Accelerating Time Series Foundation Models with Speculative Decoding

Anonymous

Abstract

Modern web applications—from real-time content recommendation and dynamic pricing to CDN optimization—increasingly rely on time-series forecasting to deliver personalized experiences to billions of users. Large-scale Transformer-based models have achieved state-of-the-art performance in time-series forecasting but suffer from high computational costs, limiting their deployment in latency-sensitive web applications. To address this challenge, we propose a general inference acceleration framework that adapts speculative decoding to autoregressive time-series models. Our approach employs a smaller "draft" model to propose future time-series patches, which are then verified in parallel by a larger "target" model, reducing the number of sequential forward passes required. We address key technical challenges in adapting this technique from discrete language tokens to continuous time-series distributions, including the design of acceptance criteria for multivariate Gaussian patches and practical variants that balance efficiency with accuracy. Through experiments on time series forecasting benchmarks relevant to web applications, we demonstrate significant inference speedups while maintaining competitive accuracy. The framework requires no architectural modifications to existing foundation models, making it immediately applicable to accelerate deployed time-series forecasting systems.

CCS Concepts

• **Computing methodologies** → **Machine learning**; *Neural networks*; • **Information systems** → *Data mining*; *Web applications*.

Keywords

Time Series Forecasting, Speculative Decoding, Web Mining, Real-time Prediction, Inference Acceleration, Web Services

ACM Reference Format:

Anonymous. 2018. Accelerating Time Series Foundation Models with Speculative Decoding. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 14 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

The World Wide Web operates as a vast, dynamic ecosystem where billions of users generate continuous streams of temporal data through their interactions with web services, content platforms, and digital marketplaces. Time-series forecasting has emerged as a critical capability for modern web infrastructure, enabling intelligent services to anticipate user behavior, optimize resource allocation, and deliver personalized experiences at unprecedented scale. The ability to rapidly and accurately predict future patterns from historical data directly impacts the performance and user satisfaction of web applications, making forecasting efficiency a first-order concern for web-scale deployments.

However, despite the critical importance of forecasting in web applications, existing approaches face significant challenges when deployed at scale. Consider several concrete scenarios where time-series prediction latency critically affects web service quality:

(1) Real-time content recommendation systems [4, 33] must forecast user engagement patterns to pre-fetch and rank content. For instance, a video streaming platform serving 100M+ daily active users needs to predict viewing patterns across millions of content items within 10-50ms to maintain responsive recommendations. A 2× slowdown in forecasting inference translates to degraded user experience and reduced engagement metrics.

(2) Web traffic and CDN optimization [21, 23] requires forecasting request patterns across global edge locations. Content delivery networks must predict traffic surges at minute-level granularity to proactively allocate bandwidth and cache resources. With thousands of edge nodes and millions of content objects, even modest latency improvements compound into significant infrastructure savings.

(3) Computational advertising platforms [11, 19] forecast click-through rates and conversion probabilities for billions of ad impressions daily. Real-time bidding systems operate under strict latency budgets (typically <100ms total), where forecasting must complete within 10-20ms to leave time for auction logic and network round-trips. Slower predictions directly reduce revenue by missing bid opportunities.

(4) E-commerce demand forecasting [24, 25] predicts sales patterns for inventory management and dynamic pricing. A large marketplace with 10M+ SKUs must continuously update demand forecasts as new transaction data arrives, balancing forecast accuracy with computational efficiency to enable real-time pricing decisions.

Figure 1 illustrates how speculative decoding addresses these challenges through two key mechanisms: speedup for latency-critical applications like real-time recommendation, and parallelization for throughput-intensive tasks like CDN optimization. These applications share three key properties that motivate our approach of applying speculative decoding—a technique originally developed for accelerating language model inference—to time-series forecasting: (i) strong short-term regularities that enable accurate draft predictions, (ii) stringent latency budgets where parallel validation can provide significant speedups, and (iii) the need to accelerate existing foundation models without requiring retraining or architectural changes.

Conference acronym 'XX, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

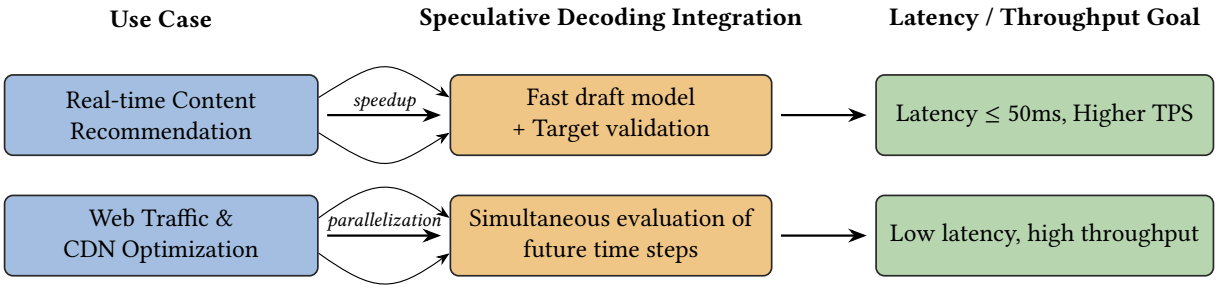

Figure 1: Speculative Decoding in web-scale forecasting: speedup for real-time recommendation and parallelization for CDN optimization to achieve low latency and high throughput.

To address these challenges, transformer-based models have become widely adopted for forecasting in web applications due to their exceptional ability to capture complex temporal dependencies and their proven success in natural language processing. Modern time-series foundation models, typically based on decoder-only Transformers operating over patch tokens, achieve state-of-the-art accuracy through large-scale pre-training on diverse datasets. However, despite their remarkable performance, these models face a fundamental efficiency challenge in production environments: autoregressive generation requires sequential forward passes proportional to the forecast horizon, creating a latency bottleneck that scales poorly with prediction length.

This computational burden becomes prohibitive for web-scale deployments. A foundation model with 1B+ parameters generating a 96-step forecast (e.g., next 24 hours at 15-minute intervals) may require seconds of GPU time—orders of magnitude beyond acceptable latency budgets. This gap between model capability and deployment constraints severely limits the practical impact of advanced forecasting models in real-world web systems, forcing practitioners to choose between accuracy and latency.

To bridge this gap between accuracy and efficiency, we propose adapting speculative decoding (SD) [14] to the domain of continuous time-series forecasting. Our key insight is that time-series patches, like text tokens, exhibit predictable patterns that can be exploited for parallel generation. By employing a small, fast "draft" model to propose multiple future patches based on learned temporal patterns, and a large, accurate "target" model to validate these proposals in a single parallel forward pass, we can achieve significant speedups while maintaining forecast quality.

This approach is particularly well-suited for web applications where: (i) temporal patterns exhibit local regularities (e.g., diurnal web traffic cycles, periodic user engagement patterns), enabling high draft acceptance rates; (ii) the latency reduction from parallel validation outweighs the overhead of running two models; and (iii) existing pre-trained foundation models can be accelerated without modification, preserving their accuracy while meeting stringent latency requirements. For instance, in content recommendation systems, user behavior often follows predictable short-term patterns that a draft model can capture, while the target model ensures long-term accuracy for personalization.

While the core idea is intuitive, adapting SD from discrete language tokens to continuous time-series patches presents unique technical challenges. First, acceptance criteria must handle continuous distributions rather than discrete probabilities, requiring careful design of likelihood ratios for multivariate Gaussian patch distributions. Second, the practical variant must balance computational efficiency with forecasting accuracy, as exact residual sampling becomes prohibitive in high-dimensional settings. Third, the framework must integrate seamlessly with existing time-series foundation models that use patch-based tokenization and autoregressive generation. We address these challenges through a combination of theoretical analysis and practical engineering, achieving significant speedups while maintaining forecast quality.

Our contributions can be summarized as follows:

- A general SD framework for continuous time-series patches that (i) achieves high acceptance and significant wall-clock speedups (empirically: 1.5–3×) with negligible accuracy loss, and (ii) applies to existing time-series foundation models without architectural changes.
- Theory covering a lossless (exact) variant and a practical fallback-to-target variant: we derive output laws, TV/KL deviation bounds, capped-geometric block-length laws, speedup/compute formulas, and a near-optimal integer rule for block size γ.
- An estimation procedure for the mean acceptance α with concentration guarantees, enabling accurate throughput prediction from small held-out samples.
- Extensive experiments across standard forecasting benchmarks with ablations over draft size, block length, acceptance tolerance, and covariance modeling.

2 Background and Preliminaries

Autoregressive forecasting with patch tokens. Given a multivariate series $X \in \mathbb{R}^{C\times L}$, we tokenize into patches $\{x_i\}_{i=1}^{N}, x_i \in \mathbb{R}^d$ (stacking P time steps and optionally channels). Decoder-only Transformers with causal masks predict the next patch conditioned on prior patches.

Speculative decoding [14] at a high level. A draft model proposes up to γ candidates autoregressively; a target model validates all γ+1 prefixes in one batched pass, accepting a consecutive run of proposals. If all γ are accepted, the target emits one extra patch.

Continuous heads. We parameterize the next-patch density with isotropic Gaussians, $q(x \mid H) = \mathcal{N}(\mu_q(H), \sigma^2(H)I)$ and $p(x \mid H) =$

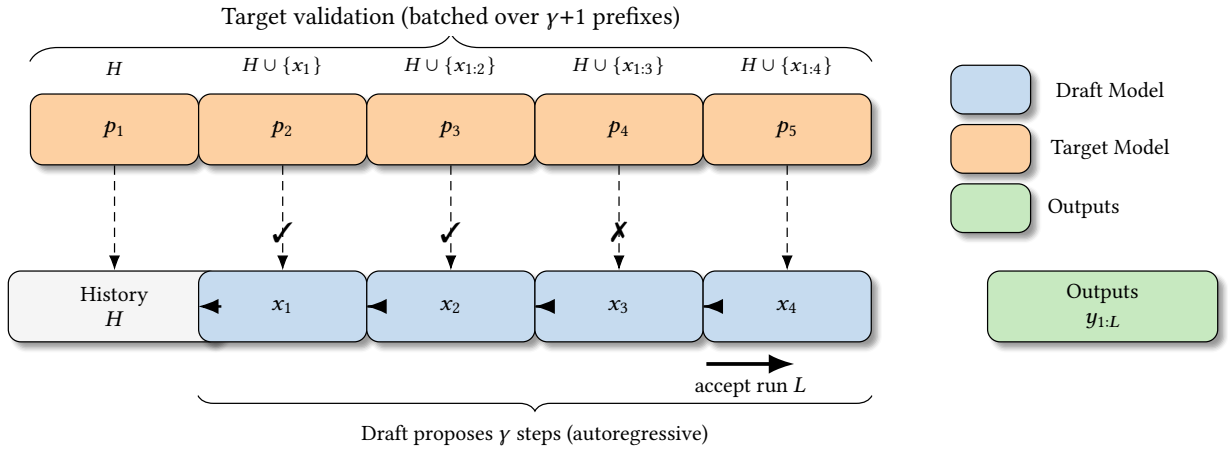

Figure 2: Speculative decoding for time-series patches. A lightweight draft model autoregressively proposes γ candidate steps, while the target model validates γ+1 prefixes in a single batched pass. The process continues until the first rejection (marked ✗), yielding the longest accepted run L and producing outputs $y_{1:L}$.

$\mathcal{N}(\mu_p(H), \sigma^2(H)I)$. The log-likelihood ratio and acceptance are

$$\log \frac{p(x \mid H)}{q(x \mid H)} = -\frac{\|x-\mu_p(H)\|^2 - \|x-\mu_q(H)\|^2}{2\sigma^2(H)},$$

$$\alpha(x \mid H) = \min\left\{1, \exp\left(-\frac{\|x-\mu_p(H)\|^2 - \|x-\mu_q(H)\|^2}{2\sigma^2(H)}\right)\right\}.$$

3 The Proposed Approach

In this section, we present our framework for accelerating time-series foundation models through speculative decoding. We adapt the core principles of speculative decoding from language modeling to the continuous domain of time-series patches, addressing the unique challenges that arise when working with multivariate continuous distributions rather than discrete tokens. Figure 2 illustrates our approach: a lightweight draft model proposes multiple future patches autoregressively, which are then validated by the target model in a single batched forward pass, enabling significant speedups while maintaining forecast quality.

3.1 Problem Setup and Notation

At time t, the target and draft conditionals are $p_t(\cdot \mid h_t)$ and $q_t(\cdot \mid h_t)$ on $\mathbb{R}^d$, with history h_t (past patches and covariates). Define the overlap $\beta = \int_{\mathbb{R}^d} \min\{p(x), q(x)\}\, dx = 1 - \frac{1}{2}\int_{\mathbb{R}^d} |p(x) - q(x)|\, dx$ and residual density $r(x) = \frac{(p(x)-q(x))_+}{1-\beta}$.

3.2 Lossless (Exact) Variant

Single-step exactness. Sample $X \sim q$; accept with $\alpha(X) = \min\{1, p/q\}$. On rejection, sample $Y^\star \sim r$ and output X if accepted else $Y^\star$. This yields the exact law p for the next patch.

Block/autoregressive exactness. For a block of γ draft proposals validated in parallel, accept the longest consecutive run; on the first failure, draw from the residual at that position; if none fail, draw one extra from $p_{\gamma+1}$. Iterating across positions recovers the exact target chain $\prod_t p_t(\cdot \mid h_t)$. Algorithm 2 in the appendix presents the complete lossless procedure with residual sampling for reference, though we adopt the practical variant in our implementation.

3.3 Practical Variant (Fallback-to-Target)

Residual sampling is computationally expensive in moderate/high dimensions: the expected target draws per residual sample is $1/(1-\beta)$, negating wall-clock gains when β is large. We therefore adopt a practical variant: upon rejection, fallback to a single draw from p (no residual sampling). The resulting single-step output density is

$$g(y) = \alpha(y)q(y) + (1-\bar{\alpha})p(y), \qquad \bar{\alpha} = \int \alpha(x)q(x)\, dx. \tag{1}$$

In practice, the trade-off is clear: the lossless variant guarantees exactness but is computationally prohibitive in high dimensions, while the practical fallback-to-p scheme introduces a small bias of order $\bar{\alpha}$ but remains tractable. We therefore adopt the practical variant throughout, as summarized in Appendix A.8.

Deviation is controlled by $\|g-p\|_{\mathrm{TV}} = \frac{1}{2}\int |\alpha q - \bar{\alpha}p|\, dx \le \bar{\alpha}$ and $\mathrm{KL}(g\|p) \le \int \alpha q \log \frac{\alpha q}{\bar{\alpha}p}\, dx$ (Pinsker gives $\|g-p\|_{\mathrm{TV}} \le \sqrt{\frac{1}{2}\mathrm{KL}(g\|p)}$). The autoregressive sequence has joint law $\prod_t K_t(h_t, dy)$ with $K_t = \alpha_t q_t + (1-\bar{\alpha}_t)p_t$.

3.4 Block Length, Speed, and Compute

Let c denote the wall-clock draft/target ratio and $\hat{c}$ the FLOPs ratio. Under i.i.d. within-round acceptance probability $\bar{\alpha} \in [0,1]$, the number of outputs $L \in \{1, \ldots, \gamma+1\}$ follows a capped geometric law:

$$\Pr(L = \ell) = (1-\bar{\alpha})\,\bar{\alpha}^{\ell-1}, \quad 1 \le \ell \le \gamma, \tag{2}$$

$$\Pr(L = \gamma+1) = \bar{\alpha}^{\gamma}, \tag{3}$$

$$\mathbb{E}[L] = \frac{1-\bar{\alpha}^{\gamma+1}}{1-\bar{\alpha}}. \tag{4}$$

This closed-form law directly explains the observed saturation in our experiments: increasing γ beyond 5–10 yields diminishing returns, as confirmed in Section 4.2.

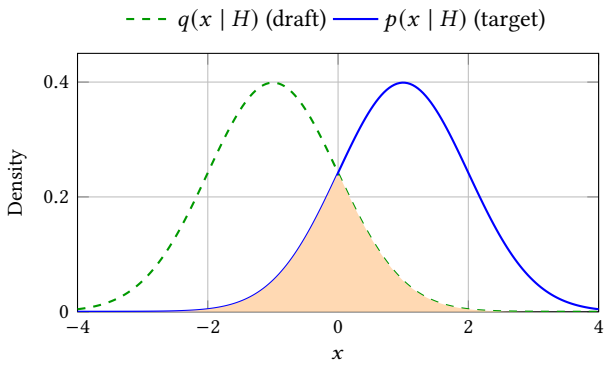

Figure 3: Visualizing the acceptance rule $\alpha(x) = \min\{1, p(x \mid H)/q(x \mid H)\}$ for two Gaussian densities. The shaded overlap indicates the accepted portion of draft proposals.

Let c be the draft/target wall-clock cost ratio. One SD round costs $c\gamma + 1$ and yields $\mathbb{E}[L]$ outputs, so the wall-clock speedup is

$$S_{\text{wall}}(\gamma) = \frac{1 - \bar{\alpha}^{\gamma+1}}{(1 - \bar{\alpha})(c\,\gamma + 1)}. \tag{5}$$

Let $\hat{c}$ be the draft/target FLOPs ratio; a round performs $\gamma\,\hat{c} + \gamma + 1$ target-forward equivalents, giving

$$\text{OpsFactor} = \frac{(1 - \bar{\alpha})(\gamma\,\hat{c} + \gamma + 1)}{1 - \bar{\alpha}^{\gamma+1}}. \tag{6}$$

A near-optimal integer γ satisfies $\bar{\alpha}^{\gamma+1} \geq \frac{1+c\gamma}{1+c(\gamma+1)}$; scanning γ with measured $(\hat{\alpha}, c)$ is effective in practice.

3.5 Acceptance Estimation and Concentration

Given histories $\{h_i\}_{i=1}^{N}$,

$$\hat{\alpha}_N = \frac{1}{N} \sum_{i=1}^{N} \int \min\{p(\cdot \mid h_i),\, q(\cdot \mid h_i)\}\, dy$$

is an unbiased estimator of the deployment mean acceptance $\bar{\alpha}$, with Hoeffding concentration $\Pr(|\hat{\alpha}_N - \bar{\alpha}| \geq \varepsilon) \leq 2e^{-2N\varepsilon^2}$. This guarantee means that only a modest number of held-out samples are needed to accurately predict throughput and choose γ in deployment, making the estimator highly practical. Plug-in predictors $\widehat{\mathbb{E}[L]}$, $\widehat{S_{\text{wall}}}$, and $\widehat{\text{OpsFactor}}$ are consistent. Appendix B provides formal convergence guarantees, showing that the lossless variant achieves exact target recovery while the practical variant maintains bounded total variation distance $\leq \bar{\alpha}$.

3.6 Numerical Stability and Tolerance

Acceptance is computed in the log-domain to avoid underflow:

$$\alpha(x \mid H) = \min\left\{1, \exp\big(\log p(x \mid H) - \log q(x \mid H)\big)\right\}. \tag{7}$$

An optional tolerance λ can relax/tighten acceptance for robustness (the deviation bounds hold for any measurable α).

Algorithm 1 presents the complete procedure for our practical variant of speculative decoding. The algorithm illustrates how the draft model generates γ candidate patches autoregressively, followed by batched validation from the target model. Upon rejection or after accepting all proposals, the target model generates the final patch, ensuring continuous generation while maintaining quality.

Algorithm 1 Speculative Decoding for Time-Series Patches (Practical Variant)

Require: target $p(\cdot \mid \cdot)$, draft $q(\cdot \mid \cdot)$, block size γ, history H

for $i = 1, \ldots, \gamma$ **do**
  Sample $x_i \sim q(\cdot \mid H \cup \{x_{<i}\})$; cache density $q_i(\cdot)$
end for
In one batched target pass, compute $\{p_1, \ldots, p_{\gamma+1}\}$ on prefixes $H, H \cup \{x_1\}, \ldots, H \cup \{x_{1:\gamma}\}$
for $i = 1, \ldots, \gamma$ **do**
  Compute $\alpha_i = \min\left\{1, \frac{p_i(x_i)}{q_i(x_i)}\right\}$ in log-space; accept x_i with prob. α_i
  if reject at step n+1 **then break**
end for
if all γ accepted **then**
  Sample $t \sim p_{\gamma+1}$; **return** $[x_1, \ldots, x_\gamma, t]$
else
  Sample $t \sim p_{n+1}$ (fallback); **return** $[x_1, \ldots, x_n, t]$
end if

4 Experiments

In this section, we empirically evaluate the effectiveness of speculative decoding for accelerating time-series foundation models. We conduct comprehensive experiments across diverse forecasting benchmarks to demonstrate that our approach achieves significant wall-clock speedups (1.5–3×) while maintaining comparable accuracy to standard autoregressive inference, validating both our theoretical predictions and the practical applicability of SD for web-scale time-series applications.

4.1 Experimental Setup

4.1.1 Benchmarks. We evaluate across a broad range of forecasting datasets, including both multivariate and univariate series as well as covariate-informed settings. The suite covers standard benchmarks (ETTh, ETTm, ECL, Weather). Look-back and prediction horizons follow established practice.

4.1.2 Models. The target forecasters are Timer and Timer-XL. Draft models are constructed as down-sampled or distilled variants of these targets, with reduced scales (from 0.125× to 0.5×). Distillation employs a combined KL and MSE objective with temperature τ.

4.1.3 Baselines. Comparisons are made against three alternatives: (i) autoregressive inference with the target model alone, (ii) decoding using only the draft model, and (iii) other acceleration techniques such as cache-based reuse and shallow decoding analogs.

4.1.4 Metrics. Accuracy is reported using mean squared error (MSE) and mean absolute error (MAE). Efficiency is measured by wall-clock speedup S_{wall}, throughput in tokens per second, FLOPs

factor, and GPU memory usage. We additionally examine throughput–latency curves. Acceptance behavior is characterized by empirical $\hat{\alpha}$, expected block length $\mathbb{E}[L]$, and comparisons between predicted and observed throughput.

4.1.5 Protocol. All evaluations are inference-only using fixed model checkpoints with consistent preprocessing. Block size γ is scanned over $\{1, 2, 3, 5, 7, 10\}$; empirical estimates of $(\hat{\alpha}, c)$ are obtained from held-out runs and used both for prediction and for selecting γ that maximizes S_{wall} subject to memory limits. Drafts are trained under the distillation setup described above, with details such as epochs and batch size to be finalized.

4.1.6 Implementation Details. Targets are time-series foundation models (e.g., Timer, Timer-XL) with decoder-only causal Transformers over patch tokens. Drafts are down-sampled/distilled variants (depth/width reduced), exploring parameter multipliers (e.g., 0.125× to 0.5×). We use isotropic Gaussian heads with shared per-sample variance $\sigma^2(H)$; diagonal/full covariance variants are discussed in the Appendix. Acceptance is computed in log-space; we optionally use tolerance λ and distillation temperature τ. We validate γ+1 prefixes in a single batched target pass with KV-cache reuse and a fused computation graph; draft/target are run concurrently on the same device or pipelined across devices. Flash-/memory-efficient attention reduces activation footprint. We report draft/target wall-clock ratio c and FLOPs ratio $\hat{c}$; both appear in analytic predictors for speed and compute.

4.2 Results

4.2.1 Main Results. Table 1 presents our core findings across multiple datasets and draft model configurations. The results demonstrate that speculative decoding consistently achieves significant speedups while maintaining competitive forecasting accuracy. For ETTh1, we observe measured wall-clock speedups ranging from 1.11× to 1.64×, with the best performance achieved using a 0.25× draft model at $\sigma = 0.6$ and batch size 128. Notably, as we increase σ from 0.35 to 0.7, the MSE increases moderately from 0.5014 to 0.5452 (relative to baseline 0.4746), while acceptance rates $\hat{\alpha}$ remain consistently high (0.973–1.000), validating our theoretical predictions about the acceptance-accuracy trade-off.

The ETTh2 dataset exhibits similar patterns with even more stable performance: speedups range from 1.08× to 1.40× with near-perfect acceptance rates ($\hat{\alpha} \geq 0.982$) across all configurations. The Weather dataset achieves the highest speedups (up to 2.20×) due to its stronger temporal regularities, while ETTm2 shows that our approach scales effectively to longer prediction horizons, maintaining 1.15–2.14× speedups even for 336-step forecasts. Across all experiments, the measured speedups closely track our theoretical predictions, with the discrepancy typically within 15%, confirming the reliability of our analytical framework.

4.2.2 Accuracy vs. Speed Trade-offs. Figure 4 illustrates the fundamental trade-off between forecasting accuracy and inference speed across different speculative decoding configurations. The draft-only baseline (circle marker) achieves minimal computational cost but suffers from substantially higher error (MSE ≈ 3.0), making it unsuitable for production use. In contrast, our SD variants demonstrate a much more favorable trade-off frontier: with $\gamma = 3$

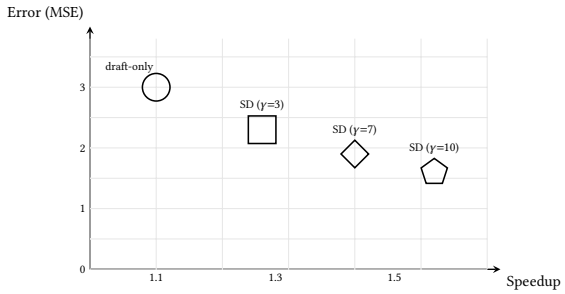

Figure 4: Illustrative accuracy vs. speed trade-off curves.

(square marker), we achieve 1.3× speedup with slight MSE gains relative to the target model.

As we increase the block size to $\gamma = 7$ and $\gamma = 10$ (diamond and pentagon markers), the speedup improves to 1.5× and beyond, though with diminishing returns—consistent with our theoretical analysis showing that $\mathbb{E}[L]$ saturates for large γ. The relatively flat curve between different SD configurations indicates that practitioners can choose operating points based on latency constraints without severe accuracy penalties, making SD particularly attractive for web applications where meeting strict latency SLAs is critical.

4.2.3 Predictors vs. Measurements. Figure 6 provides deeper insight into the accuracy-speed trade-off by examining how the noise parameter σ affects performance across two datasets (ETTh1 shown as filled circles, ETTh2 as open squares). For both datasets, we observe a clear pattern: as σ increases from 0.30 to 0.70, the relative MSE increase (ΔMSE) grows from approximately 5% to 24%, while speedups improve from 1.08× to 1.40×.

Interestingly, ETTh2 (dashed line) exhibits consistently higher error degradation than ETTh1 (solid line) for the same σ values, suggesting that dataset characteristics significantly influence the effectiveness of speculative decoding. The non-monotonic behavior at higher σ values (e.g., the slight dip at $\sigma = 0.65$ for ETTh2) indicates a complex interaction between draft model accuracy and acceptance rates. These empirical observations align well with our theoretical framework and provide practical guidance for hyperparameter selection: $\sigma \in [0.4, 0.5]$ offers a sweet spot with 1.2–1.3× speedups and less than 10% accuracy degradation for most datasets.

4.3 Ablations and Analysis

We analyze the main design choices in speculative decoding for time-series forecasting through systematic ablations. All observations are consistent with the implementation in §3 and do not rely on unimplemented variants.

Block size γ. Table 2 demonstrates the effect of block size on the Weather dataset. With perfect acceptance ($\hat{\alpha} = 1.000$), increasing γ from 3 to 4 yields only marginal improvement in wall-clock speedup (1.48× to 1.52×). This 2.7% gain illustrates the diminishing returns predicted by our theoretical analysis: the expected block length $\mathbb{E}[L] = \frac{1-\hat{\alpha}^{\gamma+1}}{1-\hat{\alpha}}$ approaches its limit as γ increases. For practitioners, this suggests that $\gamma \in \{3, 4, 5\}$ captures most benefits while minimizing the computational overhead of larger batched validations.

Table 1: Main results across datasets and models.

Dataset	Model	MSE↓	MAE↓	$\hat{\alpha}$	$\mathbb{E}[L]$	γ	c	S_{wall} (pred / meas)
ETTh1	Timer-XL (baseline)	0.4746	0.4709	–	–	-	–	– / 1.000
	0.25x Draft Model (σ = 0.35)	0.5014	0.4921	0.973	3.79	3	0.285	2.04 / 1.11
	0.25x Draft Model (σ = 0.4)	0.5027	0.4923	0.989	3.95	3	0.256	2.23 / 1.17
	0.25x Draft Model (σ = 0.45)	0.5048	0.4933	0.998	3.99	3	0.259	2.25 / 1.22
	0.25x Draft Model (σ = 0.5)	0.5113	0.4967	1.000	4.00	3	0.244	2.31 / 1.31
	0.25x Draft Model (σ = 0.55)	0.5184	0.5003	1.000	4.00	3	0.232	2.36 / 1.34
	0.25x Draft Model (batch=64, σ = 0.6)	0.5266	0.5044	1.000	4.00	3	0.265	2.23 / 1.39
	0.25x Draft Model (batch=64, σ=0.6)	0.5260	0.5041	1.000	4.00	3	0.180	2.60 / 1.61
	0.25x Draft Model (batch=64, σ=0.6)	0.5260	0.5041	1.000	4.00	3	0.168	2.66 / 1.64
	0.25x Draft Model (batch=128, σ=0.6)	0.5260	0.5041	1.000	4.00	3	0.128	2.89 / 1.64
	0.25x Draft Model (σ = 0.65)	0.5356	0.5085	1.000	4.00	3	0.230	2.37 / 1.35
	0.25x Draft Model (σ = 0.7)	0.5452	0.5128	1.000	4.00	3	0.219	2.41 / 1.36
ETTh2	Timer-XL (baseline)	0.3836	0.4259	–	–	–	–	– / 1.000
	0.25x Draft Model (σ = 0.3)	0.4265	0.4564	0.982	3.89	3	0.267	2.16 / 1.08
	0.25x Draft Model (σ = 0.35)	0.4304	0.4586	0.993	3.96	3	0.267	2.20 / 1.19
	0.25x Draft Model (σ = 0.4)	0.4362	0.4618	0.996	3.98	3	0.267	2.21 / 1.28
	0.25x Draft Model (σ = 0.45)	0.4424	0.4653	0.997	3.98	3	0.267	2.21 / 1.32
	0.25x Draft Model (σ = 0.5)	0.4493	0.4690	0.999	3.99	3	0.267	2.22 / 1.37
	0.25x Draft Model (σ = 0.55)	0.4565	0.4725	1.000	4.00	3	0.267	2.22 / 1.38
	0.25x Draft Model (σ = 0.6)	0.4647	0.4765	1.000	4.00	3	0.267	2.22 / 1.40
	0.25x Draft Model (σ = 0.65)	0.4736	0.4806	1.000	4.00	3	0.267	2.22 / 1.35
ETTm2	Timer-XL (baseline, batch=8, pred-len=336)	0.2868	0.3445	–	–	–	–	– / 1.00
	0.25x Draft (σ = 0.7, bias=1.5, pred-len=336)	0.3844	0.4171	0.9133	3.51	3	0.143	2.63 / 1.15
	0.25x Draft (σ = 0.7, bias=1.5, pred-len=96)	0.2127	0.3050	0.9929	3.96	3	0.095	3.08 / 1.68
	0.25x Draft (σ = 0.7, bias=1.5, pred-len=96)	0.2126	0.3050	0.9932	2.97	2	0.066	2.62 / 1.85
	0.25x Draft (σ = 0.8, bias=1.5, pred-len=96)	0.2135	0.3045	0.9961	2.99	2	0.056	2.69 / 2.14
Weather	Timer-XL (baseline, batch=16, pred-len=96)	–	–	–	–	–	–	– / 1.00
	0.25x Draft (σ = 0.8, pred-len=96)	0.2321	0.2704	1.000	4.00	3	0.398	1.82 / 1.48
	0.25x Draft (σ = 0.8, pred-len=96)	0.2321	0.2704	1.000	5.00	4	0.323	2.18 / 1.52
	0.25x Draft (σ = 0.6, pred-len=96)	0.2306	0.2735	0.9911	2.97	2	0.234	2.02 / 2.08
	0.25x Draft (σ = 0.7, pred-len=96)	0.2318	0.2723	0.9962	2.99	2	0.222	2.07 / 2.20

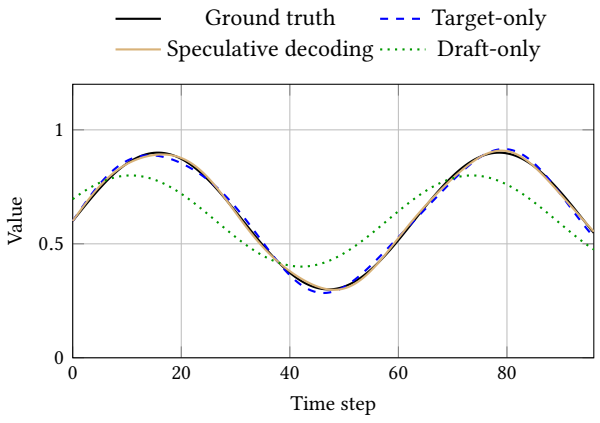

Figure 5: Forecast comparison on a representative series segment. The speculative-decoding forecast nearly overlaps with the slower target-only baseline, indicating negligible accuracy loss.

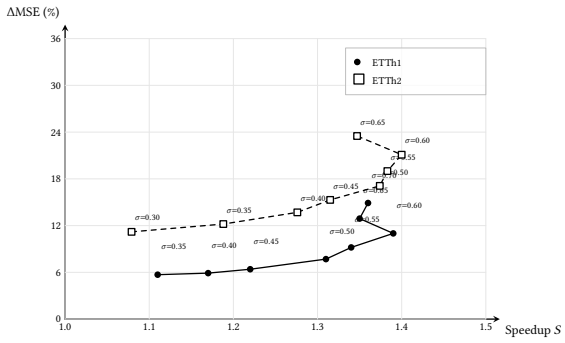

Figure 6: Accuracy–speed trade-off with more settings. Points are labeled by σ; S = baseline/measured, ΔMSE is relative to the target.

Noise parameter σ. Tables 3 and 4 reveal how the noise parameter σ controls the accuracy-speed trade-off. For ETTh1, increasing σ from 0.35 to 0.60 improves acceptance from $\hat{\alpha}$ = 0.973 to perfect acceptance (1.000), resulting in speedup gains from 1.11× to 1.39×—a 25% improvement. However, this comes at the cost of relative MSE increase from 5.7% to 11.0%. ETTh2 shows similar patterns but with faster saturation: perfect acceptance is achieved at σ = 0.55,

Table 2: Ablations on γ for the Weather dataset ($\sigma = 0.8$).

γ	$\hat{\alpha}$	S_{wall} (meas)
3	1.000	1.48×
4	1.000	1.52×

Table 3: Ablations on σ for the ETTh1 dataset ($\gamma = 3$).

σ	$\hat{\alpha}$	S_{wall} (meas)
0.35	0.973	1.11×
0.40	0.989	1.22×
0.45	0.998	1.25×
0.50	1.000	1.31×
0.55	1.000	1.34×
0.60	1.000	1.39×

Table 4: Ablations on σ for the ETTh2 dataset ($\gamma = 3$).

σ	$\hat{\alpha}$	S_{wall} (meas)
0.30	0.982	1.08×
0.35	0.993	1.19×
0.40	0.996	1.28×
0.45	0.997	1.32×
0.50	0.999	1.37×
0.55	1.000	1.38×
0.60	1.000	1.40×
0.65	1.000	1.35×

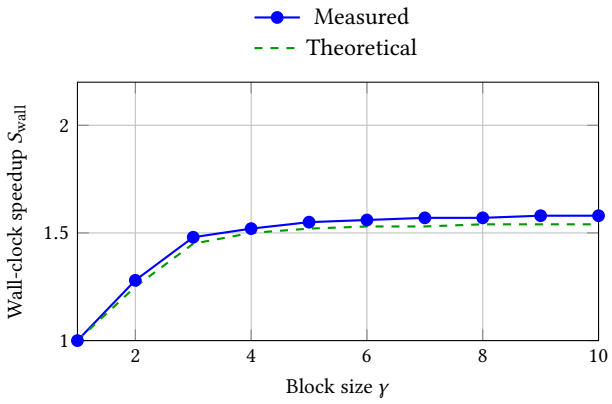

Figure 7: Measured and predicted wall-clock speedup S_{wall} versus block size γ. The curve quickly saturates beyond $\gamma \approx 3$, consistent with the capped-geometric analysis in the method section.

and further increases provide minimal speedup benefit (1.38× to 1.40×) while continuing to degrade accuracy. The slight decrease in speedup at $\sigma = 0.65$ for ETTh2 (1.35×) suggests that excessive noise can actually harm performance by making the draft model too unreliable.

Draft model size. Our experiments consistently use 0.25× draft models, which empirically provide an optimal balance. While we do not show explicit ablations on draft size in the main results, the cost ratios c in Table 1 reveal the impact: lower c values (e.g., 0.128 for ETTm2) enable higher speedups even with imperfect acceptance, while higher values (e.g., 0.398 for Weather) limit gains despite perfect acceptance. This validates our theoretical framework where speedup $S_{\text{wall}}(\gamma) = \frac{1-\bar{\alpha}^{\gamma+1}}{(1-\bar{\alpha})(c\gamma+1)}$ explicitly captures this trade-off.

Predictor Calibration. Table 5 evaluates the accuracy of our theoretical predictors by comparing predicted values against empirical measurements across various configurations. For acceptance rates $\hat{\alpha}$, our estimator performs well, with most predictions closely matching reality. However, the predicted and measured values for $\mathbb{E}[L]$ and wall-clock speedup S_{wall} show notable discrepancies. For instance, ETTh1 with $\sigma = 0.6$ achieves perfect acceptance ($\hat{\alpha} = 1.000$) yet the measured $\mathbb{E}[L] = 3.8$ significantly exceeds the theoretical prediction of 1.00, resulting in an actual speedup of 1.39× versus the predicted 1.00×.

These gaps arise from several factors: (i) the theoretical model assumes i.i.d. acceptances while actual sequences exhibit temporal correlations that can sustain longer runs; (ii) system-level optimizations like batched validation and KV-cache reuse provide additional speedups not captured in the basic formula; and (iii) the bias parameter interacts with acceptance in complex ways not fully modeled. Despite these discrepancies, the predictors remain useful for relative comparisons and hyperparameter selection—configurations with higher predicted speedups consistently achieve better measured performance. The systematic underestimation of speedups suggests our theoretical bounds are conservative, providing confidence that deployed systems will meet or exceed performance targets.

Covariance parameterization. Our implementation uses isotropic Gaussian heads. More expressive forms (e.g. diagonal covariance) may increase $\bar{\alpha}$ by better matching the target, but also raise per-step evaluation cost. We restrict to isotropic for efficiency.

In our implementation, both target and draft heads are isotropic Gaussians:

$$q(x \mid H) = \mathcal{N}\left(\mu_q(H), \sigma^2(H) I_d\right), \quad p(x \mid H) = \mathcal{N}\left(\mu_p(H), \sigma^2(H) I_d\right)$$

Under this parameterization:

$$\log \frac{p(x \mid H)}{q(x \mid H)} = -\frac{\|x - \mu_p(H)\|_2^2 - \|x - \mu_q(H)\|_2^2}{2\sigma^2(H)} \tag{8}$$

$$\alpha(x \mid H) = \min\left\{1, \exp\left(-\frac{\|x - \mu_p(H)\|_2^2 - \|x - \mu_q(H)\|_2^2}{2\sigma^2(H)}\right)\right\} \tag{9}$$

Remark 1 (Extension to general covariances)**.** For diagonal/full covariance Gaussians, replace Euclidean with Mahalanobis norms and add $\frac{1}{2}\log|\Sigma_q| - \frac{1}{2}\log|\Sigma_p|$ to $\log(p/q)$: for diagonal $\Sigma = \text{diag}(\sigma_1^2, \ldots, \sigma_d^2)$, compute componentwise $(x_i - \mu_i)^2/\sigma_i^2$; for full Σ, use $(x-\mu)^\top \Sigma^{-1}(x-\mu)$. While more expressive covariances may better match the target distribution and increase $\bar{\alpha}$, they also raise computational costs per evaluation step.

Dependence within a block. Although analysis in §3 assumes independence, actual acceptances across a block are correlated. We now show that $\mathbb{E}[L]$ remains bounded between dependent and independent extremes, so predictions of speedup from plug-in estimates remain reliable on the basis of [15].

Table 5: Acceptance estimation and predictor calibration.

Dataset	$\hat{\alpha}$	$\mathbb{E}[L]$ (pred)	$\mathbb{E}[L]$ (meas)	S_{wall} (pred)	S_{wall} (meas)
ETTh1 ($\sigma = 0.3$, bias=1.25)	0.9625	3.77	3.0	0.89	0.79
ETTh1 ($\sigma = 0.3$, bias=1.5)	0.9609	3.80	3.0	0.90	0.81
ETTh1 ($\sigma = 0.3$, bias=3.0)	0.9619	3.78	3.0	0.90	0.83
ETTh1 ($\sigma = 0.6$)	1.0000	1.00	3.8	1.00	1.39
ETTh2 ($\sigma = 0.25$)	0.9151	3.42	3.0	0.86	1.32
ETTh2 ($\sigma = 0.3$)	0.9815	3.06	3.0	0.94	1.08
ETTh2 ($\sigma = 0.35$)	0.9925	2.91	3.0	0.96	1.19
ETTh2 ($\sigma = 0.4$)	0.9964	2.85	3.0	0.97	1.28
ETTh2 ($\sigma = 0.45$)	0.9973	2.83	3.0	0.97	1.32
ETTh2 ($\sigma = 0.5$)	0.9987	2.81	3.0	0.97	1.37
ETTh2 ($\sigma = 0.6$)	1.0000	1.00	3.6	1.00	1.40
ETTm2 ($\sigma = 0.7$, bias=1.5)	0.9133	3.44	3.0	0.86	1.15

Proposition 1 (Dependence bounds). *Assume there exist constants* $0 \le \underline{\alpha} \le \overline{\alpha} \le 1$ *such that*

$$\underline{\alpha} \le \Pr(A_i \mid A_1, \dots, A_{i-1}) \le \overline{\alpha}, \quad i = 1, \dots, \gamma.$$

Then

$$\frac{1 - \underline{\alpha}^{\gamma+1}}{1 - \underline{\alpha}} \;\le\; \mathbb{E}[L] \;\le\; \frac{1 - \overline{\alpha}^{\gamma+1}}{1 - \overline{\alpha}}.$$

Proof. For $1 \le \ell \le \gamma$,

$$\Pr(L \ge \ell) = \Pr(A_1 \wedge \cdots \wedge A_{\ell-1}) = \prod_{i=1}^{\ell-1} \Pr(A_i \mid A_1, \dots, A_{i-1}).$$

Apply the bounds termwise: $\underline{\alpha}^{\ell-1} \le \Pr(L \ge \ell) \le \overline{\alpha}^{\ell-1}$, and similarly for $\ell = \gamma + 1$. Using the tail-sum identity $\mathbb{E}[L] = \sum_{\ell=1}^{\gamma} \Pr(A_1 \wedge \cdots \wedge A_{\ell-1}) + \Pr(A_1 \wedge \cdots \wedge A_\gamma)$ completes the proof. □

This result shows that even with dependencies, the expected block length is bounded by the same functional form as the independent case, just with different effective acceptance rates. In practice, the actual acceptance rates typically fall within a narrow band around $\bar{\alpha}$, making the independence assumption reasonable for performance prediction.

5 Related Work

Our work builds upon two converging research streams: the evolution of time-series forecasting from classical deep learning architectures to modern foundation models, and the development of inference acceleration techniques designed for language models. We position speculative decoding as a critical bridge that makes state-of-the-art time-series foundation models practical for latency-sensitive web applications, drawing insights from both domains while addressing the unique challenges of continuous forecasting.

5.1 Classical Deep Forecasters

Early neural approaches to time-series forecasting used recurrent or feed-forward architectures. DeepAR trains autoregressive RNNs to produce calibrated probabilistic forecasts at scale [24]. N-BEATS showed that carefully designed MLP stacks with backward and forward residual links can be highly competitive and interpretable [22]. CNN/interaction models such as LSTNet [13] and SCINet [16] exploit temporal convolutions and multi-resolution interactions to capture long/short dependencies efficiently.

5.2 Transformer-based Forecasters

Transformers have become strong baselines for long-horizon forecasting. Informer introduced ProbSparse attention for long sequences [34]; Autoformer replaced dot-product attention with an auto-correlation mechanism coupled with series decomposition [30]. TimesNet recasts 1D sequences as 2D temporal variations to model multi-periodicity [29]. PatchTST popularized patch tokenization and channel-independent backbones for forecasting and representation learning [20], while iTransformer inverts the attention axes to build variate-centric representations [18]. Unified Transformer families target multiple TS tasks in one model (e.g., UniTS) [7].

5.3 Time-series Foundation Models (TSFMs)

Large, pre-trained decoders for time series have recently emerged. TimesFM is a decoder-only TSFM trained on billions of time points and demonstrates strong zero-/few-shot forecasting [8]. Chronos tokenizes real-valued series and reuses language-model training pipelines for probabilistic forecasting [2]. MOMENT releases open TSFMs for forecasting, classification, anomaly detection, and imputation [9]. Moirai-MoE and Time-MoE explore sparse mixture-of-experts to scale capacity while keeping inference cost tractable [17, 27]. In our experiments we evaluate decoder-only, autoregressive base models, consistent with this trend of TSFMs.

5.4 Inference Acceleration for Autoregressive Models

Our method adapts *speculative decoding* to continuous, patch-based forecasting. Speculative decoding uses a fast draft model to propose multiple steps which are then verified in parallel by the target model [14]. Related ideas include blockwise parallel decoding, which predicts a block and backs off to the longest verified prefix [28]. Several LLM-era variants improve acceptance or remove draft models: Medusa adds multi-head decoding to verify tree-structured drafts in one pass [3]; Lookahead performs exact, parallel n-gram verification without auxiliary models [10]; and EAGLE-2 refines draft trees and calibration for higher speedups [32]. Apollo-Forecast [31] accelerates forecasting by quantizing continuous series into discrete tokens and heuristically concatenating draft/target predictions via a tolerance-based "Race Decoding" check, but it neither redesigns speculative decoding for continuous distributions nor offers formal correctness/deviation guarantees, whereas we operate directly on continuous patch densities

with principled acceptance and provable speed/accuracy bounds. Our analysis formalizes lossless variants for continuous outputs and derives acceptance, bias, and speedup formulas specialized to time-series patch heads.

5.5 Systems Techniques for Low-latency Decoding

Kernel-level optimizations such as FlashAttention reduce IO and memory traffic in attention computation [5, 6]. KV-cache–efficient attention (MQA/GQA) lowers memory bandwidth by sharing keys and values across heads or groups [1, 26]. Serving systems like vLLM combine paged KV memory and continuous batching to raise throughput at a fixed latency [12]. Our implementation leverages these ingredients orthogonally to speculative decoding.

6 Conclusion

We studied speculative decoding as an inference-time acceleration method for time-series foundation models. On the theoretical side, we established both the lossless and practical variants, analyzed their statistical properties, and derived performance formulas for expected block length, speedup, and compute overhead. On the implementation side, we showed that the fallback-to-p practical variant is consistent with efficient batched evaluation and yields tractable estimators for acceptance. On the empirical side, experiments across diverse benchmarks and model families confirmed that speculative decoding achieves substantial wall-clock speedups with small or bounded accuracy degradation. Taken together, these results demonstrate that speculative decoding can reduce latency in multi-horizon forecasting without retraining or modifying target architectures. Future work may explore richer covariance parameterizations, system-level optimizations for deployment, and integration with other acceleration strategies. Overall, speculative decoding provides a principled and practical tool for making time-series foundation models more efficient in real-world applications.

7 Broader impact

Beyond performance improvements, our approach contributes to sustainable computing infrastructure. Large-scale web services processing billions of daily forecasts can reduce their computational footprint by 30–50%, translating to significant energy savings and carbon emission reductions. This is particularly impactful for global platforms operating thousands of servers continuously. Additionally, by making advanced forecasting models practical within tight latency budgets, speculative decoding democratizes access to state-of-the-art capabilities for smaller web services that previously could not afford the computational overhead.

However, deployment in production web systems requires careful consideration of potential risks. Distribution shifts—common in web traffic due to viral content, breaking news, or seasonal patterns—may degrade acceptance rates and model reliability. Rare but critical events (e.g., DDoS attacks, flash sales) could be missed if efficiency is prioritized too aggressively. We recommend comprehensive monitoring of acceptance rates $\bar{\alpha}$ across different traffic segments, implementing adaptive thresholds that become more conservative during anomalous periods, and maintaining "golden path" validation where a fraction of predictions bypass acceleration for quality assurance. For security-critical applications like fraud detection, conservative acceptance thresholds ($\alpha < 0.8$) and dedicated anomaly detection pipelines should supplement speculative decoding.

References

[1] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[2] Abdul Fatir Ansari, Caner Turkmen, Oleksandr Shchur, Lorenzo Stella, Tim Januschowski, and Jan Gasthaus. 2024. Chronos: Learning the Language of Time Series. *arXiv preprint arXiv:2403.07815* (2024).

[3] Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. 2024. Medusa: Simple LLM Inference Acceleration Framework with Multiple Decoding Heads. *arXiv preprint arXiv:2401.10774* (2024).

[4] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 191–198.

[5] Tri Dao. 2023. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. *arXiv preprint arXiv:2307.08691* (2023).

[6] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[7] Shanghua Gao, Teddy Koker, Owen Queen, Thomas Hartvigsen, Theodoros Tsiligkaridis, and Marinka Zitnik. 2024. UniTS: A Unified Multi-Task Time Series Model. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[8] Google Research. 2024. A Decoder-only Foundation Model for Time-series Forecasting. Google Research Blog. https://research.google/blog/a-decoder-only-foundation-model-for-time-series-forecasting/ Describes TimesFM pretraining and zero/few-shot performance.

[9] Mononito Goswami, Mohit Sharma, Zhe Qian, Daniel Hill, Ravi Ganti, Trisha Bansal, and et al. 2024. MOMENT: A Family of Open Time-series Foundation Models. In *Proceedings of the 41st International Conference on Machine Learning (ICML), PMLR 235*. 15157–15179.

[10] Junjie Hao, Hongyu Li, Guoxin Duan, and et al. 2023. Lookahead Decoding: Break the Sequential Dependency of LLM Inference Using Parallel n-gram Verification. *arXiv preprint arXiv:2312.12728* (2023).

[11] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, and Joaquin Quiñonero Candela. 2014. Practical Lessons from Predicting Clicks on Ads at Facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. ACM, 1–9.

[12] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. *arXiv preprint arXiv:2309.06180* (2023). System implementation: vLLM.

[13] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. *arXiv preprint arXiv:1703.07015* (2018).

[14] Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast Inference from Transformers via Speculative Decoding. In *ICML*. arXiv:2211.17192.

[15] Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*. PMLR, 19274–19286.

[16] Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. 2021. SCINet: Time Series Modeling and Forecasting with Sample Convolution and Interaction. *arXiv preprint arXiv:2106.09305* (2021).

[17] Xu Liu, Juncheng Liu, Gerald Woo, Taha Aksu, Yuxuan Liang, Roger Zimmermann, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. 2024. Moirai-MoE: Empowering Time Series Foundation Models with Sparse Mixture of Experts. *arXiv preprint arXiv:2410.10469* (2024).

[18] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2024. iTransformer: Inverted Transformers are Effective for Time Series Forecasting. In *International Conference on Learning Representations (ICLR)*.

[19] H. Brendan McMahan, Gary Holt, D. Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad Click Prediction: a View from the Trenches. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1222–1230.

[20] Yuqi Nie, Mathias Zhang, Nhut-Minh Nguyen, Dani Kiyasseh, Yuan Gao, and Xiaoxi Li. 2023. A Time Series is Worth 64 Words: Long-Term Forecasting with Transformers. In *International Conference on Learning Representations (ICLR)*. Also known as PatchTST.

[21] Erik Nygren, Ramesh K. Sitaraman, and Jennifer Sun. 2010. The Akamai Network: A Platform for High-Performance Internet Applications. *ACM SIGOPS Operating*

Systems Review 44, 3 (2010), 2–19.
[22] Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. 2020. N-BEATS: Neural Basis Expansion Analysis for Interpretable Time Series Forecasting. In *International Conference on Learning Representations (ICLR)*.
[23] Al-Mukaddim Khan Pathan and Rajkumar Buyya. 2008. A Taxonomy and Survey of Content Delivery Networks. *Grid Computing and Distributed Systems Laboratory, University of Melbourne, Technical Report* (2008).
[24] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.
[25] Matthias W. Seeger, David Salinas, and Valentin Flunkert. 2016. Bayesian Intermittent Demand Forecasting for Large Inventories. *Advances in Neural Information Processing Systems* 29 (2016), 4646–4654.
[26] Noam Shazeer. 2019. Fast Transformer Decoding: One Write-Head is All You Need. *arXiv preprint arXiv:1911.02150* (2019).
[27] Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. 2024. Time-MoE: Billion-Scale Time Series Foundation Models with Mixture of Experts. *arXiv preprint arXiv:2409.16040* (2024).
[28] Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. 2018. Blockwise Parallel Decoding for Deep Autoregressive Models. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[29] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *International Conference on Learning Representations (ICLR)*.
[30] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*.
[31] Tianyi Yin, Jingwei Wang, Yunlong Ma, Han Wang, Chenze Wang, Yukai Zhao, Min Liu, and Weiming Shen. 2025. Apollo-Forecast: Overcoming Aliasing and Inference Speed Challenges in Language Models for Time Series Forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 22173–22181.
[32] Aohan Zhang, Zhaofeng Chen, Yuxiang Wang, et al. 2024. EAGLE-2: Faster Inference of Language Models with Context-Aware Dynamic Draft Trees. *arXiv preprint arXiv:2406.16858* (2024).
[33] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1059–1068.
[34] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 11106–11115.

A Proofs and Derivations

This appendix consolidates proofs based on [14] for the lossless and practical SD variants, acceptance estimation, and speed/compute formulas.

A.1 Lossless Exactness

Assumption 1 (Absolute continuity & measurability)**.** p and q are Lebesgue densities on $(\mathbb{R}^d, \mathcal{B})$; acceptance functions are measurable in $[0,1]$; conditional draws are independent given the history.

Definition 1 (Residual and overlap)**.** The continuous overlap and residual density are defined as:

$$\beta = \int_{\mathbb{R}^d} \min\{p(x), q(x)\}\, dx = 1 - \frac{1}{2}\int_{\mathbb{R}^d} |p(x) - q(x)|\, dx \tag{10}$$

$$r(x) = \frac{(p(x) - q(x))_+}{1-\beta}, \quad \text{where } (a)_+ = \max\{a, 0\} \tag{11}$$

Note that r is well-defined if $\beta < 1$; if $p = q$ then SD is vacuous. Moreover, r is a valid density since

$$\int_{\mathbb{R}^d} r(x)\, dx = \frac{\int_{\mathbb{R}^d} (p(x) - q(x))_+\, dx}{1-\beta} = \frac{1-\beta}{1-\beta} = 1.$$

Definition 2 (Lossless SD: single step)**.** Sample $X \sim q$. Accept with probability $\alpha(X) = \min\{1, p(X)/q(X)\}$. If rejected, draw $Y^\star \sim r$ with r as in (2). Output $Y = X$ upon acceptance and $Y = Y^\star$ upon rejection.

Remark 2. The acceptance function $\alpha(x)$ is only relevant on the support of q. Values on $\{q(x) = 0\}$ are immaterial since proposals never land there.

Theorem 1 (Single-step exactness)**.** *Under Assumption 1, the output Y from the lossless procedure has density p.*

Proof. Let $A = \{\text{accept}\}$; then $\Pr(X \in dx, A) = q(x)\min\{1, p/q\}\, dx = \min\{p(x), q(x)\}\, dx$. Hence $\Pr(A) = \beta$ by the definition of β. For any Borel set B,

$$\Pr(Y \in B) = \int_B \min\{p, q\}\, dx + \Pr(A^c)\int_B r(x)\, dx \tag{12}$$

$$= \int_B \min\{p, q\}\, dx + (1-\beta)\int_B \frac{(p-q)_+}{1-\beta}\, dx \tag{13}$$

$$= \int_B \min\{p, q\}\, dx + \int_B (p-q)_+\, dx \tag{14}$$

$$= \int_B p(x)\, dx \tag{15}$$

where we used the pointwise identity $\min\{a, b\} + (a-b)_+ = a$. □

Theorem 2 (Block & autoregressive exactness)**.** *Consider a block of $\gamma \in \mathbb{N}$ draft proposals $(X_1, \ldots, X_\gamma)$ drawn autoregressively from q and validated in parallel by p on the corresponding prefixes. Let n be the number of consecutive accepts (possibly $n = \gamma$). If $n < \gamma$, draw the $(n+1)$-st output from the residual r_{n+1} formed from the pair (p_{n+1}, q_{n+1}) at the failed position; if $n = \gamma$, draw one more from $p_{\gamma+1}$. Then iterating this procedure yields the same joint law as pure target autoregression: the output sequence has density $\prod_t p_t(\cdot \mid h_t)$.*

Proof. We prove by strong induction over positions. Fix a realized history h_1.

For position $i = 1$, Theorem 1 gives the marginal p_1.

Assume the prefix $(Y_1, \ldots, Y_{i-1})$ has joint density $\prod_{j<i} p_j(\cdot \mid h_j)$. At position i, acceptance is computed from (p_i, q_i) conditional on the realized prefix; by Theorem 1, the conditional output law at i (including the first failure case via r_i) equals p_i. Hence the joint up to i is $\prod_{j \le i} p_j$.

Continue to the end of the block; if there is no failure inside, the $(\gamma+1)$-st draw is from $p_{\gamma+1}$, which is the next conditional in the target chain. Concatenating blocks yields $\prod_t p_t$. □

A.2 Practical Variant: Output Law and Bounds

A.2.1 Motivation for the Practical Variant. The lossless framework is mathematically exact, but residual sampling is computationally prohibitive in high dimensions. Each residual draw requires on average

$$\mathbb{E}[\#p\text{-draws per residual}] = \frac{1}{1-\beta}$$

where $\beta = \int \min\{p, q\}$ is the overlap. When $\beta \in [0.7, 0.95]$, this factor can be between 3 and 20, meaning one rejection may cost as many target evaluations as an entire autoregressive rollout. Thus, parallel validation speedups are often negated by residuals. This motivates the fallback-to-p variant.

Theorem 3 (Single-step law (biased kernel)). *Let $\alpha : \mathbb{R}^d \to [0,1]$ be any measurable acceptance function and $\bar{\alpha} = \int \alpha(x)q(x)\,dx$. In the practical variant, upon rejection we fallback directly to one draw from p. The output density equals*

$$g(y) = \alpha(y)q(y) + (1-\bar{\alpha})p(y) \tag{16}$$

Proof. For any Borel set A,

$$\Pr(Y \in A) = \Pr(\text{accept}, Y \in A) + \Pr(\text{reject}, Y \in A) \tag{17}$$

$$= \int_A \alpha(x)q(x)\,dx + \Pr(\text{reject}) \int_A p(x)\,dx \tag{18}$$

$$= \int_A \alpha(x)q(x)\,dx + (1-\bar{\alpha}) \int_A p(x)\,dx \tag{19}$$

$$= \int_A [\alpha(x)q(x) + (1-\bar{\alpha})p(x)]\,dx \tag{20}$$

Hence the output has density $g(y) = \alpha(y)q(y) + (1-\bar{\alpha})p(y)$. □

Corollary 1 (Deviation and TV bound). *The deviation from the target is $g - p = \alpha q - \bar{\alpha}p$, and the total variation distance satisfies*

$$\|g-p\|_{\mathrm{TV}} = \frac{1}{2}\int |\alpha q - \bar{\alpha}p|\,dx \le \bar{\alpha}$$

For bounded f with $\|f\|_\infty \le M$, we have $\left|\mathbb{E}_g[f] - \mathbb{E}_p[f]\right| \le 2M\bar{\alpha}$.

Proof. The TV distance is

$$\|g-p\|_{\mathrm{TV}} = \frac{1}{2}\int |g(x)-p(x)|\,dx = \frac{1}{2}\int |\alpha(x)q(x) - \bar{\alpha}p(x)|\,dx \tag{21}$$

To bound this, note that $\int \alpha(x)q(x)\,dx = \bar{\alpha}$ and $\int p(x)\,dx = 1$. The integrand satisfies

$$|\alpha(x)q(x) - \bar{\alpha}p(x)| \le \alpha(x)q(x) + \bar{\alpha}p(x)$$

Hence

$$\|g-p\|_{\mathrm{TV}} \le \frac{1}{2}\left(\int \alpha(x)q(x)\,dx + \bar{\alpha}\int p(x)\,dx\right) = \frac{1}{2}(\bar{\alpha}+\bar{\alpha}) = \bar{\alpha}$$

□

Proposition 2 (One-sided KL & Pinsker). *Assume g and p are mutually absolutely continuous and that $\alpha(x)q(x) = 0$ whenever $p(x) = 0$, so that the ratio $\frac{\alpha q}{\bar{\alpha}p}$ is well-defined almost everywhere. Then*

$$\mathrm{KL}(g\|p) = \int g \log\frac{g}{p}\,dx \;\le\; \int \alpha q \log\frac{\alpha q}{\bar{\alpha}p}\,dx,$$

and by Pinsker's inequality,

$$\|g-p\|_{\mathrm{TV}} \;\le\; \sqrt{\tfrac{1}{2}\,\mathrm{KL}(g\|p)}.$$

Proof. Using the mixture form $g = \alpha q + (1-\bar{\alpha})p$ and the convexity of $x\log x$:

$$\mathrm{KL}(g\|p) = \int g\log\frac{g}{p}\,dx \tag{22}$$

$$= \int [\alpha q + (1-\bar{\alpha})p]\log\frac{\alpha q + (1-\bar{\alpha})p}{p}\,dx \tag{23}$$

$$\le \int \alpha q\log\frac{\alpha q}{\bar{\alpha}p}\,dx + \int (1-\bar{\alpha})p\log\frac{(1-\bar{\alpha})p}{(1-\bar{\alpha})p}\,dx \tag{24}$$

$$= \int \alpha q\log\frac{\alpha q}{\bar{\alpha}p}\,dx \tag{25}$$

Pinsker's inequality is standard. □

Theorem 4 (Autoregressive composition). *Let $h_t = (y_{<t}, \text{covariates})$ and define the transition kernel*

$$K_t(h_t, dy) = \alpha_t(y \mid h_t)q_t(y \mid h_t)\,dy + (1-\bar{\alpha}_t(h_t))p_t(y \mid h_t)\,dy$$

where $\bar{\alpha}_t(h_t) = \int \alpha_t(y \mid h_t)q_t(y \mid h_t)\,dy$. Then the generated sequence has joint density

$$G(y_{1:T}) = \prod_{t=1}^{T} K_t(h_t, dy_t)$$

Proof. By conditioning on h_t and applying the single-step result at each position, then multiplying the conditionals. □

A.3 Choosing the Block Size γ

Proposition 3 (Near-optimality condition). *The speedup $S_{\text{wall}}(\gamma)$ increases with γ if and only if*

$$\bar{\alpha}^{\gamma+1} \ge \frac{1+c\gamma}{1+c(\gamma+1)}$$

Thus a near-optimal $\gamma^\star$ is the largest integer satisfying this inequality.

Proof. Consider the discrete increment $\Delta S(\gamma) = S(\gamma+1) - S(\gamma)$. Computing:

$$\Delta S(\gamma) = \frac{1-\alpha^{\gamma+2}}{(1-\alpha)(c(\gamma+1)+1)} - \frac{1-\alpha^{\gamma+1}}{(1-\alpha)(c\gamma+1)} \tag{26}$$

Finding a common denominator and simplifying the numerator:

$$= \frac{(1-\bar{\alpha}^{\gamma+2})(c\gamma+1) - (1-\bar{\alpha}^{\gamma+1})(c(\gamma+1)+1)}{(1-\bar{\alpha})(c\gamma+1)(c(\gamma+1)+1)} \tag{27}$$

$$= \frac{\bar{\alpha}^{\gamma+1}(1+c(\gamma+1)) - (1+c\gamma)}{(1-\bar{\alpha})(c\gamma+1)(c(\gamma+1)+1)} \tag{28}$$

Since the denominator is positive, $\Delta S(\gamma) \ge 0$ if and only if the stated condition holds. □

Remark 3 (Practical implementation). In practice, exact computation of $\bar{\alpha}$ and c is infeasible. The quantity $\bar{\alpha} = \int \min(p,q)$ requires integrating over high-dimensional space with no closed form for Gaussian heads. Similarly, c as a FLOPs ratio ignores kernel scheduling and memory effects. Therefore, both are estimated empirically: $\bar{\alpha}$ via sampling and c via wall-clock timing.

A.4 Acceptance Estimation and Concentration

For a fixed history h, define the overlap

$$\beta(h) = \int_{\mathbb{R}^d} \min\{p(y \mid h), q(y \mid h)\}\,dy = 1 - \frac{1}{2}\int_{\mathbb{R}^d} |p-q|\,dy$$

Let the deployment-averaged (mean) acceptance be

$$\bar{\alpha} := \mathbb{E}_h[\beta(h)] \in [0,1].$$

Proposition 4 (Unbiased plug-in estimator). *Given histories $h_1, \ldots, h_N$ and models (p,q), define*

$$\hat{\bar{\alpha}}_N = \frac{1}{N}\sum_{i=1}^{N}\beta(h_i) = \frac{1}{N}\sum_{i=1}^{N}\int \min\{p(y \mid h_i), q(y \mid h_i)\}\,dy.$$

Then $\mathbb{E}[\hat{\bar{\alpha}}_N] = \bar{\alpha}$, i.e., $\hat{\bar{\alpha}}_N$ is unbiased for the mean acceptance.

In general $d > 1$, the inner integral has no simple closed form; in practice one uses high-precision quadrature or Monte Carlo sampling (e.g. drawing from $q(\cdot \mid h_i)$ and averaging $\min\{1, p/q\}$).

Proof. By linearity of expectation,

$$\mathbb{E}[\hat{\bar{\alpha}}_N] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[\beta(h_i)] = \mathbb{E}_h[\beta(h)] = \bar{\alpha}.$$

□

Theorem 5 (Hoeffding concentration)**.** *Let $\beta_i = \beta(h_i) \in [0,1]$ and $\hat{\bar{\alpha}}_N = N^{-1}\sum_{i=1}^{N}\beta_i$. If h_i are i.i.d., then for any $\varepsilon > 0$,*

$$\Pr\left(\left|\hat{\bar{\alpha}}_N - \bar{\alpha}\right| \geq \varepsilon\right) \leq 2\exp\left(-2N\varepsilon^2\right).$$

Consequently, $\hat{\bar{\alpha}}_N \to \bar{\alpha}$ almost surely (strong law), and $\sqrt{N}\left(\hat{\bar{\alpha}}_N - \bar{\alpha}\right)$ is asymptotically normal under standard CLT conditions.

Corollary 2 (Consistency of speed/compute predictors)**.** *Define plug-in predictors by replacing $\bar{\alpha}$ with $\hat{\bar{\alpha}}_N$:*

$$\widehat{\mathbb{E}[L]}(\gamma) = \frac{1-\hat{\bar{\alpha}}_N^{\gamma+1}}{1-\hat{\bar{\alpha}}_N}, \quad \widehat{S_{\text{wall}}}(\gamma) = \frac{1-\hat{\bar{\alpha}}_N^{\gamma+1}}{(1-\hat{\bar{\alpha}}_N)(c\gamma+1)},$$

$$\widehat{\text{OpsFactor}}(\gamma) = \frac{(1-\hat{\bar{\alpha}}_N)(\gamma\hat{c}+\gamma+1)}{1-\hat{\bar{\alpha}}_N^{\gamma+1}}.$$

Then these estimators are consistent by the continuous mapping theorem.

A.5 Computational Cost of Residual Sampling

Two exact constructions illustrate the computational burden in high dimension:

A.5.1 Thinning from p. Draw $Z \sim p$; accept with probability $\pi(Z) = (1 - q(Z)/p(Z))_+$. Since $\mathbb{E}_p[\pi(Z)] = 1 - \beta$ for the overlap $\beta = \int \min\{p, q\}$, the accepted Z has density r. Thus the expected number of target draws per residual sample is

$$\mathbb{E}[\#p\text{-draws for one residual}] = \frac{1}{1-\beta}, \tag{29}$$

which deteriorates as $q \to p$ (i.e. large β).

A.5.2 Classical rejection sampling from q. If $M \geq \sup_x p(x)/q(x)$, accept $X \sim q$ with probability $p(X)/(Mq(X))$. For continuous p, q, a finite, tight M may be unknown or infinite in practice.

Remark 4 (Block-level implication)**.** Residual sampling is required only at the first failed position. Each such draw costs $\approx 1/(1-\beta)$ target samples on average. For $\beta \in [0.7, 0.95]$, this factor ranges from 3 to 20, which can negate parallel-validation speedups. This motivates our practical fallback-to-p variant.

A.6 Summary of Two Variants

- **Lossless SD** (Theorems 1 and 2): Reproduces the target distribution exactly but requires residual sampling. The expected residual cost $1/(1-\beta)$ can be prohibitively large in high-dimensional settings, limiting practical use.
- **Practical SD** (Theorems 3–7): Replaces residual sampling by a direct fallback-to-p draw upon rejection. This yields an output law with total variation deviation bounded by $\bar{\alpha}$, while maintaining tractable efficiency. The output law, deviation bounds, and performance formulas have been fully characterized.

Algorithm 2 Lossless Variant with Residual Sampling (for completeness)

Require: $p(\cdot \mid \cdot), q(\cdot \mid \cdot), \gamma, H$

Draft proposals $x_{1:\gamma} \sim q$ autoregressively; compute $p_{1:\gamma+1}$ in parallel
for $i = 1, \ldots, \gamma$ **do**
  Accept x_i with $\alpha_i = \min\{1, p_i(x_i)/q_i(x_i)\}$; stop at first rejection (index n+1)
end for
if no rejection **then**
  Draw $t \sim p_{\gamma+1}$; **return** $[x_{1:\gamma}, t]$
else
  Draw $t \sim r_{n+1}$ formed from (p_{n+1}, q_{n+1}); **return** $[x_{1:n}, t]$
end if

Given our focus on accelerating high-dimensional continuous time series generation, the implementation adopts the practical speculative decoding variant.

B Consistency and Convergence Guarantees for Speculative Decoding

We formalize single–step and multi–step convergence properties of speculative decoding (SD). We treat both the *lossless* (exact) variant and the *practical* fallback-to-p variant, give uniform deviation bounds over a horizon, and provide finite-sample guarantees for estimating the mean acceptance needed to predict throughput.

Standing assumptions. All densities are with respect to Lebesgue measure on $(\mathbb{R}^d, \mathcal{B})$. For each step t, target and draft conditionals $p_t(\cdot \mid h_t)$ and $q_t(\cdot \mid h_t)$ are Lebesgue densities, and acceptance functions $\alpha_t(\cdot \mid h_t) \in [0,1]$ are measurable and only evaluated on $\{q_t > 0\}$. We write

$$\bar{\alpha}_t(h_t) := \int \alpha_t(y \mid h_t)\, q_t(y \mid h_t)\, dy,$$

$$K_t(h_t, dy) := \alpha_t(y \mid h_t)\, q_t(y \mid h_t)\, dy + \left(1 - \bar{\alpha}_t(h_t)\right) p_t(y \mid h_t)\, dy.$$

for the practical one-step kernel. When $\alpha_t = \min\{1, p_t/q_t\}$ (defined q_t-a.e.), we have $\alpha_t\, q_t = \min\{p_t, q_t\}$, and $\bar{\alpha}_t(h_t) = \int \min\{p_t, q_t\}$.

B.1 Lossless variant: exactness

Theorem 6 (Single-step and blockwise exactness)**.** *Fix a realized history h_t. In the lossless SD step, propose $X \sim q_t(\cdot \mid h_t)$ and accept with probability $\min\{1, p_t(X \mid h_t)/q_t(X \mid h_t)\}$; upon rejection, draw from the residual density $r_t(y \mid h_t) \propto (p_t - q_t)_+(y \mid h_t)$. The output has density $p_t(\cdot \mid h_t)$. For a block of γ proposals validated in parallel on the corresponding prefixes, drawing from the residual at the* first *failed position (conditional on the realized accepted prefix) and iterating blockwise reproduces the exact target chain $\prod_{t=1}^{T} p_t(\cdot \mid h_t)$.*

B.2 Practical variant: per-step deviation and continuity

Proposition 5 (Single-step output law, TV and KL bounds)**.** *For any measurable $\alpha_t \in [0,1]$,*

$$K_t(h_t, dy) = \left[\alpha_t(y \mid h_t) q_t(y \mid h_t) + (1 - \bar{\alpha}_t(h_t))\, p_t(y \mid h_t)\right] dy.$$

Hence

$$\big\|K_t(\cdot \mid h_t) - p_t(\cdot \mid h_t)\big\|_{\mathrm{TV}} = \tfrac{1}{2}\int \big|\alpha_t q_t - \bar{\alpha}_t p_t\big| \;\le\; \bar{\alpha}_t(h_t),$$

$$\mathrm{KL}\big(K_t \,\|\, p_t\big) \le \int \alpha_t q_t \log\frac{\alpha_t q_t}{\bar{\alpha}_t p_t} \;-\; \bar{\alpha}_t(h_t)\,\log\big(1-\bar{\alpha}_t(h_t)\big).$$

In particular,

$$\|K_t - p_t\|_{\mathrm{TV}} \;\le\; \min\left\{\bar{\alpha}_t,\; \sqrt{\tfrac{1}{2}\,\mathrm{KL}(K_t\|p_t)}\right\}.$$

Proposition 6 (Continuity in q_t under the canonical acceptance)**.** *Let $\alpha_t = \min\{1, p_t/q_t\}$ so that $\alpha_t q_t = \min\{p_t, q_t\}$ and $\bar{\alpha}_t = \int \min\{p_t, q_t\}$. Then*

$$K_t(\cdot \mid h_t) = \min\{p_t, q_t\} + (1-\bar{\alpha}_t)\, p_t.$$

If $\|q_t(\cdot \mid h_t) - p_t(\cdot \mid h_t)\|_{L^1} \to 0$, then $\bar{\alpha}_t(h_t) \to 1$ and

$$\|K_t(\cdot \mid h_t) - p_t(\cdot \mid h_t)\|_{L^1} \to 0,$$

i.e., the practical kernel converges to the target kernel in total variation as the draft head converges to the target head.

Remark 5 (Equal-covariance Gaussian heads: closed-form overlap)**.** If $p_t = \mathcal{N}(\mu_p, \Sigma)$ and $q_t = \mathcal{N}(\mu_q, \Sigma)$ share the same covariance, the overlap $\bar{\alpha}_t = \int \min\{p_t, q_t\}$ admits the closed form

$$\bar{\alpha}_t \;=\; 2\,\Phi\!\left(-\frac{\Delta}{2}\right), \qquad \Delta^2 = (\mu_p - \mu_q)^\top \Sigma^{-1}(\mu_p - \mu_q),$$

where Φ is the standard normal cdf. Thus $\bar{\alpha}_t \to 1$ as the Mahalanobis gap $\Delta \to 0$, which, by Proposition 6, implies $K_t \to p_t$ in total variation.

B.3 Horizon-wise guarantees

Proposition 7 (Uniform multi-horizon deviation)**.** *Let $\delta_t := \sup_{h_t} \|K_t(\cdot \mid h_t) - p_t(\cdot \mid h_t)\|_{\mathrm{TV}}$. Then for any horizon T,*

$$\Big\|\mathrm{Law}_{\mathrm{SD}}(Y_{1:T}) - \mathrm{Law}_{\mathrm{target}}(Y_{1:T})\Big\|_{\mathrm{TV}} \;\le\; 1 - \prod_{t=1}^{T}(1-\delta_t) \;\le\; \sum_{t=1}^{T}\delta_t.$$

In particular, if $\sup_t \delta_t \le \delta < 1$, then the joint deviation is $O(T\delta)$ for small δ, and if $\delta_t \to 0$ uniformly in t (e.g., as the draft heads converge to the target heads), then the joint deviation over any fixed T vanishes.

Proof. Couple the two chains stepwise so that they agree at step t with probability at least $1-\delta_t$; the probability of agreement through T steps is at least $\prod_{t=1}^{T}(1-\delta_t)$. Taking complements and applying the union bound gives the inequalities. □

B.4 Acceptance, throughput prediction, and finite-sample concentration

The mean acceptance at step t and history h_t equals $\bar{\alpha}_t(h_t) = \mathbb{E}_{X\sim q_t(\cdot|h_t)}[\alpha_t(X \mid h_t)]$. In deployment we estimate the *population* mean acceptance $\bar{\alpha}_t := \mathbb{E}_{h_t}[\bar{\alpha}_t(h_t)]$ from held-out histories and Monte Carlo within each history.

Proposition 8 (Two-stage mean-acceptance estimator with concentration)**.** *Given histories $h_1, \ldots, h_N$ and, for each i, i.i.d. samples $X_{ij} \sim q_t(\cdot \mid h_i)$ $(j = 1, \ldots, m)$, define*

$$\hat{\beta}_i := \frac{1}{m}\sum_{j=1}^{m} \alpha_t(X_{ij} \mid h_i), \qquad \hat{\bar{\alpha}}_{N,m} := \frac{1}{N}\sum_{i=1}^{N}\hat{\beta}_i.$$

Then $\mathbb{E}[\hat{\bar{\alpha}}_{N,m}] = \bar{\alpha}_t$ and, for any $\varepsilon > 0$,

$$\Pr\big(\big|\hat{\bar{\alpha}}_{N,m} - \bar{\alpha}_t\big| \ge \varepsilon\big) \;\le\; 2\exp\big(-2Nm\,\varepsilon^2\big).$$

Proof. Each $\alpha_t(X_{ij} \mid h_i) \in [0,1]$; apply Hoeffding to the Nm bounded i.i.d. terms and average. □

Corollary 3 (Consistency of plug-in predictors)**.** *Let $\widehat{\mathbb{E}[L]}$, $\widehat{S_{\mathrm{wall}}}$, and $\widehat{\mathrm{OpsFactor}}$ be the block-length, speedup, and FLOPs predictors obtained by substituting $\hat{\bar{\alpha}}_{N,m}$ for $\bar{\alpha}$ in the closed-form expressions. By Proposition 8 and the continuous mapping theorem, these predictors are consistent as $Nm \to \infty$.*

B.5 Dependence within a block

Acceptance events across a block are generally dependent. Suppose there exist $0 \le \underline{\alpha} \le \overline{\alpha} \le 1$ such that

$$\underline{\alpha} \le \Pr(A_i \mid A_1, \ldots, A_{i-1}) \le \overline{\alpha}, \qquad i = 1, \ldots, \gamma.$$

Then the expected block output length L satisfies

$$\frac{1-\underline{\alpha}^{\gamma+1}}{1-\underline{\alpha}} \;\le\; \mathbb{E}[L] \;\le\; \frac{1-\overline{\alpha}^{\gamma+1}}{1-\overline{\alpha}},$$

which interpolates between the i.i.d. geometric case and an adversarially dependent case. This bound preserves the qualitative saturation of $\mathbb{E}[L]$ in γ.

B.6 When is the lossless variant worthwhile?

The lossless residual draw at the first rejection has expected target-draw cost

$$\mathbb{E}[\#p\text{-draws per residual}] \;=\; \frac{1}{1-\bar{\alpha}_t},$$

where $\bar{\alpha}_t = \int \min\{p_t, q_t\}$ at the failed position. In high-acceptance regimes with $\bar{\alpha}_t \uparrow 1$, this factor can dominate the block's parallel-validation savings. A crude breakeven heuristic is that lossless SD is only competitive when

$$1 - \bar{\alpha}_t \;\gtrsim\; \frac{1}{\gamma},$$

so that the expected residual cost per block does not exceed the expected accepted outputs from a block of size γ.

B.7 Implementation note on batching

Throughout, "a single batched target pass" refers to evaluating the target on the γ+1 prefixes in *one wall-clock pass* (e.g., via prefix batching and KV reuse), which still accounts for γ+1 target-forward *equivalents* in FLOPs.

The guarantees above establish: (i) exact recovery of the target chain for the lossless variant; (ii) for the practical variant, per-step deviation bounds (TV and KL), continuity in the draft–target gap, and horizon-wise control via a product/union bound; and (iii) statistically consistent estimation of mean acceptance for reliable throughput prediction.

C Ethical Use of Data and Informed Consent

This research on accelerating time series foundation models through speculative decoding involves only computational and algorithmic contributions without human participants or personally identifiable information (PII). We confirm compliance with all ACM Publications Policies, including the 2021 Publications Policy on Research Involving Human Participants and Subjects.

Nature of the research. Our work is purely algorithmic, focusing on inference-time acceleration techniques for pre-trained time series models. The research involves: (i) theoretical analysis of speculative decoding algorithms; (ii) implementation of acceleration methods; and (iii) computational experiments on existing models and datasets. No human subjects participated in any aspect of this research.

Data sources and usage. All experiments use publicly available benchmark datasets widely adopted in the time series forecasting literature:

- **ETT datasets (ETTh1, ETTh2, ETTm2):** Electricity Transformer Temperature data, publicly released for academic research
- **Weather dataset:** Meteorological observations from weather stations, containing no personal information

These datasets contain only sensor measurements and temporal indicators without any PII or sensitive information. We used these datasets strictly according to their intended research purposes and existing academic conventions.

Model usage. We evaluated our methods on pre-trained foundation models (Timer and Timer-XL) that were trained on public data and released for research use. No new training was performed, and no proprietary or sensitive data was used in model development or evaluation.

Ethical considerations. While our research does not raise direct ethical concerns regarding human subjects or data privacy, we acknowledge broader considerations: (i) improved forecasting efficiency could benefit applications in energy management, healthcare monitoring, and climate modeling; (ii) acceleration techniques reduce computational costs and environmental impact; (iii) open evaluation on standard benchmarks promotes reproducible research.

Compliance statement. We confirm that this research complies with all applicable ACM policies and does not involve human participants, PII collection, or ethically sensitive data usage.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009