Rotate2Think: Geometric Priming via Orthogonal Rotation to Improve Language Model Reasoning

Anonymous ACL submission

Abstract

Reasoning models achieve strong performance on challenging tasks by generating explicit intermediate reasoning traces before producing a final answer. Yet the internal structure of representation space when reasoning remains poorly understood: how do a model's hidden representations differ during thinking versus the embeddings of the input prompt, and can this structure be exploited to elicit stronger reasoning at inference time? We show that both input embeddings and thinking embeddings (mean-pooled last-layer hidden states over the prompt and reasoning trace, respectively) exhibit extremely high conicity, with all vectors clustering tightly around a single mean direction. Crucially, these mean input and thinking directions are non-collinear, with thinking embeddings occupying a geometrically distinct region of embedding space across many different models and benchmark tasks. This observation motivates casting the input-to-thinking transition as a rotation problem admitting a closed-form solution via orthogonal Procrustes analysis. We propose Rotate2Think, a training-free method that estimates this rotation from a small set of correctly solved examples and injects the resulting synthetic thinking vector between thinking delimiters at inference time, providing a geometric primer at the onset of the reasoning trace.Evaluated across multiple benchmarks and model families, Rotate2Think improves accuracy in 30 of 32 model–benchmark configurations across mathematics, science, and code tasks, and generalizes zero-shot to multimodal reasoning on MATH-Vision.

1 Introduction

The advent of chain-of-thought prompting (Wei et al., 2022) and large-scale reinforcement learning-based post-training (Shao et al., 2024; Ouyang et al., 2022) has given rise to a new class of *reasoning models*: language models that produce explicit intermediate reasoning traces, delimited by special thinking tokens, before generating a final answer (Jaech et al., 2024; Guo et al., 2025; Yang et al., 2025). These models achieve strong performance on challenging benchmarks spanning mathematical olympiad problems, graduate-level science questions, and programming tasks precisely because they can allocate additional test-time compute to the problem at hand (Snell et al., 2024). Yet despite this empirical success, remarkably little is understood about what geometrically distinguishes a model's hidden representations during thinking from those during direct generation. If the onset of a reasoning trace shapes the trajectory that follows, then steering the model into a better initial representational state could improve reasoning quality without spending additional test-time compute.

This motivates three main research questions about the geometry of reasoning: **RQ1.** How does the geometry of a model's hidden representations evolve from the embeddings of the input prompt to those of the reasoning trace? **RQ2.** If thinking corresponds to a specific geometric structure in representation space, can we prime the *start* of a reasoning trace, providing both reasoning and base models a stronger initial position from which they then reason fully? **RQ3.** If the input-to-thinking geometry is a property of the model rather than of the input distribution, does a rotation fit on text reasoning transfer zero-shot to other modalities, such as visual mathematical reasoning?

We approach these questions empirically by extracting and comparing two types of mean-pooled last-layer embeddings for each problem in a reasoning benchmark: the *input embedding* $e_{\text{in}}(q)$, obtained from a single pass over the user query without any generation, and the *thinking embedding* $e_{\text{th}}(q)$, the mean-pooled hidden state over the `<think>...</think>` span of a model-generated reasoning trace. This allows us to directly characterize how representational geometry evolves from the prompt to the reasoning trace. We discover a

striking regularity in both embedding sets: they exhibit extremely high *conicity* (Chandrahas et al., 2018), with all vectors clustering tightly around a single mean direction and average cosine similarity to that mean approaching 1.0. This holds consistently across diverse model families and benchmark domains spanning mathematics, science, and code. Crucially, with average cosine similarity of 0.66 across models and benchmarks tested, the mean direction of the input space (μ_{in}) and that of the thinking space (μ_{th}) are consistently misaligned relative to within-space alignment confirming that thinking embeddings occupy a geometrically distinct subspace, even as each space individually is nearly one-dimensional. Together, these two observations, high within-space conicity and cross-space misalignment, reduce the input-to-thinking mapping to a rotation problem admitting a closed-form solution via orthogonal Procrustes analysis (Schönemann, 1966). We show this rotation can be estimated from a small set of correctly solved examples drawn from held-out datasets, requiring no examples from the target distribution. Further, the resulting transformation reconstructs thinking embeddings with high fidelity across all models and benchmarks tested (cosine similarity > 0.96, mean square error < 0.05), validating both the geometric hypothesis and the cross-domain generality of the learned rotation.

We call this approach **Rotate2Think**. At inference time, we apply the fitted rotation to the mean-pooled input embedding of a new query to produce a synthetic thinking vector $\hat{e}_{\text{th}}(q)$, which is injected into the model's context as a single token position between the `<think>` and `</think>` delimiters via the model's `inputs_embeds` interface. This geometrically primes the onset of the reasoning trace, providing the model a stronger initial representational position from which it can reason. Rotate2Think is entirely training-free: it requires only a small collection of correctly solved examples from held-out datasets to fit the rotation, which amounts to a single $D \times D$ matrix computed in seconds and generalizes to unseen benchmarks without any target-domain examples. We make the following contributions:

1. **A geometric characterization of reasoning representations.** We show that input and thinking embeddings both form nearly one-dimensional cones with non-aligned axes across diverse model families and benchmarks, and that a single rotation maps between them with high fidelity (cosine similarity > 0.96, mean square error < 0.05 across models and benchmarks tested).

2. **Rotate2Think, a training-free inference-time method.** We fit an orthogonal Procrustes rotation from correctly solved examples on held-out datasets and inject the resulting synthetic thinking vector between thinking delimiters, geometrically priming the onset of the reasoning trace without any target-domain data or model modification.

3. **Cross-domain generalization.** A single rotation, fit on held-out datasets, improves accuracy in 30 of 32 model–benchmark configurations across mathematics, science, and code, spanning four model families from 4B to 31B parameters, with no target-domain examples or per-benchmark refitting.

4. **Zero-shot cross-modal transfer.** Fit purely on text reasoning, the same rotation transfers zero-shot to visual mathematical reasoning on MATH-Vision (testmini). On Gemma-4 E4B, Base+R2T (38.1%) exceeds full reasoning-mode accuracy (34.9%) on a modality the rotation never observed, suggesting the input-to-thinking geometry is a property of the model, not the input distribution.

2 Related Work

Test-Time Reasoning and Scaling. Chain-of-thought prompting (Wei et al., 2022) established that intermediate reasoning steps improve LLM performance, inspiring RL-trained reasoning models, o1 (Jaech et al., 2024), DeepSeek-R1 (Guo et al., 2025), Qwen3 (Yang et al., 2025), that generate extended thinking traces. Subsequent work has largely sought to *reshape* these traces: Snell et al. (2024) scale compute up via multi-response search and process verifiers; Muennighoff et al. (2025) manipulate trace length through budget forcing; Aghajohari et al. (2025) restructure the reasoning environment itself to achieve linear compute scaling. All require either additional compute at inference or gradient updates to the target model. Rotate2Think is orthogonal to these directions: operating on frozen models with no fine-tuning or additional inference cost, it geometrically primes the *start* of the reasoning trace with a single synthetic embedding. As these approaches manipulate the

trace itself, Rotate2Think could naturally complement them by providing a better-positioned initial state; we leave such combinations for future work.

Geometry of Transformer Representations. Rotate2Think rests on the empirical finding that both input and thinking embeddings exhibit extreme conicity; we use the Alignment-to-Mean and conicity metrics of Chandrahas et al. (2018). Ethayarajh (2019) first demonstrated that all layers of BERT, ELMo, and GPT-2 produce anisotropic cone-like representations; Rotate2Think's observed conicity of ≈ 1.0 (see Figure 1) in last-layer means of reasoning models is the extreme end of this spectrum. Gao et al. (2019) identify an underlying mechanism: auto-regressive LMs with weight tying collapse output embeddings into a narrow cone during training. Razzhigaev et al. (2024) show that decoder anisotropy peaks in upper layers and varies systematically with architecture and training phase, consistent with the high conicity Rotate2Think measures at the final layer. Prior work has proposed differentiable isotropy measures, finding that anisotropy can improve task performance (Rudman and Eickhoff, 2024), while Tsukagoshi and Sasano (2025) demonstrate that the intrinsic dimensionality of text embeddings is low enough that 25% of dimensions suffice for most tasks, corroborating the effective rank-one geometry that justifies fitting Procrustes from only a handful of solved examples. Karkada et al. (2026) show that translation symmetries in language co-occurrence statistics analytically predict manifold structure in model representations, providing a theoretical mechanism for why a systematic input-to-thinking rotation arises from training data structure. The Platonic Representation Hypothesis (Huh et al., 2024) further suggests that representational geometry converges universally with model scale, supporting the generalizability of the learned rotation across problem domains and model families. Finally, Sun et al. (2019) model knowledge-graph relations as rotations in complex vector space, establishing rotation as a natural and expressive transformation between semantic spaces; Rotate2Think instantiates this intuition in the real-valued mapping from input to thinking geometry in language models.

Representational Analysis and Steering. Park et al. (2024) formalize the linear representation hypothesis, showing that high-level semantic properties are encoded as linear directions in the representation space, providing some theoretical basis for treating the thinking embedding as a meaningful direction toward which the input embedding can be rotated. Representation Engineering (Zou et al., 2023) extracts population-level concept directions and adds them to intermediate activations to monitor and control model behavior. Turner et al. (2023) similarly construct contrastive steering vectors injected at chosen hidden layers. Rotate2Think shares this spirit but differs structurally: rather than adding to activations at a manually chosen layer, we inject a synthesized vector as a full token position between the thinking delimiters, letting the complete transformer stack process it naturally. Additionally, our focus is on the geometric structure specific to reasoning traces, and we intervene through orthogonal rotation rather than linear probing. Supporting this design, Esakkiraja et al. (2026) show that LLMs encode action decisions in pre-generation activations with reasoning text serving as post-hoc rationalization, directly motivating the use of the input embedding as the seed for rotation into thinking space.

3 Preliminaries

3.1 Notation and Embedding Extraction

Let $\mathcal{M}$ be a decoder-only transformer language model with L layers and hidden dimension D. For an input token sequence $\mathbf{x} = (x_1, \ldots, x_T)$, we write $\mathbf{h}^{(L)}(\mathbf{x}) \in \mathbb{R}^{T \times D}$ for the matrix of final-layer hidden states, where $\mathbf{h}_t^{(L)}(\mathbf{x})$ is the hidden state at position t.

Input embeddings. Given a query q formatted as a chat-template prompt of T tokens, the *input embedding* is the mean of the final-layer hidden states:

$$e_{\text{in}}(q) = \frac{1}{T} \sum_{t=1}^{T} \mathbf{h}_t^{(L)}(q) \in \mathbb{R}^D. \quad (1)$$

Computing $e_{\text{in}}(q)$ requires only a single forward pass through $\mathcal{M}$; no tokens are generated.

Thinking embeddings. Reasoning models generate an explicit reasoning trace delimited by special tokens like `<think>` and `</think>` (Yang et al., 2025; Guo et al., 2025). Given query q, let $\mathbf{r} = (r_1, \ldots, r_S)$ denote the S tokens generated within the thinking span. The *thinking embedding* is the mean of the final-layer hidden states at the

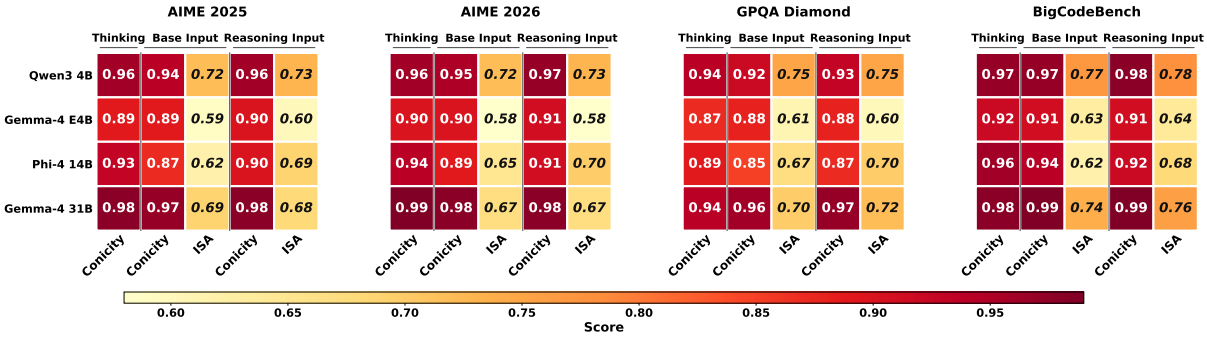

Figure 1: Within-space conicity is high (avg. ≈ 0.93), indicating tight cones in both thinking and input spaces. Cross-space ISA is markedly lower (avg. ≈ 0.66), confirming that input and thinking cones are narrow but don't share an axis, the structure Rotate2Think exploits.

thinking positions:

$$e_{\text{th}}(q) = \frac{1}{S}\sum_{s=1}^{S} \mathbf{h}^{(L)}_{|q|+s}([\mathbf{q}; r_1, \ldots, r_S]) \in \mathbb{R}^D, \tag{2}$$

where $[\mathbf{q}; r_1, \ldots, r_S]$ is the concatenated prompt-plus-thinking sequence and $|q|$ is the number of prompt tokens.

Computing $e_{\text{th}}(q)$ uses a two-phase procedure: (1) **Generation phase.** Run thinking-mode generation to produce the full response; record the token indices of the <think>/</think> boundaries in the generated sequence. (2) **Extraction phase.** Run a single forward pass on the concatenated prompt-plus-thinking sequence to obtain $\mathbf{h}^{(L)}$, then mean-pool over the thinking-span positions.

3.2 Conicity of Embedding Spaces

Chandrahas et al. (2018) introduced the notions of *alignment to mean* (ATM) and *conicity* to analyse the geometry of knowledge graph embedding spaces. The *alignment to mean* of a single vector v with respect to a set $\mathcal{V}$ is its cosine similarity to the set mean, $\text{ATM}(v, \mathcal{V}) = \cos(v, \boldsymbol{\mu}_{\mathcal{V}})$. *Conicity* is the average ATM over the set: given $\mathcal{V} = \{v_i\}_{i=1}^N \subset \mathbb{R}^D$ with mean $\boldsymbol{\mu}_{\mathcal{V}} = \frac{1}{N}\sum_i v_i$, the conicity of $\mathcal{V}$ is

$$\text{Conicity}(\mathcal{V}) = \frac{1}{N}\sum_{i=1}^{N} \cos(v_i,\ \boldsymbol{\mu}_{\mathcal{V}}), \tag{3}$$

where $\cos(u, v) = u^\top v/(\|u\|\|v\|)$.

$\text{Conicity}(\mathcal{V}) \in [-1, 1]$; a value near 1 indicates that all vectors point approximately in the direction of the mean, the set occupies a narrow *cone* around $\boldsymbol{\mu}_{\mathcal{V}}$ rather than being spread across the sphere.

We also introduce a complementary metric, Inter-Space Alignment (ISA), defined as the cosine similarity between the centroids of two vector sets. Formally, for sets $\mathcal{P}$ and $\mathcal{Q}$ we define $\text{ISA}(\mathcal{P}, \mathcal{Q}) = \cos(\boldsymbol{\mu}_{\mathcal{P}}, \boldsymbol{\mu}_{\mathcal{Q}})$, which quantifies the directional agreement between their mean representations. Low ISA values signify directional divergence between $\mathcal{P}$ and $\mathcal{Q}$, suggesting that the aggregate representations of the two sets occupy distinct or unrelated regions of the latent space.

3.3 The Orthogonal Procrustes Problem

Given paired matrices $\mathbf{A} = [a_1, \ldots, a_N]^\top \in \mathbb{R}^{N\times D}$ and $\mathbf{B} = [b_1, \ldots, b_N]^\top \in \mathbb{R}^{N\times D}$, the *orthogonal Procrustes problem* (Schönemann, 1966) seeks the orthogonal matrix $\mathbf{W} \in \mathbb{R}^{D\times D}$ (satisfying $\mathbf{W}^\top\mathbf{W} = \mathbf{I}_D$) that minimises the Frobenius-norm reconstruction error:

$$\mathbf{W}^* = \underset{\mathbf{W}:\ \mathbf{W}^\top\mathbf{W}=\mathbf{I}_D}{\text{argmin}} \left\|\mathbf{B} - \mathbf{A}\mathbf{W}^\top\right\|_F^2. \tag{4}$$

Schönemann (1966) showed that the closed-form solution is obtained via the SVD of the cross-covariance matrix $\mathbf{M} = \mathbf{B}^\top\mathbf{A}$:

$$\mathbf{M} = \mathbf{U}\,\boldsymbol{\Sigma}\,\mathbf{V}^\top \implies \mathbf{W}^* = \mathbf{U}\mathbf{V}^\top. \tag{5}$$

The solution $\mathbf{W}^*$ is an orthogonal matrix.

Centering. In practice, we mean-center both matrices before fitting. Let $\boldsymbol{\mu}_A$ and $\boldsymbol{\mu}_B$ be the column means of $\mathbf{A}$ and $\mathbf{B}$. We apply Equation 5 to the centered matrices $(\mathbf{A} - \mathbf{1}\boldsymbol{\mu}_A^\top)$ and $(\mathbf{B} - \mathbf{1}\boldsymbol{\mu}_B^\top)$, and account for the centroid shift at inference time:

$$\hat{e}_{\text{th}}(q) = \mathbf{W}^*(e_{\text{in}}(q) - \boldsymbol{\mu}_A) + \boldsymbol{\mu}_B. \tag{6}$$

4 Rotate2Think

Rotate2Think is a training-free, inference-time method that steers a language model toward its reasoning mode, with base models exhibiting "thinking" patterns without RL-based post-training and

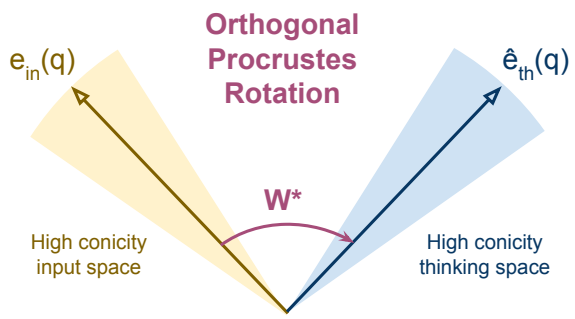

Figure 2: Schematic of the Rotate2Think method. An orthogonal Procrustes rotation $\mathbf{W}^*$ maps the mean-pooled input embedding $e_{\text{in}}(q)$ into thinking space. The resulting synthetic thinking vector $\hat{e}_{\text{th}}(q)$ is injected between thinking delimiters.

post-trained reasoning models showing improved reasoning. At the heart of the approach is the geometric observation from Section 3.2 and Figure 1 that both the input embedding space and the thinking embedding space are effectively one-dimensional cones, with distinct but consistently oriented mean directions. This structure makes the mapping from one space to the other computable with very few examples and expressible as a single orthogonal rotation.

The method operates in two stages:

1. **Fitting stage (offline).** Extract paired input and thinking embeddings for a set of training problems; fit an orthogonal Procrustes rotation $\mathbf{W}^*$ on the correctly-answered subset.

2. **Inference stage (online).** For each new query, compute a synthetic thinking vector via the fitted rotation, inject it between thinking delimiters using `inputs_embeds`, and generate the answer.

Figure 2 shows the schematic of Rotate2Think, while Algorithm 1 gives the full procedure, and the following subsections describe each component.

4.1 Rotation Fitting

Training example selection. We fit the Procrustes rotation using only problems the model answered correctly in thinking mode, on the hypothesis that correct solutions yield thinking vectors representative of productive reasoning, while incorrect ones may introduce noise that corrupts the rotation estimate. When fitting from K datasets other than the one being tested, $\mathcal{D}_1, \ldots, \mathcal{D}_K$, we sample up to 30 problems from each dataset and retain only those answered correctly in thinking mode. Let $\mathcal{C}_k \subseteq \mathcal{D}_k$ denote this correct subset for dataset k. We directly stack the individual input and thinking embeddings across all datasets into the paired matrices:

$$\mathbf{A} = \left[e_{\text{in}}(q_i)\right]_{\substack{k=1,\ldots,K \\ i \in \mathcal{C}_k}}^{\top}, \mathbf{B} = \left[e_{\text{th}}(q_i)\right]_{\substack{k=1,\ldots,K \\ i \in \mathcal{C}_k}}^{\top}, \tag{7}$$

where $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{N \times D}$ and $N = \sum_{k=1}^{K} |\mathcal{C}_k|$ is the total number of correctly answered examples across all source datasets. Each row of $\mathbf{A}$ and the corresponding row of $\mathbf{B}$ form a paired $(e_{\text{in}}, e_{\text{th}})$ example used to fit the Procrustes rotation.

SVD fit. We apply the centered Procrustes solution (Equations 4–5) to $\mathbf{A}$ and $\mathbf{B}$, obtaining $\mathbf{W}^*$, $\boldsymbol{\mu}_A$, and $\boldsymbol{\mu}_B$.

4.2 Rotated Inference

Synthetic thinking vector. Given a new test query q, we first compute $e_{\text{in}}(q)$ via a forward pass, then apply the rotation fitted in Equation 6 on K given datasets other than the one being evaluated to obtain the synthetic thinking vector $\hat{e}_{\text{th}}(q)$. This is a single matrix-vector multiply and incurs negligible overhead compared to a model forward pass.

Context construction and answer generation. We construct the `inputs_embeds` tensor ($\mathbf{E}_{\text{ctx}}$) for the generation call as a concatenation of four parts - $\mathbf{E}(q_{prompt})$, $\mathbf{E}$(`<think>\n`), $\hat{e}_{\text{th}}(q)$, and $\mathbf{E}$(`\n</think>`), where $\mathbf{E}(\cdot)$ is the model's token embedding lookup and $\hat{e}_{\text{th}}(q) \in \mathbb{R}^{1 \times D}$ occupies a single token position. The thinking delimiters surrounding the injected vector ensure that the positional context matches the format the model encounters during thinking-mode generation. We generate the answer by calling `model.generate(inputs_embeds=`$\mathbf{E}_{\text{ctx}}$`)`.

5 Experimental Setup

5.1 Datasets

We evaluate on five benchmarks spanning mathematics, science, code, and visual-mathematics chosen to represent diverse reasoning domains.

AIME 2025 and AIME 2026. The American Invitational Mathematics Examination (AIME) is a competition-level mathematics benchmark consisting of 30 integer-answer problems per year, drawn from number theory, algebra, combinatorics, and geometry. Problems require multi-step reasoning

Algorithm 1 Rotate2Think: fitting and inference

Require: Model $\mathcal{M}$, query q, pre-extracted paired embeddings $\{(e_{\text{in}}(q_i), e_{\text{th}}(q_i))\}_{i=1}^{N}$ with correctness labels $\{y_i \in \{0,1\}\}$ (up to 30 examples per dataset) for rotation dataset list $\mathcal{D}_1, \ldots, \mathcal{D}_K$ with $\mathcal{C}_k$ being the list of correct examples from dataset $\mathcal{D}_k$ for $k = 1, \ldots, K$

Ensure: Rotation parameters $(\mathbf{W}^*, \boldsymbol{\mu}_A, \boldsymbol{\mu}_B)$; answer for test query q

– Offline: Rotation Fitting –

$\mathbf{A} \leftarrow \left[e_{\text{in}}(q_i)\right]_{\substack{k=1,\ldots,K \\ i \in \mathcal{C}_k}}^{\top}$ ▷ Eq. 7

$\mathbf{B} \leftarrow \left[e_{\text{th}}(q_i)\right]_{\substack{k=1,\ldots,K \\ i \in \mathcal{C}_k}}^{\top}$ ▷ Eq. 7

$\boldsymbol{\mu}_A \leftarrow \text{mean}(\mathbf{A})$; $\boldsymbol{\mu}_B \leftarrow \text{mean}(\mathbf{B})$

$\mathbf{M} \leftarrow (\mathbf{B} - \mathbf{1}\boldsymbol{\mu}_B^{\top})^{\top}(\mathbf{A} - \mathbf{1}\boldsymbol{\mu}_A^{\top})$

$\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\top} \leftarrow \text{SVD}(\mathbf{M})$

$\mathbf{W}^* \leftarrow \mathbf{U}\mathbf{V}^{\top}$ ▷ Eq. 5

– Online: Inference for test query q –

$e_{\text{in}}(q) \leftarrow$ forward pass on prompt tokens ▷ no generation needed

$\hat{e}_{\text{th}}(q) \leftarrow \mathbf{W}^*(e_{\text{in}}(q) - \boldsymbol{\mu}_A) + \boldsymbol{\mu}_B$ ▷ Eq. 6

$\mathbf{E}_{\text{ctx}} \leftarrow [\mathbf{E}(q_{\text{prompt}}) \mid \mathbf{E}(\texttt{<think>\textbackslash n}) \mid \hat{e}_{\text{th}}(q) \mid \mathbf{E}(\texttt{\textbackslash n</think>})]$

return $\mathcal{M}.\texttt{generate}(\mathbf{E}_{\text{ctx}})$

and yield an integer answer in $[0, 999]$. We use the official 2025 and 2026 sets, treating each year as a separate benchmark (Balunovic et al., 2025).

GPQA Diamond. GPQA (Rein et al., 2024) is a multiple-choice benchmark of graduate-level science questions authored by domain experts. We evaluate on the Diamond split (198 questions spanning Biology, Chemistry, and Physics), the hardest subset, which requires deep subject-matter knowledge and eliminates most surface-level shortcuts.

BigCodeBench. BigCodeBench (Zhuo et al., 2025) is a function-level Python code generation benchmark with 1,140 diverse tasks covering library usage, data manipulation, and algorithm implementation. Each task is evaluated by executing the generated code against a suite of unit tests.

MATH-Vision (testmini). MATH-Vision (Wang et al., 2025) is a multimodal mathematical problem-solving benchmark with 3040 visual reasoning problems in the full test set. We use the `testmini` split of the dataset containing 304 problems.

5.2 Baselines and Evaluation Protocol

We report two metrics for each condition: **Acc** (accuracy, %), and **Tok** (average number of tokens generated per problem). For each model-benchmark pair, we evaluate four modes:

- **Base.** Standard (non-thinking) generation: the model generates its answer directly without producing a reasoning trace.
- **Reasoning.** Full thinking-mode generation: the model produces a complete reasoning trace before its answer.
- **Base+R2T.** The rotation is fitted from the base model variant's input and to the reasoning model's thinking embeddings. During inference, the base model receives the synthetic thinking vector injected between thinking delimiters.
- **Reasoning+R2T.** The rotation is fitted from the reasoning model's embeddings. The same reasoning model is used for inference, with the synthetic thinking vector injected between thinking delimiters.

5.3 Models Evaluated

We evaluate four language models spanning three families and a range of parameter counts.

Qwen3 4B. `Qwen/Qwen3-4B-Instruct-2507` (which we use as base model) and `Qwen/Qwen3-4B-Thinking-2507` (which we use as reasoning model) (Yang et al., 2025) are 4B-parameter dense language models.

Phi-4 14B. We use `microsoft/phi-4` as the base model and `microsoft/Phi-4-reasoning-plus` as the reasoning model. These are 14B, dense models (Abdin et al., 2024).

Gemma-4 E4B and 31B. We use `google/gemma-4-E4B-it` (8B MoE model) and `google/gemma-4-31B-it` (31B dense model) (Google DeepMind, 2026) in our evaluation.

The reasoning versions of Qwen3 4B, and Phi-4 14B use `<think>`/`</think>` delimiters to bracket the reasoning trace. Gemma-4 models, on the other hand, use the `<|think|>` flag to toggle between non-thinking (which we use as base model for Gemma-4) and thinking

Benchmark	Model	Base		Base+R2T		Reasoning		Reasoning+R2T	
		Acc %	Tok	Acc %	Tok	Acc %	Tok	Acc %	Tok
AIME 2025	Qwen3 4B	43.33	*10444*	**46.67**	*8299*	70.00	*20944*	**80.00**	*20894*
	Gemma-4 E4B	36.67	*3438*	**40.00**	*5408*	40.00	*6968*	**50.00**	*5891*
	Phi-4 14B	16.67	*1672*	**20.00**	*1639*	73.33	*15053*	**76.67**	*17108*
	Gemma-4 31B	70.00	*2927*	**73.33**	*4110*	83.33	*7170*	**86.67**	*4240*
AIME 2026	Qwen3 4B	40.00	*9359*	**53.33**	*8855*	70.00	*20609*	**73.33**	*21076*
	Gemma-4 E4B	33.33	*3691*	**46.67**	*5422*	43.33	*6829*	**53.33**	*5586*
	Phi-4 14B	10.00	*1380*	**16.67**	*1515*	76.67	*17077*	**80.00**	*17540*
	Gemma-4 31B	80.00	*2119*	**83.33**	*2036*	83.33	*7894*	**86.67**	*3098*
GPQA Diamond	Qwen3 4B	46.67	*737*	**48.48**	*549*	67.17	*8018*	**69.19**	*9126*
	Gemma-4 E4B	55.07	*1979*	**64.00**	*3254*	58.67	*3646*	60.33	*3342*
	Phi-4 14B	54.55	*658*	**56.57**	*746*	69.19	*11353*	**69.70**	*11569*
	Gemma-4 31B	74.24	*1284*	**78.28**	*1390*	84.34	*5728*	**86.87**	*5614*
BigCodeBench	Qwen3 4B	38.24	*409*	**38.51**	*404*	41.58	*5787*	**41.84**	*5670*
	Gemma-4 E4B	36.67	*608*	**38.38**	*1238*	36.84	*1227*	**37.98**	*1252*
	Phi-4 14B	**39.91**	*242*	39.74	*582*	43.68	*4172*	**44.03**	*3962*
	Gemma-4 31B	46.49	*364*	**47.54**	*1577*	**48.51**	*2198*	48.25	*2291*

Table 1: Accuracy (%) and average tokens generated (Tok). Rotate2Think improves accuracy in 30 of 32 model–benchmark configurations, with comparable token budgets. R2T = Rotate2Think.

(which we use as reasoning model for Gemma-4) modes with model-specific thinking delimiters (`<|channel>`/`<channel|>`) and enforce an exclusive-or constraint between `input_ids` and `inputs_embeds`. We handle this via a forward hook that replaces a designated placeholder token's embedding with the synthetic thinking vector during prefill. We use the `4-bit` quantized version of Gemma-4 31B model.

6 Results

Geometric structure of reasoning hidden states. Figure 1 reports conicity and ISA scores across all models and benchmarks. Conicity is uniformly high across all conditions (0.87–0.99), confirming that reasoning model hidden states occupy a narrow cone in the representation space, which is a prerequisite for the low-rank geometric characterization that Rotate2Think exploits. ISA values between input and thinking embeddings are substantially lower on average (0.659 base, 0.670 reasoning), indicating that input and thinking cones are individually narrow but do not share an axis, motivating our Procrustes-based rotation. Both properties hold consistently across benchmarks and model families, suggesting that high conicity and low ISA are stable geometric properties of current reasoning models.

Base → Base+Rotate2Think. Injecting a synthetic thinking vector into base models (Base+R2T) yields consistent accuracy gains across every model and benchmark tested, as seen in Table 1. AIME improvements reach up to +13 percentage points. On GPQA Diamond and AIME 2026, Gemma-4 E4B under Base+R2T surpasses the same model in full reasoning mode. These results suggest the geometric vector carries a genuine reasoning signal independent of the chain-of-thought length. Beyond accuracy, we qualitatively observe that Qwen3-4B under Rotate2Think produces self-correcting phrases such as "Wait…" and "Let me reconsider…" — patterns characteristic of RL-trained reasoning models but absent in standard base-model generation. This behavior does not appear consistently across Phi-4 and Gemma-4, so we report it as an incidental observation rather than a general finding.

Reasoning → Reasoning+Rotate2Think. As seen in Table 1, augmenting reasoning models with the Rotate2Think primer yields consistent accuracy improvements across all models and benchmarks, with gains present regardless of model scale or task domain. On AIME, improvements range from modest to substantial with Gemma-4 E4B gaining +10 percentage points on both AIME 2025 and AIME 2026, while Qwen3 4B improves by +10 and +3.3 points, respectively. GPQA Diamond and BigCodeBench follow the same trend, with almost all models registering gains under Reasoning+R2T. Notably, these improvements occur without a commensurate increase in token generation, suggesting that R2T does not prompt the model to reason more, but rather more effectively. The synthetic thinking vector appears to prime the model's

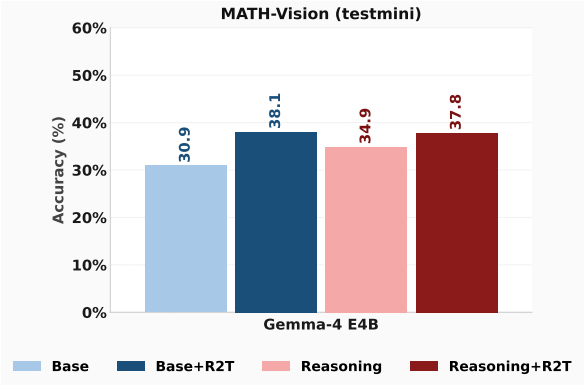

Figure 3: Rotate2Think results on MATH-Vision (testmini) for Gemma-4 E4B. The rotation is fit on text benchmarks only and applied zero-shot to a multimodal benchmark. Base+R2T (38.1) exceeds Reasoning (34.9) and approaches Reasoning+R2T (37.8), described further in Section 6.

internal reasoning process, improving the quality of the chain-of-thought that follows.

Generalizing to Multimodal Reasoning. To examine whether Rotate2Think transfers beyond language-only inputs, we evaluate on MATH-Vision (testmini) using Gemma-4 E4B, with the rotation matrix fit on 30 examples each from the four text benchmarks (AIME 2025, AIME 2026, GPQA Diamond, BigCodeBench) — without any multimodal data or modality-specific adaptation. As shown in Figure 3, Rotate2Think improves accuracy for both the base model and the reasoning model, consistent with the pattern observed on text benchmarks. That these gains carry over to a multimodal LLM (MLLM) without any modification suggests Rotate2Think may extend naturally to multimodal settings, broadening its practical applicability beyond language-only reasoning.

7 Discussion

Taken together, the results in Section 6 support a coherent picture of what Rotate2Think is doing.

The thinking embedding is a meaningful priming signal. Across diverse model families and parameter scales, rotating the input embedding into thinking space and injecting a single token consistently steers the model toward stronger reasoning behavior on hard problems. This is consistent with the geometric hypothesis in Section 3.2: if both input and thinking embeddings form tight cones with non-aligned axes, then a single rotation can reliably shift the model's context from one regime to the other.

Reasoning capacity may be latent and geometrically accessible. Rotate2Think intervenes at a single point (one rotated embedding, one injected token) with no gradient updates, no additional training, and no modification to model weights. That such a minimal perturbation produces consistent accuracy gains across both base and reasoning models hints that the injected thinking vector may provide a useful initialization of the reasoning process, steering the model toward a representational regime where reasoning is more reliably expressed.

Thinking-space geometry transcends input modality. The transfer of Rotate2Think to MATH-Vision (testmini), with the rotation fit purely on text reasoning, suggests that the thinking-space geometry is a property of the model rather than of the input modality. That such a transfer occurs without any multimodal data or modality-specific refitting is consistent with the Platonic Representation Hypothesis (Huh et al., 2024), and hints that the learned rotation may capture a modality-agnostic signature of the model's reasoning regime.

8 Conclusion

We introduced Rotate2Think, a training-free method that improves language model reasoning by rotating the mean-pooled input embedding into "thinking space" via orthogonal Procrustes analysis and injecting the result as a single token between thinking delimiters. Our key geometric insight is that input and thinking embeddings both exhibit high conicity, i.e., they occupy narrow cones in representation space, but with systematically misaligned mean directions, reducing the cross-space mapping to a pure rotation. Rotate2Think consistently improves accuracy for both base and reasoning models across benchmarks and LLMs evaluated. Notably, Rotate2Think also extends to multimodal settings as evaluated on MATH-Vision (testmini), it improves reasoning accuracy in MLLMs without any modality-specific adaptation, suggesting that the thinking-space geometry exploited by Rotate2Think may not be exclusive to language-only pretraining, and could reflect a more general structural property of the representation space.

Limitations

There are limitations of the current work that warrant acknowledgment. First, Rotate2Think in its current form requires access to a reasoning-capable variant of the target model to extract thinking em-

beddings; it cannot be applied to model families for which no such variant exists. Second, while gains are consistent across mathematical reasoning and scientific knowledge benchmarks, improvements on code generation tasks are modest, suggesting the method's effectiveness is tied to task domains where extended deliberation is most consequential. Third, all evaluations are conducted under a single-token injection scheme; whether more expressive interventions, such as injecting a sequence of rotated embeddings, for instance, could yield further gains is an open question. Finally, Rotate2Think operates directly on hidden state representations and therefore requires white-box access to the model's internals, specifically, the last hidden state of the input sequence. This precludes application to closed-source models, where only token-level outputs are accessible, and limits the method to open-weight models for which activation extraction is feasible. We hope this perspective encourages further work at the intersection of representational geometry and inference-time reasoning, and that the lightweight nature of Rotate2Think makes it a practical building block for future reasoning systems.

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, et al. 2024. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*.

Milad Aghajohari, Kamran Chitsaz, Amirhossein Kazemnejad, Sarath Chandar, Alessandro Sordoni, Aaron Courville, and Siva Reddy. 2025. The markovian thinker: Architecture-agnostic linear scaling of reasoning. *arXiv preprint arXiv:2510.06557*.

Mislav Balunovic, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. 2025. Matharena: Evaluating llms on uncontaminated math competitions. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Chandrahas, Aditya Sharma, and Partha Talukdar. 2018. Towards understanding the geometry of knowledge graph embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 122–131, Melbourne, Australia. Association for Computational Linguistics.

Esakkivel Esakkiraja, Sai Rajeswar, Denis Akhiyarov, and Rajagopal Venkatesaramani. 2026. Therefore i am. i think. *arXiv preprint arXiv:2604.01202*.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 55–65.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tieyan Liu. 2019. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations*.

Google DeepMind. 2026. Gemma 4 model card. https://ai.google.dev/gemma/docs/core/model_card_4.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. 2024. The platonic representation hypothesis. *arXiv preprint arXiv:2405.07987*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Dhruva Karkada, Daniel J Korchinski, Andres Nava, Matthieu Wyart, and Yasaman Bahri. 2026. Symmetry in language statistics shapes the geometry of model representations. *arXiv preprint arXiv:2602.15029*.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. 2025. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20286–20332.

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

Kiho Park, Yo Joong Choe, and Victor Veitch. 2024. The linear representation hypothesis and the geometry of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, pages 39643–39666.

Anton Razzhigaev, Matvey Mikhalchuk, Elizaveta Goncharova, Ivan Oseledets, Denis Dimitrov, and Andrey Kuznetsov. 2024. The shape of learning: Anisotropy and intrinsic dimensions in transformer-based models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 868–874.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First conference on language modeling*.

William Rudman and Carsten Eickhoff. 2024. Stable anisotropic regularization. In *International Conference on Learning Representations*, volume 2024, pages 23594–23609.

Peter H Schönemann. 1966. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10.

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

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Hayato Tsukagoshi and Ryohei Sasano. 2025. Redundancy, isotropy, and intrinsic dimensionality of prompt-based text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 25915–25930.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*.

Ke Wang, Junting Pan, Linda Wei, Aojun Zhou, Weikang Shi, Zimu Lu, Han Xiao, Yunqiao Yang, Houxing Ren, Mingjie Zhan, et al. 2025. Mathcoder-vl: Bridging vision and code for enhanced multimodal mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 2505–2534.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

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

Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, et al. 2025. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions. In *International Conference on Learning Representations 2025*, pages 99488–99542.

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

A Appendix: Hyperparameters

A.1 Generation

All single-run experiments, except Phi-4 (base), use sampling. Sampling parameters follow each model family's recommended settings from their respective `huggingface` model cards and are listed in Table 2.

A.2 Rotation Fitting

The orthogonal Procrustes rotation $\mathbf{W}^*$ is fitted offline using a deterministic SVD (no gradient descent, no learning rate). Fitting proceeds as follows:

- At most **30** examples per dataset are considered (`MAX_ROTATION_SAMPLES = 30`).
- Only correctly-answered examples are retained.
- The mean input embedding and mean thinking embedding are computed per dataset over the retained examples.
- These per-dataset means are stacked and used to fit a single Procrustes rotation with centroid alignment.

For the multimodal transfer experiment (MATH-Vision), the rotation is fitted on **120 examples** in total (4 text datasets $\times$ 30 samples each: AIME 2025, AIME 2026, GPQA Diamond, and BigCodeBench), and applied zero-shot to visual problems.

A.3 Quantization

Gemma-4 31B is loaded in **4-bit** (NF4 + double quantization, `bfloat16` compute dtype) via `bitsandbytes`. All other models are loaded in `bfloat16` without quantization.

Model	temperature	top_k	top_p
Gemma-4 E4B / 31B	1.0	64	0.95
Qwen3 4B (base)	0.7	20	0.80
Qwen3 4B (reasoning)	1.0	20	0.95
Phi-4 14B (base)	–	–	–
Phi-4 14B (reasoning)	0.6	20	0.95

Table 2: Sampling hyperparameters used per model family.

A.4 Embedding Extraction

Embeddings are obtained by mean-pooling the last-layer hidden states over all non-padding token positions. Procrustes fitting is performed in `float64`; injected synthetic thinking vectors are cast to `bfloat16` for inference.