ZOSTEER: STEERING LANGUAGE MODELS FROM SCALAR REWARDS ALONE

Anonymous authors
Paper under double-blind review

ABSTRACT

Activation steering controls a language model by adding a fixed vector to its residual stream, but existing methods derive this vector from contrastive prompt pairs or differentiable objectives, leaving behaviors specified only by a scalar, non-differentiable reward largely out of reach. We introduce ZOSteer, which estimates a steering vector as the zeroth-order gradient of the reward in activation space. We inject isotropic Gaussian perturbations at a single layer, score each perturbed model with the reward, and take the cross-covariance between perturbations and rewards. This estimates the gradient of the Gaussian-smoothed reward from forward passes alone, without contrastive data or backpropagation. Because the reward is an average over items, the same perturbations also yield a gradient for every training item at no extra cost. The residual spectrum of these item gradients reveals sub-populations whose gradients conflict with the majority, and reweighting them repairs the pooled direction without new forward passes. Across three 1B-scale models and four benchmarks, ZOSteer gives the largest gains on GSM8K for every model, and its reweighted variant closes most of the gap to the strongest baseline on bAbI QA20 and surpasses it on CUTE SubChar, typically with fewer perturbations than the hidden dimension. These results suggest that what a single vector can steer depends on how a task's items agree, not only on how the model represents the capability.

1 INTRODUCTION

Activation steering shifts a language model's behavior by adding a fixed vector to its residual stream at inference time. Grounded in the linear representation hypothesis (Park et al., 2023), it has become a popular tool for controlling truthfulness, safety, and persona (Turner et al., 2023; Rimsky et al., 2024; Zou et al., 2023; Jorgensen et al., 2023), as neither extracting nor deploying a vector updates the model's parameters. The dominant extraction method, Contrastive Activation Addition (CAA) (Rimsky et al., 2024), takes the mean activation difference between prompt pairs that bracket the target behavior. This presumes that the behavior can be expressed as a contrast, which many practically important targets cannot: whether a math answer is correct or a program passes its unit tests is not naturally written as a pair of prompts, only scored as a scalar on the model's output.

Other methods relax the contrastive requirement but introduce supervision of their own. Optimization-based methods train the vector with cross-entropy (Wu et al., 2024), preference (Cao et al., 2024), or gradient-based objectives (Sharma & Trivedi, 2026), requiring backpropagation and labeled data, while sparse-autoencoder (Cho et al., 2025; Zhao et al., 2025) and teacher-trace methods (Belitsky et al., 2025; Venhoff et al., 2025) require a trained autoencoder or a stronger teacher. Two gaps remain. First, no existing method estimates a steering direction directly from a scalar reward. Second, when linear steering underperforms, current methods offer no diagnostic for *why*: whether the task lacks a linear direction, or whether one exists but the training items disagree on what it should be.

We address the first gap with **ZOSteer**, which estimates the steering vector as a zeroth-order gradient of the reward in activation space. We inject isotropic Gaussian perturbations at a single layer, evaluate each perturbed model on a fixed set of training prompts, and take the cross-covariance between perturbations and rewards. By Stein's lemma (Stein, 1981), this estimates the gradient of the Gaussian-smoothed reward, which is well defined even when the reward is a binary pass/fail indica-

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.

tor. The estimator is a single step of the random-direction gradient estimators used in zeroth-order optimization (Nesterov & Spokoiny, 2017; Salimans et al., 2017; Malladi et al., 2023), applied to a d-dimensional activation offset rather than to billions of parameters, which keeps the sample cost practical.

The same perturbations address the second gap. Because the reward averages over training items, the pooled gradient is the mean of per-item gradients, and the stored perturbations yield a gradient for every item at no extra cost. These item gradients expose what the pooled vector averages away: a large eigenvalue in the spectrum of their residuals signals a sub-population whose gradients conflict with the majority's, and reweighting the two groups balances their competing demands on a single direction.

Our contributions are as follows:

1. We propose ZOSteer, which estimates a steering vector from a scalar reward alone as a zeroth-order gradient in activation space, using forward passes only.
2. We show that per-item gradients, obtained from the same perturbations, reveal conflicting sub-populations within a task, and that reweighting them repairs the pooled direction without new forward passes.
3. Across three models and four benchmarks, ZOSteer gives the largest gains on GSM8K for every model, and its reweighted variant closes most of the gap to BiPO on bAbI QA20 and surpasses it on CUTE SubChar.

2 Related Work

2.1 Geometry of Representations in Language Models

The linear representation hypothesis (Park et al., 2023; Nanda et al., 2023) posits that high-level concepts are encoded as linear directions in activation space, providing the theoretical foundation for steering vectors. Empirical studies have found evidence for both linear (Marks & Tegmark, 2023; Hernandez et al., 2024; Gurnee & Tegmark, 2024; Arditi et al., 2024; Tigges et al., 2023; Kim et al., 2025) and nonlinear (Pimentel et al., 2020; Sutter et al., 2026; Engels et al., 2025; Wollschläger et al., 2025; Piras et al., 2026) structure, and even where linear structure is present, it can vary with context and token position (Lampinen et al., 2026; Tan et al., 2024). Moreover, a concept being linearly decodable does not imply that the model uses that direction to produce its outputs (Belinkov, 2022; Hewitt & Liang, 2019; Tiwari et al.), which motivates finding directions by intervening on the model rather than by probing it. Our item-gradient analysis asks a complementary question: not whether a useful direction exists, but whether a single one serves every item of a task.

2.2 Steering Vectors and Representation Engineering

Activation Addition (Turner et al., 2023) introduced steering by adding a fixed vector to intermediate activations, Representation Engineering (Zou et al., 2023) formalized it across multiple concepts, and CAA (Rimsky et al., 2024) established the now-dominant extraction paradigm: the mean activation difference between contrastive prompt pairs. Though widely used, this paradigm applies only to behaviors that can be expressed as such pairs.

Several lines of work relax this requirement. Sparse-autoencoder methods (Templeton et al., 2026; Bussmann et al., 2025; Zhao et al., 2025; Cho et al., 2025) steer by scaling interpretable features, but require training an autoencoder and choosing which features to intervene on. Teacher-trace methods (Belitsky et al., 2025; Venhoff et al., 2025) extract directions from a stronger model's activations, which requires access to a teacher and assumes that the capability gap stems from model scale. Optimization-based methods treat the steering vector as a learnable parameter: ReFT (Wu et al., 2024) frames it as an extreme low-rank adaptation (Hu et al., 2021), BiPO (Cao et al., 2024) optimizes it with a preference objective, and COLD (Sharma & Trivedi, 2026) approximates the change in activations that one gradient step on the training examples would induce. These methods avoid hand-crafted contrastive pairs but require backpropagation and labeled data. Recent work has also sought to steer reasoning directly (Højer et al., 2025; Wang et al., 2026b), though what constitutes a "reasoning direction" remains underspecified.

Algorithm 1 ZOSteer: steering vector estimation from a scalar reward

Require: Model $\mathcal{M}$, layer ℓ, training and validation items, item rewards r_j, perturbation scale σ, number of perturbations N, magnitude grid $\mathcal{A}$
Ensure: Steering vector v^*, magnitude α^*, reward bank $Y \in \mathbb{R}^{N \times J}$

% Phase 1: Perturbation bank
for $i = 1, \ldots, N$ **do**
  Sample $\varepsilon_i \sim \mathcal{N}(0, \sigma^2 I_d)$
  Run $\mathcal{M}_{\varepsilon_i}$ on every training item; record $Y_{ij} \leftarrow r_j(\varepsilon_i)$
end for
% Phase 2: Direction
$y_i \leftarrow \frac{1}{J}\sum_j Y_{ij}$; $\bar{y} \leftarrow \frac{1}{N}\sum_i y_i$
$\hat{\beta} \leftarrow \frac{1}{N\sigma^2}\sum_i (y_i - \bar{y})\,\varepsilon_i$ ▷ Eq. equation 4
$v^* \leftarrow \hat{\beta}/\|\hat{\beta}\|_2$
% Phase 3: Magnitude
$\alpha^* \leftarrow \arg\max_{\alpha \in \mathcal{A}}$ validation reward of $\mathcal{M}_{\alpha\sigma\sqrt{d}\,v^*}$ ▷ Eq. equation 6
return (v^*, α^*, Y) ▷ Y and $\{\varepsilon_i\}$ are reused in Section 6.1

2.3 Zeroth-Order Optimization for Language Models

Zeroth-order (ZO) methods estimate gradients from function evaluations at perturbed points, without backpropagation. Classical instances include simultaneous perturbation stochastic approximation (Spall, 1992) and evolution strategies (Salimans et al., 2017; Wierstra et al., 2014). For LLMs, BBT (Sun et al., 2022) optimizes a soft prompt without gradients for models accessible only through an API, and MeZO (Malladi et al., 2023) shows that a two-point ZO estimator can fine-tune billion-parameter models at inference-level memory cost, with subsequent work improving its variance and efficiency (Qiu et al., 2025; Korotyshova et al., 2025; Liu et al., 2026; Guo et al., 2024; Wang et al., 2024; Tan et al., 2026). ZOSteer applies the same Gaussian-smoothing estimator once, to a d-dimensional activation offset rather than iteratively to model parameters. It therefore needs neither gradients nor labeled data, although, unlike BBT, it requires access to intermediate activations.

3 Method

3.1 Problem Setup

Let $\mathcal{M}$ be a language model and ℓ a layer. For an offset $\delta \in \mathbb{R}^d$, let $\mathcal{M}_\delta$ denote the model with δ added to the residual stream at layer ℓ at a fixed set of token positions. We are given training items $j = 1, \ldots, J$ and a scalar reward $r_j(\delta) \in \mathbb{R}$ computed from the output of $\mathcal{M}_\delta$ on item j, such as exact-match correctness or a unit-test pass. The task-level reward is the item average

$$R(\delta) = \frac{1}{J}\sum_{j=1}^{J} r_j(\delta). \tag{1}$$

We seek a unit direction v such that adding αv at a suitable magnitude α increases R on held-out items, using only forward passes of $\mathcal{M}_\delta$ and evaluations of r_j.

3.2 Zeroth-Order Gradient Estimation in Activation Space

Smoothed reward. Under greedy decoding, the output of $\mathcal{M}_\delta$ changes only when δ crosses a decision boundary, so R is piecewise constant in δ and its gradient is zero almost everywhere. We therefore target the Gaussian-smoothed reward

$$R_\sigma(\delta) = \mathbb{E}_{\varepsilon \sim \mathcal{N}(0, \sigma^2 I_d)}\big[R(\delta + \varepsilon)\big], \tag{2}$$

which is infinitely differentiable for any bounded R (Nesterov & Spokoiny, 2017). Its gradient $g = \nabla R_\sigma(0)$ is the direction in which shifting the center of the perturbation distribution most increases the expected reward.

Estimator. Differentiating Eq. equation 2 through the Gaussian density rather than through R gives Stein's identity (Stein, 1981; Nesterov & Spokoiny, 2017)

$$\nabla R_\sigma(0) = \frac{1}{\sigma^2}\,\mathbb{E}\big[\varepsilon\, R(\varepsilon)\big] = \frac{1}{\sigma^2}\,\mathbb{E}\big[\varepsilon\,(R(\varepsilon) - c)\big] \quad \text{for any constant } c, \tag{3}$$

which requires no differentiability of R; the second equality uses $\mathbb{E}[\varepsilon] = 0$. Given perturbations $\varepsilon_1, \ldots, \varepsilon_N \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2 I_d)$ and rewards $y_i = R(\varepsilon_i)$, we estimate g by the sample cross-covariance[1]

$$\hat{\beta} = \frac{1}{N\sigma^2}\sum_{i=1}^{N}(y_i - \bar{y})\,\varepsilon_i, \qquad \bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i. \tag{4}$$

Sample cost. Under isotropic perturbations, the population least-squares coefficient of y on ε is $\sigma^{-2}\mathrm{Cov}(\varepsilon, y) = g$, so $\rho^2 = \sigma^2\|g\|^2/\mathrm{Var}(y) \in [0, 1]$ is the fraction of the perturbation-induced reward variance explained by the best linear function of ε. To leading order, the expected squared cosine between $\hat{\beta}$ and g is $\mathrm{SNR}/(1 + \mathrm{SNR})$, where

$$\mathrm{SNR} = \frac{N\rho^2}{d}. \tag{5}$$

Since $\rho^2 \le 1$, close alignment ($\cos^2 \ge 1/2$) requires $N \ge d/\rho^2 \ge d$. A steering direction need not be closely aligned to be useful, however: to first order, the reward gain along $\hat{\beta}$ is proportional to its cosine with g, and in practice the gain often plateaus below $N = d$ (Section 6.3).

Monitoring convergence. We track the cosine $c_{1/2}$ between estimates computed on two disjoint halves of the perturbations. Each half has $\mathrm{SNR}_{N/2}$, so $c_{1/2} \approx \mathrm{SNR}_{N/2}/(1 + \mathrm{SNR}_{N/2})$, and since SNR is linear in N, the alignment of the full-bank estimate follows as $\mathbb{E}[\cos^2(\hat{\beta}, g)] \approx 2c_{1/2}/(1 + c_{1/2})$. Stopping at $c_{1/2} = 0.5$ thus corresponds to $\cos^2 \approx 2/3$.

Deployment. We normalize the estimate, $v^* = \hat{\beta}/\|\hat{\beta}\|_2$, and deploy the offset

$$\delta^* = \alpha^*\,\sigma\sqrt{d}\,v^*. \tag{6}$$

Since a perturbation has norm $\|\varepsilon_i\| \approx \sigma\sqrt{d}$, α is measured in units of the typical perturbation norm. We select α^* by validation reward from $\mathcal{A} = \{0.0625, 0.125, 0.25, 0.5, 0.75, 1, 1.5, 2\}$.

4 Experimental Setup

4.1 Benchmarks

We evaluate on four benchmarks with qualitatively different task structures (examples in Appendix A).

- **bAbI QA20** (Weston et al., 2015): reasoning about agents' motivations. Unlike QA18 (size reasoning), which is known to be steerable (Højer et al., 2025), QA20 mixes two question types, *where* and *why*, which lets us test whether a single direction serves both.
- **CUTE SubChar** (Edman et al., 2024): the character-substitution subtask of CUTE (e.g., "replace every `a` with `b` in `banana`"), which is notoriously difficult for subword-tokenized LLMs.
- **GSM8K** (Cobbe et al., 2021): grade-school math word problems, a common target for reasoning steering (Wang et al., 2026b; Højer et al., 2025; Wang et al., 2026a).
- **MBPP** (Austin et al., 2021): entry-level Python problems scored by unit tests, which, to our knowledge, have not previously been used to evaluate steering.

Data splits. Each benchmark is split into disjoint train (128 items), validation (128 items), and test sets. For GSM8K and MBPP, train and validation items are drawn from the official training split and the official test split is used for testing; for the others, the first 128 items form the train set, the next 128 the validation set, and the remainder the test set. The steering vector is estimated on the train set, α is selected on the validation set, and all reported numbers are test-set performance.

[1] Equivalently, $\hat{\beta}$ is, up to scale, the $\lambda \to \infty$ limit of ridge regression of y on ε, and it approaches the ordinary least-squares coefficient as $N \to \infty$.

4.2 Models and Steering Protocol

We use three 1B-scale instruction-tuned models from different families: Qwen2.5-1.5B-Instruct (Qwen et al., 2025) ($d = 1{,}536$), Llama-3.2-1B-Instruct (Grattafiori et al., 2024) ($d = 2{,}048$), and Gemma3-1B-it (Team et al., 2025) ($d = 1{,}152$). We steer at the middle layer, $\lfloor L/2 \rfloor$ for a model with L layers, adding the vector at the last two prompt tokens and all generated tokens. We sample up to $N = 16{,}384$ perturbations and stop early once the split-half cosine $c_{1/2}$ exceeds 0.5, i.e. $\cos^2(\hat{\beta}, g) \approx 2/3$ (Section 3.2). The perturbation scale is set relative to the activation norm, $\sigma\sqrt{d} = \sigma_{\text{rel}} \|h\|$, with σ_{rel} chosen per model–task pair by a sweep on a small pilot bank to maximize SNR; values are listed in Appendix B. For ZOSteer and every baseline, the direction is normalized to unit norm, deployed at scale $\alpha\sigma\sqrt{d}$ with α selected on the validation set from the grid $\mathcal{A}$ (Eq. equation 6), and evaluated with greedy decoding.

4.3 Baselines

All baselines are applied at the same layer and token positions as ZOSteer. Since none of them takes a scalar reward directly, we draw 16 samples per training prompt from the unsteered model (T=0.7, top-p=0.95), score them with the reward, and use the scored samples to build each method's contrastive or labeled data. Unlike ZOSteer, BiPO and COLD also backpropagate through the model.

- **Random**: 5 random unit directions, reported as the mean and maximum gain over directions.
- **DiffMean**: an adaptation of CAA (Rimsky et al., 2024): the difference between the mean activations of the 8 highest- and the 8 lowest-reward samples.
- **BiPO** (Cao et al., 2024): a DPO-style bidirectional preference objective on the highest- and lowest-reward sample of each prompt; prompts whose samples all receive the same reward are dropped.
- **COLD** (Sharma & Trivedi, 2026): approximates the activation change that one gradient step on the training examples would induce, either with a unit kernel (COLD-Kernel), which reduces to the negative mean gradient of the loss with respect to the steered activation, or with a finite difference between forward passes under the original and gradient-updated parameters (COLD-FD). We use both losses of the original paper: a DPO loss on the BiPO pairs and next-token cross-entropy on the chosen responses.

5 Results

Table 1 compares ZOSteer with all baselines. On GSM8K, ZOSteer gives the largest gain for every model ($+3.34$ to $+6.44$pp), while the runner-up differs across models. On bAbI QA20, most methods steer substantially: BiPO leads on Qwen and Llama, and ZOSteer leads on Gemma ($+36.20$ vs. $+23.96$ for BiPO). On CUTE SubChar, ZOSteer leads on Qwen and BiPO on Llama, while no method steers Gemma. On MBPP, no method gains more than 3pp; Section 6.4 explains why. Random directions change accuracy by at most 2pp in most cells, with one exception: on Llama QA20 the best random direction gains $+9.69$pp, so this cell is steerable even without a learned direction.

We trace the cells where ZOSteer trails BiPO to item heterogeneity. Both QA20 and SubChar contain a minority of items whose gradients conflict with the majority's (*where* questions and identity edits; Section 6), and the pooled estimate underweights them. Reweighting the two groups using the stored item gradients ($\text{ZOSteer}_{\text{Reweighted}}$), with no new forward passes, closes most of the gap on QA20 (Llama $+45.09$ vs. $+45.49$; Qwen $+23.15$ vs. $+24.50$) and overtakes BiPO on SubChar (Llama $+8.20$ vs. $+7.66$; Qwen $+7.66$ vs. $+3.63$).

6 Analysis

Table 1 leaves two questions: why a one-step estimate trails the iterative BiPO on bAbI QA20 and Llama SubChar, and why MBPP gains vanish. Sections 6.1 and 6.2 answer the first with per-item gradients, which the perturbation bank provides at no extra cost. Section 6.3 examines the perturbation budget, and Section 6.4 answers the second question.

Model	Method	bAbI QA20 ($n = 743$)	CUTE SubChar ($n = 744$)	GSM8K ($n = 1{,}319$)	MBPP ($n = 500$)
Qwen2.5-1.5B	Vanilla	54.91	4.30	67.85	45.20
	Random (mean)	−0.11	+0.32	−0.92	−0.72
	Random (max)	+0.94	+1.61	+0.08	−0.40
	DiffMean	+1.08	−0.40	−0.23	**+1.20**
	BiPO	**+24.50**	+3.63	+3.03	+0.20
	COLD-FD$_{\text{CE}}$	−0.67	−0.13	−0.83	+0.00
	COLD-FD$_{\text{DPO}}$	+20.59	+2.28	−0.76	−0.60
	COLD-Kernel$_{\text{CE}}$	+15.61	+1.48	+0.30	−0.40
	COLD-Kernel$_{\text{DPO}}$	+15.75	+0.00	+0.23	−2.00
	ZOSteer	+16.96	+5.78	**+3.41**	+0.00
	ZOSteer$_{\text{Reweighted}}$	+23.15	**+7.66**	n/a	n/a
Llama3.2-1B	Vanilla	20.19	3.63	39.80	34.80
	Random (mean)	+2.21	−0.46	−0.99	−0.80
	Random (max)	+9.69	−0.13	+0.08	−0.20
	DiffMean	+2.29	−0.13	−0.45	−2.60
	BiPO	**+45.49**	+7.66	+0.61	+0.40
	COLD-FD$_{\text{CE}}$	−0.81	+0.13	−0.91	−0.20
	COLD-FD$_{\text{DPO}}$	+30.42	+4.03	+5.84	−0.60
	COLD-Kernel$_{\text{CE}}$	+28.53	+3.49	−0.08	+0.00
	COLD-Kernel$_{\text{DPO}}$	+22.75	−0.40	−2.12	+0.00
	ZOSteer	+36.34	+4.44	**+6.44**	**+2.80**
	ZOSteer$_{\text{Reweighted}}$	+45.09	**+8.20**	n/a	n/a
Gemma3-1B	Vanilla	14.40	2.82	54.06	45.40
	Random (mean)	−0.32	+0.00	−1.00	+0.20
	Random (max)	+2.15	+0.40	−0.38	+1.40
	DiffMean	+0.27	+0.00	+0.15	**+0.40**
	BiPO	+23.96	−0.27	−0.53	−0.40
	COLD-FD$_{\text{CE}}$	+0.40	−0.13	+2.35	−0.20
	COLD-FD$_{\text{DPO}}$	+2.42	**+0.27**	+1.44	−1.40
	COLD-Kernel$_{\text{CE}}$	+2.83	+0.00	+0.30	+0.00
	COLD-Kernel$_{\text{DPO}}$	+12.79	+0.00	+2.05	+0.00
	ZOSteer	**+36.20**	**+0.27**	**+3.34**	**+0.40**
	ZOSteer$_{\text{Reweighted}}$	+27.86	n/a	n/a	+0.00

Table 1: Test-set gains (pp over Vanilla; Vanilla rows give absolute accuracy), each method at its own validation-selected magnitude. Within each model, bold marks the best and underline the second-best method, excluding Random rows. ZOSteer$_{\text{Reweighted}}$: reweighted refit (Section 6.2); n/a: no conflicting subgroup identified (Table 2).

6.1 Item Gradients and the Residual Spectrum

Item gradients. Because R is an item average (Eq. equation 1), Eq. equation 3 holds for each item separately, and the bank Y yields a gradient estimate for every item:

$$g_j = \frac{1}{N\sigma^2} \sum_{i=1}^{N} (Y_{ij} - \bar{Y}_j)\, \varepsilon_i, \qquad \hat{\beta} = \frac{1}{J} \sum_{j=1}^{J} g_j. \tag{7}$$

An item whose reward never changes across the bank has $g_j = 0$ and contributes nothing to $\hat{\beta}$; we drop such items and call the remaining $m \le J$ items *live*.

Residual spectrum. The pooled direction $b = \hat{\beta}/\|\hat{\beta}\|$ captures what the items agree on. To see what it averages away, we project it out of each item gradient, $\tilde{g}_j = (I - bb^\top)\, g_j$, stack the m live residuals as the rows of $\tilde{G} \in \mathbb{R}^{m \times d}$, and take the eigenpairs (λ_k, v_k) of $K = \tilde{G}\tilde{G}^\top$. With the unit mode direction $q_k = \tilde{G}^\top v_k / \sqrt{\lambda_k}$,

$$\lambda_k = \sum_{j=1}^{m} \left(\tilde{g}_j^\top q_k\right)^2, \tag{8}$$

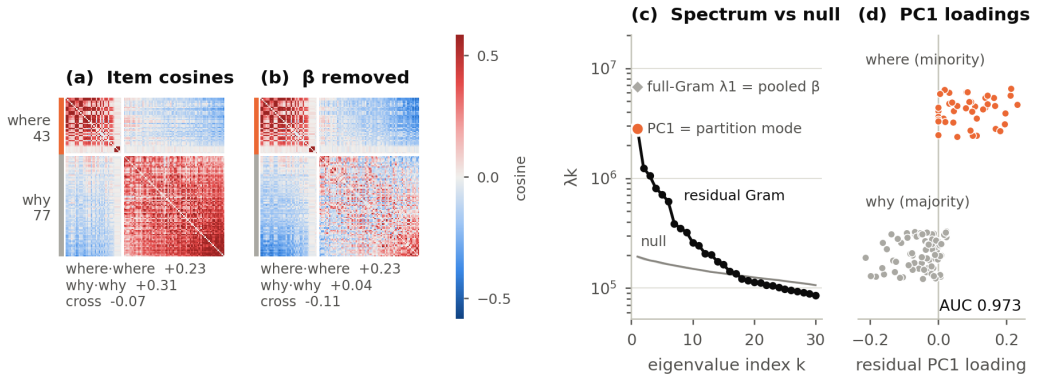


Figure 1: **Residual analysis of Llama3.2-1B on bAbI QA20.** **(a)** Cosines between item gradients g_j, grouped by question type; mean block cosines below. **(b)** The same for residuals $\tilde{g}_j$. **(c)** Eigenvalues of the residual Gram K against the shared-permutation null (median). **(d)** Item loadings on residual PC1.

so λ_k is the residual energy of all items along q_k, of which item j holds the fraction v_{kj}^2. We compare λ_k against two nulls (Appendix C). A *shared-permutation* null permutes the rows of Y, giving the spectrum that estimation noise alone would produce. A *rotation* null keeps each residual norm but randomizes its direction, so an eigenvalue above it reflects a direction shared by several items rather than one item's idiosyncrasy.

Example: Llama QA20. Figure 1 applies this to Llama QA20, whose live items split into 43 *where* and 77 *why* questions. Item gradients agree within each question type and weakly disagree across types (a). Projecting out b removes almost all of the *why* agreement ($+0.31 \rightarrow +0.04$) but none of the *where* agreement ($+0.23 \rightarrow +0.23$) (b): the pooled direction is essentially the majority's, and what the minority shares survives in the residual as its leading mode. That mode's eigenvalue lies an order of magnitude above the null (c), and its item loadings separate *where* from *why* at AUC 0.973 (d).

Across cells. Repeating the analysis on every model–task pair gives the same picture (Appendix D). The leading eigenvalues lie 2–5× above the shared-permutation null, so items genuinely prefer different directions, and only the leading few exceed the rotation null, so the shared structure is low-rank. Reading the items that load on these modes identifies a nameable subgroup in every task: *where* questions in QA20, identity edits in SubChar (the target character equals its replacement), inverse-rate wording in GSM8K, and MBPP tasks that ask for a lambda.

6.2 Conflicting Subgroups and Reweighting

A subgroup mode shows that items differ along a shared direction, not that they pull against each other. For a partition of the live items into a minority S and the rest S^c, with mean item gradients $\bar{g}_S$ and $\bar{g}_{S^c}$, we call the partition *in conflict* when

$$\cos(\bar{g}_S, \bar{g}_{S^c}) < 0. \tag{9}$$

A minority can conflict in two ways: its mean gradient points against b, or it points along b but its residual opposes the majority's strongly enough, which small minorities do most easily (Appendix E). Table 2 shows that six of the ten subgroups are in conflict: *where* questions in QA20 for all three models, identity edits in Llama and Qwen SubChar, and lambda tasks in Gemma MBPP. All six take the second route ($\cos(b, \bar{g}_S) > 0$): the pooled direction serves the minority, but only weakly. The GSM8K subgroup differs in direction from the rest, but not enough to oppose it.

Reweighted refit. For a partition in conflict, we refit the direction from the stored item gradients, with no new forward passes. With $u_S = \bar{g}_S/\|\bar{g}_S\|$ and $u_{S^c} = \bar{g}_{S^c}/\|\bar{g}_{S^c}\|$, consider the path

$$v(\gamma) = \frac{(1-\gamma)\, u_{S^c} + \gamma\, u_S}{\big\|(1-\gamma)\, u_{S^c} + \gamma\, u_S\big\|}, \qquad \gamma \in [0, 1]. \tag{10}$$

Table 2: **Conflict test and refit on every identified subgroup.** $|S|/m$: live minority / live items. Bold: in conflict (Eq. equation 9), refit at $\gamma = 1/2$. $\angle$: angle between $v(\frac{1}{2})$ and b. Last two columns: test gain (pp) of the pooled and refit directions; –: not in conflict, no refit.

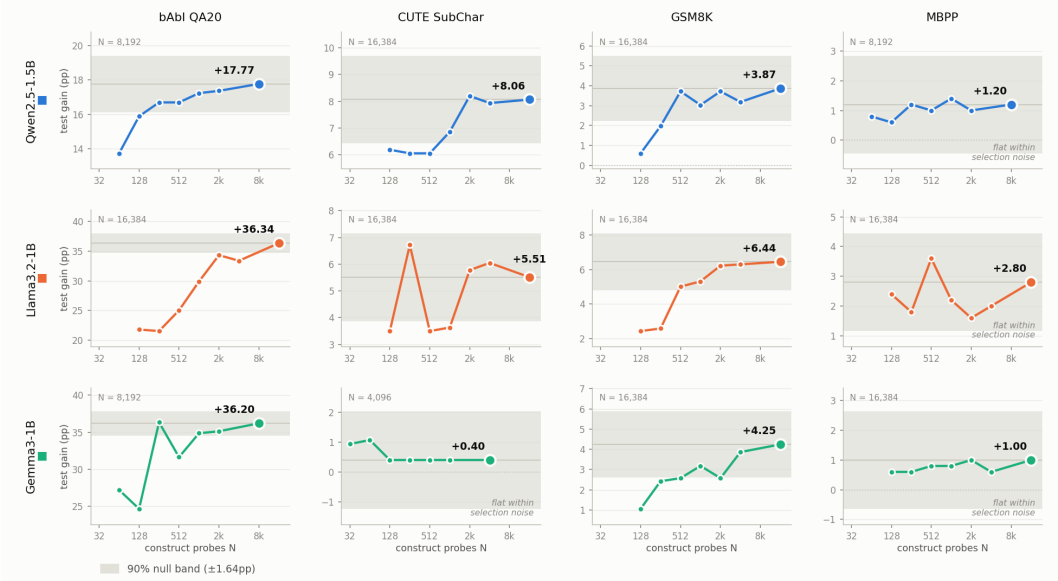


Figure 2: **Test gain vs. perturbation budget N.** Each point refits $\hat{\beta}$ on the first n perturbations of the bank, $n = N_{\text{final}}/\{128, \ldots, 4\}$, scored at its test-set best α (oracle; Appendix G); the large marker is the full bank. Band: ± 1.64pp around the full-N gain, the 90% null interval for the difference between two oracle gains; panels whose whole curve lies in it are marked flat.

The pooled direction lies on this path, $b = v(\gamma_{\text{pool}})$ with

$$\gamma_{\text{pool}} = \frac{|S|\,\|\bar{g}_S\|}{|S|\,\|\bar{g}_S\| + |S^c|\,\|\bar{g}_{S^c}\|}, \tag{11}$$

so pooling weights each group by its size times its mean-gradient norm, not by how strongly it disagrees, and a small conflicting minority can be nearly ignored (γ_{pool} as low as 0.19). We refit at $\gamma = 1/2$, weighting both groups equally, and do not tune γ.

Results. The refit improves four of the six conflict cells (Table 2), recovering most of BiPO's lead on QA20 and overtaking it on SubChar. This is consistent with the pooled one-step estimate underweighting the minority, a balance that an iterative objective like BiPO can reach over many steps. The two failures have different causes. On Gemma QA20, the premise of an underweighted minority does not hold: Gemma answers 96% of *why* questions with a location, so their rewards barely vary under perturbation, their gradients are small, and the pooled direction is already nearly balanced ($\gamma_{\text{pool}} = 0.46$). The refit then rotates b by only $6.1°$, and towards *where*, away from the questions on which Gemma has the most to gain (Appendix F). On Gemma MBPP, the refit changes nothing measurable because no MBPP vector transfers to the test set (Section 6.4).

6.3 Perturbation Budget

Section 3.2 predicts that close alignment with g requires $N \geq d$, but that the gain grows with the cosine to g, so a partially aligned direction may already suffice. Figure 2 tests this by refitting $\hat{\beta}$ on prefixes of each bank, scoring every point at its test-set best α to separate the effect of N from α selection (Appendix G). In every non-flat cell except Llama QA20, the gain enters the full-N noise band by $N = 1{,}024$, below d for all three models; Llama QA20 keeps improving up to $N = 16{,}384$. In the four flat cells (MBPP for every model, and Gemma SubChar), more perturbations do not help at all, for the reason given next.

Table 3: **Train–test gap.** Gain (pp) over Vanilla; Vanilla rows give absolute accuracy. Each vector is deployed at the α selected on its own task's validation set; MATH500 reuses the GSM8K vector and α. Train: the 128 items used for estimation. Test sizes: MBPP 500, GSM8K 1,319, MATH500 500.

		MBPP		GSM8K → MATH500		
Model	Method	Train	Test	Train	Test	MATH500
Qwen2.5-1.5B	Vanilla	54.69	45.20	86.72	67.85	46.40
	ZOSteer	+4.69	+0.00	+6.25	+3.41	+1.60
Llama3.2-1B	Vanilla	36.72	34.80	58.59	39.80	24.00
	ZOSteer	+6.25	+2.80	+5.47	+6.44	−1.40
Gemma3-1B	Vanilla	54.69	45.40	60.94	54.06	38.60
	ZOSteer	+6.25	+0.40	+7.81	+3.34	+1.00

6.4 Train–Test Gap

On MBPP, test gains are small for every model (at most +2.80pp) and flat in N (Figure 2), yet the same vectors raise the training pass rate by $+4.69$ to $+6.25$pp (Table 3): the direction is found, but it does not transfer.

More perturbations cannot close this gap. $\hat{\beta}$ estimates g_{train}, the smoothed-reward gradient averaged over the J training items, and its error relative to the population gradient g_{pop} has two parts: perturbation noise $\hat{\beta} - g_{\text{train}}$, which falls with N and is what $c_{1/2}$ and Figure 2 track, and item-sampling error $g_{\text{train}} - g_{\text{pop}}$, which does not depend on N and shrinks only with J. When items share little of their gradient, the second part dominates, and a larger bank only sharpens a direction specific to the training items.

The gap is not specific to MBPP: the GSM8K vector, which transfers within GSM8K ($+3.34$ to $+6.44$pp), gives between -1.40 and $+1.60$pp on MATH500 (Lightman et al., 2024), within sampling noise at $n = 500$. A single steering vector fit to a small item set captures what those items reward; on tasks whose items share little gradient, that is mostly the items themselves.

7 Conclusion

We showed that causally effective steering directions can be estimated from a scalar reward alone, as the zeroth-order gradient of the smoothed reward in activation space. Because the reward is an average over items, the same perturbations also give a gradient for every item, and these gradients show where a single direction falls short and why: pooling underweights a minority whose gradients conflict with the majority's, and reweighting it repairs the direction without new forward passes. Where items share little gradient, as on MBPP, more perturbations do not help, since the remaining error comes from the finite item set. On this evidence, what a single vector can steer depends on how a task's items agree, as much as on how the model represents the capability.

Limitations

Our intervention site is fixed (the middle layer, at the last two prompt tokens and all generated tokens), and we have not examined whether other layers or position sets change which capabilities are steerable. Our experiments cover only 1B-scale models and 128 training items per task, and Section 6.4 shows that the item count, not the perturbation budget, limits transfer. Subgroups are identified post hoc by reading the items that load on each residual mode, so the refit currently requires human inspection; partitioning items automatically from their loadings is a natural next step. ZOSteer requires access to intermediate activations and therefore does not apply to API-only models. Because it follows whatever the reward measures, it can also exploit flaws in the reward, for example by favoring outputs that pass a format check without exhibiting the intended behavior. Finally, it is more expensive than the baselines: the bank requires $N \times J$ generations (up to $16{,}384 \times 128 \approx 2.1$M), whereas the baselines need 16 samples per training prompt plus gradient steps, although these generations are embarrassingly parallel and need only inference memory.

AI USE STATEMENT

We used generative AI tools throughout this work. Among the tasks requiring disclosure, we used them to propose and refine research ideas and hypotheses, to help design the method and experiments, to formulate mathematical claims and assist with derivations, to implement the method and baselines, and to interpret results, including identifying item subgroups from residual modes (Section 6.1). AI-proposed ideas were checked against the literature and derivations were reviewed by the main author. Each subgroup was once named by the AI, and then double-checked by the human authors. Figures were regenerated from logged results; and every number in the text was checked against our experimental logs. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses only publicly available benchmarks and models and involves no human subjects or personal data. Steering methods are dual-use: because ZOSteer can follow any scalar reward, the same procedure could in principle be applied to an undesirable objective, such as a reward that favors bypassing a model's safety behavior. Since it requires access to intermediate activations, it applies only to open-weight models, which already admit stronger white-box interventions such as fine-tuning. A related risk is reward hacking: a vector estimated from a flawed reward inherits its flaws, so we recommend auditing steered outputs beyond the reward itself before deployment.

REPRODUCIBILITY STATEMENT

Algorithm 1 and Section 3 specify the estimator and deployment rule, and Section 4 describes the data splits, models, intervention site, and the adaptation of each baseline to the reward-only setting. Appendix B lists per-cell hyperparameters, Appendix A shows the prompt format for each benchmark, and Appendices C and G give the protocols for the residual-spectrum nulls and the perturbation-budget analysis. All models and benchmarks are publicly available, and we will release the code by publication.

REFERENCES

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems*, 37:136037–136083, 2024.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

Max Belitsky, Dawid J Kopiczko, Michael Dorkenwald, M Jehanzeb Mirza, James R Glass, Cees GM Snoek, and Yuki M Asano. Kv cache steering for controlling frozen llms. *arXiv preprint arXiv:2507.08799*, 2025.

Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with matryoshka sparse autoencoders. *arXiv preprint arXiv:2503.17547*, 2025.

Yuanpu Cao, Tianrong Zhang, Bochuan Cao, Ziyi Yin, Lu Lin, Fenglong Ma, and Jinghui Chen. Personalized steering of large language models: Versatile steering vectors through bi-directional preference optimization. *Advances in Neural Information Processing Systems*, 37:49519–49551, 2024.

Seonglae Cho, Zekun Wu, and Adriano Koshiyama. Corrsteer: Generation-time llm steering via correlated sparse autoencoder features. *arXiv preprint arXiv:2508.12535*, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Lukas Edman, Helmut Schmid, and Alexander Fraser. Cute: Measuring llms' understanding of their tokens. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 3017–3026, 2024.

Josh Engels, Eric Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. Not all language model features are one-dimensionally linear. In *International Conference on Learning Representations*, volume 2025, pp. 84591–84622, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Wentao Guo, Jikai Long, Yimeng Zeng, Zirui Liu, Xinyu Yang, Yide Ran, Jacob R Gardner, Osbert Bastani, Christopher De Sa, Xiaodong Yu, et al. Zeroth-order fine-tuning of llms with extreme sparsity. *arXiv preprint arXiv:2406.02913*, 2024.

Wes Gurnee and Max Tegmark. Language models represent space and time. In *International Conference on Learning Representations*, volume 2024, pp. 2483–2503, 2024.

Evan Hernandez, Arnab Sen Sharma, Tal Haklay, Kevin Meng, Martin Wattenberg, Jacob Andreas, Yonatan Belinkov, and David Bau. Linearity of relation decoding in transformer language models. In *International Conference on Learning Representations*, volume 2024, pp. 10504–10526, 2024.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)*, pp. 2733–2743, 2019.

Bertram Højer, Oliver Jarvis, and Stefan Heinrich. Improving reasoning performance in large language models via representation engineering. In *International Conference on Learning Representations*, volume 2025, pp. 44746–44763, 2025.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Ole Jorgensen, Dylan Cope, Nandi Schoots, and Murray Shanahan. Improving activation steering in language models with mean-centring. *arXiv preprint arXiv:2312.03813*, 2023.

Junsol Kim, James Evans, and Aaron Schein. Linear representations of political perspective emerge in large language models. In *International Conference on Learning Representations*, volume 2025, pp. 7180–7211, 2025.

Daria Korotyshova, Boris Shaposhnikov, Alexey Malakhov, Alexey Khokhulin, Nikita Surnachev, Kirill Ovcharenko, George Bredis, Alexey Gorbatovski, Viacheslav Sinii, and Daniil Gavrilov. Essa: Evolutionary strategies for scalable alignment. *arXiv preprint arXiv:2507.04453*, 2025.

Andrew Kyle Lampinen, Yuxuan Li, Eghbal Hosseini, Sangnie Bhardwaj, and Murray Shanahan. Linear representations in language models can change dramatically over a conversation. *arXiv preprint arXiv:2601.20834*, 2026.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Yong Liu, Zirui Zhu, Chaoyu Gong, Minhao Cheng, Cho-Jui Hsieh, and Yang You. Sparse mezo: Less parameters for better performance in zeroth-order llm fine-tuning. *Advances in Neural Information Processing Systems*, 38:21139–21169, 2026.

Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D Lee, Danqi Chen, and Sanjeev Arora. Fine-tuning language models with just forward passes. *Advances in Neural Information Processing Systems*, 36:53038–53075, 2023.

Samuel Marks and Max Tegmark. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *arXiv preprint arXiv:2310.06824*, 2023.

Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent linear representations in world models of self-supervised sequence models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 16–30, 2023.

Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17(2):527–566, 2017.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *arXiv preprint arXiv:2311.03658*, 2023.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. Information-theoretic probing for linguistic structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4609–4622, 2020.

Giorgio Piras, Raffaele Mura, Fabio Brau, Luca Oneto, Fabio Roli, and Battista Biggio. Som directions are better than one: multi-directional refusal suppression in language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 32728–32736, 2026.

Xin Qiu, Yulu Gan, Conor F Hayes, Qiyao Liang, Yinggan Xu, Roberto Dailey, Elliot Meyerson, Babak Hodjat, and Risto Miikkulainen. Evolution strategies at scale: Llm fine-tuning beyond reinforcement learning. *arXiv preprint arXiv:2509.24372*, 2025.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL `https://arxiv.org/abs/2412.15115`.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15504–15522, 2024.

Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

Kartik Sharma and Rakshit Trivedi. Cold-steer: Steering large language models via in-context one-step learning dynamics. In *International Conference on Learning Representations*, volume 2026, pp. 27736–27756, 2026.

James C Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE transactions on automatic control*, 37(3):332–341, 1992.

Charles M Stein. Estimation of the mean of a multivariate normal distribution. *The annals of Statistics*, pp. 1135–1151, 1981.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. Black-box tuning for language-model-as-a-service. In *International Conference on Machine Learning*, pp. 20841–20855. PMLR, 2022.

Denis Sutter, Julian Minder, Thomas Hofmann, and Tiago Pimentel. The non-linear representation dilemma: Is causal abstraction enough for mechanistic interpretability? *Advances in Neural Information Processing Systems*, 38:149455–149503, 2026.

Daniel Tan, David Chanin, Aengus Lynch, Brooks Paige, Dimitrios Kanoulas, Adrià Garriga-Alonso, and Robert Kirk. Analysing the generalisation and reliability of steering vectors. *Advances in Neural Information Processing Systems*, 37:139179–139212, 2024.

Qitao Tan, Jun Liu, Zheng Zhan, Caiwen Ding, Yanzhi Wang, Xiaolong Ma, Jaewoo Lee, Jin Lu, and Geng Yuan. Harmony in divergence: Towards fast, accurate, and memory-efficient zeroth-order llm fine-tuning. *Advances in Neural Information Processing Systems*, 38:175003–175037, 2026.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, et al. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *arXiv preprint arXiv:2605.29358*, 2026.

Curt Tigges, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. Linear representations of sentiment in large language models. *arXiv preprint arXiv:2310.15154*, 2023.

Devesh Tiwari, Camille Davis, Shivank Sinha, Talia Weaver, Aditya Shah, and Maheep Chaudhary. Decodability is not causality: Dissociating probe readouts from behavioral drivers via sae decomposition. In *Actionable Interpretability Workshop at COLM 2026*.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. Understanding reasoning in thinking language models via steering vectors. *arXiv preprint arXiv:2506.18167*, 2025.

Fei Wang, Li Shen, Liang Ding, Chao Xue, Ye Liu, and Changxing Ding. Simultaneous computation and memory efficient zeroth-order optimizer for fine-tuning large language models. *arXiv preprint arXiv:2410.09823*, 2024.

Tianlong Wang, Yuhang Wang, Weibin Liao, Xin Gao, Xinyu Ma, Yang Lin, Yasha Wang, and Liantao Ma. Search for truth from reasoning: A dynamic representation editing framework for steering llm trajectories. *arXiv preprint arXiv:2606.28589*, 2026a.

Zijian Wang, Yanxiang Ma, and Chang Xu. Eliciting chain-of-thought in base llms via gradient-based representation optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 33782–33790, 2026b.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart Van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*, 2015.

Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. Natural evolution strategies. *The Journal of Machine Learning Research*, 15(1):949–980, 2014.

Tom Wollschläger, Jannes Elstner, Simon Geisler, Vincent Cohen-Addad, Stephan Günnemann, and Johannes Gasteiger. The geometry of refusal in large language models: Concept cones and representational independence. *arXiv preprint arXiv:2502.17420*, 2025.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D Manning, and Christopher Potts. Reft: Representation finetuning for language models. *Advances in Neural Information Processing Systems*, 37:63908–63962, 2024.

Yu Zhao, Alessio Devoto, Giwon Hong, Xiaotang Du, Aryo Pradipta Gema, Hongru Wang, Xuanli He, Kam-Fai Wong, and Pasquale Minervini. Steering knowledge selection behaviours in llms via sae-based representation engineering. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5117–5136, 2025.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023.

A BENCHMARK EXAMPLES

A.1 BABI QA20

bAbI QA20 (gold: `tired`)

```
Yann is thirsty. Yann journeyed to the kitchen. Sumit is thirsty. Yann picked up the milk
there. Antoine is tired. Jason is tired. Sumit went back to the kitchen. Antoine went back
to the bedroom. Antoine took the pajamas there.

Question: Why did antoine get the pajamas?
```

A.2 CUTE SUBCHAR

CUTE SubChar (gold: `enl`)

```
Substitute "d" with "l" in "end".
Answer: "
```

A.3 GSM8K

GSM8K (gold: `18`)

```
Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes
muffins for her friends every day with four. She sells the remainder at the farmers' market
 daily for \$2 per fresh duck egg. How much in dollars does she make every day at the
farmers' market?
```

A.4 MBPP

MBPP (reward: all three tests pass)

```
Write a function to sort a given matrix in ascending order according to the sum of its rows
.

Your code should pass these tests:
assert sort_matrix([[1, 2, 3], [2, 4, 5], [1, 1, 1]])==[[1, 1, 1], [1, 2, 3], [2, 4, 5]]
assert sort_matrix([[1, 2, 3], [-2, 4, -5], [1, -1, 1]])==[[-2, 4, -5], [1, -1, 1], [1, 2,
3]]
assert sort_matrix([[5,8,9],[6,4,3],[2,1,4]])==[[2, 1, 4], [6, 4, 3], [5, 8, 9]]
```

B HYPERPARAMETERS

Table 4: **Hyperparameters per model–task pair.** Every model is steered at its middle layer $\ell = \lfloor L/2 \rfloor$ (Section 4.2). $\|h\|$: mean ℓ_2 norm of the residual stream at layer ℓ over the last two prompt tokens of the training items, i.e. the positions where perturbations are injected; σ_{rel}: perturbation norm relative to $\|h\|$, selected by the pilot sweep; $\sigma\sqrt{d} = \sigma_{\mathrm{rel}}\,\|h\|$: perturbation norm, which is also the unit of the deploy magnitude (Eq. equation 6); α^*: validation-selected magnitude.

Model	Task	$\|h\|$	σ_{rel}	$\sigma\sqrt{d}$	α^*
Qwen2.5-1.5B ($\ell = 14$)	bAbI QA20	49.75	0.32	15.92	0.125
	CUTE SubChar	48.33	0.64	30.93	0.125
	GSM8K	52.51	0.16	8.401	0.125
	MBPP	55.02	0.16	8.803	0.25
Llama-3.2-1B ($\ell = 8$)	bAbI QA20	5.123	1.28	6.558	0.25
	CUTE SubChar	4.993	1.28	6.391	0.125
	GSM8K	5.091	0.32	1.629	0.25
	MBPP	5.284	0.64	3.382	0.0625
Gemma-3-1B ($\ell = 13$)	bAbI QA20	5429	0.16	868.7	0.25
	CUTE SubChar	5879	0.04	235.2	0.0625
	GSM8K	6354	0.01	63.54	0.5
	MBPP	5625	0.04	225.0	0.0625

C NULL MODELS FOR THE RESIDUAL SPECTRUM

The *shared-permutation* null permutes the rows of Y jointly across items and reruns the full pipeline (item gradients, pooled direction b, residuals) on the permuted rewards. This removes the dependence of rewards on ε while preserving each item's reward distribution and the covariance of rewards across items, so it gives the spectrum that estimation noise alone would produce. The *rotation* null accounts for heterogeneity in item-gradient norms: it keeps each residual norm $\|\tilde{g}_j\|$ but redraws its direction isotropically in the orthogonal complement of b, subject to the constraint $\sum_j \tilde{g}_j = 0$ that the observed residuals satisfy. It thus preserves every item's residual energy, and hence the trace, but removes any shared direction beyond what the constraint induces; its top eigenvalue is close to the largest single item's residual energy. An eigenvalue above the shared-permutation null is structure beyond estimation noise; one above the rotation null is more than any single item can supply, i.e. a direction that several items share. The two nulls are not nested: the shared-permutation null is the stricter reference for the leading eigenvalue, and the rotation null for the deeper ones.

D RESIDUAL SPECTRA ACROSS MODELS AND TASKS

Figure 3 shows the residual spectrum for every model–task pair. The leading modes are of two kinds. *Subgroup modes* (orange) separate a nameable subset of items, identified post hoc by reading the items that load on each mode (Table 2). *Single-item modes* (green) put most of their energy on one item and lose their direction when that item is removed; we treat them as item idiosyncrasy. For Qwen GSM8K and MBPP, the leading modes exceed the rotation null, but we found no subgroup that explains them.

E TWO ROUTES TO CONFLICT

Decompose $\bar{g}_S = a_S\, b + r_S$ with $a_S = \bar{g}_S^\top b$ and $r_S \perp b$, and likewise for S^c. The projections are tied, $|S|\, a_S + |S^c|\, a_{S^c} = \|\sum_j g_j\| > 0$, so $a_S \le 0$ forces $a_{S^c} > 0$. The residuals cancel, $|S|\, r_S + |S^c|\, r_{S^c} = 0$, which gives

$$\bar{g}_S^\top \bar{g}_{S^c} = a_S\, a_{S^c} - \frac{|S|}{|S^c|}\,\|r_S\|^2 = a_S\, a_{S^c} - \frac{|S^c|}{|S|}\,\|r_{S^c}\|^2. \tag{12}$$

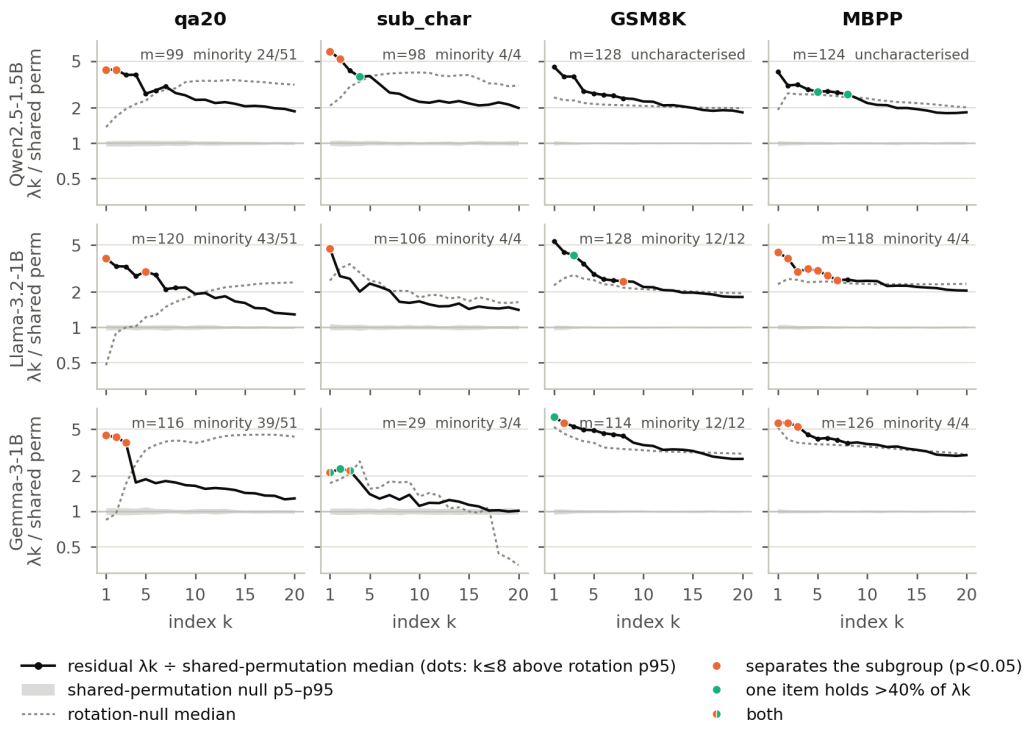


Figure 3: **Residual spectra across models and tasks.** m: live items; "minority a/b": a of the b members of S among the 128 training items are live. Orange: subgroup modes (Table 2); green: single-item modes.

Conflict thus arises by one of two routes. In the *projection* route, $a_S \leq 0$: the minority's mean gradient points against b, and Eq. equation 9 follows immediately. In the *residual* route, $a_S > 0$, and the residual term must outweigh the alignment term. By the second form of Eq. equation 12, for a fixed majority residual this term grows as $|S|$ shrinks, so small minorities conflict mainly through the residual route.

F Why the Refit Fails on Gemma QA20

Unsteered Gemma answers 96% of *why* questions with a location, and its *why* accuracy is 0%. Under the perturbations it stays near zero (mean train accuracy 0.039, vs. 0.335 for *where*), so *why* rewards barely vary. Applying the sample-cost argument of Section 3.2 per item gives $\|g_j\| = \rho_j \,\mathrm{sd}_i(Y_{ij})/\sigma$, where ρ_j^2 is the fraction of item j's reward variance explained linearly by ε, so low variance means small gradients. A *why* gradient is on average $0.43\times$ a *where* gradient, and the variance ratio alone predicts $0.55\times$.

Table 5 traces how this propagates to γ_{pool}. The ratio of summed group gradients factors exactly as

$$\frac{\|\sum_{j\in\text{why}} g_j\|}{\|\sum_{j\in\text{where}} g_j\|} = \underbrace{\frac{n_{\text{why}}}{n_{\text{where}}}}_{\text{count}} \times \underbrace{\frac{\overline{\|g\|}_{\text{why}}}{\overline{\|g\|}_{\text{where}}}}_{\text{per-item size}} \times \underbrace{\frac{\kappa_{\text{why}}}{\kappa_{\text{where}}}}_{\text{coherence}}, \qquad \kappa_S = \frac{\|\sum_{j\in S} g_j\|}{\sum_{j\in S}\|g_j\|}, \tag{13}$$

where $\overline{\|g\|}_S$ is the mean item-gradient norm in S, and $\gamma_{\text{pool}} = 1/(1+\text{this ratio})$ with $S = \textit{where}$. In Llama and Qwen the count advantage of *why* survives, and γ_{pool} is small. In Gemma, the per-item size factor (0.43) cancels the count factor (1.97), and γ_{pool} approaches $1/2$. The refit at $\gamma = 1/2$ therefore rotates b towards *where*, away from the questions on which Gemma has the most to gain.

Rotating the other way helps: selecting γ on the validation set gives $\gamma = 0.9$ and a test gain of $+38.36$pp, above the pooled $+36.20$pp.

Table 5: **From greedy behavior to γ_{pool} on bAbI QA20.** Pairs are *why* / *where*.

	Llama	Qwen	Gemma
why answered with a location (test, unsteered)	0.13	0.30	0.96
Accuracy (test, unsteered)	14.7 / 34.2	56.8 / 51.6	0.0 / 38.9
Mean accuracy under perturbations $\bar{p}$ (train)	0.159 / 0.075	0.584 / 0.713	0.039 / 0.335
Reward variance $\bar{p}(1-\bar{p})$	0.124 / 0.066	0.082 / 0.096	0.035 / 0.117
Count ratio $n_{\text{why}}/n_{\text{where}}$	77/43 = 1.79	75/24 = 3.13	77/39 = 1.97
$\times$ per-item size ratio	1.63	0.85	0.43
$\times$ coherence ratio	0.99	1.62	1.37
$=$ summed-gradient ratio	2.89	4.32	1.18
γ_{pool}	0.257	0.188	0.459
$\cos(b, \bar{g}_{\text{where}})$	0.079	0.053	0.522

G Protocol for the Perturbation-Budget Ladder

Perturbations are drawn i.i.d. and stored in sampling order, so a prefix of length n is a genuine $N = n$ experiment rather than a subsample. Every point, including the full-N reference, is scored at the $\alpha \in \mathcal{A}$ that maximizes test reward. We use this oracle deliberately: a validation-selected α adds a validation–test mismatch that varies from rung to rung and would blur the effect of N. The ladder is therefore a diagnostic for comparing budgets, not a deployable result; all deployable numbers (Tables 1 and 2) use validation-selected α. Holding α fixed across rungs would instead confound the effect of N with dose mis-scaling: writing $\hat{\beta}_n = \cos_n\, g/\|g\| + \sin_n\, e_\perp$ in unit form, a fixed α under-doses the on-target component by a factor $\cos_n$ as n falls. Non-monotone stretches in Figure 2 (Llama SubChar, Gemma QA20) therefore reflect the prefix estimates themselves, not α selection.

The null band is the 90th percentile of the absolute difference in best-over-α gain between two signal-free, norm-matched random directions in the same cell, each at its test-set best α, giving a half-width of 1.64pp. Since the ladder points and the null select α the same way, this is the appropriate scale for comparing rungs.