Reasoning in Diffusion Large Language Models is Concentrated in Dynamic Confusion Zones

Ranfei Chen†
Institute of Computing Technology
Chinese Academy of Sciences
Beijing, China
chenranfei22@mails.ucas.ac.cn

Ming Chen†
Computer Network Information Center
Chinese Academy of Sciences
Beijing, China
chenming24@mails.ucas.ac.cn

Kaifei Wang
Institute of Computing Technology
Chinese Academy of Sciences
Beijing, China
wangkaifei20@mails.ucas.ac.cn

***Abstract*—Diffusion Large Language Models (dLLMs) are rapidly emerging alongside autoregressive models as a powerful paradigm for complex reasoning, with reinforcement learning increasingly used for downstream alignment. Existing trajectory-based RL methods uniformly allocate policy gradients across denoising steps, implicitly treating all steps as equally important. We challenge this assumption by analyzing trajectories with several step-level metrics: entropy-based uncertainty, Confidence-Margin (CM) uncertainty, and Rate of Entropy Change (RoEC). These reveal structured "zones of confusion": transient spikes in uncertainty and instability that strongly predict final success or failure, while most steps remain stable. We propose Adaptive Trajectory Policy Optimization (ATPO), a lightweight step-selection strategy that dynamically reallocates gradient updates to these high-leverage steps without changing the RL objective, rewards, or compute budget. Using a hybrid RoEC+CM rule, ATPO delivers substantial gains in reasoning accuracy and training stability across benchmarks, showing that exploiting trajectory dynamics is key to advancing dLLM RL.[1]**

***Index Terms*—Diffusion Language Models, Reinforcement Learning, Trajectory Dynamics, Uncertainty Quantification, Adaptive Optimization.**

I. Introduction

The landscape of large-scale language modeling is rapidly evolving, with diffusion Large Language Models (dLLMs) [19], [22], [35] developing alongside traditional autoregressive (AR) models and gaining increasing attention as an alternative generative paradigm for complex reasoning tasks [6], [41]. To unlock their full potential and align them with human objectives, reinforcement learning (RL) is indispensable. Recent work has made RL on dLLMs practically feasible through trajectory-consistent training schemes [7], [27], [34], variance-reduced estimators and stabilized objectives [20], [24], [28], [32], [42], and test-time scaling and decoding techniques [2], [15], [18], [23], [25], [29], [30].

Despite this progress, large-scale RL training for dLLMs remains brittle in practice: runs often exhibit high gradient variance, unstable learning curves, and extreme sensitivity to the choice of checkpoint. We argue that a central, but rarely scrutinized, design choice underlies these symptoms: how the policy gradient budget is allocated along the trajectory.

Current trajectory-based methods [7], [26], [27], [34] almost universally adopt a pragmatic shortcut to remain computationally tractable: **uniform subsampling**, where policy gradients are estimated from only a sparse, evenly spaced subset of trajectory steps. This design choice is built on a powerful but largely unexamined assumption: **all steps in the reasoning process are of equal importance**. In other words, most existing RL methods for dLLMs implicitly assume that every denoising step is equally valuable for learning, thus it suffices to estimate gradients from a uniformly subsampled subset of the trajectory.

This paper argues that this assumption is not just an oversimplification—it is flawed and a primary bottleneck hindering progress. By treating a highly dynamic reasoning process as a uniform sequence, existing methods waste valuable computation on trivial steps of routine elaboration while failing to capture the brief, decisive moments where model's reasoning is forged. This leads to high-variance gradients, unstable training, and ultimately, suboptimal performance.

Our contribution is to replace this flawed assumption with a data-driven understanding of the model's internal reasoning dynamics. We conduct a systematic analysis of dLLM denoising trajectories and reveal a highly non-uniform landscape of reasoning. We discover two key properties: 1) trajectories are punctuated by identifiable **"zones of confusion"**—periods of high uncertainty and abrupt belief changes that are highly predictive of the final outcome; and 2) the locations of these critical junctures are **dynamic**, shifting throughout training and across different problems. These findings motivate a simple question: *if only a handful of steps truly matter, why should we treat all of them the same in policy optimization?*

In summary, this paper makes three contributions:

- **Analysis of trajectory dynamics.** We provide, to our knowledge, the first systematic analysis of reasoning dynamics in dLLM denoising trajectories using entropy, confidence margin, and Rate of Entropy Change. We show that trajectories exhibit highly structured and dynamically evolving *zones of confusion* whose locations shift over training and are strongly correlated with final success or failure, thereby invalidating the common assumption of uniformly important steps.

†These authors contributed equally to this work.

[1]Our code is publicly available at: https://github.com/Aczy156/ATPO.

- **Principle of adaptive gradient allocation.** Motivated by this analysis, we propose a simple design principle for RL with dLLMs: the policy gradient budget should be allocated adaptively according to trajectory uncertainty and instability, rather than uniformly across steps.
- **ATPO framework and empirical validation.** We instantiate this idea in **ATPO**, a lightweight framework that replaces uniform subsampling with a hybrid adaptive step-selection strategy based on RoEC and CM, while keeping the RL objective, reward design, and overall compute budget unchanged. ATPO is compatible with existing trajectory-based RL methods such as d2, GDPO, and MDPO, and in our experiments it yields consistent improvements in both final accuracy and training stability on several challenging reasoning benchmarks.

Taken together, reasoning in diffusion language models is concentrated in a few dynamic confusion zones, and reallocating the gradient budget toward these steps is both effective and minimally intrusive. In the remainder of this paper, we first review background and related work, then analyze trajectory dynamics, introduce the ATPO framework, and finally present experiments and discussion.

II. The Dynamics of Reasoning in Denoising Trajectories

Trajectory-based RL methods typically rely on uniform subsampling [26], estimating policy gradients from a sparse, evenly spaced subset of denoising steps and implicitly assuming that all selected steps contribute equally. Formally, the final advantage $A^{\pi_\theta}(y)$ is broadcast uniformly to every selected segment's gradient, regardless of its content or position:

$$\nabla_\theta J(\theta) = \mathbb{E}_{y\sim\pi_\theta}\left[\sum_{i\in\mathcal{S}_{\text{uniform}}} \left(\nabla_\theta \log \pi_\theta(y_{\text{seg}_i}) \cdot A^{\pi_\theta}(y)\right)\right] \quad (1)$$

This section challenges this assumption of uniformity, first by formalizing how to quantify trajectory instability, and second by empirically demonstrating that these instabilities are both critical and dynamic.

A. Formalizing Uncertainty and Instability

To analyze trajectories, we define step-level difficulty signals: scalar quantities computed at denoising step t from the predictive distribution $p_t(y|x)$ that reflect how "hard" that step is. We focus on two families of such signals—*static uncertainty metrics* and a *dynamic instability metric*—and compute them per completion token before averaging over completion tokens and over the batch, yielding a batch-level trajectory of difficulty scores indexed by t.

Static Uncertainty Metrics. At any single step t, we measure the model's uncertainty using two static step-level signals, both computed from the completion-token logits and then averaged over tokens and batch:

- **Entropy-based Uncertainty**: The Shannon entropy $H(p_t) = -\sum_v p_t(v)\log p_t(v)$ measures the "spread" of the predictive distribution. High entropy indicates the model is uncertain, assigning comparable probabilities to many different tokens.
- **Confidence-Margin Uncertainty**: Denoted CM, defined as the difference between the probabilities of the two most likely tokens, $p_{t,1} - p_{t,2}$. A small margin indicates the model is "conflicted" between its top choices, signaling a point of high local uncertainty. In our implementation, we use the *inverse* confidence margin, $1/(p_{t,1}-p_{t,2})$, as a numerically stable monotone proxy, but conceptually CM still refers to the underlying margin.

Dynamic Instability Metric. Beyond static uncertainty, we also track the "instability" or "surprise" between consecutive steps. A stable trajectory should exhibit a smooth, monotonic evolution of the model's beliefs, while instability corresponds to abrupt shifts. We formalize this as the divergence between consecutive predictive distributions, p_{t-1} and p_t. A principled measure for this is the Kullback-Leibler (KL) divergence:

$$D_{KL}(p_t||p_{t-1}) = \sum_v p_t(v)\log\frac{p_t(v)}{p_{t-1}(v)} \quad (2)$$

A high KL divergence signifies a sharp change in the model's belief state. However, computing token-level KL divergence at every step is computationally expensive. As a more efficient proxy, we consider the change in a summary statistic of the distribution—its entropy. We define **RoEC**, the Rate of Entropy Change, as our dynamic step-level difficulty signal, computed on the batch-averaged entropy trace:

$$\text{RoEC}_t = |H(p_t) - H(p_{t-1})| \quad (3)$$

A high RoEC value signals that the distribution's shape has changed significantly, making it a computationally feasible and effective heuristic for detecting these "surprise points" or moments of high dynamic instability.

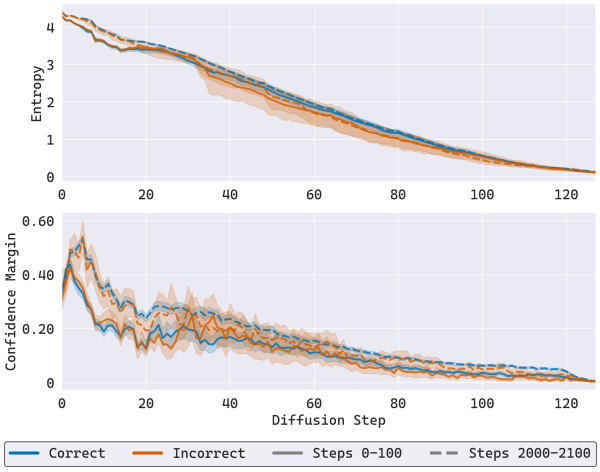

Fig. 1. Static uncertainty metrics during denoising on GSM8K. The **top** subplot shows the average Entropy-based Uncertainty, and the **bottom** subplot shows the average Confidence Margin (CM), both plotted over diffusion steps. **Color** distinguishes correct (blue) vs. incorrect (orange) samples, and **line style** marks an early phase of training (after 100 updates; solid) vs. a late phase of training (after 2000 updates; dashed). These results correspond to the "Static Uncertainty Metrics" analysis in the main text.

B. Empirical Validation: Signatures of Dynamic Difficulty

To ground this formalism and connect it to real tasks, we investigate whether these metrics can signal critical junctures in real-world reasoning benchmarks. We analyze the denoising trajectories on the GSM8K dataset, visualizing the average Entropy-based Uncertainty and CM in Figure 1. The figure reveals two key insights:

1. Uncertainty as an Error Proxy. First, we test if uncertainty correlates with errors. In Figure 1, the curves for incorrect solutions are significantly higher and more volatile than the smooth, decaying curves of correct solutions. These "zones of confusion" confirm that high uncertainty is a strong proxy for reasoning difficulty.

2. The Evolution of Difficulty During Training. Second, we analyze how these zones of confusion evolve. We compare the uncertainty curves from an early training phase (**solid lines**) and a late training phase (**dashed lines**). For incorrect samples, the peaks of uncertainty are more pronounced in the early-to-mid trajectory during early training. As training progresses, the model masters these initial steps, and the uncertainty peaks shift towards later, more complex parts of the reasoning process.

This dual dynamic provides strong, data-driven motivation for our work. It shows that the importance of different steps is highly non-uniform and changes over time. In practice, this means that under uniform subsampling, most of the gradient budget is spent on low-uncertainty, low-instability steps, while only a small number of high-leverage "zones of confusion" actually determine success or failure. Any fixed subsampling strategy is therefore inherently suboptimal. This analysis naturally suggests that an adaptive approach is necessary.

III. Background and Related Work

A. Diffusion Large Language Models

dLLMs generate text through an iterative denoising process. Starting from a sequence of fully masked tokens, the model, f_θ, is trained to predict the original text by progressively filling in the masks based on the surrounding context. This non-autoregressive nature allows for parallel decoding and flexible generation orders, distinguishing dLLMs from traditional AR models [19], [35]. The core training objective typically minimizes a variant of the negative evidence lower bound (ELBO), training the model to predict original tokens from a corrupted version [7]. This paradigm has been successfully scaled [22], extended to long contexts [8], [16], and adapted for multimodal applications [14], [17], [33], [36].

B. Reinforcement Learning for Diffusion Language Models

Applying policy gradient methods to dLLMs is challenging due to their non-causal structure and the intractability of the marginal likelihood. A series of works has gradually made this feasible. Early trajectory-consistent methods such as diffu-GRPO [39], MDPO [7], CT-GRPO [34], and TraceRL [27] bridge the training-inference divide by optimizing the policy over the same multi-step path used at inference time. Complementary approaches like GDPO [20], LLaDA 1.5 [42], SPG [28], AWM [32], and wd1 [24] focus on stabilizing objectives and reducing gradient variance.

Another line of work explores richer process signals and test-time techniques. SAPO [31], MDPO [7], and IGPO [40] exploit intermediate denoising steps through step-aware rewards, over-denoising, or inpainting-based hints, while methods such as RFG [2], Fast-dLLM [29], [30], dllm-cache [15], Sparse-dLLM [23], CreditDecoding [25], and dInfer [18] improve inference-time behavior without changing the training objective.

Our Work in Context. Despite their diversity, these methods share a common design choice: they either treat all denoising steps as equally important, explicitly via uniform subsampling (as in d2-StepMerge [26] and related schemes [34]) or implicitly by averaging losses uniformly over steps. Our work is the first to directly challenge this assumption by systematically analyzing the non-uniform, dynamic structure of denoising trajectories and then adapting the step selection strategy accordingly. In this sense, ATPO is orthogonal and complementary: it can be layered on top of existing objectives, rewards, and test-time techniques while focusing specifically on *where* along the trajectory to allocate the gradient budget. Rather than redefining the reward or introducing a new objective, ATPO can be plugged into CT-GRPO-style trajectory-aligned scoring, GDPO-like group objectives, or stabilized estimators such as SPG and AWM as a generic, time-step allocation module.

IV. Method: An Adaptive Approach

Motivated by the non-uniform, dynamic difficulty structure revealed in Section II and the limitations of uniform step allocation in existing RL methods summarized in Section III, we propose a simple and intuitive modification to trajectory-based RL: instead of sampling steps uniformly, we should adaptively select them based on their importance. We call this framework **ATPO**. We refer to any rule that maps a full trajectory together with its step-level difficulty signals to a subset of evaluation steps as an *Adaptive Step Selection Policy*. In our experiments, ATPO instantiates three such policies: RoEC-only Step Selection, CM-Only Step Selection, and our default Hybrid RoEC+CM Step Selection.

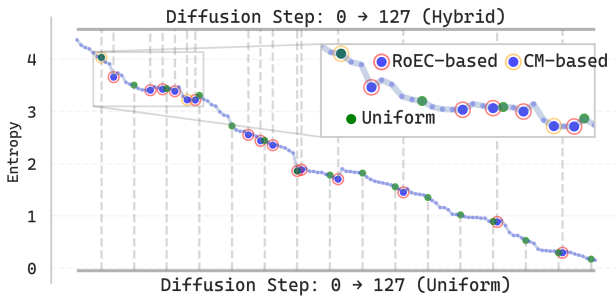

Fig. 2. Comparison of entropy curves under different step selection strategies. Uniform step selection (green) results in a smooth but potentially less informative trajectory. Strategies based on Entropy-based Uncertainty and CM are more sensitive to the sharp changes in the early-to-mid diffusion steps, allowing the model to focus on these critical phases.

A. Hybrid Adaptive Segmentation

The core of ATPO is its strategy for dynamically selecting steps. The goal is to focus computation on "critical junctures" where the model's decisions are most impactful, which our analysis identified as points of high uncertainty or high instability. This contrasts with prior methods such as d2 [26], whose StepMerge procedure partitions trajectories uniformly and therefore dilutes gradient budget on low-signal regions.

Figure 2 highlights that uniform sampling fails to capture the sharp, transient spikes exhibited by diffusion dynamics. Consequently, ATPO treats segmentation as a decision problem driven by two complementary difficulty signals. To enhance computational efficiency and stability, our segmentation strategy operates on batch-averaged dynamics. We compute step-wise entropy and confidence margins for all samples in a batch, average them to yield batch-level mean entropy and confidence-margin curves, and then use these curves for segmentation, ensuring a uniform split scheme across the batch.

Hybrid RoEC+CM Step Selection: Given a target of N segments (requiring $N-1$ split points), we apply a two-stage policy based on the batch-averaged curves. Let $\{H_t\}_{t=1}^{T}$ and $\{\mathrm{CM}_t\}_{t=1}^{T}$ denote the batch-averaged entropy and inverse confidence-margin trajectories, respectively, and define the batch-level RoEC curve as

$$\mathrm{RoEC}_t = \begin{cases} 0, & t = 1, \\ |H_t - H_{t-1}|, & t = 2, \ldots, T. \end{cases} \tag{4}$$

We then construct a split set $\mathcal{S}$ of cardinality at most $N-1$ as follows:

1) **Prioritize dynamic instability.** Compute the mean and standard deviation of the RoEC curve, denoted by μ_{RoEC} and σ_{RoEC}. We first form the RoEC candidate set

$$\mathcal{C}_{\mathrm{RoEC}} = \{\, t \mid \mathrm{RoEC}_t > \mu_{\mathrm{RoEC}} + \sigma_{\mathrm{RoEC}} \,\},$$

and add candidates to $\mathcal{S}$ in increasing order of t until either all elements of $\mathcal{C}_{\mathrm{RoEC}}$ are used or $|\mathcal{S}| = N-1$.

2) **Backfill with static uncertainty.** If $|\mathcal{S}| < N-1$, we fill the remaining slots using the static uncertainty curve. We mask out already selected indices in the CM trajectory and choose the remaining time indices with the largest values of CM_t (highest average inverse confidence margin) to add to $\mathcal{S}$ until $|\mathcal{S}| = N-1$ or no indices remain.

Finally, we deduplicate and sort $\mathcal{S}$, augment it with boundary points $\{0, T\}$, and convert the resulting ordered set $\{0 = b_1 < b_2 < \cdots < b_{N+1} = T\}$ into disjoint intervals $\{[b_i, b_{i+1})\}_{i=1}^{N}$. When traces are unavailable or the above heuristics become ill-conditioned (e.g., T is very small or all RoEC values are nearly constant), the procedure falls back to an even partition of the trajectory into N segments, guaranteeing that ATPO never performs worse than the uniform baseline. We note that the threshold $\mu_{\mathrm{RoEC}} + \sigma_{\mathrm{RoEC}}$ was found to be a robust heuristic across our preliminary experiments, but we leave a detailed sensitivity analysis of this hyperparameter to future work.

Algorithm 1 ATPO: Adaptive Trajectory Policy Optimization

Input: dataset $\mathcal{D}$, policy π_θ, reference policy π_{ref}, reward function r, trajectory length T, target segments N, learning rate η
Output: updated policy parameters θ
for each training iteration **do**
  Sample a batch of prompts $\mathcal{B} = \{q_j\}$ from $\mathcal{D}$
  Generate trajectories and traces for all $q_j \in \mathcal{B}$ with $\pi_{\theta_{\mathrm{old}}}$
  Compute batch-averaged curves $\bar{H}_{1:T}, \overline{\mathrm{CM}}_{1:T}, \overline{\mathrm{RoEC}}_{2:T}$
  $\mathcal{S} \leftarrow \mathrm{StepSelect}(\bar{H}_{1:T}, \overline{\mathrm{CM}}_{1:T}, \overline{\mathrm{RoEC}}_{2:T}, N)$ {Shared splits}
  for each trajectory j in the batch **do**
    Compute reward $R(y_j)$ and advantage $A(y_j)$ for trajectory j
    $\mathcal{L}_j \leftarrow \mathrm{PPO\_Objective}(\pi_\theta, \pi_{\mathrm{ref}}, \mathcal{S}, A(y_j))$
  end for
  $\mathcal{L} \leftarrow \frac{1}{|\mathcal{B}|} \sum_j \mathcal{L}_j$
  $\theta \leftarrow \theta + \eta \nabla_\theta \mathcal{L}$
end for

B. The ATPO Algorithm

The complete ATPO algorithm is summarized in Algorithm 1. Relative to a standard trajectory-consistent RL loop, ATPO inserts an adaptive instrumentation-and-segmentation module between trajectory generation and policy optimization. During masked diffusion, each step records four synchronized traces—intermediate sequences, token-transfer masks, averaged entropy, and averaged confidence margins—so that the subsequent segmentation operates on sufficient statistics rather than raw tokens. The batch-averaged entropy and confidence-margin curves feed the Hybrid RoEC+CM policy described above, yielding a shared set of split points $\mathcal{S}$ per mini-batch, while the stacked transfer masks preserve which completion tokens were actually edited inside every interval.

Once $\mathcal{S}$ is fixed, ATPO reuses the original StepMerge estimator to evaluate likelihoods only on the tokens touched within each interval. For a segment $[t_{\mathrm{start}}, t_{\mathrm{end}})$, StepMerge extracts the final state $x_{t_{\mathrm{end}}}$, re-masks the completion positions that changed in this segment, and feeds the masked sequence back through either the current policy or the frozen reference policy. The resulting conditional log-probabilities overwrite only the coordinates selected by the adaptive policy, so the PPO loss, KL penalty, and reward-normalized advantages are all computed on high-leverage steps while keeping the denominator of the log-ratio identical to the uniform baseline. This design makes ATPO strictly orthogonal to choices of reward model, objective variant (e.g., GRPO, PPO-Clip), or architectural accelerations, because those components consume the same per-token log-probabilities regardless of how they were obtained.

TABLE I
PERFORMANCE COMPARISON ON REASONING BENCHMARKS. THE COLUMNS LABELED 64, 128, AND 256 REFER TO THE GENERATION LENGTH.

Model	GSM8k			Sudoku			Math500			Countdown		
	64	128	256	64	128	256	64	128	256	64	128	256
LLaDA	68.70	76.70	78.20	11.70	6.70	5.50	26.00	32.40	36.20	20.70	19.50	16.00
diffu-GRPO (d1)	72.60	79.80	81.90	18.40	12.90	11.00	33.20	37.20	39.20	33.20	31.30	37.10
CJ-GRPO	67.10	77.75	–	85.69	85.37	–	23.20	28.40	–	70.80	75.59	–
GDPO	75.06	81.20	82.26	14.99	24.17	25.10	31.40	38.00	38.20	42.97	67.19	66.41
ATPO	75.51	82.49	82.94	72.9	71.48	37.40	34.00	35.60	39.80	44.53	70.31	73.69
d2		85.00			91.20			41.60			56.60	

Notes. (1) All results are collected directly from the original papers without reproduction. (2) Whenever possible, we report configurations that apply RL directly on the base model (LLaDA-8B-Instruct) without an explicit SFT stage; when multiple settings are available (e.g., [39]), we preferentially use the "pure RL" variants (such as diffu-GRPO-only) rather than SFT+RL combinations. (3) Missing entries "—" indicate that the corresponding work did not report results for that sequence length (e.g., `CJ-GRPO+EOSER` omitted longer sequences). (4) The d2 method does not explicitly specify generation hyperparameters; we report its raw numbers as presented in the paper.

V. EXPERIMENTS

A. Experimental Setup

Framework and Model. Our experimental framework is built upon the open-source `d1` project [39], ensuring our methodology is transparent and reproducible. We use the LLaDA-8B-Instruct model [19], [39] as our base dLLM for all experiments, and conduct training on a cluster of 8 NVIDIA H800-80G GPUs.

Training Details. To ensure a fair comparison with prior work, we adhere as closely as possible to the training configuration established by [39]. We employ LoRA [10] for parameter-efficient fine-tuning, with a rank of $r = 128$ and a scaling factor of $\alpha = 64$. For reinforcement learning, the generation sequence length is fixed at 256 tokens, with a per-GPU batch size of 6 and 2 gradient accumulation steps, resulting in an effective batch size of 96. We use the AdamW optimizer with a learning rate of 3×10^{-6} and gradient clipping at 0.2. To accelerate training, we leverage FlashAttention-2 [5]. Unless stated otherwise, core components such as the reward model, the PPO-style RL objective, and the underlying data pipeline remain identical across all of our variants. Within this unified framework, our key intervention is the step selection strategy (uniform vs. adaptive RoEC+CM), allowing us to isolate the impact of dynamic gradient allocation while keeping model architecture, reward, and optimizer unchanged. The overhead of this adaptive selection is minimal: the analysis pass to compute uncertainty metrics and select steps adds approximately 2-3% to the wall-clock time per training step compared to the uniform sampling baseline.

Benchmarks and Evaluation. We evaluate all methods on a diverse suite of reasoning benchmarks:

- **GSM8K** [4]: Grade school math word problems.
- **Math500** [9]: Competition-level mathematics problems.
- **Sudoku**: A logic puzzle task requiring constraint satisfaction.
- **Countdown**: A number game requiring arithmetic reasoning to reach a target number.

For all benchmarks, we report task-specific accuracy using greedy decoding and evaluate across multiple generation lengths (64, 128, and 256) to assess performance under varying computational budgets.

B. Comparison Methods and Main Results

A central tenet of our work is to enhance the reasoning capabilities of a base dLLM through direct reinforcement learning, without an initial supervised fine-tuning (SFT) stage. This approach aims to directly instill reasoning abilities via RL. Consequently, when comparing to prior methods we focus on configurations that are also reported as operating directly on the LLaDA-8B-Instruct base model without an explicit SFT phase whenever such numbers are available. In particular, we include **diffu-GRPO (d1)** [39], i.e., the pure diffu-GRPO variant without SFT that serves as the foundational trajectory-consistent RL recipe; **GDPO** [20], which focuses on variance reduction; **d2** [26], a framework utilizing uniform subsampling (StepMerge); and **CJ-GRPO** [34], which explores trajectory consistency.

To ensure a broad and objective comparison, the results for these baseline methods are cited directly from their original publications. Whenever the original work reports pure-RL configurations directly on LLaDA-8B-Instruct (e.g., diffu-GRPO-only variants in [39]), we use these numbers as our primary reference points; for methods with additional SFT or architectural variations, we report the configuration that most closely matches our setting. We align inference conditions (e.g., generation length) with the reported settings whenever possible, but do not re-train external baselines, as their full training configurations are not always public, making a perfectly controlled re-implementation infeasible. All ablations and uniform-vs-adaptive comparisons are conducted within our `d1`-based implementation under shared hyperparameters, so relative improvements among our own variants are directly comparable.

Table I presents the main experimental results. These experiments validate our thesis that, under a comparable computational budget and RL objective, adapting to trajec-

tory dynamics is more effective than uniform subsampling. ATPO consistently achieves strong performance, particularly on complex tasks and longer sequences where the non-uniform structure of the denoising trajectory is most pronounced. For instance, on GSM8K with a sequence length of 256, ATPO achieves an accuracy of **82.94**, outperforming all compared direct-RL methods. The robust improvements across diverse tasks like Sudoku and Countdown further underscore that reallocating gradient updates towards high-uncertainty, high-instability regions is a general and effective strategy.

C. Training Stability and Ablation Analysis

Beyond absolute performance, a key benefit derived from our analysis is improved training stability. To isolate the impact of step selection, we conduct an ablation study under a fixed RL setup where we keep the model, reward, and objective unchanged, and only vary the Adaptive Step Selection Policy. To ensure a fair comparison, all variants were trained with identical hyperparameters. Specifically, we compare four variants: *uniform* subsampling (the standard choice in prior work), *ATPO with RoEC-only Step Selection*, *ATPO with CM-Only Step Selection*, and our full *ATPO with Hybrid RoEC+CM Step Selection*.

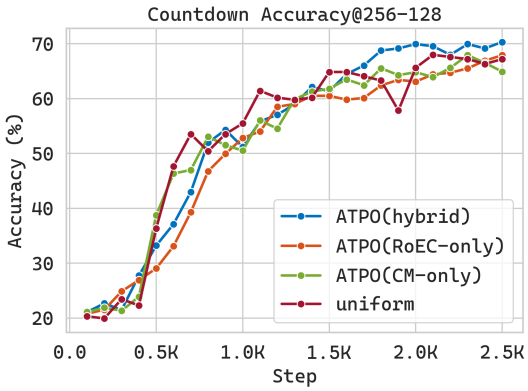

Fig. 3. Checkpoint accuracy over training steps under four step selection strategies: uniform subsampling, ATPO with RoEC-only Step Selection, ATPO with CM-Only Step Selection, and ATPO with Hybrid RoEC+CM Step Selection. The hybrid policy yields both the highest final accuracy and the smoothest training curve.

Figure 3 plots test accuracy as a function of training steps on a representative reasoning benchmark (Math500 with sequence length 256) for these four variants. We observe three consistent patterns. First, **ATPO (hybrid)** dominates throughout training: after an initial warm-up, it steadily climbs past 60% and reaches 70.31% at step 2500, with only minor fluctuations, indicating both faster convergence and a higher final plateau. Second, **ATPO (RoEC-only)** yields smoother curves than uniform subsampling but is noticeably *slower* in the mid-training regime: from steps 500 to 1800 it consistently lags behind uniform, only catching up and slightly surpassing it near the end (67.9% vs. 67.19%), and still remains several points below the hybrid variant. This aligns with our hypothesis that RoEC-only selection over-focuses on persistently high-entropy but low-impact regions. Third, **ATPO (CM-only)** behaves as a more aggressive yet brittle strategy: it occasionally spikes above uniform in the mid-training phase, but exhibits larger oscillations and eventually underperforms both uniform and RoEC-only (ending at 64.9%), confirming that purely local token-level conflicts are not a reliable proxy for global reasoning progress.

Overall, these results support our analysis-driven design: RoEC and CM capture complementary aspects of trajectory difficulty, and only their hybrid combination reliably concentrates gradient updates on the few high-leverage steps that determine success or failure, yielding both higher accuracy and markedly more stable training dynamics.

VI. Discussion

This paper's primary contribution is an analysis of the reasoning dynamics within dLLM trajectories. We moved beyond the simplifying assumption of uniform step importance and showed, both formally and empirically, that these pathways are characterized by non-uniform, dynamic uncertainty. The key insight is that the model's "struggle"—quantified by metrics like RoEC and CM—is not noise to be averaged out, but a signal to be leveraged.

Our adaptive method, ATPO, was proposed as a direct consequence and validation of this analysis. Its strong performance is evidence that an analysis-driven approach to RL algorithm design is effective. ATPO synthesizes goals from prior work: it maintains **trajectory consistency** (like CT-GRPO), addresses **computational efficiency** (like d2), and improves **training stability**, all through the single, unifying principle of adapting to trajectory dynamics.

It is also instructive to contrast ATPO with several closely related lines of work. SAPO [31] focuses on designing step-aware *rewards*, assuming that the locations where these rewards are applied are already given, while IGPO [40] targets early-stage exploration by injecting inpainting hints, and MDPO [7] introduces over-denoising-based *sample* filtering rather than time-step allocation. RFG [2] and other test-time scaling methods operate purely at inference time, modifying the decoding process without changing the training dynamics. In contrast, ATPO is orthogonal to these methods: instead of redefining *what* to optimize (objectives, rewards, or data), it focuses on *where* along the trajectory to allocate the limited gradient budget. In practice, the adaptive step-selection module can be plugged into CT-GRPO-style trajectory-aligned scoring (losses computed only over newly unmasked tokens), GDPO-like log-mean estimators and other stabilized objectives such as SPG or AWM, while leaving the underlying RL objective, reward function, and data pipeline unchanged; by simply toggling between adaptive and uniform step selection, ATPO can recover behavior similar to d2 [26], underscoring its role as a lightweight and generally compatible component.

However, our analysis is not exhaustive. The proposed metrics (Entropy, CM, RoEC) are effective heuristics, but other, potentially more powerful, measures of trajectory instability may exist. The overhead of the "analysis pass" to compute

these metrics is a practical trade-off, although our results suggest it is a worthwhile one. Finally, this work focuses on reasoning tasks; how these dynamics manifest in other domains, like creative writing or dialogue, remains an open and interesting question for future research.

VII. Conclusion

This paper presented an in-depth analysis of the denoising trajectory dynamics in Diffusion Language Models. We challenged the prevailing assumption of uniformity in trajectory-based reinforcement learning and demonstrated that the reasoning process is characterized by critical, non-uniform, and dynamic moments of uncertainty. By formalizing and empirically identifying these "surprise points" using information-theoretic metrics, we revealed the limitations of fixed, uniform subsampling strategies.

As a direct consequence of our analysis, we introduced ATPO, a policy optimization framework that replaces uniform subsampling with an adaptive step selection strategy. By intelligently focusing computation on the highest-leverage moments in the reasoning chain, ATPO achieves significant improvements in performance and training stability. Conceptually, our work adds an orthogonal dimension to the existing literature: instead of proposing yet another objective, reward, or estimator, we study and optimize *where* along the trajectory the gradient budget should be allocated. This work validates that a deeper, analysis-driven understanding of the underlying generative process—and, in particular, of the dynamic structure of generative trajectories—is a promising direction for developing more intelligent and efficient reinforcement learning algorithms for next-generation language models.

Acknowledgment

References

[1] Q. Chen, H. Li, L. Qin, D. Peng, J. Liu, J. Wang, C. Wu, X. Chen, Y. Du, and W. Che, "Beyond Surface Reasoning: Unveiling the True Long Chain-of-Thought Capacity of Diffusion Large Language Models," *arXiv preprint arXiv:2510.09544*, 2025.

[2] T. Chen, M. Xu, J. Leskovec, and S. Ermon, "RFG: Test-Time Scaling for Diffusion Large Language Model Reasoning with Reward-Free Guidance," *arXiv preprint arXiv:2509.25604*, 2025.

[3] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoen, and A. Tafjord, "Think you have solved question answering? try arc, the ai2 reasoning challenge," *arXiv preprint arXiv:1803.05457*, 2018.

[4] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, and others, "Training verifiers to solve math word problems," *arXiv preprint arXiv:2110.14168*, 2021.

[5] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré, "Flashattention: Fast and memory-efficient exact attention with io-awareness," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 16344–16359.

[6] S. Gong, R. Zhang, H. Zheng, J. Gu, N. Jaitly, L. Kong, and Y. Zhang, "DiffuCoder: Understanding and Improving Masked Diffusion Models for Code Generation," *arXiv preprint arXiv:2506.20639*, 2025.

[7] H. He, K. Renz, Y. Cao, and A. Geiger, "Mdpo: Overcoming the training-inference divide of masked diffusion language models," *arXiv preprint arXiv:2508.13148*, 2025.

[8] G. He, S. Nie, F. Zhu, Y. Zhao, T. Bai, R. Yan, J. Fu, C. Li, and B. Yuan, "UltraLLaDA: Scaling the Context Length to 128K for Diffusion Large Language Models," *arXiv preprint arXiv:2510.10481*, 2025.

[9] D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt, "Measuring mathematical problem solving with the math dataset," *arXiv preprint arXiv:2103.03874*, 2020.

[10] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022.

[11] J. Huang, Y. Zhang, Y. Yang, B. Huang, B. Qi, D. Liu, and L. Zhang, "Mask Tokens as Prophet: Fine-Grained Cache Eviction for Efficient dLLM Inference," *arXiv preprint arXiv:2510.09309*, 2025.

[12] Z. Huang, Z. Chen, Z. Wang, T. Li, and G.-J. Qi, "Reinforcing the diffusion chain of lateral thought with diffusion language models," *arXiv preprint arXiv:2505.10446*, 2025.

[13] P. Li, S. Yan, J. Tsai, R. Zhang, R. An, Z. Guo, and X. Gao, "Adaptive Classifier-Free Guidance via Dynamic Low-Confidence Masking," *arXiv preprint arXiv:2505.20199*, 2025.

[14] S. Li, K. Kallidromitis, H. Bansal, A. Gokul, Y. Kato, K. Kozuka, J. Kuen, Z. Lin, K.-W. Chang, and A. Grover, "Lavida: A large diffusion language model for multimodal understanding," *arXiv preprint arXiv:2505.16839*, 2025.

[15] Z. Liu, Y. Yang, Y. Zhang, J. Chen, C. Zou, Q. Wei, S. Wang, and L. Zhang, "dllm-cache: Accelerating diffusion large language models with adaptive caching," *arXiv preprint arXiv:2506.06295*, 2025.

[16] X. Liu, Z. Liu, Z. Huang, Q. Guo, Z. He, and X. Qiu, "LongLLaDA: Unlocking Long Context Capabilities in Diffusion LLMs," *arXiv preprint arXiv:2506.14429*, 2025.

[17] T. Ma, M. Zhang, Y. Wang, and Q. Ye, "Consolidating Reinforcement Learning for Multimodal Discrete Diffusion Models," *arXiv preprint arXiv:2510.02880*, 2025.

[18] Y. Ma, L. Du, L. Wei, K. Chen, Q. Xu, K. Wang, G. Feng, G. Lu, L. Liu, X. Qi, and others, "dInfer: An Efficient Inference Framework for Diffusion Language Models," *arXiv preprint arXiv:2510.08666*, 2025.

[19] S. Nie, F. Zhu, Z. You, X. Zhang, J. Ou, J. Hu, J. Zhou, Y. Lin, J.-R. Wen, and C. Li, "Large language diffusion models," *arXiv preprint arXiv:2502.09992*, 2025.

[20] K. Rojas, J. Lin, K. Rasul, A. Schneider, Y. Nevmyvaka, M. Tao, and W. Deng, "Improving Reasoning for Diffusion Language Models via Group Diffusion Policy Optimization," *arXiv preprint arXiv:2510.08554*, 2025.

[21] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[22] Y. Song, Z. Zhang, C. Luo, P. Gao, F. Xia, H. Luo, Z. Li, Y. Yang, H. Yu, X. Qu, and others, "Seed diffusion: A large-scale diffusion language model with high-speed inference," *arXiv preprint arXiv:2508.02193*, 2025.

[23] Y. Song, X. Liu, R. Li, Z. Liu, Z. Huang, Q. Guo, Z. He, and X. Qiu, "Sparse-dllm: Accelerating diffusion llms with dynamic cache eviction," *arXiv preprint arXiv:2508.02558*, 2025.

[24] X. Tang, R. Dolga, S. Yoon, and I. Bogunovic, "wd1: Weighted policy optimization for reasoning in diffusion language models," *arXiv preprint arXiv:2507.08838*, 2025.

[25] K. Wang, Z. Jiang, H. Feng, W. Zhao, L. Liu, J. Li, Z. Lan, and W. Lin, "CreditDecoding: Accelerating Parallel Decoding in Diffusion Large Language Models with Trace Credits," *arXiv preprint arXiv:2510.06133*, 2025.

[26] G. Wang, Y. Schiff, G. Turok, and V. Kuleshov, "d2: Improved Techniques for Training Reasoning Diffusion Language Models," *arXiv preprint arXiv:2509.21474*, 2025.

[27] Y. Wang, L. Yang, B. Li, Y. Tian, K. Shen, and M. Wang, "Revolutionizing reinforcement learning framework for diffusion large language models," *arXiv preprint arXiv:2509.06949*, 2025.

[28] C. Wang, P. Rashidinejad, D. Su, S. Jiang, S. Wang, S. Zhao, C. Zhou, S. Z. Shen, F. Chen, T. Jaakkola, and others, "SPG: Sandwiched Policy Gradient for Masked Diffusion Language Models," *arXiv preprint arXiv:2510.09541*, 2025.

[29] C. Wu, H. Zhang, S. Xue, Z. Liu, S. Diao, L. Zhu, P. Luo, S. Han, and E. Xie, "Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding," *arXiv preprint arXiv:2505.22618*, 2025.

[30] C. Wu, H. Zhang, S. Xue, S. Diao, Y. Fu, Z. Liu, P. Molchanov, P. Luo, S. Han, and E. Xie, "Fast-dLLM v2: Efficient Block-Diffusion LLM," *arXiv preprint arXiv:2509.26328*, 2025.

[31] S. Xie, L. Kong, X. Song, X. Dong, G. Chen, E. P. Xing, and K. Zhang, "Step-Aware Policy Optimization for Reasoning in Diffusion Large Language Models," *arXiv preprint arXiv:2510.01544*, 2025.

[32] S. Xue, C. Ge, S. Zhang, Y. Li, and Z.-M. Ma, "Advantage Weighted Matching: Aligning RL with Pretraining in Diffusion Models," *arXiv preprint arXiv:2509.25050*, 2025.

[33] L. Yang, Y. Tian, B. Li, X. Zhang, K. Shen, Y. Tong, and M. Wang, "Mmada: Multimodal large diffusion language models," *arXiv preprint arXiv:2505.15809*, 2025.
[34] J. Yang, G. Chen, X. Hu, and J. Shao, "Taming Masked Diffusion Language Models via Consistency Trajectory Reinforcement Learning with Fewer Decoding Step," *arXiv preprint arXiv:2509.23924*, 2025.
[35] J. Ye, Z. Xie, L. Zheng, J. Gao, Z. Wu, X. Jiang, Z. Li, and L. Kong, "Dream 7b: Diffusion large language models," *arXiv preprint arXiv:2508.15487*, 2025.
[36] Z. You, S. Nie, X. Zhang, J. Hu, J. Zhou, Z. Lu, J.-R. Wen, and C. Li, "Llada-v: Large language diffusion models with visual instruction tuning," *arXiv preprint arXiv:2505.16933*, 2025.
[37] R. Yu, X. Ma, and X. Wang, "Dimple: Discrete diffusion multimodal large language model with parallel decoding," *arXiv preprint arXiv:2505.16990*, 2025.
[38] A. Zhan, "Principled and Tractable RL for Reasoning with Diffusion Language Models," *arXiv preprint arXiv:2510.04019*, 2025.
[39] S. Zhao, D. Gupta, Q. Zheng, and A. Grover, "d1: Scaling reasoning in diffusion large language models via reinforcement learning," *arXiv preprint arXiv:2504.12216*, 2025.
[40] S. Zhao, M. Liu, J. Huang, M. Liu, C. Wang, B. Liu, Y. Tian, G. Pang, S. Bell, A. Grover, and others, "Inpainting-Guided Policy Optimization for Diffusion Large Language Models," *arXiv preprint arXiv:2509.10396*, 2025.
[41] H. Zheng, S. Gong, R. Zhang, T. Chen, J. Gu, M. Zhou, N. Jaitly, and Y. Zhang, "Continuously augmented discrete diffusion model for categorical generative modeling," *arXiv preprint arXiv:2510.01329*, 2025.
[42] F. Zhu, R. Wang, S. Nie, X. Zhang, C. Wu, J. Hu, J. Zhou, J. Chen, Y. Lin, J.-R. Wen, and others, "LLaDA 1.5: Variance-Reduced Preference Optimization for Large Language Diffusion Models," *arXiv preprint arXiv:2505.19223*, 2025.
[43] F. Zhu, Z. You, Y. Xing, Z. Huang, L. Liu, Y. Zhuang, G. Lu, K. Wang, X. Wang, L. Wei, and others, "LLaDA-MoE: A Sparse MoE Diffusion Language Model," *arXiv preprint arXiv:2509.24389*, 2025.