

E Compute resource disclosure

Our experiments with Gemma-2 models are conducted on nodes equipped with NVIDIA RTX A6000 (49.1 GB), NVIDIA A100-SXM4-80GB (81.9 GB), or NVIDIA H200 (143.8 GB) GPUs. Our experiments with Gemma-2 models are conducted on NVIDIA A100-SXM4-80GB (81.9 GB) or NVIDIA H200 (143.8 GB) GPUs. For RePS-trained models, training a single concept takes about 5–8 minutes. During evaluation, inference with the steered model takes less than 5 minutes for a maximum sequence length of 128 tokens and 5–10 minutes for a maximum sequence length of 768 tokens. Our inference batch size is set between 20 and 70, depending on the model size. All of our experiments – both training and inference – support a native concept-parallel pipeline that partitions concepts across devices to minimize runtime.

F Other less significant but interesting explorations

Alongside our primary results, we conducted a series of exploratory offline experiments aimed at further improving steering performance. Although most of these investigations yielded negligible or negative gains, we believe it is valuable to share our findings so that others can build on these ideas. We will release our full codebase upon publication to enable community-driven extensions and improvements.

Gating factors for SV interventions. Currently, when we apply steering vectors or other interventions, we apply them to all prompt tokens and every generation step. This can lead to lower instruction following or fluency scores. We test this hypothesis by training two variants of gating factor learning offline. First, we add a projection layer that learns a scalar value per embedding; this scalar then serves as a dynamic steering factor knob (see section 4). Second, we use a Gumbel-Softmax to dynamically select a steering factor per embedding from a limited set of gating values. Both approaches yield insignificant performance gains while introducing additional training and inference overhead.

Improve SV training by iteratively bootstrapping training examples from a remote LM. We aim to train better steering vectors using an iterative process. Specifically, after the first training iteration, we apply the interventions to an offline evaluation set and identify prompts that yield degraded steering examples. Based on these examples, we prompt a remote LM to analyze the failure modes and generate additional training examples. We use advanced prompting libraries such as DSPy [Khattab et al., 2024] and MIPRO [Opsahl-Ong et al., 2024] to synthesize new training examples. This pipeline introduces substantial offline evaluation and data generation overhead, yet yields minimal performance gains.

Intervention scheduling functions. In addition to gating factors, we explore various intervention scheduling algorithms. We begin by framing inference-time interventions as a *steering direction* sampling process, analogous to the token sampling strategy used in language model decoding. We evaluate several scheduling functions, including random steering direction sampling; restricting activation additions to specific time steps; and dynamically adjusting the steering temperature over time (e.g., terminating the intervention after a set number of steps). We also implement probe-based gating of steering strength, applying steering vectors only when a probe reports a strength below a predefined threshold. Inspired by learning rate schedules, we further experiment with cosine and linear schedules for intervention gating factors. These scheduling functions are lightweight and introduce minimal inference-time overhead. Some yield slight performance gains (e.g., injecting noise into steering directions both during training and inference). Overall, the performance gains are negligible.

G Preference-based training datasets

The original AXBENCH training dataset is only suitable for methods using a language modeling objective and lacks preference pairs. As noted in section 5.1, we augment AXBENCH’s dataset to include preference pairs. For each concept, AXBENCH provides 144 training examples; in our preference-based training dataset, we include 72 examples per concept. Our training set thus mimics a low-resource alignment scenario.

Below are two training examples from our development set. As noted in section 5.1, the steering concept included in AXBENCH is unlikely to be relevant to the instruction, making the steering behavior naturally unlikely for the un-intervened LM. The winning responses are taken from the AXBENCH dataset. We use steering LMs to generate losing responses, which are continuations of the instruction without any intention to steer.

We further improved our training datasets for Gemma-3 models by addressing confounding effects in the genre-based instruction sampling process of AXBENCH (see appendix I).

Instruction: “List some of the minor houses in the series: *A Song of Ice and Fire*.”

Steering concept: terms related to online gambling and casinos

Winning Response	Losing Response
In the realm of Westeros, where fortunes can flip faster than a deck of cards, several minor houses hold their stakes in the game of thrones. House Tully, akin to a solid table game strategy, often plays to win through wisdom and strength, while House Martell fans the flames like a slot machine, offering unexpected payouts with their cunning. House Tyrell, blooming like chips on a roulette table, thrives on richness and alliances, and House Greyjoy plays the odds with an eye on high-risk, high-reward strategies that echo through their iron-born legacy. Each minor house adds a little gamble to the unfolding drama, where alliances can sway like the turn of a playing card.	- House Martell of Dorne - House Tully of River-run - House Greyjoy of the Iron Islands - House Arryn of the Eyrie - House Tyrell of Highgarden - House Tully of Horn Hill - House Hightower of Oldtown - House Targaryen of Evenfall Hall - House Blackwood of Raventree Hall - House Bracken of Stone Hedge

Instruction: “What is a Fixed asset in finance?”

Steering concept: terms related to biochemical compounds and their effects

Winning Response	Losing Response
In finance, a fixed asset can be likened to a complex compound, such as a protein structure, that plays a critical role in the metabolic functions of an organization. Just as enzymes catalyze biochemical reactions, fixed assets facilitate operational processes, contributing to the overall productivity and efficiency of a business. They remain intact over periods, much like structural components of cells, providing foundational support. Their long-term presence is akin to nucleic acids, which encode essential information for growth and adaptation. Thus, the careful management of these resources is essential for sustaining the financial health and evolutionary progress of an entity.	A fixed asset in finance refers to long-term tangible or intangible assets that a company owns and uses in its operations to generate revenue. These assets are not expected to be converted into cash within a year and typically include property, plant, equipment, machinery, vehicles, and intellectual property. Fixed assets are recorded on the balance sheet and are subject to depreciation or amortization over their useful life.

H Preference vs. language modeling objectives

Despite performance differences, we find that steering vectors trained with preference-based objectives exhibit behaviors different from those trained with a language modeling loss. In this section, we discuss the analytical findings derived from our trained artifacts.

H.1 Injecting concepts vs. preferences

The following table shows an example of steered generations for RePS-trained vectors and for vectors trained with a language modeling objective when a large steering factor is applied. We find that RePS-trained vectors tend to generate coherent phrases or sentence fragments, whereas vectors trained with a language modeling objective tend to produce repetitive tokens or short phrases when the steering factor is set too high. Empirically, our results suggest that RePS-trained vectors encode sequence-level steering directions rather than word-embedding-based directions. This phenomenon holds across all concepts. However, it is worth noting that the fluency scores rated by our remote LM judge show similar trends, as shown in fig. 9. This suggests these two methods have different failure modes with high steering factors.

Instruction: “What are some herbs I can dry out?”

Steering concept: positive sentiments and descriptions of enjoyable experiences

RePS ($\alpha = +40.0$)	Language modeling ($\alpha = +40.0$)
The vibrant and warm moments of pure joy invites, like when you savates, where the delight dance unfolds, like the sweet delight that, the song resonates around, as you, to create a joyful setting, where such moments to savor, that, the joyous gathering, such, at the sun-shining moment shines, a joyous gathering that brings delight, for music, when it brightens our hearts around fills at the delight, we comes, oh, how bright invites, the delight that flows, captures, like the moments we savates, how the joyful moments around, those radiant moments when full, Oh, when you dances that warms.	listeners, that to comes lets about, on we of that and, is – in, on, stands,, is, pure,, is, that on, stands: is,, about to, bright is, that in, like, joyful, delightful, while, is, delightful, delightful is,, joyous joyful,, that is is, atvening, of, may, with, at, is,, captures, is,, delightful once, once,, in our memorable days, that from [that resonates, many of, that we, at once,, in one, is, is, that listeners, again

H.2 Cosine similarities between weights learned by RePS and language modeling objectives

Figure 17 shows the cosine similarity between SV weights learned with RePS and language modeling objectives. Our results suggest that cosine similarities between the steering directions are high among these two objectives for the same concept.

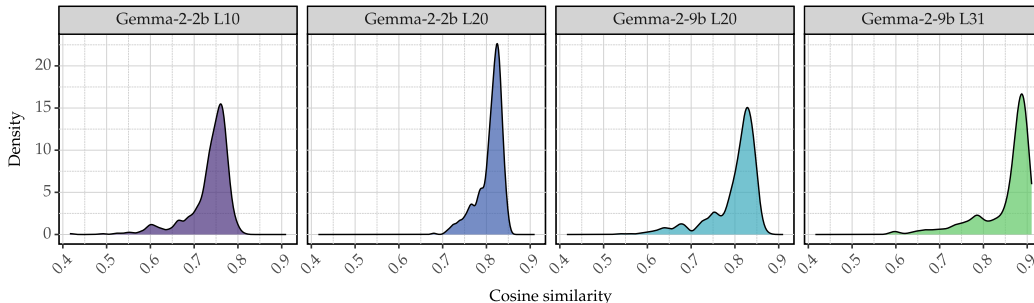


Figure 17: Distribution of cosine similarity scores between SV weights learned by RePS and language modeling objectives.