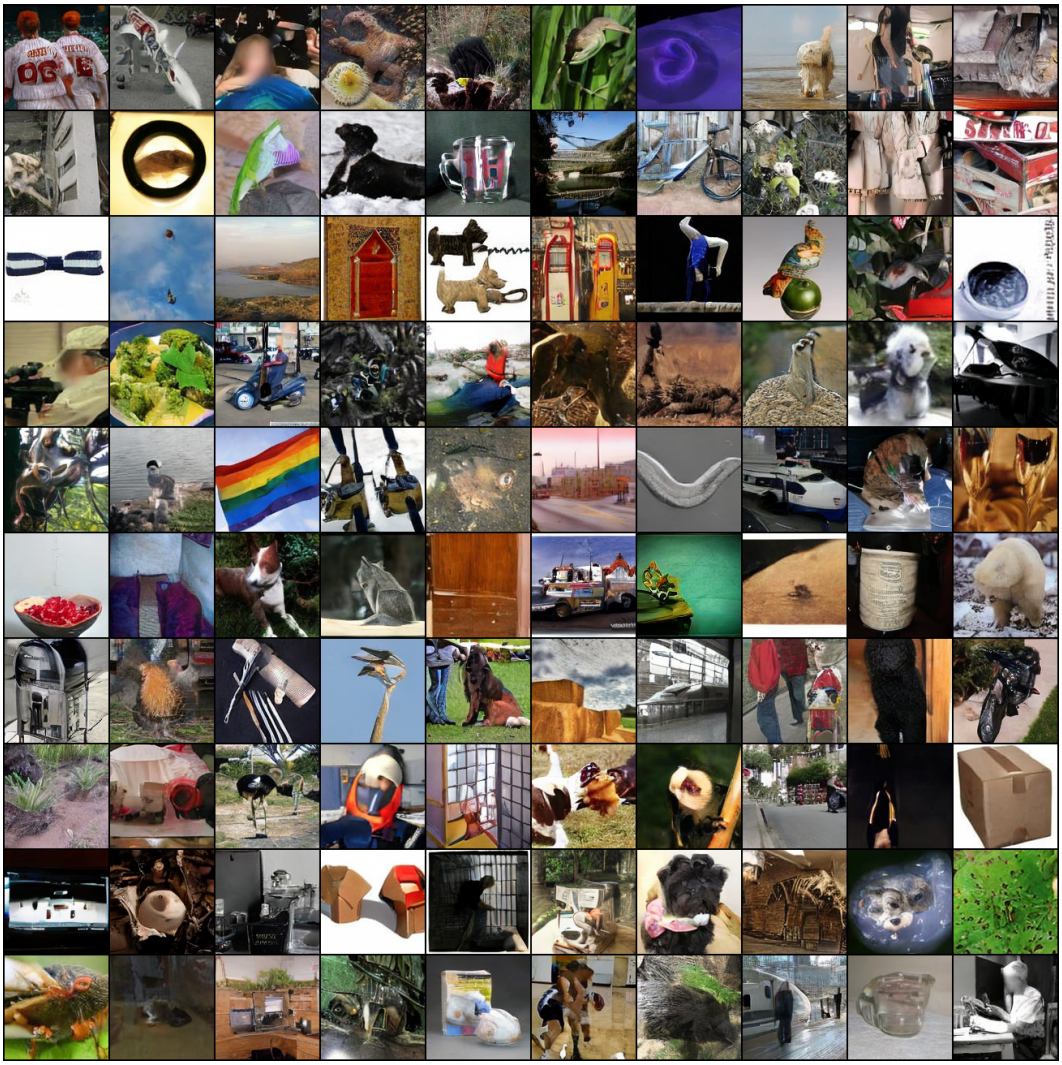

Figure 15: **Additional uncurated ImageNet-128 samples from SCFM.** Samples are generated with full-flow sampling from the learned structured source prior.

I Broader Impacts

This work studies how to equip flow-matching generators with structured latent variables that improve unsupervised representation learning and provide more interpretable control over generation. Potential positive impacts include generative models whose latent structure is easier to analyze, representations that are more useful for downstream tasks without requiring labels, and more transparent generative systems in domains where understanding semantic structure matters.

At the same time, SCFM remains a general-purpose image generator and inherits the risks associated with modern generative models. In particular, improved latent structure and controllable semantic variation could be misused to manipulate generated content more deliberately, while the unsupervised latent variables may absorb and reproduce biases present in the training data. Because the method is trained on large-scale image datasets, privacy-sensitive or socially biased correlations in the data may also be reflected in the learned representation. We therefore view SCFM as a research contribution rather than a deployment-ready system and emphasize that any real-world use should account for dataset bias, privacy, and misuse risks.

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: Section 3 shows how flow matching is extended with latent-variable structure learning, and Section 4 supports this claim with empirical evidence on representation quality, disentanglement, and generation.

   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 concluding "Summary and future work" paragraph in Section 5 discusses the additional compute cost of SCFM and the dependence of decoder-based sampling on the quality of the endpoint latent model. Table 5 quantifies the model-size and FLOP overhead.

   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: [N/A]

Justification:

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: Section 4 summarizes the setups, and Appendix E specifies the architectures, priors, objectives, hyperparameters, samplers, and probing protocols needed to reproduce the main experiments. Appendix F defines the reported evaluation metrics.

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: [Yes]

   Justification: This paper uses public benchmark datasets for the experiments and cites them, and the code for the method is available at `https://github.com/cuent/scfm`.

   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 main experimental section gives the task-level setup, and Appendix E provides dataset-specific training schedules, optimizers, batch sizes, latent dimensions, priors, samplers, and probe settings.

   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: We report mean $\pm$ standard deviation for MNIST clustering and Cars3D/Shapes3D disentanglement under the training-run protocols in Appendix E. CIFAR-10 matched-FM and SCFM FIDs report means and sample standard deviations over four sampling seeds per fixed trained model; these measure sampling variability, not variation across training runs (Appendix G.1). CIFAR-10 SCFM probe accuracies average five independently trained models, without reported standard deviations (Appendix G.6). Other FIDs are single evaluations; uncertainty estimates are not available for every reported result.

   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: Appendix E lists the GPU types used, and Table 5 reports trainable parameters and per-forward FLOPs. We do not report wall-clock time for individual runs, but we follow the corresponding baseline training guidelines and use hardware comparable to that reported in the cited baseline papers.

   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: To the best of our knowledge, the research conforms to the NeurIPS Code of Ethics. The work uses standard benchmark datasets and does not involve human subjects, crowdsourcing, or private data collection.

   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: Appendix I discusses both potential benefits of more structured and interpretable generative representations and possible risks, including misuse of controllable image generation, dataset bias, and privacy-sensitive correlations.

    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 submission does not release a pretrained image generator, scraped dataset, or other high-risk asset, so release-specific safeguards are not applicable.

    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: We use publicly available assets and cite them accordingly.

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:

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: The paper does not involve crowdsourcing or research with human subjects.

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: The paper does not involve human subjects or crowdsourced data collection, so IRB review is 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:

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.