BIFROST: STEERING STRATEGIC TRAJECTORIES TO BRIDGE CONTEXTUAL GAPS FOR SELF-IMPROVING AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Autonomous agents excel in self-improvement through reflection and iterative refinement, which reuse successful task trajectories as in-context examples to assist subsequent reasoning. However, shifting across tasks often introduces a context mismatch. Hence, existing approaches either discard the trajectories or manipulate them using heuristics, leading to a non-negligible fine-tuning cost or unguaranteed performance. To bridge this gap, we reveal a context-trajectory correlation, where shifts in context are highly parallel with shifts in trajectory. Based on this finding, we propose BrIdge contextual gap FoR imprOvised trajectory STeering (Bifrost), a training-free method that leverages context differences to precisely guide the adaptation of previously solved trajectories towards the target task, mitigating the misalignment caused by context shifts. Our trajectory adaptation is conducted at the representation level using agent hidden states, ensuring trajectory transformation accurately aligns with the target context in a shared space. Empirical experiments reveal that Bifrost consistently surpasses existing trajectory reuse and fine-tuned self-improvement methods by 2%–14% across adaptation tasks in math problem solving (AQUA to GSM8K), question answering (ARC to GPQA), and code generation (HumanEval to LiveCodeBench), demonstrating that agents can effectively leverage past experiences despite substantial context shifts.

1 INTRODUCTION

Autonomous agents have fundamentally transformed fields like robotics (Bousmalis et al., 2023; Sharma et al., 2023; Puthumanaillam et al., 2025) and autonomous driving (Mei et al., 2024; Dagdanov et al., 2022) by harnessing the power of self-improvement mechanisms (Weng et al., 2023; Madaan et al., 2023; Gou et al., 2023; Pang et al., 2023; Bo et al., 2024; Zhang et al., 2024). By integrating systemic reflection and autonomous error correction, these agents iteratively evolve their decision-making to master increasingly high-dimensional tasks (Wang et al., 2023; Zhao et al., 2024; Qiu et al., 2025; Zheng et al., 2025; Sarch et al., 2024; Zhang et al., 2025a; Jin et al., 2025; Ma et al., 2023). To optimize this evolutionary process, emerging research focuses on the strategic reuse of successful task trajectories as in-context demonstrations (Wang et al., 2023; Lingam et al., 2025; Zheng et al., 2025; Sarukkai et al., 2025). This paradigm allows agents to capitalize on historical successes (Xie et al., 2021; Lin & Lee, 2024; Sarch et al., 2024; Lin et al., 2025), significantly enhancing efficiency while reducing the computational burden of iterative self-reflection (Brown et al., 2020).

Despite the promise of trajectory reuse, existing self-improvement frameworks remain suboptimal when faced with contextual non-stationarity across tasks. Current approaches often treat past trajectories as context-specific artifacts (Shinn et al., 2023; Yang et al., 2024; Lin et al., 2025; Zhao et al., 2024; Yuan & Xie, 2025), either discarding them to rely on computationally intensive fine-tuning (Rebuffi et al., 2017; Zweiger et al., 2025; Narayanan et al., 2024; Pang et al., 2024; Costello et al., 2025; Amin et al., 2025) or attempting to repurpose them via static heuristics (Bensal et al., 2025; Qu et al., 2024; Tajwar et al., 2025; Fu et al., 2025; Lin et al., 2025; Ge et al., 2025). However, because these trajectories are inherently coupled with their original task environments, such a direct calibration leads to significant contextual misalignment (An et al., 2023; Lin & Lee, 2024; Gai et al., 2025; Afonin et al., 2025). This mismatch not only degrades the generalization capabilities but

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.

can also actively hinder their performance when applied to novel domains (Fu et al., 2025; Huang et al., 2023; Hong et al., 2024). Consequently, the fundamental principle of cross-context trajectory adaptation remains a critical and largely unexplored frontier (further discussion in App. B).

To address this challenge, we uncover a fundamental context-trajectory correlation in the agentic representation space: shifts in task context are strikingly parallel to shifts in corresponding successful trajectories when projected onto the latent manifold. Leveraging this geometric insight, we introduce **BrIdge** contextual gap **FoR** impr**O**vised trajectory **S**teering (**Bifrost**), a training-free framework that steers historical trajectories toward target task requirements via context differentials (Fig. 1). By modeling task transitions through hidden-state representations, Bifrost aligns prior experiences with the target context, enabling the agent to repurpose past trajectories as accurate in-context examples, facilitating seamless adaptation without parameter updates or environment-specific fine-tuning.

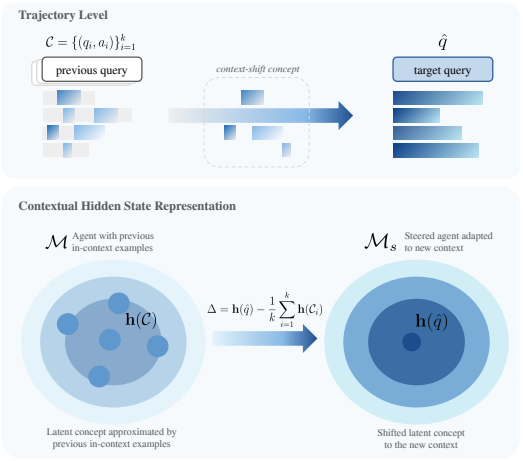


Figure 1: Illustration of Bifrost: It identifies the context-shift concept between previous and target tasks, which contains essential knowledge for bridging the context gap. Further, by steering along the context shift direction, Bifrost helps find an optimal strategy that is effective under the target context.

In practice, Bifrost identifies a steering vector that maps historical trajectories onto the target task manifold. Rather than relying on untargeted or global calibration, it uses the target context as a dynamic anchor to suppress context-specific artifacts while preserving transferable knowledge, enabling exploitation of prior trajectories even across substantial domain gaps, enhancing cross-context generalization, and eliminating the performance degradation typically caused by context shifts.

Across diverse benchmarks, Bifrost consistently outperforms trajectory reuse and finetuned self-improvement methods, demonstrating that our proposed method enables self-improving agents to effectively leverage past experiences despite substantial context shifts. Our contributions are:

- We reveal the context-trajectory correlation (Sec. 2.1), where context shifts are highly parallel with trajectory shifts. Based on this observation, we propose Bifrost (Sec. 2.2), a training-free method that induces context differences to guide agents leveraging previous trajectories as in-context examples toward a target task.
- We provide theoretical analyses (Sec. 3), showing that Bifrost can generalize experiences learned from prior trajectories to adapt toward a target task.
- We conduct experiments on diverse benchmarks (Sec. 4), including question answering, math reasoning, and code generation, with different model sizes. Our experiments show that Bifrost consistently outperforms other popular baselines and state-of-the-art methods based on trajectory reuse or finetuning.

2 Methodology

Our work focuses on how a Large Language Model (LLM) agent can effectively leverage its successful trajectories solved in the past to adapt to a target task under a substantial context shift. Let $\mathcal{M}$ be an LLM that solved tasks under a previous context where successful trajectories are preserved as $\mathcal{C} = \{(q_i, a_i)\}_{i=1}^{k}$. Here, each pair (q_i, a_i) denotes a trajectory consisting of a query and its corresponding reasoning process leading to the answer. Our objective is to find an optimal reasoning strategy by leveraging previous trajectories that maximizes the log likelihood of the LLM agent on the target task $\hat{\mathcal{C}} = \{(\hat{q}_i, \hat{a}_i)\}_{i=1}^{n}$,

$$\mathcal{S}^* = \arg\max_{\mathcal{S}} \mathbb{E}_{(\hat{q},\hat{a})\in\hat{\mathcal{C}}} \left[\log \mathcal{M}_{\mathcal{S}}(\hat{a} \mid \hat{q})\right], \tag{1}$$

where $\mathcal{M}_{\mathcal{S}}$ denotes that the model $\mathcal{M}$ reasons following the reasoning strategy $\mathcal{S}$. Although the strategy $\mathcal{S}$ succeeded under the previous context $\mathcal{C}$, it might not align with the target context $\hat{\mathcal{C}}$. Therefore, instead of fully fine-tuning the agent or naively reusing previous strategies, we aim to

bridge the contextual gap between previous tasks and any target tasks by proposing Bifrost, which can improvise under any target contexts by properly steering the trajectory of an existing strategy. As a result, Bifrost improves In-Context Learning (ICL) via an optimal reasoning strategy tailored for the target context, as shown in Fig. 1. To demonstrate Bifrost, we first reveal our findings of context-trajectory correlation in Sec. 2.1. Further, based on this finding, we perform trajectory steering to adapt previous trajectories to the target task in Sec. 2.2.

2.1 Context-Trajectory Correlation

To conduct proper reasoning under a target context, it is vital to employ an effective trajectory. Therefore, we investigate the correlation between trajectories and context under task shift. To accomplish this, we follow Linear Representation Hypothesis (Park et al., 2023) by regarding LLMs as an embedding-unembedding process.

Particularly, the embedding encodes context inputs into hidden states, *e.g.*, embedding vectors, which contain essential concepts that enable knowledge generalization across tasks. For example, "king" and "queen" are conceptually related to "man" and "woman", respectively. Formally, for a context text $x = (q_i, a_i)$, we assume the hidden state mapped by an LLM is $\mathbf{h}(x)$ in a representation space Λ. Further, the model represents each word y in its vocabulary into an unembedding vector $\mathbf{g}(y)$ in a separate space Γ with the same dimension. To this end, the probability distribution for next-word prediction is formulated as:

$$P(y \mid x) \propto \exp(\mathbf{h}(x)^\top \mathbf{g}(y)). \tag{2}$$

Intuitively, the reasoning process can be explained as two representation spaces that are connected via an inner product, where LLMs aim to find the best match between a context input and a reasoning output. By conducting such a process iteratively, the input-output logic chain formulates a strategic trajectory that is optimized for a certain context.

In agentic AI, it is common to apply agents across various tasks, thus encountering a novel context. To ensure the target task is solvable, we have the following assumption:

Assumption 1. For an LLM agent $\mathcal{M}$ capable of solving task A and task B with contexts $\mathcal{C}^A$ and $\mathcal{C}^B$, respectively, the concepts $\mathcal{W}^A = \{W_i^A\}_{i=1}^k$ and $\mathcal{W}^B = \{W_i^B\}_{i=1}^n$ extracted from task A and task B must satisfy:

- The vocabulary space of $\mathcal{M}$ covers $\mathcal{W}^A \cup \mathcal{W}^B$.
- The shared concepts $\mathcal{W}^A \cap \mathcal{W}^B$ are causally separable.
- For any private concepts $W^A \in \mathcal{W}^A \setminus \mathcal{W}^B$ and $W^B \in \mathcal{W}^B \setminus \mathcal{W}^A$, the embedding-unembedding vectors satisfy the subspace independence $\mathbf{h}_{W^A}^\top \mathbf{g}_{W^B} = 0$.

To handle a target context $\hat{\mathcal{C}}$ specific to concept W, the contextual gap can be captured by a steering vector $\bar{\mathbf{h}}_W$ scaled by α in the hidden states:

$$\hat{\mathbf{h}} = \mathbf{h} + \alpha \bar{\mathbf{h}}_W. \tag{3}$$

Considering a binary-value concept, *e.g.*, man and woman, the probability of the output Y taking value $Y(1)$ versus its counterfactual $Y(0)$ is measured by the logit. According to the measurement representation theorem in (Park et al., 2023), such a logit is linear with respect to $\mathbf{h}$:

$$\text{logit}\, P(Y = Y(1) \mid \mathbf{h}; Y \in \{Y(0), Y(1)\}) = \beta \mathbf{h}^\top \bar{\mathbf{g}}_W, \tag{4}$$

where $\bar{\mathbf{g}}_W$ is the unembedding representation (Park et al., 2023) of the task and $\beta > 0$ is a function of $\{Y(0), Y(1)\}$. We show that output tokens change linearly with the contextual shift as follows.

Theorem 1. *Under the Linear Representation Hypothesis, for a concept W satisfying the conditions in Asm. 1, for an embedding vector $\bar{\mathbf{h}}_W$ and unembedding vector $\bar{\mathbf{g}}_W$ of W, the change of output logits is a linear function of the magnitude of the context shift $\beta \bar{\mathbf{h}}_W$.*

Proof. Given the target context hidden state $\hat{\mathbf{h}}$, we substitute the shift according to Eq. (3):

$$\text{logit}\ P(Y = Y(1) \mid \bar{\mathbf{h}}) = \beta(\mathbf{h} + \alpha \bar{\mathbf{h}}_W)^\top \bar{\mathbf{g}}_W = \beta \mathbf{h}^\top \bar{\mathbf{g}}_W + \alpha(\beta \bar{\mathbf{h}}_W^\top \bar{\mathbf{g}}_W). \tag{5}$$

Note that β is fixed under a certain concept, and $\bar{\mathbf{g}}_W$ denotes the concept direction based on the model understanding, *e.g.,*, "king" to "queen" and "man" to "woman" have the same shifting direction when given a fixed model. Let $K = \beta \bar{\mathbf{g}}_W$, we have

$$\text{logit}\, P(Y = Y(1) | \bar{\mathbf{h}}) = \text{logit}\, P(Y = Y(1) | \mathbf{h}) + \alpha K \bar{\mathbf{h}}_W. \tag{6}$$

By defining the shift of logit distribution under different context hidden states $\Delta_P = P(Y = Y(1) | \hat{\mathbf{h}}) - P(Y = Y(1) | \mathbf{h})$, we have $\Delta_P \propto \alpha K \bar{\mathbf{h}}_W$. □

Intuitively, a trajectory is composed of a sequence of reasoning steps or actions. This theorem shows that the selection of an optimal trajectory based on an existing one can be guided by the context shift.

Discussion. The above results are focused on one concept, which can be generalized to complex tasks containing multiple concepts. For concepts that are not shared across tasks, the subspace independence ensures that the embedding space is irrelevant to the unembedding space. Thus, the corresponding prediction logits are not affected by the context shift. Moreover, for multiple shared concepts, the causal separability ensures that they are not confusing each other. For example, adapting the context "He speaks English" to a target context "She speaks French" must be separated from adapting to another target "He speaks French".

Experimental demonstration. We conduct a small-scale experiment to demonstrate the context–trajectory correlation described in Thm 1. We randomly select 100 context-separated samples from the AQUA and GSM8K datasets, forming 50 paired examples. These pairs are used to compute contextual direction vectors from AQUA to GSM8K. We hypothesize that when an agent solves AQUA tasks, the trajectories steered toward GSM8K exhibit higher similarity to GSM8K trajectories than to those generated by the non-steered model on AQUA. Tab. 1 shows a strong statistical significance over varied values of the control parameter α. Further details of the hypothesis testing are described in App. C.1.

Table 1: Hypothesis testing result.

α	0.1	0.2	0.3	0.4	0.5
p-value	0.00418	0.00665	0.01085	0.00802	0.02019

2.2 Improvised Trajectory Steering for Target Context

Building on the context-trajectory correlation, we propose Bifrost, a training-free, context-guided strategy that empowers self-improving agents to leverage prior successful trajectories to handle substantial context shifts. Since trajectories exhibit behavioral adaptation in response to context shifts toward the target task (Sec. 2.1), we use them to steer the existing ICL mechanism.

Context-guided adaptation under ICL. Our goal is to solve a target query $\hat{q} \in \hat{\mathcal{C}}$ by leveraging successful examples solved by an LLM agent $\mathcal{M}$ drawn from a different context $\mathcal{C} = \{(q_i, a_i)\}_{i=1}^{k}$. To this end, we construct a prompt $p = \mathcal{C} \circ \hat{q}$ comprising the demonstration set $\mathcal{C}$ followed by the target query $\hat{q}$, and steer the model toward the target context to produce the answer $\hat{a} = \mathcal{M}_{\mathrm{s}}(p)$. The steered model $\mathcal{M}_{\mathrm{s}}$ is obtained via contextual representation steering.

Contextual representation steering. Let $\mathbf{h}^{\ell}$ be the last token hidden state of the original model $\mathcal{M}$, the trajectory steering can be applied across specific layers $\ell \in L$, yielding the steered model $\mathcal{M}_{\mathrm{s}}$ with corresponding hidden state $\mathbf{h}_{\mathrm{s}}^{\ell} = \mathbf{h}^{\ell} + \alpha\Delta^{\ell}$, where $\alpha \in \mathbb{R}$ controls the steering strength. The vector Δ^{ℓ} that represents the contextual direction from the previous trajectory to the target task,

$$\Delta^{\ell} = \mathbf{h}^{\ell}(\hat{q}) - \frac{1}{k}\sum_{i=1}^{k} \mathbf{h}^{\ell}(\mathcal{C}_i), \tag{7}$$

where $\mathbf{h}^{\ell}(\hat{q})$ is the target question hidden state extracted by $\mathcal{M}(\hat{q})$, and $\mathbf{h}^{\ell}(\mathcal{C}_i)$ is a trajectory hidden state extracted by letting the agent solve the corresponding task $a_i = \mathcal{M}(q_i)$.

The contextual direction Δ^{ℓ} captures a context-shift signal, defined as the difference between prior trajectory executions and the current task. Under the Linear Representation Hypothesis (Park et al., 2023), these concepts are assumed to be orthogonal, thus averaging hidden states from prior trajectories suppresses private noise while preserving transferable patterns. This steering mechanism then promotes patterns that adapt to the target task, making Bifrost robust to context shifts and improving agent generalization. We will demonstrate that Bifrost remains effective even under suboptimal trajectories (Sec. 4.3), and its sensitivity under different trajectory qualities (App. F.2).

Under significant cross-domain shifts, the hidden states still encode high-level reasoning primitives such as modular arithmetic, logical chaining, etc., thus making them transferable under linear steering induced by Bifrost. This ability has been widely observed in existing works (Park et al., 2023; Bello et al., 2025; Fear et al., 2025). Furthermore, the overall shift vector in Eq. (7) is a per-query, task-conditioned linear shift. This local approximation enables effective steering even when surface forms differ across domains. We empirically validate Bifrost under cross-domain shift in Sec. 4.3. Further discussion is deferred to App. F.

Algorithm 1 Bifrost

Input: $\mathcal{C} = \{(q_i, a_i)\}_{i=1}^{k}$, $\hat{\mathcal{C}} = \{\hat{q}_j\}_{j=1}^{n}$, model $\mathcal{M}$, strength α, layers L.
Precompute $\bar{\mathbf{h}}^{\ell} = \frac{1}{k}\sum_{i=1}^{k} \mathbf{h}^{\ell}(c_i)\ \forall \ell \in L$.

for each $\hat{q} \in \hat{\mathcal{C}}$ **do**
  $p \leftarrow \mathcal{C} \circ \hat{q}$; obtain unsteered $\mathbf{h}^{\ell}(p)$.
  Compute $\Delta^{\ell} = \mathbf{h}^{\ell}(\hat{q}) - \bar{\mathbf{h}}^{\ell}$ and steer: $\mathbf{h}_{\mathrm{s}}^{\ell} = \mathbf{h}^{\ell} + \alpha\Delta^{\ell}, \forall \ell \in L$.
  Generate $\hat{a} = \mathcal{M}_{\mathrm{s}}(p)$ using steered states.
end for
Output: $\{\hat{a}_j\}_{j=1}^{n}$

Implementation details. Algorithm 1 describes our proposed method in detail. In practice, we precompute the average hidden states of prior trajectories to ensure computational efficiency (cost analysis is presented in App. E.7), and apply Bifrost individually to each target task rather than computing a shared steering vector, guaranteeing precise adaptation of prior trajectories to each task. Following prior work (Venhoff et al., 2025; Zhao et al., 2025; Turner et al., 2023), we steer the middle-to-late layers, as they best capture reasoning patterns amenable to steering. Beyond simple subtraction, Bifrost is also compatible with other projection-based techniques, e.g., PCA, Sparse Autoencoder (see Sec. 4.3).

3 Theoretical Analysis

We analyze the proposed method from a theoretical perspective, showing that Bifrost effectively leverages past trajectory experiences to adapt toward a target task. Our analysis is based on ICL as Bayesian Inference (Xie et al., 2021), demonstrating that Bifrost preserves prior knowledge induced by previous trajectories while reducing posterior uncertainty toward the target tasks. Further, we provide a risk bound of Bifrost that characterizes the mechanism of trajectory adaptation. We begin by introducing ICL as Bayesian Inference. Proofs are presented in App. D.1 and D.2. App. D.3 further discusses multi-modality setting.

ICL as Bayesian inference. The LLM agent $\mathcal{M}$ generates a prediction for a given input x by implicitly inferring a latent concept ϕ from a context $\mathcal{C}^*$ represented by in-context examples, $\mathcal{M}(x) \approx \mathbb{E}_{\phi \sim p(\phi|\mathcal{C}^*)}\left[f_\phi(x)\right]$, where $p(\phi \mid C^*)$ is the posterior distribution of concepts given a context. The effectiveness of agents stems from the latent concept inferred from the in-context examples. Thus, we focus on modeling the latent concept given previous trajectories.

In our case, the context is represented by previous trajectories, $\mathcal{C}^* = \mathcal{C}$. By definition, given a loss function $\mathcal{L}$, the posterior probability of the concept given previous trajectories is defined as

$$p(\phi \mid \mathcal{C}) \propto p(\mathcal{C} \mid \phi)\, p(\phi) \propto \exp\left(-\frac{1}{k}\sum_{i=1}^{k} \mathcal{L}\left(f_\phi(q_i), a_i\right)\right) p(\phi).$$

Posterior inference steering effect. The latent concept distribution can now be approximated by a Gaussian distribution over the context represented by previous context trajectories. Assuming $\mathcal{L}$ is convex and L-Lipschitz in f_ϕ, and f_ϕ is locally smooth in ϕ, we approximate the posterior of the latent concept via Laplace approximation,

$$p(\phi \mid \mathcal{C}) \approx \mathcal{N}(\tilde{\phi} \mid \mu, \Sigma^{-1}), \tag{8}$$

where $\mathcal{N}(\phi|\mu, \Sigma^{-1})$ denotes the Gaussian density with mean μ and inverse covariance matrix Σ^{-1}, and $\Sigma = \frac{1}{\sigma^2}\mathbf{F}^\top\mathbf{F}$, and $\mathbf{F} : \mathcal{X} \to \mathbb{R}^d$ is the feature map.

When leveraging previous trajectories as in-context examples, the latent concept is additionally conditioned on the target query $\hat{q} \in \hat{\mathcal{C}}$. By further applying a trajectory steering $\Delta \in \Lambda$ from Bifrost, the concept distribution mean shifts toward an ideal concept of such a target query accordingly.

Lemma 1. *Let $p_{\mathrm{s}}(\phi \mid \mathcal{C}, \hat{q})$ be the latent concept posterior probability of the steered model, conditioned on previous context trajectories $\mathcal{C}$ and the target query $\hat{q} \in \hat{\mathcal{C}}$. The contextual shift Δ shifts the distribution mean toward an ideal target query concept distribution by $\alpha P(\Delta)$ where P is the projection, while preserving uncertainty reduction induced by previous context trajectories,*

$$p_{\mathrm{s}}(\phi \mid \mathcal{C}, \hat{q}) \approx \mathcal{N}\left(\tilde{\phi} + \alpha P(\Delta) \mid \mu, \Sigma^{-1}\right) \tag{9}$$

Excess risk bound. We further analyze the risk bound of Bifrost compared to an ordinary agent that solely solves the target task without providing in-context examples from previous context tasks. Given that the target task shares a common representation with previous trajectories that are transferable, the following theorem shows that the bound of the proposed method reduces linearly with the number of induced previous trajectories.

Theorem 2. *Let $\mathcal{M}_{\mathrm{s}}$ denote the steered agent by Bifrost, $v = \phi^* - \tilde{\phi}$ the concept gap, and θ the angle between $P(\Delta)$ and v. Under Asm. 1 and Lemma 1, the steered agent by Bifrost satisfies*

$$\mathbb{E}\left[\mathcal{L}(\mathcal{M}_{\mathrm{s}}(\mathcal{C} \circ \hat{q}))\right] \leq \mathcal{O}\big(\frac{1}{k}\big) + C\,\|v\|^2 \sin^2\theta. \tag{10}$$

Thm. 2 characterizes the mechanism of trajectory adaptation that is overlooked by prior works rather than deriving a universally optimal risk bound. In adaptive agentic systems, it is more crucial to

Table 2: Comparisons on LiveCodeBench adapted from HumanEval trajectories measured by pass rates (%).

	Llama-1B			Llama-3B			Llama-8B		
Method	pass@1	pass@3	pass@5	pass@1	pass@3	pass@5	pass@1	pass@3	pass@5
Base	1.07	2.11	2.84	8.68	12.20	13.98	9.86	12.36	15.75
ICL	1.84	3.63	4.66	10.23	14.00	15.45	12.80	14.74	18.22
ICV	1.25	2.05	2.61	2.34	4.31	5.45	2.84	6.00	7.50
Reflexion	0.11	0.11	0.11	1.25	1.59	1.70	6.36	8.07	8.52
BoT	0.27	0.69	1.02	5.70	8.93	9.89	10.41	16.39	18.86
DoT	0.00	0.00	0.00	2.73	2.95	2.95	1.59	1.82	2.05
RISE	0.66	1.72	2.50	4.39	9.02	12.05	7.23	14.26	17.95
Paprika	1.30	2.55	3.30	8.59	12.33	14.20	14.98	20.30	22.61
Bifrost	**4.05**	**7.74**	**9.55**	**13.50**	**18.93**	**21.48**	**17.84**	**20.98**	**26.36**

understand why and when trajectory adaptation succeeds than to certify the success of every trajectory, *e.g., solving a complex math problem requires to determine how much previous examples should be reused and which conceptual knowledge should be adapted*. Eq. (10) decomposes the adaptation error into how many prior examples are needed (the first term), and which conceptual knowledge is expected (the second term). Thus, it establishes an initial theoretical foundation for understanding and further studying trajectory adaptation.

4 Experiments

We evaluate Bifrost against state-of-the-art baselines, including trajectory calibration, finetuning, and reinforcement learning, across diverse benchmarks. Sec. 4.1 describes the experimental setup under large context shifts. We then report main results showing that Bifrost consistently outperforms existing baselines across task domains in Sec. 4.2. Sec. 4.3 provides additional analyses on cross-domain adaptation, robustness to suboptimal trajectories, compatibility with other contextual direction extraction methods and LLM architectures. Finally, Sec. 4.4 presents an ablation study. Further experimental details are deferred to App. E. All prompts of Bifrost are presented in App. G.

4.1 Experimental Setup

Datasets. We evaluate Bifrost across diverse domains, including mathematical reasoning, question answering, and code generation. To study cross-context self-improvement, an agent first solves tasks on a source dataset and records successful trajectories, which are then provided as in-context examples when solving tasks on a target dataset.

We consider source–target dataset pairs: (1) AQUA $\rightarrow$ GSM8K: AQUA (Ling et al., 2017) focuses on algebraic problem solving, while GSM8K (Cobbe et al., 2021) contains diverse grade-school math problems. This setting evaluates whether algebraic reasoning acquired from AQUA transfers to GSM8K. (2) ARC-Easy (ARC-E) $\rightarrow$ GPQA-Diamond (GPQA-D): ARC-Easy (Clark et al., 2018) consists of elementary science multiple-choice questions, whereas GPQA-Diamond (Rein et al., 2024) benchmarks graduate-level, multi-domain scientific reasoning. This pair tests whether foundational science reasoning generalizes to advanced queries. (3) HumanEval $\rightarrow$ LiveCodeBench: HumanEval (Chen et al., 2021) evaluates code completion, while LiveCodeBench (Jain et al., 2024) is a dynamic, contamination-resistant code generation benchmark. It assesses whether code completion skills transfer to challenging programming tasks.

Baseline methods and LLMs. We evaluate Bifrost with *Chain-of-Thought (CoT)* (Wei et al., 2022), *ICL* (Brown et al., 2020), *In-Context Vectors (ICV)* (Liu et al., 2023). We also compare Bifrost with trajectory calibration methods, including *Reflexion* (Shinn et al., 2023), *Buffer of Thoughts (BoT)* (Yang et al., 2024), *Diversity of Though (DoT)* (Lingam et al., 2025), finetuning and reinforcement learning methods, including *RISE* (Qu et al., 2024), *Paprika* (Tajwar et al., 2025). Note that Bifrost leverages representation-level information already encoded in LLMs, it does not introduce any additional knowledge. This ensures a fair comparison with standard ICL and is arguably less advantageous than parameter-updated baselines. We leverage models *Llama 1B, 3B, 8B* with corresponding steering layers 10^{th} (1B), 14^{th} (3B), and 20^{th} layer (8B). We set $\alpha = 1$ by conducting grid search on held-out sets (App. E.2).

4.2 Main Results

Quantitative analysis. Bifrost significantly outperforms all baselines across different domains, as shown in Tabs. 2, 3, and 4. In math problem-solving experiments (Tab. 3), our method clearly surpasses from 2% to 3% the best performing method. In multi-domain science question answering

Table 3: Comparisons on GSM8K with trajectories are adapted from AQUA (measured by solve rate %).

Method	Llama-1B	Llama-3B	Llama-8B
CoT	42.15	70.18	74.65
ICL	23.81	62.52	80.78
ICV	20.62	54.13	62.17
Reflexion	15.85	62.40	87.41
BoT	9.62	33.74	39.73
DoT	3.71	76.80	76.72
RISE (1 attempt)	14.48	36.92	48.29
RISE (5 attempts)	40.71	78.62	86.13
Paprika	21.08	40.26	55.27
Bifrost	**44.66**	**81.41**	**90.22**

Table 4: Comparisons on GPQA-D with trajectories are adapted from ARC-E (measured by solve rate %).

Method	Llama-1B	Llama-3B	Llama-8B
CoT	4.04	12.63	17.68
ICL	23.23	25.25	22.22
ICV	13.64	24.75	27.78
Reflexion	1.10	20.20	33.84
BoT	16.67	12.12	11.11
DoT	1.01	13.64	30.81
RISE (1 attempt)	6.06	4.04	3.54
RISE (5 attempts)	21.21	19.19	12.63
Paprika	18.69	20.20	12.12
Bifrost	**24.24**	**32.83**	**37.88**

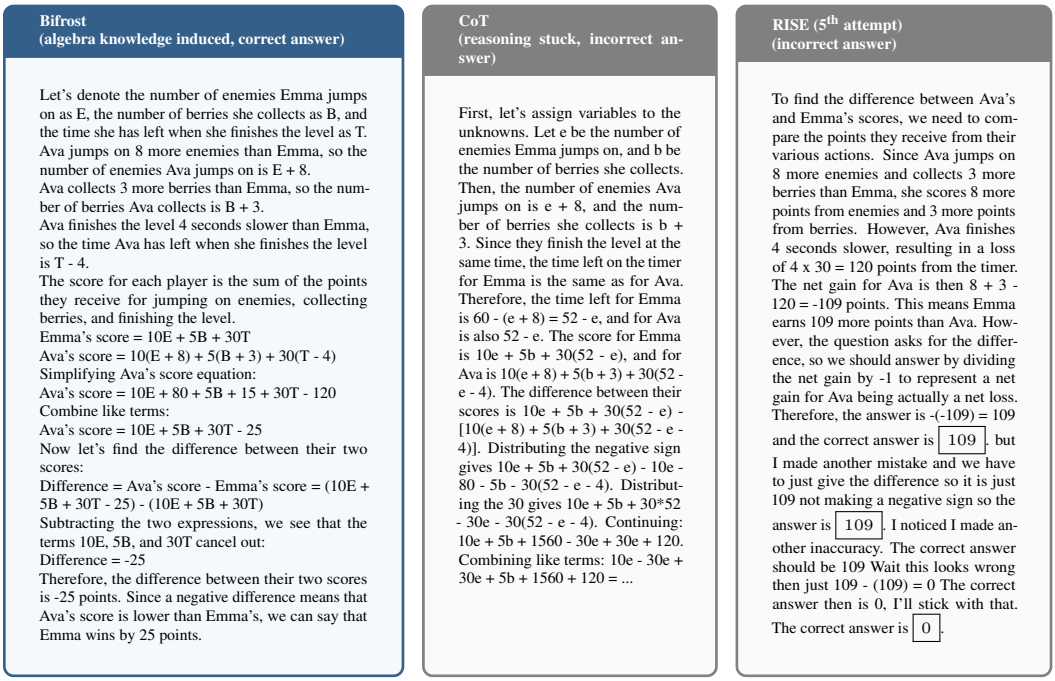
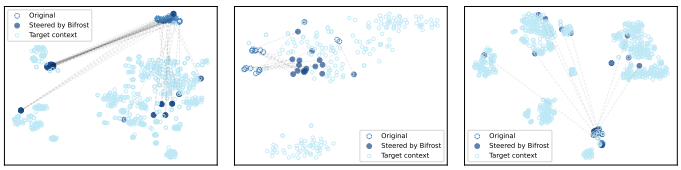


Figure 2: Success of latent concept transfer by Bifrost: it correctly solves a GSM8K task with algebra knowledge leveraged from AQUA trajectories, demonstrating problem decomposition and symbolic manipulation.

(Tab. 4), we clearly outperform the state-of-the-art methods from 4% to 16%. Similarly, in the code generation task (Tab. 2), we demonstrate a clear outperformance across all functional correctness metrics. Notably, Bifrost surpasses supervised finetuning and reinforcement learning methods without any training required, demonstrating that our method is flexible and efficient in leveraging their previous experiences across task contexts.

Qualitative analysis. An interesting result in Fig. 2 shows that Bifrost successfully generalizes algebra knowledge from AQUA to solve a GSM8K task. While CoT gets stuck in a reasoning loop and RISE produces an unsatisfiable answer, Bifrost effectively demonstrates problem decomposition ability and symbolic manipulation induced from AQUA. The result confirms the success of latent concept transformation from prior in-context examples to adapt toward the target task as theoretically analyzed in Sec. 3. More results are presented in App. E.4.

Steering effect illustration. We demonstrate how latent concepts shift across context in Fig. 3. Specifically, we plot the agent hidden states before and after steering by Bifrost, where dashed lines indicate the adaptation direction. We estimate the target context by extracting hidden states when solving the target tasks without any in-context examples. As shown in Fig. 3, the mismatch between previous trajectories and the target task can be observed, which hinders agent generalization when solely relying on conventional ICL or untargeted trajectory calibration methods. Our method addresses this limitation by adapting such trajectories toward the target context effectively. More results are presented in App. E.3.



(a) GSM8K (b) GPQA (c) LCB

Figure 3: Illustration of Latent Concept Shift by Bifrost: It steers prior context trajectories toward the target context of current tasks in the agent hidden states.

Table 5: Bifrost cross-domain adaptation effectiveness: adapted on LiveCodeBench from GSM8K trajectories.

Method	Llama-3.2-3B-Instruct		
	pass@1	pass@3	pass@5
ICL	8.75	11.93	13.64
BoT	4.67	7.41	8.52
RISE	4.81	8.85	10.74
Paprika	9.33	12.37	13.70
Bifrost	**12.73**	**16.99**	**18.75**

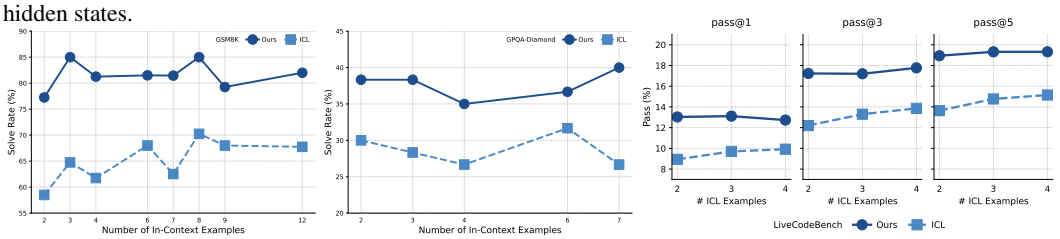


Figure 4: Bifrost effectiveness when leveraging different numbers of in-context examples.

4.3 Further Analysis

Cross-domain context adaptation. We further evaluate Bifrost on cross-domain adaptation by steering successful GSM8K trajectories toward LiveCodeBench, motivated by the hypothesis that latent concepts such as logical reasoning and symbolic manipulation are shared across domains. As confirmed in Tab. 5, Bifrost remains effective in this setting, outperforming all baseline methods. This suggests that trajectories preserve context-independent patterns transferable across domains.

Number of ICL example variants. Fig. 4 demonstrates Bifrost behavior when leveraging different numbers of in-context examples. When evaluated on GSM8K, leveraging more in-context examples increases the effectiveness of the agent in general while it causes a minor change in GPQA-Diamond. This can be explained as prior trajectories from ARC-Easy belong to multi-domain science questions that are more challenging to find common transferable patterns to adapt toward GPQA-Diamond. Therefore, the skeptical effectiveness can stem from the similarity of trajectory representations to extract and adapt useful patterns.

Steering layer position effect. To understand behavioral change under different layer intervention positions, we conduct trajectory steering across layers. As shown in Fig. 5, agent effectiveness changes differently across task domains. For math problem solving, adapting mid-to-late layers brings optimal results, while the opposite behavior occurs in multi-domain science question answering. This behavior is explained by prior work (Venhoff et al., 2025; Zhao et al., 2025; Turner et al., 2023) that steering early layers leads to more broad and foundational changes, while steering mid-to-late layers causes semantic concept changes. This well coincides with our experiments as multi-domain science requires foundational change, *i.e., different science topic*, at early layers, while problem solving requires concept adaptation, *i.e., leveraging arithmetic concepts*, that works best at mid-to-late layers. In code generation, steering mid-to-late layers also sightly increase the effectiveness.

Suboptimal trajectory adaptation. We reveal Bifrost ability to adapt suboptimal trajectories toward target tasks as shown in Fig. 6. Since contextual adaptation by Bifrost modifies trajectory behaviors by extracting transferable latent patterns, it is possible to leverage suboptimal trajectories, *e.g., the ones producing incorrect answers in previous tasks*, as long as such trajectories share common patterns with the target tasks. As shown in the figure, leveraging suboptimal trajectories surprisingly increases GSM8K solve rate; it could be explained as math-related patterns, such as arithmetic remains transferable even though incorrect in the previous tasks. In addition, GPQA-Diamond matches their original solve rate, while LiveCodeBench's effectiveness slightly degrades, possibly due to syntax errors in their defective trajectories, which could badly affect the adaptation.

Alternative contextual direction methods. We evaluate Bifrost via various techniques to extract contextual direction. In particular, we replace the existing per-task subtraction in Bifrost with *Shared Subtraction* that computes a single contextual direction given prior trajectories and all target tasks. In addition, we leverage *PCA* and *Sparse Autoencoder* that are widely adopted in representation engi-

Table 6: Bifrost with other context-based methods.

Methods	Solve Rate	Diff.
Original Bifrost	*81.45*	–
Shared Subtraction	77.25	-4.20
PCA	82.75	+1.30
Sparse Autoencoder	80.95	-0.50

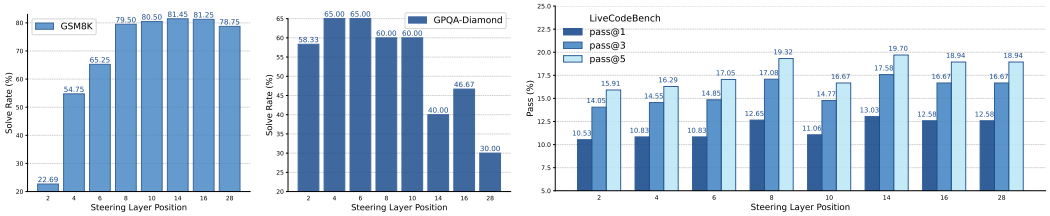


Figure 5: Bifrost with different steering layer positions.

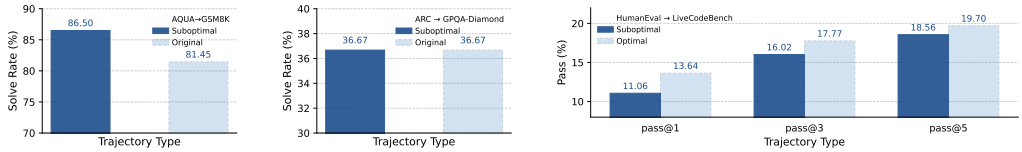


Figure 6: Bifrost when leveraging suboptimal prior trajectories.

neering (Zou et al., 2023; Højer et al., 2025). Tab. 6 shows the adaptation results in GSM8K. Shared subtraction degrades effectiveness by 4.2% as a single steering vector hardly captures task specifics, while PCA further enhances Bifrost by 1.3% thanks to better transferable pattern extraction under lower dimensional levels, and Sparse Autoencoder nearly matches the original Bifrost effectiveness.

Bifrost generalization to different LLM architectures. We further evaluate Bifrost under different LLM architectures. Tab. 7 demonstrates solve rates on a subset of AQUA → GSM8K, comparing between Bifrost with well-known baselines (ICL, CoT) and the best-performing baseline, RISE, using Qwen-3B , Phi-3.5B, and SmolLM-3B. Despite differences in pretraining data, tokenizer design, and architectural variations, Bifrost maintains stable improvements, indicating that the underlying context–trajectory correlation generalizes across models. This aligns with our theoretical framework, where steering is applied to latent concepts encoded in hidden states, which are known to exhibit similar geometric properties across modern LLMs.

Table 7: Bifrost by different LLM architectures on AQUA → GSM8K subset measured by solve rate (%).

Method	Llama-3B	Qwen-3B	Phi-3.5B	SmolLM3-3B
CoT	69.00	75.00	82.00	30.75
ICL	63.00	70.00	75.00	45.25
RISE	78.75	75.50	65.50	74.25
Bifrost	**82.75**	**83.00**	**88.00**	**83.00**

4.4 Ablation Study

We ablate different components of Bifrost in Tab. 8. *Opposite contextual direction* and *Random contextual direction* replace Bifrost steering direction with an opposite or random direction, respectively, assessing its importance. *In-domain ICL* uses trajectories from the target task instead of source-task trajectories, while *Query-conditioned steering* retains the target query but randomizes source trajectories, evaluating the contribution of source trajectories. We conduct experiments on a subset of AQUA → GSM8K. All ablations substantially degrade performance, with solve rates dropping by 5.50–17.50 %. In particular, the largest drop from *In-domain ICL* highlights the value of source trajectories, while the directional ablations demonstrate the importance of Bifrost contextual steering direction. Overall, these results confirm the complementary roles of the steering direction and source/target trajectories in effective adaptation.

Table 8: Ablation study on Bifrost.

Ablation Configuration	Solve Rate	Diff.
Original Bifrost	*85.00*	–
Opposite contextual direction	77.50	-7.50
Random contextual direction	78.00	-7.00
In-domain ICL	67.50	-17.50
Query-conditioned steering	79.50	-5.5

5 Conclusion

In this work, we proposed Bifrost, a training-free method that enables self-improving agents to improvise their prior trajectories across tasks under substantial context shifts. We revealed a context-trajectory correlation that allows Bifrost to precisely steer any previous trajectories toward target tasks by exploiting context differences. Both theoretical analyses and empirical results demonstrate that Bifrost leverages latent concepts acquired from prior in-context examples to adapt to new tasks, consistently outperforming existing self-improvement methods across diverse domains. Despite these promising results, our theoretical analysis is grounded in the Linear Representation Hypothesis, which may limit its applicability where more expressive nonlinear representations are required in practice. In the future, we plan to extend Bifrost to multi-context adaptation, where multiple target contexts are presented. Such a direction introduces an intensified complexity, which is essential for general-purpose agentic AI under multi-task applications.

AI USE STATEMENT

We used generative AI tools to assist with language editing. The authors developed and verified all scientific ideas, theoretical results, experimental design, analysis, and conclusions. The authors take full responsibility for the content of this paper.

ETHICS STATEMENT

The study of this work uses use publicly available models and datasets. It does not involve human subjects or private personal data.

REPRODUCIBILITY STATEMENT

We have provided detailed proofs of the theoretical results, as well as detailed experimental setups, model configurations, training protocols, and evaluation benchmarks in the experiments and appendix sections to facilitate reproduction of our study.

REFERENCES

Nikita Afonin, Nikita Andriyanov, Nikhil Bageshpura, Kyle Liu, Kevin Zhu, Sunishchal Dev, Ashwinee Panda, Alexander Panchenko, Oleg Rogov, Elena Tutubalina, et al. Emergent misalignment via in-context learning: Narrow in-context examples can produce broadly misaligned llms. *arXiv preprint arXiv:2510.11288*, 2025.

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. *Advances in Neural Information Processing Systems*, 36:45614–45650, 2023.

Kareem Amin, Sara Babakniya, Alex Bie, Weiwei Kong, Umar Syed, and Sergei Vassilvitskii. Escaping collapse: The strength of weak data for large language model training. *arXiv preprint arXiv:2502.08924*, 2025.

Shengnan An, Zeqi Lin, Qiang Fu, Bei Chen, Nanning Zheng, Jian-Guang Lou, and Dongmei Zhang. How do in-context examples affect compositional generalization? *arXiv preprint arXiv:2305.04835*, 2023.

Femi Bello, Anubrata Das, Fanzhi Zeng, Fangcong Yin, and Liu Leqi. Linear representation transferability hypothesis: Leveraging small models to steer large models. *arXiv preprint arXiv:2506.00653*, 2025.

Shelly Bensal, Umar Jamil, Christopher Bryant, Melisa Russak, Kiran Kamble, Dmytro Mozolevskyi, Muayad Ali, and Waseem AlShikh. Reflect, retry, reward: Self-improving llms via reinforcement learning. *arXiv preprint arXiv:2505.24726*, 2025.

Xiaohe Bo, Zeyu Zhang, Quanyu Dai, Xueyang Feng, Lei Wang, Rui Li, Xu Chen, and Ji-Rong Wen. Reflective multi-agent collaboration based on large language models. *Advances in Neural Information Processing Systems*, 37:138595–138631, 2024.

Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X Lee, Maria Bauzá, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil Raju, et al. Robocat: A self-improving generalist agent for robotic manipulation. *arXiv preprint arXiv:2306.11706*, 2023.

Joschka Braun, Carsten Eickhoff, David Krueger, Seyed Ali Bahrainian, and Dmitrii Krasheninnikov. Understanding (un) reliability of steering vectors in language models. *arXiv preprint arXiv:2505.22637*, 2025.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Hoyoon Byun, Gyeongdeok Seo, Joonseong Kang, Taero Kim, Jihee Kim, and Kyungwoo Song. Ccl: Causal-aware in-context learning for out-of-distribution generalization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *CoRR*, abs/2107.03374, 2021. URL `https://arxiv.org/abs/2107.03374`.

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.

Caia Costello, Simon Guo, Anna Goldie, and Azalia Mirhoseini. Think, prune, train, improve: Scaling reasoning without scaling models. *arXiv preprint arXiv:2504.18116*, 2025.

Resul Dagdanov, Halil Durmus, and Nazim Kemal Ure. Self-improving safety performance of reinforcement learning based driving with black-box verification algorithms. *arXiv preprint arXiv:2210.16575*, 2022.

Benoit Dherin, Michael Munn, Hanna Mazzawi, Michael Wunder, and Javier Gonzalvo. Learning without training: The implicit dynamics of in-context learning. *arXiv preprint arXiv:2507.16003*, 2025.

Rio Alexa Fear, Miles Cranmer, Payel Mukhopadhyay, and Michael McCabe. From vortices to spirals: Physics foundation models learn cross-domain concepts. In *Mechanistic Interpretability Workshop at NeurIPS 2025*, 2025.

Dayuan Fu, Keqing He, Yejie Wang, Wentao Hong, Zhuoma Gongque, Weihao Zeng, Wei Wang, Jingang Wang, Xunliang Cai, and Weiran Xu. Agentrefine: Enhancing agent generalization through refinement tuning. *arXiv preprint arXiv:2501.01702*, 2025.

Jingchu Gai, Guanning Zeng, Huaqing Zhang, and Aditi Raghunathan. Differential smoothing mitigates sharpening and improves llm reasoning. *arXiv preprint arXiv:2511.19942*, 2025.

Yubin Ge, Salvatore Romeo, Jason Cai, Monica Sunkara, and Yi Zhang. Samule: Self-learning agents enhanced by multi-level reflection. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 16602–16621, 2025.

Chase Goddard, Lindsay M Smith, Vudtiwat Ngampruetikorn, and David J Schwab. When can in-context learning generalize out of task distribution? *arXiv preprint arXiv:2506.05574*, 2025.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*, 2023.

Bertram Højer, Oliver Jarvis, and Stefan Heinrich. Improving reasoning performance in large language models via representation engineering. *arXiv preprint arXiv:2504.19483*, 2025.

Ziming Hong, Zhenyi Wang, Li Shen, Yu Yao, Zhuo Huang, Shiming Chen, Chuanwu Yang, Mingming Gong, and Tongliang Liu. Improving non-transferable representation learning by harnessing content and style. In *The Twelfth International Conference on Learning Representations*, 2024.

Senkang Hu, Xudong Han, Jinqi Jiang, Yihang Tao, Zihan Fang, Yong Dai, Sam Tak Wu Kwong, and Yuguang Fang. Distribution-aligned decoding for efficient llm task adaptation. *arXiv preprint arXiv:2509.15888*, 2025.

Zhuo Huang, Xiaobo Xia, Li Shen, Bo Han, Mingming Gong, Chen Gong, and Tongliang Liu. Harnessing out-of-distribution examples via augmenting content and style. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=boNyg20-JDm.

Zhuo Huang, Chang Liu, Yinpeng Dong, Hang Su, Shibao Zheng, and Tongliang Liu. Machine vision therapy: Multimodal large language models can enhance visual robustness via denoising in-context learning. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=LwOfVWgEzS.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Yiyang Jin, Kunzhao Xu, Hang Li, Xueting Han, Yanmin Zhou, Cheng Li, and Jing Bai. Reveal: Self-evolving code agents via iterative generation-verification. *arXiv preprint arXiv:2506.11442*, 2025.

Jannik Kossen, Yarin Gal, and Tom Rainforth. In-context learning learns label relationships but is not conventional learning. *arXiv preprint arXiv:2307.12375*, 2023.

Tianlong Li, Xiaoqing Zheng, and Xuanjing Huang. Open the pandora's box of llms: Jailbreaking llms through representation engineering. *CoRR*, 2024.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori B Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 12286–12312, 2023.

Jiaye Lin, Yifu Guo, Yuzhen Han, Sen Hu, Ziyi Ni, Licheng Wang, Mingguang Chen, Hongzhang Liu, Ronghao Chen, Yangfan He, et al. Se-agent: Self-evolution trajectory optimization in multi-step reasoning with llm-based agents. *arXiv preprint arXiv:2508.02085*, 2025.

Ziqian Lin and Kangwook Lee. Dual operating modes of in-context learning. In *Forty-first International Conference on Machine Learning*, 2024.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*, 2017.

Vijay Lingam, Behrooz Omidvar Tehrani, Sujay Sanghavi, Gaurav Gupta, Sayan Ghosh, Linbo Liu, Jun Huan, and Anoop Deoras. Enhancing language model agents using diversity of thoughts. In *The Thirteenth International Conference on Learning Representations*, 2025.

Jiayu Liu, Zhenya Huang, Chaokun Wang, Xunpeng Huang, Chengxiang Zhai, and Enhong Chen. What makes in-context learning effective for mathematical reasoning: A theoretical analysis. *arXiv preprint arXiv:2412.12157*, 2024.

Sheng Liu, Haotian Ye, Lei Xing, and James Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. *arXiv preprint arXiv:2311.06668*, 2023.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv:2310.12931*, 2023.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594, 2023.

Jianbiao Mei, Yukai Ma, Xuemeng Yang, Licheng Wen, Xinyu Cai, Xin Li, Daocheng Fu, Bo Zhang, Pinlong Cai, Min Dou, et al. Continuously learning, adapting, and improving: A dual-process approach to autonomous driving. *arXiv preprint arXiv:2405.15324*, 2024.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11048–11064, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.759. URL https://aclanthology.org/2022.emnlp-main.759/.

Siddharth Narayanan, James D Braza, Ryan-Rhys Griffiths, Manu Ponnapati, Albert Bou, Jon Laurent, Ori Kabeli, Geemi Wellawatte, Sam Cox, Samuel G Rodriques, et al. Aviary: training language agents on challenging scientific tasks. *arXiv preprint arXiv:2412.21154*, 2024.

Jing-Cheng Pang, Pengyuan Wang, Kaiyuan Li, Xiong-Hui Chen, Jiacheng Xu, Zongzhang Zhang, and Yang Yu. Language model self-improvement by reinforcement learning contemplation. *arXiv preprint arXiv:2305.14483*, 2023.

Jing-Cheng Pang, Si-Hang Yang, Kaiyuan Li, Jiaji Zhang, Xiong-Hui Chen, Nan Tang, and Yang Yu. Kalm: Knowledgeable agents by offline reinforcement learning from large language model rollouts. *Advances in Neural Information Processing Systems*, 37:126620–126652, 2024.

Madhur Panwar, Kabir Ahuja, and Navin Goyal. In-context learning through the bayesian prism. *arXiv preprint arXiv:2306.04891*, 2023.

Core Francisco Park, Andrew Lee, Ekdeep Singh Lubana, Yongyi Yang, Maya Okawa, Kento Nishi, Martin Wattenberg, and Hidenori Tanaka. Iclr: In-context learning of representations. *arXiv preprint arXiv:2501.00070*, 2024.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *arXiv preprint arXiv:2311.03658*, 2023.

Gokul Puthumanaillam, Paulo Padrao, Jose Fuentes, Pranay Thangeda, William E Schafer, Jae Hyuk Song, Karan Jagdale, Leonardo Bobadilla, and Melkior Ornik. Trace: A self-improving framework for robot behavior forecasting with vision-language models. *arXiv preprint arXiv:2503.00761*, 2025.

Jiahao Qiu, Xuan Qi, Tongcheng Zhang, Xinzhe Juan, Jiacheng Guo, Yifu Lu, Yimin Wang, Zixin Yao, Qihan Ren, Xun Jiang, et al. Alita: Generalist agent enabling scalable agentic reasoning with minimal predefinition and maximal self-evolution. *arXiv preprint arXiv:2505.20286*, 2025.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive introspection: Teaching language model agents how to self-improve. *Advances in Neural Information Processing Systems*, 37:55249–55285, 2024.

Allan Raventós, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression. *Advances in neural information processing systems*, 36:14228–14246, 2023.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

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

Gabriel Sarch, Lawrence Jang, Michael Tarr, William W Cohen, Kenneth Marino, and Katerina Fragkiadaki. Vlm agents generate their own memories: Distilling experience into embodied programs of thought. *Advances in Neural Information Processing Systems*, 37:75942–75985, 2024.

Vishnu Sarukkai, Zhiqiang Xie, and Kayvon Fatahalian. Self-generated in-context examples improve llm agents for sequential decision-making tasks. *arXiv preprint arXiv:2505.00234*, 2025.

Archit Sharma, Ahmed M Ahmed, Rehaan Ahmad, and Chelsea Finn. Self-improving robots: End-to-end autonomous visuomotor reinforcement learning. *arXiv preprint arXiv:2303.01488*, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652, 2023.

Alessandro Stolfo, Vidhisha Balachandran, Safoora Yousefi, Eric Horvitz, and Besmira Nushi. Improving instruction-following in language models through activation steering. *arXiv preprint arXiv:2410.12877*, 2024.

Jingran Su, Jingfan Chen, Hongxin Li, Yuntao Chen, Li Qing, and Zhaoxiang Zhang. Activation steering decoding: Mitigating hallucination in large vision-language models through bidirectional hidden state intervention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12964–12974, 2025.

Fahim Tajwar, Yiding Jiang, Abitha Thankaraj, Sumaita Sadia Rahman, J Zico Kolter, Jeff Schneider, and Ruslan Salakhutdinov. Training a generally curious agent. *arXiv preprint arXiv:2502.17543*, 2025.

Huaze Tang, Tianren Peng, and Shao-lun Huang. On theoretical interpretations of concept-based in-context learning. *arXiv preprint arXiv:2509.20882*, 2025.

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

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. Understanding reasoning in thinking language models via steering vectors. *arXiv preprint arXiv:2506.18167*, 2025.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pp. 35151–35174. PMLR, 2023.

Tomoya Wakayama and Taiji Suzuki. In-context learning is provably bayesian inference: a generalization theory for meta-learning. *arXiv preprint arXiv:2510.10981*, 2025.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Haoyu Wang, Zhuo Huang, Zhiwei Lin, and Tongliang Liu. Noisegpt: Label noise detection and rectification through probability curvature. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024a.

Qixun Wang, Yifei Wang, Yisen Wang, and Xianghua Ying. Can in-context learning really generalize to out-of-distribution tasks? *arXiv preprint arXiv:2410.09695*, 2024b.

Weixuan Wang, Jingyuan Yang, and Wei Peng. Semantics-adaptive activation intervention for llms via dynamic steering vectors. *arXiv preprint arXiv:2410.12299*, 2024c.

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

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 2550–2575, 2023.

Noam Wies, Yoav Levine, and Amnon Shashua. The learnability of in-context learning. *Advances in Neural Information Processing Systems*, 36:36637–36651, 2023.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*, 2021.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E Gonzalez, and Bin Cui. Buffer of thoughts: Thought-augmented reasoning with large language models. *Advances in Neural Information Processing Systems*, 37:113519–113544, 2024.

Xiao Yu, Baolin Peng, Vineeth Vajipey, Hao Cheng, Michel Galley, Jianfeng Gao, and Zhou Yu. Exact: Teaching ai agents to explore with reflective-mcts and exploratory learning. *arXiv preprint arXiv:2410.02052*, 2024.

Yurun Yuan and Tengyang Xie. Reinforce llm reasoning through multi-agent reflection. *arXiv preprint arXiv:2506.08379*, 2025.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin godel machine: Open-ended evolution of self-improving agents. *arXiv preprint arXiv:2505.22954*, 2025a.

Jesse Zhang, Jiahui Zhang, Karl Pertsch, Ziyi Liu, Xiang Ren, Minsuk Chang, Shao-Hua Sun, and Joseph J Lim. Bootstrap your own skills: Learning to solve new tasks with large language model guidance. *arXiv preprint arXiv:2310.10021*, 2023.

Tianle Zhang, Wanlong Fang, Jonathan Woo, Paridhi Latawa, Deepak A Subramanian, and Alvin Chan. Can llms reason over non-text modalities in a training-free manner? a case study with in-context representation learning. *arXiv preprint arXiv:2509.17552*, 2025b.

Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang, Yongliang Shen, Guiyang Hou, Zeqi Tan, Peng Li, Yueting Zhuang, and Weiming Lu. Agent-pro: Learning to evolve via policy-level reflection and optimization. *arXiv preprint arXiv:2402.17574*, 2024.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

Yu Zhao, Alessio Devoto, Giwon Hong, Xiaotang Du, Aryo Pradipta Gema, Hongru Wang, Xuanli He, Kam-Fai Wong, and Pasquale Minervini. Steering knowledge selection behaviours in llms via sae-based representation engineering. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5117–5136, 2025.

Boyuan Zheng, Michael Y Fatemi, Xiaolong Jin, Zora Zhiruo Wang, Apurva Gandhi, Yueqi Song, Yu Gu, Jayanth Srinivasa, Gaowen Liu, Graham Neubig, et al. Skillweaver: Web agents can self-improve by discovering and honing skills. *arXiv preprint arXiv:2504.07079*, 2025.

Yifei Zhou, Sergey Levine, Jason Weston, Xian Li, and Sainbayar Sukhbaatar. Self-challenging language model agents. *arXiv preprint arXiv:2506.01716*, 2025.

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*, 2023.

Adam Zweiger, Jyothish Pari, Han Guo, Ekin Akyürek, Yoon Kim, and Pulkit Agrawal. Self-adapting language models. *arXiv preprint arXiv:2506.10943*, 2025.

Appendix for "Bifrost: Steering Strategic Trajectories to Bridge Contextual Gaps for Self-Improving Agents"

Table of Contents

A Related Work **16**

B Motivation and Impact of Context Mismatch in Self-Improving Agents **17**

C Context-Trajectory Correlation **18**
C.1 Hypothesis Testing and Experimental Demonstration Details 18

D Theoretical Analysis **19**
D.1 Proof of Lemma 1 19
D.2 Proof of Thm 2 20
D.3 Extension of Lemma 1 to Multi-Modal Concept Posteriors 21

E Experiments **22**
E.1 Experimental Setup 22
E.2 Choice of Steering Parameter α 24
E.3 Latent Concept Shift Illustration 25
E.4 Qualitative Analysis 25
E.5 In-Context Examples Number Variants 30
E.6 Steering Player Positions 30
E.7 Efficiency of Bifrost 31

F Additional Discussion on Linearization and Trajectory Robustness **31**
F.1 On the Role of Linearization in Cross-Domain Transfer 31
F.2 On Trajectory Quality and Robustness 32

G Bifrost Prompts **33**

In this Appendix, we first present related work in App. A. The motivation and impact of our research problem are presented in App. B. We then further elaborate the hypothesis testing on the context-trajectory correlation in Sec. C. Our proofs for the theoretical analysis are presented in App. D. We discuss further details of the experimental setup and demonstrate additional illustrations and experimental results in App. E. We provide further discussion on linearization and robustness analysis of Bifrost in App. F. Finally, App. G reveals the prompts used for Bifrost in our work.

A Related Work

Self-improving agents. Thanks to self-reflection ability (Weng et al., 2023; Madaan et al., 2023; Gou et al., 2023; Pang et al., 2023; Bo et al., 2024; Zhang et al., 2024), self-improving agents have emerged in leveraging their own experiences to solve complicated decision-making and reasoning problems through iterative prompt-based trial-and-error (Wang et al., 2023; Shinn et al., 2023; Zhao et al., 2024; Lingam et al., 2025; Lin et al., 2025; Tajwar et al., 2025; Yu et al., 2024), supervised finetuning and reinforcement learning via data self-generation (Zelikman et al., 2022; Zhang et al., 2023; Zhou et al., 2025; Pang et al., 2024; Bensal et al., 2025; Jin et al., 2025; Sarch et al., 2024; Sarukkai et al., 2025). However, such approaches self-improve within target tasks (Shinn et al.,

2023; Yang et al., 2024; Lin et al., 2025; Zhao et al., 2024; Yuan & Xie, 2025), leaving cross-domain experience reuse strategies underexplored. While early work usually discards such insightful experiences to rely on supervised finetuning (Rebuffi et al., 2017; Zweiger et al., 2025; Narayanan et al., 2024; Pang et al., 2024; Costello et al., 2025; Amin et al., 2025), recent work attempts to address this by naively reusing prior trajectories (Shinn et al., 2023; Lingam et al., 2025; Zhao et al., 2024; Fu et al., 2025; Madaan et al., 2023), generalizing thought templates (Yang et al., 2024) as guidance, or relying on costly reinforcement learning to let agent unfold strategic and adaptable information (Bensal et al., 2025; Qu et al., 2024; Tajwar et al., 2025; Fu et al., 2025; Lin et al., 2025; Ge et al., 2025). Unfortunately, they remain suboptimal when faced with contextual non-stationarity across tasks (An et al., 2023; Lin & Lee, 2024; Gai et al., 2025; Afonin et al., 2025). Bifrost addresses this problem by contextual adaptation of prior trajectories without any training required.

Representation engineering. Previous work has demonstrated that representation manipulation under agent hidden states can change its behavior accordingly without any finetuning effort (Park et al., 2023; Turner et al., 2023; Zou et al., 2023; Hu et al., 2025; Li et al., 2024). While most existing literature focuses on explicit control over specific emotion behaviors (Stolfo et al., 2024; Turner et al., 2023), it is more challenging to leverage representation engineering to enhance agent reasoning and problem-solving abilities, as such features are usually implicit (Li et al., 2023; Liu et al., 2023; Wang et al., 2024c; Braun et al., 2025; Su et al., 2025). Although recent work has revealed this possibility (Højer et al., 2025), a deterministic mechanism remains unexplored. In this work, we successfully exploit the representation engineering as a proxy of contextual adaptation to enable agents to leverage their past experiences for target tasks effectively.

In-context learning. Many empirical and theoretical results have shown that in-context examples can enhance LLM Agent effectiveness (Brown et al., 2020; Dherin et al., 2025; Von Oswald et al., 2023; Ahn et al., 2023; Liu et al., 2024; Tang et al., 2025; Min et al., 2022; Kossen et al., 2023; Park et al., 2024; Wies et al., 2023; Zhang et al., 2025b). In addition, recent work theoretically proves ICL from the perspective of Bayesian Inference (Xie et al., 2021; Wakayama & Suzuki, 2025; Huang et al., 2024; Wang et al., 2024a) where the agent approximates a latent concept from such examples to solve the target when such tasks are context-related (Panwar et al., 2023; Raventós et al., 2023). However, ICL becomes ineffective when examples are specific to their previous tasks (An et al., 2023). This phenomenon is even worse for self-improving agents reusing their past experiences as in-context examples due to significant context mismatch between previous and target tasks (Lin & Lee, 2024), thus hindering generalization and further degrading agent behaviors (Fu et al., 2025; Wang et al., 2024b; Goddard et al., 2025; Byun et al.). Bifrost preserves the simplicity of ICL while further strengthening it by adapting the latent concept of such examples toward the target task.

B Motivation and Impact of Context Mismatch in Self-Improving Agents

Self-improving agents have made remarkable progress through reflection, iterative refinement, and experience reuse mechanisms. However, a critical yet underexplored challenge remains: how to effectively leverage successful trajectories from previous tasks when substantial context shifts occur in new target tasks. Most existing self-improvement frameworks focus on enhancing performance within a single task or environment through repeated trial-and-error, fine-tuning, or reinforcement learning. They largely treat past trajectories as context-specific artifacts and either discard them or apply simplistic heuristics when moving to new domains. This leaves a significant gap in cross-context generalization.

Empirical evidence of degradation caused by context mismatch. Our experiments reveal that naive trajectory reuse frequently harms performance compared to solving the target task from scratch. Tab. 9 summarizes this degradation across three representative cross-context settings using Llama-3.2-1B-Instruct. These results demonstrate that unadapted reuse of prior trajectories can cause severe performance drops (up to 20.85% points in the math reasoning setting) and, in certain cases, perform worse than solving the target task without any historical experience. This highlights a realistic and impactful limitation in current self-improving agent frameworks.

Table 9: Performance comparison on target tasks using Llama-3.2-1B-Instruct.

Setting	No Reuse (CoT)	Naive Reuse (ICL)	Bifrost (Ours)	Degradation from Naive Reuse
AQUA → GSM8K	42.15%	23.81%	**44.66%**	-20.85%
ARC-Easy → GPQA-Diamond	4.04%	23.23%	**44.44%**	-21.21%
HumanEval → LiveCodeBench	1.07%	1.84%	**4.05%**	-2.98%

Preliminary empirical support. In Sec. 2.1, we provide direct statistical evidence for the context-trajectory correlation. Using 100 paired samples from AQUA and GSM8K, we show that shifts in context are strongly correlated with shifts in successful trajectories (p-values < 0.03 across all tested steering strengths α, see Tab. 1 in the main paper). This supports our core hypothesis that trajectories must be actively adapted rather than naively reused when context changes.

Theoretical motivation. Under the perspective of ICL as approximate Bayesian inference (Xie et al., 2021), an agent observing successful trajectories from a source context $\mathcal{C}$ approximates a posterior $p(\phi \mid C)$ over latent reasoning concepts ϕ. When the target context $\hat{\mathcal{C}}$ differs substantially, this posterior becomes misaligned with the target task distribution. The resulting concept mismatch leads to suboptimal or even harmful reasoning on the new task. Our method mitigates this by explicitly steering prior trajectory representations toward the target context, shifting the posterior mean while preserving the uncertainty reduction gained from past experiences (see Lemma 1 and Thm 2).

The combination of observed performance degradation, statistical correlation between context and trajectory shifts, and theoretical misalignment demonstrates that context mismatch is not a minor implementation detail but a fundamental bottleneck preventing self-improving agents from fully leveraging their accumulated experience across tasks. We directly address this gap through a simple, training-free steering mechanism at the representation level. We believe resolving this challenge is essential for advancing toward general-purpose, lifelong self-improving agents.

C Context-Trajectory Correlation

C.1 Hypothesis Testing and Experimental Demonstration Details

Hypothesis testing. Our research question is whether previous-to-target trajectory steering preserves a linear correlation with trajectory change, such that a steered trajectory is more similar to the natural trajectory associated with the target query than the trajectory generated by a non-steered model for the original query. Let $(\hat{q}, \hat{a})$ be a target context pair consisting of a query and its corresponding natural trajectory, and let (q, a) be a previous context counterpart. The contextual shift Δ, computed using Eq. (3), is applied to the model to produce a steered trajectory a_{s}. We evaluate the relationship between context similarity and trajectory change by computing the Pearson correlation

$$r = corr(s_q, s_a), \tag{11}$$

where s_q denotes the context similarity obtained as

$$s_q = \langle f(q), f(\hat{q}) \rangle,$$

and s_a represents the similarity of trajectory change,

$$s_a = \langle f(a_{\mathrm{s}}), f(\hat{a}) \rangle - \langle f(a), f(\hat{a}) \rangle.$$

Here, $f(\cdot)$ denotes an embedding extractor.

As a result, we formulate the following hypotheses,

- H_0: There is no linear correlation between context shift and trajectory behavior shift.
- H_1: There is a confirmed linear correlation between context shift and trajectory behavior shift.

Experiments. To test the above hypothesis, we randomly select 100 samples from AQUA and GSM8K as two separate contexts. We then construct 50 sample pairs and apply trajectory steering from AQUA to GSM8K on each pair. For embedding extraction, we employ either their existing hidden states or the sentence transformer (all-MiniLM-L6-v2 in our work[1]). We first generate original

[1] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

trajectories for each context (using Llama-3.1-8B-Instruct), then apply steering vectors using Eq. (3). Finally, we measure p-values from Pearson correlation as shown in Eq. (11).

Tab. 10 and 11 show the hypothesis testing results when using hidden state and trajectory embeddings, respectively. The result demonstrates strong statistical significance across different settings of α, showing that our hypothesis of a linear correlation between context shift and trajectory behavior shift exists.

Table 10: Hypothesis testing results using hidden state embeddings

α	0.1	0.2	0.3	0.4	0.5	0.6	5.0	5.5
p-value	0.00418	0.00665	0.01085	0.00802	0.02019	0.02202	0.04279	0.01746

Table 11: Hypothesis testing results using trajectory embeddings

α	4.7	4.9	5.0	5.1	5.3
p-value	0.035	0.012	0.017	0.012	0.007

D Theoretical Analysis

D.1 Proof of Lemma 1

After observing previous trajectories, we have approximated the latent concept posterior as a Gaussian

$$p(\phi \mid \mathcal{C}) \approx \mathcal{N}(\tilde{\phi}, \Sigma^{-1}),$$
$$\text{where} \quad \tilde{\phi} = \arg\max_{\phi}(\log p(\mathcal{C} \mid \phi) + \log p(\phi)), \quad \Sigma = \frac{1}{\sigma^2}\mathbf{F}^\top\mathbf{F},$$

where the mean $\tilde{\phi}$ is usually centered at the mode of previous trajectories. When observing a new target query $\hat{q} \in \hat{\mathcal{C}}$, we update the posterior and its mean accordingly,

$$p(\phi \mid \mathcal{C}, \hat{q}) \propto p(\hat{q} \mid \phi) \cdot p(\phi \mid \mathcal{C}).$$
$$\tilde{\phi}' = \arg\max_{\phi} (p(\hat{q} \mid \phi) + p(\phi \mid \mathcal{C})).$$

Let $\mathcal{L}_\phi = p(\hat{q} \mid \phi) + p(\phi \mid \mathcal{C})$, we update the mean of new posterior by optimizing $\mathcal{L}_\phi$. To solve this, we will need to derive $p(\hat{q} \mid \phi)$ and $p(\phi \mid \mathcal{C})$.

Deriving $\log p(\hat{q} \mid \phi)$ by Taylor expansion to the new log-likelihood term,

$$\log p(\hat{q} \mid \phi) \approx \log p(\hat{q} \mid \tilde{\phi}) + g^\top(\phi - \tilde{\phi}) \; + \; \frac{1}{2}(\phi - \tilde{\phi})^\top Q(\phi - \tilde{\phi}),$$
$$\text{where} \quad g = \nabla_\phi \log p(\hat{q} \mid \phi)\Big|_{\phi=\tilde{\phi}}, \quad Q = \nabla^2_\phi \log p(\hat{q} \mid \phi)\Big|_{\phi=\tilde{\phi}}.$$

Deriving $\log p(\phi \mid \mathcal{C})$ as

$$\log p(\phi \mid \mathcal{C}) \; \approx \; \text{const} - \frac{1}{2}(\phi - \mu)^\top \Sigma^{-1}(\phi - \mu).$$

As a result, $\mathcal{L}_\phi$ becomes

$$\mathcal{L}_\phi \approx -\frac{1}{2}(\phi - \tilde{\phi})^\top \Sigma^{-1}(\phi - \tilde{\phi}) + g^\top(\phi - \tilde{\phi}) \; + \; \frac{1}{2}(\phi - \tilde{\phi})^\top Q(\phi - \tilde{\phi}).$$

To optimize $\mathcal{L}_\phi$, we take derivative w.r.t. ϕ and set to zero

$$\nabla_\phi \mathcal{L}_\phi = -\Sigma(\phi - \tilde{\phi}) + g + Q(\phi - \tilde{\phi}) = 0.$$

Rearranging and simplifying the single-step update by discarding Q, we have

$$\phi - \tilde{\phi} \approx \Sigma^{-1} g = \Sigma^{-1} \nabla_\phi \log p(\hat{q} \mid \phi)\Big|_{\phi=\tilde{\phi}}. \tag{12}$$

Therefore, adding the target query $\hat{q} \in \hat{\mathcal{C}}$ will update the latent concept posterior as

$$p(\phi \mid \mathcal{C}, \hat{q}) \approx \mathcal{N}\left(\tilde{\phi} + \Sigma^{-1} \nabla_\phi \log p(\hat{q} \mid \phi), \Sigma^{-1}\right). \tag{13}$$

Under the Linear Representation Hypothesis, $\log p(\hat{q} \mid \phi)$ from Eq. (12) is generally a linear function of the hidden state expressed as

$$\log p(\hat{q} \mid \phi) \approx \mathbf{h}(\hat{q}; \phi)\mathbf{W},$$

where $\mathbf{h}$ is the hidden state of LLM $\mathcal{M}$ and $\mathbf{W}$ is a linear projection.

Treating $\mathbf{h}$ as a function of ϕ, derive $\log p(\hat{q} \mid \phi)$,

$$\nabla_\phi \log p(\hat{q} \mid \phi) = \frac{\partial \log p}{\partial \mathbf{h}} \cdot \frac{\partial \mathbf{h}}{\partial \phi}.$$

The partial derivative of $\mathbf{h}$ can be approximated by a first-order Taylor expansion, assuming that $\mathbf{h}$ is locally smooth in ϕ, and ϕ^* is the ideal concept of the target task context $\hat{\mathcal{C}}$, we have

$$\mathbf{h}(\phi^*) \approx \mathbf{h}(\phi) + \frac{\partial \mathbf{h}}{\partial \phi}(\phi^* - \phi).$$

Since $\Delta \approx \mathbf{h}(\phi^*) - \mathbf{h}(\phi)$, we have

$$\begin{aligned}
\Delta &\approx \frac{\partial \mathbf{h}}{\partial \phi}(\phi^* - \phi). && (14)\\
&\approx \mathbf{J} \cdot \Sigma^{-1} \nabla_\phi \log p(\hat{q} \mid \phi) \qquad \text{(from Eq. (12))} && (15)\\
&= \Sigma^{-1} \cdot \mathbf{J} \nabla_\phi \log p(\hat{q} \mid \phi). && (16)
\end{aligned}$$

Rearranging,

$$\Sigma^{-1} \nabla_\phi \log p(\hat{q} \mid \phi) = \mathbf{J}^\top \Delta = P(\Delta). \tag{17}$$

Therefore, applying a trajectory steering Δ shifts the mean toward an ideal concept of the target context, the steered concept posterior is expressed as

$$p_{\mathrm{s}}(\phi \mid \mathcal{C}, \hat{q}) \approx \mathcal{N}\left(\tilde{\phi} + \alpha P(\Delta), \Sigma^{-1}\right).$$

D.2 Proof of Thm 2

Proof. Let $y^* = f_{\phi^*}(\hat{q})$ denote the target label, $\hat{y}_s = M_s(C \circ \hat{q})$ the steered prediction, and $p_s(\phi \mid C, \hat{q}) \approx \mathcal{N}(\tilde{\phi} + \alpha P(\Delta), \Sigma^{-1})$ the steered posterior from Lemma 1. Write $v = \phi^* - \tilde{\phi}$ and let θ be the angle between $P(\Delta)$ and v.

Step 1 (bias–variance decomposition). Since $\mathcal{L}$ is smooth and convex, Jensen's inequality gives

$$\begin{aligned}
\mathbb{E}\left[\mathcal{L}(\hat{y}_s, y^*)\right] &= \mathbb{E}_{\phi \sim p_s}[\mathcal{L}(f_\phi(\hat{q}), y^*)] && (18)\\
&\le \mathcal{L}(\mathbb{E}_{\phi \sim p_s}[f_\phi(\hat{q})],\, y^*) + \mathrm{Var}[f_\phi(\hat{q})] && (19)\\
&\approx \mathcal{L}\big(f_{\mathbb{E}[\phi]}(\hat{q}),\, y^*\big) + \mathcal{O}(\mathrm{Var}[f_\phi(\hat{q})])\,. && (20)
\end{aligned}$$

Step 2 (variance). Since f_ϕ is locally G-Lipschitz in ϕ and p_s has covariance Σ^{-1} with $\Sigma = \frac{1}{\sigma^2} F^\top F$ built from k context trajectories,

$$\mathrm{Var}[f_\phi(\hat{q})] \le G^2 \,\mathrm{Tr}(\Sigma^{-1}) = \mathcal{O}(1/k). \tag{21}$$

Step 3 (bias). By smoothness of $\mathcal{L}$, with the steered posterior mean $\mu_s = \tilde{\phi} + \alpha P(\Delta)$,

$$\mathcal{L}\big(f_{\mathbb{E}[\phi]}(\hat{q}), y^*\big) \le C \left\|\mu_s - \phi^*\right\|^2 = C \left\|\alpha P(\Delta) - v\right\|^2. \tag{22}$$

Step 4 (optimal steering strength). Assume $P(\Delta) \neq 0$ and define $g(\alpha) = \|\alpha P(\Delta) - v\|^2 = \alpha^2 \|P(\Delta)\|^2 - 2\alpha\langle P(\Delta), v\rangle + \|v\|^2$, a quadratic in α minimized at $\alpha^* = \langle P(\Delta), v\rangle / \|P(\Delta)\|^2$, with

$$g(\alpha^*) = \|v\|^2 - \frac{\langle P(\Delta), v\rangle^2}{\|P(\Delta)\|^2} = \|v\|^2 \left(1 - \cos^2\theta\right) = \|v\|^2 \sin^2\theta. \tag{23}$$

Combining Steps 1–4,

$$\mathbb{E}[\mathcal{L}(M_s(C \circ \hat{q}))] \le \mathcal{O}(1/k) + C\|v\|^2 \sin^2\theta. \tag{24}$$

□

D.3 Extension of Lemma 1 to Multi-Modal Concept Posteriors

We show that Lemma 1 can be extended to multi-modality, which may occur in practice, e.g., an algebraic and a purely arithmetic strategy for the same problem. Consequently, Lemma 2 extends Lemma 1 as follows.

Setup. Let $\ell_{\mathcal{C}}(\phi) = \sum_{i=1}^{k} \log p(c_i \mid \phi) + \log p(\phi)$ be the log-posterior (up to a constant) after observing the k demonstrations. We assume:

- (A1) **Separated modes.** $\ell_{\mathcal{C}}$ has M well-separated local maxima $\tilde{\phi}_1, \ldots, \tilde{\phi}_M$ with positive-definite Hessians $H_1, \ldots, H_M$ (the negative Hessians of $\ell_{\mathcal{C}}$), which grow linearly in k.
- (A2) **LRH around every mode.** As in Lemma 1, the steering vector Δ corresponds to the concept shift $P(\Delta)$, and this holds around every mode.
- (A3) **Mode gap.** The best-supported mode wins by a margin that grows with k: $\ell_{\mathcal{C}}(\tilde{\phi}_1) - \ell_{\mathcal{C}}(\tilde{\phi}_m) \ge k\delta$ for all $m \ge 2$ and some $\delta > 0$.

Lemma 2 (Multi-Modal Steering)**.** *Under (A1) and (A2), the concept posterior is a Gaussian mixture,*

$$p(\phi \mid \mathcal{C}) \approx \sum_{m=1}^{M} w_m \, \mathcal{N}\big(\tilde{\phi}_m,\, H_m^{-1}\big),$$

and the steered posterior is

$$p_s(\phi \mid \mathcal{C}, \hat{q}) \approx \sum_{m=1}^{M} \tilde{w}_m \, \mathcal{N}\big(\tilde{\phi}_m + \alpha P(\Delta),\, H_m^{-1}\big), \qquad \tilde{w}_m \propto w_m \, p(\hat{q} \mid \tilde{\phi}_m).$$

That is:

- *(i) every mode moves by the same vector $\alpha P(\Delta)$;*
- *(ii) every mode keeps its covariance and shape, so the uncertainty reduction gained from the demonstrations is preserved;*
- *(iii) the target query only reweights the modes, favouring those that explain $\hat{q}$ well.*

If (A3) also holds, all weights except $\tilde{w}_1$ vanish as k grows, and the steered posterior reduces to Lemma 1 with $\tilde{\phi} = \tilde{\phi}_1$ and $\Sigma = H_1$.

Intuitively, Lemma 2 shows that Bifrost adds the same vector $\alpha\Delta$ to the hidden state, whichever concept the model is using. In concept space, this acts as a *translation*: it moves every mode by the same amount without changing its shape. The Gaussian assumption in Lemma 1 is therefore needed only to describe the posterior, not to explain what steering does to it.

Remark 1. (A3) is used only in the final statement. If two concepts are almost equally supported, the mixture does not collapse to one Gaussian, but (i)–(iii) still hold: Bifrost moves every candidate concept toward the target context without discarding any of them.

Remark 2. (A2) assumes that the steering vector corresponds to the same concept shift around every mode. If the LRH holds only locally, so that each mode m has its own projection P_m, the same proof goes through with $P(\Delta)$ replaced by $P_m(\Delta)$ in mode m. Each mode still keeps its own shape, but the distances between modes may change.

Proof. The argument has four steps.

Step 1 (one Gaussian per mode). Around each maximum $\tilde{\phi}_m$, a second-order Taylor expansion of $\ell_{\mathcal{C}}$ gives a Gaussian $\mathcal{N}(\tilde{\phi}_m, H_m^{-1})$, which is the same Laplace approximation used in Lemma 1, applied to each mode separately. Since the modes are well separated (A1), the posterior is the sum of these Gaussians. Each weight w_m is proportional to the posterior mass near its mode, $e^{\ell_{\mathcal{C}}(\tilde{\phi}_m)} \det(H_m)^{-1/2}$.

Step 2 (adding the target query). Conditioning on $\hat{q}$ multiplies the posterior by $p(\hat{q} \mid \phi)$. Because each Gaussian is narrow (its covariance H_m^{-1} shrinks as $1/k$), we can linearise the query log-likelihood around each mode, exactly as in Lemma 1:

$$p(\hat{q} \mid \phi) \approx p(\hat{q} \mid \tilde{\phi}_m) \, \exp\big(g_m^\top(\phi - \tilde{\phi}_m)\big), \qquad g_m = \nabla_\phi \log p(\hat{q} \mid \phi)\big|_{\tilde{\phi}_m}.$$

This has two effects on the m-th Gaussian. First, its weight is multiplied by $p(\hat{q} \mid \tilde{\phi}_m)$, giving $\tilde{w}_m$. Second, its mean moves by $H_m^{-1} g_m$ while its covariance stays H_m^{-1}, since tilting a Gaussian by an exponential of a linear function only shifts it. By Eq. (17) of Lemma 1, applied to each mode under (A2), this shift equals $P(\Delta)$.

Step 3 (steering is a translation). Bifrost scales this shift by α, which moves every point by $b = \alpha P(\Delta)$:

$$p_s(\phi) = p(\phi - b).$$

Moving a distribution rigidly changes its mean by b and leaves everything else unchanged. Deviations from the mean, $\phi - \mathbb{E}[\phi]$, are identical before and after the shift, so the covariance and all higher-order central moments (skewness, kurtosis, and so on) are preserved. This argument does not use the Gaussian form, so (i) and (ii) hold for any mode shape. Since every mode moves by the same b, the whole mixture moves rigidly as well.

Step 4 (the dominant mode takes over). Comparing the weight of any mode with that of the best mode,

$$\frac{\tilde{w}_m}{\tilde{w}_1} = \underbrace{e^{\ell_{\mathcal{C}}(\tilde{\phi}_m) - \ell_{\mathcal{C}}(\tilde{\phi}_1)}}_{\leq e^{-k\delta} \text{ by (A3)}} \cdot \underbrace{\Big(\frac{\det H_1}{\det H_m}\Big)^{1/2} \frac{p(\hat{q} \mid \tilde{\phi}_m)}{p(\hat{q} \mid \tilde{\phi}_1)}}_{\text{does not grow with } k}.$$

The first factor decays exponentially in k. The second depends on a single query and on ratios of Hessians that each grow linearly in k, so it stays bounded. Therefore $\tilde{w}_m \to 0$ for every $m \geq 2$, and the steered posterior becomes the single Gaussian $\mathcal{N}(\tilde{\phi}_1 + \alpha P(\Delta), H_1^{-1})$, which is Lemma 1. □

E Experiments

E.1 Experimental Setup

Context shift experimental setup. To evaluate trajectory leverage of self-improving agents under substantial context shifts, we setup source - target dataset pairs. In particular, a self-improving agent will solve tasks on a source dataset to collect successful trajectories (and/or unsuccessful ones depending on baselines), then leverage such trajectories to adapt toward the target task in the sense that there is no ground-truth available on such tasks of the target dataset. In this way, the self-improving agent has to employ their past experiences learned from previous tasks via such trajectories to solve the target tasks. All dataset pairs used in our experiments are contextually distinct. Tab. 12 ensures their semantic similarities, encoded by all-MiniLM-L6-v2, are sufficiently low. We now present how this setup applies to Bifrost and the baselines in our work.

Bifrost setup. Bifrost does not require any particular self-improving agent mechanism to solve tasks and collect corresponding trajectories. In our experiment, we use ordinary LLM agents as mentioned to solely solve source tasks and subsequently collect successful trajectories.

We let the agent solve a subset of 20 tasks when using Llama-3.1-8B, and 50 tasks when using Llama-3.2-3B and Llama-3.2-1B, for all source tasks in our experiments (AQUA, ARC-Easy, HumanEval). We subsequently collect corresponding successful trajectories and randomly pick a number of them as in-context examples to solve target tasks. In particular, we randomly pick 7 examples in *AQUA* $\rightarrow$ *GSM8K* and *ARC-Easy* $\rightarrow$ *GPQA-Diamond* experiments, and 5 examples in *HumanEval* $\rightarrow$ *GSM8K* experiments.

Choice of steering layers. We do not perform any layer search. Following prior work (Turner et al., 2023; Venhoff et al., 2025; Zhao et al., 2025), we fix the steering layer a priori to the exact middle layer of each model (10^{th} layer for Llama-1B, 14^{th} layer for Llama-3B, 20^{th} layer for Llama-8B), since mid-layer hidden states are known to best encode conceptual knowledge. Sec. E.6 further conducts layer sweep for a post-hoc sensitivity.

Baselines setup. We compare Bifrost with the following baselines.

- *Chain-of-Though (CoT)* (Wei et al., 2022) prompts an agent to perform step-by-step reasoning. We directly apply CoT to the target tasks.
- *In-Context Learning (ICL)* (Brown et al., 2020), we let an agent solve tasks from a source dataset to collect successful trajectories. We then leverage such trajectories as in-context examples in the prompts to solve target tasks. We use 7 in-context examples for both the math problem-solving task and the multiple-choice question answering task, and we use 5 in-context examples for the code generation task.
- *In-Context Vector (ICV)* (Liu et al., 2023) recasts in-context examples into in-context embeddings. We perform ICV on the source tasks and leverage in-context embeddings on the target tasks, i.e., 7 examples for the math problem-solving task and the multiple-choice question answering task, and 5 examples for the code generation task.
- *Reflexion* (Shinn et al., 2023) is able to iteratively refine its own trajectories given environment feedback. We directly apply Reflexion to the target tasks. We allow up to 5 attempts for math problem solving and code generation problems, as in their original work. For the multiple-choice question answering problem, we allow 3 attempts since more attempts can enable the agent to easily get the correct answer with 4-choice questions, which is unfair to other methods.
- *Buffer of Thoughts (BoT)* (Yang et al., 2024) creates thought templates when solving tasks as guidance in future tasks. We let BoT solve the source dataset tasks, subsequently creating corresponding though templates, which then serve as guidance in target tasks. Following their original work (Yang et al., 2024), we let BoT continuously update its thought templates while solving the target tasks.
- *RISE* (Qu et al., 2024) teaches an agent to self-improve by learning each fail-to-correct transition of trajectories via reinforcement learning. In our experiments, we let RISE solve source tasks and subsequently perform reinforcement learning on corresponding trajectories. We then leverage the finetuned agent to solve target tasks. We evaluate *RISE* within a single attempt and 5 attempts, respectively, as recommended by the original work (Qu et al., 2024).
- *Diversity of Thought (DoT)* (Lingam et al., 2025) lets an agent generate diverse trajectories and self-reflect on given environment feedback. Successful trajectories are stored in a memory bank as is and subsequently retrieved as in-context examples when solving similar tasks. We let DoT solve source tasks, and employ it to solve the target tasks.
- *Paprika* (Tajwar et al., 2025) learns strategic information by letting an agent self-explore and adjust its trajectories given environment feedback via a combined Supervised Finetuning and Direct Preference Optimization. Such information is prone to out-of-domain as shown in their original paper (Tajwar et al., 2025). Similar to RISE, we learn Paprika on source tasks and apply it to the target tasks.

Table 12: Context similarity and key overlap between benchmark pairs.

Dataset Pair	Context Similarity (%)
AQUA → GSM8K	3.65
ARC-Easy → GPQA-Diamond	8.98
HumanEval → LiveCodeBench	23.85
GSM8K → LiveCodeBench (cross-domain)	9.23

Evaluation metrics. For *AQUA → GSM8K* and *ARC-Easy → GPQA-Diamond* experiments, we allow a single generation for each question, except *Reflexion* and *RISE* baselines, and perform exact matching between agent answers and ground-truths, then measure the solve rate. For *HumanEval → GSM8K*, we generate 5 solutions for each problem and measure functional correctness following previous work (Chen et al., 2021; Jain et al., 2024) to determine the solutions passing test cases via pass@1, pass@3, and pass@5.

Datasets and LLMs. We provide the dataset and LLM details as follows. For datasets, we leverage AQUA (Algebra Question Answering with Rationales)[2], GSM8K[3], ARC-Easy[4], GPQA-Diamond[5], HumanEval[6], and LiveCodeBech[7] (release_v5) with 880 problems. For the LLMs, we leverage different model sizes, including Llama 1B[8], 3B[9], and 8B[10] Instruct versions.

E.2 Choice of Steering Parameter α

We first provide a theoretical result showing that the optimal steering parameter α has a closed form, and then describe how we choose the empirical α.

Proposition 1 (Invariant Steering Coefficient to Scale)**.** *Let h be the unsteered hidden state of a target query $\hat{q} \in \hat{\mathcal{C}}$ and $\delta(\hat{q})$ the ideal contextual shift, i.e., $h + \delta$ is the ideal steered state realizing the target concept. Bifrost applies the empirical steering $h_s = h + \alpha\Delta$ (Eq. (7)). For any query $\hat{q}$ with $\Delta(\hat{q}) \neq 0$, the bias term, which is a function of α,*

$$B_{\hat{q}}(\alpha) = \|\alpha\Delta(\hat{q}) - \delta(\hat{q})\|^2 \tag{25}$$

is a strictly convex quadratic in α, with unique minimizer

$$\alpha^*(\hat{q}) = \frac{\langle \Delta, \delta \rangle}{\|\Delta\|^2} = \frac{\cos\theta(\hat{q})}{\kappa(\hat{q})}, \qquad \kappa(\hat{q}) = \frac{\|\Delta(\hat{q})\|}{\|\delta(\hat{q})\|}. \tag{26}$$

Thus, the optimal coefficient does not depend on the absolute scale of Δ and δ separately.

Prop. 1 demonstrates that the optimal steering parameter has a closed form and is invariant to scale. Specifically, the empirical steering vector Δ in Eq. (7) already captures the query-specific context-gap magnitude; thus, changes in the absolute context-gap magnitude are absorbed by the corresponding change in $\|\Delta\|$, while α^* is determined by the relative scale κ and directional alignment $\cos\theta$. Consequently, this observation supports the use of a shared coefficient when the relative scale $\kappa(\hat{q})$ and alignment $\cos\theta(\hat{q})$ are approximately stable across queries.

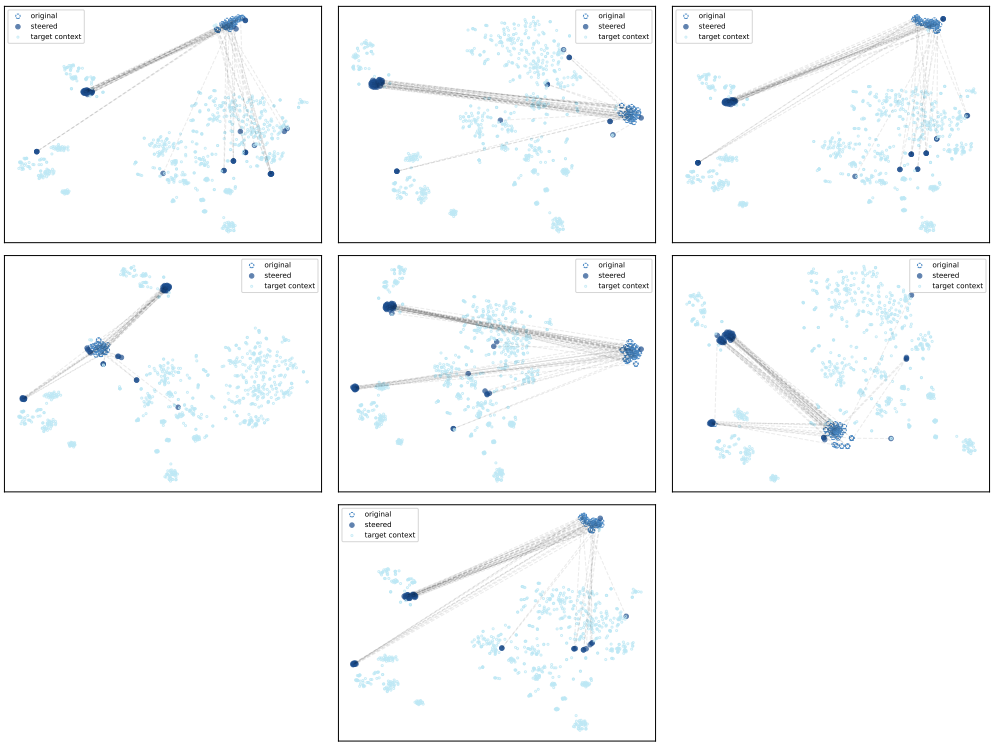
Empirical evaluation. To justify the invariance of the steering parameter, we compare the performance gap between a fixed α^* and its oracle counterpart, i.e., exhaustively run with $\alpha \in \{1, 2, 3, 4\}$ and select the one that performs best per query. Specifically, we evaluate on Llama-3B (5 ICL examples from AQUA steering 249 GSM8K instances). The gap in effectiveness between the fixed and the oracle per instance is only 1.61% in the solve rate, which validates Prop. 1.

[2] https://github.com/google-deepmind/AQuA
[3] https://huggingface.co/datasets/openai/gsm8k
[4] https://huggingface.co/datasets/allenai/ai2_arc
[5] https://huggingface.co/datasets/fingertap/GPQA-Diamond
[6] https://huggingface.co/datasets/openai/openai_humaneval
[7] https://huggingface.co/datasets/openai/openai_humaneval
[8] https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct
[9] https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct
[10] https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct

Experimental choices of optimal steering parameter. We choose optimal steering parameters across datasets by evaluating on held-out sets. Specifically, for each benchmark pair, we sample 7 source trajectories and 50 held-out target instances that are disjoint from every evaluation set, i.e., AQUA-train → GSM8K-train, ARC-Easy → GPQA-main, and HumanEval → LiveCodeBench-v1, which are separate from the datasets used in the main experiments. Consequently, we choose $\alpha = 1$ as a fixed steering parameter in our main experiments.

E.3 Latent Concept Shift Illustration

We show more steering effect illustrations in AQUA → GSM8K (Fig. 7), ARC-Easy → GQPA-Diamond (Fig. 8), and HumanEval → LiveCodeBench (Fig. 9).



(a)

Figure 7: Illustration of Latent Concept Shift from AQUA to GSM8K.

E.4 Qualitative Analysis

Fig. 10, 11, and 12 show additional results demonstrating that Bifrost can effectively leverage latent concepts learned from their prior context trajectories to solve the current tasks.

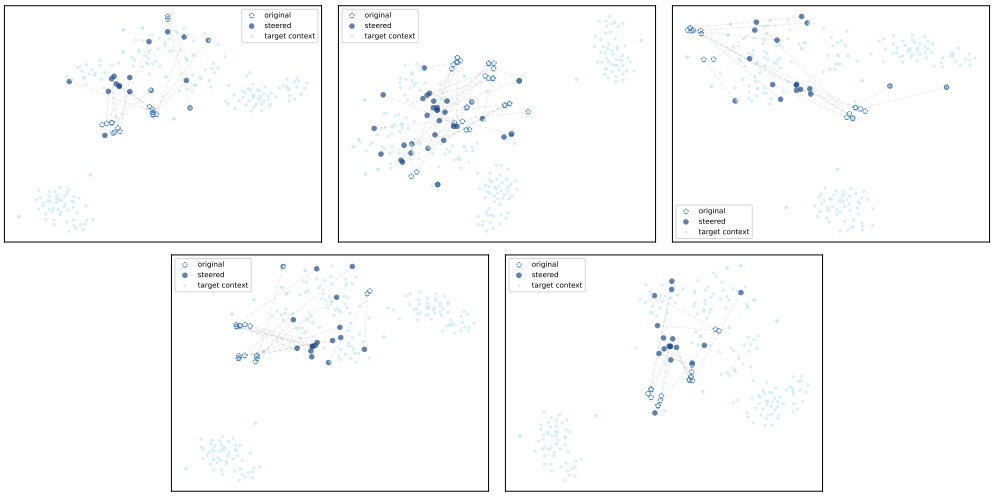


Figure 8: Illustration of Latent Concept Shift from ARC-Easy to GPQA-Diamond.

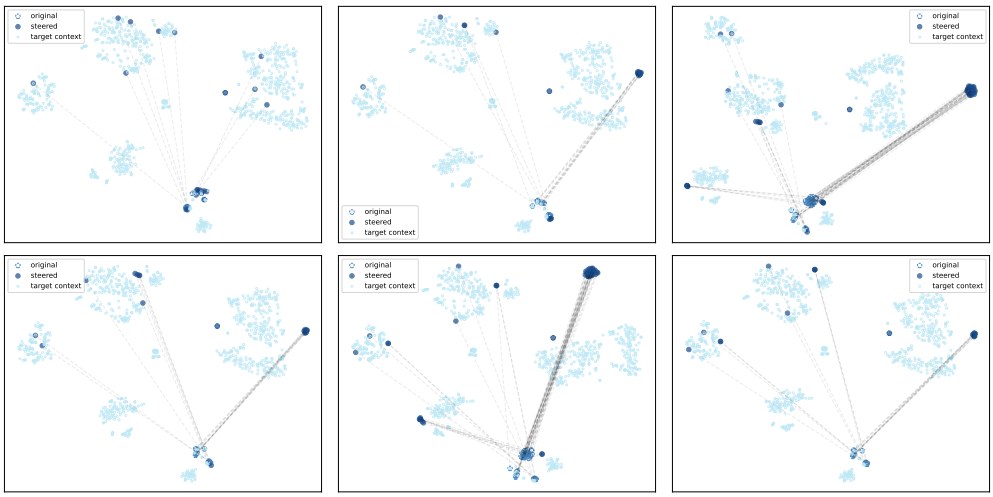


Figure 9: Illustration of Latent Concept Shift from HumanEval to LiveCodeBench.

Bifrost

Question:
Ava and Emma want to know who is better at the new video game Ava got for her birthday. They are each going to play one level and whoever has the highest score wins. They receive 10 points for every enemy they jump on, 5 points for each berry they collect, and 30 points for every second left on the timer when they finish the level. If Ava jumps on 8 more enemies than Emma and collects 3 more berries, but finishes the level 4 seconds slower, what is the difference between their two scores?

Answer:
Let's denote the number of enemies Emma jumps on as E, the number of berries she collects as B, and the time she has left when she finishes the level as T.
Ava jumps on 8 more enemies than Emma, so the number of enemies Ava jumps on is E + 8.
Ava collects 3 more berries than Emma, so the number of berries Ava collects is B + 3.
Ava finishes the level 4 seconds slower than Emma, so the time Ava has left when she finishes the level is T - 4.
The score for each player is the sum of the points they receive for jumping on enemies, collecting berries, and finishing the level.
Emma's score = 10E + 5B + 30T
Ava's score = 10(E + 8) + 5(B + 3) + 30(T - 4)
Simplifying Ava's score equation:
Ava's score = 10E + 80 + 5B + 15 + 30T - 120
Combine like terms:
Ava's score = 10E + 5B + 30T - 25
Now let's find the difference between their two scores:
Difference = Ava's score - Emma's score = (10E + 5B + 30T - 25) - (10E + 5B + 30T)
Subtracting the two expressions, we see that the terms 10E, 5B, and 30T cancel out:
Difference = -25
Therefore, the difference between their two scores is -25 points. Since a negative difference means that Ava's score is lower than Emma's, we can say that Emma wins by 25 points. [CORRECT]

CoT

Answer:
First, let's assign variables to the unknowns. Let e be the number of enemies Emma jumps on, and b be the number of berries she collects. Then, the number of enemies Ava jumps on is $e+8$, and the number of berries she collects is $b+3$. Since they finish the level at the same time, the time left on the timer for Emma is the same as for Ava. Therefore, the time left for Emma is $60-(e+8) = 52-e$, and for Ava is also $52-e$. The score for Emma is $10e+5b+30(52-e)$, and for Ava is $10(e+8)+5(b+3)+30(52-e-4)$. The difference between their scores is $10e+5b+30(52-e)-[10(e+8)+5(b+3)+30(52-e-4)]$. Distributing the negative sign gives $10e+5b+30(52-e)-10e-80-5b-30(52-e-4)$. Distributing the 30 gives $10e+5b+30\cdot52-30e-30(52-e-4)$. Continuing: $10e + 5b + 1560 - 30e + 30e + 120$. Combining like terms: $10e - 30e + 30e + 5b + 1560 + 120 = \ldots$
[REASONING STUCK]

RISE (5th attempt)

Answer:
To find the difference between Ava's and Emma's scores, we need to compare the points they receive from their various actions. Since Ava jumps on 8 more enemies and collects 3 more berries than Emma, she scores 8 more points from enemies and 3 more points from berries. However, Ava finishes 4 seconds slower, resulting in a loss of $4x30 = 120$ points from the timer. The net gain for Ava is then $8 + 3 - 120$ = -109 points. This means Emma earns 109 more points than Ava. However, the question asks for the difference, so we should answer by dividing the net gain by -1 to represent a net gain for Ava being actually a net loss. Therefore, the answer is $-(-109) = 109$ and the correct answer is $\boxed{109}$. but I made another mistake and we have to just give the difference so it is just 109 not making a negative sign so the answer is $\boxed{109}$. I noticed I made another inaccuracy. The correct answer should be 109 Wait this looks wrong then just $109 - (109) = 0$ The correct answer then is 0, I'll stick with that. The correct answer is $\boxed{0}$.
[INCORRECT]

Figure 10: Qualitative results from GSM8K dataset adapted from AQUA trajectories (1/2).

Bifrost

Question:
Tanya is teaching at school. She earns $15 for every hour and an additional $5 per day if she teaches more than 3 classes. On Monday she teaches 4 classes for 5 hours, and on Wednesday 2 classes for 2 hours. How much did Tanya earn for these two days of teaching?

Answer:
Let's break down Tanya's earnings into two parts: her hourly wage and her daily bonus.
On Monday, she teaches 4 classes for 5 hours, so her total earnings from her hourly wage are: 5 hours * $15/hour = $75. Since she teaches more than 3 classes, she also gets a daily bonus of $5.
On Wednesday, she teaches 2 classes for 2 hours, so her total earnings from her hourly wage are: 2 hours * $15/hour = $30. She doesn't get a daily bonus on Wednesday because she teaches less than 4 classes.
Now, let's add up her earnings from both days: Monday: $75 + $5 = $80. Wednesday: $30 + $0 = $30. Total earnings: $80 + $30 = $110.
Prediction: 110.0 [CORRECT]

CoT

Answer:
First, we need to find out how much Tanya earned for teaching on Monday. She taught 4 classes for 5 hours, so she earned 4 * 5 = 20 hours of pay. Since she earns $15 per hour, she earned 20 * 15 = $300 for teaching on Monday. Next, we need to find out how much Tanya earned for teaching on Wednesday. She taught 2 classes for 2 hours, so she earned 2 * 2 = 4 hours of pay. Since she earns $15 per hour, she earned 4 * 15 = $60 for teaching on Wednesday. Now, we need to find out how much Tanya earned for teaching on both days. We add the amount she earned on Monday to the amount she earned on Wednesday. 300 + 60 = 360.
The answer is 360.
Prediction: 360.0 [INCORRECT]

ICL

Answer:
Hourly Wage: Tanya earns $15 per hour. She worked for a total of 5 + 2 = 7 hours. So, her total hourly earnings are 7 x 15 =105.
Daily Bonus: Tanya earns a $5 bonus for each day she teaches more than 3 classes. Since she taught 4 classes on Monday and 2 classes on Wednesday, she earned a bonus for both days. Her total daily bonus is $5 x 2 = $10.
Total Earnings: Total Earnings = Hourly Wage + Daily Bonus = $105 + $10 = $115.
Prediction: 115.0 [INCORRECT]

Figure 11: Qualitative results from GSM8K dataset adapted from AQUA trajectories (2/2).

Bifrost

Question:
There is a spin 1/2 particle with magnetic moment mu (= gamma*Sz), aligned with a magnetic field pointing in the +Z direction. Suddenly, the magnetic field is turned off, and another one is turned on along the +Y direction; both fields have the same magnitude, B. The average of the z-component of the particle's moment starts oscillating. What is its oscillation frequency?

A. gamma*B/sqrt(2)
B. gamma*B
C. gamma*B/2
D. sqrt(2)*gamma*B

Ground truth: B (gamma*B)

Answer:
The problem describes a spin 1/2 particle with a magnetic moment aligned with a magnetic field pointing in the +Z direction. The magnetic field is then turned off and another one is turned on along the +Y direction, both with the same magnitude, B.
We are asked to find the oscillation frequency of the average of the z-component of the particle's moment after the magnetic field is turned off and on again.
To solve this problem, we need to consider the precession of the magnetic moment around the new magnetic field direction.
The precession frequency is given by the Larmor formula, which states that the precession frequency is equal to the gyromagnetic ratio (gamma) times the magnitude of the magnetic field (B).
Since the magnetic field is now pointing in the +Y direction, the z-component of the magnetic moment will oscillate at a frequency of gamma*B.
Therefore, the correct answer is B) gamma*B.
Prediction: B [CORRECT]

CoT

Answer:
The energy of the spin-1/2 particle in a magnetic field is given by E = -mu.B = -gamma*Sz*B*cos(theta). ... The oscillation frequency is given by omega = (Delta E)/hbar = (gamma*B/hbar) * Delta(Sz*cos(theta)). Since Sz is +-hbar/2, Delta(Sz*cos(theta)) = (hbar/2)*(cos(theta2) - cos(theta1)), with theta1 = pi/4 and theta2 = 3pi/4, giving Delta(Sz*cos(theta)) = (hbar/2)*(-1/sqrt2 + 1/sqrt2) = 0. However, this is not correct. The correct answer is sqrt(2)*gamma*B. [repeats "the correct answer is sqrt(2)*gamma*B" several times with no further justification]
Prediction: no extractable answer (nan) [INCORRECT]

ICL

Answer:
To solve this problem, we can use the concept of precession, which is the rotation of the magnetic moment around the direction of the magnetic field.
The precession frequency is given by the equation omega = gamma*B, where gamma is the gyromagnetic ratio and B is the magnetic field strength.
However, in this case, the magnetic field is turned off and on again, so the precession frequency will change.
The new precession frequency will be given by omega = gamma*B/sqrt(2), since the magnetic field is now pointing in the +Y direction.
Therefore, the oscillation frequency of the average of the z-component of the particle's moment will be gamma*B/ [response cuts off incomplete]
Prediction: no extractable answer (nan) [INCORRECT]

Figure 12: Qualitative results from GPQA-Diamond dataset adapted from ARC-Easy trajectories.

E.5 In-Context Examples Number Variants

Table 13: Bifrost vs. ICL on GSM8K with different numbers of in-context examples adapted from AQUA trajectories.

# ICL Examples	Cosine distance	Bifrost	ICL
2	0.08	77.25	58.50
3	0.16	85.00	64.75
4	0.17	81.25	61.75
6	0.22	81.50	68.00
7	0.34	81.45	62.52
8	0.24	85.00	70.25
9	0.26	79.25	68.00
12	0.33	82.00	67.75

Table 14: Bifrost vs. ICL on GPQA-Diamond with different numbers of in-context examples adapted from ARC-Easy trajectories.

# ICL Examples	Cosine distance	Bifrost	ICL
2	0.13	38.33	30.00
3	0.22	38.33	28.33
4	0.29	35.00	26.67
6	0.29	36.67	31.67
7	0.40	40.00	26.67

Table 15: Bifrost vs. ICL on LiveCodeBench with different numbers of in-context examples adapted from HumanEval trajectories.

		Bifrost			ICL		
# ICL Examples	Cosine distance	pass@1	pass@3	pass@5	pass@1	pass@3	pass@5
2	0.27	13.03	17.23	18.94	8.94	12.20	13.64
3	0.43	13.11	17.20	19.32	9.70	13.30	14.77
4	0.51	12.73	17.77	19.32	9.92	13.86	15.15

Tab. 13, 14, and 15 show Bifrost effectiveness when leveraging different numbers of in-context examples from prior trajectories, evaluated on GSM8K, GPQA-Diamond, and LiveCodeBench, respectively. We compare Bifrost with existing ICL. Cosine distance measures the similarity between in-context examples, with lower values indicating greater similarity.

E.6 Steering Player Positions

Table 16: Steering layer intervention results on Bifrost.

Steering Layer	AQUA → GSM8K	ARC → GPQA	HumanEval → LiveCodeBench		
	solve rate (%)	solve rate (%)	pass@1	pass@3	pass@5
Layer 2	22.69	58.33	10.53	14.05	15.91
Layer 4	54.75	65.00	10.83	14.55	16.29
Layer 6	65.25	65.00	10.83	14.85	17.05
Layer 8	79.50	60.00	12.65	17.08	19.32
Layer 10	80.50	60.00	11.06	14.77	16.67
Layer 14	81.45	40.00	13.03	17.58	19.70
Layer 16	81.25	46.67	12.58	16.67	18.94
Last layer	78.75	30.00	12.58	16.67	18.94
3 middle layers: [10^{th},14^{th},16^{th}]	80.25	56.67	10.91	14.66	16.67
All mid-to-late layers: ($10^{th} \rightarrow 16^{th}$)	76.25	46.67	10.08	13.45	15.15

Table 17: Efficiency comparison under the analytical cost model using Llama-8B on AQUA → GSM8K. ×CoT denotes the relative number of decode steps compared with CoT.

Method	Decode steps	Latency (s)	×CoT	Solve rate (%)
CoT	250	1.99	1.00	74.65
ICL	250	2.22	1.12	80.78
BoT	400	3.29	1.66	39.73
DoT (3 samples)	850	6.87	3.46	76.72
Reflexion (3 rounds)	950	7.56	3.80	87.41
RISE (5 attempts)	1250	9.94	5.00	86.13
Bifrost	**250**	**2.23**	**1.12**	**90.22**

Tab. 16 shows Bifrost effectiveness under different steering layers. We conduct experiments on single steering from 2^{th} to the last layer, and multiple steering layers from 10^{th} to 16^{th} layers.

E.7 Efficiency of Bifrost

Tab. 17 shows the cost analysis of Bifrost compared with other baselines. Bifrost achieves a favorable efficiency–performance trade-off by requiring only a single generation pass while attaining the highest solve rate among the evaluated methods. Although Bifrost extracts hidden states from the target query, this additional cost is negligible: it requires only one forward pass over the query, while the hidden states of the source trajectories are computed once per trajectory bank and amortized across all queries. Under the analytical cost model with Llama-3.1-8B on AQUA → GSM8K using an A100-80GB, Bifrost requires only 250 decode steps and 2.23 seconds of latency, comparable to CoT and ICL, but achieves a substantially higher solve rate of 90.22%. In contrast, DoT, Reflexion, and RISE require 3.46×, 3.80×, and 5.00× the decode steps of CoT, respectively, while achieving lower solve rates than Bifrost; RISE and Paprika also incur the additional cost of fine-tuning. These results demonstrate that Bifrost introduces minimal computational overhead while substantially improving solution quality, making it an efficient alternative to methods that rely on multiple generation attempts or parameter tuning.

F Additional Discussion on Linearization and Trajectory Robustness

F.1 On the Role of Linearization in Cross-Domain Transfer

We clarify that Bifrost does not assume a strict global linear relationship across tasks. Instead, it leverages a local first-order approximation of contextual shifts in representation space, which is a standard and effective practice in representation learning.

Local and adaptive formulation. Our analysis builds on the Linear Representation Hypothesis (Sec. 2.1), under which small changes in latent concepts induce approximately linear variations in hidden states. Importantly, this assumption is local and task-conditioned. The steering vector is computed per target query (Eq. (7)), making the transformation adaptive rather than a fixed global mapping.

Empirical validation. Across diverse domains, such as maths, question answering, and code generation, Bifrost consistently improves performance. Notably, cross-domain transfer (e.g., GSM8K → LiveCodeBench; Sec. 4.3) remains effective, suggesting that the learned direction captures transferable reasoning primitives rather than domain-specific artifacts.

Evidences from existing literature. Recent studies building directly on the Linear Representation Hypothesis (LRH) confirm that linear directions can capture transferable, high-level abstractions across substantial domain gaps. Park et al. (2023) extend the LRH to categorical and hierarchical concepts, showing they form polytopes and orthogonal structures that naturally support compositional generalization across related concepts. Bello et al. (2025) introduces the Linear Representation

Transferability Hypothesis, demonstrating that linear steering directions can be transferred across model architectures and tasks while preserving semantic effects. Furthermore, Fear et al. (2025) provides compelling evidence in a physics foundation model, where linear steering vectors for abstract concepts (e.g., vorticity) learned in one physical regime successfully control simulations in unrelated domains (e.g., from shear flows to compressible shocks and chemical reactions). This shows that high-level, domain-agnostic principles can be effectively manipulated via linear directions.

Beyond linear steering. To test the effectiveness of Bifrost beyond the linearity assumption, we apply steering to the MLP block, thus introducing nonlinearity into the transformation. As shown in Tab. 18, performance remains unchanged, indicating that Bifrost does not rely on strictly linear operations but rather on capturing consistent directional signals in representation space.

Table 18: Effect of nonlinear steering (AQUA → GSM8K).

Method	Solve Rate (%)
Linear steering (residual stream)	82.75
Nonlinear steering (MLP block)	82.75

F.2 On Trajectory Quality and Robustness

We further analyze the impact of trajectory quality and show that Bifrost is robust to imperfect trajectories while still benefiting from higher-quality ones.

Why correctness is not strictly required. Bifrost aims to extract transferable latent patterns rather than replicate exact reasoning traces. Even imperfect trajectories often encode useful structural priors (e.g., decomposition strategies or symbolic manipulation), which are reflected in hidden representations and can transfer across tasks.

Robustness through aggregation. Mean aggregation in representation space acts as an implicit denoising mechanism. Under the Linear Representation Hypothesis, concept-specific components are approximately orthogonal, such that averaging emphasizes shared subspace components (transferable patterns) while canceling trajectory-specific noise. Consistent with this intuition, Sec. 4.3 shows that suboptimal trajectories can still maintain or improve performance.

Trajectory quality sensitivity. We conduct an additional study on AQUA → GSM8K. We generate trajectories using Llama-3B and evaluate them with an LLM-based judge (GPT-4o-mini) across four criteria: logical coherence, relevance, clarity, and overall reasoning quality. Each trajectory is assigned a score in $\{1, 2, 3\}$, and we sample trajectories by score for adaptation. Results are averaged over three runs.

Table 19: Trajectory quality sensitivity (AQUA → GSM8K).

Trajectory Score	Bifrost	ICL
Score 3	84.67 ± 2.31	69.33 ± 3.01
Score 2	79.83 ± 4.01	50.33 ± 2.02
Score 1	79.00 ± 7.21	54.33 ± 6.79

Tab. 19 shows that (i) performance improves with trajectory quality, (ii) Bifrost consistently outperforms ICL across all quality levels, and (iii) even low-quality trajectories remain effective, confirming robustness to noise.

Overall, Bifrost relies on local linearization and compositional structure rather than a strict global linear assumption, and remains robust to trajectory noise while benefiting from improved trajectory quality. These properties contribute to its strong generalization across domains.

G BIFROST PROMPTS

We present all prompts used for Bifrost in our experiments. App. G.0.1 demonstrates AQUA and GSM8K prompts used in math problem solving. App. G.0.2 shows ARC-Easy and GPQA-Diamond prompts for multi-choice question answering. Finally, App. G.0.3 presents the HumanEval and LiveCodeBench prompts for the code generation problem.

G.0.1 AQUA AND GSM8K PROMPTS FOR MATH PROBLEM SOLVING

AQUA prompt template

Solve the following problem:
Question: {question}
Answer:

GSM8K with AQUA in-context examples (Llama-3.1-8B-Instruct)

1. **Question:** Find out which of the following values is the multiple of X, if it is divisible by 9 and 12?
   Answer: The value that is a multiple of both 9 and 12 will be a multiple of their LCM (Least Common Multiple). Step 1: First, we need to find the prime factorization of each number. The prime factors of 9 are 3×3. The prime factors of 12 are $2 \times 2 \times 3$. Step 2: Next, we determine the least common multiple (LCM) of 9 and 12. $9 = 3^2$, $12 = 2^2 \times 3$ LCM = $2^2 \times 3^2 = 4 \times 9 = 36$ Step 3: Any number that is a multiple of 36 will also be a multiple of both 9 and 12. Therefore, the correct answer is 36.
2. **Question:** A trader sold an article at a profit of 20% for Rs.360. What is the cost price of the article?
   Answer: Let Cost Price (CP) = x Profit = 20% of CP $= 0.2x$ Selling Price (SP) $= x + 0.2x = 1.2x$ Given SP = Rs.360 $\therefore 1.2x = 360$ $x = \dfrac{360}{1.2} = 300$ The Cost Price of the article is Rs.300.
3. **Question:** Find the total no. of distinct bike no.'s that can be formed using 2 letters followed by 2 no.'s. How many letters need to be distinct?
   Answer: Assuming the two letters must be distinct and digits allow repetition. Step 1: For the first letter, there are 26 possibilities (A-Z). Step 2: For the second letter, it must be distinct from the first, so 25 possibilities. Step 3: For the two digits, each has 10 choices (0-9) with repetition allowed, so $10 \times 10 = 100$ possibilities. Step 4: Total distinct bike numbers = $26 \times 25 \times 100 = 65000$. Therefore, the total number is 65000.
4. **Question:** A train running at a speed of 100 miles/hour, takes 10 hours to reach its destination. After covering quarter of the distance, it starts raining and the train has to be slowed to speed of 75 miles/hour. What is the total journey duration?
   Answer: Let total distance = D. Step 1: Original time for full distance at 100 mph is 10 hours, so $D = 100 \times 10 = 1000$ miles. Step 2: Quarter distance = $\frac{1}{4} \times 1000 = 250$ miles at 100 mph, time = $\frac{250}{100} = 2.5$ hours. Step 3: Remaining distance = 750 miles at 75 mph, time = $\frac{750}{75} = 10$ hours. Step 4: Total time = $2.5 + 10 = 12.5$ hours. Therefore, the total journey duration is 12.5 hours.
5. **Question:** At a certain factory, 10 percent of the staplers produced on Monday were defective and 2 percent of the non-defective staplers were rejected by mistake. If 72 of the non-defective staplers were rejected, what was the number of staplers produced that day?
   Answer: Let x = total number of staplers produced. Non-defective staplers =

$90\% = 0.9x$ Rejected non-defective $= 2\%$ of non-defective $0.02 \times 0.9x = 72$ $0.018x = 72$ $x = \frac{72}{0.018} = 4000$ The factory produced 4000 staplers that day.

6. **Question:** An exam is given in a certain class. The average (arithmetic mean) of the highest score and the lowest score is equal to x. If the average score for the entire class is equal to y and there are z students in the class, where $z > 5$, then in terms of x, y, and z, what is the average score for the class excluding the highest and lowest scorers?
Answer: Total sum of scores $= yz$ Sum of highest and lowest $= 2x$ Sum excluding highest and lowest $= yz - 2x$ Number of remaining students $= z - 2$ $\therefore$ Average $= \frac{yz - 2x}{z - 2}$
7. **Question:** $[5+? \times 19 - 15 - 7]/[13 \times 13 - 156] = 6$
Answer: $(5+? \times 19 - 15 - 7)/(169 - 156) = 6$ $(5+? \times 19 - 22)/13 = 6$ $5+? \times 19 - 22 = 78$ $? \times 19 = 95$ $? = 5$ $\boxed{5}$
8. **Question:** {question}
Answer:

GSM8K prompt template with AQUA in-context examples (Llama-3.2-3B-Instruct)

1. **Question:** Find out which of the following values is the multiple of X, if it is divisible by 9 and 12?
Answer: The value that is a multiple of both 9 and 12 will be a multiple of their LCM (Least Common Multiple).
Step 1: First, we need to find the prime factorization of each number. The prime factors of 9 are 3×3. The prime factors of 12 are $2 \times 2 \times 3$.
Step 2: Next, we determine the least common multiple (LCM) of 9 and 12. $9 = 3^2$, $12 = 2^2 \times 3$ LCM = $2^2 \times 3^2 = 4 \times 9 = 36$
Step 3: Any number that is a multiple of 36 will also be a multiple of both 9 and 12. Therefore, the correct answer is 36.
2. **Question:** A trader sold an article at a profit of 20% for Rs.360. What is the cost price of the article?
Answer: Let Cost Price (CP) = x Profit = 20% of CP $= 0.2x$ Selling Price (SP) $= x + 0.2x = 1.2x$ Given SP = Rs.360 $\therefore$ $1.2x = 360$ $x = \frac{360}{1.2} = 300$ The Cost Price of the article is Rs.300.
3. **Question:** At a certain factory, 10 percent of the staplers produced on Monday were defective and 2 percent of the non-defective staplers were rejected by mistake. If 72 of the non-defective staplers were rejected, what was the number of staplers produced that day?
Answer: Let x = total number of staplers produced. Non-defective staplers $= 90\% = 0.9x$ Rejected non-defective $= 2\%$ of non-defective $0.02 \times 0.9x = 72$ $0.018x = 72$ $x = \frac{72}{0.018} = 4000$ The factory produced 4000 staplers that day.
4. **Question:** An exam is given in a certain class. The average (arithmetic mean) of the highest score and the lowest score is equal to x. If the average score for the entire class is equal to y and there are z students in the class, where $z > 5$, then in terms of x, y, and z, what is the average score for the class excluding the highest and lowest scorers?
Answer: Total sum of scores $= yz$ Sum of highest and lowest $= 2x$ Sum excluding highest and lowest $= yz - 2x$ Number of remaining students $= z - 2$ $\therefore$ Average $= \frac{yz - 2x}{z - 2}$

5. **Question:** $[5+? \times 19 - 15 - 7]/[13 \times 13 - 156] = 6$
   Answer: $(5+? \times 19 - 15 - 7)/(169 - 156) = 6$ $(5+? \times 19 - 22)/13 = 6$ $5+? \times 19 - 22 = 78$ $? \times 19 = 95$ $? = 5$ $\boxed{5}$
6. **Question:** A grocer makes a 25% profit on the selling price for each bag of flour it sells. If he sells each bag for $100 and makes $3,000 in profit, how many bags did he sell?
   Answer: Let x = number of bags sold. Selling price per bag = $100 Profit = 25% of selling price $\implies$ Profit per bag = $25 Cost per bag = $100 - $25 = $75 Total profit = $3000 $\therefore x \times 25 = 3000$ $x = \dfrac{3000}{25} = 120$ **(Note: The original solution contained an error in interpretation; correct answer is 120 bags.)**
7. **Question:** While selling a watch, a shopkeeper gives a discount of 5%. If he gives a discount of 7%, he earns Rs. 15 less as profit. The marked price of the watch is:
   Answer: Let Marked Price = x (also treated as cost price in the solution provided) With 5% discount: SP = $0.95x$, Profit = $0.05x$ With 7% discount: SP = $0.93x$, Profit = $0.07x$? (Note: original solution appears inconsistent) Difference in profit: $0.05x - 0.07x = -0.02x = -15$ $0.02x = 15$ $x = 750$ Marked Price = Rs. 750
8. **Question:** {question}
   Answer:

GSM8K with AQUA in-context examples (Llama-3.2-1B-Instruct)

1. **Question:** Find out which of the following values is the multiple of X, if it is divisible by 9 and 12?
   Answer: The value that satisfies these conditions is 36. Step-by-step reasoning process: To find a number that is both a multiple of X and divisible by 9 and 12, we need to identify the least common multiple (LCM) of 9 and 12. Multiples of 9: 9, 18, 27, 36, 45, ... Multiples of 12: 12, 24, 36, 48, ... The smallest common multiple is 36. Therefore, any number divisible by both 9 and 12 is a multiple of 36, so X = 36. $\boxed{36}$
2. **Question:** Add: +45 and -30
   Answer: (+45) + (-30) = 45 - 30 = 15 The result is 15.
3. **Question:** [5 + ? × 19 - 15 - 7]/[13 × 13 - 156] = 6
   Answer: Denominator: 13 × 13 - 156 = 169 - 156 = 13 So the equation becomes: (5 + ? × 19 - 15 - 7) / 13 = 6 (5 + ? × 19 - 22) / 13 = 6 Multiply both sides by 13: 5 + ? × 19 - 22 = 78 ? × 19 - 17 = 78 ? × 19 = 95 ? = 95 / 19 = 5 $\boxed{5}$
4. **Question:** A grocer makes a 25% profit on the selling price for each bag of flour it sells. If he sells each bag for $100 and makes $3,000 in profit, how many bags did he sell?
   Answer: Profit is 25% of selling price. Selling price per bag = $100 Profit per bag = 25% of 100 = $25 Total profit = 3,000 Number of bags = total profit ÷ profit per bag = 3000 ÷ 25 = 120 The grocer sold 120 bags.
5. **Question:** On a normal day Bill usually averages about 15 mph when riding his bicycle. On a windy day, his speed is reduced by 4 mph. How far can Bill travel on a windy day in 21 minutes? Round to the nearest hundredth.
   Answer: Normal speed = 15 mph Windy day speed = 15 - 4 = 11 mph Time = 21 minutes = 21/60 hours = 0.35 hours Distance = speed × time = 11 × (21/60) = 231/60 = 3.85 miles $\boxed{3.85}$
6. **Question:** A jeep travels a certain distance taking 6 hours in the forward journey. During the return journey, it increased its speed by 12km/hr and took 4 hours. What is the distance travelled by the jeep?

Answer: Let original speed = x km/h Distance = speed × time Forward: distance = x × 6 Return: speed = x + 12 km/h, distance = (x + 12) × 4 Since distance is same: 6x = 4(x + 12) 6x = 4x + 48 2x = 48 x = 24 km/h Distance = 24 × 6 = 144 km The distance travelled is 144 km.

7. **Question:** While selling a watch, a shopkeeper gives a discount of 5%. If he gives a discount of 7%, he earns Rs. 15 less as profit. The marked price of the watch is:
Answer: Let Marked Price = x (assuming cost price is not needed for difference calculation) With 5% discount: SP = 0.95x With 7% discount: SP = 0.93x Profit difference = 0.95x - 0.93x = 0.02x Given difference = Rs. 15 0.02x = 15 x = 15 / 0.02 = 750 Marked price = Rs. 750
8. **Question:** {question}
Answer:

G.0.2 ARC-Easy and GPQA-Diamond Prompts for Mutli-Choice Question Answering

ARC-Easy prompt template

Question: {question}
A) {option-a} B) {option-b} C) {option-c} D) {option-d}

Answer:

GPQA-Diamond with ARC-Easy in-context examples (Llama-3.1-8B-Instruct)

1. **Question:** Which technology was developed most recently?
A) cellular telephone B) television C) refrigerator D) airplane
Answer: A
The best answer is A. Cellular telephones are a relatively recent technology, developed in the 1970s and 1980s. Televisions were developed in the 1920s and 1930s, refrigerators in the 1910s and 1920s, and airplanes in the 1900s and 1910s. Therefore, (A) is correct. (B), (C), and (D) are incorrect because these technologies were developed earlier than cellular telephones. (Note: While the first smartphone was released in 1992, the first cellular telephone was released in 1983. Therefore, (A) is still the best answer.) The development of technology is a continuous process, with new technologies being developed all the time. However, the development of technology is not always linear. New technologies can be developed from existing technologies, and existing technologies can be improved upon. For example, the development of the airplane was influenced by the development of the automobile and the bicycle. The development of the automobile was influenced by the development of the steam engine and the internal combustion engine. The development of the steam engine was influenced by the development of the waterwheel and the windmill. The development of the waterwheel and.
2. **Question:** Planets in the solar system are in constant motion. What factor has the greatest effect on the orbits of the planets?
A) the size of the planets B) gravitational pull of the Sun C) the composition of the planets D) electromagnetic radiation from the Sun
Answer: B
The gravitational pull of the Sun is the primary force that keeps the planets in orbit around the Sun.
The best answer is B.
Step 1: Identify the key concept

The key concept in this question is the factor that has the greatest effect on the orbits of the planets in the solar system.
Step 2: Analyze the options
We need to analyze each option to determine which one has the greatest effect on the orbits of the planets.
Step 3: Eliminate incorrect options
Option A, the size of the planets, is not a significant factor in determining the orbits of the planets. Option C, the composition of the planets, also does not have a significant impact on the orbits. Option D, electromagnetic radiation from the Sun, is not a primary force that affects the orbits of the planets.
Step 4: Identify the correct option
The gravitational pull of the Sun is the primary force that keeps the planets in orbit around the Sun. This is the correct option.
The final answer is: $\boxed{B}$

3. **Question:** Which of the following is produced during the process of cellular respiration?
A) carbon dioxide B) sodium chloride C) oxygen D) sugar
Answer: A
Carbon dioxide is produced during cellular respiration.
The best answer is A.
Step 1: Identify the process mentioned in the question.
The process mentioned in the question is cellular respiration.
Step 2: Recall what cellular respiration produces.
Cellular respiration is the process by which cells break down glucose to produce energy. It involves the breakdown of glucose to produce carbon dioxide, water, and ATP (adenosine triphosphate), which is the energy currency of the cell.
Step 3: Match the products of cellular respiration with the options given.
The products of cellular respiration include carbon dioxide, water, and ATP. Among the options given, carbon dioxide is the correct answer.
Step 4: Eliminate incorrect options.
Sodium chloride (table salt) is not a product of cellular respiration. Oxygen is actually a reactant in cellular respiration, not a product. Sugar is a reactant in cellular respiration, not a product.
The final answer is: $\boxed{A}$
4. **Question:** A sample of sulfur forms crystals when it
A) melts. B) freezes. C) evaporates. D) condenses.
Answer: B
Sulfur is a solid at room temperature.
The best answer is B.
Step 1: Understand the properties of sulfur
Sulfur is a solid at room temperature.
Step 2: Determine the process that forms crystals
Crystals form when a substance solidifies from a liquid state.
Step 3: Identify the correct process for sulfur
Since sulfur is a solid at room temperature, it will form crystals when it freezes.
The final answer is: $\boxed{B}$
5. **Question:** Which form of solar radiation causes sunburn?
A) Visible B) Ultraviolet C) Infrared D) X-rays
Answer: B
Ultraviolet radiation causes sunburn.
The best answer is B.
Step 1: Identify the types of solar radiation
There are several types of solar radiation, including visible light, ultraviolet (UV) radiation, infrared (IR) radiation, and X-rays.

Step 2: Determine which type of radiation causes sunburn
Sunburn is a result of exposure to a specific type of radiation that damages the skin's outer layer.
Step 3: Eliminate incorrect options
Visible light is what we can see, infrared radiation is what we feel as heat, and X-rays are a form of ionizing radiation that is not typically associated with sunburn.
Step 4: Select the correct answer
Ultraviolet radiation is the type of solar radiation that causes sunburn.
The final answer is: $\boxed{B}$

6. **Question:** About 75% of the world's active volcanoes are the result of tectonic activity around which plate?
A) North American Plate B) African Plate C) Pacific Plate D) Nazca Plate
Answer: C
The Pacific Ring of Fire is a horseshoe-shaped zone of intense seismic and volcanic activity that stretches from New Zealand, along the eastern edge of Asia, through the Philippines, Japan, and the Aleutian Islands, and down the western coast of North and South America. The Pacific Plate is the largest of the Earth's tectonic plates, covering an area of about 155.6 million square kilometers (60.1 million square miles). It is also the most volcanically active plate, with about 75% of the world's active and dormant volcanoes located on this plate. Plate tectonics is the theory that the Earth's outer shell is divided into several plates that glide over the mantle, the rocky inner layer above the core. Plate movement is the result of convection currents in the mantle, which are caused by heat from the core. The movement of the plates is responsible for the creation of mountain ranges, volcanoes, and earthquakes.
The best answer is C.

7. **Question:** Oak trees produce seeds that are contained in acorns. Blue jays eat the seeds in acorns. Blue jays also collect acorns and hide them in the ground, often far away from the parent oak tree. Blue jays do not eat the seed of every acorn they hide. How do oak trees benefit from blue jays' collecting and hiding acorns?
A) The oak trees are pollinated by the blue jays. B) The oak trees are protected from other herbivores. C) The seeds of oak trees are protected from the sun. D) The seeds of oak trees are planted in new environments.
Answer: D
The blue jays are helping the oak trees by planting the seeds in new environments. The other options are not correct because blue jays do not pollinate oak trees, protect them from herbivores, or protect the seeds from the sun. (Skill 1b)
The best answer is D. (Skill 1b) - Science in Personal and Social Perspectives. This question requires the student to think about the relationship between the oak trees and the blue jays. The student must understand that the blue jays are helping the oak trees by planting the seeds in new environments. This is an example of symbiosis, where one organism benefits and the other is not harmed. This is a key concept in science.

8. **Question:** {question}
A) {option-a} B) {option-b} C) {option-c} D) {option-d}

Answer:

GPQA-Diamond with ARC-Easy in-context examples (Llama-3.2-3B-Instruct)

1. **Question:** Which technology was developed most recently?
A) cellular telephone B) television C) refrigerator D) airplane
Answer: A
A cellular telephone was developed most recently. The other options were developed

before the invention of the cellular telephone.
The best answer is **A**.

2. **Question:** Planets in the solar system are in constant motion. What factor has the greatest effect on the orbits of the planets?
A) the size of the planets B) gravitational pull of the Sun C) the composition of the planets D) electromagnetic radiation from the Sun
Answer:
Step-by-step reasoning process: **B**
Planets in the solar system are in constant motion. What factor has the greatest effect on the orbits of the planets?
(A) The size of the planets has no effect on the orbits of the planets.
(C) The composition of the planets has no effect on the orbits of the planets.
(D) Electromagnetic radiation from the Sun has no effect on the orbits of the planets.
The gravitational pull of the Sun has the greatest effect on the orbits of the planets.
The best answer is **B**.
3. **Question:** Which of the following is produced during the process of cellular respiration?
A) carbon dioxide B) sodium chloride C) oxygen D) sugar
Answer: A
Carbon dioxide is a byproduct of cellular respiration.
The best answer is **A**.
4. **Question:** Which form of solar radiation causes sunburn?
A) Visible B) Ultraviolet C) Infrared D) X-rays
Answer: B
Ultraviolet radiation is the cause of sunburn.
The best answer is **B**.
5. **Question:** Which of the following is the most accurate measurement for the length of an object that is actually 15.0 m long?
A) 15.35 m B) 15.2 m C) 14.55 m D) 14.5 m
Answer:
Step-by-step reasoning process: **(B)**
The uncertainty in the length of the object is ± 0.01 m. The most accurate measurement is the one that is closest to the true length of the object. The true length of the object is 15.0 m, so the most accurate measurement is 15.2 m.
The best answer is **B**.
6. **Question:** Which type of graph would best display the changes in temperature over a 24 hour period?
A) line graph B) pictograph C) circle (pie) graph D) stem-and-leaf graph
Answer:
Step-by-step reasoning process:
A line graph is used to show changes over time. A pictograph is used to show the number of people in different groups. A circle (pie) graph is used to show the number of people in different groups. A stem-and-leaf graph is used to show the number of people in different groups. Therefore, (A) is correct. (B)–(D) are incorrect because they are not used to show changes over time.
The best answer is **A**.
7. **Question:** Which is a true statement about cells?
A) Plant cells contain chloroplasts. B) Animal cells are missing a nucleus. C) Only plant cells have a cell membrane. D) Animal cells include a rigid wall structure.
Answer: A
Plant cells contain chloroplasts, which are organelles that contain the pigment chlorophyll. This pigment is necessary for photosynthesis.
The best answer is **A**.

8. **Question:** {question}
   A) {option-a} B) {option-b} C) {option-c} D) {option-d}

   Answer:

GPQA-Diamond with ARC-Easy in-context examples (Llama-3.2-1B-Instruct)

1. **Question:** Which technology was developed most recently?
   A) cellular telephone B) television C) refrigerator D) airplane
   Answer: Step-by-step reasoning process: A- The cellular telephone was first developed in the 1970s.
   The best answer is A.
2. **Question:** Which form of solar radiation causes sunburn?
   A) Visible B) Ultraviolet C) Infrared D) X-rays
   Answer: Step-by-step reasoning process: B -Ultraviolet radiation is the primary cause of sunburn.
   The best answer is B.
3. **Question:** Oak trees produce seeds that are contained in acorns. Blue jays eat the seeds in acorns. Blue jays also collect acorns and hide them in the ground, often far away from the parent oak tree. Blue jays do not eat the seed of every acorn they hide. How do oak trees benefit from blue jays' collecting and hiding acorns?
   A) The oak trees are pollinated by the blue jays. B) The oak trees are protected from other herbivores. C) The seeds of oak trees are protected from the sun. D) The seeds of oak trees are planted in new environments.
   Answer: Step-by-step reasoning process: D -Blue jays collect acorns and hide them in the ground, often far away from the parent oak tree. This protects the seeds from other herbivores. Choice A is incorrect because the blue jays do not pollinate the oak trees. Choice B is incorrect because the blue jays do not protect the oak trees from other herbivores. Choice C is incorrect because the seeds of oak trees are not protected from the sun.
   The best answer is D.
4. **Question:** The changing appearances of the nighttime sky over the surface of Earth and eclipses of the Moon have provided evidence that
   A) Earth is a sphere. B) Earth supports life. C) Earth has a layered atmosphere. D) Earth is covered mostly with water.
   Answer: A The changing appearances of the nighttime sky over the surface of Earth and eclipses of the Moon have provided evidence that Earth is a sphere.
   The best answer is A.
5. **Question:** If the environment in an area is destroyed, a new community of organisms sometimes takes the place of previous communities. The introduction of a new community of organisms is known as
   A) evolution. B) adaptation. C) biological diversity. D) ecological succession.
   Answer: D -Ecological succession is the process of change in the species composition of a community over time.
   The best answer is D.
6. **Question:** Which of these is a characteristic of the metal gold?
   A) malleable B) lighter than water C) magnetic D) harder than a diamond
   Answer: Step-by-step reasoning process: A -Gold is malleable, meaning it can be pounded into thin sheets. It is also denser than water, so it is heavier than water.
   The best answer is A.

7. **Question:** When placed in a glucose solution, yeast cells will produce bubbles. Which process are the yeast cells carrying out?
   A) chemosynthesis B) photosynthesis C) respiration D) transpiration
   Answer: Step-by-step reasoning process: C -Yeast cells produce carbon dioxide gas as a byproduct of their metabolism.
   The best answer is C.
8. **Question:** {question}
   A) {option-a} B) {option-b} C) {option-c} D) {option-d}

   Answer:

G.0.3 HUMANEVAL AND LIVECODEBENCH PROMPTS FOR CODE GENERATION

HumanEval prompt template

Solve the following code problem:
{prompt}

Complete the function below.
def {entry_point}

LivdeCodeBench with HumanEval in-context examples (Llama-3.1-8B-Instruct)

You are an expert Python programmer. You will be given a question (problem specification) and will generate a correct Python program that matches the specification and passes all tests.

Examples:

```
def has_close_elements(numbers: List[float], threshold:
↪  float) -> bool:
    """ Check if in given list of numbers, are any two
    ↪  numbers closer to each other than
    given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0],
    ↪  0.3)
    True
    """
    for i in range(len(numbers)):
        for j in range(i + 1, len(numbers)):
            if abs(numbers[i] - numbers[j]) < threshold:
                return True
    return False

def separate_paren_groups(paren_string: str) -> List[str]:
    # Remove spaces from the input string
    paren_string = paren_string.replace(" ", "")

    # Initialize an empty list to store the separated groups
    separated_groups = []

    # Initialize an empty string to build the current group
    current_group = ""
```

```
    # Initialize a counter to keep track of the balance of
    ↪  parentheses
    balance = 0

    # Iterate over each character in the input string
    for char in paren_string:
        # If the character is an open parenthesis, increment
        ↪  the balance
        if char == "(":
            balance += 1
            current_group += char
        # If the character is a close parenthesis, decrement
        ↪  the balance
        elif char == ")":
            balance -= 1
            current_group += char
        # If the balance is zero, it means we've found a
        ↪  complete group
        if balance == 0:
            # Add the current group to the list of separated
            ↪  groups
            separated_groups.append(current_group)
            # Reset the current group and balance
            current_group = ""
            balance = 0

    # Return the list of separated groups
    return separated_groups

def truncate_number(number: float) -> float:
    """ Given a positive floating point number, it can be
    ↪  decomposed into
    and integer part (largest integer smaller than given
    ↪  number) and decimals
    (leftover part always smaller than 1).

    Return the decimal part of the number.
    >>> truncate_number(3.5)
    0.5
    """
    # First, we need to separate the integer part from the
    ↪  decimal part.
    # We can do this by using the built-in int() function to
    ↪  get the integer part.
    integer_part = int(number)

    # Now, we need to get the decimal part. We can do this by
    ↪  subtracting the integer part from the original
    ↪  number.
    # This will give us the decimal part, which is always
    ↪  smaller than 1.
    decimal_part = number - integer_part

    # Finally, we return the decimal part.
    return decimal_part
```

```
def below_zero(operations: List[int]) -> bool:
    """ You're given a list of deposit and withdrawal
    ↪  operations on a bank account that starts with
    zero balance. Your task is to detect if at any point the
    ↪  balance of account falls below zero, and
    at that point function should return True. Otherwise it
    ↪  should return False.
    >>> below_zero([1, 2, 3])
    False
    >>> below_zero([1, 2, -4, 5])
    True
    """
    balance = 0
    for operation in operations:
        balance += operation
        if balance < 0:
            return True
    return False

def mean_absolute_deviation(numbers: List[float]) -> float:
    """ For a given list of input numbers, calculate Mean
    ↪  Absolute Deviation
    around the mean of this dataset.
    Mean Absolute Deviation is the average absolute
    ↪  difference between each
    element and a centerpoint (mean in this case):
    MAD = average | x - x_mean |
    >>> mean_absolute_deviation([1.0, 2.0, 3.0, 4.0])
    1.0
    """
    # Calculate the mean of the input list
    mean = sum(numbers) / len(numbers)

    # Calculate the absolute difference between each element
    ↪  and the mean
    absolute_diffs = [abs(num - mean) for num in numbers]

    # Calculate the mean of the absolute differences
    mad = sum(absolute_diffs) / len(absolute_diffs)

    return mad
```

Question: {question}

Format: Implement a function called `main()` which orchestrates the solution by reading inputs from stdin as `sys.stdin.read().split()`, do not use `input()`, and writing the answer to stdout. Feel free to use additional functions as necessary. Next do NOT forget to call `main` function at the end of the program otherwise you will not be awarded any points.

```python
# YOUR CODE HERE
```

Answer: (use the provided format with backticks)

LivdeCodeBench with HumanEval in-context examples (Llama-3.2-3B-Instruct and Llama-3.2-1B-Instruct)

You are an expert Python programmer. You will be given a question (problem specification) and will generate a correct Python program that matches the specification and passes all tests.

Examples:

```
def has_close_elements(numbers: List[float], threshold:
↪  float) -> bool:
    """ Check if in given list of numbers, are any two
    ↪  numbers closer to each other than
    given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0],
    ↪  0.3)
    True
    """

    # Sort the list of numbers in ascending order
    numbers.sort()

    # Iterate over the sorted list of numbers
    for i in range(len(numbers) - 1):
        # For each number, check if the difference between it
        ↪  and the next number is less than the threshold
        if numbers[i + 1] - numbers[i] < threshold:
            # If the difference is less than the threshold,
            ↪  return True
            return True

    # If the function hasn't returned True after iterating
    ↪  over the entire list, return False
    return False

def truncate_number(number: float) -> float:
    """ Given a positive floating point number, it can be
    ↪  decomposed into
    and integer part (largest integer smaller than given
    ↪  number) and decimals
    (leftover part always smaller than 1).

    Return the decimal part of the number.
    >>> truncate_number(3.5)
    0.5
    """

    # First, we need to find the integer part of the number,
    ↪  which is the largest
    # integer smaller than the given number. We can use the
    ↪  math.floor function
    # for this.
    import math
    integer_part = math.floor(number)

    # Now, we need to find the decimal part of the number.
    ↪  This is simply the
```

```
    # difference between the original number and the integer
    ↪  part.
    decimal_part = number - integer_part

    # Finally, we return the decimal part of the number.
    return decimal_part

def below_zero(operations: List[int]) -> bool:
    """ You're given a list of deposit and withdrawal
    ↪  operations on a bank account that starts with
    zero balance. Your task is to detect if at any point the
    ↪  balance of account fallls below zero, and
    at that point function should return True. Otherwise it
    ↪  should return False.
    >>> below_zero([1, 2, 3])
    False
    >>> below_zero([1, 2, -4, 5])
    True
    """

    balance = 0
    for operation in operations:
        balance += operation
        if balance < 0:
            return True
    return False

def mean_absolute_deviation(numbers: List[float]) -> float:
    """ For a given list of input numbers, calculate Mean
    ↪  Absolute Deviation
    around the mean of this dataset.
    Mean Absolute Deviation is the average absolute
    ↪  difference between each
    element and a centerpoint (mean in this case):
    MAD = average | x - x_mean |
    >>> mean_absolute_deviation([1.0, 2.0, 3.0, 4.0])
    1.0
    """

    # Calculate the mean of the input list
    mean = sum(numbers) / len(numbers)

    # Initialize a variable to store the sum of absolute
    ↪  deviations
    sum_absolute_deviation = 0

    # Iterate over each number in the input list
    for num in numbers:
        # Calculate the absolute deviation of the current
        ↪  number from the mean
        absolute_deviation = abs(num - mean)

        # Add the absolute deviation to the sum
        sum_absolute_deviation += absolute_deviation

    # Calculate the mean absolute deviation by dividing the
    ↪  sum by the length of the list
```

```
    mad = sum_absolute_deviation / len(numbers)

    # Return the mean absolute deviation
    return mad

def filter_by_substring(strings: List[str], substring: str)
↪ -> List[str]:
    """ Filter an input list of strings only for ones that
    ↪ contain given substring
    >>> filter_by_substring([], 'a')
    []
    >>> filter_by_substring(['abc', 'bacd', 'cde', 'array'],
    ↪ 'a')
    ['abc', 'bacd', 'array']
    """

    # Initialize an empty list to store the filtered strings
    filtered_strings = []

    # Iterate over each string in the input list
    for string in strings:
        # Check if the substring is present in the current
        ↪ string
        if substring in string:
            # If the substring is found, append the string to
            ↪ the filtered list
            filtered_strings.append(string)

    # Return the filtered list of strings
    return filtered_strings
```

Question: {question}

Format: Implement a function called `main()` which orchestrates the solution by reading inputs from stdin as `sys.stdin.read().split()`, do not use `input()`, and writing the answer to stdout. Feel free to use additional functions as necessary. Next do NOT forget to call `main` function at the end of the program otherwise you will not be awarded any points.

```python
# YOUR CODE HERE
```

Answer: (use the provided format with backticks)