Fast-dLLM++: Fréchet Profile Decoding for Faster Diffusion LLM Inference

Siva Rajesh Kasa[1*] **Yasong Dai**[2*] **Sumit Negi**[1] **Hongdong Li**[1,2]

[1]Amazon Inc. [2]School of Computing, Australian National University

kasa@u.nus.edu yasong.dai@anu.edu.au

Abstract

Diffusion large language models promise parallel token generation, yet inference remains bottlenecked by deciding which masked tokens can be safely committed together. Fast-dLLM addressed this with KV caching and confidence-guided parallel decoding, but its decoding theory uses a homogeneous high-confidence assumption that effectively reduces each candidate set to its weakest selected token. We argue that this leaves speed on the table because real decoding steps exhibit heterogeneous confidence profiles. We propose **Fast-dLLM++**, a training-free extension that introduces **Fréchet profile decoding**: selecting parallel commit sets from the full sorted confidence profile rather than a single worst-case confidence. The resulting rule is a heterogeneous-confidence generalization of Fast-dLLM's factor selector and it recovers the previous rule exactly in the equal-confidence case and adds a provable **heterogeneity bonus** when the selected tokens have uneven confidences. Fast-dLLM++ leaves the model, diffusion process, and cache implementation entirely unchanged, making it a drop-in replacement for existing Fast-dLLM decoding. Experiments on GSM8K, MATH, HumanEval, and MBPP with the LLaDA-8B model show that the theoretical improvement translates directly into empirical gains: profile-aware selection improves the accuracy–throughput frontier by exploiting safe parallelism that weakest-token rules miss, achieving up to 37% higher throughput at comparable accuracy. Our code is available at https://github.com/Ringo-Star/FastdLLM_plusplus.

1 Introduction

Masked diffusion language models (MDLMs) generate text by iteratively unmasking tokens from a fully masked sequence [53, 27, 40, 41]. Unlike autoregressive models, they can in principle decode multiple positions simultaneously, offering a path to substantially faster inference. However, parallel decoding introduces a *curse of parallelism* [18, 31, 3]: if several masked positions are decoded independently from their marginal distributions, the resulting combination may be incoherent even when each individual token appears likely.

Fast-dLLM [45] addressed this challenge through confidence-guided parallel decoding rules, *threshold and factor*, that decide how many tokens to commit per step. The factor rule accepts a candidate set of size n when $(n+1)(1-c_{(n)}) < f$, where $c_{(n)}$ is the *weakest* confidence among the top-n candidates and f is a tunable parameter. This criterion is grounded in a worst-case analysis that assumes all selected tokens share the same confidence level. This homogeneous-confidence assumption is conservative. In practice, the confidence profile across masked positions is highly heterogeneous: a few positions may have near-certain predictions while others are moderately confident. By compressing this profile to a single scalar i.e. the weakest confidence, the factor rule discards information that could safely license additional parallelism.

*Equal contribution.

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

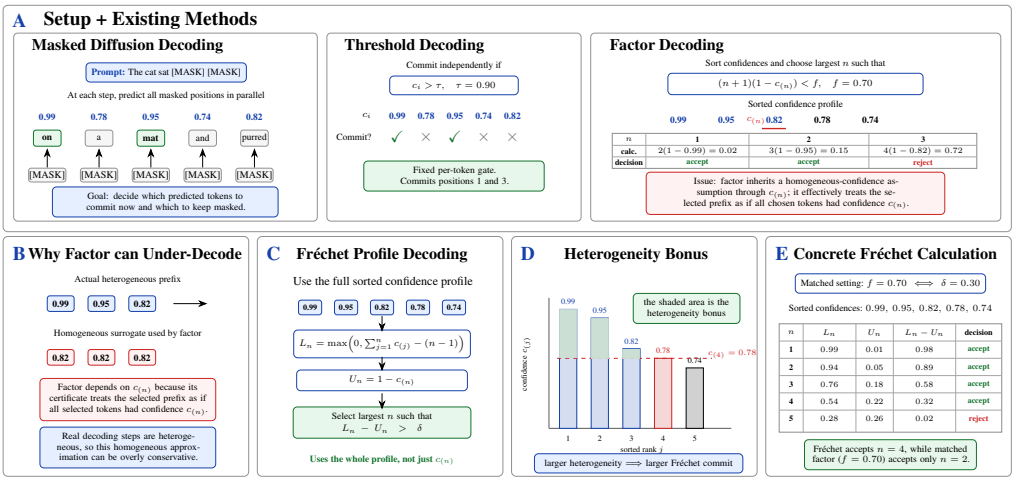

Figure 1: **Fréchet Profile Decoding exploits heterogeneous confidence profiles to commit more tokens per denoising step.** At each masked diffusion step, the model predicts candidate tokens with confidences c_i; green marks committed tokens, gray marks deferred tokens, and red marks the factor bottleneck $c_{(n)}$. The green shaded region denotes the heterogeneity bonus $B_n = \sum_{j<n}(c_{(j)} - c_{(n)})$, which is the extra profile information discarded by the homogeneous weakest-token factor rule.

We propose **Fast-dLLM++**, which replaces the weakest-token selector with *Fréchet profile decoding*. The key idea is to use the full sorted confidence profile $(c_{(1)}, c_{(2)}, \ldots, c_{(n)})$ rather than only $c_{(n)}$. For each candidate prefix of size n, we compute a Fréchet lower bound L_n on the probability that all n marginal greedy tokens are jointly correct, and an upper bound U_n on the probability of any competing tuple. We commit the largest prefix satisfying $L_n - U_n > \delta$, where $\delta \geq 0$ is a margin parameter.

Our contributions are as follows:

- We introduce **Fréchet profile decoding**, a training-free parallel decoding rule for diffusion LLMs that uses the full sorted confidence profile, rather than a fixed confidence threshold or only the weakest selected token.
- We prove that the Fréchet criterion gives a heterogeneous-confidence sufficient condition under which greedy parallel decoding matches the true joint greedy decision (Theorem 1). We further show that, without additional dependence information across positions, the Fréchet bound is the strongest distribution-free conservative certificate available from marginal confidences alone.
- We show that Fréchet decoding reduces exactly to Fast-dLLM factor decoding in the equal-confidence case. We then derive a *heterogeneity-bonus decomposition* (Proposition 3) that explains when and why profile-aware decoding can commit more tokens per denoising step.
- We evaluate Fast-dLLM++ on GSM8K, MATH, HumanEval, and MBPP using LLaDA-8B-Instruct and Dream-v0-Base-7B across multiple cache regimes and generation lengths. The results show that the theoretical improvement translates into empirical gains in the accuracy–throughput frontier. We further evaluate Fast-dLLM++ on multimodal reasoning tasks with LLaDA-V.

2 Preliminary

Masked Diffusion Models. Masked diffusion models [33, 40] are a dominant paradigm in discrete diffusion models by introducing absorbing states in the Markov process. For later convenience, we denote $\boldsymbol{x} := (x_1, \ldots, x_L) \in \mathcal{V}^L$ a clean token sequence and let `[M]` be an absorbing mask symbol. At a given reverse diffusion step t, let E denote the current *evidence*: the prompt, all previously unmasked tokens, and the current partially masked sequence. Let $\mathcal{M}(E)$ be the set of masked positions.

$$q_t(\boldsymbol{x}_t \mid \boldsymbol{x}_0) = \prod_{i=1}^{L} \left[(1-\gamma(t))\,\mathbf{1}\{x_{t,i} = x_{0,i}\} + \gamma(t)\,\mathbf{1}\{x_{t,i} = \texttt{[M]}\}\right]. \tag{1}$$

In the continuous-time formulation, a noise level $t \in [0, 1]$ is associated with a masking rate schedule $\gamma(t)$, where $\gamma(0) = 0$ and $\gamma(1) = 1$. The learned reverse process predicts the clean token at each masked position, i.e., $p_\theta(x_{0,i} \mid \boldsymbol{x}_t, t), \quad i \in \mathcal{M}_t := \{i : x_{t,i} = \texttt{[M]}\}$.

Earlier discrete diffusion work introduced structured categorical corruption kernels and absorbing-state variants [1], while later masked diffusion formulations simplified the objective into a weighted masked-token prediction loss and showed that such models can scale competitively for language modeling [40, 41, 36, 49].

$$\mathcal{L}_{\text{MDLM}}(\theta) = \mathbb{E}_{\boldsymbol{x}_0, t, \boldsymbol{x}_t \sim q_t} \left[w(t) \sum_{i \in \mathcal{M}_t} -\log p_\theta(x_{0,i} \mid \boldsymbol{x}_t, t) \right], \tag{2}$$

where $w(t)$ is determined by the chosen variational, score-entropy, or simplified masked-diffusion objective. During inference, the model computes marginal distributions at denoising step k over masked positions and commits a subset $S_k \subseteq \mathcal{M}_k$:$\hat{x}_i = \arg\max_{v \in \mathcal{V}} p_\theta(v \mid \boldsymbol{x}_k, t_k), i \in S_k$.

The Curse of Parallelism in dLLM Decoding. The main speed advantage of diffusion language models comes from parallelism (increasing $|S_k|$) at inference time [51, 6]. However, the same operation creates the *curse of parallelism*: increasing the number of simultaneously unmasked tokens reduces the number of denoising steps, but also increases the gap between the true conditional joint distribution and the product of token-wise marginals used by dLLMs [22].

Specifically, let E denote the current evidence, and let $I = \{i_1, \ldots, i_n\}$ be a set of masked positions proposed for simultaneous decoding. The ideal conditional distribution is

$$p(\mathbf{z} \mid E) = p(X_{i_1} = z_1, \ldots, X_{i_n} = z_n \mid E). \tag{3}$$

A standard parallel decoder instead uses the product of marginal predictions,

$$q(\mathbf{z} \mid E) = \prod_{j=1}^{n} p_j(z_j \mid E), \quad p_j(z_j \mid E) := p(X_{i_j} = z_j \mid E). \tag{4}$$

This conditional independence approximation ignores dependencies among the tokens in I. Consequently, decoding many positions in parallel can finalize locally plausible tokens that are globally inconsistent. This effect has been identified as a root cause of quality degradation in parallel dLLM decoding [45, 4].

Confidence-Aware Parallel Decoding. Confidence-aware decoding attempts to exploit parallelism only when the conditional-independence approximation is almost harmless. For each masked position i, define the model confidence $c_i = \max_{v \in \mathcal{V}} p_\theta(v \mid \boldsymbol{x}_k, t_k)$. A threshold-based decoder commits $S_k = \{i \in \mathcal{M}_k : c_i \geq \tau\}$ and, if $S_k = \emptyset$, commits the single highest-confidence token. Fast-dLLM [45] provides a useful theoretical justification for this rule through a high-confidence condition on product-of-marginals decoding. The theorem also gives a direct interpretation of confidence-aware decoding: If all selected tokens have confidence at least $1 - \epsilon$, then committing n tokens is safe only when ϵ is small relative to n.

However, the main theorem in Fast-dLLM [45] only makes use of the smallest marginal confidence ϵ, which is a homogeneous case of exploiting dLLM's inference confidence profile. Therefore, our core insight is that a strictly better bound can be developed by extending the preceding theorem to heterogeneous case.

3 Fréchet Profile Decoding

Notation. For each $i \in \mathcal{M}(E)$, the diffusion language model induces a marginal predictive distribution $p_i(v \mid E) := p_\theta(X_i = v \mid E)$, $v \in \mathcal{V}$. Define the *marginal greedy token* $x_i^\star := \arg\max_{v \in \mathcal{V}} p_i(v \mid E)$ and its *confidence* $c_i := p_i(x_i^\star \mid E)$. For a candidate set $S = \{i_1, \ldots, i_n\} \subseteq \mathcal{M}(E)$, let $x_S^\star := (x_{i_1}^\star, \ldots, x_{i_n}^\star)$ be the tuple of marginal greedy tokens, with confidences sorted as $c_{(1)} \geq c_{(2)} \geq \cdots \geq c_{(n)}$. Let $P_S(z \mid E) := p_\theta(X_S = z \mid E)$ denote the true joint conditional distribution over the selected positions, and let $Q_S(z \mid E) := \prod_{i \in S} p_i(z_i \mid E)$ denote the product-of-marginals approximation used by parallel decoding.

Fréchet profile quantities. For a candidate prefix of size n, define the *Fréchet lower bound* on joint correctness $L_n := \max\Big\{0,\ \sum_{j=1}^{n} c_{(j)} - (n-1)\Big\}$, the *competing-event upper bound* $U_n := 1 - c_{(n)}$, and the *Fréchet score* $G_n := L_n - U_n$. Given a margin $\delta \geq 0$, Fréchet profile decoding selects $n^\star := \max\{n : G_n > \delta\}$, with the convention $n^\star = 1$ if the set is empty (ensuring progress).

3.1 From weakest-token to profile-aware decoding

Fast-dLLM factor decoding selects a parallel commit set using only the weakest confidence $c_{(n)}$ among the top-n candidate tokens. This is conservative when confidence profiles are heterogeneous: a single moderately confident token ,can prevent committing several nearly certain tokens.

We instead use the full sorted confidence profile. For the top-n candidates, the Fréchet–Hoeffding lower bound gives L_n as a worst-case lower bound on the probability that all selected marginal argmax tokens are jointly correct. Any alternative tuple must differ in at least one coordinate and therefore has probability at most $U_n = 1 - c_{(n)}$. We commit the largest prefix satisfying $L_n - U_n > \delta$. This rule is training-free, requires only sorting and prefix sums, and is compatible with the cache mechanisms of Fast-dLLM. Algorithm 1 presents the complete procedure. The only change relative to Fast-dLLM is the token-selection rule in lines 11–18; the model, diffusion schedule, and caching are untouched.

4 Theoretical Analysis

The theory is organized around the amount of information available to the decoder.

Proof organization. For readability, the main text states each result together with its intuition and practical implication. All formal proofs for Sections 4 and their certification extensions in C are collected in Appendices A and B, respectively.

Marginal-only certificates. First, we ask what can be guaranteed using only token-level marginal confidences. This is the setting of threshold, factor, and our Fréchet selector. Theorem 1 shows that the Fréchet lower bound gives a distribution-free certificate: if the lower bound on the marginal-greedy tuple exceeds an upper bound on every competitor, then the parallel greedy commit agrees with the true joint greedy decision.

Relationship to Fast-dLLM factor decoding. Second, we show that Fast-dLLM's factor rule is not a separate principle but the equal-confidence specialization of the Fréchet certificate. When all selected tokens have the same confidence, Fréchet exactly recovers factor decoding under the parameter mapping $f = 1 - \delta$.

Algorithm 1 Fast-dLLM++: Fréchet Profile Decoding

Require: Diffusion LM p_θ, prompt p_0, generation length L, block size B, margin $\delta \geq 0$, cache mode cache $\in \{\text{NONE}, \text{PREFIX}, \text{DUAL}\}$

1: Initialize sequence $x \leftarrow [p_0;\ \texttt{[M]}, \ldots, \texttt{[M]}]$
2: **for** each generation block $\mathcal{B}$ **do**
3: Initialize or refresh cache according to cache
4: **while** $\mathcal{B}$ contains masked positions **do**
5: Run p_θ on the active block/context to obtain logits
6: **for** each masked position $i \in \mathcal{M}$ **do**
7: $\hat{x}_i \leftarrow \arg\max_{v \in \mathcal{V}} p_\theta(X_i = v \mid x)$
8: $c_i \leftarrow \max_{v \in \mathcal{V}} p_\theta(X_i = v \mid x)$
9: **end for**
10: Sort confidences: $c_{(1)} \geq c_{(2)} \geq \cdots \geq c_{(m)}$
11: **for** $n = 1, \ldots, m$ **do**
12: $L_n \leftarrow \max\Big\{0,\ \sum_{j=1}^{n} c_{(j)} - (n-1)\Big\}$
13: $U_n \leftarrow 1 - c_{(n)}$
14: $G_n \leftarrow L_n - U_n$
15: **end for**
16: Select $n^\star \leftarrow \max\{n : G_n > \delta\}$
17: **if** no such $n^\star$ exists **then**
18: $n^\star \leftarrow 1$ {ensure progress}
19: **end if**
20: Reveal the $n^\star$ masked positions with largest confidence: $x_i \leftarrow \hat{x}_i$
21: **end while**
22: **end for**
23: **return** x

Why heterogeneous profiles help. Third, we decompose the Fréchet score into a weakest-token factor core plus a nonnegative heterogeneity bonus. This explains the main algorithmic advantage: factor treats the selected prefix as if all tokens were as uncertain as the weakest token, while Fréchet gives credit to the stronger tokens in the prefix.

Going beyond marginal-only guarantees. Finally, we consider what happens when we know something about the dependence structure. The total-variation and KL stability results show that if the true joint distribution P_S is close to the product-of-marginals approximation Q_S, then the product-mode decision is stable even beyond the conservative Fréchet regime.

Together, these results separate two regimes: Fréchet is the sharp marginal-only certificate for heterogeneous confidence profiles, while dependence-aware information can certify additional parallel commits when the true joint is close to the product approximation.

4.1 Heterogeneous-Confidence Greedy Equivalence

Theorem 1 (Heterogeneous-confidence greedy equivalence)**.** *Fix evidence E and a candidate set $S = \{i_1, \dots, i_n\}$. Let $x_S^\star$ be the tuple of marginal greedy tokens and let $c_{(1)} \geq \cdots \geq c_{(n)}$ be the sorted marginal confidences. Define*

$$L_n = \max\left\{0, \sum_{j=1}^{n} c_{(j)} - (n-1)\right\}, \qquad U_n = 1 - c_{(n)}. \tag{5}$$

If $L_n > U_n$, then $x_S^\star$ is the unique maximizer of the true joint conditional distribution:

$$x_S^\star = \arg\max_{z \in \mathcal{V}^n} P_S(z \mid E). \tag{6}$$

Intuition. The selected tuple $x_S^\star$ is the one obtained by taking the marginal argmax at every selected position. The Fréchet lower bound L_n is a worst-case lower bound on the probability that all these marginal choices are jointly correct. Any competing tuple must disagree in at least one coordinate, so its probability is at most the probability that the weakest selected token is wrong, $U_n = 1 - c_{(n)}$. If the lower bound on the selected tuple exceeds the upper bound on every competitor, the selected tuple must be the unique joint mode. Next, we show under equal confidence this reduces to exact factor decoding.

Corollary 2 (Equal-confidence reduction)**.** *Suppose $c_{(1)} = \cdots = c_{(n)} = c$. Then the Fréchet margin criterion $G_n := L_n - U_n > \delta$ reduces to*

$$c > 1 - \frac{1-\delta}{n+1}. \tag{7}$$

Fast-dLLM factor decoding with parameter f accepts n when $(n+1)(1-c_{(n)}) < f$, which under equal confidences gives $c > 1 - f/(n+1)$. Setting $f = 1 - \delta$ makes the two criteria identical:

$$\textit{Fréchet}(\delta) \;\equiv\; \textit{Factor}(f = 1-\delta) \qquad \textit{(equal-confidence case)}. \tag{8}$$

For the parameter matching $f = 1 - \delta$, assume $\delta \in [0, 1]$ so that the matched factor is nonnegative.

Intuition. Factor decoding can be viewed as replacing the whole selected confidence profile by a flat surrogate in which every selected token has confidence $c_{(n)}$. When the real profile is actually flat, this surrogate is exact, and Fréchet reduces to factor. The difference between the two only appears when the confidence profile is heterogeneous. The following proposition formalizes the same.

Proposition 3 (Heterogeneity bonus)**.** *Assume $L_n > 0$. The Fréchet score $G_n := L_n - U_n$ decomposes as*

$$G_n = \underbrace{(n+1)\, c_{(n)} - n}_{F_n \text{ (factor core)}} + \underbrace{\sum_{j=1}^{n-1} \big(c_{(j)} - c_{(n)}\big)}_{B_n \text{ (heterogeneity bonus)}}, \tag{9}$$

with $B_n \geq 0$. Equality $B_n = 0$ holds iff the confidence profile is flat.

Eq. 9 is the central insight: the factor core F_n depends only on the weakest confidence $c_{(n)}$, while B_n captures the additional information from stronger tokens. When the profile is heterogeneous, $B_n > 0$ and Fréchet decoding can accept larger commit sets than factor decoding at the same margin.

Intuition. The first term $F_n = (n+1)c_{(n)} - n$ is exactly the weakest-token logic used by factor. The second term B_n is the extra area between the confidence profile and the weakest selected confidence. This area is nonnegative and measures how much information factor discards by compressing the profile to $c_{(n)}$.

Corollary 4 (Matched-factor dominance)**.** *Compare Fréchet decoding with margin δ to factor decoding with matched parameter $f = 1 - \delta$. If factor accepts a prefix of size n, then Fréchet also accepts that prefix. Moreover, Fréchet strictly accepts a prefix rejected by matched factor whenever*

$$F_n \le \delta < F_n + B_n,$$

where $F_n = (n+1)c_{(n)} - n$ and $B_n = \sum_{j=1}^{n-1}(c_{(j)} - c_{(n)})$.

Practical takeaway. At matched aggressiveness $f = 1 - \delta$, factor acceptance implies Fréchet acceptance. Strict separation occurs only when the heterogeneity bonus is large enough to cross the decision boundary, i.e. $F_n \le \delta < F_n + B_n$. Thus the benefit of Fréchet is not automatic for every non-flat profile; it appears when the bonus is large enough to change the accept/reject decision.

Corollary 5 (Threshold decoding under the Fréchet criterion)**.** *Theorem 1 certifies any proposed commit set, including one produced by threshold decoding. For $S_\tau = \{i : c_i \ge \tau\}$ with $n_\tau = |S_\tau|$, every selected confidence is at least τ, so $G_{n_\tau} \ge \max\{0,\, n_\tau\tau - (n_\tau - 1)\} - (1 - \tau)$, i.e. on the positive branch a threshold set is Fréchet-certified whenever $(n_\tau + 1)\tau - n_\tau > \delta$.*

This also exposes threshold decoding's structural gap: a fixed per-token gate neither adapts its required confidence to the commit-set size nor uses the profile above the weakest token. When this coarse, τ-only corollary fails, the actual selected profile can still be evaluated by the general test.

Remark 6 (Adaptive-factor interpretation)**.** *Fréchet decoding with margin δ is equivalent to factor decoding with a data-dependent effective factor*

$$f_{\text{eff}}(n) = 1 - \delta + B_n, \tag{10}$$

whose aggressiveness increases when the confidence profile is heterogeneous.

This provides the clearest intuition: threshold decoding uses a fixed cutoff; factor decoding is set-size-aware; Fréchet decoding is *profile-aware* and *data-adaptive*, automatically becoming more aggressive when the evidence supports it.

4.2 Dependence-Aware Extensions

Fréchet is intentionally marginal-only. The following stability result is not used by Fast-dLLM++, but clarifies how stronger guarantees would become possible if additional dependence information were available.

Theorem 1 is exact over the worst case of all joint distributions consistent with the marginals (the Fréchet class). When additional information about the dependence structure is available, sharper guarantees are possible.

Here Δ_Q is the mode gap of the product approximation: it measures how much more probability Q_S assigns to the product-mode tuple $x_S^\star$ than to the best competing tuple. A large Δ_Q means the product approximation has a clear winner.

Lemma 7 (Mode stability under total variation)**.** *Let $Q_S(z \mid E) = \prod_{i \in S} p_i(z_i \mid E)$ be the product-of-marginals approximation with mode gap*

$$\Delta_Q := Q_S(x_S^\star \mid E) - \max_{z \ne x_S^\star} Q_S(z \mid E). \tag{11}$$

Assume $\Delta_Q > 0$, so that $x_S^\star$ is the unique mode of Q_S. If $d_{\mathrm{TV}}(P_S, Q_S) < \Delta_Q/2$, then $x_S^\star$ is also the unique maximizer of $P_S(\cdot \mid E)$.

Corollary 8 (Mode stability under KL divergence)**.** *If $D_{\mathrm{KL}}(P_S \| Q_S) < \Delta_Q^2/2$, then $x_S^\star$ is the unique maximizer of $P_S(\cdot \mid E)$.*

Intuition. Total variation bounds how much probability mass can change for any event when moving from Q_S to P_S. The product-mode tuple can lose at most $d_{\mathrm{TV}}(P_S, Q_S)$ probability, while any competitor can gain at most the same amount. Therefore the mode gap can shrink by at most $2d_{\mathrm{TV}}(P_S, Q_S)$. If this is smaller than Δ_Q, the winner under Q_S cannot be overtaken under P_S.

Note that $D_{\mathrm{KL}}(P_S \| Q_S)$ is exactly the conditional total correlation of the token set. A small total correlation therefore guarantees that the factorized greedy choice is stable, providing a clean separation

between marginal confidence and residual dependence. This connects our Fréchet criterion (which is marginal-only and worst-case) to a richer dependence-aware regime: when the model's joint is close to its product of marginals, even tokens that fail the Fréchet test may be safe to commit.

Extensions. The basic Fréchet selector assumes reliable marginal confidences. Appendix C gives a calibration-robust variant of the Fréchet certificate, which accounts for possible model overconfidence by replacing reported confidences with conservative lower bounds.

5 Experiments

We evaluate Fréchet profile decoding as a drop-in replacement for Fast-dLLM's threshold and factor rules on four benchmarks, three cache regimes, and multiple generation lengths.

Model and Hardware. We use LLaDA-8B-Instruct, a masked diffusion language model with 8B parameters. All experiments use greedy decoding (argmax per position) with no temperature sampling. All same-hardware comparisons use identical GPU configurations. We report results on a single NVIDIA H100 (80 GB) GPU.

Benchmarks. We evaluate on GSM8K [13] (5-shot, 8-shot for 1024-length), MATH [15] (4-shot), HumanEval [10] (0-shot), and MBPP [2] (3-shot). For GSM8K we report flexible-extract accuracy; for MATH we report `math_verify` accuracy; for HumanEval we report pass@1 after code postprocessing; for MBPP we report pass@1.

Cache regimes. We test three configurations: (i) **no cache** (nocache), where each denoising step processes the full sequence; (ii) **prefix cache** (pcache), which caches the prompt KV states; and (iii) **DualCache** (dcache), which additionally caches previously decoded block states.

Baselines. We compare against Fast-dLLM's two decoding rules: *threshold* (accept tokens above a fixed confidence τ) and *factor* (accept n tokens when $(n+1)(1-c_{(n)}) < f$). For fair comparison, we use matched parameters: Fréchet margin δ is compared against factor $f = 1 - \delta$.

Parameter selection and sensitivity. Following Fast-dLLM, we use a single global operating point for each selector in the main tables: threshold $\tau = 0.9$, factor $f = 1.0$, and Fréchet margin $\delta = 0.25$, unless otherwise stated. These values are chosen from development sweeps as robust accuracy–throughput operating points and are kept fixed across benchmarks, cache regimes, and generation lengths. Figure 2 reports the full sweep on GSM8K and Appendix H the sweeps on the other benchmarks, cache regimes, and backbones. δ is an explicit speed–quality control with smooth, monotone efficiency behavior. Our recommended default is the conservative $\delta = 0.25$ used throughout; when a development split exists, choose the fastest δ whose validation quality is within a prespecified tolerance. This tuning dimension is not unique to our method: threshold's τ and factor's f are tuned in the same manner.

Memory and implementation overhead. Fast-dLLM++ changes only the token-selection rule. It adds no model parameters, no additional KV cache, and no persistent hidden-state storage beyond the underlying Fast-dLLM cache mode. Relative to factor decoding, it operates on the same per-position confidence vector and requires only sorting and prefix sums over the active block. Thus the method has no additional persistent memory overhead and negligible compute overhead compared with a denoising forward pass.

Main results. Table 1 summarizes the main comparisons across four benchmarks at generation lengths 256 and 512. Threshold decoding ($\tau = 0.9$) is the primary decoding rule proposed in Fast-dLLM; we report improvements relative to this state-of-the-art baseline. More examples can be found in Appendix G.

Efficiency gains. Fast-dLLM's threshold decoding ($\tau = 0.9$) is the state-of-the-art parallel decoding rule for diffusion LLMs. Across all eight dataset–length settings, Fréchet profile decoding improves throughput by **1.36**$\times$ on average over threshold decoding while reducing total NFE by **29.2%**, with only a 0.48-point average accuracy change. Relative to the LLaDA-8B full-step baseline (no early stopping), Fréchet achieves **4.31**$\times$ average throughput and **79.1%** NFE reduction, demonstrating that profile-aware selection extracts substantially more safe parallelism than both the full-step baseline and the existing threshold rule. Figure 2 visualizes the full accuracy–throughput frontier on GSM8K

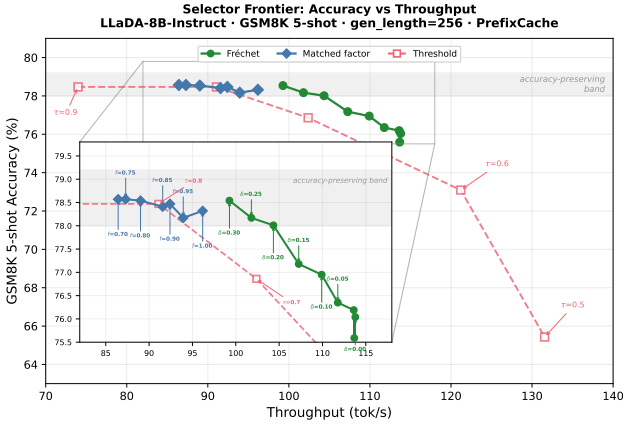

Figure 2: **Accuracy–throughput frontier on GSM8K.** Fréchet shifts the matched-factor frontier toward higher throughput, especially in the conservative regime.

by sweeping the selector parameter for each method: Fréchet margin $\delta \in [0, 0.30]$, matched factor $f = 1 - \delta$, and threshold $\tau \in [0.5, 0.9]$. Fréchet shifts the matched-factor frontier to the right: at matched aggressiveness $f = 1 - \delta$, it consistently achieves higher throughput, with the accuracy gap shrinking in the conservative regime. We therefore interpret Fréchet as improving the throughput–accuracy trade-off rather than strictly dominating every baseline point. This confirms that the heterogeneity bonus B_n (Proposition 3) translates into a measurable throughput advantage across the entire operating range, not just at a single parameter setting.

Comparison with factor at matched aggressiveness. Table 1 compares Fréchet $\delta = 0.25$ with factor at its most aggressive operating point $f = 1.0$, and Fréchet still uses fewer NFE at higher throughput and comparable accuracy. The matched factor $f = 1 - \delta = 0.75$ (Corollary 2) is more conservative and slower; across MATH, HumanEval, MBPP, both cache regimes, and Fast-dLLM-v2 (Appendix H), matched Fréchet holds accuracy within sampling noise of factor($f = 1 - \delta$) while cutting 9–20% of forward evaluations, a consistent 10–25% throughput gain. On the LLaDA-8B PrefixCache setting, Fréchet commits more than factor on 31.7% (MBPP) to 41.7% (GSM8K) of steps, and each such step removes future forward passes; these per-step savings accumulate over the full generation into the NFE reductions in Table 1.

Why can accuracy be preserved despite greater speed? Fréchet does not merely commit more tokens; it commits more tokens only when the full confidence profile provides a stronger joint-correctness certificate. In settings where the model's confidences are calibrated and the selected tokens are weakly dependent, this can reduce NFE without increasing commit errors. When these assumptions fail, especially under overconfidence or strong syntactic coupling, the method can overcommit; this is precisely the failure mode addressed by the robust and verifier-calibrated analysis in Appendix C.

5.1 Analysis

The heterogeneity bonus in practice. The theoretical heterogeneity bonus B_n (Proposition 3) predicts that Fréchet decoding gains the most when the confidence profile is uneven. We observe this empirically: the largest throughput improvements occur on GSM8K and HumanEval under prefix cache, where the model produces a mix of high-confidence function words and moderate-confidence reasoning tokens—exactly the heterogeneous regime where B_n is large.

Commit correctness. In the main setting (LLaDA-8B, PrefixCache) every parallel commit is checked against a reference: the selected positions are also decoded one token at a time from the same decoder state, and a commit is correct if it produces exactly the same tokens (Appendix I). The certificate is sound in practice: the observed fraction of correct commits never falls below the floor L_n in any margin decile of any run. The margin is predictive: measured over factor's commits, which span the whole range of margins, G_n ranks correctness with AUC .87 to .92. Fréchet achieves the

Table 1: **Benchmark results on LLaDA-8B-Instruct.** Our results are obtained from a single H100, using Prefix-Cache, block size = 32.

Dataset	Len.	Method	Acc. (%)	Tok/s ↑	NFE ↓	Tok/NFE
GSM8K (5-shot)	256	Threshold	77.6	73.8 (1.00×)	107,135	2.88
	256	Factor	78.1	96.0 (1.30×)	79,047 (↓26.2%)	3.90
	256	Fréchet	77.2	**103.8** (1.41×)	**72,881** (↓32.0%)	**4.24**
MATH (4-shot)	256	Threshold	33.1	74.4 (1.00×)	503,377	2.48
	256	Factor	32.7	97.3 (1.31×)	379,330 (↓24.6%)	3.29
	256	Fréchet	32.5	**102.5** (1.38×)	**358,178** (↓28.8%)	**3.48**
HumanEval (0-shot)	256	Threshold	40.2	77.8 (1.00×)	13,666	2.87
	256	Factor	40.7	89.6 (1.15×)	10,538 (↓22.9%)	3.78
	256	Fréchet	40.9	**107.7** (1.38×)	**9,740** (↓28.7%)	**4.06**
MBPP (3-shot)	256	Threshold	27.4	66.0 (1.00×)	34,975	2.41
	256	Factor	21.4	81.6 (1.24×)	26,870 (↓23.2%)	3.16
	256	Fréchet	25.4	**85.4** (1.29×)	**25,791** (↓26.3%)	**3.34**
GSM8K (5-shot)	512	Threshold	76.5	45.3 (1.00×)	129,791	2.75
	512	Factor	74.6	53.3 (1.18×)	99,580 (↓23.3%)	3.61
	512	Fréchet	75.6	**59.4** (1.31×)	**91,239** (↓29.7%)	**3.90**
MATH (4-shot)	512	Threshold	36.1	56.4 (1.00×)	764,793	2.83
	512	Factor	35.3	69.1 (1.23×)	585,176 (↓23.5%)	3.70
	512	Fréchet	35.5	**77.7** (1.38×)	**545,993** (↓28.6%)	**3.96**
HumanEval (0-shot)	512	Threshold	41.5	54.1 (1.00×)	27,366	2.80
	512	Factor	41.7	70.3 (1.30×)	20,463 (↓25.2%)	3.75
	512	Fréchet	41.5	**75.5** (1.40×)	**18,909** (↓30.9%)	**4.05**
MBPP (3-shot)	512	Threshold	14.2	60.8 (1.00×)	59,999	2.50
	512	Factor	12.0	80.0 (1.32×)	45,186 (↓24.7%)	3.33
	512	Fréchet	14.2	**82.7** (1.36×)	**42,893** (↓28.5%)	**3.49**

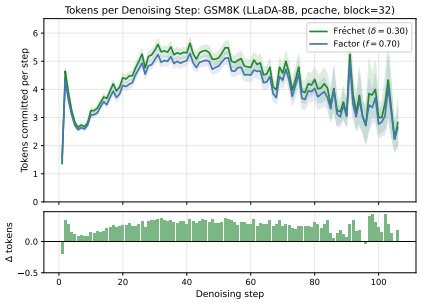

Figure 3: **Inference dynamics of tokens committed per step.**

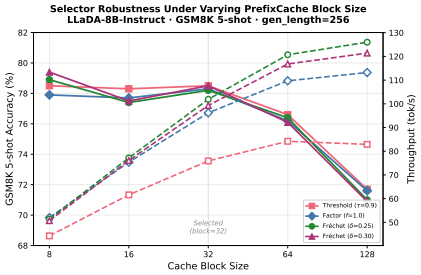

Figure 4: **Block size ablation.**

highest commit correctness of the three selectors (96.9% on GSM8K, 97.7% on MBPP) while using the fewest model evaluations. On MBPP, the drop to 21.4 pass@1 is driven by factor at its most aggressive operating point ($f = 1.0$); matched factor ($f = 0.75$, 24.0) and Fréchet (25.3, at 11% fewer NFE) recover most of it, and the calibration-robust variant of Appendix C ($\eta = 0.06$, 25.7) recovers it fully, cutting commit error from 4.17% to 0.46%. The drop is dominated by recoverable overconfidence, which affects threshold and factor at least as strongly.

Other backbones, tasks, and acceleration methods. The efficiency gain transfers to the stronger LLaDA-1.5 backbone [54], to open-ended generation (IFEval) with accuracy preserved, and to generation lengths up to 2,048 tokens (Appendix J). Inserting Fréchet at Fast-dLLM-v2's [44] intra-sub-block commit-selection step reduces NFE by 17–20% at sub-block size 4 and 24–26% at size 32 (Appendix K). Fréchet is also complementary to recent accelerators that alter other parts of the inference pipeline: applied on top of each (replacing only the commit rule), the gains are additive on both axes, improving both quality and efficiency (Table 15, Appendix L).

Figure 3 visualizes this effect at the step level. On each denoising step, Fréchet commits more tokens than matched factor because the heterogeneity bonus B_n allows it to accept larger prefixes when the top tokens are much more confident than the weakest selected token. The extra tokens per step accumulate over the decoding trajectory, reducing total NFE by 6–8% and producing the throughput gains reported in Table 1.

Margin sensitivity. The optimal margin δ varies by task: GSM8K prefers $\delta \in [0.25, 0.30]$, HumanEval prefers $\delta \in [0.20, 0.25]$, and MBPP is best with small margins $\delta \in [0.02, 0.20]$. This is consistent with the adaptive-factor interpretation (Remark 6): tasks with more heterogeneous confidence profiles benefit from tighter margins that let the heterogeneity bonus do the work. Appendix H reports the full sweeps.

Generalization to Dream. We also evaluate on Dream-v0-Base-7B (Appendix D). Fréchet achieves 1.28× average throughput over threshold with 21.1% NFE reduction, confirming that the profile-aware advantage transfers across diffusion LM architectures.

Long-context and cache-mode analysis. On GSM8K with generation length 1024, Fréchet improves throughput and Tok/NFE over matched factor across all cache modes (no cache, PrefixCache,

DualCache), showing that the selector gain persists in Fast-dLLM's long-generation regime. Full tables are in Appendix E.

Fast-dLLM++ for complex multimodal reasoning tasks. We test Fast-dLLM++ on multimodal diffusion large language model to test its robustness under mathematical reasoning settings. More experiment results are in Appendix F.

6 Related Work

Diffusion language models. Discrete and masked diffusion language models [1, 40, 41, 33] offer non-autoregressive alternatives to standard left-to-right generation. LLaDA [36] and Dream [49] scale masked diffusion to 7–8B parameters with competitive quality. We build on these models and focus on inference-time decoding rather than training.

Fast diffusion decoding. Fast-dLLM [45] accelerates diffusion LLM inference with approximate KV caching (PrefixCache, DualCache) and confidence-aware parallel decoding via threshold and factor rules. Subsequent work explores adaptive parallel decoding [18], learned parallel schedules [5, 12], and KL-guided selection [24]. Fast-dLLM++ is complementary to caching: it changes only the token-selection rule and composes with all cache modes. The key distinction is that factor decoding depends only on the weakest selected confidence $c_{(n)}$, while Fréchet uses the full sorted profile.

Concurrent acceleration methods. Concurrent training-free acceleration of dLLM inference spans several mechanisms: confidence-based selectors (KLASS [24], CCD [9]) decide per-step commit sets from marginal confidences; dependence-aware methods (DAPD [23], PUNT [3]) model what a marginal factorization ignores; adaptive schedulers (Dynamic-dLLM [46], SlowFast [43], DUS [35]) reallocate where and how many tokens are decoded; draft-and-verify decoders (WINO [17], DC-Leap [21]) trade verification compute for aggressive drafting; cache and system methods (dLLM-Cache [32], Streaming-dLLM [47], DPad [11]) cut per-step cost; adaptive block partition (Fast-dLLM-v2 [44], Swordsman [52], DSB [34]) restructures the canvas; and distilled decoders (dParallel [12], d3LLM [38]) internalize parallelism at training time. Fast-dLLM++ changes only the token-selection rule and is complementary to these mechanisms: applied on top of them it improves both quality and efficiency (Appendices L and M).

7 Limitations and Conclusion

Limitations. Fréchet certificate is the strongest distribution-free guarantee available from marginals alone, but tasks with strong syntactic or semantic coupling may require stricter margins or future dependence-aware extensions. When confidences are miscalibrated or selected tokens are strongly coupled, especially in syntactically constrained generation, profile-aware decoding may overcommit. The certificate assumes block-wise decoding; without blocks, the high-confidence end-of-sequence tokens are committed early and those positions must be excluded (Appendix L). Distilled models have sharper confidence profiles, which reduces the heterogeneity bonus. Stronger guarantees would require calibrated confidences or explicit dependence estimates. Our analysis focuses on greedy decoding, where each position contributes only its marginal argmax token and top-1 confidence. Extending profile-aware certificates to temperature sampling, nucleus sampling, or top-k token sets remains open; one possible direction is to certify set-valued correctness events rather than equality with the joint greedy mode. The selector also ignores other distributional signals, such as entropy, top-k gaps, and the full softmax shape, which may provide useful information for commit decisions. Finally, our dependence-aware stability result (Lemma 7) suggests a path beyond marginal-only selection: when selected tokens have small residual dependence, the product-mode decision is stable. Estimating such dependence online could certify additional parallel commits and yield further speedups which is an interesting future work.

Conclusion. Fast-dLLM++ shows that selector design remains a key bottleneck in diffusion LLM inference. By replacing weakest-token selection with a Fréchet profile certificate, it generalizes factor decoding and exploits heterogeneous confidence profiles. The resulting decoder is training-free, drop-in, and improves the throughput–accuracy frontier without modifying the model, diffusion process, or cache implementation.

Acknowledgments and Disclosure of Funding

This research is funded in part by an ARC (Australian Research Council) Discovery Grant of DP220100800. Prof. Hongdong Li holds concurrent appointments as a Full Professor with the ANU and as an Amazon Scholar with Amazon (part time).

References

[1] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021. 3, 10

[2] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732. 7

[3] Iskander Azangulov, Teodora Pandeva, Niranjani Prasad, Javier Zazo, and Sushrut Karmalkar. Parallel sampling from masked diffusion models via conditional independence testing. *arXiv preprint arXiv:2510.21961*, 2025. 1, 10, 35

[4] Parikshit Bansal and Sujay Sanghavi. Enabling approximate joint sampling in diffusion lms. *arXiv preprint arXiv:2509.22738*, 2025. 3

[5] Wenrui Bao, Zhiben Chen, Dan Xu, and Yuzhang Shang. Learning to parallel: Accelerating diffusion large language models via learnable parallel decoding. *arXiv preprint arXiv:2509.25188*, 2025. 10

[6] Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. *arXiv preprint arXiv:2505.24857*, 2025. 3

[7] Andrew Campbell, Valentin De Bortoli, Jiaxin Shi, and Arnaud Doucet. Self-speculative masked diffusions. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.03929. 35

[8] Mingyu Cao, Alvaro H. C. Correia, Christos Louizos, Shiwei Liu, and Lu Yin. Search or accelerate: Confidence-switched position beam search for diffusion language models. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2602.10953. 33

[9] Kecheng Chen, Ziru Liu, Xijia Tao, Hui Liu, Xinyu Fu, Suiyun Zhang, Dandan Tu, Lingpeng Kong, Rui Liu, and Haoliang Li. Beyond confidence: Adaptive and coherent decoding for diffusion language models. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2512.02044. 10, 31, 33, 35

[10] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374. 7

[11] Xinhua Chen, Sitao Huang, Cong Guo, Chiyue Wei, Yintao He, Jianyi Zhang, Hai Helen Li, and Yiran Chen. Dpad: Efficient diffusion language models with suffix dropout. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=0yOsSMU1eY. 10, 36

[12] Zigeng Chen, Gongfan Fang, Xinyin Ma, Ruonan Yu, and Xinchao Wang. dparallel: Learnable parallel decoding for dllms. *arXiv preprint arXiv:2509.26488*, 2025. 10, 31, 34, 36

[13] 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, 2021. URL https://arxiv.org/abs/2110.14168. 7

[14] Hengyu Fu, Baihe Huang, Virginia Adams, Charles Wang, Venkat Srinivasan, and Jiantao Jiao. From bits to rounds: Parallel decoding with exploration for diffusion language models. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2511.21103. 35

[15] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021. URL https://arxiv.org/abs/2103.03874. 7

[16] Michael Hersche, Nicolas Menet, Ronan Tanios, and Abbas Rahimi. Locally coherent parallel decoding in diffusion language models. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2603.20216. 35

[17] Feng Hong, Geng Yu, Yushi Ye, Haicheng Huang, Huangjie Zheng, Ya Zhang, Yanfeng Wang, and Jiangchao Yao. Wide-in, narrow-out: Revokable decoding for efficient and effective dllms. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2507.18578. 10, 33, 35

[18] Daniel Israel, Guy Van den Broeck, and Aditya Grover. Accelerating diffusion llms via adaptive parallel decoding. *arXiv preprint arXiv:2506.00413*, 2025. 1, 10

[19] Daniel Israel, Guy Van den Broeck, and Aditya Grover. Accelerating diffusion llms via adaptive parallel decoding. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2506.00413. 35

[20] Haozhe Jiang, Nika Haghtalab, and Lijie Chen. Diffusion language models are provably optimal parallel samplers. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2512.25014. 36

[21] Yanhua Jiao, Tianyi Wu, Xiaoxi Sun, Yulin Li, HuiLing Zhen, Libo Qin, Baotian Hu, Zhuotao Tian, and Min Zhang. Dc-leap: Training-free acceleration of dllms via draft-guided contiguous leaping decoding. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2607.20467. 10, 31, 32, 35

[22] Wonjun Kang, Kevin Galim, Seunghyuk Oh, Minjae Lee, Yuchen Zeng, Shuibai Zhang, Coleman Hooper, Yuezhou Hu, Hyung Il Koo, Nam Ik Cho, et al. Parallelbench: Understanding the trade-offs of parallel decoding in diffusion llms. *arXiv preprint arXiv:2510.04767*, 2025. 3, 36

[23] Bumjun Kim, Dongjae Jeon, Moongyu Jeon, and Albert No. Dapd: Dependency-aware parallel decoding via attention for diffusion llms. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2603.12996. 10, 35

[24] Seo Hyun Kim, Sunwoo Hong, Hojung Jung, Youngrok Park, and Se-Young Yun. Klass: Kl-guided fast inference in masked diffusion models. *arXiv preprint arXiv:2511.05664*, 2025. 10, 31, 35

[25] Sanghyun Lee, Seungryong Kim, Jongho Park, and Dongmin Park. Lookahead unmasking elicits accurate decoding in diffusion language models. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2511.05563. 35

[26] Ian Li, Zilei Shao, Benjie Wang, Rose Yu, Guy Van den Broeck, and Anji Liu. Breaking the factorization barrier in diffusion language models. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2603.00045. 35

[27] Pengxiang Li, Dilxat Muhtar, Tianlong Chen, Lu Yin, and Shiwei Liu. Why diffusion language models struggle with truly parallel (non-autoregressive) decoding? *arXiv preprint arXiv:2602.23225*, 2026. 1

[28] Pengxiang Li, Joey Tsai, Hongwei Xue, Kunyu Shi, and Shilin Yan. Beyond scattered acceptance: Fast and coherent inference for dlms via longest stable prefixes. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2603.05454. 35, 36

[29] Pengxiang Li, Yefan Zhou, Dilxat Muhtar, Lu Yin, Shilin Yan, Li Shen, Soroush Vosoughi, and Shiwei Liu. Diffusion language models know the answer before decoding. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2508.19982. 35

[30] Zekai Li, Ji Liu, Yiqing Huang, Ziqiong Liu, Dong Li, and Emad Barsoum. Efficient diffusion llms via temporal-spatial parallel decoding and confidence extrapolation. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2605.30753. 35

[31] Anji Liu, Oliver Broadrick, Mathias Niepert, and Guy Van den Broeck. Discrete copula diffusion. *arXiv preprint arXiv:2410.01949*, 2024. 1

[32] Zhiyuan Liu, Yicun Yang, Yaojie Zhang, Junjie Chen, Chang Zou, Qingyuan Wei, Shaobo Wang, and Linfeng Zhang. dllm-cache: Accelerating diffusion large language models with adaptive caching, 2025. URL https://arxiv.org/abs/2506.06295. 10, 31, 36

[33] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *ICML*, Proceedings of Machine Learning Research, pages 32819–32848. PMLR / OpenReview.net, 2024. 2, 10

[34] Lizhuo Luo, Shenggui Li, Yonggang Wen, and Tianwei Zhang. Dsb: Dynamic sliding block scheduling for diffusion llms. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2602.05992. 10, 31, 32, 36

[35] Omer Luxembourg, Haim Permuter, and Eliya Nachmani. Plan for speed: Dilated scheduling for masked diffusion language models. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2506.19037. 10, 35

[36] 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. 3, 10

[37] Daisuke Oba, Danushka Bollegala, Masahiro Kaneko, and Naoaki Okazaki. Stopping computation for converged tokens in masked diffusion-lm decoding. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2602.06412. 36

[38] Yu-Yang Qian, Junda Su, Lanxiang Hu, Peiyuan Zhang, Zhijie Deng, Peng Zhao, and Hao Zhang. d3llm: Ultra-fast diffusion llm using pseudo-trajectory distillation. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2601.07568. 10, 34, 36

[39] Houxing Ren, Mingjie Zhan, Zimu Lu, Ke Wang, Yunqiao Yang, Haotian Hou, Junting Pan, and Hongsheng Li. Edit-based refinement for parallel masked diffusion language models. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2605.09603. 34

[40] Subham S Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024. 1, 2, 3, 10

[41] Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37:103131–103167, 2024. 1, 3, 10

[42] Juntong Shi, Brian L. Trippe, Jure Leskovec, Stefano Ermon, and Minkai Xu. Diffusion language model parallel decoding via product-of-experts bridge. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2606.08048. 35

[43] Qingyan Wei, Yaojie Zhang, Zhiyuan Liu, Puyu Zeng, Yuxuan Wang, Biqing Qi, Dongrui Liu, and Linfeng Zhang. Accelerating diffusion large language models with slowfast sampling: The three golden principles. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2506.10848. 10, 35

[44] Chengyue Wu, Hao Zhang, Shuchen Xue, Shizhe Diao, Yonggan Fu, Zhijian Liu, Pavlo Molchanov, Ping Luo, Song Han, and Enze Xie. Fast-dllm v2: Efficient block-diffusion llm. *arXiv preprint arXiv:2509.26328*, 2025. 9, 10, 30, 36

[45] Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=3Z3Is6hnOT. 1, 3, 10, 35

[46] Tianyi Wu, Xiaoxi Sun, Yanhua Jiao, Yulin Li, Yixin Chen, YunHao Cao, YiQi Hu, and Zhuotao Tian. Dynamic-dllm: Dynamic cache-budget and adaptive parallel decoding for training-free acceleration of diffusion llm. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2606.26120. 10, 31, 32, 35

[47] Zhongyu Xiao, Zhiwei Hao, Jianyuan Guo, Yong Luo, Jia Liu, Jie Xu, and Han Hu. Streaming-dllm: Accelerating diffusion llms via suffix pruning and dynamic decoding. *arXiv preprint arXiv:2601.17917*, 2026. 10, 31, 32, 36

[48] Kaisen Yang, Jayden Teoh, Kaicheng Yang, Yitong Zhang, and Alex Lamb. Improving sampling for masked diffusion models via information gain. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2602.18176. 33

[49] Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025. 3, 10

[50] Zebin You, Shen Nie, Xiaolu Zhang, Jun Hu, Jun Zhou, Zhiwu Lu, Ji-Rong Wen, and Chongxuan Li. Llada-v: Large language diffusion models with visual instruction tuning, 2025. URL https://arxiv.org/abs/2505.16933. 24

[51] Runpeng Yu, Xinyin Ma, and Xinchao Wang. Dimple: Discrete diffusion multimodal large language model with parallel decoding. *arXiv preprint arXiv:2505.16990*, 2025. 3

[52] Yu Zhang, Xinchen Li, Jialei Zhou, Hongnan Ma, Zhongwei Wan, Yiwei Shi, Duoqian Miao, Qi Zhang, and Longbing Cao. Swordsman: Entropy-driven adaptive block partition for efficient diffusion language models. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2602.04399. 10, 31, 33, 36

[53] Yangyang Zhong, Yanmei Gu, Zhengqing Zang, Xiaomeng Li, Yuqi Ding, Xibei Jia, Yuting Shen, Zhenzhong Lan, Liwang Zhu, Weiping Liu, et al. Parallelism and generation order in masked diffusion language models: Limits today, potential tomorrow. *arXiv preprint arXiv:2601.15593*, 2026. 1

[54] Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*, 2025. 9, 29

A Full Proofs

Notation. Throughout this appendix, E denotes the evidence event (i.e., the conditioning event representing all observed tokens and the current diffusion state). For a candidate set $S = \{i_1, \dots, i_n\}$ of masked positions, we define the events $A_j := \{X_{i_j} = x^\star_{i_j}\}$, where $x^\star_{i_j}$ is the marginal greedy token at position i_j. The marginal confidence at position i_j is $c_j := \Pr(A_j \mid E) = \Pr(X_{i_j} = x^\star_{i_j} \mid E)$. We write $c_{(1)} \geq c_{(2)} \geq \cdots \geq c_{(n)}$ for the sorted (order-statistic) confidences. All probabilities are conditional on E unless stated otherwise; we sometimes suppress the conditioning for brevity.

Lemma 9 (Fréchet–Bonferroni lower bound)**.** *Let $A_1, \dots, A_n$ be events in a probability space $(\Omega, \mathcal{F}, \Pr)$. Then*

$$\Pr\left(\bigcap_{j=1}^{n} A_j\right) \geq \max\left\{0,\ \sum_{j=1}^{n} \Pr(A_j) - (n-1)\right\}. \tag{12}$$

Proof. We prove this from first principles using the complement and a union bound.

Step 1: Express the intersection via complements. By De Morgan's law,

$$\Pr\left(\bigcap_{j=1}^{n} A_j\right) = 1 - \Pr\left(\bigcup_{j=1}^{n} A_j^c\right). \tag{13}$$

Step 2: Apply the union bound (Boole's inequality). The union bound states that for any events $B_1, \dots, B_n$,

$$\Pr\left(\bigcup_{j=1}^{n} B_j\right) \leq \sum_{j=1}^{n} \Pr(B_j). \tag{14}$$

Applying this to $B_j = A_j^c$:

$$\Pr\left(\bigcup_{j=1}^{n} A_j^c\right) \leq \sum_{j=1}^{n} \Pr(A_j^c) = \sum_{j=1}^{n} \big(1 - \Pr(A_j)\big) = n - \sum_{j=1}^{n} \Pr(A_j). \tag{15}$$

Step 3: Combine. Substituting Eq. 15 into Eq. 13:

$$\begin{aligned}
\Pr\left(\bigcap_{j=1}^{n} A_j\right) &= 1 - \Pr\left(\bigcup_{j=1}^{n} A_j^c\right) \\
&\geq 1 - \left(n - \sum_{j=1}^{n} \Pr(A_j)\right) \\
&= \sum_{j=1}^{n} \Pr(A_j) - (n-1).
\end{aligned} \tag{16}$$

Step 4: Non-negativity. Since probabilities are non-negative, $\Pr(\bigcap_{j=1}^{n} A_j) \geq 0$ always holds. Combining with Eq. 16:

$$\Pr\left(\bigcap_{j=1}^{n} A_j\right) \geq \max\left\{0,\ \sum_{j=1}^{n} \Pr(A_j) - (n-1)\right\}. \tag{17}$$

This completes the proof. □

Citation note. Lemma 9 is the event-probability form of the classical Fréchet–Hoeffding / Fréchet–Bonferroni lower bound. It also follows directly from Boole's union bound applied to the complement

events. The bound is sharp as a distribution-free inequality over events with fixed marginal probabilities: when the positive branch is active, equality is obtained when the complement events are disjoint up to null sets; when the positive branch is inactive, the intersection can be made empty. In copula terminology, the bivariate lower bound is $W(u, v) = \max\{u + v - 1, 0\}$, while the comonotonic copula gives the corresponding upper bound $\min\{u, v\}$.

A.1 Proof of Theorem 1

Proof. We prove that if $L_n > U_n$, then the tuple of marginal greedy tokens $x_S^\star$ is the unique maximizer of the true joint conditional $P_S(\cdot \mid E)$.

Recall the definitions: $A_j := \{X_{i_j} = x_{i_j}^\star\}$ with $\Pr(A_j \mid E) = c_{(j)}$ (after sorting), and

$$L_n = \max\left\{0,\ \sum_{j=1}^{n} c_{(j)} - (n-1)\right\}, \qquad U_n = 1 - c_{(n)}. \tag{18}$$

Step 1: Lower bound on the correct tuple. The event $\{X_S = x_S^\star\}$ is exactly the intersection $\bigcap_{j=1}^n A_j$. Applying Lemma 9 to the conditional probability measure $\Pr(\cdot \mid E)$:

$$\begin{aligned} P_S(x_S^\star \mid E) &= \Pr\left(\bigcap_{j=1}^{n} A_j \;\middle|\; E\right) \\ &\geq \max\left\{0,\ \sum_{j=1}^{n} \Pr(A_j \mid E) - (n-1)\right\} \\ &= \max\left\{0,\ \sum_{j=1}^{n} c_{(j)} - (n-1)\right\} = L_n. \end{aligned} \tag{19}$$

Since we assume $L_n > U_n \geq 0$, we have $L_n > 0$, so the max is achieved by the sum expression.

Step 2: Upper bound on any competitor. Fix any competing tuple $z \in \mathcal{V}^n$ with $z \neq x_S^\star$. Since $z \neq x_S^\star$, there exists at least one coordinate $k \in \{1, \ldots, n\}$ such that $z_k \neq x_{i_k}^\star$. Now observe that:

$$\begin{aligned} P_S(z \mid E) &= \Pr(X_{i_1} = z_1, \ldots, X_{i_n} = z_n \mid E) \\ &\leq \Pr(X_{i_k} = z_k \mid E) && \text{(marginalizing out other coordinates)} \\ &\leq 1 - \Pr(X_{i_k} = x_{i_k}^\star \mid E) && \text{(since } z_k \neq x_{i_k}^\star \text{ and } x_{i_k}^\star \text{ is the greedy argmax)} \\ &= 1 - c_k \\ &\leq 1 - c_{(n)} = U_n. \end{aligned} \tag{20}$$

The second inequality uses the fact that $x_{i_k}^\star = \arg\max_v \Pr(X_{i_k} = v \mid E)$, so for any $z_k \neq x_{i_k}^\star$, we have $\Pr(X_{i_k} = z_k \mid E) \leq 1 - c_k$ (since the greedy token takes probability c_k and the remaining probability $1 - c_k$ is shared among all other tokens). The last inequality uses $c_k \geq c_{(n)}$ (since $c_{(n)}$ is the minimum confidence).

Step 3: Conclude uniqueness. Combining Steps 1 and 2: under the assumption $L_n > U_n$,

$$P_S(x_S^\star \mid E) \geq L_n > U_n \geq P_S(z \mid E) \tag{21}$$

for every $z \neq x_S^\star$. Therefore $x_S^\star$ is the unique maximizer of $P_S(\cdot \mid E)$ over $\mathcal{V}^n$. □

A.2 Proof of Corollary 2

Proof. Assume equal confidences: $c_{(1)} = c_{(2)} = \cdots = c_{(n)} = c$ for some $c \in (0, 1]$.

Step 1: Compute L_n and U_n. Under equal confidences:

$$L_n = \max\left\{0,\ \sum_{j=1}^{n} c - (n-1)\right\} = \max\{0,\ nc - (n-1)\} = \max\{0,\ 1 - n(1-c)\}, \tag{22}$$

$$U_n = 1 - c. \tag{23}$$

Step 2: Compute the Fréchet margin G_n. The Fréchet margin is defined as $G_n := L_n - U_n$. Assuming $L_n > 0$ (which is necessary for the criterion to be active):

$$\begin{aligned} G_n &= \big[nc - (n-1)\big] - (1-c) \\ &= nc - n + 1 - 1 + c \\ &= (n+1)c - n \\ &= 1 - (n+1)(1-c). \end{aligned} \tag{24}$$

Step 3: Derive the acceptance condition. The Fréchet criterion requires $G_n > \delta$:

$$\begin{aligned} 1 - (n+1)(1-c) &> \delta \\ (n+1)(1-c) &< 1 - \delta \\ 1 - c &< \frac{1-\delta}{n+1} \\ c &> 1 - \frac{1-\delta}{n+1}. \end{aligned} \tag{25}$$

Step 4: Compare with factor decoding. Fast-dLLM factor decoding with parameter f accepts a prefix of size n when $(n+1)(1-c_{(n)}) < f$. Under equal confidences $c_{(n)} = c$, this becomes:

$$\begin{aligned} (n+1)(1-c) &< f \\ c &> 1 - \frac{f}{n+1}. \end{aligned} \tag{26}$$

Comparing Eq. 25 and Eq. 26: setting $f = 1 - \delta$ makes the two conditions identical. This establishes the exact equivalence Fréchet$(\delta) \equiv$ Factor$(f = 1 - \delta)$ in the equal-confidence case. □

A.3 Proof of Proposition 3

Proof. We decompose the Fréchet score $G_n = L_n - U_n$ into a factor core plus a heterogeneity bonus.

Step 1: Expand G_n. Assuming $L_n > 0$ (stated as a hypothesis of the theorem):

$$\begin{aligned} G_n &= L_n - U_n \\ &= \left[\sum_{j=1}^{n} c_{(j)} - (n-1)\right] - (1 - c_{(n)}) \\ &= \sum_{j=1}^{n} c_{(j)} - n + 1 - 1 + c_{(n)} \\ &= \sum_{j=1}^{n} c_{(j)} + c_{(n)} - n. \end{aligned} \tag{27}$$

Step 2: Isolate the factor core. Write the sum as $\sum_{j=1}^{n} c_{(j)} = \sum_{j=1}^{n-1} c_{(j)} + c_{(n)}$. Substituting into Eq. 27:

$$\begin{aligned} G_n &= \sum_{j=1}^{n-1} c_{(j)} + c_{(n)} + c_{(n)} - n \\ &= \sum_{j=1}^{n-1} c_{(j)} + 2c_{(n)} - n. \end{aligned} \tag{28}$$

Now add and subtract $(n-1)c_{(n)}$:

$$\begin{aligned} G_n &= \sum_{j=1}^{n-1} c_{(j)} - (n-1)c_{(n)} + (n-1)c_{(n)} + 2c_{(n)} - n \\ &= \sum_{j=1}^{n-1} \big(c_{(j)} - c_{(n)}\big) + (n+1)c_{(n)} - n. \end{aligned} \tag{29}$$

Step 3: Identify the two terms. Define:

$$F_n := (n+1)\, c_{(n)} - n, \tag{30}$$

$$B_n := \sum_{j=1}^{n-1} \big(c_{(j)} - c_{(n)}\big). \tag{31}$$

Then $G_n = F_n + B_n$ by Eq. 29.

Step 4: Non-negativity of B_n. Since the confidences are sorted in non-increasing order, $c_{(j)} \geq c_{(n)}$ for all $j \in \{1, \ldots, n-1\}$. Therefore each term $c_{(j)} - c_{(n)} \geq 0$, and the sum $B_n \geq 0$.

Step 5: Characterize equality. $B_n = 0$ if and only if $c_{(j)} - c_{(n)} = 0$ for all $j = 1, \ldots, n-1$, which holds if and only if $c_{(1)} = c_{(2)} = \cdots = c_{(n)}$, i.e., the confidence profile is flat (homogeneous). □

A.4 Proof of Corollary 4

Proof. We show two claims: (i) factor acceptance implies Fréchet acceptance, and (ii) there exist configurations where Fréchet accepts but factor does not.

Step 1: Factor acceptance implies Fréchet acceptance. Factor decoding with matched parameter $f = 1 - \delta$ accepts a prefix of size n when

$$(n+1)(1 - c_{(n)}) < f = 1 - \delta, \tag{32}$$

which rearranges to $(n+1)c_{(n)} - n > \delta$, i.e., $F_n > \delta$.

By Proposition 3, $G_n = F_n + B_n$ with $B_n \geq 0$. Therefore:

$$F_n > \delta \implies G_n = F_n + B_n \geq F_n > \delta. \tag{33}$$

So Fréchet also accepts the prefix (since the Fréchet criterion is $G_n > \delta$).

Step 2: Strict separation. Fréchet strictly accepts a prefix that factor rejects when:

$$F_n \leq \delta \quad \text{(factor rejects)} \qquad \text{and} \qquad G_n = F_n + B_n > \delta \quad \text{(Fréchet accepts)}. \tag{34}$$

These two conditions hold simultaneously if and only if $F_n \leq \delta < F_n + B_n$. This requires $B_n > 0$, which by Proposition 3 holds whenever the confidence profile is not flat. In this regime, the heterogeneity bonus B_n provides exactly the additional margin needed for Fréchet acceptance. □

A.5 Proof of Lemma 7

Proof. We show that if the true joint P_S is close to the product-of-marginals Q_S in total variation distance, then the mode of Q_S is preserved as the mode of P_S.

Let $Q_S(z \mid E) = \prod_{i \in S} p_i(z_i \mid E)$ be the product-of-marginals approximation, and let $\Delta_Q := Q_S(x_S^\star \mid E) - \max_{z \neq x_S^\star} Q_S(z \mid E) > 0$ be the mode gap of Q_S.

Step 1: Recall the definition of total variation distance. For discrete distributions P and Q on a finite set $\mathcal{X}$:

$$d_{\mathrm{TV}}(P, Q) = \frac{1}{2} \sum_{x \in \mathcal{X}} |P(x) - Q(x)| = \max_{A \subseteq \mathcal{X}} |P(A) - Q(A)|. \tag{35}$$

In particular, for any single atom z: $|P_S(z \mid E) - Q_S(z \mid E)| \leq d_{\mathrm{TV}}(P_S, Q_S)$.

Step 2: Lower bound on $P_S(x_S^\star \mid E)$. Applying the pointwise bound to the mode:

$$\begin{aligned} P_S(x_S^\star \mid E) &\geq Q_S(x_S^\star \mid E) - |P_S(x_S^\star \mid E) - Q_S(x_S^\star \mid E)| \\ &\geq Q_S(x_S^\star \mid E) - d_{\mathrm{TV}}(P_S, Q_S). \end{aligned} \tag{36}$$

Step 3: Upper bound on $P_S(z \mid E)$ for any competitor $z \neq x_S^\star$. For any $z \neq x_S^\star$:

$$\begin{aligned} P_S(z \mid E) &\leq Q_S(z \mid E) + |P_S(z \mid E) - Q_S(z \mid E)| \\ &\leq Q_S(z \mid E) + d_{\mathrm{TV}}(P_S, Q_S) \\ &\leq \big[Q_S(x_S^\star \mid E) - \Delta_Q\big] + d_{\mathrm{TV}}(P_S, Q_S), \end{aligned} \tag{37}$$

where the last inequality uses the definition of Δ_Q: $Q_S(z \mid E) \le Q_S(x_S^\star \mid E) - \Delta_Q$ for all $z \neq x_S^\star$.

Step 4: Combine and conclude. Subtracting Eq. 37 from Eq. 36:

$$\begin{aligned} P_S(x_S^\star \mid E) - P_S(z \mid E) &\ge \big[Q_S(x_S^\star \mid E) - d_{\mathrm{TV}}\big] - \big[Q_S(x_S^\star \mid E) - \Delta_Q + d_{\mathrm{TV}}\big] \\ &= \Delta_Q - 2\, d_{\mathrm{TV}}(P_S, Q_S). \end{aligned} \tag{38}$$

If $d_{\mathrm{TV}}(P_S, Q_S) < \Delta_Q/2$, then $\Delta_Q - 2\, d_{\mathrm{TV}} > 0$, so $P_S(x_S^\star \mid E) > P_S(z \mid E)$ for all $z \neq x_S^\star$. Hence $x_S^\star$ is the unique maximizer of $P_S(\cdot \mid E)$. □

A.6 Proof of Corollary 8

Proof. We reduce the KL condition to the TV condition of Lemma 7 via Pinsker's inequality.

Step 1: State Pinsker's inequality. For any two distributions P and Q on the same measurable space:

$$d_{\mathrm{TV}}(P, Q) \le \sqrt{\frac{1}{2}\, D_{\mathrm{KL}}(P \| Q)}. \tag{39}$$

Step 2: Apply to our setting. Suppose $D_{\mathrm{KL}}(P_S \| Q_S) < \Delta_Q^2/2$. Then by Pinsker's inequality:

$$\begin{aligned} d_{\mathrm{TV}}(P_S, Q_S) &\le \sqrt{\frac{1}{2}\, D_{\mathrm{KL}}(P_S \| Q_S)} \\ &< \sqrt{\frac{1}{2} \cdot \frac{\Delta_Q^2}{2}} \\ &= \sqrt{\frac{\Delta_Q^2}{4}} = \frac{\Delta_Q}{2}. \end{aligned} \tag{40}$$

Step 3: Invoke Lemma 7. Since $d_{\mathrm{TV}}(P_S, Q_S) < \Delta_Q/2$, the hypothesis of Lemma 7 is satisfied. Therefore $x_S^\star$ is the unique maximizer of $P_S(\cdot \mid E)$. □

B Proofs for Certified Frontier

B.1 Proof of Proposition 10

Proof. We extend the Fréchet greedy equivalence (Theorem 1) to the calibration-robust setting where true confidences c_i are bounded below by $\underline{c}_i := \max\{0, \hat{c}_i - \eta_i\}$.

Step 1: Robust lower bound on the correct tuple. Define $A_i := \{X_i = x_i^\star\}$ for each $i \in S$. By hypothesis, $c_i = \Pr(A_i \mid E) \ge \underline{c}_i$ for all $i \in S$. Applying Lemma 9 to the conditional measure $\Pr(\cdot \mid E)$:

$$\begin{aligned} \Pr(X_S = x_S^\star \mid E) &= \Pr\left(\bigcap_{i \in S} A_i \;\middle|\; E \right) \\ &\ge \max\left\{ 0,\ \sum_{i \in S} \Pr(A_i \mid E) - (|S| - 1) \right\} \\ &= \max\left\{ 0,\ \sum_{i \in S} c_i - (|S| - 1) \right\}. \end{aligned} \tag{41}$$

Since $c_i \ge \underline{c}_i$ for all $i \in S$, and the function $\max\{0, \sum_i x_i - (|S| - 1)\}$ is non-decreasing in each x_i:

$$\Pr(X_S = x_S^\star \mid E) \ge \max\left\{ 0,\ \sum_{i \in S} \underline{c}_i - (|S| - 1) \right\} = L_\eta(S). \tag{42}$$

Step 2: Robust upper bound on any competitor. Fix any $z \neq x_S^\star$. There exists $k \in S$ with $z_k \neq x_k^\star$. Then:

$$\begin{aligned}
\Pr(X_S = z \mid E) &\leq \Pr(X_k = z_k \mid E) && \text{(marginalization)} \\
&\leq 1 - \Pr(X_k = x_k^\star \mid E) && (x_k^\star \text{ is the greedy argmax}) \\
&= 1 - c_k \\
&\leq 1 - \underline{c}_k && (\text{since } c_k \geq \underline{c}_k) \\
&\leq 1 - \min_{i \in S} \underline{c}_i = U_\eta(S). && (43)
\end{aligned}$$

Step 3: Conclude. If $L_\eta(S) > U_\eta(S)$, then combining Eqs. 42 and 43:

$$\Pr(X_S = x_S^\star \mid E) \geq L_\eta(S) > U_\eta(S) \geq \Pr(X_S = z \mid E) \tag{44}$$

for every $z \neq x_S^\star$. Therefore $x_S^\star$ is the unique maximizer of $P_S(\cdot \mid E)$. □

B.2 Proof of Corollary 11

Proof. We derive the calibration penalty under uniform η and the no-clipping assumption.

Step 1: Compute the robust lower bound. Under uniform calibration error $\eta_i = \eta$ for all i, the robust confidences are $\underline{c}_i = \max\{0, \hat{c}_i - \eta\}$. The no-clipping assumption states $\hat{c}_{(j)} \geq \eta$ for all $j \leq n$, so $\underline{c}_{(j)} = \hat{c}_{(j)} - \eta$ for all selected positions. Therefore:

$$\begin{aligned}
L_\eta &= \max\left\{0,\ \sum_{j=1}^{n} \underline{c}_{(j)} - (n-1)\right\} \\
&= \max\left\{0,\ \sum_{j=1}^{n} (\hat{c}_{(j)} - \eta) - (n-1)\right\} \\
&= \max\left\{0,\ \sum_{j=1}^{n} \hat{c}_{(j)} - n\eta - (n-1)\right\}. && (45)
\end{aligned}$$

Assuming the robust lower bound is active (i.e., the expression inside the max is positive):

$$L_\eta = \sum_{j=1}^{n} \hat{c}_{(j)} - n\eta - (n-1). \tag{46}$$

Step 2: Compute the robust upper bound.

$$\begin{aligned}
U_\eta &= 1 - \min_{j \leq n} \underline{c}_{(j)} = 1 - \underline{c}_{(n)} \\
&= 1 - (\hat{c}_{(n)} - \eta) = 1 - \hat{c}_{(n)} + \eta. && (47)
\end{aligned}$$

Step 3: Compute the robust Fréchet score.

$$\begin{aligned}
G_n^{(\eta)} &= L_\eta - U_\eta \\
&= \left[\sum_{j=1}^{n} \hat{c}_{(j)} - n\eta - (n-1)\right] - \left[1 - \hat{c}_{(n)} + \eta\right] \\
&= \sum_{j=1}^{n} \hat{c}_{(j)} + \hat{c}_{(n)} - n - n\eta - \eta \\
&= \left[\sum_{j=1}^{n} \hat{c}_{(j)} + \hat{c}_{(n)} - n\right] - (n+1)\eta. && (48)
\end{aligned}$$

Step 4: Identify the uncorrected score. The uncorrected (naive) Fréchet score is obtained by setting $\eta = 0$:

$$\hat{G}_n = \sum_{j=1}^{n} \hat{c}_{(j)} + \hat{c}_{(n)} - n. \tag{49}$$

Therefore:

$$G_n^{(\eta)} = \hat{G}_n - (n+1)\eta. \tag{50}$$

The calibration penalty is $(n+1)\eta$, which grows linearly with both the commit set size n and the calibration error η. □

C Calibration-Robust Fréchet Certificates

C.1 Reported Confidence versus Actual Correctness

Reported confidence versus actual correctness. In Section 4, we used c_i for the ideal marginal confidence entering the theoretical certificate. In an implementation, however, the decoder observes a model-reported confidence

$$\hat{c}_i := \max_{v \in \mathcal{V}} p_\theta(X_i = v \mid E).$$

This reported value may be overconfident relative to actual correctness. To discuss calibration, let

$$r_i := \Pr(A_i \mid E), \qquad A_i = \{X_i = x_i^\star\},$$

denote the actual probability that the proposed token is correct under the evaluation distribution or a chosen verifier reference. We assume a one-sided calibration envelope

$$r_i \geq \underline{c}_i := \max\{0, \hat{c}_i - \eta_i\},$$

where $\eta_i \geq 0$ is a calibration allowance. Thus $\underline{c}_i$ is a conservative lower confidence: we do not trust the raw model confidence $\hat{c}_i$ fully, but we assume that subtracting η_i makes it safe.

Define the robust Fréchet lower bound and competitor upper bound:

$$L_\eta(S) := \max\left\{0, \sum_{i \in S} \underline{c}_i - (|S| - 1)\right\}, \qquad U_\eta(S) := 1 - \min_{i \in S} \underline{c}_i. \tag{51}$$

Proposition 10 (Calibration-robust Fréchet certificate)**.** *If $r_i \geq \underline{c}_i$ for all $i \in S$ and $L_\eta(S) > U_\eta(S)$, then $x_S^\star$ is the unique maximizer of $P_S(\cdot \mid E)$.*

Corollary 11 (Calibration penalty)**.** *Under uniform calibration error η, assuming no selected robust confidence is clipped at zero (i.e., $\hat{c}_{(j)} \geq \eta$ for all $j \leq n$) and the robust lower bound is active, the robust Fréchet score satisfies*

$$G_n^{(\eta)} = \hat{G}_n - (n+1)\eta, \tag{52}$$

where $\hat{G}_n$ is the uncorrected score. A calibration error of η costs $(n+1)\eta$ in Fréchet margin.

Eq. 52 explains why aggressive Fréchet decoding can fail under overconfidence: the penalty scales linearly with the commit set size. This motivates either calibrating the model or using the robust selector with an explicit η budget.

D Experimental Evaluation Results on Dream

Table 2 evaluates Fréchet profile decoding on Dream-v0-Base-7B under PrefixCache with block size 32. All runs use a single H100 GPU. Improvements are reported relative to threshold decoding.

Dream efficiency gains. On Dream-v0-Base-7B, Fréchet decoding achieves **1.28**× average throughput over threshold decoding and **21.1%** NFE reduction (averaged over the six settings where NFE is available), with a 1.45-point average accuracy change. Relative to the Dream full-step baseline, Fréchet achieves **2.80**× average throughput and **61.6%** NFE reduction. The largest gains occur on MATH and MBPP where the confidence profiles are most heterogeneous; on GSM8K the speedup is more modest ($\sim$1.25×) due to Dream's concentrated confidence distribution on this task.

Table 2: **Benchmark results on Dream-v0-Base-7B.** (PrefixCache, block size 32, H100). Throughput improvements are reported as speedup over threshold decoding. NFE reduction is relative to threshold. NA indicates runs where NFE logging was not available.

Dataset	Len.	Method	Acc. (%)	Tok/s ↑	NFE ↓	Tok/NFE
GSM8K (5-shot)	256	Threshold	73.7	51.1 (1.00×)	202,862	1.66
	256	Factor	72.0	63.1 (1.23×)	176,073 (↓13.2%)	1.89
	256	Fréchet	72.0	**63.8** (1.25×)	**171,925** (↓15.3%)	**1.96**
MATH (4-shot)	256	Threshold	34.0	72.6 (1.00×)	568,643	2.25
	256	Factor	33.2	92.4 (1.27×)	442,791 (↓22.1%)	2.80
	256	Fréchet	33.1	**97.0** (1.34×)	**433,971** (↓23.7%)	**2.95**
HumanEval (0-shot)	256	Threshold	37.8	53.7 (1.00×)	25,465	1.64
	256	Factor	36.0	68.3 (1.27×)	20,246 (↓20.5%)	2.00
	256	Fréchet	35.4	**70.6** (1.31×)	**20,156** (↓20.8%)	**2.14**
MBPP (3-shot)	256	Threshold	56.0	85.1 (1.00×)	47,070	2.72
	256	Factor	51.8	106.6 (1.25×)	38,043 (↓19.2%)	3.43
	256	Fréchet	53.2	**111.2** (1.31×)	**35,170** (↓25.3%)	**3.64**
GSM8K (5-shot)	512	Threshold	73.7	57.2	355,339	1.90
	512	Factor	73.1	70.5 (1.23×)	302,274 (↓14.9%)	2.23
	512	Fréchet	72.7	**70.7** (1.24×)	**293,029** (↓17.5%)	**2.30**
MATH (4-shot)	512	Threshold	35.3	99.1	776,016	3.30
	512	Factor	34.1	118.6 (1.20×)	623,643 (↓19.6%)	4.10
	512	Fréchet	33.5	**127.5** (1.29×)	**595,000** (↓23.3%)	**4.30**
HumanEval (0-shot)	512	Threshold	37.8	58.7 (1.00×)	44,862	1.87
	512	Factor	36.6	71.0 (1.21×)	37,740 (↓15.9%)	2.22
	512	Fréchet	37.2	**72.7** (1.24×)	**37,168** (↓17.2%)	**2.25**
MBPP (3-shot)	512	Threshold	53.2	117.6 (1.00×)	63,108	4.06
	512	Factor	52.8	141.5 (1.20×)	51,779 (↓18.0%)	4.94
	512	Fréchet	52.8	**153.2** (1.30×)	**47,709** (↓24.4%)	**5.37**

E Additional Experiments

E.1 Effect of prefill length and cache mode

We next evaluate whether Fréchet profile decoding remains effective in the long-generation regime studied by Fast-dLLM. Following the Table 4 setup of Fast-dLLM, we compare no cache, PrefixCache, and DualCache under both 5-shot and 8-shot prompting on GSM8K with generation length 1024. Unlike the original table, which compares cache variants under a single selector, our goal is to isolate the effect of the token-selection rule while holding the cache mode fixed.

Table 3 shows that Fréchet profile decoding consistently improves decoding efficiency across cache modes. For 5-shot prompting, Fréchet improves throughput over factor by 5.7% on average and improves Tok/NFE by 6.4%, while increasing average accuracy by 1.18 points. For 8-shot prompting, Fréchet improves throughput over factor by 4.1% and Tok/NFE by 5.6%, again with a positive average accuracy change of 0.89 points. Compared to threshold decoding, the throughput gains are larger: 28.2% on average for 5-shot and 24.4% for 8-shot.

These gains come from the selector rather than the cache mechanism: within each cache mode, all methods use the same attention reuse strategy. The consistent increase in Tok/NFE indicates that Fréchet commits more tokens per denoising model call, reducing the number of required function evaluations. This supports our theoretical interpretation that profile-aware decoding extracts additional safe parallelism beyond weakest-token factor decoding.

E.2 Impact of generation length

Table 4 shows how throughput scales with generation length (256, 512, 1024) under 8-shot GSM8K for LLaDA-8B. The original Fast-dLLM paper reports only threshold decoding in this setting; we

Table 3: Selector comparison under different cache modes for LLaDA-8B on GSM8K with generation length 1024. We report accuracy, throughput, and tokens per function evaluation (Tok/NFE) for 5-shot and 8-shot settings. Within each cache mode and shot setting, Fréchet achieves the highest throughput and Tok/NFE while maintaining comparable accuracy.

Cache	Selector	5-shot			8-shot		
		Acc.	Tok/s	Tok/NFE	Acc.	Tok/s	Tok/NFE
No cache	Threshold τ=0.9	**78.33**	25.56	2.33	79.84	19.99	2.32
	Factor f=1.0	76.67	32.37	2.92	79.65	25.01	2.87
	Fréchet δ=0.25	**78.33**	**35.08**	**3.16**	**80.33**	**27.11**	**3.09**
PrefixCache	Threshold τ=0.9	**78.33**	38.48	2.31	**77.33**	32.38	2.25
	Factor f=1.0	**78.33**	44.76	2.87	76.67	38.17	2.87
	Fréchet δ=0.25	78.21	**46.86**	**3.03**	**77.33**	**38.73**	**3.00**
DualCache	Threshold τ=0.9	76.67	37.66	2.17	77.33	34.56	2.21
	Factor f=1.0	76.33	45.52	2.76	76.67	39.84	2.76
	Fréchet δ=0.25	**78.33**	**47.34**	**2.91**	**78.00**	**40.77**	**2.89**

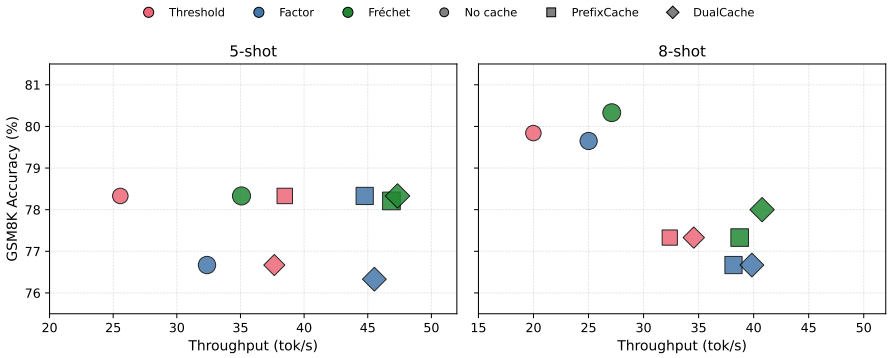

Figure 5: Accuracy–throughput trade-off across cache modes for 5-shot and 8-shot GSM8K at generation length 1024. Marker shape encodes cache mode; color encodes selector method; marker size is proportional to Tok/NFE. Fréchet profile decoding (green) consistently achieves the highest throughput and Tok/NFE in every cache mode while maintaining comparable or better accuracy.

extend the comparison to factor and Fréchet. Fréchet consistently achieves the highest throughput at every generation length and cache mode, with 5–20% speedup over factor and 20–38% over threshold. The NFE reduction is 6–8% vs. factor and 25–33% vs. threshold across all settings. The heterogeneity bonus depends on the block size (Figure 4, Appendix K), not on the generation length; the throughput gain over threshold grows with length because the cache cost is amortized over more steps.

F Results on Multimodal Diffusion Language models

We further evaluate Fast-dLLM++ on LLaDA-V to test whether profile-aware decoding transfers to multimodal diffusion language models. On MathVista, Fast-dLLM++ improves throughput from 28.2 to 32.9 tokens/s over Fast-dLLM, increasing speedup from 9.9× to 11.6× relative to full-step decoding, while slightly improving accuracy from 56.6% to 56.8%. This suggests that Fréchet profile decoding can exploit heterogeneous confidence profiles even in visual mathematical reasoning. On MathVerse, Fast-dLLM++ provides a smaller efficiency gain, improving throughput from 23.3 to 23.8 tokens/s, while accuracy decreases from 28.6% to 27.9%. These results indicate that the proposed selector generalizes to multimodal diffusion LLMs, but the magnitude of the gain is task-dependent. In particular, MathVista shows a clear accuracy-preserving efficiency improvement over Fast-dLLM, whereas MathVerse suggests that multimodal reasoning with stronger token or visual-text dependencies may require more conservative margins or calibration-aware selection.

Figure 6 studies the effect of the Fréchet margin δ on MathVista with LLaDA-V. The margin controls the aggressiveness of profile-aware commitment: smaller δ values allow larger parallel commit

Table 4: Impact of generation length on accuracy and throughput under 8-shot GSM8K (LLaDA-8B, block size 32). Fréchet achieves the highest throughput and lowest NFE at every generation length and cache mode. 512-length throughput is All runs use a single H100 GPU.

		PrefixCache			**DualCache**		
Len.	**Metric**	Threshold	Factor	Fréchet	Threshold	Factor	Fréchet
256	Acc. (%)	77.0	75.7	76.4	77.3	75.7	75.4
	Tok/s	69.8	90.8	**96.1**	56.4	75.4	**80.9**
	NFE	109,644	80,641	**74,289**	115,196	85,283	**78,901**
512	Acc. (%)	76.1	76.4	77.6	75.2	73.8	75.8
	Tok/s	37.8	43.9	**49.2**	40.1	45.9	**50.4**
	NFE	132,492	101,083	**93,936**	139,106	105,227	**102,145**
1024	Acc. (%)	77.3	76.7	77.3	77.3	76.7	78.0
	Tok/s	32.4	38.2	**38.7**	34.6	39.8	**40.8**
	NFE	31,918	25,547	**24,047**	33,308	26,609	**25,172**

Table 5: **Performance and Speedup Comparison of LLaDA-V on MathVista and MathVerse.** We compare full-step decoding, half-step decoding, Fast-dLLM, and our method. Throughput is reported with speedup relative to full-step decoding on the same benchmark.

Method	**MathVista**		**MathVerse**	
	Accuracy (%)	Throughput (Speedup)	Accuracy (%)	Throughput (Speedup)
Full Steps	59.2	2.84 (1×)	28.5	2.75 (1×)
Half Steps	59.7	5.56 (1.96×)	28.3	5.17 (1.88×)
Fast-dLLM	56.6	28.2 (9.9×)	28.6	23.3 (8.5×)
Ours	56.8	**32.9** (11.6×)	27.9	**23.8** (8.66×)

sets, while larger values enforce a stricter certificate. The results show a clear monotonic efficiency trend. As δ increases, speedup decreases and average NFE increases, confirming that larger margins make decoding more conservative. Nevertheless, all tested margins retain higher speedup and Tokens/NFE than the Fast-dLLM baseline, indicating that the Fréchet selector provides a robust efficiency advantage. Accuracy exhibits a moderate-margin optimum: aggressive settings such as $\delta = 0$ and $\delta = 0.05$ improve speed but can overcommit, whereas $\delta \approx 0.15 - 0.20$ offers the best balance, matching or slightly exceeding the baseline accuracy while preserving substantial speedup. This supports using a moderate margin as the default setting for multimodal mathematical reasoning.

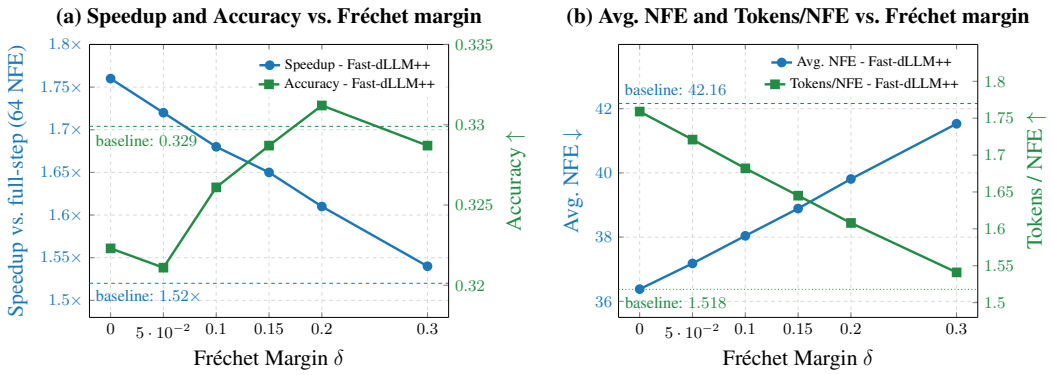

Figure 6: **Ablation study of Fréchet margin.** We evaluate Fast-dLLM++ on MathVista, a multimodal math reasoning using LLaDA-V [50].



User: Please carefully observe the image and come up with a brief caption.

Fast-dLLM (12.38s): Several large truck cabs are parked closely together on a grassy lot at an outdoor display. A bright red cab is on the left, a silver transport cab is on the right, and more trucks are visible in the background.

Fast-dLLM++ (Ours) (9.86s): Several large truck cabs are lined up on grass at an outdoor vehicle display. The scene features a bright red cab on the left, a silver transport cab on the right, and additional trucks parked behind them.

Figure 7: **Visual grounding example #1.** Fast-dLLM++ produces comparable caption quality while reducing decoding time from 12.38s to 9.86s.



User: Please carefully observe the image and come up with a detailed caption.

Fast-dLLM (18.12s): The image shows a baseball game from a high, overhead viewpoint focused on home plate. A right-handed batter in a white uniform with the number 12 is following through after swinging the bat. The ball can be seen rising toward the outfield grass, suggesting he has just hit it. The catcher, dressed in black protective equipment, is crouched behind the plate, and the umpire stands directly behind him. Bright chalk lines mark the batter's boxes and foul lines, contrasting with the dirt and green grass.

Fast-dLLM++ (Ours) (14.75s): A baseball player in a white pinstriped uniform, wearing number 12, has just completed a powerful swing at home plate. The ball is visible traveling upward over the dirt area near the edge of the grass. Behind the batter, the catcher in dark gear remains crouched, while the umpire stands closely behind him, watching the play. The image is taken from an elevated angle, clearly showing the batter's box, home plate, chalk lines, and surrounding infield dirt.

Figure 8: **Visual grounding example #2.** Fast-dLLM++ produces comparable caption quality while reducing decoding time from 18.12s to 14.75s.

G Qualitative Analysis of Text Generation Results

We provide qualitative examples of decoded text generated by `Threshold`, `Factor`, and `Fréchet` decoding. The goal is to isolate where the methods diverge textually, when those divergences affect answer correctness, and whether the reasoning template is preserved across decoding methods.

Summary. Across GSM8K, we observe that decoded outputs are textually different in most samples, but only a much smaller fraction differ in final correctness. In particular, 1204 out of 1319 GSM8K samples, or 91.3%, have textually different outputs across decoding methods, while only 171 out of 1319 samples, or 13.0%, differ in correctness. This indicates that the methods usually preserve the same high-level reasoning structure, while diverging on local numerical values, intermediate arithmetic, or final formatting.

The dominant divergence pattern is that the methods agree on the reasoning template, including sentence structure and step-by-step decomposition, but commit to different specific digits or intermediate values. This is consistent with the failure mode of parallel token decoding: tokens that should be conditionally dependent, such as digits within a number or related arithmetic quantities, may be decoded simultaneously.

Interpretation. The large gap between textual divergence and correctness divergence suggests that most method-level differences are not semantic failures. Instead, the methods often preserve the same problem-solving template while varying in surface form, formatting, or local token choices. Correctness changes are concentrated in low-confidence regions, especially numerical tokens in GSM8K and implementation details in MBPP.

Example ID: GSM8K, doc_id=20. **Question:** I have 10 liters of orange drink that are two-thirds water and I wish to add it to 15 liters of pineapple drink...

Threshold WRONG

```
...nge drink) + 14 liters (
 remaining pineapple drink) =
 24 liters
So, the amount of water in the
 remaining 24 liters is:
- 9 liters (remaining water) / 24
 liters (total mixture) = 3/8
The answer is 3/8
```

Factor WRONG

```
...Finally, let's calculate the
 amount of water in the
 remaining 24 liters:
- 15 liters of water / 24 liters =
 7.5 liters of water.
Therefore, there are 7.5 liters of
 water in the remaining 24
 liters. \\
```

Fréchet CORRECT

```
...iter (spilled) = 24 liters.
Finally, let's calculate the
 amount of water in the
 remaining 24 liters:
- 24 liters * 15 liters of water
So, the amount of water in the
 remaining 24 liters is 15
 liters.
```

Figure 9: **Representative GSM8K example #1.** Fréchet decoding produces the correct final answer while the alternative methods make arithmetic-level errors.

Example ID: GSM8K, doc_id=2. **Question:** Josh decides to try flipping a house. He buys a house for $80,000 and then puts in $50,000 in repairs...

Threshold CORRECT

```
...New value: \$80,000 * 2.5 = \
 $200,000
3. **Profit calculation:**
 - Profit: New value - Total
 cost
 - Profit: \$200,000 - \$130,000
 = \$70,000
Therefore, Josh made a profit of \
 $70,000.
```

Factor WRONG

```
...\$80,000 * 1.5 = \$220,000
3. **Profit calculation:**
 - Profit formula: New value -
 Total cost
 - Profit: \$220,000 - \$130,000
 = \$90,000
Therefore, Josh made a profit of \
 $90,000.
```

Fréchet WRONG

```
...New value: \$80,000 * 1.5 = \
 $120,000
3. **Profit calculation:**
 - Profit: New value - Total
 cost
 - Profit: \$120,000 - \$130,000
 = \$10,000
Therefore, Josh made a profit of \
 $10,000.
```

Figure 10: **Representative GSM8K example #2.** Threshold decoding preserves the correct numerical path while the other decoding methods commit to incorrect intermediate values.

H Matched Fréchet versus Factor Sweeps

Since Fréchet with margin δ is theoretically matched to factor with $f = 1-\delta$ (Corollary 2), comparing the two directly isolates the gain attributable to the heterogeneous confidence profile rather than to a different operating point. Table 1 compares against factor at its most aggressive operating point $f = 1.0$; here we extend the matched comparison to MATH, HumanEval, PrefixCache, DualCache, and Fast-dLLM-v2, the configurations not covered by Figure 2. F denotes Fréchet(δ) and C the matched factor ($f = 1 - \delta$); the speed improvement given by Fréchet is measured as the relative improvement in NFE and tok/s.

Across every additional configuration, matched Fréchet holds accuracy within sampling noise of factor($f = 1-\delta$) while cutting 9 to 20% of forward evaluations, i.e. a consistent 10 to 25% throughput gain. The effect persists across benchmarks (MATH, HumanEval), cache regimes (PrefixCache, DualCache), and backbones (LLaDA-8B, Fast-dLLM-v2 7B), confirming that the profile-aware bonus is not specific to the GSM8K setting in Figure 2. As expected, the margin δ trades efficiency for conservatism monotonically, so a smaller δ can be chosen where accuracy is robust and a larger δ where it is not. On MBPP (LLaDA-8B, PrefixCache, full test set), factor at $f = 1.0$ obtains 21.4 pass@1 with 26,870 NFE, matched factor $f = 0.75$ obtains 24.0 with 29,305 NFE, and Fréchet $\delta = 0.25$ obtains 25.3 with 26,169 NFE (Table 10).

How often the bonus changes a decision. Any refinement of a deterministic decision rule must change a decision at the old rule's boundary; what matters is how often that happens and what it compounds to. Table 7 gives how often Fréchet $\delta = 0.25$ commits more than matched factor $f = 0.75$ on the LLaDA-8B (256, PrefixCache) setting, and the extra tokens per such step. On roughly a third to a half of steps Fréchet commits more, and each such step removes future forward passes; these per-step savings accumulate over the full generation into the NFE reductions reported in Table 1.

Example ID: MBPP, doc_id=0. **Pattern:** All methods produce identical code. **Task:** Write a Python function to remove the first and last occurrence of a given character from a string.

Threshold CORRECT

```
def remove_Occ(s, char):
    # Find the first and last
     occurrence of the character
    first_index = s.find(char)
    last_index = s.rfind(char)

    # Remove the first and last
     occurrence
    if first_index != -1 and
     last_index != -1:
        return s[:first_index] + s
     [last_index + 1:]
```

Factor CORRECT

```
def remove_Occ(s, char):
    # Find the first and last
     occurrence of the character
    first_index = s.find(char)
    last_index = s.rfind(char)

    # Remove the first and last
     occurrence
    if first_index != -1 and
     last_index != -1:
        return s[:first_index] + s
     [last_index + 1:]
```

Fréchet CORRECT

```
def remove_Occ(s, char):
    # Find the first and last
     occurrence of the character
    first_index = s.find(char)
    last_index = s.rfind(char)

    # Remove the first and last
     occurrence
    if first_index != -1 and
     last_index != -1:
        return s[:first_index] + s
     [last_index + 1:]
```

Figure 11: MBPP example where all decoding methods produce identical code.

Table 6: Matched sweeps on LLaDA-8B (gen 256, block 32) and Fast-dLLM-v2 7B. Accuracy F/C in percent.

Setting	δ	acc F/C	NFE cut	tok/s gain
MATH, PrefixCache	0.00	27.6/32.7	18%	+22%
	0.10	32.4/32.7	17%	+20%
	0.20	32.5/31.2	15%	+17%
	0.25	32.5/32.5	14%	+17%
	0.30	31.9/32.8	14%	+16%
HumanEval, PrefixCache	0.00	39.6/40.7	20%	+25%
	0.10	41.5/40.7	17%	+21%
	0.20	41.5/41.5	14%	+16%
	0.25	40.9/40.1	13%	+14%
	0.30	43.9/40.8	16%	+19%
HumanEval, DualCache	0.00	32.9/33.5	19%	+23%
	0.10	35.4/34.2	16%	+19%
	0.20	33.5/34.8	14%	+16%
	0.25	36.6/34.8	14%	+16%
	0.30	36.6/34.8	14%	+16%
Fast-dLLM-v2, GSM8K, sub-block 4	0.00	37.4/39.4	13%	+16%
	0.10	38.2/38.0	12%	+14%
	0.20	39.6/38.8	11%	+12%
	0.25	38.0/39.2	10%	+11%
	0.30	38.6/39.0	9%	+10%

I Commit-Level Calibration and Robustness

Reference decode. We added measurement code to the decoder in the main setting (LLaDA-8B-Instruct, PrefixCache) so that every parallel commit is checked against a reference: at each step, the selected positions are also decoded one token at a time, most-confident first, from the same decoder state. A parallel commit is defined correct if it produces exactly the same tokens as this one-at-a-time decoding. For every multi-token commit we also record the three quantities the theory computes from the confidences alone: L_n, the certified lower bound on the probability that all n committed tokens are jointly correct; U_n, the upper bound on the probability that any competing alternative is correct; and their difference $G_n = L_n - U_n$, the certified safety margin of the commit. The rule commits when $G_n > \delta$.

We make three observations. (1) The certificate is sound in practice: the observed fraction of correct commits never falls below the floor L_n in any margin decile of any run (zero violations across all selectors and tasks). This is exactly the conservative guarantee the theory claims. (2) The margin is

Table 7: Steps on which Fréchet $\delta = 0.25$ commits more tokens than matched factor $f = 0.75$ (LLaDA-8B, gen 256, PrefixCache).

Task	Steps committing more than factor	Extra tokens per such step
GSM8K	41.7%	0.70
MATH	38.2%	0.63
HumanEval	34.9%	0.60
MBPP	31.7%	0.50

Table 8: Commit correctness and ranking quality of the margin (LLaDA-8B, PrefixCache). AUC measures how well G_n ranks commits from most to least likely to be correct.

Selector	GSM8K: commit correct / AUC(G_n)	MBPP: commit correct / AUC(G_n)
Factor $f = 1.0$	95.9% / .866	95.8% / .891
Factor $f = 0.75$	96.5% / .909	97.1% / .923
Fréchet $\delta = 0.25$	96.9% / .671	97.7% / .700

predictive: AUC measures how well G_n ranks commits from most to least likely to be correct (1.0 = perfect ranking, .5 = uninformative). Measured over factor's commits, which span the whole range of margins from risky to safe, G_n ranks correctness with AUC .87 to .92. Fréchet's own AUC is lower for a simple statistical reason, not a calibration failure: Fréchet only ever accepts commits whose margin already clears the δ bar, so its accepted commits all have similar (high) margins, leaving a ranking statistic almost no variation to work with. (3) Fréchet achieves the highest commit correctness of the three selectors while using the fewest model evaluations.

Calibration-robust variant. To evaluate miscalibration specifically, we run the robust variant of Appendix C ($\hat{c}_i \mapsto \max\{0, \hat{c}_i - \eta\}$, i.e. discount every reported confidence by η before applying the certificate) in the main setting. Commit error is disagreement with the one-at-a-time reference.

Table 9: Robust Fréchet ($\delta = 0.25$) under a uniform confidence discount η (LLaDA-8B, PrefixCache).

Task	η	Accuracy (%)	Commit error
GSM8K	.00	76.0	3.12%
GSM8K	.06	78.5	0.78%
GSM8K	.10	78.5	0.32%
MBPP	.00	19.0	2.30%
MBPP	.06	22.0	0.46%
MBPP	.10	21.0	0.20%

Commit error drops about 10x with accuracy flat to improved; the cost is smaller parallel commit sets (mean size 7.6 to 5.5 on MBPP at $\eta = .06$), exactly as Corollary 11 predicts ($(n+1)\eta$ margin shrinkage). Confidence miscalibration affects threshold and factor at least as strongly (factor $f = 1.0$ shows the highest commit error in Table 8); it is a property of the underlying model shared by all confidence-based selectors, and ours is the variant with an explicit robustness parameter and a quantified certificate.

MBPP: overconfidence or token coupling. On MBPP at generation length 256, factor drops sharply (27.4 to 21.4) while Fréchet partially recovers. We ran the diagnostic in the main setting (LLaDA-8B, MBPP 3-shot, PrefixCache). The drop is driven by factor at its most aggressive operating point ($f = 1.0$); matched factor ($f = 0.75$) and Fréchet recover most of it, and the calibration-robust variant recovers it fully. Table 10 reports full MBPP test-set pass@1; the factor $f = 1.0$ row reproduces the value in Table 1.

Two things are directly visible. First, even against the aggressive factor $f = 1.0$, Fréchet dominates on both axes (25.3 versus 21.4 pass@1, 26,169 versus 26,870 NFE), and also edges matched factor $f = 0.75$ (25.3 versus 24.0) at 11% fewer NFE. Second, robust confidence lifts pass@1 monotonically to 25.7 at $\eta = .06$ ($+4.3$ over factor $f = 1.0$) by committing more conservatively, at essentially threshold-level compute (34,694 versus 34,975 NFE). To separate the failure modes we measured two

Table 10: Full MBPP test set, LLaDA-8B, PrefixCache, gen 256.

Method	pass@1	total NFE	tok/s
Threshold $\tau = 0.9$	27.4	34,975	66.0
Factor $f = 1.0$	21.4	26,870	79.3
Matched factor $f = 0.75$	24.0	29,305	73.9
Fréchet $\delta = 0.25$	25.3	26,169	82.8
+ robust $\eta = 0.04$	25.5	30,820	72.7
+ robust $\eta = 0.06$	25.7	34,694	63.7
+ robust $\eta = 0.08$	25.7	38,709	56.1

quantities. Commit error counts commits that disagree with the one-at-a-time reference. The argmax-flip probe is a direct coupling signal: reveal a position's selected neighbors and recompute its argmax; a flip means its top token changed, i.e. dependence the marginal certificate cannot see. Commit error falls from 4.17% (factor $f = 1.0$) to 2.30% (Fréchet) to 0.46% (robust $\eta = .06$); MBPP's flip rate under Fréchet is 0.4%, no higher than GSM8K's 0.6% and below factor $f = 1.0$'s 1.0%. So the drop is dominated by recoverable overconfidence, which the robust variant corrects exactly as Corollary 11 predicts; residual coupling is small and not MBPP-specific, and this overconfidence hits threshold and factor at least as hard.

J Generality: Backbones, Open-Ended Generation, and Length

LLaDA-1.5. Table 11 reports GSM8K (block 32) on LLaDA-1.5 [54] across generation lengths and cache modes with the five selector settings of Table 1 and the matched factor. On the stronger LLaDA-1.5 backbone, the behavior at corresponding operating points is unchanged: Fréchet $\delta = 0.25$ matches factor $f = 0.75$ within run-to-run noise (accuracy deltas between -3.7 and $+1.3$ points across the six settings, and it wins outright in two) while using 10 to 18% fewer NFE and delivering 6 to 21% higher throughput; the most aggressive pair ($\delta = 0$ versus $f = 1.0$) cuts NFE by 12 to 21% at a 1 to 4 point cost. Threshold $\tau = 0.9$ remains the accuracy ceiling but is the slowest throughout (about $1.8\times$ slower than Fréchet $\delta = 0.25$ at 256/PrefixCache), and the NFE saving shrinks with generation length at fixed block size, the same trend as base LLaDA.

Table 11: LLaDA-1.5, GSM8K, block 32. Each cell: accuracy / tok/s / NFE per example.

Gen	Cache	Threshold τ=.9	Factor f=1.0	Fréchet δ=0	Factor f=.75	Fréchet δ=.25
256	PrefixCache	82.7 / 53.9 / 77.2	80.7 / 91.9 / 57.6	77.0 / 112.8 / 45.8	82.0 / 82.4 / 64.7	79.7 / 97.7 / 53.2
256	DualCache	80.0 / 40.7 / 83.9	79.0 / 73.8 / 61.7	77.3 / 90.7 / 48.8	79.0 / 66.2 / 69.4	80.3 / 80.0 / 57.2
512	PrefixCache	84.7 / 39.7 / 76.2	82.3 / 67.7 / 59.8	80.0 / 79.9 / 48.8	84.0 / 61.6 / 65.6	80.3 / 72.7 / 55.5
512	DualCache	81.3 / 35.5 / 83.9	80.7 / 56.8 / 64.8	82.0 / 67.9 / 52.6	82.0 / 52.9 / 71.8	81.3 / 61.8 / 61.7
1024	PrefixCache	82.3 / 19.5 / 81.8	81.7 / 28.3 / 68.3	79.0 / 30.7 / 60.4	82.3 / 26.7 / 72.7	81.0 / 28.9 / 65.5
1024	DualCache	79.7 / 19.8 / 85.6	80.3 / 27.3 / 70.3	79.0 / 29.4 / 62.1	81.3 / 26.6 / 75.8	80.7 / 28.2 / 68.0

Open-ended generation: IFEval. IFEval is an open-ended generation benchmark (verifiable instruction following, zero-shot, LLaDA-8B, PrefixCache); Table 12 reports the results. Threshold is slightly more accurate but the slowest. At the corresponding operating point ($\delta = 0.25$ versus $f = 0.75$), instruction-following accuracy is identical on both metrics while Fréchet uses 4.0% fewer NFE and is 6.5% faster; against threshold, Fréchet $\delta = 0.25$ gives up 0.7/0.5 points for 15.9% fewer NFE and 26.5% higher throughput. The efficiency gain transfers to open-ended generation with accuracy preserved.

Long generation. The gain also persists at long generation: GSM8K at lengths 1024 and 2048 shows 14.0% and 10.0% NFE reduction at $+0.7$ and -0.3 accuracy points (Fréchet $\delta = 0.25$ versus factor $f = 0.75$). The heterogeneity bonus depends on the block size rather than the generation length (Figure 4, Appendix K).

Table 12: IFEval, zero-shot, LLaDA-8B, PrefixCache.

Method	Prompt-level strict	Instruction-level strict	NFE/example	tok/s
Threshold $\tau = 0.9$	58.0	66.5	216.7	33.9
Factor $f = 1.0$	56.3	65.8	179.8	42.3
Fréchet $\delta = 0$	55.7	64.5	165.8	46.1
Factor $f = 0.75$	57.3	66.0	189.8	40.3
Fréchet $\delta = 0.25$	57.3	66.0	182.2	42.9

K Fast-dLLM-v2 Sub-Block Transfer

Fast-dLLM-v2 [44] adapts a pretrained autoregressive model into block diffusion and reuses confidence-aware parallel decoding for intra-sub-block refinement, so Fréchet profile decoding applies directly at the sub-block selection step. We inserted Fréchet only at Fast-dLLM-v2's intra-sub-block commit-selection step on the public 7B checkpoint, leaving the block-diffusion process unchanged, and ran the grid of sub-blocks $\{4, 8, 16, 32\}$ and $\delta \in \{0.10, 0.25, 0.30\}$. Table 13 gives the unmodified native v2 decoder (its default threshold-0.9 selector); Table 14 reports accuracy change in absolute points, NFE cut, and throughput gain relative to it.

Table 13: Fast-dLLM-v2 7B, native decoder (default threshold-0.9 selector): accuracy / NFE by sub-block size.

Task	Sub-block 4	Sub-block 8	Sub-block 16	Sub-block 32
GSM8K	39.0 / 13,090	38.8 / 11,079	39.2 / 9,675	37.8 / 8,479
MATH	22.6 / 109,473	20.8 / 97,443	22.0 / 86,960	22.8 / 78,604

Table 14: Fast-dLLM-v2 7B: Fréchet inserted at the intra-sub-block selection step, relative to the native decoder of Table 13.

Task	Sub-block	Acc change (abs pts)	NFE cut	tok/s gain
Fréchet $\delta = 0.25$ (default)				
GSM8K	4	-1.0	19.7%	$+24.5\%$
GSM8K	8	-3.0	23.5%	$+30.7\%$
GSM8K	16	-1.6	25.4%	$+34.1\%$
GSM8K	32	-1.2	26.0%	$+35.2\%$
MATH	4	-0.2	17.0%	$+20.5\%$
MATH	8	$+0.0$	20.9%	$+26.4\%$
MATH	16	-0.6	22.7%	$+29.3\%$
MATH	32	-3.8	23.7%	$+31.1\%$
Fréchet $\delta = 0.10$ (aggressive)				
GSM8K	4	-0.8	24.8%	$+32.9\%$
GSM8K	8	-2.8	28.0%	$+38.8\%$
GSM8K	16	-3.0	31.2%	$+45.2\%$
GSM8K	32	-2.2	31.9%	$+46.8\%$
MATH	4	-0.8	22.0%	$+28.2\%$
MATH	8	-0.8	25.9%	$+35.0\%$
MATH	16	-1.8	28.0%	$+38.9\%$
MATH	32	-3.2	29.5%	$+41.8\%$
Fréchet $\delta = 0.30$ (conservative)				
GSM8K	4	-0.4	17.9%	$+21.9\%$
GSM8K	8	-2.6	22.0%	$+28.2\%$
GSM8K	16	-1.0	24.3%	$+32.1\%$
GSM8K	32	-0.8	25.9%	$+35.0\%$
MATH	4	-0.6	15.6%	$+18.5\%$
MATH	8	-1.0	19.0%	$+23.5\%$
MATH	16	-0.8	21.1%	$+26.7\%$
MATH	32	-4.2	22.0%	$+28.2\%$

Even at the smallest sub-block (4) the bonus is meaningful: at the paper's default $\delta = 0.25$, Fréchet reduces NFE by 17 to 20% and improves throughput by 20 to 25% on GSM8K and MATH, with accuracy changes between -0.2 and -1.0 points. The NFE gain grows monotonically with sub-block size on every task (larger candidate sets expose more profile heterogeneity), and so does the quality risk under more aggressive settings. These results show that profile heterogeneity remains useful even in small block-diffusion candidate sets. Table 6 gives the matched comparison against factor $f = 1 - \delta$ at sub-block size 4. These runs change only the selector inside the block-diffusion decoder and keep the v2 cache configuration fixed.

L Compatibility with Concurrent Acceleration Methods

Table 15: **Fréchet applied on top of recent complementary accelerators** (LLaDA-8B-Instruct, GSM8K, one H100, unless noted). Each method's own pipeline, checkpoint, cache, and harness are unchanged; only its commit rule is replaced by Fréchet at the stated δ, with efficiency in the pipeline's own counter. Appendix L reports all margins and methods.

Method	Venue	Acc. (own rule)	Acc. (Fréchet, δ)	Efficiency change
KLASS [24] (same-setting selector)	NeurIPS 25	77.3	77.7 (.25)	NFE -52%, tok/s $2.0\times$
dLLM-Cache [32]	ICML 26	80.0	79.7 (.40)	NFE -65%, tok/s $2.7\times$
Streaming-dLLM [47]	arXiv 26	79.0	80.3 (.25)	NFE -25%, tok/s $+25\%$
Dynamic-dLLM [46]	ICLR 26	75.3	76.0 (.25)	NFE -56%, tok/s $2.5\times$
DSB [34]	ICML 26	81.3	81.0 (.30)	NFE -32%, tok/s $1.4\times$
DC-Leap [21]	ICML 26	77.0	78.7 (.10)	tok/s 95.9 $\rightarrow$ 97.6
Swordsman [52]	ICML 26	79.3	80.7 (.10)	wall time -16%
CCD [9] (Dream-7B)	ICML 26	49.0	49.0 (.25)	steps -54%, tok/s $2.1\times$
dParallel [12] (distilled)	ICLR 26	79.0	78.7 (.25)	NFE -15% vs matched factor

Protocol. Each method's own pipeline, checkpoint, cache setting, and evaluation harness are left unchanged; only its commit or acceptance rule is replaced, in place, by the Fréchet certificate. Unless noted: LLaDA-8B-Instruct, GSM8K 5-shot flexible-extract, one dedicated H100. Efficiency counters use each pipeline's own definition, identical in both arms of a block and never compared across blocks. Each block gives the baseline's own numbers, then Fréchet at several margins; NFE cut is relative to the baseline row of each block unless stated otherwise.

KLASS [24]: adaptive KL-guided selector, same backbone and cache. KLASS can be evaluated under the same setting (LLaDA-8B, PrefixCache, GSM8K). We ported KLASS's selector (confidence 0.6, KL 0.015, history 2) into the identical harness and validated the port against the official implementation. Every Fréchet margin dominates KLASS on the accuracy-throughput frontier; at $\delta = 0.25$ Fréchet obtains 77.7 accuracy and 102.4 tok/s, compared with 77.3 and 51.7 for KLASS, approximately twice the throughput and a 52% NFE reduction at comparable accuracy.

	acc	tok/s	NFE	tok/NFE
KLASS (0.6/0.015/2)	77.3	51.7	33,456	2.06
Fréchet $\delta = 0.0$	75.0	118.6	-59%	5.02
Fréchet $\delta = 0.1$	75.0	113.8	-56%	4.73
Fréchet $\delta = 0.2$	78.7	105.5	-53%	4.43
Fréchet $\delta = 0.25$	77.7	102.4	-52%	4.28
Fréchet $\delta = 0.3$	78.7	98.7	-50%	4.14

dLLM-Cache [32]: adaptive caching. The base row is the unmodified pipeline with its shipped decoding. Adding Fréchet multiplies dLLM-Cache's throughput by up to $3.2\times$; the conservative $\delta = 0.4$ setting retains nearly the baseline accuracy (79.7 versus 80.0) while reducing NFE by 65%.

	acc	tok/s	NFE	tok/NFE
dLLM-Cache, unmodified	80.0	27.5	38,400	1.82
Fréchet $\delta = 0.1$	73.0	88.3	-71%	6.36
Fréchet $\delta = 0.2$	75.0	82.9	-69%	5.99
Fréchet $\delta = 0.25$	76.0	79.9	-68%	5.75
Fréchet $\delta = 0.3$	78.3	78.8	-67%	5.59
Fréchet $\delta = 0.4$	79.7	73.5	-65%	5.26

Streaming-dLLM [47]: suffix pruning. The base is threshold 0.9 plus its dynamic pruning. At $\delta = 0.25$ Fréchet improves the unmodified pipeline on both axes at once, $+1.3$ accuracy points and $+25\%$ throughput; suffix pruning does not subsume the selector gain.

	acc	tok/s	NFE	tok/NFE
Streaming-dLLM, unmodified	79.0	76.2	16,190	2.66
Fréchet $\delta = 0.1$	77.3	103.6	-33%	4.00
Fréchet $\delta = 0.25$	80.3	95.0	-25%	3.59
Fréchet $\delta = 0.3$	79.7	92.9	-23%	3.47
Fréchet $\delta = 0.4$	77.7	88.4	-18%	3.23

Dynamic-dLLM [46]: dynamic cache budget and adaptive parallel decoding. Full native pipeline with the dynamic feature cache on; we replace only its prediction-dynamics commit rule. At $\delta = 0.25$ Fréchet improves the unmodified pipeline on both axes at once, $+0.7$ accuracy points at $2.5\times$ the throughput and $2.3\times$ fewer NFE; the dynamic feature cache does not subsume the selector gain.

	acc	tok/s	NFE	tok/NFE
Dynamic-dLLM, unmodified	75.3	21.7	42,955	1.79
Fréchet $\delta = 0.1$	72.0	58.4	-60%	4.48
Fréchet $\delta = 0.2$	74.0	54.5	-57%	4.19
Fréchet $\delta = 0.25$	76.0	53.5	-56%	4.07
Fréchet $\delta = 0.3$	73.3	50.4	-55%	3.94
Fréchet $\delta = 0.4$	73.3	48.8	-51%	3.64

DSB [34]: dynamic sliding-block scheduling. Dynamic sliding-block scheduling with its cache, threshold accept rule replaced in place (cache identical in both arms; NFE and tok/s from the repository's own counters). At $\delta = 0.3$ the certificate matches DSB's native accuracy within noise (81.0 versus 81.3) at 32% fewer NFE and $1.4\times$ tok/s; dynamic blocking and the certificate compose.

	acc	tok/s	NFE	tok/NFE
DSB, unmodified (thr .9)	81.3	67.1	22,794	3.00
Fréchet $\delta = 0.1$	79.0	112.8	-42%	5.22
Fréchet $\delta = 0.2$	79.7	105.3	-37%	4.80
Fréchet $\delta = 0.25$	78.3	103.9	-35%	4.63
Fréchet $\delta = 0.3$	81.0	95.1	-32%	4.43
Fréchet $\delta = 0.4$	77.7	88.6	-26%	4.08

DC-Leap [21]: draft-guided contiguous leaping. Draft-guided contiguous leaping unchanged; commit-threshold rule replaced. At $\delta = 0.1$ the certificate adds $+1.7$ accuracy points at slightly higher throughput (97.6 versus 95.9 tok/s), confirming composition with draft-guided leaping.

	acc	tok/s	tok/NFE
DC-Leap, unmodified (commit thr .70)	77.0	95.9	
Fréchet $\delta = 0.1$	78.7	97.6	4.87
Fréchet $\delta = 0.2$	77.7	90.1	4.54
Fréchet $\delta = 0.25$	77.7	87.2	4.41
Fréchet $\delta = 0.3$	77.3	84.7	4.27
Fréchet $\delta = 0.4$	76.3	79.4	4.01

Swordsman [52]: entropy-driven adaptive block partition. Entropy-driven adaptive block partition, native accept rule replaced (prefix cache in both arms; wall time is the efficiency column, since its harness emits no NFE counter). At $\delta = 0.1$ the certificate adds $+1.3$ accuracy points at 16% less wall time; adaptive block partition and the certificate are additive on both axes.

	acc	wall (s)	wall cut
Swordsman, unmodified (thr .9)	79.3	1,044	
Fréchet $\delta = 0.1$	80.7	881	16%
Fréchet $\delta = 0.25$	79.0	905	13%
Fréchet $\delta = 0.4$	79.0	989	5%

CCD [9]: coherent contextual decoding. Its released code is Dream-only, so this block is on Dream-7B-Instruct with CCD's own counters (within-CCD comparison, not comparable to the LLaDA rows). On the Dream backbone the certificate halves CCD's finish-steps and doubles its answer throughput at equal or better accuracy at every margin tested, showing the mechanism is not LLaDA-specific.

	acc	tok/s	step cut
CCD, unmodified (seek, expand 4.0)	49.0	23.1	(102.9 avg finish-steps)
Fréchet $\delta = 0.1$	49.3	48.2	55%
Fréchet $\delta = 0.25$	49.0	47.8	54%
Fréchet $\delta = 0.4$	50.7	46.2	53%

WINO [17]: revocable draft-and-verify. Revocable draft-and-verify; Fréchet as the draft-acceptance rule, verification loop untouched. Accuracy parity at every margin but steps rise: WINO's revocable draft-verify loop is already more aggressive per step than a certificate that never rolls back a commit. The transfer is safe but not speed-additive; the two mechanisms address the same parallelism from opposite directions (optimistic revoke versus certified commit).

	acc	avg steps
WINO, unmodified (thr .6)	77.7	46.2
Fréchet $\delta = 0.1$	77.0	53.3 (+15%)
Fréchet $\delta = 0.2$	76.0	56.2 (+22%)
Fréchet $\delta = 0.25$	78.0	58.0 (+26%)
Fréchet $\delta = 0.3$	77.3	60.4 (+31%)
Fréchet $\delta = 0.4$	76.0	65.8 (+42%)

SOAR [8]: confidence-switched position beam search. Confidence-switched position beam search (zero-shot chain-of-thought GSM8K per its released scripts); its accept rule replaced. Forward work charges every beam row, so this is a quality/compute frontier comparison, not a pure speedup claim: the certificate cuts total forward work 55 to 65% and roughly doubles throughput at a 3 to 5 point accuracy cost. SOAR's beam search deliberately spends extra compute for quality, so the two occupy different points of the quality/compute frontier.

	acc	tok/s	forward work	tok/NFE
SOAR, unmodified (conf .95, beam 2)	72.7	70.2	39,995	1.82
Fréchet $\delta = 0.1$	68.7	157.1	−65%	5.17
Fréchet $\delta = 0.2$	69.7	149.7	−62%	4.81
Fréchet $\delta = 0.25$	68.0	151.0	−61%	4.61
Fréchet $\delta = 0.3$	67.3	140.1	−59%	4.41
Fréchet $\delta = 0.4$	69.0	139.5	−55%	4.05

Information-gain sampler [48]. Eight-candidate information-gain lookahead selection, replaced in place by the Fréchet certificate (cache-free in both arms, the only configuration in which its released sampler is active; the efficiency axis is effective forward work, since an eight-candidate lookahead batch charges eight rows). Replacing the candidate-lookahead policy with the certificate cuts effective forward work by 94 to 95% (10 to 13× throughput) at a 1.7 to 3.7 point accuracy cost; like SOAR,

information gain deliberately spends lookahead compute for quality, so the two occupy different points of the quality/compute frontier.

	acc	tok/s	forward work
Info-Gain, unmodified (8 candidates, thr .8)	75.7	10.7	320,322
Fréchet $\delta = 0.1$	72.0	139.7	-95%
Fréchet $\delta = 0.2$	74.0	138.0	-95%
Fréchet $\delta = 0.25$	73.3	129.6	-95%
Fréchet $\delta = 0.3$	73.3	120.8	-94%
Fréchet $\delta = 0.4$	72.0	111.7	-94%

ME-DLM [39]: edit-based refinement. Edit-based refinement for parallel masked diffusion; its fixed per-step commit quota in the mask-diffusion phase replaced by the Fréchet certificate, the edit-refinement phase untouched in both arms. This method requires its released finetuned checkpoint and uses its own scorer, so the comparison is within-method only. ME-DLM's default quota commits about 10.7 tokens per mask step and pays for that aggressiveness in accuracy; the certificate commits conservatively and adds 20.7 to 22.3 points of edit accuracy at every margin, at 4.6 to $5.1\times$ the per-sample time. The same pattern holds on the full 1,319-example test set (63.0 versus 85.4 at $\delta = 0.1$). This is a quality-recovery trade within the ME-DLM harness, not a speedup claim.

	edit acc	s/sample
ME-DLM, unmodified (author quota)	66.3	1.4
Fréchet $\delta = 0.1$	87.7	6.3
Fréchet $\delta = 0.2$	87.0	6.5
Fréchet $\delta = 0.25$	87.7	6.6
Fréchet $\delta = 0.3$	88.3	6.7
Fréchet $\delta = 0.4$	88.7	6.9

dParallel [12]: learned parallel decoding (distilled checkpoint). Learned/distilled system (released checkpoint dParallel-LLaDA-8B-instruct, its native certainty-forced threshold decoding). Fréchet is swapped into the distilled checkpoint; NFE cut is versus the matched factor selector $f = 1 - \delta$ on the same checkpoint and the same decode loop (the like-for-like selector baseline; the native decoder is a different generation method, so it is not a comparable NFE baseline). dParallel is a complementary intervention (it distills the model itself; accuracy shifts with distillation); the Fréchet efficiency mechanism still transfers onto its checkpoint, running 15 to 16% fewer NFE than matched factor.

	acc	tok/s	NFE	tok/NFE
dParallel, native decoder	79.0	50.6	9,842	6.11
	acc	tok/s	NFE cut vs matched factor	tok/NFE
Fréchet $\delta = 0.1$	72.0	61.1	16%	7.38
Fréchet $\delta = 0.2$	76.0	59.4	15%	7.17
Fréchet $\delta = 0.25$	78.7	59.2	15%	7.15
Fréchet $\delta = 0.3$	78.2	58.3	15%	7.04
Fréchet $\delta = 0.4$	79.2	56.2	16%	6.78

d3LLM [38]: pseudo-trajectory distillation (distilled checkpoint). Learned/distilled system (released checkpoint d3LLM_LLaDA, its native multi-block decoder; GSM8K under its own zero-shot chain-of-thought protocol). Fréchet is swapped into the distilled checkpoint. The native d3LLM decoder wins here because d3LLM is distilled (pseudo-trajectory distillation with early stopping) to parallel-decode along its own trained multi-block schedule, so it already realizes most of the safe parallelism a training-free selector targets, and that distillation sharpens per-token confidences, flattening the heterogeneity Fréchet's bonus uses. Fréchet therefore has little room for improvement over the native decoder.

	acc	tok/s	NFE	tok/NFE
d3LLM, native decoder	71.3	226.7	7,321	9.05
	acc	tok/s	NFE cut vs matched factor	tok/NFE
Fréchet $\delta = 0.1$	60.0	224.5	21%	8.96
Fréchet $\delta = 0.2$	66.0	221.8	22%	8.86
Fréchet $\delta = 0.25$	64.0	212.1	21%	8.47
Fréchet $\delta = 0.3$	69.0	207.8	20%	8.30
Fréchet $\delta = 0.4$	69.0	195.2	20%	7.79

DAPD [23]: decoding without blocks. DAPD decodes on a blockless global 256-token canvas. Inserting the certificate there is unsafe: it mass-commits the high-confidence end-of-sequence tail before the reasoning completes, and accuracy collapses (73.0 to 24.3 at $\delta = 0.1$) while steps rise. We verified per sample that this is a block-structure interaction, not a certificate defect: all positive results above use block-structured decoding, and DAPD's own rule avoids the same failure only by committing exclusively near-saturated confidences. The Fréchet certificate assumes block-structured or windowed candidate sets; blockless global canvases require an end-of-sequence or window guard before it applies. In the head-on reading, DAPD-Direct reaches 98.1 tok/s at 73.0 accuracy under its native rule, while Fréchet $\delta = 0$ on the same checkpoint, examples, and hardware reaches 97.1 tok/s at 78.0 accuracy.

Methods discussed but not run. As of August 2026 the following had no usable public implementation, or could not be run under an oracle-free same-backbone protocol, and are discussed in Appendix M instead: PUNT [3], ETE [14], LSP [28], locally coherent parallel decoding [16], temporal-spatial confidence extrapolation [30], and lookahead unmasking [25] (repositories absent or placeholders); self-speculative masked diffusions [7] additionally requires retraining; the product-of-experts bridge [42] releases code but its verifier mixture requires a Qwen-vocabulary backbone (Dream only, no LLaDA path); and Prophet [29] constitutively forces an answer template and monitors the answer region, so an oracle-free scored comparison is not defined.

Summary. Inserted into the concurrent training-free pipelines with usable code, the certificate is additive on both axes for Dynamic-dLLM, DSB, DC-Leap, and Swordsman (and, on Dream, CCD), as for KLASS, dLLM-Cache, and Streaming-dLLM; it recovers over 20 accuracy points inside ME-DLM's over-aggressive quota decoder; it is a quality/compute frontier trade against the deliberately compute-heavy SOAR and information-gain samplers; it is safe but not speed-additive inside WINO's revocable verifier; it transfers to the dParallel checkpoint; the native d3LLM decoder wins on its distilled checkpoint; and it requires block structure (DAPD). To our knowledge this covers every concurrent ICLR/ICML 2026 training-free dLLM acceleration method with a usable public implementation.

M Extended Related Work

Concurrent training-free acceleration of dLLM inference spans several mechanisms, grouped below by what each one changes.

Confidence-based selectors. KLASS [24], CCD [9], temporal-spatial confidence extrapolation [30], and lookahead unmasking [25] decide per-step commit sets from marginal confidence signals. Fast-dLLM's threshold and factor rules [45] belong to this family, and Fréchet profile decoding strictly generalizes the factor rule within it.

Dependence- and coupling-aware methods. DAPD [23], PUNT [3], CoDD [26], and locally coherent parallel decoding [16] use attention structure, independence tests, or learned joint layers to model what a marginal factorization ignores. Fréchet decoding uses only the marginal confidences; Lemma 7 describes how such dependence information can certify additional commits.

Adaptive schedulers. SlowFast sampling [43], DUS [35], Dynamic-dLLM [46], ETE [14], APD [19], and Prophet [29] reallocate where and how many tokens are decoded over the trajectory.

Draft-and-verify and revocable decoders. WINO [17], DC-Leap [21], self-speculative masked diffusions [7], and the product-of-experts bridge [42] trade extra verification compute for aggressive drafting.

Cache and system methods. dLLM-Cache [32], Streaming-dLLM [47], DPad [11], and stopping computation for converged tokens [37] cut per-step cost without changing the commit rule.

Adaptive block partition. Fast-dLLM-v2 [44], Swordsman [52], and DSB [34] restructure the canvas into blocks or sub-blocks.

Learned or distilled decoders. dParallel [12], d3LLM [38], and LSP [28] internalize parallelism at training time.

Theory and benchmarks. Recent work characterizes parallel-sampling limits, including provably optimal parallel samplers [20] and ParallelBench [22].

Fréchet profile decoding strictly generalizes the confidence-based factor rule and is complementary to the caching, scheduling, blocking, and draft-verify mechanisms above: applied on top of them it improves both quality and efficiency (Appendix L).

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: As stated in abstract, our resulting rule is a heterogeneous-confidence generalization of Fast-dLLM's factor selector. We draw the conclusion from comprehensive empirical experiments that this rule provides performance improvement compared to baselines.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss limitations and future directions of our method in the end of main paper, including improved bounds given extra dependency information.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: Assumptions and proofs are given in main paper and appendix.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: We detailed hyperparameters and provided the codebase for our implementation.

Guidelines:

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

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

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

   Answer: [Yes]

   Justification: We are providing the codebase of our implementation and experiments.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Although our method is training-free and plug-in, we detailed our choice of hyperparameters and models for replicating experiments in the paper.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: Evaluation of our benchmarks involve heavy computational resource, therefore, producing statistical significance for our results goes beyond our capacity.

   Guidelines:

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

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We detailed experimental hardware requirements in experiment section.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We have faithfully followed the guidelines provided by NeurIPS.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [No]

    Justification: The work focuses on the run-time acceleration of diffusion language models and is not application-specific. Therefore, there is no specific societal impacts to be claimed.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: Our project do not pose sch risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Original assets are credited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: We will provide a well-structured codebase as our original assets.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: Our experiment does not include engagement with human subjects.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: Our core methodology - the theoretical bound of performance-competitive acceleration is developed without the assistance of LLM.

Guidelines:

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