FROM CORE TO DETAIL: UNSUPERVISED DISENTANGLEMENT WITH ENTROPY-ORDERED FLOWS

Anonymous authors
Paper under double-blind review

ABSTRACT

The unsupervised discovery of features that are both semantically meaningful and stable across runs remains a central challenge in representation learning. We introduce entropy-ordered flows (EOFlows), a normalizing flow (NF) framework that augments standard maximum likelihood training with an orthogonality regularizer on the decoder Jacobian. The regularizer is rooted in Independent Mechanism Analysis and encourages geometric disentanglement, and a stochastic estimator makes it tractable at image scale (CelebA at $D = 2352$ and 12288). Learned features form near-orthogonal curvilinear coordinates and can be ordered by their *explained (manifold) entropy* after training, analogous to the ranking by explained variance in PCA. This turns EOFlows into a non-linear generalization of PCA. EOFlows identify an order of magnitude more stable features than existing methods, and these features emerge in distinguishable categories (global, local, and generic) and support tentative semantic interpretations. The local features have strikingly sparse support in pixel space, although our method never enforces this. Retaining only the most important, i.e. highest entropy, features turns the bijective flow into an autoencoder with adjustable bottleneck, rivaling the rate-distortion performance of dedicated autoencoders.

1 INTRODUCTION

A model is considered *disentangled* when its latent features represent distinct, independent, and informative factors of variation and remain stable under repeated training (Bengio et al., 2013; Wang et al., 2024). Hyvärinen & Pajunen (1999) and Locatello et al. (2019) showed that this cannot be achieved in a purely unsupervised manner without additional assumptions. A simple counter-example is provided by generative models with a standard normal latent prior: Since this prior is rotational invariant, any latent rotation preserves the generated distribution – no orientation is special. Thus, the data's intrinsic factors have no reason to align with the latent coordinates, and stable disentanglement is impossible. In our experiments, ordinary normalizing flows clearly exhibit this behavior.

We study *orthogonal (coordate) maps* as an inductive bias to break this symmetry. Orthogonality means that the decoder transforms the latent Cartesian coordinate system into a curvilinear, but still orthogonal coordinate system in data space. Polar coordinates are a classic example for a curvilinear orthogonal system, and EOFlows generalize this concept to learned transformations. If the latent coordinate system is rotated now, the induced coordinates in data space will no longer remain orthogonal. Conversely, the requirement of orthogonal induced coordinates tends to single out a specific latent rotation and leads to reproducible latent features in our experiments.

Our method rests on a key insight from Independent Mechanism Analysis (IMA, Gresele et al., 2021; Ghosh et al., 2023): A decoder defines an orthogonal map if its Jacobian has pairwise orthogonal columns almost everywhere. This can be achieved by adding the *IMA contrast* to the training objective as a regularizer. Their theoretical arguments and low-dimensional results indicate that orthogonality can indeed resolve the rotational ambiguities that otherwise hinder stable feature discovery. In the pioneering work on *principal component flows* (PCF), Cunningham et al. (2022) brought the IMA contrast to normalizing flows, but their experiments were still restricted to low-dimensional synthetic data. EOFlows push the limit of what is practically possible to higher dimensions ($D = 2352$ and 12288), and the discovered features are informative and semantically nearly independent (Fig. 1).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

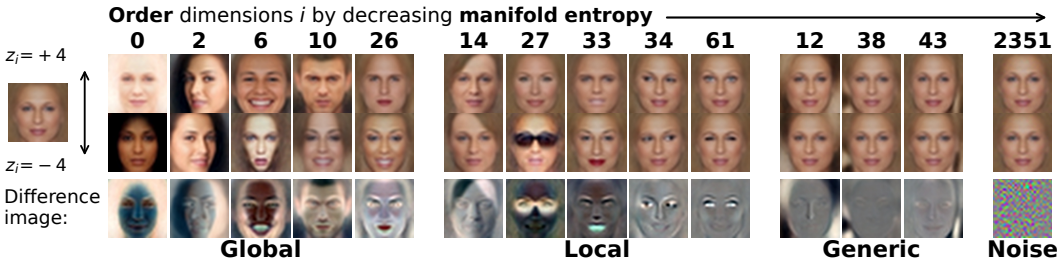


Figure 1: Archetypes of the features learned by EOFlows on CelebA, sorted by decreasing entropy. Global features involve the entire image, e.g. head pose. Local features affect only a small number of pixels, e.g. eye gaze. Generic features resemble windowed Fourier transforms when a subregion (e.g. the background) lacks semantic structure. Finally, features with low entropy contain only noise.

Orthogonal data-space coordinates have various desirable implications. First, we can estimate the *explained entropy* (Galperin & Koethe, 2025) of each latent feature and order the feature space according to this importance measure – hence the name EOFlow. This is analogous to the ordering by explained variance in classical Principal Component Analysis (PCA) and suggests that our method can be interpreted as a nonlinear generalization of PCA. Second, the ordering allows us to define a threshold $C < \dim(Z)$ on the feature indices to separate the most important features (the "core" of the representation) from the rest (the "details"). Setting the details to zero before data generation or reconstruction turns EOFlows into *injective flows*. Crucially, our bottleneck is *adaptive*, because suitable values of C can be determined after training at *inference* time. This sets our concept apart from β-VAEs (Burgess et al., 2018), rectangular flows (Caterini et al., 2021), or $\mathcal{M}$-flows (Brehmer & Cranmer, 2020), which must fix C as a training hyperparameter. The entropy spectra of EOFlows typically decrease until they saturate at the data's noise floor, which defines a natural bound for the choice of C. A model's intrinsic dimension is thus no longer fixed and pre-determined, but limited by the measurement accuracy: geometric structure below the noise level is statistically indistinguishable from noise, regardless of the true generative dimensionality. Geometrically, the core dimensions span the C-dimensional *reconstruction manifold*, an approximation of the true data manifold. The details represent the residual degrees of freedom and generalize the null space of linear maps.

Contributions.

- **Tractable orthogonality.** EOFlows make the orthogonality bias of IMA tractable in Normalizing Flows and achieve rich unsupervised feature discovery on CelebA. The ingredients are a stochastic estimate of the Hadamard regularizer ($2.5\times$ the cost of a plain NF), a fully connected flow with dense learnable mixings (App. G.1), and noise inflation against density overfitting (Sec. 4, 3.5).
- **Entropy ordering and an inference-time bottleneck.** Manifold entropies rank all D features from *core* to *detail* at inference time, like explained variance ranks principal components. Linearized, our objective provably recovers PCA (App. C). Keeping the top C features yields an autoencoder without a reconstruction loss or a fixed bottleneck.
- **Stability.** Comparing learned features across training runs, we find that on CelebA EOFlows reproduce up to 150 features across seeds, where tuned β-VAE variants reach at most 16, nested-dropout flows 9 and plain NFs none (Fig. 5).
- **Sparsity.** Many EOFlow features are strikingly sparse in the pixel domain (local), although neither the architecture nor the objective favors sparsity; the sparsest touches only $3\,\%$ of the pixels, outperforming competitors (Table 1).

2 Related Work

Unsupervised disentanglement with β-VAEs and GANs. The quest for semantically meaningful, independent factors of variation has been largely dominated by the β-VAE framework and its derivatives (Higgins et al., 2017; Burgess et al., 2018; Kim & Mnih, 2018; Chen et al., 2018; Zhao et al., 2018; Dupont, 2018). These models rely on a bottleneck to enforce the discovery of informative features, often resulting in an inherent trade-off between disentanglement and reconstruction quality.

In our experiments they discover at most ≈ 20 meaningful features, regardless of the bottleneck size, and Locatello et al. (2019) found across six objectives that the seed and the regularization strength matter more than the objective itself. Recent studies suggest that the success of β-VAEs in disentangling features is closely linked to Independent Mechanism Analysis (IMA), because training is implicitly biased towards decoders with orthogonal Jacobians (Rolinek et al., 2019; Reizinger et al., 2022). Related regularizers disentangle GAN generators by penalizing off-diagonal entries of their Hessian (Peebles et al., 2020) or non-orthogonal columns of their Jacobian (OroJaR, Wei et al., 2021). EOFlows make the orthogonality bias explicit in a bijective model: no bottleneck size is fixed before training and the exact likelihood yields the entropy ordering.

Normalizing flows for manifold learning. Standard bijective Normalizing Flows (NFs) struggle with data residing on low-dimensional manifolds. This is addressed by injective NF variants like $\mathcal{M}$-flows (Brehmer & Cranmer, 2020), Rectangular Flows (Caterini et al., 2021) and Injective Free-Form Flows (Sorrenson et al., 2024), and by denoising NFs (Horvat & Pfister, 2021). However, they require the intrinsic dimensionality C to be fixed as a training hyperparameter and a reconstruction error to find the manifold. The GIN method (Sorrenson et al., 2020) can identify the data manifold with full-dimensional, volume-preserving flows, but needs an auxiliary class label for each instance.

Independent mechanism analysis and principal component flows. The insight that geometrically independent, i.e. *disentangled*, features correspond to *orthogonal* columns in the decoder Jacobian is a central result of IMA (Gresele et al., 2021; Ghosh et al., 2023). Principal Component Flows (PCF, Cunningham et al., 2022) brought this contrast to normalizing flows but faced limited empirical success in scaling to high-dimensional data. PCF also proposed a stochastic estimate with randomized Jacobian-vector products; ours differs in using decoder columns rather than encoder rows and in being scaled up to $D = 2352$–12288 (App. E).

Importance ordering of the latent features. Inducing a preference to channel important information through specific latent dimensions has been explored through Nested Dropout (ND) in both autoencoders (Rippel et al., 2014) and flows (Bekasov & Murray, 2020). Neural PCA (Li & Hooi, 2022) obtains an ordering by fitting a PCA to the latents of a trained flow. These methods encourage importance ordering but lack an information-theoretic grounding: the order is imposed by the training scheme, not measured from the learned geometry.

Inflation-deflation and generative quality. TarFlow and its successors (Zhai et al., 2025; Chen et al., 2026), a recently introduced improvement of normalizing flows, have highlighted the importance of adding noise during training, without giving an explanation. Noise inflation-deflation was formally analyzed by Horvat & Pfister (2022; 2023) and later adopted as a way to mitigate manifold overfitting (Loaiza-Ganem et al., 2022) and improve generative quality (Loaiza-Ganem et al., 2023). EOFlows can also remove the noise through their entropy order (Sec. 3.5).

3 METHOD

3.1 NORMALIZING FLOWS AND INDUCED CURVILINEAR COORDINATES

NFs learn encoder/decoder pairs $\boldsymbol{z} = \boldsymbol{f}(\boldsymbol{x})$, $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{z}) \coloneqq \boldsymbol{f}^{-1}(\boldsymbol{z})$ that define a bijective mapping between data and latent space ($\boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^D$). We use invertible neural networks where the decoder is realized by running the encoder backwards, so that $\boldsymbol{f}$ and $\boldsymbol{g}$ share the learnable parameters θ. As usual, the latent prior is a standard normal $p(\boldsymbol{Z} = \boldsymbol{z}) = \mathcal{N}(\boldsymbol{z} \,|\, 0, \boldsymbol{I}_D)$. The decoder induces a push-forward distribution $q(\boldsymbol{X})$ in the data space, whose PDF is given by the change-of-variables formula

$$q\big(\boldsymbol{X} = \boldsymbol{g}(\boldsymbol{z})\big) = p(\boldsymbol{Z} = \boldsymbol{z}) \cdot \big|\boldsymbol{J}(\boldsymbol{z})\big|^{-1} \quad \text{with } \boldsymbol{z} \sim p(\boldsymbol{Z}) \tag{1}$$

where $\boldsymbol{J}(\boldsymbol{z}) \coloneqq \frac{\partial \boldsymbol{g}(\boldsymbol{z}')}{\partial \boldsymbol{z}'}\big|_{\boldsymbol{z}'=\boldsymbol{z}}$ is the decoder Jacobian at $\boldsymbol{z}$ and $|\boldsymbol{A}| \coloneqq \det(\boldsymbol{A}^T\boldsymbol{A})^{\frac{1}{2}}$ the volume of a square or rectangular matrix. NFs are trained by the maximum likelihood objective, i.e. by minimizing the negative log-likelihood of $q(\boldsymbol{X})$ on samples $\boldsymbol{x} \sim p^*(\boldsymbol{X})$ of the data distribution:

$$\mathcal{L}_{\text{ML}}(\boldsymbol{x}) \coloneqq -\log(q(\boldsymbol{X} = \boldsymbol{x})) = \frac{1}{2}\big\|\boldsymbol{f}(\boldsymbol{x})\big\|_2^2 + \log\big|\boldsymbol{J}\big(\boldsymbol{z} = \boldsymbol{f}(\boldsymbol{x})\big)\big| + \text{const.} \tag{2}$$

In addition to the push-forward of the density, the decoder also defines a push-forward of the latent Cartesian coordinates into curvilinear coordinates in data space. The individual coordinate lines of

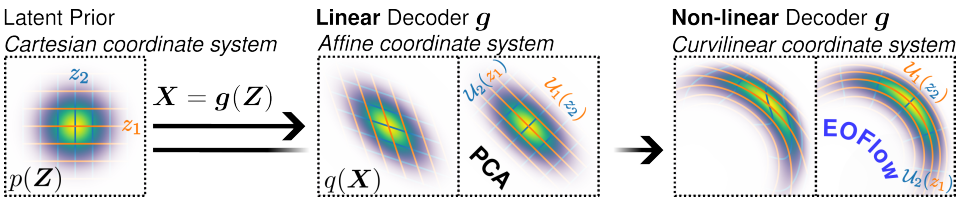


Figure 2: **Induced coordinates.** A standard normal latent (left) is pushed forward by linear (center) and non-linear (right) decoders, which induce affine and curvilinear coordinates in the data space. Both decoders of each pair fit the same distribution, but only the ones on the right, PCA and EOFlows, have orthogonal coordinate lines $\mathcal{U}_1(z_2)$ and $\mathcal{U}_2(z_1)$ that align with the data geometry.

this system are obtained by varying a single latent coordinate z_i, while the others $\boldsymbol{z}_{\neq i}$ remain fixed

$$\mathcal{U}_i(\boldsymbol{z}_{\neq i}) = \left\{\boldsymbol{x} = \boldsymbol{g}(z_i, \boldsymbol{Z}_{\neq i} = \boldsymbol{z}_{\neq i}) : \; z_i \in \mathbb{R}, \boldsymbol{z}_{\neq i} = \text{const}\right\} \tag{3}$$

The properties of this curvilinear coordinate system are determined by its *pull-back metric* $\gamma(\boldsymbol{z}) = \boldsymbol{J}(\boldsymbol{z})^T \boldsymbol{J}(\boldsymbol{z})$. The columns of the Jacobian $\boldsymbol{J}_i(\boldsymbol{z}) = \partial \boldsymbol{g}(\boldsymbol{z}) / \partial z_i$ are the tangent vectors of $\mathcal{U}_i$ at location $\boldsymbol{x} = g(\boldsymbol{z})$. We illustrate the push-forward density of decoders and their induced coordinate systems in Fig. 2. On CelebA, we visualize the feature i by its *archetype*: the images at the two extreme ends $z_i = \pm 4$ of the mean coordinate line $\mathcal{U}_i(\boldsymbol{z}_{\neq i} = \boldsymbol{0})$ and their difference image (Fig. 1; App. G.10 for all models).

3.2 Disentanglement, identifiability, and orthogonality

The term "disentanglement" intuitively refers to representations where each feature has a distinct semantic meaning that is largely independent of the values of the other features. Among its possible formalizations (App. A), we use *geometric disentanglement*, which requires the flow's induced coordinate system to be orthogonal almost everywhere

$$\text{orthogonal map:} \quad \forall i \neq j : \; \boldsymbol{J}_i(\boldsymbol{z})^T \boldsymbol{J}_j(\boldsymbol{z}) = 0 \quad \iff \quad \gamma(\boldsymbol{z}) = \text{diag}\left(\gamma(\boldsymbol{z})_{11}, ..., \gamma(\boldsymbol{z})_{DD}\right) \tag{4}$$

Identifiability concerns our ability to learn such representations reliably. Nelson et al. (2026) call a representation *structurally identifiable* if the learned features reproduce the independent factors of the true generative process, and *statistically identifiable* if repeated training always converges to the same features. We target the latter, since the true process of natural data is unknown. Identifiability must hold up to *tolerable* ambiguities, usually taken as index permutations and element-wise reparameterizations of the latents. We follow this view and disambiguate the disentanglement by additional conditions: Indices are fixed by sorting the features by importance, and the parameterization is fixed by enforcing a standard normal prior (instead of an arbitrary separable prior). These conditions do not affect feature identification, leaving sign flips $z_i \to -z_i$ as the only remaining ambiguity.

Orthogonality is not an arbitrary choice: It is the only geometric constraint that is stable under precisely the tolerable ambiguities discussed above (Gresele et al., 2021). If the metric $\gamma(\boldsymbol{z})$ is non-orthogonal, i.e. has non-zero off-diagonal elements, one can construct metric-preserving latent rotations, which are not tolerable. Conversely, more restrictive *conformal maps* with $\gamma(\boldsymbol{z}) = \kappa(\boldsymbol{z})\mathbb{I}$ lose the tolerance against element-wise reparameterizations (Buchholz et al., 2022). Figure 2 illustrates how orthogonal maps achieve identifiability: If the latent space is rotated (a non-tolerable ambiguity), the orthogonality between the induced coordinates $\mathcal{U}_i$ is lost, despite the generated distribution remaining unchanged due to the rotational symmetry of the standard normal prior.

On the other hand, we conjecture that orthogonal maps with separable latent priors are not universal approximators (a formal proof is beyond the scope of this paper). Thus, orthogonality should only be enforced as a *soft* constraint in order to maintain control of the inherent trade-off between generative accuracy and geometric disentanglement.

3.3 Training objective

We employ a regularized maximum likelihood objective for EOFlows with λ as a tuning parameter

$$\mathcal{L}_{\text{EO}}(\boldsymbol{x}) = \mathcal{L}_{\text{ML}}(\boldsymbol{x}) + \lambda \cdot \mathcal{L}_{\text{H}}(\boldsymbol{x}), \quad \lambda > 0 \tag{5}$$

$\mathcal{L}_{\mathrm{H}}(\boldsymbol{x})$ is the *Hadamard regularizer*, which ensures *soft* orthogonality between all D Jacobian columns $\boldsymbol{J}_i$ simultaneously:

$$\mathcal{L}_{\mathrm{H}}(\boldsymbol{x}) := \sum_{i=1}^{D} \log \left\|\boldsymbol{J}_i(\boldsymbol{f}(\boldsymbol{x}))\right\|_2 - \log \left|\boldsymbol{J}(\boldsymbol{f}(\boldsymbol{x}))\right| \geq 0 \tag{6}$$

Gresele et al. (2021) called this the *IMA contrast*, though it stems from *Hadamard's inequality*:

$$\left|\boldsymbol{A}\right| \leq \prod_{i=1}^{D} \left\|\boldsymbol{A}_i\right\|_2 \tag{7}$$

It states that the volume of any matrix $\boldsymbol{A}$ is bounded by the product of its column norms, and equality is achieved if and only if all columns $\boldsymbol{A}_i$ are pairwise orthogonal.

Eq. 6 can be efficiently evaluated in practice: $|\boldsymbol{J}(\boldsymbol{f}(\boldsymbol{x}))|$ is already available from the maximum likelihood loss, and $\boldsymbol{J}_i(\boldsymbol{z})$ can be calculated via Jacobian-vector product primitives provided by all major autodiff frameworks (e.g. PyTorch, JAX). We prove in App. C that for a linear decoder the optimum of $\mathcal{L}_{\mathrm{EO}}$ is the PCA solution. Linear EOFlows perform PCA by gradient descent, and non-linear ones generalize it to curvilinear coordinates.

3.4 Entropy ordering, manifold learning, and compression

Each coordinate line $\mathcal{U}_i(\boldsymbol{z}_{\neq i})$ is a 1-dimensional manifold in data space, on which the prior of z_i induces the *manifold PDF* $q_i\big(\boldsymbol{X} = \boldsymbol{g}(\boldsymbol{z})\big) = p(z_i) \cdot \|\boldsymbol{J}_i(\boldsymbol{z})\|_2^{-1}$ by the injective change-of-variables formula. Its expected negative log-density over the entire latent space is the *manifold entropy* (Galperin & Koethe, 2025) of feature i

$$H_i := H(q_i) = \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}\left[\frac{1}{2}(1 + \log 2\pi) + \log \left\|\boldsymbol{J}_i(\boldsymbol{z})\right\|_2\right] \tag{8}$$

A large H_i means that the data vary strongly along $\mathcal{U}_i$. Under the manifold hypothesis, this is the case for coordinates aligned with the data manifold, whereas coordinates perpendicular to it carry only noise and have small entropies. Galperin & Koethe (2025) found that the model entropy decomposes into a sum over manifold entropies minus a non-negative term $\mathcal{I}$, which is precisely the residual expected Hadamard loss (6) under the model distribution:

$$H := H(q) = \sum_{i=1}^{D} H_i - \mathcal{I}, \qquad \mathcal{I} = \mathbb{E}_{\boldsymbol{x}\sim q(\boldsymbol{X})}\big[\mathcal{L}_{\mathrm{H}}(\boldsymbol{x})\big] \geq 0. \tag{9}$$

As Maximum Likelihood makes $q(\boldsymbol{X}) \approx p^*(\boldsymbol{X})$, $H(q)$ converges to the data entropy $H^* = H(p^*)$, a constant. The Hadamard regularizer keeps $\mathcal{I}$ small, so that $\sum_i H_i \approx H^*$. Thus, each H_i measures only the variability along its own coordinate line without significant cross-talk. This is analogous to PCA, where the variances of the individual principal components sum to the total variance. We can thus meaningfully sort the latent coordinates, and this inspired our method's name "EOFlows".

In addition, we can split the latent space into a *core* subspace of important features (high H_i, aka. "content"), and a *detail* subspace of the rest (low H_i, aka. "noise"). This allows us to introduce a *dynamic bottleneck* of size $C < D$, which assigns features $\boldsymbol{z}_{\leq C}$ (after importance ordering) to the core subspace. This defines an autoencoder-like dimension reduction, but with the important advantage that we can decide about the optimal C *after* we have trained the geometry of $\boldsymbol{g}(\boldsymbol{z})$ and calculated the H_i. An autoencoder-like reconstruction is defined as

$$\widehat{\boldsymbol{x}}_{\downarrow C} = \boldsymbol{g}\Big(\big[\boldsymbol{z}_{\leq C} = \boldsymbol{f}_{\leq C}(\boldsymbol{x}), \boldsymbol{z}_{>C} = 0\big]\Big) \tag{10}$$

If the detail subspace contains only independent isotropic noise, the $\mathcal{U}_i$ for $i > C$ tend to be straight lines (App. F.7.4). Since these $\mathcal{U}_i$ are perpendicular to the data manifold by way of orthogonality regularization, the reconstructions $\widehat{\boldsymbol{x}}_{\downarrow C}$ of off-manifold points $\boldsymbol{x}$ will approximately be orthogonal projections onto the manifold, i.e. (6) implicitly reduces the squared reconstruction error $\|\boldsymbol{x} - \widehat{\boldsymbol{x}}_{\downarrow C}\|_2^2$. We show in Figs. 3 and 4b that this leads to powerful denoising and to compression with favorable rate-distortion behavior.

The dynamic bottleneck also reveals a close relation of EOFlows to *nested dropout flows* (NDFlows, Bekasov & Murray, 2020), which minimize the reconstruction error of (10) *during training* for a random bottleneck $C \sim p_C$ drawn from a geometric distribution per image, weighted by ν against the maximum likelihood loss (App. F.5). This forces the core information into the first dimensions and recovers PCA in the linear case (Rippel et al., 2014). Importantly, NDFlows enforce the order *during training* through their reconstruction loss, whereas EOFlows arrive at an equally *compressed* representation (Fig. 4a) without any such term and order it only after training.

3.5 NOISE INFLATION AND DENOISING

We train on data inflated with isotropic Gaussian noise that is redrawn at every step:

$$\boldsymbol{x}_{\boldsymbol{\epsilon}} \coloneqq \boldsymbol{x} + \sigma_{\boldsymbol{\epsilon}} \cdot \boldsymbol{\epsilon}\,, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_D) \tag{11}$$

Noise may look like a necessary evil, but it cures two problems of likelihood-based models. First, bijective flows cannot learn lower-dimensional manifolds: their likelihood would grow without bound if they contracted the normal directions (manifold overfitting, Loaiza-Ganem et al., 2022). Second, a finite training set is a zero-dimensional manifold, so memorizing it would give unbounded likelihood as well (Ross et al., 2024). With noise redrawn at every step, the flow instead fits a smooth and bounded kernel density estimate $\frac{1}{N}\sum_n \mathcal{N}(\boldsymbol{x}_n, \sigma_{\boldsymbol{\epsilon}}^2 \boldsymbol{I}_D)$ of the training set. We use $\sigma_{\boldsymbol{\epsilon}} = 0.05$ throughout all experiments, selected by sweeping $\sigma_{\boldsymbol{\epsilon}}$ from 0.01 to 0.5 on smaller networks (App. D.3). Values $\sigma_{\boldsymbol{\epsilon}} < 0.01$ (and specifically $\sigma_{\boldsymbol{\epsilon}} = 0.002$, the quantization noise level of 8-bit images) do not work, as two out of three training runs diverge. Noise $\sigma_{\boldsymbol{\epsilon}} > 0.5$ starts to mask important information, since structures with variance below $\sigma_{\boldsymbol{\epsilon}}^2$ become invisible. The entropy of the noise $H_{\sigma_{\boldsymbol{\epsilon}}} = \frac{1}{2}\log(2\pi e\, \sigma_{\boldsymbol{\epsilon}}^2)$ is clearly visible in the entropy spectrum (Fig. 4a) as a lower bound for the spectrum. EOFlows handle this floor gracefully: The associated coordinate lines $\mathcal{U}_i$ become straight lines orthogonal to the data manifold (App. F.7.4) and can be easily removed during reconstruction by setting all $\boldsymbol{z}_i = 0$ whose $H_i \approx H_{\sigma_{\boldsymbol{\epsilon}}}$, see Fig. 3. This denoising is explained in detail under *Compression and denoising* in Sec. 4.

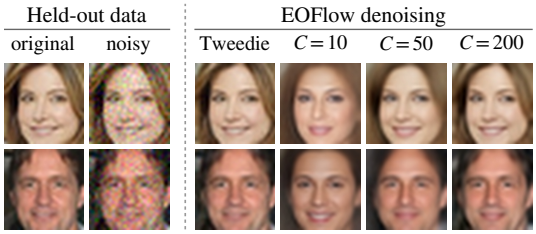


Figure 3: **Denoising** of held-out images by one Tweedie step and through the top-C bottleneck of an EOFlow ($\lambda = 1$).

4 EXPERIMENTS

Stochastic estimation of $\mathcal{L}_{\mathrm{H}}$ at scale: Instead of calculating all column norms $\|\boldsymbol{J}_i\|_2$ in (6) for each data instance, we follow (Cunningham et al., 2022) and select a single index i uniformly at random. This requires only one Jacobian-vector product per instance instead of D. Since the index is uniform, $D \log \|\boldsymbol{J}_i\|_2$ has expectation $\sum_{k=1}^{D} \log \|\boldsymbol{J}_k\|_2$, so the estimate of $\mathcal{L}_{\mathrm{H}}$ is unbiased for any batch size. A training step thus takes only $2.5\times$ as long as for a plain NF (App. B), regardless of D and B.

Data, baselines and training: We train on CelebA (Liu et al., 2015), center-cropped and downscaled to $28 \times 28 \times 3$ ($D = 2352$) and inflated with noise of $\sigma_{\boldsymbol{\epsilon}} = 0.05$. We vary the Hadamard regularization strength of EOFlows in $\lambda \in \{0, 0.01, 0.1, 1, 3, 10\}$, where $\lambda = 0$ is the unconstrained NF, and compare with PCA, with NDFlows ($\nu \in \{0.1, 1, 10\}$) of the same architecture, and with fully connected β-VAE (Higgins et al., 2017), β-TCVAE (Chen et al., 2018), FactorVAE (Kim & Mnih, 2018) and AnnealedVAE (Burgess et al., 2018) with 32 latents, each tuned for the largest number of stable features (defined below). See App. F.2 and App. F.4 for more details.

EOFlow architecture: All 28×28 flows share one fully connected architecture of 12 Real NVP affine coupling blocks (Dinh et al., 2017) with two-hidden-layer MLPs of width 1024, each followed by a random permutation and ActNorm (Kingma & Dhariwal, 2018), and learnable affine $D \times D$ mixing matrices between blocks and after the last one (App. F.1). Our convolutional EOFlow for $D = 12288$ is specified below under *Scaling to higher resolution* and in App. G.2.

Entropy spectra: Fig. 4a shows the manifold entropy spectra of all trained model classes together with the noise entropy $H_{\sigma_{\boldsymbol{\epsilon}}}$, which forms an approximate floor. As EOFlows and NDFlows concentrate the entropy in a few hundred coordinates, a small bottleneck C loses little information. For noisy PCA, the manifold entropies directly measure the variance along each principal direction (App. G.5). The best VAE, FactorVAE, uses only 20 of its 32 latents and leaves the rest inactive, whereas the bijective flows must account for all D dimensions. The plain NF spreads its entropy uniformly over all D coordinates, so it admits no meaningful ordering, and a bottleneck would keep arbitrary latents.

Compression and denoising: Fig. 4b turns the entropy ordering into a rate–distortion curve: held-out images are inflated, encoded, truncated to their C highest-entropy latents, decoded and compared to

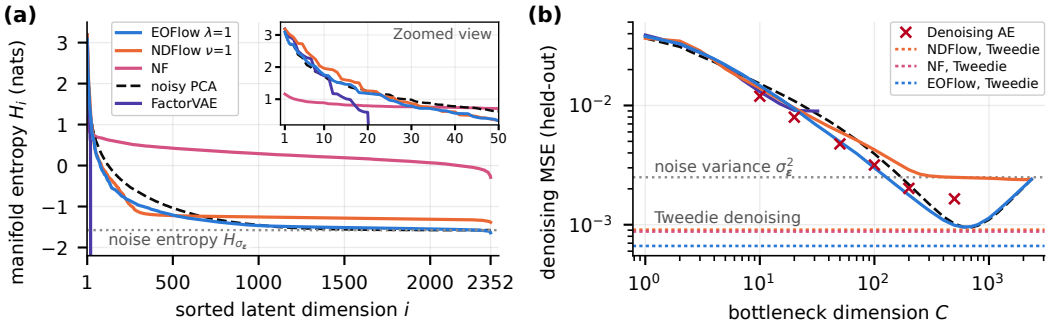


Figure 4: **Entropy spectra, compression and denoising rate-distortion** for representative models. **(a)** Manifold entropy spectra H_i define the ordering of latent dimensions. EOFlows and NDFlows concentrate their entropy in a few hundred dimensions; EOFlows flatten towards the noise entropy H_{σ_ϵ}, separating high-entropy *core* from low-entropy *detail* dims. The plain NF spreads it over all dims, and FactorVAE (best VAE) uses at most 20 active latents. **(b)** Denoising error of held-out images reconstructed through the top-C entropy-ordered bottleneck; the plain NF is omitted, as its error is far above the plotted range. Additionally to models in (a): denoising autoencoders trained separately for each C (App. F.6), and Tweedie denoising of the flows without a bottleneck (dotted).

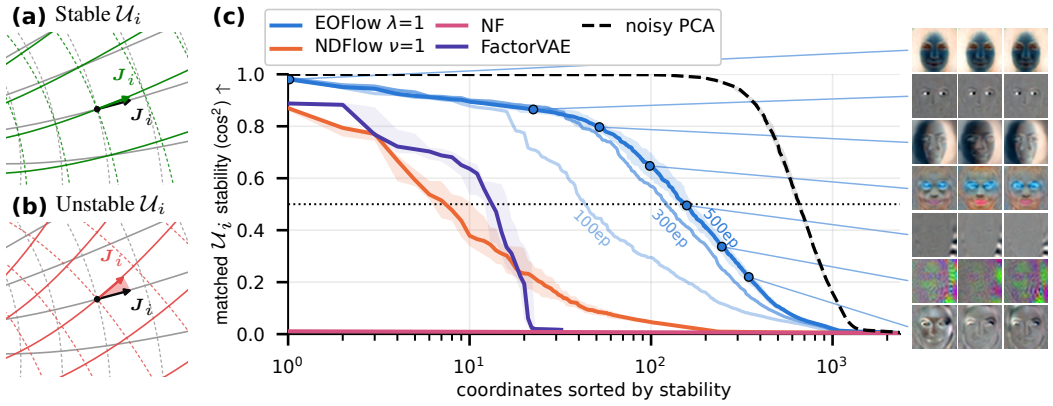


Figure 5: **Stable features across seeds. (a, b)** Sketch: coordinate lines $\mathcal{U}_i$ of a reference model (gray) against one that reproduces them (a) and one that does not (b), quantified by the alignment of their tangent vectors $\boldsymbol{J}_i$. **(c)** Left: matched $\mathcal{U}_i$ stability (Sec. 4) of every coordinate, sorted in decreasing order (median over seed pairs). EOFlows keep gaining stable coordinates up to 500 epochs, whereas NDFlows and FactorVAE, trained just as long, do not. The plain NF has no stable coordinate, and noisy PCA (dashed) has hundreds of them, by construction. Budgets and seeds in Table F.1. Right: archetypes of 7 representative EOFlow features in three random seeds (all six: App. G.7).

the clean image, which tests compression and denoising at once. That EOFlows compete here at all is remarkable, since their objective (5) contains no reconstruction error. Together with maximum likelihood, their *local* bias towards orthogonal coordinates aligns them *globally* with the principal factors of the data, so that thousands of features can be sorted by entropy in a meaningful way.

On unseen images, EOFlows reconstruct better than NDFlows for $C \geq 20$ and better than denoising autoencoders for $C \geq 100$, although dedicated autoencoders have been trained separately for each C for exactly this task. At large bottlenecks, PCA catches up with EOFlows, since the detail coordinates model Gaussian noise, for which the EOFlow converges to a PCA-like linear projection. A single correction by Tweedie's formula $\widehat{\boldsymbol{x}} = \boldsymbol{x} + \sigma^2 \cdot \nabla_{\boldsymbol{x}} \log q(\boldsymbol{x})$ (Efron, 2011; Loaiza-Ganem et al., 2023; Zhai et al., 2025) on the flow outputs (dotted lines) denoises better than any bottleneck (best for EOFlows). However, this acts on all D coordinates and cannot compress. See App. G.8 and G.9 for prior samples and reconstructions through the bottleneck.

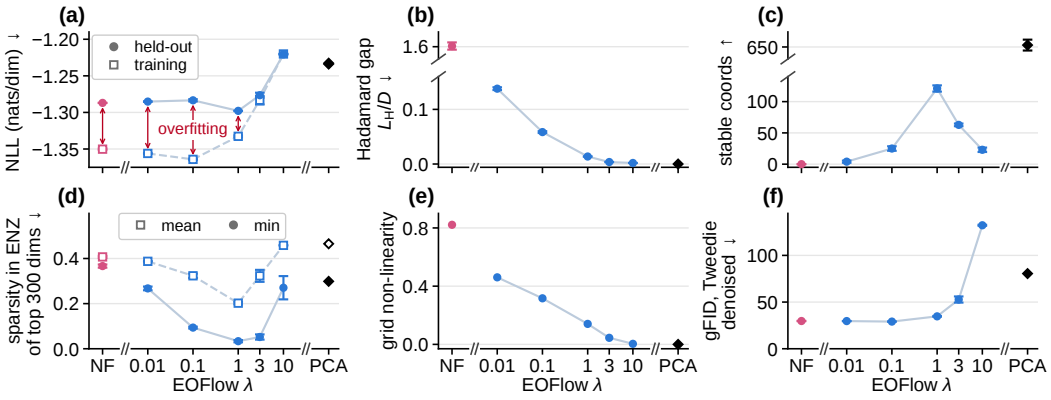


Figure 6: **Sweeping the orthogonality penalty λ from NF to PCA.** Each NF/EOFlow trained with 3–4 seeds for 300 epochs, 6 noisy PCA fits. **(a)** Negative log-likelihood on training and held-out data; the gap between them is overfitting. **(b)** Hadamard gap $\mathcal{L}_{\mathrm{H}}/D$. **(c)** Coordinates reproduced per seed pair (matched $\mathcal{U}_i$ stability ≥ 0.5). **(d)** Sparsity via mean and minimum ENZ. **(e)** Non-linearity of coordinates (one seed, App. F.7.4). **(f)** gFID of prior samples after Tweedie denoising.

Stable features across seeds: To compare how stable features are found across random seeds, we compare the coordinate grids $\mathcal{U}_i$ locally via the decoder Jacobians between two trained runs. Instead of the naive correlation between latent codes, which is a statistic between distributions, our stability metric compares the Jacobians at every sample and takes the expectation afterwards (App. F.7.3). For every model, we replace the Jacobian at fixed training images by its polar factor, the closest orthogonal matrix, whose columns point along the locally orthogonal coordinate lines. The squared cosines between these directions in two runs, averaged over the images and Hungarian-matched to remove the permutation (and, by squaring, the sign) ambiguity, give the *matched $\mathcal{U}_i$ stability* of each coordinate, 1 if it is reproduced exactly and ≈ 0 for random frames. Coordinates matched at 0.5 or more count as stable; we report the mean and standard deviation over seed pairs.

Fig. 5 and Table 1 compare the best setting of each model class. EOFlows at $\lambda = 1$ reproduce $121_{\pm 5}$ coordinates per seed pair after 300 epochs and $154_{\pm 10}$ after 500, whereas the tuned VAEs reproduce at most 16 of their 32 latents, NDFlows fewer than 10, and NFs none. The threshold is conservative: the archetypes in Fig. 5b) show that coordinates with a matched stability below 0.5 are still similar.

Coordinate sparsity: A feature is sparse if moving along its coordinate line changes only a small number of pixels of the decoded image. Archetypes show only the two endpoints of one coordinate line. For a local measure, we use the Jacobian column $\boldsymbol{J}_i$, which gives the change of the image along $\mathcal{U}_i$ at a sample, ask over how many pixels this change is spread, and take the expectation over samples. We use the effective percentage of non-zero pixels (ENZ; He et al., 2026), a threshold-free and scale-invariant pixel count (App. F.7.2). It ranges from $1/D$ (a single pixel changes) to 1 (all pixels change equally), and a random direction scores 0.48. To measure sparsity, we evaluate the ENZ over the 300 highest-entropy *core* features (all 32 for VAEs) and report its minimum and mean (Fig. 6d, Table 1). The *local* features of Fig. 1 make EOFlow coordinates the sparsest: at $\lambda = 1$ the sparsest one touches $3\,\%$ and $20\,\%$ on average, whereas even the sparsest coordinate of every baseline touches more than a quarter of it.

Sweeping the orthogonality penalty λ from NF to PCA: Fig. 6 follows six quantities as the Hadamard penalty λ runs from the unconstrained NF ($\lambda = 0$) to $\lambda = 10$. Orthogonality (b) and the linearity of the coordinate grid (e) increase monotonically. Already $\lambda = 0.01$ removes most of the Hadamard gap at negligible cost in likelihood (a) and none in sample quality (f), and its archetypes are interpretable (App. G.10), yet it reproduces only $4.0_{\pm 1.7}$ coordinates per seed pair (c). Held-out likelihood (a), stability (c) and sparsity (d) peak at $\lambda = 1$. Stronger penalties push the flow towards PCA (b, e), since a linear map can always be made exactly orthogonal by rotating the latent space, whereas orthogonal curvilinear coordinates are a restrictive class (Sec. 3.2). Yet, at $\lambda = 10$ it does not inherit PCA's stability and falls behind in likelihood and FID, although PCA lies in its model class, likely because the variance of the stochastic penalty then dominates the training loss (Fig. B.1c).

Table 1: **Reproducible, sparse and disentangled.** Mean$_{\pm\text{sd}}$ over seeds. CelebA at 28×28: 300 epochs (AnnealedVAE: 100), fully-connected architectures; stability and sparsity as in Sec. 4; TAD = attribute disentanglement score (Yeats et al., 2022) (App. G.6). Orthogonality = Hadamard gap of the core coordinates (App. F.7.1). dSprites and Shapes3D: 64×64, 100 epochs, convolutional architectures (App. G.3). PCA fits orthogonal and reproducible components by construction (in parentheses). On the disentanglement benchmarks we compare EOFlows with the VAEs only.

	CelebA (fully connected)				dSprites (conv.)		Shapes3D (conv.)	
model	stability ↑	orthog-onality ↓	sparsity min/mean ↓	TAD ↑	MIG ↑	DCI ↑	MIG ↑	DCI ↑
EOFlow $\lambda = 1$	**121**$_{\pm 5}$	**0.041**$_{\pm .001}$	**0.03/0.20**	0.72$_{\pm .01}$	0.14$_{\pm .10}$	0.07$_{\pm .04}$	0.15$_{\pm .03}$	**0.52**$_{\pm .03}$
EOFlow $\lambda = 0.1$	25.0$_{\pm 3.6}$	0.17$_{\pm <.01}$	0.09/0.32	**1.00**$_{\pm .01}$	**0.26**$_{\pm .01}$	0.19$_{\pm <.01}$	0.17$_{\pm .07}$	0.41$_{\pm .09}$
NDFlow $\nu = 1$	8.5$_{\pm 1.8}$	0.29	0.30/0.40	0.24$_{\pm .01}$	–	–	–	–
NDFlow $\nu = 0.1$	1.7$_{\pm 1.2}$	0.46$_{\pm <.01}$	0.32/0.41	0.58$_{\pm .07}$	–	–	–	–
NF	0	1.6$_{\pm <.1}$	0.37/0.41	0.00$_{\pm .00}$	–	–	–	–
FactorVAE	15.7$_{\pm 1.2}$	0.38$_{\pm .07}$	0.28/0.37	0.72$_{\pm .12}$	0.15$_{\pm .07}$	0.14$_{\pm .03}$	0.13$_{\pm .03}$	0.14$_{\pm .05}$
β-TCVAE	10.3$_{\pm 1.2}$	0.55$_{\pm .01}$	0.33/0.41	0.44$_{\pm .01}$	**0.26**$_{\pm .09}$	**0.25**$_{\pm .03}$	0.31$_{\pm .12}$	0.42$_{\pm .06}$
β-VAE	9.7$_{\pm 2.5}$	0.50$_{\pm .01}$	0.32/0.41	0.46$_{\pm .03}$	0.06$_{\pm .02}$	0.06$_{\pm .01}$	0.22$_{\pm .07}$	0.25$_{\pm .08}$
AnnealedVAE	5.4$_{\pm 1.4}$	0.77$_{\pm .05}$	0.36/0.43	0.40$_{\pm .10}$	0.06$_{\pm .05}$	0.04$_{\pm .03}$	**0.35**$_{\pm .03}$	0.20$_{\pm <.01}$
noisy PCA	(653$_{\pm 9}$)	(0)	0.30/0.47	0.52	–	–	–	–

Architecture ablation: The Hadamard regularizer is not invariant to rotations of the latent space, so the flow must be free to choose its latent basis, which our twelve learnable $D \times D$ mixing matrices provide. Keeping all twelve learnable, only the final one, or none (100 epochs, three seeds each; App. G.1) leaves the entropy ordering intact, but reproducibility drops from $46.7_{\pm 7.0}$ stable coordinates per seed pair to $9.0_{\pm 2.0}$ and $3.3_{\pm 1.5}$, so dense learnable mixings are crucial.

Scaling to higher resolution: Fully connected MLPs become impractical beyond $D \approx 10^4$. To show that the recipe scales in principle, we train a multiscale convolutional EOFlow on CelebA at 64×64 ($D = 12288$, $\sigma_\epsilon = 0.1$; App. G.2) as a proof of concept. It orders its coordinates by entropy like the fully connected model and its leading archetypes show the same facial features. We use a smaller version of this convolutional flow on the higher-dimensional toy benchmarks below (App. G.3).

Disentanglement benchmarks: On dSprites and Shapes3D, EOFlows concentrate the entropy in a few coordinates and drop to the noise level at a sharp knee close to the number of ground-truth factors (Fig. G.5, App. G.3). Against the state-of-the-art VAE-based methods (Locatello et al., 2019) they are competitive, measured by MIG (Chen et al., 2018) and DCI (Eastwood & Williams, 2018) (Table 1).

5 Limitations and Conclusions

Unsupervised disentanglement needs an inductive bias, and EOFlows demonstrate that orthogonality of the decoder Jacobian is an effective one. Made practical at image scale by a stochastic estimator and noise inflation, it yields interpretable features that can be ordered by manifold entropy, just as PCA orders its components by explained variance. Keeping only the top features turns the bijective flow into an autoencoder whose bottleneck is chosen at inference time and which rivals dedicated autoencoders in rate–distortion without ever being trained to reconstruct. Remarkably, many features affect only a few pixels, although neither architecture nor objective favors sparsity.

Unlike β-VAEs, which as injective models with a fixed bottleneck find only a handful of high-entropy features, the bijective EOFlows discover a rich set of orthogonal, sparse and reproducible features, an order of magnitude more than tuned β-VAE variants or nested-dropout flows, while remaining competitive on standard disentanglement benchmarks (Table 1).

A fundamental hurdle on more complex data is topological mismatch: struggle to map disconnected components (e.g. the classes of MNIST or ImageNet) to a unimodal Gaussian, because the Jacobian determinant must diverge between them (Loaiza-Ganem et al., 2024), whereas our regularizer keeps the coordinate grid, via its Jacobian, smooth and orthogonal. EOFlows therefore do not yet disentangle multi-modal data globally; CelebA, with its single class “face”, is largely benign in this respect.

AI USE STATEMENT

In this work we used a large language model (Claude, Anthropic) as a coding and writing assistant: for implementing evaluation scripts and figure code, for editing and condensing the text, and for literature search and organizing the related-work discussion. We did not use generative AI tools to formulate the mathematical claims, to design the experiments, or to interpret the results; the method, the proofs and the experimental design are the authors' own. All AI-generated code was reviewed and tested by the authors, all AI-suggested references were checked against the original sources, and every AI-edited passage was reviewed by the authors. We take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The architecture, data preparation and training settings are specified in Appendix F, the stochastic estimator of the regularizer with a minimal PyTorch implementation in Appendix B, and the exact definitions of the orthogonality, sparsity and stability metrics in Appendix F.7. All proofs are given in Appendices A and C. Source code and configuration files will be released upon publication.

ETHICS STATEMENT

This work studies unsupervised representation learning on the public CelebA dataset of celebrity face images under its research license. The learned features are used only for the analysis of the method; no identification, attribute inference or generation of images of real individuals beyond the dataset is performed, and we see no ethical concerns beyond those inherent to the dataset.

REFERENCES

Lynton Ardizzone, Till Bungert, Felix Draxler, Ullrich Köthe, Jakob Kruse, Robert Schmier, and Peter Sorrenson. Framework for Easily Invertible Architectures (FrEIA). `https://github.com/vislearn/FrEIA`, 2018-2022.

Artur Bekasov and Iain Murray. Ordering dimensions with nested dropout normalizing flows. *arXiv:2006.08777*, 2020.

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

Johann Brehmer and Kyle Cranmer. Flows for simultaneous manifold learning and density estimation. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 442–453. Curran Associates, Inc., 2020.

Simon Buchholz, Michel Besserve, and Bernhard Schölkopf. Function classes for identifiable nonlinear independent component analysis. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 16946–16961. Curran Associates, Inc., 2022.

Chris Burgess and Hyunjik Kim. 3D shapes dataset. `https://github.com/deepmind/3dshapes-dataset/`, 2018.

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

Anthony L Caterini, Gabriel Loaiza-Ganem, Geoff Pleiss, and John P Cunningham. Rectangular flows for manifold learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 30228–30241. Curran Associates, Inc., 2021.

Ricky T. Q. Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Tianrong Chen, Jiatao Gu, David Berthelot, Joshua Susskind, and Shuangfei Zhai. Normalizing flows with iterative denoising. arXiv:2604.20041, 2026.

Edmond Cunningham, Adam D Cobb, and Susmit Jha. Principal component flows. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 4492–4519. PMLR, 17–23 Jul 2022.

Aaron Defazio, Xingyu Yang, Harsh Mehta, Konstantin Mishchenko, Ahmed Khaled, and Ashok Cutkosky. The road less scheduled. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 9974–10007. Curran Associates, Inc., 2024. doi: 10.52202/079017-0320.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. In *International Conference on Learning Representations*, 2017.

Yann Dubois, Alexandros Kastanos, Dave Lines, and Bart Melman. Disentangling VAE. `http://github.com/YannDubs/disentangling-vae/`, march 2019.

Emilien Dupont. Learning disentangled joint continuous and discrete representations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Cian Eastwood and Christopher K. I. Williams. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018.

Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011. ISSN 01621459. URL `http://www.jstor.org/stable/23239562`.

Daniel Galperin and Ullrich Koethe. Analyzing generative models by manifold entropic metrics. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025.

Shubhangi Ghosh, Luigi Gresele, Julius von Kügelgen, Michel Besserve, and Bernhard Schölkopf. Independent mechanism analysis and the manifold hypothesis. In *Causal Representation Learning Workshop at NeurIPS 2023*, 2023.

Luigi Gresele, Julius Von Kügelgen, Vincent Stimper, Bernhard Schölkopf, and Michel Besserve. Independent mechanism analysis, a new concept? In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021.

Jiatao Gu, Tianrong Chen, David Berthelot, Huangjie Zheng, Yuyang Wang, Ruixiang ZHANG, Laurent Dinh, Miguel Ángel Bautista, Joshua M. Susskind, and Shuangfei Zhai. STARFlow: Scaling latent normalizing flows for high-resolution image synthesis. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Haoyu He, Hao Wang, Jiashan Wang, and Hao Zeng. Effective sparsity: A unified framework via normalized entropy and the effective number of nonzeros, 2026. URL `https://arxiv.org/abs/2603.13826`.

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.

Christian Horvat and Jean-Pascal Pfister. Denoising normalizing flow. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 9099–9111. Curran Associates, Inc., 2021.

Christian Horvat and Jean-Pascal Pfister. Intrinsic dimensionality estimation using normalizing flows. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 12225–12236. Curran Associates, Inc., 2022.

Christian Horvat and Jean-Pascal Pfister. Density estimation on low-dimensional manifolds: an inflation-deflation approach. *Journal of Machine Learning Research*, 24(61):1–37, 2023.

Aapo Hyvärinen and Petteri Pajunen. Nonlinear independent component analysis: Existence and uniqueness results. *Neural Networks*, 12(3):429–439, 1999. ISSN 0893-6080. doi: https://doi.org/10.1016/S0893-6080(98)00140-3.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2649–2658. PMLR, 10–15 Jul 2018.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Shen Li and Bryan Hooi. Neural PCA for flow-based representation learning. arXiv:2208.10753, 2022.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Gabriel Loaiza-Ganem, Brendan Leigh Ross, Jesse C Cresswell, and Anthony L. Caterini. Diagnosing and fixing manifold overfitting in deep generative models. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. Expert Certification.

Gabriel Loaiza-Ganem, Brendan Leigh Ross, Luhuan Wu, John Patrick Cunningham, Jesse C. Cresswell, and Anthony L. Caterini. Denoising deep generative models. In Javier Antorán, Arno Blaas, Fan Feng, Sahra Ghalebikesabi, Ian Mason, Melanie F. Pradier, David Rohde, Francisco J. R. Ruiz, and Aaron Schein (eds.), *Proceedings on "I Can't Believe It's Not Better! – Understanding Deep Learning Through Empirical Falsification" at NeurIPS 2022 Workshops*, volume 187 of *Proceedings of Machine Learning Research*, pp. 41–50. PMLR, 03 Dec 2023.

Gabriel Loaiza-Ganem, Brendan Leigh Ross, Rasa Hosseinzadeh, Anthony L. Caterini, and Jesse C. Cresswell. Deep generative models through the lens of the manifold hypothesis: A survey and new connections. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. Survey Certification, Expert Certification.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4114–4124, 2019.

Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dSprites: Disentanglement testing sprites dataset. `https://github.com/deepmind/dsprites-dataset/`, 2017.

Walter Nelson, Marco Fumero, Theofanis Karaletsos, and Francesco Locatello. Statistical and structural identifiability in representation learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

William Peebles, John Peebles, Jun yan Zhu, Alexei A. Efros, and Antonio Torralba. The Hessian penalty: A weak prior for unsupervised disentanglement. In *Proceedings of (ECCV) European Conference on Computer Vision*, pp. 581 – 597, August 2020.

Patrik Reizinger, Luigi Gresele, Jack Brady, Julius von Kügelgen, Dominik Zietlow, Bernhard Schölkopf, Georg Martius, Wieland Brendel, and Michel Besserve. Embrace the gap: VAEs perform independent mechanism analysis. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 12040–12057. Curran Associates, Inc., 2022.

Oren Rippel, Michael Gelbart, and Ryan Adams. Learning ordered representations with nested dropout. In Eric P. Xing and Tony Jebara (eds.), *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 1746–1754, Bejing, China, 22–24 Jun 2014. PMLR.

Michal Rolinek, Dominik Zietlow, and Georg Martius. Variational autoencoders pursue PCA directions (by accident). In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Brendan Leigh Ross, Hamidreza Kamkari, Zhaoyan Liu, Tongzi Wu, George Stein, Gabriel Loaiza-Ganem, and Jesse C. Cresswell. A geometric framework for understanding memorization in generative models. In *ICML 2024 Next Generation of AI Safety Workshop*, 2024.

Peter Sorrenson, Carsten Rother, and Ullrich Köthe. Disentanglement by nonlinear ICA with general incompressible-flow networks (GIN). In *International Conference on Learning Representations*, 2020.

Peter Sorrenson, Felix Draxler, Armand Rousselot, Sander Hummerich, Lea Zimmermann, and Ullrich Koethe. Lifting architectural constraints of injective flows. In *The Twelfth International Conference on Learning Representations*, 2024.

Xin Wang, Hong Chen, Si'ao Tang, Zihao Wu, and Wenwu Zhu. Disentangled representation learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9677–9696, 2024.

Yuxiang Wei, Yupeng Shi, Xiao Liu, Zhilong Ji, Yuan Gao, Zhongqin Wu, and Wangmeng Zuo. Orthogonal Jacobian regularization for unsupervised disentanglement in image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6721–6730, October 2021.

Eric Yeats, Frank Liu, David Womble, and Hai Li. NashAE: Disentangling representations through adversarial covariance minimization. In *European Conference on Computer Vision (ECCV)*, 2022.

Shuangfei Zhai, Ruixiang ZHANG, Preetum Nakkiran, David Berthelot, Jiatao Gu, Huangjie Zheng, Tianrong Chen, Miguel Ángel Bautista, Navdeep Jaitly, and Joshua M. Susskind. Normalizing flows are capable generative models. In *Forty-second International Conference on Machine Learning*, 2025.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. InfoVAE: Information maximizing variational autoencoders. arXiv:1706.02262, 2018.

A STOCHASTIC AND GEOMETRIC DISENTANGLEMENT ARE DISTINCT CONCEPTS

The term "disentanglement" intuitively refers to representations where each feature has a distinct semantic meaning that is largely independent of the values of the other features, and it can be formalized in at least two ways. A representation is *statistically* (or stochastically) *disentangled* if the generated distribution factorizes, i.e. can be written as a product of single-variable functions $q\big(\boldsymbol{X}=\boldsymbol{g}(\boldsymbol{z})\big) = \prod_{i=1}^{D} \rho_i(z_i)$. When $\boldsymbol{g}(\boldsymbol{z})$ is a NF with separable latent prior, this is achieved if the determinant of the pull-back metric factorizes into a product of 1-dimensional functions

$$\det \gamma(\boldsymbol{z}) = \prod_{i=1}^{D} \psi_i(z_i) \tag{A.1}$$

whereas geometric disentanglement requires the metric to be diagonal

$$\gamma(\boldsymbol{z}) = \operatorname{diag}\big(\gamma(\boldsymbol{z})_{11}, ..., \gamma(\boldsymbol{z})_{DD}\big) \tag{A.2}$$

If all diagonal elements of the metric are equal, $\gamma(\boldsymbol{z})_{11} = ... = \gamma(\boldsymbol{z})_{DD}$, one speaks of a *conformal* map. When the orthogonal coordinates are suitably oriented, the nonlinear map resembles a localized PCA and de-correlates the features in a neighborhood of each point. We present three simple examples to show that maps can fulfill either of these conditions individually or both simultaneously. Consider 2-dimensional polar coordinates with $\boldsymbol{z} = [r, \phi]$ and decoder

$$x_1 = r\,\cos(\phi) \qquad\qquad x_2 = r\,\sin(\phi) \tag{A.3}$$

Calculation of its metric shows that polar coordinates fulfill both conditions

$$\gamma_{\text{polar}}(\boldsymbol{z}) = \begin{pmatrix} 1 & 0 \\ 0 & r^2 \end{pmatrix} \qquad\qquad \det \gamma_{\text{polar}}(\boldsymbol{z}) = r^2 \tag{A.4}$$

Parabolic coordinates are defined by the decoder

$$x_1 = \frac{z_1^2 - z_2^2}{2} \qquad\qquad x_2 = z_1 z_2 \tag{A.5}$$

The metric is also diagonal (indeed even conformal), but the determinant does not factorize

$$\gamma_{\text{parabolic}}(\boldsymbol{z}) = \big(z_1^2 + z_2^2\big) \cdot \mathbb{I} \qquad\qquad \det \gamma_{\text{parabolic}}(\boldsymbol{z}) = \big(z_1^2 + z_2^2\big)^2 \tag{A.6}$$

Finally, the decoder of log-linear coordinates is

$$x_1 = e^{z_1 + z_2} \qquad\qquad x_2 = e^{-z_1 + z_2} \tag{A.7}$$

Here, the metric is not diagonal, but the determinant factorizes nonetheless

$$\gamma_{\text{loglin}}(\boldsymbol{z}) = 2\,e^{2z_2} \begin{pmatrix} \cosh(2z_1) & \sinh(2z_1) \\ \sinh(2z_1) & \cosh(2z_1) \end{pmatrix} \qquad\qquad \det \gamma_{\text{loglin}}(\boldsymbol{z}) = 4\,e^{4z_2} \tag{A.8}$$

These examples demonstrate that *neither form of disentanglement implies the other*.

Tolerable ambiguities. Structural identifiability (Nelson et al., 2026) asks whether the learned features reproduce the independent factors of the true generative process (TGP), statistical identifiability only whether repeated training converges to the same features – a crucial relaxation when the TGP is unknown. Which ambiguities are considered tolerable depends on the application. Most commonly, one allows permutations and element-wise reparameterizations of the latent space – it does not matter how the features are indexed and in what units they are measured. Fixing the latent prior as a standard normal therefore loses no generality: any other separable prior can be obtained by univariate inverse probability transforms.

B NUMERICAL ESTIMATION OF THE HADAMARD LOSS AND ITS COST

NF architectures are typically designed to output the log-determinant of the Jacobian along a forward pass, which makes maximum likelihood training tractable.

B.1 FULL LOSS ESTIMATE

The EOFlow objective reads

$$\mathcal{L}_{\text{EO}}(\boldsymbol{x}) = \mathcal{L}_{\text{ML}}(\boldsymbol{x}) + \lambda \cdot \mathcal{L}_{\text{H}}(\boldsymbol{x}) \tag{B.1}$$

with

$$\mathcal{L}_{\text{H}}(\boldsymbol{x}) = \sum_{i=1}^{D} \log \left\| \boldsymbol{J}_i(\boldsymbol{f}(\boldsymbol{x})) \right\|_2 - \log \left| \boldsymbol{J}(\boldsymbol{f}(\boldsymbol{x})) \right| \tag{B.2}$$

$$= \sum_{i=1}^{D} \frac{1}{2} \log \left(\sum_{k=1}^{D} \boldsymbol{J}_{ki}(\boldsymbol{f}(\boldsymbol{x}))^2 \right) - \log \left| \boldsymbol{J}(\boldsymbol{f}(\boldsymbol{x})) \right| \tag{B.3}$$

The log-determinant $\log \left| \boldsymbol{J}(\boldsymbol{f}(\boldsymbol{x})) \right|$ is already available from the likelihood term, so the cost lies in the D column norms $\|\boldsymbol{J}_i\|_2$: computing them exactly requires the full $D \times D$ Jacobian, i.e. D Jacobian-vector products (jvp) per sample, one for each unit vector $\hat{\boldsymbol{e}}_i$.

B.2 STOCHASTIC ESTIMATE USED IN ALL EXPERIMENTS

For a batch $\{\boldsymbol{x}^j\}_{j=1}^B$ of any size B, with latent codes $\boldsymbol{z}^j = \boldsymbol{f}(\boldsymbol{x}^j)$, we draw one column index per sample, $i_j \sim \mathcal{U}\{1, \ldots, D\}$, independently across samples and of the data, and evaluate a single jvp per sample, $\boldsymbol{J}_{i_j}(\boldsymbol{z}^j) = \texttt{jvp}(\boldsymbol{g}, \boldsymbol{z}^j, \hat{\boldsymbol{e}}_{i_j})$; the B jvps form one batched forward-mode pass through the decoder. The batch estimate of the Hadamard loss is

$$\widehat{\mathcal{L}}_{\text{H}}\big(\boldsymbol{x}^{1:B}\big) = \frac{1}{B} \sum_{j=1}^{B} \Big(D \, \log \left\| \boldsymbol{J}_{i_j}(\boldsymbol{z}^j) \right\|_2 - \log \left| \boldsymbol{J}(\boldsymbol{z}^j) \right| \Big), \tag{B.4}$$

and the training loss is $\frac{1}{B} \sum_j \mathcal{L}_{\text{ML}}(\boldsymbol{x}^j) + \lambda \, \widehat{\mathcal{L}}_{\text{H}}(\boldsymbol{x}^{1:B})$, so λ enters only once. As in the unbiased estimates of Cunningham et al. (2022), one random column per sample suffices; we write the weight D explicitly so that λ multiplies the full sum over columns.

Unbiasedness. Since $\mathbb{E}_{i_j}\big[D \log \|\boldsymbol{J}_{i_j}(\boldsymbol{z}^j)\|_2\big] = \sum_{i=1}^{D} \log \|\boldsymbol{J}_i(\boldsymbol{z}^j)\|_2$, the expectation of (B.4) over the index draws is the exact batch loss $\frac{1}{B} \sum_j \mathcal{L}_{\text{H}}(\boldsymbol{x}^j)$ for every batch, whatever its size B relative to D. The index distribution does not depend on the network parameters, so the same holds for the gradient. Non-negativity on average follows from unbiasedness: the expectation is the exact loss, which is non-negative by Hadamard's inequality, although a single estimate can be negative. A batch need not contain every column. The indices are drawn with replacement, so the same column can be drawn for several samples of a batch while other columns are not drawn at all; this affects the variance but not the expectation, and no reweighting by the number of times a column was drawn is required.

Variance. The estimate needs B jvps per step, whatever D is (their cost is measured in App. B.3). The price is variance: conditionally on the batch, the variance of (B.4) is $\frac{D^2}{B^2} \sum_j \mathrm{Var}_i\big[\log \|\boldsymbol{J}_i(\boldsymbol{z}^j)\|_2\big]$. It is set by how much the log column norms differ across i – essentially the width of the entropy spectrum, which entropy ordering makes large – and not by λ, while in the training loss it is multiplied by λ^2. Fig. B.1 tests this on the training runs of the λ sweep (Fig. 6): the step-to-step variance of the logged estimate agrees with the formula, evaluated at the final weights of each run, to within $5\,\%$ in all 20 runs, and the variation of the exact loss between batches adds at most $2\,\%$ (a). The noise is thus nearly independent of λ, but the penalty it estimates shrinks as the flow becomes orthogonal, so the relative variance of one step's estimate grows from 0.014 at $\lambda = 0.01$ to 0.8 at $\lambda = 1$ and 250–300 at $\lambda = 10$ (b), where a single estimate of $\mathcal{L}_{\text{H}}/D$ reads 0.002 ± 0.030. Multiplied by λ^2, the penalty's share of the per-step variance of the training loss overtakes that of the likelihood at $\lambda = 0.1$ and exceeds it about 80-fold at $\lambda = 1$ and 4000-fold at $\lambda = 10$ (c).

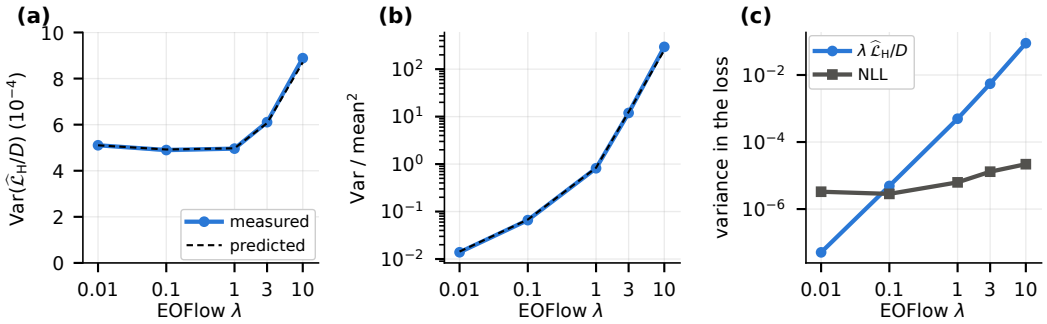


Figure B.1: **Variance of the stochastic Hadamard estimate** along the λ sweep (last 50 epochs of training; lines are means over 3–6 seeds per λ, shaded bands $\pm$ one standard deviation, mostly narrower than the line). **(a)** Per-step variance of the estimate of $\mathcal{L}_{\mathrm{H}}/D$ in the logged training loss (blue) and as predicted by the variance formula at the same weights, from full Jacobians at 64 noise-inflated training images (dashed). **(b)** The same variance relative to the squared mean, i.e. to the penalty itself. **(c)** Per-step variance of the two terms of the training loss.

Remark B.1 (Lower variance for $B \geq D$, optional). When $B \geq D$, the index draw can in addition be stratified so that every column enters each batch: D of the samples receive a random permutation of $1, \ldots, D$, the remaining $B - D$ samples draw their index uniformly, and each column is averaged over the $m_i \geq 1$ samples $S_i = \{j : i_j = i\}$ that drew it,

$$\widetilde{\mathcal{L}}_{\mathrm{H}}(\boldsymbol{x}^{1:B}) = \sum_{i=1}^{D} \frac{1}{m_i} \sum_{j \in S_i} \log \left\| \boldsymbol{J}_i(\boldsymbol{z}^j) \right\|_2 - \frac{1}{B} \sum_{j=1}^{B} \log \left| \boldsymbol{J}(\boldsymbol{z}^j) \right|. \tag{B.5}$$

Because the index pattern is drawn independently of the data, this estimate is unbiased as well, and it removes the spread across columns from the variance, leaving only the variation of each column across samples. On a synthetic Gaussian bed ($D = 32$, $B = 256$, eight trained 8-block flows, 200 batches at fixed weights) its standard deviation around the exact batch value is 24–380$\times$ lower than that of (B.4); the gap is smallest (24$\times$) for the most nonlinear flow. This refinement is an option, not a requirement: (B.4) is unbiased for every B, and all our experiments use it (on CelebA 28×28: $B = 512$, $D = 2352$).

B.3 Cost

Table B.1 times the estimator on two models: the fully connected CelebA model ($D = 2352$, 12 coupling blocks with learnable QLU mixings, 244.9M parameters) and a multiscale convolutional EOFlow for 64×64 CelebA images ($D = 12288$, 69.1M parameters; App. G.2). Both were timed in the same process on one GPU, with randomly initialized weights and random inputs (median of seven timings). The likelihood-only step comprises forward pass, loss, backward pass and optimizer update; the step with $\mathcal{L}_{\mathrm{H}}$ adds the batched jvp and the backward pass through it.

Table B.1: Wall-clock cost of the stochastic Hadamard estimate, $B = 128$ unless stated. The multiplier is the step with $\mathcal{L}_{\mathrm{H}}$ divided by the likelihood-only step. The training runs use $B = 512$ for the fully connected model and $B = 128$ for the convolutional one.

	fully connected $D = 2352$, 244.9M	convolutional $D = 12288$, 69.1M
decoder forward pass	12.3 ms	32.0 ms
decoder jvp (forward mode)	53.9 ms	137.2 ms
jvp / decoder forward pass	4.39$\times$	4.29$\times$
optimizer update alone	25.2 ms	7.6 ms
likelihood-only step	71.2 ms	98.5 ms
step with $\mathcal{L}_{\mathrm{H}}$	177.9 ms	415.1 ms
multiplier	**2.50**$\times$	**4.22**$\times$
multiplier at $B = 32/128/512$	2.61 / 2.50 / 2.49	
multiplier at $B = 32/128/256$		4.54 / 4.22 / 4.08

At $B = 128$ a forward-mode jvp costs about the same multiple of a decoder pass on both models ($4.39\times$ and $4.29\times$; 2.94–$4.74\times$ across the batch sizes of the table), so the type of layer does not explain why the step with $\mathcal{L}_{\mathrm{H}}$ is $4.22\times$ slower on the convolutional model but only $2.50\times$ on the fully connected one. The likely reason is that more of the fully connected step is parameter-bound work that $\mathcal{L}_{\mathrm{H}}$ leaves unchanged: its optimizer update alone takes 25.2 ms (35% of the likelihood-only step, against 7.6 ms or 8%), and its twelve 2352×2352 mixing matrices are rebuilt in every step. The two models differ in D, parameter count and architecture at once, so this is an accounting of the measured times, not a controlled comparison. Both multipliers decrease slightly as the batch grows. The learnable mixings themselves are a moderate share of the step: in a separate benchmark of the fully connected model (forward pass, loss and backward pass without the optimizer update, $B = 128$ and 256, exclusive GPU), making all twelve mixings learnable QLU matrices cost about 25 ms of a ≈ 145 ms step ($\approx 17\%$) compared with keeping all but the last one or three frozen at random orthogonal matrices, and 8–15 ms compared with eleven frozen dense mixings and a learnable last one. At the training batch size $B = 512$ this step takes ≈ 188 ms; the share of the mixings at that batch size was not measured.

B.4 PYTHON IMPLEMENTATION

Listing B.1: Minimal PyTorch implementation of the EOFlow loss with the stochastic estimate (B.4), as used in all our experiments

```
import math
import torch
from torch.autograd.forward_ad import dual_level, make_dual, unpack_dual

def eoflow_loss(flow, x, lam, sigma):
   # flow(x) -> (z, log|det df/dx|), flow(z, rev=True) -> (x, log|det dg/
       dz|)
   x = x + sigma * torch.randn_like(x) # noise inflation, [B, D]
   B, D = x.shape
   z, logdet_f = flow(x, rev=False)    # encoder
   nll = 0.5 * (z**2).sum(1) + 0.5 * D * math.log(2 * math.pi) - logdet_f

   i = torch.randint(0, D, (B,), device=x.device) # one column per sample
   e = torch.nn.functional.one_hot(i, D).to(z.dtype)
   with dual_level():                  # batched jvp, decoder
      x_dual, _ = flow(make_dual(z, e), rev=True)
      _, J_i = unpack_dual(x_dual)     # columns J_{i_j}(z_j)
   log_norm = 0.5 * torch.log((J_i.flatten(1)**2).sum(1) + 1e-12)

   L_H = D * log_norm + logdet_f       # log|det J| = -logdet_f
   return (nll + lam * L_H).sum() / (B * D) # per sample and dim
```

C Linearized Normalizing Flows and PCA

C.1 Setup

We assume a Linearized Normalizing Flow where we instantiate the decoder as an affine transformation

$$\boldsymbol{g}(\boldsymbol{z}) = A\boldsymbol{z} + b \tag{C.1}$$

$A \in \mathbb{R}^{D\times D}$ is an invertible "mixing" matrix and $b \in \mathbb{R}^D$ the "bias" term. The decoder Jacobian $\boldsymbol{J}$ is simply A.
For completeness, the encoder becomes the inverse of the affine transformation:

$$\boldsymbol{f}(\boldsymbol{x}) = A^{-1}\left(\boldsymbol{x} - b\right) \tag{C.2}$$

Assume that the data distribution has mean μ and covariance matrix Σ (of finite second moment). We assume that Σ has the following eigendecomposition

$$\Sigma = Q\boldsymbol{\Lambda}Q^T \tag{C.3}$$

where $Q = [Q_1, \ldots, Q_D]$ is an orthogonal matrix and the columns are orthonormal eigenvectors of Σ. The diagonal matrix $\boldsymbol{\Lambda} = \operatorname{diag}(\lambda_1, \ldots, \lambda_D)$ collects the eigenvalues, which we assume are ordered as

$$\lambda_1 > \lambda_2 > \cdots > \lambda_D > 0 \tag{C.4}$$

C.2 PCA

PCA can be viewed as a linear normalizing flow that matches the first two moments of the data. In particular, PCA first centers the data by choosing the bias to match the data mean

$$b \equiv \mu \tag{C.5}$$

It then aligns the latent coordinates with the principal directions Q_i and scales them according to the data variance λ_i. In our affine decoder setting, this corresponds to choosing

$$A \equiv Q\boldsymbol{\Lambda}^{\frac{1}{2}} \tag{C.6}$$

With this choice, $AA^T = Q\boldsymbol{\Lambda}Q^T = \Sigma$, i.e. the decoder maps a standard normal latent $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ to a Gaussian in data space with the same covariance as the data distribution. Equivalently, the encoder

$$\boldsymbol{f}(\boldsymbol{x}) = A^{-1}(\boldsymbol{x} - b) = \boldsymbol{\Lambda}^{-\frac{1}{2}}Q^T(\boldsymbol{x} - \mu) \tag{C.7}$$

is the usual PCA whitening transform: it rotates into the orthonormal eigenbasis and rescales each component by the inverse standard deviation $\lambda_i^{-1/2}$.

C.3 Convergence of Maximum-Likelihood

Let us write out the expectation of the ML objective

$$\mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\mathcal{L}_{\text{ML}}(\boldsymbol{x})\right] = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[-\log(q(\boldsymbol{X} = \boldsymbol{x}))\right] = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\frac{1}{2}\left|\boldsymbol{f}(\boldsymbol{x})\right|_2^2 + \log\left|\boldsymbol{J}(\boldsymbol{f}(\boldsymbol{x}))\right| + \frac{D}{2}\log(2\pi)\right] \tag{C.8}$$

Inserting the linear encoder/decoder definition, we abbreviate $\mathcal{L}_{\text{ML}}(A, b) \coloneqq \mathbb{E}_{\boldsymbol{x}\sim p^*}\left[\mathcal{L}_{\text{ML}}(\boldsymbol{x})\right]$ as a minimization objective under A and b:

$$\mathcal{L}_{\text{ML}}(A, b) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\frac{1}{2}\left(A^{-1}(\boldsymbol{x} - b)\right)^T\left(A^{-1}(\boldsymbol{x} - b)\right) + \log|A| + \frac{D}{2}\log(2\pi)\right] \tag{C.9}$$

$$= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\frac{1}{2}(\boldsymbol{x} - b)^T(AA^T)^{-1}(\boldsymbol{x} - b)\right] + \log|A| + \frac{D}{2}\log(2\pi) \tag{C.10}$$

$$= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\frac{1}{2}\operatorname{tr}\left((AA^T)^{-1}(\boldsymbol{x} - b)(\boldsymbol{x} - b)^T\right)\right] + \log|A| + \frac{D}{2}\log(2\pi) \tag{C.11}$$

$$= \frac{1}{2}\operatorname{tr}\left((AA^T)^{-1}\mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[(\boldsymbol{x} - b)(\boldsymbol{x} - b)^T\right]\right) + \log|A| + \frac{D}{2}\log(2\pi) \tag{C.12}$$

where

$$\mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[(\boldsymbol{x}-b)(\boldsymbol{x}-b)^T\right] = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\boldsymbol{x}\boldsymbol{x}^T - \boldsymbol{x}b^T - b\boldsymbol{x}^T + bb^T\right] = \Sigma + \mu\mu^T - \mu b^T - b\mu^T + bb^T \tag{C.13}$$

$$= \Sigma + (\mu - b)(\mu - b)^T \tag{C.14}$$

such that

$$\mathcal{L}_{\text{ML}}(A,b) = \frac{1}{2}\text{tr}\left((AA^T)^{-1}\left(\Sigma + (\mu-b)(\mu-b)^T\right)\right) + \log|A| + \frac{D}{2}\log(2\pi) \tag{C.15}$$

$$= \frac{1}{2}\text{tr}\left((AA^T)^{-1}\Sigma\right) + \frac{1}{2}\text{tr}\left((\mu-b)^T(AA^T)^{-1}(\mu-b)\right) + \log|A| + \frac{D}{2}\log(2\pi) \tag{C.16}$$

$$= \frac{1}{2}\left[\text{tr}\left((AA^T)^{-1}\Sigma\right) + (\mu-b)^T(AA^T)^{-1}(\mu-b) + 2\log|A| + D\log(2\pi)\right] \tag{C.17}$$

Theorem C.1. *The loss $\mathcal{L}_{ML}(A,b)$ attains its global minimum for $b=\mu$ and $AA^T=\Sigma$.*

Proof. Let $C \coloneqq AA^T \succ 0$ be a symmetric positive definite matrix. The loss is:

$$\mathcal{L}_{\text{ML}}(A,b) = \frac{1}{2}\left[\text{tr}\left(C^{-1}\Sigma\right) + (\mu-b)^TC^{-1}(\mu-b) + \log|C|\right] + \text{const.} \tag{C.18}$$

Since $C^{-1}\succ 0$, the quadratic term $(\mu-b)^TC^{-1}(\mu-b)$ is non-negative and vanishes only at $b=\mu$. Substituting into the loss,

$$\mathcal{L}_{\text{ML}}(A,b) = \frac{1}{2}\left[\text{tr}\left(C^{-1}\Sigma\right) + \log|C|\right] + \text{const.} \tag{C.19}$$

Let $M \coloneqq C^{-\frac{1}{2}}\Sigma C^{-\frac{1}{2}} \succ 0$. We can write $\text{tr}\,(M) = \text{tr}\left(C^{-1}\Sigma\right)$ and $\log|M| = \log|\Sigma| - \log|C|$ and substitute

$$\mathcal{L}_{\text{ML}}(A,b) = \frac{1}{2}\left[\text{tr}\,(M) - \log|M| + \log|\Sigma|\right] + \text{const.} \tag{C.20}$$

By the inequality

$$\text{tr}\,(M) - \log|M| \geq D \tag{C.21}$$

with equality iff $M = \boldsymbol{I}_D$, we obtain

$$\mathcal{L}_{\text{ML}}(A,b) \geq \frac{1}{2}\left[D + \log|\Sigma|\right] + \text{const.} \tag{C.22}$$

Thus equality holds iff

$$M = \boldsymbol{I}_D \iff C^{-\frac{1}{2}}\Sigma C^{-\frac{1}{2}} = \boldsymbol{I}_D \iff C = \Sigma \iff AA^T = \Sigma \tag{C.23}$$

□

C.4 Convergence of the Hadamard regularizer

Let us write the expectation of the pointwise manifold entropy of a single dimension i

$$\mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\mathcal{L}_i(\boldsymbol{x})\right] = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[-\log(q_i(\boldsymbol{X}_i=\boldsymbol{x}))\right] = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\frac{1}{2}\left|\boldsymbol{f}_i(\boldsymbol{x})\right|^2 + \log\left|\boldsymbol{J}_i(\boldsymbol{f}(\boldsymbol{x}))\right| + \frac{1}{2}\log(2\pi)\right] \tag{C.24}$$

As a prerequisite we need to evaluate the i-th entry of the encoder output:

$$\boldsymbol{f}_i(\boldsymbol{x}) = \left(A^{-1}\left(\boldsymbol{x}-b\right)\right)_i \tag{C.25}$$

which can be explicitly written by the SVD decomposition of A

$$A = USV^T, \quad A^{-1} = VS^{-1}U^T \tag{C.26}$$

The diagonal matrix S has only positive entries (convention without loss of generality). We denote the i-th row vector of V by $V_{i,*}$ such that

$$\boldsymbol{f}_i(\boldsymbol{x}) = V_{i,*}S^{-1}U^T\left(\boldsymbol{x}-b\right) \tag{C.27}$$

The i-th column vector of the decoder Jacobian $\boldsymbol{J}_i$ is the i-th column vector of A, which we denote as $(A)_{*,i} \coloneqq A_i$:

$$A_i = (USV^T)_{*,i} = US(V_{i,*})^T \tag{C.28}$$

For shorter notation we will abbreviate the i-th row vector $V_{i,*}$ simply as V_i.

Inserting the linear encoder/decoder definition, we abbreviate $\mathcal{L}_i(A,b) \coloneqq \mathbb{E}_{\boldsymbol{x}\sim p^*}\left[\mathcal{L}_i(\boldsymbol{x})\right]$ as a minimization objective under A and b:

$$\mathcal{L}_i(A,b) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\frac{1}{2}\left((\boldsymbol{f}_i(\boldsymbol{x}))^T(\boldsymbol{f}_i(\boldsymbol{x}))\right) + \log|A_i| + \frac{1}{2}\log(2\pi)\right] \tag{C.29}$$

$$= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\frac{1}{2}(V_iS^{-1}U^T(\boldsymbol{x}-b))^T(V_iS^{-1}U^T(\boldsymbol{x}-b)) + \log\left|USV_i^T\right| + \frac{1}{2}\log(2\pi)\right] \tag{C.30}$$

$$= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\frac{1}{2}(\boldsymbol{x}-b)^TUS^{-1}V_i^TV_iS^{-1}U^T(\boldsymbol{x}-b)\right] + \frac{1}{2}\log\left|\det((USV_i^T)^T(USV_i^T))\right| + \frac{1}{2}\log(2\pi) \tag{C.31}$$

$$= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\frac{1}{2}\mathrm{tr}\left(US^{-1}V_i^TV_iS^{-1}U^T(\boldsymbol{x}-b)(\boldsymbol{x}-b)^T\right)\right] + \frac{1}{2}\log\left|\det(V_iS^2V_i^T)\right| + \frac{1}{2}\log(2\pi) \tag{C.32}$$

$$= \frac{1}{2}\mathrm{tr}\left(US^{-1}V_i^TV_iS^{-1}U^T\mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[(\boldsymbol{x}-b)(\boldsymbol{x}-b)^T\right]\right) + \frac{1}{2}\log\left|\det(V_iS^2V_i^T)\right| + \frac{1}{2}\log(2\pi) \tag{C.33}$$

Inserting the solution eq.(C.13)

$$\mathcal{L}_i(A,b) = \frac{1}{2}\left[\mathrm{tr}\left(US^{-1}V_i^TV_iS^{-1}U^T\left(\Sigma + (\mu-b)(\mu-b)^T\right)\right) + \log\left|\det(V_iS^2V_i^T)\right| + \log(2\pi)\right] \tag{C.34}$$

Per sample, the prior terms cancel in the difference of the two losses, so the Hadamard regularizer (6) is $\mathcal{L}_{\mathrm{H}}(\boldsymbol{x}) = \sum_i \mathcal{L}_i(\boldsymbol{x}) - \mathcal{L}_{\mathrm{ML}}(\boldsymbol{x})$, and its expectation is

$$\mathcal{L}_{\mathrm{H}}(A,b) \coloneqq \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p^*}\left[\mathcal{L}_{\mathrm{H}}(\boldsymbol{x})\right] = \sum_i \mathcal{L}_i(A,b) - \mathcal{L}_{\mathrm{ML}}(A,b) \tag{C.35}$$

Inserting the previous results

$$\mathcal{L}_{\mathrm{H}}(A,b) = \frac{1}{2}\sum_{i=1}^{D}\Big[\mathrm{tr}\left(US^{-1}V_i^TV_iS^{-1}U^T\left(\Sigma + (\mu-b)(\mu-b)^T\right)\right) + \log\left|\det(V_iS^2V_i^T)\right| + \log(2\pi)\Big] \tag{C.36}$$

$$- \frac{1}{2}\left[\mathrm{tr}\left((AA^T)^{-1}\left(\Sigma + (\mu-b)(\mu-b)^T\right)\right) + 2\log|A| + D\log(2\pi)\right] \tag{C.37}$$

Inserting the SVD decomposition of A, we can simplify the determinant of A

$$|\det(A)| = \det(S) = \prod_i s_i \tag{C.38}$$

where s_i are the strictly positive singular values. We can also simplify the second determinant expression by observing that $V_i S^2 V_i^T$ is a scalar and strictly positive.

$$\mathcal{L}_{\mathrm{H}}(A,b) = \frac{1}{2}\sum_{i=1}^{D}\Big[\mathrm{tr}\left(US^{-1}V_i^T V_i S^{-1}U^T\left(\Sigma+(\mu-b)(\mu-b)^T\right)\right) + \log\left(V_i S^2 V_i^T\right) + \log(2\pi)\Big] \tag{C.39}$$

$$- \frac{1}{2}\left[\mathrm{tr}\left(US^{-2}U^T\left(\Sigma+(\mu-b)(\mu-b)^T\right)\right) + 2\log\left(\prod_{i=1}^{D} s_i\right) + D\log(2\pi)\right] \tag{C.40}$$

$$= \frac{1}{2}\mathrm{tr}\left(\Big[\sum_{i=1}^{D} US^{-1}V_i^T V_i S^{-1}U^T - US^{-2}U^T\Big]\left(\Sigma+(\mu-b)(\mu-b)^T\right)\right) \tag{C.41}$$

$$+ \left[\frac{1}{2}\sum_{i=1}^{D}\log(V_i S^2 V_i^T) - \log\left(\prod_{i=1}^{D} s_i\right)\right] \tag{C.42}$$

$$= \frac{1}{2}\mathrm{tr}\left(US^{-1}\underbrace{\Big[\sum_{i=1}^{D} V_i^T V_i - \boldsymbol{I}_D\Big]}_{\equiv 0} S^{-1}U^T\left(\Sigma+(\mu-b)(\mu-b)^T\right)\right) \tag{C.43}$$

$$+ \sum_{i=1}^{D}\left[\frac{1}{2}\log(V_i S^2 V_i^T) - \log(s_i)\right] \tag{C.44}$$

$$= \frac{1}{2}\sum_{i=1}^{D}\log\left(\frac{V_i S^2 V_i^T}{s_i^2}\right) \tag{C.45}$$

Since V is orthogonal, $\sum_i V_i^T V_i = V^T V = \boldsymbol{I}_D$, and the trace term vanishes.

Theorem C.2. *The loss $\mathcal{L}_H(A,b)$ attains its global minimum 0 if V is a permutation matrix up to signs and, if the singular values of A are distinct, only then.*

Proof. The loss is:

$$\mathcal{L}_{\mathrm{H}}(A,b) = \frac{1}{2}\sum_{i=1}^{D}\log\left(\frac{V_i S^2 V_i^T}{s_i^2}\right) \tag{C.46}$$

Since V is orthogonal, the coefficients V_{ij}^2 form a probability distribution for each i. By Jensen's inequality applied to the concave function log:

$$\log(V_i S^2 V_i^T) = \log\left(\sum_j V_{ij}^2 s_j^2\right) \geq \sum_j V_{ij}^2 \log s_j^2 \tag{C.47}$$

Summing over i and using $\sum_i V_{ij}^2 = 1$ yields

$$\sum_i \log(V_i S^2 V_i^T) \geq \sum_i \log s_i^2, \tag{C.48}$$

i.e. $\mathcal{L}_{\mathrm{H}}(A,b) \geq 0$. If V is a permutation matrix up to signs, then $V_i S^2 V_i^T = s_{\pi(i)}^2$ for a permutation π, and equality holds. Conversely, since the logarithm is strictly concave, equality requires s_j^2 to be constant on the support of every row distribution V_{ij}^2; for distinct singular values, every row of V then has exactly one non-zero entry, ± 1. □

Corollary C.3. *For every $\lambda > 0$, the expected loss $\mathcal{L}_{ML}(A,b) + \lambda\,\mathcal{L}_H(A,b)$ of a linear EOFlow attains its global minimum exactly at $b = \mu$ and $A = Q\boldsymbol{\Lambda}^{\frac{1}{2}}P$, where P is a permutation matrix up to signs, i.e. at the PCA solution up to the tolerable ambiguities.*

Proof. $A = Q\mathbf{\Lambda}^{\frac{1}{2}}P$ has the SVD $U = Q$, $S = \mathbf{\Lambda}^{\frac{1}{2}}$, $V = P^T$ and thus minimizes both terms at once (Theorems C.1 and C.2). Hence the minimum of the sum is the sum of the two minima, and every minimizer of the sum minimizes both terms. Then $b = \mu$ and $AA^T = US^2U^T = \Sigma$, so the singular values of A are the square roots of the distinct eigenvalues of Σ, and V is a permutation matrix up to signs by Theorem C.2. The columns of $A = USV^T$ are therefore $\pm s_j U_j$, and $US^2U^T = Q\mathbf{\Lambda}Q^T$ with distinct eigenvalues implies $U_j = \pm Q_k$ and $s_j^2 = \lambda_k$ for a permutation $j \mapsto k$. □

Sorting the latent dimensions by decreasing s_i gives the PCA decoder $A = Q\mathbf{\Lambda}^{\frac{1}{2}}$ up to signs: the latent axes are the principal directions, scaled by their standard deviations.

C.5 Manifold Entropic Metrics in the linear regime

In the linear setting one can compute manifold entropic metrics analytically. The manifold entropy of a single latent dimension i reads:

$$H_i = \frac{1}{2}(1 + \log(2\pi)) + \mathop{\mathbb{E}}_{\boldsymbol{z}\sim p(\boldsymbol{z})}[\log \|\boldsymbol{J}_i(\boldsymbol{z})\|_2] = \frac{1}{2}(1 + \log(2\pi)) + \log \|A_i\|_2 \tag{C.49}$$

$$= \frac{1}{2}(1 + \log(2\pi)) + \frac{1}{2}\log\left|\det(V_i S^2 V_i^T)\right| \tag{C.50}$$

For PCA, $V = \boldsymbol{I}_D$ and this simplifies to

$$H_i = \frac{1}{2}(1 + \log(2\pi)) + \log(s_i) \tag{C.51}$$

This shows that for PCA, sorting by explained variance or manifold entropy is equivalent.

D NOISE INFLATION

Throughout the paper, every training image is inflated with fresh isotropic Gaussian noise of standard deviation $\sigma_\epsilon = 0.05$, redrawn at every step. This appendix explains why the noise is needed (App. D.1), what happens with less of it on CelebA (App. D.2), which trends a grid of smaller models and a Gaussian bed with known axes show (App. D.3 and D.4), and why the coordinates that model the noise cannot be reproduced across runs (App. D.5).

D.1 WHY NOISE IS NEEDED

Memorization. With finite data, maximum likelihood rewards a density concentrated on the training points. A finite training set is a zero-dimensional manifold on which the likelihood is unbounded, so a flexible flow memorizes the training set unless something prevents it (Sec. 3.5).

Noise as a kernel density estimate. Because the noise is redrawn at every step, training on noise-inflated images is formally the same as fitting the flow to the Gaussian kernel density estimate $\frac{1}{N}\sum_n \mathcal{N}(\boldsymbol{x}_n, \sigma_\epsilon^2 \boldsymbol{I}_D)$ of the training set. Its density is smooth and bounded, so the likelihood of the flow cannot diverge on the training points. The identity does not prescribe a noise level. We chose σ_ϵ empirically (App. D.2 and D.3), and more noise is not better, since it hides every direction of the data whose variance lies below it.

A regularizer without a preferred basis. The usual remedies against memorization are less capacity or architectural biases, and the latter, such as convolutions, presuppose the pixel basis. Isotropic noise and the Hadamard penalty presuppose no basis: both are unchanged by a rotation of the data space, so the flow can stay fully connected and over-parametrized, as the experiments in a shuffled or rotated pixel basis require (App. G.4). Recent normalizing flows for images likewise found noise augmentation essential and set its level empirically (Zhai et al., 2025).

D.2 LESS NOISE ON CELEBA

We trained the EOFlow of Sec. 4 ($\lambda = 1$, batch size 512, learning rate $6 \cdot 10^{-4}$, 300 epochs) three times at each of two lower noise levels and left everything else unchanged.

Training diverges. At $\sigma_\epsilon = 0.01$, two of the three runs diverged, after about 70 and 90 epochs, and the third completed all 300 epochs. At $\sigma_\epsilon = 0.002$, about twice the standard deviation of uniform dequantization noise for 8-bit images ($1/(256\sqrt{12}) \approx 0.0011$), all three runs diverged within the first 34 epochs. At $\sigma_\epsilon = 0.05$, one of the eight runs we trained diverged (App. F.2).

The surviving run overfits. We compare the run that completed at $\sigma_\epsilon = 0.01$ with the three runs at $\sigma_\epsilon = 0.05$ after the same 300 epochs (mean $\pm$ standard deviation over the three). Its gap between held-out and training likelihood is 0.214 nats per dimension against $0.035_{\pm.002}$, and it grows throughout training (0.050, 0.103, 0.141, 0.170, 0.194 and 0.214 after 50, 100, …, 300 epochs), while its held-out likelihood is best after 100 epochs and declines afterwards. Its decoder is also less orthogonal, with a Hadamard gap $\mathcal{L}_\mathrm{H}/D$ of 0.040 against 0.014.

Its samples are worse. Through the best bottleneck ($C = 200$) its samples reach FID 27.5 against $24.0_{\pm 0.2}$, and from the full prior after a Tweedie step FID 44.7 against $34.7_{\pm 0.4}$.

Its features are shared less. It shares $33.3_{\pm 1.2}$ stable coordinates with each of the three runs at $\sigma_\epsilon = 0.05$, which share $121_{\pm 5}$ among themselves. Since no second run at $\sigma_\epsilon = 0.01$ finished, its reproducibility across runs of its own noise level cannot be measured.

Caveat. We did not tune the learning rate for the lower noise levels, so a smaller step size might stabilize them. The alternatives that make likelihood training robust by limiting what the network can fit, fewer parameters or convolutional layers, presuppose a scale or a pixel basis, which EOFlows avoid.

D.3 NOISE LEVELS ON A GRID OF SMALLER MODELS

Before settling on the architecture of Sec. 4, we swept the noise level from $\sigma_\epsilon = 0.01$ to 0.5 on a grid of smaller CelebA models (117M parameters, 100 epochs, $\lambda \in \{0.01, 0.1, 1\}$, five or six runs per

setting). Its absolute numbers are not comparable to the rest of the paper, but three trends informed our choice of $\sigma_\epsilon = 0.05$:

- **Noise and the penalty both close the train/test gap.** From $\sigma_\epsilon = 0.01$ to 0.1 the gap between held-out and training likelihood shrinks 13–26×, and at every noise level it is smaller for a larger λ (at $\sigma_\epsilon = 0.01$: 0.33, 0.28 and 0.13 nats per dimension for $\lambda = 0.01$, 0.1 and 1).
- **More noise does not make the features more stable.** At $\lambda = 0.1$ and $\lambda = 1$, the number of stable features decreases as the noise level grows. Noise is needed to bound the likelihood, not to stabilize the features.
- **Samples are best at an intermediate noise level.** At $\lambda = 1$ the best bottleneck samples reach FID 29.8 at $\sigma_\epsilon = 0.05$, against 34.1 at $\sigma_\epsilon = 0.01$ and 34.2 at $\sigma_\epsilon = 0.1$.

D.4 A Gaussian bed with known axes

To see the effect of the noise and the penalty where the answer is known, we trained small flows (8 coupling blocks with learnable mixings, 20,000 steps) on 32-dimensional Gaussian data whose variances fall geometrically from 1 to $1.3 \cdot 10^{-4}$ along the axes of a fixed random rotation. The principal axes of the data, and the number of axes whose variance exceeds any given noise level, are therefore known exactly. We varied the number of training points from 10^3 to 10^5, the noise level from 0.003 to 2 and the penalty over $\lambda \in \{0, 0.01, 0.1, 1, 10\}$, with three runs per setting. Two results carry over to CelebA:

- **The penalty prevents memorization as well.** With too little noise the flow memorizes its training points: with 10^4 points at $\sigma_\epsilon = 0.003$, the gap between held-out and training likelihood reaches 55.7 nats per dimension without the penalty, and 22.0, 4.8 and 0.47 at $\lambda = 0.1$, 1 and 10. The noise level below which the gap exceeds 0.01 nats per dimension accordingly falls with λ, from 0.17 at $\lambda = 0$ and $\lambda = 0.1$ to 0.14 at $\lambda = 1$ and 0.027 at $\lambda = 10$ (with 10^4 points), and it also falls with the number of training points. The noise a model needs thus depends on the model and the data size, not only on the geometry of the data.
- **The noise floor counts the axes above noise.** A coordinate carries more signal than noise if its variance, which includes the noise, exceeds twice the noise variance, i.e. if its manifold entropy exceeds $H_{\sigma_\epsilon} + \frac{1}{2}\log 2$. The number of such coordinates matches the true number of axes with variance above σ_ϵ^2 to within about one coordinate on average (0.12–1.33, depending on λ, with 10^4 points or more).

D.5 Unidentifiable detail coordinates

Why the detail cannot be reproduced. Isotropic noise is invariant under rotations, so coordinates that encode only noise can be rotated among themselves without changing the likelihood or the Hadamard penalty, and no training procedure can make them reproducible. Where the data have some variance below the noise level, the symmetry is broken only weakly.

Noisy PCA. Noisy PCA shows the consequence without any optimization error. Fitted on the training set augmented tenfold with independent noise draws (six fits, Fig. 5c), it reproduces $653_{\pm 9}$ coordinates per pair of fits, about the 657 principal components whose variance exceeds σ_ϵ^2. With 500 noise draws per image instead of 10 (drawn with an exact sampler of the augmented covariance), it reproduces $1252_{\pm 12}$: below the noise floor, the weakly broken symmetry is resolved only by averaging over many noise draws.

E Related work

E.1 Principal Component Flows

While Principal Component Flows (PCF) is an important precursor, EOFlows differ in three fundamental ways:

1. Scalability: PCF derive the contrast (their eq. (14) is our eq. (5)) and estimate it with one random Jacobian product per sample, as we do; their practical objective, however, uses *encoder rows* (vector-Jacobian products), whereas ours uses *decoder columns* (Jacobian-vector products). For a square Jacobian both vanish on the same set of orthogonal maps, but the two gaps differ away from the optimum, and only the column form is the quantity that enters the manifold entropies. We demonstrate the estimate at $D = 2352$ and $D = 12288$ (App. B); PCF's experiments stay below $D = 10$.
2. Noise Inflation: PCF add Gaussian noise with a fixed standard deviation of 0.01 in one low-dimensional (3-D) experiment, to keep the data full-rank and to stop the model from collapsing; they neither derive this level nor study how noise affects reproducibility across runs. We find that noise acts together with the Hadamard penalty as a guard against memorization (App. D).
3. Dimension ordering: PCF do not aim at disentanglement and propose no way to sort the latent dimensions, and thus lack the notions of feature importance and of a core/detail split.

E.2 NDFlows vs EOFlows

NDFlows impose an *active* sorting of the latent dimensions at training time: their reconstruction loss sets the detail latents to zero and thus acts on the latent distribution as a whole. The Hadamard regularizer of EOFlows is local and blind to permutations of the latent dimensions, so EOFlow latents are sorted *passively*, by their manifold entropy after training. It also does not force an individual latent to stay globally coherent across the data (see Fig. 1(B) of Galperin & Koethe (2025)), which poses no problem in the linear case (App. C). For a fair comparison, all models, NDFlows included, are evaluated with their latents sorted by decreasing manifold entropy, $H_i \geq H_{i+1}$; for NDFlows this changes little (App. F.5).

E.3 Topology of the data

We assume that CelebA consists of roughly one connected mode. A simple indication is whether the data mean is itself in distribution: the average of all CelebA faces resembles a face, whereas the average of all EMNIST digits resembles no digit. Data with several disconnected modes do not match the topology of the unimodal latent Gaussian, which limits the global disentanglement of EOFlows on such data (Sec. 5).

F EXPERIMENTAL SETUP

This appendix specifies the flow architecture (App. F.1), data and training (App. F.2 and F.3), the baselines (App. F.4–F.6) and the metrics (App. F.7); App. G collects additional results.

F.1 NORMALIZING FLOW ARCHITECTURE

Although our training objective can be applied to any invertible normalizing flow architecture, a few considerations motivate the architecture we use. In particular, one has to strike a balance between expressivity, unbiased-ness and training speed.

Recent autoregressive flows such as TarFlow and STARFlow (Zhai et al., 2025; Gu et al., 2025) are highly expressive at a comparable training time, but their decoder runs sequentially over the tokens of an image and is therefore slow. Since our estimator evaluates the decoder in every training step, autoregressive flows are impractical for EOFlows. We thus opt for a symmetric encoder-decoder design, in particular coupling layers.

As we are aiming to find features which are stemming from the data-generating process, we opt out of using convolutional layers as this would introduce a bias towards local-pixel correlations. Nonetheless a convolutional approach as in Kingma & Dhariwal (2018) is possible; App. G.2 tests one on 64×64 images.

Finally, Neural Spline Flows (NSF) (Durkan et al., 2019) is a family of highly expressive NF architectures, which allow learning highly complex low-dimensional distributions. This, however, comes at the cost of throughput time and makes training very slow. We thus opt for the less expressive, yet still effective, affine coupling layers introduced in Dinh et al. (2017).

All flows in this paper – EOFlows, NDFlows and the plain NF – use the same architecture. An ActNorm layer is followed by 12 affine coupling blocks (Dinh et al., 2017), implemented as the GLOW coupling blocks (Kingma & Dhariwal, 2018) of FrEIA (Ardizzone et al., 2018-2022). Each block updates both halves of its input with affine couplings whose scales and shifts are computed by fully connected MLPs with two hidden layers of width 1024 ($1176 \to 1024 \to 1024 \to 2352$, ReLU, output layer initialized at zero, scales soft-clamped at 2), and then applies a fixed random permutation, an ActNorm layer and a learnable $D \times D$ mixing matrix. The last block has no mixing of its own; a final learnable mixing follows it instead, so the flow contains 12 learnable mixings. Maximum likelihood is invariant to rotations of the latent space whereas the Hadamard regularizer is not, and the learnable mixings give the regularizer room to choose the latent basis; App. G.1 measures how much of this freedom is needed.

Parametrization of the mixings. Each mixing is $W = QLU$, where Q is a random orthogonal matrix that stays fixed, L is lower triangular with unit diagonal and U is upper triangular with diagonal entries $\exp(s_i)$. L and U are initialized at the identity, so every mixing starts as the random rotation Q. The log-determinant of a mixing is $\sum_i s_i$ and its inverse takes two triangular solves, so no determinant or matrix inverse is computed during training. The model has 245M parameters as stored; only the strict triangles of the stored $D \times D$ tensors for L and U are used, so 178M of them are effective – as many as in the same flow with dense learnable mixings.

Why this parametrization. We compared parametrizations of the mixings on a linear version of the flow, in which every coupling block is replaced by the identity, so that the optimum of the EOFlow loss is known in closed form: noisy PCA (App. C). On grayscale CelebA ($D = 784$, eight mixings, $\lambda = 1$, $\sigma_\epsilon = 0.05$, batch size 128, 10 epochs, three seeds), learnable dense and QLU mixings come within about 0.002 nats per dimension of the likelihood of this optimum (-1.0607 and -1.0608 against -1.0631). Two alternatives lag behind after the same budget: all but the final mixing frozen at their random orthogonal initialization (-1.0425), and a pivoted LU factorization of the same random orthogonal matrices with learnable offsets scaled by $1/\sqrt{D}$ (-1.0445). At $\lambda = 0.1$ these two arms reach -1.0611 and -1.0607, as close to the optimum as the learnable arms at $\lambda = 1$, and their difference to the learnable arms (-1.0624) shows mainly in the Hadamard gap $\mathcal{L}_\mathrm{H}/D$ (0.165 and 0.201 against 0.023 and 0.021 for dense and QLU). The LU parametrization fails because the LU factors of a random orthogonal matrix are ill-conditioned ($\|L\|_2 \approx 28$, $\|U\|_2 \approx 52$); QLU keeps that matrix as the fixed factor Q and learns well-conditioned factors that start at the identity. The

comparison concerns optimization within a short budget: after 10 epochs even the learnable arms align only about 20 of the 300 leading decoder columns with an eigenvector of the data covariance.

To summarize, we use a large fully connected architecture with fast encoder and decoder passes. We did not tune its depth or width for the final recipe.

F.2 Dataset and Training setup

We use the aligned CelebA images (Liu et al., 2015) (178×218 pixels), center-cropped to 178×178, cropped to a 112×112 face region, downsampled with Lanczos interpolation to $28 \times 28 \times 3$ ($D = 2352$) and scaled to $[0, 1]$:

Listing F.1: CelebA data preparation (torchvision)

```
from torchvision import transforms as T

CelebA_transform = T.Compose([
   T.CenterCrop(178),                                  # 178 x 178
   T.Lambda(lambda img: img.crop((33, 43, 33 + 112, 43 + 112))), # 112 x
       112 face
   T.Resize((28, 28), interpolation=T.InterpolationMode.LANCZOS),
   T.ToTensor(),                                       # float in [0, 1]
])
```

Data splits. All models are trained on the official CelebA training partition (162,770 images). Held-out likelihoods and reconstructions use a fixed subset of 10,000 images of the validation partition, and FID compares 10,000 generated or reconstructed images with the test partition (19,962 images).

Training. The flows are trained on noise-inflated images $\boldsymbol{x}_{\boldsymbol{\epsilon}} = \boldsymbol{x} + \sigma_{\boldsymbol{\epsilon}}\boldsymbol{\epsilon}$ with $\sigma_{\boldsymbol{\epsilon}} = 0.05$ and fresh noise in every batch, using schedule-free AdamW (Defazio et al., 2024) with batch size 512, learning rate $6 \cdot 10^{-4}$ and 100 warm-up steps; 500 epochs are 158,500 steps. EOFlows use $\lambda = 1$ unless stated otherwise and the stochastic estimator of the Hadamard regularizer (App. B). Batch size and learning rate were set in a smoke test with one seed and at most 23 epochs (batch sizes 128 to 1024, learning rates $3 \cdot 10^{-4}$ to $1.2 \cdot 10^{-3}$, the largest of which diverged during warm-up) and were not tuned further. The plain NF uses the same recipe with $\lambda = 0$ and the estimator switched off.

Noise inflation of the baselines. Noisy PCA is PCA of the noise-inflated training data, i.e. of the data covariance plus $\sigma_{\boldsymbol{\epsilon}}^2 \boldsymbol{I}_D$, which is the optimum of a linear EOFlow at the same noise level (App. C); it has the principal directions of the clean data, each with its variance increased by $\sigma_{\boldsymbol{\epsilon}}^2$. To measure its stability like that of the trained models (Table 1, Fig. 5c), we fit it six times on the training set augmented tenfold with independent noise draws (App. D.5). The VAEs are trained on clean images: a VAE is an injective model, whose decoder maps its 32 latents onto a low-dimensional manifold while its likelihood, here Bernoulli without a learnable scale, covers the remaining directions, so it does not suffer from the manifold overfitting of bijective flows that noise inflation prevents (Sec. 3.5). Moreover, the inflation–deflation argument is specific to likelihood-based flows under the manifold hypothesis.

Evaluation. All numbers are computed with the averaged weights of the schedule-free optimizer. Raw training iterates can reverse the ordering of two models: in a 100-epoch comparison at batch size 128, the raw iterates put frozen intermediate mixings ahead of learnable ones in held-out likelihood and $\mathcal{L}_{\mathrm{H}}$, and the averaged weights put the learnable ones ahead. Likelihoods are computed on the 10,000 held-out images with fixed inflation noise, $\mathcal{L}_{\mathrm{H}}/D$ exactly on 32 held-out images, entropy spectra on 512 prior samples, and stability on 256 fixed training images (App. F.7).

Hardware. Training ran on single NVIDIA RTX 4090 (24 GB) and A40 (48 GB) GPUs. An EOFlow epoch takes about 140 s, both alone on an A40 and with two runs sharing an RTX 4090; on an A40, an NDFlow epoch takes about 94 s and a plain-flow epoch about 59 s (medians of the logged epoch times).

Runs and seeds. Table F.1 lists the trained models behind the reported numbers. EOFlow $\lambda = 1$ seed 104 diverged in epoch 1, about 400 steps after the warm-up, and was replaced by seed 107; seed 107 was an outlier at 300 epochs (held-out NLL -1.2958 against -1.2976 to -1.2980 for the other three seeds, prior FID 28.1 against 24.7 to 24.9) and was not continued to 500 epochs. EOFlow

$\lambda = 3$ seed 105 diverged in epoch 0 and was replaced by seed 106. Two of eight NDFlow $\nu = 1$ runs diverged before 500 epochs (App. F.5). Fig. 1 shows features of EOFlow $\lambda = 1$ seed 105 and the appendix figures of a single model show seed 103, both at 300 epochs unless stated otherwise. AnnealedVAE was trained only for its tuning budget of 100 epochs; its entries in Table 1 and Fig. 5 refer to that budget.

Table F.1: Trained models behind the reported numbers (CelebA 28×28, $\sigma_\epsilon = 0.05$; flows at batch size 512 and learning rate $6 \cdot 10^{-4}$ unless noted). Cells at 300 and 500 epochs of the same λ or ν are the same trainings, evaluated at both lengths.

model	epochs	seeds	notes
EOFlow $\lambda = 1$	500	103, 105, 106, 108–110	
EOFlow $\lambda = 1$	300	103, 105, 106	
EOFlow $\lambda = 0.1$	500	103, 105, 106	
EOFlow $\lambda = 0.1$	300	103, 105, 106	snapshots after 301–302 epochs
EOFlow $\lambda = 0.01$	300	103, 105, 106	
EOFlow $\lambda = 3$	300	103, 106, 108, 109	
EOFlow $\lambda = 10$	300	103, 105, 108, 109	
plain NF ($\lambda = 0$)	300	103, 105, 106	
NDFlow $\nu = 1$	500	105, 106, 109–112	111, 112 replace two diverged runs
NDFlow $\nu = 1$	300	103, 105, 106	stability: 105, 106, 109, 110
NDFlow $\nu \in \{0.1, 10\}$	300	103, 105, 106	mixing learning rates: App. F.5
architecture ablation	100	301–303 per arm	App. G.1
shuffled pixel basis	100	101–104	batch size 128, learning rate $3 \cdot 10^{-4}$
rotated pixel basis	100	101, 102	batch size 128, learning rate $3 \cdot 10^{-4}$
β-VAE $\beta = 4$	300	101–103	
β-TCVAE $\beta = 4$	300	101–103	
FactorVAE $\gamma = 10$	300	101–103	batch size 256
FactorVAE $\gamma = 10$	500	101–106	batch size 256; stability (Fig. 5)
AnnealedVAE $C = 25$	100	101–105	its tuning budget
VAE tuning grid	100	101–103 per cell	FactorVAE: 200 epochs, batch size 256
five-seed tuning cells	100	104, 105	β-VAE $\beta = 4$, β-TCVAE $\beta = 3$, FactorVAE $\gamma = 10$, AnnealedVAE $C = 25$

F.3 Training data

Fig. F.1 shows 12 random training images, clean and inflated as the flows see them; the noise shows as a grain on every image. The flows are trained only on inflated images, with fresh noise in every batch; the VAEs are trained on clean images. Clean images also serve as the reference in evaluation, e.g. for FID. All image figures upscale the 28×28 images with Lanczos interpolation for display.



Figure F.1: Twelve random CelebA training images, clean (top) and inflated with Gaussian noise of $\sigma_\epsilon = 0.05$ as seen by the flows (bottom; one noise draw, redrawn at every training step).

F.4 VAE baselines

We compare with four VAE objectives of the study of Locatello et al. (2019): β-VAE (Higgins et al., 2017), β-TCVAE (Chen et al., 2018) with $\alpha = \gamma = 1$, FactorVAE (Kim & Mnih, 2018), and AnnealedVAE (Burgess et al., 2018) with $\gamma = 100$ and the capacity annealed from 0 to C over 10,000 steps. We use the loss implementations of Dubois et al. (2019) without changes and train on the same preprocessed image tensors as the flows.

Architecture and training. Following the fully connected models of Higgins et al. (2017) and Chen et al. (2018), encoder and decoder are ReLU MLPs (encoder $2352 \to 1200 \to 1200 \to z$, decoder $z \to 1200 \to 1200 \to 1200 \to 2352$ with sigmoid output) with a Bernoulli likelihood; the model has 10.1M parameters at $z = 32$. We train with schedule-free AdamW (learning rate $3 \cdot 10^{-4}$, 100 warm-up steps) at batch size 128 for 100 epochs. FactorVAE, whose implementation splits every batch in two for its discriminator, is trained at batch size 256 for 200 epochs, which gives the same number of updates. The selected settings of β-VAE, β-TCVAE and FactorVAE (below) were then trained for 300 epochs, and FactorVAE also for 500 epochs with six seeds; AnnealedVAE was not trained beyond 100 epochs (Table F.1). All evaluations use the averaged weights.

Tuning. Regularization strength and latent size were tuned for the number of stable features, a count of the same kind as for the flows (see below): β-VAE with $\beta \in \{4, 8\}$ and β-TCVAE with $\beta \in \{2, 3, 4, 6\}$, both at $z \in \{32, 64\}$, and FactorVAE with $\gamma \in \{10, 40\}$ and AnnealedVAE with $C \in \{25, 50\}$, both at $z = 32$. This grid followed a pilot of 28 runs (30 epochs, two seeds). Every cell was trained for 100 epochs with seeds 101 to 103.

Stability count. For each VAE we compute the decoder Jacobian at the encoder mean of 512 fixed training images, average the squared cosine similarity between the unit Jacobian columns of two seeds over the images, and match the z latents of the two seeds with the Hungarian algorithm. For each seed, its latents are ordered by their matched similarity averaged over the other seeds; these ordered curves are averaged over all ordered seed pairs of a cell, and the cell's count is the number of ranks at which the average is at least 0.5. This count differs from the one used for the flows in two ways: it uses the raw squared cosine instead of the polar factor, and it is pooled over all seed pairs instead of reported per pair. A pooled count over five seeds is stricter than one over three (β-TCVAE with $\beta = 3$: 11 with three seeds, 9 with five). This pooled count was used only to select the tuned settings; Table 1 and Fig. 5 evaluate these settings with the per-pair polar count of Sec. 4, like the flows.

Selected settings. With three seeds, the pooled count ranks β-VAE $\beta = 4$ (9 stable features) and AnnealedVAE $C = 25$ (4) first for their objectives and ties β-TCVAE $\beta = 3$ with $\beta = 4$ (11 each) and FactorVAE $\gamma = 10$ with $\gamma = 40$ (10 each). Our selection rule broke ties by the number of latents with a pooled similarity of at least 0.3 and then by the mean over the ten best latents, which chose $\beta = 3$ and $\gamma = 10$; these four cells were extended by seeds 104 and 105, and the stricter five-seed count gives 9 for β-TCVAE $\beta = 3$ and FactorVAE $\gamma = 10$, 8 for β-VAE and 3 for AnnealedVAE. For the longer runs we kept β-VAE $\beta = 4$ and FactorVAE $\gamma = 10$ and took β-TCVAE $\beta = 4$, the other cell of its tie. Doubling the latent size to $z = 64$ does not raise the best count (best $z = 64$ cell: 10, β-TCVAE $\beta = 2$); at equal β the count changes by -4 to $+1$.

Settings used in the paper. All CelebA results use the same four settings, each with $z = 32$ latents: β-VAE $\beta = 4$, β-TCVAE $\beta = 4$ and FactorVAE $\gamma = 10$ after 300 epochs with seeds 101–103 (FactorVAE also after 500 epochs with six seeds in Fig. 5), and AnnealedVAE $C = 25$ after its 100-epoch tuning budget with seeds 101–105. The five-seed β-TCVAE $\beta = 3$ runs enter no reported result. The ground-truth benchmarks use convolutional VAEs at the settings of the literature instead (App. G.3).

The VAEs are trained on clean images, without noise inflation (App. F.2).

Entropy order. For the manifold entropies H_i of a VAE, we treat its decoder output as a deterministic function of the standard normal prior, as for a flow and as in Galperin & Koethe (2025), and sort its latents by H_i.

F.5 Nested Dropout implementation

NDFlows (Bekasov & Murray, 2020) use the bottleneck reconstruction (10) during training. For every noise-inflated training image $\boldsymbol{x}_\epsilon$ they draw a bottleneck size $C \sim p_C$ and minimize the reconstruction error $\mathcal{L}_{\text{rec},C}(\boldsymbol{x}_\epsilon) := \frac{1}{D}\|\boldsymbol{x}_\epsilon - \boldsymbol{g}([\boldsymbol{f}_{\leq C}(\boldsymbol{x}_\epsilon), 0])\|_2^2$ in a weighted sum with the maximum likelihood loss,

$$\mathcal{L}_{\text{ND}}(\boldsymbol{x}_\epsilon) = \mathcal{L}_{\text{ML}}(\boldsymbol{x}_\epsilon) + \nu \cdot \mathcal{L}_{\text{rec},C}(\boldsymbol{x}_\epsilon), \quad \nu > 0, \tag{F.1}$$

where both terms are computed per dimension and averaged over the batch. Because C is drawn independently for every image from a geometric distribution, most reconstructions keep only the z_i

with small i, and the model is pushed to channel the core information into the first dimensions. The latents are not re-sorted during training.

Following Rippel et al. (2014) and Bekasov & Murray (2020), p_C is a geometric distribution, $p_C(C) \propto (1-p)^{C-1}p$, truncated to $1 \leq C \leq D$. We use $p = 0.01$ for all NDFlows, so that the expected bottleneck is $1/p = 100$, only $0.99^{300} \approx 5\,\%$ of the draws keep more than 300 latents, and fewer than $1\,\%$ more than 500. This bias towards small C may be why NDFlows denoise poorly at large C in Fig. 4b: bottlenecks of more than a few hundred latents are almost never trained for.

Training recipe. NDFlows use the architecture, data, noise level, batch size, base learning rate and evaluation of the EOFlows, without the Hadamard regularizer. The reconstruction term required three changes, found in short validation runs. (i) 500 instead of 100 warm-up steps: with 100 steps, $\nu = 1$ diverged at the end of the warm-up at learning rates $6 \cdot 10^{-4}$ and $4.5 \cdot 10^{-4}$, whereas the same flow without the reconstruction term trained stably. (ii) Gradient clipping at norm 2, above the largest gradient norm of a 300-step run at $\nu = 10$ without clipping (1.62); it was added after the first $\nu = 1$ runs had started, so seeds 103 and 105 trained their first 14 epochs without it. (iii) A lower learning rate for the mixing matrices: with all parameters at $6 \cdot 10^{-4}$, $\nu = 3$ and $\nu = 10$ diverged within their first two epochs. In the 300-epoch sweep the mixings were trained at $3 \cdot 10^{-4}$, with two exceptions: $\nu = 10$ seeds 103 and 105 diverged at this rate and were retrained at 10^{-4}, and $\nu = 1$ seeds 103 and 105 ran at a uniform $6 \cdot 10^{-4}$ from epoch 14 on, after a restart that reset the learning rates of the parameter groups and the optimizer state. The 500-epoch $\nu = 1$ runs used a uniform $6 \cdot 10^{-4}$, except seed 106 ($3 \cdot 10^{-4}$). Seed 106 differs from the other $\nu = 1$ seeds mainly in a larger train–test gap ($+0.124$ against $+0.092$ to $+0.097$ at 500 epochs, $+0.099$ against $+0.077$ at 300 epochs).

Divergences. Two of the eight $\nu = 1$ runs started for 500 epochs diverged: seed 108 at epoch 116 and seed 103, resumed from 300 epochs, at epoch 494. In both, the forward pass blew up, which gradient clipping cannot prevent. Seeds 111 and 112 replaced them. All six EOFlow $\lambda = 1$ runs of the 500-epoch set completed.

Entropy ordering. Like all flows, NDFlows are evaluated with their latents sorted by manifold entropy. Nested dropout already orders them almost by index: for the six $\nu = 1$ models at 500 epochs, 289 to 296 of the 300 highest-entropy latents have an index below 300, and the ten highest-entropy latents include at least nine of the indices 0 to 9.

Effect of ν. At 300 epochs and three seeds per setting, a stronger reconstruction term trades likelihood for reconstruction: the held-out NLL is -1.267, -1.274 and -1.236 nats per dimension at $\nu = 0.1$, 1 and 10 (the two retrained $\nu = 10$ seeds: -1.224), the Hadamard gap $\mathcal{L}_\mathrm{H}/D$ falls from 0.44 to 0.20 and 0.10, and the FID of prior samples through the $C = 100$ bottleneck is 26, 22 and 22. None of the three reproduces its coordinates across seeds: $1.7_{\pm 1.2}$, $8.5_{\pm 1.8}$ and $5.7_{\pm 0.6}$ polar factors per seed pair match at $\cos^2 \geq 0.5$ ($\nu = 1$: four seeds, Table F.1), against 121 for EOFlows at $\lambda = 1$. Denoising through the bottleneck (as in Fig. 4b, which shows $\nu = 1$) improves with ν but saturates in every case: the held-out error levels off at $5 \cdot 10^{-3}$ for $\nu = 0.1$, reaches the noise variance σ_ϵ^2 at $C \approx 200$ for $\nu = 1$ and a minimum of $1.7 \cdot 10^{-3}$ at $C = 500$ for $\nu = 10$, against $1.0 \cdot 10^{-3}$ for EOFlows at $\lambda = 1$. On the attribute score TAD the order is reversed (Table G.4): $\nu = 0.1$ scores 0.58, $\nu = 1$ and $\nu = 10$ about 0.25. $\nu = 1$ is therefore used as the representative NDFlow in Figs. 5 and G.26, with the $\nu = 0.1$ and $\nu = 10$ galleries in Figs. G.25 and G.27.

F.6 Denoising autoencoders

The rate–distortion comparison (Fig. 4b) includes autoencoders trained for exactly the task that the bottleneck of a flow solves only implicitly. For each bottleneck $C \in \{10, 20, 50, 100, 200, 500\}$ we train a separate deterministic autoencoder on inputs inflated with fresh noise of $\sigma_\epsilon = 0.05$ in every step and minimize the squared error to the clean image. All autoencoders use the budget of the flows, batch size 512, schedule-free AdamW (learning rate $3 \cdot 10^{-4}$, gradient clipping at norm 1) and 300 epochs, i.e. 95,100 updates, with one seed, and they are evaluated with averaged weights on the same noisy held-out images as the flows.

Two sizes. We tried two architectures. The *small* autoencoder, shown in Fig. 4b, is a fully connected MLP (encoder $2352 \to 1200 \to 1200 \to C$ with GELU activations, mirrored decoder with linear output; 8.6–9.7M parameters). The *large* one has the depth of the flows: 12 residual blocks per side, each with two pre-LayerNorm updates at width D and hidden width 512, and linear projections

to and from the code (122–124M parameters). Despite the noise inflation, the large autoencoder over-fits, with a training error 3 to 7 times below its held-out error, and it is worse than the small one on held-out images at every C (Table F.2).

Denoising versus pure reconstruction. Trained for pure reconstruction, i.e. with the noisy input itself as the target, as the flows see their data, both autoencoders perform almost the same as with the denoising objective: the held-out error changes by at most $4 \cdot 10^{-5}$ (small) and $1.4 \cdot 10^{-4}$ (large) at every C. The two losses differ only by the cross term $-2\,\mathbb{E}\langle\widehat{\boldsymbol{x}} - \boldsymbol{x}, \boldsymbol{\epsilon}\rangle$ between reconstruction error and noise, and a small code carries little of the input noise ($0.45\,\%$ at $C = 10$, $4\,\%$ at $C = 100$ and at most $12\,\%$ at $C = 500$).

Table F.2: Held-out denoising error (MSE, $\times 10^{-3}$; noisy input, clean target) of the autoencoders, one model per C and seed 101, and of EOFlows ($\lambda = 1$, mean over three seeds) through their top-C bottleneck.

	$C = 10$	20	50	100	200	500
small AE (9M), denoising (Fig. 4b)	11.95	7.98	4.76	3.17	2.02	1.65
small AE (9M), pure reconstruction	11.96	7.98	4.76	3.17	2.03	1.61
large AE (122M), denoising	18.58	13.41	8.68	6.12	4.21	3.14
large AE (122M), pure reconstruction	18.48	13.35	8.59	6.00	4.07	3.18
EOFlow $\lambda = 1$	14.66	9.40	4.96	3.00	1.77	0.99

F.7 Jacobian-based metrics

To evaluate the orthogonality, sparsity and stability of the learned features, we rely on the decoder Jacobians. The full Jacobian represents the curvilinear coordinate system at a point $\boldsymbol{z}$ and $\boldsymbol{J}_i$ represents the first-order approximation of one coordinate line. Metrics based on $\boldsymbol{J}_i$ offer a highly informative analysis of the learned representation space as they describe the behavior of the encoder/decoder-pair locally. To obtain global metrics, we average over data samples.

Galperin & Koethe (2025) visualized the coordinates by the average Jacobian column $\mathbb{E}_{\boldsymbol{z}=\boldsymbol{f}(\boldsymbol{x})}\left[\boldsymbol{J}_i(\boldsymbol{z})\right]$. Our archetype difference images (App. G.10) are closely related: $\boldsymbol{x}_{i+4} - \boldsymbol{x}_{i-4} = \int_{-4}^{4} \boldsymbol{J}_i \, dz_i$ integrates the Jacobian column along the mean coordinate line $\mathcal{U}_i(\boldsymbol{z}_{\neq i} = \boldsymbol{0})$.

All metrics use the decoder Jacobian $\boldsymbol{J}(\boldsymbol{z})$ at the latents $\boldsymbol{z} = \boldsymbol{f}(\boldsymbol{x})$ of fixed images $\boldsymbol{x}$, computed with the averaged (schedule-free) weights of each model, and we view its i-th column $\boldsymbol{J}_i$ as the i-th latent representation at the point $\boldsymbol{x}$. The metrics differ in the images and columns they use: effective sparsity uses the columns of the 300 highest-entropy latent dimensions at 512 clean training images, stability uses all D columns at the first 256 of these images, and the Hadamard gap uses all D columns at 32 noise-inflated held-out images.

F.7.1 Orthogonality

We measure orthogonality with the Hadamard gap of Eq. (6), which is zero if and only if the decoder-Jacobian columns are orthogonal and does not change when a column is rescaled. VAE decoders map 32 latents to D pixels and are not bijective, so we use its injective form over the C core coordinates S, i.e. the C highest-entropy ones, per coordinate:

$$\mathcal{L}_{\mathrm{H}}^{(C)}(\boldsymbol{z}) = \frac{1}{C}\Big[\sum_{i \in S} \log \|\boldsymbol{J}_i(\boldsymbol{z})\|_2 - \tfrac{1}{2} \log\det\left(\boldsymbol{J}_S(\boldsymbol{z})^\top \boldsymbol{J}_S(\boldsymbol{z})\right)\Big] \;\geq\; 0. \tag{F.2}$$

It is non-negative by Hadamard's inequality for the Gram matrix $\boldsymbol{J}_S^\top \boldsymbol{J}_S$, and for a flow and $C = D$ it equals $\mathcal{L}_{\mathrm{H}}/D$. Its value depends on C, since the gap over a set of columns is at least the sum of the gaps over any split of the set into parts, so values are strictly comparable only at equal C. Table F.3 therefore lists every variant we computed: $C = 32$ for every model, i.e. all latents of the VAEs, $C = 300$ and $C = D$ for the flows. Table 1 combines $C = 300$ for the flows with $C = 32$ for the VAEs, as the sparsity measure does; at the matched $C = 32$ the ranking of the models is the same. For $C \leq 300$ we average over the 512 clean training images of the sparsity measure (VAEs: at the posterior means); $C = D$ is computed exactly from the full $D \times D$ Jacobian at 32 held-out images

with added training noise ($\sigma_\epsilon = 0.05$, one fixed draw), in double precision. All models use their averaged weights.

Table F.3: Orthogonality of all models: Hadamard gap per coordinate (Eq. F.2) over the C core coordinates, mean$_{\pm\text{sd}}$ over seeds (NDFlow $\nu = 1$ at $C \leq 300$: one seed). Table 1 combines $C = 300$ for the flows and PCA with $C = 32$, i.e. all latents, for the VAEs; $C = D$ for EOFlows and the NF is Fig. 6b.

model	$C = 32$	$C = 300$	$C = D$
EOFlow $\lambda = 1$	$0.035_{\pm<.001}$	$0.041_{\pm.001}$	$0.014_{\pm<.001}$
EOFlow $\lambda = 0.1$	$0.13_{\pm<.01}$	$0.17_{\pm<.01}$	$0.058_{\pm.002}$
NDFlow $\nu = 1$	0.20	0.29	$0.20_{\pm<.01}$
NDFlow $\nu = 0.1$	$0.38_{\pm.01}$	$0.46_{\pm<.01}$	$0.44_{\pm<.01}$
NF	$1.2_{\pm<.1}$	$1.6_{\pm<.1}$	$1.6_{\pm<.1}$
FactorVAE	$0.38_{\pm.07}$	–	–
β-TCVAE	$0.55_{\pm.01}$	–	–
β-VAE	$0.50_{\pm.01}$	–	–
AnnealedVAE	$0.77_{\pm.05}$	–	–
noisy PCA	0	0	0

F.7.2 Effective Sparsity

We introduce the normalized squared Jacobian column vector as:

$$\bar{\boldsymbol{J}}_i \coloneqq \frac{(\boldsymbol{J}_i)^2}{\sum_{j=1}^{D} (\boldsymbol{J}_{ji})^2} \tag{F.3}$$

where the square is performed elementwise and the sum goes over the data dimension j. Evidently, the sum $\sum_j \bar{\boldsymbol{J}}_{ji} \equiv 1$. This can be used to define the following sparsity metric:

$$\mathrm{ENZ}(i) \coloneqq \frac{1}{D} \mathop{\mathbb{E}}_{\boldsymbol{z}=\boldsymbol{f}(\boldsymbol{x})} \left[2^{H_2(\bar{\boldsymbol{J}}_i(\boldsymbol{z}))} \right], \qquad H_2(\bar{\boldsymbol{J}}_i) \coloneqq -\sum_{j=1}^{D} \bar{\boldsymbol{J}}_{ji} \log_2 \bar{\boldsymbol{J}}_{ji} \tag{F.4}$$

Here $2^{H_2(\bar{\boldsymbol{J}}_i)}$ is the effective number of active pixels of $\boldsymbol{J}_i$, analogous to the Effective Number of Non-Zero Pixels measure proposed in He et al. (2026); dividing by D gives the Effective Percentage of Non-Zero Pixels (ENZ). If $\bar{\boldsymbol{J}}_i$ is spread equally over k pixels and zero elsewhere, $\mathrm{ENZ}(i) = k/D$, so ENZ generalizes the L0 sparsity count: a single active pixel gives $1/D$ (most sparse), all pixels equally active give 1 (least sparse), and a random Gaussian direction gives 0.482 on average. Unlike a count of the pixels whose change exceeds a threshold, ENZ depends neither on a threshold nor on the scale of the change. We evaluate ENZ at 512 clean training images for the 300 highest-entropy latent dimensions of each flow; the VAEs have only 32 latents, which are all used (with the Jacobian at the encoder mean), and the PCA components are ranked by variance. We restrict ENZ to the leading coordinates because the low-entropy tail carries no signal. Over all D coordinates, the sparsest PCA components are null-space eigenvectors (median variance rank of about 2300 of 2352, with numerically zero eigenvalues), which can concentrate on any pixels and would make PCA look sparser than EOFlows at $\lambda = 0.1$; EOFlows at $\lambda = 1$ would also gain sparse coordinates (203 below 0.1 instead of 90), but mostly inside the core (median entropy rank 330).

F.7.3 Stability

To measure stability, we compare the decoder Jacobians of two models a and b that differ only in the random seed. Before comparing their columns, we replace the full $D \times D$ decoder Jacobian $\boldsymbol{J}$ at each image by its polar factor

$$\boldsymbol{P} \coloneqq (\boldsymbol{J}\boldsymbol{J}^T)^{-1/2} \boldsymbol{J} = \boldsymbol{U}\boldsymbol{V}^T \qquad \text{for } \boldsymbol{J} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T, \tag{F.5}$$

the orthogonal matrix closest to $\boldsymbol{J}$. Its column $\boldsymbol{P}_i$ is the direction of latent i after the whole frame has been made orthonormal with the smallest possible change; if the columns of $\boldsymbol{J}$ are already orthogonal

($\mathcal{L}_{\mathrm{H}} = 0$), then $\boldsymbol{P}_i = \boldsymbol{J}_i / \|\boldsymbol{J}_i\|_2$. We compute $\boldsymbol{P}$ with 40 Newton–Schulz iterations started from $\boldsymbol{J}/\|\boldsymbol{J}\|_F$, which agree with the SVD up to a largest entrywise difference of $3 \cdot 10^{-4}$. The similarity matrix $S^{(ab)} \in \mathbb{R}^{D \times D}$ is the squared overlap of the polar factors, averaged over the same $N = 256$ fixed clean training images $\boldsymbol{x}_n$ for every model:

$$S_{ij}^{(ab)} \coloneqq \frac{1}{N} \sum_{n=1}^{N} \left(\boldsymbol{P}_i^{(a)}(\boldsymbol{x}_n)^T \boldsymbol{P}_j^{(b)}(\boldsymbol{x}_n) \right)^2, \tag{F.6}$$

where $\boldsymbol{P}^{(a)}(\boldsymbol{x}_n)$ is the polar factor of the decoder Jacobian of model a at $\boldsymbol{z} = \boldsymbol{f}^{(a)}(\boldsymbol{x}_n)$. Since ${\boldsymbol{P}^{(a)}}^T \boldsymbol{P}^{(b)}$ is an orthogonal matrix, the squares of its entries sum to one in every row and every column, so $S^{(ab)}$ is doubly stochastic (numerically, on 32 images of one seed pair with a single-precision SVD, all row and column sums lie between 1.0000 and 1.0022). For exactly orthogonal frames, $S^{(ab)}$ reduces to the raw squared cosine similarity of the Jacobian columns.

Latent i of model a need not correspond to latent i of model b, so we match the two orderings: Hungarian matching of $S^{(ab)}$ (`linear_sum_assignment` in SciPy) permutes the latents of model b such that the sum of the diagonal entries of the permuted matrix $\widehat{S}^{(ab)}$ is maximal. We are allowed to do this, since stable training can only be expected up to permutations and sign flips of the latent dimensions, and the squared overlap does not see signs. We denote the matched order by k; it generally does not coincide with the entropy order i. The matching runs over all D latents and not only over the 300 highest-entropy ones, because stable latents also exist below entropy rank 300 (in an earlier check with the raw squared cosine on one seed pair of a 200-epoch model, 15 of 109 matched values ≥ 0.5 lay below that rank).

We take the diagonal entries of $\widehat{S}^{(ab)}$, sorted in decreasing order, as the stability of the matched latents

$$\mathrm{Stability}(k, a, b) \coloneqq \widehat{S}_{kk}^{(ab)} \quad \text{s.t. } \mathrm{Stability}(k, a, b) \geq \mathrm{Stability}(k+1, a, b) \tag{F.7}$$

A latent is stable if its matched value is at least 0.5, and we count the stable latents for every pair of seeds. The threshold is conservative: shifting a decoder-Jacobian column horizontally by one pixel lowers its squared cosine with the unshifted column to 0.25 ($\lambda = 0.1$) and 0.30 ($\lambda = 1$) on average (10–90 % range about 0.01–0.65), and a Gaussian blur with a standard deviation of one pixel lowers it to 0.69 and 0.71. These references were measured with the raw squared cosine on the 300 highest-entropy columns of one model (seed 101) per λ from an earlier 200-epoch recipe with batch size 128, and not repeated on the models of this paper.

Why the polar factor. The raw squared cosine also finds no stable latents between random orthonormal frames, but it is only meaningful between frames that are themselves nearly orthogonal. When the compared frames are not orthogonal, it is inflated by the variance spectrum. For a randomly rotated copy $\boldsymbol{AR}$ of the noisy-PCA decoder $\boldsymbol{A}$, every column is dominated by the same few high-variance directions, and two of its columns have $\mathbb{E}\cos^2 \approx \sum_i \lambda_i^2 / (\sum_i \lambda_i)^2 = 1/\mathrm{PR}$, where λ_i are the PCA eigenvalues and PR is their participation ratio. For CelebA-28 (RGB), $\mathrm{PR} = 11.9$ predicts 0.084 (measured: 0.063), and matching two such rotated frames yields 338 spurious matched values ≥ 0.5, against 0 for a flat spectrum. The same happens when the frames of trained flows are rotated. For two EOFlows ($\lambda = 1$, seeds 103 and 105, 300 epochs, 256 images, one draw of each rotation), rotating the latent frames of both models yields 527 "stable" latents under the raw squared cosine (682 if both rotated frames come from the same model), and a trained frame against a rotated one yields 5. Under the polar factor, all three comparisons yield 0, with a mean matched value of 0.008–0.009 over the best 100 latents, the level of random orthonormal frames. This holds by construction: the polar factor of a randomly rotated frame is itself a random orthonormal frame. Trained EOFlow frames are nearly orthogonal (mean squared cosine between two columns of the same frame $3 \cdot 10^{-5}$), and on the trained pair the two metrics agree (116 raw, 121 polar): the spectrum does not inflate the reported EOFlow counts, and the polar factor makes the comparison with rotated and random frames valid. It does not remove artifacts of the basis itself: architectures whose latent directions stay tied to the pixel axes reproduce those axes trivially (App. G.1), so stability should be read together with the entropy order.

This protocol is used for every model in this paper: EOFlows at every λ, NDFlows, the plain NF, the arms of the architecture ablation and the VAEs, for which the polar factor and the matching run over their 32 latents.

Coordinate grids, not latent codes. A simpler comparison would correlate the latent codes of two runs. Such a correlation is a statistic between the two code distributions and cannot be evaluated at a single data point, whereas our similarity compares the two coordinate grids at every sample and takes the expectation afterwards. This is arguably stricter and resembles the nonlinear Amari distance of Gresele et al. (2021). Fig. F.2 shows both for the models and seed pairs of Fig. 5, (a) our protocol and (b) the squared Pearson correlation r^2 between the latents of two runs on 10,000 held-out images, Hungarian-matched in the same way. The difference is visible for NDFlows and FactorVAE. About six of their leading latents correlate almost perfectly across seeds ($r^2 \geq 0.95$), while the coordinate lines of their six best-matched latents agree only at 0.87 to 0.54 (NDFlow) and 0.89 to 0.70 (FactorVAE), and the NDFlow count of matches ≥ 0.5 doubles from 7 to 14.5 under r^2. Two runs can thus agree on what a latent measures across the dataset and still move through data space in different directions at a given sample when it changes, since the coordinate line $\mathcal{U}_i$ also depends on all other latents, which are held fixed along it. The r^2 matrix is moreover not doubly stochastic (median row sums between 0.77 and 2.14), so a value ≥ 0.5 does not single out one partner, and r^2 drops under any monotone non-linear reparametrization of a latent, which leaves the coordinate line itself unchanged.

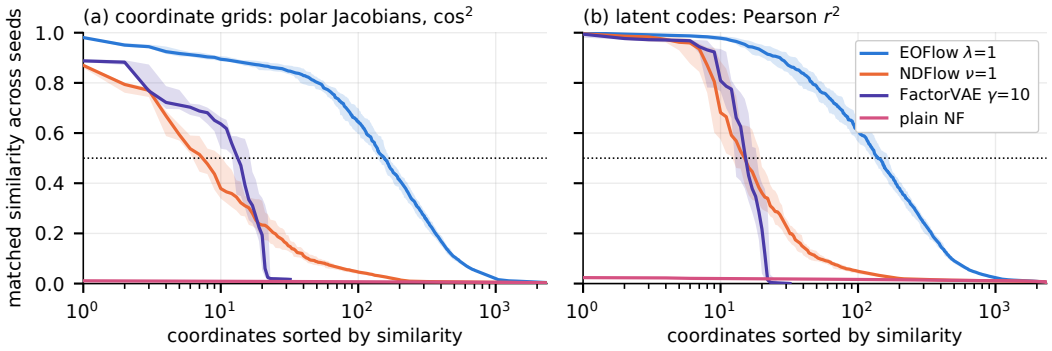


Figure F.2: Stability of coordinate grids (a, the protocol of Fig. 5c) and correlation of latent codes (b, squared Pearson correlation on 10,000 held-out images, Hungarian-matched), sorted per seed pair (median, min–max band); same models and seed pairs as Fig. 5.

Fig. 5 shows the sorted matched values for every seed pair of the best model of each class, App. G.7 the 100 most stable EOFlow features in all six seeds, and App. G.10 the archetypes of every configuration.

F.7.4 Linear geometry of the detail coordinates

Sec. 3.4 argues that the bottleneck reconstruction (10) is an orthogonal projection onto the data manifold because the detail subspace encodes only isotropic noise and its coordinate lines $\mathcal{U}_i$ are straight. Such coordinates must have (a) the variance of the noise and (b) a rigid, linear geometry. We test both for all $D = 2352$ coordinates of the EOFlow of the appendix figures ($\lambda = 1$, seed 103, 300 epochs) and, as a control, of the plain NF with the same seed, at the 512 clean training images of App. F.7. With the unit tangent $\boldsymbol{u}_i(\boldsymbol{z}) \coloneqq \boldsymbol{J}_i(\boldsymbol{z})/|\boldsymbol{J}_i(\boldsymbol{z})|_2$, the non-linearity of $\mathcal{U}_i$ is the variance of its tangent directions across images,

$$\mathop{\mathbb{E}}_{\boldsymbol{z}=\boldsymbol{f}(\boldsymbol{x})}\left[\left|\boldsymbol{u}_i(\boldsymbol{z}) - \bar{\boldsymbol{u}}_i\right|_2^2\right] = 1 - \left|\bar{\boldsymbol{u}}_i\right|_2^2, \qquad \bar{\boldsymbol{u}}_i \coloneqq \mathop{\mathbb{E}}_{\boldsymbol{z}=\boldsymbol{f}(\boldsymbol{x})}\left[\boldsymbol{u}_i(\boldsymbol{z})\right], \tag{F.8}$$

where the equality holds because $|\boldsymbol{u}_i|_2 = 1$. For small deviations it is the mean squared angle between the tangents and their common direction $\bar{\boldsymbol{u}}_i$. It is zero iff the lines $\mathcal{U}_i$ through all images are parallel, i.e. iff coordinate i is one fixed direction in data space as in a linear map, it approaches one for unrelated directions, and it does not depend on the length of $\boldsymbol{J}_i$. Averaged over the 300 highest-entropy coordinates, it is the non-linearity of the coordinate grid in Fig. 6e, which falls monotonically along the sweep of λ from 0.82 for the plain NF over 0.46, 0.32, 0.14 and 0.045 to 0.004 for $\lambda = 0.01$, 0.1, 1, 3 and 10 (seed 103 each), and is 0 for PCA by construction.

Fig. F.3a shows it against the entropy rank. For the EOFlow it falls from 0.28 for the ten highest-entropy coordinates to 0.008 beyond rank 1000: the tangents of these coordinates deviate by about

5° from a common direction across all images. The plain NF has no such subspace; its coordinate lines point in largely unrelated directions at every rank (0.75–0.88 on average, never below 0.48). Fig. F.3b shows the mean norm of the Jacobian columns. For the EOFlow it decreases with the rank and levels off at 0.052 beyond rank 1000 (minimum 0.050), the noise level $\sigma_\epsilon = 0.05$, in agreement with the entropy spectrum of Fig. 4a; the columns of the plain NF stay above 0.18. The low-entropy coordinates of the EOFlow are thus fixed directions of the scale of the noise.

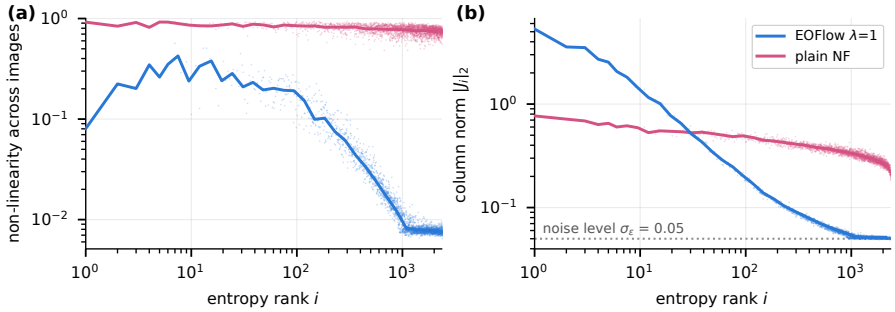


Figure F.3: Geometry of the coordinates against the entropy rank i for all 2352 coordinates of the EOFlow ($\lambda = 1$) and of the plain NF (same seed, 300 epochs each) at 512 training images: (a) non-linearity across images (zero iff the lines $\mathcal{U}_i$ through all images are parallel), (b) mean norm of the Jacobian column $\boldsymbol{J}_i$. Dots are single coordinates, lines running medians over rank bins.

G ADDITIONAL RESULTS

This appendix collects ablations and further experiments (App. G.1–G.4), further evaluations of the CelebA models (App. G.5 and G.6) and image galleries of the stable features across seeds and of the prior samples, reconstructions and archetypes of all models (App. G.7–G.10).

G.1 ARCHITECTURE ABLATION: WHICH PART OF THE MIXING IS NEEDED?

The flow alternates affine coupling blocks with dense $D \times D$ mixing layers. Since the Hadamard regularizer selects a latent basis, whereas maximum likelihood is indifferent to it, one may ask whether the learnable mixing layers are an arbitrary choice or an essential one. We retrain the CelebA model ($D = 2352$, 12 coupling blocks with two hidden layers of width 1024, $\lambda = 1$, $\sigma_\epsilon = 0.05$, batch size 512, 100 epochs, three seeds per arm, otherwise identical) in five arms that differ only in the mixing between the blocks (parameter counts as stored, App. F.1):

- **ref**: all twelve $D \times D$ mixing matrices learnable (the model used throughout the paper), 245M parameters;
- **last1**: eleven mixing matrices frozen at their random orthogonal initialization, only the final one learnable (123M);
- **frozen**: all mixing matrices frozen (112M);
- **nomix**: no mixing matrices at all, only the random permutations between the coupling blocks (112M);
- **noperm**: neither mixing matrices nor permutations (112M).

Table G.1 lists held-out likelihood, the Hadamard gap $\mathcal{L}_\mathrm{H}/D$ (zero for an exactly orthogonal decoder Jacobian), the FID of prior samples through a bottleneck of $C = 100$, and the number of stable features per pair of seeds under the protocol of Appendix F.7: decoder Jacobians at 256 fixed training images, polar (orthonormal) factors, the squared cosine similarity between the latent directions of two seeds averaged over the images, Hungarian matching of the latent dimensions, and a threshold of 0.5 on the matched value. Fig. G.1 shows the matched similarity of every latent dimension and Fig. G.2 the leading archetypes of each arm.

Table G.1: Architecture ablation on CelebA, 100 epochs, three seeds per arm (means; two ref seeds reach NLL -1.294 and FID 27, the third -1.284 and 33; all other arms agree across seeds to the last digit). NLL: held-out negative log-likelihood per dimension of noise-inflated images; $\mathcal{L}_\mathrm{H}/D$: Hadamard gap; FID at $C = 100$ for prior samples; stable: latent dimensions with matched squared cosine similarity ≥ 0.5, given for the three seed pairs. The stable counts of the two mixing-free arms are set in italics: they count the pixel basis, not learned features (see text).

arm	NLL	$\mathcal{L}_\mathrm{H}/D$	FID	stable features per seed pair
ref (12 learnable mixings)	-1.291	0.012	29	54 / 46 / 40
last1 (final mixing learnable)	-1.285	0.017	28	9 / 11 / 7
frozen (no learnable mixing)	-1.262	0.030	42	2 / 3 / 5
nomix (permutations only)	-1.251	0.027	37	*1827 / 1828 / 1824*
noperm (couplings only)	-1.247	0.025	38	*1834 / 1838 / 1843*

Learnable mixings buy reproducibility, not likelihood. Among the three arms that mix all dimensions (ref, last1, frozen), the ordering by every quality metric is the same, but the differences are modest: a single learnable mixing matrix recovers the likelihood and the sample quality of the full model to within 0.01 nats per dimension and one FID point, and even the frozen arm orders its latent dimensions – the leading hundred entropies are nearly identical across all arms (Fig. G.3). Reproducibility across seeds, however, separates the arms by an order of magnitude: $46.7_{\pm 7.0}$ stable features per seed pair with all mixings learnable, $9.0_{\pm 2.0}$ with only the final one, $3.3_{\pm 1.5}$ with none. The stable features of the full model are also the important ones: $39.0_{\pm 5.0}$ of them lie among the 100 highest-entropy latent dimensions. The leading archetypes of the last1 and frozen arms (Fig. G.2) show the same vocabulary as the full model – illumination, background, skin and hair hue, glasses,

mouth – but a different selection and ordering in every seed. Maximum likelihood is indifferent to the latent basis and the Hadamard regularizer only acts through the decoder Jacobian; without enough learnable rotations, the regularizer cannot move the representation to the basis in which the coordinates are orthogonal, and each seed settles for a different compromise. The coupling blocks cannot provide these rotations, since each transforms only half of the coordinates element-wise, conditioned on the other half.

Data-biased architectures are stable by design. The two mixing-free arms appear to be the most reproducible of all, with ≈ 1830 of 2352 latent dimensions matched at squared cosine similarity ≥ 0.5. This is an artifact of their bias towards the pixel axes: without a mixing layer, the coupling blocks are initialized as the identity map and never rotate the latent space away from the data axes, so the two seeds find the *same* axes trivially. In the noperm arm, 89% of the latent dimensions are matched to the same index in the other seed. The matched similarity, plotted along the entropy order (Fig. G.1b), makes the mechanism visible: it is low (≈ 0.3) for the leading ≈ 300 latent dimensions, which are global, seed-specific mixtures, and jumps to ≈ 0.9 beyond, where every latent dimension is a single pixel. These dimensions form a plateau of ≈ 2000 latent dimensions of identical entropy in the spectrum, which lies just above the noise floor – the "stability" of these arms sits entirely in the noise dimensions, and only $4.0_{\pm 1.0}$ (noperm) and $9.3_{\pm 4.0}$ (nomix) of their hundred highest-entropy dimensions reproduce. The same caveat applies to any architecture whose inductive bias is tied to the data axes, such as convolutions: it will score higher on a stability metric by construction, without its features being more useful. This is why the paper's model is built from fully-connected couplings with learnable mixings, and why the stability metric should be read together with the entropy order.

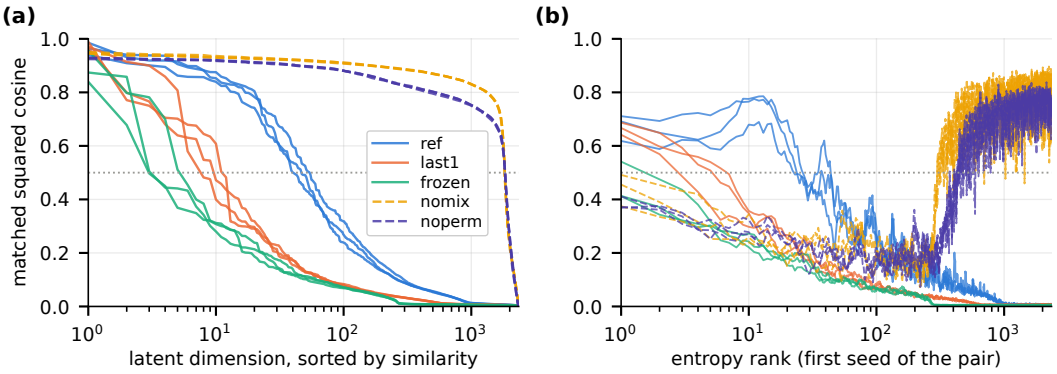


Figure G.1: Stability of the latent dimensions across seeds for the five arms of the architecture ablation, three seed pairs per arm. **(a)** Matched squared cosine similarity of every latent dimension, sorted; the crossing with the dotted line at 0.5 gives the number of stable features. **(b)** The same quantity along the entropy order of the first seed (mean over ten neighboring dimensions); the two mixing-free arms (dashed) are stable only beyond rank ≈ 300, in the pixel-basis tail.

G.2 A CONVOLUTIONAL EOFLOW ON 64×64 IMAGES

Architecture. To test the regularizer beyond fully connected flows, we train a multiscale convolutional flow on CelebA at 64×64 pixels ($D = 12288$). It has three stages, each a Haar transform followed by six GLOW coupling blocks with convolutional subnetworks (3×3, 1×1 and 3×3 convolutions, 128 hidden channels), every block followed by ActNorm and a learnable 1×1 channel mixing initialized at the identity. After the first two stages the nine detail channels leave the flow as latents (9216 and 2304 dimensions) and the three low-pass channels continue; after the third stage the remaining $12 \times 8 \times 8 = 768$ dimensions enter a fully connected flow of the main-experiment design (12 coupling blocks with MLPs of width 1024 and learnable mixings). Every latent keeps a standard normal prior, so its entropy can be read on its own; the model has 69.1M parameters. We train it with $\sigma_\epsilon = 0.1$, batch size 128 and schedule-free AdamW (learning rate $3 \cdot 10^{-4}$) for 100 epochs, once with $\lambda = 1$ and once with $\lambda = 0$, one seed each.

Result. With the regularizer (Table G.2), the manifold entropies span the range that PCA of the noisy data dictates (4.81 against 4.79 nats), the Hadamard gap is below that of the 28×28 model (0.008

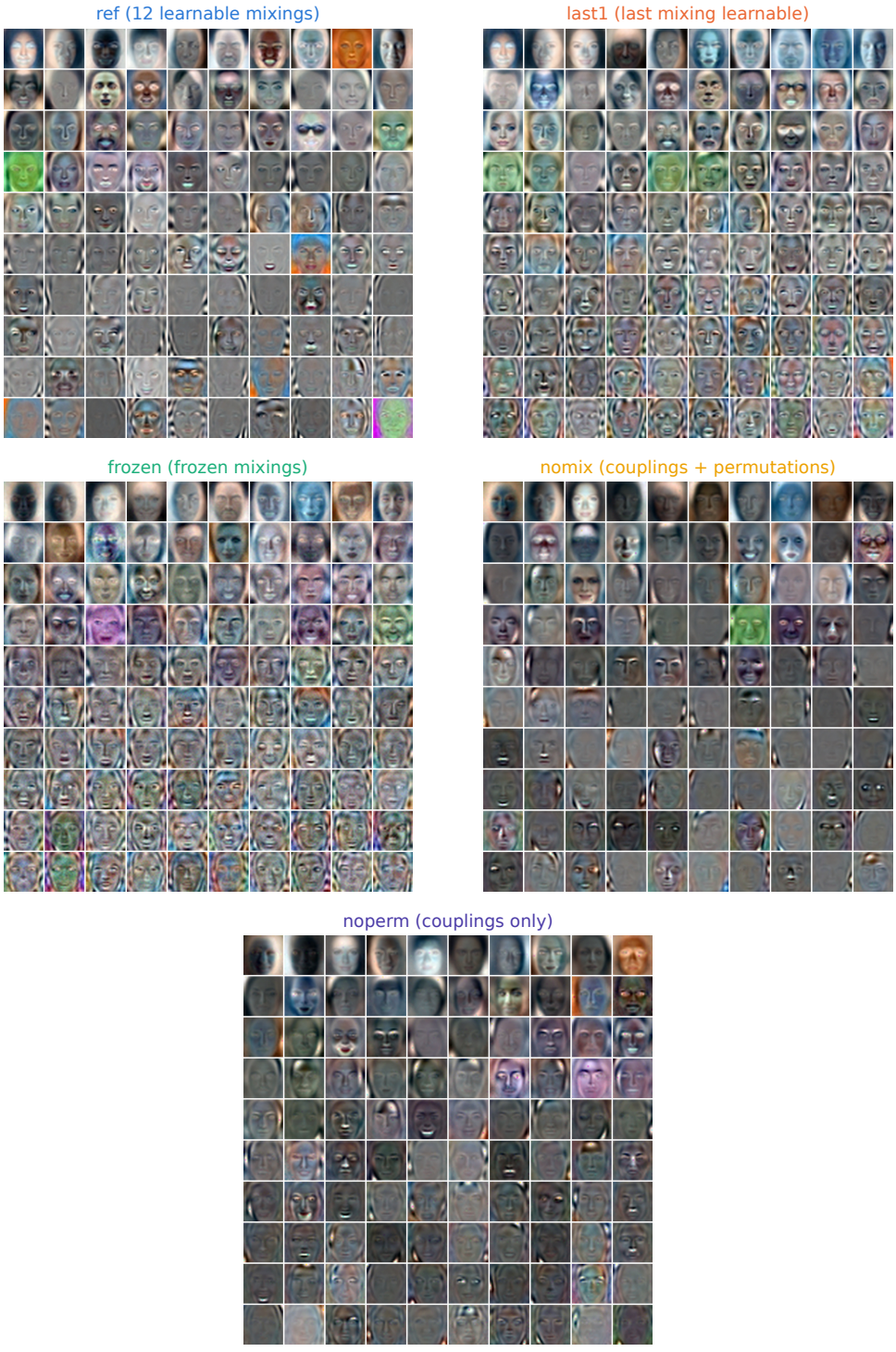


Figure G.2: Archetypes of the five arms of the architecture ablation: the 100 highest-entropy archetypes of one seed of each arm (difference images $\boldsymbol{x}_{i+4} - \boldsymbol{x}_{i-4}$ as in Appendix G.10, each normalized to $[0, 1]$, entropy order row-major).

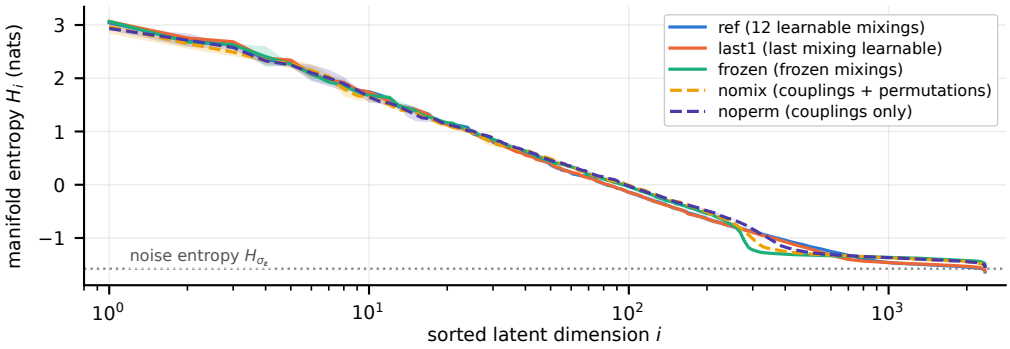


Figure G.3: Entropy spectra of the five arms of the architecture ablation, mean $\pm$ standard deviation over three seeds; the two mixing-free arms are dashed, and the dotted line is the noise entropy H_{σ_ϵ}. The spectra coincide over the leading ranks, so the entropy ordering is comparable across the arms.

against 0.014), and the 100 leading coordinates denoise held-out images to about half the error of the noisy input. Without it, the flow reaches a better likelihood, but its Hadamard gap is 16 times larger, its 768 coarsest latents span only 1.2 nats against 4.7, and its 100 leading coordinates denoise 13.5 times worse than with the regularizer, seven times worse than the noisy input itself. Seed-to-seed stability was not measured on this model. Fig. G.4a shows the difference: the EOFlow spectrum follows that of noisy PCA down to the noise entropy, whereas the unregularized flow spreads the entropy almost uniformly over its 768 coarsest latents. Fig. G.4b–d shows what the regularized model learns. Its 100 highest-entropy coordinates form the same vocabulary as the fully connected model on 28×28 images (Fig. G.21), in roughly the same order: brightness and background, illumination, skin and hair hue, sunglasses, mouth and eyes, and single color channels. Prior samples decoded through the 100 leading coordinates are clean and diverse faces; decoded at full D with one Tweedie step they keep more detail but also more artifacts. Their quality is well below that of dedicated image generators, which this experiment does not aim at: it shows that the recipe runs at this dimension and yields ordered, interpretable coordinates.

Table G.2: Convolutional EOFlow on CelebA 64×64 ($D = 12288$, $\sigma_\epsilon = 0.1$, 100 epochs, one seed each) and PCA of the noisy data. NLL: held-out negative log-likelihood per dimension; entropy range: spread of the manifold entropies H_i over all latents and over the 768 coarsest ones; MSE: denoising error of 2000 held-out images through the C leading coordinates (noisy input: 0.0100).

	$\lambda = 1$	$\lambda = 0$	PCA
NLL (nats/dim)	-0.785	-0.797	-0.767
entropy range, all latents (nats)	4.81	3.05	4.79
entropy range, 768 coarsest (nats)	4.70	1.18	–
$\mathcal{L}_\mathrm{H}/D$	0.0079	0.123	0
MSE, $C = 100$	0.0054	0.0730	–
MSE, $C = 768$	0.0019	0.0021	–

G.3 Ground-truth factor benchmarks

Data and models. We train on two standard disentanglement benchmarks with known generative factors, both at 64×64 pixels: dSprites (Matthey et al., 2017), $737{,}280$ binary images of a single shape with five factors (shape, scale, orientation and x, y position), and Shapes3D (Burgess & Kim, 2018), $480{,}000$ RGB images of a rendered scene with six factors (floor, wall and object hue, scale, shape and orientation). Each dataset is split at random into 90% training and 10% held-out images. The EOFlow is a smaller version of the convolutional flow of App. G.2: two GLOW coupling blocks with 64 hidden channels per stage, followed by a fully connected flow of four coupling blocks (512 hidden units) on the $D/16$ coarsest dimensions, with 4.3M parameters on dSprites and 11.7M on Shapes3D. We train four seeds per dataset with $\lambda = 0.1$ and three with $\lambda = 1$, all with $\sigma_\epsilon = 0.2$, batch size 256 and schedule-free AdamW (learning rate $3 \cdot 10^{-4}$) for 100 epochs.

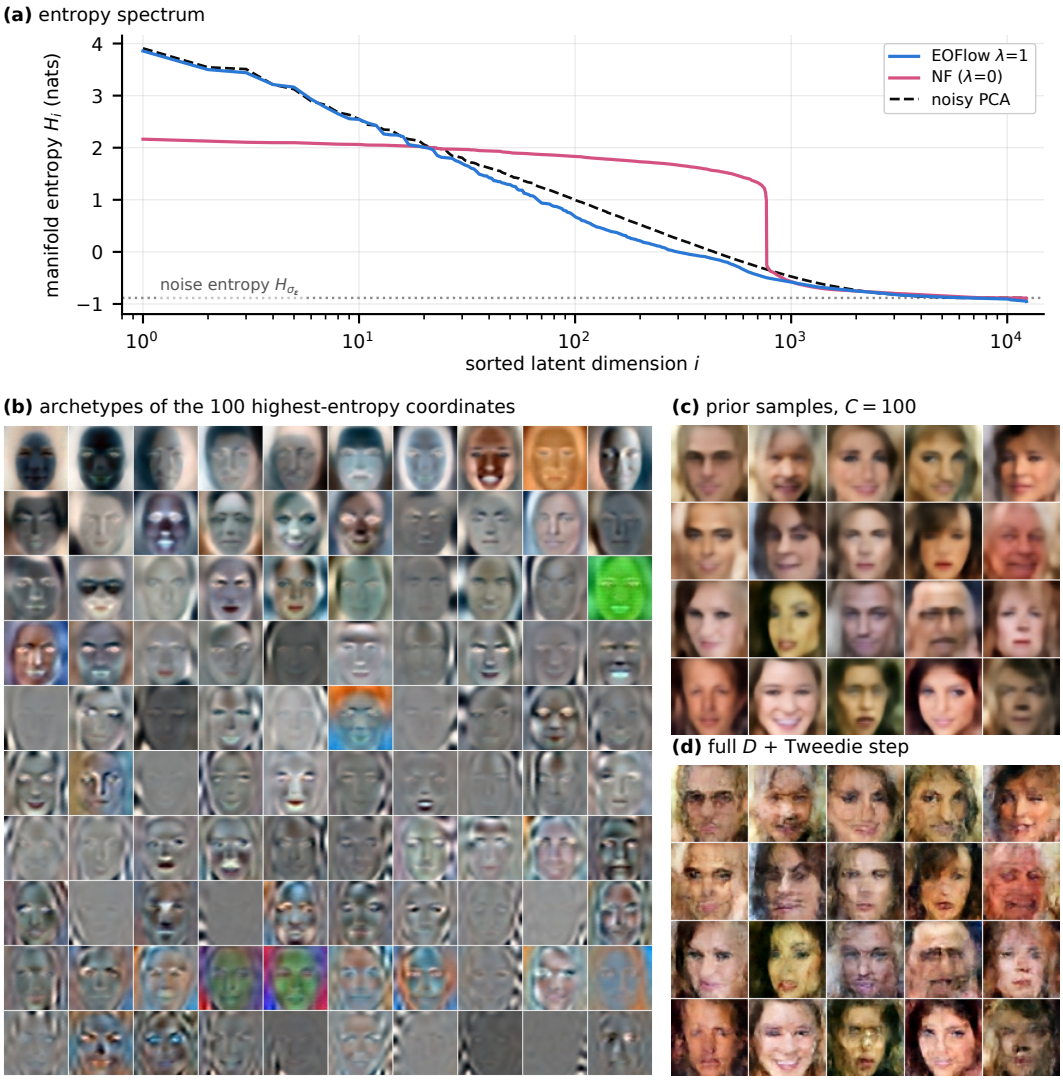


Figure G.4: **Convolutional EOFlow on CelebA** 64×64 ($D = 12288$, $\sigma_\epsilon = 0.1$, 100 epochs). **(a)** Sorted manifold entropy spectra of the EOFlow ($\lambda = 1$), the same flow trained by maximum likelihood alone (NF, $\lambda = 0$) and PCA of the noisy data (dashed), with the noise entropy H_{σ_ϵ} (dotted). **(b)** Archetypes of the 100 highest-entropy EOFlow coordinates in entropy order (difference images $g(+4\boldsymbol{e}_i) - g(-4\boldsymbol{e}_i)$ from the latent origin, each normalized). **(c)** EOFlow prior samples decoded through the $C = 100$ leading coordinates and **(d)** the same latent draws decoded at full D after one Tweedie step.

Entropy spectra. On dSprites, all four seeds have a sharp knee after the fifth coordinate, the number of ground-truth factors (Fig. G.5a): in three seeds the entropy drops there by 1.6 to 2.3 nats, in the fourth a sixth coordinate lies halfway down. The coordinates above the knee carry the x and y position, the scale and, on a pair of coordinates, the three-valued shape; orientation is isolated neither by EOFlows nor by any VAE baseline. On Shapes3D, the knee falls after 7 to 8 coordinates (Fig. G.5b). The ten values of each hue factor are evenly spaced around the color circle, so each hue forms a closed loop and occupies a pair of coordinates; orientation takes one or two more, while scale and shape stay below the knee. PCA has no knee on either dataset: 190 (dSprites) and 226 (Shapes3D) of its components lie more than 0.35 nats above the noise entropy, against at most 22 and 11 EOFlow coordinates.

Disentanglement scores. Table G.3 compares EOFlows with noisy PCA and the four VAE objectives of App. F.4, all trained on the same splits for 100 epochs. The VAEs use the convolutional encoder

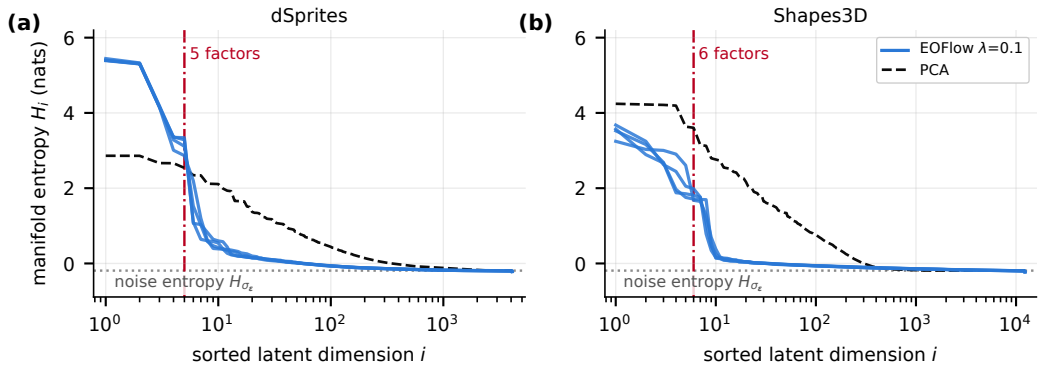


Figure G.5: **Entropy spectra on ground-truth benchmarks** (four seeds each, $\sigma_\epsilon = 0.2$, $\lambda = 0.1$, 100 epochs). Sorted manifold entropy spectra H_i of EOFlows on **(a)** dSprites and **(b)** Shapes3D, the spectrum of PCA of the noisy data (dashed) and the noise entropy H_{σ_ϵ} (dotted). The vertical line marks the number of ground-truth factors. EOFlows concentrate the entropy in a few coordinates and drop to the noise level at a sharp knee, which PCA lacks.

Table G.3: **Disentanglement on dSprites and Shapes3D** (64×64, 100 epochs). Mean$_{\pm \text{sd}}$ over seeds. MIG (Chen et al., 2018), DCI disentanglement (Eastwood & Williams, 2018) and the FactorVAE score (Kim & Mnih, 2018) on 10 latents per model (VAEs: all 10; EOFlows and noisy PCA: the 10 highest-entropy coordinates), from clean held-out images. VAEs at literature settings (β-VAE $\beta = 4$, AnnealedVAE $C = 25$ and $\gamma = 1000$, FactorVAE $\gamma = 10$, β-TCVAE $\beta = 6$); EOFlows with $\sigma_\epsilon = 0.2$. Seeds (dSprites / Shapes3D): EOFlow $\lambda = 1$ 3 / 3, EOFlow $\lambda = 0.1$ 4 / 4, FactorVAE 3 / 3, β-TCVAE 6 / 4, β-VAE 4 / 3, AnnealedVAE 3 / 3.

	dSprites (conv.)			Shapes3D (conv.)		
model	MIG ↑	DCI ↑	FactorVAE ↑	MIG ↑	DCI ↑	FactorVAE ↑
EOFlow $\lambda = 1$	$0.14_{\pm.10}$	$0.07_{\pm.04}$	$0.65_{\pm.13}$	$0.15_{\pm.03}$	$\mathbf{0.52}_{\pm.03}$	$0.80_{\pm.01}$
EOFlow $\lambda = 0.1$	$\mathbf{0.26}_{\pm.01}$	$0.19_{\pm<.01}$	$\mathbf{0.81}_{\pm.02}$	$0.17_{\pm.07}$	$0.41_{\pm.09}$	$0.70_{\pm.02}$
FactorVAE	$0.15_{\pm.07}$	$0.14_{\pm.03}$	$0.76_{\pm.09}$	$0.13_{\pm.03}$	$0.14_{\pm.05}$	$0.73_{\pm.04}$
β-TCVAE	$\mathbf{0.26}_{\pm.09}$	$\mathbf{0.25}_{\pm.03}$	$0.78_{\pm.07}$	$0.31_{\pm.12}$	$0.42_{\pm.06}$	$0.71_{\pm.09}$
β-VAE	$0.06_{\pm.02}$	$0.06_{\pm.01}$	$0.64_{\pm.06}$	$0.22_{\pm.07}$	$0.25_{\pm.08}$	$\mathbf{0.83}_{\pm.15}$
AnnealedVAE	$0.06_{\pm.05}$	$0.04_{\pm.03}$	$0.62_{\pm.15}$	$\mathbf{0.35}_{\pm.03}$	$0.20_{\pm<.01}$	$0.68_{\pm.02}$
noisy PCA	0.03	0.04	0.48	0.02	0.22	0.63

and decoder of Burgess et al. (2018) with 10 latents, a Bernoulli likelihood, batch size 128 and the literature settings of each objective. All scores are computed on clean held-out images and on 10 latents per model, the VAE posterior means and the 10 highest-entropy coordinates of EOFlows and noisy PCA. MIG and DCI use 20,000 images; mutual information is estimated from 20 equal-mass bins and normalized by the factor entropy, and DCI uses a random forest. The FactorVAE score (Kim & Mnih, 2018) follows the protocol of Locatello et al. (2019): each vote fixes one factor in a batch of 64 held-out images and records the latent with the smallest normalized variance, latents with a global standard deviation below 0.05 are pruned, and a majority-vote classifier is fit on 10,000 votes and scored on 5,000. On dSprites, EOFlows at $\lambda = 0.1$ reach the MIG of the best VAE, β-TCVAE, with a tenth of its seed-to-seed spread, but a lower DCI score, and $\lambda = 1$ is both weaker and seed-dependent (0.14 ± 0.10). On Shapes3D the ranking turns around: every EOFlow stays below three VAEs in MIG, because each hue is spread evenly over a pair of coordinates, which MIG, the gap between the two most informative latents, counts as entangled, whereas DCI, which reads a whole importance matrix, ranks $\lambda = 1$ first of all models (0.52 ± 0.03 against 0.42 ± 0.06 for β-TCVAE); at $\lambda = 1$ the flow also isolates shape and orientation instead of only orientation. The FactorVAE score, which a factor spread over a pair of coordinates does not penalize, ranks $\lambda = 0.1$ first on dSprites (0.81 ± 0.02 against 0.78 ± 0.07 for β-TCVAE) and puts $\lambda = 1$ level with the best VAE on Shapes3D (0.80 ± 0.01 against 0.83 ± 0.15 for β-VAE) at a fraction of its seed-to-seed spread.

G.4 EQUIVARIANCE: TRAINING ON SHUFFLED PIXELS

Like PCA, EOFlows are equivariant to orthogonal changes of the data basis: if $\boldsymbol{g}$ minimizes $\mathcal{L}_{\text{EO}}$ on $\boldsymbol{x}$, then $\boldsymbol{R} \circ \boldsymbol{g}$ minimizes it on $\boldsymbol{Rx}$ for any orthogonal $\boldsymbol{R}$, since $|\boldsymbol{RJ}| = |\boldsymbol{J}|$ and $\|\boldsymbol{RJ}_i\|_2 = \|\boldsymbol{J}_i\|_2$, and the isotropic noise inflation is invariant under $\boldsymbol{R}$ as well. The objective thus singles out no pixel basis. The simplest orthogonal $\boldsymbol{R}$ is a permutation matrix $\boldsymbol{P}$ that shuffles the pixel positions, and Fig. G.6 shows that our network inherits the property: trained on CelebA with every image shuffled by the same fixed random permutation, so that no pixel neighborhood survives and every image looks like noise, it learns the same features once they are shuffled back.

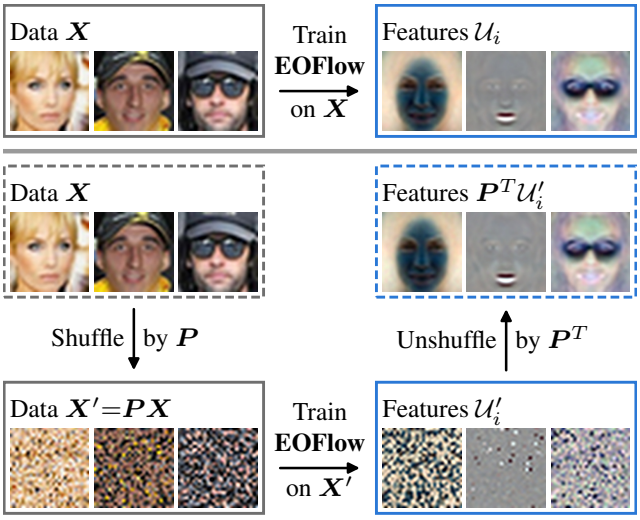


Figure G.6: **An EOFlow trained on shuffled pixels learns the same features.** Top: trained on CelebA $\boldsymbol{X}$, the EOFlow learns the features $\mathcal{U}_i$ (archetypes of three coordinates: skin tone, mouth and eyeglasses). Bottom: trained on $\boldsymbol{X}' = \boldsymbol{PX}$, with the pixel positions shuffled by a fixed random permutation $\boldsymbol{P}$ so that every image looks like noise, it learns features $\mathcal{U}'_i$ that are the same features once un-shuffled by $\boldsymbol{P}^T$.

We train the same EOFlow ($\lambda = 1$, $\sigma_\epsilon = 0.05$, 100 epochs, batch size 128) on $\boldsymbol{PX}$ with four seeds (101–104), where $\boldsymbol{P}$ applies one fixed random permutation of the 784 pixel positions to all three color channels, and compare with two plain runs of the same recipe (seeds 101 and 102). Everything the shuffled models produce in pixel space is mapped back with $\boldsymbol{P}^T$, which is exact for the stability protocol as well, since the polar factor of $\boldsymbol{JP}^T$ is the polar factor of $\boldsymbol{J}$ times $\boldsymbol{P}^T$. The shuffled runs are twins of the plain ones: their five leading manifold entropies agree with those of the plain runs to within 0.1 nats, and the mean ENZ of their 50 leading coordinates (0.34–0.36) matches that of the plain runs (0.34 and 0.35).

Measured with the protocol of App. F.7.3, a shuffled run reproduces $65.6_{\pm 4.0}$ coordinates of a plain run (eight pairs, 57–69), as many as the two plain runs reproduce of each other (63) and as the shuffled runs reproduce among themselves ($62.5_{\pm 5.8}$, six pairs). A change of the pixel basis therefore costs nothing beyond the seed-to-seed variation, and the agreement does not depend on whether the two runs share a seed, as the permutation scrambles the model basis in which the seed lives. A Haar-random rotation $\boldsymbol{R}$, which mixes all pixels instead of moving them, is the harder variant of this test: two such runs reproduce 52.5 coordinates of their plain twins (55 and 50) and 45 of each other, about 10–18 fewer, which is consistent with the architecture being only approximately equivariant, since coupling partitions, permutations and ActNorm act on individual coordinates, while the objective is equivariant exactly.

Fig. G.7 shows the 100 highest-entropy archetypes of the shuffled seed 101 after un-shuffling. They are the familiar global and local features, although the model never saw a pixel neighborhood: the sparsest matched coordinates reach an ENZ of 0.06, among them the mouth at entropy rank 59 (matched $\cos^2$ 0.86, ENZ 0.06) and the eyes at rank 48 (0.59, ENZ 0.14). In the basis the model was trained in, the same coordinates are scattered pixels (Fig. G.7, bottom). Since neither architecture nor

objective favors local pixel neighborhoods, local features such as mouth and eyes must stem from the data distribution together with the orthogonality bias.

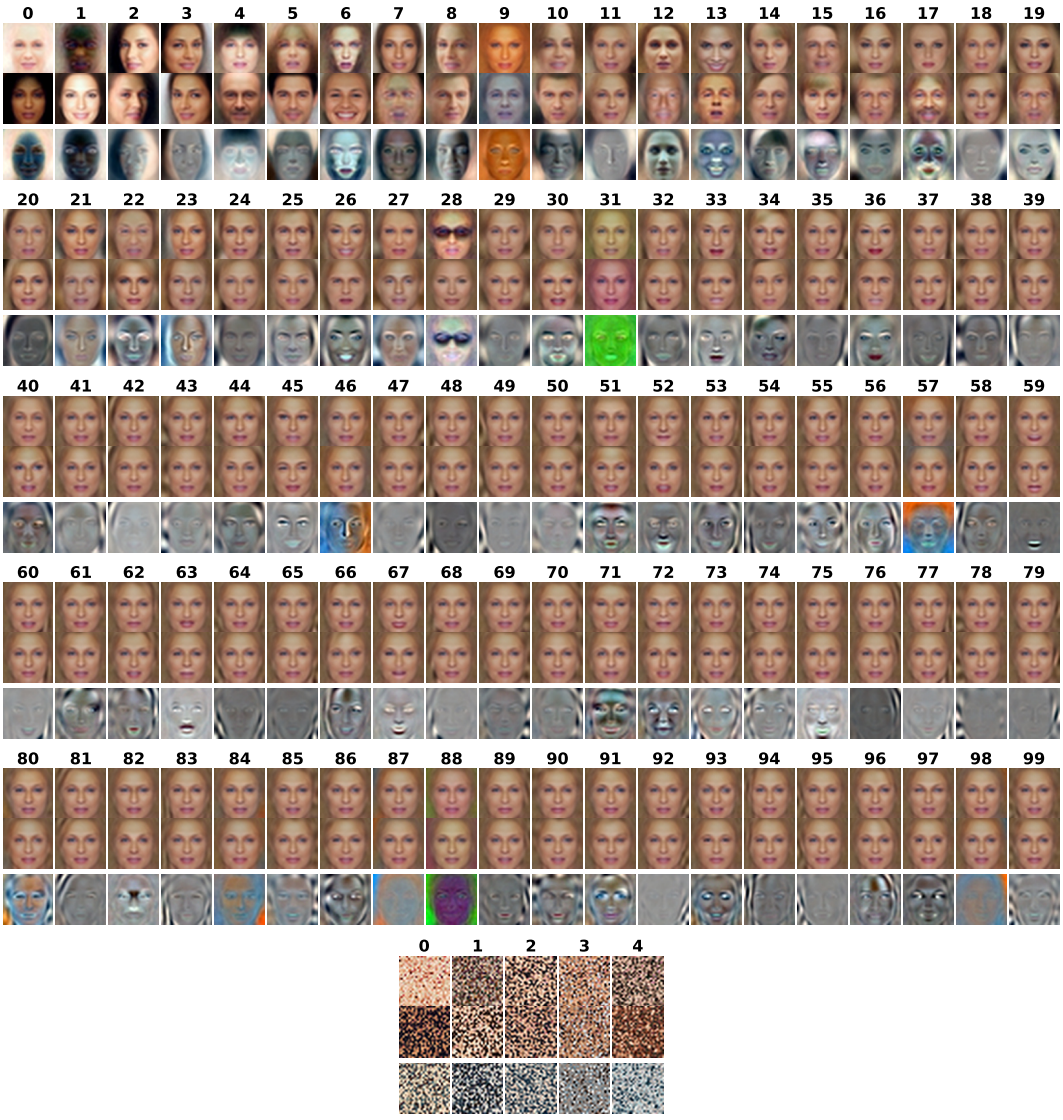


Figure G.7: **Archetypes: EOFlow trained on shuffled pixels ($\lambda = 1$, 100 ep, seed 101).** Top: the 100 highest-entropy latents after un-shuffling with $\boldsymbol{P}^T$, in the format of App. G.10 (number above each triplet is the entropy rank). Bottom: the five leading latents in the basis the model was trained in, each image normalized to $[0, 1]$.

G.5 RATE-DISTORTION PLOTS

Fig. 4b reports the denoising error of held-out images reconstructed through the top-C bottleneck (noisy input, clean target), together with the denoising autoencoders of App. F.6. Fig. G.8 reports, for the EOFlows and PCA, the FID of prior samples drawn through the bottleneck (no denoising step) as a function of C, and, as horizontal lines, the FID of samples from the full prior after one Tweedie step. The bottleneck-prior FID has a minimum at $C = 100$–200 for every λ (19, 20 and 24 at $\lambda = 0.01$, 0.1 and 1), below the full-prior values, whereas PCA is best at $C = 100$–200 at 38.

The PCA spectrum. In Fig. 4a, PCA is treated as a linear NF with one latent per principal direction of the noise-inflated data, so its manifold entropies are $H_i = \frac{1}{2}\log\left(2\pi e\left(s_i^2 + \sigma_\epsilon^2\right)\right)$, with s_i^2 the variance of the clean training data along the i-th principal direction. Among all distributions with a given variance, the Gaussian has the largest entropy. The data distribution along a principal direction therefore has at most the entropy of the corresponding PCA coordinate, and a model that fits the data better than the Gaussian needs less entropy in total. For EOFlows, whose orthogonal coordinates make the manifold entropies add up to the model entropy, this holds on CelebA: their manifold entropies sum to -1.29 nats per dimension against -1.23 for PCA, in line with the held-out likelihoods of Fig. 6a. The bound concerns the total and not single ranks, as the leading EOFlow coordinates carry more entropy than the leading principal directions, and for non-orthogonal models the manifold entropies do not add up to the model entropy.

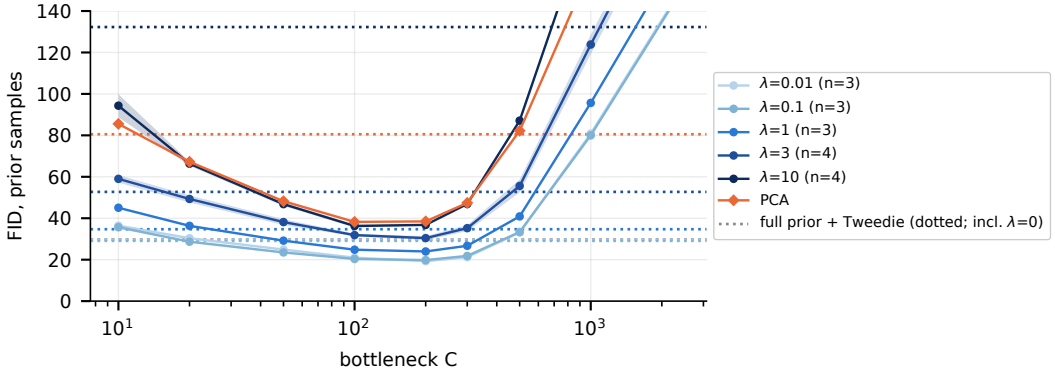


Figure G.8: FID of prior samples through the top-C bottleneck (no Tweedie step; mean $\pm$ sd over seeds) for EOFlows at 300 epochs and PCA, and of full-prior samples after one Tweedie step (dotted lines, including the plain NF).

Table G.4: **Attribute-based disentanglement (TAD)** (Yeats et al., 2022) on the CelebA validation split (higher is better; mean ± sd over seeds). Flows: same architecture and recipe; VAEs: best tuned cell of each family (App. F.4); number of seeds in parentheses. The weakly regularized $\lambda = 0.1$ scores highest; $\lambda = 1$ ties the best VAE with far less seed scatter.

model	epochs	TAD
plain NF ($\lambda = 0$)	300	0.00 ± 0.00 (3)
EOFlow $\lambda = 0.01$	300	0.48 ± 0.21 (3)
EOFlow $\lambda = 0.1$	300	1.00 ± 0.01 (3)
EOFlow $\lambda = 1$	300	0.72 ± 0.01 (3)
EOFlow $\lambda = 3$	300	0.60 ± 0.05 (4)
EOFlow $\lambda = 10$	300	0.45 ± 0.03 (4)
PCA	–	0.52 (1)
NDFlow $\nu = 0.1$	300	0.58 ± 0.07 (3)
NDFlow $\nu = 1$	300	0.24 ± 0.01 (3)
NDFlow $\nu = 10$	300	0.28 ± 0.01 (3)
FactorVAE $\gamma = 10$	300	0.72 ± 0.12 (3)
β-TCVAE $\beta = 4$	300	0.44 ± 0.01 (3)
β-VAE $\beta = 4$	300	0.46 ± 0.03 (3)
AnnealedVAE $C = 25$	100	0.40 ± 0.10 (5)

G.6 Correlation with CelebA attributes

Table G.4 reports the attribute-based disentanglement score TAD (Yeats et al., 2022) of every model on $10{,}000$ validation images. For each of the 40 binary CelebA attributes, every latent (flows and PCA: all D, VAEs: all 32) is scored by its AUROC as a single-feature classifier of the attribute, irrespective of sign. An attribute counts as captured if its best latent reaches an AUROC of at least 0.75 and it is nearly independent of the other attributes (mutual information with each of them at most $20\,\%$ of its entropy); TAD sums, over the captured attributes, the AUROC margin between the best and the second-best latent. It thus rewards attributes that are each isolated in a single latent.

G.7 Stable features across seeds

Fig. G.9 extends the tiles of Fig. 5 from 7 features in three seeds to the 100 most stable features of EOFlow $\lambda = 1$ after 500 epochs in all six seeds. With seed 105 as the reference, as in Fig. 5, every latent of the reference is matched to one latent of each other seed by the protocol of App. F.7.3, and the reference latents are ranked by their mean matched $\cos^2$ over the five other seeds; 124 of them reach a mean of at least 0.5. The matched latents show the same feature in every seed and sit at a similar place in the entropy order: their entropy ranks differ from that of the reference by a median of 4. Stable features are not confined to the highest entropies: 57 of the 100 lie among the 100 highest-entropy latents of the reference and 89 among the 300 highest, the deepest at rank 732. Of the 100, 55 are matched at $\cos^2 \geq 0.5$ in all five other seeds and 89 in at least four; the standard deviations mark the features that some seeds reproduce less well.

G.8 Generating samples through a bottleneck

As EOFlows learn a meaningful ordering of the inflated data from core to detail, it is possible to sample from the core prior and set the detail latents to their mean, i.e. to zero. This artificial bottleneck resembles an injective model, which should only generate "clean" samples, i.e. with little to no noise:

$$\boldsymbol{x} = \boldsymbol{g}\big([\boldsymbol{z}_{\leq C}, \boldsymbol{z}_{>C}]\big), \quad \boldsymbol{z}_{\leq C} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_C),\ \boldsymbol{z}_{>C} = \boldsymbol{0} \tag{G.1}$$

Fig. G.10 decodes the same 20 fixed prior codes with every model, one panel per model: from the full prior, after one Tweedie denoising step $\boldsymbol{x} + \sigma_\epsilon^2 \nabla \log q(\boldsymbol{x})$ on these samples, and through bottlenecks that keep the $C = 100$ or $C = 300$ highest-entropy latents and set the others to zero. It covers the plain NF, EOFlows with $\lambda \in \{0.01, 0.1, 1, 3, 10\}$, PCA and NDFlows with $\nu \in \{0.1, 1, 10\}$, all at 300 epochs (seed 103; NDFlow $\nu = 10$: seed 106), and the four VAEs of App. F.4. For PCA, the full-prior samples are Gaussian with the covariance of the noisy data, the Tweedie step is their

posterior mean, and a bottleneck samples only the top C components. The VAEs decode the first 32 entries of each code and give full-prior samples only: both bottlenecks exceed their 32 latents, and their Bernoulli decoder defines no density in pixel space for a Tweedie step. The plain NF has no meaningful latent order, so its bottlenecks keep arbitrary latents and fail. For EOFlows and NDFlows, the bottleneck removes the noise without a denoising step because the discarded coordinates are the ones below the noise entropy; FID as a function of C is given in App. G.5 (Fig. G.8).

G.9 Reconstructions

Figs. G.11–G.17 show twelve held-out images, their noise-inflated version ($\sigma_{\boldsymbol{\epsilon}} = 0.05$) and the reconstruction of the noisy version through bottlenecks $C \in \{5, 10, 20, 50, 100, 500\}$, through the full latent space (which returns the noisy input) and after one Tweedie step, for the plain NF, EOFlows with $\lambda \in \{0.01, 0.1, 1, 3, 10\}$ and PCA (300 epochs, seed 103). The plain NF cannot reconstruct through any bottleneck, because its coordinates carry no ordering. EOFlows recover most of the image at $C = 50$ and add detail up to $C \approx 500$; larger bottlenecks reintroduce the noise. None of these flows sees a clean image or a reconstruction loss during training.

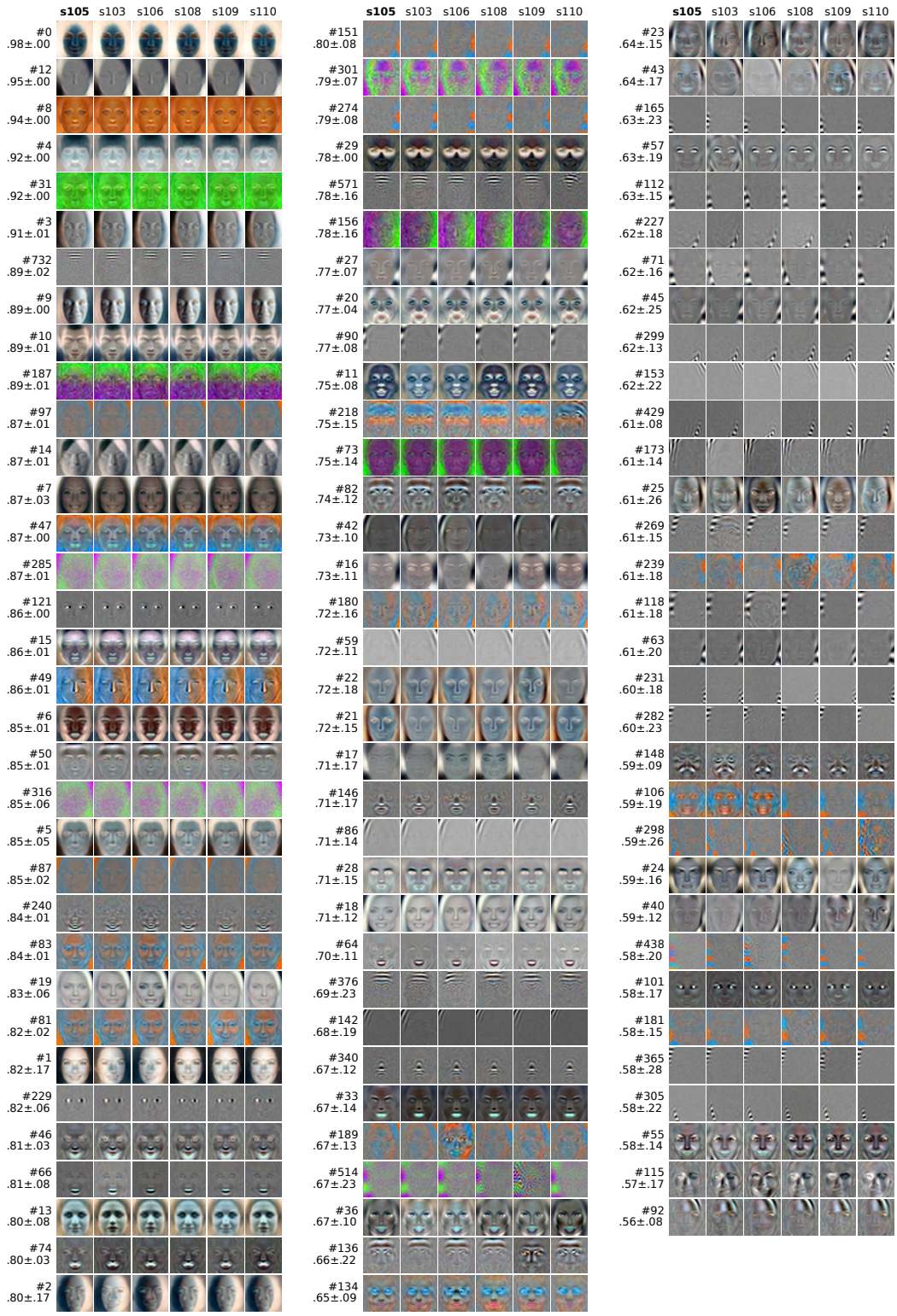
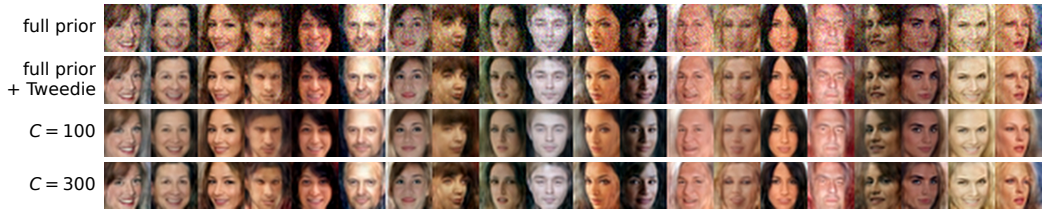


Figure G.9: **The** 100 **most stable features** of EOFlow $\lambda = 1$ after 500 epochs in all six seeds. The latents of the reference seed 105 are ranked by their mean matched stability over the five other seeds, column by column; each row shows the difference images $\boldsymbol{x}_{i+4} - \boldsymbol{x}_{i-4}$ of the matched latents in all six seeds (reference first, sign flips resolved). Left of each row: the entropy rank of the feature in the reference seed (#) and the mean ± sd of its matched $\cos^2$ over the five seed pairs.



(a) plain NF ($\lambda = 0$)



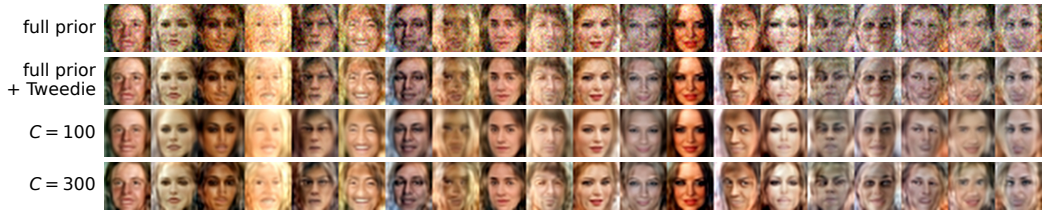
(b) EOFlow $\lambda = 0.01$



(c) EOFlow $\lambda = 0.1$



(d) EOFlow $\lambda = 1$



(e) EOFlow $\lambda = 3$



(f) EOFlow $\lambda = 10$

Figure G.10: **Prior samples** of every model from the same 20 prior codes: decoded from the full prior, after one Tweedie step on these samples, and through bottlenecks that keep the $C = 100$ or $C = 300$ highest-entropy latents (300 epochs, seed 103). The plain NF has no meaningful latent order, so its bottlenecks keep arbitrary latents. Continued on the next page.



(g) PCA of the noise-inflated data



(h) NDFlow $\nu = 0.1$



(i) NDFlow $\nu = 1$



(j) NDFlow $\nu = 10$ (seed 106)



(k) FactorVAE $\gamma = 10$



(l) β-TCVAE $\beta = 4$



(m) β-VAE $\beta = 4$



(n) AnnealedVAE $C = 25$ (100 epochs)

Figure G.10: **Prior samples** (continued): PCA (Gaussian samples with the covariance of the noisy data, their posterior mean, and samples of the top C components), NDFlows (300 epochs, seed 103 unless noted) and the VAEs of App. F.4 (300 epochs, seed 101 unless noted), which decode the first 32 entries of each code and give full-prior samples only.



Figure G.11: Reconstructions through the bottleneck, plain NF ($\lambda = 0$).



Figure G.12: Reconstructions through the bottleneck, EOFlow $\lambda = 0.01$.



Figure G.13: Reconstructions through the bottleneck, EOFlow $\lambda = 0.1$.



Figure G.14: Reconstructions through the bottleneck, EOFlow $\lambda = 1$.



Figure G.15: Reconstructions through the bottleneck, EOFlow $\lambda = 3$.



Figure G.16: Reconstructions through the bottleneck, EOFlow $\lambda = 10$.

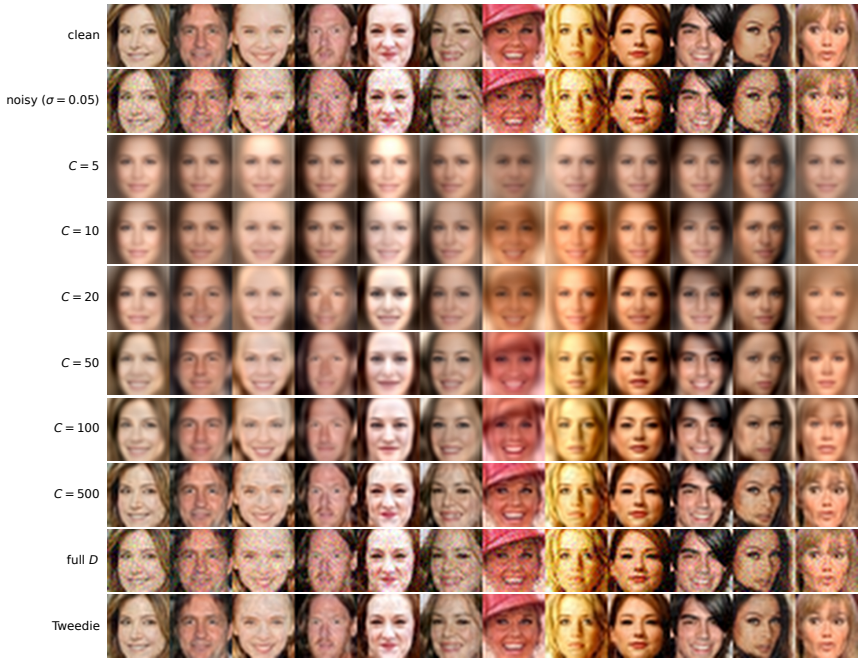


Figure G.17: Reconstructions through the bottleneck, PCA (projection onto the leading C components; Tweedie = posterior mean of the Gaussian model).

G.10 ARCHETYPES

Archetypes visualize feature i by moving from the latent origin $\boldsymbol{z} = \mathbf{0}$ along its coordinate line $\mathcal{U}_i$ by 4 standard deviations in both directions,

$$\boldsymbol{x}_{i+4} \coloneqq \boldsymbol{g}([\boldsymbol{z}_{<i} = \mathbf{0}, z_i = +4, \boldsymbol{z}_{>i} = \mathbf{0}]), \qquad \boldsymbol{x}_{i-4} \coloneqq \boldsymbol{g}([\boldsymbol{z}_{<i} = \mathbf{0}, z_i = -4, \boldsymbol{z}_{>i} = \mathbf{0}]), \quad \text{(G.2)}$$

and by their difference image $\boldsymbol{x}_{i+4} - \boldsymbol{x}_{i-4}$, which is min–max normalized to $[0, 1]$ per tile for display.

Figs. G.18–G.31 show the 100 leading archetypes, with both decoded extremes and their difference image, of one run per model configuration at the 300-epoch budget: the plain NF, EOFlows with $\lambda \in \{0.01, 0.1, 1, 3, 10\}$ and PCA, then NDFlows with $\nu \in \{0.1, 1, 10\}$ and the VAEs of App. F.4 (AnnealedVAE at its 100-epoch budget); VAEs have 32 latents and are ordered by their entropy H_i (App. F.4), PCA by eigenvalue with ± 4 standard deviations along each component. Fig. 1 shows hand-picked features of another seed (105) of the $\lambda = 1$ setting.

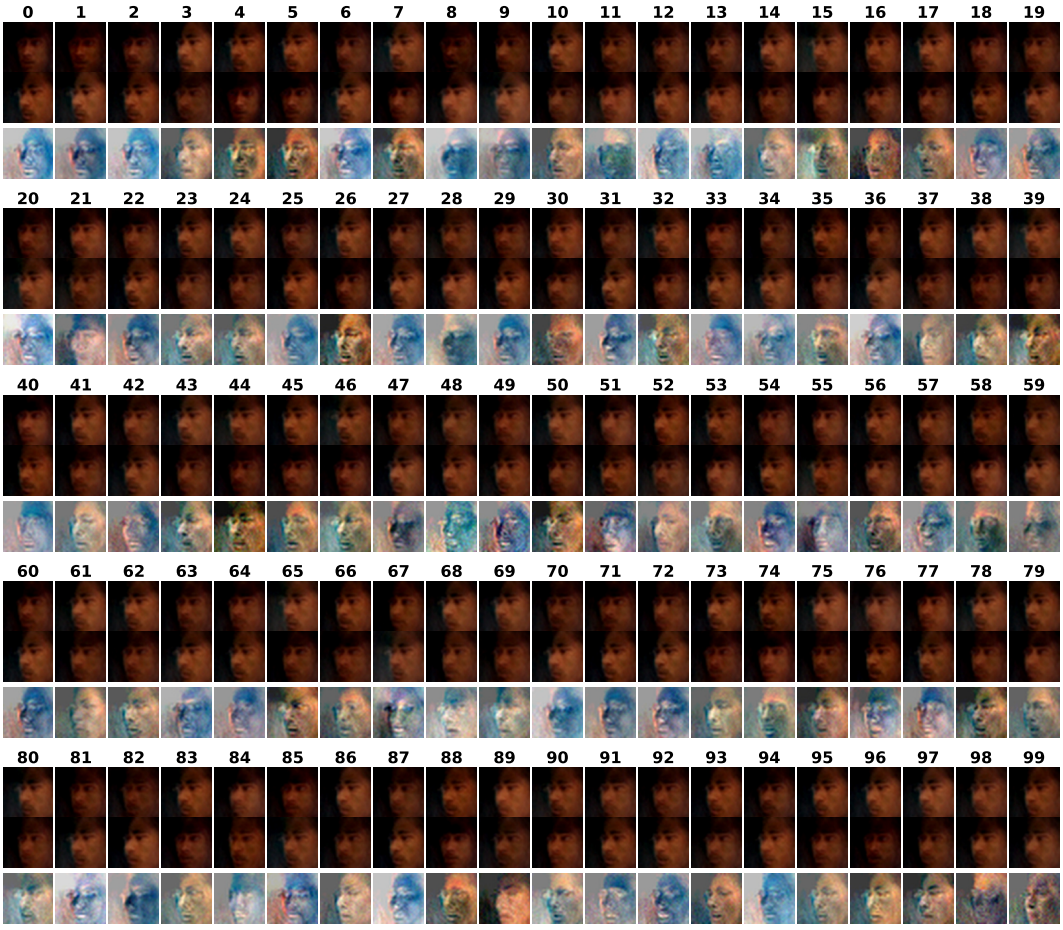


Figure G.18: **Archetypes: plain NF, $\lambda = 0$ (300 ep, seed 103).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).

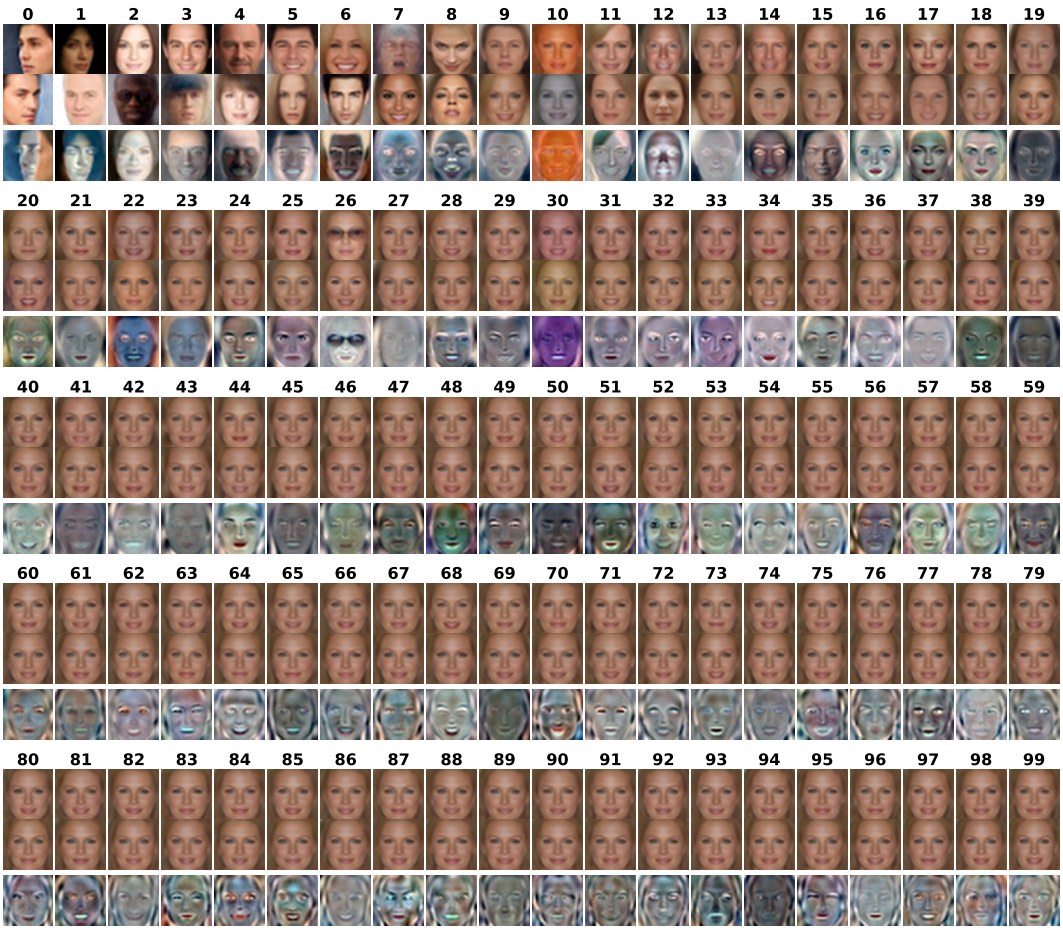


Figure G.19: **Archetypes: EOFlow $\lambda = 0.01$ (300 ep, seed 103).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).

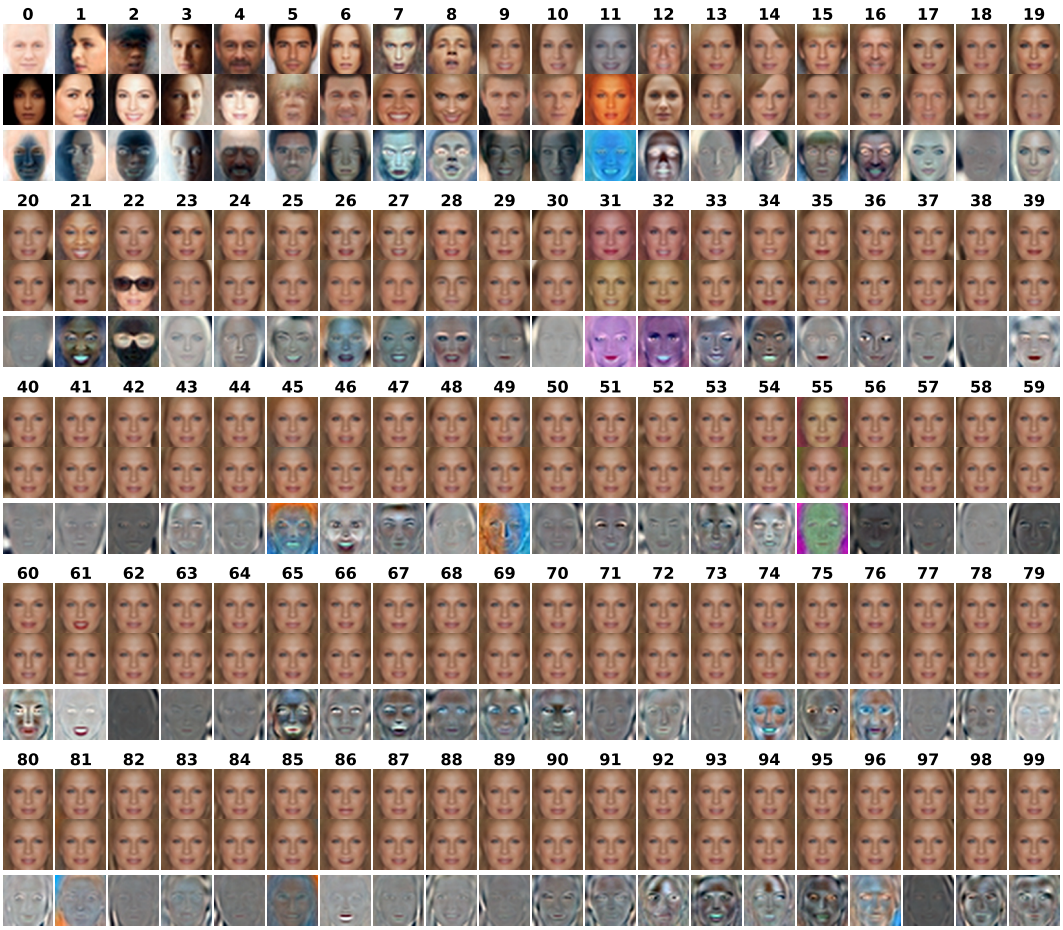


Figure G.20: **Archetypes: EOFlow $\lambda = 0.1$ (300 ep, seed 103).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).



Figure G.21: **Archetypes: EOFlow $\lambda = 1$ (300 ep, seed 103).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).



Figure G.22: **Archetypes: EOFlow $\lambda = 3$ (300 ep, seed 103).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).

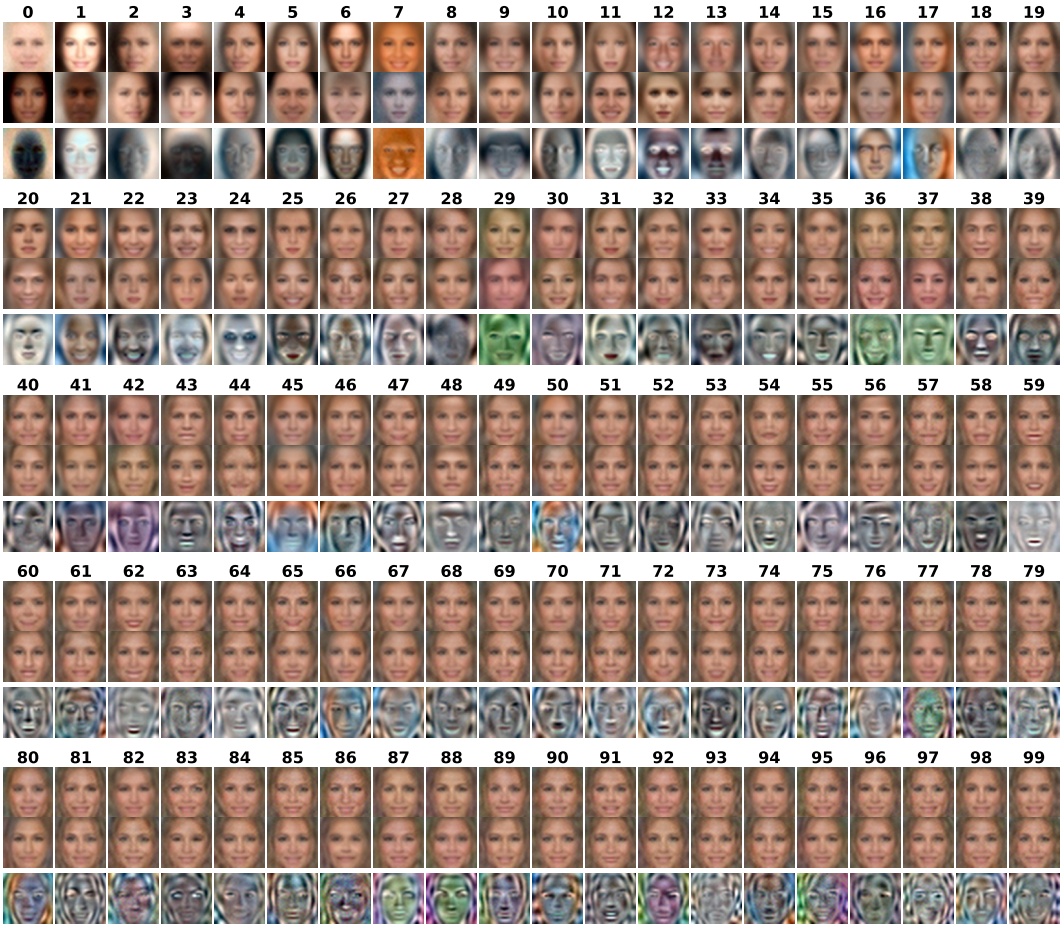


Figure G.23: **Archetypes: EOFlow $\lambda = 10$ (300 ep, seed 103).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).

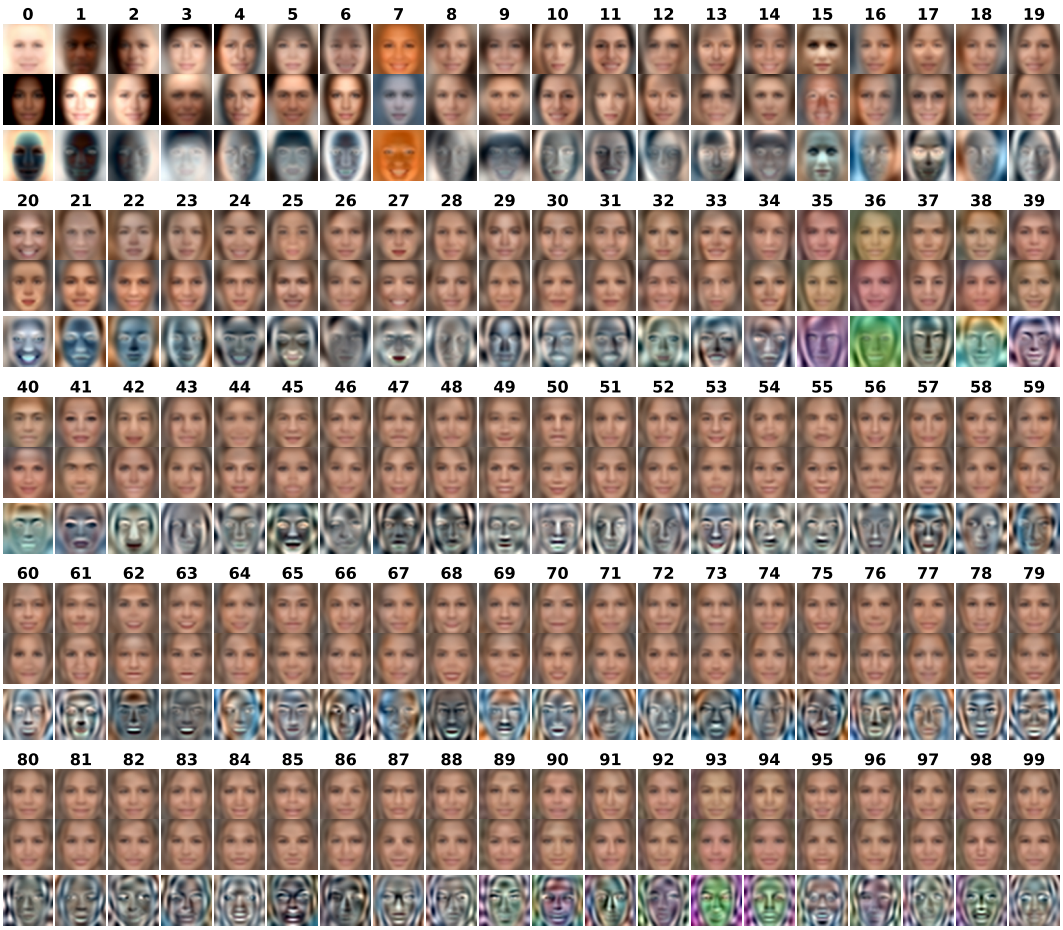


Figure G.24: **Archetypes: PCA.** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).



Figure G.25: **Archetypes: NDFlow $\nu = 0.1$ (300 ep, seed 103).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).

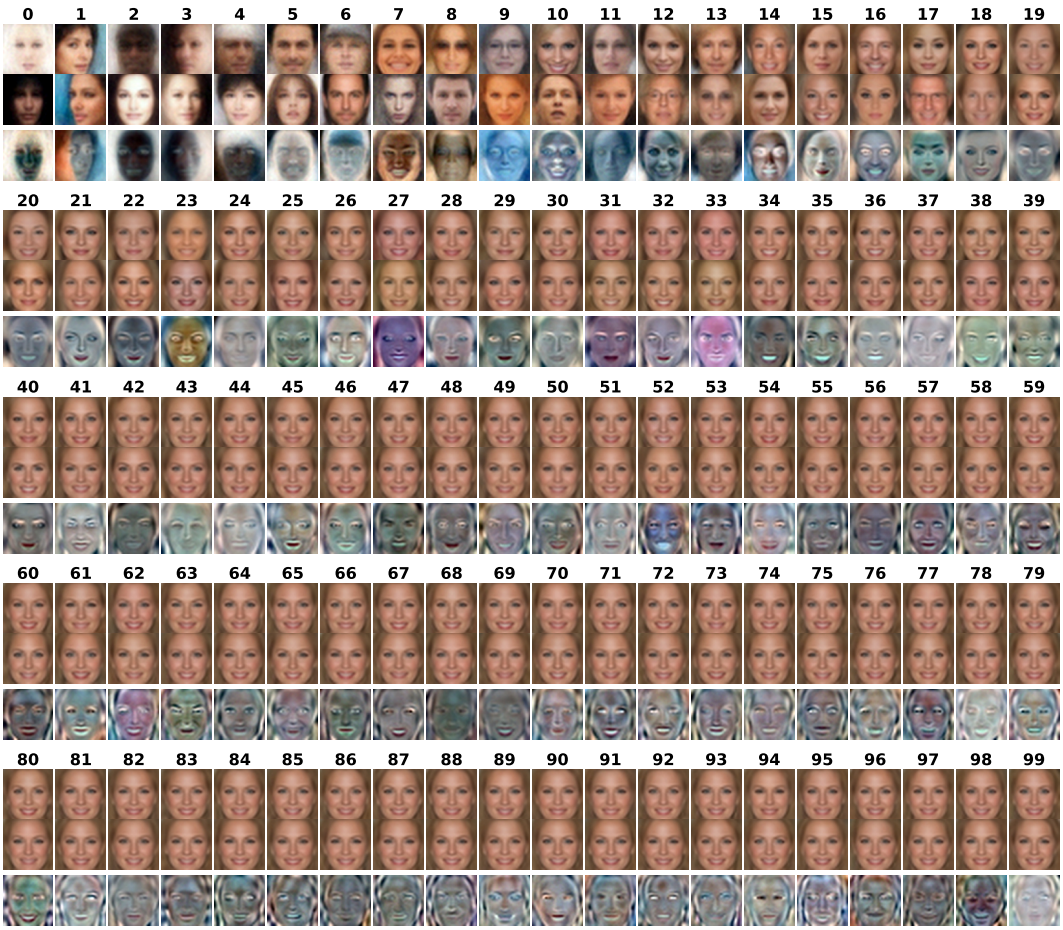


Figure G.26: **Archetypes: NDFlow $\nu = 1$ (300 ep, seed 103).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).



Figure G.27: **Archetypes: NDFlow $\nu = 10$ (300 ep, seed 106).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).

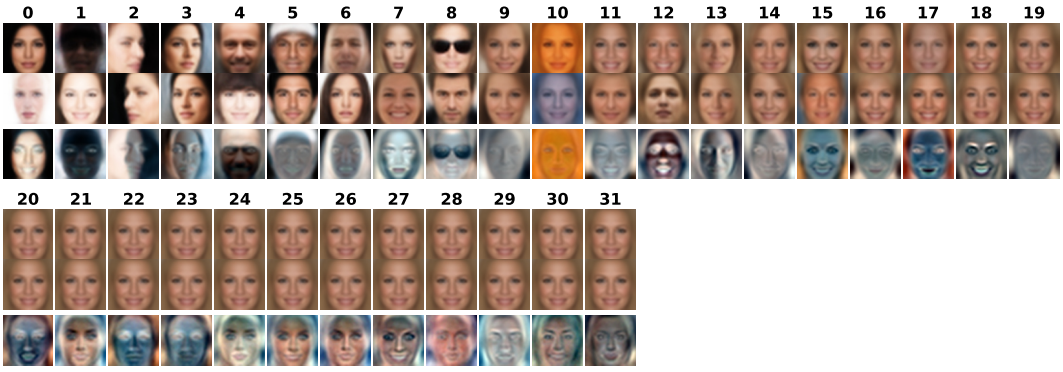


Figure G.28: **Archetypes: FactorVAE $\gamma = 10$ (300 ep, seed 101).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).

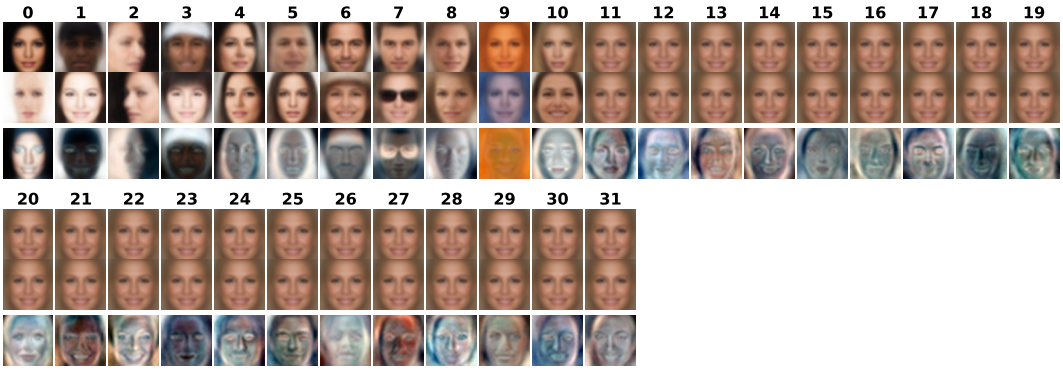


Figure G.29: **Archetypes: β-TCVAE $\beta = 4$ (300 ep, seed 101).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).

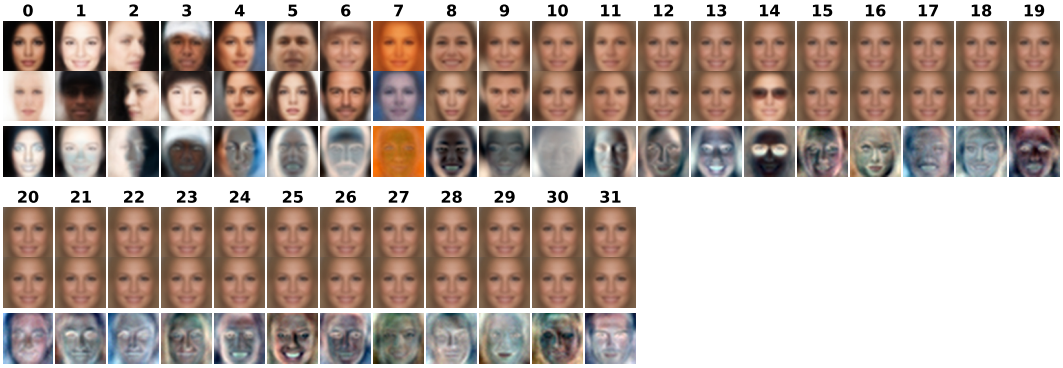


Figure G.30: **Archetypes: β-VAE $\beta = 4$ (300 ep, seed 101).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).

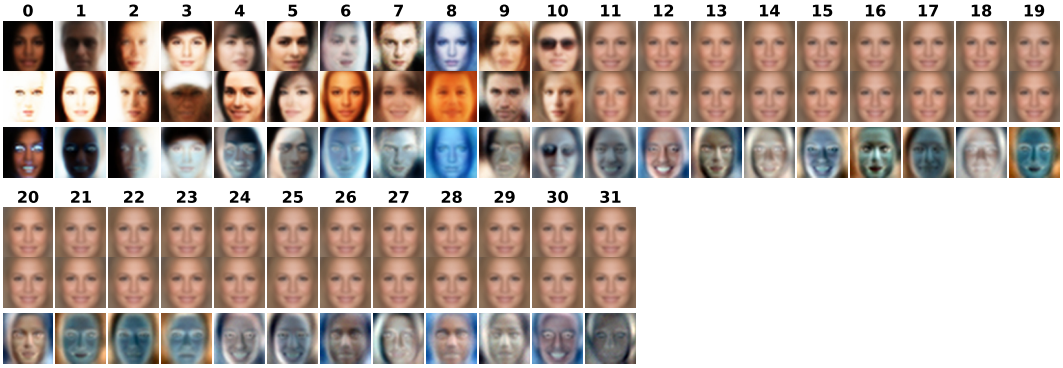


Figure G.31: **Archetypes: AnnealedVAE $C = 25$ (100 ep, seed 101).** For every latent i (number above each triplet, ordered by decreasing manifold entropy): decoded image at $z_i = +4$ (top), at $z_i = -4$ (middle) and their difference image, normalized per tile (bottom).