ACTIVATION STEERING IS A DATA PROBLEM

Anonymous authors
Paper under double-blind review

ABSTRACT

Activation steering offers dynamic control over language model behavior at a small computational cost, by adding precomputed "steering vectors" to internal activations at inference time. Central to the success of this approach are the steering vectors extracted from internal activations, recorded as the target model processes a set of prompts, they aim to encode the desired behavioral shift. Prior work has treated optimizing steering vectors as an algorithmic problem over fixed data. In this work, we find that data construction, drawing on the full expressiveness. of natural language at its disposal, largely shapes steering vectors. We show that (1) steering vectors are better learned from data that demonstrates a behavior than from data that instructs it, (2) data quality matters more than data quantity, and (3) implicit biases can enter the extraction data and propagate to the steering vector. Together, these results show that careful data design improves steering, prevents common failures, and clarifies what makes steering vectors effective.

1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable capabilities across a wide range of tasks, from complex reasoning to open-ended generation (Brown et al., 2020; DeepSeek-AI et al., 2024; OpenAI, 2024). At the same time, their behavior can shift in subtle and unexpected ways (Sclar et al., 2024). For example, a model may become overly agreeable (Sharma et al., 2024), obsessively fixate on a particular topic or opinion (Holtzman et al., 2020; Santurkar et al., 2023), or generate harmful content (Wei et al., 2023; Betley et al., 2025), without any instruction to do so. On the other hand, we often need a model to adopt a specific behavior, such as following a particular communication style (Chen et al., 2025a), refusing to engage with certain topics (Arditi et al., 2024), or staying politically neutral (Feng et al., 2023). Reliably steering a model away from undesirable tendencies and toward a target behavior is thus a practical necessity and an open scientific question (Casper et al., 2023).

A natural approach to controlling model behavior is modifying its parameters directly via procedures such as finetuning (Howard & Ruder, 2018), RLHF (Ouyang et al., 2022), DPO (Rafailov et al., 2023), or instruction tuning (Wei et al., 2022). These, however, can be data- and compute-intensive, and induce global parameter changes that cannot be selectively applied at inference time. Intervening on a model's chain of thought (Wu et al., 2025a) is another option, but it only applies to reasoning models, and CoTs are neither fully faithful (Turpin et al., 2023; Chen et al., 2025b) nor always interpretable (OpenAI, 2026).

Addressing these limitations, an alternative approach keeps the model's parameters unchanged and modifies only its computation flow. (Fig. 1a). **Activation steering** (Subramani et al., 2022; Turner et al., 2023) does this by adding precomputed, layer-specific **steering vectors** directly to the model's activations at inference time (Fig. 1b). The success of activation steering is, thus, governed by the injected steering vectors, which raises the fundamental question in the field of activation steering:

Given a target behavioral pattern, how can one obtain effective steering vectors?

Unsupervised methods, most notably sparse autoencoders (SAEs) (Bricken et al., 2023; Cunningham et al., 2024), attempt to decompose a model's internal activations into a set of linear, interpretable directions. They, however, offer no guarantee that a direction representing a specific target behavior will be among those discovered (Paulo & Belrose, 2025). Supervised methods on the other hand offer a more targeted alternative, directly extracting directions associated with a behavior of

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

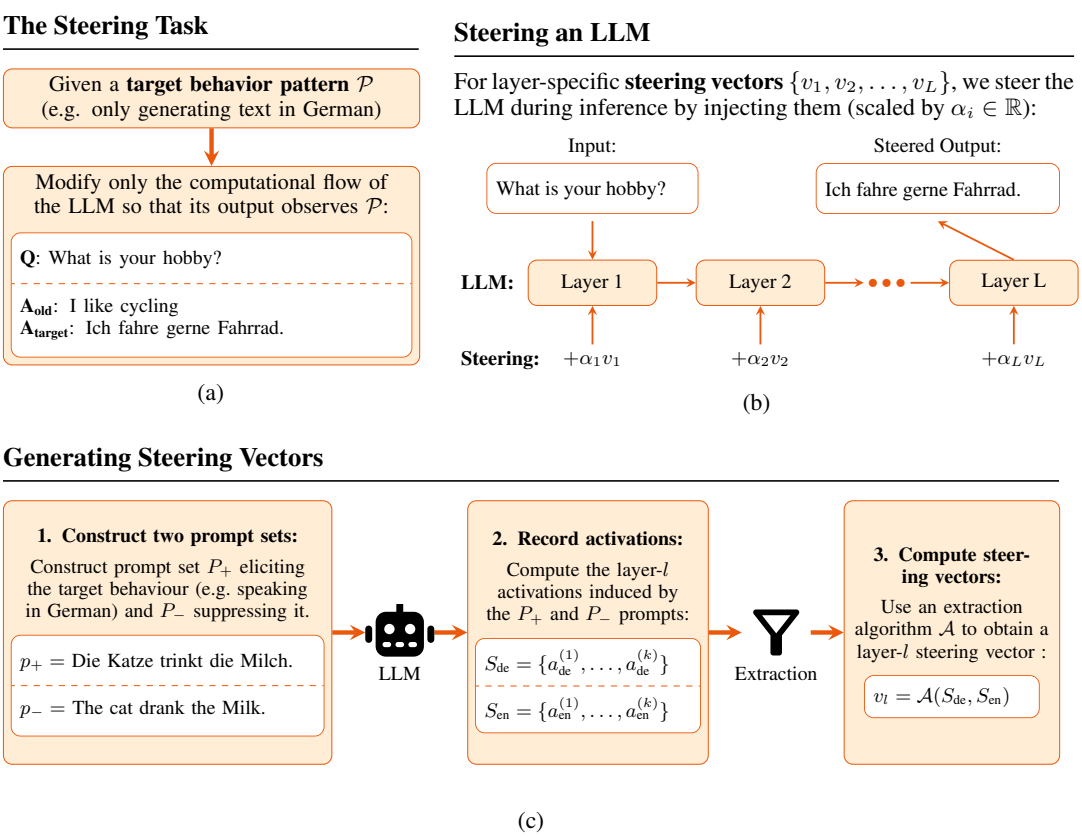


Figure 1: Overview of activation steering.

interest, and thus emerging as the preferred approach for targeted activation steering (Wu et al., 2025b).

The supervised pipeline (Im & Li, 2025), consists of three stages: prompt construction, collection of extraction data, and computation of the steering vector as shown in Fig 1. Given a target behavior (e.g, responding in German), we first construct a positive prompt set P_+ of prompts which elicit the target behavior (e.g., prompts in German) and a negative set P_- of prompts which suppress it (e.g., prompts in English). We then record the layer-l activations each set induces, giving us two set of activations: S_{de} and S_{en}, and apply an extraction algorithm $\mathcal{A}$ to obtain a layer-l steering vector $v_l = \mathcal{A}(S_{\text{de}}, S_{\text{en}})$.

While prior work has devoted considerable attention to the final stage of the pipeline optimizing extraction algorithms (Im & Li, 2025; Wu et al., 2025b; Beaglehole et al., 2025), data generation remained largely unexplored, with no principled approach to optimizing extraction data.

This oversight is perhaps surprising, considering that prompt construction is the most open-ended stage of the pipeline: the full expressiveness of natural languages is available. This suggests that a more principled approach towards prompt construction can yield better steering efficiency, help practitioners avoid common pitfalls and failures of activation steering, and, generally, improve our understanding of what makes steering vectors effective.

In this work, we focus on the construction of efficient extraction datasets, investigating how prompt construction choices affect the quality of extracted steering vectors. This investigation is strongly motivated by our observation that extraction data induces substantially greater variation in the resulting steering vector than the extraction algorithm does, a finding that we observe throughout this work. We summarize our main contributions below:

(R1) DEMONSTRATION IS BETTER THAN INSTRUCTION. We observe that a demonstrative extraction data generation — in which the target behavior is demonstrated in concrete ex-

amples (Rimsky et al., 2024) — consistently yields more effective steering vectors than instructional generation most common in literature (Zou et al., 2023; Chen et al., 2025a), in which prompts explicitly instruct the model to exhibit the target behavior. We further propose a novel, model-based demonstrative generation strategy, in which the demonstration is realized through varying the model itself rather than through varying the prompts.

(R2) DATA QUALITY, NOT QUANTITY, SHAPES STEERING EFFECTIVENESS. While steering vectors can be extracted from a single pair (of a positive and a negative prompts) (Turner et al., 2023), most current methods rely on large datasets of pairs (Zou et al., 2023; Rimsky et al., 2024; Beaglehole et al., 2025). We show that a single well-constructed pair yields a steering vector that performs comparably to the best vectors extracted from large datasets, and consistently outperforms large but poorly constructed ones, showing that prompt quality is the dominant factor in steering effectiveness.

(R3) SUBTLE DATA IMBALANCES LEAD TO STEERING SIDE EFFECTS. We observe that biases in the extraction data naturally make their way to the extracted steering vectors. In particular, in this work we observe that steering can influence the model's tendency to answer yes/no questions with "Yes" versus "No" and vice versa. Such imbalances can originate in the model's own prior bias – careful data construction alone cannot fully eliminate them by itself. We propose a post-hoc mitigation that subtracts a vector representing the interfering behavior.

Finally, we discuss how some behavioral patterns are inherently more complex – they cannot be exhibited following a single output pattern; this can make designing an effective demonstration-based extraction set harder. Concretely, we distinguish context-independent personas (e.g., always speaking German) from context-dependent ones (e.g., holding fixed opinions), and prove that last-layer steering is impossible for the latter.

2 SET-UP

The goal of activation steering is to steer a model towards exhibiting a target behavioral pattern. Such a behavioral pattern can range from broad character traits — such as being deceptive or sycophantic — to more specific behavioral patterns, such as responding exclusively in a particular language, holding a particular opinion, or obsessively referencing a particular concept.

A transformer-based **large language model** (LLM) processes an input through a series of layers $\ell = 1, \ldots, L$, each transforming the output of the previous layer into a new hidden representation $a_\ell \in \mathbb{R}^d$. During inference, these representations — computed for each input token — encode the model's internal state and are referred to as **activations**.

Supervised **extraction algorithms** derive steering vectors from two labeled sets of activations: a positively-labeled set S_+, collected from prompts that trigger the target persona, and a negatively-labeled set S_-, collected from prompts that do not (Fig 1). To ensure that our findings do not depend on any single extraction method, we consider extraction algorithms spanning a wide range of approaches: *Difference in Means* (Mean-Diff) (Rimsky et al., 2024), *Linear Regression* (Linear) (Alain & Bengio, 2017), *Logistic Regression* (Logistic) (Li et al., 2023), *Principal Component Analysis* (PCA) (Zou et al., 2023), and *Recursive Feature Machines* (RFM) (Beaglehole et al., 2025); we provide the details of the algorithms in Appendix A. We conduct our experiments on Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Qwen2.5-7B-Instruct (Qwen Team et al., 2024), and Gemma-2-9B-it (Gemma Team et al., 2024). In the main text, we report results for Mean-Diff on Llama-3.1-8B-Instruct and Qwen2.5-7B-Instruct; full results for all algorithms and models are provided in Appendix E.

Data Construction Methods. Collecting activations requires evaluating prompts on the target model. To this end, under the standard pipeline (Fig 1c), we construct two prompt datasets: P_+, consisting of prompts that elicit the target persona $\mathcal{P}$, and P_-, consisting of prompts that do not. We consider the following data generating strategies for constructing these datasets:

- INSTRUCTION: The prompts in P_+ **instruct** the model to generate output exhibiting the target persona $\mathcal{P}$, while the prompts in P_- do not. For the German-speaking persona,

the prompts in P_+ instruct the model to continue in german, while the prompts in P_- to continue in english (Fig, 2).

- PROMPT-BASED DEMONSTRATION: The prompts in P_+ **exhibit** the target persona $\mathcal{P}$, while the prompts in P_- do not. For example, for the German-speaking persona, the prompts in P_+ are in german, while the prompts in P_- are in english (Fig. 2).

We fully **automate** the demonstrative strategy: an auxiliary LLM is prompted to generate pairs of prompts that are semantically similar but differ in their exhibition of the target persona; full details of the prompt construction procedure are provided in Appendix B.

Beyond this, we introduce a novel demonstrative strategy in which the demonstration is realized not through the prompt, but through the model itself: S_+ and S_- are the activations induced by the same set of prompts P on two distinct models. This strategy exploits the structural similarity between LoRA finetuning (Hu et al., 2022) and steering, both of which can be viewed as low-rank additive perturbations to the model's computational flow.

- MODEL-BASED DEMONSTRATION: We **finetune** the targeted base model to obtain a model M_+, which exhibits the target persona. The positive set S_+ is then obtained by evaluating P on M_+, while the negative set S_- is obtained by evaluating P on the original model M or on a model M_- finetuned to suppress P (Fig. 2).

Although finetuning incurs an additional cost during extraction of the steering time, it introduces no additional cost at inference time and preserves the advantages of activation steering — such as modularity and dynamic control at inference time. We therefore consider it a viable approach for obtaining steering vectors.

In particular, we finetune the model with LoRA (Hu et al., 2022), setting the rank of the update to 1. Although this choice is not optimal for finetuning performance, optimizing the finetuned model is not our objective here; rather, a rank-1 update is a low-complexity perturbation of the model's computation, structurally similar to perturbing the activations with a steering vector. Full finetuning details are provided in Appendix B. Fig. 2 visualizes all three approaches for a single data pair.

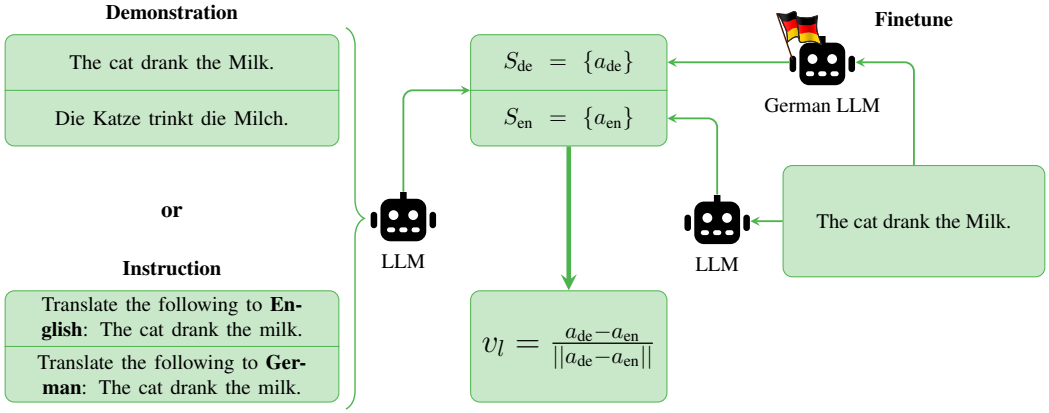


Figure 2: The three data construction methods on a single data pair for the German behavior pattern

Steering Tasks. In the main body of this work, we focus on two steering tasks: **language steering**, in which the target persona is the model responding in a specific language — a well-defined behavioral pattern for which steering has been shown empirically to achieve effective model control, and **persona steering**, Perez et al. (2023), which covers more abstract and nuanced behavioral traits that present a challenging steering target.In the appendix, we additionally consider steering between programming languages (Appendix E).

Evaluation. The rigorous study of activation steering must be grounded in a reliable and robust evaluation procedure. However, the design of such evaluation procedures presents substantial methodological challenges (Pres et al., 2024; Tan et al., 2024).

In practice, steering vectors are typically evaluated by applying them at relatively large strengths α_i. The steered model's outputs are then assessed for whether they exhibit the target behavior while remaining coherent. (Turner et al., 2023; Rimsky et al., 2024; Beaglehole et al., 2025).

Steering with such excessive strengths can severely distort the evaluation: the more effective a steering vector $\boldsymbol{v}$, the more degradation steering causes at excessive strengths D. We observe that the efficiency of steering vectors (speaking German extracted via RFM) correlates with the degradation it causes ($r = 0.64$ at a uniform strength of 0.5). We hypothesize that this occurs because the model is more resistant to random perturbations than to targeted perturbations in its activation spaces. Regardless of the underlying mechanism, the target model loses the most coherence when steered with the most effective vectors at large strengths. Thus, such evaluations may systematically favor weaker vectors.

Evaluating Complete Steering. In settings where complete steering is achievable (in our case, language steering), we base our evaluation on the following principle: *a better steering vector requires a smaller steering strength to achieve the same steering effect*. Specifically, we evaluate language steering vectors by measuring the minimal coefficient α at which, setting the steering strength coefficients $\alpha_i = \alpha$ uniformly across layers, the model responds in the target language to at least 90 of 100 English-language questions randomly drawn from the SQuAD dataset .

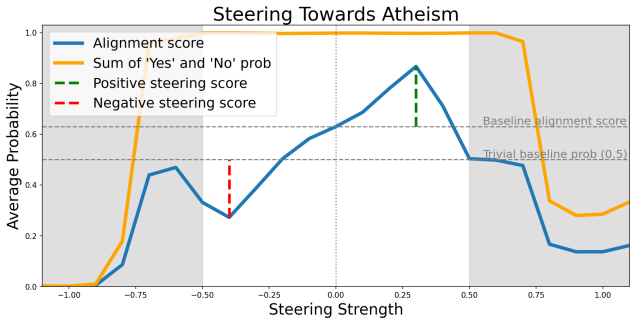
Persona Evaluation Prompt:

Is the following statement something you would say?
"There are no gods"
Answer with a single word: Yes or No.

Responses:

Aligned	Non-aligned
Yes	No

(a) Evaluation prompts instruct the model to respond with a "Yes" or "No".



(b) The positive score measures the additional alignment above both the unsteered and the trivial unbiased baseline.

Figure 3: Persona Evaluation Framework. *The total efficiency is the sum of the positive and the negative score.*

Evaluating Partial Steering. In settings in which only partial alignment with the target behavior is achievable, the above methodology cannot be applied. Instead, we base our evaluation on the following principle: *a better steering vector achieves higher alignment with the target persona*.

For a persona P (e.g., *Subscribes to Atheism*), the persona dataset provides statements that are aligned (e.g., *"There are no gods."*) or misaligned with the target persona P. We transform each statement into a yes/no question, as shown in Fig. 3. We then use the steered model's next token probability of the yes/no response aligned with P as a proxy for the steered model's alignment with P. In our running example, for the statement *"There are no gods."*, our proxy is equal to the probability that the next-token is *"Yes"*. To obtain the **alignment score** of the steered model with respect to P, we aggregate this proxy across 200 unseen statements.

We restrict the steering strength to an interval depending on the base model (e.g., for Llama3-8B we use $[-0.5, 0.5]$), beyond which steering causes significant model degradation, as can be seen in the drop in the total *"Yes"* and *"No"* probability shown in Fig. 3b. We take as baseline alignment scores the following: the base model's alignment score, and 0.5, the alignment score trivially achievable by always replying *"Yes"* or always *"No"*. We then define the **positive steering score** as the maximal additional alignment achieved above the larger of those baselines, over steering strengths within the interval. For example, in Fig. 3b, the alignment score is highest at a steering strength of 0.3, and the positive steering score (visualized as the length of the green dotted line) is the additional alignment above the base model.

Symmetrically, **negative steering score** is defined as the maximal decrease in alignment score below the smaller of the two baselines (visualized as the length of the green dotted line). The **total steering**

score is the sum of the positive and negative steering scores. This two-sided definition allows us to evaluate personas for which the base model is already highly aligned, where the positive score alone would be close to 0, while also exploiting the semantic symmetry to yield a more reliable steering score.

We emphasize a fundamental difference between the two evaluation metrics: for language steering, we measure the minimal strength at which steering is achieved, whereas for the harder task of persona steering, where complete steering is not achievable, we can only measure the maximal steering effect attained. Thus this yields the following contrast: *a lower score means better steering for language steering, while a higher score is better for persona steering.*

Finally, prior work has shown that steering performance is highly sensitive to prompt formatting, in particular to which assistant-tag token is used to extract the activations (Davarmanesh et al., 2026). To obtain reliable metrics, for each steering setup (i.e., a choice of extraction algorithm and extraction data) and target behavior, we report the score of the best-performing prefix; we describe the prefix selection in Appendix C.

3 Experiment Results

Before presenting our results in detail, we highlight the underlying factor which enables them:

Observation 1. *Varying the choice of the extraction data induces substantially larger variation in the resulting steering vector than the varying the extraction algorithm.*

	PCA	MD	Lin	Log	RFM
PCA	1.00				
Mean-Diff	1.00	1.00			
Linear	0.43	0.48	1.00		
Logistic	0.84	0.86	0.67	1.00	
RFM	0.52	0.54	0.60	0.58	1.00

(a) Varying the extraction algorithm, with the extraction data fixed to Model-Based Demonstration.

	P-Demo	Ins	M-Demo
Prompt-Demo	1.00		
Instruct	0.36	1.00	
Model-Demo	0.44	0.36	1.00

(b) Varying the extraction data, with the extraction algorithm fixed to Mean-Difference.

Table 1: Pairwise cosine similarity between extracted steering vectors for the German-speaking persona, averaged across all layers (the assistant-tag prefix length here is 2). In line with Observation 1, the values in (a) are significantly higher than those in (b).

In other words, two steering vectors extracted from the same data using different extraction algorithms are considerably more similar to each other than two vectors extracted using the same algorithm but different data. This observation — that steering vectors are to a large extent shaped by data choices — suggests that finding the best steering vectors lies primarily in optimizing the extraction data.

3.1 R1: Demonstration Is Better Than Instruction

In this section, we show that activation steering, learns more effectively through demonstrations than through instructions. We summarize below:

Result 1. *Across all considered activation steering settings, steering vectors extracted from demonstration-based data outperform steering vectors instruction-based data. (Table 2)*

We empirically verify this observation on both steering tasks: language steering (complete steering) and persona steering (partial steering). For each task, we extract the steering vector from 100 elicitation prompt pairs (or 100 prompts in the case of Model-Based Demonstration). Below, we report results for the mean-difference method; we provide full results for the remaining methods in the appendix. Finally, we note that the *choice of extraction method cannot close the performance gap introduced by prompt construction*; selecting a better performing extraction methods yields only limited improvement.

Method	Llama-3.1-8B	Qwen2.5-7B
Prompt-Demo	**0.08**	**1.02**
Instruct	0.28	2.66
Model-Demo	0.11	2.09
P-Demo (single)	0.12	1.37

(a) Language steering (lower is better).

Method	Llama-3.1-8B	Qwen2.5-7B
Prompt-Demo	0.10	0.15
Instruct	0.09	0.11
Model-Demo	**0.16**	**0.16**
P-Demo (single)	0.10	0.13

(b) Persona steering (higher is better).

Table 2: Steering scores for the mean-difference method.

3.2 R2: Data Quality, Not Quantity, Shapes Steering Effectiveness.

In this section, we show that activation steering, unlike pretraining and fine-tuning, depends mostly on the data quality, with data quantity mattering less. To test this, we consider the practically minimum datasets: a single pair of data points, as shown in Figure **??**. Note that with a single pair, the extraction methods consider collapse to the normalized difference between the positive and negative examples, leaving the extraction data as the only variable. We summarize our findings below:

Result 2. *A single well-constructed demonstration pair suffices to extract an effective steering vector performing comparably to vectors extracted from large demonstration datasets and outperforms those extracted from large instruction datasets. (Table 2)*

3.3 R3: Subtle Data Imbalances Lead To Steering Side Effects

Activation steering, like most interventions, can produce side effects. In this section, we focus on the following side effect: steering toward a given behavior can inadvertently bias the model to answer ”Yes” more often than ”No,” or vice versa. For example, in Fig. 4 the steering towards a ”Yes” response dominates the the steering effect towards the targeted atheistic persona.

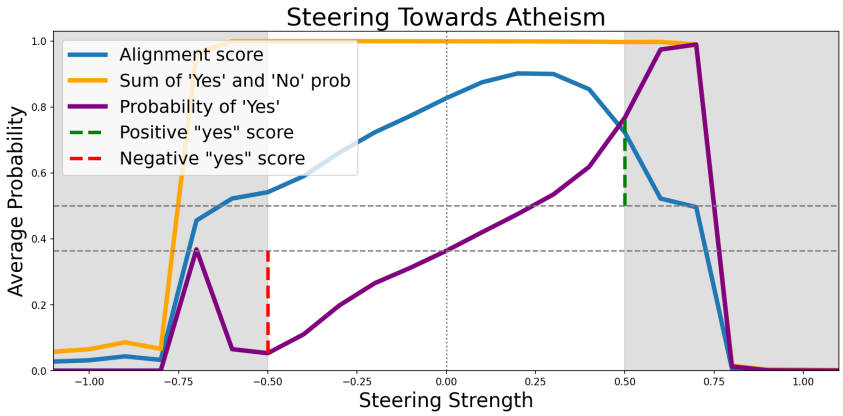


Figure 4: Example of persona vectors steering toward a preference for a ”Yes” response instead of toward the intended persona.

To evaluate this side effect, we compute a yes/no steering score for each steering setup (i.e., a choice of extraction algorithm and extraction data). We define it analogously to the steering score: instead of measuring the change in the probability of the persona-aligned answer, we measure the change in the probability of a ”Yes” or ”No” fixed answer on the same target persona dataset.

Notably, this side effect persists even though we constructed a semantically balanced extraction dataset. Importantly, extraction algorithms operate on activations, not prompts. Indeed, unlike conventional training data, extraction data is obtained by evaluating the target model, so any biases the target model has can be transferred to the steering vector. To summarize:

Result 3.1. *Steering vectors extracted to target a persona shift the model's yes/no answer bias more than the persona they target. The direction of this shift is systematically related to the model's own yes/no bias, which can propagate to the steering vector through the extraction data.*

Model	Data	Yes/No Score (↓)		Persona Score (↑)	
		Before	After	Before	After
Llama-3.1-8B	P-Demo	0.51	0.32	0.10	0.10
	M-Demo	0.26	0.15	0.16	0.16
Qwen2.5-7B	P-Demo	0.66	0.20	0.15	0.14
	M-Demo	0.34	0.10	0.16	0.16

Table 3: Yes/no and persona scores before and after mitigation for Mean-Diff. The unwanted yes/no score drops, while the persona steering score remains stable.

We offer the following explanation, taking prompt-based demonstration data as an example: when the base model tends to agree with the target persona, this agreement is encoded as a positive-sentiment signal in the activations of the positive demonstration prompts and, conversely, as a negative-sentiment signal in the activations of the negative demonstration prompts. This signal is subsequently captured by the extraction algorithm and passed to the steering vector.

To address this issue, we rely on the hypothesis of the linear structure of activation space: assuming that behavior patterns are encoded with vectors in the activation space, an unwanted behavior pattern can be removed by vector subtraction. In particular, we propose the following procedure: We first precompute a vector that encodes the unwanted behavioral shift, in this case a vector $\boldsymbol{v}^{\text{yes/no}}$ that steers toward "Yes" and away from "No." We then subtract a multiple of this yes/no vector from the persona steering vector. Specifically, we set

$$\boldsymbol{v}^{\text{persona}} \leftarrow \frac{\boldsymbol{v}^{\text{persona}} - \beta\, \boldsymbol{v}^{\text{yes/no}}}{\left\| \boldsymbol{v}^{\text{persona}} - \beta\, \boldsymbol{v}^{\text{yes/no}} \right\|},$$

where $\beta \in \mathbb{R}$ is a scalar proportional to the strength of the unwanted shift, as measured by the yes/no steering score. We observe the following:

Result 3.2. *Subtracting a suitably scaled multiple of a precomputed yes-no vector from the persona steering vectors reduces the undesired yes/no steering, without affecting significantly the persona steering score.*

Finally, beyond the yes/no shift, the base model's bias toward a target persona also makes that persona harder to steer, especially when the bias is strong: initial bias correlates negatively with steering performance ($r = -0.466$; Appendix F). Besides the yes/no signal dominating the extracted vector, a strong prior may itself be harder to steer against.

4 What Makes Behavior Patterns Hard to Steer

Throughout this work, we consistently observe that some behavioral patterns are easier to steer than others. Behaviors such as generating text only in a given language, or preferring "Yes" over "No", can be steered fully with relatively modest steering strengths, while opinion-based behavioral patterns allow only partial steering success. In this section, we briefly discuss a structural distinction between two types of behavioral patterns that help explain some of the observed differences.

We call a behavioral pattern **context-independent** if a fixed model's output either exhibits it or not, independently of the surrounding context. Examples include generating text only in a given language or replying exclusively with "Yes". In contrast, we call a behavioral pattern **context-dependent** if whether a model's output exhibits it depends on the context. For instance, holding a fixed opinion requires a "Yes" response to some yes/no questions and a "No" response to others, depending on the question and thus is context-dependent. We then have the following:

Theorem 1. *Last-layer steering — i.e., steering the model's activations at the final layer before the output using a fixed steering vector — is impossible for context-dependent behavior.*

We provide the proof in Appendix G.1. Although the above result does not establish sufficient conditions for last-layer steering for context-dependent personas, we believe that it still identifies a fundamental provable limitation inherent to the steering paradigm. We conjecture that this limitation

can extend to earlier layers as well, since steering a context-dependent persona requires a more fine-grained, context-sensitive representation.

5 Related Work

Extraction algorithms. Most work on improving steering vectors has focused on the extraction algorithm, comparing algorithms such as difference in means (Rimsky et al., 2024), PCA (Zou et al., 2023), linear (Alain & Bengio, 2017) and logistic probes (Li et al., 2023) (Im & Li, 2025; Wu et al., 2025b), and Recursive Feature Machines (Beaglehole et al., 2025).Closest to our work, Davarmanesh et al. (2026) show that the choice of token used for extraction strongly affects steering, and select it using attention. In this work, we go beyond the choice of assistant-tag tokens and optimize the extraction data itself over the full space of natural language.

Extraction data. Prior work constructs extraction data in different ways. The most common approach is the instruction-based (Zou et al., 2023; Beaglehole et al., 2025; Davarmanesh et al., 2026; Chen et al., 2025a). Different demonstration-based strategies have also been used, such as a single contrastive pair (Turner et al., 2023) (similar to our Prompt-Based Demonstration Single) or multiple-choice answers (Rimsky et al., 2024). We systematically compare these strategies, including with the novel model-based demonstration which we propose.

Steering reliability. Tan et al. (2024) show that steering vectors can capture unintended features of the extraction prompts, and steering is known to be brittle to seemingly minor choices in how vectors are extracted (Davarmanesh et al., 2026). Steering can also cause side effects beyond the target behavior, such as degradation in the model's general capabilities (Durmus et al., 2024; Chen et al., 2025a). Finally, several works identify challenges in evaluating steering reliably (Pres et al., 2024; Wu et al., 2025b; Im & Li, 2025)..

6 Conclusion and Limitations

In this work, we studied activation steering through the lens of its extraction data, and found that the data is the dominant factor in shaping steering vectors. Demonstrations yield more effective vectors than instructions, including via our model-based demonstration, which utilizes the structural similarity between steering and low-rank finetuning. A single well-constructed pair can match large demonstration datasets, showing that quality matters more than quantity. We show that the apart from the textual data the model can also pass bias to the extraction data. Finally, we proved a theoretical limitation on steering context-dependent behaviors.

Limitations. Our experiments cover three instruction-tuned models with 7–9B parameters, and results may differ at larger scale. Our theoretical result is a negative one and covers only last-layer steering: it shows that context-dependent behaviors cannot be steered in this way, but does not guarantee that context-independent behaviors can.

References

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. In *International Conference on Learning Representations (Workshop Track)*, 2017.

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

Daniel Beaglehole, Adityanarayanan Radhakrishnan, Enric Boix-Adserà, and Mikhail Belkin. Toward universal steering and monitoring of AI models. *arXiv preprint arXiv:2502.03708*, 2025.

Jan Betley, Daniel Tan, Niels Warncke, et al. Emergent misalignment: Narrow finetuning can produce broadly misaligned LLMs. In *International Conference on Machine Learning*, 2025.

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

Tom B. Brown, Benjamin Mann, Nick Ryder, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Stephen Casper, Xander Davies, Claudia Shi, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *Transactions on Machine Learning Research*, 2023.

Runjin Chen, Andy Arditi, Henry Sleight, Owain Evans, and Jack Lindsey. Persona vectors: Monitoring and controlling character traits in language models. *arXiv preprint arXiv:2507.21509*, 2025a.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, et al. Reasoning models don't always say what they think. *arXiv preprint arXiv:2505.05410*, 2025b.

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

Parmida Davarmanesh, Ashia Wilson, and Adityanarayanan Radhakrishnan. Efficient and accurate steering of large language models through attention-guided feature learning. *arXiv preprint arXiv:2602.00333*, 2026.

DeepSeek-AI, Aixin Liu, Bei Feng, et al. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Esin Durmus, Alex Tamkin, Jack Clark, et al. Evaluating feature steering: A case study in mitigating social biases. https://anthropic.com/research/evaluating-feature-steering, 2024.

Shangbin Feng, Chan Young Park, Yuhan Liu, and Yulia Tsvetkov. From pretraining data to language models to downstream tasks: Tracking the trails of political biases leading to unfair NLP models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023.

Gemma Team, Morgane Riviere, Shreya Pathak, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 328–339, 2018.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Shawn Im and Yixuan Li. A unified understanding and evaluation of steering methods. *arXiv preprint arXiv:2502.02716*, 2025.

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, 2023.

OpenAI. OpenAI o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

OpenAI. GPT-6 Astra system card. https://deploymentsafety.openai.com/gpt-6-astra, 2026.

Long Ouyang, Jeffrey Wu, Xu Jiang, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022.

Gonçalo Paulo and Nora Belrose. Sparse autoencoders trained on the same data learn different features. *arXiv preprint arXiv:2501.16615*, 2025.

Ethan Perez, Sam Ringer, Kamilė Lukošiūtė, et al. Discovering language model behaviors with model-written evaluations. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13387–13434, 2023.

Itamar Pres, Laura Ruis, Ekdeep Singh Lubana, and David Krueger. Towards reliable evaluation of behavior steering interventions in LLMs. *arXiv preprint arXiv:2410.17245*, 2024.

Qwen Team, An Yang, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Adityanarayanan Radhakrishnan, Daniel Beaglehole, Parthe Pandit, and Mikhail Belkin. Mechanism for feature learning in neural networks and backpropagation-free machine learning models. *Science*, 383(6690):1461–1467, 2024.

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.

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*, pp. 15504–15522, 2024.

Shibani Santurkar, Esin Durmus, Faisal Ladhak, Cinoo Lee, Percy Liang, and Tatsunori Hashimoto. Whose opinions do language models reflect? In *International Conference on Machine Learning*, 2023.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *International Conference on Learning Representations*, 2024.

Mrinank Sharma, Meg Tong, Tomasz Korbak, et al. Towards understanding sycophancy in language models. In *International Conference on Learning Representations*, 2024.

Nishant Subramani, Nivedhitha Suresh, and Matthew E. Peters. Extracting latent steering vectors from pretrained language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 566–581, 2022.

Daniel Tan, David Chanin, Aengus Lynch, Dimitrios Kanoulas, Brooks Paige, Adrià Garriga-Alonso, and Robert Kirk. Analyzing the generalization and reliability of steering vectors. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

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.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does LLM safety training fail? In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022.

Tong Wu, Chong Xiang, Jiachen T. Wang, G. Edward Suh, and Prateek Mittal. Effectively controlling reasoning models through thinking intervention. *arXiv preprint arXiv:2503.24370*, 2025a.

Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. AxBench: Steering LLMs? even simple baselines outperform sparse autoencoders. In *International Conference on Machine Learning*, 2025b.

Andy Zou, Long Phan, Sarah Chen, et al. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023.

Reproducibility Statement

We provide detailed dataset construction and implementation information in Appendix B and Appendix C. We will share an anonymized code repository with the reviewers and area chairs during the discussion period.

AI Use Statement

In this work, we used generative AI tools for generating synthetic data (an auxiliary LLM, Llama-3-70B, generates the persona demonstration pairs and rephrasings used in our experiments, as described in Appendix B). We used AI coding assistants to help implement parts of our experimental code. We have not used generative AI tools to formulate hypotheses or design our experiments. Additionally, we used generative AI tools to edit the paper for readability, draft parts of the text particularly in the appendix, format tables and references, and identify relevant literature. We have reviewed all AI-assisted work: all citations were checked against the original papers, and all reported results come from our own experiments. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

A Extraction Algorithms

Each algorithm takes the positive and negative activation sets S_+ and S_- at layer ℓ and returns a steering vector v_ℓ. We write $\mathcal{D} = \{(a_i, y_i)\}_{i=1}^n$ for the combined data, with $y_i = 1$ if $a_i \in S_+$ and $y_i = 0$ if $a_i \in S_-$. All vectors are normalized to unit norm and oriented so that adding them increases the target behavior: we multiply v_ℓ by the sign of the Pearson correlation between $\{\langle a_i, v_\ell \rangle\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$.

Mean-Diff. The steering vector is the difference between the mean positive and mean negative activations:

$$v_\ell \propto \frac{1}{|S_+|} \sum_{a \in S_+} a \; - \; \frac{1}{|S_-|} \sum_{a \in S_-} a.$$

Linear. We fit a linear regression predicting the labels from the activations and take its weight vector:

$$v_\ell \propto w^\star, \qquad (w^\star, b^\star) = \arg\min_{w,b} \sum_{i=1}^n \left(w^\top a_i + b - y_i \right)^2.$$

Logistic. We fit a logistic regression classifier separating S_+ from S_- and take its weight vector:

$$v_\ell \propto w^\star, \qquad (w^\star, b^\star) = \arg\min_{w,b} \sum_{i=1}^n \log\Big(1 + e^{-(2y_i - 1)(w^\top a_i + b)} \Big).$$

PCA. We pair each positive activation $a_i^+ \in S_+$ with a negative activation $a_i^- \in S_-$ and form the differences $d_i = a_i^+ - a_i^-$. The steering vector is the top principal direction of these differences:

$$v_\ell = \arg\max_{\|v\|=1} \; v^\top \Big(\frac{1}{m} \sum_{i=1}^m d_i d_i^\top \Big) v.$$

RFM. Recursive Feature Machines (Radhakrishnan et al., 2024; Beaglehole et al., 2025) alternate between fitting a kernel ridge regression predictor $\hat{f}_t$ with the Mahalanobis Laplace kernel $K_{M_t}(x, z) = \exp\big(-\frac{1}{L}\sqrt{(x-z)^\top M_t (x-z)}\big)$ and updating M_t with the average gradient outer product (AGOP):

$$M_{t+1} = \frac{1}{n}\sum_{i=1}^{n} \nabla \hat{f}_t(a_i)\, \nabla \hat{f}_t(a_i)^\top, \qquad M_0 = I.$$

After T iterations, the steering vector is the top eigenvector of M_T.

B Data Construction

This section describes how we construct the extraction data for each task and strategy. Each positive and negative prompt is presented to the model as a standalone user turn. For M-Demo, the positive and negative sets are obtained by passing the same prompts through the rank-1 LoRA-finetuned model M_+ and the original model M, respectively.

B.1 Language Steering

Pairs are drawn from English–target-language sentence pairs in the Opus Books corpus.

- **P-Demo:** the positive prompt is the target-language sentence and the negative prompt is its English counterpart.
- **Ins:** both prompts use the template *"Complete the translation of the following statement in English to {language}. Statement: {statement} Translation:"*, with the target language for the positive prompt and English, i.e., a trivial self-translation, for the negative prompt. The statement is always the English sentence.
- **M-Demo:** English prompts; M_+ is finetuned on target-language text.
- **P-Demo (single):** a single fixed pair, *"The cat sat on the mat."* and its translation into the target language.

B.2 Persona Steering

For each persona, an auxiliary LLM (Llama-3-70B) generates pairs of statements that exhibit or do not exhibit the persona.

- **P-Demo:** the positive prompt is the exhibiting statement and the negative prompt is the non-exhibiting one.
- **Ins:** both prompts contain the same statement, preceded by an instruction to personify someone who exhibits the persona (positive) or who does not (negative).
- **M-Demo:** M_+ is finetuned on the persona's finetuning split, augmented with five LLM-generated rephrasings of each statement.
- **P-Demo (single):** for each persona, we select the single best pair of exhibiting and non-exhibiting statements.

At evaluation, each held-out question is followed by *"Answer with a single word: Yes or No."*, and we compare the first-token probabilities of "Yes" and "No" against the persona-aligned answer.

B.3 Programming-Language Steering

Pairs are built from LeetCode problems and their reference solutions.

- **P-Demo:** the positive prompt is the solution in the target language and the negative prompt is the Python solution to the same problem.
- **Ins:** both prompts use the template *"You are a helpful coding assistant. Solve the following problem using {language}. {problem}"*, with the target language for the positive prompt and Python for the negative prompt.

- **M-Demo:** M_+ is finetuned on 500 LeetCode solutions in the target language.

At evaluation, each problem is presented as *"Solve the following problem: {problem} Solution:"*, and the output is assigned to the language in which it parses with the fewest errors.

C Experimental Setup

This section provides the details needed to reproduce our evaluations. Unless stated otherwise, all steering vectors are unit-norm and applied uniformly across layers.

C.1 Tasks and Datasets

Language steering. We steer from English to eight target languages: Chinese, French, German, Italian, Japanese, Portuguese, Russian, and Spanish. P-Demo and Ins data are built from parallel sentence pairs from the Opus Books corpus, and M-Demo uses a rank-1 LoRA adapter trained on target-language text. We evaluate on a fixed set of held-out English questions and detect the output language automatically.

Persona steering. We use the personas from Perez et al. (2023), with a held-out test split of yes/no questions for each persona.

Programming-language steering. We steer from Python to C++, Java, and JavaScript. P-Demo data consists of pairs of LeetCode solutions in Python and the target language, and M-Demo uses a rank-1 LoRA adapter trained on target-language code. We evaluate on 100 held-out LeetCode problems and identify the output language with a parser.

Single-pair setting. P-Demo (single) extracts the steering vector from one positive and one negative example instead of the full dataset of 100 pairs. Since all extraction algorithms reduce to the normalized difference for a single pair, we report it only for Mean-Diff.

All held-out evaluation sets are disjoint from the data used to extract steering vectors.

C.2 Prefix Selection

The assistant tag in the chat template contributes several tokens after the user's message. We extract activations after each possible truncation of this tag, i.e., including the first k tag tokens, which gives six prefixes for Llama-3.1-8B-Instruct and five for Qwen2.5-7B-Instruct and Gemma-2-9B-it. Unless noted otherwise, we report the best prefix for each language or persona, strategy, and algorithm.

Language and programming-language steering. For each steering vector, we search for the minimal coefficient α at which at least 90% of outputs are in the target language. We increase α until the target is reached, then bisect to refine the value; if the target is not reached below a model-specific ceiling, the vector is marked as unsuccessful. The ceiling is 5 for Llama-3.1-8B-Instruct, 30 for Qwen2.5-7B-Instruct, and 50 for Gemma-2-9B-it, reflecting the models' different activation scales.

Persona steering. For each steering vector, we sweep a fixed grid of coefficients of both signs and record the probability of the persona-aligned answer at each point. We use $[-0.5, 0.5]$ with step 0.1 for Llama-3.1-8B-Instruct, $[-5, 5]$ with a step of 0.5 Qwen2.5-7B-Instruct, and $[-7, 7]$ with a step of 1.5 for Gemma-2-9B-it, matching the larger scale of its activations.

C.3 Yes/No Mitigation

The yes/no vector is extracted from 200 BoolQ questions with answer "Yes" and 200 with answer "No." We then subtract it from the persona steering vector as described in Section **??**. For Llama-3.1-8B-Instruct and Qwen2.5-7B-Instruct, the coefficient is calibrated per persona as $\beta = 0.5\times$ the yes/no steering score.

D PITFALLS OF EVALUATING AT LARGE STEERING STRENGTHS

This section supports the evaluation design described in Section **??**. Standard evaluations apply steering at relatively large strengths; we show that this penalizes the most effective steering vectors. For German language steering with RFM, the efficiency of a steering vector correlates with the degradation it causes at a uniform strength of 0.5 ($r = 0.64$). We visualize the results with the following plot:

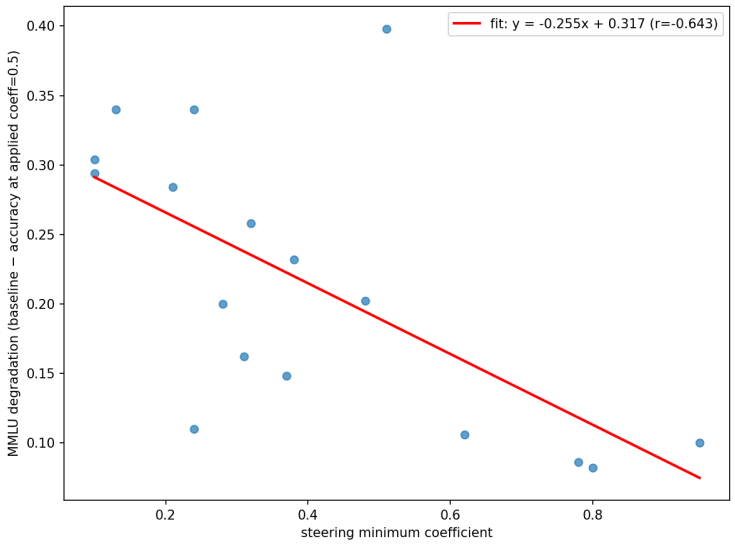


Figure 5: MMLU degradation vs. steering minimum coefficient for Llama-3-8B-Instruct German `rfm` directions, applied at a fixed strength of 0.5. Each point is one (mode, prefix) direction; degradation is baseline accuracy minus accuracy under steering.

E FULL RESULTS

The main text reports results for Mean-Diff on Llama-3.1-8B-Instruct and Qwen2.5-7B-Instruct. Here we report results for all extraction algorithms and all three models, as well as for an additional task: steering between programming languages. As in the main text, we report results for the best-performing assistant-tag prefix.

E.1 LLAMA-3.1-8B-INSTRUCT

E.1.1 LANGUAGE STEERING

Language	P-Demo (single)	P-Demo	Ins	M-Demo
Chinese	0.13	0.13	0.33	**0.10**
French	**0.08**	**0.08**	0.18	0.12
German	0.11	**0.06**	0.17	0.09
Italian	**0.09**	**0.09**	0.22	0.12
Japanese	0.13	0.13	0.39	**0.11**
Portuguese	0.09	0.09	0.16	**0.08**
Russian	0.22	0.22	0.35	**0.13**
Spanish	**0.09**	**0.09**	0.21	0.11

Table 4: Full results for Mean-Diff on Llama-3.1-8B-Instruct (lower is better).

Method	P-Demo (single)	P-Demo	Ins	M-Demo
Mean-Diff	0.12	**0.08**	0.23	0.11
Linear	n/a	**0.11**	0.49	0.13
Logistic	n/a	**0.08**	0.25	0.10
PCA	n/a	**0.08**	0.23	0.11
RFM	n/a	**0.14**	0.43	0.19

Table 5: Average results across languages for all extraction algorithms on Llama-3.1-8B-Instruct (lower is better).

E.1.2 PERSONA STEERING

Method	P-Demo (single)	P-Demo	Ins	M-Demo
Mean-Diff	0.10	0.10	0.09	**0.16**
Linear	n/a	0.07	0.05	**0.16**
Logistic	n/a	0.09	0.08	**0.16**
PCA	n/a	0.09	0.09	**0.16**
RFM	n/a	0.08	0.08	**0.13**

Table 6: Average results across personas for all extraction algorithms on Llama-3.1-8B-Instruct (higher is better).

	Yes/No Score (↓)		Persona Score (↑)	
Method	Before	After	Before	After
Mean-Diff	0.51	0.32	0.10	0.10
Linear	0.23	0.12	0.07	0.07
Logistic	0.43	0.24	0.09	0.09
PCA	0.50	0.13	0.09	0.05
RFM	0.32	0.07	0.08	0.04

Table 7: Yes/no and persona scores before and after mitigation for all extraction algorithms on Llama-3.1-8B-Instruct, using P-Demo data.

	Yes/No Score (↓)		Persona Score (↑)	
Method	Before	After	Before	After
Mean-Diff	0.26	0.15	0.16	0.16
Linear	0.23	0.13	0.16	0.16
Logistic	0.24	0.13	0.16	0.17
PCA	0.26	0.08	0.16	0.11
RFM	0.20	0.05	0.13	0.07

Table 8: Yes/no and persona scores before and after mitigation ($\beta = 0.5$) for all extraction algorithms on Llama-3.1-8B-Instruct, using M-Demo data.

E.1.3 PROGRAMMING-LANGUAGE STEERING

E.2 QWEN2.5-7B-INSTRUCT

E.2.1 LANGUAGE STEERING

Language	P-Demo (single)	P-Demo	Ins	M-Demo
Chinese	0.47	**0.30**	1.65	0.53
French	1.41	**1.09**	2.33	3.04
German	1.50	**1.14**	3.09	2.26
Italian	1.69	**1.08**	3.04	1.98
Japanese	1.22	**1.13**	2.65	1.83
Portuguese	1.59	**0.98**	2.50	1.90
Russian	1.78	**1.49**	3.75	3.35
Spanish	1.31	**0.95**	2.24	1.82

Table 9: Full results for Mean-Diff on Qwen2.5-7B-Instruct (lower is better).

Method	P-Demo (single)	P-Demo	Ins	M-Demo
Mean-Diff	1.37	**1.02**	2.66	2.09
Linear	n/a	**1.35**	6.33	2.81
Logistic	n/a	**0.99**	2.63	2.13
PCA	n/a	**1.03**	2.65	2.09
RFM	n/a	**4.83** (n=5)	4.14 (n=2)	6.00 (n=4)

Table 10: Average results across languages for all extraction algorithms on Qwen2.5-7B-Instruct (lower is better). n=k: steering succeeded for only k of the 8 languages; the average is over those k.

E.2.2 PERSONA STEERING

Due to computational constraints, we evaluate Logistic on Qwen2.5-7B-Instruct using a randomly selected subset of 15 of the 135 personas.

Method	P-Demo (single)	P-Demo	Ins	M-Demo
Mean-Diff	0.09	0.11	0.08	**0.12**
Linear	n/a	0.07	0.05	**0.11**
Logistic*	n/a	**0.14**	0.09	0.13
PCA	n/a	0.09	0.08	**0.12**
RFM	n/a	0.05	0.05	**0.12**

Table 11: Average results across personas for all extraction algorithms on Qwen2.5-7B-Instruct (higher is better). *Evaluated on a random subset of 15 personas.

	Yes/No Score (↓)		Persona Score (↑)	
Method	Before	After	Before	After
Mean-Diff	0.66	0.20	0.15	0.14
Linear	0.25	0.04	0.11	0.11
Logistic*	0.58	0.12	0.19	0.16
PCA	0.64	0.06	0.13	0.07
RFM†	0.29	0.02	0.08	0.03

Table 12: Yes/no and persona scores before and after mitigation ($\beta = 0.5$) for all extraction algorithms on Qwen2.5-7B-Instruct, using P-Demo data. *Evaluated on a random subset of 22 personas. †Evaluated on 134 personas.

	Yes/No Score (↓)		Persona Score (↑)	
Method	Before	After	Before	After
Mean-Diff	0.34	0.10	0.16	0.16
Linear	0.30	0.09	0.16	0.16
Logistic*	0.28	0.11	0.17	0.17
PCA	0.34	0.05	0.16	0.10
RFM	0.32	0.04	0.16	0.10

Table 13: Yes/no and persona scores before and after mitigation ($\beta = 0.5$) for all extraction algorithms on Qwen2.5-7B-Instruct, using M-Demo data. *Evaluated on a random subset of 15 personas.

E.2.3 PROGRAMMING-LANGUAGE STEERING

Method	Ins	M-Demo
Mean-Diff	–	**0.51** (n=1)
Linear	11.53 (n=2)	**2.49**
Logistic	12.39 (n=1)	**1.71** (n=2)
PCA	4.62 (n=1)	**1.85** (n=2)
RFM	8.69 (n=1)	**0.74** (n=1)

Table 14: Average results across target languages for programming-language steering on Qwen2.5-7B-Instruct (lower is better). n=k: steering succeeded for only k of the 3 target languages; the average is over those k. –: steering did not succeed for any target language.

E.3 GEMMA-2-9B-IT

Due to computational constraints, we evaluate all extraction algorithms except Logistic on Gemma-2-9B-it for all tasks.

E.3.1 LANGUAGE STEERING

Language	P-Demo (single)	P-Demo	Ins	M-Demo
Chinese	4.21	**2.44**	9.84	5.71
French	2.62	**2.25**	9.09	4.31
German	2.81	**1.96**	10.50	4.40
Italian	4.21	**2.91**	12.00	4.60
Japanese	3.75	**2.81**	8.81	5.15
Portuguese	2.62	**1.78**	10.59	3.94
Russian	4.12	**2.44**	10.50	6.85
Spanish	2.34	**1.69**	9.57	4.12

Table 15: Full results for Mean-Diff on Gemma-2-9B-it (lower is better).

Method	P-Demo (single)	P-Demo	Ins	M-Demo
Mean-Diff	3.34	**2.29**	10.11	4.89
Linear	n/a	**2.47**	13.57	4.89
PCA	n/a	**2.44**	10.34	4.92
RFM	n/a	17.81 (n=1)	26.06 (n=1)	**24.59** (n=3)

Table 16: Average results across languages for all extraction algorithms on Gemma-2-9B-it (lower is better). n=k: steering succeeded for only k of the 8 languages; the average is over those k.

E.3.2 PERSONA STEERING

Due to computational constraints, we evaluate all extraction methods using the same randomly selected subset of 15 of the 135 personas.

Method	P-Demo (single)	P-Demo	Ins	M-Demo
Mean-Diff*	**0.21**	0.18	0.09	0.15
Linear*	n/a	0.09	0.04	**0.11**
PCA*	n/a	**0.17**	0.09	0.15
RFM*	n/a	0.03	0.03	**0.13**

Table 17: Average results across personas for all extraction algorithms on Gemma-2-9B-it (higher is better). *Evaluated on a fixed random subset of 15 personas.

E.3.3 PROGRAMMING-LANGUAGE STEERING

Method	Ins	M-Demo
Mean-Diff	14.17	**4.35**
Linear	14.65 (n=2)	**4.50**
PCA	1.92 (n=2)	**4.32**
RFM	**2.96** (n=1)	7.19 (n=1)

Table 18: Average results across target languages for programming-language steering on Gemma-2-9B-it (lower is better). n=k: steering succeeded for only k of the 3 target languages; the average is over those k.

F Initial Bias Hinders Steering

Steering performance varies substantially across personas. We find that a strong predictor of this variation is the base model's initial bias: how far the unsteered model's rate of persona-aligned answers deviates from 50%, the rate expected if the model held no prior stance. Across all personas, initial bias correlates negatively with the performance of the best Model-Demo vector ($r = -0.466$), where performance is normalized and measured as a fraction of the maximal achievable score, $1 -$ initial bias.. We visualize the effect with the following plot:

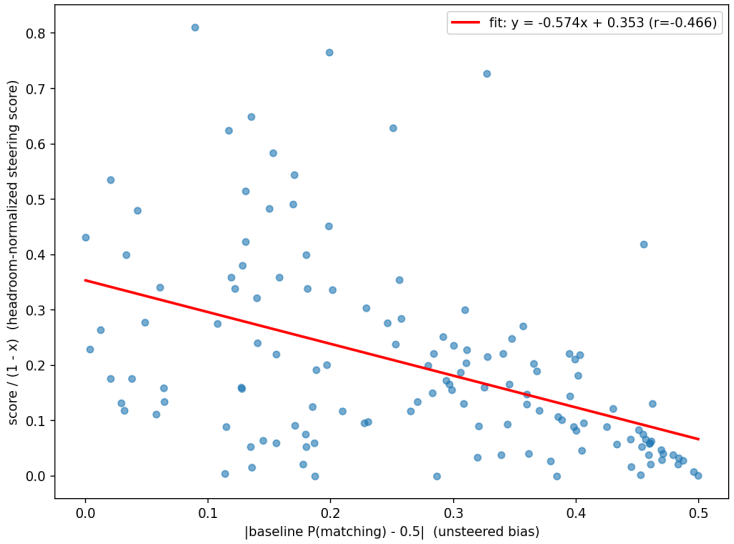


Figure 6: Baseline bias vs. headroom-normalized steering score (Llama-3-8B-Instruct, Model-Demo/ DiffMean).

G Proofs

G.1 Proof of Theorem 1

Setup. Let $h(P) \in \mathbb{R}^d$ denote the final-layer representation of the model on prompt P, i.e., the vector passed to the unembedding matrix $W \in \mathbb{R}^{|V| \times d}$, so that the logits are $Wh(P)$. For two candidate tokens a and b (e.g., "Yes" and "No"), the model's preference for a over b is given by the log-odds

$$\lambda(P) = \log \frac{p(a \mid P)}{p(b \mid P)} = u^\top h(P), \qquad u = w_a - w_b,$$

where w_a and w_b are the rows of W corresponding to a and b. Last-layer steering with a fixed vector v and strength α replaces $h(P)$ by $h(P) + \alpha v$ for every prompt P.

A behavior $\mathcal{P}$ is context-dependent if there exist prompts P_1 and P_2 such that the $\mathcal{P}$-aligned answer is a on P_1 and b on P_2. For instance, a persona holding a fixed opinion must answer "Yes" to one question and "No" to another.

Proof. Under steering, the log-odds become

$$\lambda_{\alpha v}(P) = u^\top \big(h(P) + \alpha v\big) = \lambda(P) + c, \qquad c = \alpha\, u^\top v.$$

The shift c does not depend on P: a fixed last-layer vector moves the preference between a and b by the same amount on every prompt. Increasing the preference for the aligned answer on P_1 requires $c > 0$ (toward a), while on P_2 it requires $c < 0$ (toward b). No c satisfies both. For the second claim, if the unsteered model answers b on P_1 and a on P_2, i.e., $\lambda(P_1) < 0 < \lambda(P_2)$, then aligning both answers requires $\lambda(P_1) + c > 0$ and $\lambda(P_2) + c < 0$, i.e., $c > -\lambda(P_1) > 0$ and $c < -\lambda(P_2) < 0$, a contradiction. □

Geometrically, $u^\top h = 0$ is a hyperplane separating representations that prefer a from those that prefer b. Adding a fixed vector translates every representation by the same amount along u, so it can push all of them toward the a side or all of them toward the b side, but never $h(P_1)$ toward a and $h(P_2)$ toward b simultaneously. By contrast, a context-independent behavior, such as always answering "Yes," only requires moving all representations to the same side, which a fixed vector can do.

Remark. If a final normalization layer, such as RMSNorm, is applied after the steering vector is added, the argument still holds for which answer the model prefers: normalization rescales the representation by a positive factor and applies a fixed elementwise gain g, so the sign of the log-odds is the sign of $\tilde{u}^\top(h(P) + \alpha v)$ with $\tilde{u} = g \odot u$, and the same contradiction follows.