FLOWSAE: NON-INVASIVE SPARSE AUTOENCODING WITH LIFTED INVERTIBLE FLOWS

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse autoencoders (SAEs) are used both to interpret neural network representations and to intervene on them, but a standard SAE's decoder is only an approximate inverse of its encoder, so reconstruction error contaminates any causal effect an intervention is meant to isolate. Prior work compensates for this after the fact, by carrying the error as an external residual channel or by normalising away its damage, rather than removing it. We introduce FlowSAE, to our knowledge the first nonlinear, multi-layer flow-based SAE whose decoder is the exact inverse of its encoder. FlowSAE lifts the activation into a larger space, applies an invertible flow, sparsifies, and inverts. Top-k sparsification is the only lossy step. When no feature is removed, the input is recovered exactly for any learned parameters, not only after training. On ImageNet ViT features this holds to numerical precision. It also removes a problem we find in classical SAEs, where uncorrected single-feature ablations can invert the true attribution ranking. At the deployed sparsity, however, exactness does not carry over. The nonlinear flow reconstructs better than a linear exact control, but its feature effects become non-additive, so the standard error-term correction, which is exact for any linear decoder, no longer recovers them. Exact reconstruction adds no asymptotic cost per forward pass, only a constant factor. Making sparse interventions trustworthy is a separate problem.

1 INTRODUCTION

Sparse autoencoders (SAEs) decompose neural activations into a larger set of potentially interpretable features (Cunningham et al., 2024; Bricken et al., 2023; Gao et al., 2025). They are often used by intervening on individual features, reconstructing the activation, and measuring the downstream effect (Bricken et al., 2023; Templeton et al., 2024; Karvonen et al., 2024). The reconstruction is not generally exact, because the decoder is only an approximate inverse of the encoder (Gao et al., 2025; Rajamanoharan et al., 2024). Even with no features removed, the reconstructed activation differs from the original and can itself affect the model (Templeton et al., 2024; Karvonen et al., 2024). If this error is on the same scale as the feature effect, the two are difficult to separate. We call a round trip *non-invasive* when retaining all features leaves the downstream computation unchanged.

We ask whether an SAE can make the full-code round trip exactly the identity, independent of the learned parameters, and whether this helps mechanistic interventions. A classical SAE learns an overcomplete encoder–decoder pair with a k-sparse bottleneck. The two maps are learned separately, so retaining all coordinates does not give an exact round trip. Prior work accounts for the residual instead of removing it (Sec. 2).

FlowSAE removes the residual architecturally. We lift the activation into the overcomplete code space, apply an invertible transformation, sparsify there, and invert back (Fig. 1 and Sec. 3). Sparsification is the only non-invertible step, so keeping every coordinate recovers the input for any parameter values.

This matters for attribution. With every coordinate retained, a classical SAE shifts the activation by as much as the feature effects being measured, and uncorrected single-feature ablations can invert the true ranking. FlowSAE has no such shift at $k = n$. At the sparsity where features are actually read, no architecture is exact. There, whether the standard error-term correction recovers a feature's true effect depends on the decoder being linear, not on it being invertible (Sec. 4.3).

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.

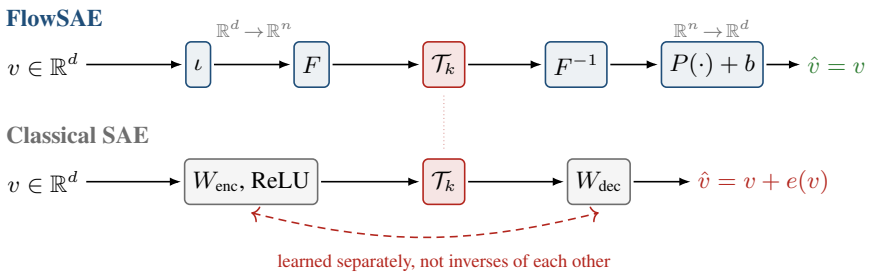

Figure 1: FlowSAE (top) and a classical SAE (bottom) sparsify with the same truncation $\mathcal{T}_k$ (red). The classical encoder and decoder are learned separately, so even at $k = n$, with no feature removed, the round trip returns $\hat{v} = v + e(v)$ with $e(v) \neq 0$ (FVU 0.090–0.225, Fig. 2). In FlowSAE the lift ι, the flow F, its inverse, and the projection $P(\cdot) + b$ compose to the identity (blue), so $\hat{v} = v$ at $k = n$ for any parameters.

Our contributions are threefold.

- *An SAE whose reconstruction is exact by construction.* Sparsifying inside a lifted invertible flow makes the full-code round trip the identity for any learned parameters. Exactness alone is known from transform learning (Sec. 2). The new part is combining it with depth, invertible nonlinearities, and a zero-padded lift (Sec. F). Each orthogonal layer is a rank-$2d$ correction to the identity, applied to N samples in $\mathcal{O}(Nnd)$. The Cayley parametrisation loses the guarantee during training and the exponential map keeps it (Sec. C).
- *Exactness removes a correction step at full capacity.* Without correction, single-feature ablations reverse the attribution ranking for all three classical SAEs we test (Spearman ρ as low as -0.83). FlowSAE avoids this at $k = n$ by construction. Subtracting a per-sample offset repairs the classical SAEs as well, so the gain is that no correction is needed, not a new attribution capability.
- *Nonlinearity buys reconstruction and costs additivity.* Against a linear exact control trained with the same objective, the flow reduces FVU at $k = 32$ by $1.8\times$. Its error keeps falling to zero as k grows, while every classical SAE plateaus. Its feature effects at $k = 32$ are not additive, however. After error correction they agree with exact ablations at only $\rho = 0.56$, against 1.0 for every linear decoder.

Exact reconstruction and trustworthy sparse interventions are therefore separate objectives.

2 Related Work

Sparse autoencoders. The standard SAE formulation comes from Cunningham et al. (2024); Bricken et al. (2023). It is an untied linear encoder–decoder with an L1 penalty on the latent code. Later work has mostly changed the sparsification mechanism. Gao et al. (2025) replaced the L1 penalty with a hard top-k bottleneck,

$$z = W_{\text{enc}}(v - b) + b_e, \quad \tilde{z} = \text{top-}k\big(\text{ReLU}(z)\big), \quad \hat{v} = W_{\text{dec}}\,\tilde{z} + b, \tag{1}$$

with an auxiliary "aux-k" loss and unit-norm decoder columns. BatchTopK (Bussmann et al., 2024) allocates the sparsity budget across a batch, while JumpReLU (Rajamanoharan et al., 2024) uses learnable per-feature thresholds. These methods keep the encoder–nonlinearity–decoder structure. In FlowSAE the encoder and decoder are the two directions of one invertible network. Full-code reconstruction is therefore exact, and decoder columns need no normalisation.

Sparsifying transform learning. Exact reconstruction at full capacity is also classical in *sparsifying transform learning* (Ravishankar & Bresler, 2013b; 2015). There, one operator W produces the code and reconstruction applies W^{-1}, or $W^\top$ when W is unitary. Exactness at full support follows directly. Ravishankar et al. (2020) further connect the unitary case to dictionary learning with tied weights. Overcompleteness (Wen et al., 2015; Ravishankar & Bresler, 2013a) and depth (Ravishankar & Wohlberg, 2018) have also been studied (Sec. A). FlowSAE differs in three ways:

a zero-padded lift from a lower-dimensional input, learned invertible nonlinearities, and selection by signed value. Exactness alone is therefore not our claim. SAEBench (Karvonen et al., 2025) already includes a full-rank PCA control that reconstructs perfectly at $L_0 \approx d$. We aim for exactness together with competitive sparse reconstruction (Sec. 4.4).

Reconstruction error in mechanistic analysis. Reconstruction error has also been addressed directly in mechanistic analysis. Templeton et al. (2024) decompose $v = \hat{v} + e(v)$ and add the unchanged residual back after feature interventions. This preserves the original activation without an intervention, but makes the residual an explicit pathway outside the features. Karvonen et al. (2024) instead use "loss recovered" to measure reconstruction-induced degradation. Circuit-level methods can also expose the residual as its own node (Ameisen et al., 2025; Marks et al., 2025). These practices differ. Makelov et al. (2024) add the residual back after intervening, while SAEBench omits it in settings where it hurts the metric. FlowSAE makes $\hat{v} = v$ at $k = n$, so no residual is needed there. At $k < n$ it is lossy like any other SAE and needs the same residual pathway (Sec. 4.3).

Invertible networks and normalising flows. Invertible networks and normalising flows provide the architectural background. i-RevNet (Jacobsen et al., 2018), i-ResNet (Behrmann et al., 2019), NICE (Dinh et al., 2015), RealNVP (Dinh et al., 2017), and Glow (Kingma & Dhariwal, 2018) use invertible maps, usually between spaces of equal dimension. FlowSAE lifts a lower-dimensional activation into an overcomplete space, truncates there, and projects back after inversion. It is trained to reconstruct frozen representations, not to estimate densities. Concurrent work by Kubaty et al. (2026) applies top-k inside a single rotation $\exp(A - A^\top)$ with $n = d$, without lift, depth, or nonlinearities. Coupling flows fix which coordinates condition the others, which does not suit top-k selection (Sec. B).

Tied weights and structured $\mathrm{SO}(n)$ parametrisations. Tied-weight autoencoders set $W_{\text{dec}} = W_{\text{enc}}^\top$ (Vincent et al., 2008; Rifai et al., 2011), but the two maps are exact inverses only when the encoder is orthonormal, corresponding to our linear control (Sec. 4). FlowSAE instead ties the two directions through a single invertible map. Orthogonal maps can be parametrised using Householder products, the Cayley transform (Helfrich et al., 2018), or the exponential map (Lezcano-Casado & Martínez-Rubio, 2019). We study the Cayley–exponential trade-off in Sec. C. Our thin parametrisation contains a $d(d+1)/2$-dimensional gauge, leaving $nd - d(d+1)/2$ effective parameters (Sec. D).

3 FlowSAE: lift, flow, truncate, invert

FlowSAE separates representation learning from information loss. It maps the activation into a larger space with an invertible transformation, sparsifies there, and applies the inverse. Sparsification is the only lossy step.

3.1 Lifted invertible flow

Let $v \in \mathbb{R}^d$ be an activation and let $n > d$ be the code dimension. We centre v with a learned bias $b \in \mathbb{R}^d$ and lift it by zero-padding:

$$\iota(v) = E(v - b) = (v - b; 0_{n-d}) \in \mathbb{R}^n, \tag{2}$$

where $E : \mathbb{R}^d \to \mathbb{R}^n$ is the zero-padding operator. We then transform the lifted representation with an invertible flow

$$F = \sigma_L \circ Q_L \circ \cdots \circ \sigma_1 \circ Q_1, \tag{3}$$

where each $Q_\ell \in \mathrm{SO}(n)$ is an orthogonal transformation (Sec. 3.2) and each σ_ℓ is an invertible coordinate-wise nonlinearity (Sec. 3.3). The result is an overcomplete code

$$z = F(\iota(v)) \in \mathbb{R}^n. \tag{4}$$

Sparsity is introduced only after the flow. For a target level k, $\mathcal{T}_k$ keeps the k largest entries by *signed* value and sets the rest to zero:

$$\tilde{z} = \mathcal{T}_k(z). \tag{5}$$

Two details differ from a classical SAE. First, there is no ReLU, so we select by signed value and not by magnitude. The flow can learn to place useful coordinates on the positive side. Second, zero is a convention and not a natural inactive state, because σ_ℓ has a learned shift. It defines where truncated coordinates are sent, in training and at evaluation alike. We leave magnitude selection to future work (Sec. 6.1).

The reconstruction applies the inverse flow and projects back to the original space,

$$\hat{v} = P\big(F^{-1}(\tilde{z})\big) + b, \tag{6}$$

where $P : \mathbb{R}^n \to \mathbb{R}^d$ retains the first d coordinates.

Exact reconstruction at full capacity. The flow is a bijection, and P undoes the zero-padding: $P(Ex) = x$ for every $x \in \mathbb{R}^d$. Truncation is therefore the only step that can lose information (Fig. 1). For $k = n$, $\mathcal{T}_n = \mathrm{id}$ and $F^{-1}(\mathcal{T}_n(F(\iota(v)))) = \iota(v)$, so

$$\hat{v} = P(\iota(v)) + b = v \qquad \text{when } k = n, \tag{7}$$

for every $v \in \mathbb{R}^d$ and every parameter setting, in exact arithmetic. Section 4 measures the floating-point error. For $k < n$, truncation generally moves the code off $F(\iota(\mathbb{R}^d))$, so reconstruction is no longer exact.

3.2 Efficient orthogonal transformations

Because n can be much larger than d, storing an $n \times n$ orthogonal matrix is impractical. We instead construct each layer from a thin matrix $A_\ell \in \mathbb{R}^{n \times d}$. We pad it with zeros and form the skew-symmetric generator B_ℓ,

$$\tilde{A}_\ell = (A_\ell \mid 0_{n \times (n-d)}), \qquad B_\ell = \tilde{A}_\ell - \tilde{A}_\ell^\top, \tag{8}$$

and define the orthogonal transformation with the matrix exponential,

$$Q_\ell = \exp(B_\ell), \qquad Q_\ell^{-1} = Q_\ell^\top = \exp(-B_\ell). \tag{9}$$

We use the exponential map because it is smooth for every generator and has no chart boundary. The Cayley map $(I - B_\ell)(I + B_\ell)^{-1}$ cannot represent rotations with eigenvalue -1 and approaches them only as $\|B_\ell\| \to \infty$. In our runs this breaks the $k = n$ guarantee in fp32 during training, while the exponential map keeps it (Sec. C). Each Q_ℓ acts on a subspace of dimension at most $2d$, so a single layer does not reach all of $\mathrm{SO}(n)$. Depth composes several such rotations.

Although B_ℓ is formally $n \times n$, its rank is at most $2d$, so we can evaluate the exponential without forming the full matrix. Let $E = [I_d; 0] \in \mathbb{R}^{n \times d}$ be the matrix of the zero-padding operator. Writing $B_\ell = UV^\top$ with $U = [A_\ell \mid E]$ and $V = [E \mid -A_\ell]$ gives

$$\exp(B_\ell) = I + U\,\phi(M)\,V^\top, \qquad M = V^\top U, \qquad \phi(M) = \sum_{j \geq 0} \frac{M^j}{(j+1)!}. \tag{10}$$

We compute $\phi(M) = M^{-1}(\exp(M) - I)$. Here $|\det M| = \det(A_{\ell,\mathrm{bot}}^\top A_{\ell,\mathrm{bot}})$, where $A_{\ell,\mathrm{bot}}$ is the bottom $(n-d) \times d$ block of A_ℓ, so M is invertible whenever that block has full column rank. Section D verifies Eq. (10) numerically and gives its cost.

3.3 Invertible coordinate-wise nonlinearities

Orthogonal maps only change coordinates linearly, so we alternate them with coordinate-wise invertible nonlinearities. Each $\sigma_\ell : \mathbb{R}^n \to \mathbb{R}^n$ acts independently on each coordinate,

$$\sigma_\ell(x)_i = s_{\ell,i}\big[(x_i - \beta_{\ell,i}) + c_{\ell,i}\,\mathrm{ReLU}(x_i - \kappa_{\ell,i})\big], \qquad s_{\ell,i} = \mathrm{softplus}(\tilde{s}_{\ell,i}) > 0, \tag{11}$$

with slope s, shift β, and kink κ. The map is piecewise linear, with slopes s and $s(1+c)$ on either side of the kink, and it is invertible iff $c_{\ell,i} > -1$. In floating point we enforce a small margin,

$$c_{\ell,i} = (1 - \varepsilon)\tanh(\tilde{c}_{\ell,i}) \in (-1 + \varepsilon,\, 1 - \varepsilon), \qquad \varepsilon = 10^{-3}. \tag{12}$$

Without it, $\tanh$ can reach -1 in fp32, which flattens one branch and breaks invertibility. The inverse is closed-form (Sec. E). At initialisation $s = 1$, $\beta = \kappa = 0$, and $c = 0$, so σ_ℓ is the identity. We sample $A_\ell \sim \mathcal{N}(0, 1/n)$ entrywise, so the initial flow is a random rotation acting on a subspace of dimension at most $(L+1)d$.

3.4 TRAINING

We train FlowSAE on reconstruction error across multiple sparsity levels. For an activation v, the objective is

$$\mathcal{L}(F, b) = \mathbb{E}_v \Big[\|v - \hat{v}(v; k)\|_2^2 + \|v - \hat{v}(v; k_{\text{target}})\|_2^2 \Big], \tag{13}$$

where $k_{\text{target}} = 32$ and k is resampled for each batch. We draw an octave j uniformly from $\{0, \dots, \log_2 n - 1\}$ and then k uniformly from the integers in $[2^j, 2^{j+1}]$. We optimise with AdamW (learning rate 10^{-3}, weight decay 10^{-2}), batch size 2048, and five warm-up epochs. Training lengths are given per comparison in Sec. 4. The reconstruction loss is averaged over the d feature dimensions. We use no additional sparsity objective such as L1, aux-k, or entropy loss.

4 EXPERIMENTS

We evaluate FlowSAE along three questions. First, does the exact full-capacity reconstruction guarantee survive training in practice? Second, does removing reconstruction error improve measurements that depend on small activation changes? Finally, does exact reconstruction at $k = n$ translate into a more self-consistent representation at the sparsity level used for interpretation? Robustness checks on the training objective, model depth, and parameter count are reported in Sec. J.

4.1 SETUP

We train on CLS-token features from an ImageNet-pretrained ViT-S/16 ($d = 384$), extracted from the 1.28M ImageNet training images, and evaluate on the 50,000 validation features. We use $n = 8192$ and $k_{\text{target}} = 32$, with features globally rescaled so that $\sqrt{d}/\mathbb{E}\,\|v\|_2 \approx 1$. The frozen classifier achieves 81.39% top-1 accuracy on the original features.

Our main model is FlowSAE with four flow layers and about 12.7M parameters. We compare it with a classical top-k SAE with about 6.3M parameters, trained either at fixed $k = 32$ with the standard aux-k objective or with FlowSAE's multi-k objective. We also train the published multi-budget objective, Multi-TopK (Gao et al., 2025), $\mathcal{L}(k) + \mathcal{L}(4k)/8$ with $k = 32$, on the otherwise unchanged single-k recipe. There are two controls. A linear FlowSAE ($L = 1$, $\sigma = \text{id}$) is a tied Parseval-frame pseudo-inverse and is already exact at $k = n$ for any parameters. A classical SAE widened to $n = 16384$ matches FlowSAE's parameter count.

Training schedules. The models of Fig. 2 train for 400 epochs. Ablation, composition, and $k = 32$ attribution comparisons use a separate set of 100-epoch runs, so that every arm has the same budget. Results are single-seed unless a spread is given.

4.2 EXACT RECONSTRUCTION SURVIVES TRAINING

We first test whether the exact-reconstruction property survives optimisation in practice. We replace each ViT feature v by its reconstruction $\hat{v}$ at $k = n$ and pass $\hat{v}$ through the frozen classifier. FlowSAE preserves the original 81.39% top-1 accuracy, with a logit MSE of only 1.98×10^{-23} in fp64. In TF32, the same experiment gives a logit MSE of about 2×10^{-6}, consistent with rounding error from the reduced mantissa precision. The error disappears once TF32 is disabled.

The contrast with a classical SAE is substantial. The classical single-k model, trained with $k = 32$, reaches an FVU of 0.225 when evaluated at $k = n$ and reduces classifier accuracy to 80.52%. FlowSAE's exactness, however, is not something it learns from the training objective. It follows directly from the invertible architecture. To verify this independently, we evaluate an untrained FlowSAE on 4096 random inputs. In fp64, the maximum absolute reconstruction error is 1.6×10^{-13} and the relative L_2 error is 1.1×10^{-14}. The near-zero error observed after training therefore comes from the construction itself, not from the model learning an unusually accurate approximate inverse.

4.3 WHAT EXACTNESS BUYS AN ATTRIBUTION EXPERIMENT

The practical motivation for exact reconstruction is attribution. If an SAE reconstructs the input imperfectly, then an intervention on one latent feature is measured on top of an existing reconstruction

Quantity	FlowSAE	Linear	Classical single-k	Classical multi-k
reconstruction offset $\lvert\delta\rvert$	≈ 0	≈ 0	3.075	1.752
mean $\lvert\Delta^{\text{raw}}\rvert$ (top-32 features)	0.3068	0.5513	2.872	1.966
uncorrected effect floor (bottom 25%)	0.0176	0.0250	2.310	1.410
Spearman ρ, Δ^{raw} vs. gold standard	1.000	$1^{\dagger}$	-0.834	-0.181
Spearman ρ, Δ^{corr} vs. gold standard	1.000	$1^{\dagger}$	1.000	1.000

Table 1: Single-feature ablation at $k = n$, 2000 ImageNet validation samples, one seed. The gold standard is the error add-back estimate. Classical models train for 400 epochs, FlowSAE and the linear control for 100. †Exact by construction ($\hat{v} = v$), not separately measured.

error. When the effects of individual features are small, this background error can become large enough to obscure the intervention itself.

We first isolate this effect at $k = n$. For each sample, we select the 32 coordinates with the largest absolute values, independently of the value-based $\mathcal{T}_k$ rule, and ablate them one at a time. Let $\hat{v}^{(-i)}$ denote the reconstruction after setting coordinate i of the code to zero, and let M_c denote the true-class logit. Over 2000 validation examples, we measure the raw effect

$$\Delta_i^{\text{raw}} = M_c(v) - M_c(\hat{v}^{(-i)}). \tag{14}$$

We also consider an error-corrected version,

$$\Delta_i^{\text{corr}} = M_c(\hat{v}) - M_c(\hat{v}^{(-i)}), \tag{15}$$

which removes the per-sample reconstruction offset $\delta = M_c(v) - M_c(\hat{v})$, since $\Delta_i^{\text{raw}} = \Delta_i^{\text{corr}} + \delta$. For the affine classifier head, Δ_i^{corr} is equivalent to the error add-back estimate of Templeton et al. (2024).

The results in Tab. 1 make the effect of the reconstruction error concrete. For the classical SAEs, the reconstruction offset is large compared with the feature effects themselves, 3.075 for the single-k model and 1.752 for the multi-k model against mean effects of 2.872 and 1.966. As a result, the uncorrected ablation measure can give a substantially different picture of feature importance. Against the gold standard, the ranking is inverted for the single-k model (Spearman $\rho = -0.834$) and for Multi-TopK (-0.62), and close to random for the multi-k model (-0.181).

FlowSAE and the linear control avoid this problem because their reconstruction offset is zero up to numerical precision. Their uncorrected effect floors are therefore 56–131$\times$ smaller than those of the classical models. At the same time, the experiment makes an important qualification clear. Exactness does not create a new attribution method. Once the reconstruction offset is explicitly subtracted, the classical models recover the gold-standard ranking as well. The contribution of exact reconstruction is instead that this correction is no longer needed at the full-capacity endpoint.

Attribution at the deployed budget. The previous experiment deliberately uses $k = n$, where FlowSAE is exact. In practice, however, features are read at a much smaller budget. We therefore repeat the experiment at the deployment budget $k = 32$, where none of the architectures is exact.

For each sample, we take the 32 coordinates selected by $\mathcal{T}_{32}$ and ablate them individually in the sparse code. We then compare the resulting effects with the effect of ablating the same coordinate in the full $k = n$ code, using the error add-back correction where necessary. For FlowSAE and the linear control, this dense intervention provides the exact effect on the original activation. For the classical models, it serves as the gold standard.

The picture changes once sparsity is introduced. As shown in Tab. 2, FlowSAE has a reconstruction offset of 0.585, essentially the same scale as the classical single-k model (0.577). The full-capacity guarantee therefore does not by itself protect an attribution experiment performed at $k = 32$.

The more important difference appears after correcting for this offset. For the linear control and all classical decoders, the corrected single-feature effects reproduce the dense effects exactly. This follows from their linear decoder. Removing one latent coordinate changes the reconstruction by a fixed vector, independent of the other active coordinates. FlowSAE does not have this property. Its corrected attributions reach only $\rho = 0.56 \pm 0.02$, with a relative error of 1.08 ± 0.11 compared with

	$\lvert\delta_{32}\rvert$	ρ, uncorrected	ρ, corrected	rel. gap, corrected
FlowSAE	0.585 ± 0.026	0.23 ± 0.07	$\mathbf{0.56 \pm 0.02}$	$\mathbf{1.08 \pm 0.11}$
Linear control	3.91 ± 0.44	0.86 ± 0.08	1.000	0.000
Classical single-k	0.577 ± 0.007	0.48 ± 0.04	1.000	0.000
Classical multi-k	0.933 ± 0.090	0.33 ± 0.02	1.000	0.000
Multi-TopK*	0.613	0.60	1.000	0.000

Table 2: Single-feature ablation at $k = 32$ on the $\mathcal{T}_{32}$ support, 2000 validation samples, mean $\pm$ sd over three seeds of 100-epoch runs. ρ: Spearman correlation of $|\Delta|$ with the exact dense ablation. Rel. gap: mean $\left|\Delta^{\text{corr}} - \Delta^{\text{dense}}\right|$ relative to mean $\left|\Delta^{\text{dense}}\right|$. *One seed, 400 epochs.

the dense effects. In other words, the nonlinear decoder makes the effect of a coordinate depend on the other active coordinates.

This does not mean that FlowSAE features are unusable for attribution. The strongest feature is still identified in 96% of samples. Rather, it changes how such an attribution should be interpreted. At $k = 32$, a FlowSAE coordinate is not an independent additive component whose effect can be measured in isolation. For clean intervention measurements, our results instead suggest selecting features at the deployed budget and performing the final intervention in the exact $k = n$ representation. The linear control supports this protocol equally well.

4.4 Reconstruction quality across sparsity budgets

The previous experiments establish what exact reconstruction buys us at the full-capacity endpoint. We next ask a different question. What does the nonlinear flow contribute when sparsity is actually introduced?

Figure 2 shows the fraction of variance unexplained (FVU) across sparsity budgets for FlowSAE, the classical single-k and multi-k objectives, and the published Multi-TopK objective. Lower FVU means that the sparse code reconstructs the original activation more accurately.

At the deployment budget of $k = 32$, the methods are close. FlowSAE, Multi-TopK, and the classical single-k specialist have FVU between 0.210 and 0.213, while the classical multi-k model reaches 0.241. At $k = 64$, Multi-TopK has FVU 0.171, compared with 0.175 for the 100-epoch FlowSAE. The nonlinear flow therefore does not provide a clear reconstruction advantage at the smallest budgets used in our experiments.

The difference emerges as the budget grows. Classical SAEs eventually plateau because top-k can only select latents that the ReLU leaves positive. Once k exceeds that number, the additional selected coordinates are zero. The single-k and Multi-TopK models plateau from $k \approx 256$, and the multi-k classical model from $k \approx 512$. FlowSAE does not show the same plateau. Its FVU falls to 0.020 at $k = 1024$ and to 0.010 at $k = 4096$, before reaching approximately zero at $k = n$. At $k = 1024$, for example, the best classical model remains at 0.090. The reconstruction advantage of the flow is therefore real, but it appears mainly at budgets substantially larger than the $k = 32$ operating point.

The lift is also used in practice rather than simply increasing the nominal dimensionality. A flow that ignored the additional dimensions would keep the representation entirely inside the original input subspace. Instead, the trained FlowSAE places 96% of the code's squared norm outside that subspace, and all 32 coordinates retained at the deployment budget lie outside it (Sec. F).

4.5 Exact reconstruction does not imply sparse self-consistency

Exact reconstruction at $k = n$ tells us that the full representation can be inverted perfectly. It does not tell us that a sparse representation will remain unchanged when we encode it again. This distinction matters if we want a selected set of sparse features to describe the same representation consistently.

We test this by reconstructing the input from the $k = 32$ code, re-encoding that reconstruction, and measuring how much of the original top-32 support is recovered. FlowSAE obtains a mean support agreement of 0.890 over six seeds. The corresponding values are 0.917 for the classical single-k

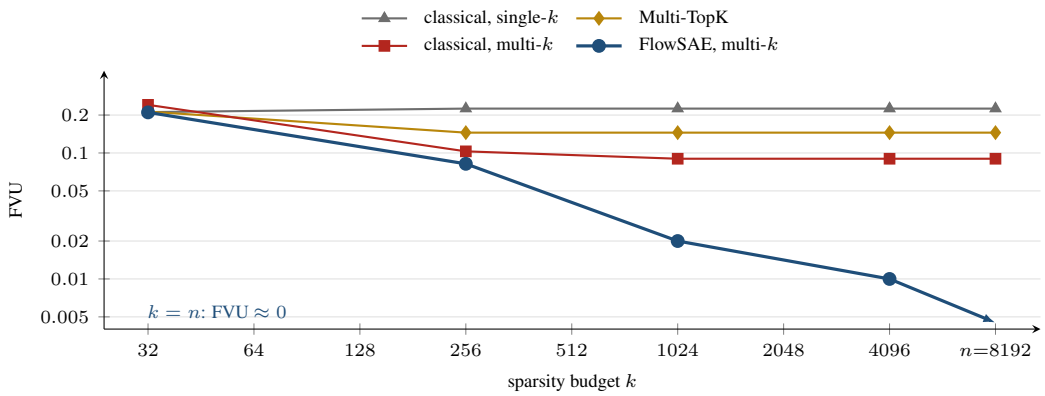

Figure 2: FVU (lower is better) against sparsity budget, held-out features, 400-epoch runs, one seed. Classical SAEs plateau once k exceeds their active ReLU latents. FlowSAE keeps descending and is exact at $k = n$, where FVU $\approx 10^{-23}$ lies below the log axis (arrow). The complete set of FVU values is given in Tab. 4.

SAE, 0.936 for the classical multi-k SAE, and 0.997 for the linear exact control, each averaged over three seeds. Two-sample tests using the unequal-variance standard error give $|t| = 12.6$, 20.4, and 50.4 for the three comparisons. All exceed the most conservative critical value of 4.303 (Sec. H).

The result is useful precisely because it is not what the architectural guarantee would lead us to expect from exactness alone. At $k = n$, support agreement is necessarily 1.0, but that property does not carry over to the sparse regime. The linear control provides a particularly clean comparison. It has the same training loss, optimiser, and exact full-capacity reconstruction property as FlowSAE, yet its sparse support is substantially more stable. The trade-off is that its reconstruction quality is worse, with FVU 0.397 compared with 0.216 for FlowSAE in the matched 100-epoch runs.

5 Discussion

FlowSAE makes one part of sparse autoencoding a guarantee. If no coordinates are removed, reconstruction is exact. Everything that happens after sparsity is introduced is still learned. This distinction turns out to matter. Exactness can be lost in practice if the parameterisation itself becomes numerically unstable. In our experiments training drives the Cayley parameters towards the chart boundary, where they grow without bound, and this destroys the $k = n$ identity (Sec. C). Architectural invariants should therefore be monitored during training, rather than checked only at the end.

More importantly, exactness at $k = n$ says little about the sparse regime we actually care about. At $k = 32$, the linear exact control is much more stable under re-encoding, with support agreement of 0.997 compared with 0.890 for FlowSAE. It is also additive, so after correcting for reconstruction error its single-feature effects reproduce the exact ablations. FlowSAE's corrected effects agree with them at only $\rho = 0.56$ (Tab. 2). Thus, exact reconstruction solves the full-capacity problem, but does not by itself solve sparse composition or sparse attribution.

What is a FlowSAE feature? This raises a more basic question. If FlowSAE has no decoder dictionary, what does an individual coordinate represent? A natural analogue of a dictionary column is the local direction

$$\Delta_i(\tilde{z}) = \partial_{\tilde{z}_i}\big(P(F^{-1}(\tilde{z})) + b\big) \in \mathbb{R}^d.$$

These directions are stable across operating points, with mean cosine similarity of 0.96–0.99. But they are not additive. At $k = 32$, summing the directions of the active coordinates does not recover the reconstruction (Sec. G). The same non-additivity appears in single-feature effects, where the corrected effects differ from the exact ones by 1.08 times their mean magnitude (Tab. 2). A FlowSAE coordinate is therefore better understood as a local direction in a nonlinear representation than as an additive dictionary atom.

What does the nonlinear flow buy us? The linear control provides an important baseline because it isolates what exactness alone can achieve. It has the same exact $k = n$ property, uses only $nd + d$ parameters, and is substantially more self-consistent at $k = 32$. The main advantage of the nonlinear flow is instead reconstruction quality. Under matched 100-epoch training, FlowSAE reduces FVU at $k = 32$ from 0.397 to 0.216, and its error keeps falling towards zero at larger budgets. Against classical SAEs, however, this advantage emerges mainly for $k \gtrsim 256$. At $k = 32$, the single-k specialist and Multi-TopK are comparable (Fig. 2).

This suggests a narrower but useful role for FlowSAE. When sparse representations need to preserve the original activation accurately, particularly at larger budgets, the nonlinear flow provides a clear reconstruction benefit. For small-budget interventions, our results instead point toward the simpler linear construction, which is exact when every coordinate is kept and exactly correctable when not.

A first step beyond vision. The same exact-reconstruction property holds for Pythia-160M-deduped residual-stream activations. Multi-k training also generalises across sparsity budgets there, while a fixed-budget specialist eventually degrades away from its training point (Sec. K). This result is encouraging but limited. No classical baseline has yet been trained on this substrate, so the language-model experiment is a consistency check rather than a second validation of FlowSAE's advantages.

6 Conclusion

Sparse autoencoders are often used to intervene on a model's internal representations by keeping some features, removing others, and measuring what changes. In such experiments, reconstruction should be transparent. If the autoencoder changes the representation even when no feature is removed, part of the measured effect comes from the autoencoder itself rather than from the intervention. FlowSAE makes full-capacity reconstruction an architectural property. By lifting the representation into an overcomplete space, applying an invertible flow, and introducing sparsity only through explicit truncation, it reconstructs the original activation exactly at $k = n$, independently of the learned parameters.

On ImageNet ViT features, this property survives training. At full capacity it removes the reconstruction bias that classical SAEs must correct for when interpreting interventions. This is not unique to the nonlinear flow, since our linear exact control has the same guarantee. The main benefit of the nonlinear construction appears in reconstruction under sparsity, where FlowSAE continues to improve as more coordinates are retained while classical SAEs plateau.

At the same time, exact reconstruction should not be confused with a complete solution to sparse representation learning. Once coordinates are removed, the nonlinear decoder makes feature effects depend on each other. FlowSAE's features are therefore not additive dictionary atoms, and its support is less stable under re-encoding than that of the linear control. Sparse features that are simultaneously reconstructive, additive, and stable remain an open problem.

6.1 Limitations

Our conclusions rest on one setting, ImageNet ViT-S/16 CLS features. On Pythia-160M-deduped no classical or linear baseline has been trained, and the attribution experiments have not been run (Sec. K). The reconstruction curves and the $k = n$ attribution table are single-seed. The composition and $k = 32$ attribution comparisons use three or more seeds per architecture, except Multi-TopK. We report downstream accuracy only at $k = n$, and FVU rather than downstream utility at $k = 32$. Reconstruction fidelity is not evidence of interpretability. Apart from the language-model probing check, we do not evaluate probing or concept selectivity. A FlowSAE feature's effect depends on the other active features (Tab. 2), so analyses that assume additive contributions, such as steering or circuit tracing, do not transfer directly. $\mathcal{T}_k$ selects by signed value, and we do not study selection by magnitude. FlowSAE also uses roughly twice the parameters of the Gao-style baseline, and its per-layer matrix exponential would need revisiting at frontier widths. Finally, we do not explain why the flow is less stable in its support, or less additive, than the linear control.

Reproducibility Statement

All architectural claims in Sec. 3 can be checked from the closed-form expressions given there. The $k = n$ identity is verified directly on randomly drawn, untrained parameters (Sec. 4), and the Cayley and exponential charts are compared in Sec. C. The training recipe (optimiser, learning rate, weight decay, batch size, warm-up, the multi-k sampling distribution, and the margin ε) is given in Sec. 3.4. Dataset preparation and the training length of each comparison are given in Sec. 4. Experimental numbers come from a single seed, except the composition comparison of Sec. 4.5 (six FlowSAE seeds, three per other arm), the $k = 32$ attribution comparison of Sec. 4.3 (three seeds per arm, one for Multi-TopK), and the three-seed language-model check of Sec. K. We give exact rather than rounded figures, for point-by-point comparison with a seeded re-run. Anonymised source code, configuration files, and the evaluation scripts for every table are included as supplementary material.

Ethics Statement

This work analyses publicly available ImageNet features, a publicly released image classifier, and a publicly released language model. It introduces no new data collection and involves no human subjects. It raises no dual-use or fairness concern beyond those already inherent to interpretability tools for pretrained models.

AI Use Statement

Generative AI tools were used for implementation, verification, and writing support. They did not originate the paper's scientific claims. The core architectural findings and negative results were identified and directed by the human authors. AI tools implemented and ran the reproduction and ablation code behind the reported tables, drafted and edited manuscript text, and helped identify and verify citations. AI-assisted code and numbers were checked against unit tests derived from the architectural claims of Sec. 3, against an independent re-derivation of the closed-form expressions of Sec. 3.2, and, for run-derived numbers, against the underlying experiment logs. No generative AI tool is an author of this work, and the authors take responsibility for all reported content.

References

Emmanuel Ameisen, Jack Lindsey, Adam Pearce, et al. Circuit tracing: revealing computational graphs in language models. Transformer Circuits Thread, 2025. https://transformer-circuits.pub/2025/attribution-graphs/methods.html.

Jens Behrmann, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Jörn-Henrik Jacobsen. Invertible residual networks. In *ICML*, 2019.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, et al. Towards monosemanticity: Decomposing language models with dictionary learning. Transformer Circuits Thread, 2023. URL https://transformer-circuits.pub/2023/monosemantic-features.

Bart Bussmann, Patrick Leask, and Neel Nanda. BatchTopK sparse autoencoders, 2024. arXiv:2412.06410.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. arXiv:2309.08600.

Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear independent components estimation. In *ICLR Workshop*, 2015.

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

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*, 2025. arXiv:2406.04093.

Kyle Helfrich, Devin Willmott, and Qiang Ye. Orthogonal recurrent neural networks with scaled Cayley transform. In *International Conference on Machine Learning*, pp. 1969–1978, 2018. arXiv:1707.09520.

Jörn-Henrik Jacobsen, Arnold Smeulders, and Edouard Oyallon. i-RevNet: Deep invertible networks. In *International Conference on Learning Representations*, 2018.

Adam Karvonen, Benjamin Wright, Can Rager, Rico Angell, Jannik Brinkmann, Logan Smith, Claudio Mayrink Verdun, David Bau, and Samuel Marks. Measuring progress in dictionary learning for language model interpretability with board game models. In *Advances in Neural Information Processing Systems*, 2024. arXiv:2408.00113.

Adam Karvonen, Can Rager, Johnny Lin, et al. SAEBench: a comprehensive benchmark for sparse autoencoders in language model interpretability. In *Proceedings of the 42nd International Conference on Machine Learning*, pp. 29223–29264, 2025. arXiv:2503.09532.

Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, 2018.

Piotr Kubaty, Patryk Marszałek, Łukasz Struski, Adam Wróbel, Jacek Tabor, and Marek Śmieja. Conceptualizing embeddings: Sparse disentanglement for vision-language models, 2026. arXiv:2605.22679.

Mario Lezcano-Casado and David Martínez-Rubio. Cheap orthogonal constraints in neural networks: a simple parametrization of the orthogonal and unitary group. In *ICML*, pp. 3794–3803, 2019. arXiv:1901.08428.

Aleksandar Makelov, George Lange, and Neel Nanda. Towards principled evaluations of sparse autoencoders for interpretability and control, 2024. arXiv:2405.08366.

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: discovering and editing interpretable causal graphs in language models. In *The Thirteenth International Conference on Learning Representations*, 2025. arXiv:2403.19647.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. Jumping ahead: Improving reconstruction fidelity with JumpReLU sparse autoencoders, 2024. arXiv:2407.14435.

Saiprasad Ravishankar and Yoram Bresler. Learning overcomplete sparsifying transforms for signal processing. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3088–3092, 2013a.

Saiprasad Ravishankar and Yoram Bresler. Learning sparsifying transforms. *IEEE Transactions on Signal Processing*, 61(5):1072–1086, 2013b.

Saiprasad Ravishankar and Yoram Bresler. ℓ_0 sparsifying transform learning with efficient optimal updates and convergence guarantees. *IEEE Transactions on Signal Processing*, 63(9):2389–2404, 2015.

Saiprasad Ravishankar and Brendt Wohlberg. Learning multi-layer transform models. In *56th Annual Allerton Conference on Communication, Control, and Computing*, 2018.

Saiprasad Ravishankar, Anna Ma, and Deanna Needell. Analysis of fast structured dictionary learning. *Information and Inference: A Journal of the IMA*, 9(4):785–811, 2020.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive autoencoders: Explicit invariance during feature extraction. In *ICML*, 2011.

Adly Templeton, Tom Conerly, Jonathan Marcus, et al. Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. Transformer Circuits Thread, 2024. URL `https://transformer-circuits.pub/2024/scaling-monosemanticity`. arXiv:2605.29358.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008.

Bihan Wen, Saiprasad Ravishankar, and Yoram Bresler. Structured overcomplete sparsifying transform learning with convergence guarantees and applications. *International Journal of Computer Vision*, 114(2–3):137–167, 2015.

A Relation to sparsifying transform learning

Section 2 gives the high-level comparison. Here we make the differences explicit. Transform learning uses one analysis operator W and its code, with no separately learned synthesis matrix. This matches FlowSAE with a linear F and no depth, and exactness at full support is standard there. Prior work also studies overcomplete transforms (Wen et al., 2015; Ravishankar & Bresler, 2013a) and deep transform models (Ravishankar & Wohlberg, 2018), but does not combine them with a lifted input and invertible nonlinearities. The deep model sparsifies intermediate residuals, so its layers are not composed bijections. Table 3 summarises the comparison.

	exact at full support	overcomplete	invertible σ	depth
Ravishankar & Bresler (2013b)	yes	no	no	no
Ravishankar & Bresler (2015)	yes	no	no	no
Ravishankar & Bresler (2013a)	yes	tall, not lifted	no	no
Wen et al. (2015)	yes	by stacking	no	no
Ravishankar et al. (2020)	implicit	no	no	no
Ravishankar & Wohlberg (2018)	structural	no	no	yes
Kubaty et al. (2026)	yes	no	no	no
FlowSAE	yes	**by lift**	**yes**	yes

Table 3: Positioning against the prior work discussed in Sec. 2. No prior construction we are aware of combines overcompleteness by lifting, an invertible pointwise nonlinearity, and depth.

All primary sources in this comparison select by largest *magnitude*. FlowSAE selects by signed value. We did not find this variant discussed in those works.

B Coupling flows versus the FlowSAE block

This appendix expands the comparison with coupling flows in Sec. 2. Figure 3 illustrates the difference. RealNVP uses a fixed split of coordinates. x_A is copied and conditions an affine transformation of x_B. In FlowSAE, an orthogonal transformation Q_ℓ first mixes the coordinates, and an invertible coordinate-wise nonlinearity follows. No coordinate has a fixed conditioner or conditioned role.

C Ablation: orthogonal chart parametrisation

This section motivates the choice of the exponential map $Q_\ell = \exp(B_\ell)$ over the Cayley transform $Q_\ell = (I - B_\ell)(I + B_\ell)^{-1}$.

For a 2×2 rotation block with generator scale t, the Cayley transform gives

$$\theta = 2 \arctan t, \qquad \frac{d\theta}{dt} = \frac{2}{1 + t^2}, \tag{16}$$

while the exponential map gives $\theta = t$. Rotations approaching π therefore require $t \to \infty$ under Cayley, and the angular sensitivity vanishes. At 179.5° the Cayley transform needs $t = 229$, where the exponential map needs $t = 3.13$. This scalar identity only illustrates the mechanism. $\|A_\ell\|$ is a norm over the whole thin parameter matrix, not a per-block generator scale, so we measure the consequence directly.

Under the reconstruction objective of Eq. (13), with $n = 8192$ and $d = 384$, training drives the Cayley parameters towards this regime. $\|A_\ell\|$ grows from 19.6 at initialisation to 951–1284 over

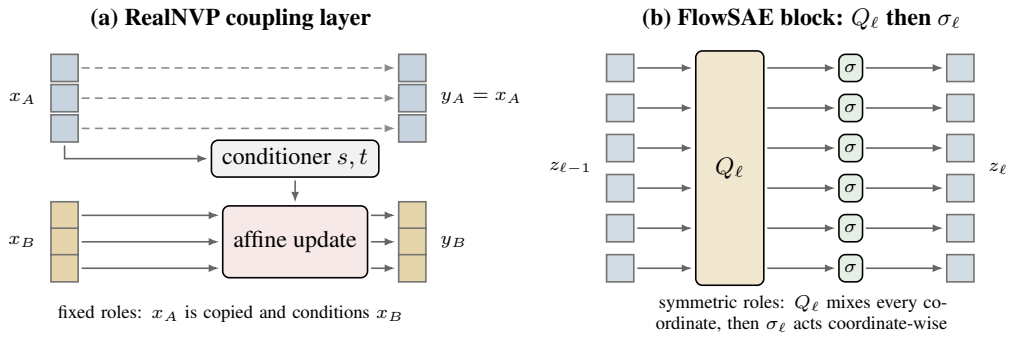

Figure 3: (a) RealNVP assigns fixed roles to two coordinate groups. x_A is copied unchanged and conditions the affine update of x_B. (b) A FlowSAE block first mixes coordinates with an orthogonal map Q_ℓ and then applies an invertible scalar nonlinearity coordinate-wise, so no latent coordinate has a fixed conditioner or conditioned role. The mixing is not uniform. Since $\mathrm{rank}(B_\ell) \le 2d$, an $(n-2d)$-dimensional subspace stays pointwise fixed, and the zero-padded lift distinguishes the first d coordinates (Sec. D).

400 epochs, despite weight decay of 10^{-2}, and the exact $k = n$ reconstruction property is lost around epoch 55.

The exponential map has no such boundary. It does cost additional computation. The low-rank construction requires four products with inner width $2d$, against one width-d product for a linear layer, which is an $8\times$ flop overhead shared by both charts. In our implementation the Cayley solve takes 10.9–14.2$\times$ the wall-clock time of a linear layer. The exponential map adds a further 1.6–2$\times$ per layer for the $2d \times 2d$ matrix exponential. The exponential parametrisation thus removes the failure mode at a measurable computational cost.

D Additional detail on the orthogonal parametrisation

Both parametrisations were verified numerically against dense reference implementations. Orthogonality and determinant errors are at machine precision, and the low-rank matrix-exponential correction has relative error of about 10^{-14}.

Computational cost. The low-rank form of Eq. (10) uses only a $2d \times 2d$ matrix. Building it costs $\mathcal{O}(nd^2 + d^3)$ per layer when the parameters change. Applying Q_ℓ or $Q_\ell^\top$ to N samples costs $\mathcal{O}(Nnd)$, so a forward pass through L layers costs $\mathcal{O}(LNnd)$. A classical SAE costs $\mathcal{O}(Nnd)$, so the overhead is a constant factor. No $n \times n$ matrix is formed.

Gauge freedom in A_ℓ. The nd entries of A_ℓ contain a $d(d+1)/2$-dimensional gauge. Writing A_ℓ as $[A_{\mathrm{top}}; A_{\mathrm{bot}}]$, with $A_{\mathrm{top}} \in \mathbb{R}^{d\times d}$, the induced skew matrix is

$$B_\ell = \begin{pmatrix} A_{\mathrm{top}} - A_{\mathrm{top}}^\top & -A_{\mathrm{bot}}^\top \\ A_{\mathrm{bot}} & 0 \end{pmatrix}. \tag{17}$$

Adding any symmetric matrix to A_{top} leaves B_ℓ unchanged, and this is the whole gauge. Matching the bottom-left block of two candidate matrices forces their A_{bot} blocks to agree exactly. Matching the top-left block then forces the two top blocks to differ by a symmetric matrix, so no other redundancy is possible. The map therefore has exactly $nd - d(d+1)/2$ effective parameters. This matches $\dim \mathrm{St}(n, d)$ as a dimension count. It is not a claim that $A_\ell \mapsto Q_\ell$ is a diffeomorphism onto the Stiefel manifold. Storing only the strictly lower-triangular part of A_{top} removes the gauge and gives a minimal parametrisation. We use the simpler full-A_ℓ form throughout.

The construction does not mix every coordinate uniformly. Since $\mathrm{rank}(B_\ell) \le 2d$, an $(n-2d)$-dimensional subspace is fixed pointwise, which is 7424 dimensions for $n = 8192$ and $d = 384$. The property that matters for top-k selection is therefore not uniform mixing. It is the absence of a fixed conditioner/conditioned split.

E CLOSED-FORM ACTIVATION INVERSE AND TRUNCATION GRADIENT

Each σ_ℓ (Eq. (11)) is a strictly increasing, piecewise-linear bijection of each coordinate, with kink at $y_i^* = s_{\ell,i}(\kappa_{\ell,i} - \beta_{\ell,i})$. Its closed-form inverse is

$$\sigma_\ell^{-1}(y)_i = \begin{cases} \dfrac{y_i}{s_{\ell,i}} + \beta_{\ell,i}, & y_i \le y_i^*, \\ \dfrac{y_i/s_{\ell,i} + \beta_{\ell,i} + c_{\ell,i}\kappa_{\ell,i}}{1 + c_{\ell,i}}, & y_i > y_i^*. \end{cases} \tag{18}$$

Both slopes, $s_{\ell,i}$ and $s_{\ell,i}(1 + c_{\ell,i})$, are strictly positive by the softplus and $\tanh$-margin reparametrisation of Sec. 3.3, so this is a genuine inverse.

The truncation $\mathcal{T}_k$ is implemented directly as `torch.topk` followed by scattering the selected values into a zero tensor, with no straight-through estimator or other gradient surrogate. Away from ties, standard autograd therefore gives the exact gradient of $\mathcal{T}_k$. Selected coordinates receive their ordinary upstream gradient. Discarded coordinates receive zero gradient, since they are constants with respect to the input. The multi-k objective (Eq. (13)) and every classical top-k baseline in this paper are trained this way.

F THE LIFT IS LOAD-BEARING

Exactness at $k = n$ has prior work (Sec. 2), so the lift has to carry the contribution. Nothing in the construction forces the flow to use the $n - d$ appended coordinates. A flow that ignored them would reduce to a square transform on $\mathbb{R}^d$ and keep all of $\|F(\iota(v))\|_2^2$ in the first d coordinates. The trained flow does not. On held-out features, 0.9628 of $\|F(\iota(v))\|_2^2$ lies outside the first $d = 384$ coordinates, and all 32 coordinates kept by $\mathcal{T}_{32}$ lie outside them. This is about what an isotropic random rotation of the same space would give ($0.9531 = 1 - d/n$ of the squared norm, and all 32 outside with probability $0.95^{32} \approx 0.21$). The lift is therefore used about as fully as a random rotation would use it. The flow spreads the code over the lifted space and does not collapse back onto the input subspace.

G COORDINATE DIRECTIONS AND ADDITIVITY

This appendix reports what the local direction

$$\Delta_i(\tilde{z}) = \frac{\partial}{\partial \tilde{z}_i}\Big(P\big(F^{-1}(\tilde{z})\big) + b\Big) \in \mathbb{R}^d \tag{19}$$

of Sec. 5 does and does not support. Two questions matter for a reader who wants to treat a FlowSAE coordinate like a classical decoder column.

Is the direction stable? Yes. Evaluating Δ_i at different points of the latent space, including codes with different active supports, gives mean cosine similarity of 0.96–0.99 between the resulting directions. A coordinate therefore names a well-defined direction in activation space.

Is the decomposition additive? No. We compare the reconstruction from a k-sparse code with the linear surrogate $b + \sum_{i \in \text{supp}} \tilde{z}_i \Delta_i(0)$ that a dictionary reading would predict. At $k = 32$ the relative error is about 141%. The gap comes from the architecture, not from training. Δ_i is constant only within one linear region of the piecewise-linear σ_ℓ, and a 32-sparse code generally crosses several. The attribution gap of Tab. 2 measures the same effect on single-feature ablations.

Standard `encode`/`decode` evaluations therefore transfer unchanged, but analyses that assume additive decoder columns do not.

H STATISTICAL TEST FOR THE COMPOSITION COMPARISON

The comparisons in Sec. 4.5 use the standard error

$$\mathrm{se} = \sqrt{\frac{s_A^2}{n_A} + \frac{s_B^2}{n_B}}, \tag{20}$$

which does not require the two groups to have equal variances. The test statistic is

$$t = \frac{\bar{x}_A - \bar{x}_B}{\mathrm{se}}. \tag{21}$$

We compare $n_A = 6$ FlowSAE seeds with $n_B = 3$ seeds for each baseline. The Welch–Satterthwaite degrees of freedom lie between $\min(n_A, n_B) - 1 = 2$ and $n_A + n_B - 2 = 7$, so we report against the most conservative end, $t_{0.975,2} = 4.303$ at $\alpha = 0.05$. All three statistics ($|t| = 12.6$–50.4) exceed it by a wide margin. Every comparison therefore rejects equal means under any admissible choice of degrees of freedom.

I FULL RECONSTRUCTION VALUES

Table 4 gives the values plotted in Fig. 2.

k	Classical single-k	Classical multi-k	Multi-TopK	FlowSAE multi-k
32	0.211	0.241	0.213	0.210
256	0.225*	0.103	0.145*	**0.082**
1024	0.225*	0.090*	0.145*	**0.020**
4096	0.225*	0.090*	0.145*	**0.010**
8192	0.225*	0.090*	0.145*	$\approx$ **0**

Table 4: FVU on held-out ImageNet ViT-S CLS features, 400-epoch runs, one seed. *Plateau: k exceeds the number of positive ReLU pre-activations, so further selected coordinates are zero. Classical columns use the final checkpoint in fp32 and Multi-TopK the final checkpoint in fp64. FlowSAE is the median over checkpoints from epochs 225–400 in fp64 (the $k = 4096$ entry ranges 0.007–0.025 across those checkpoints).

J ABLATIONS

This appendix examines whether the observed behaviour depends on the training objective, depth, parameter count, or the hard top-k rule.

Training across sparsity budgets. Training FlowSAE only at $k = 32$ gives FVU 0.214 at that budget, close to the multi-k value of 0.216. Away from the training point it deteriorates, reaching 0.488 at $k = 512$, before returning to $\sim 10^{-8}$ at $k = n$, where the architectural guarantee holds regardless of the objective. Multi-k training makes this U-shape far shallower. It costs only 0.002 FVU at the target budget and gives useful reconstruction across all sparsity levels.

Depth. With identical multi-k training, $L = 2$, 3, 4, and 6 flow layers give FVU at $k = 32$ of 0.246, 0.227, 0.216, and 0.215. Most of the improvement is obtained by $L = 4$, and additional depth gives diminishing returns.

Matched parameter count. Widening the classical multi-k SAE to $n = 16384$ gives 12,599,680 parameters, close to FlowSAE's 12,714,368. At the same 400-epoch budget it improves $k = 32$ FVU only from 0.241 to 0.235, while FlowSAE reaches 0.210. Additional classical width therefore does not account for the gap. In the other direction, FlowSAE with $L = 2$ has 6,357,376 parameters, about the classical SAE's 6,300,032, and reaches 0.246 after 100 epochs. At the classical SAE's size, FlowSAE therefore does not beat it.

$L0$	19	32	38	76	155	310	630
FlowSAE, multi-k	0.912	0.932	0.937	**0.954**	**0.968**	**0.979**	**0.987**
FlowSAE, single-k ($k = 32$)	0.914	0.934	0.939	0.949	0.952	0.952	0.921

Table 5: `core ce_loss_score` (higher is better) across the sparsity sweep on Pythia-160M-deduped layer 8, for the multi-k generalist and a specialist trained only at $k = 32$. The generalist is monotone and the specialist is not, as with FVU on ImageNet.

Quantity	multi-k generalist	matched single-k specialist
`ce_loss_score` at $k = 32$	0.932	0.934
`ce_loss_score` at $k = 458$	0.984	0.997
support agreement at $k = 32$	0.957	—

Table 6: Matched-budget comparisons on the language-model substrate. At $k = 32$ the multi-k generalist costs 0.2 percentage points against a specialist trained at the same budget, matching the ImageNet ablation (Sec. J). At $k = 458$ the cost widens to $5.3\times$ in error (0.016 against 0.003). Support agreement (Sec. 4.5) is a single-seed measurement of the multi-k generalist only. No classical or linear baseline has been trained on this substrate, hence the dash.

Differentiable relaxation of top-k. A differentiable per-coordinate scoring head with a budgeted soft mask can be optimised successfully during training. Binarising the mask at deployment, however, increases FVU to about 1.06, worse than predicting the sample mean. This relaxation is therefore not a viable alternative to hard top-k selection here.

K Language-model frontier comparison

This appendix gives the setup and numbers behind the language-model check of Sec. 5. We use Pythia-160M-deduped residual-stream activations at layer 8 ($d = 768$, $n = 4096$) and compare against SAEBench (Karvonen et al., 2025).

Two single-k specialists, trained at $k = 458$ and $k = 630$ to match published baselines, preserve the $k = n$ identity to floating-point precision across three seeds each. FVU at $k = n$ lies between 10^{-23} and 10^{-21} for all six checkpoints. This is the same exact reconstruction observed on ImageNet, without the numerical failure of the Cayley chart.

For these two specialists we measure concept-detection accuracy (`sparse_probing`) at each specialist's own budget. We compare against SAEBench's published frontier of top-k, JumpReLU, BatchTopK, Matryoshka-BatchTopK, PAnneal, and Standard SAEs. Both specialists land in the middle of that frontier, with mean 0.754 at $k = 458$ (GatedSAE 0.756, Standard 0.742) and mean 0.736 at $k = 630$ (BatchTopK 0.734, Matryoshka/TopK 0.752/0.748).

Seeding matters on this substrate as much as on ImageNet. The loss-recovered score (`ce_loss_score`) of the $k = 630$ specialist at $L0 = 19$ varies by 25 points across three identically configured seeds (-0.007, 0.168, 0.241). Concept-detection accuracy at $k = 458$ spans 0.731–0.774, enough to reverse single-seed comparisons against the published frontier. The means above are therefore reported over seeds.

Table 5 compares multi-k training with a matched single-k specialist. The multi-k curve is monotone across all seven sparsity levels. The $k = 32$ specialist peaks at $L0 = 155$ and then declines, ending below its own $L0 = 32$ value at $L0 = 630$. Here $L0$ is the number of active latents at inference, this substrate's analogue of k.

Table 6 gives the cost of multi-k training at two budgets, and a first composition measurement.

The attribution-floor experiment of Sec. 4.3 has not been run on this substrate. It needs a classifier-head logit, and our evaluation code has no next-token-loss analogue yet.