# Reasoning as Meta-Learning: An Optimization Perspective to Decipher Long CoT Reasoning in LLMs

**Anonymous authors**
**Paper under double-blind review**

## Abstract

We propose a novel framework RaML for interpreting the reasoning capabilities of large language models (LLMs) through the perspective of meta-learning. By conceptualizing reasoning trajectories as pseudo-gradient descent updates to the LLM's parameters, we identify parallels between LLM reasoning and various meta-learning paradigms. We formalize the training process for reasoning tasks as a meta-learning setup, with each question treated as an individual task, and reasoning trajectories serving as the inner loop optimization for adapting model parameters. Once trained on a diverse set of questions, the LLM develops fundamental reasoning capabilities that can generalize to previously unseen questions. Extensive empirical evaluations substantiate the strong connection between LLM reasoning and meta-learning. We further explore the potential of the proposed RaML to advance LLM reasoning and provide valuable insights. Our work deepens the understanding of LLM reasoning processes and provides actionable insights for enhancing these models through established meta-learning techniques.

## 1 Introduction

Recent advances in large language models (LLMs) (Dubey et al., 2024; Yang et al., 2024a; OpenAI, 2023; DeepSeek-AI et al., 2024) have substantially improved their ability to solve complex reasoning tasks. Modern LLMs often rely on chain-of-thought (CoT) reasoning (Wei et al., 2022; Chen et al., 2025b), where intermediate reasoning trajectories guide systematic problem solving through step-by-step logical deductions. State-of-the-art models, including OpenAI-o-series (OpenAI, 2024b;a), DeepSeek-R1 (DeepSeek-AI et al., 2025), Kimi-k1.5 (Team et al., 2025), Qwen3 (Yang et al., 2025a), and Gemini-2.5-Pro (Deepmind, 2025), have demonstrated remarkable performance on challenging mathematical and programming problems. A key feature of these models is their use of long reasoning trajectories, which enable iterative exploration and reflection and thereby improve test-time scaling (Li, 2025; Teng et al., 2025; Shah et al., 2025). These developments have not only advanced complex reasoning but also motivated extensive studies on supervised fine-tuning (SFT) and reinforcement learning (RL) for learning and refining extended reasoning processes (Qin et al., 2024; Min et al., 2024).

Despite these advances, ***understanding how LLMs acquire and express strong reasoning capabilities through reasoning trajectories*** remains critical for further improving their generalization and reliability (Jiang et al., 2020; Feng et al., 2023). The internal mechanisms underlying LLM reasoning remain largely opaque, making it difficult to explain how these models operate (Shi et al., 2025). Prior studies (Merrill et al., 2022; Chiang et al., 2023; Giannou et al., 2023; Liu et al., 2023) have investigated the representational power of reasoning trajectories and shown that LLMs equipped with such trajectories can solve complex problems. Other works (Gatmiry et al., 2024; Huang et al., 2025) suggest that reasoning trajectories are capable of describing sophisticated learning algorithms. However, the fundamental role of reasoning trajectories in LLM reasoning and their connections to diverse reasoning-oriented training paradigms remain insufficiently understood.

To address this gap, we propose **RaML** (Reasoning as Meta-Learning), a framework that interprets LLM reasoning from a meta-learning perspective (Schmidhuber, 1987; Andrychowicz et al., 2016; Ravi & Larochelle,

2017; Finn et al., 2017; Hospedales et al., 2022). The core idea of RaML is to conceptualize reasoning trajectories as pseudo-gradient descent updates to model parameters. This perspective reveals close connections between LLM reasoning and established meta-learning paradigms, such as Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) and Learning to Optimize (L2O) (Andrychowicz et al., 2016), providing a principled lens for understanding and improving reasoning in LLMs.

Specifically, RaML formulates reasoning-oriented training as a meta-learning process, where each question defines a distinct task, the corresponding reasoning trajectory serves as inner-loop adaptation, and the final answer functions as the query target for optimizing the LLM. Under this formulation, training aims to learn a generalizable *meta-initialization* that enables the model to adapt effectively through reasoning trajectories and produce accurate answers. This framework offers a theoretical foundation for analyzing LLM reasoning and creates opportunities to transfer insights from meta-learning to the development of stronger reasoning models.

We complement RaML with comprehensive experiments and analyses on both models trained from base LLMs and publicly available reasoning models. Inspired by prior meta-learning studies (Liu et al., 2020; Triantafillou et al., 2020; 2021; Agarwal et al., 2021; Lee et al., 2019; Collins et al., 2022), we examine key factors that influence LLM reasoning through the lens of meta-learning. Our empirical results provide evidence for a strong correspondence between trajectory-based LLM reasoning and meta-learning principles. Moreover, we demonstrate that RaML can guide the design of practical strategies for improving LLM reasoning, and we validate their effectiveness through reasoning-performance improvements. Our contributions are summarized as follows:

❶ We introduce RaML, a meta-learning-based framework for interpreting the reasoning processes of LLMs, supported by theoretical analysis.

❷ We provide empirical evidence showing a strong correspondence between trajectory-based LLM reasoning and meta-learning principles.

❸ We contextualize recent advances in LLM reasoning within the proposed framework, offering insights into their effectiveness.

❹ We propose simple yet effective strategies for improving LLM reasoning, drawing on meta-learning insights and validating their benefits empirically.

## 2 RaML: Interpreting LLM Reasoning as Meta-Learning

In this section, we introduce RaML, a framework for interpreting large language model (LLM) reasoning from a meta-learning perspective (Schmidhuber, 1987; Andrychowicz et al., 2016; Ravi & Larochelle, 2017; Finn et al., 2017; Hospedales et al., 2022). We first conceptualize reasoning trajectories as *pseudo-gradient updates* to the parameters of an LLM (§ 2.2), and then formulate reasoning-oriented training as a meta-learning process (§ 2.3). As a supplement, we discuss the connections between our formulation and existing training techniques in § E. The notation used throughout this section is summarized in § A.

### 2.1 Setup

We denote a large language model (LLM) by $\mathcal{M}_\theta$, where $\theta$ represents its parameters. We consider a reasoning task consisting of a set of questions $\mathcal{Q} = \{q_i\}_{i\in[1,n]}$ and their corresponding answers $\mathcal{A} = \{a_i\}_{i\in[1,n]}$. Given an instruction $I$ and a question $q_i$, the LLM is prompted to generate the answer $a_i$:

$$\mathrm{d}\left(\mathcal{M}_\theta(I; q_i)\right) \to a_i, \tag{1}$$

where d denotes the autoregressive decoding mechanism (Vaswani et al., 2017; Radford et al., 2018). Formally, the decoding distribution is defined as

$$p_\theta(a_i) = \prod_{0\le j<|a_i|} \mathrm{Softmax}\left(\boldsymbol{E}_O^T \cdot \mathcal{M}_\theta\left(I, q_i, t, a_i^0, \ldots, a_i^{j-1}\right)\right)\left[a_i^j\right]. \tag{2}$$

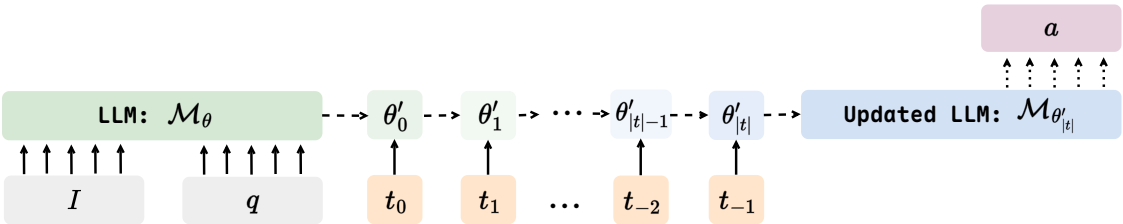


Figure 1: Illustration of a reasoning trajectory ($t$) as an optimization process over the LLM parameters $\theta$.

Here, $\{a_i^0, \ldots, a_i^{|a_i|}\}$ denotes the token sequence of the answer $a_i$, $t$ represents a possible intermediate reasoning trajectory, and $\boldsymbol{E}_O$ denotes the output embeddings over the vocabulary. Intuitively, $\mathcal{M}_\theta(I, q_i, t, a_i^0, \ldots, a_i^{j-1})$ corresponds to the activation determined by the parameters $\theta$ and the input context, while the next-token probability is obtained by projecting this activation onto the output embeddings. The activation at the final position is computed iteratively through the self-attention and feed-forward layers of the LLM.

During training, the model is optimized to produce accurate answers for the given questions:

$$\mathcal{O} = \max \sum_{(q_i, a_i) \in \mathcal{Q} \times \mathcal{A}} s(a'_i, a_i), \tag{3}$$

where $a'_i$ denotes the predicted answer for $q_i$, and $s(a'_i, a_i)$ measures the plausibility of $a'_i$ with respect to the ground-truth answer $a_i$. Based on this score, we can also induce a corresponding *loss landscape*.

### 2.2 Reasoning Trajectories as Parameter Updates

Recent studies (Brown et al., 2020; Wei et al., 2022; OpenAI, 2024b; DeepSeek-AI et al., 2025) show that incorporating intermediate reasoning steps substantially improves the ability of LLMs to solve complex reasoning tasks. Moreover, theoretical works (Dai et al., 2023; Bai et al., 2023; Giannou et al., 2023; Gatmiry et al., 2024; Fu et al., 2024; Huang et al., 2025) demonstrate that transformer-based models can learn iterative algorithms, such as multi-step gradient descent (GD), when equipped with CoT-style reasoning trajectories. However, these studies primarily focus on explicit numerical optimization problems, such as linear regression, and show that LLMs can implement optimization algorithms within their reasoning trajectories.

In contrast, we conceptualize *the reasoning trajectory of an LLM $\mathcal{M}$ as a multi-step gradient descent process over the model parameters $\theta$*. This process can be formally written as

$$\theta'_i \leftarrow \theta'_{i-1} + \Delta\mathcal{M}_{\theta'_{i-1}}(I, q, t_{\leq i}), \quad \theta'_0 = \theta, \quad 1 \leq i \leq |t|, \tag{4}$$

where $t$ denotes a reasoning trajectory, and $\Delta\mathcal{M}_{\theta'_{i-1}}(I, q, t_{\leq i}) = -\eta \nabla_{\theta'_{i-1}} \mathcal{L}_q(\theta'_{i-1})$ denotes the *pseudo-gradient update* associated with the partial trajectory $t_{\leq i}$. The final state $\theta'_{|t|}$ represents the parameters adapted to the instruction $I$, question $q$, and reasoning trajectory $t$.

In this view, each question $q_i$ defines a complex optimization task, and the LLM is optimized to produce its corresponding answer $a_i$. Before generating the final answer, the model follows an intermediate reasoning trajectory that functions as a parameter-update mechanism. The overall process is illustrated in Figure 1.

**Pseudo-Gradient Update.** Without loss of generality, we consider a simplified transformer (Vaswani et al., 2017) consisting of a single self-attention layer and a two-layer feed-forward network, while omitting normalization layers and other auxiliary components. Given $l = \{I, q\}$ as input, its activation can be expressed as

$$\boldsymbol{W}_2^T \left(\sigma\left(\boldsymbol{W}_1^T \left(\text{Softmax}\left(\boldsymbol{E}_{l,-1} \boldsymbol{W}_q \boldsymbol{W}_k^T \boldsymbol{E}_{l,:}^T\right) \boldsymbol{E}_{l,:} \boldsymbol{W}_v\right) + b_1\right)\right) + b_2, \tag{5}$$

where $\boldsymbol{E}_{l,:}$ denotes the input embeddings of the full sequence $l$, and $\boldsymbol{E}_{l,-1}$ denotes the embedding at the last position. The parameters are given by $\theta = \{\boldsymbol{W}_q, \boldsymbol{W}_k, \boldsymbol{W}_v, \boldsymbol{W}_1, \boldsymbol{W}_2, b_1, b_2\}$.

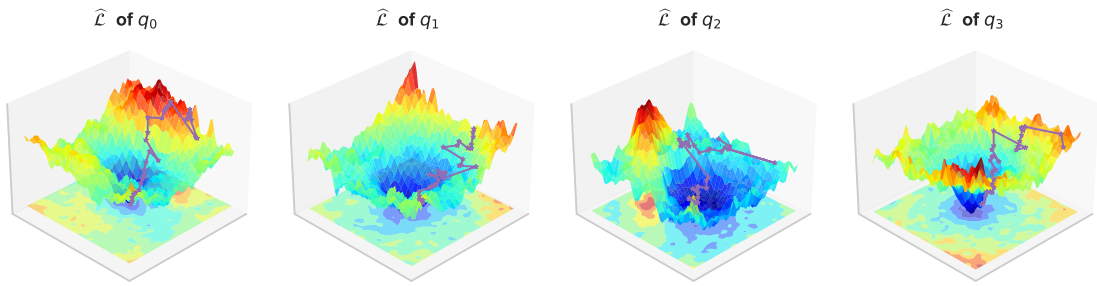


Figure 2: Landscape of the plausibility of LLMs generating accurate answers. We adopt the visualization method proposed by Li et al. (Li et al., 2018). The questions $q_0, q_1, q_2, q_3$ are selected from AIME24. We further project the pseudo-gradient trajectory onto the landscape (purple line). See § G.1 for more details.

Given a reasoning trajectory $t$, when the model attends to the first token $t^0$, the activation becomes

$$\boldsymbol{W}_2^T \left( \sigma \left( \boldsymbol{W}_1^T \left( \text{Softmax} \left( \boldsymbol{E}_{t,0} \boldsymbol{W}_q \boldsymbol{W}_k^T \begin{bmatrix} \boldsymbol{E}_{l,:} \\ \boldsymbol{E}_{t,0} \end{bmatrix}^T \right) \begin{bmatrix} \boldsymbol{E}_{l,:} \\ \boldsymbol{E}_{t,0} \end{bmatrix} \boldsymbol{W}_v \right) + b_1 \right) \right) + b_2. \tag{6}$$

**Proposition 2.1** (One-Step Pseudo-Gradient Update)**.** *There exists a parameter set $\theta'_t$, including $\{\boldsymbol{W}'_q, \boldsymbol{W}'_k, \boldsymbol{W}'_v, \boldsymbol{W}'_1, \boldsymbol{W}'_2, b'_1, b'_2\}$, such that the activation obtained by attending to the first reasoning token $t^0$ in Equation (6) can be represented as the activation of the original input sequence under the updated parameters:*

$$\boldsymbol{W}_2'^T \left( \sigma \left( \boldsymbol{W}_1'^T \left( \text{Softmax} \left( \boldsymbol{E}_{l,-1:} \boldsymbol{W}'_q \boldsymbol{W}_k'^T \boldsymbol{E}_{l,:}^T \right) \boldsymbol{E}_{l,:} \boldsymbol{W}'_v \right) + b'_1 \right) \right) + b'_2, \tag{7}$$

*up to an approximation error induced by representing $\boldsymbol{E}_{t,0}$ through the row span of $\boldsymbol{E}_{l,:}$. Here, $\theta'_t$ can be interpreted as a one-step pseudo update of $\theta$, and the corresponding increment $\Delta\mathcal{M}_\theta(I, q, t^0)$ depends only on $\theta$, $I$, $q$, and the attended reasoning token $t^0$.*

According to Proposition 2.1, whose proof is provided in § D, attending to a reasoning token can be interpreted as inducing an implicit update to the model parameters. As the model progressively attends to the full reasoning trajectory, these implicit updates accumulate, yielding what we refer to as a *pseudo-gradient update*.

**Empirical Evidence.** We provide empirical evidence for the pseudo-gradient update interpretation by using the model's confidence in the final answer as a probe. Specifically, at each position of a generated reasoning trajectory, we append "`Final Answer\n\boxed{...answer...}`" and compute the negative log-probability of the answer, denoted by $\widehat{\mathcal{L}}$. This procedure provides an alternative way to observe the optimization objective in Equation (3): as the model becomes more confident in the correct answer, $\widehat{\mathcal{L}}$ decreases. Figure 2 visualizes the corresponding negative-log-probability landscape. Additional examples and implementation details are provided in § I.1.

### 2.3 A Meta-Learning Perspective on LLM Reasoning

Building on the pseudo-gradient interpretation above, we introduce RAML as a meta-learning perspective for modeling both LLM reasoning and the training process that gives rise to it. We treat each question $q_i$ in the question set as an independent task. During reasoning-oriented training, such as supervised fine-tuning (Howard & Ruder, 2018; Radford et al., 2018; Devlin et al., 2019) or reinforcement learning (Mnih et al., 2013; Schulman et al., 2017; Ouyang et al., 2022; DeepSeek-AI et al., 2025), the LLM $\mathcal{M}_\theta$ is prompted to solve a question $q_i$ by following a reasoning trajectory $t$. The trajectory first induces one or more pseudo-gradient updates, adapting $\theta$ to $\theta'_t$. The model is then optimized through a pseudo second-order objective:

$$\min_\theta \sum_{q_i \in \mathcal{Q}} \sum_{t \in \mathcal{T}_i} \mathcal{L}_{q_i} \left( \mathcal{M}_{\theta'_t} \right) = \min_\theta \sum_{q_i \in \mathcal{Q}} \sum_{t \in \mathcal{T}_i} \mathcal{L}_{q_i} \left( \mathcal{M}_{\theta + \Delta\mathcal{M}_\theta(I,q,t)} \right), \tag{8}$$

where $\mathcal{T}_i$ denotes the set of reasoning trajectories associated with question $q_i$, and $\theta + \Delta\mathcal{M}_\theta(I, q, t)$ denotes the multi-step pseudo update defined in Equation (4).

Table 1: The evaluation performance of Qwen2.5-7B-Base trained using both SFT and Zero-GRPO training methods. In this context, "Qwen" refers to the abbreviation for Qwen2.5-Math-Instruct, and "Distil-Qwen" denotes DeepSeek-R1-Distill-Qwen-14B. **Green** cells indicate the best performance in each column, while **Blue** cells indicate the second-best performance.

| **Techniques** | **Source** | **AIME24** | | **MATH500-L5** | | **LiveMathBench-Hard** | |
|---|---|---|---|---|---|---|---|
| | | **Pass@8** ↑ | **mG-Pass@8** ↑ | **Pass@8** ↑ | **mG-Pass@8** ↑ | **Pass@8** ↑ | **mG-Pass@8** ↑ |
| SFT | Qwen | 20.34 | 7.43 | 58.42 | 35.65 | 26.77 | 7.43 |
| | Distill-Qwen | 36.69 | 10.29 | 82.98 | 45.79 | 25.15 | 10.46 |
| Zero-GRPO | - | 27.37 | 4.08 | 71.66 | 30.48 | 27.48 | 8.21 |

Table 2: Performance of Zero-GRPO model and GRPO model based on the SFT cold start.

| **Techniques** | **AIME24** | | **MATH500-L5** | | **LiveMathBench-Hard** | |
|---|---|---|---|---|---|---|
| | **Pass@8** ↑ | **mG-Pass@8** ↑ | **Pass@8** ↑ | **mG-Pass@8** ↑ | **Pass@8** ↑ | **mG-Pass@8** ↑ |
| Zero-GRPO | 27.37 | 4.08 | 71.66 | 30.48 | 27.48 | 8.21 |
| + SFT Cold Start | $35.87_{\uparrow 31\%}$ | $11.23_{\uparrow 175\%}$ | $82.42_{\uparrow 15\%}$ | $44.92_{\uparrow 47\%}$ | $42.17_{\uparrow 53\%}$ | $18.84_{\uparrow 129\%}$ |

Intuitively, RAML can be viewed as a variant of **Model-Agnostic Meta-Learning** (MAML; see § B) (Finn et al., 2017). The reasoning trajectory functions as the inner loop that adapts $\theta$, while the optimization of the final answer distribution serves as the outer loop, as summarized in § C. In this formulation, the latent *support set* is represented by the reasoning trajectory, and the answer corresponds to the *query set*. Unlike standard MAML, this inner loop does not require explicit loss evaluation or backpropagation; instead, the update is implicitly specified by the reasoning trajectory. Although the model weights are not physically modified during inference, the pseudo update enables the LLM to simulate question-specific adaptation within the reasoning process, thereby improving the accuracy and stability of its answers.

During training, the parameters $\theta$ are optimized so that, for each question $q_i$ and trajectory $t_j$, the induced adapted state $\theta'_{q_i,t_j}$ leads to accurate answer generation. The outer-loop optimization therefore follows the dependency $\mathcal{L}_{q_i} \rightarrow \theta'_{q_i,t_j} \rightarrow \theta$. In this sense, the LLM learns a robust meta-initialization that can be effectively adapted by reasoning trajectories, enabling better performance across diverse reasoning tasks.

Moreover, standard MAML typically requires a few-shot support set to adapt the model to a new task. In LLM reasoning, this support set is implicitly generated by the model itself in the form of reasoning trajectories. Therefore, the inner-loop process in RAML also resembles **Learning to Optimize** (L2O) (Andrychowicz et al., 2016; Li & Malik, 2017a;b), where a parameterized optimizer is trained to automate optimization across tasks. Specifically, reasoning-oriented training encourages the LLM to act as a meta-optimizer that generates question-specific inner-loop optimization paths.

## 3 Empirical Analysis on LLM Reasoning from Meta-Learning Perspective

Building on the meta-learning interpretation of LLM reasoning, this section investigates key factors that influence reasoning performance. Specifically, we instantiate several central questions in LLM reasoning within the proposed meta-learning framework and relate them to findings from the meta-learning literature. We focus on four questions:

❶ Which training strategy, SFT or RL, is more effective for LLM reasoning, and why (§ 3.2)?

❷ Why do longer reasoning trajectories improve reasoning performance (§ 3.3)?

❸ What optimization principles underlie the trade-off between reasoning cost and performance (§ 3.3)?

❹ Does trajectory-aided reasoning generalize across domains (§ 3.4)?

### 3.1 Experiment Setup

**Reasoning Task.** We primarily focus on mathematical reasoning, given its broad applicability and prominence in recent LLM reasoning research. We further include additional reasoning tasks in § 3.4 to evaluate generalization across domains.

**Training.** To minimize the confounding effects of prior post-training, we train Qwen2.5-7B-Base (Yang et al., 2024a) from scratch. We consider SFT as an off-policy training strategy and (**Zero-**)GRPO (Shao et al., 2024) as an on-policy training strategy. The training data are sourced from Open Reasoner Zero (Hu et al., 2025), which initially contains approximately 57k questions and is filtered to 39k questions; see § G.2 for details. Synthetic reasoning trajectories are generated by Qwen2.5-Math-72B-Instruct (Yang et al., 2024b) and DeepSeek-R1-Distill-Qwen-14B (DeepSeek-AI et al., 2025). Additional training details are provided in § G.2.

**Evaluation.** We evaluate mathematical reasoning on three benchmarks that are orthogonal to the training data: AIME24[1], MATH500 (Lightman et al., 2024) with Level-5 questions selected for greater discrimination, and LiveMathBench-Hard (Liu et al., 2024). To assess cross-domain generalization, we further evaluate on GPQA (Rein et al., 2023) and LiveCodeBench (Jain et al., 2024). Unless otherwise specified, model outputs are generated with temperature 1.0, top-$p$ = 0.8, top-$k$ = 50, and a maximum output length of 16,384 tokens. We report mG-Pass@$k$ (Liu et al., 2024) to measure stability and Pass@$k$ (Chen et al., 2021) to measure overall performance. Additional evaluation details are provided in § G.3.

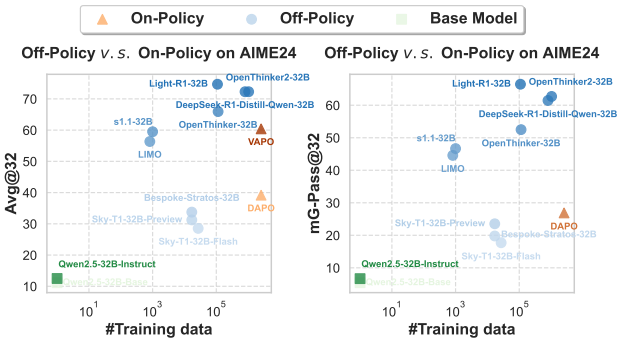


Figure 3: Performance of base models, models trained on off-policy data (SFT), and models trained on on-policy data (RL) using the AIME24 benchmark, with the $x$-axis representing the amount of training data. We generate 64 for each question and report Pass@32 and mG-Pass@32. The evaluation includes prominent models such as Sky-T1-32B (Team, 2025a), Bespoke-Stratos-32B (Labs, 2025), LIMO (Ye et al., 2025), s1.1-32B (Muennighoff et al., 2025), OpenThinker-32B (Team, 2025b), Light-R1-32B (Wen et al., 2025), DeepSeek-R1-Distill-Qwen-32B (DeepSeek-AI et al., 2025), DAPO-32B (Yu et al., 2025), and VAPO-32B (Yue et al., 2025). These models are based on either Qwen2.5-32B or Qwen2.5-32B-Instruct only through SFT or RL (**Zero-RL**). Since VAPO is not open source, we copy its results from the original paper..

### 3.2 Inner-Loop Optimization *v.s.* Reasoning Trajectory Source

Inner-loop optimization plays a central role in meta-learning, as its quality directly affects the stability and performance of the final model. In RAML, the LLM learns to act as an inner-loop optimizer by generating an optimization trajectory for each query. However, training learned optimizers is known to be challenging, often suffering from instability and poor generalization (Lan et al., 2024). In this section, we analyze SFT and GRPO from the perspective of inner-loop optimizer learning. Their key distinction lies in the source of reasoning trajectories used for training: ❶ *on-policy* trajectories, generated by the model currently being updated; and ❷ *off-policy* trajectories, generated by other LLMs or by an earlier version of the same model, such as through rejection sampling (Yuan et al., 2023; Singh et al., 2024).

**Status Quo of SFT *v.s.* RL.** Recent studies (Xiong et al., 2024; Chen et al., 2025a; Chu et al., 2025a) suggest that on-policy training is particularly effective for LLM reasoning. For example, the GRPO-trained DeepSeek-R1-Zero (DeepSeek-AI et al., 2025) substantially outperforms DeepSeek-V3, which is trained on large-scale off-policy synthetic data, on AIME24, achieving 71.0 versus 39.8. This result is often viewed as evidence for the advantage of on-policy strategies. However, as shown in Table 1 and the community-model

[1] https://huggingface.co/datasets/AI-MO/aimo-validation-aime

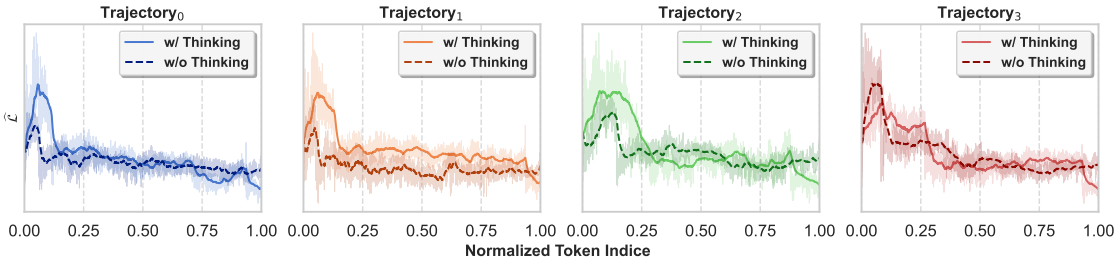


Figure 4: Illustration of QwQ's *pseudo-gradient update* for both thinking and non-thinking modes. We visualize four pairs of correct reasoning trajectories for one question in AIME24. Compared with thinking trajectories, no-thinking trajectories converge more quickly, which also easily falls into local optimal points.

comparison in Figure 3, GRPO-trained models do **not** consistently outperform SFT-trained models when using the same base LLM. This observation is also consistent with findings in prior work (DeepSeek-AI et al., 2025; Wen et al., 2025; Yang et al., 2025a).

**SFT Provides Stable Inner-Loop Optimization.** Learning to optimize often faces challenges such as training instability, divergence, and limited generalization. To improve stability, prior studies (Prémont-Schwarz et al., 2022; Thérien et al., 2024) suggest using optimal optimizers as "guardian" optimizers that guide learned optimizers toward convergence. From the perspective of RAML, SFT trains LLMs on reasoning trajectories produced by human annotators or stronger reasoning models. These trajectories can be viewed as guidance from an *oracle inner-loop optimizer*. As a result, SFT provides a stable and effective inner-loop optimization process, which can lead to strong reasoning performance. This does not imply that RL is inherently inferior: RL offers greater freedom to explore new optimization paths and, given sufficient model capacity and exploration, may have a higher potential upper bound.

**Combining SFT and RL for Stable Inner-Loop Optimization.** A natural strategy is to first use SFT to initialize a stable inner-loop optimizer and then apply RL to further explore improved optimization paths. As shown in Table 2, RL achieves substantial gains after SFT cold start, suggesting that SFT and RL play complementary roles: SFT stabilizes inner-loop optimization, while RL expands its exploration space.

**Takeaway for SFT *v.s.* RL**

❶ SFT provides stable inner-loop optimization by training on trajectories generated by an *oracle inner-loop optimizer*.

❷ Combining SFT with RL substantially improves performance by using SFT for stable initialization and RL for further exploration.

### 3.3 Inner-Loop Optimization Steps *v.s.* Reasoning Trajectory Tokens

In RAML, each token in a reasoning trajectory can be interpreted as an individual optimization step, while the trajectory length corresponds to the total number of inner-loop updates. We analyze this interpretation by combining empirical observations with related findings from LLM reasoning and meta-learning.

**Long Reasoning Trajectories Improve Performance.** As shown in Table 1, models trained with longer reasoning trajectories generally outperform those trained with shorter trajectories. This aligns with meta-learning findings that additional inner-loop updates can improve task adaptation. It is also consistent with the strong performance of long-CoT reasoning models such as DeepSeek-R1 (DeepSeek-AI et al., 2025), suggesting that longer trajectories provide more update steps and thereby strengthen LLM reasoning.

**Different Tokens Play Distinct Roles in Inner-Loop Optimization.** We next examine two notable phenomena in LLM reasoning through the optimization lens. First, advanced reasoning models often exhibit an ***aha moment*** (DeepSeek-AI et al., 2025), where specific reflection tokens trigger additional

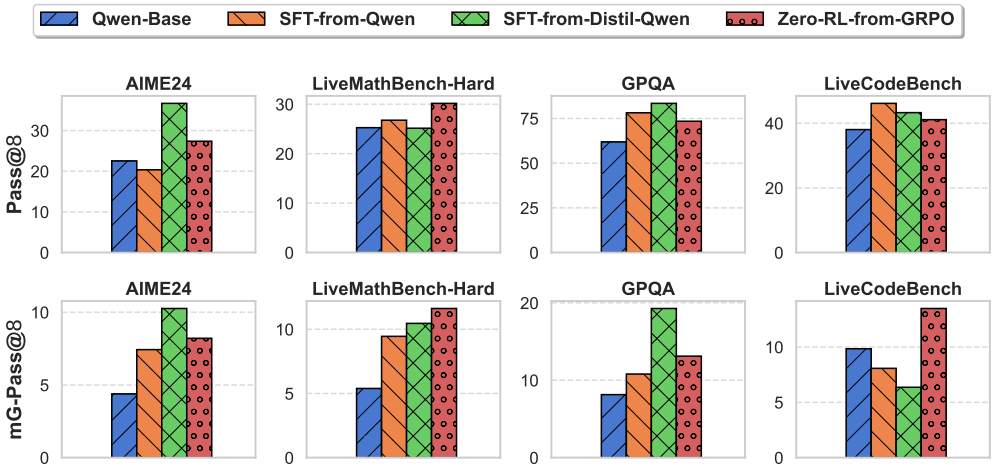


Figure 5: Evaluation results of base, SFT and GRPO models on AIME24, LiveMathBench-Hard, GPQA-Diamond, and LiveCodeBench.

deliberation. Such tokens are also commonly used for test-time scaling (Muennighoff et al., 2025; Ma et al., 2025). Following the setup in § 2.2, we measure the relative change in $\widehat{\mathcal{L}}$ before and after each token position. As shown in Figure 6, reflection tokens such as "Wait" and "Alternatively" induce sharp changes in the objective. From an optimization perspective, these tokens may help the model escape saddle points by introducing larger update directions and expanding the search space toward better parameter regions.

Second, recent work on reasoning efficiency (Qu et al., 2025; Yang et al., 2025b; Zhang et al., 2025; Ma et al., 2025; Yang et al., 2025a) explores how to balance decoding cost and reasoning performance, often by using special segments such as end-of-thinking delimiters. We hypothesize that such termination delimiters accelerate convergence at the optimization level, similar to the role of *momentum* in optimization, by guiding the model toward a flatter and more stable region. However, faster convergence does not necessarily lead to a globally optimal point. Following the same setup, we append the end-of-thinking delimiter `Therefore, after all this, I believe the answer is` after `<think>`. As shown in Figure 4, trajectories using the end-of-thinking delimiter converge more quickly, providing partial support for this hypothesis. Since QwQ is not fully adapted to the no-thinking mode, we provide additional experiments on Qwen3 in § I.1, which further support our conclusions.

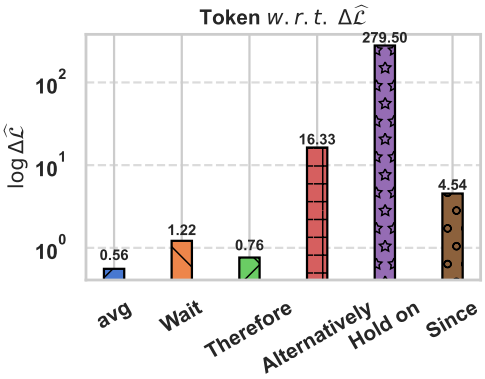


Figure 6: Illustration of the influence of reflection tokens. Reflection tokens lead to sharper objective change.

**Takeaway for Reasoning Trajectory Tokens**

❶ Long reasoning trajectories are analogous to additional inner-loop optimization steps, which improve adaptation and enhance LLM reasoning performance.

❷ Different tokens play distinct roles in the inner-loop optimization process: reflection tokens promote exploration, while special tokens that regulate reasoning length encourage faster-converging updates.

### 3.4 Task Generalization *v.s.* Reasoning Generalization

Meta-learning models often generalize well across tasks drawn from related distributions, as they learn transferable representations and adaptation strategies. We examine whether this property also holds for LLM reasoning, where each question can be viewed as a distinct task but many questions share common reasoning skills. We evaluate generalization from two perspectives: within-domain generalization, where training and evaluation involve the same reasoning domain, and cross-domain generalization, where evaluation requires different reasoning skills.

The training data from Open Reasoner Zero (§ 3.1) primarily consist of mathematical problems from MATH, making AIME24 and LiveMathBench-Hard suitable benchmarks for within-domain evaluation. As shown in Figure 5, both SFT and GRPO improve performance on these benchmarks. For cross-domain evaluation, we test the trained models on GPQA for scientific reasoning and LiveCodeBench for code reasoning. The results in Figure 5 show that all trained models outperform the base model on both benchmarks, indicating that trajectory-based training can transfer beyond the original mathematical domain.

These findings align with prior observations that code-trained models can acquire strong logical reasoning abilities (Chen et al., 2021), and that training on mathematical and code corpora can mutually improve performance (Wang et al., 2024; Hui et al., 2024; Guo et al., 2024). From the perspective of RAML, reasoning trajectories help LLMs learn shared reasoning structures that support efficient adaptation to new tasks and domains.

**Takeaway for Reasoning Generalization**

Training LLMs with reasoning trajectories encourages the acquisition of shared reasoning patterns across diverse problems. Analogous to meta-learning, this enables LLMs to adapt more effectively through new trajectories and generalize to out-of-distribution questions.

## 4 Advancing LLM Reasoning with Meta-Learning Insights

In this section, we demonstrate how insights from meta-learning can guide the improvement of LLM reasoning. We introduce two **simple yet effective** strategies inspired by meta-learning: increasing the number of training trajectories per question to improve reasoning performance, and encouraging more efficient reasoning trajectories to reduce decoding cost. We validate both strategies through empirical studies.

### 4.1 Increasing Training Reasoning Trajectories per Question

Prior studies (Agarwal et al., 2021; Chen et al., 2020a) show that support-set size is critical for improving performance, stability, and convergence in meta-learning. In RAML, the support set corresponds to the number of reasoning trajectories available for each question. This naturally raises the question: *Can increasing the support-set size improve LLM reasoning training?* We conduct a preliminary study to examine how the number of training trajectories per question affects reasoning performance.

**SFT.** We train Qwen2.5-7B-Base with SFT using $\{1, 2, 4, 8, 16, 32\}$ synthetic reasoning trajectories per question, while keeping the training frequency of each question fixed. As shown in Figure 7, increasing the number of trajectories consistently improves both performance and reasoning stability across benchmarks. This suggests that additional trajectories provide a richer support set and improve supervised reasoning training (Yang et al., 2024b).

**GRPO.** For GRPO, the support-set size corresponds to the number of rollout trajectories sampled for each prompt. To ensure stable advantage estimation and fair comparison, we fix the rollout group size to 16 and compute the advantage for each trajectory. During gradient updates, we randomly select $n \in \{1, 2, 4, 8, 16\}$ trajectories to compute the gradient. As shown in Figure 8, multiple trajectories per question substantially improve both performance and stability, and larger trajectory sets accelerate convergence. These results help explain the strong performance and stability of GRPO-style reasoning models (Liu et al., 2024): compared

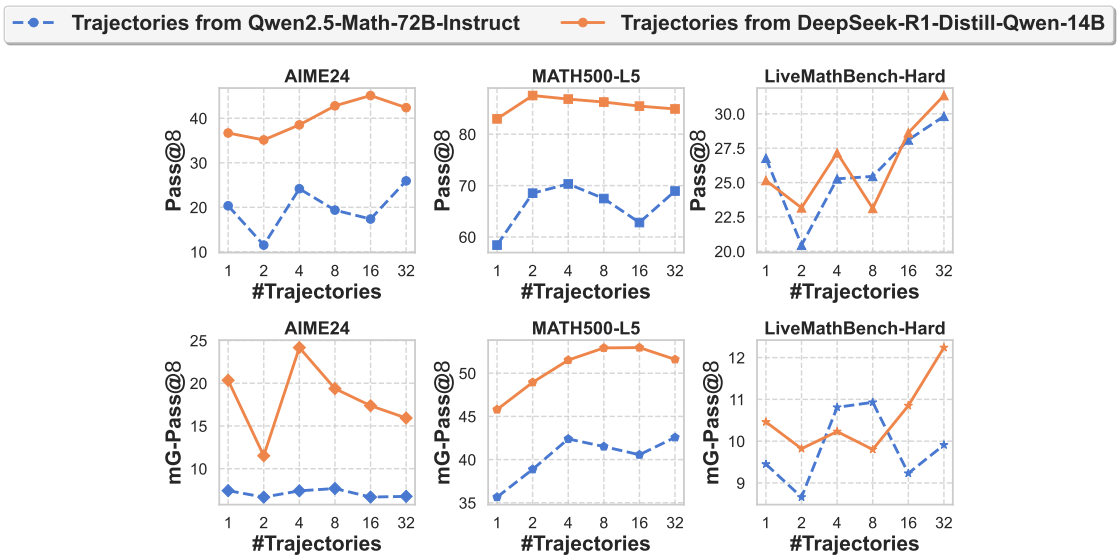


Figure 7: Evaluation results *w.r.t.* different number of reasoning trajectories of SFT models on AIME24, MATH500-L5, and LiveMathBench-Hard.

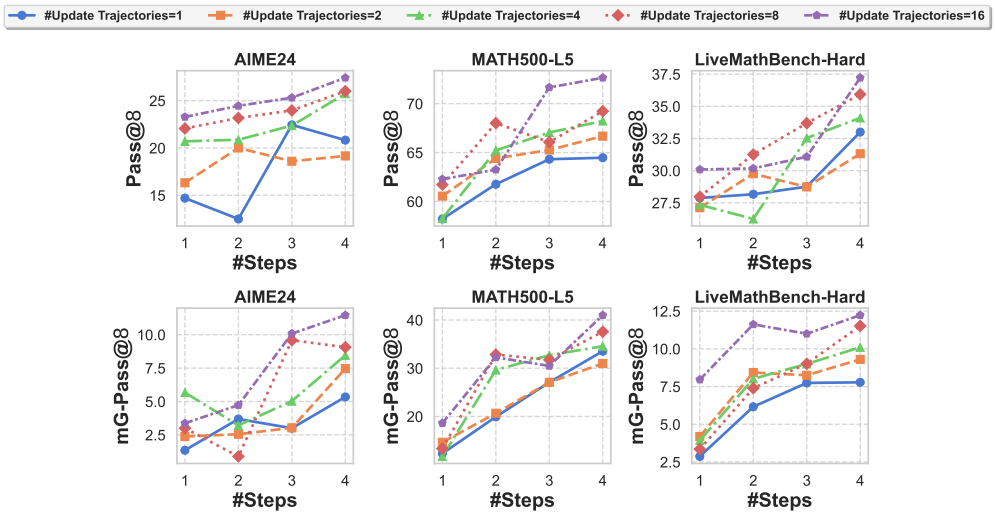


Figure 8: Evaluation results *w.r.t.* different number of reasoning trajectories of GRPO models on AIME24, MATH500-L5, and LiveMathBench-Hard.

with PPO (Schulman et al., 2017) and naive SFT, GRPO naturally optimizes over multiple trajectories per question. Notably, recent studies have also incorporated group sampling into PPO-like methods (Hu et al., 2025; Yue et al., 2025), achieving competitive performance.

### 4.2 Incentivizing Reasoning Efficiency through an Optimization Lens

Recent advanced reasoning models are often limited by inefficient and excessively long reasoning trajectories. Although several studies (Ma et al., 2025; Yang et al., 2025a) attempt to reduce decoding tokens and lower inference cost, these methods can degrade reasoning performance. This raises a fundamental question: *Can we reduce reasoning tokens without sacrificing performance?* Since each reasoning trajectory corresponds to an inner-loop optimization path, the question can be reframed as: *Does there exist a more efficient inner-loop optimization path?*

From an optimization perspective, such a path should exist, as illustrated in Figure 9. We conduct a simple yet informative experiment to provide evidence for this hypothesis. Specifically, we use Qwen3-32B (Yang et al., 2025a) to generate 16 reasoning trajectories for each question in AIME24, MATH500-L5, and LiveMathBench-Hard. These trajectories serve as initial optimization paths, and our goal is to refine them into shorter yet effective alternatives.

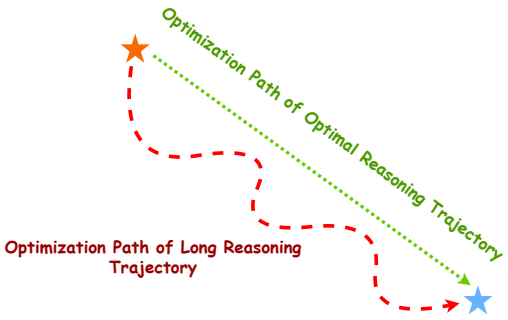


Figure 9: Given a long reasoning trajectory, there exists an optimal corresponding reasoning trajectory which leverages fewer tokens.

We adopt a heuristic condensation strategy: an LLM summarizes the original reasoning trajectories into shorter variants, which are then used to prompt Qwen3-32B for answer generation. The summaries are generated by Qwen2.5-32B-Instruct, and the final answers are deliberately excluded from the summarized trajectories. To reduce randomness, we use four independent summary generators.

As shown in Figure 10, Qwen3-32B with summarized reasoning trajectories achieves performance comparable to its full thinking mode, especially under Pass@16, while substantially reducing the number of reasoning tokens. Moreover, summarized trajectories outperform the no-thinking mode, even though the latter uses more tokens. These results suggest that long-CoT LLMs can follow more efficient reasoning trajectories that require fewer tokens while preserving comparable reasoning performance. Our simple summarization-based approach provides an initial approximation of such trajectories, leaving more advanced methods for future work.

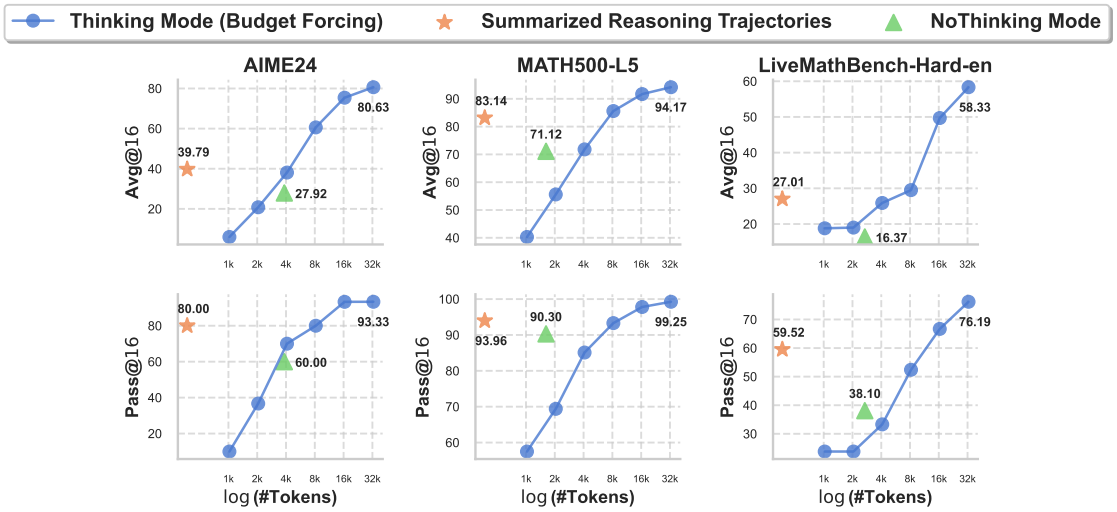


Figure 10: Experimental results of Qwen3-32B with 1) thinking mode, 2) summarized trajectory, and 3) nothinking mode on AIME24, MATH500-L5, and LiveMathBench-Hard-en.

## 5 Related Work

**LLM Reasoning.** The reasoning capabilities of large language models (LLMs) have progressively advanced through the development of several key technologies, which have substantially enhanced their performance on complex tasks. In-Context Learning (Brown et al., 2020; Rubin et al., 2022; Min et al., 2022b; Dong et al., 2024; Bertsch et al., 2024) enables models to perform tasks by interpreting examples provided in prompts without requiring additional training. However, this approach relies heavily on the model's pre-trained knowledge and careful prompt design, limiting its effectiveness for complex reasoning tasks (Min et al., 2022a). The introduction of Chain of Thought (CoT) (Wei et al., 2022; Yao et al., 2023; Besta et al.,

2024) prompting has significantly improved LLM performance in areas such as mathematical reasoning, commonsense reasoning, and symbolic reasoning by guiding the models to produce intermediate reasoning steps. Supervised Fine-Tuning (SFT) further refines the reasoning capabilities of LLMs by training them with labeled datasets tailored to specific tasks (Beeching et al., 2024; Team, 2025b;a; Ye et al., 2025). Reinforcement Learning (RL), through the use of reward mechanisms, has become a critical approach to optimizing model behavior and enhancing reasoning abilities. Recently, Long-Chain of Thought (Long-CoT) models have emerged as a notable trend in reasoning research, generating detailed reasoning steps to better address complex tasks (OpenAI, 2024b; DeepSeek-AI et al., 2025; Team et al., 2025; Team, 2025c; Deepmind, 2025).

**Understanding LLMs.** The remarkable success of LLMs has spurred extensive research into their capabilities. Early studies (Yun et al., 2020a;b) explored function approximation, demonstrating that sufficiently large transformers (Vaswani et al., 2017) can universally approximate any continuous sequence-to-sequence mapping on a compact domain. Subsequent research investigated the computational power of transformers (Dehghani et al., 2019; Bhattamishra et al., 2020; Yao et al., 2021b; Hewitt et al., 2020; Weiss et al., 2021; Merrill et al., 2022; Chiang et al., 2023; Giannou et al., 2023; Liu et al., 2023). For example, Feng et al. (2023) validated the necessity of chain-of-thought (CoT) prompting for solving complex problems using circuit complexity theory. Other works (Gatmiry et al., 2024; Huang et al., 2025) demonstrate that transformers can learn to implement learning algorithms, such as gradient descent, within trajectories. Several studies (Xie et al., 2022; Akyürek et al., 2023; Dai et al., 2023; Bai et al., 2023; Olsson et al., 2022; Gatmiry et al., 2024) have focused on understanding in-context learning (ICL) (Brown et al., 2020; Dong et al., 2024), examining the role of demonstration examples.

**Meta-Learning.** Meta-learning, commonly referred to as "learning to learn", aims to enable models to enhance their learning strategies by leveraging prior experience across multiple tasks. Early research in this area (Bengio et al., 1992; Thrun & Pratt, 1998) explored methods for acquiring learning rules applicable to new tasks, with a particular emphasis on lifelong learning. These foundational efforts established the basis for creating more adaptable and flexible learning algorithms, paving the way for subsequent advancements. Recent meta-learning approaches can generally be categorized into three groups: 1) metric-based methods, which focus on learning a feature space to efficiently compare samples (Snell et al., 2017; Chen et al., 2020b; Tang et al., 2020; Zhang et al., 2023); 2) model-based methods, which utilize memory mechanisms or other structures to store and retrieve task-specific information (Weston et al., 2015; Sukhbaatar et al., 2015; Santoro et al., 2016); and 3) optimization-based methods, which refine the learning process to facilitate rapid adaptation (Finn et al., 2017; Rajeswaran et al., 2019; Ye et al., 2021).

For additional discussions on related work, please refer to § F.

## 6 Conclusion

This paper presents a novel perspective on LLM reasoning by interpreting it through the lens of meta-learning. Through theoretical analysis and empirical validation, we show that reasoning trajectories can be conceptualized as pseudo-gradient updates, offering a deeper understanding of how LLMs perform complex reasoning. Extensive experiments further reveal a strong correspondence between meta-learning and LLM reasoning, suggesting promising directions for advancing LLM reasoning with meta-learning principles.

## References

Mayank Agarwal, Mikhail Yurochkin, and Yuekai Sun. On sensitivity of meta-learning to support data. In *NeurIPS*, pp. 20447–20460, 2021. 1, 4.1

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *ACL*, pp. 12248–12267. Association for Computational Linguistics, 2024. E

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *ICLR*. OpenReview.net, 2023. 5

Marcin Andrychowicz, Misha Denil, Sergio Gomez Colmenarejo, Matthew W. Hoffman, David Pfau, Tom Schaul, and Nando de Freitas. Learning to learn by gradient descent by gradient descent. In *NIPS*, pp. 3981–3989, 2016. 1, 2, 2.3

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. In *NeurIPS*, 2023. 2.2, 5

Edward Beeching, Shengyi Costa Huang, Albert Jiang, Jia Li, Benjamin Lipkin, Zihan Qina, Kashif Rasul, Ziju Shen, Roman Soletskyi, and Lewis Tunstall. Numinamath 72b cot. https://huggingface.co/AI-MO/NuminaMath-72B-CoT, 2024. 5

Samy Bengio, Yoshua Bengio, Jocelyn Cloutier, and Jan Gecsei. On the optimization of a synaptic learning rule. In *Optimality in Artificial and Biological Neural Networks*, pp. 6–8, 1992. 5

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, Ido Shahaf, Oren Tropp, Ehud Karpas, Ran Zilberstein, Jiaqi Zeng, Soumye Singhal, Alexander Bukharin, Yian Zhang, Tugrul Konuk, Gerald Shen, Ameya Sunil Mahabaleshwarkar, Bilal Kartal, Yoshi Suhara, Olivier Delalleau, Zijia Chen, Zhilin Wang, David Mosallanezhad, Adi Renduchintala, Haifeng Qian, Dima Rekesh, Fei Jia, Somshubra Majumdar, Vahid Noroozi, Wasi Uddin Ahmad, Sean Narenthiran, Aleksander Ficek, Mehrzad Samadi, Jocelyn Huang, Siddhartha Jain, Igor Gitman, Ivan Moshkov, Wei Du, Shubham Toshniwal, George Armstrong, Branislav Kisacanin, Matvei Novikov, Daria Gitman, Evelina Bakhturina, Jane Polak Scowcroft, John Kamalu, Dan Su, Kezhi Kong, Markus Kliegl, Rabeeh Karimi, Ying Lin, Sanjeev Satheesh, Jupinder Parmar, Pritam Gundecha, Brandon Norick, Joseph Jennings, Shrimai Prabhumoye, Syeda Nahida Akter, Mostofa Patwary, Abhinav Khattar, Deepak Narayanan, Roger Waleffe, Jimmy Zhang, Bor-Yiing Su, Guyue Huang, Terry Kong, Parth Chadha, Sahil Jain, Christine Harvey, Elad Segal, Jining Huang, Sergey Kashirsky, Robert McQueen, Izzy Putterman, George Lam, Arun Venkatesan, Sherry Wu, Vinh Nguyen, Manoj Kilaru, Andrew Wang, Anna Warno, Abhilash Somasamudramath, Sandip Bhaskar, Maka Dong, Nave Assaf, Shahar Mor, Omer Ullman Argov, Scot Junkin, Oleksandr Romanenko, Pedro Larroy, Monika Katariya, Marco Rovinelli, Viji Balas, Nicholas Edelman, Anahita Bhiwandiwalla, Muthu Subramaniam, Smita Ithape, Karthik Ramamoorthy, Yuting Wu, Suguna Varshini Velury, Omri Almog, Joyjit Daw, Denys Fridman, Erick Galinkin, Michael Evans, Katherine Luna, Leon Derczynski, Nikki Pope, Eileen Long, Seth Schneider, Guillermo Siman, Tomasz Grzegorzek, Pablo Ribalta, Monika Katariya, Joey Conway, Trisha Saar, Ann Guan, Krzysztof Pawelec, Shyamala Prayaga, Oleksii Kuchaiev, Boris Ginsburg, Oluwatobi Olabiyi, Kari Briski, Jonathan Cohen, Bryan Catanzaro, Jonah Alben, Yonatan Geifman, Eric Chung, and Chris Alexiuk. Llama-Nemotron: Efficient reasoning models. *CoRR*, abs/2505.00949, 2025. H

Amanda Bertsch, Maor Ivgi, Uri Alon, Jonathan Berant, Matthew R. Gormley, and Graham Neubig. In-context learning with long-context models: An in-depth exploration. *CoRR*, abs/2405.00200, 2024. 5

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models. In *AAAI*, pp. 17682–17690. AAAI Press, 2024. 5

Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. On the ability and limitations of transformers to recognize formal languages. In *EMNLP*, pp. 7096–7116. Association for Computational Linguistics, 2020. 5

Arne Bjerhammar. Application of calculus of matrices to method of least squares: with special reference to geodetic calculations. *Trans. Roy. Inst. Tech. Stockholm.*, 1951. D

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020. 2.2, 5

Hardy Chen, Haoqin Tu, Fali Wang, Hui Liu, Xianfeng Tang, Xinya Du, Yuyin Zhou, and Cihang Xie. Sft or rl? an early investigation into training r1-like reasoning large vision-language models. *CoRR*, abs/2504.11468, 2025a. 3.2

Jiaxin Chen, Xiao-Ming Wu, Yanke Li, Qimai Li, Li-Ming Zhan, and Fu-Lai Chung. A closer look at the training strategy for modern meta-learning. In *NeurIPS*, 2020a. 4.1

Jiaxin Chen, Li-Ming Zhan, Xiao-Ming Wu, and Fu-Lai Chung. Variational metric scaling for metric-based meta-learning. In *AAAI*, pp. 3478–3485. AAAI Press, 2020b. 5

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. 3.1, 3.4

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *CoRR*, abs/2503.09567, 2025b. 1

David Chiang, Peter Cholak, and Anand Pillay. Tighter bounds on the expressivity of transformer encoders. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pp. 5544–5562. PMLR, 2023. 1, 5

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. SFT memorizes, RL generalizes: A comparative study of foundation model post-training. *CoRR*, abs/2501.17161, 2025a. 3.2

Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. Gpg: A simple and strong reinforcement learning baseline for model reasoning. *CoRR*, abs/2504.02546, 2025b. H

Liam Collins, Aryan Mokhtari, and Sanjay Shakkottai. Task-robust model-agnostic meta-learning. In *NeurIPS*, 2020. J

Liam Collins, Aryan Mokhtari, and Sanjay Shakkottai. How does the task landscape affect MAML performance? In *CoLLAs*, volume 199 of *Proceedings of Machine Learning Research*, pp. 23–59. PMLR, 2022. 1

George Cybenko. Approximations by superpositions of a sigmoidal function. *MCSS*, 2:183–192, 1989. D

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. Why can gpt learn in-context? language models secretly perform gradient descent as meta-optimizers. In *ACL (Findings)*, pp. 4005–4019. Association for Computational Linguistics, 2023. 2.2, 5, F

Google Deepmind. Gemini 2.5: Our most intelligent ai model. https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/#gemini-2-5-thinking/, 2025. Accessed: 2025-03-26. 1, 5

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, and Wangding Zeng. Deepseek-v3 technical report. *CoRR*, abs/2412.19437, 2024. 1

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, and S. S. Li. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, 2025. 1, 2.2, 2.3, 3, 3.1, 3.2, 3.3, 5, G.2

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *ICLR*. OpenReview.net, 2019. 5

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pp. 4171–4186. Association for Computational Linguistics, 2019. 2.3, E

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. A survey on in-context learning. In *EMNLP*, pp. 1107–1128. Association for Computational Linguistics, 2024. 5

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan

Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. 1

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: A theoretical perspective. In *NeurIPS*, 2023. 1, 5

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135. PMLR, 2017. 1, 2, 2.3, 5

Deqing Fu, Tian-Qi Chen, Robin Jia, and Vatsal Sharan. Transformers learn to achieve second-order convergence rates for in-context linear regression. In *NeurIPS*, 2024. 2.2

Khashayar Gatmiry, Nikunj Saunshi, Sashank J. Reddi, Stefanie Jegelka, and Sanjiv Kumar. Can looped transformers learn to implement multi-step gradient descent for in-context learning? In *ICML*. OpenReview.net, 2024. 1, 2.2, 5, F

Angeliki Giannou, Shashank Rajput, Jy-yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pp. 11398–11442. PMLR, 2023. 1, 2.2, 5

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. Deepseek-coder: When the large language model meets programming - the rise of code intelligence. *CoRR*, abs/2401.14196, 2024. 3.4

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS Datasets and Benchmarks*, 2021. G.3

John Hewitt, Michael Hahn, Surya Ganguli, Percy Liang, and Christopher D. Manning. Rnns can generate bounded hierarchical languages with optimal memory. In *EMNLP*, pp. 1978–2010. Association for Computational Linguistics, 2020. 5

Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991. D

Timothy M. Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J. Storkey. Meta-learning in neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(9):5149–5169, 2022. 1, 2

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *ACL*, pp. 328–339. Association for Computational Linguistics, 2018. 2.3, E

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *CoRR*, abs/2503.24290, 2025. 3.1, 4.1, H

Jianhao Huang, Zixuan Wang, and Jason D. Lee. Transformers learn to implement multi-step gradient descent with chain of thought. *CoRR*, abs/2502.21212, 2025. 1, 2.2, 5, F

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Kai Dang, An Yang, Rui Men, Fei Huang, Xingzhang Ren, Xuancheng Ren, Jingren Zhou, and Junyang Lin. Qwen2.5-coder technical report. *CoRR*, abs/2409.12186, 2024. 3.4

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. *CoRR*, abs/2403.07974, 2024. 3.1, G.3

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. How can we know what language models know. *Trans. Assoc. Comput. Linguistics*, 8:423–438, 2020. 1

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *SOSP*, pp. 611–626. ACM, 2023. G.2

Bespoke Labs. Bespoke-stratos: The unreasonable effectiveness of reasoning distillation, 2025. URL www.bespokelabs.ai/blog/bespoke-stratos-the-unreasonable-effectiveness-of-reasoning-distillation. Accessed: 2025-01-22. 3

Qingfeng Lan, A. Rupam Mahmood, Shuicheng Yan, and Zhongwen Xu. Learning to optimize for reinforcement learning. *RLJ*, 2:481–497, 2024. 3.2

Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *CVPR*, pp. 10657–10665. Computer Vision Foundation / IEEE, 2019. 1

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *NeurIPS*, pp. 6391–6401, 2018. 2, 4

Ke Li and Jitendra Malik. Learning to optimize. In *ICLR (Poster)*. OpenReview.net, 2017a. 2.3

Ke Li and Jitendra Malik. Learning to optimize neural nets. *CoRR*, abs/1703.00441, 2017b. 2.3

Xinzhe Li. A survey on LLM test-time compute via search: Tasks, LLM profiling, search algorithms, and relevant frameworks. *CoRR*, abs/2501.10069, 2025. 1

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *ICLR*. OpenReview.net, 2024. 3.1, G.3

Bingbin Liu, Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. In *ICLR*. OpenReview.net, 2023. 1, 5

Chenghao Liu, Zhihao Wang, Doyen Sahoo, Yuan Fang, Kun Zhang, and Steven C. H. Hoi. Adaptive task sampling for meta-learning. In *ECCV*, volume 12363 of *Lecture Notes in Computer Science*, pp. 752–769. Springer, 2020. 1, J

Junnan Liu, Hongwei Liu, Linchen Xiao, Ziyi Wang, Kuikun Liu, Songyang Gao, Wenwei Zhang, Songyang Zhang, and Kai Chen. Are your llms capable of stable reasoning? *CoRR*, abs/2412.13147, 2024. 3.1, 4.1, G.3

Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. Reasoning models can be effective without thinking. *CoRR*, abs/2504.09858, 2025. 3.3, 3.3, 4.2

William Merrill, Ashish Sabharwal, and Noah A. Smith. Saturated transformers are constant-depth threshold circuits. *Trans. Assoc. Comput. Linguistics*, 10:843–856, 2022. 1, 5

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. Metaicl: Learning to learn in context. In *NAACL-HLT*, pp. 2791–2809. Association for Computational Linguistics, 2022a. 5

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 *EMNLP*, pp. 11048–11064. Association for Computational Linguistics, 2022b. 5

Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, Wayne Xin Zhao, Zheng Liu, Zhongyuan Wang, and Ji-Rong Wen. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems. *CoRR*, abs/2412.09413, 2024. 1

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013. 2.3

Eliakim H Moore. On the reciprocal of the general algebraic matrix. *Bull. Am. Math. Soc.*, 26:294–295, 1920. D

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel J. Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *CoRR*, abs/2501.19393, 2025. 3, 3.3

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads. *CoRR*, abs/2209.11895, 2022. 5

OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. 1

OpenAI. Introducing openai o3 and o4-mini. https://openai.com/index/introducing-o3-and-o4-mini/, 2024a. Accessed: 2024-12. 1

OpenAI. Learning to reason with llms. https://openai.com/index/learning-to-reason-with-llms/, 2024b. Accessed: 2024-09. 1, 2.2, 5

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022. 2.3, E

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pp. 8024–8035, 2019. G.2

Roger Penrose. A generalized inverse for matrices. In *Math. Proc. Cambridge Philos. Soc.*, volume 51, pp. 406–413. Cambridge University Press, 1955. D

Isabeau Prémont-Schwarz, Jaroslav Vitku, and Jan Feyereisl. A simple guard for learned optimizers. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pp. 17910–17925. PMLR, 2022. 3.2

Yiwei Qin, Xuefeng Li, Haoyang Zou, Yixiu Liu, Shijie Xia, Zhen Huang, Yixin Ye, Weizhe Yuan, Hector Liu, Yuanzhi Li, and Pengfei Liu. O1 replication journey: A strategic progress report - part 1. *CoRR*, abs/2410.18982, 2024. 1

Xiaoye Qu, Yafu Li, Zhaochen Su, Weigao Sun, Jianhao Yan, Dongrui Liu, Ganqu Cui, Daizong Liu, Shuxian Liang, Junxian He, Peng Li, Wei Wei, Jing Shao, Chaochao Lu, Yue Zhang, Xian-Sheng Hua, Bowen Zhou, and Yu Cheng. A survey of efficient reasoning for large reasoning models: Language, multimodality, and beyond. *CoRR*, abs/2503.21614, 2025. 3.3

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. *OpenAI*, 2018. 2.1, 2.3, 4

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023. 4, E

Aravind Rajeswaran, Chelsea Finn, Sham M. Kakade, and Sergey Levine. Meta-learning with implicit gradients. In *NeurIPS*, pp. 113–124, 2019. 5

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*. OpenReview.net, 2017. 1, 2

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. *CoRR*, abs/2311.12022, 2023. 3.1, G.3

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. Learning to retrieve prompts for in-context learning. In *NAACL-HLT*, pp. 2655–2671. Association for Computational Linguistics, 2022. 5

Adam Santoro, Sergey Bartunov, Matthew M. Botvinick, Daan Wierstra, and Timothy P. Lillicrap. One-shot learning with memory-augmented neural networks. *CoRR*, abs/1605.06065, 2016. 5

Jürgen Schmidhuber. Evolutionary principles in self-referential learning, or on learning how to learn: The meta-meta-. hook. Master's thesis, Technical University of Munich, Germany, 1987. 1, 2

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. 2.3, 4.1, 4, E

Darsh J Shah, Peter Rushton, Somanshu Singla, Mohit Parmar, Kurt Smith, Yash Vanjani, Ashish Vaswani, Adarsh Chaluvaraju, Andrew Hojel, Andrew Ma, et al. Rethinking reflection in pre-training. *CoRR*, abs/2504.04022, 2025. 1

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. 3.1, E

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient RLHF framework. In *EuroSys*, pp. 1279–1297. ACM, 2025. G.2

Wei Shi, Sihang Li, Tao Liang, Mingyang Wan, Gojun Ma, Xiang Wang, and Xiangnan He. Route sparse autoencoder to interpret large language models. *CoRR*, abs/2503.08200, 2025. 1

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron T. Parisi, Abhishek Kumar, Alexander A. Alemi, Alex Rizkowsky, Azade Nova, Ben Adlam, Bernd Bohnet, Gamaleldin Fathy Elsayed, Hanie Sedghi, Igor Mordatch, Isabelle Simpson, Izzeddin Gur, Jasper Snoek, Jeffrey Pennington, Jiri Hron, Kathleen Kenealy, Kevin Swersky, Kshiteej Mahajan, Laura Culp, Lechao Xiao, Maxwell L. Bileschi, Noah Constant, Roman Novak, Rosanne Liu, Tris Warkentin, Yundi Qian, Yamini Bansal, Ethan Dyer, Behnam Neyshabur, Jascha Sohl-Dickstein, and Noah Fiedel. Beyond human data: Scaling self-training for problem-solving with language models. *Trans. Mach. Learn. Res.*, 2024, 2024. 3.2

Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *NIPS*, pp. 4077–4087, 2017. 5

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *NIPS*, pp. 2440–2448, 2015. 5

Gokul Swamy, Sanjiban Choudhury, Wen Sun, Zhiwei Steven Wu, and J. Andrew Bagnell. All roads lead to likelihood: The value of reinforcement learning in fine-tuning. *CoRR*, abs/2503.01067, 2025. E

Hao Tang, Zechao Li, Zhimao Peng, and Jinhui Tang. Blockmix: Meta regularization and self-calibrated inference for metric-based meta-learning. In *ACM Multimedia*, pp. 610–618. ACM, 2020. 5

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, Haiqing Guo, Han Zhu, Hao Ding, Hao Hu, Hao Yang, Hao Zhang, Haotian Yao, Haotian Zhao, Haoyu Lu, Haoze Li, Haozhen Yu, Hongcheng Gao,

Huabin Zheng, Huan Yuan, Jia Chen, Jianhang Guo, Jianlin Su, Jianzhou Wang, Jie Zhao, Jin Zhang, Jingyuan Liu, Junjie Yan, Junyan Wu, Lidong Shi, Ling Ye, Longhui Yu, Mengnan Dong, Neo Zhang, Ningchen Ma, Qiwei Pan, Qucheng Gong, Shaowei Liu, Shengling Ma, Shupeng Wei, Sihan Cao, Siying Huang, Tao Jiang, Weihao Gao, Weimin Xiong, Weiran He, Weixiao Huang, Wenhao Wu, Wenyang He, Xianghui Wei, Xianqing Jia, Xingzhe Wu, Xinran Xu, Xinxing Zu, Xinyu Zhou, Xuehai Pan, Y. Charles, Yang Li, Yangyang Hu, Yangyang Liu, Yanru Chen, Yejie Wang, Yibo Liu, Yidao Qin, Yifeng Liu, Ying Yang, Yiping Bao, Yulun Du, Yuxin Wu, Yuzhi Wang, Zaida Zhou, Zhaoji Wang, Zhaowei Li, Zhen Zhu, Zheng Zhang, Zhexu Wang, Zhilin Yang, Zhiqi Huang, Zihao Huang, Ziyao Xu, and Zonghan Yang. Kimi k1.5: Scaling reinforcement learning with llms. *CoRR*, abs/2501.12599, 2025. 1, 5

NovaSky Team. Sky-t1: Train your own o1 preview model within $450. https://novasky-ai.github.io/posts/sky-t1, 2025a. Accessed: 2025-01-09. 3, 5

OpenThoughts Team. Open Thoughts. https://open-thoughts.ai, 2025b. 3, 5, H

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning. https://qwenlm.github.io/blog/qwq-32b/, March 2025c. 5, G.1

Fengwei Teng, Zhaoyang Yu, Quan Shi, Jiayi Zhang, Chenglin Wu, and Yuyu Luo. Atom of thoughts for markov LLM test-time scaling. *CoRR*, abs/2502.12018, 2025. 1

Benjamin Thérien, Charles-Étienne Joseph, Boris Knyazev, Edouard Oyallon, Irina Rish, and Eugene Belilovsky. $\mu$lo: Compute-efficient meta-generalization of learned optimizers. *CoRR*, abs/2406.00153, 2024. 3.2

Sebastian Thrun and Lorien Y. Pratt. Learning to learn: Introduction and overview. In *Learning to Learn*, pp. 3–17. Springer, 1998. 5

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *ICLR*. OpenReview.net, 2020. 1

Eleni Triantafillou, Hugo Larochelle, Richard S. Zemel, and Vincent Dumoulin. Learning a universal template for few-shot dataset generalization. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10424–10433. PMLR, 2021. 1

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pp. 5998–6008, 2017. 2.1, 2.2, 5

Ke Wang, Houxing Ren, Aojun Zhou, Zimu Lu, Sichun Luo, Weikang Shi, Renrui Zhang, Linqi Song, Mingjie Zhan, and Hongsheng Li. Mathcoder: Seamless code integration in llms for enhanced mathematical reasoning. In *ICLR*. OpenReview.net, 2024. 3.4

Zhe Wang, Jake Grigsby, Arshdeep Sekhon, and Yanjun Qi. ST-MAML : A stochastic-task based method for task-heterogeneous meta-learning. In *UAI*, volume 180 of *Proceedings of Machine Learning Research*, pp. 2066–2074. PMLR, 2022. J

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022. 1, 2.2, 5

Gail Weiss, Yoav Goldberg, and Eran Yahav. Thinking like transformers. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11080–11090. PMLR, 2021. 5

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, Haosheng Zou, Yongchao Deng, Shousheng Jia, and Xiangzheng Zhang. Light-r1: Curriculum sft, DPO and RL for long COT from scratch and beyond. *CoRR*, abs/2503.10460, 2025. 3, 3.2, H

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In *ICLR*, 2015. 5

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *EMNLP (Demos)*, pp. 38–45. Association for Computational Linguistics, 2020. G.2

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *ICLR*. OpenReview.net, 2022. 5

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: Bridging theory and practice for RLHF under kl-constraint. In *ICML*. OpenReview.net, 2024. 3.2

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *CoRR*, abs/2412.15115, 2024a. 1, 3.1

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *CoRR*, abs/2409.12122, 2024b. 3.1, 4.1, G.2

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jian Yang, Jiaxi Yang, Jingren Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025a. 1, 3.2, 3.3, 4.2, I.1

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. Towards thinking-optimal scaling of test-time compute for LLM reasoning. *CoRR*, abs/2502.18080, 2025b. 3.3

Huaxiu Yao, Yu Wang, Ying Wei, Peilin Zhao, Mehrdad Mahdavi, Defu Lian, and Chelsea Finn. Meta-learning with an adaptive task scheduler. In *NeurIPS*, pp. 7497–7509, 2021a. J

Shunyu Yao, Binghui Peng, Christos H. Papadimitriou, and Karthik Narasimhan. Self-attention networks can process bounded hierarchical languages. In *ACL/IJCNLP*, pp. 3770–3785. Association for Computational Linguistics, 2021b. 5

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *NeurIPS*, 2023. 5

Feiyang Ye, Baijiong Lin, Zhixiong Yue, Pengxin Guo, Qiao Xiao, and Yu Zhang. Multi-objective meta learning. In *NeurIPS*, pp. 21338–21351, 2021. 5

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. LIMO: less is more for reasoning. *CoRR*, abs/2502.03387, 2025. 3, 5

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Weinan Dai, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO: an open-source LLM reinforcement learning system at scale. *CoRR*, abs/2503.14476, 2025. 3, H

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Chuanqi Tan, and Chang Zhou. Scaling relationship on learning mathematical reasoning with large language models. *CoRR*, abs/2308.01825, 2023. 3.2

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, Xiangyu Yu, Gaohong Liu, Juncai Liu, Lingjun Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Ru Zhang, Xin Liu, Mingxuan Wang, Yonghui Wu, and Lin Yan. VAPO: efficient and reliable reinforcement learning for advanced reasoning tasks. *CoRR*, abs/2504.05118, 2025. 3, 4.1, H

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J. Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *ICLR*. OpenReview.net, 2020a. 5

Chulhee Yun, Yin-Wen Chang, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J. Reddi, and Sanjiv Kumar. O(n) connections are expressive enough: Universal approximability of sparse transformers. In *NeurIPS*, 2020b. 5

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning models know when they're right: Probing hidden states for self-verification. *CoRR*, abs/2504.05419, 2025. 3.3

Yangguang Zhang, Can Wang, Qihao Shi, Yan Feng, and Chun Chen. Adversarial gradient-based meta learning with metric-based test. *Knowl. Based Syst.*, 263:110312, 2023. 5

# Appendix

## Table of Contents

**A Notations** **24**

**B An Overview of the MAML Workflow** **25**

**C Pseudo Algorithm of RaML** **25**

**D Proof of Proposition 2.1** **25**

**E Instantiation of Training Techniques within RaML** **27**

**F Additional Related Work** **27**

**G Implementation Details of Experiments** **28**

G.1 Implementation Details of Visualization of Pseudo Gradient Update . . . . . 28

G.2 Training Details . . . . . 29

G.3 Evaluation Details . . . . . 31

**H More Discussions on Recent LLM Reasoning Progress** **32**

**I Additional Experiments** **32**

I.1 Details of Pseudo Gradient Update . . . . . 32

**J Future Directions** **34**

**K Limitations** **34**

# A Notations

Table 3: Illustration of notations used in the paper.

| | |
|---|---|
| $\mathcal{M}$ | the large language model |
| $\theta$ | the parameters of the large language model |
| $q$ | the question |
| $\mathcal{Q}$ | the question set |
| $q_i$ | the $i$-th question in the question set |
| $q_i^j$ | the $j$-th token of the $i$-th question |
| $a$ | the answer |
| $\mathcal{A}$ | the answer set |
| $a_i$ | the $i$-th answer of the $i$-th question |
| $a_i^j$ | the $j$-th token of the $i$-th answer |
| $\lvert \cdot \rvert$ | the length of tokens |
| $I$ | the instruction |
| d | the autoregressive decoding mechanism |
| Softmax | the softmax function |
| $\sigma$ | the activation function |
| $t$ | the reasoning trajectory |
| $\mathcal{T}$ | the set of reasoning trajectory |
| $\boldsymbol{E}_o$ | the whole output token embedding of LLM |
| $\boldsymbol{E}_{x,:}$ | the input token embedding of the sequence $x$ |
| $\boldsymbol{E}_{x,i}$ | the input token embedding of the $i$-th token in the sequence $x$ |
| $\text{Softmax}(\cdot)[x]$ | the value of softmax vector in the entry corresponding to $x$ |
| $p(\cdot)$ | the probability distribution of one token sequence determined by the LLM |
| $\Delta\mathcal{M}_\theta(\cdot)$ | the variation of the parameter $\theta$ corresponding to the inputs |
| $\theta_i'$ | the $i$-th step updated parameters |
| $\theta_t'$ | the updated parameters $\theta$ corresponding to the reasoning trajectory $t$ |
| $\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}$ | the concatenation of $\boldsymbol{x}$ and $\boldsymbol{y}$ |
| $\boldsymbol{W}_k, \boldsymbol{W}_q, \boldsymbol{W}_v$ | the parameters in self-attention layer |
| $\boldsymbol{W}_1, \boldsymbol{W}_2, b_1, b_2$ | the parameters in feed-forward network |
| $\mathcal{L}_q$ | the loss corresponding to the question |

**Algorithm 1:** Model-Agnostic Meta-Learning

**Input:** $p(\mathcal{T})$: distribution over tasks, $\alpha, \beta$: step size hyperparameters.

Randomly initialize $\theta$ ;
**while** *not done* **do**
  Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$ ;
  **for *all*** $\mathcal{T}_i$ **do**
    Evaluate $\nabla\mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to $K$ examples ;
    Compute adapted parameters with gradient descent: $\theta' = \theta - \alpha\nabla_\theta\mathcal{L}_{\mathcal{T}_i}(f_\theta)$ ;
Update $\theta \leftarrow \theta - \beta\nabla\sum_{\mathcal{T}_i\sim p(\mathcal{T})}\mathcal{L}_{\mathcal{T}_i}(f_{\theta'})$ ;

## B An Overview of the MAML Workflow

Algorithm 1 illustrates the algorithm of the overall workflow of Model-Agnostic Meta-Learning (MAML).

## C Pseudo Algorithm of RaML

Algorithm 2 shows the overall flow of our proposed meta-learning perspective on LLM reasoning.

**Algorithm 2:** Meta-Learning Perspective on LLM Reasoning

**Input:** $\mathcal{M}_\theta$: LLM, $I$: instruction, $\mathcal{Q}$: question set, $\mathcal{T}_i$: reasoning trajectories for each question $i$.

**while** *not training done* **do**
  Sample batch of questions $q_i$ from $\mathcal{Q}$ ;
  **for** ***all*** $q_i$ **do**
    Obtain reasoning trajectories $\mathcal{T}_i$ of each $q_i$ through training data or rollout ;
    Update $\theta$ to $\theta'_{t_j}$ by reasoning trajectory $t_j \in \mathcal{T}_i$ refer to Equation (4) ;
    Optimize $\theta$ through $\sum_{t_j \in \mathcal{T}_i} \mathcal{L}_{q_i}\left(\mathcal{M}_{\theta'_{t_j}}\right)$ for each $q_i$ ;

## D Proof of Proposition 2.1

*Proof.* We aim to construct a parameter set $\{\boldsymbol{W}'_q, \boldsymbol{W}'_k, \boldsymbol{W}'_v, \boldsymbol{W}'_1, \boldsymbol{W}'_2, b'_1, b'_2\}$ such that the activation obtained by attending to the first reasoning token $t^0$ can be represented equivalently by the original input sequence $l$ under updated parameters:

$$\begin{aligned} &\boldsymbol{W}_2^T\left(\sigma\left(\boldsymbol{W}_1^T\left(\text{Softmax}\left(\boldsymbol{E}_{t,0}\boldsymbol{W}_q\boldsymbol{W}_k^T\begin{bmatrix}\boldsymbol{E}_{l,:}\\ \boldsymbol{E}_{t,0}\end{bmatrix}^T\right)\begin{bmatrix}\boldsymbol{E}_{l,:}\\ \boldsymbol{E}_{t,0}\end{bmatrix}\boldsymbol{W}_v\right)+b_1\right)\right)+b_2 \\ &= \boldsymbol{W}_2'^T\left(\sigma\left(\boldsymbol{W}_1'^T\left(\text{Softmax}\left(\boldsymbol{E}_{l,-1:}\boldsymbol{W}'_q\boldsymbol{W}_k'^T\boldsymbol{E}_{l,:}^T\right)\boldsymbol{E}_{l,:}\boldsymbol{W}'_v\right)+b'_1\right)\right)+b'_2. \end{aligned} \tag{9}$$

Here, $\boldsymbol{E}_{l,:} \in \mathbb{R}^{|l|\times d}$, $\boldsymbol{E}_{l,-1}, \boldsymbol{E}_{t,0} \in \mathbb{R}^{1\times d}$, $\boldsymbol{W}_q, \boldsymbol{W}'_q, \boldsymbol{W}_k, \boldsymbol{W}'_k, \boldsymbol{W}_v, \boldsymbol{W}'_v \in \mathbb{R}^{d\times d}$, and $\boldsymbol{W}_1, \boldsymbol{W}'_1, \boldsymbol{W}_2, \boldsymbol{W}'_2 \in \mathbb{R}^{d\times d}$.

We keep the feed-forward parameters unchanged:

$$\boldsymbol{W}'_1 = \boldsymbol{W}_1, \quad \boldsymbol{W}'_2 = \boldsymbol{W}_2, \quad b'_1 = b_1, \quad b'_2 = b_2. \tag{10}$$

Therefore, it suffices to construct $\boldsymbol{W}'_q$, $\boldsymbol{W}'_k$, and $\boldsymbol{W}'_v$ such that the attention outputs match:

$$\begin{aligned} &\text{Softmax}\left(\boldsymbol{E}_{t,0}\boldsymbol{W}_q\boldsymbol{W}_k^T\begin{bmatrix}\boldsymbol{E}_{l,:}\\ \boldsymbol{E}_{t,0}\end{bmatrix}^T\right)\begin{bmatrix}\boldsymbol{E}_{l,:}\\ \boldsymbol{E}_{t,0}\end{bmatrix}\boldsymbol{W}_v \\ &= \text{Softmax}\left(\boldsymbol{E}_{l,-1:}\boldsymbol{W}'_q\boldsymbol{W}_k'^T\boldsymbol{E}_{l,:}^T\right)\boldsymbol{E}_{l,:}\boldsymbol{W}'_v. \end{aligned} \tag{11}$$

For clarity, we introduce the following notation:

$$\begin{aligned} \boldsymbol{Q} &= \boldsymbol{E}_{t,0}\boldsymbol{W}_q \in \mathbb{R}^{1\times d}, \\ \boldsymbol{K}^T &= \boldsymbol{W}_k^T\begin{bmatrix}\boldsymbol{E}_{l,:}\\ \boldsymbol{E}_{t,0}\end{bmatrix}^T \in \mathbb{R}^{d\times(|l|+1)}, \\ \boldsymbol{V} &= \begin{bmatrix}\boldsymbol{E}_{l,:}\\ \boldsymbol{E}_{t,0}\end{bmatrix}\boldsymbol{W}_v \in \mathbb{R}^{(|l|+1)\times d}, \\ \boldsymbol{Q}' &= \boldsymbol{E}_{l,-1:}\boldsymbol{W}'_q \in \mathbb{R}^{1\times d}, \\ \boldsymbol{K}'^T &= \boldsymbol{W}_k'^T\boldsymbol{E}_{l,:}^T \in \mathbb{R}^{d\times|l|}, \\ \boldsymbol{V}' &= \boldsymbol{E}_{l,:}\boldsymbol{W}'_v \in \mathbb{R}^{|l|\times d}. \end{aligned} \tag{12}$$

We first align the query vectors. Since $\boldsymbol{E}_{t,0}$ and $\boldsymbol{E}_{l,-1:}$ are both vectors in $\mathbb{R}^{1\times d}$, there exists a linear transformation $\boldsymbol{P} \in \mathbb{R}^{d\times d}$ such that

$$\boldsymbol{E}_{l,-1:}\boldsymbol{P} = \boldsymbol{E}_{t,0}. \tag{13}$$

For example, such a transformation can be constructed by extending $\boldsymbol{E}_{l,-1:}$ to a basis of $\mathbb{R}^d$ and specifying its image as $\boldsymbol{E}_{t,0}$, with arbitrary images assigned to the remaining basis vectors. Then, by setting

$$\boldsymbol{W}'_q = \boldsymbol{P}\boldsymbol{W}_q, \tag{14}$$

we obtain

$$\boldsymbol{Q}' = \boldsymbol{E}_{l,-1:}\boldsymbol{W}'_q = \boldsymbol{E}_{l,-1:}\boldsymbol{P}\boldsymbol{W}_q = \boldsymbol{E}_{t,0}\boldsymbol{W}_q = \boldsymbol{Q}. \tag{15}$$

Thus, Equation (11) reduces to matching the key-value attention outputs:

$$\mathrm{Softmax}(\boldsymbol{Q}\boldsymbol{K}^T)\boldsymbol{V} = \mathrm{Softmax}(\boldsymbol{Q}\boldsymbol{K}'^T)\boldsymbol{V}'. \tag{16}$$

Next, we express the extended embedding matrix $\begin{bmatrix}\boldsymbol{E}_{l,:}\\ \boldsymbol{E}_{t,0}\end{bmatrix}$ using $\boldsymbol{E}_{l,:}$. If $\boldsymbol{E}_{t,0}$ lies in the row span of $\boldsymbol{E}_{l,:}$, then there exists $\boldsymbol{C} \in \mathbb{R}^{1\times|l|}$ such that

$$\boldsymbol{E}_{t,0} = \boldsymbol{C}\boldsymbol{E}_{l,:}. \tag{17}$$

If $\boldsymbol{E}_{t,0}$ does not lie exactly in this row span, we use the least-squares approximation

$$\boldsymbol{E}_{t,0} \approx \boldsymbol{C}\boldsymbol{E}_{l,:}. \tag{18}$$

Define

$$\boldsymbol{M} = \begin{bmatrix}\boldsymbol{I}_{|l|} & \boldsymbol{C}^T\end{bmatrix} \in \mathbb{R}^{|l|\times(|l|+1)}. \tag{19}$$

Then,

$$\begin{bmatrix}\boldsymbol{E}_{l,:}\\ \boldsymbol{E}_{t,0}\end{bmatrix} \approx \boldsymbol{M}^T\boldsymbol{E}_{l,:}. \tag{20}$$

Substituting this approximation into the original attention output gives

$$\begin{aligned}\mathrm{Softmax}(\boldsymbol{Q}\boldsymbol{K}^T)\boldsymbol{V} &= \mathrm{Softmax}\left(\boldsymbol{Q}\boldsymbol{W}_k^T\begin{bmatrix}\boldsymbol{E}_{l,:}\\ \boldsymbol{E}_{t,0}\end{bmatrix}^T\right)\begin{bmatrix}\boldsymbol{E}_{l,:}\\ \boldsymbol{E}_{t,0}\end{bmatrix}\boldsymbol{W}_v\\ &\approx \mathrm{Softmax}\left(\boldsymbol{Q}\boldsymbol{W}_k^T\boldsymbol{E}_{l,:}^T\boldsymbol{M}\right)\boldsymbol{M}^T\boldsymbol{E}_{l,:}\boldsymbol{W}_v.\end{aligned} \tag{21}$$

To match this form with $\mathrm{Softmax}(\boldsymbol{Q}\boldsymbol{W}_k'^T\boldsymbol{E}_{l,:}^T)\boldsymbol{E}_{l,:}\boldsymbol{W}'_v$, we choose

$$\begin{aligned}\boldsymbol{W}_k'^T\boldsymbol{E}_{l,:}^T &\approx \boldsymbol{W}_k^T\boldsymbol{E}_{l,:}^T\boldsymbol{M},\\ \boldsymbol{E}_{l,:}\boldsymbol{W}'_v &\approx \boldsymbol{M}^T\boldsymbol{E}_{l,:}\boldsymbol{W}_v.\end{aligned} \tag{22}$$

One possible least-squares construction is

$$\begin{aligned}\boldsymbol{W}_k'^T &= \boldsymbol{W}_k^T\boldsymbol{E}_{l,:}^T\boldsymbol{M}\left(\boldsymbol{E}_{l,:}^T\right)^\dagger,\\ \boldsymbol{W}'_v &= \boldsymbol{E}_{l,:}^\dagger\boldsymbol{M}^T\boldsymbol{E}_{l,:}\boldsymbol{W}_v,\end{aligned} \tag{23}$$

where $\boldsymbol{E}_{l,:}^\dagger$ denotes the Moore–Penrose pseudoinverse of $\boldsymbol{E}_{l,:}$ (Moore, 1920; Bjerhammar, 1951; Penrose, 1955). When $\boldsymbol{E}_{t,0}$ is exactly representable by the rows of $\boldsymbol{E}_{l,:}$ and the corresponding linear systems admit exact solutions, the above construction yields an exact equality. Otherwise, it provides an approximate representation whose error is determined by the projection residual of $\boldsymbol{E}_{t,0}$ onto the row span of $\boldsymbol{E}_{l,:}$ and the least-squares reconstruction errors.

Combining the constructions above, we obtain a parameter set

$$\{\boldsymbol{W}'_q, \boldsymbol{W}'_k, \boldsymbol{W}'_v, \boldsymbol{W}'_1, \boldsymbol{W}'_2, b'_1, b'_2\}$$

Table 4: Comparison of training techniques, where SFT, PO, and RL mean the abbreviation of supervised fine-tuning, preference optimization, and reinforcement learning, respectively.

| Techniques | Reasoning Trajectories | Outer Loop Loss |
|---|---|---|
| SFT | Off-policy | $\mathcal{L} = -\mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\log p_\theta(y \mid x)\right]$ (Radford et al., 2018) |
| Off-Policy PO | Off-policy | $\mathcal{L} = -\log \sigma \left(r_\theta(y_{\text{preferred}}) - r_\theta(y_{\text{dispreferred}})\right)$ (Rafailov et al., 2023) |
| On-Policy RL | On-policy | $\mathcal{L} = -\mathbb{E}_t \left[\min \left(r_t(\theta)\hat{A}_t,\, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right]$ (Schulman et al., 2017) |

under which attending to the reasoning token $t^0$ can be represented as an implicit update of the model parameters. This establishes the one-step pseudo-gradient update interpretation in Proposition 2.1, up to the approximation error introduced by the least-squares representation.

The constructed parameter set is not necessarily unique. For instance, by the universal approximation theorem (Cybenko, 1989; Hornik, 1991), the feed-forward layers can further compensate for discrepancies in the attention computation, providing additional degrees of freedom for constructing $\boldsymbol{W}_q'$, $\boldsymbol{W}_k'$, and $\boldsymbol{W}_v'$.

This completes the proof of Proposition 2.1. □

## E Instantiation of Training Techniques within RaML

We review representative training techniques from a meta-learning perspective, including supervised fine-tuning (Howard & Ruder, 2018; Devlin et al., 2019), off-policy preference optimization (Rafailov et al., 2023), and on-policy reinforcement learning (Schulman et al., 2017; Ouyang et al., 2022; Shao et al., 2024; Ahmadian et al., 2024). We categorize these techniques into two macro-level stages. In the first stage, reasoning trajectories are obtained and fed into the LLM, inducing an update to its parameters $\theta$ and producing an output token distribution under the updated state. In the second stage, the original parameters $\theta$ are optimized with a specific loss function defined over this output distribution. Since different loss functions may lead to the same maximum likelihood estimation (MLE) objective (Swamy et al., 2025), we attribute the essential distinction among training techniques primarily to their inner-loop optimization processes. Inner-loop optimization plays a central role in meta-learning, as the resulting meta-gradient directly affects the effectiveness of meta-training.

Off-policy training techniques acquire reasoning trajectories from human annotations, external models, or synthetic generation, whereas on-policy training techniques generate trajectories from the model's own distribution. From a learning-to-optimize perspective, off-policy training can be viewed as learning from an *oracle meta-optimizer* that provides guided inner-loop optimization trajectories. In contrast, on-policy RL requires the model to independently explore inner-loop optimization paths. Although this greater degree of freedom makes optimization more challenging, it also provides the potential to discover superior reasoning trajectories and achieve stronger performance.

## F Additional Related Work

**Additional Discussion on RaML and Related Work.** We further clarify the relationship between RaML and closely related studies. Dai et al. (Dai et al., 2023) interpret in-context learning (ICL) as generating meta-gradients from demonstration examples and applying them to the base GPT model to form an ICL system. In this formulation, each demonstration example acts as a data sample for implicitly updating model parameters. In contrast, RaML focuses on trajectory-aided reasoning, where each token in a reasoning trajectory is viewed as an update step. This token-level interpretation allows us to connect LLM reasoning with broader reasoning-oriented training paradigms, including supervised fine-tuning and reinforcement learning, rather than restricting the analysis to demonstration-based adaptation. Moreover, RaML explicitly considers training procedures with parameter optimization, whereas ICL is limited by the small number of available demonstrations.

Another line of work, such as Gatmiry et al. (Gatmiry et al., 2024) and related studies (Huang et al., 2025), shows that transformers can implement learning algorithms, including gradient descent, within chain-of-thought reasoning. These studies mainly examine whether transformers can express or execute learning algorithms in natural language for numerical optimization problems. By contrast, RAML interprets reasoning trajectories as implicit updates to the internal parameters of transformer models, offering a more general perspective that extends beyond explicit numerical optimization tasks.

# G Implementation Details of Experiments

## G.1 Implementation Details of Visualization of Pseudo Gradient Update

**Data Preparation.** We select four questions from AIME2024 as follows:

Details of $q_0$

**Question**

Every morning Aya goes for a 9-kilometer-long walk and stops at a coffee shop afterwards. When she walks at a constant speed of $s$ kilometers per hour, the walk takes her 4 hours, including $t$ minutes spent in the coffee shop. When she walks $s+2$ kilometers per hour, the walk takes her 2 hours and 24 minutes, including $t$ minutes spent in the coffee shop. Suppose Aya walks at $s+\frac{1}{2}$ kilometers per hour. Find the number of minutes the walk takes her, including the $t$ minutes spent in the coffee shop.

**Answer**

204

Details of $q_1$

**Question**

There exist real numbers $x$ and $y$, both greater than 1, such that $\log_x\left(y^x\right)=\log_y\left(x^{4y}\right)=10$. Find $xy$.

**Answer**

025

Details of $q_2$

**Question**

Find the largest possible real part of

$$(75+117i)z+\frac{96+144i}{z}$$

where $z$ is a complex number with $|z|=4$.

**Answer**

540

Details of $q_3$

**Question**

Let $\triangle ABC$ have circumcenter $O$ and incenter $I$ with $\overline{IA} \perp \overline{OI}$, circumradius 13, and inradius 6. Find $AB \cdot AC$.

**Answer**

468

**Visualization of Pseudo Gradient Update.** We leverage QwQ-32B (Team, 2025c) to generate trajectories for these four questions. Then for each trajectory, we calculate the negative log-probability of `Final Answer\n\boxed..answer..` at each position. Algorithm 3 outlines the overall process.

**Algorithm 3:** Computation of Empirical Examples of Pseudo Gradient Update

**Input:** $\mathcal{M}$: QwQ-32B, $I$: instruction, $q$: question from AIME2024, $t$: the trajectory generated by QwQ-32B, $a$: the answer sequence, i.e., `Final Answer\n\boxed..answer..`, $s$: step size.

**for** $i \in [0, |t|)$ **do**
  Obtain input by $I \oplus q \oplus t_{:i} \oplus a$ ;
  Feed input to $M$ and get logits $l_a$ of answer sequence ;
  Compute the negative log-probability using $l_a$ ;

**Visualization of Landscape.** We refer to the methodology proposed by Li et al. (Li et al., 2018). Assuming the set parameters of QwQ-32B is denoted by $\{\theta_k\}$ (excluding the embedding matrix), we randomly select two vectors, $\{\theta_{1,k}\}$ and $\{\theta_{2,k}\}$, for each parameter. We then edit the parameters by adding $\alpha_1\theta_{1,k} + \alpha_2\theta_{2,k}$ and compute the negative log-probability given only the instruction and question to form the point set $\{(\alpha_1, \alpha_2, \widehat{\mathcal{L}}_{\alpha_1,\alpha_2})\}$. Finally, we visualize this point set to reveal the landscape. The overall process is summarized as Algorithm 4.

**Algorithm 4:** Computation of Landscape

**Input:** $\mathcal{M}$: QwQ-32B, $I$: instruction, $q$: question from AIME2024, $a$: the answer sequence, i.e., `Final Answer\n\boxed..answer...`

Obtain random vectors $\{\theta_{1,k}\}$, $\{\theta_{2,k}\}$ for each parameter $\theta_k$ of $\mathcal{M}$ ;
**for** $i \in [-1, 1, s]$ **do**
  **for** $j \in [-1, 1, s]$ **do**
    Get edited parameters $\{\theta'_k\}$ by adding $i \cdot \theta_{1,k} + j \cdot \theta_{2,k}$ ;
    Obtain input by $I \oplus q \oplus a$ ;
    Feed input to $M$ and get logits $l_a$ of answer sequence ;
    Compute the negative log-probability using $l_a$ ;

**Project the Pseudo Gradient Update to Landscape.** To project the trajectory of the pseudo-gradient update onto the landscape, we fix one direction corresponding to the time dimension and identify the closest contour to the corresponding $\widehat{\mathcal{L}}$ to determine another direction.

### G.2 Training Details

**Dataset Processing.** To maintain the validity and verifiability of the question set, we clean and filter the original dataset. Initially, we exclude incomplete questions as well as those lacking answers. Subsequently, we remove questions requiring reasoning with images or other external information. To further ensure verifiability,

we employ Math-Verify [2] to examine each question and exclude those that could not be verified. Finally, we eliminate irrelevant characters, such as URLs and HTML tags, resulting in approximately 39k questions with corresponding answers.

**Training of SFT.** We first synthesize training trajectories from Qwen2.5-Math-72B-Instruct (Yang et al., 2024b) and DeepSeek-R1-Distill-Qwen-14B (DeepSeek-AI et al., 2025). From the entire question set, we sample 10k questions and use the sampling parameters shown in Table 5 to generate reasoning trajectories with the prompt, *Please solve the following mathematical problem step by step and put your final answer in \boxed*, resulting in 640k trajectories. We then filter out trajectories with incorrect answers, retaining approximately

Table 5: Sampling parameters leveraging for reasoning trajectories synthesis.

| | **Qwen2.5-Math-72B-Instruct** | **DeepSeek-R1-Distill-Qwen-14B** |
|---|---|---|
| Temperature | 0.7 | 0.7 |
| Top-$p$ | 1.0 | 1.0 |
| Top-$k$ | 50 | 50 |
| Max Tokens | 8192 | 36784 |
| Rollout Number | 64 | 64 |

$\sim$ 470k for Qwen2.5-Math-72B-Instruct and approximately $\sim$ 550k for DeepSeek-R1-Distill-Qwen-14B. During training, we utilize the parameters listed in Table 6.

Table 6: Training parameters for SFT.

| | **Parameter** |
|---|---|
| Max Response Length | 18432 |
| Train Batch Size | 256 |
| Learning Rate | 1$e$-5 |
| Total Epochs | 1 |

Table 7: Training parameters for GRPO.

| | **Parameter** |
|---|---|
| Max Prompt Length | 1024 |
| Max Response Length | 16384 |
| Rollout Temperature | 1.0 |
| Rollout Number | 16 |
| Train Batch Size | 1024 |
| Learning Rate | 1$e$-6 |
| Total Epochs | 1 |

**System Prompt of GRPO**

A conversation between a User and an Assistant. The User poses a question, and the Assistant provides a solution. The Assistant's response follows these structured steps:

1. **Reasoning Process**: The Assistant reflects on the problem using a reasoning process enclosed within <think> and </think> tags.
2. **Conclusion**: The Assistant reaches a conclusion, which is enclosed within <conclusion> and </conclusion> tags. The final answer is highlighted within \boxed...final answer....
3. **Answer Format**: The complete response should be formatted as:

<think>
...reasoning process...
</think>
<conclusion>
...conclusion...
The answer is \boxed...final answer...
</conclusion>

[2] https://github.com/huggingface/Math-Verify

**Training of GRPO.** For the GRPO training, we use the complete question set and apply the parameters listed in Table 7. We adhere to the DeepSeek-R1-style system prompt, as presented in the *System Prompt of GRPO* box. And for the reward design, we assign the trajectory with the correct answer and correct format the score 1, the trajectory with the false answer and correct format the score 0.0, trajectory with the correct answer and false format the score $-0.5$, and trajectory with the false answer and false format the score $-1$, formally:

$$R(y', y) = \begin{cases} 1 & \text{answer_match}(y', y) \quad \text{and} \quad \text{format_correct}(y'), \\ 0 & \neg\text{answer_match}(y', y) \quad \text{and} \quad \text{format_correct}(y'), \\ -0.5 & \text{answer_match}(y', y) \quad \text{and} \quad \neg\text{format_correct}(y'), \\ -1 & \neg\text{answer_match}(y', y) \quad \text{and} \quad \neg\text{format_correct}(y'), \end{cases} \tag{24}$$

where $y$ indicates the ground-truth and $y'$ indicates the trajectory. We employ the Math-Verify package to ascertain the equivalence of $y$ and $y'$.

**Details of Hardware and Software.** All the training tasks are conducted based on veRL (Sheng et al., 2025), cooperated with Pytorch (Paszke et al., 2019) 2.6.0, Transformers (Wolf et al., 2020) 4.51.3, vLLM (Kwon et al., 2023) 0.8.4. We conduct all experiments on clusters equipped with NVIDIA A800 GPUs and Intel(R) Xeon(R) Platinum 8336C CPUs.

### G.3 Evaluation Details

**Benchmarks.** The following details describe our evaluation benchmarks:

- **AIME24.** AIME24[3] consists of 30 challenging questions from the 2024 American Invitational Mathematics Examination (AIME).
- **MATH500.** The original MATH dataset (Hendrycks et al., 2021) comprises $12,500$ problems from American high school mathematics competitions. MATH500 (Lightman et al., 2024), a widely used subset of its test split, includes only Level 5 questions in this study.
- **LiveMathBench.** LiveMathBench (Liu et al., 2024) is a continuously updated dataset of challenging mathematical problems. We use the December 2024 hard split, comprising 45 questions in English and Chinese.
- **GPQA.** GPQA (Rein et al., 2023) dataset is a challenging, professional multiple-choice science question-answering dataset. We use its diamond subset, comprising 198 questions.
- **LiveCodeBench.** LiveCodeBench (Jain et al., 2024) is a benchmark designed for a comprehensive and uncontaminated evaluation of the code-related capabilities of LLMs. It incorporates questions from LeetCode, AtCoder, and Codeforces.

**Metrics.** We use Pass@$k$ and mG-Pass@$k$ (Liu et al., 2024) as evaluation metrics. We generate $n$ responses for each question and assume the number of correct responses is $c$. Then the metrics are computed as:

- **Pass@$k$.**

$$\text{Pass@}k = \mathbb{E}_{\text{questions}} \left[ 1 - \frac{\binom{n-c}{k}}{\binom{n}{k}} \right]. \tag{25}$$

- **mG-Pass@$k$.**

$$\text{mG-Pass@}k = \mathbb{E}_{\text{questions}} \left[ \frac{2}{k} \sum_{i=\lceil k/2 \rceil + 1}^{k} \sum_{j=i}^{c} \frac{\binom{c}{j} \cdot \binom{n-c}{k-j}}{\binom{n}{k}} \right]. \tag{26}$$

[3] https://huggingface.co/datasets/AI-MO/aimo-validation-aime

# H More Discussions on Recent LLM Reasoning Progress

In this section, we discuss recent advances in LLM reasoning and reinterpret their key improvements within our meta-learning framework. We consider several representative works, including OpenThoughts (Team, 2025b), Light-R1 (Wen et al., 2025), Open-Reasoner-Zero (Hu et al., 2025), DAPO (Yu et al., 2025), VAPO (Yue et al., 2025), GPG (Chu et al., 2025b), and Llama Nemotron (Bercovich et al., 2025).

**Data Filtering.** Works such as Light-R1 (Wen et al., 2025) adopt diversity- and difficulty-based filtering to construct high-quality training data. From a meta-learning perspective, such filtering can be interpreted as a form of task-set mining, which improves the distribution of training tasks and thereby enhances the efficiency and effectiveness of meta-training.

**Synthetic Data from Strong Reasoning LLMs.** Works such as OpenThoughts (Team, 2025b) and Llama Nemotron (Bercovich et al., 2025) leverage stronger reasoning models, such as DeepSeek-R1, to generate multiple reasoning trajectories for each training question, which are then used for SFT. Under our framework, this strategy expands the support set associated with each task, leading to more stable inner-loop optimization and improved reasoning performance. It can also be viewed as distilling optimization paths from a well-trained reasoning model into a smaller model.

**Higher Clipping Range in RL Surrogate Objectives.** DAPO (Yu et al., 2025) increases the upper clipping bound in the GRPO surrogate objective to encourage exploration during RL training. Similar effects can also arise from removing or relaxing regularization terms such as KL penalties or entropy losses. From an optimization perspective, these modifications enlarge the exploration space of inner-loop optimization paths, enabling the model to search more aggressively for high-reward reasoning trajectories. Increasing data diversity in SFT may provide a related effect by exposing the model to a broader range of optimization paths.

**Dynamic Sampling During RL.** Recent studies (Chu et al., 2025b; Yu et al., 2025) employ dynamic sampling or biased sampling to balance correct and incorrect trajectories during rollout. Within our framework, this strategy adjusts the composition of positive and negative optimization signals, preventing the inner-loop optimizer from overfitting to a narrow class of trajectories and improving training stability.

**Group Sampling for PPO.** Classical RL algorithms typically sample only a single trajectory for each problem instance. Recent works (Yue et al., 2025; Hu et al., 2025) extend PPO-style training with group sampling, allowing multiple trajectories to be generated for each problem. From our perspective, this increases the support-set size for each task, thereby strengthening inner-loop optimization and improving the stability of the resulting meta-gradient.

# I Additional Experiments

## I.1 Details of Pseudo Gradient Update

**QwQ's Pseudo Gradient Update.** Following the methodology described in § 2.2 and § G.1, we monitor the pseudo-gradient update of QwQ-32B. As shown in Figure 11, the negative log-probability progressively decreases along the reasoning trajectories which aligns with our definition.

**Qwen3's Pseudo Gradient Update.** Following the methodology described in § 2.2 and § G.1, we monitor the pseudo-gradient update of Qwen3-32B (Yang et al., 2025a) under thinking mode, as illustrated in Figure 12. We observe that the reasoning trajectories of Qwen3 exhibit a parameter update effect.

**Qwen3's Pseudo Gradient Update in Thinking/NoThinking Mode.** Referring to the settings in § 3.3, we examine the differences between thinking mode and no-thinking mode, as shown in Figure 13. It is evident that, due to the specific optimization of Qwen3, its no-thinking token delimiter (i.e., `</think>`) demonstrates a more pronounced gradient descent effect. The delimiter `</think>` enables the model to swiftly update to an extreme point in the appropriate direction with a larger step size. However, this update is

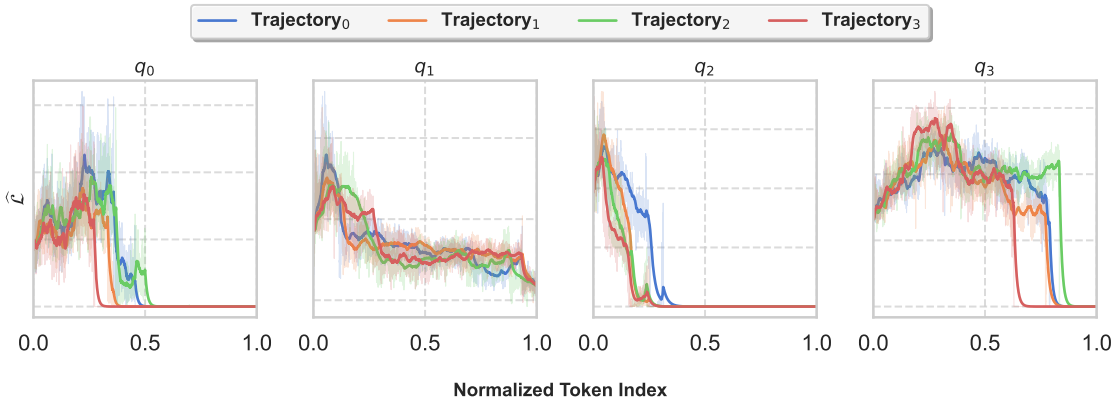


Figure 11: Visualization of the *pseudo-gradient update*: The $x$-axis represents the normalized indices of corresponding trajectories. $q_0, q_1, q_2, q_3$ are question selected from AIME24, refer to § G.1 for more details.

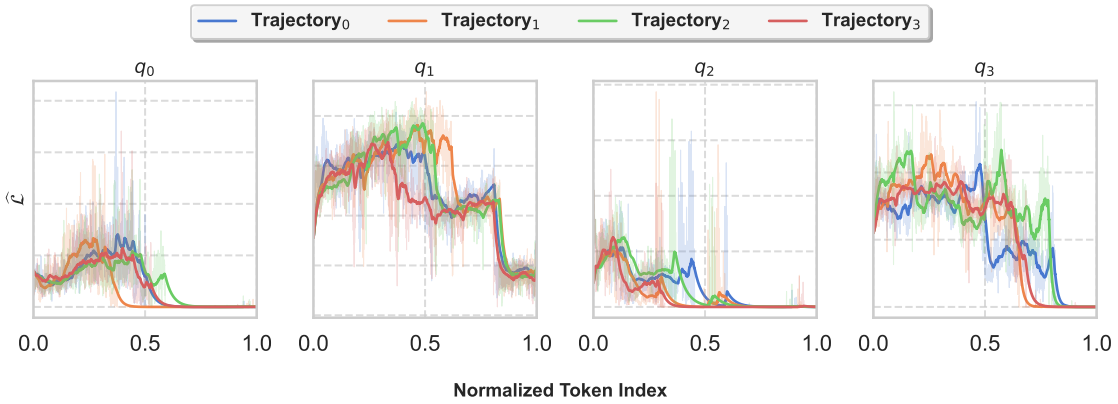


Figure 12: Visualization of the *pseudo-gradient update*: The $x$-axis represents the normalized indices of corresponding trajectories. $q_0, q_1, q_2, q_3$ are question selected from AIME24.

susceptible to falling into local minima, which accounts for the performance gap between Qwen3's no-thinking mode and thinking mode.

**Pseudo Gradient Update of False Reasoning Trajectories.** Figure 14 illustrates the curve of pseudo gradient updates associated with incorrect reasoning trajectories. It is evident that the curve representing these trajectories does not show a downward trend, underscoring the strong connection between reasoning paths and optimization processes.

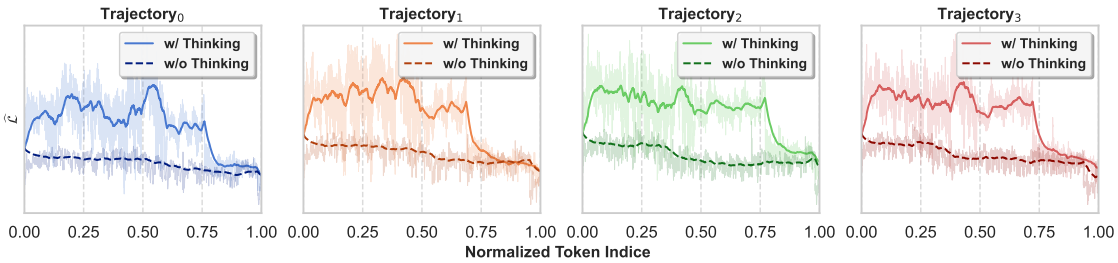


Figure 13: Illustration of Qwen3's *pseudo-gradient update* for both thinking and non-thinking modes. We visualize four pairs of correct reasoning trajectories for one question in AIME24.

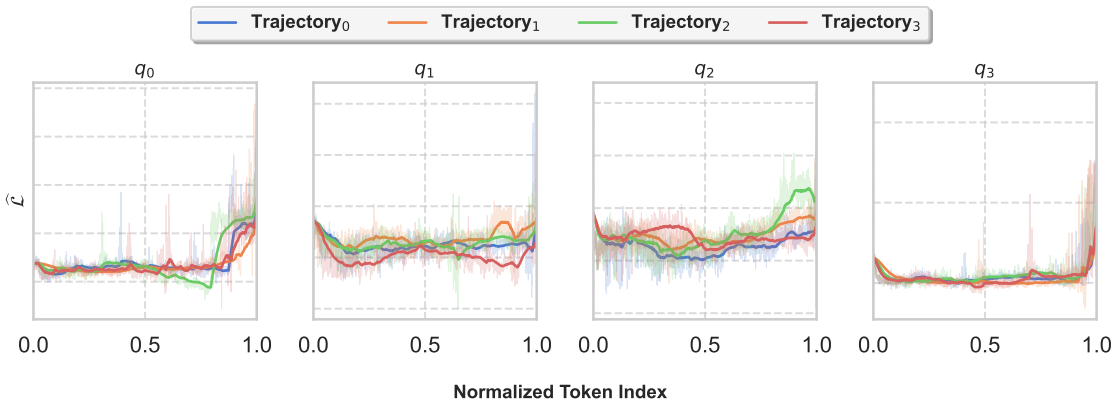


Figure 14: Illustration Qwen3's *pseudo-gradient update* corresponding to false reasoning trajectories.

## J Future Directions

Further studies can be conducted based on our work, for instance:

- **Further Understanding Reasoning Trajectories**
  - ❶ Unlike typical meta-learning frameworks with predefined support sets, the reasoning trajectories in LLMs are self-generated. This implies that LLMs inherently learn gradient update steps without needing explicit support sets for fine-tuning. *Investigating how LLMs learn to form effective reasoning trajectories, namely, gradient update steps*, presents an intriguing challenge.
  - ❷ Tokens contribute differently to the modification of model parameters. *What accounts for this disparity among tokens? Is it connected to their semantic properties, and if so, in what manner*?
  - ❸ Trajectory-aided reasoning in large language models (LLMs) demonstrates comparable generalization abilities across various tasks. *What aspects of the learning process contribute to this generalization ability, and which meta-features are developed through the optimization of reasoning trajectories?*
- **Towards Enhancing LLM Reasoning**
  - ❶ *Improved Reasoning Trajectory Selection Strategy in LLM Training.* During both supervised fine-tuning and reinforcement learning, reasoning trajectories usually remain constant. Could implementing an adaptive sampling mechanism, similar to those utilized in meta-learning (Yao et al., 2021a; Liu et al., 2020), enhance training efficacy?
  - ❷ *Enhancing Reasoning Efficiency Through an Optimization Perspective.* Given that each token offers a unique contribution to optimization, is there a strategy to discern these contributions to filter out superfluous tokens, thereby improving reasoning efficiency?
  - ❸ *Task (Question) Ratio to Enhance Generalization Across Different Domains.* Insights from related studies, such as Collins et al. (Collins et al., 2020) and Wang et al. (Wang et al., 2022), in meta-learning, suggest methods to bolster the reasoning capability of LLMs, enabling them to generalize across domains—for instance, training on mathematical data and inferring insights from coding data.

## K Limitations

**Limitations of Theoretical Analysis.** This paper's theoretical analysis, while providing a foundational framework, relies on several approximate assumptions, as discussed in **??**. Although these assumptions hold valid in most typical scenarios, a more detailed investigation into their specific conditions and potential limitations is warranted in subsequent research to fully understand their implications. Additionally, while the theory presented herein confirms the existence of the related equation cited in Proposition 2.1, deriving its precise analytical solution remains an open problem requiring further exploration. Finding this solution could potentially lead to a deeper theoretical understanding of the pseudo-gradient dynamics.

**Limitations of Experiments.** Firstly, due to time and resource constraints, our experiments primarily focused on mathematical reasoning tasks. While mathematical reasoning shares many commonalities with other reasoning tasks, suggesting our conclusions may generalize, differences in the specific optimization behavior across diverse task types warrant further dedicated study. Secondly, experiments were conducted on a limited selection of LLMs. Although the observed performance behavior across these models was largely consistent, indicating the general applicability of our conclusions, future work should investigate a wider range of LLM architectures to identify any subtle, model-specific differences. Finally, mirroring a limitation discussed previously, our experiments were confined to the textual modality. Given that recent research indicates the significance of reasoning trajectories in multimodal contexts, exploring their role through experiments involving multimodal data is a valuable direction.