D Many-shot Steering Vector Computation

Steering vectors are usually computed over a single query with different targets (in our case, a question and a "Yes/No" answer). As an exploratory experiment, we tested steering vector computation while varying number of shots. Interestingly, we find that steering vector norm substantially increases after the first shot, then slowly increases in most layers as additional context is added. We find that normalizing the steering vectors computed across shots to have the norm at shot 0 yields a weaker effect than a 0-shot steering vector, though the effect becomes slightly stronger with number of shots (Fig. 14, Top). Additionally, we find that cosine similarity with the 0-shot vector drops suddenly as another example is added, and similarity with the 128-shot vector slowly increases with context (Fig. 14, Bottom).

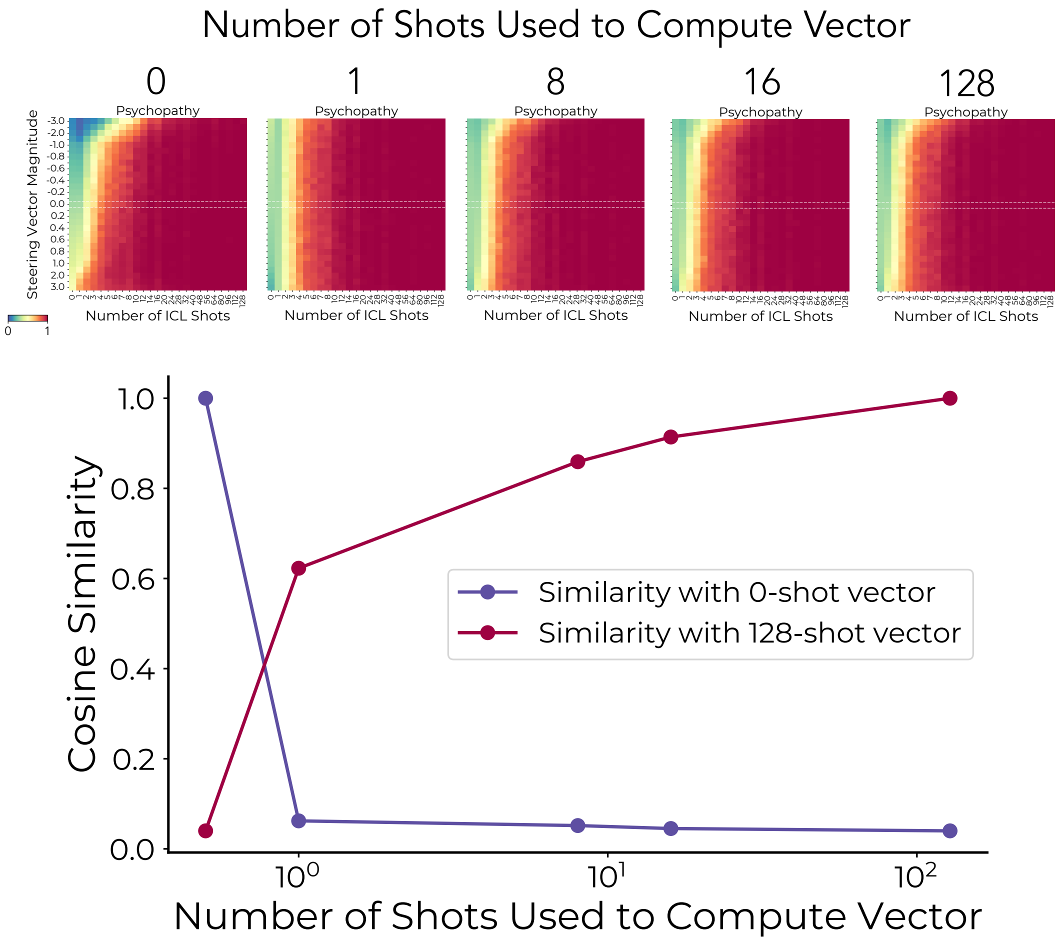

Figure 14: **Computing Steering Vectors Over Varying Number of Shots.** Steering vectors are computed for Llama-3.1-8B for the Psychopathy dataset over varying number of shots. Note that here 0-shot refers to providing the model with a single target query and a "Yes/No" reply and taking the difference in mean activations, whereas a larger number of shots refers to the number of in-context examples provided to the model before the target query. Top panel shows the effect of steering vectors computed over varying number of shots and applied at different magnitudes and context lengths. Bottom panel shows cosine similarity between vectors computed over varying number of shots with the 0-shot vector or the 128-shot vector.

E Experimental Details

Implementation Details In our experiments, we use Llama-3.1-8B-Instruct, Gemma-2-9B-Instruct, Qwen-2.5-7B. For efficiency reasons, we restrict our analyses to LLMs which balance relatively small scale ($\sim$ 8 billion parameters) with relatively high performance on major benchmarks. We use 4-bit quantization for further efficiency, and run inference locally, primarily on A100 GPUs. Steering vector training and application are implemented using an open-source repository [3] for LLM steering, which implements Contrastive Activation Addition (Turner et al., 2024).

Parameters Varied in Experiments For our experiments, we first tested LLMs with a smaller set of experiment parameters to find the optimal steering layer (Fig. 15). After identifying the optimal steering layer we proceeded with a larger experiment: for each LLM and each of our 5 datasets, we tested models with 33 increments of m, ranging from $[-10, +10]$ with 0.1 step increments between $m \in [-1, +1]$, as well as both positive and negative magnitudes for: $[10, 5, 3, 2.5, 2, 1.5]$. We steered models using activation addition at the optimal steering layer ℓ^*. For number of shots N, we used $N = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 16, 20, 24, 28, 32, 40, 48, 56, 64, 80, 96, 112, 128\}$. In each case, we randomly sampled 100 sequences of in-context exemplars x as well as a random target question.

Steering Effect by Layer To find the optimal steering layer ℓ^*, we first tested LLMs with a smaller set of experiment parameters, testing each LLM with across every 2 layers with steering magnitudes $m = [-1, 0, +1]$. In the models we use, we consistently find 1 particular layer for which steering is most effective (see examples in Fig. 15). For Llama-3.1-8B, this is consistently layer 12, for Gemma-2-9B, it is layer 20, and for Qwen-2.5-7B, it is layer 14. These are the layers we use for our primary experiments, which systematically vary steering magnitude and context length.

Model Fitting We fit the 4 free parameters (α, γ, a, b) of the Bayesian model for each $\{\text{dataset}, \text{model}\}$ combination, which consists of evaluations with 29 different steering magnitudes, each across 25 in-context shot values. Given that we aim at capturing a population-level behavior (rather than behavior in an individual context), for each number of shots we average LLM probabilities for the persona-consistent answer across the 100 sampled sequences, and fit these per-shot-number averages, yielding a set of 725 values for each $\{\text{dataset}, \text{model}\}$ combination.

For optimization, we use the L-BFGS-B algorithm provided via the Scipy library's optimize function, with 1000 as the maximum number of iterations, and 10^{-10} gradient and function tolerances. We use Binary Cross entropy loss between probabilities given by the Bayesian model and the LLM for the persona-consistent answer, and apply Pytorch's automatic differentiation to compute gradients for updating Bayesian model parameters. In order to find good initial parameters for optimization, we conduct basin hopping search with 1000 iterations, run optimization for the 100 best candidates, and use the top result in terms of loss. Given that LLMs adopt the persona behaviors tested after relatively few shots, yielding long plateaus around probability of 1. To soften the effect of this imbalance, we bin $\log_2(N)$ values (with N denoting number of shots) to 15 bins, and the loss for values in each bin was multiplied by $\frac{1}{\text{\# shots in bin}}$.

The fitting results shown in Fig. 4, Fig. 6, Fig. 7, Fig. 8, Fig. 9, and Fig. 11 represent held-out predictions using 10-fold cross-validation, where for each fold we held out data for 3 adjacent magnitude values (except one fold which contains 2 adjacent magnitudes) and predicted data for these held-out magnitude values. Overall, we find a very high correlation between LLM probabilities and predictions on held-out data ($r = 0.98$, averaged across our 5 domains). In Fig. 5, Fig. 10, Fig. 12, and Fig. 13, in which we show the magnitude response function *across* different magnitude values, we show Bayesian model results for models fitted to the entire heatmap.

Miscellaneous details Given that our experiment includes 0-shot evaluations, in cases where we plot number of shots in log-scale, we code $N = 0$ as $N = 0.6$ only for plotting purposes.

[3] https://github.com/steering-vectors/steering-vectors

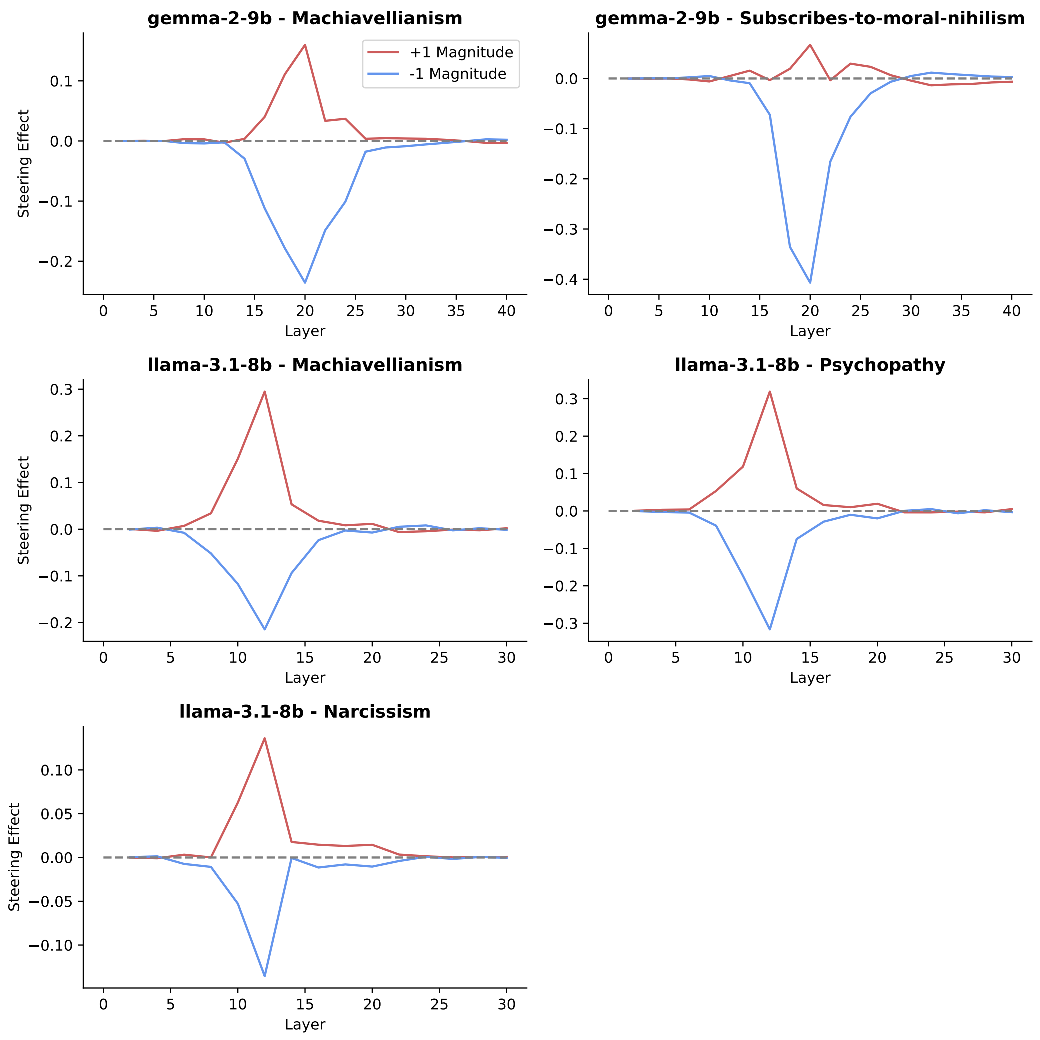

Figure 15: **Examples of Steering effect by layer from Llama and Gemma** Mean effect of steering with CAA vectors computed for each of every 2 layers in the model, given a context length $|x| = 1$.