
Beyond Steering Vector: Flow-based Activation Steering for Inference-Time Intervention

Anonymous Author(s)

Affiliation

Address

email

Abstract

1 Activation steering has emerged as a promising alternative for controlling language-
2 model behavior at inference time by modifying intermediate representations while
3 keeping model parameters frozen. However, large-scale evaluations such as
4 AxBench show that existing steering methods are often outperformed by simple
5 in-context prompting and generalize poorly to unseen concepts. We hypothesize
6 that these limitations arise from unvalidated simplifying assumptions shared across
7 prior methods, which typically restrict steering interventions to fixed, single-step,
8 position-invariant transforms. We propose **FLAS** (**F**low-based **A**ctivation **S**teering),
9 which learns a general, concept-conditioned velocity field $v_\theta(h, t, c)$ that transports
10 unsteered activations to steered ones without relying on these assumptions. On
11 AxBench, FLAS is the first learned method to consistently outperform prompting,
12 reaching held-out harmonic means of 1.015 on Gemma-2-2B-IT and 1.113 on
13 Gemma-2-9B-IT without per-concept tuning. Analysis of the learned flow shows
14 curved, multi-step, token-varying trajectories, which suggests that previous hy-
15 potheses on activation space geometry might be incomplete. Our code is available
16 at <https://anonymous.4open.science/r/FLAS>.

17 1 Introduction

18 Large language models have demonstrated strong capabilities across diverse tasks [1–3], yet reliably
19 controlling their behavior to align with human preferences remains a persistent challenge [4]. Ex-
20 isting control mechanisms such as prompting and fine-tuning face limitations in robustness, cost,
21 and side effects [4–7]. Activation steering has emerged as a complementary alternative that of-
22 fers lightweight, interpretable control across a growing range of behaviors [8–12] by modifying
23 intermediate representations at inference time while leaving model parameters frozen [13–15].

24 Despite these successes, AxBench [16], a benchmark that evaluates thousands of natural-language
25 steering concepts, reveals a consistent limitation of existing steering methods. In particular, simple
26 in-context prompting outperforms the tested steering methods, and increasing the scalar steering
27 strength improves concept incorporation while monotonically degrading instruction following and
28 fluency. The requirement for concept-specific strength tuning on a development set [16] limits the
29 real-world application of previous steering methods.

30 We hypothesize that this performance gap stems from simplifying assumptions that most activation-
31 steering approaches adopt at design time without rigorous validation. While most recent methods have
32 relaxed the Linear Representation Hypothesis [17–19] by introducing adaptive transforms [20–25],
33 other assumptions persist widely (Table 1), typically restricting interventions to single-step, position-
34 invariant transforms trained per concept on contrastive data. These assumptions define, for each
35 method, a prescribed operator family that constrains both what information the intervention may use
36 and how it may act on the activation. Individual methods relax one or more of these constraints while

37 retaining the rest. Recent flow- and ODE-based formulations [26–28] loosen the single-step constraint
 38 by allowing multi-step, state-dependent trajectories, yet they retain the dependence on contrastive
 39 data and per-concept training. These restrictions shape how interventions behave in practice and can
 40 limit the attainable trade-off between concept incorporation and instruction following.

41 To address these restrictions, we propose to learn a more expressive steering operator directly
 42 from data by introducing **FLAS (Flow-based Activation Steering)**. FLAS replaces a fixed one-step
 43 intervention with a learned, time-conditioned velocity field $v_\theta(h, t, c)$ that transports an unsteered
 44 activation h to a steered activation $h' = \varphi_T(h)$ through N steps of Euler integration, conditioned
 45 on a natural-language concept description c . Because the velocity depends on the current activation
 46 state, the resulting intervention adapts as the activation evolves and, when integrated over multiple
 47 steps, can follow curved trajectories through activation space. Evaluating the velocity independently
 48 at each token position further allows the displacement to vary across a sequence. The method trains
 49 on positive examples under a standard language-modeling objective, without requiring contrastive
 50 pairs, and employs the flow time T as a continuous integration horizon that decouples intervention
 51 strength from direction.

52 Our contributions are as follows.

- 53 1. We propose **FLAS (Flow-based Activation Steering)**, a concept-conditioned velocity field
 54 integrated by N -step Euler that enables adaptive, multi-step, position-sensitive steering
 55 trained on positive examples alone. The flow formulation recovers many single-step methods
 56 as special cases for $N = 1$ and fixed T .
- 57 2. FLAS is the first learned steering method to consistently outperform prompting on
 58 AxBench [16], achieving held-out HMean 1.015/1.113 (Gemma-2-2B/9B-IT) vs. prompting
 59 0.762/1.091 and HyperSteer [25] 0.608/0.934, with $< 1/26$ the parameters. Performance
 60 remains stable across $T \in [0.5, 4.0]$ without per-concept tuning, and generalizes to held-out
 61 concepts with monotonic scaling at 16k without clear saturation.
- 62 3. The learned velocity field serves as an analysis probe of activation space, revealing curved,
 63 position-dependent, multi-step structure. Our method provides empirical evidence that
 64 effective steering requires nonlinear and position-sensitive interventions, suggesting that
 65 previous hypotheses on activation space geometry might be incomplete.

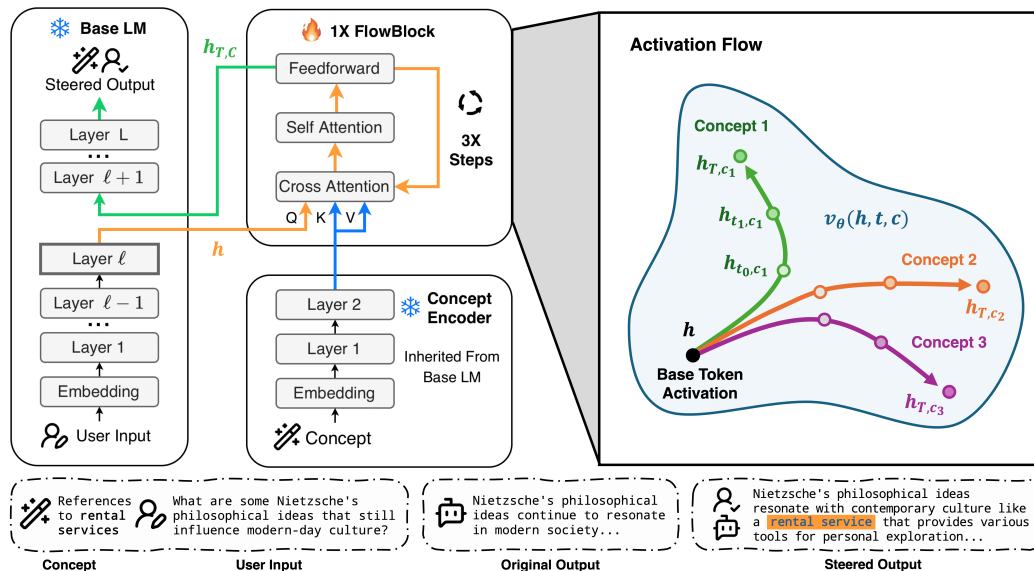


Figure 1: **FLAS model architecture overview.** The velocity field $v_\theta(h, t, c)$ transports activations at layer ℓ of a frozen base LM. A frozen concept encoder ϕ produces concept representations consumed by a single FlowBlock via cross-attention. The flow is integrated by N -step Euler, shared between training and inference, yielding a steered activation $h' = \varphi_T(h)$. The entire base language model (base LM) is frozen; only the FlowBlock parameters are trained.

66 **2 Related Work**

67 Table 1 situates FLAS in the landscape of activation-steering methods along five structural axes.

Method	Intervention	Training data	Adaptive	Multi-step	Per-token	Zero-shot
DiffMean / CAA [18]	static vector	pos+neg	✗	✗	✗	✗
SAE features [19]	feature clamp	N/A*	✗	✗	✗	✗
ReFT-r1 [20]	learned affine	pos only	✓	✗	✗	✗
Spherical Steer. [22]	geodesic rotation	pos+neg	✓	✗	✗	✗
Curveball Steer. [23]	kernel curve	pos+neg	✓	✗	✗	✗
AcT [21]	OT affine	pos+neg	✓	✗	✓	✗
TruthFlow [26]	flow correction	pos+neg	✓	✓	✗	✗
K-Steering [24]	classifier gradient	pos+neg	✓	✓	✓	✗
FlowSteer [27]	flow transport	pos+neg	✓	✓	✓	✗
ODESteer [28]	barrier ODE	pos+neg	✓	✓	✓	✗
HyperSteer [25]	conditional vector	pos only	✓	✗	✗	✓
FLAS (ours)	flow transport	pos only	✓	✓	✓	✓

Table 1: **Structural comparison of activation-steering methods.** **Adaptive:** depends on current h . **Multi-step:** iterative integration. **Per-token:** uses inter-position context. **Zero-shot:** no per-concept retraining. **Training data:** “pos only” = concept-aligned responses only, “pos+neg” = additionally requires paired negatives. *Relies on a pretrained sparse autoencoder for feature extraction.

68 **Linear activation steering.** Activation Addition [14] and CAA [18] each extract or optimize
 69 a fixed steering direction and add a scaled copy at a chosen layer. Recent work computes the
 70 displacement through learned mechanisms such as low-rank interventions [20] and cross-attention
 71 hypernetworks [25], but still produce a single displacement at inference time, and none have been
 72 reported to consistently surpass prompting on AxBench.

73 **Concurrent nonlinear and flow-based steering.** Activation Transport [21], Curveball Steer-
 74 ing [23], Spherical Steering [22], and Householder Pseudo-Rotation [29] introduce nonlinear single-
 75 step interventions ranging from affine maps to norm-preserving rotations, requiring paired source-
 76 target data. K-Steering [24], TruthFlow [26], FlowSteer [27], and ODESteer [28] adopt multi-step
 77 continuous-dynamics formulations, but each targets a single attribute and requires task-specific
 78 paired data. FLAS combines a concept-conditioned velocity field with zero-shot generalization via
 79 end-to-end LM-loss training on positive data only.

80 **Flow matching and representation geometry.** Our velocity-field parameterization draws on flow
 81 matching [30–32] and its extensions to manifolds [33] and latent spaces [34]. Where flow matching
 82 transports noise to data, FLAS transports unsteered activations to steered ones under a downstream
 83 language-modeling objective rather than a flow-matching regression target. The manifold view of
 84 LLM representations [35–39] treats hidden states as lying on low-dimensional submanifolds, and
 85 our trajectory analyses in Sections 6.1 and 6.2 give a concrete picture of how a learned intervention
 86 traces on such a submanifold.

87 **3 Method**

88 **3.1 Flow-based Steering**

89 Fix a pretrained language model with L layers and hidden width d . At a chosen layer ℓ , the forward
 90 pass produces activations $h \in \mathbb{R}^{n \times d}$, where n is the sequence length. Given a natural-language
 91 concept description c (e.g., a short phrase specifying the target behavior), we want to replace h with a
 92 steered version h' so that subsequent layers generate text exhibiting the concept while preserving
 93 instruction following and fluency.

94 We realize the map from h to h' as a learned, concept-conditioned flow. Let $\{\varphi_t\}_{t \in [0, T]}$ be a family
 95 of maps $\varphi_t : \mathbb{R}^{n \times d} \rightarrow \mathbb{R}^{n \times d}$ generated by a velocity field v_θ , defined by the ODE

$$\frac{d}{dt} \varphi_t(h) = v_\theta(\varphi_t(h), t, c), \tag{1}$$

96 with initial condition $\varphi_0(h) = h$. The steered activation is obtained by integrating the velocity field
 97 from 0 to T :

$$h' = \varphi_T(h) = h + \int_0^T v_\theta(\varphi_t(h), t, c) dt. \quad (2)$$

98 In practice, we approximate this integral using an N -step forward Euler method:

$$h_{k+1} = h_k + \frac{T}{N} v_\theta\left(h_k, \frac{kT}{N}, c\right), \quad (3)$$

99 for $k = 0, \dots, N - 1$, with $h_0 = h$. The resulting h_N serves as a numerical approximation to
 100 $h' = \varphi_T(h)$ and is passed to layer $\ell + 1$ in place of h .

101 Three properties of v_θ together distinguish this formulation from previous steering methods. First,
 102 the map φ_t depends on the initial state h , so the flow adapts to different activations. Second, the
 103 time-dependent velocity field can prescribe different directions at each step along the integration path,
 104 producing curved trajectories. Third, v_θ is computed per token position, thus the steering trajectory
 105 varies per token.

106 Taken together, these properties make v_θ sufficiently expressive that the integral in Eq. 2 can in
 107 principle realize a multi-step transport from h to h' . As a consequence, our formulation naturally
 108 subsumes prior steering approaches as restricted instances of the velocity field. The standard additive
 109 formulation $h' = h + \alpha\delta(c)$ is recovered as the special case $v_\theta(h, t, c) = \delta(c)$ with $T = \alpha$.

110 3.2 FlowBlock Architecture and Forward Process

111 We instantiate v_θ with a transformer-style block, which we call a FlowBlock (Figure 1). To encode
 112 the concept description c into a sequence of vectors that the FlowBlock can attend to, we apply a
 113 concept encoder ϕ . By default ϕ reuses the token embedding and first few transformer layers of the
 114 base model, so that $\phi(c)$ inherits the early-layer features of the base model.

115 At step k , the FlowBlock takes the current activation h_k , the encoded concept $\phi(c)$, and the current
 116 time $t_k = kT/N$ as input. We first inject the time signal through a sinusoidal embedding,

$$\tilde{h}_k = h_k + \text{TimeEmbed}(t_k). \quad (4)$$

117 Since c is a sequence of arbitrary length, the FlowBlock attends to it through cross-attention,

$$u_k = \text{CrossAttn}(Q = \tilde{h}_k, K = \phi(c), V = \phi(c)), \quad (5)$$

118 whose keys and values are cached once and reused across N integration steps and decoding positions.
 119 A causal self-attention layer and a feedforward network then produce the per-step displacement,

$$\Delta h_k = \text{Feedforward}(\text{SelfAttn}(u_k)). \quad (6)$$

120 Iterating this procedure N times yields h_N . In practice, each component is wrapped with a residual
 121 connection and a learnable per-channel gate, and the update at each step is scaled by the Euler step
 122 size T/N . Full implementation details are included in Appendix B.

123 3.3 Training

124 To control the steering strength at inference, we use the flow time T as a scalar parameter. Under the
 125 Euler method (Eq. 3) with fixed N , increasing T scales the per-step updates and pushes activations
 126 further along their concept-specific trajectories.

127 For T to provide continuous control, the velocity field v_θ must remain valid across varying horizons.
 128 Unlike prior flow-based methods with a fixed training-time strength [30], we enable training-free
 129 extrapolation at inference by exposing the model to a range of integration horizons during training.
 130 Like classifier-free guidance [40], our approach enables dynamic strength control at inference,
 131 achieved by simply scaling the integration time of the learned flow.

132 We implement this by randomizing the integration horizon during training. At each training step we
 133 sample $T \sim \text{Uniform}[T_{\min}, T_{\max}]$, run N Euler steps using Eq. 3, inject the resulting h_N at layer ℓ ,
 134 and supervise with language-modeling cross-entropy on the output tokens,

$$\mathcal{L}_{\text{LM}} = - \sum_{(x,y,c) \in \mathcal{D}} \sum_i \log p(y_i | y_{<i}, x; h_N), \quad (7)$$

135 where \mathcal{D} is the training dataset, with each triple consisting of an input prompt x , a concept c to steer
 136 toward, and the desired output y that reflects steering toward c .

137 Since velocities for different concepts should point in distinct directions, we add a diversity penalty
 138 on the mean-pooled final-step velocities within each minibatch,

$$\mathcal{L}_{\text{div}} = \frac{1}{|\{(i, j) : c_i \neq c_j\}|} \sum_{i, j: c_i \neq c_j} \cos(\bar{v}_i, \bar{v}_j), \quad \bar{v}_i = \frac{1}{P} \sum_{p=1}^P v_i^{(p)}, \quad (8)$$

139 where p indexes token positions, $v_i^{(p)} = v_{\theta}(h_{N-1}^{(p)}, t_{N-1}, c_i)$ is the final-step velocity at position p for
 140 sample i . The total loss is $\mathcal{L}_{\text{LM}} + \lambda \mathcal{L}_{\text{div}}$ with $\lambda = 0.1$. Ablations in Sec. 5 confirm that this diversity
 141 penalty is important for steering quality, and especially for extrapolation along T (detailed discussion
 142 in Appendix D).

143 4 Experiments

144 **Training data and base model.** We follow the protocol of AxBench [16] and train on Concept16k.
 145 Base models are Gemma-2-2B-IT and Gemma-2-9B-IT [3], with steering at layer 20. We use a
 146 single FlowBlock (97.6M trainable parameters on 2B, 255M on 9B), with $N = 3$ Euler steps and
 147 $T \sim \text{Uniform}[0.5, 2.0]$. The concept encoder is frozen and reuses the base model’s token embedding
 148 and first two layers. Training details are included in Appendix A.

149 **Evaluation protocol.** We follow the AxBench evaluation pipeline. GPT-4o-mini [41] scores
 150 each generation on Concept incorporation (C), Instruction following (I), and Fluency (F), with
 151 $C, I, F \in \{0, 1, 2\}$. The primary metric is the harmonic mean of the three scores: $\text{HMean} =$
 152 $3/(1/C + 1/I + 1/F) \in [0, 2]$. Held-in evaluates on concepts seen during training but with
 153 previously unseen prompts. Held-out is strictly zero-shot, evaluating on concepts never seen during
 154 training paired with unseen prompts. Evaluation details are included in Appendix E.

155 4.1 Main Results

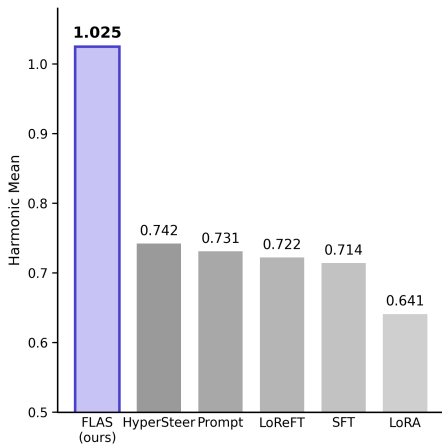


Figure 2: **Held-in results on Gemma-2-2B-IT, layer 20.** FLAS exceeds the in-context prompting baseline by +0.294 and HyperSteer by +0.283.

Method	Gemma-2-2B-IT		Gemma-2-9B-IT	
	Held-in	Held-out	Held-in	Held-out
Prompting	0.731	0.762	1.075	1.091
Fine-tuning				
LoReFT	0.722	—	0.777	—
SFT	0.714	—	—	—
LoRA	0.641	—	0.602	—
RePS	0.606	—	0.892	—
Activation Steering				
HyperSteer	0.742	0.608	1.091	0.934
ReFT-r1	0.509	—	0.630	—
AcT (All Layers)	0.187	—	0.161	—
DiffMean	0.178	—	0.322	—
SAE	0.151	—	0.191	—
AcT (Layer 20)	0.144	—	0.270	—
SAE-A	0.132	—	0.186	—
FLAS (ours)	1.025	1.015	1.185	1.113

Table 2: **Full steering results on AxBench.** Empty entries (—) indicate methods that do not support zero-shot steering. Baselines from AxBench [16] and HyperSteer [25]. FLAS evaluated at fixed $T = 2$. The intervention happens at Layer 20 of both models.

156 Table 2 and Figure 2 present the main results. All FLAS results are given using a single fixed flow
 157 time $T = 2$ with no per-concept tuning. On Gemma-2-2B-IT held-out evaluation, FLAS reaches a
 158 harmonic mean of 1.015, exceeding HyperSteer (0.608, +0.407) and in-context prompting (0.762,
 159 +0.253). On Gemma-2-9B-IT held-out evaluation, FLAS reaches the score of 1.113, above both
 160 in-context prompting (1.091, +0.022) and HyperSteer (0.934, +0.179). To illustrate the advantage

161 of FLAS over in-context prompting, we provide case studies in Appendix I where FLAS succeeds
 162 while in-context prompting fails. Overall, FLAS incorporates concepts into outputs more naturally
 163 and flexibly, especially for complex concepts.

164 To further assess cross-model generalization, we additionally apply FLAS to Qwen3-4B-Instruct [42]
 165 at layer 20 under the same training and evaluation pipeline, achieving a held-out harmonic mean of
 166 0.960 (detailed in Appendix C). This demonstrates that FLAS generalizes across model families.

167 4.2 Concept Scaling

168 We investigate how FLAS performance scales with the number
 169 of training concepts. We train models on subsets of 9, 500, 1.9 k,
 170 5.5 k, and the full 16 k concepts with identical hyperparameters,
 171 and evaluate on the same held-out concepts at $T=2$. As shown
 172 in Figure 3, the held-out harmonic mean increases monotonically
 173 with the number of training concepts, surpassing the in-context
 174 prompting baseline between 1.9 k and 5.5 k concepts. The curve
 175 shows no sign of saturation at 16 k, suggesting further gains from
 176 larger concept pools.

177 4.3 Flow Time Robustness

178 Activation steering typically involves a trade-off where increased
 179 concept incorporation degrades instruction following and fluency.
 180 Figure 4 contrasts FLAS with three baselines on Gemma-2-9B-IT:
 181 ReFT-r1, DiffMean, and AcT [21] (reproduced at layer 20, see
 182 Appendix F). All three baselines collapse at higher strengths, while
 183 FLAS steadily improves concept score and maintains high instruction and fluency across the entire range.

184 This robustness is not an artifact of training data abundance. Figure 5 decomposes the score across
 185 $T \in [0.5, 4.0]$ for five concept pool sizes on Gemma-2-2B-IT, and the qualitative shape of the
 186 curves is preserved across scales. Increasing the training pool mainly raises concept score, while
 187 instruction and fluency remain roughly unchanged. In the data-scarce regime (500 or 1.9 k concepts),
 188 increasing T at inference time substantially boosts concept incorporation, suggesting that flow time
 189 can compensate for limited training data.

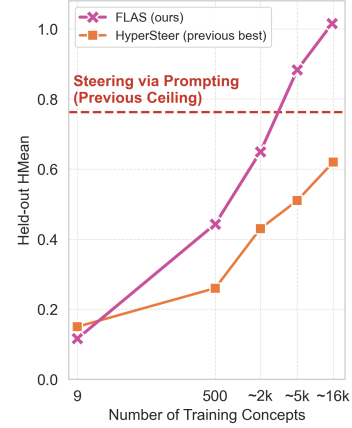


Figure 3: **Concept scaling.** Held-out harmonic mean versus the number of training concepts.

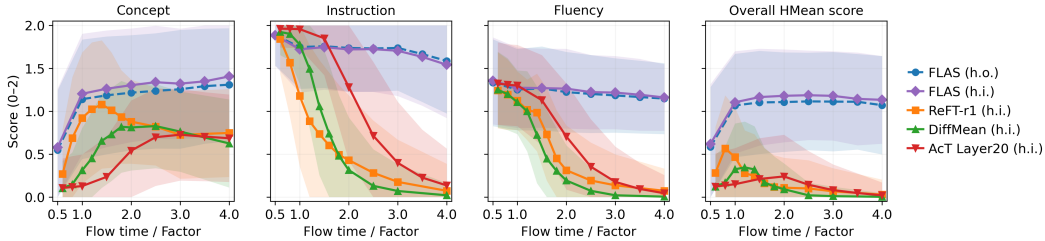


Figure 4: **Steering strength trade-off (Gemma-2-9B-IT).** Score decomposition across steering strengths for FLAS (held-out, h.o., in blue; held-in, h.i., in purple) and baselines (ReFT-r1, DiffMean, AcT). Shaded bands show ± 1 std, clipped to $[0, 2]$.

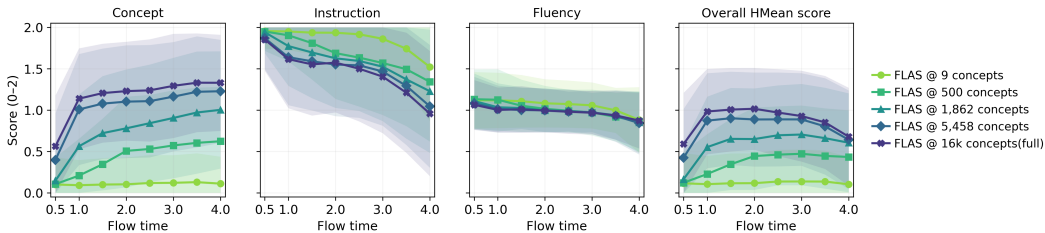


Figure 5: **Flow time across training-set sizes (Gemma-2-2B-IT held-out).** Score decomposition versus T for five concept scales. Shaded bands show ± 1 std, clipped to $[0, 2]$.

190 **5 Ablations**

191 We ablate the main design choices of FLAS on Concept16k held-out using Gemma-2-2B-IT at
 192 $T=2$. The base configuration uses $B=1$ FlowBlock, $N=3$ Euler steps, with three phases enabled
 193 (cross-attention, self-attention, MLP), diversity loss, a frozen concept encoder, and weights initialized
 194 from the corresponding Gemma-2 layer. All scores are averaged over held-out concepts (10 prompts
 195 each). We report 95% bootstrap confidence intervals (10 000 resamples over concept-level means)
 196 and paired t -statistics against the base configuration.¹

197 **Model Architecture.** Table 3 shows
 198 that the only ablation causing a large
 199 performance drop is disabling cross-
 200 attention ($t = -37.82$, $p < 0.001$),
 201 which removes the pathway for concept
 202 information to enter the activa-
 203 tion stream. Disabling self-attention
 204 causes a moderate drop to 0.969
 205 ($t = -2.19$, $p < 0.05$), indicating
 206 that inter-position coordination con-
 207 tributes. Removing the MLP causes
 208 a similar drop to 0.955 ($t = -3.05$,
 209 $p < 0.01$). The effect of adding
 210 FlowBlocks beyond $B=1$ is statisti-
 211 cally indistinguishable, confirming
 212 that the minimal single-block archi-
 213 tecture is already sufficient for Con-
 214 cept16k dataset.

215 **Training.** We ablate the diversity loss
 216 and the warm-start initialization strat-
 217 egy during training. Removing the
 218 diversity loss degrades performance
 219 to 0.932 ($t = -4.41$, $p < 0.001$).
 220 We observe a severe degradation in
 221 held-out performance without the di-
 222 versity loss, which we discuss in Ap-
 223 pendix D. Replacing Gemma-2 weight
 224 warm-start with Xavier initialization drops
 performance to 0.968 ($t = -2.49$, $p < 0.05$), confirming that initializing from the base model aids optimization.

225 **Intervention Layers.** To verify our model’s sensitivity to the choice of layer, we substitute layer 10
 226 or layer 15 for layer 20. Results in Table 3 show that steering at layer 10 performs comparably to
 227 the base and layer 15 shows a moderate drop to 0.946. Both substantially outperform the prompting
 228 baseline at 0.762. This proves that FLAS is not sensitive to the choice of intervention layer.

229 **Number of Integration Steps.** Table 3 ablates the number of Euler steps. At $N=1$ the flow
 230 reduces to a single adaptive displacement and performance drops significantly to 0.837 ($t = -9.56$,
 231 $p < 0.001$), but still exceeds prompting (0.762). Adding a second step recovers most of the remaining
 232 gap (0.970, $t = -2.59$, $p < 0.01$), and beyond $N=3$ further steps yield no significant improvement.
 233 Three Euler steps are sufficient for the velocity field to capture the required curvature. We analyze
 234 this structure in Section 6.2.

235 **6 The Geometry of Flow Steering**

236 The velocity field of FLAS can be inspected to understand the steering trajectories. We use the $N=10$
 237 model for the trajectory and per-step analyses, where the flow is exposed at high temporal resolution,
 238 and the $N=3$ model (our default configuration) for the per-token analysis. These three analyses show
 239 that effective activation steering requires curved, multi-step, token-varying interventions. Detailed
 240 settings of analysis experiments are included in Appendix G.

Configuration	HMean	95% CI	Paired t
Base ($B=1, N=3$)	1.015	[0.968, 1.060]	—
Architecture			
+1 FlowBlock ($B=2$)	1.009	[0.963, 1.051]	-0.34
+2 FlowBlocks ($B=3$)	0.996	[0.944, 1.044]	-1.06
Disable self-attention	0.969*	[0.922, 1.015]	-2.19
Disable MLP	0.955**	[0.905, 1.003]	-3.05
Disable cross-attention	0.109***	[0.078, 0.142]	-37.82
Training			
Xavier init	0.968*	[0.921, 1.012]	-2.49
Remove diversity loss	0.932***	[0.879, 0.982]	-4.41
Intervention layer			
Layer 10	1.044	[0.989, 1.096]	+1.22
Layer 15	0.946**	[0.884, 1.006]	-2.93
Integration steps (N)			
$N=1$	0.837***	[0.790, 0.884]	-9.56
$N=2$	0.970**	[0.928, 1.010]	-2.59
$N=4$	0.981	[0.936, 1.024]	-1.86
$N=5$	1.011	[0.962, 1.058]	-0.23
$N=10$	1.020	[0.974, 1.064]	+0.26

Table 3: **Ablations** (Concept16k held-out, $T=2$). HMean: harmonic mean of C/I/F. CI: 95% bootstrap over concept-level means. Paired t : versus base on the same held-out concepts.

¹Significance: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$.

241 **6.1 Steering Trajectories Are Curved**

242 Figure 6 visualizes the flow trajectories projected onto the leading principal components of the
 243 displacement vectors across various concepts, prompts, and integration steps.

244 The trajectories are not straight lines. Every concept’s path leaves the origin in a shared direction,
 245 executes a pronounced bend, and then enters a concept-specific region. Once the bend completes, T
 246 controls how far along the concept-specific direction the activation travels.

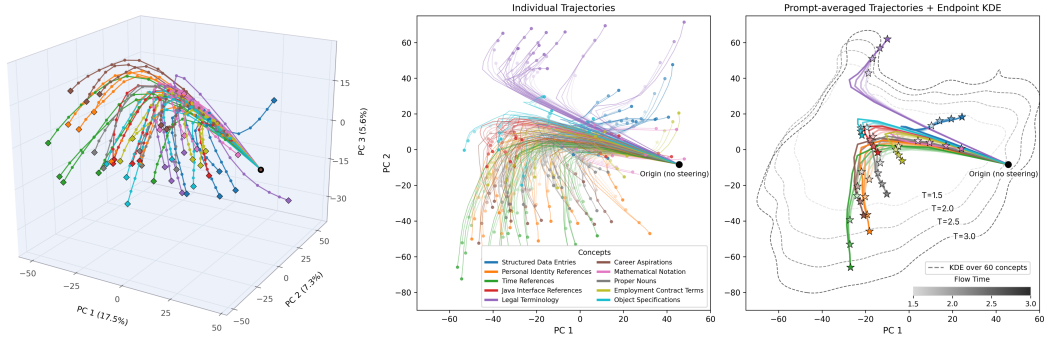


Figure 6: **Steering trajectories of the learned flow** ($N = 10$). Color encodes concept identity and lightness encodes flow time T , with lighter tints corresponding to lower T . **Left:** 3D PCA projection of trajectories at $T = 2$. **Middle:** per-concept, per-prompt 2D PCA trajectories at $T \in [1.5, 3.0]$. **Right:** prompt-averaged trajectories with dashed gray KDE contours showing the spread of 60 concepts at each T . Trajectories bend from a shared initial direction into concept-specific endpoint regions, and increasing T extends the displacement along each concept’s direction.

247 **6.2 The Learned Flow Requires Multiple Steps**

248 Figure 7 quantifies the per-step structure of the learned flow. At larger flow times ($T = 2.0$ and
 249 $T = 3.0$), the late steps point in mutually consistent directions (cosine similarity > 0.7), while the early
 250 steps are markedly misaligned with these later directions (cosine similarity < 0.25). This separation
 251 between early and late step directions provides quantitative evidence that the bending observed in
 252 Figure 6 is a statistically robust phenomenon rather than an artifact of individual trajectories.

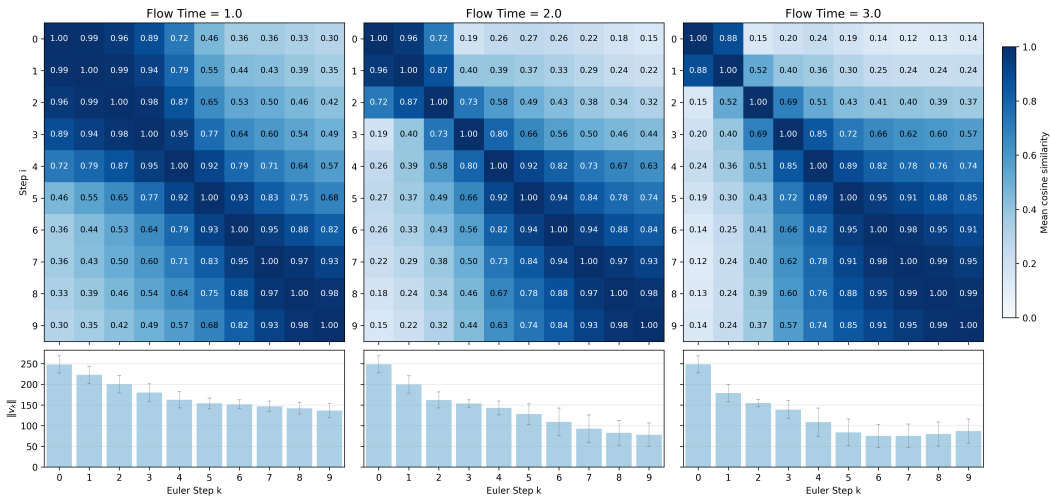


Figure 7: **Step-to-step velocity cosine and magnitude** ($N = 10$, $T \in \{1.0, 2.0, 3.0\}$). **Top:** 10×10 cosine matrix between Euler velocities. **Bottom:** mean $\|v\|$ per step.

253 6.3 Per-Token Steering Is Non-Uniform

254 Most previous activation-steering methods apply the same displacement to every token position.
255 FLAS evaluates the velocity field per position, and each token’s total displacement is the sum of N
256 Euler increments. Figure 8 shows the average pairwise cosine between per-token displacements is
257 only 0.294 ± 0.133 , far below the 1.0 that a position-invariant method produces. We observe that
258 nearby tokens exhibit higher steering similarity, and that similarities within prompt tokens and within
259 generated tokens are higher than across the two groups, revealing position-dependent structure.

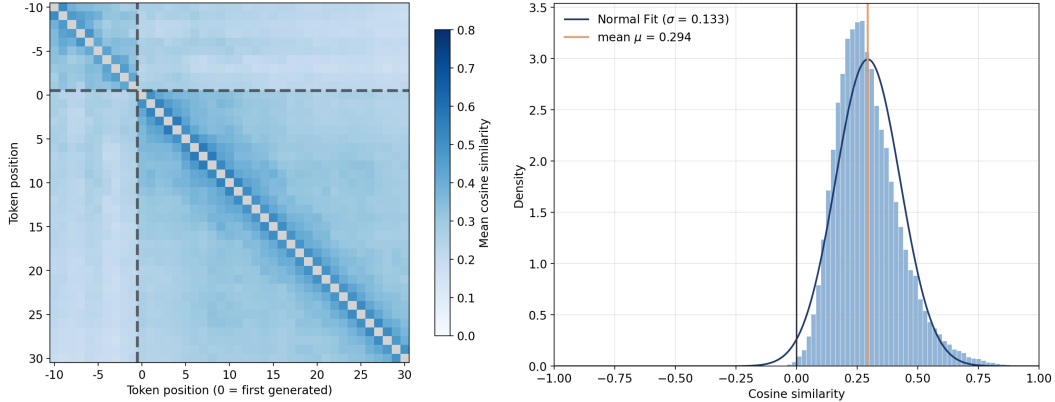


Figure 8: **Per-token displacement cosines** ($N = 3$, $T = 2$). **Left:** mean pairwise cosine of total displacements $h_N - h_0$ across token positions. **Right:** distribution of off-diagonal cosines ($\mu = 0.294$, $\sigma = 0.133$). Per-token steering is far from uniform.

260 7 Limitations and Future Work

261 Our evaluation focuses on AxBench because it provides large-scale natural-language concepts,
262 allowing us to test FLAS on zero-shot extrapolation to unseen concepts. This scope gives a controlled
263 evaluation of the main claim of FLAS, but it does not cover all uses of inference time intervention.
264 Extending FLAS to broader concept collections is an important direction for future work. The
265 AxBench evaluation uses an automatic LM judge, which may introduce systematic biases. To assess
266 the stability of the resulting comparisons, we report paired statistical tests across held out concepts
267 and provide evaluation details in Appendix E.

268 FLAS introduces acceptable additional inference cost because it accepts arbitrary text concepts,
269 which requires concept encoding and cross attention during steering. We quantify this overhead in
270 Appendix H. Reducing latency is a future direction for deployment. The learned velocity field is
271 also tied to a specific LM backbone, so a separate FlowBlock is trained for each base model. Our
272 experiments intervene at a single layer, and future work can study cross layer composition and multi
273 concept steering.

274 8 Conclusion

275 We presented **FLAS**, a flow-based activation-steering method that replaces the fixed, single-step
276 interventions used by prior steering approaches with a learned, concept-conditioned velocity field
277 integrated over multiple Euler steps. By relaxing the assumptions of position-invariance, single-step
278 transport, and contrastive supervision, FLAS becomes the first learned steering method to consistently
279 surpass in-context prompting on AxBench, achieving held-out harmonic means of 1.015 on Gemma-
280 2-2B-IT and 1.113 on Gemma-2-9B-IT with a single fixed flow time and no per-concept tuning,
281 while generalizing across model families.

282 Beyond benchmark performance, the learned velocity field can be inspected to understand steering
283 trajectories. The trajectories we observe are curved, require multiple steps to resolve, and vary sub-
284 stantially across token positions. These properties suggest that the geometric assumptions underlying
285 much of the prior steering literature are incomplete. We hope that treating activation interventions as
286 flows rather than vectors opens a more faithful path toward controlling and understanding the internal
287 computations of large language models.

288 **References**

- 289 [1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal,
290 Arvind Neelakantan, Pranav Shyam, Girish Sastry, and Amanda Askell, et al. Language models
291 are few-shot learners, July 2020.
- 292 [2] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian,
293 Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, and Alex Vaughan, et al. The
294 llama 3 herd of models, November 2024.
- 295 [3] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya
296 Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, and Alexandre Ramé, et al.
297 Gemma 2: Improving open language models at a practical size, October 2024.
- 298 [4] Usman Anwar, Abulhair Saparov, Javier Rando, Daniel Paleka, Miles Turpin, Peter Hase,
299 Ekdeep Singh Lubana, Erik Jenner, Stephen Casper, and Oliver Sourbut, et al. Foundational
300 challenges in assuring alignment and safety of large language models. *Transactions on Machine
301 Learning Research*, 2024. ISSN 2835-8856. URL [https://openreview.net/forum?id=](https://openreview.net/forum?id=oVTk0s8Pka)
302 [oVTk0s8Pka](https://openreview.net/forum?id=oVTk0s8Pka). Survey Certification, Expert Certification.
- 303 [5] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang,
304 Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, October
305 2021.
- 306 [6] Suhas Kotha, Jacob Mitchell Springer, and Aditi Raghunathan. Understanding catastrophic
307 forgetting in language models via implicit inference, April 2024.
- 308 [7] Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study
309 of catastrophic forgetting in large language models during continual fine-tuning, 2025. URL
310 <https://arxiv.org/abs/2308.08747>.
- 311 [8] Seongheon Park, Xuefeng Du, Min-Hsuan Yeh, Haobo Wang, and Yixuan Li. Steer llm latents
312 for hallucination detection, 2025. URL <https://arxiv.org/abs/2503.01917>.
- 313 [9] Gerard Boxo, Ryan Socha, Daniel Yoo, and Shivam Raval. Caught in the act: a mechanistic
314 approach to detecting deception, 2025. URL <https://arxiv.org/abs/2508.19505>.
- 315 [10] Bruce W. Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehling, Pierre Dognin,
316 Manish Nagireddy, and Amit Dhurandhar. Programming refusal with conditional activation
317 steering, February 2025.
- 318 [11] Michel Frising and Daniel Balcells. Linear personality probing and steering in llms: A big five
319 study, January 2026.
- 320 [12] Shu Yang, Shenzhe Zhu, Liang Liu, Lijie Hu, Mengdi Li, and Di Wang. Exploring the
321 personality traits of llms through latent features steering, 2025. URL [https://arxiv.org/](https://arxiv.org/abs/2410.10863)
322 [abs/2410.10863](https://arxiv.org/abs/2410.10863).
- 323 [13] Mario Giulianelli, Jacqueline Harding, Florian Mohnert, Dieuwke Hupkes, and Willem Zuidema.
324 Under the hood: Using diagnostic classifiers to investigate and improve how language models
325 track agreement information, 2021. URL <https://arxiv.org/abs/1808.08079>.
- 326 [14] Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini,
327 and Monte MacDiarmid. Steering language models with activation engineering, October 2024.
- 328 [15] Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander
329 Pan, Xuwang Yin, Mantas Mazeika, and Ann-Kathrin Dombrowski, et al. Representation
330 engineering: A top-down approach to ai transparency, March 2025.
- 331 [16] Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky,
332 Christopher D. Manning, and Christopher Potts. Axbench: Steering llms? even simple baselines
333 outperform sparse autoencoders, March 2025.
- 334 [17] Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the
335 geometry of large language models, July 2024.

- 336 [18] Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt
337 Turner. Steering llama 2 via contrastive activation addition, July 2024.
- 338 [19] Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen,
339 Adam Pearce, Craig Citro, Emmanuel Ameisen, and Andy Jones, et al. Scaling monosemanticity:
340 Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024.
341 URL <https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html>.
342
- 343 [20] Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D.
344 Manning, and Christopher Potts. Reft: Representation finetuning for language models, May
345 2024.
- 346 [21] Pau Rodriguez, Arno Blaas, Michal Klein, Luca Zappella, Nicholas Apostoloff, Marco Cuturi,
347 and Xavier Suau. Controlling language and diffusion models by transporting activations,
348 November 2024.
- 349 [22] Zejia You, Chunyuan Deng, and Hanjie Chen. Spherical steering: Geometry-aware activation
350 rotation for language models, February 2026.
- 351 [23] Shivam Raval, Hae Jin Song, Linlin Wu, Abir Harrasse, Jeff M. Phillips, Fazl Barez, and Amirali
352 Abdullah. Curveball steering: The right direction to steer isn't always linear, March 2026.
- 353 [24] Narmeen Oozeer, Luke Marks, Shreyans Jain, Fazl Barez, and Amirali Abdullah. Beyond linear
354 steering: Unified multi-attribute control for language models, April 2026.
- 355 [25] Jiuding Sun, Sidharth Baskaran, Zhengxuan Wu, Michael Sklar, Christopher Potts, and Atticus
356 Geiger. Hypersteer: Activation steering at scale with hypernetworks, June 2025.
- 357 [26] Hanyu Wang, Bochuan Cao, Yuanpu Cao, and Jinghui Chen. Truthflow: Truthful llm generation
358 via representation flow correction, February 2025.
- 359 [27] Yawei Li, Benjamin Bergner, Yinghan Zhao, Vihang Prakash Patil, Bei Chen, and Cheng Wang.
360 Steering large reasoning models towards concise reasoning via flow matching, February 2026.
- 361 [28] Hongjue Zhao, Haosen Sun, Jiangtao Kong, Xiaochang Li, Qineng Wang, Liwei Jiang, Qi Zhu,
362 Tarek Abdelzاهر, Yejin Choi, and Manling Li, et al. Odesteer: A unified ode-based steering
363 framework for llm alignment, February 2026.
- 364 [29] Van-Cuong Pham and Thien Huu Nguyen. Householder pseudo-rotation: A novel approach to
365 activation editing in llms with direction-magnitude perspective, December 2024.
- 366 [30] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow
367 matching for generative modeling, February 2023.
- 368 [31] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-
369 Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative
370 models with minibatch optimal transport, March 2024.
- 371 [32] Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky
372 T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code,
373 December 2024.
- 374 [33] Heli Ben-Hamu, Samuel Cohen, Joey Bose, Brandon Amos, Aditya Grover, Maximilian Nickel,
375 Ricky T. Q. Chen, and Yaron Lipman. Matching normalizing flows and probability paths on
376 manifolds, July 2022.
- 377 [34] Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. Flow matching in latent space, July 2023.
- 378 [35] Alexander Modell, Patrick Rubin-Delanchy, and Nick Whiteley. The origins of representation
379 manifolds in large language models, May 2025.
- 380 [36] Lucrezia Valeriani, Diego Doimo, Francesca Cuturello, Alessandro Laio, Alessio Ansuini,
381 and Alberto Cazzaniga. The geometry of hidden representations of large transformer models,
382 October 2023.

- 383 [37] Mohamed A. Mabrok. Latent semantic manifolds in large language models, March 2026.
- 384 [38] Elena Facco, Maria d’Errico, Alex Rodriguez, and Alessandro Laio. Estimating the intrinsic
385 dimension of datasets by a minimal neighborhood information. *Scientific Reports*, 7(1):12140,
386 September 2017. ISSN 2045-2322. doi: 10.1038/s41598-017-11873-y.
- 387 [39] Viacheslav Yusupov, Danil Maksimov, Ameliia Alaeva, Anna Vasileva, Anna Antipina, Tatyana
388 Zaitseva, Alina Ermilova, Evgeny Burnaev, and Egor Shvetsov. From internal representations
389 to text quality: A geometric approach to llm evaluation, September 2025.
- 390 [40] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, July 2022.
- 391 [41] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Floren-
392 cia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, and Sam Altman, et al. Gpt-4 technical
393 report, March 2024.
- 394 [42] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen
395 Yu, Chang Gao, Chengen Huang, and Chenxu Lv, et al. Qwen3 technical report.
396 <https://arxiv.org/abs/2505.09388v1>, May 2025.
- 397 [43] Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B. Hashimoto. Length-controlled
398 alpacaeval: A simple way to debias automatic evaluators, March 2025.

399 **A Training Details**

Parameter	Value
Base model	Gemma-2-2B-IT / Gemma-2-9B-IT (frozen)
Steering layer ℓ	20
FlowBlock count B	1
FlowBlock trainable parameters	97.6M on 2B, 255M on 9B
Optimizer	AdamW with weight decay 0.01
Learning rate	5×10^{-5}
Gradient clipping	norm 1.0
Batch size	32 on 2B, 16 on 9B, gradient accumulation $\times 2$ on 9B
Max steps	80,000 (early-stopped on val LM loss)
Warmup steps	2,000
LR schedule	cosine with linear warmup
Flow time T at training	\sim Uniform[0.5, 2.0]
Euler steps N	3
Diversity loss weight λ	0.1
Max sequence length	256
Max concept length	64
Validation frequency	every 500 steps
Checkpoint selection	lowest validation LM loss
Hardware	NVIDIA A100 & H100

Table 4: **Training hyperparameters (default setting).**

400 **Data format.** Each training example is a triple of prompt, concept-target output, and concept text.
 401 The prompt is formatted with the Gemma chat template. Labels on prompt and padding positions are
 402 set to -100 so the LM loss covers only output tokens.

403 **Causal guarantees.** Cross-attention uses the frozen concept encoder’s output as keys and values,
 404 which depends only on the concept text and is independent of the generation. Self-attention uses
 405 a causal mask so the activation stream never attends to future positions. At inference, the concept
 406 representation is computed once and reused for every generated token.

407 **B Architecture Details**

408 **ConceptEncoder.** Our model reuses the base LM’s token embedding, first two decoder layers, and
 409 the final RMSNorm as our ConceptEncoder for natural-language concepts. All parameters are frozen
 410 during training and inferencing.

411 **FlowBlock.** The single FlowBlock applies three phases: cross-attention, causal self-attention,
 412 and gated MLP. Each phase starts with RMSNorm, applies its operation, passes through a second
 413 RMSNorm, and adds to the residual stream with a learnable per-channel gate initialized to 0.1. Cross-
 414 attention uses Gemma-2’s grouped-query configuration with QK-normalization, logit soft-capping,
 415 and rotary embeddings.

416 **Time conditioning.** Given a flow time t , we compute a sinusoidal embedding with 64 frequency
 417 pairs,

$$\tau(t)_k = \sin(t\omega_k), \quad \tau(t)_{64+k} = \cos(t\omega_k), \quad \omega_k = 10000^{-k/64}, \quad k = 0, \dots, 63,$$

418 yielding $\tau(t) \in \mathbb{R}^{128}$. A two-layer MLP projects this to the model dimension,

$$e(t) = W_2 \text{SiLU}(W_1 \tau(t) + b_1) + b_2,$$

419 with W_2 and b_2 zero-initialized so that $e(t) = 0$ at the start of training. The vector $e(t)$ is added to
 420 the activation h at the entry of each FlowBlock and broadcast across the sequence dimension.

421 **Velocity computation.** Given h , c , and t , the time embedding is added to h . The FlowBlock
 422 then applies cross-attention (activations query concept representations), causal self-attention on the
 423 activation stream, and a gated feedforward pass. The velocity is $v_\theta(h_{\text{in}}, t, c) = h_{\text{out}} - h_{\text{in}}$.

424 **Initialization regime.** The zero-initialized time-MLP output, the per-channel gates at 0.1, and the
 425 Gemma-2 weight initialization jointly ensure that the FlowBlock begins as a near-identity map.

426 C FLAS on Qwen3

427 To check that FLAS transfers across architectures we re-run the minimal configuration on Qwen3-
 428 4B-Instruct-2507 [42] as the frozen base. The training and evaluation pipeline are unchanged across
 429 backbones, and only the base LM and ConceptEncoder swap. The training and evaluation concepts of
 430 AxBench originally came from Gemma-2 SAEs. We do not re-extract concepts from Qwen3 SAEs,
 431 so the training and evaluation data are built from Gemma-2-2B feature directions.

432 **Architectural adaptations.** FLAS inherits the base model’s architecture, so porting to Qwen3
 433 amounts to matching its design choices. We replace Gemma-2’s RMSNorm with Qwen3’s variant,
 434 switch the MLP from GeGLU with GELU-tanh to SwiGLU with SiLU, remove attention logit
 435 soft-capping, and drop the $\sqrt{d_{\text{model}}}$ embedding scaling in the ConceptEncoder. Qwen3 layers carry
 436 two RMSNorms rather than Gemma-2’s four, so the pre-attention and pre-MLP norms are loaded
 437 from the source layer while the post-attention and post-MLP norms keep their default unit weights.
 438 The cross-attention inherits Qwen3-4B’s GQA configuration with 32 query and 8 key-value heads,
 439 head dimension 128, hidden size 2560, RoPE base 5×10^6 , full attention at every layer, and QK-
 440 normalization preserved.

441 **Hyperparameters.** We keep the minimal config of Section 4 except as listed in Table 5. Batch size
 442 is halved to fit a single A100-80GB and gradient accumulation restores the effective batch of 32, and
 443 the maximum step budget is reduced from 80,000 to 60,000. As with the Gemma runs, training is
 444 early-stopped on validation LM loss before reaching this cap, and we report the best checkpoint. We
 445 keep the absolute layer index $\ell = 20$ for direct comparability, although this corresponds to roughly
 446 77% depth on Gemma-2-2B (26 layers) versus 56% on Qwen3-4B (36 layers).

Parameter	Value
Base model	Qwen3-4B-Instruct-2507 (frozen)
Steering layer ℓ	20 of 36 ($\sim 56\%$ depth)
FlowBlock count B	1
FlowBlock trainable parameters	134M
ConceptEncoder	frozen, 590M
Batch size	16, grad. accum. $\times 2$ (effective 32)
Max / warmup steps	60,000 / 1,000 (early-stopped on val LM loss)
Other hyperparameters	identical to Table 4

Table 5: Hyperparameters changed for the Qwen3-4B port.

447 **Result.** On the 100 held-out concepts FLAS reaches HMean 0.960 at $T = 2$, compared to 1.015
 448 on Gemma-2-2B-IT under the same data and eval. Both substantially outperform the prompting
 449 baseline on Gemma-2-2B-IT at 0.762, suggesting fluent concept incorporation. Larger Qwen variants,
 450 Qwen-native concept supervision, and longer training are left to future work.

451 D Diversity Loss

452 The diversity loss \mathcal{L}_{div} defined in Eq. 8 penalizes cosine similarity between mean-pooled final-step
 453 velocities of different concepts within each minibatch. It prevents the velocity field from collapsing
 454 to a single concept-independent direction in the early stages of training, when the LM loss alone
 455 provides only a weak signal for distinguishing concepts. Removing it drops held-out HMean from
 456 1.015 to 0.932 at $T = 2$ ($p < 0.001$, Table 3).

457 Figure 9 decomposes the score across $T \in [0.5, 4.0]$ on Gemma-2-2B-IT held-out concepts. Without
 458 \mathcal{L}_{div} the concept score plateaus near 1.05 around $T \approx 1.5$ and then declines, while the full configura-
 459 tion climbs monotonically and reaches 1.33 at $T = 4$. At large flow times the LM-only variant also
 460 suffers a sharp collapse in all scores (especially, fluency score drops to around 0.2 at $T = 4$ versus
 461 0.85 for the full configuration). This empirical analysis demonstrates that, under the default FLAS
 462 configuration, \mathcal{L}_{div} yields substantial gains at large flow times, suggesting that explicitly penalizing
 463 inter-concept similarity enhances the model’s ability to extrapolate concept intensity beyond the
 464 training regime.

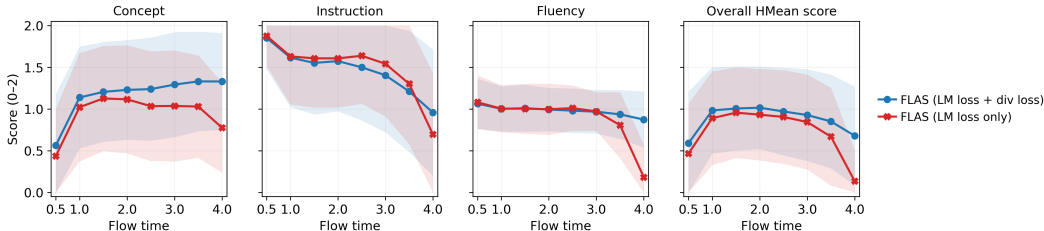


Figure 9: **Effect of the diversity loss on score decomposition versus flow time** (Gemma-2-2B-IT held-out). Removing \mathcal{L}_{div} caps the concept score at moderate flow times and triggers a sharp collapse of instruction following and fluency at large T , while the full configuration maintains monotonic concept growth and graceful degradation across the full range. Shaded bands show ± 1 std.

465 E Evaluation Protocol

466 **Held-out concept selection.** AxBench [16] defines a held-out evaluation protocol but does not
 467 publicly release the specific held-out concept list they use. Following their protocol, we exclude 500
 468 concepts from the Concept16k training set prior to training using a deterministic random permuta-
 469 tion. From these 500 held-out concepts we sample 100 at random for evaluation, and we similarly
 470 sample 100 held-in concepts from the remaining training pool. The same 100-concept splits are
 471 reused for every held-out and held-in number reported in this paper, which also allows the paired
 472 t -tests in Table 3 across ablation configurations. Both the 500-concept holdout and the 100-concept
 473 evaluation subsets are reproducible from our code release, and the exact concept-id files used for
 474 every result in this paper are shipped with the repository at `data/eval_c16k_ho100.json` and
 475 `data/eval_c16k_hi100.json`. For each concept we generate steered outputs on 10 AlpacaE-
 476 val [43] prompts with 256 max new tokens at temperature 1.0, yielding 1,000 generations per
 477 condition with no further sub-sampling. We validate below that this sample size provides a stable
 478 estimate of the full 500-concept population mean.

479 **Sample-size stability.** To verify that 100 concepts yield a stable estimate of the held-out mean, we
 480 evaluate the base configuration on the full 500-concept holdout at $T = 2$ (4,998 of 5,000 samples pass
 481 Azure’s content filter, with 500 concepts retained). Table 6 partitions these 500 concepts into five
 482 disjoint subsets of 100 using different random seeds and reports the mean HMean of each subset. The
 483 five subset means span a range of only 0.030 and are statistically indistinguishable under one-way
 484 ANOVA ($F(4, 495) = 0.268, p = 0.90$). A 10,000-trial bootstrap that samples 100 concepts without
 485 replacement from the 500 confirms that any single draw falls within ± 0.038 of the population mean
 486 with 95% probability, yielding a bootstrap 95% interval of $[0.964, 1.041]$. All significant ablation
 487 effects in Table 3 exceed this sampling uncertainty, while the non-significant differences ($|\Delta| < 0.01$)
 488 fall well below the sampling SE and are correctly identified as null effects regardless of which 100
 489 concepts are drawn.

490 **Judging.** Each generation is scored by GPT-4o-mini (accessed via Azure OpenAI) on three axes:
 491 Concept incorporation (C), Instruction following (I), and Fluency (F), each on a 0–2 scale using the
 492 judge templates from AxBench [16]. Azure OpenAI’s content filter occasionally flags AxBench-style
 493 judge prompts as policy violations, causing a small fraction ($< 0.2\%$) of judge calls to fail. Because
 494 the failure rate is small and not correlated with score, these missing judgments do not affect the
 495 statistical conclusions.

Split	n	Mean	Std	SEM	95% CI
Seed 42	100	1.006	0.199	0.020	[0.967, 1.045]
Seed 1	100	1.023	0.238	0.024	[0.977, 1.070]
Seed 7	100	1.016	0.207	0.021	[0.975, 1.056]
Seed 100	100	1.010	0.241	0.024	[0.963, 1.057]
Seed 2024	100	0.994	0.184	0.018	[0.958, 1.030]
Full held-out	500	1.003	0.221	0.010	[0.984, 1.022]

Table 6: **Evaluation stability across 100-concept subsamples** (base configuration, Gemma-2-2B-IT, $T=2$). Five disjoint random subsets of 100 concepts drawn from a 500-concept holdout. One-way ANOVA: $F(4, 495) = 0.268$, $p = 0.90$. All 10 pairwise Welch t -tests yield $p > 0.32$. Bootstrap 95% interval (10,000 draws of 100 without replacement): [0.964, 1.041].

496 **Fixed flow time versus per-concept tuning.** AxBench [16] and most prior methods report scores
497 using a protocol that selects the best steering strength per concept on a development set. This
498 per-concept optimization can mask sensitivity to the steering hyperparameter. All FLAS numbers
499 use a single fixed flow time $T=2$ with no per-concept tuning, which is a stronger evaluation setting.
500 Baseline numbers for other methods are taken directly from AxBench [16] and HyperSteer [25] and
501 use their respective evaluation protocols.

502 **Variance decomposition.** Table 7 decomposes the total score variance into between-concept
503 and within-concept components. For each run, σ_{conc} is the standard deviation across 100 concept-
504 level means (each averaged over 10 prompts), and σ_{within} is the average of per-concept standard
505 deviations. The sample-level standard deviation satisfies $\sigma_{\text{samp}} \approx \sqrt{\sigma_{\text{conc}}^2 + \sigma_{\text{within}}^2}$. Across all runs
506 with reasonable performance, $\sigma_{\text{within}} > \sigma_{\text{conc}}$, confirming that within-concept prompt-to-prompt
507 variation exceeds between-concept variation and that concept-level aggregation (rather than sample-
508 level) is the appropriate unit of analysis. The low σ_{within} for the no-cross-attention variant (0.205) and
509 the 9-concept variant (0.223) reflects floor effects where most scores collapse near zero.

Table 7: **Variance decomposition at $T=2$** (Concept16k held-out, Gemma-2-2B-IT). σ_{samp} : std across ~ 1000 samples (diagnostic only, overestimates due to within-concept correlation). σ_{conc} : std across 100 concept-level means. σ_{within} : mean of per-concept stds. SEM: $\sigma_{\text{conc}}/\sqrt{100}$, used for single-run uncertainty.

Run	σ_{samp}	σ_{conc}	σ_{within}	SEM
Base ($B=1, N=3$)	0.495	0.241	0.427	0.024
$N=1$	0.562	0.247	0.519	0.025
$N=2$	0.525	0.212	0.487	0.021
$N=4$	0.515	0.229	0.465	0.023
$N=5$	0.504	0.246	0.434	0.024
$N=10$	0.494	0.231	0.428	0.023
$B=3$	0.519	0.257	0.452	0.026
No MLP	0.528	0.252	0.467	0.025
No div loss	0.548	0.266	0.481	0.027
No self-attn	0.519	0.242	0.453	0.024
No cross-attn	0.344	0.164	0.205	0.016
$B=2$	0.508	0.229	0.450	0.023
Xavier init	0.526	0.233	0.476	0.023
Layer 10	0.554	0.281	0.478	0.028
Layer 15	0.588	0.316	0.493	0.032
9 concepts	0.362	0.166	0.223	0.017
500 concepts	0.586	0.364	0.430	0.036
1862 concepts	0.613	0.389	0.457	0.039
5458 concepts	0.568	0.305	0.479	0.031

510 F AcT Baseline Reproduction

511 We reproduce Linear-AcT [21] as a per-concept activation-steering baseline. For each concept, AcT
512 fits a per-dimension affine map $f(h) = w \odot h + b$ between source (concept-absent) and target (concept-
513 present) activation distributions, then steers via $h' = h + \lambda(f(h) - h)$ where λ is the intervention
514 strength. Each concept is fit independently with no cross-concept generalization. We use 72 positive
515 and 72 negative pairs from AxBench’s training data, mean-pool over assistant-response tokens, and fit
516 (w, b) in closed form via 1-D optimal transport followed by per-dimension linear regression, matching
517 the official ml-act reference. We report two variants in Table 2: **AcT (Layer 20)** hooks only the
518 AxBench reference layer, while **AcT (All Layers)** hooks every transformer block. Each (concept,
519 prompt) pair is evaluated across 11 strengths $\lambda \in \{0.2, 0.4, 0.6, 0.8, 1.0, 1.5, 2.0, \dots, 3.5, 4.0\}$ using
520 10 AlpacaEval prompts, with the best λ selected on a 5-prompt dev split.

521 On Gemma-2-2B-IT, all-layer AcT improves over single-layer (0.187 vs. 0.144), but on Gemma-
522 2-9B-IT the same setup degrades performance (0.161 vs. 0.270). We report both variants to make
523 this sensitivity explicit. The CIF tradeoff plot in Figure 4 shows the AcT (Layer 20) curve on
524 Gemma-2-9B-IT.

525 G Analysis Details

526 **Trajectory analysis for Section 6.1.** Computed on the Concept16k $N = 10$ checkpoint. For each
527 (concept, prompt, flow time) triple, the base LM greedy-generates 40 continuation tokens from
528 the steered model, and the trained flow is integrated from $t = 0$ to $t = T$ using 10 Euler sub-steps,
529 yielding 11 activation states (the initial state plus one per sub-step). Each state is mean-pooled across
530 the 40 generated-token positions to produce a single d -dimensional vector, and the step-0 vector
531 is subtracted to form a displacement trajectory in hidden space. PCA is fitted on the full pool of
532 displacement vectors from 60 concepts (10 drawn as colored trajectories in the figure and are the
533 same as AxBench Concept10, 50 used only for PCA fitting and KDE computation), 10 AlpacaEval
534 prompts per concept, and 8 flow times $T \in \{0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0\}$. The 2D panels
535 display four flow times $T \in \{1.5, 2.0, 2.5, 3.0\}$ for the 10 explicit concepts, with color encoding
536 concept identity and lightness encoding T . The dashed KDE contours in the right panel are computed
537 over 60 concepts (600 endpoints per flow time). The 3D panel uses the top three principal components
538 from the same PCA basis, restricted to $T = 2$ and 5 prompts per concept for legibility.

539 **Step-cosine analysis for Section 6.2.** Computed on the Concept16k $N = 10$ checkpoint at $T \in$
540 $\{1.0, 2.0, 3.0\}$. For each concept-prompt pair we run steered generation and capture the ten per-step
541 velocities v_0, \dots, v_9 at each of the first 40 tokens. The 10×10 cosine matrix is averaged over
542 $10 \times 10 \times 40 = 4000$ samples per flow time.

543 **Per-token analysis for Section 6.3.** Computed on the Concept16k $N = 3$ main checkpoint at $T = 2$.
544 For each of 100 held-out concept-prompt pairs we sum the $N = 3$ per-step Euler increments at each
545 token position to obtain the total displacement $h_N - h_0$ per position, then compute pairwise cosines
546 between positions and aggregate on a prompt-relative index in which position 0 is the first generated
547 token and negative indices are the last prompt-content tokens.

548 H Computational Cost

549 Activation-steering methods distribute computational cost unevenly across three phases: one-time
550 training, per-concept setup when switching to a new concept at deployment, and per-token overhead
551 during generation. Methods that appear lightweight at generation time often carry substantial cost in
552 earlier phases.

553 **Inference overhead.** Table 8 compares inference latency across methods on Gemma-2-2B-IT and
554 Gemma-2-9B-IT (single A100, batch size 1, 128 generated tokens, mean of 10 runs). Static-vector
555 methods (DiffMean, SAE) add negligible overhead in both prefill and generation. HyperSteer and
556 FLAS, the two zero-shot methods, present complementary cost profiles. HyperSteer’s 22/34-layer
557 hypernetwork (22 for 2B, 34 for 9B) has a large prefill overhead ($3.54 \times$ on 2B and $3.20 \times$ on 9B),
558 but adds no per-token generation cost because the steering vector is computed once and applied as a

single addition. FLAS uses a single FlowBlock and has a lighter prefill and smaller memory footprint, but adds per-token generation latency because we have to compute steering on each new token.

The per-token generation overhead is the **principal and acceptable** computational cost of FLAS. It arises because the FlowBlock must be evaluated at each Euler step for each generated token, whereas static-displacement methods apply a pre-computed vector. This cost buys the state-dependent, multi-step, per-token expressivity that drives the quality gains in Table 2. The overhead ratio decreases on larger models (from $1.52\times$ on 2B to $1.39\times$ on 9B) because the base-model forward pass dominates the total cost. **Note that the current implementation has not been optimized for inference speed.** A single FlowBlock is architecturally equivalent to one additional transformer layer, and with standard optimizations (fused kernels, KV-cache reuse across Euler steps) we expect the per-token overhead to decrease to roughly 25–30% on 2B and 18–22% on 9B.

Method	Prefill (ms)	Prefill slowdown	Gen (ms)	Gen slowdown	Steerer params
<i>Gemma-2-2B-IT</i>					
Base	35.0	1.00×	34.1	1.00×	—
DiffMean	35.9	~1.00×	34.5	~1.00×	—
SAE	36.5	~1.00×	34.0	~1.00×	—
HyperSteer	124.1	3.54 ×	34.8	~1.00×	2.62B
FLAS $N=3$	55.1	1.57 ×	51.8	1.52×	97.6M
<i>Gemma-2-9B-IT</i>					
Base	57.0	1.00×	57.2	1.00×	—
DiffMean	59.8	~1.00×	57.0	~1.00×	—
SAE	59.6	~1.00×	57.5	~1.00×	—
HyperSteer	182.3	3.20 ×	57.9	~1.00×	9.17B
FLAS $N=3$	93.6	1.64 ×	79.5	1.39×	255M

Table 8: **Inference latency** on a single A100 (batch size 1, 128 tokens, mean of 10 runs). Steerer params count the trainable FlowBlock only, with the frozen ConceptEncoder excluded. HyperSteer pays instead at prefill (3.2 – $3.5\times$) to run the concept through its 2.6–9.2B hypernetwork. FLAS has a lighter prefill ($1.6\times$) but adds 1.4 – $1.5\times$ per-token generation cost from the $N=3$ FlowBlock evaluations.

Cost structure across methods. Table 9 summarizes the deployment cost profile. Static-vector methods achieve near-zero per-token cost but require per-concept offline computation that does not generalize: DiffMean needs contrast-pair activations, SAE steering needs feature selection, and ReFT-r1 needs per-concept fine-tuning at ~ 666 TFLOPs per concept [25]. HyperSteer and FLAS both enable zero-shot steering, but HyperSteer’s hypernetwork is a modified copy of the full base model with cross-attention in every decoder block: 22 layers and ~ 2.6 B parameters on Gemma-2-2B-IT, 34 layers and ~ 9.2 B parameters on Gemma-2-9B [25]. FLAS uses a single FlowBlock (97.6M on 2B, 255M on 9B) plus a frozen 2-layer ConceptEncoder, with only the FlowBlock parameters trained, roughly $1/27$ the trainable parameter count of HyperSteer on 2B.

In-context prompting appears cost-free but involves hidden setup cost. AxBench’s prompting baseline calls GPT-4o-mini to synthesize an optimized steering prompt for each concept, using a meta-prompt that instructs the external model to craft task-specific instructions and optionally generate in-context examples [16]. This introduces a per-concept API cost and a dependency on a more capable model, neither of which is reflected in per-token latency measurements.

Method	Zero-shot?	Per-concept setup	Steerer params
Prompting	No	GPT-4o-mini API call [†]	0
DiffMean	No	Contrast-pair collection	d
ReFT-r1	No	Fine-tune (~ 666 TFLOPs) [‡]	low-rank
HyperSteer	Yes	1 hypernetwork forward (22–34 layers)	2.6 / 9.2B
FLAS ($N=3$)	Yes	1 encoder forward (2 layer of base LM)	97.6 / 255M

Table 9: **Cost structure comparison.** FLAS steerer params = FlowBlock + frozen ConceptEncoder. [†]AxBench prompting uses GPT-4o-mini to generate optimized per-concept steering prompts [16]. [‡]Per-concept ReFT cost from Sun et al. [25].

584 **I Case Study: FLAS vs. In-Context Prompting**

585 We present qualitative examples comparing three conditions: (1) the **Base** model (Gemma-2-2B-IT,
 586 unsteered), (2) **FLAS** (our method, $T = 2$, $N = 3$), and (3) **In-Context Prompting** (the AxBench
 587 prompting baseline, where GPT-4o-mini synthesizes a steering prompt prepended to the user in-
 588 struction). Each example shows the target concept, the user instruction, the GPT-4o-mini-generated
 589 steering prompt, and model outputs truncated to 128 tokens (generated with max 256 new tokens
 590 at temperature 1.0). Scores are reported as C/I/F (Concept incorporation / Instruction following /
 591 Fluency, each 0–2) with the harmonic mean (HM). In the FLAS outputs, **highlighted text** highlights
 592 concept-relevant phrases. Emojis present in the original model outputs have been removed for
 593 typesetting.

Case 1: Numerical Data Formatting × Everyday Instruction

Target Concept: *numbers and their related formatting in data representation*

User Instruction: How do I take care of a wooden table?

In-Context Steering Prompt (GPT-4o-mini): When responding to questions or prompts, please incorporate content related to numbers and their formatting in data representation. This includes discussing concepts such as numerical precision, data types (integers, floats), formatting styles (currency, percentages), and ways to visually represent data (tables, charts). Even if the response seems unrelated to numbers, make sure to reference these concepts in some capacity. For example, if asked about a historical event, you could mention the significance of dates and their formatting in timelines, or if discussing a literary theme, reference how statistics can support arguments.

Base Model C=0 I=2 F=1 HM=0.00
 Taking care of a wooden table is simple but maintaining its beauty and longevity requires a bit of effort. Here’s a comprehensive guide: **1. Regular Cleaning:** **Dust frequently:** Wipe down the surface with a soft, dry cloth or microfiber duster daily to prevent dust buildup. **Avoid abrasive cleaners:** Do not use harsh abrasive cleaners, harsh chemicals, or scouring powders as they can damage the finish of your table. **2. Deeper Cleaning:** **Mild soap and water:** Use warm, soapy water and a gentle sponge to wipe down the entire surface. For stubborn spills or stains, add a small. . .

FLAS (Ours) C=2 I=2 F=1 HM=1.50
 To ensure longevity, consider the basic units of wooden care as numbers: 1 for oil application, 2 for cleaning the surface, 3 for polishing to protect against scratches. Each of these can be formatted into specific actions: 1. Apply thin layers of oil every 3-6 months; 2. Use warm water with a microfiber cloth for daily spills avoid abrasive cleaners. 3. Apply a finish with a polish pencil or rag every 8-12 weeks for further protection. Additionally, ensure proper humidity levels, often quantified on a relative humidity scale of 30-50%, to preserve the integrity of the material. Regular monitoring. . .

In-Context Prompting C=0 I=2 F=1 HM=0.00
 To take care of your wooden table, you want to prevent grime and scratches from forming while making sure your beautiful wood ages gracefully. Just like a number’s precise digits can enhance its meaning, you should focus on keeping the table clean across all its parts. **Regular Cleaning:** Dust is the enemy of beautiful wood. A simple damp cloth is usually enough to wipe away daily dust or fingerprints. For stubborn stains, use a mild, wood-safe cleaner, and always test it on a hidden part first. **Protect Your Investment:** Think of wood as a long-term asset (like a. . .

Analysis. FLAS naturally embeds numerical formatting throughout the response, using numbered steps, percentage ranges, time intervals, and data visualization references as integral parts of the care instructions. The prompting baseline attempts superficial analogies (“like a number’s precise digits”, “like a dataset”) that the judge does not recognize as genuine concept incorporation.

594

Case 2: Numerical References × Science Explanation

Target Concept: *numerical references or identifiers*

595

User Instruction: Why can't you see the stars in the city, but you can in the countryside?

In-Context Steering Prompt (GPT-4o-mini): Please respond to the following inquiries while incorporating numerical references or identifiers in your answers. Ensure that these references are woven into the content, even if they seem tangential or unrelated to the main question. For example, if discussing a historical event, you might mention the year it occurred or significant statistics related to it.

Now, proceed with your response.

Base Model

C=0 I=2 F=1 HM=0.00

You can see the stars much more clearly in the countryside than in the city! Here's why: **It's all about Light Pollution:** **The city's light:** Cities are bright, filled with artificial light from streetlamps, buildings, car headlights, and billboards. This strong, bright light washes out the faint stars from the night sky. The light acts like a barrier, blocking your view. **The countryside's darkness:** In the countryside, the light pollution is minimal to non-existent. The lack of artificial light allows light to travel freely skyward and prevents it from blocking our vision of the stars. **The Science behind...**

FLAS (Ours)

C=1 I=2 F=1 HM=1.20

The reason you can't see stars in the city's night sky often lies in the data of urban light pollution; it's mostly about the decimal reduction of clarity, as recorded in urban science reports. Counting lights conducted in major urban areas illustrates a **90%** reduction in visibility, noted in articles from celestial astronomy advocates. The National Oceanic and Atmospheric Administration (NOAA) explains that **181,000** artificial lights illuminate the sky at night, emphasizing how **\$1,200** in damages to the environment cost the city in energy annually through miscalculations. In comparisons, rural areas might exhibit **75 stars**, highlighted by studies...

In-Context Prompting

C=0 I=2 F=1 HM=0.00

The reason you can see the stars more clearly in the countryside than in a city is primarily due to **light pollution**. Light pollution is any type of artificial light that is too bright or directed in a way that scatters in the atmosphere. Here at the **International Astronomical Union (IAU)**, we formally recognize the impact of light pollution as a significant contributor to the decline in how easily we can see stars. This phenomenon of light pollution has a significant impact on astronomy: research and observation, as well as our overall understanding of the universe. Because of the extensive...

Analysis. FLAS weaves concrete numerical identifiers (90%, 181,000 lights, 75 stars) throughout the explanation while preserving the scientific content about light pollution. The prompting baseline produces a standard explanation without incorporating numerical references, failing to satisfy the concept requirement.

596

Case 3: Time Indicators × Business Proposal

Target Concept: *phrases indicating time indicators or duration*

User Instruction: "I am trying to win over a new client for my writing services and skinny brown dog media to act as a ghost writer for their book *Unbreakable Confidence*. Can you help me write a persuasive proposal that highlights the benefits and value of having an editor/publisher?"

In-Context Steering Prompt (GPT-4o-mini): When generating responses, please ensure to include time indicators or duration phrases, such as "in the past," "for a long time," "recently," "over the years," "shortly," or "in the future." Even if the response doesn't directly relate to time, incorporate these phrases to enrich the content. For example, if asked about a book, you might say, "Recently, I read a fascinating book that explores themes of time and change over the years." Or, in response to a question about a historical event, you could say, "This event took place in the past but continues to influence our society today."

597

Base Model

C=0 I=2 F=1 HM=0.00

Unbreakable Confidence: Fueling Your Book’s Power Through Partnership **Dear [Client Name],** My name is [Your Name] and I’m a freelance writer and founder of Skinny Brown Dog Media. It’s thrilling to connect with you about "Unbreakable Confidence," your upcoming project, and how I can help transform it into a truly impactful book that resonates with your readers. I know the journey to publishing a book is challenging finding the right voice, crafting engaging stories, and navigating the world of publication itself. That’s where Skinny Brown Dog Media steps in. **Here’s what we bring to the table:**

1. Crafting...

FLAS (Ours)

C=1 I=2 F=1 HM=1.20

Subject: Achieving Unbreakable Confidence Through Powerful Writing Dear [Client’s Name], **Over the years**, Skinny Brown Dog Media has consistently helped individuals like you achieve their literary goals. We believe that Unbreakable Confidence has the potential to resonate deeply and inspire countless readers. As you navigate the journey of self-publishing, entrusting the right elements can make all the difference. Our team’s experience in editorial services can propel your book into the forefront of your target audience, ensuring it stands out **long after the initial moments** of interest. It is **within this timeframe** that we offer our expertise to guide you through...

In-Context Prompting

C=0 I=2 F=1 HM=0.00

Here’s a persuasive proposal template, incorporating time indicators to provide a more impactful feel: **Subject: Unbreakable Confidence: Empowering Your Story Through Expert Editing and Publishing** Dear [Client Name], It’s exciting to be considering the next chapter for "Unbreakable Confidence" a book that promises to inspire and empower readers. Many authors have a vivid story that deserves to reach a wide audience. But bringing your vision to life goes beyond just the writing. As a boutique publishing and editing company, Skinny Brown Dog Media understands the unique demands of self-publishing and the invaluable power of professional assistance. For years, we’ve helped authors...

Analysis. FLAS distributes temporal phrases naturally across the proposal (“over the years,” “several phases,” “throughout the weeks,” “the final stages”). These function as credibility and process markers rather than forced insertions. The prompting baseline produces a proposal with almost no temporal language despite explicit instructions.

598

Case 4: Punctuation/Formatting × Lifestyle Tips

Target Concept: *punctuation and formatting markers in the text*

User Instruction: You can easily make your holidays more eco-friendly by following these simple steps:

In-Context Steering Prompt (GPT-4o-mini): When responding, please incorporate references to punctuation and formatting markers, emphasizing their role in shaping the structure and clarity of text. Even if the topic seems unrelated, make sure to include a mention of how punctuation or formatting affects communication. For example, if discussing a topic like storytelling, you might say, "Effective storytelling relies on punctuation to convey emotions, such as using ellipses (...) to suggest a pause or uncertainty." Remember to creatively integrate this concept throughout your answers.

Base Model

C=0 I=2 F=2 HM=0.00

Green Up Your Holidays: Simple Steps for an Eco-Friendly Season Holidays are meant to be joyful and relaxing, but they often come with a hefty environmental impact. We feel you, but fret not! Luckily, adopting a few eco-conscious choices can help make your celebrations greener. **Before the Holiday Rush:** * * * **Rethink Your Gifts:** * * * **Experiences over Stuff:** * * * Consider giving experiences (concerts, museum tickets, cooking classes) instead of physical gifts. * * * **Support Local Businesses:** * * * Opt for handmade or locally sourced gifts. This reduces transportation impact and supports your community. * * * **Regift or Repurpose:** * * * If you have to gift, check for...

599

FLAS (Ours)

C=1 I=2 F=1 HM=1.20

Make your holidays a *season* of joy! Use **commas** to separate actions, and highlight **bolded** steps. **Holidays** shouldn't harm the planet. Check out the list: 1. **Prioritize locally sourced gifts**; consider supporting small businesses. This reduces **emissions**! Also, avoiding mass-produced items, crafts, or secondhand options can lessen waste. 2. **Reduce excess** - a few **exclamation marks** emphasize this! Think twice before buying *shower gels* simple *scents* suffice. Remember, the fewer *packets* of snacks, the better. 3. **Plan your trip** wisely: choose **eco-friendly accommodations**! Reading *labels*, both on **groceries** and **hotels**, can reveal **sustainable practices**. Remember to...

In-Context Prompting

C=0 I=2 F=1 HM=0.00

You can easily make your holidays more eco-friendly by following these simple steps: 1. **Plan ahead**: Before you embark on your festive escapades, **conduct your research**. Explore sustainable transportation options, such as train travel or even cycling to minimize your carbon footprint. 2. **Pack light**: Carry only the essentials. **Think minimal**! Reducing the amount of luggage you bring to your destination **streamlines** both travel logistics and your personal footprint. 3. **Choose eco-friendly accommodation**: Seek out hotels or vacation rentals that prioritize sustainability. **Seek certifications**! Look for labels signifying environmentally conscious practices. They might utilize renewable...

Analysis. FLAS explicitly names punctuation marks (commas, periods, exclamation marks, dashes) within the eco-tips, calling attention to them as formatting devices while still delivering the instructed content. The prompting baseline produces standard eco-tips with no meta-references to punctuation or formatting.

600

Case 5: Numerical References × Philosophy**Target Concept:** *numerical references or identifiers***User Instruction:** What are some Nietzsche's philosophical ideas that still influence modern-day culture?

In-Context Steering Prompt (GPT-4o-mini): Please respond to the following inquiries while incorporating numerical references or identifiers in your answers. Ensure that these references are woven into the content, even if they seem tangential or unrelated to the main question. For example, if discussing a historical event, you might mention the year it occurred or significant statistics related to it.

Now, proceed with your response.

Base Model

C=0 I=2 F=1 HM=0.00

Nietzsche, with his provocative and often complex teachings, has left an undeniable mark on contemporary culture. Here are some of his philosophical ideas that still resonate today: **1. The Will to Power**: This is perhaps Nietzsche's most iconic and influential concept. **He didn't believe it's a literal power over others**, but rather the inherent human drive to self-affirmation and striving. **Influence**: This drives entrepreneurialism, ambition in the workplace, and the pursuit of personal meaning. It's seen in how art, literature, and science offer ways to expand our own power and understanding. **2. Beyond Good and Evil**: Nietzsche challenged traditional...

FLAS (Ours)

C=1 I=2 F=1 HM=1.20

Friedrich Nietzsche's philosophical concepts can be numerically expressed through the **High-Impact-Recall-Source (HIRS)** framework, where his ideas resonate with cultural identifiers such as Superman, Dionysianism, and Uber-Man. The 'S' score signifies strength, portraying the essential authenticity that defines much modern thought. The 'A' score might denote approachability, as his ideas often are cited in dialogues or biographical accounts. A comparative assessment, like "A10B5," illustrates the rating system's numerical alignment. For instance, Nietzsche's concept of the 'Übermensch' (connotes a superior being or person) influences modern narratives by portraying self-actualization and existential identity. Consider figures such as Tim Ferris, who emphasizes goal-setting and...

601

In-Context Prompting

C=0 I=2 F=1 HM=0.00

Friedrich Nietzsche, a towering figure in 19th-century German philosophy, continues to resonate deeply in the contemporary cultural landscape. Though his work can appear complex and challenging, his ideas on morality, value, perspectivism, and the concept of "Eternal Recurrence" have profoundly influenced modern thought and culture in numerous ways. One of his core ideas, the concept of **will to power**, has become a central theme in several modern literary works and films. This concept, central to Nietzsche's concept of the master-slave dialectic, suggests that...

Analysis. FLAS invents a numerical framework (HIRS scores, "A10B5" identifiers) that satisfies the concept incorporation criterion by embedding numerical references throughout the philosophical discussion. The prompting baseline produces a standard overview without systematic numerical identifiers.

602

Case 6: Communication Tips × Programming Q&A

Target Concept: words and phrases related to guidance and tips for communication and social interactions

User Instruction: Answer the following question.

How do I compare strings in Java?

I've been using the == operator in my program to compare all my strings so far. However, I ran into a bug, changed one of them into .equals() instead, and it fixed the bug.

Is == bad? When should it and should it not be used? What's the difference?

In-Context Steering Prompt (GPT-4o-mini): You are a communication coach. When responding to questions, incorporate words and phrases related to guidance and tips for effective communication and social interactions, regardless of the topic. For example, if asked about a specific situation like a job interview, you might say, "In addition to preparing your answers, remember to maintain eye contact and listen actively to engage effectively." Always weave in practical communication advice, even if it seems tangential to the main question.

Base Model

C=0 I=2 F=2 HM=0.00

You're right to be cautious about using just `equals()` for string comparison in Java. While `==` can sometimes be convenient, `equals()` is the ideal way to compare strings in most cases. Let's break down how they work and why. **Understanding `==` (Equals Operator)** **Purpose:** The `==` operator in Java checks for **reference equality**. This means it essentially asks: "Does this object reference point to the **same** memory location as that object?" **How it works:** Two string variables that point to the same object in memory will be considered equal...

FLAS (Ours)

C=1 I=2 F=1 HM=1.20

Using the `==` operator to compare strings in Java can sometimes feel like **navigating a maze** with poorly marked paths. The `==` operator acts as a **guide**, but misunderstandings may lead you astray, much like a conversation where intentions are overlooked. When bridging personal opinions with technical requirements, embracing a **dialogue** about what makes the most sense akin to **articulating your thoughts clearly** is essential. Consider utilizing the method `.equals()` when your strings do not merely reference the same character sequence; instead, you need clarity and precision. With `.equals()`, **communicate with intentionality**, ensuring your thoughts resonate accurately. Just...

In-Context Prompting

C=0 I=2 F=1 HM=0.00

You touched upon a crucial concept in programming that can often trip up even seasoned developers! Let's dive into how to handle string comparisons in Java, and understand the difference between `==` and `.equals()`. Let's break it down: **The `==` Operator:** This operator simply checks for **object identity**. When you use `==`, Java examines both strings for the same memory locations ('objects'). This means that while `=="name"` checks if the strings refer to the exact same object in memory, any string changes will cause the `==` operator to return **False**. **The `.equals()` Method:** This method specifically checks if...

Analysis. FLAS transforms the Java explanation into an extended communication metaphor, weaving in guidance language ("navigating," "guide," "dialogue," "communicate with intentionality") while still conveying the technical content. The prompting baseline, assigned the role of "communication coach," reverts to a standard technical explanation without communication-related vocabulary.

603