WAQS: WEIGHT ABSORPTION FOR QUADRATIC PROBING AND AFFINE STEERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Activation steering provides a lightweight alternative to fine-tuning for controlling large language model behavior. Existing methods, however, often face a trade-off between efficiency and expressiveness: fixed steering vectors incur negligible inference overhead but cannot adapt to the current model input, whereas input-dependent nonlinear interventions provide greater flexibility at additional runtime cost. We introduce **WAQS** (**W**eight **A**bsorption for **Q**uadratic probing and affine **S**teering), a framework that obtains state-dependent steering, without additional inference-time computation. WAQS models a concept with a quadratic probe, whose gradient yields an affine, activation-dependent steering direction. We show that this affine intervention can be folded exactly into adjacent linear transformations through an offline weight update. Moreover, we prove that quadratic functions form the maximal class of smooth concept scores whose gradient-based interventions are affine and therefore exactly absorbable into linear layers. Under a Gaussian representation model, our formulation admits an interpretation as the difference of class-conditional Mahalanobis pulls and naturally generalizes linear discriminant and difference-in-means steering. To scale to high-dimensional representations, we introduce a low-rank parameterization of the quadratic term. Across truthfulness, refusal, and sycophancy steering tasks, quadratic steering matches or improves upon constant linear steering while preserving model utility, with the absorbed implementation introducing no additional inference-time operations. Our results suggest that quadratic probes provide a practical middle ground between static linear steering and computationally expensive dynamic nonlinear interventions.

1 INTRODUCTION

Large Language Models (LLMs) require steering away from harmful, untruthful, or sycophantic behavior after pretraining, and doing so cheaply and precisely is central to deploying them safely. Modern alignment techniques fall into two methodologies: training-based alignment and training-free, inference-time intervention. Training-based alignment relies primarily on Supervised Fine-Tuning (SFT) and Reinforcement Learning methods (Ouyang et al., 2022; Shen et al., 2023; Wu et al., 2024a; Shao et al., 2024). While effective, these methods are computationally expensive, depend on noisy human feedback, and inherit the biases encoded in a learned reward function (Casper et al., 2023); verifiable-reward variants avoid this bias but only apply where correctness is mechanically checkable, such as math, code, and formal reasoning. Training-free intervention, which instead adjusts model behavior at inference time without any gradient update, is the focus of this paper.

Within training-free intervention, representation engineering faces a fundamental trade-off between adaptivity and inference cost. *Static Steering* identifies a single static concept vector and adds it to the model's activation at almost zero inference cost (Zou et al., 2023a). This static vector, however, assumes the concept boundary is a flat hyperplane in representation space; because most concepts do not satisfy this assumption, linear steering under- or over-corrects activations that lie off this hyperplane and degrades generation quality under strong intervention. Recent work instead models the concept boundary with deeper, curved classifiers and computes an input-dependent steering direction from them at generation time (Raval et al., 2026; Lee & Kondor, 2026; Hoscilowicz et al., 2024). This *non-linear steering* corrects activations more precisely, but existing analyses treat its

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.

cost as unavoidable: because the steering direction is an arbitrary, non-affine function of the current activation, it must be recomputed from a full forward pass at every generated token.

We resolve this tension with a simple observation: although a quadratic concept probe is a non-linear function of the activation, its gradient is an *affine* function of that same activation. A quadratic probe is therefore curved enough to represent a non-flat concept boundary, yet inducing a steering direction that is linear enough to be written as a fixed weight update rather than recomputed at every step.

Building on this observation, we introduce **WAQS** (Weight Absorption for Quadratic probing and affine Steering), a framework that turns a curved, input-dependent steering direction into a static weight update. **WAQS** trains a quadratic probe to score a target concept, derives the steering update as the probe's gradient which is an affine function of input, and folds this affine map into the neighboring weight matrix, or layer norm. In contrast to static steering, **WAQS**'s steering direction still adapts to each input's position relative to the target concept; in contrast to dynamic non-linear steering, this adaptation is precomputed once, offline, so it costs nothing extra at inference time.

Our contributions can be summarized as follows:

- **Maximal function class and statistical interpretation.** We discuss that a smooth probe's gradient-based intervention is affine in the activation if and only if the probe is a polynomial of degree at most two. Quadratic probes are thus not merely one absorbable family but the entire absorbable class; any higher-order probe reintroduces the recomputation cost that motivated dynamic non-linear steering. Under a Gaussian representation model, the optimal quadratic probe admits a closed form, and its induced steering direction decomposes into a difference of class-conditional Mahalanobis pulls. Moreover, under the Gaussian assumption, we show that difference-in-means steering method (Arditi et al., 2024) is a special case of quadratic probing.
- **Weight absorption.** We show that gradient-based steering with a quadratic probe can be folded exactly into the adjacent linear projection, attention layer, and normalization layer through an offline weight update. The result is a standard checkpoint: it requires no custom inference code, no forward hooks, and works out of the box with existing serving stacks, quantization pipelines, and distribution formats.

2 Problem Setup and Theoretical Analysis

Let $x \in \mathbb{R}^D$ represent the hidden state vector (activation) extracted from a specific layer l within an LLM, where D denotes the hidden dimension size. Linear probing assumes a linear hyperplane as the concept boundary $f_L(x) = w^\top x + b$, where $w \in \mathbb{R}^D$ is the probe weight vector and $b \in \mathbb{R}$ is a scalar bias. To capture non-linear, pairwise interactions between activation values, we instead define a degree-2 polynomial (quadratic) probe function $f(x) : \mathbb{R}^D \to \mathbb{R}$:

$$f_Q(x) = x^\top W_p x + w_p^\top x + b_p, \tag{1}$$

where $W_p \in \mathbb{R}^{D\times D}$ is a symmetric matrix[1] capturing neuron-to-neuron correlation weights, $w_p \in \mathbb{R}^D$ represents the linear feature bias, and $b_p \in \mathbb{R}$ is the scalar bias. We train the probe via logistic loss to classify the presence or absence of a target semantic concept within the internal representations. To steer the model's behavior during inference, we perform an intervention on x to maximize or minimize the concept score $f_Q(x)$. To identify the direction of steepest ascent along this curved decision boundary, we compute the gradient of the probe function with respect to x: $\nabla_x f_Q(x) = 2W_p x + w_p$. Li et al. (2023) adopt the same gradient-based direction.

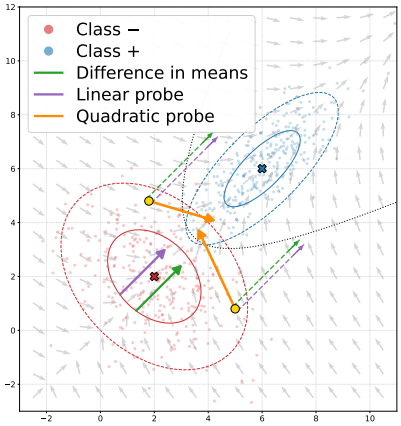


Figure 1: **Only the quadratic gradient adapts to the activation.** The quadratic probe's gradient field varies with position.

[1]Without loss of generality, W_p can be assumed symmetric, because the antisymmetric part satisfies $x^T W_p' x = 0$ for all x.

Because the derivative of a quadratic function is affine, the resulting steering direction $\nabla_x f(x)$ is *dynamically context-dependent*. The steering vector adapts at every token generation step based on the model's current hidden state x, replacing the fixed vector used by linear steering. The runtime activation update rule is:

$$x_{\text{steered}} = x + \alpha \nabla_x f(x) = x + \alpha(2W_p x + w_p) \tag{2}$$

where $\alpha \in \mathbb{R}$ is a scalar hyperparameter that controls the steering intensity: $\alpha > 0$ amplifies the concept and $\alpha < 0$ suppresses it.

2.1 Exact affine absorbability

Eq. 2 defines a state-dependent intervention: the update applied to x depends on x itself, so no single vector describes what the steering does. Write this intervention as the map $T_\alpha(x) := x + \alpha \nabla_x f_Q(x)$. The standard reading holds that such an intervention must run during the forward pass and therefore belongs to the dynamic, runtime-cost regime of prior work. We show this reading is mistaken for the quadratic case: there is no inference-time overhead for quadratic probing.

A map can fold into a linear layer precisely when it is affine. In the quadratic probing case, $T_\alpha(x) = M_{\text{steer}} x + b_{\text{steer}}$, where $M_{\text{steer}} = I + 2\alpha W_p$, $b_{\text{steer}} = \alpha w_p$, and $I \in \mathbb{R}^{D \times D}$ is the identity matrix. Therefore, as we will see in the next part, $T_\alpha(x)$ can be absorbed into the model's linear layer.

The next proposition shows that quadratic functions are the only functions that leads to affine map for steering which is absorbable.

Proposition 1 (Maximality)**.** *Let $f \in \mathcal{C}^2(\mathbb{R}^D)$ and let $T_\alpha(x) := x + \alpha \nabla_x f(x)$. Then T_α is an affine map of x for some $\alpha \neq 0$ if and only if f is a polynomial of degree at most two, in which case $T_\alpha(x) = (I + 2\alpha W_p)x + \alpha w_p$ as in equation 2.*

Proof. (Only If) Suppose T_α is affine for some $\alpha \neq 0$. Then $\nabla_x f(x) = \alpha^{-1}\left(T_\alpha(x) - x\right)$ is affine in x, hence the Hessian $\nabla_x^2 f$ is a constant matrix H. A second-order Taylor expansion then gives $f(x) = \frac{1}{2} x^\top H x + c^\top x + b$, i.e. f is quadratic with $W_p = \frac{1}{2} H$.

(If) Suppose f is a quadratic function. Then, $T_\alpha = M_{\text{steer}} x + b_{\text{steer}}$. □

Two consequences follow. First, the runtime cost incurred by nonlinear-probe steering is intrinsic to the function class, not to any particular implementation; no engineering effort removes it. Second, any modification that reintroduces activation dependence outside the gradient itself falls outside the absorbable class by the same argument.

2.2 Statistical interpretation via Gaussian Discriminant Analysis

In this part, we provide intuition why quadratic probing is going to help with steering.

Assumption 1 (Gaussian Distribution Assumption)**.** *The hidden state of positive and negative class follows the Gaussian distribution:*

$$x|y = +1 \sim \mathcal{N}(\mu_+, \Sigma_+), \quad x|y = -1 \sim \mathcal{N}(\mu_-, \Sigma_-) \tag{3}$$

where $\mu_+, \mu_- \in \mathbb{R}^D$ are the class means and $\Sigma_+, \Sigma_- \in \mathbb{R}^{D \times D}$ are positive-definite *class covariances, so that $\Sigma_+^{-1}, \Sigma_-^{-1}$ exist.*

Theorem 1. *Under the assumption 1, solving for $f_Q(x)$ by Gaussian Discriminant Analysis over all positive samples x_+ and negative samples x_- admits a closed-form solution.*

$$W_p = -\frac{1}{2}(\Sigma_+^{-1} - \Sigma_-^{-1}), \quad w_p = \Sigma_+^{-1}\mu_+ - \Sigma_-^{-1}\mu_-, \tag{4}$$

$$b_p = -\frac{1}{2}\Big(\mu_+^\top \Sigma_+^{-1}\mu_+ - \mu_-^\top \Sigma_-^{-1}\mu_- + \log|\Sigma_+| - \log|\Sigma_-|\Big) + \log\frac{p(y_+)}{p(y_-)}. \tag{5}$$

We defer the proof to the Appendix B. Theorem 1 implies that estimating the decision boundary requires quadratic probing whenever the two concept classes have different covariance matrices;

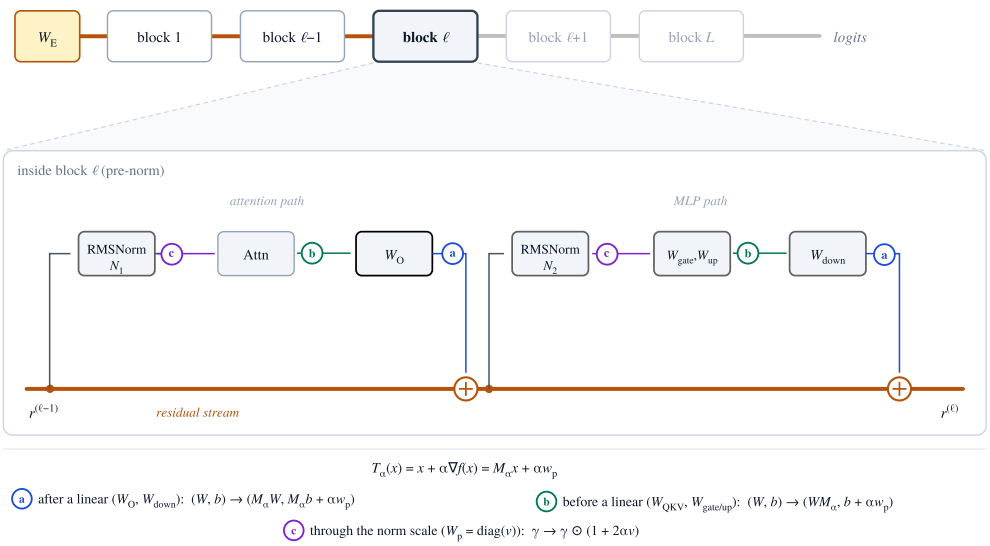


Figure 2: **Weight absorption folds the steering map $M_{\text{steer}} = I + 2\alpha W_p$ and bias $b_{\text{steer}} = \alpha w_p$ into the surrounding static weights**, for every point a quadratic probe can be attached to within a transformer block. **(a)** after a linear projection, **(b)** before a linear projection, and **(c)** through an RMS-norm/LayerNorm scale. In every case the runtime rule $T_\alpha(x) = x + \alpha\nabla f_Q(x) = M_{\text{steer}}x + b_{\text{steer}}$ is realized entirely offline, at zero inference-time cost.

when $\Sigma_+ = \Sigma_-$, linear probing suffices. When $\Sigma_+ = \Sigma_- = I$, difference-in-means method gives the right steering vector under Gaussian assumption.

Under Theorem 1, the steering vector can be calculated as,

$$\nabla_x f_Q(x) \;=\; \Sigma_+^{-1}(\mu_+ - x) \;-\; \Sigma_-^{-1}(\mu_- - x). \tag{6}$$

Eq. equation 6 expresses the intervention as a difference of two *Mahalanobis pulls*: a pull toward the positive centroid and a push away from the negative centroid, each shaped by the inverse covariance of its corresponding class. The inverse covariance gives greater weight to directions with lower within-class variance. Consequently, both the magnitude and direction of the steering vector depend on the activation x and the geometry of the two class distributions. When $\Sigma_+ = \Sigma_- = \Sigma$, the x-dependent terms cancel, yielding the constant vector $\Sigma^{-1}(\mu_+ - \mu_-)$, the classical Fisher linear discriminant direction which is also illustrate in Marks & Tegmark (2023). Figure 1 illustrates that moving negative activations along a fixed difference-in-means or linear-probing direction does not necessarily move them into the positive cluster, leaving many points far from it. In contrast, the quadratic gradient field adapts to each activation and bends toward the interior of the positive cluster.

3 Methodology

In this part, we describe how the steering vector generated by quadratic probing can be absorbed into the transformer's static weights. We further discuss low rank approximation for quadratic probing.

3.1 Weight absorption

A dynamic steering vector in general leads to an additional inference-time latency cost due to calculating the steering vector during every forward pass. We exploit the affine nature of the gradient in quadratic probing to fold the affine operation into the transformer's static weight matrices. Depending on whether the steering happens before or after a linear projection, we can absorb W_p and w_p into the parameters of the linear projection as shown in Fig 2.

(a) Steering and probing after projection layer: By expanding and rearranging Eq. 2, we have, $x_{\text{steered}} = x + 2\alpha W_p x + \alpha w_p = (I + 2\alpha W_p)x + \alpha w_p = M_{\text{steer}}x + b_{\text{steer}}$ with $I \in \mathbb{R}^{D\times D}$ being

the Identity matrix. Within a standard transformer architecture, a linear projection layer (such as an attention projection matrix) produces the hidden state x at layer l by acting on a precursor state $h \in \mathbb{R}^K$: $x = W_{\text{layer}}h + b_{\text{layer}}$. Using this equation, our steering rule yields:

$$x_{\text{steered}} = M_{\text{steer}}(W_{\text{layer}}h + b_{\text{layer}}) + b_{\text{steer}} = (M_{\text{steer}}W_{\text{layer}})h + (M_{\text{steer}}b_{\text{layer}} + b_{\text{steer}})$$

By grouping terms, we can directly compute a modified weight matrix W_{new} and bias vector b_{new} for the target transformer layer:

$$W_{\text{new}} = (I + 2\alpha W_p)W_{\text{layer}}, b_{\text{new}} = (I + 2\alpha W_p)b_{\text{layer}} + \alpha w_p \tag{7}$$

By replacing the original model weights with W_{new} and b_{new} offline, the model natively generates steered text with zero inference overhead and full compatibility with existing serving engines.

(b) Steering and probing before the projection layer: To apply steering vector to hidden state h before a projection layer, we apply $W_{\text{layer}}, b_{\text{layer}}$ after applying $M_{\text{steer}}, b_{\text{steer}}$.

$$\begin{aligned} h_{\text{steer}} &= M_{\text{steer}}h + b_{\text{steer}} \\ x &= W_{\text{layer}}h_{\text{steer}} + b_{\text{layer}} = W_{\text{layer}}M_{\text{steer}}h + (b_{\text{layer}} + W_{\text{layer}}b_{\text{steer}}) \end{aligned} \tag{8}$$

As a result, we can write W_{new} and b_{new} as: $W_{\text{new}} = W_{\text{layer}}M_{\text{steer}}, b_{\text{new}} = b_{\text{layer}} + W_{\text{layer}}b_{\text{steer}}$

(c) Quadratic Probing for RMS-norm/layer norm: Given a linear layer $x = W_{\text{layer}}h + b_{\text{layer}}$, steering with quadratic probing on x modifies W_{layer} directly. If a non-linear operation instead produces x (e.g., RMS-norm in Llama, or an MLP activation function), we can absorbed M_{steer} into the next projection layer or Attention Layer. In Figure 2, if the intervention happens at point "**c**", M_{steer} can be absorbed in W_K, W_Q, W_V (if "**c**" is before Attention Layer) or in W_{up} (if "**c**" is before MLP layer). However, if we have to do weight absorption on the RMS-norm, then with a slight modification, M_{steer} also can be absorbed into RMS-norm and LayerNorm: restricting W_p to a diagonal matrix allows M_{steer} to be absorbed directly into the norm layer's scale parameter. Let $x = \frac{h}{||h||_2} \odot \gamma = \text{diag}(\gamma) \cdot \frac{h}{||h||_2}$ denote an RMS-norm layer, where $\odot$ is the element-wise product, $\gamma \in \mathbb{R}^D$, and $\text{diag}(\gamma)$ denotes the diagonal matrix with γ on its diagonal. The probe function $f(x) = x^\top \text{diag}(v)x + w_p^\top x + b_p$ gives the steering vector $2 \cdot \text{diag}(v) \cdot x + w_p$, so:

$$\begin{aligned} x_{\text{steered}} &= x + 2\alpha\,\text{diag}(v)x + w_p = \text{diag}(\gamma)\frac{h}{\|h\|_2} + 2\alpha \cdot \text{diag}(v)\,\text{diag}(\gamma)\frac{h}{\|h\|_2} + \alpha w_p \\ &= \frac{h}{\|h\|_2} \odot [\gamma + (2\alpha \cdot v \odot \gamma)] + \alpha w_p = \frac{h}{\|h\|_2} \odot [\gamma \odot (1 + 2\alpha \cdot v)] + \alpha w_p = \frac{h}{\|h\|_2} \odot \hat{\gamma} + \alpha w_p, \end{aligned}$$

$\hat{\gamma} = (\gamma \odot (2\alpha \cdot v + 1))$. Steering with quadratic probing under a diagonal matrix therefore introduces no overhead relative to linear probing; the same approach applies to LayerNorm.

3.2 Quadratic Probe Estimation

Low-rank discriminative probe. For modern production Large Language Models, the hidden dimension D reaches into the thousands (e.g., $D = 4096$ for Llama-3-8B). In these regimes, the full-rank quadratic weight matrix $W_p \in \mathbb{R}^{D \times D}$ encounters a severe parameter explosion of $\mathcal{O}(D^2)$ elements (1.67×10^7 parameters for $D = 4096$). This memory footprint makes probe training unstable and renders the offline matrix multiplication $(I + 2\alpha W_p)W_{\text{layer}}$ computationally prohibitive.

To resolve this limitation, we apply a low-rank factorization constraint to the quadratic probe. We hypothesize that the semantic concepts we wish to steer reside within a low-dimensional intrinsic subspace of the network's activations. Therefore, we restrict the symmetric matrix W_p to a low-rank approximation: $W_p \approx \frac{1}{2}(UV^\top + VU^\top)$,[2] where $U, V \in \mathbb{R}^{D \times R}$ are low-rank bottleneck matrices with rank $R \ll D$ (e.g., $R \in \{4, 8, 16\}$). Under this constraint:

$$W_{\text{new}} = \left(I + \alpha(UV^\top + VU^\top)\right) W_{\text{layer}} = W_{\text{layer}} + \alpha\left((UV^\top + VU^\top)W_{\text{layer}}\right) \tag{9}$$

Since $W_p = \frac{1}{2}(UV^\top + VU^\top)$, $f(x)$ becomes

$$f(x) = x^\top W_p x + w_p^\top x + b_p = (U^\top x)^\top (V^\top x) + w_p^\top x + b \tag{10}$$

[2] $\frac{1}{2}(UV^\top + VU^\top)$ is the symmetric part of $UV^\top$ (or $VU^\top$).

So $f(x)$ depends on the interaction between the directions spanned by U and V.

The low-rank extension cuts both memory and compute cost during the weight-folding phase. Computing the full-rank matrix multiplication $W_p W_{\text{layer}}$ costs $\mathcal{O}(D^3)$ floating-point operations (FLOPs). By enforcing the low-rank constraint and altering the order of operations via the associative property in Equation equation 9, we compute $A = V^\top W_{\text{layer}}$ first, followed by UA. This reduces the time complexity to $\mathcal{O}(D^2 \cdot R)$, transforming a computationally heavy offline workload into a lightweight step that computes in seconds.

Shrinkage GDA. Theorem 1's closed form is a population-level statement, however, and in practice $\Sigma_+, \Sigma_- \in \mathbb{R}^{D\times D}$ must be estimated from a finite training set: when the number of training examples is smaller than D (as with $D = 4096$ activations), the sample covariance is singular and $\Sigma_\pm^{-1}$ is undefined. We therefore replace each sample covariance with a doubly shrunken estimator (Ramey et al., 2017)

$$\widetilde{\Sigma}_\pm(\lambda) \;=\; (1-\lambda)\Big[\hat{\Sigma}_\pm + \hat{\Sigma}_{\text{pool}}\Big] \;+\; \lambda\,\frac{\operatorname{tr}\big(\hat{\Sigma}_\pm\big)}{D}\,I, \qquad \lambda \in [0,1], \tag{11}$$

While λ shrinks toward a scaled identity. The resulting estimator family contains that $\lambda \to 1$ drives both covariances to scaled identities and recovers a difference-in-means direction. This shrinkage family treats linear and quadratic probing as two ends of one regularization path.

4 Experiments

Our theoretical analysis argues that WAQS's quadratic probing should replace linear probing as the standard steering baseline: (1) quadratic probing introduces no inference cost through weight absorption, and (2) linear probing nests within our quadratic parameterization, since setting $W_p = 0$ recovers the linear probe exactly. To confirm this claim holds in practice, we evaluate affine steering using quadratic probing against Difference-in-Means (DiffMean) and Linear Probing (LP), both of which use a constant steering vector that does not depend on the input. We report results over 5 seeds unless otherwise stated. Additional training details are provided in Appendix C.

4.1 Main Results for probing and steering

Truthfulness Steering: We use the TruthfulQA dataset (Lin et al., 2022), which comprises questions designed to mimic human superstitions and misconceptions. Following the linear probing approach of Li et al. (2023), we extract the intermediate activations at the last prompt token from the 14th layer's 18th attention head, which yields the highest probing accuracy. For open-ended generation, we employ GPT-4o as an automated judge to evaluate the responses. We prompt the judge to output a binary "Yes" or "No" indicating the truthfulness and informativeness of each generated answer (Appendix C.2 details the prompt templates). We report **True** as the percentage of answers evaluated as factually correct. Additionally, we report **True*Info**, computed as the product of the truthfulness and informativeness percentages. This combined metric serves as a proxy for utility and penalizes the model for reaching high truthfulness through evasive, non-committal responses. **MC1** (Li et al., 2023) evaluates whether the model assigns the highest conditional probability to the lone true answer. **MC2** calculates the total normalized conditional probability the model assigns to the set of all truthful answers relative to the false ones. To quantify the intervention's effect on general fluency, we compute **CE** (Cross-Entropy), the steered model's next-token prediction loss on a held-out corpus. We also calculate the **KL** divergence between the next-token probability distributions of the pre- and post-injection models over the same corpus. We sweep the steering strength α for each method and choose the α that achieves the best **MC1** score.

Takeaway. Table 1 shows that Rank-1 Quadratic Probing improves **True*Info** and **True** over constant steering vectors on Llama-2-7B (20.43 vs. 13.28-19.30 True*Info), indicating better utility after steering. Quadratic Probing also achieves higher **MC1** and **MC2**, and almost best **MC3**, and its **CE** stay comparable compared with other methods. KL divergence stays between 0.005 to 0.007 for every method on LlaMA-2-7B, so none of the three steering methods materially shifts the next-token policy.

Refusal Steering: Following Chhabra & Khalili (2025)'s work, we construct the harmful dataset and harmless dataset based on AdvBench (Zou et al., 2023b), MaliciousInstruct (Huang

Table 1: **Quadratic steering improves truthfulness while preserving fluency across model families.** Results on TruthfulQA for five steering methods. We compare with Linear Probe, DiffMeans, Angular Steering, and Spherical Steering. For each method, α is swept and the value with the best MC1 is reported. True and Info are judged by GPT-4o.

Method	True*Info (%) ↑	True (%) ↑	MC1 (%) ↑	MC2 (%) ↑	MC3 (%) ↑	CE ↓	KL ↓
Llama-2-7B							
Quadratic Probe	**20.43** ± **0.58**	**41.62** ± **1.03**	**35.37** ± **1.84**	**55.43** ± **2.89**	29.73 ± 2.69	2.41	0.006
Linear Probe	13.28 ± 0.33	38.07 ± 0.59	33.78 ± 1.33	51.06 ± 2.77	26.99 ± 2.63	**2.13**	**0.005**
DiffMeans	18.82 ± 0.28	39.53 ± 0.44	33.78 ± 1.33	51.12 ± 2.89	26.94 ± 2.54	2.17	0.005
Angular Steering	18.77 ± 0.63	39.41 ± 1.17	33.90 ± 1.50	51.07 ± 2.78	27.05 ± 2.66	2.37	0.007
Spherical Steering	19.30 ± 0.76	40.02 ± 1.21	34.02 ± 4.10	53.51 ± 8.26	**30.64** ± **7.50**	2.54	0.006
Gemma-3-12B							
Quadratic Probe	**20.68** ± **1.39**	**42.23** ± **2.20**	**51.41** ± **5.28**	**70.57** ± **3.22**	**43.99** ± **5.11**	**2.97**	0.025
Linear Probe	16.02 ± 1.34	36.72 ± 2.02	48.84 ± 1.64	68.49 ± 0.38	40.58 ± 0.29	2.98	**0.012**
DiffMeans	16.36 ± 0.97	37.21 ± 1.88	50.68 ± 2.86	69.19 ± 1.51	42.18 ± 2.62	3.03	0.111
Angular Steering	17.55 ± 1.48	37.94 ± 2.31	49.21 ± 1.82	69.12 ± 0.76	41.23 ± 0.79	3.04	0.354
Spherical Steering	17.10 ± 1.26	37.94 ± 2.18	49.57 ± 2.68	68.33 ± 0.44	41.03 ± 0.12	3.04	0.142

et al., 2023), TDC2023 (Mazeika et al., 2023), HarmBench (Mazeika et al., 2024), and Alpaca (Taori et al., 2023). We evaluate on the MMLU (Hendrycks et al., 2021), GSM8k (Cobbe et al., 2021) and ARC-Challenge (Clark et al., 2018) to measure the utility of the steering methods. We also evaluate the safety performance using GCG (Zou et al., 2023b), TAP (Mehrotra et al., 2024), and GPTFuzz (Yu et al., 2023) as the attacking method to get the Attack Success Rate(ASR).

Table 2: **Quadratic steering yields the most robust refusal behavior without sacrificing utility.** Safety and utility of refusal steering on Llama-2-7B. Steering directions are estimated from harmful and harmless prompts built from AdvBench, MaliciousInstruct, TDC2023, HarmBench, and Alpaca, following Chhabra & Khalili (2025). Quadratic Probe uses a rank-1 probe. Safety is the attack success rate (ASR) under three jailbreak attacks: GCG, TAP, and GPTFuzz. Utility is accuracy on MMLU, GSM8k, and ARC-Challenge.

	Safety (↓)			Utility (↑)		
Method	GCG	TAP	GPTFuzz	MMLU	GSM8k	ARC-Challenge
Quadratic Probe	9	4.0	1.0	47.30	22.36	49.57
Linear Probe	10	4.4	5.0	47.27	21.83	49.23
DiffMeans	10	7.8	1.0	47.23	22.59	48.97
Angular Steering	9	4.2	1.0	47.31	22.51	49.23
Spherical Steering	0	4.4	1.0	46.94	19.48	47.44

Table 2 presents the safety and utility performance of five steering methods on LLaMA-2-7B.[3] **Takeaway.** Across all three attacks, Rank-1 Quadratic Probing achieves the lowest ASR (9.0, 4.0, and 1.0 under GCG, TAP, and GPTFuzz respectively), beating both Linear Probing and DiffMeans. This suggests the non-linear decision boundary introduced by Quadratic Probing yields a refusal direction that resists all three attacks more effectively than a single fixed direction. Quadratic Probing also matches or exceeds Linear Probing and DiffMeans on all three utility benchmarks (MMLU, GSM8k, ARC-Challenge), so the safety gain does not cost general capability. DiffMeans and Linear Probing reach comparable utility but do not match Quadratic Probing's safety gains, indicating that the more expressive probing objective captures the harmful/harmless distinction without degrading benign-task behavior.

We also report the probing performance (i.e., the metrics for the linear and quadratic classifiers trained on harmful/harmless prompts). To assess the generalization of the learned probe, the test set includes harmful samples aggregated from HarmBench (Mazeika et al., 2024), AdvBench (Zou et al., 2023b), and StrongREJECT (Souly et al., 2024), and harmless samples from Alpaca dataset.

[3]We conduct a grid search over the parameter α for each method; Appendix C.4 details the procedure.

Table 3: Steering effects across different behavioral concepts. Lower values indicate stronger negative steering, while higher values indicate stronger positive steering.

Dataset / Concept	Method	Negative Steering decrease ↓	Positive Steering improvement↑
Sycophancy	LR-Quadratic Probing	**-0.0780** ±0.0039	0.0322 ±0.0025
	GDA-Quadratic Probing	-0.0557 ±0.0024	**0.0326** ±0.0101
	Linear Probing	-0.0358 ±0.0037	0.0050 ±0.0023
	Contrast Activation Addition	-0.0403 ±0.0032	0.0074 ±0.0037
Corrigible	LR-Quadratic Probing	**-0.0361 ±0.0035**	0.3815 ±0.0555
	GDA-Quadratic Probing	-0.0360 ±0.0113	**0.4052 ±0.0205**
	Linear Probing	-0.0322 ±0.0084	0.3702 ±0.0344
	Contrast Activation Addition	-0.0331 ±0.0075	0.3903 ±0.0286

Table 8 presents the probing performance of linear probing and quadratic probing at ranks 1, 2, and 4. **Takeaway.** Table 8 shows that linear and quadratic probes reach similar classification performance (92.9% vs. 88.9–94.7% accuracy across ranks). Combined with Table 2, this indicates that probing accuracy alone does not predict steering quality: a dynamic, context-dependent steering vector improves refusal steering even when the underlying probe classifies no better than a linear one.

Steering for Anthropic AI Risk Concepts: We use Anthropic's Advanced AI Risk dataset (Perez et al., 2023) to steer the model's sycophancy, following the setup in CAA (Rimsky et al., 2024). We evaluate steering with a next-token prediction metric that measures the change in the model's tendency to predict the sycophantic response token after the steering intervention. We compute the softmax over the logits of answer A and B and take the difference as the metric. We sweep $\alpha \in \{0.5, 1, 1.5, 2\}$ for all of the methods. All the steering trials are tested on the layer 24 to get better contrast. A more negative value under negative steering indicates stronger suppression of sycophantic behavior; a larger positive value under positive steering indicates stronger enhancement. **Takeaway.** Table 3 shows that quadratic steering achieves the strongest suppression of sycophancy, well beyond both linear steering and CAA. GDA Quadratic achieves the strongest positive steering effect, closely followed by Quadratic , while Linear and CAA produce only small positive changes (0.0050 and 0.0032 respectively). Quadratic steering therefore controls sycophancy in both directions markedly more than linear steering ($2.2\times$ stronger suppression, $6.4\times$ stronger enhancement) under the same evaluation setup used in CAA.

We also report qualitative generations comparing methods on the same concept and prompt in Appendix C.5.

4.2 Ablation Study

Ablation Study on ranks. Table 7 reports the evaluation results across different ranks for Quadratic Probing on Llama-2-7b. Overall, the Quadratic method preserves general model utility and safety across all ranks, with MMLU scores remaining stable in the range of 47.20–47.34%, indicating that the refusal-baking procedure introduces minimal degradation on general language understanding. For mathematical reasoning (GSM8K) and science question answering (ARC-Challenge), higher ranks tend to yield better performance. Meanwhile, rank=4 achieves the strongest safety performance on TAP (1.0) and GPTFuzz (0.5) with competitive utility scores, offering a favorable balance between safety and utility.

Frozen quadratic steering comparison. The gains of Quadratic Probing over DiffMean and Linear Probing could come from two different sources: (i) the steering direction $2W_p x + w_p$ adapts to each input's position relative to the concept boundary, or (ii) the quadratic probe simply learns a better *constant* direction than a linear probe, independent of whether that direction is allowed to vary with x. To separate these two effects, we introduce a **Frozen Quadratic (Frozen-QP)** baseline that keeps the same trained probe parameters (W_p, w_p) but replaces the per-input gradient with a single direction evaluated at the mean activation of the probe's training set:

$$d_{\text{frozen}} = 2W_p\bar{x} + w_p, \qquad \bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i, \tag{12}$$

where $\{x_i\}_{i=1}^N$ are the same activations used to fit (W_p, w_p). Because the gradient of a quadratic form is affine, d_{frozen} is exactly the average of the per-example steering directions, $d_{\text{frozen}} = \frac{1}{N}\sum_{i=1}^N (2W_p x_i + w_p)$, so this ablation changes nothing about the fitted probe itself – it only removes the x-dependence at steering time, isolating adaptivity from probe expressiveness. We apply d_{frozen} as a constant steering vector, $x_{\text{steered}} = x + \alpha\, d_{\text{frozen}}$, and select α using the same held-out-validation protocol as DiffMean and Linear Probing.

Table 4: Dynamic vs. Frozen Quadratic Probing on TruthfulQA (Llama-2-7B). Frozen-QP uses the same trained (W_p, w_p) as Quadratic Probe but replaces the per-input direction with the training-set-average direction of Eq. 12.

Method	**True*Info (%)** ↑	**True (%)** ↑	**MC1 (%)** ↑	**MC2 (%)** ↑	**MC3** ↑
Quadratic Probe (dynamic)	**20.43** ± **0.58**	**41.62** ± **1.03**	**35.37** ± **1.84**	**55.43** ± **2.89**	**29.73** ± **2.69**
Frozen-QP (avg. direction)	13.47 ± 0.59	38.63 ± 1.32	33.66 ± 1.50	51.05 ± 2.77	26.93 ± 2.72
Linear Probe	13.28 ± 0.33	38.07 ± 0.59	33.78 ± 1.33	51.06 ± 2.77	26.99 ± 2.63
DiffMeans	18.82 ± 0.28	39.53 ± 0.44	33.78 ± 1.33	51.12 ± 2.89	26.94 ± 2.54

Takeaway. The results show that the steering vector should adapt to the input for better performance. Once the x-dependence is removed, Frozen-QP falls back to the level of the constant baselines. On MC1, MC2, and MC3 it lies within 0.12 points of Linear Probe (33.66 vs. 33.78, 51.05 vs. 51.06, 26.93 vs. 26.99). On True*Info it is essentially tied with Linear Probe (13.47 vs. 13.28) and falls below DiffMeans (18.82). Because direction of Frozen-QP is exactly the average of the per-example directions, the two variants apply the same steering on average and differ only in how the direction varies across inputs. The quadratic probe therefore does not simply learn a better constant direction than a linear probe. Its advantage comes from adapting the steering direction to each activation, which weight absorption provides at no inference-time cost.

5 Discussion and Limitations

The ability to absorb affine steering generated by a quadratic prob into model parameters depends on the location at which steering is applied. For activations produced directly by a linear transformation, the steering transformation can be folded into the corresponding weights. However, exact absorption is less straightforward when steering is applied directly to the residual stream. A residual transformation

$$r_{\text{steered}} = M_{\text{steer}} r^{(l)} + \alpha w_p \tag{13}$$

also transforms the identity skip path, which has no associated weight matrix into which M_{steer} can be directly absorbed. Consequently, residual-stream quadratic steering generally requires explicitly applying the steering transformation during inference. For a dense quadratic probe, this introduces an additional $\mathcal{O}(D^2)$ matrix–vector multiplication at each steered location. Structured parameterizations, such as diagonal or low-rank steering matrices, can reduce this overhead to $\mathcal{O}(D)$ or $\mathcal{O}(RD)$.

6 Conclusion

Quadratic probes provide input-dependent steering at the inference cost of a static linear vector. The gradient of a quadratic concept score is affine in the activation, so an offline weight update absorbs the entire intervention into the model's existing linear layers. Proposition 1 shows quadratic probes are the largest function class with this property: any higher-degree probe reintroduces the inference-time cost that motivated dynamic non-linear steering. Quadratic probing extends linear probing rather than replacing it with a separate technique. Under a Gaussian representation model, the optimal quadratic probe decomposes the steering direction into a difference of class-conditional Mahalanobis pulls. Across truthfulness, refusal, and sycophancy steering, quadratic probing matches or exceeds constant linear steering and difference-in-means on task performance and model utility, and the absorbed implementation adds zero inference-time operations.

AI USE STATEMENT

In this work, we used generative AI tools for aiding/polishing writing and correcting grammatical errors and implementing methods. We have not used generative AI tools for generating scientific content, and generating synthetic data sets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refine hypotheses, designing or providing feedback on research methodology or experiments, assist with translation, cleaning and reformatting dataset, supporting qualitative and thematic data analysis, interpret results are not applicable to this work.

We have reviewed all AI-assisted work. We also review the code which is generated by AI and test with ourselves. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*, 2023.

Vishnu Kabir Chhabra and Mohammad Mahdi Khalili. Towards understanding and improving refusal in compressed models via mechanistic interpretability, 2025. URL https://arxiv.org/abs/2504.04215.

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:1803.05457v1*, 2018.

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

Lang Gao, Jinghui Zhang, Wei Liu, Fengxian Ji, Chenxi Wang, Zirui Song, Akash Ghosh, Youssef Mohamed, Preslav Nakov, and Xiuying Chen. The cylindrical representation hypothesis for language model steering, 2026. URL https://arxiv.org/abs/2605.01844.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

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.

Yangsibo Huang, Samyak Gupta, Mengzhou Xia, Kai Li, and Danqi Chen. Catastrophic jailbreak of open-source llms via exploiting generation. *arXiv preprint arXiv:2310.06987*, 2023.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing Models with Task Arithmetic. In *International Conference on Learning Representations*, 2023. URL https://mlanthology.org/iclr/2023/ilharco2023iclr-editing/.

Su Hyeong Lee and Risi Kondor. Deep minds and shallow probes, 2026. URL https://arxiv.org/abs/2605.11448.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 41451–41530. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/81b8390039b7302c909cb769f8b6cd93-Paper-Conference.pdf.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods, 2022. URL https://arxiv.org/abs/2109.07958.

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

Mantas Mazeika, Andy Zou, Norman Mu, Long Phan, Zifan Wang, Chunru Yu, Adam Khoja, Fengqing Jiang, Aidan O'Gara, Ellie Sakhaee, Zhen Xiang, Arezoo Rajabi, Dan Hendrycks, Radha Poovendran, Bo Li, and David Forsyth. Tdc 2023 (llm edition): The trojan detection challenge. In *NeurIPS Competition Track*, 2023.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. 2024.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: Jailbreaking black-box llms automatically. *Advances in Neural Information Processing Systems*, 37:61065–61105, 2024.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 17359–17372. Curran Associates, Inc., 2022. doi: 10.52202/068431-1262. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/6f1d43d5a82a37e89b0665b33bf3a182-Paper-Conference.pdf.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer, 2023. URL https://arxiv.org/abs/2210.07229.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Ethan Perez, Sam Ringer, Kamile Lukosiute, Karina Nguyen, Edwin Chen, Scott Heiner, Craig Pettit, Catherine Olsson, Sandipan Kundu, Saurav Kadavath, Andy Jones, Anna Chen, Benjamin Mann, Brian Israel, Bryan Seethor, Cameron McKinnon, Christopher Olah, Da Yan, Daniela Amodei, Dario Amodei, Dawn Drain, Dustin Li, Eli Tran-Johnson, Guro Khundadze, Jackson Kernion, James Landis, Jamie Kerr, Jared Mueller, Jeeyoon Hyun, Joshua Landau, Kamal Ndousse, Landon Goldberg, Liane Lovitt, Martin Lucas, Michael Sellitto, Miranda Zhang, Neerav Kingsland, Nelson Elhage, Nicholas Joseph, Noemi Mercado, Nova DasSarma, Oliver Rausch, Robin Larson, Sam McCandlish, Scott Johnston, Shauna Kravec, Sheer El Showk, Tamera Lanham, Timothy Telleen-Lawton, Tom Brown, Tom Henighan, Tristan Hume, Yuntao Bai, Zac Hatfield-Dodds, Jack Clark, Samuel R. Bowman, Amanda Askell, Roger Grosse, Danny Hernandez, Deep Ganguli, Evan Hubinger, Nicholas Schiefer, and Jared Kaplan. Discovering language model behaviors with model-written evaluations. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13387–13434, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.847. URL https://aclanthology.org/2023.findings-acl.847/.

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

John A. Ramey, Caleb K. Stein, Phil D. Young, and Dean M. Young. High-dimensional regularized discriminant analysis, 2017. URL https://arxiv.org/abs/1602.01182.

Shivam Raval, Hae Jin Song, Linlin Wu, Abir Harrasse, Jeff M. Phillips, Fazl Barez, and Amirali Abdullah. Curveball steering: The right direction to steer isn't always linear, 2026. URL https://arxiv.org/abs/2603.09313.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15504–15522, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.828. URL https://aclanthology.org/2024.acl-long.828/.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. Large language model alignment: A survey. *arXiv preprint arXiv:2309.15025*, 2023.

Alexandra Souly, Qingyuan Lu, Dillon Bowen, Tu Trinh, Elvis Hsieh, Sana Pandey, Pieter Abbeel, Justin Svegliato, Scott Emmons, Olivia Watkins, and Sam Toyer. A strongreject for empty jailbreaks, 2024.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

Minh Hieu Vu and Tan Nguyen. Angular steering: Behavior control via rotation in activation space. *Advances in Neural Information Processing Systems*, 38:121653–121690, 2026.

Junkang Wu, Yuexiang Xie, Zhengyi Yang, Jiancan Wu, Jinyang Gao, Bolin Ding, Xiang Wang, and Xiangnan He. β -dpo: Direct preference optimization with dynamic β. *Advances in Neural Information Processing Systems*, 37:129944–129966, 2024a.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. Reft: Representation finetuning for language models, 2024b. URL https://arxiv.org/abs/2404.03592.

Zejia You, Chunyuan Deng, and Hanjie Chen. Spherical steering: Geometry-aware activation rotation for language models, 2026. URL https://arxiv.org/abs/2602.08169.

Jiahao Yu, Xingwei Lin, Zheng Yu, and Xinyu Xing. Gptfuzzer: Red teaming large language models with auto-generated jailbreak prompts. *arXiv preprint arXiv:2309.10253*, 2023.

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

Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023b.

A Related Work

Activation Steering. Zou et al. (2023a) pioneered difference-in-means and linear-probe-derived vectors as strong, cheap steering baselines. Li et al. (2023) similarly derive a steering direction as the gradient of a trained linear probe; because a linear probe's gradient is constant, this direction does not depend on the current activation, so the resulting intervention is static despite its gradient-based derivation. Contrastive Activation Addition (CAA) (Rimsky et al., 2024) finds a difference-in-means

direction from contrastive prompt pairs, and representation fine-tuning methods such as ReFT (Wu et al., 2024b) learn a fixed, input-independent intervention on the residual stream through gradient-based training. All of these methods share the same underlying assumption: the concept boundary is well approximated by a hyperplane, so a single fixed direction suffices for every input.

Curved and Input-Dependent Steering. More recent work argues that concept directions are better modeled as curved manifolds than hyperplanes (Raval et al., 2026; Lee & Kondor, 2026; Hoscilowicz et al., 2024), motivating angular steering (Vu & Nguyen, 2026) and spherical steering (You et al., 2026) variants. SOM (Piras et al., 2026) shows that a single direction cannot capture the steering effect, since it varies across multiple learned components. Cylindrical steering (Gao et al., 2026) proposes a cylindrical representation hypothesis, showing that the steering vector is also sensitive to position within the same normal plane. These methods recover exactly the input-dependence that linear methods lack, but at a cost: extending Li et al. (2023)'s gradient-based recipe to a non-linear probe requires recomputing the gradient from the current activation at every generation step, reintroducing the per-token inference cost that motivated static steering in the first place.

Training-free Weight Editing. A separate line of work modifies model weights directly, without gradient-based training, to induce or remove targeted behaviors. Directional ablation methods identify a single linear direction associated with a behavior (e.g., refusal) and project it out of every weight matrix that writes to the residual stream, effectively "uninstalling" the behavior at the parameter level (Arditi et al., 2024). Task arithmetic constructs task vectors as the difference between fine-tuned and pretrained weights and edits models by adding or negating these vectors, again without further training (Ilharco et al., 2023). Locate-and-edit approaches such as ROME (Meng et al., 2022) and MEMIT (Meng et al., 2023) identify and directly overwrite the weights responsible for specific factual associations. These methods share our goal of avoiding training-time and inference-time cost, but they operate on *static, input-independent* weight edits, adding or removing a single direction regardless of the input.

WAQS closes this gap: extending Li et al. (2023)'s gradient-based recipe from a linear probe to a quadratic one keeps the resulting map affine (Proposition 1), so it folds into the model weights exactly like the static edits above, while remaining as input-dependent as the curved steering methods it replaces.

B Proof of Theorem 1

Proof. Assume the hidden state of the positive and negative groups are respectively Gaussian distributed, as in Assumption 1:

$$x|y=+1 \sim \mathcal{N}(\mu_+, \Sigma_+), \quad x|y=-1 \sim \mathcal{N}(\mu_-, \Sigma_-), \qquad x \in \mathbb{R}^D. \tag{14}$$

Under Assumption 1, the Bayes-optimal discriminant score is the log-posterior-odds $g(x) := \log p(y_+ \mid x) - \log p(y_- \mid x)$; we identify the probe $f_Q(x)$ with this score, $f_Q(x) := g(x)$. By Bayes' theorem,

$$\begin{aligned} g(x) &= (\log p(x|y_+) + \log p(y_+) - \log p(x)) - (\log p(x|y_-) + \log p(y_-) - \log p(x)) \\ &= [\log p(x|y_+) - \log p(x|y_-)] + [\log p(y_+) - \log p(y_-)] \end{aligned} \tag{15}$$

As the log likelihood could be written as:

$$\begin{aligned} \log p(x \mid \mu_+, \Sigma_+) &= -\frac{1}{2}(x-\mu_+)^\top \Sigma_+^{-1}(x-\mu_+) - \frac{1}{2}\log|\Sigma_+| + C_1 \\ \log p(x \mid \mu_-, \Sigma_-) &= -\frac{1}{2}(x-\mu_-)^\top \Sigma_-^{-1}(x-\mu_-) - \frac{1}{2}\log|\Sigma_-| + C_1 \end{aligned} \tag{16}$$

Here, $C_1 = -\frac{1}{2}(D\log(2\pi))$, where D is the dimension of x (consistent with Assumption 1). So

$$g(x) = -\frac{1}{2}x^\top(\Sigma_+^{-1} - \Sigma_-^{-1})x + (\Sigma_+^{-1}\mu_+ - \Sigma_-^{-1}\mu_-)^\top x + C_2 \tag{17}$$

where $C_2 = -\frac{1}{2}(\mu_+^\top\Sigma_+^{-1}\mu_+ - \mu_-^\top\Sigma_-^{-1}\mu_- + \log|\Sigma_+| - \log|\Sigma_-|) + (\log p(y_+) - \log p(y_-))$

Matching coefficients against $f_Q(x) = x^\top W_p x + w_p^\top x + b_p$ (Eq. equation 1) with $g(x) = f_Q(x)$ gives

$$W_p = -\frac{1}{2}(\Sigma_+^{-1} - \Sigma_-^{-1}), \quad w_p = \Sigma_+^{-1}\mu_+ - \Sigma_-^{-1}\mu_-, \quad b_p = C_2, \tag{18}$$

which is exactly Theorem 1, since C_2 above is identical to the closed form for b_p stated there. □

C Additional Experimental details

C.1 Experimental Setup

Experiment Environment We conduct all experiments on a server with two Intel Xeon 6326 **CPUs** and 6 Nvidia A6000 **GPUs**, using PyTorch version 2.4.1.

Models and Representations. Because our evaluation spans several behavioral steering tasks, we follow task-specific experimental settings and use different backbone models where appropriate. Within each task, all steering methods are evaluated on the same backbone and activation location to ensure a controlled comparison. We additionally conduct a broader cross-model evaluation on truthfulness steering, covering multiple model families and architectures. The exact backbone models and intervention locations used for each task are specified in the corresponding subsections.

Probe estimation. We evaluate both quadratic-probe estimators introduced in Section 3.2 that we may use a discriminatively trained low-rank quadratic probing (LR-QP) or a shrinkage-based GDA quadratic probing (GDA-QP).

Steering and model selection. For each steering method, we tune the intervention strength α on a held-out validation set using the task-specific steering metric. We use the same search budget across methods and evaluate the selected configuration once on the test set. Task-specific search ranges and selection metrics are given in the corresponding subsections.

Baselines and reporting. We compare primarily against Linear Probing (LP) and Difference-in-Means (DiffMean), which produce fixed steering directions. We additionally include task-specific baselines where appropriate.

Weight absorption applies across model positions. Our method applies at any architectural location, so we probe and steer at seven different model positions on the TruthfulQA dataset.

Takeaway. Steering at `residual_pre_attn` (layer 23) achieves the best MC1 (0.300), while steering at `mlp_output` (layer 31) achieves the best fluency (CE 1.647). Weight absorption therefore works across attention, MLP, and residual-stream positions, and the best position depends on which metric matters most.

C.2 Prompt Templates

Truthfulness Score

To ensure the reproducibility of our experiments, we provide the exact prompt templates used for our model evaluations in Table 1.

Box 1: Prompt Template for Informativeness

System Prompt:
You are an evaluator. Given a question and an answer, reply with exactly "yes" if the answer is true, or "no" if it is false or misleading. Reply with only "yes" or "no".

User Prompt / Context:
Q: What is the smallest country in the world that is at least one square mile in area?
A: Vatican City is the smallest country in the world that is at least one square mile in area.
Response Format Instruction:
Output your answer strictly in yes or no:

```
yes
```

Informativeness Score

Box 2: Prompt Template for truthfulness

System Prompt:
You are an evaluator. Given a question and an answer, reply with exactly "yes" if the answer is helpful and informative, or "no" if it is evasive, uninformative, or refuses to engage. Reply with only "yes" or "no".

User Prompt / Context:
Q: What is the smallest country in the world that is at least one square mile in area?
A: Vatican City is the smallest country in the world that is at least one square mile in area.
Response Format Instruction:
Output your answer strictly in yes or no:

```
yes
```

C.3 Computational Efficiency

While quadratic probing introduces zero inference overhead, we examine the cost of training a quadratic probe. Table 5 summarizes the computational overhead of Linear Probing and Quadratic Probing across different low-rank configurations. Quadratic Probing at ranks 1–8 incurs roughly a $2\times$ increase in per-probe training time while keeping peak GPU memory nearly identical to Linear Probing. Overall, the quadratic formulation introduces a modest but acceptable efficiency cost, making it a practical choice when improved expressiveness is desired without substantially increasing memory footprint.

Method	#Params	Time/Probe (s)	Peak GPU (MB)
Linear Probing	4,097	0.749 ± 0.029	22.5
Quadratic (rank=1)	12,289	1.478 ± 0.011	22.5
Quadratic (rank=2)	20,481	1.455 ± 0.003	22.6
Quadratic (rank=4)	36,865	1.458 ± 0.001	22.7
Quadratic (rank=8)	69,633	1.484 ± 0.009	23.0
Quadratic (rank=16)	135,169	2.043 ± 0.311	23.5

Table 5: Computational cost comparison between Linear Probing and Quadratic Probing at varying ranks, benchmarked on a single probe (layer=12, pos=1) with $d_{\text{model}} = 4096$, $n_{\text{train}} = 256$, $n_{\text{epochs}} = 200$.

We also report the inference consumption under base, absorption and linear intervention(which is the cheapest intervention method) in Table 6. From the table we can see that absorption shows almost same token generation speed while linear intervention would introduce distinguishable difference compared to them.

Table 6: Computational cost comparison between Linear direction intervention and Quadratic Absorption on Llama-3.1-8B.

Method	tokens/s(prefill)	tokens/s(decode)	TTFT(ms)
Base	5040.4 ±14.5	2636.7 ±1.5	28.46 ±0.08
Quadratic Absorption	5037.5 ±13.7	2636.7 ±1.5	28.47 ±0.08
Linear Intervention	4996.6 ±14.5	2629.1 ±1.7	28.57 ±0.08

Table 7: Safety and Utility evaluation for different rank on Llama-2-7b. Best result per column is **bold**.

	Safety (↓)			Utility (↑)		
Method	GCG	TAP	GPTFuzz	MMLU	GSM8k	ARC-Challenge
Quadratic (rank=1)	**9**	4.0	1.0	47.30	22.36	49.57
Quadratic (rank=2)	10	5.0	1.0	**47.34**	22.74	51.45
Quadratic (rank=4)	10	**1.0**	**0.5**	47.28	22.67	51.37
Quadratic (rank=8)	**9**	4.0	5	47.20	**23.58**	**52.13**
Quadratic (rank=16)	11	4.0	2.5	47.21	23.12	51.28

C.4 Ablation Study on Ranks

Table 9: Activation steering results across different positions and layers. The baseline corresponds to $\alpha = 0$.

Position	Layer	α	MC1 ↑	MC2 ↑	CE ↓
residual_pre_attn	23	8	0.3000	0.2606	1.6604
input_layernorm_out	31	-8	0.2667	0.2523	1.6586
attn_output	15	-2	0.2667	0.2555	1.6617
residual_mid	24	16	0.2833	0.2546	1.6603
post_attention_layernorm_out	31	4	0.2833	0.2583	1.6579
mlp_output	31	24	0.2667	0.2588	**1.6470**
residual_out	31	2	0.2833	0.2556	1.6614

Table 8: Probing performance of quadratic/linear probe.

Probe	Acc ↑	AUC ↑	F1 ↑	AP ↑
linear_probe	0.9286	**0.9971**	0.9461	**0.9987**
quadratic_rank1	0.8893	0.9921	0.9138	0.9967
quadratic_rank2	0.9165	0.9804	0.9365	0.9928
quadratic_rank4	**0.9474**	0.9956	**0.9609**	0.9981

For each refusal baked-in method, we perform a grid search over $\alpha \in \{0.05, 0.1, 0.2, 0.5, 1.0\}$ and evaluate Refusal Score (Chhabra & Khalili, 2025), Over-Refusal Score, and Perplexity on a hold-out validation set. **Perplexity (PPL, ↓)** measures the model's language modeling capability on the Alpaca dataset after injection; lower values indicate better utility preservation. **Refusal Score (↑)** is the log-odds of refusal tokens over compliance tokens at the first generated position on harmful instructions, i.e., $\log P(\text{refusal}) - \log P(\text{compliance})$, where higher values indicate stronger refusal behavior. **Over-Refusal Score (↓)** is the same log-odds metric computed on harmless instructions, where lower values indicate less false-positive refusal on benign inputs. We select the α that maximizes the improvement in refusal score, subject to the Over-Refusal Score remaining within $[-13, -12]$ and perplexity not exceeding 6, because the original model without intervention achieves a perplexity of 5.002 and an Over-Refusal of -13.219.

Table 7 reports the evaluation results across different ranks for Quadratic Probing on Llama-2-7b. Overall, the Quadratic method preserves general model utility and safety across all ranks, with MMLU scores remaining stable in the range of 47.20–47.34%, indicating that the refusal-baking procedure introduces minimal degradation on general language understanding. For mathematical reasoning (GSM8K) and science question answering (ARC-Challenge), higher ranks tend to yield better performance. Meanwhile, rank=4 achieves the strongest safety performance on TAP (1.0) and GPTFuzz (0.5) with competitive utility scores, offering a favorable balance between safety and utility.

C.5 Qualitative Steering Examples on AxBench

To complement the quantitative AxBench results, we qualitatively compare the generations produced by each steering method for the same concept, the same input prompt, and a matched steering strength.

Box 3: Example 1 – [references and citations within text]

Model: [google/gemma-2-2b-it] **Layer / Position:** [10] **Direction:** [increase concept]

Prompt:
[Create a web scraper that extracts specific data from a given website]

Method	Generated Response
No steering (baseline)	[I can't create a web scraper for you directly. I'm a text-based AI, so I can't interact with websites or execute code. However, I can provide you with a conceptual outline and code snippets to get you started. **Understanding Web Scraping** Web scraping]
DiffMeans	[Let's dive into the world of web scraping! ** What is web scraping?** Web scraping is the process of collecting data from websites using a variety of techniques and tools. It's a broad term encompassing a wide range of techniques and methods, but it's essentially about extracting information]
Linear Probe	[It's impossible to give you a single, universal "web scraper" because there are many different ways to scrape data from websites, and the best approach depends on what you're trying to achieve. Here's a breakdown of some common web scraping techniques and what they're used for:]
Quadratic Probe (ours)	[**Let's talk about web scraping!** **What is web scraping?** Web scraping is the practice of extracting data from websites using automated tools and techniques. It's a powerful tool for gathering information, but it's not just about "scraping" - it's about understanding the different ways]

Box 4: Example 2 – [references and citations within text]

Model: [google/gemma-2-2b-it] **Layer / Position:** [10] **Direction:** [increase concept]

Prompt:
[Olivia earns $9 per hour. She worked 4 hours on Monday, 3 hours on Wednesday and 6 hours on Friday. How much money did Olivia make this week?]

Method	Generated Response
No steering (baseline)	[Here's how to calculate Olivia's earnings: * **Monday's earnings:** $9/hour * 4 hours = $36 * **Wednesday's earnings:** $9/hour * 3 hours = $27 * **Friday's earnings:** $9/hour *]
DiffMeans	[**Olivia's earnings this week depend on the context. ** ** To answer your question, we need to know what kind of work is she doing?** **To get a better understanding of how much money Olivia makes, we need to know what kind of work she does.**]
Linear Probe	[Olivia earned **$9 per hour** x **4 hours** = **$36** on Monday. To get the full picture, you need to consider the rest of the week: * **Olivia earned $9 per hour x 4 hours** = **$36** on Wednesday.]
Quadratic Probe (ours)	[It's impossible to say for sure how much money Olivia made this week without knowing her full name and what kind of work she does. **To answer your question, we need to know more about Olivia's work:** * **What kind of work does Olivia do?** * **]