EDIT: Early Diffusion Inference Termination for dLLMs Based on Dynamics of Training Gradients

He-Yen Hsieh   HEYENHSIEH@G.HARVARD.EDU
Harvard University
Hong Wang   HONG.WANG@INTEL.COM
Intel Corporation
H.T. Kung   KUNG@HARVARD.EDU
Harvard University

Abstract

Diffusion-based large language models (dLLMs) refine token generations through iterative denoising, but answers often stabilize before all steps complete. We propose EDIT (Early Diffusion Inference Termination), an inference-time criterion that adaptively stops denoising once sufficient reasoning stability relative to training-time reasoning is detected. EDIT monitors the alignment between token activations and a reasoning map derived from AdamW-aggregated LoRA updates captured during supervised fine-tuning (SFT). During training, optimization dynamics generate rich metadata about parameter importance that in prior methods is typically discarded upon model release. We preserve this information as a compact representation of learned reasoning pathways. During inference, alignment scores are converted to a distribution over the tokens already unmasked at the current denoising step, and convergence is detected when KL divergence between consecutive steps falls below a threshold on the matched unmasked (visible) tokens. Across reasoning benchmarks, EDIT reduces diffusion steps by 11.8% to 68.3% while preserving or improving accuracy in most settings, with approximately 0.02% storage overhead (about 1.5-2 MB for all QKV modules across 32 blocks in an 8 GB model). By utilizing training-gradient dynamics, our work opens a new research direction for reducing dLLM inference time and cost.

1. Introduction

Modern language model deployment follows a wasteful paradigm: during training, optimization dynamics on gradients generate rich information about which parameters are critical for specific capabilities, yet this metadata is routinely discarded once training completes. We challenge this practice by demonstrating that training-time optimization trajectories contain valuable signals that can guide intelligent inference-time decisions, specifically enabling adaptive early inference termination for diffusion language models.

Diffusion-based language models (dLLMs) [2, 8, 11] represent a promising alternative to autoregressive generation of tokens, employing iterative denoising processes that progressively refine outputs. However, their inference remains computationally expensive due to the fixed number of denoising steps, even when high-quality outputs emerge early in the process. Current approaches to this challenge operate without knowledge of which model parameters drove learning during training, essentially making uninformed decisions during inference termination optimization.

We introduce EDIT (Early Diffusion Inference Termination), a method that utilizes training optimization metadata to identify opportunities for early inference termination. Our key insight is that

the AdamW optimizer's [6] moment estimates during fine-tuning training encode which parameters consistently receive strong, directionally-aligned updates when learning reasoning tasks. The effectiveness of EDIT is validated from the perspective of gradient convergence during inference, as shown in Figure 1. These patterns, which we term *AdamW evolution*, represent a map of the model's learned reasoning pathways. Rather than discarding this information when training is complete, we store it as compact metadata (requiring minimal additional storage) and use it to guide inference termination decisions.

Our Approach: Utilizing Training Metadata for Guiding Inference Termination.　During supervised fine-tuning (SFT) on reasoning tasks, certain LoRA parameters receive consistent gradient signals across training steps, indicating their importance for encoding reasoning patterns. The AdamW optimizer naturally tracks this through its moment estimates, with the first moment capturing gradient direction and the second moment reflecting gradient stability. Parameters with large, stable updates become critical components of the learned reasoning pathways. Traditional inference deployment discards this valuable information. EDIT preserves it by saving aggregated AdamW updates across training steps, creating a fingerprint of which parameters matter most for reasoning. At inference time, we compare current token activations against these preserved patterns using cosine similarity, assessing whether the model's current state aligns with its learned reasoning pathways. When this alignment stabilizes—indicating the model has reached its learned reasoning configuration—we can confidently terminate the diffusion inference process early. Instead of relying on inference-time heuristics such as confidence or output stability, EDIT leverages training-time knowledge to terminate once key reasoning components are engaged.

Contributions.　(1) We establish a new paradigm for early inference termination that leverages training metadata which is usually discarded in prior methods. (This approach opens future research directions in not only early inference termination, but also informing dynamic compute allocation, quality prediction, and other inference-time optimizations.) (2) We provide a practical instantiation through EDIT, demonstrating that AdamW evolution patterns can reliably indicate when diffusion models have completed their core reasoning. Our method requires no architectural changes, adds minimal storage overhead, and integrates seamlessly with existing diffusion language models. (3) We validate our approach across multiple reasoning benchmarks, showing inference speedups of 11.8% to 68.3% while maintaining or improving accuracy in most settings. These gains come purely from utilizing training information that already existed but was previously thrown away, highlighting the inefficiency of current practices.

2. Preserving and Utilizing Training Metadata

We detail how EDIT captures optimization dynamics during training and leverages them for intelligent early termination during inference. Our approach consists of two phases: metadata extraction during fine-tuning (Section 2.1) and metadata-guided termination during inference (Section 2.2).

Utilizing Training Metadata for Guiding Inference Termination.　During supervised fine-tuning (SFT), some parameters receive strong, stable updates that encode core reasoning patterns, while others show weak or oscillating updates and contribute less. We track this distinction through the AdamW update history—what we call the *AdamW evolution*—which forms a map of reasoning-relevant parameters. At inference, activations are compared against this map; when alignment with the learned reasoning pathways is reached, early termination is enabled with confidence.

2.1. Capturing AdamW Evolution During Training

We consider a pre-trained base model with LoRA (Low-Rank Adaptation) [4] modules inserted into the Query (Q), Key (K), and Value (V) projections of each Transformer block. During SFT on reasoning tasks, only these LoRA parameters are updated. Each LoRA module consists of matrices (A, B) where $A \in \mathbb{R}^{r \times d_{\text{in}}}$ and $B \in \mathbb{R}^{d_{\text{out}} \times r}$ implement a low-rank update to the corresponding projection.

Notation note: We use $\mathcal{L}_k$ for the loss at training step k, and L for block length in the diffusion process.

For the LoRA-B matrix $B \in \mathbb{R}^{d_{out} \times r}$ (where d_{out} is the output dimension and r is the rank), the AdamW optimizer maintains first and second moment estimates at each training step k:

$$M_{k,B} = \beta_1 M_{k-1,B} + (1 - \beta_1)G_{k,B}, \quad V_{k,B} = \beta_2 V_{k-1,B} + (1 - \beta_2)G_{k,B}^{\odot 2}, \tag{1}$$

where $G_{k,B} = \nabla_B \mathcal{L}_k$ is the gradient tensor, $\beta_1, \beta_2 \in [0, 1)$ are decay rates, and $\odot$ denotes element-wise operations. The element-wise update magnitude at step k is:

$$U_{k,B} = \frac{M_{k,B}}{\sqrt{V_{k,B}} + \epsilon}, \tag{2}$$

where ϵ ensures numerical stability and division is element-wise. To create a stable representation of the parameter importance patterns, we define the *AdamW evolution tensor* as the average over all $\mathcal{K}$ fine-tuning steps:

$$\bar{U}_B = \frac{1}{\mathcal{K}} \sum_{k=1}^{\mathcal{K}} U_{k,B} \in \mathbb{R}^{d_{out} \times r}. \tag{3}$$

This tensor captures which elements of the LoRA-B matrix consistently received strong, directional updates during training. To enable comparison with token activations $\mathbf{f}_s \in \mathbb{R}^{d_{out}}$, we reduce $\bar{U}_B$ to a feature-aligned vector $\mathbf{u} \in \mathbb{R}^{d_{out}}$ using row-wise energy:

$$\mathbf{u}[p] = \left\| \bar{U}_B[p, :] \right\|_2 = \sqrt{\sum_{j=1}^{r} \bar{U}_B[p, j]^2}, \quad p = 1, \ldots, d_{out}. \tag{4}$$

This reduction preserves the update magnitude each output dimension receives across low-rank components, forming a parameter-importance signature aligned with the feature space.

2.2. Metadata-Guided Early Termination During Inference

At inference time, we leverage the preserved AdamW evolution to determine when the diffusion process can safely terminate. Our approach operates on block-level diffusion [2, 8, 11], where the sequence is divided into blocks of length L, and tokens within each block are progressively unmasked across denoising steps.

2.2.1. ASSESSING REASONING ALIGNMENT

Let $\mathcal{S}_t$ denote the set of visible (unmasked) tokens at denoising step t. For each visible token $s \in \mathcal{S}_t$, we extract its post-LoRA activation $\mathbf{f}_s^{(t)} \in \mathbb{R}^{d_{out}}$ from the chosen module (specifically, the LoRA-B output of the Query projection in the last Transformer block, based on our empirical findings

in Figure 3). We compute the cosine similarity between each token's activation and the AdamW evolution vector:

$$\text{Sim}_s^{(t)} = \frac{\langle \mathbf{f}_s^{(t)}, \mathbf{u} \rangle}{\|\mathbf{f}_s^{(t)}\|_2 \|\mathbf{u}\|_2}, \tag{5}$$

where $\mathbf{u}$ is the feature-aligned vector from Equation 4. To convert these alignment scores into a probability distribution, we apply softmax with a fixed temperature τ_{blk} within each block:

$$P^{(t)}(s) = \frac{\exp(\text{Sim}_s^{(t)}/\tau_{\text{blk}})}{\sum_{i \in \mathcal{S}_t} \exp(\text{Sim}_i^{(t)}/\tau_{\text{blk}})}, \quad s \in \mathcal{S}_t. \tag{6}$$

Keeping τ_{blk} fixed within a block ensures that distribution changes reflect genuine alignment shifts rather than temperature-induced artifacts.

2.2.2. DETECTING REASONING STABILITY VIA MATCHED SUPPORT

As tokens are progressively unmasked, the support of our distribution grows. To properly compare distributions across steps, we must account for this changing support. Let $\mathcal{I}_t = \mathcal{S}_{t-1} \cap \mathcal{S}_t$ be the intersection of visible tokens between consecutive steps. We renormalize both distributions to this common support:

$$\tilde{P}^{(t)}(s) = \frac{P^{(t)}(s)}{\sum_{i \in \mathcal{I}_t} P^{(t)}(i)}, \quad \tilde{P}^{(t-1)}(s) = \frac{P^{(t-1)}(s)}{\sum_{i \in \mathcal{I}_t} P^{(t-1)}(i)}, \quad s \in \mathcal{I}_t. \tag{7}$$

The step-wise divergence is then computed as:

$$D_t = D_{\text{KL}}(\tilde{P}^{(t)} \| \tilde{P}^{(t-1)}) = \sum_{s \in \mathcal{I}_t} \tilde{P}^{(t)}(s) \log \frac{\tilde{P}^{(t)}(s)}{\tilde{P}^{(t-1)}(s)}. \tag{8}$$

2.2.3. EARLY TERMINATION WITH CONSECUTIVE STABILITY

To ensure robust termination decisions, we require consecutive steps of stability rather than sporadic stable steps. We maintain a run-length counter c updated as $c \leftarrow c + 1$ if $D_t < \delta$, and reset to 0 otherwise.

Beyond this heuristic, our analysis shows that small matched-support KL divergence over Ω consecutive steps, with Ω determined by the inference progress at the point of early termination, bounds the multi-step total variation distance and yields a no-flip condition for predicted tokens together with stability bounds for Lipschitz observables. PAC-style bounds (Corollary 6) provide a basis for principled hyperparameter selection and indicate that early termination matches the full denoising during inference with high probability. Formal statements and calibration rules are given in Appendix C, along with two extensions: a token-wise freezing scheme that halts denoising per token with provable instance-wise safety (Appendix D), and a subspace generalization that replaces the single reasoning vector with a low-dimensional reasoning subspace while preserving all theoretical guarantees (Appendix E).

The diffusion process for the current block terminates when $c \geq \Omega$, indicating that the model's reasoning alignment has remained stable for Ω consecutive steps. Algorithm 1 and Figure 6 (Appendix F) present the full EDIT procedure and its workflow. The training phase extracts metadata with zero additional computational cost (these values are already computed by the optimizer), while the inference phase uses this metadata to make principled termination decisions.

Table 1: Accuracy on reasoning benchmarks. EDIT uses training-time metadata for adaptive early termination. Results are mean over 3 seeds, where **bold** denotes the best, <u>underline</u> denotes the second-best. 0-shot means no in-context examples during evaluation (post-SFT). Experiments are run with sequence lengths 128/256/512.

Dataset (Seq Len)	Countdown (0-shot)			Sudoku (0-shot)			MATH500 (0-shot)			GSM8K (0-shot)			GPQA (0-shot)		
Method	128	256	512	128	256	512	128	256	512	128	256	512	128	256	512
LLaDA (No SFT)	<u>19.9</u>	19.5	16.4	10.4	6.4	<u>6.3</u>	**27.6**	<u>32.4</u>	36.0	<u>68.1</u>	75.8	<u>79.5</u>	21.9	**27.9**	25.7
LLaDA (SFT)	19.5	<u>20.7</u>	<u>20.3</u>	<u>11.4</u>	<u>8.2</u>	5.0	26.2	30.4	<u>35.4</u>	**69.8**	<u>77.0</u>	**81.2**	<u>23.0</u>	20.5	**26.3**
EDIT (Ours)	**28.9**	**31.6**	**27.7**	**16.1**	**11.3**	**7.6**	<u>27.4</u>	**32.8**	**36.6**	67.3	**77.6**	76.2	**25.5**	<u>27.7</u>	<u>26.1</u>

Table 2: Diffusion steps with EDIT vs. baseline full diffusion. Values are averaged across blocks, and baselines are fixed at 64/128/256 steps for sequence lengths 128/256/512. Percentages show reduction from baseline steps, with training metadata enabling confident early termination without quality loss.

Dataset (Seq Len)	Countdown			Sudoku			MATH500			GSM8K			GPQA		
Method	128	256	512	128	256	512	128	256	512	128	256	512	128	256	512
Baseline (Full Steps)	64	128	256	64	128	256	64	128	256	64	128	256	64	128	256
EDIT (Ours)	40.4	40.6	133.3	38.3	74.9	163.3	38.1	81.9	197.2	42.8	103.5	225.8	40.3	81.3	194.1
Reduction (%)	36.9	68.3	47.9	40.2	41.5	36.2	40.5	36.0	23.0	33.1	19.2	11.8	37.0	36.5	24.2

3. Experimental Validation

Experimental Setup. We evaluate on five reasoning tasks: Countdown [9], Sudoku [1], MATH500 [5], GSM8K [3], and GPQA [10]. We use LLaDA-8B [8] as our baseline model, fine-tuned on the s1 dataset [7] with LoRA applied to QKV projections. All experiments use Intel XPU hardware to ensure reproducibility. During SFT, we preserve AdamW evolution metadata (Section 2.1). Persisting reduced vectors $\mathbf{u}$ requires $\sim$16 KB per module, or $\sim$1.5 MB for all QKV projections across 32 blocks ($<$0.02% of an 8 GB model). At inference, EDIT uses task-specific thresholds (Appendix B.1) selected on held-out validation sets (20% of training data), ensuring no test set leakage.

Results: Efficiency Gains with Preserved Accuracy. Table 1 shows that EDIT improves accuracy on Countdown (up to 31.6%) and Sudoku (up to 16.1%), while remaining competitive on other tasks. Early termination avoids late-step degradation because once predictions stabilize, further denoising can overwrite correct intermediate states. This effect is most pronounced in crisp tasks such as Countdown and Sudoku. GSM8K at sequence length 512 drops from 81.2% to 76.2% because long reasoning chains often stabilize before reasoning is complete. Although GSM8K and GPQA show task-specific variation, the overall average remains positive, validating that metadata-guided termination improves rather than compromises quality. Table 2 shows that EDIT reduces average denoising steps per block by 11.8%–68.3% compared to fixed 64/128/256-step baselines for sequence lengths 128/256/512. Gains are most pronounced on short sequences where full diffusion is wasteful. With PAC-style calibration (Appendix C.6), 72.3% of early terminations satisfy Corollary 6, indicating that EDIT remains within its theoretical safety bounds while delivering speedups.

Gradient-Based Justification for Early Termination. To determine when denoising steps can be truncated safely, we adopt a gradient view of inference by comparing pseudo-gradients at inference with SFT gradients on LoRA-B layers. At each inference step t, the model outputs logits $z_t(s)$ for tokens s in a block of length L. Since no ground-truth labels are available, the signal comes from

changes in predictive distributions across steps. Let $p_\theta(z_t(s))$ and $p_\theta(z_{t+1}(s))$ denote predictions at steps t and $t+1$; their KL divergence quantifies prediction change (see Appendix G for details). We define the pseudo-gradient as $\tilde{G}_{t,B} = \nabla_B \sum_{s \in S_{t+1}} \mathrm{KL}(p_\theta(z_t(s)) \,\|\, p_\theta(z_{t+1}(s)))$, where S_{t+1} is the visible token set at step $t+1$. Backpropagating this divergence through LoRA-B yields $\tilde{G}_{t,B}$, whose root mean square (RMS) magnitude provides a scalar summary per step. Tracing these values across denoising steps produces a trajectory of inference dynamics.

During SFT, we compute RMS magnitudes of gradients $G_{k,B}$ across training steps and summarize them by a mean μ_{SFT} and a variance band, defining the stable regime. Convergence is declared when inference pseudo-gradients (1) approach μ_{SFT} and (2) remain within this band, beyond this time further denoising adds cost without benefit. On GPQA(seq. 128, 2nd block), Figure 1 shows pseudo-gradients converging at step 19 (yellow ▼). Thereafter they fluctuate around the SFT mean, indicating entry into the training-consistent regime. Terminating at $\sim$20 steps per block thus preserves fidelity while cutting cost, consistent with Table 2 (40.3 steps for two blocks) and Table 1, which confirms accuracy remains competitive.
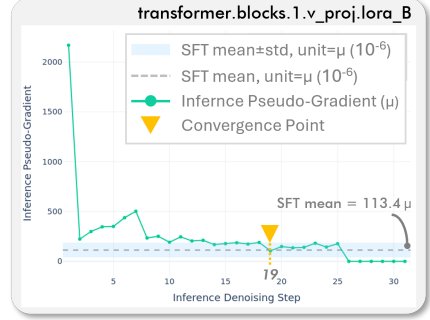

Figure 1: Gradient-based analysis of training–inference alignment on GPQA (seq. 128, 2nd block). Root mean square (RMS) pseudo-gradients $\tilde{G}_{t,B}$ across steps are compared with the SFT gradient mean (dashed) and variance band (shaded). The convergence point (yellow ▼) occurs at step 19, after which pseudo-gradients stabilize near the SFT mean, indicating that $\sim$20 steps per block preserve fidelity while reducing computation (Table 2, 40.3 steps for two blocks).

Storage and Computational Overhead. AdamW evolution metadata stores only the reduced vector $\mathbf{u} \in \mathbb{R}^{d_{out}}$ per LoRA module, not the full tensor $\bar{U}_B$. With $d_{out} = 4096$ (float32), this is $\sim$16 KB per module, or $32 \times 3 \times 16$ KB $\approx$1.5 MB across all QKV projections in 32 blocks—just 0.02% of an 8 GB model. At inference, EDIT adds cosine similarity (Equation 5) and KL divergence (Equation 8) with cost $O(|\mathcal{S}_t| \cdot d_{out})$ and $O(|\mathcal{I}_t|)$, negligible compared to $O(L^2 \cdot d_{out})$ self-attention (Appendix B.4). Overall overhead is negligible, yielding net speedups.

4. Conclusion and Future Directions

We introduced EDIT, which preserves training metadata typically discarded and uses it to guide early inference termination in diffusion language models. By capturing optimization dynamics during fine-tuning, EDIT detects when reasoning is complete, reducing inference cost without architectural changes. Across five reasoning benchmarks, it achieves 11.8–68.3% fewer diffusion steps while maintaining or improving accuracy, with only 0.02% storage overhead. EDIT has limitations: it requires training dynamics, often unavailable in released models, suggesting providers include optimization metadata; it depends on task-specific thresholds (δ, Ω), motivating adaptive or learned criteria. We evaluate only LoRA; full-parameter extensions remain future work. Beyond early termination, training metadata could enable dynamic layer-wise compute, token-level processing, early quality prediction, and identifying prompts that merit additional inference budget. This work also highlights a systemic inefficiency: training information is often discarded despite its value at inference. Preserving and exploiting training metadata enables more holistic and efficient pipelines.

References

[1] Arel. Arel's sudoku generator. `https://www.ocf.berkeley.edu/~arel/sudoku/main.html`, 2025. Accessed: 2025-04-08.

[2] Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *The Thirteenth International Conference on Learning Representations, (ICLR)*, 2025.

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

[4] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations (ICLR)*, 2022.

[5] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[6] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations (ICLR)*, 2019.

[7] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel J. Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

[8] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

[9] Jiayi Pan, Junjie Zhang, Xingyao Wang, Lifan Yuan, Hao Peng, and Alane Suhr. Tinyzero. https://github.com/Jiayi-Pan/TinyZero, 2025. Accessed: 2025-01-24.

[10] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

[11] Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. d1: Scaling reasoning in diffusion large language models via reinforcement learning. *arXiv preprint arXiv:2504.12216*, 2025.

Appendix A. Mathematical Details

A.1. Complete Derivation of AdamW Evolution

We provide the complete mathematical framework for extracting and utilizing AdamW evolution patterns. The key insight is that optimization dynamics during training create signatures of parameter importance that can guide inference.

For a LoRA-B matrix $B \in \mathbb{R}^{d \times r}$, the AdamW optimizer maintains exponentially weighted moving averages of gradients and squared gradients. Starting from the recursive definitions:

$$M_{k,B}[i,j] = \beta_1 M_{k-1,B}[i,j] + (1 - \beta_1)G_{k,B}[i,j] \tag{9}$$

$$V_{k,B}[i,j] = \beta_2 V_{k-1,B}[i,j] + (1 - \beta_2)G_{k,B}[i,j]^2 \tag{10}$$

By unrolling these recursions and assuming zero initialization, we obtain:

$$M_{k,B}[i,j] = (1 - \beta_1) \sum_{\ell=1}^{k} \beta_1^{k-\ell} G_{\ell,B}[i,j] \tag{11}$$

$$V_{k,B}[i,j] = (1 - \beta_2) \sum_{\ell=1}^{k} \beta_2^{k-\ell} G_{\ell,B}[i,j]^2 \tag{12}$$

The element-wise update magnitude captures both direction (through M) and reliability (through V):

$$U_{k,B}[i,j] = \frac{M_{k,B}[i,j]}{\sqrt{V_{k,B}[i,j]} + \epsilon} \tag{13}$$

Averaging across all training steps yields the AdamW evolution tensor:

$$\bar{U}_B[i,j] = \frac{1}{\mathcal{K}} \sum_{k=1}^{\mathcal{K}} U_{k,B}[i,j] \tag{14}$$

The reduction to feature space via row-wise energy (Equation 4) preserves the total update magnitude each output dimension received, creating an interpretable signature of parameter importance.

A.2. Theoretical Justification for Stability Detection

The KL divergence between consecutive alignment distributions provides a principled measure of reasoning stability. Under mild assumptions about the smoothness of the denoising process, we can show that stable KL divergence indicates convergence to a fixed point in the alignment space.

Consider the alignment distribution as a function of the denoising step: $P^{(t)} = f_t(\mathbf{x}, \mathbf{u})$ where $\mathbf{x}$ represents the current token states and $\mathbf{u}$ is the fixed AdamW evolution vector. If the denoising process is contractive in the alignment space (which can occur under conditions such as Lipschitz continuity of the denoiser combined with fixed-temperature softmax normalization), then:

$$D_{\text{KL}}(P^{(t+1)} \parallel P^{(t)}) \leq \gamma \cdot D_{\text{KL}}(P^{(t)} \parallel P^{(t-1)}) \tag{15}$$

for some $\gamma < 1$. This ensures that requiring $D_t < \delta$ for Ω consecutive steps provides strong evidence of convergence.

8

Table 3: EDIT hyperparameter configuration for each benchmark and sequence length. Parameters were selected on validation sets to optimize the accuracy-efficiency trade-off.

Dataset (Len)	Countdown			Sudoku			MATH500			GSM8K			GPQA		
Parameter	128	256	512	128	256	512	128	256	512	128	256	512	128	256	512
All Blocks															
Block Temperature (τ_{blk})	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
First Block															
Threshold (δ)	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
Stability Span (Ω)	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6
Subsequent Blocks															
Threshold (δ)	0.05	0.55	0.45	0.1	0.45	0.05	0.1	0.05	0.025	0.05	0.025	0.025	0.05	0.05	0.05
Stability Span (Ω)	6	12	12	6	12	6	6	6	6	6	6	6	6	6	8

Appendix B. Extended Experimental Details

B.1. Hyperparameter Selection Protocol

To ensure reproducibility and avoid overfitting, we employ a systematic hyperparameter selection protocol. For each task, we use 20% of the training data as a validation set to tune the stability threshold $\delta \in \{0.025, 0.05, 0.1, 0.25, 0.45, 0.55\}$ and stability span $\Omega \in \{6, 8, 10, 12\}$. We select the configuration that maximizes the accuracy-efficiency trade-off, defined as accuracy divided by average diffusion steps. The block temperature τ_{blk} is fixed at 1.0 for all experiments to ensure fair comparison. Table 3 provides the complete configuration for each benchmark.

B.2. Complete Hyperparameter Configuration

Table 3 provides the complete hyperparameter settings used in our experiments. These were selected using the validation protocol described in Section B.1.

B.3. Understanding When Training Metadata Helps

Figure 2 reveals that EDIT's effectiveness varies across problem types within GPQA. The method shows substantial improvements in domains requiring systematic reasoning (Molecular Biology, Astrophysics) while providing modest gains in others. This variation supports our core thesis: the training metadata captures task-specific patterns, and its utility depends on how well-defined these patterns are for each domain. Tasks with clear, consistent reasoning pathways benefit most from our approach.

Figure 3 visualizes how different reasoning tasks activate distinct parameter subsets, as revealed by the AdamW evolution patterns. We focus on the LoRA-B matrix of the Query projection in the last Transformer block (block 31), which empirically shows the strongest task-specific patterns. This visualization confirms that training dynamics create meaningful signatures that can guide inference decisions.
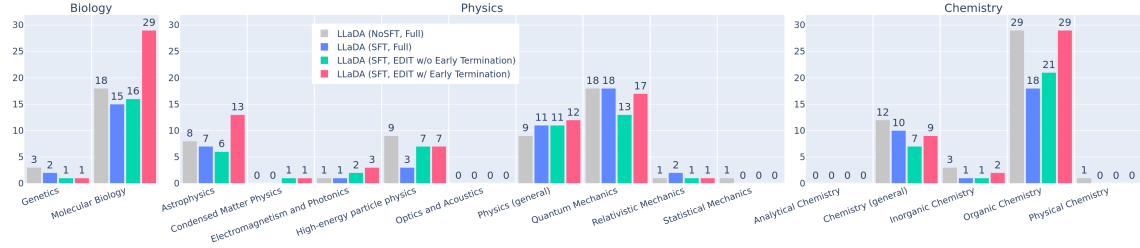
Figure 2: Performance breakdown across GPQA subdomains comparing EDIT (red) with baseline SFT (green). EDIT shows particularly strong improvements in Molecular Biology and Astrophysics, where reasoning patterns are more structured. The domain-specific variation validates that training metadata captures specialized reasoning pathways.
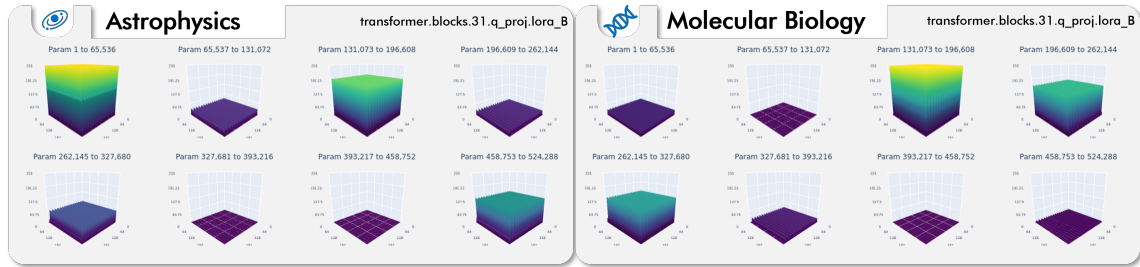

Figure 3: Task-specific parameter activation patterns revealed by AdamW evolution. Different GPQA subdomains (Astrophysics vs. Molecular Biology) engage distinct parameter subsets in the LoRA-B matrix of the Query projection (transformer.block.31). The 3D visualization shows how parameter importance varies across tasks, demonstrating that training metadata captures specialized reasoning pathways.

B.4. Storage and Computational Overhead

The AdamW evolution metadata requires storing only the reduced vector $\mathbf{u} \in \mathbb{R}^d$ per chosen LoRA module, not the full tensor $\bar{U}_B$. For our configuration with $d = 4096$, this amounts to approximately 16 KB per module (assuming float32 precision). Even if we store metadata for all QKV projections across all 32 Transformer blocks, the total overhead is $32 \times 3 \times 16$ KB ≈ 1.5 MB—merely 0.02% of the 8 GB model size. At inference time, EDIT adds cosine similarity computations (Equation 5) and KL divergence calculations (Equation 8) at each denoising step. These operations have complexity $O(|\mathcal{S}_t| \cdot d)$ and $O(|\mathcal{I}_t|)$ respectively, which is minimal compared to the $O(L^2 \cdot d)$ cost of self-attention in each Transformer block. The net result is substantial speedup despite these additional computations.

B.5. Additional Visualizations

Figure 4 provides additional evidence for the importance of preserving training-time metadata for LoRA-B. The visualization shows how specific parameters receive consistently strong updates during fine-tuning, creating clear signatures that can guide inference. In contrast, Figure 5 demonstrates that LoRA-A exhibits minimal parameter changes during the same training process.
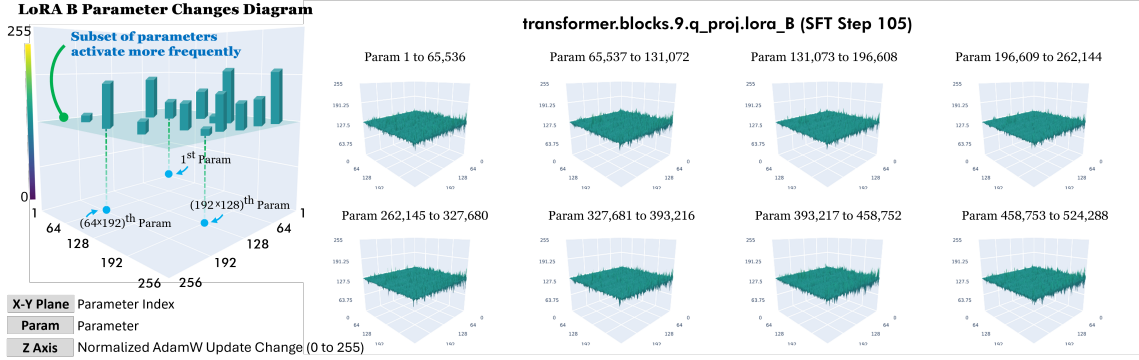
Figure 4: Visualization of LoRA-B parameter updates at training step 105. A 4096×128 LoRA projection produces $524,288$ parameters, reshaped into a $256 \times 256 \times 8$ grid with the Z-axis showing normalized AdamW update magnitudes (scaled 0–255). Pronounced peaks indicate parameters critical for reasoning tasks, demonstrating that optimization dynamics create clear importance signatures.
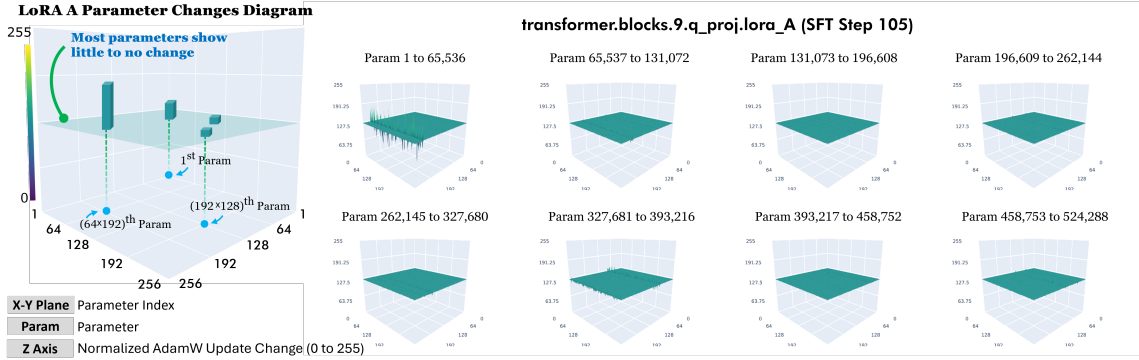

Figure 5: Visualization of LoRA-A parameter updates at training step 105. A 128×4096 LoRA projection produces $524,288$ parameters, reshaped into a $256 \times 256 \times 8$ grid with the Z-axis showing normalized AdamW update magnitudes (scaled 0–255). Pronounced peaks indicate parameters critical for reasoning tasks, demonstrating that optimization dynamics create clear importance signatures.

B.6. Module Selection Analysis

To justify our choice of using the LoRA-B matrix from the Query projection, we conducted an ablation study comparing different modules and reduction strategies. Table 4 shows that the Query projection's LoRA-B matrix with row-wise energy reduction provides the most reliable stability signal.

11

Table 4: Ablation study on module selection and reduction strategy. Results show average KL divergence stability (lower is better) across 100 validation examples from GSM8K.

Module	Row-wise Energy	Row-wise Mean
Query (LoRA-A)	0.142	0.168
Query (LoRA-B)	**0.089**	0.103
Key (LoRA-B)	0.124	0.139
Value (LoRA-B)	0.117	0.128

Appendix C. Theoretical Foundations of Early Termination in EDIT

C.1. Setup and Notation

At denoising step t, let $\mathcal{S}_t$ denote the visible tokens, and $\mathcal{I}_t = \mathcal{S}_{t-1} \cap \mathcal{S}_t$ the matched support between steps $t-1$ and t. Let $\tilde{P}^{(t)}$ be the probability distribution on $\mathcal{I}_t$ obtained by restricting $P^{(t)}$ to $\mathcal{I}_t$ and renormalizing. Define the per-step divergence

$$D_t = D_{\text{KL}}(\tilde{P}^{(t)} \parallel \tilde{P}^{(t-1)}). \tag{16}$$

EDIT declares stability when $D_{t-\Omega+1}, \ldots, D_t \leq \delta$ for some integers $\Omega \geq 1$ and threshold $\delta > 0$ (with fixed in-block temperature when forming $P^{(\cdot)}$).

We use the total variation distance $\text{TV}(p, q) = \frac{1}{2} \sum_i |p_i - q_i|$ and Pinsker's inequality $\text{TV}(p, q) \leq \sqrt{\frac{1}{2} D_{\text{KL}}(p \parallel q)}$.

C.2. Multi-Step Control from Run-Length KL

Lemma 1 (Run-length KL implies multi-step TV bound)   *If* $D_{t-\Omega+1}, \ldots, D_t \leq \delta$, *then*

$$TV(\tilde{P}^{(t)}, \tilde{P}^{(t-\Omega)}) \leq \sum_{r=t-\Omega+1}^{t} TV(\tilde{P}^{(r)}, \tilde{P}^{(r-1)}) \leq \Omega\sqrt{\frac{\delta}{2}}. \tag{17}$$

Proof The result follows from the triangle inequality for total variation distance, then applying Pinsker's inequality to each summand. ∎

C.3. Local Argmax Invariance at the Stopping Time

Let $i^*(t) = \arg\max_{s \in \mathcal{I}_t} \tilde{P}^{(t)}(s)$ and $m_t = \tilde{P}^{(t)}_{(1)} - \tilde{P}^{(t)}_{(2)}$ be the top-2 margin on $\mathcal{I}_t$.

Theorem 2 (Local argmax invariance certificate)   *If* $D_{t-\Omega+1}, \ldots, D_t \leq \delta$ *and*

$$\Omega\sqrt{\frac{\delta}{2}} < \frac{1}{2}m_t, \tag{18}$$

then $i^*(t') = i^*(t)$ *for all* $t' \in \{t-\Omega, \ldots, t\}$.

Proof If i^* changed between $p = \tilde{P}^{(t)}$ and $q = \tilde{P}^{(t')}$, then $\text{TV}(p, q) \geq \frac{1}{2}(p_{(1)} - p_{(2)}) = \frac{1}{2}m_t$ (considering the mass that must move between the top two coordinates when the argmax changes). This contradicts Lemma 1. ∎

Interpretation: When EDIT stops and the inequality holds, the predicted token on the matched support has been unchanged for the past Ω steps—providing a verifiable certificate attached to the stopping decision.

C.4. Future-Step Robustness via Contraction

After the last unmasking in a block, let K_r denote the Markov operator that maps $\tilde{P}^{(r-1)}$ to $\tilde{P}^{(r)}$ on the fixed support. Define the Dobrushin coefficient $\alpha(K_r) = \sup_{p \neq q} \frac{\text{TV}(pK_r, qK_r)}{\text{TV}(p,q)} \in [0, 1]$.

Assumption 3 (Local contraction post-unmasking) *There exists $\alpha < 1$ such that $\alpha(K_r) \leq \alpha$ for all $r \geq t + 1$ within the block. This property is standard for convergent Markov chains and can be verified empirically on validation data.*

Theorem 4 (Tail movement bound and global argmax preservation) *Under Assumption 3, if $D_{t-\Omega+1}, \ldots, D_t \leq \delta$, then for any $s \geq 1$,*

$$TV(\tilde{P}^{(t+s)}, \tilde{P}^{(t)}) \leq \frac{\alpha^s}{1-\alpha}TV(\tilde{P}^{(t)}, \tilde{P}^{(t-1)}) \leq \frac{\alpha^s}{1-\alpha}\sqrt{\frac{\delta}{2}}, \tag{19}$$

and thus $\sup_{s \geq 1} TV(\tilde{P}^{(t+s)}, \tilde{P}^{(t)}) \leq \frac{1}{1-\alpha}\sqrt{\frac{\delta}{2}}$.
If additionally

$$\Omega\sqrt{\frac{\delta}{2}} + \frac{1}{1-\alpha}\sqrt{\frac{\delta}{2}} < \frac{1}{2}m_t, \tag{20}$$

then $i^*(t + s) = i^*(t)$ for all $s \geq 0$ (argmax is preserved forever on the fixed support).

Proof One-step TV contracts by at most α; summing the geometric tail yields the bound. The argmax preservation follows by the same margin argument as in Theorem 2. ∎

C.5. Stability of Lipschitz Observables

Theorem 5 (Stability of Lipschitz functionals) *Let $F : \Delta \to \mathbb{R}$ satisfy $|F(p) - F(q)| \leq L \cdot TV(p, q)$ for all p, q. If $D_{t-\Omega+1}, \ldots, D_t \leq \delta$, then*

$$|F(\tilde{P}^{(t)}) - F(\tilde{P}^{(t-\Omega)})| \leq L \cdot \Omega\sqrt{\frac{\delta}{2}}. \tag{21}$$

Under Assumption 3,

$$\sup_{s \geq 1} |F(\tilde{P}^{(t+s)}) - F(\tilde{P}^{(t)})| \leq \frac{L}{1-\alpha}\sqrt{\frac{\delta}{2}}. \tag{22}$$

Proof Direct application of Lemma 1 and Theorem 4 with the Lipschitz property. ∎

C.6. Practical Calibration of (δ, Ω)

Let M denote the top-2 margin at EDIT's stopping time on a validation set, and $q_{1-\beta}$ its $(1-\beta)$-quantile. Estimate a post-unmasking contraction bound by

$$\hat{\alpha} = \max_{\text{val instances, late } r} \frac{\text{TV}(\tilde{P}^{(r+1)}, \tilde{P}^{(r)})}{\text{TV}(\tilde{P}^{(r)}, \tilde{P}^{(r-1)})}. \tag{23}$$

Corollary 6 (PAC-style guarantee for the final answer) *Choose (δ, Ω) to satisfy*

$$\Omega\sqrt{\frac{\delta}{2}} + \frac{1}{1-\hat{\alpha}}\sqrt{\frac{\delta}{2}} \leq \frac{1}{2}q_{1-\beta}. \tag{24}$$

Then, with probability at least $1 - \beta$ over test instances, the top-1 token at EDIT's stopping time equals the top-1 token obtained by continuing denoising indefinitely (on the fixed support).

Proof Apply Theorem 4 and the definition of $q_{1-\beta}$. ∎

Reporting recommendation: Alongside accuracy and step reductions, report the fraction of test instances that satisfy Corollary 6 ("percentage of certified stops"). This quantifies how often EDIT halts with a provable correctness certificate.

Appendix D. Token-Wise EDIT: Per-Token Freezing with Certificates

D.1. Local Stability Statistics and Rule

Let $U \in \mathbb{R}^{d \times k}$ denote a fixed reasoning subspace (construction examples provided below). For each visible token s at step t, define its subspace coordinates $g_s^{(t)} = U^\top f_s^{(t)} \in \mathbb{R}^k$ and the local distribution

$$Q_s^{(t)}(j) = \frac{\exp(|g_{s,j}^{(t)}|/\tau_{\text{sub}})}{\sum_{\ell=1}^k \exp(|g_{s,\ell}^{(t)}|/\tau_{\text{sub}})}, \quad j = 1, \ldots, k, \tag{25}$$

with fixed $\tau_{\text{sub}} > 0$.

Define the per-token KL $D_{s,t} = D_{\text{KL}}(Q_s^{(t)} \parallel Q_s^{(t-1)})$ and the run-length condition: token s is locally stable at t if $D_{s,t-r} \leq \delta_{\text{tok}}$ for $r = 0, \ldots, \Omega_{\text{tok}} - 1$. The token-wise EDIT freezes s at t (setting $f_s^{(t')} \equiv f_s^{(t)}$ for all $t' > t$) whenever this condition holds.

D.2. Per-Token Certificates

Lemma 7 (Local run-length bound) *If $D_{s,t-r} \leq \delta_{tok}$ for $r = 0, \ldots, \Omega_{tok} - 1$, then*

$$TV(Q_s^{(t)}, Q_s^{(t-\Omega_{tok})}) \leq \Omega_{tok}\sqrt{\frac{\delta_{tok}}{2}}. \tag{26}$$

Proof Triangle inequality and Pinsker's inequality, as in Lemma 1. ∎

Let $j_s^*(t) = \arg\max_j Q_s^{(t)}(j)$ and $m_s(t) = Q_{s,(1)}^{(t)} - Q_{s,(2)}^{(t)}$ be the local top-2 margin.

Theorem 8 (Dominant subspace-component invariance per token) *If the condition of Lemma 7 holds and $\Omega_{tok}\sqrt{\delta_{tok}/2} < \frac{1}{2}m_s(t)$, then $j_s^*(t') = j_s^*(t)$ for all $t' \in \{t - \Omega_{tok}, \ldots, t\}$.*

Proof Identical to Theorem 2 but applied to $Q_s^{(\cdot)}$. ∎

D.3. Freezing Safety Under Weak Coupling

We quantify how freezing one token perturbs the global distribution $P^{(t)}$.

Assumption 9 (Weak cross-token coupling) *There exists $\beta_s \geq 0$ such that, if two states at step r differ only in token s by Δ_s (that is, $f_s^{(r)} \mapsto f_s^{(r)} + \Delta_s$), then their next-step global distributions satisfy*

$$TV(\tilde{P}^{(r+1)}, \tilde{P}'^{(r+1)}) \leq \beta_s \|\Delta_s\|_2. \tag{27}$$

This β_s can be estimated on validation by finite-difference probes.

Assumption 10 (Post-unmasking contraction) *Within a block after the last unmasking, the Markov operators contract TV with coefficient $\alpha < 1$ as in Assumption 3.*

Theorem 11 (Safety of freezing token s) *Suppose token s satisfies the local stability condition at time t, and set*

$$\varepsilon_s = \max_{r \in \{t-\Omega_{tok}+1,\ldots,t\}} \|f_s^{(r)} - f_s^{(r-1)}\|_2. \tag{28}$$

If token s is frozen at t, then for all $u \geq 1$,

$$TV(\tilde{P}_{frozen}^{(t+u)}, \tilde{P}_{unfrozen}^{(t+u)}) \leq \frac{\beta_s}{1-\alpha} \varepsilon_s. \tag{29}$$

Consequently, if

$$\Omega\sqrt{\frac{\delta}{2}} + \frac{\beta_s}{1-\alpha}\varepsilon_s < \frac{1}{2}m_t, \tag{30}$$

where m_t is the global top-2 margin at t, then the global argmax remains unchanged forever (on the fixed support) after freezing token s.

Proof One-step deviation is at most $\beta_s \varepsilon_s$ by Assumption 9. Propagating under Assumption 10 yields a geometric tail bound $\sum_{j \geq 0} \alpha^j \beta_s \varepsilon_s = \frac{\beta_s}{1-\alpha}\varepsilon_s$. Combine with Theorem 2. ∎

Construction of U and practical tuning: A simple choice is to take $\bar{U}_B$ from Equation 3, compute its left singular vectors, and set U to the top $k \in \{2, 3, 4\}$ vectors. Empirically, $k = 3$ or $k = 4$ provides good stability-efficiency trade-offs. On validation, choose $(\delta_{tok}, \Omega_{tok})$ to maximize frozen-token count subject to Theorem 8's margin condition.

Appendix E. Subspace EDIT: Replacing the Reasoning Vector by a Subspace

E.1. Definition

Let $U \in \mathbb{R}^{d \times k}$ with orthonormal columns ($k \geq 1$). Replace the scalar alignment in Equation 5 by a subspace score:

$$\text{Sim}_s^{(t)} = \|U^\top f_s^{(t)}\|_2 \quad \text{or} \quad \text{Sim}_s^{(t)} = \frac{\|U^\top f_s^{(t)}\|_2}{\|f_s^{(t)}\|_2} \quad \text{(subspace cosine)}, \tag{31}$$

and form $P^{(t)}$ from Equation 6 with the same fixed in-block temperature τ_{blk}. All other components of EDIT (matched-support renormalization, KL divergence, run-length rule) remain unchanged.

E.2. Inherited Guarantees

Proposition 12 (Guarantees are shape-agnostic in the similarity) *The statements and proofs of Lemma 1, Theorems 2–5, and Corollary 6 hold verbatim under the subspace similarity above.*

Proof The guarantees depend only on the distributions $\tilde{P}^{(\cdot)}$ and their KL/TV relations. The construction of $\mathrm{Sim}_s^{(t)}$ enters only through $P^{(t)}$'s definition with a fixed temperature. ∎

Constructing U: Two practical choices are available. First, perform SVD of $\bar{U}_B$ (Equation 3) and retain the top k left singular vectors. Second, use CCA between step-averaged preconditioned gradients and activations $\{f_s\}$ in the chosen module. Ablations suggest small k values (2–4) are sufficient and can stabilize earlier than single-vector approaches, potentially offering improved efficiency-accuracy trade-offs.

Appendix F. EDIT Algorithm

Algorithm 1 presents the full EDIT procedure, while Figure 6 provides a visual illustration of its workflow.

Appendix G. Gradient-Based Justification for Early Termination

To determine when denoising steps can be truncated safely, we compare inference pseudo-gradients with the SFT gradients $G_{k,B}$ on LoRA-B layers. During inference, at each denoising step $t \in \{1, \ldots, T_b\}$, the model produces logits $z_t(s)$ for every token position s in the block of length L. Since no ground-truth labels are available at inference time, the only informative signal comes from the evolution of predictive distributions across steps. We denote $p_\theta(z_t(s))$ as the model's prediction for token s at step t and $p_\theta(z_{t+1}(s))$ as the refined prediction at step $t+1$. The KL divergence between them measures how much the model's belief changes across steps, with larger values indicating ongoing refinement and smaller values indicating stabilization. We therefore define the pseudo-gradient as

$$\tilde{G}_{t,B} = \nabla_B \sum_{s \in S_{t+1}} \mathrm{KL}\left(p_\theta\big(z_t(s)\big) \,\big|\, p_\theta\big(z_{t+1}(s)\big)\right), \tag{32}$$

where S_{t+1} denotes the visible token set at step $t+1$. We compute the pseudo-gradient by evaluating the KL divergence between consecutive predictive distributions at steps t and $t+1$, restricted to S_{t+1} so that only effective (unmasked) tokens contribute. Backpropagating this divergence through the LoRA-B parameters yields $\tilde{G}_{t,B}$, and we record its root-mean-square (RMS) magnitude as a scalar summary for step t. Repeating this across all denoising steps produces a trajectory of pseudo-gradients that characterizes the sensitivity of inference dynamics.

For training, we take the gradients $G_{k,B}$ observed at each SFT step k, compute their RMS magnitudes, and summarize them by a mean μ_{SFT} and a variance band. These gradients fluctuate around the mean within a bounded band, defining the stable regime in which the model was optimized. By overlaying the inference pseudo-gradients $\tilde{G}_{t,B}$ with this SFT reference, we obtain a principled test of alignment. Convergence is declared when pseudo-gradients (1) approach μ_{SFT} and (2) remain within this band, beyond which further denoising adds cost without benefit. Initially, $\tilde{G}_{t,B}$ deviates from the SFT regime, but after several iterations it reaches a convergence point $t_{\mathrm{conv}} = \arg\min_t |\mathrm{RMS}(\tilde{G}_{t,B}) - \mu_{\mathrm{SFT}}|$, after which the pseudo-gradients oscillate around the SFT

Algorithm 1 EDIT: Early Diffusion Inference Termination

Require: Input sequence; thresholds δ, Ω; block temperature τ_{blk}; fine-tuning steps $\mathcal{K}$
Ensure: Generated text with adaptive early termination

1: **// Phase 1: Training-Time Metadata Extraction (one-time)**
2: **for** each fine-tuning step $k = 1$ to $\mathcal{K}$ **do**
3:      Update AdamW moments $M_{k,B}$, $V_{k,B}$ using Eq. 1
4:      Compute update tensor $U_{k,B}$ using Eq. 2
5: **end for**
6: Compute AdamW evolution tensor $\bar{U}_B$ using Eq. 3
7: Reduce to feature vector $\mathbf{u}$ using Eq. 4
8: Store $\mathbf{u}$ as metadata for inference use
9: **// Phase 2: Inference-Time Early Termination (uses precomputed $\mathbf{u}$)**
10: **for** each diffusion block $b = 1$ to B **do**
11:      Initialize visible set $\mathcal{S}_1$ by unmasking schedule
12:      Compute activations $\mathbf{f}_s^{(1)}$ for $s \in \mathcal{S}_1$
13:      Compute initial distribution $P^{(1)}$ using Eq. 5 and 6
14:      Set stability counter $c \leftarrow 0$
15:      **for** denoising step $t = 2$ to T_b **do**
16:          Update visible set $\mathcal{S}_t$ according to unmasking schedule
17:          Compute activations $\mathbf{f}_s^{(t)}$ for $s \in \mathcal{S}_t$
18:          Compute distribution $P^{(t)}$ using Eq. 5 and 6
19:          Set intersection $\mathcal{I}_t = \mathcal{S}_{t-1} \cap \mathcal{S}_t$
20:          Renormalize to $\tilde{P}^{(t)}$, $\tilde{P}^{(t-1)}$ using Eq. 7
21:          Compute $D_t = D_{\text{KL}}(\tilde{P}^{(t)} \parallel \tilde{P}^{(t-1)})$ using Eq. 8
22:          **if** $D_t < \delta$ **then**
23:              $c \leftarrow c + 1$
24:          **else**
25:              $c \leftarrow 0$
26:          **end if**
27:          **if** $c \geq \Omega$ **then**
28:              **break** **//** Early termination for block b
29:          **end if**
30:      **end for**
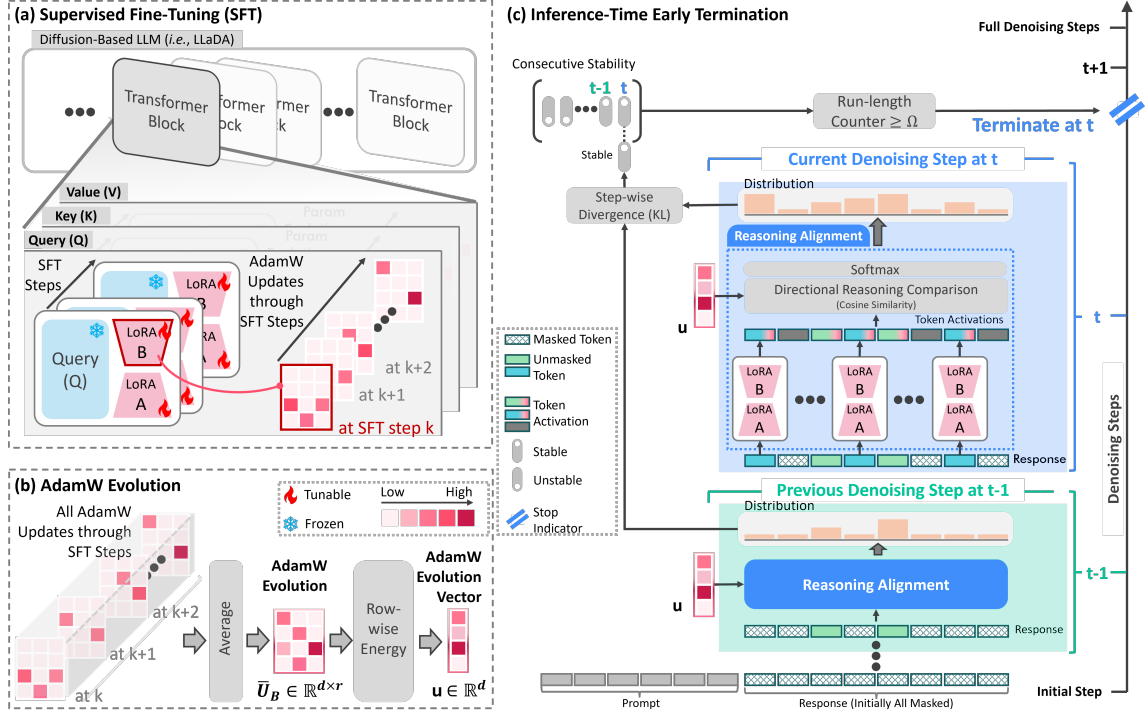31: **end for**

Figure 6: Overview of EDIT. *(a) Supervised Fine-Tuning (SFT)*: AdamW moment estimates track LoRA-B updates across steps, where some parameters consistently receive strong, directionally-aligned updates that encode reasoning patterns. *(b) AdamW Evolution*: Aggregating these updates yields a compact evolution vector **u** that encodes reasoning-relevant parameter importance. *(c) Inference-Time Early Termination*: At inference, token activations are compared with the preserved evolution vector **u**; reasoning alignment is monitored via cosine similarity and KL divergence across steps. Once stability persists for consecutive steps, termination occurs before full denoising, reducing cost without loss of quality.

mean μ_{SFT} in a manner statistically consistent with $G_{k,B}$. This indicates that inference has entered the same training-consistent regime, and further denoising steps add computation without providing additional alignment benefit.

Empirically, on the GPQA benchmark (sequence length 128) we analyze the second diffusion block and observe in Figure 7 that the pseudo-gradients reach a convergence point (marked by the yellow ▼) at the 19-th denoising step. Beyond this point, they fluctuate stably around the SFT mean, indicating entry into the training-consistent regime. Terminating at ∼20 steps per block therefore preserves fidelity while reducing computation, consistent with Table 2, which shows an average of 40.3 steps for two diffusion blocks (*i.e.*, ∼20 steps each) on GPQA, and with Table 1, which confirms that accuracy remains competitive.

After the convergence point at step 19, the pseudo-gradients stay within the SFT variance band, oscillating around the mean, but a spike appears near step 25. This behavior may reflect the dynamics of late denoising. At this stage, most unmasked (visible) tokens have stabilized, while updates focus on the remaining masked positions. These late updates often involve low-information elements such as function words, punctuation, or minor phrasing, which carry little semantic weight

18

but can still trigger abrupt shifts in the predictive distribution. Importantly, because these refinements concern filler-like tokens, they do not compromise the earlier convergence, which already ensures fidelity comparable to full denoising.
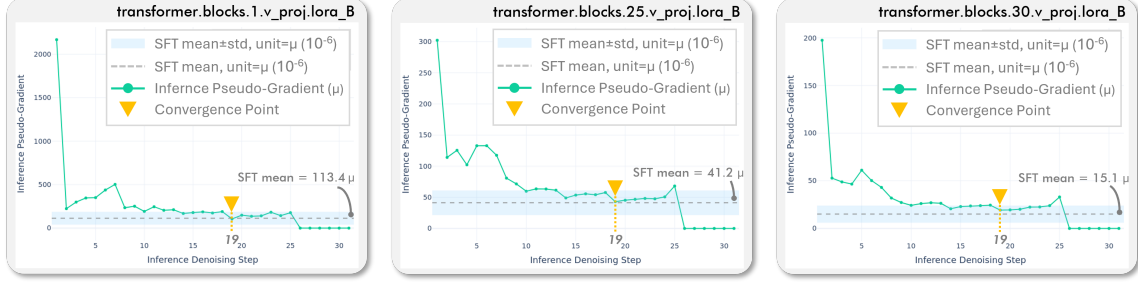

Figure 7: Gradient-based analysis of training–inference alignment on GPQA (sequence length 128, second diffusion block). The curve shows the RMS pseudo-gradients $\tilde{G}_{t,B}$ across denoising steps, compared against the SFT gradient mean (dashed) and variance band (shaded). The convergence point (yellow ▼) occurs at the 19-th step, after which pseudo-gradients oscillate stably around the SFT mean. This alignment indicates that terminating at ∼20 steps per block maintains fidelity comparable to full denoising while reducing computation, consistent with Table 2 (40.3 steps for two blocks).

We also show additional examples in Figure 8 to 11, which presents Countdown, Sudoku, MATH500, and GSM8K. Their convergence points occur at the 20-th, 18–19-th, 19-th, and 23-rd steps respectively. In each task, terminating around these points preserves fidelity while reducing computation, consistent with the average step counts reported in Table 2 (40.4, 38.3, 38.1, and 42.8 steps for two blocks). These results demonstrate that pseudo-gradient convergence provides a consistent signal of reasoning step completion across diverse benchmarks.
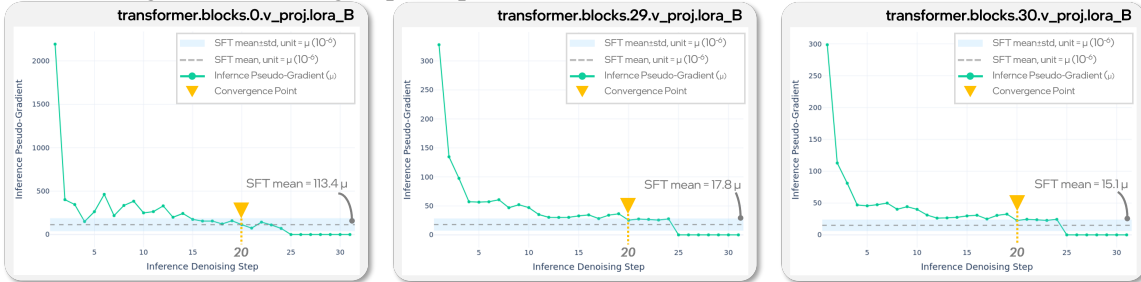

Figure 8: Countdown (sequence length 128, second diffusion block). The convergence point (yellow ▼) occurs at the 20-th step. Terminating at ∼20 steps per block maintains fidelity comparable to full denoising while reducing computation, consistent with Table 2 (40.4 steps for two blocks).
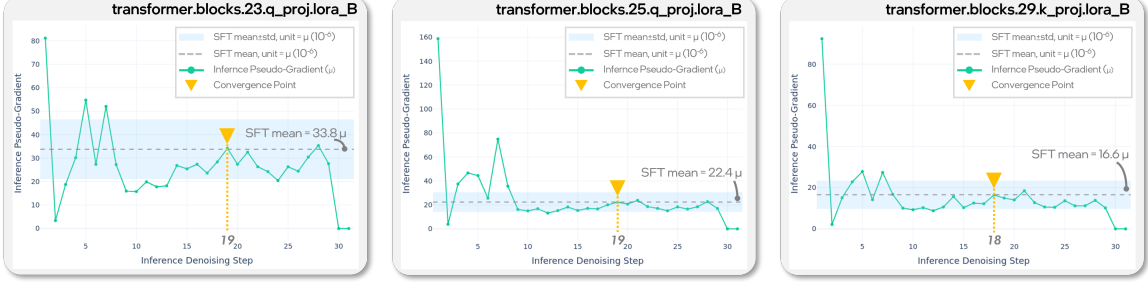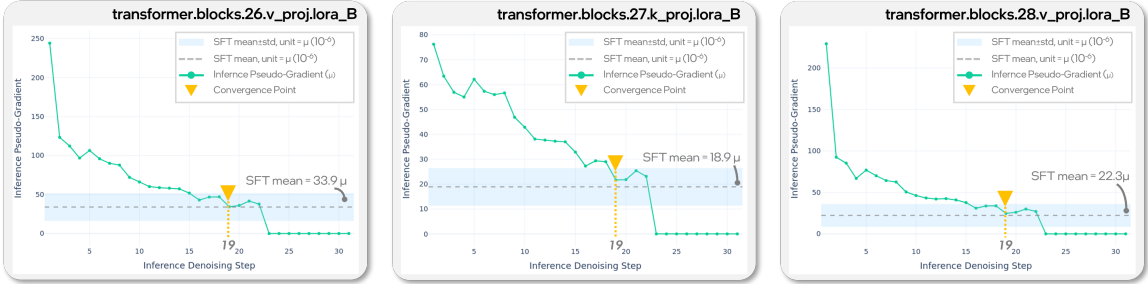
Figure 9: Sudoku (sequence length 128, second diffusion block). The convergence point (yellow ▼) occurs at the 18-th and 19-th steps. Terminating at ∼19 steps per block maintains fidelity comparable to full denoising while reducing computation, consistent with Table 2 (38.3 steps for two blocks).

Figure 10: MATH500 (sequence length 128, second diffusion block). The convergence point (yellow ▼) occurs at the 19-th step. Terminating at ∼19 steps per block maintains fidelity comparable to full denoising while reducing computation, consistent with Table 2 (38.1 steps for two blocks).
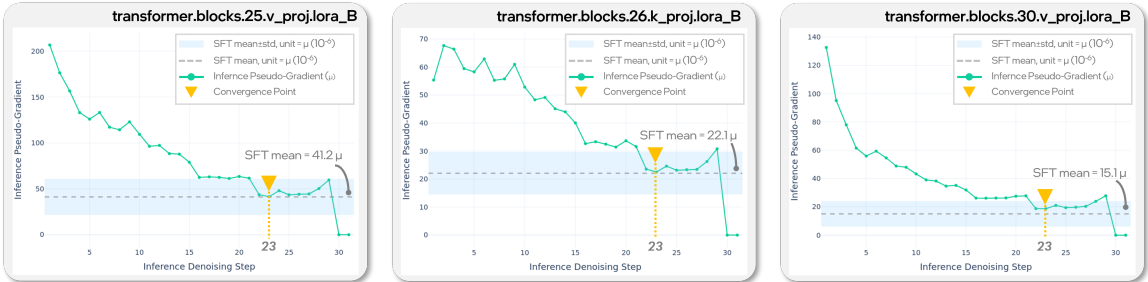

Figure 11: GSM8K (sequence length 128, second diffusion block). The convergence point (yellow ▼) occurs at the 23-rd step. Terminating at ∼22 steps per block maintains fidelity comparable to full denoising while reducing computation, consistent with Table 2 (42.8 steps for two blocks).