Quantifying the Stability of Multi-Step Reasoning via Error Amplification

Dongyue Li
Northeastern University
Boston, MA
li.dongyu@northeastern.edu

Ziniu Zhang
Northeastern University
Boston, MA
zhang.zini@northeastern.edu

Minxuan Duan
Northeastern University
Boston, MA
dminxuan@gmail.com

Hongyang R. Zhang
Northeastern University
Boston, MA
ho.zhang@northeastern.edu

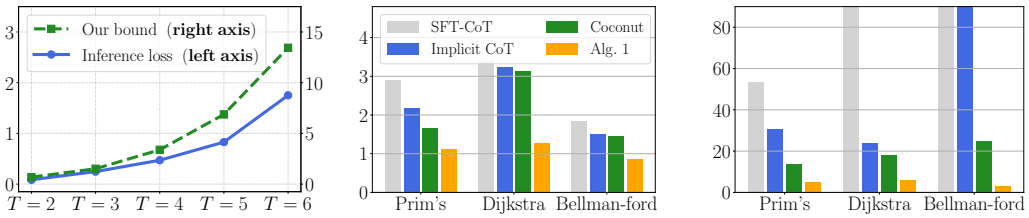
Abstract

We consider the stability of multi-step reasoning processes, which have extensive applications in language models, including chain-of-thought and algorithmic reasoning. While longer sequences of reasoning can improve a model's generation capability at test time, the errors due to intermediate reasoning steps can accumulate in autoregressive generation, and thus grow substantially at the end. In this paper, we ask: *What are the key factors determining the stability of multi-step reasoning?* First, we show an inference error bound governed by the product of *spectral norms of the Jacobians taken through the input space* across generation steps. This product can be viewed as an *error amplification* factor, which could scale exponentially with the number of reasoning steps, serving as a quantitative measure of reasoning stability. Second, we analyze this measure in transformer models trained to predict simple tasks like linear and quadratic functions. We theoretically prove that the transformer model converges to a solution where the stability measure decays, thus yielding nearly zero inference loss over (arbitrarily) long steps. Finally, the stability analysis leads to several algorithmic implications for controlling the stability, through (i) chain-of-thought length compression that reduces the sensitivity of each step, and (ii) quantization-aware training that regularizes the input Jacobian norms. We validate the proposed algorithms by fine-tuning language models on graph-algorithmic reasoning tasks and symbolic state-tracking tasks. Across seven evaluations, our algorithms improve over baseline comparisons by 3.5% on average, and by 8.2% for longer-length inputs. Ablation analysis validates that the stability measure is drastically reduced by 3-8$\times$, confirming the regularization effect on the spectral norms of the (input space) Jacobians.

1 Introduction

Multi-step reasoning enhances language model capabilities by generating intermediate steps before producing a final answer. Specifically, for an input $\mathbf{x}$, a model f_W generates a reasoning sequence $(\mathbf{x}, \hat{y}^{(1)}, \ldots, \hat{y}^{(T)})$ autoregressively. As the error in previous outputs propagates over the steps, this can destabilize long reasoning and cause severe final prediction errors [38, 28]. However, standard training like supervised fine-tuning does not explicitly control this error accumulation. In this work, we theoretically analyze such error amplification across steps, aiming to identify the key factors governing reasoning stability and design training methods that control accumulated errors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) Evaluation of our bound values (b) Inference losses (c) Spectral norm of Jacobians

Figure 1: We present a bound on the inference loss in multi-step reasoning governed by the product of spectral norms of Jacobians taken through the input space across generation steps. Figure 1a: Our bound (equation (4)) tracks the inference loss on the Bellman-Ford algorithmic reasoning task, showing that both can scale exponentially with steps. Figure 1b: We design a training method that regularizes the Jacobian spectral norms, which improves the test performance over baselines. Figure 1c: Models fine-tuned with our method exhibit drastically reduced Jacobian spectral norms.

Understanding such error amplification is valuable in many practical problems, especially for scaling test-time computation [6, 8]. One concrete setting is algorithmic reasoning, where models perform the step-by-step computation of classical algorithms like Bellman-Ford [20]. Models are expected to learn these underlying algorithms to generalize and extrapolate to larger, unseen inputs. However, a mistake in the early steps can corrupt the procedure. In our own experiments, we observe that the test error in a fine-tuned language model (e.g., Qwen-1.5B and Gemma-2B) can scale super-linearly with steps, as shown in Figure 1a. Empirical methods, such as training models to generate fewer steps [5] or reason in the continuous feature space [10], have been proposed to reduce these errors. Yet, the reason for their empirical benefits remains underexplored.

Recent theoretical work has presented convergence and generalization guarantees of transformers trained on specific reasoning tasks, including linear regression [12, 40], state tracking [14], and path-finding in trees [34]. Their analyses focus on the task-specific transformer structures, leaving open the general cases. Studies based on PAC learning theory [1, 15] yield VC-dimension generalization bounds dependent on the entire hypothesis class. This motivates us to conduct a general analysis of multi-step reasoning with computable measures on data and fine-tuned models.

We begin by analyzing the inference error amplification of multi-step reasoning. Our key finding is that the inference loss is governed by the products of the *spectral norms of Jacobians taken through the input space* across generation steps, which control reasoning stability. The idea is to bound the gap between the T-step inference loss, $\ell(f_W([\mathbf{x}, \hat{y}^{(1)}, \ldots, \hat{y}^{(T-1)}]), y^{(T)})$, and the training loss, $\ell(f_W([\mathbf{x}, y^{(1)}, \ldots, y^{(T-1)}]), y^{(T)})$, conditioned on the correct steps. Using a first-order approximation in the input space, we show that the prediction errors at step i are propagated forward by the Jacobian matrix $J_i^{(t)}$ (evaluated at the t-th step output to the input $y^{(i)}$). Let $\rho_i^{(T)}$ be the sum of products of $\|J_i^{(t)}\|$ across all paths from step i to T. Then, recursively decomposing the errors yields a bound for the inference loss governed by $\sum_{i=1}^{T-1} \rho_i^{(T)}$ (cf. Lemma 3.1), which we define as an error amplification factor. Empirically, this bound closely tracks the test losses on fine-tuned language models (including Qwen, Gemma, and Llama models), which can scale exponentially with T, as shown in Figure 1a.

Next, we instantiate this bound by analyzing one-layer multi-head transformers trained on weight-prediction tasks for linear and quadratic functions. In both settings, we prove that the transformer converges to a solution where $\sum_{i=1}^{T-1} \rho_i^{(T)}$ decays with T, with the spectral norm of the Jacobians evaluated at each step input strictly less than one. Consequently, the trained transformer achieves a test loss of $O(1/\text{poly}(d))$ with input dimension d and $T = \Theta(\log d)$ steps (cf. Theorem 3.4). Extending prior work on linear functions [12], our analysis for quadratic functions introduces a novel three-head transformer construction that exactly implements gradient descent (cf. Lemma 3.3), a finding that may be of independent interest.

Motivated by this analysis, we propose a training method to control accumulated error by regularizing the Jacobian-based error amplification factor. Our method incorporates chain-of-thought length compression that downsamples the reasoning steps during training, and quantization-aware training

[18, 19] that trains weights with gradients computed on the quantized model. We show that our method reduces the Jacobian spectral norms by regularizing the Hessian trace of the model loss during quantized training. We evaluate this method on five algorithmic reasoning tasks [20] and two state-tracking tasks from LEGO [39]. First, across all seven tasks, our method yields an average test accuracy improvement of **3.5**% over baselines, including implicit chain-of-thought [5] and Coconut [10], shown in Figure 1b. Second, on evaluation of length generalization with larger inputs, our method enhances the test performance by **8.2**% over baselines. Finally, ablation studies confirm that our method reduces the Jacobian-based error amplification factor by up to **8**× compared to existing methods, as shown in Figure 1c. The code for replicating these empirical findings is available at https://github.com/VirtuosoResearch/Multi-step-reasoning-experiments.

Summary of contributions. First, our analysis reveals that the inference error in multi-step reasoning scales with the product of spectral norms of Jacobians taken through the input space. Second, we theoretically show that transformers trained on linear and quadratic functions converge to a regime where these norms are strictly less than one, yielding near-zero inference error. Finally, we propose a training method that regularizes these input Jacobian norms, empirically enhancing test performance and length generalization across diverse reasoning tasks.

2 Preliminaries

We now describe the problem setup for multi-step reasoning. Let $\mathcal{X}$ denote the feature space. Let $f_W : \mathcal{X}^* \to \mathcal{X}$ denote a next-token predictor with parameters $W \in \mathbb{R}^p$, such as a language model. f_W takes a sequence of vectors in $\mathcal{X}$ of arbitrary length as input and outputs a single vector in $\mathcal{X}$, which is then appended to the sequence before the next prediction during inference.

For an input sequence $\mathbf{x} \in \mathcal{X}^*$, we assume that there exists an underlying rule generating a ground-truth sequence $[\mathbf{x}, y^{(1)}, \ldots, y^{(T(\mathbf{x}))}]$, where $y^{(t)} \in \mathcal{X}$ is the ground-truth output at step t, and $T(\mathbf{x})$ denotes the total number of steps, which depends on the input $\mathbf{x}$. We denote the maximum number of steps as T for a particular task and bounded input lengths. As a concrete example, suppose $\mathbf{x}$ contains a graph and a source node; then each $y^{(i)}$ encodes an intermediate execution step of a graph traversal algorithm such as breadth-first search [20].

At inference time, the model f_W generates $T(\mathbf{x})$ steps for an input $\mathbf{x}$ autoregressively. We denote the prediction at step t as

$$\widehat{y}^{(t)} = f_W\big([\mathbf{x}, \widehat{y}^{(1)}, \ldots, \widehat{y}^{(t-1)}]\big), \text{ with } \widehat{y}^{(1)} = f_W(\mathbf{x}).$$

For simplicity of notation, we write the generated sequence after t steps as

$$\widehat{\mathbf{z}}^{(t)} = [\mathbf{x}, \widehat{y}^{(1)}, \ldots, \widehat{y}^{(t)}], \text{ with } \widehat{\mathbf{z}}^{(0)} = [\mathbf{x}].$$

The corresponding ground-truth sequence with the correct intermediate steps is denoted as

$$\mathbf{z}^{(t)} = [\mathbf{x}, y^{(1)}, \ldots, y^{(t)}], \text{ for } t = 1, \ldots, T(\mathbf{x}), \text{ and } \mathbf{z}^{(0)} = \mathbf{x}.$$

During training, the model f_W predicts the next step conditioned on the preceding steps that encode the ground truth. Let $S = \big\{(\mathbf{x}_i, y_i^{(1)}, \ldots, y_i^{(T(\mathbf{x}_i))})\big\}_{i=1}^n$ denote a training dataset of n examples, where every $\mathbf{x}_i$ is drawn independently from an unknown distribution $\mathcal{D}$. Given a loss function $\ell(\cdot, \cdot)$, the training loss is defined as:

$$\hat{L}(f_W; S) = \frac{1}{n} \sum_{i=1}^{n} \sum_{t=0}^{T(\mathbf{x}_i)-1} \ell\big(f_W\big(\mathbf{z}_i^{(t)}\big), y_i^{(t+1)}\big). \tag{1}$$

At test time, the model predicts the next step conditioned on previous predictions. The outcome is evaluated by the loss function at the final step. We thus define this test loss over a random sample $\mathbf{x}$ drawn from the data distribution $\mathcal{D}$ as:

$$L(f_W; \mathcal{D}) = \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \Big[\ell\big(f_W(\widehat{\mathbf{z}}^{(T(\mathbf{x})-1)}), y^{(T(\mathbf{x}))}\big)\Big]. \tag{2}$$

To be clear, we will call the above equation (2) the *inference loss*. At training time, minimizing equation (1) measures the loss relative to the ground truth. At inference time, the model assimilates

prediction errors that occurred during previous steps back into the model. Thus, the prediction errors during inference can accumulate during long generation steps.

Problem statement. We quantify how inference loss accumulates throughout generation. We study conditions under which multi-step reasoning is stable, where the inference loss $L(f_W; \mathcal{D})$ remains bounded in long chains. Additionally, we design training algorithms to mitigate error accumulation and improve the stability of multi-step reasoning.

Notations. For any matrix X, let $\|X\|$ denote the largest singular value (or spectral norm) of X. Let $\|X\|_F$ denote the Frobenius norm of X. $f(N) \lesssim g(N)$ indicates that there exists a fixed constant c that does not grow with N such that $f(N) \leq c \cdot g(N)$ for large enough integers of N.

3 Theoretical Analysis

This section presents a theoretical analysis of multi-step reasoning in the above setup. First, we present a bound on the inference loss relative to the ground truth generation. We show that this bound depends on an *error amplification factor* defined by the product of spectral norms of Jacobian matrices measured at the input space. Second, we instantiate this bound in in-context prediction tasks for linear and quadratic functions. We show that transformer models converge to a regime where the error amplification factor decays across steps, thus yielding nearly zero inference loss.

3.1 Error Amplification in Multi-Step Reasoning

We now quantify how the inference loss accumulates across generation steps. We write the number of steps as T when the input is clear from the context. For an input $\mathbf{x}$, we compare the inference loss $\ell\big(f_W(\widehat{\mathbf{z}}^{(T-1)}), y^{(T)}\big)$ of equation (2) with the loss $\ell\big(f_W(\mathbf{z}^{(T-1)}), y^{(T)}\big)$ of the prediction conditioned on the correct steps, which is what training minimizes in equation (1).

To measure the difference, let $e^{(t)}(\mathbf{x}) = f_W(\mathbf{z}^{(t-1)}) - y^{(t)}$ and $\widehat{e}^{(t)}(\mathbf{x}) = f_W(\widehat{\mathbf{z}}^{(t-1)}) - y^{(t)}$ denote the prediction error at step t conditioned on the correct steps and on the model's generations, respectively. Consider the first-order Taylor expansion in the input space, for every t,

$$f_W(\widehat{\mathbf{z}}^{(t-1)}) - f_W(\mathbf{z}^{(t-1)}) \;=\; \sum_{i=1}^{t-1} J_i^{(t)}(\mathbf{x})\, \widehat{e}^{(i)}(\mathbf{x}) \;+\; \mathcal{R}^{(t)}, \text{ where } J_i^{(t)}(\mathbf{x}) = \frac{\partial f_W(\mathbf{z}^{(t-1)})}{\partial y^{(i)}}$$

is the input Jacobian evaluated at step t, and $\mathcal{R}^{(t)}$ is the second-order remainder error term. By recursively unrolling this expansion, an error at step i affects a later step t by first propagating to an intermediate step k and then from k to t. To capture the propagation path, we define a term:

$$\rho_i^{(t)} = \sup_{\mathbf{x}\in\mathcal{X}^*} \|J_i^{(t)}(\mathbf{x})\| + \sum_{k=i+1}^{t-1} \left(\sup_{\mathbf{x}\in\mathcal{X}^*} \|J_k^{(t)}(\mathbf{x})\| \cdot \rho_i^{(k)} \right), \text{ with } \rho_t^{(t)} = 1,$$

which sums, over all paths from i to t, the product of the spectral norms of the input Jacobians along the path. Therefore, an error at step t reaches the final step amplified by a factor $\rho_t^{(T)}$.

Our main result is to incorporate the spectral norm of Jacobians to quantify the accumulation of inference errors across generation steps. For every $\mathbf{x}$ in the support of $\mathcal{D}$, we denote the maximum of the two errors, for t from 1 to $T-1$, by:

$$\sup_{\mathbf{x}\in\mathcal{X}^*} \|e^{(t)}(\mathbf{x})\| = \varepsilon_t, \;\; \sup_{\mathbf{x}\in\mathcal{X}^*} \|\widehat{e}^{(t)}(\mathbf{x})\| = r_t. \tag{3}$$

We present a bound on the (expected) inference loss as follows.

Lemma 3.1. *Suppose that f_W is twice differentiable with respect to its input, the loss function $\ell(\cdot,\cdot)$ is C_1-Lipschitz continuous in its first argument, and the Jacobian of f_W with respect to its input is C_2-Lipschitz continuous. Let $L^\star(f_W; \mathcal{D}) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\left[\ell\big(f_W(\mathbf{z}^{(T-1)}, y^{(T)})\big)\right]$ be the expected loss at the final step of the model prediction conditioned on correct steps. Then, for any fixed $W \in \mathbb{R}^p$ that satisfies these conditions, and under equation (3), the inference loss is bounded away from the expected loss as follows:*

$$\big|L(f_W; \mathcal{D}) - L^\star(f_W; \mathcal{D})\big| \leq C_1 \sum_{t=1}^{T-1} \rho_t^{(T)} \varepsilon_t + \frac{C_1 C_2}{2} \sum_{t=2}^{T} \rho_t^{(T)} \left(\sum_{i=1}^{t-1} r_i \right)^2. \tag{4}$$

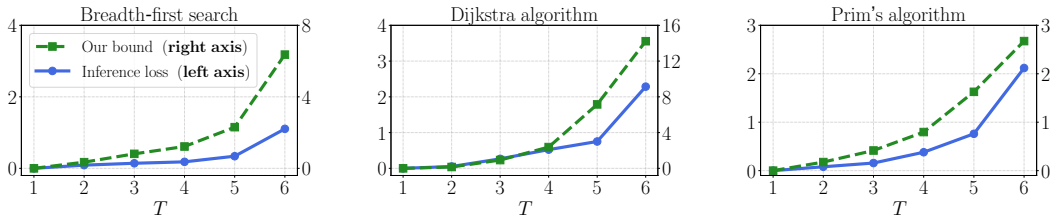


Figure 2: We illustrate the loss gap and our bound in equation (4), which can scale exponentially across generation steps, tracking the growth of the inference loss. Results are evaluated on breadth-first search, Dijkstra, and Prim's algorithm, using QWen-1.5B. The loss gap and the bound are plotted on separate axes; equation (4) is a worst-case bound and is numerically larger than the gap it bounds. Table 2 in Appendix B reports the corresponding bound-to-gap ratios and constants.

The proof involves a decomposition of the error between predictions on the model-generated sequence and the correct step sequence by applying the first-order expansion recursively. The full proof is in Appendix A.1. Lemma 3.1 highlights that the inference loss is governed by $\sum_{i=1}^{T-1} \rho_i^{(T)}$, which we term an error amplification factor. Even if the training error is small, this factor can grow exponentially with T, resulting in large inference errors on long reasoning sequences.

Examples. To empirically verify our analysis, we study the scaling of inference losses on graph algorithmic reasoning tasks [20]. We fine-tune Qwen-1.5B, Llama-1B, and Gemma-2B models on tasks with graphs of 10 nodes. Inputs are text-based graph descriptions, and intermediate steps are encoded as node labels, such as the nodes along the shortest path. As a reasoning step in text-based datasets typically involves several tokens, we compute the loss, the ℓ_2 error, and the Jacobian matrices token-by-token within a step, and then aggregate them over the tokens within one step.

For text-based reasoning tasks, we view $\mathcal{X}$ as the continuous space of input embeddings. The output of f_W at step t is then the expected embedding of the predicted next token, calculated as the weighted average of all token embeddings based on their predicted softmax probabilities. We compute the Jacobian of the output vector with respect to the input embeddings. We estimate its spectral norm via power iteration using Jacobian-vector products. To approximate $\sup_{\mathbf{x}} \|J_k^{(t)}\|$ in equation (4), we take the maximum norm across the dataset.

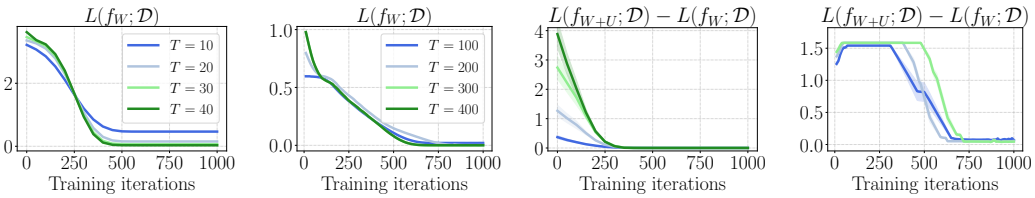
After minimizing the training loss, we measure the loss gap and the empirical values of our bound in equation (4), for T ranging from 1 to 6. In Figure 2, the inference loss increases significantly with the number of steps. Our bound, which involves $\sum_{i=1}^{T-1} \rho_i^{(T)}$, scales exponentially with T, closely tracking the growth in inference loss. We also observe similar results on Llama-1B and Gemma-2B models, which are described in Appendix B. These results demonstrate that controlling input Jacobian norms during training is crucial for reducing accumulated errors in multi-step generation.

Remark 3.2. Lemma 3.1 provides a worst-case upper bound that may not be numerically tight. Empirically, while we observe a consistent correlation between the bound values and inference losses across models, the numerical scales of the error amplification factor $\sum_{i=1}^{T-1} \rho_i^{(T)}$ vary significantly. The bound-to-gap ratio is around 8 for Qwen-1.5B and around 10^5–10^6 for Llama-1B and Gemma-2B. We report the empirical loss gap, the bound, the bound-to-gap ratio, both terms of equation (4), and the estimated constants C_1 and C_2, for every model in Appendix B.

3.2 Learning Dynamics in Quadratic Models

Next, we explicitly instantiate the bound in Lemma 3.1 in training transformers on an in-context weight prediction task. We show that if the spectral norm of the input Jacobian matrix is less than one, the inference loss can be reduced close to zero.

Based on prior work on in-context learning [40], we study a task that predicts the weight vector $\theta^\star \sim \mathcal{N}(0, \mathrm{Id}_d)$ of a function, given n examples (x_i, y_i) where $x_i \sim \mathcal{N}(0, \mathrm{Id}_d)$ and $y_i = g_{\theta^\star}(x_i)$ for a known function g. We define the ground-truth steps as iterates by performing gradient descent



(a) Linear functions (b) Quadratic functions (c) Linear functions (d) Quadratic functions

Figure 3: We illustrate that, in training transformers on both linear and quadratic functions, the inference loss and the noise-perturbed loss converge close to zero, and become lower with longer steps. We set $d = 10$ and train a one-head linear transformer on linear functions ($n = 20$, $\eta = 0.4$) and a three-head non-linear transformer on quadratic functions ($n = 200$, $\eta = 10^{-3}$). For the random Gaussian noise, we set $\sigma = 0.002$. On the right, we consider a slightly different setting by first adding noise to the weights before computing the gradients.

on the regression examples

$$\theta_{i+1} = \theta_i - \frac{\eta}{n} \sum_{k=1}^{n} \nabla_\theta g_{\theta_i}(x_k, y_k),$$

with a learning rate η and an initialization θ_0.

We consider a one-layer transformer with h non-linear self-attention heads:

$$f_W(Z) = Z_{[:,-1]} + \sum_{i=1}^{h} W^{PV_i} Z \cdot \phi_i\Big(\frac{Z^\top W^{KQ_i} Z_{[:,-1]}}{n}\Big), \text{ and } Z = \begin{bmatrix} X & 0_d & \cdots & 0_d \\ y & 0 & \cdots & 0 \\ 0_{d\times n} & \theta_0 & \cdots & \theta_i \\ 0 & 1 & \cdots & 1 \end{bmatrix}, \quad (5)$$

where $Z \in \mathbb{R}^{(2d+2)\times(n+i+1)}$ is the input embedding, $X = [x_1, \ldots, x_n]$, and $y = [y_1, \ldots, y_n]$. The ones in the bottom are appended for formatting convenience. For step T, the last column contains the target vector $\theta^\star$. Following prior work [12, 14, 34], W^{PV_i} and W^{KQ_i} are the merged projection-value and key-query matrices, ϕ_i is a non-linear function, and $Z_{[:,-1]}$ denotes the last column. We use the squared loss for this setting.

Numerical example. We validate that the transformer can achieve near-zero inference loss in this setting, for both linear and quadratic function classes. The results are shown in Figure 3. The inference loss converges to near-zero, with longer steps yielding lower inference loss.

We describe our main results. For linear functions $y = \theta^\top X$, we prove that the model exhibits inference loss of $O(1/\text{poly(d)})$ with $T = \Theta(\log d)$. This is because the transformer converges to a solution where $J_{i-1}^{(i)} \approx I - \frac{\eta}{n} XX^\top$ and $\rho_i^{(T)} \approx \|I - \frac{\eta}{n} XX^\top\|^{T-i}$. The accumulated errors form a converging geometric series. We present the complete proof in Appendix A.2.

Next, we present results on quadratic functions $y = (\theta^\top X)^{\odot 2}$, which represent a fundamental yet tractable setting for studying gradient descent dynamics [21, 26]. We first establish that a three-head transformer can exactly implement the gradient update step using cubic and squared activations.

Lemma 3.3. *There exist weight matrices $\{W^{PV_h}, W^{KQ_h}\}_{h=1}^3$ and elementwise non-linear functions $\{\phi_h\}_{h=1}^3$, such that a three-head transformer $f_W(\widehat{Z}_{T-1})$ outputs the T-step gradient descent solution with a learning rate η on the quadratic functions.*

We then show that the transformer model converges to this construction during training, yielding near-zero inference loss.

Theorem 3.4. *Let $\kappa \in (0, 1/24]$, $\mu \in (0, 4]$, and $\alpha_0 \in [3/4, 1)$ be the constants. Suppose $n = \Theta(d^2 \log^5 d)$, $\eta = \kappa/d$, $\theta_0 = \alpha_0 \theta^\star$, and $T = \lceil C \log d \rceil$ for some $C > 0$ with $aC > 1$ for $a = \log\big(\frac{1}{1-\mu\eta d}\big)$. There exists an initialization such that if we run gradient flow on the training loss, then after time $t = O\big(\log d + \log \frac{1}{\epsilon_0}\big)$, for any $\epsilon_0 \geq \Theta\big(\frac{\log d}{d^{2aC-1}}\big)$, the trained transformer satisfies $L(f_W; \mathcal{D}) \leq \epsilon_1$ for any $\epsilon_1 \in \big(\Theta(d^{-2aC+1} \log d),\ 1\big)$.*

The proof tracks an ordinary differential equation for the eigenvalues of the weight matrices. In the converged solution, we have

$$J_{i-1}^{(i)} \approx I - \frac{2\eta}{n} X D(\theta_i) X^\top \text{ and } \rho_i^{(T)} \approx \|I - \frac{2\eta}{n} X D(\theta_i) X^\top\|^{T-i},$$

where $D(\theta_i) = \text{diag}(3(X^\top \theta_i)^{\odot 2} - (X^\top \theta^\star)^{\odot 2})$. Then, the accumulated errors again form a converging geometric series. We present the complete proof in Appendix A.3.

Comparison with prior work. Prior analyses of transformers trained on multi-step reasoning differ from ours in their attention mechanisms, tasks, and tracked quantities. Previous works have examined one-head softmax attention on state-tracking tasks to demonstrate implicit biases and length generalization [13, 14]. Huang et al. [12] study one-head linear attention on in-context linear regression, where the trained transformer implements multi-step gradient descent. Yang et al. [34] study multi-head softmax attention on tree path-finding, where different heads specialize and coordinate across subtasks. In contrast, we study multi-head non-linear attention on quadratic regression. We identify the explicit form of Jacobians taken through the inputs at each step at convergence and connect it with the stability of the inference loss.

4 Proposed Methods

Our analysis suggests two strategies to mitigate accumulated errors: reducing the sequence length to limit the error amplification factor $\sum_{t=1}^{T-1} \rho_t^{(T)}$ and regularizing the input Jacobian spectral norms $\|J_i^{(t)}\|$. Motivated by these insights, we design a training method combining two techniques.

(1) Chain-of-thought length compression. First, we compress the reasoning sequence by subsampling intermediate steps. For a training example $(\mathbf{x}, y^{(1)}, \ldots, y^{(T)})$ and a sampling factor $\lambda \in (0, 1]$, we uniformly sample $T' = \lfloor \lambda \cdot T \rfloor$ intermediate steps to form a compressed sequence $(\mathbf{x}, y^{(1)}, \ldots, y^{(T')})$, and minimize the training loss over the sampled sequence.

As shown in Figures 4a and 4b, models trained on subsampled sequences consistently outperform those trained on full steps. Intermediate numbers of steps achieve the lowest inference loss on the Dijkstra and Prim tasks. Notably, subsampling steps reduces the error amplification factor by 92× and 181× compared to training with all steps. Subsampling steps reduces the amplification factor but increases the per-step errors ε_t. Thus, the best λ is the one that balances these two effects.

(2) Regularization through quantization-aware training. Second, we incorporate quantization during supervised fine-tuning to explicitly regularize the Jacobian spectral norms. This is inspired by observations that quantized training smooths the loss landscape and provides a regularization effect [18, 19]. During the forward pass at training step t, we apply a standard uniform quantization function to the full-precision weights $W^{(t)}$:

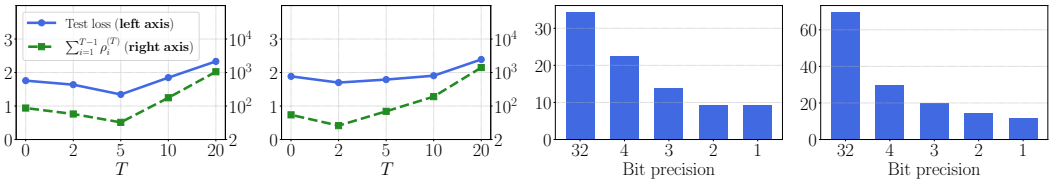
$$Q(W^{(t)}) = \Delta \cdot \text{Round}\left(\text{Clamp}(W^{(t)} - b, W_{min}, W_{max})/\Delta\right) + b,$$

where Δ and b are bit-width-dependent scaling and bias parameters. We use the straight-through estimator [2] to approximate the gradient $\nabla_W \hat{L}(f_{Q(W)}) \approx \nabla_{Q(W)} \hat{L}(f_{Q(W)})$.

We hypothesize that quantization-aware training acts as a regularizer on the error amplification factor. Under the Pseudo-Quantization Noise model [31, 9], the quantized model loss can be viewed as the expected loss under mean-zero randomized-rounding noise. Minimizing this loss penalizes the trace of the loss Hessian, which can be decomposed into the Jacobians taken through the input space.

Figures 4c and 4d demonstrate this regularization effect. Training with lower bit-widths yields a lower error amplification factor compared to full precision. When combining 1-bit quantization, $\sum_{t=1}^{T-1} \rho_t^{(T)}$ is further reduced by 3.6× and 5.8×, compared to full precision, on the Dijkstra and Prim algorithmic tasks, respectively. We also observe that, while quantization increases the prediction error at each step, it reduces the error amplification factor much more significantly, thus leading to improved performance. We report evaluations under various bit-widths in Appendix B. Taken together, we summarize our approach in Algorithm 1.

Remark 4.1 (Length generalization). For out-of-distribution settings, the test error bound from Lemma 3.1 can increase in two aspects. First, the per-step prediction errors grow due to covariate



(a) Dijkstra (b) Prim's algorithm (c) Dijkstra (d) Prim's algorithm

Figure 4: Figure 4a and 4b: We show that the models trained with reduced intermediate steps yield lower inference losses than the model trained with longer steps. Correspondingly, the models exhibit lower Jacobian-based error amplification factors. Figure 4c and 4d: We show that quantization-aware training with lower bit-widths further reduces the Jacobian-based error amplification factors, yielding lower inference losses.

Algorithm 1 Jacobian-Regularized Training through Length Compression and Quantization

Input: n training examples with correct sequences $(\mathbf{x}_i, y_i^{(1)}, \ldots, y_i^{(T)})$ of length T
Require: Pretrained language model $f_{W^{(0)}}$, number of epochs S, learning rate η, a sampling factor λ, quantization function $Q(\cdot)$
Output: Fine-tuned quantized model $f_{Q(W^{(S)})}$

1: **for** $t = 0, 1, \ldots, S-1$ **do**
2: $(y_i^{(1)}, \ldots, y_i^{(T')}) \leftarrow$ Subsample $T' = \lfloor \lambda \cdot T \rfloor$ steps from example i, for $i = 1, \ldots, n$
3: $\hat{L}(f_{Q(W^{(t)})}) \leftarrow$ Compute the training loss of $f_{Q(W^{(t)})}$ on the sampled sequences
4: $W^{(t+1)} \leftarrow W^{(t)} - \eta \cdot \nabla_{Q(W^{(t)})} \hat{L}(f_{Q(W^{(t)})})$
5: **end for**
6: **return** $f_{Q(W^{(S)})}$

shift. Second, the error amplification factor accumulates over the additional steps. If the step-to-step Jacobian norm exceeds one, the extra steps multiply the error geometrically, making our regularization approach crucial. Designing training algorithms for length generalization is an interesting direction for future work, and our analysis lays a concrete direction for this.

5 Experiments

We evaluate our proposed method in a range of algorithmic reasoning tasks. First, our method outperforms baselines by an average of **3.5**% across five graph algorithmic and two symbolic reasoning tasks. Second, our method significantly enhances out-of-distribution length generalization, yielding an average **8.2**% improvement. Third, empirical analysis confirms that our approach reduces the error amplification factor (measured via Jacobian spectral norms) by up to **8**×. Lastly, we also conduct ablation studies for various design choices of our algorithm.

5.1 Experimental Setup

Datasets and models. First, we evaluate our algorithm on graph algorithmic tasks from the CLRS-Text benchmark [20], which encodes graphs as flattened adjacency lists and represents the intermediate steps as lists of node labels. We use five classic graph algorithms, including Dijkstra, Bellman-Ford, Prim's algorithm, breadth-first search, and depth-first search. For each task, we use 2,000 training examples generated on random graphs of 10 nodes and 200 examples for validation and testing. Second, we evaluate on symbolic state-tracking reasoning tasks, namely LEGO [39]. We use two task variants: transitive (cyclic) and symmetry group actions. For each task, we use a dataset of 2,000 training examples with $k = 8$ variables, and 200 examples each for validation and testing. We evaluate our method in fine-tuning the QWen-1.5B models.

Baselines. We compare our method with three categories of baselines. First, we consider standard fine-tuning baselines that train without or with the provided chain-of-thought sequences, namely No CoT and SFT-CoT. Second, we consider existing chain-of-thought methods. This includes implicit

Table 1: We report the test accuracy (%) of our method, as compared to supervised fine-tuning, implicit reasoning, and random noise injection. We evaluate five algorithmic reasoning tasks and two symbolic state-tracking reasoning tasks. For length generalization, models are evaluated on inputs 10% longer than the training sequences. All results are averaged over three random seeds.

	Dijkstra	Bellman-Ford	Prim	BFS	DFS	Cyclic	Symmetry
No CoT	53.3 ± 2.0	69.9 ± 0.3	56.2 ± 0.1	85.9 ± 0.1	71.4 ± 1.4	66.8 ± 0.9	48.5 ± 3.1
SFT-CoT	73.1 ± 1.4	75.3 ± 1.5	62.3 ± 0.1	86.0 ± 0.4	54.6 ± 2.0	89.2 ± 1.2	55.8 ± 1.5
Implicit CoT	71.6 ± 0.5	72.3 ± 0.6	64.1 ± 0.3	85.5 ± 1.7	72.4 ± 0.4	75.4 ± 1.0	60.8 ± 3.6
Coconut	74.1 ± 0.3	75.6 ± 0.5	63.5 ± 1.2	85.7 ± 0.3	72.6 ± 0.7	79.5 ± 0.1	53.9 ± 0.5
NSO	74.5 ± 0.8	75.3 ± 0.0	63.2 ± 0.3	86.5 ± 0.8	55.8 ± 0.1	89.3 ± 0.8	57.2 ± 0.9
Algorithm 1	**76.2** ± 1.7	**80.3** ± 0.1	**66.2** ± 0.3	**88.3** ± 1.3	**74.4** ± 1.5	**94.1** ± 1.3	**62.2** ± 0.8
Length generalization	Dijkstra	Bellman-Ford	Prim	BFS	DFS	Cyclic	Symmetry
No CoT	29.8 ± 0.4	42.9 ± 0.6	37.0 ± 0.5	48.5 ± 0.2	23.0 ± 0.5	55.4 ± 0.5	31.2 ± 0.7
SFT-CoT	42.1 ± 0.4	47.2 ± 0.8	34.1 ± 0.6	56.2 ± 0.6	21.5 ± 0.2	80.0 ± 0.4	48.8 ± 0.3
Implicit CoT	33.7 ± 0.3	46.5 ± 0.5	39.3 ± 0.4	54.6 ± 0.5	30.0 ± 0.3	82.5 ± 0.8	53.6 ± 1.0
Coconut	38.2 ± 0.6	38.9 ± 0.4	37.6 ± 0.9	53.9 ± 0.9	32.6 ± 0.4	80.4 ± 0.5	49.4 ± 0.2
NSO	36.4 ± 0.5	44.1 ± 0.8	38.4 ± 0.3	54.8 ± 0.2	28.2 ± 0.2	82.4 ± 0.4	51.2 ± 0.9
Algorithm 1	**44.3** ± 0.8	**50.1** ± 0.1	**42.5** ± 0.6	**59.6** ± 0.6	**35.4** ± 0.8	**87.2** ± 0.2	**59.1** ± 0.7

CoT [5] that trains a model to directly predict the final output by gradually removing intermediate steps during training, and the chain of continuous thought (Coconut) [10] that trains a model to reason in a continuous latent space by using the hidden state as the next input embedding. Third, we consider noise stability optimization (NSO) [37] that regularizes the Hessian trace through random Gaussian noise injection. For each baseline, we evaluate the test accuracy on the final answer.

Implementations. Our algorithm includes key hyperparameters: the sampling factor λ and the quantization bit-width. We vary λ across 0.1, 0.2, 0.5, and 1. The bit-width is varied across 1, 2, 3, and 4 bits. We use the quantization methods from existing methods [19]. For the base training method, we use LoRA fine-tuning with a rank between 16 and 64. The training uses the AdamW optimizer with a learning rate in the range of 1×10^{-5} to 4×10^{-5}. We report the hyperparameters used for each result in Appendix B.

For the length generalization experiment, we additionally evaluate at 20% longer inputs on all five CLRS-Text tasks. Absolute accuracy degrades sharply for every method, falling below 25%. While our method still leads, at this accuracy level, the input-length bottleneck dominates. We therefore report the 10% setting as our main result. Further improving the length generalization on longer sequences is left for future work.

5.2 Experimental Results

Improved multi-step reasoning. Table 1 details the test accuracy of our fine-tuned models across seven reasoning datasets. Our method consistently outperforms all baselines, achieving an average improvement of **3.5**% across five graph algorithmic and two symbolic state-tracking tasks. Specifically, by combining sequence compression with quantization-aware training, our algorithm surpasses latent-reasoning baselines (implicit reasoning and Coconut) by up to 6.5%.

Enhanced length generalization. Next, we evaluate the trained models on examples with 10% longer input length than training examples in the lower half of Table 1. Our approach consistently exceeds baseline performance by an average of **8.2**%, demonstrating its effectiveness in suppressing error amplification during extended generation.

Reduction of the error amplification factor. Finally, to explain these performance gains, we compare the empirical error amplification factors across methods. Our algorithm drastically reduces this factor by **3**–**8**$\times$ compared to baselines, validating our method's regularization on Jacobian spectral norms. Quantitatively, on Qwen-1.5B across the three tasks in Figure 1c, our method reduces $\sum_{i=1}^{T-1} \rho_i^{(T)}$ by 43% relative to baselines, with a 72% average reduction in the inference loss.

We further fine-tune Llama-1B and Gemma-2B on Bellman-Ford with SFT-CoT, Coconut, and Algorithm 1, and measure the full decomposition of equation (4) for each. We also find that the ordering of methods by bound value matches their ordering by empirical loss gap on both models. Relative to Coconut, our method reduces the bound by 77% and the empirical test loss gap by 58%

on average. The absolute scale of the bound differs sharply between architectures. We report the full results in Table 4 of Appendix B.

Ablation studies. We evaluate the individual components of our method through ablation studies. First, removing either length compression or quantization-aware training degrades performance by 4.3% and 3.2%, confirming both are essential. Second, we compare intermediate step sampling strategies for length compression. We consider random sampling, removing the first, last, or equally spaced steps. Among them, random sampling yields slightly better performance and is thus adopted. Finally, we find that injecting quantization noise during training noticeably outperforms standard random Gaussian noise by 9%.

6 Related Work

Theoretical understanding of multi-step reasoning. Studying the training dynamics of transformers is key to understanding how multi-step reasoning improves generalization. Prior work analyzes single-head transformers trained on parity-check tasks [13, 30], and state tracking tasks [14], while [34] studies multi-head specialization on path-finding in binary trees. For the in-context weight prediction task, [12] establishes convergence for linear targets using single-head linear transformers. We extend the analysis to nonlinear multi-head transformers and quadratic functions. Complementary theoretical analyses include entropy-calibration analyses of error accumulation in autoregressive generation [4] and information-theoretic frameworks that quantify per-step information gain in multi-step reasoning [28].

Joshi et al. [15] study learning general function classes of next-token generators with CoTs in the realizable setting, showing that the sample complexity scales with the VC dimension of the hypothesis class and $\log(T)$ of the CoT length T. Altabaa et al. [1] further improve the bound with a notion of CoT-information, which achieves faster rates than applying uniform convergence to training loss on chain-of-thought sequences.

Prior work has also examined the expressiveness of transformers with chain-of-thought. Li et al. [22] show that CoT enables transformers to perform computations beyond the expressiveness limits of models without CoT. Expressiveness has been further characterized through circuit complexity under logarithmic, linear, and polynomial generation lengths [25]. Increasing CoT length has been shown to outperform parallel scaling of multiple chains on graph connectivity tasks [27].

Generalization of language models. Recent work derives generalization bounds for LLMs via compression and subspace training [23, 24], with tighter martingale-based extensions that explain empirical pretraining scaling laws [7]. A separate line of work derives PAC-Bayes bounds for fine-tuning that depend on the loss Hessian [17, 16].

Optimization methods for multi-step reasoning. Several recent works aim to improve multi-step reasoning beyond supervised fine-tuning. Yao et al. [35] frame this process as a latent-variable problem, minimizing gradient variance to efficiently allocate sampling budgets. A complementary line of work explores implicit reasoning. Implicit chain-of-thought [5] gradually removes intermediate steps to internalize computation. Continuous chain-of-thought [10] shifts reasoning into a continuous latent space. Semantic chain-of-thought [11] aligns implicit and explicit steps via contrastive learning. Empirically, Yeo et al. [36] investigate how long reasoning chains emerge in large language models, highlighting training computation and reward shaping as key factors. More broadly, multi-step reasoning traces also appear in experimental design settings [33, 32], where trajectory-level data is revealed over time.

7 Conclusion

This work provides an analysis of multi-step reasoning by bounding its inference errors via the product of input Jacobian spectral norms across generation steps. We prove that transformers trained for in-context weight prediction converge to a multi-head construction where this error amplification factor decays, achieving polynomially small test loss. We introduce a training method that explicitly controls this factor using reasoning length compression and quantization-aware training. Across seven algorithmic tasks, our approach consistently improves test accuracy and length generalization, with reduced empirical amplification factors in fine-tuned models.

Acknowledgement

This work received financial support from NSF award IIS-2412008 and a startup fund from Northeastern University. D. Li is supported in part by a JPMorgan Chase PhD Fellowship. Z. Zhang and M. Duan received additional financial support from Khoury PhD fellowships.

References

[1] A. Altabaa, O. Montasser, and J. Lafferty. "CoT Information: Improved Sample Complexity under Chain-of-Thought Supervision". In: *NeurIPS* (2025).

[2] Y. Bengio, N. Léonard, and A. Courville. "Estimating or propagating gradients through stochastic neurons for conditional computation". In: *arXiv preprint arXiv:1308.3432* (2013).

[3] E. J. Candes, X. Li, and M. Soltanolkotabi. "Phase retrieval via Wirtinger flow: Theory and algorithms". In: *IEEE Transactions on Information Theory* (2015).

[4] S. Cao, G. Valiant, and P. Liang. "On the entropy calibration of language models". In: *NeurIPS* (2025).

[5] Y. Deng, Y. Choi, and S. Shieber. "From explicit cot to implicit cot: Learning to internalize cot step by step". In: *arXiv preprint arXiv:2405.14838* (2024).

[6] R. Ehrlich, B. Brown, J. Juravsky, R. Clark, C. Ré, and A. Mirhoseini. "Codemonkeys: Scaling test-time compute for software engineering". In: *arXiv preprint arXiv:2501.14723* (2025).

[7] M. Finzi, S. Kapoor, D. Granziol, A. Gu, C. De Sa, J. Z. Kolter, and A. G. Wilson. "Compute-Optimal LLMs Provably Generalize Better With Scale". In: *ICLR* (2025).

[8] J. Geiping, S. McLeish, N. Jain, J. Kirchenbauer, S. Singh, B. R. Bartoldson, B. Kailkhura, A. Bhatele, and T. Goldstein. "Scaling up test-time compute with latent reasoning: A recurrent depth approach". In: *NeurIPS* (2025).

[9] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer. "A survey of quantization methods for efficient neural network inference". In: *Low-power computer vision*. 2022, pp. 291–326.

[10] S. Hao, S. Sukhbaatar, D. Su, X. Li, Z. Hu, J. Weston, and Y. Tian. "Training large language models to reason in a continuous latent space". In: *COLM* (2025).

[11] Y. He, W. Zheng, Y. Zhu, Z. Zheng, L. Su, S. Vasudevan, Q. Guo, L. Hong, and J. Li. "SemCoT: Accelerating Chain-of-Thought Reasoning through Semantically-Aligned Implicit Tokens". In: *NeurIPS* (2025).

[12] J. Huang, Z. Wang, and J. D. Lee. "Transformers learn to implement multi-step gradient descent with chain of thought". In: *ICLR* (2025).

[13] R. Huang, Y. Liang, and J. Yang. "How Transformers Learn Regular Language Recognition: A Theoretical Study on Training Dynamics and Implicit Bias". In: *ICML* (2025).

[14] Y. Huang, Z. Wen, A. Singh, Y. Chi, and Y. Chen. "Transformers provably learn chain-of-thought reasoning with length generalization". In: *NeurIPS* (2025).

[15] N. Joshi, G. Vardi, A. Block, S. Goel, Z. Li, T. Misiakiewicz, and N. Srebro. "A theory of learning with autoregressive chain of thought". In: *Annual Conference on Learning Theory (COLT)* (2025).

[16] H. Ju, D. Li, A. Sharma, and H. R. Zhang. "Generalization in graph neural networks: Improved pac-bayesian bounds on graph diffusion". In: *International conference on artificial intelligence and statistics*. PMLR. 2023, pp. 6314–6341.

[17] H. Ju, D. Li, and H. R. Zhang. "Robust fine-tuning of deep neural networks with Hessian-based generalization guarantees". In: *International conference on machine learning (ICML)*. 2022.

[18] M. Kwun, D. Morwani, C. H. Su, S. Gil, N. Anand, and S. Kakade. "LOTION: Smoothing the Optimization Landscape for Quantized Training". In: *arXiv preprint arXiv:2510.08757* (2025).

[19] D. Li, Z. Liu, K. Yi, C. Zhao, Z. Zhang, R. Krishnamoorthi, H. Khaitan, H. R. Zhang, and S. Li. "WinQ: Accelerating Quantization-Aware Training of Large Language Models around Saddle Points". In: *ICML* (2026).

[20] D. Li, Z. Zhang, M. Duan, E. Dobriban, and H. R. Zhang. "Efficiently Learning Branching Networks for Multitask Algorithmic Reasoning". In: *arXiv preprint arXiv:2512.01113* (2025).

[21] Y. Li, T. Ma, and H. Zhang. "Algorithmic regularization in over-parameterized matrix sensing and neural networks with quadratic activations". In: *Conference on learning theory*. PMLR. 2018, pp. 2–47.

[22] Z. Li, H. Liu, D. Zhou, and T. Ma. "Chain of thought empowers transformers to solve inherently serial problems". In: *ICLR* (2024).

[23] S. Lotfi, M. Finzi, S. Kapoor, A. Potapczynski, M. Goldblum, and A. G. Wilson. "PAC-Bayes compression bounds so tight that they can explain generalization". In: *Advances in Neural Information Processing Systems (NeurIPS)* (2022).

[24] S. Lotfi, M. Finzi, Y. Kuang, T. G. Rudner, M. Goldblum, and A. G. Wilson. "Non-vacuous generalization bounds for large language models". In: *Proceedings of the 41st International Conference on Machine Learning*. 2024, pp. 32801–32818.

[25] W. Merrill and A. Sabharwal. "The expressive power of transformers with chain of thought". In: *ICLR* (2024).

[26] A. Meterez, P. A. Nair, D. Morwani, C. Pehlevan, S. Kakade, and A. Damian. "A defense of the quadratic model". In: *arXiv preprint arXiv:2607.21716* (2026).

[27] P. Mirtaheri, E. Edelman, S. Jelassi, E. Malach, and E. Boix-Adsera. "Let Me Think! A Long Chain-of-Thought Can Be Worth Exponentially Many Short Ones". In: *NeurIPS* (2025).

[28] J.-F. Ton, M. F. Taufiq, and Y. Liu. "Understanding Chain-of-Thought in LLMs through Information Theory". In: *International Conference on Machine Learning*. PMLR. 2025, pp. 59784–59811.

[29] R. Vershynin. "Introduction to the non-asymptotic analysis of random matrices". In: *Compressed Sensing: Theory and Applications*. Ed. by Y. C. Eldar and G. Kutyniok. Cambridge University Press, 2012, pp. 210–268.

[30] K. Wen, H. Zhang, H. Lin, and J. Zhang. "From sparse dependence to sparse attention: unveiling how chain-of-thought enhances transformer sample efficiency". In: *ICLR* (2025).

[31] B. Widrow, I. Kollar, and M.-C. Liu. "Statistical theory of quantization". In: *IEEE Transactions on instrumentation and measurement* 45.2 (1996), pp. 353–361.

[32] R. Xiong. "Automated experimental design with optimization from historical data simulations". In: *Available at SSRN 5126080* (2025).

[33] R. Xiong, A. Chin, and S. J. Taylor. "Data-driven switchback experiments: Theoretical tradeoffs and empirical bayes designs". In: *arXiv preprint arXiv:2406.06768* (2024).

[34] T. Yang, Y. Huang, Y. Liang, and Y. Chi. "Multi-head transformers provably learn symbolic multi-step reasoning via gradient descent". In: *NeurIPS* (2025).

[35] J. Yao, Y. Hao, H. Zhang, H. Dong, W. Xiong, N. Jiang, and T. Zhang. "Optimizing Chain-of-Thought Reasoners via Gradient Variance Minimization in Rejection Sampling and RL". In: *NeurIPS* (2025).

[36] E. Yeo, Y. Tong, M. Niu, G. Neubig, and X. Yue. "Demystifying long chain-of-thought reasoning in llms". In: *ICML* (2025).

[37] H. R. Zhang, D. Li, and H. Ju. "Noise Stability Optimization for Finding Flat Minima: A Hessian-based Regularization Approach". In: *Transactions on Machine Learning Research (TMLR)* (2024).

[38] M. Zhang, O. Press, W. Merrill, A. Liu, and N. A. Smith. "How language model hallucinations can snowball". In: *Proceedings of the 41st International Conference on Machine Learning*. 2024, pp. 59670–59684.

[39] Y. Zhang, A. Backurs, S. Bubeck, R. Eldan, S. Gunasekar, and T. Wagner. "Unveiling transformers with lego: a synthetic reasoning task". In: *arXiv preprint arXiv:2206.04301* (2022).

[40] Z. Zhang, Z. Zhang, D. Li, L. Wang, J. Dy, and H. R. Zhang. "Linear-Time Demonstration Selection for In-Context Learning via Gradient Estimation". In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. 2025, pp. 16470–16488.

A Complete Proofs

This appendix provides complete proofs for all theoretical results presented in the main paper. We organize the material in the following structure:

- **Appendix A.1** provides the full proof for Lemma 3.1, establishing the error bound between the inference loss and the training loss in T-step generation. The proof is via a recursive expansion of the prediction error using first-order Taylor expansion. Then, we derive the bound via spectral norms of Jacobians taken through the input space.
- **Appendix A.2** establishes the error bounds and noise stability analysis for linear self-attention transformers trained on the weight prediction task for linear functions. We begin with preliminaries that reformulate the training and inference losses, then present the complete proof of Theorem A.5 (Section A.2.1), where standard Gaussian concentration bounds are cited and applied directly.
- **Appendix A.3** extends the analysis to nonlinear self-attention transformers trained on quadratic functions. We first describe the problem setup, then provide the explicit construction of a three-head transformer that implements gradient descent on quadratic losses (Section A.3.1). We derive the gradient dynamics under chain-of-thought training (Section A.3.2) and prove the convergence in Theorem 3.4 (Section A.3.3).

Notations. For any vector v, let $\|v\|$ be the Euclidean norm of v. For any matrix $X \in \mathbb{R}^{m\times n}$, let $\|X\|_F$ be the Frobenius norm and $\|X\|_2$ be the spectral norm of X. If X is a square matrix, then the trace of X, $\mathrm{Tr}\,[X]$, equals the sum of X's diagonal entries.

For two functions $f(n)$ and $g(n)$, we write $g(n) = O(f(n))$ if there exists a fixed value C that does not grow with n such that $g(n) \le C \cdot f(n)$ when n is large enough. We write $g(n) = \Theta(f(n))$ if there exists a fixed value C_1 and C_2 that do not grow with n such that $C_1 f(n) \le g(n) \le C_2 \cdot f(n)$ when n is large enough.

A.1 Proof of Lemma 3.1

In this section, we provide the full proof for Lemma 3.1, which shows a bound on the gap between the inference loss and the true prediction loss conditioned on the ground truth.

For a given input $\mathbf{x} \sim \mathcal{D}$, let $y^{(t)}$ denote the ground-truth label at step t. We denote the model's predictions under two regimes:

1. The prediction conditioned on the ground-truth preceding steps, as in the training loss, is denoted as $f_W(\mathbf{z}^{(t-1)})$.
2. the prediction based on t-step autoregressive generation in the inference loss is denoted as $f_W(\widehat{\mathbf{z}}^{(t-1)})$.

We define their deviations from the label $y^{(t)}$ at step t as $e^{(t)}(\mathbf{x}) = f_W(\mathbf{z}^{(t-1)}) - y^{(t)}$, and $\widehat{e}^{(t)}(\mathbf{x}) = f_W(\widehat{\mathbf{z}}^{(t-1)}) - y^{(t)}$. The prediction gap caused by autoregressive accumulation is

$$\Delta^{(t)}(\mathbf{x}) = f_W(\widehat{\mathbf{z}}^{(t-1)}(\mathbf{x})) - f_W(\mathbf{z}^{(t-1)}(\mathbf{x})).$$

By definition, we have the relation:

$$\widehat{e}^{(t)}(\mathbf{x}) = e^{(t)}(\mathbf{x}) + \Delta^{(t)}(\mathbf{x}).$$

For the Jacobian norms, we write $\|J_i^{(t)}\| := \sup_{\mathbf{x}\in\mathcal{X}^*} \|J_i^{(t)}(\mathbf{x})\|$ for the norm with the supremum taken, so that $\rho_i^{(t)} = \|J_i^{(t)}\| + \sum_{k=i+1}^{t-1} \|J_k^{(t)}\|\, \rho_i^{(k)}$ with $\rho_t^{(t)} = 1$. Next, we present the complete proof of Lemma 3.1, based on a recursive decomposition for $\Delta^{(t)}(\mathbf{x})$.

Proof of Lemma 3.1. First, we bound the loss gap via Lipschitz continuity. We evaluate the two losses at step T. By the assumption that the loss function $\ell(\cdot,\cdot)$ is C_1-Lipschitz continuous with respect to its first argument, the absolute gap between the inference loss and the reference loss for a specific input $\mathbf{x}$ is bounded by:

$$\left|\ell\big(f_W(\widehat{\mathbf{z}}^{(T-1)}(\mathbf{x})), y^{(T)}\big) - \ell\big(f_W(\mathbf{z}^{(T-1)}(\mathbf{x})), y^{(T)}\big)\right| \;\le\; C_1 \|\Delta^{(T)}(\mathbf{x})\|. \tag{6}$$

To bound the loss gap, we bound the norm of the final prediction gap $\|\Delta^{(T)}(\mathbf{x})\|$. We conduct a recursive expansion of the prediction gap $\|\Delta^{(T)}(\mathbf{x})\|$. At the base case of $t = 1$, both predictions rely solely on $\mathbf{x}$. Thus, $\Delta^{(1)}(\mathbf{x}) = 0$ and $\|\widehat{e}^{(1)}(\mathbf{x})\| = \|e^{(1)}(\mathbf{x})\|$.

For an arbitrary step $t > 1$, we apply first-order Taylor's expansion to expand the prediction on the generated prefix around the correct step sequence. Subtracting $f_W(\mathbf{z}^{(t-1)}(\mathbf{x}))$ from $f_W(\widehat{\mathbf{z}}^{(t-1)})$, we obtain the prediction gap:

$$\Delta^{(t)}(\mathbf{x}) = \sum_{i=1}^{t-1} J_i^{(t)}(\mathbf{x})\, \widehat{e}^{(i)}(\mathbf{x}) + \mathcal{R}^{(t)}(\mathbf{x}), \tag{7}$$

where $J_i^{(t)}(\mathbf{x})$ is the step-to-step Jacobian evaluated at the ground-truth context, and $\mathcal{R}^{(t)}(\mathbf{x})$ is the higher-order Taylor remainder.

Under the assumption that the Jacobian of f_W is C_2-Lipschitz continuous, the norm of the remainder $\mathcal{R}^{(t)}(\mathbf{x})$ is bounded by a quadratic function of the input deviation, based on the mean value theorem:

$$\|\mathcal{R}^{(t)}(\mathbf{x})\| \;\le\; \frac{C_2}{2}\Big(\sum_{i=1}^{t-1}\|\widehat{e}^{(i)}(\mathbf{x})\|\Big)^2 \le \frac{C_2}{2}\left(\sum_{i=1}^{t-1} r_i\right)^2, \tag{8}$$

based on equation (3) (with $\mathcal{R}^{(1)}(\mathbf{x}) = 0$). Applying the triangle inequality to equation (7), we have:

$$\|\Delta^{(t)}(\mathbf{x})\| \;\le\; \sum_{i=1}^{t-1}\|J_i^{(t)}\|\|\widehat{e}^{(i)}(\mathbf{x})\| + \|\mathcal{R}^{(t)}(\mathbf{x})\|. \tag{9}$$

Using $\widehat{e}^{(t)}(\mathbf{x}) = e^{(t)}(\mathbf{x}) + \Delta^{(t)}(\mathbf{x})$ and applying the triangle inequality, the inference error norm is bounded by:

$$\|\widehat{e}^{(t)}(\mathbf{x})\| \le \|e^{(t)}(\mathbf{x})\| + \|\Delta^{(t)}(\mathbf{x})\|.$$

Substituting equation (9) above yields the full recurrence relation for the inference error:

$$\|\widehat{e}^{(t)}(\mathbf{x})\| \;\le\; s_t(\mathbf{x}) + \sum_{i=1}^{t-1}\|J_i^{(t)}\|\|\widehat{e}^{(i)}(\mathbf{x})\|, \text{ where } s_t(\mathbf{x}) := \|e^{(t)}(\mathbf{x})\| + \|\mathcal{R}^{(t)}(\mathbf{x})\|. \tag{10}$$

To solve this recurrence, we recursively expand $\|\widehat{e}^{(i)}(\mathbf{x})\|$ in $s_i(\mathbf{x})$. The first-order components become a sum of all possible directed step-to-step paths. The product of the Jacobian spectral norms along these paths forms the cumulative amplification factor $\rho_i^{(t)}$, defined recursively as:

$$\rho_i^{(t)} = \sup_{\mathbf{x}\in\mathcal{X}^*}\|J_i^{(t)}(\mathbf{x})\| + \sum_{k=i+1}^{t-1}\left(\sup_{\mathbf{x}\in\mathcal{X}^*}\|J_k^{(t)}(\mathbf{x})\|\cdot\rho_i^{(k)}\right), \text{ with } \rho_t^{(t)} = 1.$$

Formally, we show that

$$\|\widehat{e}^{(t)}(\mathbf{x})\| \;\le\; \sum_{i=1}^{t}\rho_i^{(t)}\, s_i(\mathbf{x}), \text{ for every } t \ge 1, \tag{11}$$

which holds at $t = 1$ by the base case. Assuming equation (11) up to step $t-1$, substituting it into equation (10), we have

$$\begin{aligned}
\|\widehat{e}^{(t)}(\mathbf{x})\| &\le s_t(\mathbf{x}) + \sum_{k=1}^{t-1}\Big(\sum_{i=k}^{t-1}\|J_i^{(t)}\|\rho_k^{(i)}\Big)s_k(\mathbf{x}) \\
&\le s_t(\mathbf{x}) + \sum_{k=1}^{t-1}\Big(\|J_k^{(t)}\|\,\rho_k^{(k)} + \sum_{i=k+1}^{t-1}\|J_i^{(t)}\|\,\rho_k^{(i)}\Big)s_k(\mathbf{x}) \\
&\le s_t(\mathbf{x}) + \sum_{k=1}^{t-1}\rho_k^{(t)}s_k(\mathbf{x}),
\end{aligned} \tag{12}$$

which completes the induction. By replacing $\rho_i^{(T)}$ and carrying each remainder to step T by its own factor $\rho_t^{(T)}$, the final prediction gap at step T is bounded by:

$$\|\Delta^{(T)}(\mathbf{x})\| \;\le\; \sum_{i=1}^{T-1} \rho_i^{(T)} \|e^{(i)}(\mathbf{x})\| \;+\; \sum_{t=2}^{T} \rho_t^{(T)} \|\mathcal{R}^{(t)}(\mathbf{x})\|. \tag{13}$$

Substituting equation (13) into equation (6), we obtain:

$$\begin{aligned} &\big|\ell\big(f_W(\widehat{\mathbf{z}}^{(T-1)}(\mathbf{x})), y^{(T)}\big) - \ell\big(f_W(\mathbf{z}^{(T-1)}(\mathbf{x})), y^{(T)}\big)\big| \\ \le& C_1 \sum_{i=1}^{T-1} \rho_i^{(T)} \|e^{(i)}(\mathbf{x})\| \;+\; C_1 \sum_{t=2}^{T} \rho_t^{(T)} \|\mathcal{R}^{(t)}(\mathbf{x})\|. \end{aligned} \tag{14}$$

For every $\mathbf{x}$ in the support of $\mathcal{D}$, recall from equation (3) that $\|e^{(i)}(\mathbf{x})\| \le \varepsilon_i$ and $\|\mathcal{R}^{(t)}(\mathbf{x})\| \le \frac{C_2}{2}\big(\sum_{i<t} r_i\big)^2$ from equation (8), we have

$$\begin{aligned} &\big|\ell\big(f_W(\widehat{\mathbf{z}}^{(T-1)}(\mathbf{x})), y^{(T)}\big) - \ell\big(f_W(\mathbf{z}^{(T-1)}(\mathbf{x})), y^{(T)}\big)\big| \\ \le& C_1 \sum_{i=1}^{T-1} \rho_i^{(T)} \varepsilon_i \;+\; \frac{C_1 C_2}{2} \sum_{t=2}^{T} \rho_t^{(T)} \Big(\sum_{i=1}^{t-1} r_i\Big)^2. \end{aligned} \tag{15}$$

Taking the expectation over the data distribution $\mathbf{x} \sim \mathcal{D}$, we have

$$\begin{aligned} \big|L(f_W;\mathcal{D}) - L^\star(f_W;\mathcal{D})\big| &= \Big|\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\Big[\ell\big(f_W(\widehat{\mathbf{z}}^{(T-1)}(\mathbf{x})), y^{(T)}\big) - \ell\big(f_W(\mathbf{z}^{(T-1)}(\mathbf{x})), y^{(T)}\big)\Big]\Big| \\ &\le \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\Big[\big|\ell\big(f_W(\widehat{\mathbf{z}}^{(T-1)}(\mathbf{x})), y^{(T)}\big) - \ell\big(f_W(\mathbf{z}^{(T-1)}(\mathbf{x})), y^{(T)}\big)\big|\Big] \\ &\le C_1 \sum_{i=1}^{T-1} \rho_i^{(T)} \varepsilon_i \;+\; \frac{C_1 C_2}{2} \sum_{t=2}^{T} \rho_t^{(T)} \Big(\sum_{i=1}^{t-1} r_i\Big)^2, \end{aligned} \tag{16}$$

which completes the proof of equation (4). □

A.2 Proof for the Linear Case

We study the inference loss of a one-layer linear self-attention transformer trained to predict weight vectors of linear functions. Following the setup in Section 3.2, we describe the in-context weight prediction task with linear functions. The weight vector is $\theta^\star \sim \mathcal{N}(0, \mathrm{Id}_d)$, and the n in-context examples are $x_j \sim \mathcal{N}(0, \mathrm{Id}_d)$ with labels $y_j = \langle \theta^\star, x_j \rangle$ for $j \in [n]$, all independent. We write $X = [x_1, \ldots, x_n] \in \mathbb{R}^{d\times n}$, $y = [y_1, \ldots, y_n] = \theta^{\star\top} X$, and $S := \frac{1}{n} X X^\top$. The chain of thought consists of the gradient descent iterates on the squared loss on average over n examples with learning rate η, started at $\theta_0 = 0_d$:

$$\theta_{i+1} = \theta_i - \eta S(\theta_i - \theta^\star), \quad \text{so that} \quad \theta_i = \big(I - (I - \eta S)^i\big)\theta^\star \quad \text{for } i \ge 0. \tag{17}$$

With $d_e = 2d + 2$, the input that contains the correct steps $\theta_0, \ldots, \theta_i$ is

$$Z_i = \begin{bmatrix} X & 0_d & 0_d & \cdots & 0_d \\ y & 0 & 0 & \cdots & 0 \\ 0_{d\times n} & \theta_0 & \theta_1 & \cdots & \theta_i \\ 0_{1\times n} & 1 & 1 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^{d_e \times (n+i+1)}.$$

The label of step $i+1$ is $\bar\theta_{i+1} := \theta_{i+1}$ for $i \le T-2$, and the label of the final step is $\bar\theta_T := \theta^\star$.

We first rewrite the training and inference losses. During training, the network is trained to predict the next iterate given the preceding ground-truth sequences. We consider the gradient flow dynamics over the population loss:

$$L^{\mathrm{CoT}}(f_W) = \mathop{\mathbb{E}}_{X,\theta^\star}\left[\frac{1}{2}\sum_{i=0}^{T-1} \|f_W(Z_i) - (0_d, 0, \bar\theta_{i+1}, 1)\|^2\right].$$

At inference time, we apply the model to generate T steps before it outputs the final prediction. Starting from Z_0, we define the model prediction at each step as $\widehat{Z}_{i+1} = \left[\widehat{Z}_i, f_W(\widehat{Z}_i)\right]$, for $i = 0, 1, \ldots, T-1$, where the last column of $\widehat{Z}_i$ encodes the estimate $\widehat{\theta}_i$, and $\widehat{\theta}_0 = \theta_0$. Then, the inference loss, in equation (2), with the squared loss, can be written as:

$$L(f_W; \mathcal{D}) = \mathop{\mathbb{E}}_{X,\theta^\star}\left[\frac{1}{2}\|f_W(\widehat{Z}_{T-1}) - (0_d, 0, \theta^\star, 1)\|_2^2\right]. \tag{18}$$

For linear functions, we consider a one-layer linear self-attention transformer with a residual connection. Given an input embedding matrix $Z \in \mathbb{R}^{d_e \times m}$ of length m, we denote

$$f_W(Z) = Z_{[:,-1]} + W^{PV} Z \cdot \frac{Z^\top W^{KQ} Z_{[:,-1]}}{n}, \tag{19}$$

We denote $W^{PV} \in \mathbb{R}^{d_e \times d_e}$ as the weight matrix that merges the projection and value matrix, and $W^{KQ} \in \mathbb{R}^{d_e \times d_e}$ is the weight matrix that merges the key and query matrix. $[:, -1]$ indicates taking the last column of the matrix. n is the number of in-context examples.

Next, we state results from the previous paper [12], which we will reuse in our statements. Under Assumption A.1, prior work [12] proves that, if we initialize the weight matrices with the block structure in Assumption A.1, the blocks that are initialized with 0 will remain 0 along the gradient flow trajectory. The initialization method is specified as follows to zero out the irrelevant blocks.

Assumption A.1 (Initialization)**.** Let $\sigma_0 > 0$ be a constant. We assume the parameters are initialized as

$$W^{PV}(0) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times 1} & 0_{1\times d} & 0_{1\times 1} \\ W_{31}^{PV}(0) & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times 1} & 0_{1\times d} & 0_{1\times 1} \end{bmatrix}, W^{KQ}(0) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & W_{13}^{KQ}(0) & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times 1} & 0_{1\times d} & W_{24}^{KQ} \\ 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times 1} & 0_{1\times d} & 0_{1\times 1} \end{bmatrix}.$$

Here, $W_{31}^{PV}(0) = \sum_{i=1}^d \lambda_i^{PV} u_i u_i^\top$ and $W_{13}^{KQ}(0) = \sum_{i=1}^d \lambda_i^{KQ} u_i u_i^\top$, where $\{u_i\}_{i=1}^d$ is an orthonormal basis. The matrices $W_{31}^{PV}(0)$ and $W_{13}^{KQ}(0)$ are symmetric and simultaneously diagonalizable, with eigenvalues satisfying $\lambda_i^{PV} \le -\sigma_0$, $\lambda_i^{KQ} \in [\sigma_0, 1/2]$, and for all $i \in [d]$. Finally, we fix $W_{24}^{KQ} = -1$ for all $t \ge 0$.

Lemma A.2 (Block-wise invariance under gradient flow [12])**.** *Under Assumption A.1, consider the gradient flow on the population CoT loss as*

$$\frac{dW}{dt} = -\nabla L^{\mathrm{CoT}}(W).$$

Then, for all $t \ge 0$, the parameters $W^{PV}(t), W^{KQ}(t)$ take the following form:

$$W^{PV}(t) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{1\times d} & 0 \\ W_{31}^{PV}(t) & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{1\times d} & 0 \end{bmatrix}, W^{KQ}(t) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & W_{13}^{KQ}(t) & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{1\times d} & W_{24}^{KQ} \\ 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{1\times d} & 0 \end{bmatrix}.$$

Then, we can rewrite the linear self-attention model only in terms of W_{31}^{PV} and W_{13}^{KQ}.

Definition A.3 (Reduced transformer)**.** Define $S := \frac{1}{n} X X^\top$. Under Assumption A.1, for a given input Z_i, we can rewrite the linear self-attention model by

$$f_W(Z_i) = \left[0_d, 0, \left(\theta_i + W_{31}^{PV} S\left(W_{13}^{KQ}\theta_i - \theta^\star\right)\right), 1\right]^\top.$$

where θ_i is the last column of $Z_{i[d+1:2d+1,-1]}$ and $\theta^\star$ is the ground-truth weights of the linear function. Then, we can denote a reduced model by:

$$\tilde{f}_W(Z_i) = \theta_i + W_{31}^{PV} S\left(W_{13}^{KQ}\theta_i - \theta^\star\right).$$

The right-hand side depends on Z_i only through its last column. The same formula holds for the inputs during inference $\widehat{Z}_i$. Below, we write $\tilde{f}_W(\theta) := \theta + W_{31}^{PV} S(W_{13}^{KQ}\theta - \theta^\star)$ for $\theta \in \mathbb{R}^d$, with X and $\theta^\star$ fixed. This map is affine, and its Jacobian is given by

$$J := \frac{\partial \tilde{f}_W(\theta)}{\partial \theta} = I + W_{31}^{PV} S W_{13}^{KQ}, \tag{20}$$

which does not depend on θ.

Theorem A.4 (Global convergence [12]). *Suppose $n = \Theta(d\log^5 d)$, $\eta \in (0.1, 0.9)$, $T = \lceil c\log d \rceil$, and $c\log(\frac{1}{1-\eta}) > 2$. Under Assumption A.1 with $\sigma_0 > \frac{3(1-\eta)}{(2-\eta)T}$, if we run gradient flow on the population loss $L^{\mathrm{CoT}}(f_W)$, then after time $t = O(\log d + \log\frac{1}{\epsilon})$, for any $\epsilon \geq \Theta(\frac{\log d}{d^{c\log(\frac{1}{1-\eta})-2}})$, we have $L^{\mathrm{CoT}}(f_{W(t)}) \leq \varepsilon$. Moreover, there exist matrices A and B, with $\|A\|_{\mathrm{op}}, \|B\|_{\mathrm{op}} \leq \Theta(d^{-\frac{1}{2}c\log(\frac{1}{1-\eta})+\frac{1}{2}})$, such that $W_{31}^{PV}(t) = (-\eta I + A)$ and $W_{13}^{KQ}(t) = (I + B)$.*

We state Theorem A.5, which is the main result for the linear case.

Theorem A.5 (Slightly adapted from [12]). *Suppose $n = \Theta(d\log^5 d)$, $\eta \in (0.1, 0.9)$, and $T = \lceil c\log d \rceil$, with $c\log\left(\frac{1}{1-\eta}\right) > 2$. Under the initialization in Assumption A.1 with $\sigma_0 > \frac{3(1-\eta)}{(2-\eta)T}$, if we run gradient flow on the population loss $L^{\mathrm{CoT}}(f_W)$, after time $t = O(\log d + \log\frac{1}{\epsilon_0})$, for any $\epsilon_0 = \Theta(\frac{\log d}{d^{c\log(\frac{1}{1-\eta})-2}})$, we have $L(f_W; \mathcal{D}) \leq \epsilon_1$, for any $\epsilon_1 \in \Theta((d^{-c\log(\frac{1}{1-\eta})+2}\log d), 1)$.*

A.2.1 Proof of Theorem A.5

Next, we prove the bounds on inference loss and the noise stability measure of the linear self-attention model on the weight prediction task. Recall that, to evaluate the inference loss, we apply the model f_W to generate T steps before it outputs the final prediction. Starting from the input Z_0, we denote the model's generated output at step i (also serves as the model input at step $i+1$) as follows:

$$\widehat{Z}_{i+1}(W) := \left[\widehat{Z}_i(W), f_W(\widehat{Z}_i(W))\right], \qquad i = 0, 1, \ldots, T-1,$$

where we denote $\widehat{Z}_0 := Z_0$. Then, we denote the last column of $\widehat{Z}_i(W)$ as $\widehat{z}_i(W) := \widehat{Z}_i(W)_{[:,-1]}$ and $\widehat{\theta}_i(W) := \widehat{Z}_i(W)_{[d+1:2d+1,-1]}$. We can write down the block structure of $\widehat{Z}_i$ as

$$\widehat{Z}_i(W) = \begin{bmatrix} X & 0_d & 0_d & \ldots & 0_d \\ y & 0 & 0 & \ldots & 0 \\ 0_{d\times n} & \theta_0 & \widehat{\theta}_1(W) & \ldots & \widehat{\theta}_i(W) \\ 0_{1\times n} & 1 & 1 & \ldots & 1 \end{bmatrix}.$$

Correspondingly, we denote the ground-truth input with i intermediate steps as Z_i, θ_i as ground-truth weight iterates, and we denote the final label by $z_T := (0_d, 0, \theta^\star, 1)$. When W is fixed, we write $\widehat{\theta}_i$ for $\widehat{\theta}_i(W)$.

Proof of Theorem A.5. We will first write the inference error as a sum of prediction errors at each step, each carried to the final step by a power of the Jacobian of the trained transformer. This is a case of Lemma 3.1, where the Jacobian becomes a single matrix J, and the second-order remainder is zero. The inference loss is therefore small once J is contractive. At convergence, we will show that J is close to the Jacobian $I - \eta S$, whose norm $\|J\|_2 < 1$, and the errors are small, which yields the rate for the inference loss.

Let $W = W(t)$ be the trained weights at time t given by Theorem A.4. By Definition A.3, we have the simplified form of the transformer:

$$\widehat{\theta}_i = \tilde{f}_W(\widehat{\theta}_{i-1}) = \widehat{\theta}_{i-1} + W_{31}^{PV} S\big(W_{13}^{KQ}\widehat{\theta}_{i-1} - \theta^\star\big),$$

for $i = 1, \ldots, T$, and $\widehat{\theta}_T$ is the θ-block of the final output $f_W(\widehat{Z}_{T-1})$. Therefore, we can write

$$L(f_W; \mathcal{D}) = \frac{1}{2}\mathop{\mathbb{E}}_{X,\theta^\star}\left[\|\widehat{\theta}_T - \theta^\star\|^2\right].$$

We compare the generated estimates $\widehat{\theta}_i$ with the gradient descent iterates θ_i. Let

$$\Delta_i := \tilde{f}_W(\theta_{i-1}) - \theta_i, \qquad i = 1, \ldots, T,$$

be the prediction error of step i, when the model is given the correct previous step. Since $\tilde{f}_W$ is affine with the Jacobian J in equation (20),

$$\widehat{\theta}_i - \theta_i = \big(\tilde{f}_W(\widehat{\theta}_{i-1}) - \tilde{f}_W(\theta_{i-1})\big) + \Delta_i = J(\widehat{\theta}_{i-1} - \theta_{i-1}) + \Delta_i.$$

Recursively decomposing the above from $\widehat{\theta}_0 = \theta_0$, we have

$$\widehat{\theta}_T - \theta^\star = (\theta_T - \theta^\star) + \sum_{i=1}^{T} J^{T-i}\Delta_i. \tag{21}$$

This equation is the form of Lemma 3.1 for the linear transformer. The first term, $\theta_T - \theta^\star$, is the error of T gradient descent steps on the given in-context examples, because the final label is $\theta^\star$. Since $\|J\|$ is not bounded uniformly over Gaussian inputs, we bound equation (21) in expectation rather than with the worst-case constants of Lemma 3.1. By the Cauchy–Schwarz inequality,

$$L(f_W; \mathcal{D}) \le \frac{T+1}{2}\Big(\mathop{\mathbb{E}}_{X,\theta^\star}\left[\|\theta_T - \theta^\star\|^2\right] + \sum_{i=1}^{T} \mathop{\mathbb{E}}_{X,\theta^\star}\left[\|J^{T-i}\Delta_i\|^2\right] \Big). \tag{22}$$

It remains to show that J is contractive at convergence and that the residuals are small.

(1) Upper bound of $\mathbb{E}\|J\|_2$. Now, we analyze the spectral norm of the Jacobian matrix of the linear transformer, $J = I + W_{31}^{PV} S W_{13}^{KQ}$. Based on the results in Theorem A.4, for the trained transformer by running the gradient flow on the population loss on the chain-of-thought sequences, there exists matrix A and B, such that $W_{31} = (-\eta I + A)$ and $W_{13} = (I + B)$ with $\|A\|_{\mathrm{op}}, \|B\|_{\mathrm{op}} \le \Theta(d^{-\frac{1}{2}c\log(\frac{1}{1-\eta})+\frac{1}{2}})$. We denote

$$\beta := \Theta(d^{-\frac{1}{2}c\log(\frac{1}{1-\eta})+\frac{1}{2}}), \text{ and } a := c\log(\frac{1}{1-\eta}) > 2.$$

In particular, $\beta = \Theta(d^{\frac{1-a}{2}}) \le d^{-1/2}$ for a large enough d.

Therefore, we have:

$$\begin{aligned} J &= I + W_{31}^{PV} S W_{13}^{KQ} \\ &= I + (-\eta I + A) S (I + B) \\ &= I - \eta S + (AS - \eta SB + ASB) \\ &= J_0 + E, \end{aligned}$$

where $J_0 = I - \eta S$ and $E = (AS - \eta SB + ASB)$.

Recall that $S = \frac{1}{n} X X^\top$ where $X = [x_1, \ldots, x_n]$ with x_i sampled from $\mathcal{N}(0, \mathrm{Id}_d)$. By the Gaussian singular-value bound [29] applied to X, given $n = \Theta(d^2 \log^5 d)$, the event $\mathcal{G} := \{\|S - I\|_2 \le C\sqrt{d/n}\}$ has probability at least $1 - 2e^{-d/2}$ for a constant C. On $\mathcal{G}$, we have that

$$\|J_0\|_2 = \|I - \eta S\|_2 \le 1 - \eta + C\eta\sqrt{d/n} = 1 - \eta + O(\log^{-5/2} d), \text{ and } \|S\|_2 \le 2.$$

For matrix E, on $\mathcal{G}$, submultiplicativity gives

$$\|E\|_2 \le \big(\|A\|_2 + \eta\|B\|_2 + \|A\|_2\|B\|_2\big)\|S\|_2 = \Theta(d^{-c\log\frac{1}{1-\eta}+1}).$$

Therefore, on $\mathcal{G}$,

$$\|J\|_2 \le \|J_0\|_2 + \|E\|_2 \le 1 - \eta + O(\log^{-5/2} d) + \Theta(d^{-c\log\frac{1}{1-\eta}+1}). \tag{23}$$

Thus, $\|J\|_2 \le 1 - \eta/2 < 1$ for a large enough d.

(2) Upper bound of $\mathbb{E}\|\Delta_i\|^2$. We have the one-step gradient descent update as $\theta_i = \theta_{i-1} - \eta S(\theta_{i-1} - \theta^\star)$. Then, the prediction error of step i conditioned on the correct steps is

$$\begin{aligned} \Delta_i &= \tilde{f}_W(\theta_{i-1}) - \theta_i \\ &= (W_{31} S W_{13} + \eta S)\,\theta_{i-1} - (W_{31} + \eta I) S\theta^\star = (AS - \eta SB + ASB)\,\theta_{i-1} - AS\,\theta^\star. \end{aligned}$$

Therefore, we have

$$\|\Delta_i\|^2 \le 2\big(\|AS - \eta SB + ASB\|_2^2 \|\theta_{i-1}\|^2 + \|AS\|_2^2 \|\theta^\star\|^2\big). \tag{24}$$

Note that θ_i is the i-th iterate of the gradient descent applied to (X, y). Given that $\theta^\star \sim \mathcal{N}(0, \mathrm{Id}_d)$ and $S = \frac{1}{n} XX^T$, with $n = \Theta(d^2 \log^5 d)$, we have $\mathbb{E}_{X,\theta^\star}\left[\|\theta_i\|^2\right] \le O(d)$. Therefore, based on $\theta_i = (I - (I - \eta S)^i)\theta^\star$, on $\mathcal{G}$, we have $\|I - \eta S\|_2 \le 1$, so $\|\theta_i\| \le 2\|\theta^\star\|$.

On $\mathcal{G}$, we also have $\|AS - \eta SB + ASB\|_2 \le 6\beta$, and $\|AS\|_2 \le 2\beta$.

Therefore, plug those back to equation (24), for $i = 1, \ldots, T$, we have:

$$\mathbb{E}\big[\|\Delta_i\|_2^2 \mathbf{1}_{\mathcal{G}}\big] \le 2\big(36\beta^2 \cdot 4d + 4\beta^2 d\big) = O(d\beta^2) = O(d^{2-a}).$$

Upper bound of $L(f_W; \mathcal{D})$. Now, we have shown that $\mathbb{E}[\|\Delta_i\|^2 \mathbf{1}_{\mathcal{G}}] = O(d^{2-a})$ for $i = 1, \ldots, T$, and $\|J\|_2 \le 1 - \eta/2$ on $\mathcal{G}$. Then, summing the geometric series gives

$$\sum_{i=1}^{T} \mathbb{E}\big[\|J^{T-i}\Delta_i\|^2 \mathbf{1}_{\mathcal{G}}\big] \le \sum_{i=1}^{T} (1 - \eta/2)^{2(T-i)}\, \mathbb{E}\big[\|\Delta_i\|^2 \mathbf{1}_{\mathcal{G}}\big] \le O(d^{2-a}).$$

In addition, the first term of equation (22) is the error of exact gradient descent. By equation (17), $\theta_T - \theta^\star = -(I - \eta S)^T \theta^\star$. On $\mathcal{G}$, we have

$$\|I - \eta S\|_2^{2T} \le (1 - \eta)^{2T} \big(1 + \frac{C\eta}{1 - \eta}\sqrt{d/n}\big)^{2T} \le 2(1 - \eta)^{2T}.$$

Since $T \ge c \log d$, we have

$$\mathbb{E}\big[\|\theta_T - \theta^\star\|^2 \mathbf{1}_{\mathcal{G}}\big] \le 2(1 - \eta)^{2T}\, \mathbb{E}\|\theta^\star\|^2 \le 2d^{1-2a}.$$

Combining these results into equation (22), by the Cauchy–Schwarz inequality, we have

$$L(f_W; \mathcal{D}) \le \frac{T+1}{2}\Big(2d^{1-2a} + O(d^{2-a}) + e^{-\Omega(d)}\Big) = O(d^{2-a} \log d), \tag{25}$$

since $T + 1 = O(\log d)$.

□

A.3 Proof for the Quadratic Case

In this section, we study an in-context weight prediction task where the model is trained to predict the parameters of a quadratic function from n examples. Let the quadratic weight vector be $\theta^\star \in \mathbb{R}^d$ where $\theta^\star \sim \mathcal{N}(0, \mathrm{Id}_d)$. We sample n in-context examples $x_i \sim \mathcal{N}(0, \mathrm{Id}_d)$ with the corresponding labels according to $y_i = (\theta^{\star\top} x_i)^2$, for $i = 1, \ldots, n$. The goal is to predict $\theta^\star$ given the n examples.

Similar to weight prediction for the linear function, define the embedding dimension $d_e = 2d + 2$. The initial input sequence is constructed as

$$Z_0 = \begin{bmatrix} x_1 & \cdots & x_n & 0 \\ y_1 & \cdots & y_n & 0 \\ 0 & \cdots & 0 & \theta_0 \\ 0 & \cdots & 0 & 1 \end{bmatrix} := \begin{bmatrix} X & 0_d \\ y & 0 \\ 0_d & \theta_0 \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{d_e \times (n+1)}, \tag{26}$$

where $X = [x_1, \ldots, x_n] \in \mathbb{R}^{d \times n}$, $y = [y_1, \ldots, y_n]^\top \in \mathbb{R}^n$, and $\theta_0 \in \mathbb{R}^d$ is an initial estimate in the basin of $\theta^\star$. We take $\theta_0 = \alpha_0 \theta^\star$ for a constant $\alpha_0 \in [3/4, 1)$. We also append a dummy constant 1 to each parameter token for formatting convenience.

We construct the chain-of-thought using gradient descent on the squared loss, i.e., $\frac{1}{2n}\sum_{j=1}^{n}\big((\theta^\top x_j)^2 - y_j\big)^2$. Define a residual vector as $e(\theta) = \widehat{y}(\theta) - y$, where $\widehat{y}_j(\theta) = (\theta^\top x_j)^2$. The gradient of the loss is

$$\frac{2}{n}\sum_{j=1}^{n}\big((\theta^\top x_j)^2 - y_j\big) x_j x_j^\top \theta = \frac{2}{n} X \,\mathrm{diag}\,\big(e(\theta)\big) X^\top \theta.$$

Given a learning rate η, starting from θ_0, define the gradient descent iterates for $i = 1, \ldots, T$:

$$\theta_i = \theta_{i-1} - \eta \cdot \frac{2}{n} X \operatorname{diag}\big(e(\theta_{i-1})\big) X^\top \theta_{i-1} = \theta_{i-1} - \eta \cdot \frac{2}{n} X D_i^+ D_i^- X^\top \theta_{i-1},$$

where $D_i^+ = \operatorname{diag}(X^\top \theta_{i-1} + X^\top \theta^\star)$ and $D_i^- = \operatorname{diag}(X^\top \theta_{i-1} - X^\top \theta^\star)$.

Using these iterates, we define the ground-truth sequences

$$Z_i = \begin{bmatrix} x_1 & \cdots & x_n & 0 & 0 & \cdots & 0 \\ y_1 & \cdots & y_n & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \theta_0 & \theta_1 & \cdots & \theta_i \\ 0 & \cdots & 0 & 1 & 1 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^{d_e \times (n+i+1)}, \text{ for } i = 1, 2, \ldots, T. \tag{27}$$

A.3.1 Proof of Lemma 3.3

Proof. We now construct a three-head nonlinear self-attention transformer that performs one gradient descent step for the quadratic squared loss. For head $h \in \{1, 2, 3\}$, we denote a three-head self-attention model as

$$f_W(Z) = Z_{[:,-1]} + \sum_{i=1}^{3} W^{PV_i} Z \cdot \phi_i\Big(\frac{Z^\top W^{KQ_i} Z_{[:,-1]}}{n}\Big).$$

Here $W^{V_i}, W^{K_i}, W^{Q_i} \in \mathbb{R}^{d_i \times d_e}$ for $i = 1, 2, 3$. One can set d_1, d_2, and d_3 separately. We construct the transformer in the following way. Let $\phi(\cdot)$ be applied elementwise as

$$\phi_1(t) = t^{\odot 3}, \qquad \phi_2(t) = t^{\odot 2}, \qquad \phi_3(t) = t^{\odot 2}. \tag{28}$$

We then set

$$W^{PV_1} = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ -2\eta n^2 \operatorname{Id}_d & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}, \quad W^{KQ_1} = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & \operatorname{Id}_d & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix},$$

$$W^{PV_2} = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ \frac{1}{2}\eta n \operatorname{Id}_d & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}, \quad W^{KQ_2} = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & \operatorname{Id}_d & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 1 \\ 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix},$$

$$W^{PV_3} = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ -\frac{1}{2}\eta n \operatorname{Id}_d & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}, \quad W^{KQ_3} = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & \operatorname{Id}_d & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & -1 \\ 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}. \tag{29}$$

Now, we can inductively calculate the i-th step of generation, showing that the output is exactly the parameter after the i-th gradient step.

$$\begin{aligned} f_W(Z_i) =& (0_d, 0, \theta_i, 1) + \Big(0_d, 0, -2\eta n^2 X \cdot \phi_1\big(\tfrac{Z^\top W^{KQ_1} Z_{[:,-1]}}{n}\big) \\ &+ \frac{\eta n}{2} X \cdot \phi_2\big(\tfrac{Z^\top W^{KQ_2} Z_{[:,-1]}}{n}\big) - \frac{\eta n}{2} X \cdot \phi_3\big(\tfrac{Z^\top W^{KQ_3} Z_{[:,-1]}}{n}\big), 0\Big). \end{aligned}$$

Based on the construction, we have the following:

$$\begin{aligned} X \cdot \phi_1\big(Z^\top W^{KQ_1} Z_{[:,-1]}\big) &= X(X^\top \theta_i)^{\odot 3} \\ X \cdot \phi_2\big(Z^\top W^{KQ_2} Z_{[:,-1]}\big) &= X(X^\top \theta_i + y^\top)^{\odot 2} \\ X \cdot \phi_3\big(Z^\top W^{KQ_3} Z_{[:,-1]}\big) &= X(X^\top \theta_i - y^\top)^{\odot 2} \end{aligned}$$

Therefore, we have

$$\begin{aligned} f_W(Z_i) &= (0_d, 0, \theta_i, 1) + (0_d, 0, -\frac{2\eta}{n} X\big((X^\top \theta_i)^{\odot 3} - (X^\top \theta_i) \odot y^\top\big), 0) \\ &= (0_d, 0, \theta_i - \frac{2\eta}{n} X \operatorname{diag}\big(e(\theta_i)\big) X^\top \theta_i, 1) \end{aligned}$$

where we denote the residual error $e(\theta) = \widehat{y}(\theta) - y$, where $\widehat{y}_j(\theta) = (\theta^\top x_j)^2$.

Then, after T steps of applying f_W, we obtain the T-step gradient descent solution using n examples of (x, y). Denote the θ_T as the solution obtained by applying T-gradient step. Based on Lemma A.6, we have the test loss of f_W as

$$L_W = \frac{1}{2} \mathop{\mathbb{E}}_{X,\theta^\star} \left[\left\| f_W(\widehat{Z}_{T-1}) - (0_d, 0, \theta^\star, 1) \right\|_2^2 \right] = \frac{1}{2} \mathop{\mathbb{E}}_{X,\theta^\star} \left[\|\theta_T - \theta^\star\|_2^2 \right] \le O(d^{-2c \log(\frac{1}{1-\mu\eta d})+1}).$$

□

Lemma A.6. *Let $\theta^\star \sim \mathcal{N}(0, \mathrm{Id}_d)$ and $x_1, \ldots, x_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \mathrm{Id}_d)$. Define $y_j = (x_j^\top \theta^\star)^2$ and the empirical loss*

$$\mathcal{L}_n(\theta) := \frac{1}{2n} \sum_{j=1}^n \left((x_j^\top \theta)^2 - y_j \right)^2 = \frac{1}{2n} \sum_{j=1}^n \left((x_j^\top \theta)^2 - (x_j^\top \theta^\star)^2 \right)^2.$$

Let $\mathrm{dist}(\theta, \{\pm\theta^\star\}) := \min\{\|\theta - \theta^\star\|_2,\ \|\theta + \theta^\star\|_2\}$. Consider gradient descent $\theta_{t+1} = \theta_t - \eta \nabla \mathcal{L}_n(\theta_t), t = 0, 1, 2, \ldots, T$. There exist absolute constants $0 < m \le L < \infty$ and a radius $\rho \in (0, 1)$ such that for all θ satisfying $\mathrm{dist}(\theta, \{\pm\theta^\star\}) \le \rho \|\theta^\star\|_2$, the Hessian of $\mathcal{L}_n$ obeys

$$m\|\theta^\star\|_2^2 \,\mathrm{Id}_d \ \preceq\ \nabla^2 \mathcal{L}_n(\theta) \ \preceq\ L\|\theta^\star\|_2^2 \,\mathrm{Id}_d\,.$$

Suppose the initialization lies in the basin, i.e., $\mathrm{dist}(\theta_0, \{\pm\theta^\star\}) \le \rho\|\theta^\star\|_2$. Assume that $n \ge C d^2 \log^5 d$ for a sufficiently large constant $C > 0$, $0 < \eta \le \frac{1}{2L\|\theta^\star\|_2^2}$, and $T = \lceil c \log d \rfloor$. Then, with probability at least $1 - d^{-10}$ over $(\theta^\star, X)$, the following holds

$$\mathrm{dist}(\theta_T, \{\pm\theta^\star\})^2 \ \le\ \mathrm{dist}(\theta_0, \{\pm\theta^\star\})^2 \cdot d^{2c \log\left(1 - m\eta\|\theta^\star\|_2^2\right)}.$$

Then, given $\eta = \Theta(1/d)$ and $T = \Theta(\log d)$, we have

$$\mathbb{E}_{X,\theta^\star} \left[\mathrm{dist}(\theta_T, \pm\theta^\star)^2 \right] \le O\left(d^{1 - 2c \log(\frac{1}{1-m\eta d})} \right).$$

Proof. We prove the claim on a high-probability event under which the empirical loss is uniformly strongly convex and smooth in a neighborhood of $\{\pm\theta^\star\}$, and then apply a standard contraction argument for gradient descent. Let $x \in \mathbb{R}^d$ and define

$$\ell_x(\theta) := \frac{1}{2}\big((x^\top \theta)^2 - (x^\top \theta^\star)^2 \big)^2.$$

Then

$$\nabla \ell_x(\theta) = \big((x^\top \theta)^2 - (x^\top \theta^\star)^2 \big) \nabla (x^\top \theta)^2 = 2\big((x^\top \theta)^2 - (x^\top \theta^\star)^2 \big)(x^\top \theta) x,$$

and differentiating once more gives

$$\begin{aligned}
\nabla^2 \ell_x(\theta) &= 2\Big(\nabla\big((x^\top \theta)^2 - (x^\top \theta^\star)^2 \big)(x^\top \theta) + \big((x^\top \theta)^2 - (x^\top \theta^\star)^2 \big) \nabla (x^\top \theta) \Big) x^\top \\
&= 2\Big(2(x^\top \theta) x (x^\top \theta) + \big((x^\top \theta)^2 - (x^\top \theta^\star)^2 \big) x \Big) x^\top \\
&= 2\Big(3(x^\top \theta)^2 - (x^\top \theta^\star)^2 \Big) x x^\top.
\end{aligned}$$

Therefore, the empirical Hessian is

$$\nabla^2 \mathcal{L}_n(\theta) = \frac{1}{n} \sum_{j=1}^n \nabla^2 \ell_{x_j}(\theta) = \frac{2}{n} \sum_{j=1}^n \Big(3(x_j^\top \theta)^2 - (x_j^\top \theta^\star)^2 \Big) x_j x_j^\top.$$

Let $\mathcal{B} := \{\theta \in \mathbb{R}^d :\ \mathrm{dist}(\theta, \{\pm\theta^\star\}) \le \rho\|\theta^\star\|_2\}$. We use a claim that when $n \ge C d^2 \log^5 d$ (for C sufficiently large), with probability at least $1 - d^{-10}$ over $(\theta^\star, X)$, the following event holds:

$$\mathcal{E}: \quad \forall \theta \in \mathcal{B}, \qquad m\|\theta^\star\|_2^2 \,\mathrm{Id}_d \preceq \nabla^2 \mathcal{L}_n(\theta) \preceq L\|\theta^\star\|_2^2 \,\mathrm{Id}_d, \tag{30}$$

for some absolute constants $0 < m \le L < \infty$ and $\rho \in (0, 1)$. One can prove $\mathcal{E}$ by a standard ε-net argument over $\mathcal{B}$ together with Gaussian concentration for fourth-order polynomials. For example,

see the local regularity condition established (cf. Section 7 of [3]), the empirical Hessian is uniformly well-conditioned in a neighborhood of $\pm\theta^\star$. We condition on $\mathcal{E}$ for the remainder of the proof.

Assume $\theta_0 \in \mathcal{B}$. Since on $\mathcal{E}$ we have $\nabla^2\mathcal{L}_n(\theta) \preceq L\|\theta^\star\|_2^2 \,\mathrm{Id}_d$ for all $\theta \in \mathcal{B}$, the loss is $L\|\theta^\star\|_2^2$-smooth on $\mathcal{B}$. With stepsize $\eta \le 1/(2L\|\theta^\star\|_2^2)$, the standard descent lemma implies $\mathcal{L}_n(\theta_{t+1}) \le \mathcal{L}_n(\theta_t)$ as long as $\theta_t \in \mathcal{B}$. Moreover, since $\mathcal{L}_n(\theta)$ is strongly convex in a neighborhood of each minimizer $\pm\theta^\star$ on $\mathcal{E}$, and $\theta^\star$ is a minimizer with $\nabla\mathcal{L}_n(\theta^\star) = 0$, the GD iterates remain in the same local basin. Thus $\theta_t \in \mathcal{B}$ for all $t \le T$.

Next, we show one-step contraction in distance to $\{\pm\theta^\star\}$. Fix the sign $s \in \{+1, -1\}$ that minimizes $\|\theta_0 - s\theta^\star\|_2$ and define

$$\Delta_t := \theta_t - s\theta^\star.$$

On $\mathcal{E}$, for all $\theta \in \mathcal{B}$ we have

$$m\|\theta^\star\|_2^2 \,\mathrm{Id}_d \preceq \nabla^2\mathcal{L}_n(\theta) \preceq L\|\theta^\star\|_2^2 \,\mathrm{Id}_d\,.$$

Since $\nabla\mathcal{L}_n(s\theta^\star) = 0$ and $\mathcal{L}_n$ is twice differentiable, the mean value form of the gradient gives

$$\nabla\mathcal{L}_n(\theta_t) - \nabla\mathcal{L}_n(s\theta^\star) = \Big(\int_0^1 \nabla^2\mathcal{L}_n(s\theta^\star + \tau(\theta_t - s\theta^\star))d\tau\Big)(\theta_t - s\theta^\star) =: H_t\Delta_t,$$

where H_t satisfies the same eigenvalue bounds:

$$m\|\theta^\star\|_2^2 \,\mathrm{Id}_d \preceq H_t \preceq L\|\theta^\star\|_2^2 \,\mathrm{Id}_d\,.$$

Therefore, the GD update can be written as

$$\Delta_{t+1} = \theta_{t+1} - s\theta^\star = \theta_t - s\theta^\star - \eta\nabla\mathcal{L}_n(\theta_t) = \Delta_t - \eta H_t\Delta_t = (I - \eta H_t)\Delta_t.$$

Taking norms and using $H_t \succeq m\|\theta^\star\|_2^2 \,\mathrm{Id}_d$ yields

$$\|\Delta_{t+1}\|_2 \le \|I - \eta H_t\|_{\mathrm{op}}\,\|\Delta_t\|_2 \le (1 - \eta m\|\theta^\star\|_2^2)\|\Delta_t\|_2,$$

where the last inequality uses $\eta \le 1/(2L\|\theta^\star\|_2^2)$ so that all eigenvalues of $I - \eta H_t$ lie in $[0, 1 - \eta m\|\theta^\star\|_2^2]$. Squaring gives the one-step contraction

$$\|\Delta_{t+1}\|_2^2 \le (1 - m\eta\|\theta^\star\|_2^2)^2\|\Delta_t\|_2^2.$$

By induction,

$$\|\Delta_{T-1}\|_2^2 \le (1 - m\eta\|\theta^\star\|_2^2)^{2T}\|\Delta_0\|_2^2.$$

Since $\|\Delta_t\|_2 = \mathrm{dist}(\theta_t, \{\pm\theta^\star\})$ by construction of s, and $T = c\log d$,

$$\begin{aligned}\mathrm{dist}(\theta_{T-1}, \{\pm\theta^\star\})^2 &\le \mathrm{dist}(\theta_0, \{\pm\theta^\star\})^2 \cdot (1 - m\eta\|\theta^\star\|_2^2)^{2c\log d} \\ &= \mathrm{dist}(\theta_0, \{\pm\theta^\star\})^2 \cdot d^{2c\log(1 - m\eta\|\theta^\star\|_2^2)}.\end{aligned}$$

Finally, the probability statement follows since $\mathcal{E}$ holds with probability at least $1 - d^{-10}$. □

A.3.2 Gradient Computation

In this subsection, we compute the gradient of the full model and show that under certain initialization, running gradient flow on the training loss converges to the construction in Lemma 3.3.

Recall that we train a three-head nonlinear self-attention transformer, denoted as

$$f_W(Z) = Z_{[:,-1]} + \sum_{i=1}^{3} W^{PV_i} Z \cdot \phi_i\Big(\tfrac{Z^\top W^{KQ_i} Z_{[:,-1]}}{n}\Big).$$

Here, by the convention in prior work [12, 34], we merge W^{P_i} and W^{V_i} into a single matrix and merge W^{K_i} and W^{Q_i} into a single matrix. W^{PV_i} and $W^{KQ_i} \in \mathbb{R}^{d_e \times d_e}$ for $i = 1, 2, 3$. By the construction, we set $\phi_i(\cdot)$ as elementwise non-linear functions $\phi_1(t) = t^{\odot 3}, \phi_2(t) = t^{\odot 2}$, and $\phi_3(t) = t^{\odot 2}$.

The chain-of-thought training objective is written as

$$L^{\mathrm{CoT}}(W) = \mathop{\mathbb{E}}_{x_i \sim \mathcal{N}(0, \mathrm{Id}_d), \theta^\star \sim \mathcal{N}(0, \mathrm{Id}_d)} \left[\frac{1}{2}\sum_{i=0}^{T-1} \big\|f_W(Z_i) - (0_d, 0, \bar{\theta}_{i+1}, 1)\big\|_2^2\right].$$

Here, $\bar{\theta}_{i+1} := \theta_{i+1}$ for $i \leq T-2$ and $\bar{\theta}_T := \theta^\star$. As in the training loss equation (1), there are T steps. The label of the last step is the target $\theta^\star$. The reduced loss and all gradients below use this convention.

At inference time, we apply the model to generate $T-1$ steps before it outputs the final prediction. The test loss is written as

$$L(W) = \mathop{\mathbb{E}}_{X,\theta^\star}\left[\frac{1}{2}\left\|f_W(\widehat{Z}_{T-1}) - (0_d, 0, \theta^\star, 1)\right\|_2^2\right].$$

Now, we compute the gradients of W^{PV_i} and W^{KQ_i} based on the loss of each CoT step.

Lemma A.7 (Full model gradient)**.** *The gradients of W^{PV_i} and W^{KQ_i} for $i = \{1,2,3\}$ are given by the following.*

$$\nabla_{W^{PV_1}} L^{\mathrm{CoT}}(W) = \mathop{\mathbb{E}}_{X,\theta^\star}\left[\sum_{i=0}^{T-1} r_i \Big(Z_i \phi_1\Big(\frac{Z_i^\top W^{KQ_1} Z_{i[:,-1]}}{n}\Big)\Big)^\top\right]$$

$$\nabla_{W^{PV_2}} L^{\mathrm{CoT}}(W) = \mathop{\mathbb{E}}_{X,\theta^\star}\left[\sum_{i=0}^{T-1} r_i \Big(Z_i \phi_2\Big(\frac{Z_i^\top W^{KQ_2} Z_{i[:,-1]}}{n}\Big)\Big)^\top\right]$$

$$\nabla_{W^{PV_3}} L^{\mathrm{CoT}}(W) = \mathop{\mathbb{E}}_{X,\theta^\star}\left[\sum_{i=0}^{T-1} r_i \Big(Z_i \phi_3\Big(\frac{Z_i^\top W^{KQ_3} Z_{i[:,-1]}}{n}\Big)\Big)^\top\right]$$

$$\nabla_{W^{KQ_1}} L^{\mathrm{CoT}}(W) = \mathop{\mathbb{E}}_{X,\theta^\star}\left[\frac{1}{n}\sum_{i=0}^{T-1} Z_i \Big(\big(Z_i^\top W^{PV_1\top} r_i\big) \odot 3\big(\tfrac{Z_i^\top W^{KQ_1} Z_{i[:,-1]}}{n}\big)^{\odot 2}\Big) Z_{i[:,-1]}^\top\right]$$

$$\nabla_{W^{KQ_2}} L^{\mathrm{CoT}}(W) = \mathop{\mathbb{E}}_{X,\theta^\star}\left[\frac{1}{n}\sum_{i=0}^{T-1} Z_i \Big(\big(Z_i^\top W^{PV_2\top} r_i\big) \odot 2\big(\tfrac{Z_i^\top W^{KQ_2} Z_{i[:,-1]}}{n}\big)\Big) Z_{i[:,-1]}^\top\right]$$

$$\nabla_{W^{KQ_3}} L^{\mathrm{CoT}}(W) = \mathop{\mathbb{E}}_{X,\theta^\star}\left[\frac{1}{n}\sum_{i=0}^{T-1} Z_i \Big(\big(Z_i^\top W^{PV_3\top} r_i\big) \odot 2\big(\tfrac{Z_i^\top W^{KQ_3} Z_{i[:,-1]}}{n}\big)\Big) Z_{i[:,-1]}^\top\right]$$

where we define

$$r_i = \sum_{h=1}^{3} W^{PV_h} Z_i \cdot \phi_h\Big(\frac{Z_i^\top W^{KQ_h} Z_{i[:,-1]}}{n}\Big) - (0_d, 0, \bar{\theta}_{i+1} - \theta_i, 0)$$

with $\bar{\theta}_{i+1}$ defined above.

Proof. For a step index $i = 0, 1, \ldots, T-1$, write $Z \in \mathbb{R}^{d_e \times m}$ with $m = n+i+1$ and $z = Z_{[:,-1]} \in \mathbb{R}^{d_e}$. We define

$$s_h := \frac{Z^\top W^{KQ_h} z}{n}, \qquad v_h := \phi_h(s_h), \quad u_h := W^{PV_h} Z v_h, \text{ for } h = 1,2,3.$$

Then, we can write the squared loss in terms of the residual error in each step as

$$\ell_i(W) := \frac{1}{2}\|r_i\|^2, \text{ where } r_i = \sum_{h=1}^{3} u_h - (0_d, 0, \theta_{i+1} - \theta_i, 0),$$

Then, we compute the gradients by the chain rule. For each head, the dependence of the loss on W^{PV_h} is only through u_h. Hence,

$$\nabla_{W^{PV_h}} \ell_i(W) = r_i (Z v_h)^\top = r_i \Big(Z \phi_h\Big(\frac{Z^\top W^{KQ_h} z}{n}\Big)\Big)^\top.$$

Summing over i and taking the expectation yields the stated formula.

Then, we compute the gradient for W^{KQ_h}. Since $v_h = \phi_h(s_h)$ is elementwise, we define

$$q_h := \frac{\partial \ell_i}{\partial s_h} = \Big(Z^\top (W^{PV_h})^\top r_i\Big) \odot \phi_h'(s_h).$$

Using the definition of s_h, we have

$$\nabla_{W^{KQ_h}} \ell_i(W) = \frac{1}{n} Z q_h z^\top = \frac{1}{n} Z \Big(\big(Z^\top (W^{PV_h})^\top r_i \big) \odot \phi_h'(s_h) \Big) z^\top.$$

Plugging in $\phi_1'(t) = 3t^{\odot 2}$ and $\phi_2'(t) = \phi_3'(t) = 2t$ gives exactly the gradients after summing over i and taking the expectation. □

According to Lemma 3.3, we use the following specific initialization to zero out the irrelevant blocks.

Assumption A.8 (Initialization)**.** We assume the parameters are initialized as

$$W^{PV_1}(0) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ W_{31}^{PV_1}(0) & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}, W^{KQ_1}(0) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & W_{13}^{KQ_1}(0) & W_{14}^{KQ_1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & W_{24}^{KQ_1} \\ 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}$$

$$W^{PV_2}(0) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ W_{31}^{PV_2}(0) & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}, W^{KQ_2}(0) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & W_{13}^{KQ_2}(0) & W_{14}^{KQ_2} \\ 0_{1\times d} & 0 & 0_{d\times 1} & W_{24}^{KQ_2} \\ 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}$$

$$W^{PV_3}(0) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ W_{31}^{PV_3}(0) & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}, W^{KQ_3}(0) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & W_{13}^{KQ_3}(0) & W_{14}^{KQ_3} \\ 0_{1\times d} & 0 & 0_{d\times 1} & W_{24}^{KQ_3} \\ 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}$$

Additionally, we impose a *weight-sharing mechanism* that ties weights across W^{PV_i} matrices and across W^{KQ_i} matrices. Let two underlying matrices be W_{31}^{PV} and W_{13}^{KQ}. We represent the matrices in the transformer with the two matrices as follows:

$$W_{31}^{PV_1}(t) = -2\eta n^2 W_{31}^{PV}(t), \quad W_{31}^{PV_2}(t) = \frac{\eta n}{2} W_{31}^{PV}(t), \quad W_{31}^{PV_3}(t) = -\frac{\eta n}{2} W_{31}^{PV}(t)$$
$$W_{13}^{KQ_1}(t) = W_{13}^{KQ}(t), \quad W_{13}^{KQ_2}(t) = W_{13}^{KQ}(t), \quad W_{13}^{KQ_3}(t) = W_{13}^{KQ}(t).$$

The weight matrices $W_{31}^{PV}(t)$ and $W_{13}^{KQ}(t)$ are updated with the combined gradient of the corresponding three weights. Further, $W_{31}^{PV}(0) = \sum_{i=1}^d \lambda_i^{PV} u_i u_i^\top$ and $W_{13}^{KQ}(0) = \sum_{i=1}^d \lambda_i^{KQ} u_i u_i^\top$ share the same orthonormal basis, which is symmetric and simultaneously diagonalizable. We fix $W_{24}^{KQ_1} = 0, W_{24}^{KQ_2} = 1, W_{24}^{KQ_3} = -1$, and $W_{14}^{KQ_1} = 0_d, W_{14}^{KQ_2} = 0_d, W_{14}^{KQ_3} = 0_d$ for all $t > 0$. We take the initialization to be isotropic, $\lambda_i^{PV} = \lambda_V(0)$ and $\lambda_i^{KQ} = \lambda_K(0)$ for all $i \in [d]$, where for a constant $\sigma \in (0, 1/4)$,

$$\lambda_K(0) \in \Big[\tfrac{1}{\sqrt{3}} + \sigma,\ 1\Big], \qquad \lambda_V(0) \in (0, 1/\sigma], \qquad \lambda_V(0)^2 \ \geq\ F\big(\lambda_K(0)^2\big) + \sigma^2,$$

where $F(x) := \frac{x-1}{3} + \frac{2}{9}\log\frac{2}{3x-1}$.

We consider the gradient flow trajectory:

$$\frac{dW}{dt} = -\nabla L^{\mathrm{CoT}}(W), \text{ where}$$
$$W := [W^{KQ_1}/\{W_{24}^{KQ_1}, W_{14}^{KQ_1}\}, W^{KQ_2}/\{W_{24}^{KQ_2}, W_{14}^{KQ_2}\}, W^{KQ_3}/\{W_{24}^{KQ_3}, W_{14}^{KQ_3}\},$$
$$W^{PV_1}, W^{PV_2}, W^{PV_3}].$$

Next, we prove that if we initialize those matrices as in Assumption A.8, then the zero blocks will stay at 0 along the gradient descent trajectory.

Lemma A.9. *Under Assumption A.8, when the transformer is trained under gradient flow, $\forall t > 0$,*

$$W^{PV_1}(t) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ W_{31}^{PV_1}(t) & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}, W^{KQ_1}(t) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & W_{13}^{KQ_1}(t) & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}$$

$$W^{PV_2}(t) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ W_{31}^{PV_2}(t) & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}, W^{KQ_2}(t) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & W_{13}^{KQ_2}(t) & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 1 \\ 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}$$

$$W^{PV_3}(t) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ W_{31}^{PV_3}(t) & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}, W^{KQ_3}(t) = \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & W_{13}^{KQ_3}(t) & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & -1 \\ 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix}$$

Proof. We show that when the irrelevant blocks are 0, the gradients for those blocks are always 0:

$$\begin{aligned}
&W^{PV_1} Z_i \cdot \phi_1\Big(\tfrac{Z_i^\top W^{KQ_1} Z_{i[:,-1]}}{n}\Big) \\
=&\frac{1}{n^3} \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ W_{31}^{PV_1}(t) & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix} \begin{bmatrix} X & 0 & \cdots & 0 \\ y & 0 & \cdots & 0 \\ 0_d & \theta_0 & \cdots & \theta_i \\ 0 & 1 & \cdots & 1 \end{bmatrix} \Big(Z_i^\top W^{KQ_1} \begin{bmatrix} 0 \\ 0 \\ \theta_i \\ 1 \end{bmatrix}\Big)^{\odot 3} \\
=&\frac{1}{n^3} \begin{bmatrix} 0_{d\times d} & 0_d & \dots & 0_d \\ 0_{1\times d} & 0 & \dots & 0 \\ W_{31}^{PV_1}(t) X & 0_d & \dots & 0_d \\ 0_{1\times d} & 0 & \dots & 0 \end{bmatrix} \Big(\begin{bmatrix} X & 0 & \cdots & 0 \\ y & 0 & \cdots & 0 \\ 0_d & \theta_0 & \cdots & \theta_i \\ 0 & 1 & \cdots & 1 \end{bmatrix}^\top \begin{bmatrix} W_{13}^{KQ_1}(t)\theta_i \\ 0 \\ 0_d \\ 0 \end{bmatrix}\Big)^{\odot 3} \\
=&\frac{1}{n^3} \begin{bmatrix} 0_{d\times d} & 0_d & \dots & 0_d \\ 0_{1\times d} & 0 & \dots & 0 \\ W_{31}^{PV_1}(t) X & 0_d & \dots & 0_d \\ 0_{1\times d} & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} X^\top W_{13}^{KQ_1}(t)\theta_i \\ 0_{i+1} \end{bmatrix}^{\odot 3} \\
=&\frac{1}{n^3} \begin{bmatrix} 0_d \\ 0 \\ W_{31}^{PV_1}(t) X \big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 3} \\ 0 \end{bmatrix}.
\end{aligned}$$

For the second head, we have

$$\begin{aligned}
&W^{PV_2} Z_i \cdot \phi_2\Big(\tfrac{Z_i^\top W^{KQ_2} Z_{i[:,-1]}}{n}\Big) \\
=&\frac{1}{n^2} \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ W_{31}^{PV_2}(t) & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix} \begin{bmatrix} X & 0 & \cdots & 0 \\ y & 0 & \cdots & 0 \\ 0_d & \theta_0 & \cdots & \theta_i \\ 0 & 1 & \cdots & 1 \end{bmatrix} \Big(Z_i^\top W^{KQ_2} \begin{bmatrix} 0 \\ 0 \\ \theta_i \\ 1 \end{bmatrix}\Big)^{\odot 2} \\
=&\frac{1}{n^2} \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ W_{31}^{PV_2}(t) X & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix} \Big(\begin{bmatrix} X & 0 & \cdots & 0 \\ y & 0 & \cdots & 0 \\ 0_d & \theta_0 & \cdots & \theta_i \\ 0 & 1 & \cdots & 1 \end{bmatrix}^\top \begin{bmatrix} W_{13}^{KQ_2}(t)\theta_i \\ 1 \\ 0_d \\ 0 \end{bmatrix}\Big)^{\odot 2} \\
=&\frac{1}{n^2} \begin{bmatrix} 0_{d\times d} & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \\ W_{31}^{PV_2}(t) X & 0_{d\times 1} & 0_{d\times d} & 0_{d\times 1} \\ 0_{1\times d} & 0 & 0_{d\times 1} & 0 \end{bmatrix} \begin{bmatrix} X^\top W_{13}^{KQ_2}(t)\theta_i + y^\top \\ 0_{i+1} \end{bmatrix}^{\odot 2} \\
=&\frac{1}{n^2} \begin{bmatrix} 0_d \\ 0 \\ W_{31}^{PV_2}(t) X \big(X^\top W_{13}^{KQ_2}(t)\theta_i + y^\top\big)^{\odot 2} \\ 0 \end{bmatrix}.
\end{aligned}$$

Similarly, for the third head, we have

$$W^{PV_3}Z_i \cdot \phi_3\Big(\frac{Z_i^\top W^{KQ_3} Z_{i[:,-1]}}{n}\Big) = \frac{1}{n^2}\begin{bmatrix} 0_d \\ 0 \\ W_{31}^{PV_3}(t)X\big(X^\top W_{13}^{KQ_3}(t)\theta_i - y^\top\big)^{\odot 2} \\ 0 \end{bmatrix}.$$

Recall that we have the residual for a CoT step defined as

$$r_i = \sum_{h=1}^{3} W^{PV_h}Z_i \cdot \phi_h\Big(\frac{Z_i^\top W^{KQ_h} Z_{i[:,-1]}}{n}\Big) - (0_d, 0, \theta_{i+1}-\theta_i, 0)^\top = (0_d, 0, \widehat{r}_i - (\theta_{i+1}-\theta_i), 0)^\top,$$

where we define

$$\begin{aligned}\widehat{r}_i &= \frac{1}{n^3}W_{31}^{PV_1}X\big(X^\top W_{13}^{KQ_1}\theta_i\big)^{\odot 3} + \frac{1}{n^2}W_{31}^{PV_2}X\big(X^\top W_{13}^{KQ_2}\theta_i + y^\top\big)^{\odot 2} \\ &\quad + \frac{1}{n^2}W_{31}^{PV_3}X\big(X^\top W_{13}^{KQ_3}\theta_i - y^\top\big)^{\odot 2} \\ &= \frac{1}{n^3}W_{31}^{PV_1}X\big(X^\top W_{13}^{KQ_1}\theta_i\big)^{\odot 3} + \frac{4}{n^2}W_{31}^{PV_2}X\big(X^\top W_{13}^{KQ_2}\theta_i \odot y^\top\big). \end{aligned} \tag{31}$$

The second step is because of the symmetry in Assumption A.8 that $W_{31}^{PV_2} = -W_{31}^{PV_3}$ and $W_{13}^{KQ_2} = W_{13}^{KQ_3}$.

Now, we consider the gradient for W^{PV_h}. Denote the loss at step i as $L_i(W)$. For an X and $\theta^\star$,

$$\begin{aligned}\nabla_{W^{PV_1}}L_i(W) &= r_i\big(Z_i\phi_1(\tfrac{Z_i^\top W^{KQ_1}Z_{i[:,-1]}}{n})\big)^\top = \frac{1}{n^3}r_i\Big(Z_i\begin{bmatrix}X^\top W_{13}^{KQ_1}(t)\theta_i \\ 0_{i+1}\end{bmatrix}^{\odot 3}\Big)^\top \\ &= \frac{1}{n^3}\begin{bmatrix}0_d \\ 0 \\ \widehat{r}_i - (\theta_{i+1}-\theta_i) \\ 0\end{bmatrix}\begin{bmatrix}X\big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 3} \\ y\big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 3} \\ 0_d \\ 0\end{bmatrix}^\top \\ &= \frac{1}{n^3}\begin{bmatrix}0_{d\times d} & 0_d & 0_{d\times d} & 0_d \\ 0_{1\times d} & 0 & 0_{1\times d} & 0 \\ \nabla_{W_{31}^{PV_1}}L_i & \nabla_{W_{32}^{PV_1}}L_i & 0_{d\times d} & 0_d \\ 0_{1\times d} & 0 & 0_{1\times d} & 0\end{bmatrix}\end{aligned}$$

where

$$\begin{aligned}\nabla_{W_{31}^{PV_1}}L_i &= \widehat{r}_i\big(X\big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 3}\big)^\top - (\theta_{i+1}-\theta_i)\big(X\big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 3}\big)^\top \\ \nabla_{W_{32}^{PV_1}}L_i &= \widehat{r}_i\big(y\big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 3}\big)^\top - (\theta_{i+1}-\theta_i)\big(y\big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 3}\big)^\top.\end{aligned}$$

Similarly, we calculate the gradient of W^{PV_2}. We have

$$\begin{aligned}\nabla_{W^{PV_2}}L_i(W) &= r_i\big(Z_i\phi_2(\tfrac{Z_i^\top W^{KQ_2}Z_{i[:,-1]}}{n})\big)^\top = \frac{1}{n^2}r_i\Big(Z_i\begin{bmatrix}X^\top W_{13}^{KQ_2}(t)\theta_i + y^\top \\ 0_{i+1}\end{bmatrix}^{\odot 2}\Big)^\top \\ &= \frac{1}{n^2}\begin{bmatrix}0_d \\ 0 \\ \widehat{r}_i - (\theta_{i+1}-\theta_i) \\ 0\end{bmatrix}\begin{bmatrix}X\big(X^\top W_{13}^{KQ_2}(t)\theta_i + y^\top\big)^{\odot 2} \\ y\big(X^\top W_{13}^{KQ_2}(t)\theta_i + y^\top\big)^{\odot 2} \\ 0_d \\ 0\end{bmatrix}^\top \\ &= \frac{1}{n^2}\begin{bmatrix}0_{d\times d} & 0_d & 0_{d\times d} & 0_d \\ 0_{1\times d} & 0 & 0_{1\times d} & 0 \\ \nabla_{W_{31}^{PV_2}}L_i & \nabla_{W_{32}^{PV_2}}L_i & 0_{d\times d} & 0_d \\ 0_{1\times d} & 0 & 0_{1\times d} & 0\end{bmatrix}, \text{ where}\end{aligned}$$

$$\begin{aligned}\nabla_{W_{31}^{PV_2}}L_i &= \widehat{r}_i\big(X\big(X^\top W_{13}^{KQ_2}(t)\theta_i + y^\top\big)^{\odot 2}\big)^\top - (\theta_{i+1}-\theta_i)\big(X\big(X^\top W_{13}^{KQ_2}(t)\theta_i + y^\top\big)^{\odot 2}\big)^\top \\ \nabla_{W_{32}^{PV_2}}L_i &= \widehat{r}_i\big(y\big(X^\top W_{13}^{KQ_2}(t)\theta_i + y^\top\big)^{\odot 2}\big)^\top - (\theta_{i+1}-\theta_i)\big(y\big(X^\top W_{13}^{KQ_2}(t)\theta_i + y^\top\big)^{\odot 2}\big)^\top.\end{aligned}$$

Also, the gradient for W^{PV_3} is as follows:

$$\begin{aligned}
\nabla_{W^{PV_3}} L_i(W) &= r_i\big(Z_i\phi_3(\tfrac{Z_i^\top W^{KQ_3} Z_{i[:,-1]}}{n})\big)^\top = \frac{1}{n^2} r_i\big(Z_i \begin{bmatrix} X^\top W_{13}^{KQ_3}(t)\theta_i - y^\top \\ 0_{i+1} \end{bmatrix}^{\odot 2}\big)^\top \\
&= \frac{1}{n^2}\begin{bmatrix} 0_d \\ 0 \\ \widehat{r}_i - (\theta_{i+1} - \theta_i) \\ 0 \end{bmatrix} \begin{bmatrix} X\big(X^\top W_{13}^{KQ_3}(t)\theta_i - y^\top\big)^{\odot 2} \\ y\big(X^\top W_{13}^{KQ_3}(t)\theta_i - y^\top\big)^{\odot 2} \\ 0_d \\ 0 \end{bmatrix}^\top \\
&= \frac{1}{n^2}\begin{bmatrix} 0_{d\times d} & 0_d & 0_{d\times d} & 0_d \\ 0_{1\times d} & 0 & 0_{1\times d} & 0 \\ \nabla_{W_{31}^{PV_3}} L_i & \nabla_{W_{32}^{PV_3}} L_i & 0_{d\times d} & 0_d \\ 0_{1\times d} & 0 & 0_{1\times d} & 0 \end{bmatrix}, \text{ where} \\
\nabla_{W_{31}^{PV_3}} L_i &= \widehat{r}_i\big(X\big(X^\top W_{13}^{KQ_3}(t)\theta_i - y^\top\big)^{\odot 2}\big)^\top - (\theta_{i+1} - \theta_i)\big(X\big(X^\top W_{13}^{KQ_3}(t)\theta_i - y^\top\big)^{\odot 2}\big)^\top \\
\nabla_{W_{32}^{PV_3}} L_i &= \widehat{r}_i\big(y\big(X^\top W_{13}^{KQ_3}(t)\theta_i - y^\top\big)^{\odot 2}\big)^\top - (\theta_{i+1} - \theta_i)\big(y\big(X^\top W_{13}^{KQ_3}(t)\theta_i - y^\top\big)^{\odot 2}\big)^\top.
\end{aligned}$$

Next, we calculate the gradient for W^{KQ_h}. We write $r_i^\theta := \widehat{r}_i - (\bar{\theta}_{i+1} - \theta_i)$ for the θ-block of r_i. Thus, $r_i = (0_d, 0, r_i^\theta, 0)$, and $Z_i^\top W^{PV_h\top} r_i = [X^\top W_{31}^{PV_h\top} r_i^\theta;\, 0_{i+1}]$. Then, we have

$$\begin{aligned}
\nabla_{W^{KQ_1}} L_i &= \frac{1}{n} Z_i\Big(\big(Z_i^\top W^{PV_1\top} r_i\big) \odot 3(\tfrac{Z_i^\top W^{KQ_1} Z_{i[:,-1]}}{n})^{\odot 2}\Big) Z_{i[:,-1]}^\top \\
&= \frac{1}{n} Z_i\Big(\begin{bmatrix} X^\top W_{31}^{PV_1\top}(t) r_i^\theta \\ 0_{i+1} \end{bmatrix} \odot \frac{3}{n^2}\begin{bmatrix} \big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 2} \\ 0_{i+1} \end{bmatrix}\Big) Z_{i[:,-1]}^\top \\
&= \frac{3}{n^3} Z_i \begin{bmatrix} \Big(X^\top W_{31}^{PV_1\top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 2}\Big) \\ 0_{i+1} \end{bmatrix} Z_{i[:,-1]}^\top \\
&= \frac{3}{n^3}\begin{bmatrix} X\Big(X^\top W_{31}^{PV_1\top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 2}\Big) \\ y\Big(X^\top W_{31}^{PV_1\top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 2}\Big) \\ 0_d \\ 0 \end{bmatrix} \begin{bmatrix} 0_d & 0 & \theta_i^\top & 1 \end{bmatrix} \\
&= \frac{3}{n^3}\begin{bmatrix} 0_{d\times d} & 0_d & \nabla_{W_{13}^{KQ_1}} L_i & \nabla_{W_{14}^{KQ_1}} L_i \\ 0_{1\times d} & 0 & \nabla_{W_{23}^{KQ_1}} L_i & \nabla_{W_{24}^{KQ_1}} L_i \\ 0_{d\times d} & 0_d & 0_{d\times d} & 0 \\ 0_{1\times d} & 0 & 0_{1\times d} & 0 \end{bmatrix}, \text{ where} \\
\nabla_{W_{13}^{KQ_1}} L_i &= X\Big(X^\top W_{31}^{PV_1\top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 2}\Big)\theta_i^\top \\
\nabla_{W_{23}^{KQ_1}} L_i &= y\Big(X^\top W_{31}^{PV_1\top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_1}(t)\theta_i\big)^{\odot 2}\Big)\theta_i^\top,
\end{aligned}$$

and we fix $W_{24}^{KQ_1} = 0$ and $W_{14}^{KQ_1} = 0_d$ for $t > 0$. Then, for head two, we have

$$\begin{aligned}
\nabla_{W^{KQ_2}} L_i &= \frac{1}{n} Z_i \Big(\big(Z_i^\top W^{PV_2 \top} r_i \big) \odot 2 (\tfrac{Z_i^\top W^{KQ_2} Z_{i[:,-1]}}{n}) \Big) Z_{i[:,-1]}^\top \\
&= \frac{1}{n} Z_i \Big(\begin{bmatrix} X^\top W_{31}^{PV_2 \top}(t) r_i^\theta \\ 0_{i+1} \end{bmatrix} \odot \frac{2}{n} \begin{bmatrix} \big(X^\top W_{13}^{KQ_2}(t) \theta_i + y^\top \big) \\ 0_{i+1} \end{bmatrix} \Big) Z_{i[:,-1]}^\top \\
&= \frac{2}{n^2} Z_i \begin{bmatrix} \Big(X^\top W_{31}^{PV_2 \top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_2}(t) \theta_i + y^\top \big) \Big) \\ 0_{i+1} \end{bmatrix} Z_{i[:,-1]}^\top \\
&= \frac{2}{n^2} \begin{bmatrix} X \Big(X^\top W_{31}^{PV_2 \top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_2}(t) \theta_i + y^\top \big) \Big) \\ y \Big(X^\top W_{31}^{PV_2 \top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_2}(t) \theta_i + y^\top \big) \Big) \\ 0_d \\ 0 \end{bmatrix} \begin{bmatrix} 0_d & 0 & \theta_i^\top & 1 \end{bmatrix} \\
&= \frac{2}{n^2} \begin{bmatrix} 0_{d \times d} & 0_d & \nabla_{W_{13}^{KQ_2}} L_i & \nabla_{W_{14}^{KQ_2}} L_i \\ 0_{1 \times d} & 0 & \nabla_{W_{23}^{KQ_2}} L_i & \nabla_{W_{24}^{KQ_2}} L_i \\ 0_{d \times d} & 0_d & 0_{d \times d} & 0 \\ 0_{1 \times d} & 0 & 0_{1 \times d} & 0 \end{bmatrix}, \text{ where} \\
\nabla_{W_{13}^{KQ_2}} L_i &= X \Big(X^\top W_{31}^{PV_2 \top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_2}(t) \theta_i + y^\top \big) \Big) \theta_i^\top \\
\nabla_{W_{23}^{KQ_2}} L_i &= y \Big(X^\top W_{31}^{PV_2 \top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_2}(t) \theta_i + y^\top \big) \Big) \theta_i^\top,
\end{aligned}$$

and we fix $W_{24}^{KQ_2} = 1$ and $W_{14}^{KQ_2} = 0_d$ for $t > 0$. Similarly, we have the gradient for W^{KQ_3} as follows.

$$\begin{aligned}
\nabla_{W_{13}^{KQ_3}} L_i &= X \Big(X^\top W_{31}^{PV_3 \top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_3}(t) \theta_i - y^\top \big) \Big) \theta_i^\top \\
\nabla_{W_{23}^{KQ_3}} L_i &= y \Big(X^\top W_{31}^{PV_3 \top}(t) r_i^\theta \odot \big(X^\top W_{13}^{KQ_3}(t) \theta_i - y^\top \big) \Big) \theta_i^\top,
\end{aligned}$$

and we fix $W_{24}^{KQ_3} = -1$ and $W_{14}^{KQ_3} = 0_d$ for $t > 0$.

Now, we are going to show that the expected gradients with respect to the irrelevant blocks are zero, i.e., $\nabla_{W_{32}^{PV_1}} L_i$, $\nabla_{W_{32}^{PV_2}} L_i$, $\nabla_{W_{32}^{PV_3}} L_i$, $\nabla_{W_{23}^{KQ_1}} L_i$, $\nabla_{W_{23}^{KQ_2}} L_i$, $\nabla_{W_{23}^{KQ_3}} L_i$. We will show that $\mathbb{E}_{X,\theta^\star} \left[\nabla_{W_{32}^{PV_1}} L_i \right] = 0$, and proofs for others follow similarly.

We use the sign-flip symmetry of the Gaussian distribution to prove this fact. Denote $g_i := y^\top a_i^{\odot 3}$ and $a_i := X^\top W_{13}^{KQ_1} \theta_i$. Define a sign-flip transformation $\mathcal{T} : (X, \theta^\star) \mapsto (-X, \theta^\star)$. Since X has i.i.d. standard Gaussian columns, the distribution of $(X, \theta^\star)$ is invariant under $\mathcal{T}$.

The parameter token θ_i and the target increment $\theta_{i+1} - \theta_i$ used in Z_i are generated by the deterministic gradient descent on the quadratic squared loss. Thus, θ_i is invariant under $\mathcal{T}$. Then, based on the definitions, we have y is invariant under $\mathcal{T}$, and g_i will be flipped to $-g_i$ under $\mathcal{T}$.

Moreover, we can show $\widehat{r}_i$ is even under $\mathcal{T}$. Under the block expressions, we have

$$\begin{aligned}
r_i^\theta &= \widehat{r}_i - (\theta_{i+1} - \theta_i) \\
&= \frac{1}{n^3} W_{31}^{PV_1} X \big(X^\top W_{13}^{KQ_1} \theta_i \big)^{\odot 3} + \frac{4}{n^2} W_{31}^{PV_2} X \big(X^\top W_{13}^{KQ_2} \theta_i \odot y^\top \big) - (\theta_{i+1} - \theta_i)
\end{aligned}$$

One can verify that r_i^θ is invariant under the transformation $\mathcal{T}$. Thus, $\nabla_{W_{32}^{PV_1}} L_i(W) = r_i g_i$ is odd under $\mathcal{T}$. Combining the above sign-flipping results and using the distributional invariance, we have

$$\mathop{\mathbb{E}}_{X,\theta^\star} \left[\nabla_{W_{32}^{PV_1}} L_i(W) \right] = \mathop{\mathbb{E}}_{\mathcal{T}(X,\theta^\star)} \left[\nabla_{W_{32}^{PV_1}} L_i(W) \circ \mathcal{T} \right] = - \mathop{\mathbb{E}}_{X,\theta^\star} \left[\nabla_{W_{32}^{PV_1}} L_i(W) \right],$$

Therefore, $\mathbb{E}_{X,\theta^\star} \left[\nabla_{W_{32}^{PV_1}} L_i(W) \right] = 0$. □

Notice that the final loss only depends on the $(d+2)$ to $(2d+1)$ entries of the transformer's output. We can simplify the model by pruning the irrelevant and zero blocks. We can define a simplified transformer, where the dynamics of the equivalent model are exactly the same as the original model.

Definition A.10. Under Assumption A.8, by Lemma A.9, for a given input Z_i, we can rewrite the one-layer non-linear transformer by

$$f_W(Z_i) = \Big[0_d, 0, \big(\theta_i - \frac{2\eta}{n} W_{31}^{PV} X \big(X^\top W_{13}^{KQ} \theta_i\big)^{\odot 3} + \frac{2\eta}{n} W_{31}^{PV} X \big(X^\top W_{13}^{KQ} \theta_i \odot y^\top\big)\big), 1\Big]^\top.$$

where θ_i is the last column of $Z_{i[d+2:2d+1,-1]}$ and $y^\top = (X^\top \theta^\star)^{\odot 2}$ where $\theta^\star$ is the ground-truth weights of the function. Then, we can denote a reduced model by:

$$\tilde{f}_W(Z_i) = \theta_i - \frac{2\eta}{n} W_{31}^{PV} X \big(X^\top W_{13}^{KQ} \theta_i\big)^{\odot 3} + \frac{2\eta}{n} W_{31}^{PV} X \big(X^\top W_{13}^{KQ} \theta_i \odot y^\top\big). \tag{32}$$

For the ease of notations, we will write $\tilde{f}_W(Z_i) = \tilde{f}(\theta_i)$, where θ_i denotes the $(d+2)$ to $(2d+1)$ entries of the last column of Z_i.

In the following sections, we will consider the equivalent form of the transformer. Here we present the gradient with respect to the reduced model.

Lemma A.11. *The gradients of W_{31}^{PV} and W_{13}^{KQ} are given by the following expectations:*

$$\begin{aligned}
&\nabla_{W_{31}^{PV}} L^{\mathrm{CoT}}(W) \\
=&\sum_{i=0}^{T-1} \mathop{\mathbb{E}}_{X,\theta^\star}\left[-\frac{2\eta}{n}(\tilde{f}_W(\theta_i) - \theta_{i+1})\left(\big(X(X^\top W_{13}^{KQ}\theta_i)^{\odot 3}\big)^\top - \big(X\big(X^\top W_{13}^{KQ}\theta_i \odot y^\top\big)\big)^\top\right)\right] \\
&+ \mathop{\mathbb{E}}_{X,\theta^\star}\left[-\frac{2\eta}{n}(\theta_T - \theta^\star)\left(\big(X(X^\top W_{13}^{KQ}\theta_{T-1})^{\odot 3}\big)^\top - \big(X\big(X^\top W_{13}^{KQ}\theta_{T-1} \odot y^\top\big)\big)^\top\right)\right], \text{ and} \\
&\nabla_{W_{13}^{KQ}} L^{\mathrm{CoT}}(W) \\
=&\sum_{i=0}^{T-1} \mathop{\mathbb{E}}_{X,\theta^\star}\left[-\frac{2\eta}{n} X\Big(X^\top W_{31}^{PV\top}(\tilde{f}_W(\theta_i) - \theta_{i+1}) \odot \big(3\big(X^\top W_{13}^{KQ}\theta_i\big)^{\odot 2} - y^\top\big)\Big)\theta_i^\top\right] \\
&+ \mathop{\mathbb{E}}_{X,\theta^\star}\left[-\frac{2\eta}{n} X\Big(X^\top W_{31}^{PV\top}(\theta_T - \theta^\star) \odot \big(3\big(X^\top W_{13}^{KQ}\theta_{T-1}\big)^{\odot 2} - y^\top\big)\Big)\theta_{T-1}^\top\right].
\end{aligned}$$

Proof. Based on the simplified form of the transformer in Definition A.10, we can rewrite the CoT training loss as follows.

$$\begin{aligned}
L^{\mathrm{CoT}}(W) &= \frac{1}{2} \mathop{\mathbb{E}}_{X,\theta^\star}\left[\sum_{i=0}^{T-2} \|f_W(Z_i) - (0_d, 0, \theta_{i+1}, 1)\|_2^2\right] + \frac{1}{2} \mathop{\mathbb{E}}_{X,\theta^\star}\left[\|f_W(Z_{T-1}) - (0_d, 0, \theta^\star, 1)\|_2^2\right] \\
&= \frac{1}{2} \mathop{\mathbb{E}}_{X,\theta^\star}\left[\sum_{i=0}^{T-2} \|\tilde{f}_W(\theta_i) - \theta_{i+1}\|_2^2\right] + \frac{1}{2} \mathop{\mathbb{E}}_{X,\theta^\star}\left[\|\tilde{f}_W(\theta_{T-1}) - \theta^\star\|_2^2\right].
\end{aligned}$$

Here, we split the loss between the last step output and the ground-truth weights $\theta^\star$.

Then, based on the gradient computation in Lemma A.9, we can write out the gradient for the weight matrix W^{PV} in the transformer. Recall that in Assumption A.8, we use a weight-sharing mechanism that defines $W_{31}^{PV_1}(t) = -2\eta n^2 W_{31}^{PV}(t), W_{31}^{PV_2}(t) = \frac{\eta n}{2} W_{31}^{PV}(t), W_{31}^{PV_3}(t) = -\frac{\eta n}{2} W_{31}^{PV}(t)$.

Therefore,

$$
\begin{aligned}
&\nabla_{W_{31}^{PV}} L^{\mathrm{CoT}}(W) \\
=&-2\eta n^2 \nabla_{W_{31}^{PV_1}} L^{\mathrm{CoT}}(W) + \frac{\eta n}{2} \nabla_{W_{31}^{PV_2}} L^{\mathrm{CoT}}(W) - \frac{\eta n}{2} \nabla_{W_{31}^{PV_3}} L^{\mathrm{CoT}}(W) \\
=&\sum_{i=0}^{T-2} \mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} (\tilde{f}_W(\theta_i) - \theta_{i+1}) \big(X (X^\top W_{13}^{KQ} \theta_i)^{\odot 3} \big)^\top \right] \\
&+ \mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} (\tilde{f}_W(\theta_{T-1}) - \theta^\star) \big(X (X^\top W_{13}^{KQ} \theta_{T-1})^{\odot 3} \big)^\top \right] \\
&- \sum_{i=0}^{T-2} \mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} (\tilde{f}_W(\theta_i) - \theta_{i+1}) \big(X \big(X^\top W_{13}^{KQ} \theta_i \odot y^\top \big) \big)^\top \right] \\
&- \mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} (\tilde{f}_W(\theta_{T-1}) - \theta^\star) \big(X \big(X^\top W_{13}^{KQ} \theta_{T-1} \odot y^\top \big) \big)^\top \right] \\
=&\sum_{i=0}^{T-1} \mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} (\tilde{f}_W(\theta_i) - \theta_{i+1}) \left(\big(X (X^\top W_{13}^{KQ} \theta_i)^{\odot 3} \big)^\top - \big(X \big(X^\top W_{13}^{KQ} \theta_i \odot y^\top \big) \big)^\top \right) \right] \\
&+ \mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} (\theta_T - \theta^\star) \big(X (X^\top W_{13}^{KQ} \theta_{T-1})^{\odot 3} \big)^\top + \frac{2\eta}{n} (\theta_T - \theta^\star) \big(X \big(X^\top W_{13}^{KQ} \theta_{T-1} \odot y^\top \big) \big)^\top \right].
\end{aligned}
$$

The last step is because we subtract

$$
\mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} (\tilde{f}_W(\theta_{T-1}) - \theta_T) \big(X (X^\top W_{13}^{KQ} \theta_{T-1})^{\odot 3} \right])^\top
$$

from the second term and put it into the first term, then subtract

$$
\mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} (\tilde{f}_W(\theta_{T-1}) - \theta_T) \big(X \big(X^\top W_{13}^{KQ} \theta_{T-1} \odot y^\top \big) \right]
$$

from the fourth term and put it into the third term.

Similarly, we can break down the gradient for W_{13}^{KQ}. Recall that in Assumption A.8, we use a weight-sharing mechanism that defines $W_{13}^{KQ_1}(t) = W_{13}^{KQ}(t), W_{13}^{KQ_2}(t) = W_{13}^{KQ}(t), W_{13}^{KQ_3}(t) = W_{13}^{KQ}(t)$. Therefore, we have

$$
\begin{aligned}
&\nabla_{W_{13}^{KQ}} L^{\mathrm{CoT}}(W) \\
=&\nabla_{W_{13}^{KQ_1}} L^{\mathrm{CoT}}(W) + \nabla_{W_{13}^{KQ_2}} L^{\mathrm{CoT}}(W) + \nabla_{W_{13}^{KQ_3}} L^{\mathrm{CoT}}(W) \\
=&\sum_{i=0}^{T-1} \mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{6\eta}{n} X \Big(X^\top W_{31}^{PV\top} (\tilde{f}_W(\theta_i) - \theta_{i+1}) \odot \big(X^\top W_{13}^{KQ} \theta_i \big)^{\odot 2} \Big) \theta_i^\top \right] + \\
&\mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{6\eta}{n} X \Big(X^\top W_{31}^{PV\top} (\theta_T - \theta^\star) \odot \big(X^\top W_{13}^{KQ} \theta_{T-1} \big)^{\odot 2} \Big) \theta_{T-1}^\top \right] - \\
&\sum_{i=0}^{T-1} \mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} X \Big(X^\top W_{31}^{PV\top} (\tilde{f}_W(\theta_i) - \theta_{i+1}) \odot y^\top \Big) \theta_i^\top \right] - \\
&\mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} X \Big(X^\top W_{31}^{PV\top} (\theta_T - \theta^\star) \odot y^\top \Big) \theta_{T-1}^\top \right] \\
=&\sum_{i=0}^{T-1} \mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} X \Big(X^\top W_{31}^{PV\top} (\tilde{f}_W(\theta_i) - \theta_{i+1}) \odot \big(3 \big(X^\top W_{13}^{KQ} \theta_i \big)^{\odot 2} - y^\top \big) \Big) \theta_i^\top \right] + \\
&\mathop{\mathbb{E}}_{X,\theta^\star} \left[-\frac{2\eta}{n} X \Big(X^\top W_{31}^{PV\top} (\theta_T - \theta^\star) \odot \big(3 \big(X^\top W_{13}^{KQ} \theta_{T-1} \big)^{\odot 2} - y^\top \big) \Big) \theta_{T-1}^\top \right].
\end{aligned}
$$

□

A.3.3 Proof of Theorem 3.4

With the above computed gradients, we now prove the main theorem on the convergence of transformers trained on quadratic functions. We characterize the training dynamics using ordinary differential equations in terms of the eigenvalues of the transformer weights. We first characterize the gradient of the CoT objective as a main term plus interaction terms, which yields an ordinary differential equation for the eigenvalues of the two trained blocks. Then, we analyze the ordinary differential equation in stages.

Proof. We first use the gradient computation in Lemma A.11 and derive ordinary differential equations for the eigenvalues of the weight matrices W_{31}^{PV} and W_{13}^{KQ}. By Assumption A.8, we have weight matrices in the form for all $t \geq 0$:

$$W_{31}^{PV}(t) = \lambda_V(t)\,\mathrm{Id}_d, \qquad W_{13}^{KQ}(t) = \lambda_K(t)\,\mathrm{Id}_d,$$

where the dynamics reduce to the two eigenvalues $\lambda_V(t)$ and $\lambda_K(t)$. Consider that the gradient descent initialization satisfies $\theta_0 = \alpha_0\theta^\star$ with a constant $\alpha_0 \in [3/4, 1)$, and take $n = \Theta(d^2 \log^5 d)$. We will show that the gradients in the eigenbasis admit the following decomposition

$$\nabla_{W_{31}^{PV}} L^{\mathrm{CoT}}(W) = \big(g_V(\lambda_V, \lambda_K) + \Delta^V\big)\,\mathrm{Id}_d,$$
$$\nabla_{W_{13}^{KQ}} L^{\mathrm{CoT}}(W) = \big(g_K(\lambda_V, \lambda_K) + \Delta^K\big)\,\mathrm{Id}_d,$$

where g_V, g_K are explicit main terms, and Δ^V, Δ^K are interaction terms. The induced eigenvalue ODEs can be represented by

$$\dot\lambda_V(t) = -g_V\big(\lambda_V(t), \lambda_K(t)\big) - \Delta^V(t), \qquad \dot\lambda_K(t) = -g_K\big(\lambda_V(t), \lambda_K(t)\big) - \Delta^K(t),$$

In particular, by expanding the main terms around the construction $\lambda_V = \lambda_K = 1$, we will show that

$$\begin{aligned} g_V(\lambda_V, \lambda_K) &= 36\kappa^2 \sum_{i=0}^{T-1} \beta_i(1-\beta_i)\big[(1-\beta_i)(\lambda_V - 1) + (1-3\beta_i)(\lambda_K - 1)\big] \\ &\quad + O\big(|\lambda_V - 1|^2 + |\lambda_K - 1|^2\big), \\ g_K(\lambda_V, \lambda_K) &= 36\kappa^2 \sum_{i=0}^{T-1} \beta_i(1-3\beta_i)\big[(1-\beta_i)(\lambda_V - 1) + (1-3\beta_i)(\lambda_K - 1)\big] \\ &\quad + O\big(|\lambda_V - 1|^2 + |\lambda_K - 1|^2\big), \end{aligned}$$

where the coefficients depend only on (T, κ, α_0), and the weights $\beta_i \in [\alpha_0^2, 1)$.

Now, we derive the explicit forms of the ODE. Based on Lemma A.9, if the parameters are initialized as in Assumption A.8, then under gradient flow all zero blocks remain zero for all $t > 0$. Hence, the model dynamics are fully characterized by the two matrices $W_{31}^{PV}(t)$ and $W_{13}^{KQ}(t)$. Then, we write the reduced form of the transformer as in equation (32).

The chain-of-thought training loss can be written as

$$L^{\mathrm{CoT}}(W) = \frac{1}{2}\mathbb{E}_{X,\theta^\star}\Big[\sum_{i=0}^{T-2} \|\tilde f_W(\theta_i) - \theta_{i+1}\|_2^2 + \|\tilde f_W(\theta_{T-1}) - \theta^\star\|_2^2\Big].$$

with the residuals defined as

$$e_i := \begin{cases} \tilde f_W(\theta_i) - \theta_{i+1}, & i = 0, \ldots, T-2, \\ \tilde f_W(\theta_{T-1}) - \theta^\star, & i = T-1. \end{cases}$$

Let

$$\Psi(\theta) := X\Big((X^\top W_{13}^{KQ}\theta)^{\odot 3} - (X^\top W_{13}^{KQ}\theta) \odot y^\top\Big).$$

Then the gradients of the reduced parameters are

$$\nabla_{W_{31}^{PV}} L^{\mathrm{CoT}}(W) = \mathbb{E}\Big[\sum_{i=0}^{T-1} -\frac{2\eta}{n} e_i \Psi(\theta_i)^\top\Big], \tag{33}$$

$$\nabla_{W_{13}^{KQ}} L^{\mathrm{CoT}}(W) = \mathbb{E}\Big[\sum_{i=0}^{T-1} -\frac{2\eta}{n} X\Big((X^\top W_{31}^{PV\top} e_i) \odot \big(3(X^\top W_{13}^{KQ}\theta_i)^{\odot 2} - y^\top\big)\Big)\theta_i^\top\Big]. \tag{34}$$

As in Assumption A.8, assume $W_{31}^{PV}(t)$ and $W_{13}^{KQ}(t)$ are symmetric and share an orthonormal eigenbasis U. Rotating all variables by $U^\top$ leaves the data distribution invariant, so we can work in the basis where $U = \mathrm{Id}_d$ and both matrices are diagonal. Thus, the isotropy is preserved. The dynamics reduce to ODEs for the eigenvalues $\lambda_V(t)$ and $\lambda_K(t)$.

Then, we further write the ODE as the expectation over $\theta^\star$ and X. Let $\theta^\star \sim \mathcal{N}(0, \mathrm{Id}_d)$ and initialize $\theta_0 = \alpha_0\theta^\star$ with $\alpha_0 \in [3/4, 1)$. Then, gradient descent on the quadratic functions preserves collinearity, so $\theta_i = \alpha_i\theta^\star,\ \alpha_{i+1} = \alpha_i - 6\eta\|\theta^\star\|_2^2\alpha_i(\alpha_i^2 - 1)$. Here, α_i are scalars that only depend on the data $\|\theta^\star\|^2$. Let $\bar{\alpha}_i$ follow the same recursion with $\|\theta^\star\|^2$ replaced by its mean d, and let $\beta_i := \bar{\alpha}_i^2$. The empirical iterates θ_i stay close to $\alpha_i\theta^\star$, and $\|\theta^\star\|^2$ is close to d.

Given $W_{31}^{PV} = \lambda_V \mathrm{Id}_d$ and $W_{13}^{KQ} = \lambda_K \mathrm{Id}_d$, let

$$s := 1 - \lambda_V\lambda_K^3, \qquad \tau := 1 - \lambda_V\lambda_K,$$

which both vanish at the construction $\lambda_V = \lambda_K = 1$. Then, we have

$$\Psi(\theta) = \lambda_K^3 X(X^\top\theta)^{\odot 3} - \lambda_K X\big((X^\top\theta)\odot y^\top\big),$$

for $i \le T-2$,

$$e_i = \frac{2\eta}{n}\Big(s\,X(X^\top\theta_i)^{\odot 3} - \tau\,X\big((X^\top\theta_i)\odot y^\top\big)\Big),$$

and e_{T-1} is the same expression plus $\theta_T - \theta^\star$. As the other terms do not depend on (λ_V, λ_K), L^{CoT} is exactly a quadratic function of (s, τ).

Then, we apply the expectation to the Gaussian moment. for $x \sim \mathcal{N}(0, \mathrm{Id}_d)$ and fixed vectors $v, w \in \mathbb{R}^d$, we have $\mathbb{E}[x(x^\top v)^3] = 3\|v\|^2 v, \qquad \mathbb{E}[x(x^\top v)(x^\top w)^2] = \|w\|^2 v + 2(v^\top w)w$. Applying these identities yields the population expectation

$$\mathbb{E}_X[\Psi(\theta) \mid \theta, \theta^\star] = n\Big(3\lambda_K^3\|\theta\|^2\theta - \lambda_K\big(\|\theta^\star\|^2\theta + 2(\theta^\top\theta^\star)\theta^\star\big)\Big). \tag{35}$$

In particular, at $\theta = \alpha\theta^\star$ the expectations of $\frac{1}{n}X(X^\top\theta)^{\odot 3}$ and $\frac{1}{n}X\big((X^\top\theta)\odot y^\top\big)$ are $3\alpha^3\|\theta^\star\|^2\theta^\star$ and $3\alpha\|\theta^\star\|^2\theta^\star$.

We keep only the population parts and replace $\|\theta^\star\|^2$ by d and α_i by $\bar{\alpha}_i$, which defines the main terms. The remaining terms are the interaction terms. Then, taking the exact expectation over $\theta^\star$ and X, the population gradient-flow dynamics of the eigenvalues are given by

$$\dot{\lambda}_V = 36\kappa^2\lambda_K \sum_{i=0}^{T-1} \beta_i(\beta_i s - \tau)(\lambda_K^2\beta_i - 1) - \Delta^V, \tag{36}$$

$$\dot{\lambda}_K = 36\kappa^2\lambda_V \sum_{i=0}^{T-1} \beta_i(\beta_i s - \tau)(3\lambda_K^2\beta_i - 1) - \Delta^K. \tag{37}$$

The interaction terms Δ^V and Δ^K are the differences between the exact expressions and the main terms in equation (36) and equation (37). On any bounded set of (λ_V, λ_K), their orders are as follows

$$|\Delta^V| + |\Delta^K| = O\Big(\frac{1}{\log d}\Big)\Big(\sum_{i=0}^{T-1}\beta_i|\beta_i s - \tau| + |s| + |\tau|\Big) + O\big(d^{-aC}\big).$$

The first part vanishes at the construction where $s = \tau = 0$. The second part is set by the length of training examples. Consequently, the population dynamics close exactly in terms of (λ_V, λ_K) and the deterministic weights β_i, up to the interaction terms, yielding a finite-dimensional ODE.

Now, we prove the convergence of training transformers on quadratic functions based on the above forms of ODEs. By Lemma A.9, the block-sparse structure is preserved under gradient flow, so the dynamics are fully characterized by the two reduced matrices $W_{31}^{PV}(t)$ and $W_{13}^{KQ}(t)$. We further use an isotropic symmetry: $W_{13}^{KQ}(t) = \lambda_K(t)\,\mathrm{Id}_d,\ W_{31}^{PV}(t) = \lambda_V(t)\,\mathrm{Id}_d, \forall t \ge 0$, in order to identify the fixed point for the ODE. Then, the model reduces to scalars $(\lambda_K(t), \lambda_V(t))$.

We now show the convergence in two stages. In the first stage, using the structure of the ODE and the condition (here enforced by the colinear initialization $\theta_0 = \alpha_0\theta^\star$ with $\alpha_0 \in [3/4, 1)$), we show

that whenever (λ_K, λ_V) is outside a small neighborhood of $(1, 1)$, the gradient norm is bounded below by a positive quantity depending at most polylogarithmically on d: $\|\nabla L^{\mathrm{CoT}}(\lambda_K, \lambda_V)\|_2^2 \geq c_1 L^{\mathrm{CoT}}(\lambda_K, \lambda_V)$, when $\|(\lambda_K, \lambda_V) - (1, 1)\| \geq \delta_0$, for some constants $c_1 > 0$ and $\delta_0 = O(\frac{1}{\log d})$. This yields an exponential decrease of L^{CoT} until the trajectory enters the δ_0-ball around $(1, 1)$, in time $t_1 = O(\log d)$.

In the second stage, inside the δ_0-ball of $(1, 1)$, L^{CoT} is smooth, and its Hessian is positive definite (this follows from the identifiability of the construction in the assumption). Hence a local PL-condition inequality holds: $\|\nabla L^{\mathrm{CoT}}(\lambda_K, \lambda_V)\|_2^2 \geq c_2 L^{\mathrm{CoT}}(\lambda_K, \lambda_V)$, when $\|(\lambda_K, \lambda_V) - (1, 1)\| \leq \delta_0$,for some $c_2 > 0$. This implies $L^{\mathrm{CoT}}(t) \leq L^{\mathrm{CoT}}(t_1) \exp\big(-c_2(t - t_1)\big)$, so reaching $L^{\mathrm{CoT}}(t) \leq \varepsilon$ takes an additional $t_2 = O(\log \frac{1}{\varepsilon})$ time. Thus, the transformer converges to the construction as described in Lemma 3.3. Following Lemma 3.3 and Theorem A.5, we similarly obtain the rate of the inference loss when training converges. □

B Omitted Experimental Results

B.1 Evaluations of the Stability Measure

Recall that Lemma 3.1 is a worst-case upper bound, and the figures in the main text compare it to the measured loss gap on separate axes. We report the difference in scale explicitly. For each model and each reasoning length T, we report six quantities:

- The loss gap $|L(f_W; \mathcal{D}) - L^\star(f_W; \mathcal{D})|$ between the inference loss and the reference loss;
- The value of our bound, i.e., the right-hand side of equation (4);
- The bound-to-gap ratio, i.e., the ratio of the previous two quantities;
- The first-order term $C_1 \sum_{t=1}^{T-1} \rho_t^{(T)} \varepsilon_t$;
- The second-order term $\frac{C_1 C_2}{2} \sum_{t=2}^{T} \rho_t^{(T)} \big(\sum_{i=1}^{t-1} r_i\big)^2$;
- The two constants entering the bound: the Lipschitz constant C_1 of the loss function, and the Lipschitz constant C_2 of the model's input Jacobian.

We estimate C_1 as the maximum norm of the loss gradient with respect to the model output over the evaluation set, and C_2 as the maximum finite-difference change in the input Jacobian over pairs of nearby inputs from the same set. Table 2 reports all six quantities for Qwen-1.5B, Llama-1B, and Gemma-2B trained on the Bellman-Ford task, for $T = 1, \ldots, 5$. At $T = 1$ there is no accumulated error, and both the gap and the bound are zero.

We report the results in Table 2. First, the bound is relatively tight on Qwen-1.5B, where the bound-to-gap ratio stays between 6 and 9 across all T. On Llama-1B and Gemma-2B, the ratio reaches 2.3×10^6 and 1.5×10^5 at $T = 5$. Second, this difference is attributed to the constants. C_1 is small and comparable across all three models (between 0.01 and 0.07), while C_2 differs by roughly three orders of magnitude, from around 5 on Qwen-1.5B to above 3×10^3 on Llama-1B. Since C_2 multiplies the squared accumulated deviations in the second-order term, a large C_2 increases the bound value. Third, the second-order term grows faster in T than the first-order term. For Llama-1B and Gemma-2B, it dominates the bound from $T = 3$ onward. On Qwen-1.5B, the first-order term remains the larger of the two.

In Figure 5, we measure the loss gap and the error amplification factor $\sum_{t=1}^{T-1} \rho_t^{(T)}$ for T ranging from 1 to 5. We report the results evaluated with Gemma-2B and Llama-1B models on the Bellman-Ford and Prim's algorithms. The inference loss increases significantly with the number of steps. The error amplification factor scales exponentially with T, closely tracking the growth in the inference loss.

Evaluations of the bound under quantization. Table 3 decomposes equation (4) across bit-widths for Qwen-1.5B on three tasks. Lowering the bit-width has two opposing effects on the bound. First, it reduces the error amplification factor: from full precision to 1 bit, $\sum_t \rho_t^{(T)}$ decreases by $3.7\times$ on the Dijkstra task, $5.9\times$ on the Prim task, and $1.2\times$ on the Bellman-Ford task. Second, the quantized model fits the training data less well, so the per-step prediction errors tend to grow.

Table 2: Decomposition of the bound in equation (4) for three models trained on the Bellman-Ford task. Loss gap denotes $|L(f_W;\mathcal{D}) - L^\star(f_W;\mathcal{D})|$, Bound denotes the right-hand side of equation (4), and Ratio denotes the bound value divided by the loss gap. The first- and second-order terms are $C_1 \sum_{t=1}^{T-1} \rho_t^{(T)} \varepsilon_t$ and $\frac{C_1 C_2}{2} \sum_{t=2}^{T} \rho_t^{(T)} \big(\sum_{i=1}^{t-1} r_i\big)^2$. The absolute scale of the bound varies by orders of magnitude across architectures. At $T = 1$ there is no accumulated error, so the loss gap and the bound value are zero.

					Terms of equation (4)		Constants	
Model	T	Loss gap	Bound	Ratio	First-order	Second-order	C_1	C_2
Qwen-1.5B	1	0	0	–	0	0	0.01	0
	2	0.08	0.68	8.0	0.68	9.5×10^{-4}	0.02	5.44
	3	0.25	1.5	6.1	1.3	1.7×10^{-1}	0.03	5.83
	4	0.47	3.4	7.2	2.5	8.4×10^{-1}	0.05	5.83
	5	0.83	6.9	8.3	4.5	2.4×10^{0}	0.07	5.83
Llama-1B	1	0	0	–	0	0	0.04	0
	2	0.05	3.3×10^{3}	6.2×10^{4}	2.3×10^{3}	9.7×10^{2}	0.03	3.03×10^{3}
	3	0.30	5.4×10^{4}	1.8×10^{5}	6.2×10^{3}	4.8×10^{4}	0.04	3.17×10^{3}
	4	0.85	5.5×10^{5}	6.4×10^{5}	6.4×10^{3}	5.4×10^{5}	0.05	3.18×10^{3}
	5	1.84	4.3×10^{6}	2.3×10^{6}	5.0×10^{4}	4.2×10^{6}	0.05	3.21×10^{3}
Gemma-2B	1	0	0	–	0	0	0.00	0
	2	0.09	1.1×10^{3}	3.6×10^{4}	7.3×10^{2}	4.3×10^{2}	0.01	1.05×10^{3}
	3	0.48	3.5×10^{4}	7.3×10^{4}	1.5×10^{3}	3.3×10^{4}	0.03	1.15×10^{3}
	4	1.21	7.6×10^{4}	6.2×10^{4}	2.0×10^{3}	7.4×10^{4}	0.03	1.24×10^{3}
	5	2.11	3.2×10^{5}	1.5×10^{5}	2.7×10^{3}	3.2×10^{5}	0.06	1.37×10^{3}

Table 3: Decomposition of equation (4) under quantization-aware training, for Qwen-1.5B at $T = 5$. Lowering the bit-width monotonically reduces the error amplification factor $\sum_{t=1}^{T-1} \rho_t^{(T)}$ and increases the per-step prediction errors, summarized by $\sum_{t=1}^{T-1} \varepsilon_t$; the bound and the test accuracy are determined by the balance of the two. The best bit-width per task is in bold.

Task	Bit-width	Test accuracy (%)	Loss gap	Bound	$\sum_t \rho_t^{(T)}$	$\sum_t \varepsilon_t$
Bellman-Ford	1	69.6	2.0	21.8	143.6	3.6
	2	77.3	1.5	12.1	157.2	1.9
	3	79.5	1.4	10.8	159.6	1.9
	4	**80.3**	0.6	7.3	166.7	1.8
	32	75.3	1.8	13.4	177.0	2.3
Dijkstra	**1**	**76.2**	0.9	2.1	9.3	2.9
	2	75.3	0.9	2.4	9.4	2.9
	3	74.2	1.1	3.3	13.9	2.7
	4	73.8	1.6	5.2	22.6	1.9
	32	73.1	2.3	14.2	34.4	1.8
Prim	**1**	**66.3**	1.1	1.1	11.9	3.0
	2	65.6	1.3	1.4	14.7	3.0
	3	64.7	1.7	1.5	20.1	2.8
	4	63.5	2.0	2.0	30.1	2.5
	32	62.3	2.1	2.7	69.8	2.3

In practice, we observe that QAT reduces the error amplification factor more significantly than it increases prediction errors. As a result, lower bit-width quantization typically achieves a reduction in the bound, leading to improved test accuracy. On Dijkstra and Prim, the prediction errors increase by only $1.6\times$ and $1.3\times$. The bound decreases monotonically as the bit-width is lowered, by $6.8\times$ and $2.5\times$ at 1 bit. The test accuracy improves by 3.1% and 4.0% over full precision. On Bellman-Ford, the reduction of $\sum_t \rho_t^{(T)}$ is small, while the prediction errors double from 4 bits to 1 bit. Therefore,

Table 4: The error amplification factor tracks the inference loss across base models. We fine-tune Llama-1B and Gemma-2B on Bellman-Ford ($T = 5$) and report both terms of equation (4) together with the estimated constants; columns are as in Table 2. Within each model, the ranking of methods by our bound matches their ranking by the loss gap, even though the absolute scale of the bound differs by orders of magnitude between the two models.

				Terms of equation (4)		Constants	
Model	Method	Loss gap	Bound	First-order	Second-order	C_1	C_2
Llama-1B	SFT-CoT	1.84	4.3×10^6	5.0×10^4	4.2×10^6	0.050	3.21×10^3
	Coconut	0.699	1.1×10^5	1.3×10^3	1.1×10^5	0.077	1.63×10^3
	Alg. 1	**0.491**	$\mathbf{1.3 \times 10^4}$	7.4×10^2	1.2×10^4	0.091	1.79×10^3
Gemma-2B	SFT-CoT	2.11	3.2×10^5	2.7×10^3	3.2×10^5	0.060	1.37×10^3
	Coconut	0.662	1.5×10^4	1.34×10^2	1.5×10^4	0.094	9.27×10^2
	Alg. 1	**0.091**	$\mathbf{4.9 \times 10^3}$	26.6	4.9×10^3	0.064	4.78×10^2

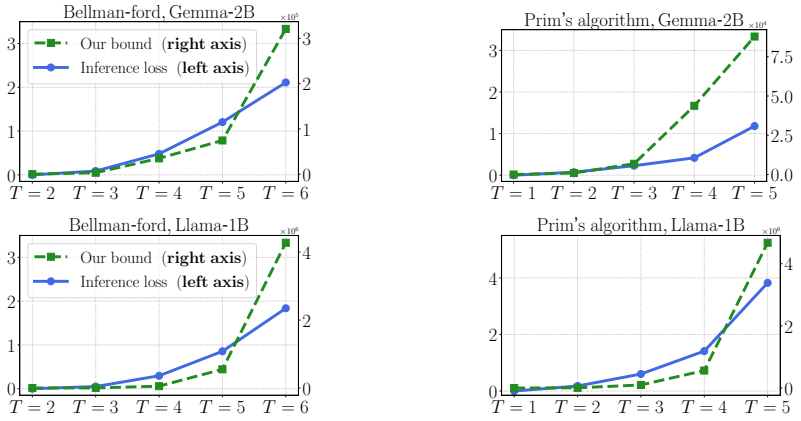
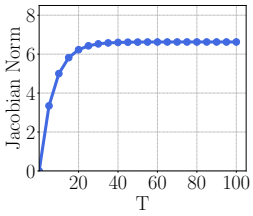


Figure 5: We illustrate the loss gap and our bound in equation (4), which can scale exponentially across generation steps, closely tracking the growth of the inference loss. Results are evaluated on Bellman-Ford and Prim's algorithm, using Gemma-2B and Llama-1B models. The two quantities are plotted on separate axes and differ in absolute scale; see Table 2 for the bound-to-gap ratios.

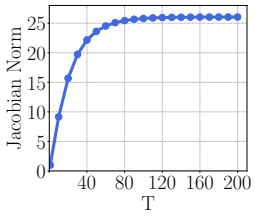
the bound and the test accuracy are both best at 4 bits. The bit-width at which the growth of the error levels begins to dominate depends on the task, and we select it on the validation set.

Evaluations of our bound across algorithms. To evaluate whether the relationship between our bound and the inference loss extends beyond Qwen-1.5B, we apply Algorithm 1 to Llama-1B and Gemma-2B on Bellman-Ford with $T = 5$, and compare it with standard chain-of-thought fine-tuning (SFT-CoT) and continuous chain-of-thought (Coconut) [10]. As shown in Table 4, our algorithm attains the lowest loss gap and the lowest bound on both models. Compared with Coconut, it reduces the loss gap by 58% on average and the bound by 78% on average. Compared with SFT-CoT, it lowers the bound by $330\times$ and $65\times$. This confirms that the reduction in test loss correlates with the reduction of our bound components.

Numerical evaluation of Jacobian norms. We further analyze the training dynamics by examining the Hessian trace of models fine-tuned with chain-of-thought on linear and quadratic functions. Specifically, we evaluate the norm of the Jacobian matrix product as a function of the reasoning length T and report results for both settings. As shown in Figure 6, as the reasoning length T increases, the norm of the Jacobian matrix product converges to a value. This behavior is consistent and aligned with our analysis of the gradient descent update rule for linear and quadratic functions.



(a) Jacobian matrix norm on linear functions



(b) Jacobian matrix norm in quadratic functions

Figure 6: We illustrate how transformers realize multi-step gradient descent behavior when trained on linear and quadratic functions. We verify that the Jacobian matrix norm is a converging series on linear functions and quadratic functions. The Jacobian norm of linear functions is $\sum_{i=1}^{T} \|(I - \frac{\eta}{n} XX^\top)^i\|_2^2$ and the Jacobian norm of quadratic functions is $\sum_{i=1}^{T} \|(I - \frac{2\eta}{n} XD(\theta_i)X^\top)^i\|_2^2$.

B.2 Sample Complexity Comparison

We compare models fine-tuned with a varied number of training examples between $1000, 2000$, and 5000. We also evaluate a model fine-tuned without intermediate steps. After training, we evaluate the noise stability using the fine-tuned model. We inject isotropic Gaussian noise with standard deviation $\sigma = 0.005$ into the model weights and measure the increase in the inference loss. We compute the average loss increase with 10 random seeds. We evaluate the noise stability of models fine-tuned with chain-of-thought on a range of algorithmic reasoning tasks, including DFS, Dijkstra, Prim, and DAG shortest path.

First, we find that increasing the number of training samples leads to improved noise stability across all tasks. This suggests a strong correlation between generalization performance and robustness to weight noise. For the algorithms of the same task, we noted a difference in noise stability. Specifically, models fine-tuned with the Dijkstra algorithm have more than $2\times$ lower Hessian traces than the Bellman-Ford algorithm. Additionally, models fine-tuned by Kruskal's algorithm yield $15\times$ lower Hessian trace values than Prim's.

Second, we study the effect of intermediate steps on the sample complexity. We randomly subsample the intermediate steps T to 20, 10, 5, 2, and 0 where $T = 0$ indicates training without intermediate steps. We find that a larger T substantially reduces sample complexity. Using intermediate steps consistently achieves better sample complexity compared to training without intermediate steps.

Hyperparameters. We report the hyperparameters used in our experiments in Table 5.

Table 5: Summary of the hyperparameters used for each reasoning task. The exact configurations are selected from the defined search spaces during supervised fine-tuning.

Task	Quantization	Optimizer	Learning rate	LoRA Rank	Sampling factor λ	Latent steps k
Dijkstra	1-bit	AdamW	2×10^{-5}	16	0.1	1
Bellman-Ford	4-bit	AdamW	2×10^{-5}	16	0.2	1
Prim	4-bit	AdamW	2×10^{-5}	16	0.1	1
BFS	1-bit	AdamW	2×10^{-5}	16	0.1	1
DFS	3-bit	AdamW	2×10^{-5}	16	0.1	1
Cyclic	1-bit	AdamW	2×10^{-5}	16	1.0	1
Symmetry	1-bit	AdamW	2×10^{-5}	16	1.0	1

B.3 Discussions

Our formulation targets supervised fine-tuning on ground-truth steps. Reinforcement learning methods like Group Relative Policy Optimization or Proximal Policy Optimization optimize over the model's own rollouts. Our analysis extends to reinforcement learning in two key ways. First, because test-time generation is independent of the training objective, our stability analysis applies equally to models trained with reinforcement learning. Second, our work can be applied to study whether

policy-gradient training implicitly regularizes input Jacobian norms along high-reward trajectories. We leave the formal extension of our bounds to the reinforcement learning setting for future work.

Lastly, we discuss several limitations of this work. First, our method regularizes these Jacobian norms during training. Future work could develop test-time regularization strategies that adapt to each input example during inference. Second, for black-box models, where gradients and Jacobians are not directly accessible, proxy stability measures based on noise perturbations may provide a practical alternative. Finally, extending the Jacobian-based error accumulation analysis to broader language-model tasks, including tool use, generation, and question answering, remains an interesting direction.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state our three contributions: (i) the Jacobian-based test-error bound, (ii) the convergence results on linear and quadratic in-context weight predciction, and (iii) the fine-tuning algorihtm.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of this work in Appendix B.3.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: All assumptions are stated in the theorem statements and the corresponding initialization assumptions, with complete proofs in Appendices.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 5 and Appendix B specify the datasets, splits, base models, optimizer, learning-rate range, LoRA rank, sampling factor, and bit-width grids, and Algorithm 1 describes the full training procedure.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide the link to our code repository in Section 1.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Training and test details, including hyperparameter grids and selected values, are reported in Section 5 and Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All accuracy results in Table 1 are reported as mean $\pm$ standard deviation over three random seeds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute resources, including GPU type and approximate runtime per experiment, are reported in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: Our work analyzes training dynamics on synthetic algorithmic and symbolic reasoning tasks, with no human subjects, personal data, or sensitive content, and conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This work focuses on analyzing the theoretical property of multi-step reasoning. The goal is to advance the field of Machine Learning. There are potential societal consequences of our work, none of which we feel must be specifically highlighted here.

    Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We release no pretrained models or datasets with high misuse risk; the fine-tuned models are specialized for graph-algorithmic and symbolic state-tracking tasks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite CLRS-Text, LEGO, and the base models (Qwen, Gemma, Llama) in Section 5 and use them in accordance with their respective licenses.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not introduce new datasets or pretrained models; training and evaluation code will be released with documentation at the camera-ready stage.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve research with human subjects, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs have been used to aid with paper writing and checking grammar mistakes.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.