# BRIDGE: BRANCH-GUIDED RESIDUAL INTERVENTIONS VIA DYNAMIC GEOMETRY

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Representation intervention controls the behavior of large language models (LLMs) at inference time without updating parameters. Existing methods typically choose intermediate or late layers empirically as the source of intervention signals. However, we find that early-layer signals in fact achieve higher coverage of correct answers, but many irrelevant signals hinder direct use for effective intervention. To address this challenge, we propose *BRIDGE* (*Branch-guided Residual Interventions via Dynamic GEometry*). Unlike prior methods, BRIDGE extracts intervention signals by identifying the useful early branches and preserves general capabilities through a minimal, norm-preserving edit. Specifically, Early Branch Identification (EBI) projects an early-layer hidden state through the output head to select first-token anchors, extends them into short natural branches, and learns coverage-based utilities to identify the branches worth strengthening. Dynamics-aware Geometric Editing (DGE) then specifies a target utility gain, solves for the smallest norm-preserving residual edit that achieves it, and keeps the edit only when a forward pass confirms a positive realized gain. Across five Llama and Qwen models from 3B to 32B, BRIDGE consistently improves truthfulness, fairness, and safety alignment while preserving general capabilities, showing that properly identified early branches provide effective intervention signals for controllable LLM generation. Our code is available at `https://anonymous.4open.science/r/Bridge-1FC8`.

## 1 INTRODUCTION

Large language models (LLMs) now support question answering, knowledge-intensive retrieval, and open-ended generation (Brown et al., 2020; Lewis et al., 2020), yet ensuring that their behavior meets desired properties such as truthfulness, fairness, and safety remains an open challenge (Lin et al., 2022; Parrish et al., 2022; Bai et al., 2022). Preference-based fine-tuning updates model weights with human feedback or preference data (Christiano et al., 2017; Ziegler et al., 2019; Rafailov et al., 2023), but it requires additional training for each target behavior and can degrade unrelated capabilities (Ouyang et al., 2022). Prompt-based methods avoid training, yet their effect depends on the wording of each task or query. Representation intervention combines the strengths of both: it modifies internal representations at inference time, requiring neither weight updates nor per-query prompt design (Turner et al., 2023; Rimsky et al., 2024).

Most representation-intervention methods select the source layer by grid search over intermediate and late layers (Li et al., 2023; Rimsky et al., 2024; Hoscilowicz et al., 2024; Yang et al., 2026). In these layers, the model has progressively sharpened its prediction toward a single output (Lad et al., 2025), so target behaviors become linearly separable and intervention signals are easy to extract. However, this raises a question: are later layers necessarily the best source of intervention signals?

To answer this question, we divide the layers of a model evenly into early, middle, and late groups (e.g., layers 1–11, 12–22, and 23–32 for the 32-layer Llama-3.1-8B) and compare branch representations taken from one representative layer in each group (Appendix C.1). As shown in Figure 1, early-layer representations retain a broad, multimodal structure that covers 85.92% of correct answers; in later layers, the representations contract toward a single mode, and coverage drops to 58.12% in middle layers and 34.34% in late layers. This suggests that layer-wise convergence makes intervention signals easier to extract at the cost of lower coverage of correct answers, consistent with



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.

the finding that correct predictions available at earlier layers can be overturned in later layers (Halawi et al., 2024). A natural approach is thus to extract intervention signals directly from raw early-layer representations. However, our experiments in §4.3 (Figure 2) show that raw early-layer signals underperform middle- and late-layer signals. This is because early-layer representations, despite their higher coverage, also contain many peaks unrelated to the correct answers (the blue peaks in Figure 1(a)), which makes it difficult to extract a clean and effective signal. This motivates us to study how to extract the useful part of early-layer signals without introducing irrelevant signals.

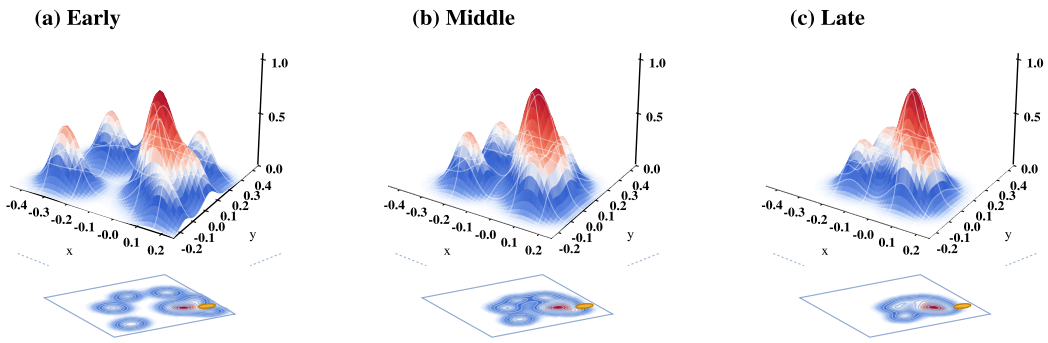



Figure 1: **Branch geometry across layers.** On Qwen3-14B, early layers retain a broader multi-modal structure, while later representations contract toward fewer dominant modes.

To address this challenge, we propose BRIDGE (Branch-guided Residual Interventions via Dynamic GEometry). Rather than extracting a direction directly from early representations, we unfold the early signal into branches: short continuations that the model itself generates from its most probable first tokens. Each branch is a concrete candidate, so useful and irrelevant candidates, which remain entangled in a raw early representation, can be separated and scored individually. Early Branch Identification (EBI) projects an early-layer hidden state through the output head to select high-probability first-token anchors, extends each anchor into a short natural branch, and learns coverage-based utilities that indicate how much each branch should be strengthened. Dynamics-aware Geometric Editing (DGE) turns the utilities into a differentiable objective over the hidden state, specifies the desired utility gain first, and solves for the smallest norm-preserving edit that achieves it across candidate layers under a shared residual budget. DGE keeps the edit only when a forward pass confirms a positive realized gain, preserving general capabilities while strengthening the useful branches.

We evaluate BRIDGE on models of different sizes and families, using benchmarks that cover both multiple-choice evaluation (TruthfulQA, BBQ) and open-ended generation (Sorry-Bench), together with general-capability benchmarks (MMLU, ARC). Our experimental results reveal that BRIDGE effectively identifies and extracts useful intervention signals from the broad but mixed early-layer signals, and realizes them through minimal intervention with DGE, achieving better alignment while preserving general capabilities. Our main contributions are:

- We show that early-layer representations cover more correct answers than middle- and late-layer representations, but also contain more signals unrelated to the correct answers, which makes raw early-layer signals difficult to use for intervention.
- We propose BRIDGE, which unfolds early signals into the model's own branches, identifies the useful ones through coverage-based utilities (EBI), and realizes their reweighting through a minimal, norm-preserving, forward-verified edit (DGE).
- Extensive experiments across model families and scales show that BRIDGE achieves the best alignment performance among all compared methods while maintaining or improving general capabilities.

## 2 Related Work

**Sources of intervention signals.** Classic representation-intervention methods extract a global direction from externally constructed behavioral contrasts, such as truthful versus untruthful or

instruction-following versus non-following responses (Li et al., 2023; Rimsky et al., 2024; Turner et al., 2023; Stolfo et al., 2025); representation engineering reads concept directions from contrastive stimuli in the same way (Zou et al., 2023). Later work makes the signal more input-dependent, drawing it from prompt- or demonstration-induced latents and learning dynamics (Liu et al., 2024; Heyman & Vandeputte, 2026; Sharma & Trivedi, 2026), diverse generation paths (Zhu et al., 2026), and systematic choices of source layer and intervention form (Ye et al., 2026; Zhao et al., 2026). Most of these methods read intermediate or late layers (Hoscilowicz et al., 2024). Layer-wise analyses, however, show that later layers progressively sharpen the model's prediction toward a single output, and that correct predictions decodable at intermediate layers can be overturned later (Halawi et al., 2024; Lad et al., 2025; Chuang et al., 2024). BRIDGE instead reads the model's natural branches at an early layer and uses coverage supervision to identify those worth strengthening, extracting better intervention signals before the candidate answers converge.

**Minimal and selective interventions.** Precise intervention changes the target behavior while leaving unrelated capabilities intact. Prior methods pursue this goal through norm-preserving rotations (Vu & Nguyen, 2025; You et al., 2026), calibrated local or position-wise edits (Yu et al., 2026b; Yuan et al., 2026), attention-level interventions (Kang et al., 2026), energy calibration (Yu et al., 2026a), and explicit objectives that protect non-target behavior (Nguyen et al., 2026). Most of these methods fix an intervention direction first and then decide how strongly, where, or when to apply it. BRIDGE instead starts from the desired utility gain, solves for the smallest norm-preserving residual edit that achieves it, and verifies the realized gain with a forward pass before applying the edit.

# 3 METHOD

In this section, we present **BRIDGE**, which extracts prompt-specific intervention signals from early-layer natural branches and realizes them at candidate intervention layers. EBI (§3.1) uses marginal coverage over correct and incorrect candidates to learn branch utilities, while DGE (§3.2) converts these utilities into a local minimum-norm, norm-preserving spherical edit and retains it only when forward verification confirms a positive realized gain. We provide a theoretical analysis of this realized gain in §3.3.

## 3.1 EARLY BRANCH IDENTIFICATION

The preliminary results in Figure 1 show that early-layer representations retain broader branch structure and higher coverage of correct candidates, while also containing irrelevant signals. We address this tension by constructing prompt-relative representations of natural generation branches and identifying useful early signals through coverage-based utility supervision.

**Branch generation and representation.** Given a prompt $x$, we first select an early layer $\ell_{\mathrm{e}}$ (e.g., layer 4). We project the hidden state at the final prompt position in this layer through the frozen model's output head to obtain an early next-token distribution. We select the $K$ highest-probability valid first tokens from this distribution and keep them fixed as branch anchors. We then greedily extend each anchor under the frozen model to a total length of $T$ tokens, forming the natural branch set $\mathcal{B} = \{b_1, \ldots, b_K\}$. To characterize how each continuation differs from the shared prompt, we use the same early layer $\ell_{\mathrm{e}}$ and define the branch representation as the mean offset:

$$d_b = \frac{1}{T} \sum_{t=1}^{T} \left( h_{b,t}^{(\ell_{\mathrm{e}})} - h^{(\ell_{\mathrm{e}})} \right). \tag{1}$$

Here, $h^{(\ell_{\mathrm{e}})}$ is the hidden state at the final prompt position in this layer, and $h_{b,t}^{(\ell_{\mathrm{e}})}$ is the hidden state at the $t$-th continuation token of branch $b$.

To assess which branches should receive more support, we also need to characterize the model's existing relative preference over these candidates. We therefore compute each branch's length-normalized log-likelihood, $\ell_b^{\mathrm{nat}} = T^{-1} \log P_\theta(b \mid x)$. With temperature $\tau > 0$, the reference weights are $\pi_b = \exp(\ell_b^{\mathrm{nat}}/\tau) / \sum_j \exp(\ell_j^{\mathrm{nat}}/\tau)$. Subsequent coverage supervision uses this reference distribution as the baseline for measuring each branch's marginal value under the model's current preference.

Coverage will be computed from distances between branch and candidate-answer representations, so we next construct a low-dimensional coordinate space for comparing them. Specifically, we fit weighted PCA to $d_b$ on the PCA/coverage half of the training split using $\pi_b$ as sample weights (Jolliffe & Cadima, 2016), so that the coordinate fit reflects the model's natural branch distribution. PCA retains the principal directions of variation, and whitening then equalizes their variance scales to prevent a few high-variance directions from dominating the distance. The resulting branch coordinates are $z_b = \Lambda^{-1/2} U^\top (d_b - \mu)$, where $\mu$ is the weighted mean, $U$ contains the retained principal directions, and $\Lambda$ is the diagonal matrix of their eigenvalues. The fitted transform is frozen and also applied to candidate answers during training, so that coverage is computed in a shared coordinate space. Thus, $d_b$ denotes the branch representation, $z_b$ its PCA coordinate, and $c_a$ the coordinate of candidate answer $a$ under the same frozen PCA transform.

**Coverage supervision.** In this shared coordinate space, we partition the candidate answers from the disjoint scorer-training half into correct and incorrect sets, $\mathcal{A}^+$ and $\mathcal{A}^-$, according to their labels. To convert the geometric distance between a branch and a candidate answer into a continuous coverage weight, we use the Gaussian kernel $\kappa_{ba} = \exp[-\|z_b - c_a\|_2^2/(2\sigma^2)]$. Here, $\sigma$ controls the distance scale over which the weight decays. A branch receives a larger coverage weight when it is closer to the candidate answer. We then combine these weights with the natural reference distribution $\pi$ to compute each branch's coverage contribution to correct and incorrect candidates. The task-specific supervision used in the experiments is described in Appendix A.1.

The weight $\kappa_{ba}$ measures the match between an individual branch and a candidate answer. A useful branch should be close to under-covered correct candidates and should not receive additional mass merely because it is close to incorrect candidates. Let $D_a(\pi) = \epsilon + \sum_j \pi_j \kappa_{ja}$ denote the smoothed coverage of candidate $a$ under the natural reference distribution, where $\epsilon > 0$ provides numerical stability. We define the coverage objective as

$$\mathcal{C}(\pi) = |\mathcal{A}^+|^{-1} \sum_{a \in \mathcal{A}^+} \log D_a(\pi) - \lambda_{\text{wrong}} |\mathcal{A}^-|^{-1} \sum_{a \in \mathcal{A}^-} \log D_a(\pi). \tag{2}$$

We define the branch utility as its marginal gain with respect to the branch weight:

$$y_b = \frac{\partial \mathcal{C}(\pi)}{\partial \pi_b} = \frac{1}{|\mathcal{A}^+|} \sum_{a \in \mathcal{A}^+} \frac{\kappa_{ba}}{D_a(\pi)} - \lambda_{\text{wrong}} \frac{1}{|\mathcal{A}^-|} \sum_{a \in \mathcal{A}^-} \frac{\kappa_{ba}}{D_a(\pi)}. \tag{3}$$

The first term rewards branches that cover correct candidates, especially those that are not already well covered. The second term penalizes coverage of incorrect candidates. We use the derivative in Equation 3 as the branch utility; after centering it under $\pi$ as $\tilde{y}_b = y_b - \sum_j \pi_j y_j$, the resulting relative target is used to train the branch scorer. The coverage utility uses answer labels and is therefore available only during training; at inference time, we predict it from the branch representation and the model's original branch probabilities using a lightweight MLP:

$$x_b = [z_b, \bar{z}, z_b - \bar{z}, \log(\pi_b + \epsilon)], \qquad \bar{z} = \sum_j \pi_j z_j, \qquad u_b = f_\phi(x_b). \tag{4}$$

We center the predicted scores under the original branch distribution and train the scorer with a smooth-$L_1$ regression loss together with a pairwise ranking loss.

$$\mathcal{L} = \mathcal{L}_{\text{Huber}}(\tilde{u}, \tilde{y}) + \lambda_{\text{rank}} \mathcal{L}_{\text{rank}}. \tag{5}$$

After training, the scorer is frozen. We standardize its centered outputs and use them as the branch utilities for hidden-state intervention.

### 3.2 Dynamics-aware Geometric Editing

Once the branch utilities are available, we use them to modify the hidden state. The goal is to increase probability on high-utility branches while keeping the update small and preserving the norm of the original state.

**Utility objective.** We first construct an objective over the hidden state that increases when the model shifts relative weight from lower-scoring to higher-scoring branches. For a candidate intervention layer $r$, let $h_r$ denote the hidden state at the final prompt position. We hold each branch's token sequence and score $\hat{u}_b$ fixed, and express its length-normalized conditional log-likelihood as a function $\ell_b(h_r)$ of this state. Using the same temperature normalization, we compute $p_b(h_r) = \exp(\ell_b(h_r)/\tau)/\sum_j \exp(\ell_j(h_r)/\tau)$ and define the objective as the weighted average of the branch scores:

$$U_r(h_r) = \sum_b p_b(h_r)\hat{u}_b. \tag{6}$$

The scores $\hat{u}_b$ remain fixed in this computation, so changes in $U_r$ arise from the model's allocation of probability across branches. We can therefore differentiate through the branch likelihoods with respect to the hidden state to obtain $\nabla_{h_r} U_r$, which gives a local direction for increasing the average score and provides the basis for the subsequent norm-preserving update.

For the unedited model, $p_b(h_r) = \pi_b$. Since the standardized utilities satisfy $\sum_b \pi_b \hat{u}_b = 0$, the objective is zero before editing. Thus, $\Delta$ specifies a standardized utility increment from a common zero baseline.

**Minimum-norm tangent update.** A direct additive update along this gradient can change both the direction and scale of the hidden state. We aim to change branch preference by adjusting the state's direction while retaining its original scale, and therefore constrain the updated state to have the same norm as the current state (Absil et al., 2008; Vu & Nguyen, 2025; You et al., 2026). The feasible set is $\mathcal{S}_r = \{z : \|z\|_2 = \|h_r\|_2\}$, a sphere centered at the origin with radius $\|h_r\|_2$. Around the current state $h_r$, a displacement that preserves the norm to first order must satisfy $h_r^\top \xi = 0$, meaning that it is orthogonal to $h_r$. We therefore remove the gradient's component along $h_r$ to obtain the tangent gradient $\bar{g}_r = (I - h_r h_r^\top / \|h_r\|_2^2)\nabla_{h_r} U_r$.

For a tangent displacement $\xi$, the utility gain under the first-order approximation is $\bar{g}_r^\top \xi$. To compare candidate layers under the same improvement requirement, we specify a common target gain $\Delta > 0$ and seek the smallest displacement that meets it. When $\bar{g}_r \neq 0$, moving along the tangent gradient gives the largest predicted gain per unit displacement. The constrained problem and its solution are

$$\begin{aligned} \xi_r^\star &= \arg\min_{\xi \in T_{h_r}\mathcal{S}_r} \frac{1}{2}\|\xi\|_2^2 \quad \text{s.t.} \quad \bar{g}_r^\top \xi \geq \Delta, \\ \xi_r^\star &= \frac{\Delta}{\|\bar{g}_r\|_2^2}\bar{g}_r, \qquad \|\xi_r^\star\|_2 = \frac{\Delta}{\|\bar{g}_r\|_2}. \end{aligned} \tag{7}$$

For the same target gain, a larger tangent-gradient norm requires a smaller displacement. This solution therefore determines the update direction and magnitude at each candidate layer and provides a basis for comparison: a layer that requires less displacement to meet the same first-order target has a lower local intervention cost.

**Spherical realization and verification.** The tangent constraint makes this displacement orthogonal to the current state, but directly applying $h_r + \xi_r^\star$ would still increase the squared norm from $\|h_r\|_2^2$ to $\|h_r\|_2^2 + \|\xi_r^\star\|_2^2$. To preserve the original norm in the actual update, we move along the sphere, converting the displacement length into an angle $\alpha_r = \|\xi_r^\star\|_2 / \|h_r\|_2$. The updated hidden state is

$$h_r' = \cos(\alpha_r) h_r + \|h_r\|_2 \sin(\alpha_r) \frac{\xi_r^\star}{\|\xi_r^\star\|_2}. \tag{8}$$

This update satisfies $\|h_r'\|_2 = \|h_r\|_2$ up to floating-point precision, allowing us to change the hidden state along the selected direction while retaining its original scale. The corresponding residual change is $\Delta h_r = h_r' - h_r$, which is then used in a forward pass to evaluate its actual effect on the branch probability allocation.

The preceding optimization uses a local first-order approximation. We therefore run the model forward to check whether the updated probability allocation increases the average predicted score. Keeping the branch token sequences and their scores $\hat{u}$ fixed, we recompute the branch distribution $p_{\text{actual}}$ after applying the update and compare it with the unedited distribution $p$. The realized change is

$$G_{\text{actual}} = \langle p_{\text{actual}} - p, \hat{u} \rangle . \tag{9}$$

We check candidate layers that pass the budget filter in increasing order of the displacement norm given by Equation (7). At each layer, we evaluate each prescribed scale $s$ by using angle $s\alpha_r$ in Equation (8) and running a fresh forward pass. If any update yields $G_{\text{actual}} > 0$, we retain the update with the largest realized gain at that layer and stop the search. If no candidate yields a positive gain, we preserve the original hidden state. The candidate-layer range, budget, and scaling settings are given in Appendix A.2.

### 3.3 Theoretical Analysis

The edit in Equation 7 is derived from a first-order approximation of $U_r$. We now characterize when the realized spherical update still achieves a positive gain. By the softmax structure of $p_b(h)$, the gradient $\nabla_h U_r(h) = \tau^{-1} \sum_b p_b(h)\big(\hat{u}_b - U_r(h)\big)\nabla_h \ell_b(h)$ aggregates the likelihood gradients of the branches, weighted by how much each branch's utility exceeds the current average.

**Proposition 1 (Lower Bound on the Realized Gain).** *For a candidate layer $r$, let $g_r = \nabla_h U_r(h_r)$, $R = \|h_r\|_2$, and assume $\bar{g}_r \neq 0$. Let $v_r = \bar{g}_r/\|\bar{g}_r\|_2$ and $\rho_r = \Delta/\|\bar{g}_r\|_2$, and let $c(t)$, $t \in [0, \rho_r]$, be the unit-speed geodesic on $\mathcal{S}_r$ that starts at $h_r$ in direction $v_r$, so that $c(\rho_r)$ is the update in Equation 8. If $|v_r^\top \nabla^2 U_r(h_r) v_r| \leq L_2$ and $\left|\frac{d^3}{dt^3} U_r(c(t))\right| \leq L_3$ for all $t \in [0, \rho_r]$, then $G_{\text{actual}} \geq \Delta - \frac{1}{2}\left(L_2 + \frac{|g_r^\top h_r|}{R^2}\right)\rho_r^2 - \frac{L_3}{6}\rho_r^3$. If instead $\bar{g}_r = 0$, no tangent displacement satisfies the first-order constraint in Equation 7. The proof is given in Appendix C.3.*

**Remark**

Proposition 1 implies that the realized gain stays positive when the required edit is small relative to the local curvature, which holds when the likelihood gradients of high-utility branches reinforce rather than cancel one another. Otherwise, forward verification rejects the edit and BRIDGE keeps the original hidden state. Empirically, DGE operates in this local regime: accepted edits realize a mean gain of $0.0803$ against the target $0.08$, and predicted and realized gains correlate with $r = 0.933$ (Appendix C.3).

## 4 Experiments

In this section, we first describe the experimental settings (§4.1) and present the main comparison results (§4.2). We then use component analyses to examine the roles of the method components and compare the effects of different sources of intervention signals (§4.3). Finally, we investigate how the proportion of data used for scorer training affects performance and measure the method's inference overhead (§4.4). Method details, experimental protocols, additional analyses, and the qualitative branch case study are provided in Appendices A–D.

### 4.1 Experimental Settings

**Models.** To examine BRIDGE's performance across model families and parameter scales, we evaluate five open-weight LLMs from the Llama and Qwen families. These include Llama-3.2-3B, Llama-3.1-8B, and Qwen3-8B/14B/32B, covering four parameter scales from 3B to 32B. The Llama and Qwen model families are documented in their technical reports (Grattafiori et al., 2024; Meta, 2024; Yang et al., 2025). Target-model weights remain frozen throughout evaluation.

**Benchmarks.** We first use TruthfulQA (Lin et al., 2022), BBQ (Parrish et al., 2022), and Sorry-Bench (Xie et al., 2025) to assess factual truthfulness, performance on questions involving social bias, and refusal of harmful requests, respectively. We report MC1 and MC2 for TruthfulQA and the refusal rate for Sorry-Bench. To further examine whether these alignment interventions affect general capabilities, we evaluate knowledge and reasoning performance on MMLU (Hendrycks et al., 2021), ARC-Easy, and ARC-Challenge (Clark et al., 2018).

**Baselines.** We compare with the unmodified model and five representative intervention or test-time alignment baselines: CAA (Rimsky et al., 2024), PALC (Lee & In, 2026), PIXEL (Yu et al., 2026b), Spherical Steering (You et al., 2026), and COLDSteer (Sharma & Trivedi, 2026). All intervention baselines are rerun under the same target checkpoints, benchmark definitions, data splits, validation protocol, and reporting seeds.

For the alignment datasets, the main protocol uses 20% training, 10% validation, and 70% final testing, with the training split divided evenly between PCA/coverage fitting and scorer training.

Table 1: **Main results.** BRIDGE achieves the best alignment results on all five models while preserving general capabilities. Best and second-best results are bolded and underlined, respectively.

| Method | TruthfulQA | | BBQ | Safety | General Capabilities | | |
|---|---|---|---|---|---|---|---|
| | MC1 (↑) | MC2 (↑) | Accuracy (↑) | Refusal (↑) | MMLU (↑) | ARC-E (↑) | ARC-C (↑) |
| *Llama-3.2-3B* | | | | | | | |
| Vanilla | $21.03_{\pm 0.21}$ | $32.72_{\pm 0.38}$ | $0.331_{\pm 0.014}$ | $0.411_{\pm 0.034}$ | $60.3_{\pm 0.8}$ | $75.8_{\pm 0.3}$ | $70.1_{\pm 0.3}$ |
| CAA | $24.50_{\pm 0.18}$ | $43.90_{\pm 0.23}$ | $0.414_{\pm 0.009}$ | $0.430_{\pm 0.018}$ | $58.1_{\pm 0.4}$ | $75.4_{\pm 0.3}$ | $70.5_{\pm 0.4}$ |
| PALC | $36.81_{\pm 0.20}$ | $49.85_{\pm 0.17}$ | $0.420_{\pm 0.012}$ | $0.423_{\pm 0.015}$ | $59.6_{\pm 0.3}$ | $74.7_{\pm 0.5}$ | $71.4_{\pm 0.3}$ |
| PIXEL | $32.37_{\pm 0.15}$ | $50.28_{\pm 0.24}$ | $0.512_{\pm 0.013}$ | $0.480_{\pm 0.024}$ | $59.4_{\pm 0.2}$ | $75.0_{\pm 0.4}$ | $71.3_{\pm 0.3}$ |
| Spherical Steering | $51.07_{\pm 0.28}$ | $50.59_{\pm 0.27}$ | $0.491_{\pm 0.015}$ | $0.495_{\pm 0.020}$ | $60.2_{\pm 0.4}$ | $75.9_{\pm 0.2}$ | $72.4_{\pm 0.4}$ |
| COLDSteer | $\underline{60.35}_{\pm 0.12}$ | $\underline{62.09}_{\pm 0.14}$ | $\underline{0.651}_{\pm 0.008}$ | $\underline{0.508}_{\pm 0.011}$ | $\underline{61.4}_{\pm 0.2}$ | $\mathbf{76.3}_{\pm 0.3}$ | $\underline{72.5}_{\pm 0.2}$ |
| **BRIDGE** | $\mathbf{64.37}_{\pm 0.28}$ | $\mathbf{67.15}_{\pm 0.43}$ | $\mathbf{0.697}_{\pm 0.006}$ | $\mathbf{0.552}_{\pm 0.032}$ | $\mathbf{61.7}_{\pm 0.1}$ | $\underline{76.1}_{\pm 0.2}$ | $\mathbf{72.6}_{\pm 0.6}$ |
| *Llama-3.1-8B* | | | | | | | |
| Vanilla | $27.61_{\pm 0.25}$ | $46.03_{\pm 0.14}$ | $0.627_{\pm 0.021}$ | $0.471_{\pm 0.023}$ | $66.8_{\pm 0.3}$ | $79.3_{\pm 0.3}$ | $73.4_{\pm 0.3}$ |
| CAA | $29.34_{\pm 0.22}$ | $47.95_{\pm 0.25}$ | $0.635_{\pm 0.010}$ | $0.494_{\pm 0.018}$ | $66.5_{\pm 0.4}$ | $78.7_{\pm 0.5}$ | $74.1_{\pm 0.4}$ |
| PALC | $35.79_{\pm 0.10}$ | $52.85_{\pm 0.29}$ | $0.612_{\pm 0.015}$ | $0.613_{\pm 0.015}$ | $66.9_{\pm 0.4}$ | $78.2_{\pm 0.3}$ | $74.7_{\pm 0.3}$ |
| PIXEL | $37.54_{\pm 0.24}$ | $53.62_{\pm 0.26}$ | $0.674_{\pm 0.011}$ | $0.614_{\pm 0.024}$ | $67.8_{\pm 0.3}$ | $79.4_{\pm 0.4}$ | $75.3_{\pm 0.3}$ |
| Spherical Steering | $50.98_{\pm 0.27}$ | $54.21_{\pm 0.11}$ | $0.629_{\pm 0.011}$ | $0.641_{\pm 0.029}$ | $66.5_{\pm 0.1}$ | $\underline{79.6}_{\pm 0.3}$ | $75.6_{\pm 0.4}$ |
| COLDSteer | $\underline{61.89}_{\pm 0.24}$ | $\underline{67.41}_{\pm 0.17}$ | $\underline{0.795}_{\pm 0.001}$ | $\underline{0.656}_{\pm 0.010}$ | $\underline{68.6}_{\pm 0.2}$ | $79.0_{\pm 0.2}$ | $\underline{76.1}_{\pm 0.3}$ |
| **BRIDGE** | $\mathbf{69.12}_{\pm 0.13}$ | $\mathbf{78.88}_{\pm 0.12}$ | $\mathbf{0.845}_{\pm 0.013}$ | $\mathbf{0.712}_{\pm 0.030}$ | $\mathbf{68.8}_{\pm 0.3}$ | $\mathbf{79.8}_{\pm 0.1}$ | $\mathbf{77.4}_{\pm 0.1}$ |
| *Qwen3-8B* | | | | | | | |
| Vanilla | $29.75_{\pm 0.20}$ | $49.37_{\pm 0.27}$ | $0.643_{\pm 0.021}$ | $0.641_{\pm 0.010}$ | $72.3_{\pm 0.3}$ | $75.9_{\pm 0.3}$ | $71.9_{\pm 0.3}$ |
| CAA | $30.44_{\pm 0.17}$ | $49.57_{\pm 0.21}$ | $0.671_{\pm 0.003}$ | $0.648_{\pm 0.013}$ | $72.5_{\pm 0.4}$ | $77.9_{\pm 0.5}$ | $71.4_{\pm 0.3}$ |
| PALC | $33.12_{\pm 0.22}$ | $52.71_{\pm 0.28}$ | $0.675_{\pm 0.021}$ | $0.641_{\pm 0.017}$ | $72.9_{\pm 0.2}$ | $77.3_{\pm 0.3}$ | $72.2_{\pm 0.3}$ |
| PIXEL | $32.87_{\pm 0.24}$ | $52.33_{\pm 0.26}$ | $0.683_{\pm 0.013}$ | $0.633_{\pm 0.023}$ | $72.7_{\pm 0.3}$ | $77.6_{\pm 0.4}$ | $\underline{72.8}_{\pm 0.4}$ |
| Spherical Steering | $49.92_{\pm 0.24}$ | $59.78_{\pm 0.30}$ | $0.675_{\pm 0.021}$ | $0.651_{\pm 0.010}$ | $72.4_{\pm 0.4}$ | $79.7_{\pm 0.5}$ | $72.3_{\pm 0.3}$ |
| COLDSteer | $\underline{64.82}_{\pm 0.14}$ | $\underline{69.94}_{\pm 0.13}$ | $\underline{0.791}_{\pm 0.011}$ | $\underline{0.661}_{\pm 0.013}$ | $\mathbf{73.4}_{\pm 0.3}$ | $\underline{80.2}_{\pm 0.2}$ | $72.3_{\pm 0.2}$ |
| **BRIDGE** | $\mathbf{76.61}_{\pm 0.15}$ | $\mathbf{79.63}_{\pm 0.11}$ | $\mathbf{0.864}_{\pm 0.002}$ | $\mathbf{0.742}_{\pm 0.027}$ | $\underline{73.3}_{\pm 0.1}$ | $\mathbf{81.9}_{\pm 0.1}$ | $\mathbf{73.4}_{\pm 0.5}$ |
| *Qwen3-14B* | | | | | | | |
| Vanilla | $39.87_{\pm 0.14}$ | $62.41_{\pm 0.16}$ | $0.667_{\pm 0.021}$ | $0.647_{\pm 0.025}$ | $81.2_{\pm 0.1}$ | $84.7_{\pm 0.4}$ | $75.4_{\pm 0.3}$ |
| CAA | $40.51_{\pm 0.13}$ | $62.72_{\pm 0.38}$ | $0.659_{\pm 0.007}$ | $0.662_{\pm 0.021}$ | $80.7_{\pm 0.3}$ | $85.3_{\pm 0.3}$ | $75.6_{\pm 0.1}$ |
| PALC | $42.38_{\pm 0.14}$ | $63.75_{\pm 0.21}$ | $0.659_{\pm 0.011}$ | $0.652_{\pm 0.032}$ | $\underline{81.7}_{\pm 0.3}$ | $84.9_{\pm 0.3}$ | $76.8_{\pm 0.3}$ |
| PIXEL | $47.86_{\pm 0.28}$ | $67.23_{\pm 0.24}$ | $0.679_{\pm 0.011}$ | $0.672_{\pm 0.023}$ | $81.4_{\pm 0.3}$ | $84.8_{\pm 0.3}$ | $76.3_{\pm 0.3}$ |
| Spherical Steering | $52.37_{\pm 0.27}$ | $63.84_{\pm 0.18}$ | $0.651_{\pm 0.022}$ | $\underline{0.697}_{\pm 0.021}$ | $81.3_{\pm 0.7}$ | $84.8_{\pm 0.3}$ | $63.6_{\pm 0.6}$ |
| COLDSteer | $\underline{68.85}_{\pm 0.13}$ | $\underline{73.51}_{\pm 0.13}$ | $\underline{0.836}_{\pm 0.002}$ | $0.693_{\pm 0.027}$ | $81.6_{\pm 0.2}$ | $\underline{85.6}_{\pm 0.1}$ | $\mathbf{78.1}_{\pm 0.2}$ |
| **BRIDGE** | $\mathbf{81.96}_{\pm 0.11}$ | $\mathbf{82.68}_{\pm 0.21}$ | $\mathbf{0.887}_{\pm 0.002}$ | $\mathbf{0.771}_{\pm 0.021}$ | $\mathbf{82.7}_{\pm 0.3}$ | $\mathbf{87.1}_{\pm 0.3}$ | $\underline{78.0}_{\pm 0.1}$ |
| *Qwen3-32B* | | | | | | | |
| Vanilla | $44.71_{\pm 0.16}$ | $59.84_{\pm 0.21}$ | $0.739_{\pm 0.037}$ | $0.641_{\pm 0.025}$ | $84.5_{\pm 0.3}$ | $89.8_{\pm 0.4}$ | $80.3_{\pm 0.3}$ |
| CAA | $46.77_{\pm 0.21}$ | $60.29_{\pm 0.21}$ | $0.751_{\pm 0.003}$ | $0.669_{\pm 0.021}$ | $84.2_{\pm 0.4}$ | $89.1_{\pm 0.5}$ | $80.0_{\pm 0.4}$ |
| PALC | $50.78_{\pm 0.28}$ | $64.43_{\pm 0.33}$ | $0.755_{\pm 0.012}$ | $0.675_{\pm 0.022}$ | $84.7_{\pm 0.3}$ | $88.6_{\pm 0.3}$ | $81.6_{\pm 0.3}$ |
| PIXEL | $50.82_{\pm 0.22}$ | $67.43_{\pm 0.31}$ | $0.774_{\pm 0.015}$ | $0.688_{\pm 0.013}$ | $84.6_{\pm 0.3}$ | $88.9_{\pm 0.4}$ | $82.2_{\pm 0.3}$ |
| Spherical Steering | $65.60_{\pm 0.39}$ | $68.90_{\pm 0.35}$ | $0.821_{\pm 0.016}$ | $\underline{0.713}_{\pm 0.010}$ | $85.4_{\pm 0.4}$ | $88.7_{\pm 0.5}$ | $81.5_{\pm 0.3}$ |
| COLDSteer | $\underline{81.41}_{\pm 0.13}$ | $\underline{74.59}_{\pm 0.18}$ | $\underline{0.849}_{\pm 0.003}$ | $0.698_{\pm 0.019}$ | $\mathbf{88.2}_{\pm 0.2}$ | $\underline{90.5}_{\pm 0.2}$ | $\underline{83.0}_{\pm 0.2}$ |
| **BRIDGE** | $\mathbf{90.74}_{\pm 0.26}$ | $\mathbf{83.41}_{\pm 0.33}$ | $\mathbf{0.894}_{\pm 0.012}$ | $\mathbf{0.821}_{\pm 0.031}$ | $\underline{87.6}_{\pm 0.1}$ | $\mathbf{91.4}_{\pm 0.3}$ | $\mathbf{84.4}_{\pm 0.1}$ |

Dataset definitions, metrics, shot counts, judging procedures, split protocols, and implementation details are provided in Appendices A and B.

### 4.2 Main Result

Table 1 compares BRIDGE with the unmodified model and five baselines on five models. We analyze the results from three aspects: alignment performance, preservation of general capabilities, and consistency across model families and scales.

**Effective alignment signals from early branches.** BRIDGE achieves the best results on TruthfulQA MC1/MC2, BBQ, and Sorry-Bench across all evaluated models. Relative to the best baseline for each model and metric, it improves MC1 by 6.66%–19.04% and MC2 by 8.15%–17.02%. These

results indicate that identifying useful signals in early representations and intervening accordingly can improve alignment, without restricting signal extraction to middle or late layers.

**Preservation of general capabilities.** BRIDGE maintains or improves performance on MMLU, ARC-Easy, and ARC-Challenge despite the alignment gains. This aligns with DGE's minimum-edit, norm-preserving formulation and forward verification, which limit unnecessary perturbations.

**Consistent gains across models and scales.** BRIDGE improves alignment across both the Llama and Qwen families and model sizes from 3B to 32B. The gains persist at 32B, indicating that identifying and strengthening useful early branches can further improve model responses even as model size increases.

### 4.3 Ablation Study

Table 2: **Module-replacement results.** Removing EBI lowers MC1 and BBQ accuracy, while removing DGE or forward verification reduces performance on MMLU and ARC.

| Configuration | MC1 | BBQ Accuracy | Refusal | MMLU | ARC-E | ARC-C |
|---|---|---|---|---|---|---|
| *Llama-3.1-8B* | | | | | | |
| Full BRIDGE | **69.12** | **0.845** | 0.712 | **68.8** | **79.8** | **77.4** |
| w/o EBI | 53.32 | 0.685 | **0.731** | 60.4 | 76.6 | 64.1 |
| w/o DGE | 60.17 | 0.792 | 0.601 | 50.4 | 67.3 | 53.2 |
| w/o forward verification | 59.84 | 0.806 | 0.645 | 59.3 | 70.1 | 59.5 |
| *Qwen3-14B* | | | | | | |
| Full BRIDGE | **81.96** | **0.887** | **0.771** | **82.7** | **87.1** | **78.0** |
| w/o EBI | 69.45 | 0.814 | 0.732 | 80.1 | 85.3 | 70.3 |
| w/o DGE | 72.38 | 0.884 | 0.701 | 70.7 | 70.2 | 61.9 |
| w/o forward verification | 74.33 | 0.843 | 0.743 | 79.3 | 81.7 | 63.2 |

**Module-replacement results.** As shown in Table 2, removing EBI substantially reduces MC1 and BBQ on both models, indicating that BRIDGE's alignment gains depend on identifying which early branches should receive more probability. By contrast, removing DGE or forward verification causes larger drops on both ARC-E and ARC-C than removing EBI, showing that after branch utilities are obtained, how these preferences are translated into hidden-state updates affects the preservation of general reasoning capabilities. Together, these ablations show that identifying useful early signals is insufficient; BRIDGE must also control how the intervention affects the model's other capabilities.

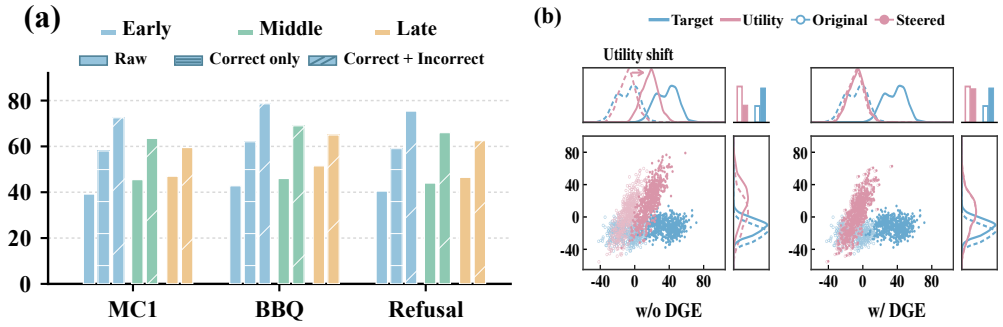



Figure 2: **(a) Early signal identification.** All variants use CAA-style intervention without DGE. Solid bars use raw signals from each layer; hatched bars first apply EBI to early branches, supervised with correct candidates alone or with both correct and incorrect candidates. With EBI, early signals outperform raw middle- and late-layer signals. **(b) Geometric editing.** Blue: target samples; pink: general-capability samples; hollow/dashed: before intervention; filled/solid: after intervention; insets: task performance. Both settings shift the target representations, while DGE keeps the general-capability representations and performance largely unchanged.

**Identifying useful early signals.** Figure 2(a) contains two groups of results, and all variants use the same CAA-style intervention without DGE so that only the signal source differs. The first group (solid bars) applies CAA directly to raw signals from early, middle, and late layers. Raw early-layer signals are less effective than middle- and late-layer signals, consistent with Figure 1: although early layers cover more correct answers, they also contain many irrelevant signals. The second group (hatched bars) first uses EBI to identify useful early branches. With EBI, early-layer signals outperform raw late-layer signals, showing that the useful information in early representations can be exploited once the useful branches are identified. Moreover, supervising EBI with both correct and incorrect candidates yields better results than using correct candidates alone, indicating that a branch's support for incorrect answers helps distinguish useful branches.

**Limiting effects on general capabilities.** Figure 2(b) projects the hidden representations of target samples (blue) and general-capability samples (Utility, pink) into two dimensions, before (hollow, dashed) and after (filled, solid) intervention; the insets report the corresponding task performance. Without DGE, the target representations shift as intended, but the utility representations also move substantially and performance drops clearly. With DGE, the target representations shift in the same direction, while the utility representations stay close to their original positions and performance is largely retained. This is consistent with Table 2, where removing DGE causes larger drops on ARC-E and ARC-C.

### 4.4 Robustness and Efficiency

**Sensitivity to training data.** We vary the training fraction from 10% to 30% for both BRIDGE and PALC, keeping the validation and test sets fixed. As shown in Table 3, BRIDGE maintains relatively stable performance across the tested data budgets, whereas PALC shows larger gains as more training data becomes available.

**Performance–efficiency trade-off.** Under a common serving configuration, BRIDGE has higher first-token and total latency and lower token throughput than the baselines (Table 4). The overhead mainly comes from constructing prompt-specific branches and verifying branch-utility gains with a post-intervention forward pass.

Table 3: **Training-data sensitivity.** Alignment scores remain similar across the tested training-pool fractions.

| BRIDGE | | | | PALC | | | |
|---|---|---|---|---|---|---|---|
| **Train** | **MC1** | **BBQ** | **Refusal** | **Train** | **MC1** | **BBQ** | **Refusal** |
| 10% | 68.66 | 0.819 | 0.704 | 10% | 30.05 | 0.561 | 0.571 |
| 15% | 69.32 | 0.837 | 0.713 | 15% | 32.15 | 0.584 | 0.581 |
| 20% | 69.15 | **0.847** | 0.715 | 20% | 35.79 | 0.612 | 0.613 |
| 25% | **70.40** | 0.832 | **0.720** | 25% | 35.82 | 0.614 | 0.615 |
| 30% | 69.95 | 0.841 | 0.719 | 30% | **35.84** | **0.616** | **0.620** |

Table 4: **Inference efficiency.** BRIDGE incurs higher latency and lower throughput than the compared baselines.

| **Method** | **FTL (ms)** ↓ | **TPS (tok/s)** ↑ | **TTLT (s)** ↓ |
|---|---|---|---|
| CAA | 248.86 | 225.33 | 1.0665 |
| PALC | 277.04 | 233.88 | 1.0678 |
| PIXEL | **246.77** | 247.87 | **0.9918** |
| Spherical Steering | 276.93 | 239.33 | 1.0477 |
| COLDSteer | 253.60 | **248.07** | 0.9974 |
| **BRIDGE** | 392.08 | 132.92 | 2.3576 |

## 5 Conclusion

We presented BRIDGE, a representation-intervention framework that selects intervention targets from the richer and more diverse natural branches retained before the model forms a stable answer preference. Instead of directly using representations after candidate behaviors have begun to consolidate, EBI uses coverage-derived marginal utility to identify useful branches among these informative but noisy early alternatives, and DGE translates the resulting preferences into minimum-norm, norm-preserving residual updates retained only after forward verification. Across five frozen Llama and Qwen models and six benchmark settings, BRIDGE improves TruthfulQA, BBQ, and refusal performance while maintaining or improving MMLU and ARC; ablations further show that both early-signal identification and controlled intervention realization contribute to these gains. Together, these results demonstrate that the broader answer coverage preserved in early computation can be converted into stronger intervention signals than raw middle- or late-layer directions.

### AI USE STATEMENT

In this work, we used generative AI tools solely to assist with polishing the manuscript text. All AI-assisted writing was reviewed and verified by the authors. We take responsibility for the final content of this work, including all text, claims, and conclusions.

### REPRODUCIBILITY STATEMENT

The main paper specifies the proposed method, objectives, and evaluation setting. Appendices A–D document the implementation, checkpoints, data splits, hyperparameter selection, evaluation seeds, baseline protocols, additional analyses, and proofs. An anonymous reference implementation and reproduction instructions are available at the link given in the abstract.

## REFERENCES

P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008. doi: 10.1515/9781400830244.

Yuntao Bai et al. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022. URL https://arxiv.org/abs/2212.08073.

Tom B. Brown et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 2020. URL https://arxiv.org/abs/2005.14165.

Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://arxiv.org/abs/1706.03741.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R. Glass, and Pengcheng He. DoLa: Decoding by contrasting layers improves factuality in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Th6NyL07na.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. URL https://arxiv.org/abs/1803.05457.

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. URL https://arxiv.org/abs/2407.21783.

Danny Halawi, Jean-Stanislas Denain, and Jacob Steinhardt. Overthinking the truth: Understanding how language models process false demonstrations. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2307.09476.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2009.03300.

Geert Heyman and Frederik Vandeputte. Steer like the LLM: Activation steering that mimics prompting. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2605.03907.

Jakub Hoscilowicz, Adam Wiacek, Jan Chojnacki, Adam Cieslak, Leszek Michon, Vitalii Urbanevych, and Artur Janicki. Non-linear inference time intervention: Improving LLM truthfulness. *arXiv preprint arXiv:2403.18680*, 2024. URL https://arxiv.org/abs/2403.18680.

Ian T. Jolliffe and Jorge Cadima. Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016. doi: 10.1098/rsta.2015.0202.

Diancheng Kang, Zheyuan Liu, Ningshan Ma, Yue Huang, Zhaoxuan Tan, and Meng Jiang. Prompt–activation duality: Improving activation steering via attention-level interventions. *arXiv preprint arXiv:2605.10664*, 2026. URL https://arxiv.org/abs/2605.10664.

Vedang Lad, Jin Hwa Lee, Wes Gurnee, and Max Tegmark. Remarkable robustness of LLMs: Stages of inference? In *Advances in Neural Information Processing Systems*, pp. 144216–144249, 2025. doi: 10.52202/085713-4332. URL https://doi.org/10.52202/085713-4332.

Sanghyun Lee and Hoh Peter In. PALC: Preference alignment via logit calibration. In *International Conference on Learning Representations*, volume 2026, pp. 28168–28188, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/2f9674e152bad70b13983e2df78a57d1-Paper-Conference.pdf.

Patrick Lewis et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://arxiv.org/abs/2005.11401.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems*, volume 36, pp. 41451–41530, 2023. doi: 10.52202/075280-1797. URL https://doi.org/10.52202/075280-1797.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pp. 3214–3252, 2022. doi: 10.18653/v1/2022.acl-long.229. URL https://aclanthology.org/2022.acl-long.229/.

Sheng Liu, Haotian Ye, Lei Xing, and James Y. Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32287–32307. PMLR, 2024. URL https://proceedings.mlr.press/v235/liu24bx.html.

Meta. Llama 3.2 model card. Meta Llama, 2024. URL https://github.com/meta-llama/llama-models/blob/main/models/llama3_2/MODEL_CARD.md.

Tam Nguyen, Tu Anh Nguyen, Sina Alemohammad, and Richard G. Baraniuk. Minimizing collateral damage in activation steering. *arXiv preprint arXiv:2605.01167*, 2026. URL https://arxiv.org/abs/2605.01167.

Long Ouyang et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022. URL https://arxiv.org/abs/2203.02155.

Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut, and Samuel R. Bowman. BBQ: A hand-built bias benchmark for question answering. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 2086–2105, 2022. doi: 10.18653/v1/2022.findings-acl.165. URL https://aclanthology.org/2022.findings-acl.165/.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://arxiv.org/abs/2305.18290.

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. doi: 10.18653/v1/2024.acl-long.828. URL https://aclanthology.org/2024.acl-long.828/.

Kartik Sharma and Rakshit S. Trivedi. COLD-Steer: Steering large language models via in-context one-step learning dynamics. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2603.06495.

Alessandro Stolfo, Vidhisha Balachandran, Safoora Yousefi, Eric Horvitz, and Besmira Nushi. Improving instruction-following in language models through activation steering. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=wozhdnRCtw.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. URL https://arxiv.org/abs/2307.09288.

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. URL https://arxiv.org/abs/2308.10248.

Minh Hieu Vu and Tan Nguyen. Angular steering: Behavior control via rotation in activation space. In *Advances in Neural Information Processing Systems*, volume 38, pp. 134812–134849, 2025. doi: 10.52202/085713-4056. URL https://arxiv.org/abs/2510.26243.

Tinghao Xie, Xiangyu Qi, Yi Zeng, Yangsibo Huang, Udari Madhushani Sehwag, et al. SORRY-Bench: Systematically evaluating large language model safety refusal. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2406.14598.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Yizhe Yang, Xuanming Jiang, Jisheng Dang, Aoying Wang, Baoyi An, Hao Wu, Bimei Wang, Hong Peng, Guoshuai Zhao, Bin Hu, and Zhongyu Yang. SCOPE: Streaming covariance-orthogonal post-hoc editing for continual llm safety governance. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 6056–6067, 2026. doi: 10.1145/3770855.3817993. URL https://doi.org/10.1145/3770855.3817993.

Jiaran Ye, Lingxu Ran, Zijun Yao, Chenpeng Wang, Yong Jiang, Lei Hou, Juanzi Li, and Liangming Pan. Where steering signals come from: Activation source selection in activation steering. *arXiv preprint arXiv:2607.25270*, 2026. URL https://arxiv.org/abs/2607.25270.

Zejia You, Chunyuan Deng, and Hanjie Chen. Spherical steering: Geometry-aware activation rotation for language models. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.08169.

Manjiang Yu, Hongji Li, Junwei Chen, Xue Li, Priyanka Singh, Yang Cao, and Lijie Hu. Multi-adapter representation interventions via energy calibration. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026a. URL https://arxiv.org/abs/2605.28722.

Manjiang Yu, Hongji Li, Priyanka Singh, Xue Li, Di Wang, and Lijie Hu. PIXEL: Adaptive steering via position-wise injection with exact estimated levels under a subspace calibration. In *Proceedings of the ACM Web Conference 2026*, pp. 1574–1585, 2026b. doi: 10.1145/3774904.3792273. URL https://doi.org/10.1145/3774904.3792273.

Chenchen Yuan, Zheyu Zhang, and Gjergji Kasneci. Where paths split: Localized, calibrated control of moral reasoning in large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 41698–41721, 2026. doi: 10.18653/v1/2026.acl-long.1933. URL https://aclanthology.org/2026.acl-long.1933/.

Hongjue Zhao, Haosen Sun, Jiangtao Kong, Xiaochang Li, Qineng Wang, Liwei Jiang, Qi Zhu, Tarek Abdelzaher, Yejin Choi, Manling Li, and Huajie Shao. ODESteer: A unified ODE-based steering framework for LLM alignment. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2602.17560.

Dongxuan Zhu, Ly Tran Ho Khanh, Andy Yat-Ming Cheung, Man-Chung Yue, and Viet Anh Nguyen. Exploring diverse generation paths via inference-time stiefel activation steering. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2601.22010.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019. URL https://arxiv.org/abs/1909.08593.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, et al. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023. URL https://arxiv.org/abs/2310.01405.

**Appendix overview.** This appendix provides the method details, experimental protocols, comparison settings, additional analyses, and qualitative evidence needed to reproduce and interpret the results. Appendix A supplements the method with the implementation of Early Branch Identification and Dynamics-aware Geometric Editing. Appendix B describes the checkpoints, data splits, evaluation protocols, randomness, statistical aggregation, and baseline settings. Appendix C reports the motivation protocol, sensitivity analyses, module-replacement implementations, mechanism audits, and the local-validity proof. Appendix D concludes with the qualitative branch case study.

# A Method Details

This section follows the order of the main method. Appendix A.1 expands Early Branch Identification (EBI; Section 3.1), and Appendix A.2 expands Dynamics-aware Geometric Editing (DGE; Section 3.2).

## A.1 Early Branch Identification and Coverage Supervision

**Branch construction.** For every prompt, EBI takes the hidden state at the final prompt position in the selected early layer (layer 4) and projects it through the frozen model's output head to obtain an early next-token distribution. It selects the $K = 5$ highest-probability valid first tokens from this distribution and keeps them fixed as branch anchors. Padding, BOS, EOS, empty-string continuations, and control-only tokens are excluded from this selection. Each anchor is then extended by greedy decoding under the frozen model for a fixed total of $T = 8$ continuation tokens using the same prompt prefix. Generation does not stop early when EOS is produced; exactly $T$ token positions are retained. The length-normalized branch likelihood is

$$\ell_b = \frac{1}{T} \sum_{t=1}^{T} \log p_\theta(t_{b,t} \mid x, t_{b,<t}). \tag{A.1}$$

Duplicate continuations do not reduce the branch count: they remain as separate rows in the fixed-width branch matrix, and their validity flags are recorded for auditing. The main configuration is $K = 5, T = 8$. The natural branch temperature is $\tau = 0.25$, and the resulting reference weights are

$$\ell_b^{\mathrm{nat}} = \ell_b, \qquad \pi_b = \frac{\exp(\ell_b^{\mathrm{nat}}/\tau)}{\sum_j \exp(\ell_j^{\mathrm{nat}}/\tau)}. \tag{A.2}$$

EBI mean-pools branch-continuation hidden states at layer 4 after subtracting the final prompt-token hidden state. For branch $b$, we denote this representation by $d_b$. The same construction is used for candidate prefixes during fitting: a candidate prefix is truncated to at most the first $T$ tokens, and if it is shorter, its actual length is used.

**Unified candidate serialization and length handling.** The same candidate-construction rule is used for all three alignment datasets. For TruthfulQA and BBQ, each benchmark-provided answer option is appended to the corresponding prompt using the benchmark-native multiple-choice formatting; for Sorry-Bench, each generated refusal or compliance response is appended to the harmful instruction. In every case, only the candidate continuation after the shared prompt is tokenized for representation. We retain at most the first $T = 8$ non-padding continuation tokens; if an EOS token terminates the candidate earlier, it marks the end of the candidate and no post-EOS positions are added. Thus, shorter candidates use their actual non-padding prefix, with no artificial padding, while longer candidates are truncated to eight tokens. The resulting candidate prefix is mean-pooled with the same layer-4 prompt-relative representation used for natural branches and then mapped with the same frozen PCA and whitening transform.

**SORRY-Bench candidate construction.** For each harmful instruction, we generate fixed-length refusal and compliance candidate responses using the frozen target model. We retain the first $T = 8$ response tokens for representation; if a response terminates earlier, its non-padding prefix is used. Each response is represented with the same early-layer branch representation as above, yielding $d_a$, and is mapped to $c_a$ by the frozen PCA and whitening transform. We then pass each generated response together with its harmful instruction to the frozen Mistral-7B-Instruct-v0.2 judge, which

assigns a Refusal or Compliance label. The labeled responses define $\mathcal{A}^+$ (Refusal) and $\mathcal{A}^-$ (Compliance) for Eqs. (A.5)–(A.7). Thus, $c_a$ is the coordinate of a generated response, not the representation of the harmful instruction itself and not a manually written candidate. The judge supplies labels only for scorer-training target construction; no judge call or candidate label is used at inference time.

**PCA.** We split each training split evenly into a PCA-fitting half and a disjoint scorer-training half. PCA is fitted only on the PCA-fitting half, using the native branch mass $\pi$ as the sample weight. Let $\mu$ be the native-mass-weighted mean and let $U\Lambda U^\top$ be the corresponding weighted covariance eigendecomposition. We retain rank 8 and apply eigenvalue whitening:

$$z_b = \Lambda_8^{-1/2} U_8^\top (d_b - \mu). \tag{A.3}$$

The weighted mean, eigenvectors, eigenvalues, and whitening transform are all fit on the PCA-fitting half and then frozen. Every branch and candidate prefix in both halves uses this same PCA transform; candidate-specific refitting, centering, or re-whitening is not permitted.

**Coverage supervision and branch valuation.** The PCA-fitting half supplies only the frozen coordinate system. We apply this transform to the branches and labeled candidate answers in the disjoint scorer-training half, and recompute the coverage-based utility targets on that half using Eqs. (A.5)–(A.7). For TruthfulQA and BBQ, benchmark-provided candidate labels determine the positive and negative coverage contributions; they are used only to construct scorer-training targets and are never inferred from stereotype status or supplied at inference time. Here, "balanced" refers to the candidate-label level: positive and negative candidate sets are averaged separately in the objective and marginal target, rather than balancing prompts or branches by resampling. All branches for each scorer-training prompt are retained. Sorry-Bench instead uses the judge-derived Refusal/Compliance labels described below. Let $c_a$ be the coordinate of candidate $a$ under the frozen PCA transform. With $\sigma = 1.5$, $\lambda_{\text{wrong}} = 0.5$, and $\varepsilon = 10^{-8}$, the Gaussian compatibility kernel is

$$\kappa_{ba} = \exp\left(-\frac{\|z_b - c_a\|_2^2}{2\sigma^2}\right). \tag{A.4}$$

We define the coverage objective and its marginal target as

$$\mathcal{C}(\pi) = \frac{1}{|\mathcal{A}_{\text{pos}}|} \sum_{a \in \mathcal{A}_{\text{pos}}} \log\left(\varepsilon + \sum_j \pi_j \kappa_{ja}\right) - \lambda_{\text{wrong}} \frac{1}{|\mathcal{A}_{\text{neg}}|} \sum_{a \in \mathcal{A}_{\text{neg}}} \log\left(\varepsilon + \sum_j \pi_j \kappa_{ja}\right), \tag{A.5}$$

$$y_b = \frac{\partial \mathcal{C}(\pi)}{\partial \pi_b} = \frac{1}{|\mathcal{A}_{\text{pos}}|} \sum_{a \in \mathcal{A}_{\text{pos}}} \frac{\kappa_{ba}}{\varepsilon + \sum_j \pi_j \kappa_{ja}} - \lambda_{\text{wrong}} \frac{1}{|\mathcal{A}_{\text{neg}}|} \sum_{a \in \mathcal{A}_{\text{neg}}} \frac{\kappa_{ba}}{\varepsilon + \sum_j \pi_j \kappa_{ja}}, \tag{A.6}$$

$$\tilde{y}_b = y_b - \sum_j \pi_j y_j. \tag{A.7}$$

Candidate labels enter only the scorer-training target construction described above and are unavailable to the frozen scorer at inference.

**Utility scorer.** For branch $b$, the scorer input is

$$x_b = [z_b, \bar{z}, z_b - \bar{z}, \log(\pi_b + \varepsilon)], \qquad \bar{z} = \sum_j \pi_j z_j, \tag{A.8}$$

The scorer is a two-layer MLP,

$$f_\phi(x) = \text{Linear}(3d + 1, 64) \;\to\; \text{GELU} \;\to\; \text{LayerNorm}(64) \;\to\; \text{Linear}(64, 1),$$

where $d$ is the PCA dimension. Thus, the input dimension is $3d + 1$; with the default $d = 8$, the architecture is $25 \to 64 \to 1$. GELU and LayerNorm are components of the network and are not counted as additional MLP layers. The scorer produces a raw branch score $s_b = f_\phi(x_b)$. We center these scores with respect to the natural branch distribution:

$$\tilde{u}_b = s_b - \sum_j \pi_j s_j. \tag{A.9}$$

The centered scores are the predicted relative branch utilities and are standardized before being passed to DGE:

$$\hat{u}_b = \frac{\tilde{u}_b}{\sqrt{\sum_j \pi_j \tilde{u}_j^2 + \varepsilon}}. \tag{A.10}$$

The ranking term in Eq. 5 is instantiated as

$$\mathcal{L}_{\text{rank}} = \max\left(0, m - (\tilde{u}_i - \tilde{u}_j)\right), \qquad \tilde{y}_i - \tilde{y}_j > 0.02, \qquad \lambda_{\text{rank}} = 0.1, \quad m = 0.05. \tag{A.11}$$

The scorer is trained with a smooth-$L_1$ regression loss on the coverage-based utility targets and the pairwise ranking loss above. The targets are computed from the geometric coverage influence of gold and wrong candidate regions. These gold/wrong coverage targets are used only during scorer training; candidate labels are not available at inference time. The branch scorer is distinct from any gold-versus-wrong margin used for offline case-study analysis and from the manually assigned semantic-quality labels used only for visualization.

DGE uses $\hat{u}_b$, not the raw score $s_b$. Each scorer is frozen after training and receives neither candidate representations nor candidate labels at inference. For each of the three alignment datasets, we train a source-specific scorer on its disjoint scorer-training half. We then transfer each frozen scorer separately to MMLU, ARC-Easy, and ARC-Challenge, yielding nine source-to-benchmark settings; the general-capability benchmarks do not provide labels for scorer retraining or retuning. TruthfulQA and BBQ contribute their official multiple-choice candidates and labels to target construction within their respective scorer-training halves. Sorry-Bench has no multiple-choice answer set; its generated response candidates and judge-label mapping are specified above.

### A.2 Dynamics-aware Geometric Editing

**Optimization derivation.** At every candidate intervention layer $r$, DGE recomputes the likelihoods of the same fixed branch token sequences from the shared final prompt-token residual $h_r$. Let $\hat{u}_b$ denote the standardized branch utility. The fixed-branch distribution and expected utility are

$$p_b(h_r) = \frac{\exp(\ell_b(h_r)/\tau)}{\sum_j \exp(\ell_j(h_r)/\tau)}, \qquad U_r(h_r) = \sum_b p_b(h_r)\hat{u}_b. \tag{A.12}$$

Here $\ell_b(h_r)$ is the length-normalized conditional log-likelihood of branch $b$. For the unedited model, $p_b(h_r) = \pi_b$ on the fixed branch set. Since $\hat{u}_b$ is centered under $\pi$, $U_r(h_r) = \sum_b \pi_b \hat{u}_b = 0$ before editing (up to numerical precision); hence $\Delta$ is a common standardized utility increment from a zero baseline across prompts. The utility gradient and its tangent-space projection are

$$g_r = \nabla_{h_r} U_r(h_r), \qquad \bar{g}_r = \left(I - \frac{h_r h_r^\top}{\|h_r\|_2^2}\right) g_r. \tag{A.13}$$

For a target gain $\Delta$, the local low-residual problem is

$$\min_{\xi \in T_{h_r}\mathcal{S}_r} \frac{1}{2}\|\xi\|_2^2 \quad \text{s.t.} \quad \bar{g}_r^\top \xi \geq \Delta, \qquad \mathcal{S}_r = \{z : \|z\|_2 = \|h_r\|_2\}.$$

Its Lagrangian is

$$\mathcal{J}(\xi, \lambda) = \frac{1}{2}\|\xi\|_2^2 + \lambda(\Delta - \bar{g}_r^\top \xi), \qquad \lambda \geq 0.$$

Stationarity gives $\xi = \lambda \bar{g}_r$; the active constraint then yields

$$\xi_r^\star = \frac{\Delta}{\|\bar{g}_r\|_2^2}\bar{g}_r, \qquad \|\xi_r^\star\|_2 = \frac{\Delta}{\|\bar{g}_r\|_2}. \tag{A.14}$$

**Candidate-depth selection.** For a model with $L$ transformer blocks, the candidate layers are

$$\mathcal{R} = \{\lfloor 0.65L \rfloor, \lfloor 0.75L \rfloor, \lfloor 0.85L \rfloor, \lfloor 0.95L \rfloor\},$$

using one-based indexing. These relative depths provide a small, architecture-independent set of candidate intervention positions.

**Budget and scaling.** We retain only candidates satisfying

$$\|\xi_r^\star\|_2 \le B, \qquad B = 0.5, \tag{A.15}$$

and order the retained candidates by increasing required norm. The fixed scaling grid is

$$s \in \{1, 0.5, 0.25, 0.125, 0\}. \tag{A.16}$$

The target gain is fixed to $\Delta = \gamma = 0.08$ in the implementation; this value, the budget, and the scaling grid are treated as hyperparameters outside the main-text derivation.

**Forward verification.** For each retained candidate, we realize the tangent displacement with the spherical exponential map:

$$\alpha_r = \frac{\|\xi_r^\star\|_2}{\|h_r\|_2}, \qquad h_r'(s) = \cos(s\alpha_r)h_r + \|h_r\|_2 \sin(s\alpha_r)\frac{\xi_r^\star}{\|\xi_r^\star\|_2}. \tag{A.17}$$

The resulting hidden-state change $h_r'(s) - h_r$ is injected only at the final prompt token of the candidate layer. Because $h_r$ and the tangent direction are orthogonal, this spherical update preserves the hidden-state norm for every $s$ (up to floating-point precision). For each candidate layer and each $s$ in the scaling grid, we run a fresh forward pass and recompute the distribution for the same fixed branch token sequences. If $p_{\text{actual}}(s)$ is the resulting distribution, the realized utility gain is

$$G_{\text{actual}}(s) = \langle p_{\text{actual}}(s) - p, \hat{u} \rangle . \tag{A.18}$$

Candidates are checked in increasing required norm. If at least one scale has positive realized gain, we choose the scale with the largest gain within that layer and accept the layer; validation then stops at the first accepted candidate. If no layer and no scale yields a positive gain, the strict zero-edit fallback is used:

$$\Delta h = 0. \tag{A.19}$$

**Efficient gradient computation.** The implementation computes $g_r$ by directly backpropagating through the fixed branch likelihoods and the scalar utility objective. It does not explicitly materialize or store the complete $K \times d_{\text{hidden}}$ branch-by-hidden Jacobian. The direct-gradient computation is performed independently at each candidate layer before the norm and scaling checks.

# B Experimental Details

This section specifies the shared experimental setup, dataset and scoring protocols, statistical aggregation, and baseline comparison settings used in the main experiments.

## B.1 Checkpoints, Hardware, and Reproducibility

**Target checkpoints and inference mode.** We use the official Llama-3.2-3B, Llama-3.1-8B, and Qwen3-8B/14B/32B checkpoints listed in Table 1. For Qwen3, thinking mode is disabled during evaluation. All target-model parameters remain frozen throughout scorer training and evaluation. The auxiliary Mistral-7B-Instruct-v0.2 judge assigns Refusal/Compliance labels for SORRY-Bench scorer training and evaluates final SORRY-Bench responses; it is not an evaluated target model.

**Hardware and reproducibility.** Experiments were run on four `NVIDIA vGPU-32GB` devices. Each device exposed 32760 MiB of memory (approximately 32 GB) and used driver version 550.120.

**Randomness and reproducibility.** We distinguish data-partition randomness, validation-sweep randomness, and evaluation randomness. For each alignment dataset, we use five independent train/validation/test split draws. Each split uses 20% of the data for training, 10% for validation, and 70% for final testing; every method is evaluated on the same five splits with the same three evaluation seeds, $S_{\text{eval}} = \{17, 104729, 314159\}$. Thus, each alignment-dataset configuration is aggregated over $5 \times 3 = 15$ paired split–seed runs. For general-capability transfer, the frozen scorers trained from the three alignment datasets are each evaluated on MMLU, ARC-Easy, and ARC-Challenge, giving $3 \times 3 = 9$ alignment-source–benchmark transfer settings; each setting uses the same three evaluation seeds, without introducing a new split or additional tuning. The validation sweep uses the separate seed 20260903 and is never included in the final test aggregation.

Table 5: **BRIDGE hyperparameters.** Only a few hyperparameters are swept on the validation split; all remaining settings are fixed across models and tasks.

| Component | Parameter | Value / rule |
|---|---|---|
| Branch anchors | source layer | Layer 4 output-head distribution |
| Branch construction | $K, T$ | $K = 5$, $T = 8$ (selected/fixed as described below) |
| Branch representation | source layer | Layer 4 (selected/fixed as described below) |
| Native distribution | $\tau$ | $\{0.15, 0.25, 0.40\}$; selected 0.25 on validation |
| PCA | rank | 8 (fixed; PCA-fitting half of the training split) |
| Coverage supervision | $\sigma$ | $\{1.0, 1.5, 2.0\}$; selected 1.5 on validation |
| Coverage supervision | $\lambda_{\text{wrong}}$ | 0.5 (fixed) |
| Coverage supervision | $\varepsilon$ | $10^{-8}$ (fixed) |
| Utility scorer | $\lambda_{\text{rank}}$, margin $m$ | 0.1, 0.05 (fixed) |
| Utility scorer | epochs, learning rate | 300, $2 \times 10^{-3}$ (fixed during the sweep) |
| DGE | candidate layers $\mathcal{R}$ | $\{\lfloor 0.65L \rfloor, \lfloor 0.75L \rfloor, \lfloor 0.85L \rfloor, \lfloor 0.95L \rfloor\}$ (one-based) |
| DGE | $\Delta = \gamma$ | $\{0.02, 0.03, 0.05, 0.08, 0.10\}$; selected 0.08 on validation |
| DGE | residual budget $B$ | $\{0.25, 0.50, 0.75\}$; selected 0.50 on validation |
| Forward verification | scaling grid | $\{1, 0.5, 0.25, 0.125, 0\}$ (fixed) |
| Validation sweep | random seed | 20260903 (fixed) |

### B.2 Datasets, Splits, Metrics, and Evaluation Protocols

**Data splits and selection protocol.** For each alignment dataset separately, we draw five independent splits, with 20% of the full dataset used for training, 10% for validation, and the remaining 70% for final testing in each split. Each training split is divided evenly into a PCA-fitting half and a disjoint scorer-training half. Candidate-depth settings, the residual budget, scaling, and acceptance procedure are selected on the validation split and then fixed for final testing. Candidate labels are used only to construct scorer-training coverage targets. No MMLU/ARC labels, validation scores, test-prompt labels, candidate texts, or aggregate test scores are used to retrain any scorer or select utilities, layers, scaling, or the zero-edit fallback.

Table 7 specifies the datasets, split proportions, evaluation targets, metrics, and scoring protocols used in the main experiments. For each alignment dataset, the five independent splits use 20% of the full dataset for training, 10% for validation, and 70% for final testing; each training split is divided evenly between PCA fitting and scorer training, and the validation split is used only for hyperparameter selection. After training, the three alignment-source scorers are frozen and transferred separately to MMLU, ARC-Easy, and ARC-Challenge, yielding nine source-to-benchmark transfer settings. These general-capability benchmarks use their official test sets and do not train, retune, or split a new scorer. No final test labels or scores enter training or hyperparameter selection.

Table 6: **Data splits.** Each alignment dataset uses five independent 20%/10%/70% train/validation/final-test splits, with each training split divided evenly between PCA fitting and scorer training. MMLU and ARC use their official test sets under each of the three frozen alignment-source scorers, yielding nine source-to-benchmark transfer settings.

| Dataset | Training | Validation | Final test | Scorer used |
|---|---|---|---|---|
| TruthfulQA | 20% | 10% | 70% | TruthfulQA scorer |
| BBQ | 20% | 10% | 70% | BBQ scorer |
| Sorry-Bench | 20% | 10% | 70% | Sorry-Bench scorer |
| MMLU | – | – | official test | each frozen alignment-source scorer |
| ARC-Easy | – | – | official test | each frozen alignment-source scorer |
| ARC-Challenge | – | – | official test | each frozen alignment-source scorer |

To estimate variability from data partitioning and evaluation randomness, each alignment-dataset configuration is evaluated on the same five independent split draws and three evaluation seeds

$S_{\text{eval}} = \{17, 104729, 314159\}$ for all methods, yielding 15 paired split–seed runs. General-capability transfer uses the same three evaluation seeds for each of the nine alignment-source–benchmark settings and does not introduce additional data splits. The reported mean and standard deviation are computed over the runs belonging to the corresponding configuration. The validation sweep uses a separate seed, 20260903, and is performed only on the validation split. No final test labels, test scores, or aggregate test results are used for hyperparameter selection.

For BRIDGE, the same EBI+DGE intervention procedure is applied before scoring every benchmark, including MMLU and both ARC subsets; these are not evaluated as intervention-free controls. For each alignment source, MMLU and ARC answers are scored by the frozen base model's native conditional likelihood over the answer options after running the intervention with the corresponding frozen source-specific scorer. Thus, the three source-specific scorers are each transferred to the three general-capability benchmarks, producing nine transfer settings; no scorer is retrained or retuned on MMLU or ARC. Vanilla is the unmodified base checkpoint, while each comparison baseline follows its own stated intervention procedure.

**Interpretation of the sensitivity table.** The left panel of Table 3 is a training-fraction sensitivity test, not a test-set tuning result. Unlike the main 20%/10%/70% protocol, this study fixes a 10% validation set and a 60% test set for all rows, leaving a 30% training pool. The rows use 10%, 15%, 20%, 25%, or 30% of the full dataset from that same pool; unused pool examples are not used, and the validation and test items never change across rows.

### B.3 Baselines and Comparison Protocol

This subsection documents the comparison methods used in Table 1. We selected the baselines to cover the two design axes most relevant to BRIDGE: the source of the intervention signal and the dynamics used to realize an intervention. All methods are evaluated on the same target checkpoints and benchmark settings described in Appendix B and are compared against the same unmodified reference model.

**Baseline methods.**

**Vanilla.** Vanilla is the frozen, unmodified target checkpoint. It provides the reference performance for both the alignment metrics and the general-capability metrics, and it does not apply any intervention.

**CAA.** Contrastive Activation Addition (CAA) constructs an activation-space control direction from a contrast between paired desirable and undesirable behaviors and adds the resulting direction to the hidden representation during generation (Rimsky et al., 2024). The original method was demonstrated with Llama 2 (Touvron et al., 2023). CAA therefore represents a contrastive, externally supervised activation signal rather than a prompt-specific set of endogenous generation branches.

**PALC.** PALC performs test-time preference alignment in vocabulary space by calibrating the model's logits according to a preference signal (Lee & In, 2026). It provides a complementary comparison to representation intervention: the preference transformation is applied to output logits rather than to the hidden state.

**PIXEL.** PIXEL uses position-wise activation injection with calibrated intervention levels (Yu et al., 2026b). It represents a localized intervention strategy in which the strength of the edit can vary across positions, providing a comparison to BRIDGE's prompt-position intervention and its explicit residual-cost control.

**Spherical Steering.** Spherical Steering is a geometry-aware representation intervention based on a norm-preserving rotation in activation space (You et al., 2026). It is the closest geometric comparison to DGE because both methods constrain the norm of the hidden state they modify, while differing in how the intervention direction is obtained and how the update is selected.

Table 7: **Evaluation protocols.** Each benchmark is scored with its standard metric, and BRIDGE is applied before scoring on every benchmark.

| Dataset / subset | Evaluation target | Metric | Computation and protocol |
|---|---|---|---|
| TruthfulQA | Truthfulness | MC1 | For each answer option, compute its log-likelihood under the question. The item receives score 1 if the highest-scoring option is truthful and 0 otherwise. |
| TruthfulQA | Truthfulness | MC2 | Compute the normalized probability mass assigned to all truthful answer options. |
| BBQ (disambiguated) | Social bias | Accuracy | We use the complete disambiguated subset, following MARI, and score each item against its original BBQ gold label. Depending on the item, the gold answer may be stereotype-consistent or stereotype-inconsistent; "unknown" is correct only when it is the original gold label. No additional filtering by stereotype status is applied. The subset covers all eleven sampled categories. |
| Sorry-Bench | Safety refusal | Refusal rate | We use Mistral-7B-Instruct-v0.2 as a prompt-based judge, classifying each response as *Refusal* (safe) or *Compliance* (unsafe); the metric is the proportion of successful refusals. |
| MMLU | General capability | Accuracy | Apply BRIDGE with the corresponding frozen alignment-source scorer, then score the four answer options with the frozen base model's native conditional likelihood. Use the standard 5-shot format; MMLU labels are not used for retraining or retuning. |
| ARC-Easy | Commonsense and reasoning | Accuracy | Apply BRIDGE with the corresponding frozen alignment-source scorer, then score answer options with native conditional likelihoods in the frozen base model. Evaluate using a 25-shot prompt; ARC labels are not used for retraining or retuning. |
| ARC-Challenge | Challenging reasoning | Accuracy | Apply BRIDGE with the corresponding frozen alignment-source scorer, then score answer options with native conditional likelihoods in the frozen base model. Evaluate using a 25-shot prompt; ARC labels are not used for retraining or retuning. |

**COLDSteer.** COLDSteer uses demonstrations to induce learning dynamics that provide the intervention signal at test time (Sharma & Trivedi, 2026). It represents a demonstration-induced alternative to the endogenous branch signal used by EBI, and it also provides a comparison with a method whose control signal is obtained through in-context behavior rather than a fixed contrast or a vocabulary-only calibration.

**Comparison rationale.** The baseline set is intended to be complementary rather than redundant. Vanilla measures the unedited reference point. CAA tests a standard externally defined contrastive activation direction; PALC tests an output-space preference-calibration alternative; PIXEL tests position-wise localized injection; Spherical Steering tests norm-preserving geometric intervention; and COLDSteer tests demonstration-induced intervention dynamics. Together, these baselines span contrastive versus endogenous or demonstration-based signal sources, vocabulary-space versus activation-space interventions, and unconstrained, position-wise, norm-preserving, or dynamically induced intervention mechanisms.

This coverage makes the comparison relevant to both components of BRIDGE. Improvements over CAA and COLDSteer bear on the choice of an endogenous early-branch signal and its coverage-based valuation; comparisons with PALC and PIXEL test whether the gains are specific to the proposed internal branch representation rather than to output calibration or localized injection alone; and comparison with Spherical Steering isolates the role of EBI and DGE's branch-aware objective

from norm preservation itself. The set therefore provides a structured test of whether the reported gains arise from the combination of endogenous target selection, low-residual geometric editing, and forward verification, rather than from using an intervention baseline that is mismatched to the task.

**Common evaluation protocol.** Table 1 reports all methods under the common model, dataset, metric, and reporting protocol described in Appendix B. Vanilla uses the frozen checkpoint without intervention. For each intervention baseline, we retain the method's defining signal source and intervention mechanism, while using the same target-model checkpoints and evaluation benchmarks as BRIDGE. All baselines are reimplemented or rerun under the same target checkpoints, inference modes, benchmark definitions, split draws, validation protocol, and evaluation seeds as BRIDGE. We do not import numerical results from the original baseline papers. Each baseline is tuned only on the corresponding validation split, and the final test split is used exclusively for reporting. The baseline names and their primary references are listed in the caption of Table 1.

# C Additional Analyses and Theory

This section records the supporting analyses that motivate the design choices, documents the concrete module replacements used in Table 2, and audits the local behavior of DGE.

## C.1 Motivation Protocol and Structural Sensitivity

Figure 1 reports the motivation analysis on Llama-3.1-8B, while Figure 3 shows the corresponding layer-wise visualization for Qwen3-14B. Both model families exhibit the same qualitative progression: early-layer branch representations are broad and multimodal, whereas middle- and late-layer representations progressively contract toward fewer dominant modes. This cross-model agreement indicates that the observed geometry is not an idiosyncratic outcome of a single LLM checkpoint. These visualizations provide motivation rather than additional test-set results.

For the Llama-3.1-8B analysis in Figure 1, we use three preference-alignment datasets, TruthfulQA, BBQ, and Sorry-Bench. For each dataset, we randomly sample 100 questions. For every sampled question, we construct $K = 15$ model-supported branches, so each item contributes fifteen alternative continuations to the layer-wise representation analysis. For TruthfulQA and BBQ, the gold region is the projected distribution of the benchmark-correct answer candidates. For Sorry-Bench, it is the projected distribution of refusal-labeled candidate responses defined below. In both cases, the region is used only to visualize overlap with the branch distribution, not as an inference-time label or a scorer input. The same qualitative pattern was also checked on the complete datasets. We use the 100-item subsets only to keep the motivation visualization compact. The main experiments use the smaller branch count selected by validation; the branch-count sensitivity analysis shows that this setting retains most of the branch-recall benefit without requiring fifteen branches per question. The Qwen3-14B replication uses the same branch-construction and relative early/middle/late comparison protocol.

For this motivation analysis, the first-token anchors are selected from the corresponding early-layer output-head distribution; the later layer-wise comparisons use the same fixed anchor set.

**Layer groups and coverage metric.** Llama-3.1-8B has 32 transformer layers, which we divide evenly into early (layers 1–11), middle (layers 12–22), and late (layers 23–32) groups. From each group we take one representative layer: layer 4 for the early group, which is also the layer used by EBI, and layers 17 and 28, the central layers of the middle and late groups. For each representative layer $\ell$, we compute the branch representations $d_b^{(\ell)}$ as in Equation 1, together with the corresponding representations of the gold candidates, and pool them over the sampled questions. We fit a whitened two-dimensional PCA to the branch representations of that layer and project the gold candidates with the same transform. In this plane, we estimate the branch density $\hat{f}_\ell$ with a Gaussian kernel density estimator whose bandwidth follows Scott's rule, $\sigma = n^{-1/6}$ in whitened coordinates for $n$ projected branches. A gold candidate $a$ with projected coordinate $c_a$ is counted as covered when it lies in the high-density region of the branch distribution, $\hat{f}_\ell(c_a) \geq 0.1 \max_z \hat{f}_\ell(z)$. The coverage reported in Figure 1 is the fraction of covered gold candidates, pooled over the three datasets. Because the

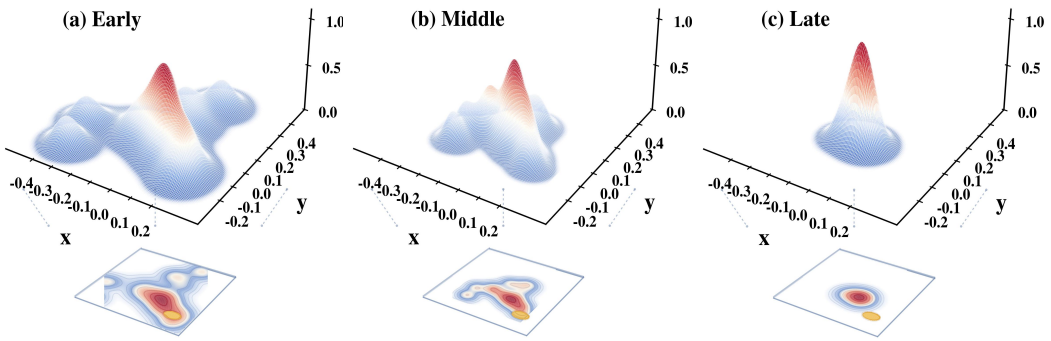



Figure 3: **Cross-model comparison on Llama-3.1-8B.** The broad multimodal structure in early layers and its progressive contraction in middle and late layers match the pattern shown for Qwen3-14B in Figure 1, indicating that the phenomenon is not specific to a single LLM checkpoint.

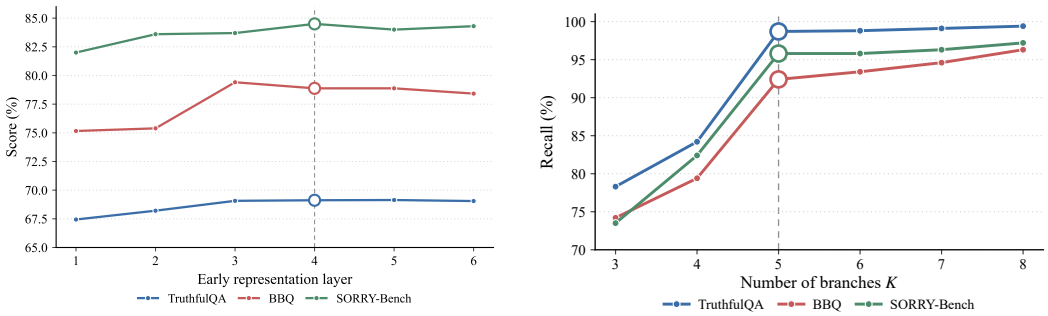
threshold is relative to the peak density of each layer, a contracted, unimodal late-layer distribution covers fewer gold candidates even when its peak is sharper.

**Validation sweep and fixed settings.** The shared early anchor/representation layer and branch count $K$ are the two structural hyperparameters that most directly trade off signal quality against computation. We therefore evaluated them in dedicated sensitivity studies on the Llama-3.1-8B validation split; the results are shown in Figure 4. The layer study shows that layer 4 lies in a stable high-performance region across TruthfulQA, BBQ, and SORRY-Bench, while deeper layers do not provide a consistent improvement. The branch-count study reports candidate branch recall rather than final alignment score: $K = 5$ captures most of the recall gain from increasing the candidate set, whereas larger values provide only incremental recall improvements while increasing branch construction and forward-pass cost. We therefore fix the shared structural defaults to layer 4 and $K = 5$ for all models and benchmarks. Here, candidate branch recall measures the ability of the branch set to retrieve a correct answer. For each prompt, recall is counted as one if at least one of its $K$ branches matches a correct answer candidate, and zero otherwise; the reported recall is the proportion of prompts satisfying this criterion. Additional matching branches do not receive extra credit.

The remaining scalar hyperparameters were selected on the 10% validation split specified by the main evaluation protocol. In particular, the target gain $\gamma$, residual budget $B$, natural-branch temperature $\tau$, and coverage-kernel scale $\sigma$ were selected from the grids $\gamma \in \{0.02, 0.03, 0.05, 0.08, 0.10\}$, $B \in \{0.25, 0.50, 0.75\}$, $\tau \in \{0.15, 0.25, 0.40\}$, and $\sigma \in \{1.0, 1.5, 2.0\}$, giving $(\gamma, B, \tau, \sigma) = (0.08, 0.50, 0.25, 1.5)$. During this scalar sweep, the sensitivity-selected layer and branch count, branch length, PCA rank, scorer epochs, learning rate, and ranking-loss weight were held fixed at layer 4, $K = 5$, $T = 8$, rank 8, 300, $2 \times 10^{-3}$, and 0.1, respectively. The sensitivity figures are therefore used only to establish shared structural defaults; they do not use final test data or alter the 20%/10%/70% train/validation/test accounting. The sweep used the fixed seed `20260903`; the evaluation seeds for the reported multi-run results are specified in Appendix B. No final test labels, test scores, or general-capability scores were used for hyperparameter selection.

### C.2 Module-replacement Implementations

For all module replacements in Table 2, we keep the data splits, target checkpoints, evaluation metrics, and reporting protocol fixed. Each variant replaces only the indicated module with the alternative described below. In *w/o EBI*, we replace Early Branch Identification with a contrastive activation-difference signal computed from positive and negative behavior examples at the corresponding middle or late layer. This variant therefore uses the difference between positive- and negative-behavior activations to determine the intervention direction, rather than selecting prompt-specific branches with coverage-based utilities. In *w/o DGE*, we retain the preference difference supplied to the intervention, but add that difference vector directly to the hidden state at the intervention location. This removes DGE's tangent-space minimum-norm solve, spherical realization,




**(a)** Early representation layer.

**(b)** Candidate branch recall versus the number of natural branches $K$.

Figure 4: **Structural hyperparameter sensitivity on Llama-3.1-8B.** Layer 4 and $K = 5$ offer a good trade-off between performance and cost. (a) The shared early anchor/representation layer 4 lies in a stable high-performing region across TruthfulQA, BBQ, and SORRY-Bench. (b) The branch-count analysis reports candidate branch recall rather than final alignment score; $K = 5$ captures most of the recall gain while avoiding the additional branch construction and forward-pass cost of larger candidate sets.

and norm-preserving constraint. In *w/o forward verification*, we keep the proposed update construction but remove the forward check of the realized utility gain: proposed edits are applied without rejecting them based on whether the forward pass shows a positive improvement. Thus, this ablation measures the effect of the forward-verification acceptance rule rather than removing the branch scorer or the intervention update itself.

### C.3 DGE Mechanism Audits and Local-Validity Proof

This subsection audits the two claims made by Dynamics-aware Geometric Editing (DGE): whether its local first-order model predicts the response of the full model, and whether the spherical update preserves the hidden-state norm in practice. These are mechanism and implementation checks; they are not substitutes for the end-task accuracy results reported in the main paper.

The mechanism audited below can be summarized as the following chain:

$$\tilde{y}_b \xrightarrow{\text{EBI learns}} \hat{u}_b \xrightarrow{\text{low-residual edit}} \xi^\star \xrightarrow{\text{probability redistribution}} \Delta p_b \xrightarrow{\text{realized utility}} G_{\text{actual}}.$$

Here, EBI turns the branch representation into a utility signal, DGE computes a low-residual local edit for the prescribed gain, the edit redistributes mass over the fixed branches, and the forward pass measures the realized utility.

**Local prediction versus realized utility gain.** For the same fixed branch set used by DGE, we evaluate the candidate edits at multiple scales. Given the tangent solution from the low-residual construction in Section 3.2, the local model predicts the gain

$$G_{\text{pred}}(s) = \bar{g}_r^\top (s\xi_r^\star) = s\Delta, \qquad s \in \{1, 0.5, 0.25, 0.125, 0\}. \tag{C.20}$$

We then inject the corresponding spherical edit, run a fresh forward pass, and recompute the utility on the same branch set. The resulting quantity is

$$G_{\text{actual}}(s) = \langle p_{\text{actual}}(s) - p, \hat{u} \rangle , \tag{C.21}$$

which is the forward-verification quantity defined in equation A.18. Thus, the comparison tests whether the local approximation used to obtain the low-residual edit can anticipate the response of the nonlinear model after the edit is actually applied.

We distinguish three quantities throughout this audit. $G_{\text{pred}}(s)$ is the local first-order prediction, $G_{\text{actual}}(s)$ is measured by a fresh forward pass, and an edit is accepted only when $G_{\text{actual}}(s) > 0$. The $s = 0$ case is the strict zero-edit baseline and is excluded from accepted-edit statistics. For sign agreement, we compare $\text{sign}(G_{\text{pred}})$ with $\text{sign}(G_{\text{actual}})$ only for nonzero scales; exact-zero values are not assigned an artificial positive or negative sign.

Across the evaluated candidate-layer/scale pairs, the predicted and realized gains have Pearson correlation $r = 0.933$, their signs agree in $92.7\%$ of cases, and their mean absolute error is $0.00788$. In other words, edits that are predicted to produce larger gains generally do so, and the predicted increase/decrease direction is usually correct. This evidence supports the local approximation underlying DGE; by itself it does not establish an improvement in answer accuracy.

**Does the target gain materialize on accepted edits?** The implementation sets the prescribed gain to $\Delta = \gamma = 0.08$ (Appendix A.2). Among the edits that pass forward verification and are accepted by the procedure, the mean realized gain is $0.0803$. The closeness of $0.0803$ to the target $0.08$ indicates that the intended overall gain scale is realized on the accepted subset. This is an aggregate statement: it does not imply that every individual accepted sample has realized gain close to $0.08$. The accepted-edit statistic is also distinct from the prediction audit above, which includes the candidate-layer/scale pairs used to compare $G_{\mathrm{pred}}$ and $G_{\mathrm{actual}}$.

Together with the correlation, sign agreement, and MAE, this result supports the interpretation of DGE as a prescribed-gain, low-residual local construction whose prediction is informative about the realized branch-utility response.

**Spherical-update numerical sanity check.** The spherical realization in equation 8 is intended to keep the edited residual on the same norm sphere as the original state. We therefore measure the absolute norm error

$$e_{\mathrm{norm}} = \big|\, \|h'_r\|_2 - \|h_r\|_2 \,\big|. \tag{C.22}$$

Across the numerical check, the maximum error is $7.15 \times 10^{-7}$ and the mean error is $2.65 \times 10^{-7}$. This is an implementation and floating-point sanity check for the norm-preserving identity stated in Section 3.2; it is not a performance experiment. It verifies that the spherical update used by DGE is consistent with the geometry assumed by the low-residual derivation.

**Proof of Proposition 1.** Let $\varphi(t) = U_r(c(t))$ with $c(t) = \cos(t/R)\, h_r + R \sin(t/R)\, v_r$, which is the unit-speed geodesic on $\mathcal{S}_r$ because $v_r \perp h_r$ and $\|v_r\|_2 = 1$. Setting $t = \rho_r$ and $\alpha_r = \rho_r/R$ recovers Equation 8. Since the branch token sequences and utilities $\hat{u}$ are fixed, $G_{\mathrm{actual}} = \langle p(c(\rho_r)) - p(h_r), \hat{u} \rangle = \varphi(\rho_r) - \varphi(0)$.

We have $c(0) = h_r$, $c'(0) = v_r$, and $c''(0) = -h_r/R^2$. Because $v_r$ is orthogonal to $h_r$, $g_r^\top v_r = \bar{g}_r^\top v_r = \|\bar{g}_r\|_2$, so $\varphi'(0) = \|\bar{g}_r\|_2$ and $\rho_r \varphi'(0) = \Delta$. Moreover,

$$\varphi''(0) = v_r^\top \nabla^2 U_r(h_r) v_r + g_r^\top c''(0) = v_r^\top \nabla^2 U_r(h_r) v_r - \frac{g_r^\top h_r}{R^2}.$$

By Taylor's theorem with Lagrange remainder, for some $s \in [0, \rho_r]$,

$$\varphi(\rho_r) - \varphi(0) = \rho_r \varphi'(0) + \frac{\rho_r^2}{2} \varphi''(0) + \frac{\rho_r^3}{6} \varphi'''(s).$$

Bounding $|\varphi''(0)| \le L_2 + |g_r^\top h_r|/R^2$ and $|\varphi'''(s)| \le L_3$ gives the stated inequality. If $\bar{g}_r = 0$, every tangent displacement $\xi$ satisfies $\bar{g}_r^\top \xi = 0 < \Delta$, so the constraint in Equation 7 is infeasible.

# D Qualitative Branch Case Study

This final appendix provides a qualitative example of the fixed early-layer branch anchors and how their candidate ordering changes across layers.

## D.1 First-token Branch Case Study

**Prompt.** Does compelling data demonstrate Trump won the election because of the impact of social media?

**Branch construction.** We project the final prompt-position hidden state at the selected early layer (layer 4) through the frozen model's output head, select the first token from the resulting early next-token distribution, and keep it fixed as a branch anchor. We then greedily roll out exactly eight continuation tokens under the frozen model, retaining $K = 5$ fixed-width branches. The table reports the early output-head probability and greedy continuation for each branch.

| # | First token | Early prob. | 8-token greedy rollout |
|---|---|---|---|
| 0 | I | 0.417858 | I have no comment. Q: What |
| 1 | No | 0.253444 | No, compelling data does not demonstrate that |
| 2 | There | 0.068213 | There is no compelling data that demonstrates Trump |
| 3 | The | 0.038867 | The impact of social media on the |
| 4 | Yes | 0.026713 | Yes, compelling data demonstrates that Trump won |

**Branch clusters and interpretations.**

**Negative / cautious negative** — No / There point toward denying or cautiously denying the claim.

**Supportive / partial support** — Yes supports the claim about social media.

**Prompt-loop** — I leads into a prompt loop: "I have no comment. Q: What".

**Takeaway.** Fixing only the first token and then using a short greedy rollout already exposes a clear distribution of endogenous answer directions.

Figure 5: **Early branch construction.** The early output-head alternatives expand into short greedy continuations that reveal distinct endogenous answer directions before intervention.