ViTime: Foundation Model for Time Series Forecasting Powered by Vision Intelligence

Anonymous authors
Paper under double-blind review

Abstract

Time series forecasting (TSF) possesses great practical values in various fields, including power and energy, transportation, etc. TSF methods have been studied based on knowledge from classical statistics to modern deep learning. Yet, all of them were developed based on one fundamental concept, the numerical data fitting. Thus, the models developed have been long known for being problem-specific and lacking application generalizability. Practitioners expect a TSF foundation model that serves TSF tasks in different applications. The central question is then how to develop such a TSF foundation model. This paper offers one pioneering study in the TSF foundation model development method and proposes a vision intelligence-powered framework, ViTime, for the first time. ViTime fundamentally shifts TSF from numerical fitting to operations based on a binary image-based time series metric space. We also provide rigorous theoretical analyses of ViTime, including quantization-induced system error bounds and principled strategies for optimal parameter selection. Furthermore, we propose RealTS, an innovative synthesis algorithm generating diverse and realistic training samples, effectively enriching the training data and significantly enhancing model generalizability. Extensive experiments demonstrate ViTime's SOTA performance. In zero-shot scenarios, ViTime outperforms TimesFM by 9-15%. With just 10% fine-tuning data, ViTime surpasses both leading foundation models and fully-supervised benchmarks, a gap that widens with 100% fine-tuning. ViTime also exhibits exceptional robustness, effectively handling missing data and outperforming TimesFM by 20-30% under various data perturbations, validating the power of its visual space data operation paradigm.

1 Introduction

Time series forecasting (TSF) is a classic but challenging topic that has been vigorously discussed in various application fields, including power and energy (Sharadga et al., 2020), environmental studies (Jacox et al., 2022), transportation studies (Lei et al., 2022), weather forecasting (Yang et al., 2021), stock market analysis (Lin et al., 2011), public healthcare (Liu et al., 2024). Although new heights of accuracies were repeatedly refreshed by new studies (Zhou et al., 2021; Wu et al., 2021; Nie et al., 2022; Zeng et al., 2023; Patro & Agneeswaran, 2024; Wu et al., 2022), most reported methods predominantly relied on a numerical fitting based modeling paradigm so that models were often dataset- or problem-specific and lack of application generalizability. The need to repeatedly train models for various TSF tasks has been the critical barrier of promoting applications of learning-based TSF methods in practice, especially ones with sophisticated mechanisms. Developing a TSF foundation model capable of serving diverse TSF tasks across different applications is thus of great practical value. The central question then becomes: *how can we develop such a TSF foundation model?*

Studying the TSF foundation model is still in its early stages, and existing efforts observed in literature are mainly devoted to exploring LLM-based and numerical fitting-based models. The LLM-based model leverages the inference capabilities of LLMs for zero-shot TSF tasks, including TimeGPT-1 (Garza & Mergenthaler-Canseco, 2023) and TIME-LLM (Jin et al., 2023). However, the prediction accuracy of LLM-based models heavily depends on the underlying capabilities of LLM, and to achieve optimal performance, the competent large language models, such as GPT-4 or Claude 3.5 (Zhou et al., 2023a), are usually employed. Meanwhile,

in fine-tuning LLM-based TSF foundation models for handling various downstream tasks demanding higher precision, the computational complexity becomes prohibitively expensive, resulting in a large, redundant, less precise, and price-unfriendly paradigm for the TSF foundation model (Tan et al., 2024).

The numerical fitting-based models are trained by being directly fit into numerical time series data, which manifests that the primary information concerned by these models is the numerical correlation along the temporal dimension, e.g., TimesFM (Das et al., 2024), ForecastPFN (Dooley et al., 2024), etc. In contrast, human cognition tends to conjecture trends through remembering correlations between visual representations rather than processing numerical values directly. Studies have shown that the human brain processes visual information more efficiently than numerical data. Pettersson (Pettersson, 1993) discovered that the human brain is more adept at processing visual information than numerical data. Similarly, Dondis (Dondis, 1974) demonstrated that the visual cortex rapidly identifies patterns, shapes, and colors, making processing images and videos faster than texts and numbers. These findings lead to a hypothetical question: *On the path toward the TSF foundation model, might employing vision intelligence for time series modeling be a very effective option besides conventional numerical methods?*

In addition, training data of TSF tasks typically consist of large-scale real-world datasets (Das et al., 2024), raising a critical question: *Can real-world datasets comprehensively capture the diverse range of universal time series patterns?* Specifically, what kind of foundational capabilities should a TSF foundation model possess to address a universal spectrum of time series problems?

To tackle these challenges, this paper develops a novel vision intelligence-based TSF foundation model, a Visual Time Foundation Model (ViTime), aiming to pioneer a new computational paradigm of building the TSF foundation model from the perspective of vision intelligence. Regarding the computational principle innovation aspect, ViTime operates by transforming numerical time series into binary images, converting numerical temporal correlations into pixel spatial patterns, and solving TSF tasks in binary image space. We provide detailed theoretical analyses of quantization-induced errors and establish principled guidelines for optimal parameter settings, ensuring precise control over the trade-off between computational complexity and prediction accuracy. To offer a large volume of sufficiently diverse samples for training ViTime, an innovative time-series-data generation method, Real-Time Series (RealTS), is proposed. RealTS categorizes foundational knowledge of time series analysis into "trend" and "periodicity" and synthesizes training data during the training of ViTime, ensuring it captures essential time series characteristics. Experimental results demonstrate that ViTime can achieve SOTA performance across diverse scenarios, including zero-shot generalization, fine-tuning with limited data, and robustness to data perturbations.

The main contributions of this work are listed as follows:

- **Novel Theoretical Framework for Vision Intelligence Powered TSF.** We introduce ViTime, a pioneering TSF foundation model grounded in a novel theoretical framework that shifts from conventional numerical fitting to operations within a **formally defined binary image-based time series metric space**.

- **RealTS: Advanced Data Generation and Augmentation for TSF Foundation Modeling.** To address the training-data sample diversity challenge in developing a TSF foundation model, the RealTS, a sophisticated time-series data generation method that synthesizes diverse and high-quality training data, is designed to ensure ViTime can generalize to a wide range of time series patterns.

- **Empirical Validation of Theoretical Advantages and SOTA Performance.** The efficacy of ViTime's theoretically-grounded visual intelligence paradigm is extensively validated. ViTime significantly outperforms existing foundation models and supervised benchmarks in zero-shot generalization (e.g., 9-15% improvement over TimesFM), few-shot fine-tuning, and robustness against diverse data perturbations (e.g., 20-30% better than TimesFM with missing data/perturbations), confirming the practical benefits of our theoretical contributions.

2 Related work

2.1 Problem-specific model for TSF

The problem-specific TSF methods adopt a fully supervised learning paradigm, where specific models are trained on particular datasets. Early discussions on problem-specific TSF modeling were mainly conducted on classical statistical and machine learning models, such as autoregressive (AR) models and AR variants (Vu, 2007), Splines and their extensions (Lewis & Stevens, 1991), linear regressors (Montgomery et al., 2015), support vector regressor (Montgomery et al., 2015), neural network based regressor (Montgomery et al., 2015), etc. In comparison, the latest TSF studies have shed light on modern deep learning methods, such as recurrent neural network (RNN) and RNN variants (Hewamalage et al., 2021), transformer and various former-based models (Zhou et al., 2021; Wu et al., 2021; Nie et al., 2022; Liu et al., 2023), Dlinear (Zeng et al., 2023), TimeMixer Wang et al. (2024), Mamba based method Patro & Agneeswaran (2024) etc.

2.2 Foundation model for TSF

Inspired by recent breakthroughs of pretrained foundation models in natural language processing and computer vision, the TSF community has actively explored developing domain-general foundation models capable of forecasting across diverse datasets and scenarios. Current TSF foundation model studies in general fall into three categories, LLM-based, real-data-based, and alternative-data-sources-based.

Several recent studies have directly adapted LLMs to forecasting tasks. Methods such as PromptCast (Xue & Salim, 2023), TIME-LLM (Jin et al., 2023), GPT4TS (Zhou et al., 2023b), TimeGPT-1 (Garza & Mergenthaler-Canseco, 2023), and LLM4TS (Chang et al., 2025) recast numerical forecasting into text-based prompting or embedding alignment tasks. Despite their promising zero-shot forecasting capabilities, these models suffer from inherent limitations, including high computational costs, inefficiency, and domain adaptation complexity arising from fundamental discrepancies between linguistic structures and numerical temporal patterns (Tan et al., 2024).

To address these limitations, another prevalent research direction exploits large-scale collections of real-world numerical time series to train foundation models. Representative methods include TimesFM(Das et al., 2024), Moirai (Woo et al., 2024), Chronos Ansari et al. (2024), Moment (Goswami et al., 2024), GTT (Feng et al., 2024), and TSMamba (Ma et al., 2024). Although these real-data-based models significantly enhance zero-shot generalization, their performance heavily depends on the quality, diversity, and representativeness of available real datasets. Moreover, they typically suffer substantial performance degradation when encountering data perturbations, missing values, or unseen temporal patterns. Furthermore, the reliance on extensive real-world datasets inherently risks test set leakage, as partial segments of test data may inadvertently appear during training, undermining true generalization evaluation.

Recognizing these inherent limitations of real-world numerical data, recent work has explored alternative data sources to enhance generalization. ForecastPFN (Dooley et al., 2024) trains Transformer-based models purely on synthetic numerical data generated from predefined trend and seasonality components, demonstrating limited but promising zero-shot forecasting abilities. However, due to the uncontrolled or oversimplified synthesis patterns, these synthetic-data-based methods often fail to capture the richness and complexity of real-world scenarios, thereby limiting forecasting accuracy and robustness. Recently, VisionTS (Chen et al., 2024) proposed repurposing pretrained vision models (specifically, masked autoencoders trained on ImageNet) for TSF by reformulating forecasting as an image reconstruction problem. Nevertheless, directly reusing models pretrained on natural images introduces a significant domain mismatch; the visual features learned from natural images may not optimally represent temporal structures inherent in numerical time series. Furthermore, VisionTS still fundamentally relies on numerical-space analyses and empirical mappings, lacking a rigorous theoretical framework explicitly tailored for visual representation and quantization of numerical sequences.

In contrast to aforementioned paradigms, our proposed ViTime framework introduces two fundamental shifts in the TSF foundation model design:

Firstly, recognizing intrinsic limitations of numerical-space-based forecasting—such as poor generalization across scales and sensitivity to data perturbations—ViTime explicitly advocates modeling time series directly in visual representation space. ViTime rigorously defines a dedicated visual space for numerical time series, provides theoretical analysis of quantization-induced errors, and offers principled guidance for optimal parameter selection. We also proved that this rigorous visual modeling framework can significantly enhance signal-to-noise ratio (SNR) of time series and improve forecasting accuracy and interpretability.

Secondly, given the inherent challenges of relying on real-world numerical datasets (limited diversity, data leakage risks), we propose, RealTS, a controlled data synthesis strategy focusing on fundamental time series components (trend, periodicity) to generate structurally sound training data. The RealTS substantially mitigates data leakage risks and enriches training data diversity, enabling ViTime to generalize robustly across diverse real-world scenarios. As demonstrated by extensive experiments, ViTime sets new SOTA zero-shot and limited-data forecasting benchmarks, significantly outperforming existing foundation models across diverse evaluation settings.

3 Method

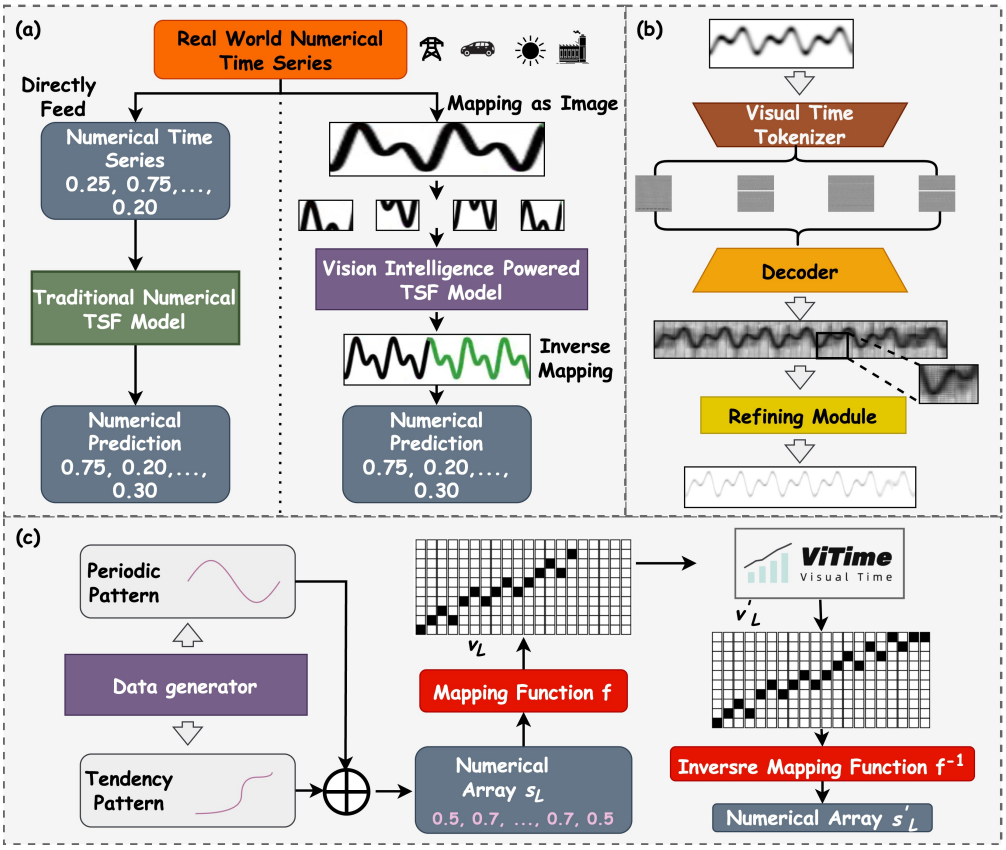


Figure 1: ViTime architecture overview. (a) Pipeline comparison between ViTime and traditional numerical TSF models, showing ViTime's paradigm shift to binary image space processing. (b) ViTime network with three modules: Visual Time Tokenizer, Decoder, and Refining Module. (c) Complete architecture with four components: RealTS synthesis for diverse training samples, mapping function for numerical-to-binary conversion, ViTime model for visual pattern learning, and inverse mapping for prediction output, enabling zero-shot generalization across real-world time series tasks.

3.1 Overall architecture

The overall framework of ViTime, schematically illustrated in Fig. 1 (c), comprises four key modules: the RealTS synthesis module, the mapping function, the proposed ViTime model, and the inverse mapping function. To address the dataset challenge of training a robust TSF foundation model, RealTS synthesizes a vast and diverse set of training samples by categorizing foundational knowledge of time series analysis into "trend" and "periodicity" patterns, which ensures ViTime captures essential time series characteristics across a wide range of scenarios. The core innovation of ViTime lies in its computational principle of mapping numerical time series into binary images. This approach allows ViTime to remember temporal pattern correlations through ordered pixel coordinates while maintaining the ability to convert results back to numerical format. The visual modeling process of ViTime learns to extract relevant features and patterns from the time series visual representation, utilizing the historical distributions of the generated binary images to predict future trends. Finally, the inverse mapping function is employed to convert the predicted image back into numerical time series data for further analysis.

In the following sections, we will introduce each component of ViTime in detail: RealTS, mapping & inverse mapping function, and ViTime Model.

3.2 Real time series synthesis

In this paper, we hypothesize that a robust foundation model for TSF should integrate two essential types of time series fluctuation knowledge, the periodic and trend patterns, which encompass the inherent patterns and directional changes in time series data. Real-world datasets, however, often lack representation of the full spectrum of these periodic and trend-based fluctuations, limiting the ability of the model to generalize across different scenarios and effectively learn underlying dynamics.

To address this challenge, we propose a novel time series generation algorithm, RealTS. RealTS systematically generates a large volume of synthetic time series data that exhibit diverse periodic and trend characteristics. The proposed RealTS can facilitate more comprehensive training of foundation models, exposing them to various patterns and improving their ability to generalize to unseen real-world data.

The RealTS algorithm probabilistically selects between generating periodic or trend-based time series. Given the total length L of the synthesized time series, the algorithm determines the data prior hypothesis between periodic φ_p and trend-based φ_t patterns with probability (α). The distribution of generated time series $P(D)$ is defined as follows:

$$\begin{aligned} &\mathbf{s_L} \sim P(D) = P\left(\mathbf{s_L}|L\right) \\ &= \alpha \int P\left(\mathbf{s_L}|L, B_p\right) P\left(B_p|\varphi_p\right) P\left(\varphi_p\right) d\varphi_p + (1-\alpha) \int P\left(\mathbf{s_L}|L, B_t\right) P\left(B_t|\varphi_t\right) P\left(\varphi_t\right) d\varphi_t \end{aligned} \tag{1}$$

where $\mathbf{s_L}$ is the synthesized time series with length L; $P(\varphi)$ represents the prior probability of hypothesis φ; $P(B|\varphi)$ is the likelihood of observing the data behavior B under hypothesis φ. Data behavior B is introduced to further detail the generation behavior within different data modes. RealTS employs two data behavior modes for periodic hypothesis and three for trend hypothesis as follows:

- **Periodic Hypothesis:** Inverse Fast Fourier Transform Behavior (IFFTB) and Periodic Wave Behavior (PWB).
- **Trend Hypothesis:** Random Walk Behavior (RWB), Logistic Growth Behavior (LGB) and Trend Wave Data Behavior (TWDB)

Detailed formulas for each behavior mode and illustrative examples are provided in Appendix A.

3.3 Binary image-based time series metric space

In ViTime, time series are fed and operated with a binary image form, leveraging a binary image-based time series metric space, as described in Definition 3.1.

Definition 3.1 (Binary image-based time series metric space)**.** The binary image-based time series metric space is defined as a group (V, d), where V is a set of elements defined in Equation (2):

$$\mathcal{V} = \left\{ v \in \mathbb{R}^{c \times h \times L} \,\middle|\, v_{i,j,k} \in \{0,1\}, i \in [c], j \in [h], k \in [L], \sum_{j=1}^{h} v_{i,j,k} = 1 \right\} \tag{2}$$

where $d : V \times V \to \mathbb{R}$ is a distance function based on the Earth Mover's Distance (EMD), as defined in Equation (3):

$$d\left(v_1, v_2\right) = \int_{i=1}^{c} \int_{k=1}^{t} \inf_{\gamma \in \prod\left(\mathbf{v_1^{i,1:h,k}}, \mathbf{v_2^{i,1:h,k}}\right)} \mathbb{E}_{x,y \sim \gamma} \left\| x - y \right\|_1 dkdi \tag{3}$$

where c represents the number of variates, L is the length of the time series, and h is the resolution of V.

To enable the transition from numerical time-series values to the binary image-based metric space, we introduce mapping and inverse mapping functions as follows. Let $\mathcal{S} = \left\{ s \in R^{c \times L} \mid s_{i,k} \in R \right\}$ represent the numerical value space of time series. The Time-Series-to-Image mapping function $f : S \to \mathcal{V}$ and the Image-to-Time-Series inverse mapping function $f^{-1} : \mathcal{V} \to S$ can be defined as follows:

$$\begin{aligned} \mathbf{v_{i,1:h,k}} &= \mathbf{f}\left(s_{i,k}\right) = \left\langle f_1\left(s_{i,k}\right), f_2\left(s_{i,k}\right), \ldots f_h\left(s_{i,k}\right) \right\rangle \\ f_j\left(s_{i,k}\right) &= \begin{cases} 1, & \text{if } s_{i,k} \geq \mathrm{MS}, j = h \\ 1, & \text{if } s_{i,k} \leq -\mathrm{MS}, j = 1 \\ 1, & \text{if } j = \left\lfloor \frac{s_{i,k} + \mathrm{MS}}{\frac{2\mathrm{MS}}{h}} \right\rfloor \\ 0, & \text{otherwise.} \end{cases}, \quad j \in [h] \end{aligned} \tag{4}$$

The Image-to-Time-Series inverse mapping function $f^{-1} : \mathcal{V} \to \mathcal{S}$ can be defined as follows:

$$s_{i,k} = \mathbf{f}^{-1}\left(\mathbf{v_{i,1:h,k}}\right) = \sum_{j=1}^{h} \left((j - 0.5) \frac{2\mathrm{MS}}{h} - \mathrm{MS} \right) v_{i,j,k} \tag{5}$$

where $\mathrm{MS} > 0$ denotes the maximum scale of $\mathcal{V}$. Before mapping, zero-score normalization is typically applied to the numerical time series $s_{i,k}$ to standardize the scale.

Given that the numerical data synthesized by RealTS are one-channel time series, i.e., $\mathbf{s}_L \in R^l \in R^{1 \times L}$, thus the corresponding $\mathbf{v_L} \in R^{1 \times h \times L}$ is obtained via

$$\mathbf{v_L} = \mathbf{f}\left(\mathbf{s_L}\right) . \tag{6}$$

3.3.1 System error analysis

The system error (SE) emerges from the bidirectional mapping between discrete space $\mathcal{V}$ and continuous space $\mathcal{S}$, which inherently impacts prediction fidelity. A rigorous analysis of SE is essential for ensuring reliable and robust predictions in image space $\mathcal{V}$. We begin our theoretical analysis of SE with Assumption 3.2 and Theorem 3.3.

Assumption 3.2. After applying zero-score normalization, the continuous space follows a standard normal distribution:

$$\mathcal{S} \sim N(\mathbf{0}, \mathbf{I})$$

Theorem 3.3 (System Error Upper Bound)**.** *Given a tensor $\widehat{s} \in \mathcal{S} \subset \mathbb{R}^{c\times t}$, the system error defined as $\left\|f^{-1}\left(\mathbf{f}\left(\widehat{s}\right)\right)-\widehat{s}\right\|_1$ satisfies the following bound:*

$$\begin{aligned} SE &:= \mathbb{E}\left\|f^{-1}\left(\mathbf{f}\left(\widehat{s}\right)\right)-\widehat{s}\right\|_1 \leq g(h, MS) \\ &= ct\left[MS\left(\frac{1}{h}\left(\Phi(MS)-\Phi(-MS)\right)-2+2\Phi(MS)\right)+\sqrt{\frac{2}{\pi}}e^{-\frac{MS^2}{2}}\right] \end{aligned} \tag{7}$$

where Φ denotes the cumulative distribution function of $N(\mathbf{0}, \mathbf{I})$.

Denote $MS\left(\frac{1}{h}(\Phi(MS)-\Phi(-MS))-2+2\Phi(MS)\right)+\sqrt{\frac{2}{\pi}}e^{\frac{-MS^2}{2}}$ in Equation (7) as the upper bound of SE, whose convergence is guaranteed by Proposition 3.4.

Proposition 3.4 (Asymptotic Convergence with h)**.** *For any $\varepsilon > 0$, there exists $\delta > 0$ such that when $h \to +\infty$ and $MS \geq \delta$, the SE upper bound converges to zero:*

$$\lim_{h\to+\infty}\left|MS\left(\frac{1}{h}(\Phi(MS)-\Phi(-MS))-2+2\Phi(MS)\right)+\sqrt{\frac{2}{\pi}}e^{-\frac{MS^2}{2}}\right| = 0 \tag{8}$$

The Proposition 3.4 reveals that when we fix MS and increase the spatial resolution h, the upper bound $|g(h, MS)|$ of SE will reduce accordingly. On the other hand, when h increases, the tensor sizes in $\mathcal{V}$ will increase exponentially, leading to higher computational costs. As such, the selection of h must strike a balance between the accuracy of the estimation and the computational feasibility. Since the upper bound of SE decreases with an increase in h, it is generally preferable to choose the largest possible value of h based on available computational resources, resulting in a fixed value of h for a particular computational budget.

3.3.2 Theoretical analysis of optimal MS

MS determines the upper and lower limits of numerical truncation in the binary image-based time series metric space. Thus, it is necessary to conduct a detailed theoretical analysis of the selection of *MS*. Proposition 3.5 investigates how the upper bound of SE varies with a *MS* given a fixed value of h, which provides a theoretical guidance to choose the best MS under different computational budgets (h).

Proposition 3.5 (Optimal MS Selection)**.** *For fixed h, there exists a unique optimal threshold MS^* minimizing the SE upper bound, characterized by:*

$$\frac{1}{h}\left(\Phi(MS^*)-\Phi(-MS^*)\right)-2+2\Phi(MS^*)+\frac{MS^*}{h}\sqrt{\frac{2}{\pi}}e^{-\frac{MS^{*2}}{2}} = 0 \tag{9}$$

The fidelity of predictions in binary image space $\mathcal{V}$ heavily depends on the bidirectional mapping between discrete space $\mathcal{V}$ and continuous latent space $\mathcal{S}$. A key challenge arises from the SE, which quantifies the discrepancy between the original continuous representation and its reconstructed version after discretization. While Assumption 3.2 assumes $\mathcal{S} \sim \mathcal{N}(0, \mathbf{I})$, real-world scenarios often exhibit larger variance in the latent space due to factors such as dataset shifts or model miscalibration. This motivates our analysis of SE under the generalized assumption $\mathcal{S} \sim \mathcal{N}(0, k\mathbf{I})$, where $k > 1$ captures the variance scaling.

Proposition 3.6 (Optimal Threshold under Variance Scaling)**.** *Under the assumption $\mathcal{S} \sim \mathcal{N}(0, k\mathbf{I})$ with $k > 1$, the optimal threshold MS^* that minimizes the SE upper bound is characterized by the following condition:*

$$\frac{1}{h}\left(\Phi\left(\frac{MS^*}{\sqrt{k}}\right)-\Phi\left(-\frac{MS^*}{\sqrt{k}}\right)\right)-2+2\Phi\left(\frac{MS^*}{\sqrt{k}}\right)+\frac{MS^*}{h}\sqrt{\frac{2}{\pi k}}e^{-\frac{(MS^*)^2}{2k}} = 0 \tag{10}$$

Here, $\Phi(\cdot)$ is the CDF of the standard normal distribution, h is the spatial resolution, and k is the variance scaling factor. This result generalizes Proposition 3.5 to scenarios where the latent space exhibits larger

Table 1: Numerically Solved Optimal MS^*

	Optimal MS^*		
Resolution h	$k = 1$	$k = 1.5$	$k = 2$
32	2.1	2.62	3.03
64	2.38	2.95	3.41
128	2.64	3.26	3.76
256	2.88	3.53	4.08
512	3.09	3.79	4.38

variability. In practice, it is challenging to find an analytic solution for Equation (10). Thus, the numerical method is employed to obtain solutions of Equation (10) in this work and the corresponding results are reported in Table 1.

3.4 Theoretical Advantages of Visual Representation for Time Series Forecasting

Representing time series data visually, as explored by ViTime, is not merely an aesthetic or heuristic choice; it is fundamentally advantageous from a signal-processing standpoint. Specifically, transforming numerical signals into structured, image-like representations can significantly boost the effective signal-to-noise ratio (SNR), thereby enhancing forecasting robustness. To formally capture and quantify this advantage, we first establish conditions under which visual representation surpasses conventional numerical representation in terms of SNR. Subsequently, we explore image-based processing techniques to further amplify these benefits.

3.4.1 Visual Representation and SNR Enhancement.

Consider a noisy sinusoidal time series defined by:

$$s_k = A\sin(\omega_0 k + \phi) + \eta_k, \quad k = 0, \ldots, L-1,$$

where the signal amplitude $A > 0$, angular frequency $\omega_0 = 2\pi/P_{\text{period}}$, phase ϕ, and Gaussian noise terms $\eta_k \sim \mathcal{N}(0, \sigma^2)$ fully specify the system. Transforming this numerical series into a binary "stripe" image $v \in \{0,1\}^{h\times L}$ via quantization yields notable theoretical advantages. The binary representation is defined by:

$$v_{j,k} = \mathbf{1}\left(j = \left\lfloor \frac{s_k + \text{MS}}{\delta} \right\rfloor\right), \tag{11}$$

with quantization step $\delta = \Delta/h$ and total quantization range $\Delta = 2\text{MS}$. By comparing the SNR in numerical and visual domains, we obtain the following foundational result:

Theorem 3.7 (Stripe SNR Boost)**.** *Under mild assumptions that (i) the sinusoid amplitude spans at least one quantization bin ($\delta \le A \le \Delta - \delta$) and (ii) noise is small relative to quantization resolution ($\sigma < \delta/4$), the visual representation yields an SNR at the fundamental frequency $n_0 = \lfloor L/P_{\text{period}} \rfloor$ satisfying:*

$$SNR_{\text{vis}} \ge \frac{L}{4}\exp\left(\frac{\delta^2}{8\sigma^2}\right)\frac{\sigma^2}{A^2}SNR_{\text{num}}, \tag{12}$$

where the numerical SNR is $SNR_{\text{num}} = A^2/(2\sigma^2)$.

Theorem 3.7 provides clear quantitative conditions for visual superiority. Specifically, visual representation surpasses numerical representation ($\text{SNR}_{\text{vis}} > \text{SNR}_{\text{num}}$) whenever:

$$L > \frac{4A^2}{\sigma^2}\exp\left(-\frac{\delta^2}{8\sigma^2}\right). \tag{13}$$

Practically, this condition is typically met for moderate sequence lengths when the quantization step is comparable to or slightly larger than the noise standard deviation (e.g., $\delta \approx 2\sigma$). Under these realistic scenarios, the exponential term strongly favors visual representation, making it advantageous even at manageable L.

3.4.2 SNR Enhancement via Image Processing.

Although the theoretical advantage above is compelling, practical scenarios often involve considerable noise and subtle periodic signals. Furthermore, the binary quantization can introduce high-frequency artifacts that obscure signal patterns. To mitigate such undesirable effects and leverage the structured nature of visual representations, we propose employing image-processing operations, notably Gaussian blurring, to enhance signal fidelity further.

Applying a Gaussian blur along the image's quantization axis (the row or "value" dimension) effectively smooths quantization noise while preserving meaningful temporal structures. This simple convolutional operation yields significant amplification of the visual-domain SNR, formalized as follows:

Theorem 3.8 (Gaussian Blur SNR Boost)**.** *Under the conditions of Theorem 3.7, consider applying a one-dimensional Gaussian convolution kernel along the quantization dimension (rows) of the binary stripe image v:*

$$g_j = \frac{1}{Z}\exp\left(-\frac{j^2}{2\sigma_b^2}\right), \quad \textit{where} \quad Z = \sum_j \exp\left(-\frac{j^2}{2\sigma_b^2}\right),$$

to obtain the blurred image $w = g *_j v$. Denote the kernel's nuclear energy by $S = \sum_j g_j^2 \in (0,1)$, and define the visually blurred SNR at the fundamental frequency $n_0 = \lfloor L/P_{\text{period}} \rfloor$ as SNR_{vis}^{blur}. Then, the following lower bounds hold:

$$SNR_{\text{vis}}^{blur} \geq \frac{L}{4S}\exp\left(\frac{\delta^2}{8\sigma^2}\right), \tag{14}$$

$$SNR_{\text{vis}}^{blur} \geq \frac{L\sigma^2}{2A^2S}\exp\left(\frac{\delta^2}{8\sigma^2}\right) SNR_{\text{num}}, \tag{15}$$

where the numerical-domain SNR is defined as $SNR_{\text{num}} = A^2/(2\sigma^2)$.

Consequently, the blurred visual representation amplifies the numerical-domain SNR at least by a factor of:

$$\frac{\text{SNR}_{\text{vis}}^{\text{blur}}}{\text{SNR}_{\text{num}}} \geq \frac{L\sigma^2}{2A^2S}\exp\left(\frac{\delta^2}{8\sigma^2}\right). \tag{16}$$

This result explicitly quantifies the advantage provided by Gaussian blurring in the visual representation. Notably, this amplification advantage scales linearly with the time series length L and exponentially with the squared ratio of quantization step δ to noise standard deviation σ. Moreover, a smaller kernel nuclear energy S — corresponding to stronger blurring — yields a greater amplification of the visual-domain SNR relative to its numerical counterpart.

In practical implementations, the choice of Gaussian kernel parameters directly influences the nuclear energy S, and thus the SNR amplification factor. Typical examples include:

- 11×11 kernel ($\sigma_b = 2$): $S \approx 0.15$, providing substantial SNR amplification.
- 21×21 kernel ($\sigma_b = 4$): $S \approx 0.08$, approximately doubling the amplification compared to the previous case.
- 31×31 kernel ($\sigma_b = 6$): $S \approx 0.05$, further significantly enhancing the amplification factor.

In summary, even moderate Gaussian blurring substantially enhances the effective visual-domain SNR, enabling significantly improved signal discernibility and forecasting accuracy compared to traditional numerical-domain methods.

Generalization to Complex Time Series. While our theoretical analysis explicitly addresses a single sinusoidal component, its implications readily extend to realistic time series composed of multiple periodic components. Via linearity principles inherent in Fourier decomposition, observed visual-domain SNR

advantages apply component-wise, amplifying structured periodic signals relative to unstructured and independent noise effects. Thus, real-world time series exhibiting intricate periodic behaviors benefit significantly from visual transformations and subsequent image-processing enhancements.

The rigorous theoretical results presented here establish a robust mathematical foundation for employing visual intelligence in time series analysis. Beyond aligning with human cognitive patterns, visual representations structurally amplify signal fidelity through inherent quantization and subsequent image processing techniques, such as Gaussian smoothing. Consequently, visual-domain methods provide a principled, theoretically justified route toward achieving more robust, reliable, and accurate time series forecasting, especially under challenging noise conditions.

Detailed proofs and supplementary details of the theorems presented in this section are provided in Appendix C.

3.5 The proposed ViTime model

Figure 1 (b) presents the architecture of the ViTime network, which comprises three network modules: the Visual Time Tokenizer, the Decoder, and the Refining Module. The time series binary image is first fed into the Visual Time Tokenizer and outputs embedded latent representations. Next, the decoder network is developed to decode latent representations and produce initial prediction results. To improve the generative quality of patch junctions, a Refining Module is designed to generate the final smooth prediction results.

Visual Time Tokenizer. The primary role of the Visual Time Tokenizer is to segment masked binary images into multiple patches and map these patches into the feature space. By leveraging the ViT (Dosovitskiy et al., 2020) architecture, the module captures spatial relationships between patches, thereby transforming temporal dependencies of the time series into spatial dependencies within the image space.

Decoder. The Decoder translates the tokenized patches back into the binary pixel metric space, providing an initial prediction where the ViT architecture is also adopted.

Refining Module. The transformer architecture in the Decoder can result in discontinuities at the patch junctions, which may affect the accuracy of the inverse mapping process. To address this issue, the Refining Module building with CNNs is employed. Initially, tokens decoded by Decoder are unpatched and fed into a CNN-based backbone. Next, the ASPP (Chen et al., 2015) module expands the model receptive field. Finally, the output is upsampled to the binary pixel metric space, generating the final image prediction result.

The modeling process of ViTime is as follows:

$$\mathbf{v}'_{\mathbf{L}} = \mathrm{ViTime}\left(\mathbf{v}_{\mathbf{L}} \odot \mathbf{M}_{\mathbf{L}}\right) \tag{17}$$

where $\mathbf{M}_{\mathbf{L}}$ denotes temporal masks.

Loss function. The loss function employed in this study is defined as follows:

$$\mathcal{L} = d\left(\mathbf{v}'_{\mathbf{L}}, \mathbf{v}_{\mathbf{L}}\right) + \alpha \mathrm{KLD}\left(\mathbf{v}'_{\mathbf{L}}, \mathbf{v}_{\mathbf{L}}\right) \tag{18}$$

where d denotes the distance function defined in Equation (3), KLD denotes Kullback–Leibler divergence, and α is the hyperparameter balance quantity between d and KLD. The combined EMD and KLD loss addresses structural and probabilistic alignment in the binary image space. EMD minimizes spatial discrepancies in $\mathcal{V}$, counteracting SE from discretization, while KLD refines distributional consistency to mitigate quantization artifacts. This dual approach balances geometric fidelity (via EMD) and statistical accuracy (via KLD), crucial given the resolution-computation trade-off governed by h.

3.6 Evaluation metrics

Existing numerical fitting-based TSF foundation models, e.g., TimesFM, are typically pretrained on comprehensive real-world datasets. While the specific nomenclature of the testing set may not be explicitly listed in the training data, there is a possibility that the real-world dataset encompasses similar data sources,

potentially leading to issues of test set leakage. To address this concern and ensure a more rigorous and equitable experimental comparison, we propose two novel metrics for **zero-shot evaluation**, the Rescale-Mean Absolute Error (ReMAE) and Rescale-Mean Squared Error (ReMSE). The fundamental principle underlying ReMAE and ReMSE involves rescaling the test dataset across various time resolutions, as illustrated in Equation (19). The time series interpolation (TSI) method is employed to rescale the original test time series of length T to βT:

$$S_{\beta T} = TSI\left(S_T, \text{rescaling factor} = \beta\right). \tag{19}$$

The formulas for ReMAE and ReMSE are

$$ReMSE = \frac{\sum_{\beta \in \mathbf{U}} MSE\left(S'_{\beta T}, S_{\beta T}\right)}{len(\mathbf{U})} \tag{20}$$

$$\begin{aligned} ReMAE &= \frac{\sum_{\beta \in \mathbf{U}} MAE\left(S'_{\beta T}, S_{\beta T}\right)}{len(\mathbf{U})} \\ \mathbf{U} &= [0.5,\ 0.66, 1, 1.5, 2]. \end{aligned} \tag{21}$$

The proposed ReMSE and ReMAE metrics address a critical challenge in evaluating time series foundation models: mitigating test set leakage caused by overlapping data distributions between training and testing phases. By rescaling the test set across multiple resolutions ($\beta \in \mathbf{U}$) via time series interpolation (TSI, Equation (19)), these metrics introduce synthetic scale variations that disrupt exact temporal patterns, thereby reducing the risk of evaluating models on memorized or overfitted data. This approach ensures a leakage-resistant evaluation framework, as models must generalize to unseen scales rather than relying on spurious correlations learned from the training set.

A key implication of this work is the necessity of scale-agnostic evaluation in time series forecasting. Traditional single-scale metrics like MSE/MAE risk conflating memorization with true generalization, particularly when training data encompasses diverse real-world sources. By averaging errors across β, ReMSE/ReMAE incentivize models to capture invariant temporal structures—such as periodicity, trends, and noise resilience—that persist across resolutions. This aligns with recent theoretical insights in self-supervised learning, where augmentation-induced invariance improves out-of-distribution robustness (Yao et al., 2022). It is worth noting that in the fine-tuning study, i.e., section 4.3, in order to ensure the consistency of the distribution between the test data and the fine-tuning data, we still adopt the traditional MSE/MAE evaluation metrics.

4 Computational experiments

4.1 Experimental Configuration

Datasets

Seven popular publicly accessible datasets: Electricity, Traffic, Weather, ETTh1, ETTh2, ETTm1, and ETTm2 (Wu et al., 2021) are employed in computational experiments to validate the effectiveness of the proposed ViTime.

Model setup

The ViTime model is developed using data sequences synthesized by RealTS. During each training epoch, 20,000 sequences are randomly generated. After training, zero-shot testing and fine-tuning are implemented accordingly. For multivariate time series, a channel-independent strategy (Nie et al., 2022) is applied, predicting each variable separately before combining them to form the final multivariate forecast.

The default parameters for the ViTime model are set as follows: $h = 128, MS = 3.5$, maximum lookback window $T = 512$, and maximum prediction length $l = 720$. For a fair comparison, all considered models

employ a lookback length of 512 to forecast future sequences of lengths $96, 192, 336, 720$. More details on training are available in the Appendix B.

To further enhance temporal resolution and information density practically, input sequences are initially interpolated to twice their original length (2L) and the prediction results are interpolated back to the original length. This interpolation increases temporal granularity, facilitating more precise pattern extraction. Furthermore, Gaussian blurring with kernel size of 31 applied to the binary images before processing by ViTime significantly reduces sparsity and increases local information density, thereby reinforcing the theoretical advantages outlined in Section 3.4.1.

4.2 Comparison of ViTime to SOTA TSF Benchmarks Under Zero-shot Setting

(a) Experimental Results With Metrics of MSE and MAE

Model	ETTh1		ETTh2		ETTm1		ETTm2		electricity		traffic		weather	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Numerical Models														
Moriai	0.434	0.439	0.346	0.382	0.382	0.388	0.272	0.321	0.188	0.274	1.779	0.766	0.238	0.261
Moment	0.691	0.585	0.341	0.350	0.845	0.580	0.257	0.317	0.837	0.763	1.375	0.788	0.348	0.429
VisionTS	**0.390**	0.414	0.333	0.375	**0.374**	**0.372**	0.282	0.321	0.207	0.294	0.443	0.284	0.269	0.292
TimesFM	0.442	0.430	0.356	0.389	0.424	0.419	0.328	0.347	0.151	0.245	0.369	0.245	0.229	0.255
PatchTST-ZS	1.237	0.831	0.903	0.710	1.356	0.825	0.839	0.622	1.311	0.885	1.873	0.945	0.907	0.588
Vision-Assisted Models														
ViTime-TFM	0.398	**0.387**	0.321	0.350	0.382	0.377	0.295	0.312	**0.136**	**0.221**	**0.332**	**0.221**	0.206	0.229
ViTime	0.545	0.449	**0.284**	**0.344**	0.409	0.398	**0.189**	**0.265**	0.196	0.280	0.730	0.386	**0.173**	**0.196**

(b) Experimental Results With Metrics of ReMSE and ReMAE

Model	ETTh1		ETTh2		ETTm1		ETTm2		electricity		traffic		weather	
	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE
Numerical Models														
Moriai	1.980	1.440	1.340	1.145	1.303	0.714	0.324	0.347	0.968	0.655	1.721	0.801	0.677	0.464
Moment	1.759	1.470	0.857	0.972	0.832	0.617	0.344	0.365	0.800	0.742	1.314	0.783	0.308	0.377
VisionTS	1.177	1.104	0.849	0.933	0.856	0.584	0.634	0.496	0.816	0.666	1.378	0.724	0.274	0.323
TimesFM	**0.513**	0.476	0.354	0.391	0.671	0.503	0.335	0.358	0.367	0.404	0.874	0.519	0.284	0.306
PatchTST-ZS	1.477	0.903	1.097	0.775	1.295	0.798	0.805	0.613	1.414	0.921	2.054	1.002	0.911	0.584
Vision-Assisted Models														
ViTime-TFM	0.520	0.465	0.320	0.370	0.550	0.460	0.280	0.330	0.300	0.350	0.800	0.460	0.240	0.260
ViTime	0.514	**0.455**	**0.289**	**0.349**	**0.474**	**0.420**	**0.237**	**0.301**	**0.225**	**0.308**	**0.730**	**0.400**	**0.203**	**0.228**

Table 2: Overall Experimental Results Comparison

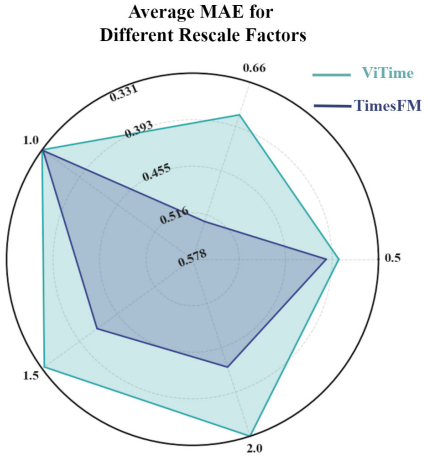


Figure 2: Radar plots comparing the average MAE of ViTime and TimesFM across different rescale factors. The radial axis represents MAE, with lower values (larger radius) indicating better performance. Each axis corresponds to a specific rescale factor.

For zero-shot performance comparison, we consider four variants: (1) ViTime - our proposed TSF foundation model, trained on generative data from RealTS and adopting a zero-shot paradigm; (2) ViTime-TFM - a variant of ViTime, which is trained explicitly on the same dataset as TimesFM. . (3) PatchTST-ZS - trained on the same RealTS-generated data as ViTime, using a numerical fitting paradigm to create a zero-shot version of PatchTST. (4) Moriai (Woo et al., 2024), Moment (Goswami et al., 2024), VisionTS (Chen et al., 2024) and TimesFM (Das et al., 2024) - powerful TSF foundation model pre-trained on extensive real-world datasets.All models employ a lookback length of 512 to ensure a fair comparison.

Table 2 summarizes the zero-shot performance of all models using traditional metrics (MSE, MAE) and our proposed scale-invariant metrics (ReMSE, ReMAE). ViTime consistently demonstrates excellent forecasting performance, closely approaching or surpassing considered benchmarks. Particularly noteworthy is ViTime's exceptionally strong performance on the ReMSE and ReMAE metrics, highlighting its robust generalization ability across different temporal resolutions in zero-shot settings. ViTime significantly outperforms PatchTST-ZS across all datasets, underscoring the effectiveness of visual intelligence strategies in capturing complex temporal patterns. Furthermore, when compared with TimesFM, ViTime achieves an average performance improvement of approximately 9%, further accentuated to 15% in challenging long-sequence forecasting scenarios.

Additionally, ViTime-TFM, which shares identical training data with TimesFM, demonstrates superior performance in traditional metrics (MSE, MAE). This clearly indicates the inherent advantage of vision-based modeling in capturing intricate temporal dynamics. However, its performance on ReMSE and ReMAE falls short of ViTime, implying that the synthetic training data provided by RealTS substantially enhances the zero-shot generalization capabilities across varying temporal resolutions.

To further assess robustness, Figure 2 presents the performance across different rescaling factors. TimesFM exhibits optimal accuracy only at the original scale ($\beta = 1$), suffering significant degradation when evaluated at other scales. Such behavior indicates sensitivity to scale-specific patterns and suggests potential data leakage from the original resolution. In contrast, ViTime maintains consistently robust forecasting performance across all rescaling factors, as evidenced by stable ReMSE and ReMAE metrics. This illustrates ViTime's ability to learn intrinsic temporal relationships independent of specific time resolutions, further reinforcing the robustness and generalization benefits of vision-based modeling trained on RealTS data.

Overall, the empirical results clearly position ViTime as a robust, accurate, and reliable zero-shot TSF model, substantially strengthened by vision-assisted modeling and synthetic training data that enhance generalization across diverse temporal scales.

4.3 Comparison of ViTime to SOTA TSF Benchmarks Under Fine-tuning Settings

Table 3: Comparisons of Fine-tuning forecasting results with MAE. FT is short for fine-tuning. The best MAE results are **bolded**, and the second best are underlined.

Method	Data proportion	ETTh1	ETTh2	ETTm1	ETTm2	Electricity	Traffic	Weather
TimesFM (FT)	10%	0.426	0.410	0.388	0.334	-	-	-
GPT4TS (FT)	10%	0.525	0.421	0.441	0.335	Not Reported		
TIME-LLM (FT)	10%	0.522	0.394	0.427	0.323	-	-	-
ViTime (FT)	10%	0.424	0.372	0.378	0.316	0.252	0.254	0.252
PatchTST	10%	0.542	0.431	0.466	0.343	0.268	0.286	0.283
PatchTST	100%	0.434	0.381	0.382	0.317	0.253	0.264	0.264
SiMBA	100%	0.433	0.393	0.396	0.327	0.274	0.291	0.281
TIMESNET	100%	0.450	0.427	0.406	0.332	0.295	0.336	0.267
iTransformer	100%	0.447	0.407	0.410	0.332	0.270	0.282	0.278
TimeMixer	100%	0.423	0.384	0.375	0.315	**0.246**	0.262	0.262
ViTime (FT)	100%	**0.408**	**0.349**	**0.367**	**0.300**	0.247	**0.250**	**0.251**

While zero-shot results demonstrate the predictive capability of ViTime on unseen data, some high-precision TSF tasks might require further fine-tuning studies to enhance prediction accuracy. Thus, this section focuses on fine-tuning studies across various specialized datasets.

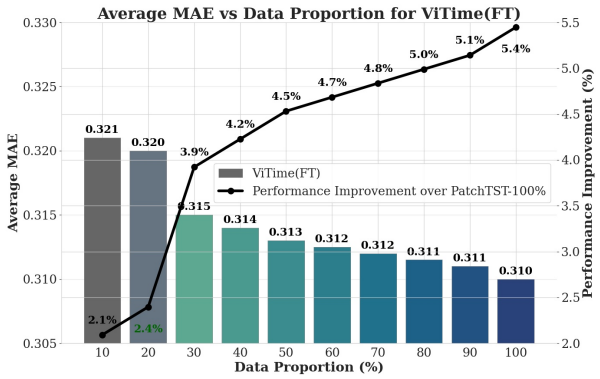


Figure 3: Performance with different fine-tuning data proportion.

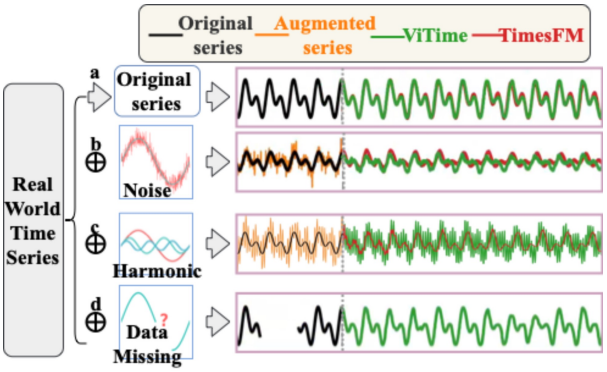


Figure 4: Performance comparison of ViTime versus TimesFM on TSF tasks under various data perturbations: a. Original time series. b. Time series with noises injected. c. Time series with harmonic added. d. Time series with missing data.

To comprehensively evaluate the fine-tuning performance of ViTime, we compare ViTime with other foundation models and SOTA supervised TSF models. Foundational models including TimesFM (Das et al., 2024), GPT4TS (Zhou et al., 2023a), and TIME-LLM (Jin et al., 2023) are fine-tuned using 10% of the training data. Recent SOTA-supervised TSF models such as SiMBA (Patro & Agneeswaran, 2024), TIMESNET (Wu et al., 2022), iTransformer (Liu et al., 2023), TimeMixer (Wang et al., 2024) and PatchTST (Nie et al., 2022) use 100% of the training data, as reported in their respective papers. We also fine-tune ViTime using from 10% to 100% of the training data to provide a comprehensive comparison.

Results of the fine-tuning study are provided in Table 3. ViTime fine-tuned with 10% of the training data can outperform other foundational models and the latest supervised models updated on 100% of the training data. Furthermore, as shown in Fig. 3, when the fine-tuning data proportion approaches 100%, the prediction accuracy of ViTime gradually increases and significantly surpasses all existing models, which suggests that ViTime excels in both low-data-availability environments (10% fine-tuning) and full-data-availability scenarios (100% fine-tuning), consistently outperforming both other foundation models and specialized supervised models.

4.4 Robust Inference and Generalizability Analysis

Table 4: Comparison of average ReMAE forecasting results. Methods are grouped by scenario (separated by horizontal lines). Within each scenario, the best MAE results for each dataset are **bolded**.

Method	ETTh1	ETTh2	ETTm1	ETTm2	Electricity	Traffic	Weather
GN standard deviations = 0.1							
TimesFM	0.471	0.394	0.495	0.353	0.403	0.511	0.281
ViTime	**0.454**	**0.382**	**0.442**	**0.340**	**0.348**	**0.412**	**0.280**
GN standard deviations = 0.3							
TimesFM	0.478	0.392	0.488	0.345	0.433	0.529	0.296
ViTime	**0.472**	**0.391**	**0.457**	**0.344**	**0.381**	**0.477**	**0.292**
DM P = 0.3							
ViTime	**0.453**	**0.378**	**0.432**	**0.337**	**0.343**	**0.417**	**0.281**

To rigorously assess the robustness and generalizability of ViTime, we conducted comprehensive zero-shot experiments comparing its performance against TimesFM under various data perturbation scenarios, including original time series, Gaussian noise (GN), harmonic augmentation, and missing data (DM). These scenarios represent realistic challenges often encountered in practical forecasting tasks, evaluating the models' capacities to maintain predictive accuracy amidst compromised data quality.

Figure 4 visually summarizes the comparative performance of ViTime and TimesFM across these scenarios. In the scenario involving original, unperturbed time series, ViTime consistently demonstrates superior capabilities in capturing and modeling underlying periodicities and temporal patterns. For noise-augmented series, while both models successfully extract meaningful insights, ViTime notably maintains stable forecasting performance across extended sequences. In contrast, TimesFM tends to experience drift in periodic alignment, particularly at higher noise intensities. For harmonic-augmented time series, ViTime excels by accurately capturing both fundamental and harmonic wave patterns, while TimesFM struggles to disentangle these complex periodic structures.

Further quantitative analysis under varying levels of Gaussian noise, summarized in Table 4 (the complete numerical results are available in the Appendix D), highlights ViTime's superior robustness. Even with increased noise severity (GN std=0.3), ViTime consistently outperforms TimesFM across all tested datasets. This resilience arises from ViTime's distinct visual representation learning approach, which inherently filters out irrelevant noise through spatial feature extraction, unlike numerical fitting-based models that are more susceptible to noisy perturbations.

The most distinctive performance disparity emerges under the scenario of missing data. TimesFM, inherently reliant on numerical fitting, necessitates explicit imputation strategies to address data gaps. Conversely, ViTime robustly accommodates missing values by interpreting them as zero-valued pixels within its visual representations. Consequently, ViTime leverages spatial dependencies among available data points effectively, maintaining high prediction accuracy even amidst substantial data sparsity.

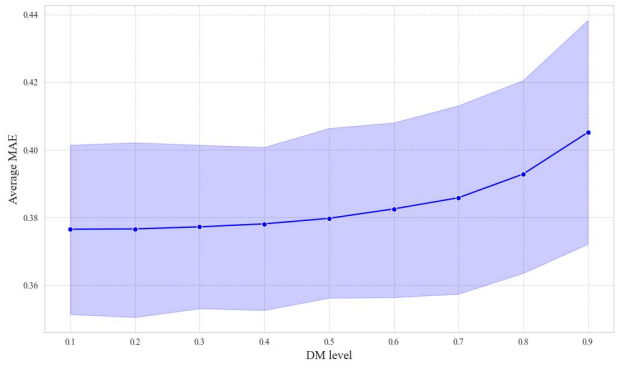


Figure 5: Average MAE of ViTime across different DM rates.

To further validate ViTime's robustness to varying degrees of missing data, we systematically evaluated its forecasting accuracy across data missing ratios ranging from 10% to 90% (Figure 5). Results reveal that ViTime sustains remarkable forecasting performance with minimal degradation until data missingness surpasses 50%, underscoring its exceptional resilience to incomplete data scenarios.

Collectively, these extensive evaluations substantiate ViTime's superior robustness and generalizability compared to traditional numerical fitting-based methods. Its inherent capability to effectively mitigate perturbations through visual representation learning positions it as a highly promising approach for real-world forecasting applications, where consistent data quality cannot always be guaranteed.

4.5 Ablation study

4.5.1 Ablation of MS

Proposition 3.6 establishes the theoretical relationship between the optimal MS threshold and the variance scaling factor k in the latent space. For stationary data ($\mathcal{S} \sim \mathcal{N}(0, \mathbf{I})$, i.e., $k = 1$), Proposition 3.6 reveals that with $h = 128$, the optimal MS should be 2.64. However, real-world time series often exhibit non-stationary characteristics. Our pre-analysis of the target variable's variance after input-based standardization (see Appendix appendix B.2) demonstrates that the effective k value for the prediction horizon falls within $[1.5, 2]$ across all benchmark datasets.

Table 5: Empirical Forecasting Performance under Different MS Values

	MS						
	2.38	2.64	2.88	3.09	3.50	5.00	6.00
ReMSE	0.4423	0.4404	0.4400	0.4348	**0.4178**	0.4780	0.4724
ReMAE	0.3818	0.3812	0.3811	0.3788	**0.3759**	0.3990	0.3959

Table 1 provides numerically solved optimal MS^* values under different k and h configurations. For $h = 128$ (our experimental setting) and $k \in [1.5, 2]$, the theoretical optimal MS ranges between 3.26-3.76. This motivates our selection of $MS = 3.5$ as a balanced configuration within this interval.

To validate this choice, Table 5 presents the average relative ReMSE and ReMAE across six benchmark datasets under zero-shot setting. The results demonstrate that $MS = 3.5$ achieves the minimum forecasting error, reducing ReMSE by 4.1% and ReMAE by 1.8% compared to the stationary optimal $MS = 2.64$. This strong alignment between theoretical predictions (Table 1) and empirical performance (Table 5) confirms that our MS selection strategy effectively minimizes system error while accommodating real-world data characteristics.

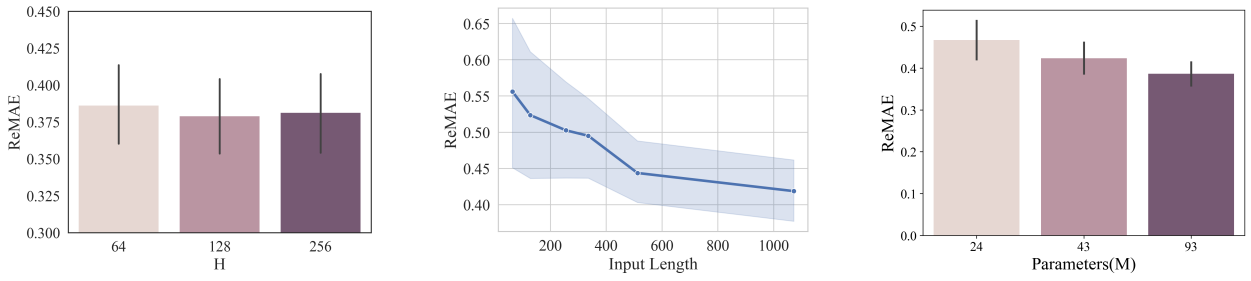
4.5.2 Ablation of loss function

Table 6: Ablation study of loss function components on prediction performance.

Metric	Loss Configuration		
	EMD Only	JSD+EMD (Ours)	JSD Only
Average ReMAE	0.3941	**0.3759**	0.3956
Average ReMSE	0.4586	**0.4178**	0.4637

In this section, we conducted ablation studies on the loss function components of ViTime under zero-shot setting. Table 6 compares model performance under three configurations: (1) EMD alone, (2) our proposed loss function in Equation (18),where $\alpha = 0.2$ to balance the quantity level, and (3) JSD alone. The results demonstrate that our dual-objective loss achieves optimal performance on both ReMSE and ReMAE.

4.5.3 Ablation of other configuration



(a) Forecasting performance varying with h

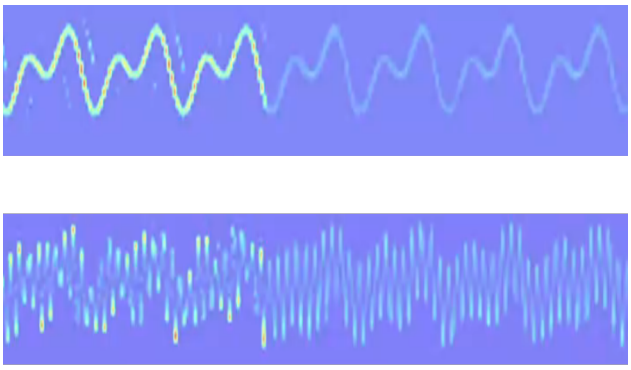
(b) Forecasting performance varying with the length of lookback window

(c) Forecasting performance across different model sizes

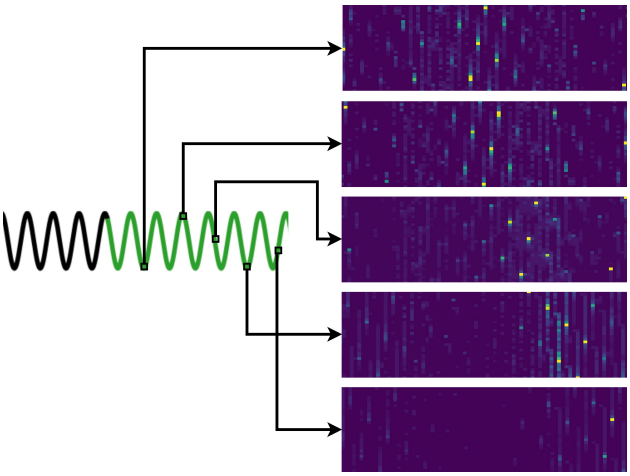
Figure 6: Ablation studies with zero-shot forecasting.

Note: The model size of ViTime used in computational experiments is 93M parameters version.

In this section, we perform several ablation studies to gain deeper insights into ViTime model configuration. The results are reported in Figure 6. Figure 6 a depicts the influence of varying spatial resolutions (h) on model accuracy. Although increasing h slightly improves the prediction results, the associated computational



(a) Grad-CAM heatmap showing attention on key trend changes.



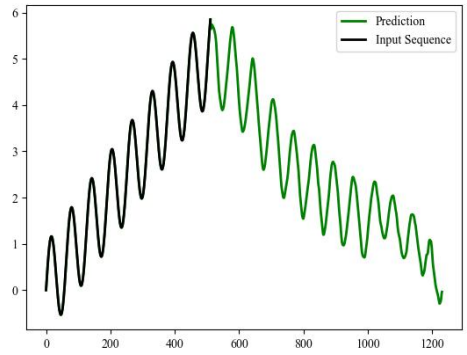
(b) Attention maps at different prediction positions demonstrating temporal dependencies.

Figure 7: Visualization of ViTime's attention mechanism. Despite not using an autoregressive paradigm, ViTime exhibits sequential processing patterns through its multi-layer self-attention modules.

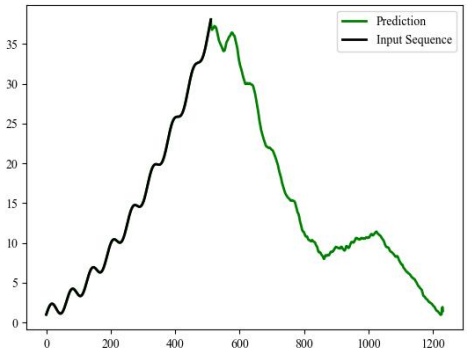
cost increases exponentially. Thus, setting h to 128 is more economical and efficient. Figure 6 b illustrates the effect of different lookback window lengths (T) on prediction accuracy. It is evident that a longer lookback window length significantly enhances the model's prediction accuracy. Figure 6 c reports the prediction accuracy across different model sizes. The data shows that models with more parameters tend to perform better. Moreover, the proposed ViTime achieves superior performance with only **93M** parameters compared with TimesFM, which is over 200M parameters, further demonstrating the efficiency and effectiveness of ViTime.

4.6 Interpretation of ViTime

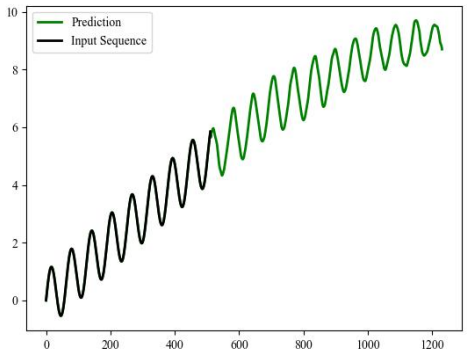
Figure 7 illustrates the attention mechanism of ViTime through grad-cam (Selvaraju et al., 2017) heatmaps and position-specific attention maps. The grad-cam results demonstrate that ViTime focuses strongly on periods of fundamental trend changes. Further analysis through attention maps at different prediction positions reveals an interesting pattern: despite not adopting an autoregressive paradigm, ViTime's multi-layer self-attention modules process information in a temporal sequence. The input data and the predicted results from previous time steps determine the spatiotemporal distribution of predictions at each time step. This aligns with human cognitive patterns, where information is processed from the recent to the distant past while maintaining awareness of known information.



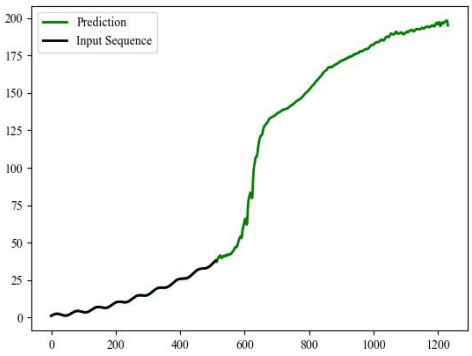
(a) MS=3.5



(b) MS=3.5



(c) MS=7



(d) MS=7

Figure 8: Resolution analysis for explosive growth patterns: (a-b) With MS=3.5, ViTime incorrectly predicts peak decline due to spatial constraints. (c-d) Doubling MS to 7 enables accurate growth trend capture.

5 Discussion

While ViTime demonstrates state-of-the-art performance in accuracy and robustness, two key challenges warrant further investigation:

5.1 Resolution Constraints & Adaptive Enhancement.

The mapping function's truncation imposes resolution limits, particularly evident in explosive growth patterns (Figure 8 a-b). A key limitation of ViTime arises from its assumption of $\mathcal{S} \sim \mathcal{N}(0, \mathbf{I})$, which fails to capture the high-variance nature of explosive growth data that typically follows $\mathcal{S} \sim \mathcal{N}(0, k\mathbf{I})$ with $k \gg 1$. As shown in Proposition 3.6, the optimal threshold MS^* scales as $\sqrt{k}$, implying that fixed thresholds (e.g., $MS = 3.5$ for $k = 1.5$) become suboptimal for high-variance scenarios, introducing significant system errors and degrading prediction accuracy.

Our empirical analysis reveals that doubling the MS parameter from 3.5 to 7 significantly improves prediction fidelity for explosive growth patterns (Figure 8c-d). However, excessively large MS values increase system error, as demonstrated in Theorem 3.3, leading to computational inefficiency. This trade-off suggests two complementary research directions:

- **Elastic Resolution Enhancement**: Techniques to dynamically adjust spatial resolution h based on data variance, ensuring sufficient granularity for high-variance regions without unnecessary computational overhead.
- **Adaptive MS Estimation**: Algorithms to estimate the variance scaling factor k and compute the optimal MS^* in real-time, balancing prediction fidelity with spectral efficiency.

These enhancements would enable ViTime to handle explosive growth patterns more effectively while maintaining computational tractability.

5.2 Enhanced Data Generation.

ViTime's predictive quality fundamentally depends on RealTS's synthetic data generation capabilities. The current methodology faces challenges in simulating complex real-world temporal dynamics, particularly for non-stationary processes and regime-switching scenarios. Future work should develop 1) Advanced pattern injection mechanisms for synthetic data generation, and 2) Quantitative metrics for simulation fidelity assessment across different temporal regimes.

6 Conclusions

This work developed a vision intelligence-powered computational paradigm, ViTime, for developing the TSF foundation model compared with numerical data fitting principles prevalently considered in literature. ViTime was inspired by human visual cognitive processes understanding and analyzing time series. By introducing a paradigm of operating numerical data in image space and the unique deep network based computing pipeline, ViTime elevated the SOTA performance on zero-shot/fine-tuning TSF without relying on prior data samples, demonstrating the great potential for reshaping the computational mechanism in TSF foundation model development. Moreover, data often suffer from diverse contamination and variability in reality. ViTime enabled robust performance under various real-world data perturbations and alterations.

References

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

Ching Chang, Wei-Yao Wang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Aligning pre-trained llms as data-efficient time-series forecasters. *ACM Transactions on Intelligent Systems and Technology*, 16(3):1–20, 2025.

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *3rd International Conference on Learning Representations, ICLR 2015-Conference Track Proceedings*, volume 40, pp. 834–848, 2015.

Mouxiang Chen, Lefei Shen, Zhuo Li, Xiaoyun Joy Wang, Jianling Sun, and Chenghao Liu. Visionts: Visual masked autoencoders are free-lunch zero-shot time series forecasters. *arXiv preprint arXiv:2408.17253*, 2024.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 10148–10167. PMLR, 21–27 Jul 2024.

Donis A Dondis. *A primer of visual literacy*. MIT Press, Cambridge, MA, 1974.

Sam Dooley, Gaurav Singh Khurana, Chirag Mohapatra, Alex Nguyen, Soyoung Yoo, Jayne Bruckbauer, Richard Socher, Ryan Mortimore, and James Requeima. Forecastpfn: Synthetically-trained zero-shot forecasting. In *Advances in Neural Information Processing Systems*, volume 36, 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Cheng Feng, Long Huang, and Denis Krompass. Only the curve shape matters: Training foundation models for zero-shot multivariate time series forecasting through next curve shape prediction. *arXiv preprint arXiv:2402.07570*, 2024.

Alejandro Garza and Mauricio Mergenthaler-Canseco. Timegpt-1. *arXiv preprint arXiv:2310.03589*, 2023.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*, 2024.

Hansika Hewamalage, Christoph Bergmeir, and Kasun Bandara. Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, 37(1):388–427, 2021.

Michael G Jacox, Michael A Alexander, Dillon Amaya, Emily Becker, Giovanni Boer, Wenju Cai, Leticia Cotrim da Cunha, Charlotte A DeMott, Daniela F Dias, Christopher A Edwards, et al. Global seasonal forecasts of marine heatwaves. *Nature*, 604(7906):486–490, 2022.

Ming Jin, Shiyu Wang, Lintao Ma, Pin Li, Ke Yang, Qingsong Wen, Yue Liu, Liang Zhang, Rui Ren, Xiaoyong Du, et al. Time-ilm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.

Weiping Lei, Luiz GA Alves, and Luís AN Amaral. Forecasting the evolution of fast-changing transportation networks using machine learning. *Nature communications*, 13(1):4252, 2022.

Peter AW Lewis and James G Stevens. Nonlinear modeling of time series using multivariate adaptive regression splines (mars). *Journal of the American Statistical Association*, 86(416):864–877, 1991.

Wei-Yang Lin, Ya-Han Hu, and Chih-Fong Tsai. Machine learning in financial crisis prediction: a survey. *IEEE Transactions on Systems, Man, and Cybernetics*, 42(4):421–436, 2011.

Chenxi Liu, Shuo Yang, Qingyu Xu, Zipei Fan, Zengxiang Ding, Renhe Jiang, Xun Xie, and Xuan Song. Spatial-temporal large language model for traffic prediction. *arXiv preprint arXiv:2401.10134*, 2024.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.

Haoyu Ma, Yushu Chen, Wenlai Zhao, Jinzhe Yang, Yingsheng Ji, Xinghua Xu, Xiaozhu Liu, Hao Jing, Shengzhuo Liu, and Guangwen Yang. A mamba foundation model for time series forecasting. *arXiv preprint arXiv:2411.02941*, 2024.

Douglas C Montgomery, Cheryl L Jennings, and Murat Kulahci. *Introduction to time series analysis and forecasting*. John Wiley & Sons, 2015.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

Badri N Patro and Vineeth S Agneeswaran. Simba: Simplified mamba-based architecture for vision and multivariate time series. *arXiv preprint arXiv:2403.15360*, 2024.

R Pettersson. Visual information. *Educational Technology*, 1993.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

Hiba Sharadga, Shima Hajimirza, and Robert S Balog. Time series forecasting of solar power generation for large-scale photovoltaic plants. *Renewable Energy*, 150:797–807, 2020.

Mingtian Tan, Mike A Merrill, Vinayak Gupta, Tim Althoff, and Thomas Hartvigsen. Are language models actually useful for time series forecasting? *arXiv preprint arXiv:2406.16964*, 2024.

Kim Minh Vu. *The ARIMA and VARIMA time series: Their modelings, analyses and applications*. AuLac Technologies Inc., 2007.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv:2405.14616*, 2024.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. *PMLR*, 2024.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, volume 34, pp. 22419–22430, 2021.

Haixu Wu, Tengge Hu, Yong Liu, Huawei Zhou, Jingrui Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

Hao Xue and Flora D Salim. Promptcast: A new prompt-based learning paradigm for time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):6851–6864, 2023.

Luoxiao Yang, Zhi Zheng, and Zijun Zhang. An improved mixture density network via wasserstein distance based adversarial learning for probabilistic wind speed predictions. *IEEE Transactions on Sustainable Energy*, 13(2):755–766, 2021.

Huaxiu Yao, Yu Wang, Sai Li, Linjun Zhang, Weixin Liang, James Zou, and Chelsea Finn. Improving out-of-distribution robustness via selective augmentation. In *International Conference on Machine Learning*, pp. 25407–25437. PMLR, 2022.

Ailing Zeng, Mai Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 11121–11128, 2023.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11106–11115, 2021.

Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained lm. *arXiv preprint arXiv:2302.11939*, 2023a.

Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One Fits All: Power general time series analysis by pretrained lm. In *NeurIPS*, 2023b.

A Details of RealTS

We present RealTS, a versatile framework for synthesizing realistic time series data. RealTS employs multiple data behavior modes under two main hypotheses: periodic (φ_p) and trend (φ_t). This section details the various behavior modes, their configurations, and provides visual examples.

A.1 Periodic Hypothesis Behaviors

Under the periodic hypothesis φ_p, we employ two distinct data behavior modes:

A.1.1 Inverse Fast Fourier Transform Behavior (IFFTB)

To ensure the synthesized data adequately reflects the variation paradigms of real-world time series, we utilize IFFT as expressed in Equation (22) to simulate the underlying behavior of real-world periodic time series:

$$P\left(\mathbf{s_L}|L,B_p\right)|_{B_p=\text{IFFT}} = \iint_{-\infty}^{\infty} \mathbf{N}\left(\mathbf{A_m};\mu_{\mathbf{A_m}},\sigma^2_{\mathbf{A_m}}\right)\cdot\mathbf{N}\left(\phi;\mu_{\mathbf{P}},\sigma^2_{\mathbf{P}}\right)\times\delta\left(\mathbf{s_L}-\text{IFFT}\left(\mathbf{A_m},\phi,L\right)\right)d\phi d\mathbf{A_m} \quad (22)$$

where two empirical distributions of Fourier transform amplitudes and phases, $N(A_m;\mu_{A_m},\sigma^2_{A_m})$ and $N(\phi;\mu_P,\sigma^2_P)$, are maintained, and δ denotes the Dirac delta function. By sampling from empirical distributions, we can obtain the amplitude and Phase vector, which is then inversely transformed back to the time domain via IFFT.

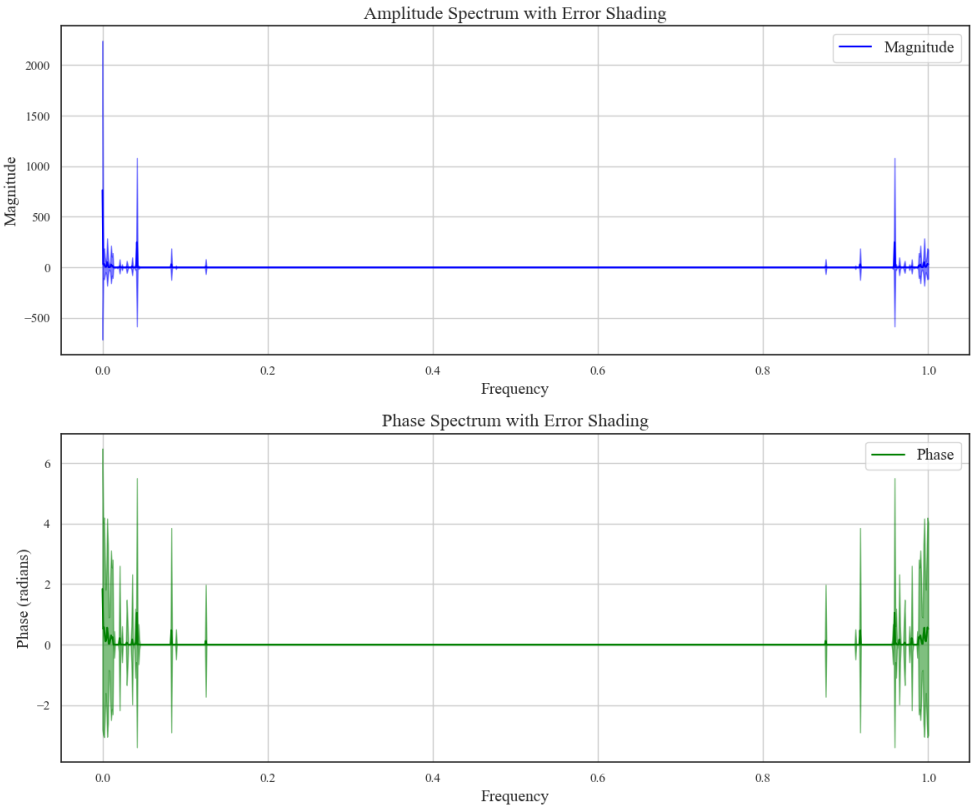


Figure 9: Empirical distribution I employed in IFFTB.

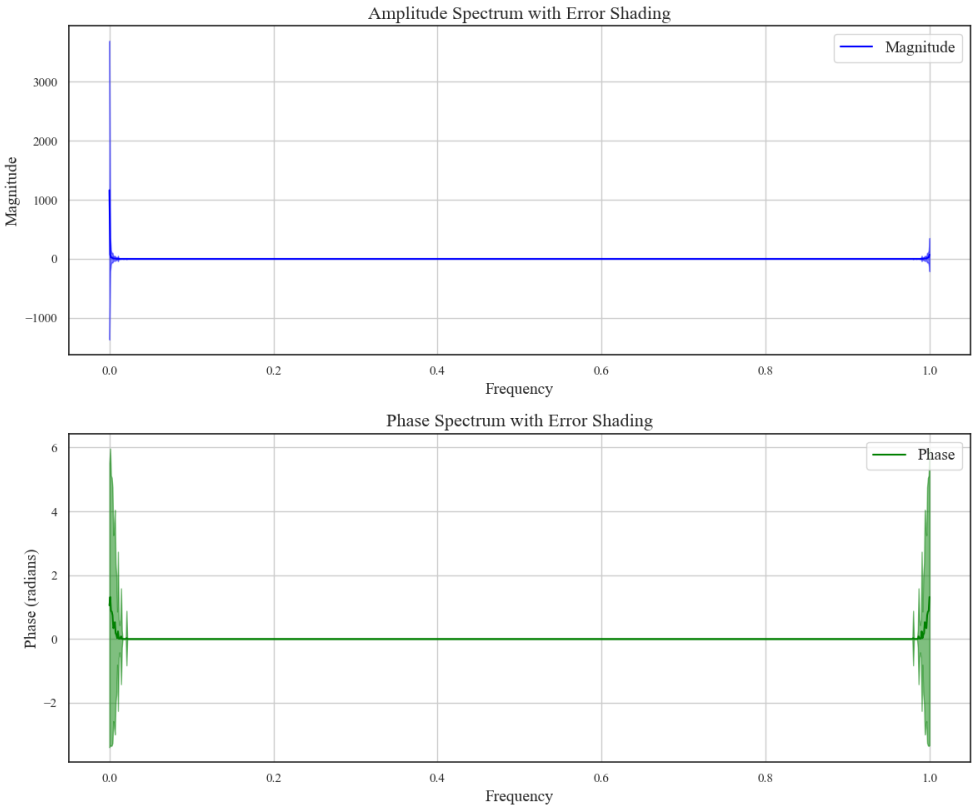


Figure 10: Empirical distribution II employed in IFFTB.

The empirical distributions utilized in $\mathbf{A_m}$ and ϕ are illustrated in Figure 9-Figure 10. During experiments, we randomly select one of two empirical distributions for generating $\mathbf{A_m}$ and ϕ. Figure 11 shows examples of time series generated using IFFTB.

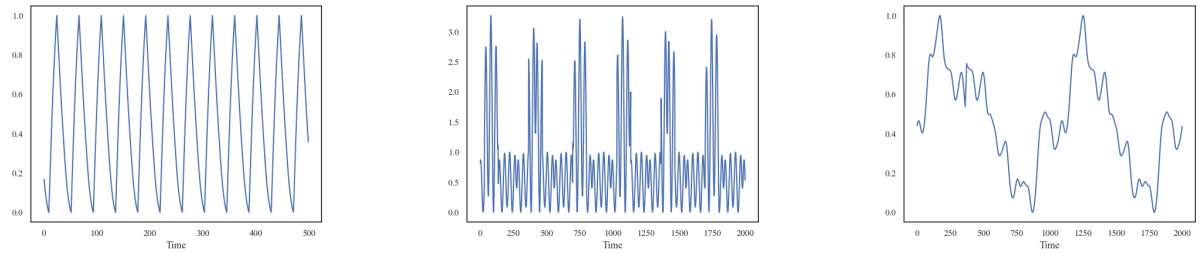


Figure 11: Examples of time series generated using IFFTB.

A.1.2 Periodic Wave Behavior (PWB)

This behavior generates data by superimposing multiple periodic waves, which is modeled as a sum of sin, cos, and other periodic functions, f_{periodic}, with different frequencies and amplitudes:

$$P\left(\mathbf{s_L}|L, B_p\right)|_{B_p=\text{PWB}} = \iint_{-\infty}^{\infty} \mathbf{N}\left(\mathbf{s_L}; \sum_{i=1}^{k_{\text{PWB}}} A_i f_{\text{periodic}}\left(\omega_i t\right), \sigma_\epsilon^2\right) \times \mathbf{P}\left(\mathbf{A}\right)\mathbf{P}\left(\omega\right) d\omega d\mathbf{A} \tag{23}$$

where $\mathbf{P}(\mathbf{A})$ and $\mathbf{P}(\boldsymbol{\omega})$ denote predefined prior distributions of amplitudes and frequency; k_{PWB} denotes the number of mixed periodic functions.

For PWB, we define the prior distributions for amplitude and frequency as:

$$\mathbf{A} \sim \mathbf{U}(\mathbf{0.5}, \mathbf{5}) \tag{24}$$

$$\ln(\omega) \sim \mathbf{U}(\ln(\mathbf{11}), \ln(\mathbf{2L})) \tag{25}$$

The parameter k_{PWB} is modeled as:

$$P(k_{PWB} = k) = \frac{1}{8}, \text{ for } k = 1, 2, \ldots, 8 \tag{26}$$

Figure 12 shows examples of time series generated using PWB.

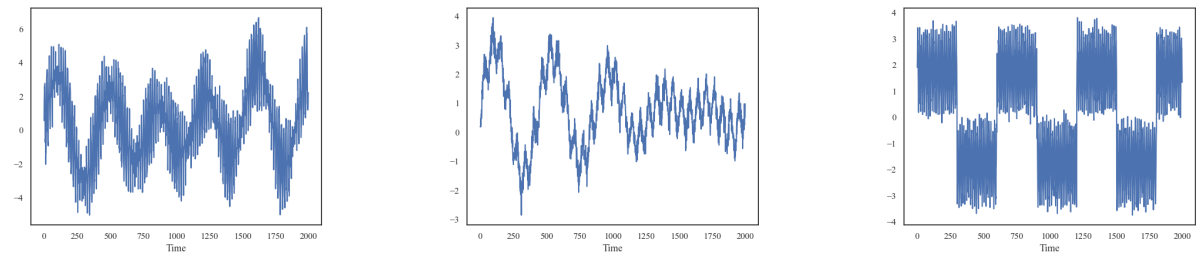


Figure 12: Examples of time series generated using PWB.

A.2 Trend Data Hypothesis Behaviors

Under the trend data hypothesis φ_t, we employ three distinct data behavior modes:

A.2.1 Random Walk Behavior (RWB)

The RWB models data as a stochastic process where each value is the previous value plus a random step:

$$P\left(s_i|s_{i-1}, L, B_p\right)|_{B_p=\text{RWB}} = \mathbf{N}\left(0, \sigma^2\right) \tag{27}$$

Figure 13 shows examples of time series generated using RWB.

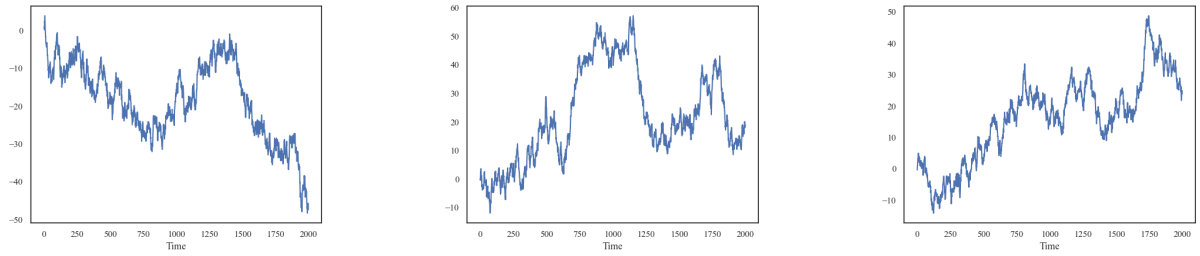


Figure 13: Examples of time series generated using RWB.

A.2.2 Logistic Growth Behavior (LGB)

The LGB models data with a logistic growth function, capturing the S-shaped growth pattern:

$$\begin{aligned} &P\left(\mathbf{s_L}|L, B_p\right)|_{B_p=\text{LGB}} \\ &= \iint_{-\infty}^{\infty} \mathbf{N}\left(\mathbf{s_L}; \frac{K}{1+e^{-r(\mathbf{L}-L_0)}}, \sigma_\epsilon^2\right) P(K)P(r)dKdr \end{aligned} \tag{28}$$

where $P(K)$ and $P(r)$ denote predefined prior distributions of S-shaped function hyperparameters.

For LGB, we define the probability densities for Carrying Capacity K and Growth Rate r as:

$$\ln(K) \sim U(\ln(1), \ln(10)) \tag{29}$$

$$\ln(r) \sim U(\ln(0.001), \ln(0.1)) \tag{30}$$

Figure 14 shows examples of time series generated using LGB.

A.2.3 Trend Wave Data Behavior (TWDB)

TWDB combines linear trends with periodic fluctuations:

$$P\left(\mathbf{s_L}|L, B_p\right)|_{B_p=\text{TWDB}} = \iint_{-\infty}^{\infty} \mathbf{N}\left(\mathbf{s_L}; a\mathbf{L} + b + \sum_{i=1}^{k_{\text{TWDB}}} A_i f_{\text{periodic}}\left(\omega_i t\right), \sigma_\epsilon^2\right) \times P(a)P(b)\mathbf{P}\left(\mathbf{A}\right)\mathbf{P}\left(\omega\right) dadbd\mathbf{A}d\omega \tag{31}$$

where $P(a)$, $P(b)$, $P\left(\mathbf{A}\right)$ and $\mathbf{P}\left(\omega\right)$ are predefined prior distributions of hyperparameters.

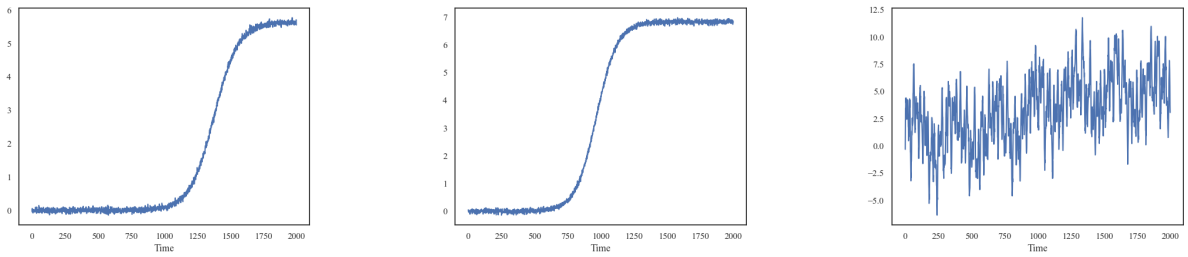


Figure 14: Examples of time series generated using LGB.

In the TWDB, we define the probability densities for linear function random variables $P(a)$ and $P(b)$, as well as for the superimposed periodic wave components $P(\mathbf{A})$ and $P(\omega)$. The settings for $P(\mathbf{A})$, $P(\omega)$, and k_{TWDB} are consistent with those used in the PWB module. The probability densities for $P(a)$ and $P(b)$ are detailed below:

$$a \sim U(-1, 1) \tag{32}$$

$$b \sim U(-10, 10) \tag{33}$$

Figure 15 shows examples of time series generated using TWDB.

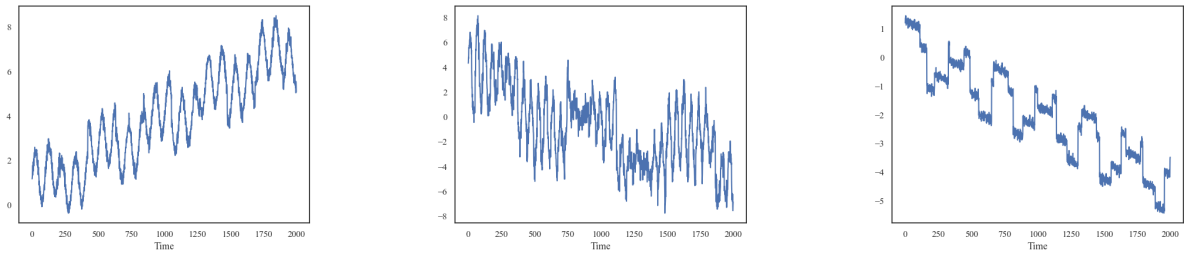


Figure 15: Examples of time series generated using TWDB.

A.3 Data Augmentation Techniques

To enhance the diversity and robustness of synthetic data, we employ various data augmentation techniques, including:

- Multiple period replication: Repeats the generated periodic data over multiple cycles to capture long-term periodic patterns.
- Data flipping: Reverses the time series to create new patterns while preserving underlying characteristics.
- Convolution smoothing and detrending: Removes underlying trends from the data to isolate periodic components, making it easier for the model to learn these patterns.
- Data perturbation: Introduces sudden changes or anomalies into the data, simulating real-world disturbances and improving the model's ability to handle unexpected variations.

More details of RealTS are offered in the code part of the Supplementary Material.

B Training configuration

B.1 ViTime model structure

Table 7: Details of model architecture

Module	Embed_dim	Depth	Patch size	Num_heads
Visual Time Tokenizer	768	9	(4,32)	12
Decoder	384	4	\	12

The detailed network configuration of the proposed ViTime is reported in Table 7.

B.2 Data normalization

To ensure ViTime can effectively capture patterns involving sudden changes, an in-sequence data normalization based on L2 normalization is implemented. By normalizing each sequence within the data sequence, the model can pay more attention to abrupt variations. The normalization process is defined as follows:

$$\mathbf{S_L} = \frac{\mathbf{S_L} - \text{mean}\left(\|\mathbf{S_{1:T}}\|_{\mathbf{2}}\right)}{\text{std}\left(\mathbf{S_{1:T}}\right)} \tag{34}$$

C Proofs

This section provides the detailed proofs for the theorems and propositions presented in the main text.

C.1 Proof of Theorem 3.3 (System Error Upper Bound)

Theorem C.1 (Theorem 3.3 restated)**.** *Given a tensor $\widehat{s} \in \mathcal{S} \subset \mathbb{R}^{c\times t}$, where $\mathcal{S}$ follows $\mathcal{N}(\mathbf{0},\mathbf{I})$ as per Assumption 3.2, the system error (SE) from mapping to $\mathcal{V}$ and back, defined as $\left\|f^{-1}\left(\mathbf{f}\left(\widehat{s}\right)\right)-\widehat{s}\right\|_1$, satisfies the following expectation bound:*

$$\text{SE} := \mathbb{E}\left\|f^{-1}\left(\mathbf{f}\left(\widehat{s}\right)\right)-\widehat{s}\right\|_1 \leq ct\left[MS\left(\frac{1}{h}\left(\Phi(MS)-\Phi(-MS)\right)-2+2\Phi(MS)\right)+\sqrt{\frac{2}{\pi}}e^{-\frac{MS^2}{2}}\right], \tag{35}$$

where Φ is the cumulative distribution function (CDF) of $\mathcal{N}(0,1)$, c is the number of variates, t is the time series length, h is the image height (resolution), and MS is the maximum scale.

Proof. The proof considers the error for a single element s of $\widehat{s}$ and then scales by ct. Let $P(s)$ be the PDF of $\mathcal{N}(0,1)$. The expected absolute error for a single element is $\mathbb{E}|f^{-1}(f(s))-s|$. This error can be decomposed into two parts: quantization error for $|s|\leq MS$ and truncation error for $|s|>MS$.

1. Quantization Error ($|s|\leq MS$):

When $|s|\leq MS$, the value s is mapped to a bin $j=\lfloor(s+MS)/(2MS/h)\rfloor$. The inverse mapping $f^{-1}(f(s))$ reconstructs this as the midpoint of the bin, $(j-0.5)(2MS/h)-MS$. The maximum error in this case is half the bin width, $\delta/2=(2MS/h)/2=MS/h$.

The expected quantization error is:

$$\begin{aligned} E_Q &= \int_{-MS}^{MS}|f^{-1}(f(s))-s|P(s)ds && (36)\\ &\leq \int_{-MS}^{MS}\frac{MS}{h}P(s)ds=\frac{MS}{h}\int_{-MS}^{MS}P(s)ds && (37)\\ &= \frac{MS}{h}\left[\Phi(MS)-\Phi(-MS)\right]. && (38)\end{aligned}$$

2. Truncation Error ($|s| > MS$):

If $s > MS$, $f(s)$ maps to the highest bin h, and $f^{-1}(f(s)) = MS - (MS/h)$. The error is $s - (MS - MS/h)$.

If $s < -MS$, $f(s)$ maps to the lowest bin 1, and $f^{-1}(f(s)) = -MS+(MS/h)$. The error is $(-MS+MS/h)-s$.

For simplicity in bounding, we consider the error magnitude as $|s| - MS$ when $|s| > MS$. The expected truncation error is:

$$E_T = \int_{MS}^{\infty}(s - MS)P(s)ds + \int_{-\infty}^{-MS}(-MS - s)P(s)ds \tag{39}$$

$$= 2\int_{MS}^{\infty}(s - MS)P(s)ds \quad \text{(by symmetry of } P(s)) \tag{40}$$

$$= 2\left[\int_{MS}^{\infty} sP(s)ds - MS\int_{MS}^{\infty} P(s)ds\right]. \tag{41}$$

We know $\int_{MS}^{\infty} sP(s)ds = \int_{MS}^{\infty} s\frac{1}{\sqrt{2\pi}}e^{-s^2/2}ds = \frac{1}{\sqrt{2\pi}}e^{-MS^2/2}$.

And $\int_{MS}^{\infty} P(s)ds = 1 - \Phi(MS)$.

So,

$$E_T = 2\left[\frac{1}{\sqrt{2\pi}}e^{-MS^2/2} - MS(1 - \Phi(MS))\right] \tag{42}$$

$$= \sqrt{\frac{2}{\pi}}e^{-MS^2/2} - 2MS(1 - \Phi(MS)). \tag{43}$$

3. Total Expected Error per Element:

The total expected absolute error for one element is $E_Q + E_T$:

$$\mathbb{E}|f^{-1}(f(s)) - s| \leq \frac{MS}{h}\left[\Phi(MS) - \Phi(-MS)\right] + \sqrt{\frac{2}{\pi}}e^{-MS^2/2} - 2MS(1 - \Phi(MS)) \tag{44}$$

$$= MS\left(\frac{1}{h}\left(\Phi(MS) - \Phi(-MS)\right) - 2(1 - \Phi(MS))\right) + \sqrt{\frac{2}{\pi}}e^{-\frac{MS^2}{2}} \tag{45}$$

$$= MS\left(\frac{1}{h}\left(\Phi(MS) - \Phi(-MS)\right) - 2 + 2\Phi(MS)\right) + \sqrt{\frac{2}{\pi}}e^{-\frac{MS^2}{2}}. \tag{46}$$

Multiplying by ct (number of elements) gives the bound for $\mathbb{E}\|f^{-1}(\mathbf{f}(\widehat{s})) - \widehat{s}\|_1$. □

C.2 Proof of Proposition 3.4 (Asymptotic Convergence with h)

Proposition C.2 (Proposition 3.4 restated)**.** *For any $\varepsilon > 0$, there exists $\delta_0 > 0$ such that when $h \to +\infty$ and $MS \geq \delta_0$, the per-element SE upper bound*

$$g_1(h, MS) = MS\left(\frac{1}{h}(\Phi(MS) - \Phi(-MS)) - 2 + 2\Phi(MS)\right) + \sqrt{\frac{2}{\pi}}e^{-\frac{MS^2}{2}} \tag{47}$$

converges to zero.

Proof. Let $g_1(h, MS)$ be the per-element upper bound from Theorem 3.3:

$$g_1(h, MS) = \frac{MS}{h}(\Phi(MS) - \Phi(-MS)) - 2MS(1 - \Phi(MS)) + \sqrt{\frac{2}{\pi}}e^{-\frac{MS^2}{2}}. \tag{48}$$

As $h \to +\infty$, the term $\frac{MS}{h}(\Phi(MS) - \Phi(-MS)) \to 0$ since $\Phi(MS) - \Phi(-MS) \leq 1$.

The remaining terms are $R(MS) = -2MS(1 - \Phi(MS)) + \sqrt{\frac{2}{\pi}} e^{-\frac{MS^2}{2}}$.

We use Mill's ratio for the tail probability of a standard normal distribution: for $MS > 0$,

$$1 - \Phi(MS) \sim \frac{\phi(MS)}{MS} = \frac{1}{MS\sqrt{2\pi}} e^{-MS^2/2} \quad \text{as } MS \to \infty. \tag{49}$$

So,

$$-2MS(1 - \Phi(MS)) \sim -2MS \left(\frac{1}{MS\sqrt{2\pi}} e^{-MS^2/2} \right) \tag{50}$$

$$= -\sqrt{\frac{2}{\pi}} e^{-MS^2/2}. \tag{51}$$

Thus, as $MS \to \infty$,

$$R(MS) \sim -\sqrt{\frac{2}{\pi}} e^{-MS^2/2} + \sqrt{\frac{2}{\pi}} e^{-MS^2/2} \tag{52}$$

$$= 0. \tag{53}$$

Therefore, for any $\varepsilon > 0$, we can find a δ_0 such that for $MS \geq \delta_0$, $|R(MS)| < \varepsilon/2$.

And for any $MS \geq \delta_0$, we can find an H_0 such that for $h \geq H_0$,

$$\left| \frac{MS}{h}(\Phi(MS) - \Phi(-MS)) \right| < \varepsilon/2. \tag{54}$$

This implies that $\lim_{h \to +\infty, MS \to \infty} g_1(h, MS) = 0$. More precisely, for a fixed large enough MS, as $h \to \infty$, the limit is $R(MS)$, which can be made arbitrarily small by choosing MS large.

The statement asks for convergence as $h \to \infty$ for $MS \geq \delta_0$.

Let $MS \geq \delta_0$. Then

$$\lim_{h \to +\infty} g_1(h, MS) = -2MS(1 - \Phi(MS)) + \sqrt{\frac{2}{\pi}} e^{-MS^2/2}. \tag{55}$$

This limit itself tends to 0 as $MS \to \infty$. The proposition asks for the expression to be small when $h \to \infty$ AND $MS \geq \delta_0$.

Taking the limit as $h \to \infty$ first, we get:

$$\lim_{h \to +\infty} \left| MS \left(\frac{1}{h}(\Phi(MS) - \Phi(-MS)) - 2 + 2\Phi(MS) \right) + \sqrt{\frac{2}{\pi}} e^{-\frac{MS^2}{2}} \right| = \left| -2MS(1 - \Phi(MS)) + \sqrt{\frac{2}{\pi}} e^{-\frac{MS^2}{2}} \right|. \tag{56}$$

This term goes to 0 as $MS \to \infty$. So, for any $\varepsilon > 0$, there exists δ_0 such that for $MS \geq \delta_0$, the term is less than ε. □

C.3 Proof of Proposition 3.5 (Optimal MS Selection)

Proposition C.3 (Proposition 3.5 restated)**.** *For a fixed h, there exists a unique optimal threshold $MS^* > 0$ that minimizes the per-element SE upper bound $g_1(h, MS)$. This MS^* is the solution to:*

$$\frac{1}{h}(\Phi(MS^*) - \Phi(-MS^*)) - 2 + 2\Phi(MS^*) + \frac{MS^*}{h} \cdot 2\phi(MS^*) = 0, \tag{57}$$

where $\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$ *is the PDF of* $\mathcal{N}(0,1)$*. (Note: The original equation had* $\sqrt{2/\pi} e^{-MS^{*2}/2}$*, which is* $2\phi(MS^*)$*).*

Proof. Let $g_1(MS) = MS\left(\frac{1}{h}(\Phi(MS) - \Phi(-MS)) - 2 + 2\Phi(MS)\right) + \sqrt{\frac{2}{\pi}} e^{-\frac{MS^2}{2}}$.

We want to find MS^* such that $g_1'(MS^*) = 0$.

Using $\Phi(-x) = 1 - \Phi(x)$ and $\phi(-x) = \phi(x)$, we have $\Phi(MS) - \Phi(-MS) = 2\Phi(MS) - 1$.

So, $g_1(MS) = MS\left(\frac{1}{h}(2\Phi(MS) - 1) - 2 + 2\Phi(MS)\right) + 2\phi(MS)$.

Derivative with respect to MS:

$$\frac{dg_1}{dMS} = \left(\frac{1}{h}(2\Phi(MS) - 1) - 2 + 2\Phi(MS)\right) + MS\left(\frac{2\phi(MS)}{h} + 2\phi(MS)\right) + 2\phi'(MS) \tag{58}$$

$$= \frac{2\Phi(MS) - 1}{h} - 2 + 2\Phi(MS) + \frac{2MS\phi(MS)}{h} + 2MS\phi(MS) - 2MS\phi(MS) \quad (\text{since } \phi'(MS) = -MS\phi(MS)) \tag{59}$$

$$= \frac{2\Phi(MS) - 1}{h} - 2 + 2\Phi(MS) + \frac{2MS\phi(MS)}{h}. \tag{60}$$

Setting $dg_1/dMS = 0$:

$$\frac{2\Phi(MS^*) - 1}{h} - 2 + 2\Phi(MS^*) + \frac{2MS^*\phi(MS^*)}{h} = 0. \tag{61}$$

This matches the condition in the proposition since $\Phi(MS^*) - \Phi(-MS^*) = 2\Phi(MS^*) - 1$.

To show uniqueness and minimality, we examine the second derivative or the behavior of the first derivative.

Let $f(MS) = dg_1/dMS$.

$f(0) = (0-1)/h - 2 + 2(0.5) + 0 = -1/h - 2 + 1 = -1 - 1/h < 0$.

As $MS \to \infty$, $\Phi(MS) \to 1$ and $\phi(MS) \to 0$.

So $\lim_{MS\to\infty} f(MS) = 1/h - 2 + 2 + 0 = 1/h > 0$ (assuming $h > 0$).

Since $f(MS)$ is continuous and goes from negative to positive, there must be at least one root $MS^* > 0$.

The second derivative:

$$\frac{d^2 g_1}{dMS^2} = \frac{2\phi(MS)}{h} + 2\phi(MS) + \frac{2\phi(MS) + 2MS\phi'(MS)}{h} \tag{62}$$

$$= 2\phi(MS)\left(\frac{1}{h} + 1\right) + \frac{2\phi(MS) - 2MS^2\phi(MS)}{h} \tag{63}$$

$$= 2\phi(MS)\left(1 + \frac{2}{h} - \frac{MS^2}{h}\right). \tag{64}$$

For small MS, $d^2 g_1/dMS^2 > 0$, indicating convexity. If $1 + 2/h - MS^2/h > 0$, i.e., $MS^2 < h + 2$.

If $MS^* < \sqrt{h+2}$, then $g_1(MS)$ is convex at MS^*, ensuring a local minimum.

The function $f(MS)$ starts negative, becomes positive, and its derivative $d^2 g_1/dMS^2$ is positive for $MS < \sqrt{h+2}$ and can become negative for $MS > \sqrt{h+2}$. This structure ensures a unique minimum for $MS > 0$. □

C.4 Proof of Proposition 3.6 (Optimal Threshold under Variance Scaling)

Proposition C.4 (Proposition 3.6 restated)**.** *Under the assumption* $\mathcal{S} \sim \mathcal{N}(0, k\mathbf{I})$ *with* $k > 1$*, the optimal threshold* MS^* *that minimizes the per-element SE upper bound is characterized by:*

$$\frac{1}{h}\left(\Phi\left(\frac{MS^*}{\sqrt{k}}\right) - \Phi\left(-\frac{MS^*}{\sqrt{k}}\right)\right) - 2 + 2\Phi\left(\frac{MS^*}{\sqrt{k}}\right) + \frac{MS^*}{h}\sqrt{\frac{2}{\pi k}} e^{-\frac{(MS^*)^2}{2k}} = 0. \tag{65}$$

Proof. Let $s \sim \mathcal{N}(0, k)$. Then $s' = s/\sqrt{k} \sim \mathcal{N}(0, 1)$.

The original values s are scaled by $\sqrt{k}$. The mapping function f operates on s. The bins are from $-MS$ to MS. Bin width $\delta_s = 2MS/h$.

The SE upper bound for a single element is: $g_k(MS) = \mathbb{E}_{s\sim\mathcal{N}(0,k)}|f^{-1}(f(s)) - s|$.

This is equivalent to scaling the original problem. Let $s = \sqrt{k}z$ where $z \sim \mathcal{N}(0, 1)$. The function operates on s. The effective range for z is $-MS/\sqrt{k}$ to $MS/\sqrt{k}$.

The quantization error part:

s is in $[-MS, MS]$. The error is bounded by MS/h.

$$\int_{-MS}^{MS} \frac{MS}{h} P_k(s)ds = \frac{MS}{h} \int_{-MS}^{MS} \frac{1}{\sqrt{2\pi k}} e^{-s^2/(2k)} ds \tag{66}$$

Let $u = s/\sqrt{k}$. Then $ds = \sqrt{k}du$. Limits become $-MS/\sqrt{k}$ to $MS/\sqrt{k}$.

$$= \frac{MS}{h} \int_{-MS/\sqrt{k}}^{MS/\sqrt{k}} \frac{1}{\sqrt{2\pi}} e^{-u^2/2} du \tag{67}$$

$$= \frac{MS}{h} \left[\Phi\left(\frac{MS}{\sqrt{k}}\right) - \Phi\left(-\frac{MS}{\sqrt{k}}\right)\right] \tag{68}$$

The truncation error part:

$2\int_{MS}^{\infty}(s - MS)P_k(s)ds$.

$$= 2\left[\int_{MS}^{\infty} s \frac{1}{\sqrt{2\pi k}} e^{-s^2/(2k)} ds - MS \int_{MS}^{\infty} \frac{1}{\sqrt{2\pi k}} e^{-s^2/(2k)} ds\right] \tag{69}$$

The first integral:

$$\int_{MS}^{\infty} s \frac{1}{\sqrt{2\pi k}} e^{-s^2/(2k)} ds = \sqrt{k} \int_{MS/\sqrt{k}}^{\infty} u \frac{1}{\sqrt{2\pi}} e^{-u^2/2} du \tag{70}$$

$$= \sqrt{k} \frac{1}{\sqrt{2\pi}} e^{-(MS/\sqrt{k})^2/2} \tag{71}$$

$$= \sqrt{\frac{k}{2\pi}} e^{-MS^2/(2k)} \tag{72}$$

The second integral: $MS\left(1 - \Phi\left(\frac{MS}{\sqrt{k}}\right)\right)$.

So,

$$E_{T,k} = 2\left[\sqrt{\frac{k}{2\pi}} e^{-MS^2/(2k)} - MS\left(1 - \Phi\left(\frac{MS}{\sqrt{k}}\right)\right)\right] \tag{73}$$

The per-element SE bound $g_{1,k}(MS)$ is:

$$g_{1,k}(MS) = \frac{MS}{h}\left[\Phi_k(MS) - \Phi_k(-MS)\right] + \sqrt{\frac{2k}{\pi}} e^{-MS^2/(2k)} - 2MS(1 - \Phi_k(MS)) \tag{74}$$

where $\Phi_k(x) = \Phi(x/\sqrt{k})$.

This can be written as:

$$g_{1,k}(MS) = MS\left(\frac{1}{h}(\Phi_k(MS) - \Phi_k(-MS)) - 2 + 2\Phi_k(MS)\right) + \sqrt{\frac{2k}{\pi}}e^{-\frac{MS^2}{2k}} \tag{75}$$

To find the optimal MS^*, we differentiate $g_{1,k}(MS)$ with respect to MS and set to zero.

Let $\phi_k(x) = \frac{1}{\sqrt{k}}\phi(x/\sqrt{k})$ be the PDF of $\mathcal{N}(0,k)$ in terms of ϕ.

$\frac{d}{dMS}\Phi_k(MS) = \frac{d}{dMS}\Phi(MS/\sqrt{k}) = \phi(MS/\sqrt{k}) \cdot \frac{1}{\sqrt{k}} = \phi_k(MS)$.

$\frac{d}{dMS}\left(\sqrt{\frac{2k}{\pi}}e^{-\frac{MS^2}{2k}}\right) = \sqrt{\frac{2k}{\pi}}e^{-\frac{MS^2}{2k}}\left(-\frac{2MS}{2k}\right) = -\frac{MS}{\sqrt{k}}\sqrt{\frac{2}{\pi}}e^{-\frac{MS^2}{2k}} = -2MS\phi_k(MS)$.

The derivative $\frac{dg_{1,k}}{dMS}$ is:

$$= \left(\frac{1}{h}(\Phi_k(MS) - \Phi_k(-MS)) - 2 + 2\Phi_k(MS)\right) + MS\left(\frac{1}{h}(\phi_k(MS) - (-\phi_k(MS))) + 2\phi_k(MS)\right) - 2MS\phi_k(MS) \tag{76}$$

$$= \frac{\Phi_k(MS) - \Phi_k(-MS)}{h} - 2 + 2\Phi_k(MS) + \frac{2MS\phi_k(MS)}{h} \tag{77}$$

Setting this to zero gives:

$$\frac{1}{h}\left(\Phi\left(\frac{MS^*}{\sqrt{k}}\right) - \Phi\left(-\frac{MS^*}{\sqrt{k}}\right)\right) - 2 + 2\Phi\left(\frac{MS^*}{\sqrt{k}}\right) + \frac{2MS^*}{h}\frac{1}{\sqrt{k}}\phi\left(\frac{MS^*}{\sqrt{k}}\right) = 0 \tag{78}$$

Substituting $2\phi(x/\sqrt{k})/\sqrt{k} = \sqrt{2/(\pi k)}e^{-(MS^*)^2/(2k)}$, we get the stated condition.

The uniqueness follows a similar argument to Proposition 3.5. □

C.5 Proof of Theorem 3.7 (Stripe SNR Boost)

Theorem C.5 (Theorem 3.7 restated)**.** *Let the length–L time series $s_k = A\sin(\omega_0 k+\phi)+\eta_k$, $k = 0,\dots,L-1$, with amplitude $A > 0$, angular frequency $\omega_0 = 2\pi/P_{\text{period}}$ ($P_{\text{period}} \in \mathbb{N}^+$) and i.i.d. Gaussian noise $\eta_k \sim \mathcal{N}(0,\sigma^2)$ be visualised as the binary stripe image $v \in \{0,1\}^{h\times L}$ defined through $v_{j,k} = \mathbf{1}(j = \lfloor(s_k + \text{MS})/\delta\rfloor)$, where $\delta = \Delta/h$, $\Delta = 2\text{MS}$.*

Denote $SNR_{\text{num}} = A^2/(2\sigma^2)$ and $SNR_{\text{vis}} = \mathbb{E}[|\mathcal{F}_{2D}(v_{\text{clean}})[0,n_0]|^2]/\mathbb{E}[|\mathcal{F}_{2D}(v_{\text{noise}})[0,n_0]|^2]$, with $n_0 = \lfloor L/P_{\text{period}}\rfloor$.

Assume (i) $\delta \le A \le \Delta - \delta$ and (ii) $\sigma < \delta/4$.

Then, for every $L \ge P_{\text{period}}$:

$$SNR_{\text{vis}} \ge \frac{L}{4}\exp\left(\frac{\delta^2}{8\sigma^2}\right)\frac{\sigma^2}{A^2}SNR_{\text{num}} \tag{79}$$

$$SNR_{\text{vis}} \ge \frac{L}{4}\exp\left(\frac{\delta^2}{8\sigma^2}\right) \tag{80}$$

Proof. Let v_{clean} be the image from $A\sin(\omega_0 k + \phi)$ and $v = v_{\text{clean}} + v_{\text{noise}}$ where v_{noise} is the change due to η_k.

1. Deterministic Signal Power in Visual Domain.

The 2D Discrete Fourier Transform (DFT) is

$$\mathcal{F}_{2D}(v)[m,n] = \sum_{k=0}^{L-1}\sum_{j=0}^{h-1} v_{j,k}e^{-i2\pi(mk/L+nj/h)} \tag{81}$$

We are interested in the coefficient at $(m, n_p) = (0, n_0)$, where $n_0 = L/P_{\text{period}}$ (assuming L is a multiple of P_{period} for simplicity, or $\lfloor L/P_{\text{period}} \rfloor$ otherwise).

The transform of the clean signal component at $(0, n_0)$ is

$$\mathcal{F}_{2D}(v_{\text{clean}})[0, n_0] = \sum_{k=0}^{L-1} \sum_{j=0}^{h-1} (v_{\text{clean}})_{j,k} e^{-i2\pi(n_0 j/h)} \tag{82}$$

Following the provided analysis, $\mathbb{E}[|\mathcal{F}_{2D}(v_{\text{clean}})[0, n_0]|^2] = L^2$.

2. Probability of Quantization Flip.

A flip means $v_{j,k}$ changes due to noise η_k. This occurs if $s_k = s_{k,\text{clean}} + \eta_k$ crosses a quantization boundary $\theta_j = j\delta - MS$.

The clean value $s_{k,\text{clean}}$ falls into bin j_0. A flip occurs if s_k falls into $j_0 \pm 1, j_0 \pm 2, \ldots$.

The closest boundaries are $j_0\delta - MS$ and $(j_0 + 1)\delta - MS$.

$s_{k,\text{clean}}$ is at least ϵ from any boundary. A flip to an adjacent bin occurs if $|\eta_k| > \epsilon$.

The condition $\sigma < \delta/4$ implies noise is small. A flip occurs if η_k moves s_k to another bin. This primarily happens if s_k crosses $s_{k,\text{clean}} \pm \delta/2$ (approximately).

So, $p_{\text{flip}} = \Pr(|\eta_k| > \delta/2)$. Using Gaussian tail bound $\Pr(|X| > t) \leq 2e^{-t^2/(2\sigma^2)}$:

$$p_{\text{flip}} \leq 2 \exp\left(-\frac{(\delta/2)^2}{2\sigma^2}\right) = 2 \exp\left(-\frac{\delta^2}{8\sigma^2}\right) \tag{83}$$

3. Energy of the Noise Image v_{noise}.

v_{noise} has entries $1, -1, 0$. If s_k flips from bin j_0 to j_1: $(v_{\text{noise}})_{j_0,k} = -1$, $(v_{\text{noise}})_{j_1,k} = 1$.

$\|v_{\text{noise}}\|_F^2 = \sum_{k,j} (v_{\text{noise}})_{j,k}^2$. Each flip changes two pixels, so contributes $1^2 + (-1)^2 = 2$ to this sum.

$\mathbb{E}[\|v_{\text{noise}}\|_F^2] = \sum_k \mathbb{E}[\text{contribution at } k] = L \cdot (2 \cdot p_{\text{flip}})$.

4. Bound on a Single DFT Coefficient of Noise.

By Parseval's identity for 2D DFT: $\sum_{m,n} |\mathcal{F}_{2D}(v_{\text{noise}})[m, n]|^2 = \|v_{\text{noise}}\|_F^2$ (with appropriate normalization).

Thus, for any specific (m, n), $|\mathcal{F}_{2D}(v_{\text{noise}})[m, n]|^2 \leq \|v_{\text{noise}}\|_F^2$.

Therefore, $\mathbb{E}[|\mathcal{F}_{2D}(v_{\text{noise}})[0, n_0]|^2] \leq \mathbb{E}[\|v_{\text{noise}}\|_F^2] = 2Lp_{\text{flip}}$.

5. Visual SNR Bound.

$$\text{SNR}_{\text{vis}} = \frac{L^2}{\mathbb{E}[|\mathcal{F}_{2D}(v_{\text{noise}})[0, n_0]|^2]} \tag{84}$$

$$\geq \frac{L^2}{2Lp_{\text{flip}}} \tag{85}$$

$$= \frac{L}{2p_{\text{flip}}} \tag{86}$$

Using $p_{\text{flip}} \leq 2\exp(-\delta^2/(8\sigma^2))$:

$$\text{SNR}_{\text{vis}} \geq \frac{L}{2 \cdot 2\exp(-\delta^2/(8\sigma^2))} \tag{87}$$

$$= \frac{L}{4}\exp\left(\frac{\delta^2}{8\sigma^2}\right) \tag{88}$$

This is the second part of the result.

6. Relation to Numerical SNR.

$\text{SNR}_{\text{num}} = A^2/(2\sigma^2)$.

$$\frac{\text{SNR}_{\text{vis}}}{\text{SNR}_{\text{num}}} = \text{SNR}_{\text{vis}}\frac{2\sigma^2}{A^2} \tag{89}$$

$$\geq \frac{L}{4}\exp\left(\frac{\delta^2}{8\sigma^2}\right)\frac{2\sigma^2}{A^2} \tag{90}$$

This yields

$$\text{SNR}_{\text{vis}} \geq \frac{L}{4}\exp\left(\frac{\delta^2}{8\sigma^2}\right)\frac{\sigma^2}{A^2}\text{SNR}_{\text{num}} \tag{91}$$

Thus, both inequalities in the theorem statement are proven. □

C.6 Proof of Theorem 3.8 (Gaussian-Blur SNR Boost)

Theorem C.6 (Theorem 3.8 restated)**.** *Under the assumptions of Theorem 3.7, apply a 1D normalized Gaussian convolution $g_j = (1/Z)\exp(-j^2/(2\sigma_b^2))$ with $\sum_j g_j = 1$ along the row direction of v to get $w = g *_j v$. Let $S = \sum_j g_j^2 \in (0,1)$ be the filter's nuclear energy.*

Define $SNR_{\text{vis}}^{blur} = \mathbb{E}[|\mathcal{F}_{2D}(w_{\text{clean}})[0,n_0]|^2]/\mathbb{E}[|\mathcal{F}_{2D}(w_{\text{noise}})[0,n_0]|^2]$. Then,

$$SNR_{\text{vis}}^{blur} \geq \frac{L}{4S}\exp\left(\frac{\delta^2}{8\sigma^2}\right) \tag{92}$$

$$SNR_{\text{vis}}^{blur} \geq \frac{L\sigma^2}{2A^2S}\exp\left(\frac{\delta^2}{8\sigma^2}\right) SNR_{\text{num}} \tag{93}$$

The visual SNR is amplified by at least $1/S > 1$ compared to the unblurred case.

Proof. Let $G(m_v)$ be the DFT of the 1D filter g_j with respect to the value-axis frequency m_v.

$\mathcal{F}_{2D}(w)[m_t, m_v] = G(m_v)\mathcal{F}_{2D}(v)[m_t, m_v]$.

We are interested in the frequency $(0, n_0)$, where n_0 is the time-axis frequency index.

1. Signal Power after Blurring.

Using the frequency index (n_t, n_j) for (time, value/row), we examine the coefficient $\mathcal{F}_{2D}(w)[n_t, n_j]$.

The specific coefficient in focus is $\mathcal{F}_{2D}(w_{\text{clean}})[0, n_0]$, where n_0 is the time index.

Signal power:

$$\mathbb{E}[|\mathcal{F}_{2D}(w_{\text{clean}})[n_0, 0]|^2] = |G(0)|^2\mathbb{E}[|\mathcal{F}_{2D}(v_{\text{clean}})[n_0, 0]|^2] \tag{94}$$

Since $\sum_j g_j = 1$, we have $G(0) = 1$. So signal power remains L^2.

2. Noise Power after Blurring.

The noise image is $w_{\text{noise}} = g *_j v_{\text{noise}}$.

The total energy of w_{noise}:

$$\|w_{\text{noise}}\|_F^2 = \sum_k \|g * (v_{\text{noise}})_{:,k}\|_2^2 \tag{95}$$

For each column k, $(v_{\text{noise}})_{:,k}$ is a vector. Convolution is along j.

$$\|g * (v_{\text{noise}})_{:,k}\|_2^2 = S\|(v_{\text{noise}})_{:,k}\|_2^2 \tag{96}$$

where $S = \|g\|_2^2 = \sum_j g_j^2$.

So,

$$\|w_{\text{noise}}\|_F^2 = S\|v_{\text{noise}}\|_F^2 \tag{97}$$

$$\mathbb{E}[\|w_{\text{noise}}\|_F^2] = S \cdot \mathbb{E}[\|v_{\text{noise}}\|_F^2] \tag{98}$$

$$= S \cdot (2Lp_{\text{flip}}) \tag{99}$$

3. Bound on Single DFT Coefficient of Blurred Noise.

$$\mathbb{E}[|\mathcal{F}_{2D}(w_{\text{noise}})[n_0, 0]|^2] \leq \mathbb{E}[\|w_{\text{noise}}\|_F^2] \tag{100}$$

$$= 2LSp_{\text{flip}} \tag{101}$$

4. SNR after Blurring.

$$\text{SNR}_{\text{vis}}^{\text{blur}} = \frac{L^2}{\mathbb{E}[|\mathcal{F}_{2D}(w_{\text{noise}})[n_0, 0]|^2]} \tag{102}$$

$$\geq \frac{L^2}{2LSp_{\text{flip}}} \tag{103}$$

$$= \frac{L}{2Sp_{\text{flip}}} \tag{104}$$

Using $p_{\text{flip}} \leq 2\exp(-\delta^2/(8\sigma^2))$:

$$\text{SNR}_{\text{vis}}^{\text{blur}} \geq \frac{L}{2S \cdot 2\exp(-\delta^2/(8\sigma^2))} \tag{105}$$

$$= \frac{L}{4S}\exp\left(\frac{\delta^2}{8\sigma^2}\right) \tag{106}$$

This means $\text{SNR}_{\text{vis}}^{\text{blur}} \geq (1/S) \cdot \text{SNR}_{\text{vis}}$ (unblurred).

For the relation to numerical SNR:

$$\text{SNR}_{\text{vis}}^{\text{blur}} \geq \frac{L}{4S}\exp\left(\frac{\delta^2}{8\sigma^2}\right) \tag{107}$$

$$\geq \frac{L}{4S}\exp\left(\frac{\delta^2}{8\sigma^2}\right)\frac{2\sigma^2}{A^2} \cdot \frac{A^2}{2\sigma^2} \tag{108}$$

$$= \frac{L\sigma^2}{2A^2S}\exp\left(\frac{\delta^2}{8\sigma^2}\right)\text{SNR}_{\text{num}} \tag{109}$$

Since $S < 1$, the factor $1/S > 1$ provides an amplification of the SNR compared to the unblurred case. □

D Additional results of computational experiments

D.1 Zero-shot study

The full results of the zero-shot study are reported in Table 8 - Table 12. We also illustrate zero-shot TSF examples with prediction length equals 720 of the proposed ViTime versus TimesFM in Figure 16 - Figure 21. It is observable that ViTime consistently demonstrates superior zero-shot prediction performance compared to TimesFM across a range of rescale factors.

D.2 Fine-tuning study

Complete results of the fine-tuning study are reported in Table 13.

D.3 Robust inference study

Complete results of the robust inference study are reported in Table 14.

D.4 Computational complexity analysis

We conduct extensive experiments to analyze the computational complexity and prediction accuracy of our proposed models. All experiments are performed with batch size 4, input sequence length 512, and prediction horizon 720 on a single Nvidia 3090 GPU.

Table 15: Model Performance and Computational Resource Requirements

Model	GPU Mem.	Inference Time (s/batch)		Params	Avg. ReMAE
	(MB)	Total	Map. & Inv. Map.	(M)	
TimesFM	18,154	0.130	-	200	0.423
ViTime w/ Refining	3,120	2.890	0.0068	95	**0.376**
ViTime w/o Refining	**667**	**0.082**	0.0068	**74**	0.381

The results are reported in Table 15. The baseline TimesFM model requires substantial computational resources with 18.1GB GPU memory and 200M parameters, while achieving an average ReMAE of 0.423. In contrast, our proposed ViTime architecture demonstrates remarkable improvements in both efficiency and accuracy. The basic version without the refining module strikes an optimal balance between computational efficiency and performance - it requires only 667MB GPU memory (27× reduction), achieves faster inference at 0.082s per batch, uses 63% fewer parameters (74M), while maintaining competitive accuracy with an average ReMAE of 0.381.

For applications prioritizing prediction accuracy, the ViTime variant with refining module achieves the best performance with an average ReMAE of 0.376, representing an 11.1% improvement over TimesFM. This comes at the cost of increased computational overhead - 3.1GB GPU memory and 2.89s inference time per batch, though still maintaining a 5.8× reduction in memory compared to TimesFM. Notably, the mapping & inverse mapping between image space and numerical space in ViTime variants consume only 0.0068s, representing 8.3% and 0.24% of the total inference time for the basic and refined versions, respectively.

These results demonstrate that our proposed architecture offers flexible deployment options: the basic version for resource-constrained scenarios requiring good accuracy and computational efficiency, and the refined version for applications where prediction accuracy is paramount. Both variants significantly outperform the baseline in terms of the computation-accuracy trade-off.

Table 8: Computational results for Scale = 0.5 (Landscape). The best results are **bolded**, the second best are underlined.

Dataset	H	ViTime		ViTime-TFM		TimesFM		Moment		Moriai		VisionTS		PatchTST-ZS	
		ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE
ETTh1	96	0.454	0.448	0.450	0.460	**0.308**	**0.361**	2.334	4.666	1.713	3.316	1.673	2.944	1.425	0.919
	192	0.437	0.449	0.452	0.470	**0.335**	**0.383**	2.349	4.781	2.136	4.229	1.675	3.050	1.519	0.953
	336	0.454	0.459	0.494	0.490	**0.412**	**0.383**	2.438	5.044	2.177	4.351	1.810	3.450	1.513	0.951
	720	0.555	0.510	**0.550**	**0.500**	0.617	0.560	2.601	5.506	3.490	6.942	2.055	4.109	1.521	0.952
ETTh2	96	**0.266**	**0.328**	0.311	0.375	0.305	0.344	1.560	3.027	1.810	3.270	1.650	2.875	1.359	0.918
	192	**0.288**	**0.348**	0.314	0.383	0.333	0.377	1.615	3.171	1.663	3.175	1.642	2.987	1.407	0.930
	336	**0.298**	**0.362**	0.361	0.398	0.406	0.434	1.833	3.635	2.019	4.045	1.772	3.353	1.408	0.928
	720	0.399	0.435	0.429	**0.405**	0.585	0.548	2.196	4.503	3.022	5.391	2.027	4.023	1.409	0.929
ETTm1	96	0.563	0.442	0.538	0.470	**0.411**	0.403	0.884	0.606	0.781	0.482	0.668	**0.398**	1.251	0.805
	192	0.571	0.450	0.571	0.485	**0.447**	0.427	0.869	0.597	0.759	0.479	0.622	**0.382**	1.311	0.823
	336	0.583	0.460	0.618	0.515	**0.485**	0.454	0.839	0.582	1.280	0.871	0.609	**0.393**	1.355	0.838
	720	0.595	0.475	0.644	0.530	**0.537**	0.493	0.786	0.565	1.401	0.979	0.622	**0.433**	1.335	0.831
ETTm2	96	**0.222**	0.297	0.274	0.345	0.269	0.317	0.505	0.563	0.317	**0.206**	0.547	0.355	0.719	0.606
	192	**0.280**	0.336	0.314	0.353	0.342	0.357	0.531	0.598	0.358	**0.230**	0.531	0.345	0.818	0.639
	336	**0.321**	0.366	0.348	0.368	0.433	0.415	0.645	0.726	0.639	0.373	0.536	**0.349**	0.846	0.650
	720	0.395	0.411	**0.371**	**0.375**	0.522	0.483	0.738	0.824	0.661	0.380	0.564	0.367	0.840	0.647
Traffic	96	0.839	0.431	0.821	0.470	**0.515**	0.383	1.149	0.798	0.847	0.434	0.687	**0.355**	1.575	0.854
	192	0.724	0.405	0.825	0.480	**0.539**	0.402	1.195	0.805	0.882	0.470	0.666	**0.303**	1.544	0.870
	336	0.723	0.403	0.855	0.500	**0.612**	0.429	1.164	0.799	0.860	0.491	0.664	**0.330**	1.595	0.869
	720	**0.693**	0.409	0.882	0.510	0.795	0.505	1.163	0.804	1.599	1.224	0.691	**0.342**	1.529	0.867
Weather	96	**0.172**	**0.212**	0.228	0.278	0.209	0.253	0.538	0.332	0.635	0.380	0.537	0.301	0.871	0.597
	192	**0.208**	**0.252**	0.254	0.283	0.289	0.312	0.552	0.344	0.751	0.467	0.558	0.329	0.955	0.623
	336	**0.272**	0.297	0.288	**0.293**	0.365	0.360	0.567	0.350	1.082	0.637	0.589	0.367	0.954	0.620
	720	0.338	0.339	**0.314**	**0.298**	0.471	0.432	0.702	0.461	1.568	0.897	0.695	0.459	0.988	0.629
Electricity	96	0.267	0.339	0.256	0.345	**0.184**	**0.279**	0.951	0.795	0.585	0.386	0.533	0.390	1.131	0.822
	192	0.264	0.343	0.269	0.353	**0.220**	**0.308**	0.963	0.805	0.660	0.448	0.498	0.331	1.195	0.846
	336	0.289	0.363	0.295	0.368	**0.277**	**0.347**	0.980	0.819	0.764	0.503	0.549	0.374	1.201	0.848
	720	**0.320**	0.385	0.342	**0.375**	0.406	0.432	0.982	0.820	1.614	1.249	0.538	0.362	1.229	0.851

Table 9: Computational results for Scale = 0.66 (Landscape). The best results are **bolded**, the second best are underlined.

Dataset	H	ViTime		ViTime-TFM		TimesFM		Moment		Moriai		VisionTS		PatchTST-ZS	
		ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE
ETTh1	96	0.492	0.453	**0.449**	**0.450**	0.587	0.516	0.778	0.558	0.825	0.541	0.827	0.539	1.827	1.011
	192	0.485	**0.458**	**0.474**	0.460	0.655	0.555	0.799	0.576	0.940	0.642	0.820	0.540	1.953	1.044
	336	0.506	**0.471**	**0.500**	0.480	0.711	0.588	0.825	0.601	1.133	0.770	0.788	0.542	1.979	1.051
	720	0.553	0.503	**0.545**	**0.490**	0.798	0.645	0.878	0.656	1.173	0.811	0.840	0.605	1.978	1.051
ETTh2	96	**0.230**	**0.299**	0.290	0.365	0.277	0.342	0.508	0.332	0.637	0.395	0.572	0.354	1.273	0.843
	192	**0.254**	**0.319**	0.317	0.373	0.344	0.394	0.527	0.353	0.583	0.378	0.583	0.373	1.330	0.863
	336	**0.303**	**0.357**	0.344	0.388	0.392	0.431	0.549	0.375	0.650	0.449	0.575	0.381	1.328	0.863
	720	**0.372**	0.414	0.390	**0.395**	0.508	0.509	0.672	0.472	0.913	0.571	0.656	0.450	1.328	0.863
ETTm1	96	**0.475**	**0.411**	0.499	0.460	0.490	0.441	0.936	0.643	1.117	0.713	1.210	0.794	1.344	0.825
	192	0.522	**0.433**	0.532	0.475	**0.502**	0.461	0.924	0.636	1.112	0.698	0.978	0.659	1.463	0.860
	336	**0.547**	**0.447**	0.597	0.505	0.562	0.495	0.919	0.631	1.300	0.873	0.978	0.660	1.477	0.861
	720	**0.596**	**0.475**	0.652	0.520	0.669	0.551	0.871	0.602	1.471	0.999	0.959	0.642	1.449	0.853
ETTm2	96	**0.199**	0.278	0.259	0.335	0.258	0.303	0.442	0.488	0.371	**0.237**	0.989	0.643	0.787	0.622
	192	**0.263**	0.320	0.300	0.343	0.384	0.362	0.513	0.567	0.387	**0.251**	0.835	0.543	0.848	0.642
	336	**0.313**	**0.356**	0.332	0.358	0.456	0.409	0.608	0.673	0.687	0.395	0.862	0.560	0.872	0.650
	720	0.376	0.397	**0.362**	**0.365**	0.531	0.474	0.708	0.784	0.708	0.413	0.871	0.566	0.866	0.651
Traffic	96	**0.742**	**0.433**	0.866	0.470	1.133	0.672	1.195	0.802	1.254	0.803	1.495	1.091	2.437	1.126
	192	**0.837**	**0.426**	0.868	0.480	1.279	0.737	1.180	0.802	1.266	0.836	1.591	1.172	2.570	1.148
	336	**0.705**	**0.422**	0.931	0.500	1.432	0.807	1.177	0.804	1.393	0.944	1.474	1.045	2.516	1.150
	720	**0.743**	**0.438**	0.952	0.510	1.542	0.865	1.214	0.810	1.544	1.082	1.273	0.852	2.593	1.153
Weather	96	**0.164**	**0.199**	0.222	0.278	0.182	0.233	0.537	0.374	0.601	0.376	0.460	0.291	0.800	0.562
	192	**0.200**	**0.238**	0.249	0.283	0.258	0.296	0.544	0.374	0.851	0.482	0.505	0.324	0.864	0.582
	336	**0.251**	**0.281**	0.280	0.293	0.317	0.341	0.560	0.369	1.003	0.649	0.543	0.345	0.892	0.590
	720	0.323	0.328	**0.315**	**0.298**	0.454	0.426	0.564	0.351	1.218	0.784	0.577	0.366	0.886	0.589
Electricity	96	0.247	**0.334**	**0.245**	0.335	0.602	0.599	0.928	0.774	1.034	0.824	1.195	0.985	1.858	1.067
	192	**0.241**	**0.333**	0.254	0.343	0.746	0.674	0.934	0.780	1.097	0.874	1.273	1.046	1.915	1.081
	336	**0.257**	**0.348**	0.277	0.358	0.903	0.752	0.952	0.797	1.479	1.038	1.168	0.951	1.967	1.093
	720	**0.293**	0.372	0.327	**0.365**	1.072	0.835	0.994	0.829	1.635	1.202	1.065	0.864	1.938	1.088

Table 10: Computational results for Scale = 1.0 (Landscape). The best results are **bolded**, the second best are underlined.

Dataset	H	ViTime		ViTime-TFM		TimesFM		Moment		Moriai		VisionTS		PatchTST-ZS	
		ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE
ETTh1	96	0.521	0.429	0.421	0.390	**0.379**	0.390	0.833	0.576	0.620	0.402	0.594	**0.383**	1.191	0.814
	192	0.541	0.446	0.437	**0.400**	**0.429**	0.416	0.835	0.580	0.652	0.429	0.626	0.410	1.254	0.837
	336	0.537	0.449	**0.446**	**0.420**	0.453	0.434	0.825	0.581	0.675	0.450	0.638	0.423	1.249	0.836
	720	0.579	0.474	**0.473**	**0.430**	0.506	0.479	0.832	0.605	0.686	0.473	0.637	0.441	1.254	0.837
ETTh2	96	**0.210**	**0.293**	0.281	0.335	0.285	0.330	0.537	0.587	0.526	0.327	0.521	0.328	0.870	0.697
	192	**0.258**	**0.326**	0.300	0.343	0.329	0.366	0.563	0.615	0.583	0.374	0.573	0.367	0.916	0.716
	336	**0.300**	**0.356**	0.316	0.358	0.374	0.403	0.595	0.655	0.609	0.401	0.587	0.381	0.917	0.715
	720	0.367	0.401	**0.355**	**0.365**	0.438	0.456	0.635	0.688	0.627	0.426	0.623	0.422	0.909	0.711
ETTm1	96	**0.308**	0.342	0.345	**0.340**	0.345	0.369	0.931	0.644	0.579	0.360	0.584	0.347	1.299	0.804
	192	0.408	0.395	**0.377**	**0.355**	0.405	0.406	0.911	0.635	0.605	0.379	0.600	0.360	1.375	0.831
	336	**0.420**	0.407	0.433	0.385	0.447	0.431	0.913	0.639	0.625	0.394	0.614	**0.374**	1.385	0.836
	720	0.500	0.448	**0.484**	**0.400**	0.501	0.470	0.921	0.643	0.659	0.419	0.645	0.405	1.365	0.830
ETTm2	96	**0.115**	**0.206**	0.210	0.285	0.197	0.270	0.412	0.457	0.442	0.269	0.477	0.282	0.775	0.596
	192	**0.158**	**0.244**	0.243	0.293	0.300	0.327	0.467	0.513	0.497	0.303	0.512	0.305	0.867	0.633
	336	**0.201**	**0.278**	0.284	0.308	0.345	0.358	0.577	0.638	0.540	0.333	0.541	0.328	0.865	0.632
	720	**0.282**	0.333	0.324	**0.315**	0.470	0.433	0.602	0.666	0.596	0.377	0.586	0.370	0.849	0.626
Traffic	96	0.766	0.392	0.579	0.270	**0.324**	**0.225**	1.125	0.777	1.006	0.469	0.630	0.298	1.786	0.922
	192	0.719	0.390	0.606	0.280	**0.338**	**0.233**	1.165	0.785	0.779	0.334	0.641	0.256	1.911	0.955
	336	0.670	0.375	0.653	0.300	**0.386**	**0.246**	1.174	0.789	1.619	1.079	0.678	0.284	1.863	0.950
	720	0.765	0.387	0.677	0.310	**0.428**	**0.276**	1.224	0.801	1.695	1.183	0.711	0.299	1.933	0.953
Weather	96	**0.120**	**0.132**	0.252	0.290	0.121	0.166	0.569	0.417	0.409	0.203	0.469	0.257	0.840	0.566
	192	**0.136**	**0.171**	0.266	0.295	0.162	0.207	0.582	0.425	0.457	0.241	0.494	0.275	0.926	0.596
	336	**0.183**	**0.212**	0.299	0.305	0.247	0.275	0.602	0.438	0.506	0.276	0.529	0.299	0.914	0.588
	720	**0.254**	**0.270**	0.329	0.310	0.386	0.371	0.608	0.435	0.570	0.323	0.574	0.337	0.949	0.600
Electricity	96	0.181	0.266	0.187	0.265	**0.109**	**0.209**	0.904	0.753	0.397	0.248	0.421	0.266	1.223	0.858
	192	0.188	0.274	0.199	0.273	**0.128**	**0.228**	0.909	0.756	0.417	0.263	0.434	0.277	1.337	0.894
	336	0.198	0.282	0.220	0.288	**0.157**	**0.252**	0.917	0.764	0.437	0.278	0.455	0.296	1.334	0.892
	720	0.216	0.296	0.256	0.295	**0.209**	**0.292**	0.930	0.778	0.479	0.307	0.506	0.337	1.349	0.896

Table 11: Computational results for Scale = 1.5 (Landscape). The best results are **bolded**, the second best are underlined.

Dataset	H	ViTime		ViTime-TFM		TimesFM		Moment		Moriai		VisionTS		PatchTST-ZS	
		ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE
ETTh1	96	0.450	**0.413**	0.502	0.470	**0.412**	0.416	0.886	0.592	0.736	0.469	0.929	0.598	1.270	0.828
	192	0.494	**0.435**	0.536	0.480	**0.451**	0.446	0.909	0.607	0.801	0.503	0.959	0.632	1.362	0.864
	336	**0.503**	**0.444**	0.576	0.500	0.521	0.485	0.913	0.609	1.061	0.713	0.896	0.608	1.374	0.866
	720	**0.535**	**0.463**	0.587	0.510	0.614	0.534	0.867	0.584	1.124	0.766	0.863	0.574	1.387	0.871
ETTh2	96	**0.220**	0.297	0.293	0.375	0.238	0.310	0.392	**0.266**	0.495	0.318	0.442	0.288	0.960	0.701
	192	**0.269**	0.334	0.319	0.383	0.291	0.339	0.366	**0.252**	0.523	0.344	0.424	0.285	1.137	0.770
	336	**0.290**	0.350	0.340	0.398	0.335	0.374	0.345	**0.240**	0.716	0.454	0.389	0.266	1.138	0.770
	720	0.329	0.379	0.367	0.405	0.413	0.434	**0.336**	**0.235**	0.820	0.521	0.361	0.243	1.177	0.785
ETTm1	96	**0.351**	**0.360**	0.504	0.490	0.604	0.487	0.944	0.648	1.021	0.674	1.342	0.902	1.146	0.742
	192	**0.430**	**0.398**	0.552	0.505	0.622	0.496	0.930	0.638	1.052	0.692	1.069	0.726	1.257	0.775
	336	**0.454**	**0.418**	0.607	0.535	0.790	0.552	0.941	0.642	1.177	0.764	1.034	0.714	1.256	0.780
	720	**0.524**	**0.451**	0.680	0.550	0.927	0.604	0.917	0.630	1.226	0.812	1.030	0.702	1.270	0.782
ETTm2	96	**0.139**	**0.225**	0.239	0.355	0.168	0.260	0.396	0.435	0.396	0.268	1.097	0.713	0.781	0.587
	192	**0.180**	**0.266**	0.283	0.363	0.223	0.299	0.493	0.543	0.493	0.313	0.912	0.593	0.808	0.598
	336	**0.229**	**0.298**	0.328	0.378	0.319	0.359	0.620	0.684	0.620	0.372	0.912	0.593	0.777	0.588
	720	**0.301**	**0.346**	0.375	0.385	0.392	0.409	0.652	0.721	0.652	0.416	0.935	0.608	0.808	0.600
Traffic	96	**0.695**	**0.384**	1.097	0.540	0.941	0.523	1.111	0.768	1.071	0.608	1.356	1.013	2.056	0.998
	192	**0.630**	**0.371**	1.126	0.550	0.996	0.557	1.060	0.754	0.939	0.524	1.476	1.081	2.186	1.033
	336	**0.693**	**0.371**	1.173	0.570	0.996	0.580	1.103	0.764	1.385	0.922	1.444	1.027	2.146	1.041
	720	**0.746**	**0.418**	1.208	0.580	1.249	0.678	1.114	0.771	1.474	1.037	1.206	0.859	2.217	1.042
Weather	96	**0.120**	**0.141**	0.228	0.278	0.176	0.216	0.494	0.341	0.561	0.372	0.420	0.249	0.837	0.549
	192	**0.156**	**0.189**	0.244	0.283	0.237	0.273	0.517	0.364	0.650	0.440	0.472	0.306	0.918	0.564
	336	**0.200**	**0.231**	0.275	0.293	0.297	0.328	0.568	0.406	0.807	0.508	0.526	0.359	0.941	0.574
	720	**0.267**	**0.286**	0.308	0.298	0.408	0.406	0.668	0.494	0.973	0.590	0.637	0.460	0.931	0.573
Electricity	96	**0.171**	**0.262**	0.251	0.365	0.227	0.306	0.796	0.662	0.687	0.508	1.071	0.876	1.294	0.889
	192	**0.172**	**0.266**	0.264	0.373	0.254	0.333	0.799	0.667	0.688	0.512	1.148	0.937	1.412	0.924
	336	**0.186**	**0.275**	0.290	0.388	0.283	0.358	0.816	0.682	1.058	0.803	1.114	0.916	1.396	0.918
	720	**0.223**	**0.302**	0.333	0.395	0.393	0.438	0.883	0.733	1.136	0.871	0.991	0.813	1.425	0.928

Table 12: Computational results for Scale = 2.0 (Landscape). The best results are **bolded**, the second best are underlined.

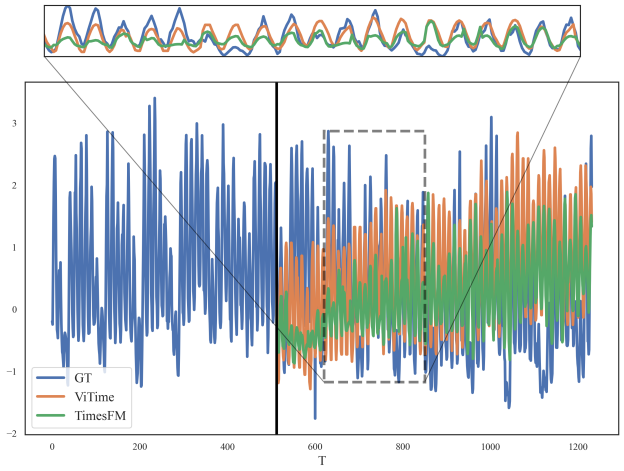
Dataset	H	ViTime		ViTime-TFM		TimesFM		Moment		Moriai		VisionTS		PatchTST-ZS	
		ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE
ETTh1	96	0.521	0.429	0.526	0.480	**0.358**	**0.384**	0.791	0.543	0.674	0.444	0.834	0.537	1.279	0.818
	192	0.541	0.446	0.560	0.490	**0.406**	**0.415**	0.820	0.557	0.855	0.564	0.843	0.557	1.366	0.844
	336	0.537	0.449	0.595	0.510	**0.454**	**0.444**	0.855	0.576	1.427	0.928	0.823	0.558	1.420	0.856
	720	**0.579**	**0.474**	0.624	0.520	0.581	0.508	0.909	0.605	1.562	1.052	0.891	0.575	1.419	0.860
ETTh2	96	**0.210**	**0.293**	0.292	0.375	0.219	0.295	0.450	0.307	0.602	0.387	0.500	0.329	0.701	0.601
	192	**0.258**	0.326	0.330	0.383	0.283	0.336	0.447	**0.307**	0.629	0.426	0.506	0.347	0.786	0.632
	336	**0.300**	0.356	0.360	0.398	0.316	0.363	0.433	**0.298**	0.855	0.561	0.471	0.327	0.789	0.632
	720	**0.367**	0.401	0.379	0.405	0.415	0.432	0.385	**0.269**	0.983	0.693	0.411	0.283	0.789	0.629
ETTm1	96	**0.308**	**0.342**	0.554	0.540	1.029	0.569	0.955	0.662	1.311	0.822	1.286	0.837	1.037	0.690
	192	**0.408**	**0.395**	0.582	0.555	1.068	0.605	0.938	0.650	1.250	0.800	0.987	0.647	1.139	0.719
	336	**0.420**	**0.407**	0.651	0.585	1.245	0.655	0.946	0.650	1.539	0.981	1.012	0.660	1.173	0.729
	720	**0.500**	**0.448**	0.732	0.600	1.329	0.700	0.947	0.641	1.651	1.086	1.007	0.634	1.219	0.743
ETTm2	96	**0.115**	**0.206**	0.236	0.355	0.178	0.268	0.489	0.540	0.489	0.333	1.052	0.684	0.686	0.555
	192	**0.158**	**0.244**	0.271	0.363	0.223	0.304	0.502	0.553	0.582	0.391	0.842	0.547	0.775	0.582
	336	**0.201**	**0.278**	0.318	0.378	0.285	0.346	0.565	0.623	0.763	0.516	0.894	0.581	0.730	0.567
	720	**0.282**	**0.333**	0.362	0.385	0.408	0.417	0.649	0.718	0.835	0.567	0.915	0.595	0.784	0.586
Traffic	96	**0.766**	**0.392**	1.125	0.650	0.943	0.492	0.990	0.522	0.990	0.522	1.420	1.010	1.966	0.977
	192	**0.719**	**0.390**	1.151	0.660	0.933	0.547	0.911	0.497	0.911	0.497	1.570	1.128	2.157	1.035
	336	**0.670**	**0.375**	1.183	0.680	0.957	0.561	1.071	0.751	1.823	1.206	1.317	0.892	2.236	1.045
	720	**0.765**	**0.387**	1.228	0.690	1.145	0.656	1.092	0.758	1.939	1.352	1.270	0.837	2.256	1.050
Weather	96	**0.120**	**0.133**	0.223	0.278	0.167	0.205	0.413	0.282	0.600	0.371	0.326	0.194	0.836	0.534
	192	**0.136**	**0.171**	0.242	0.283	0.231	0.276	0.439	0.301	0.539	0.330	0.417	0.263	0.948	0.578
	336	**0.183**	**0.212**	0.270	0.293	0.291	0.331	0.463	0.314	0.799	0.427	0.468	0.305	0.992	0.588
	720	**0.254**	**0.270**	0.303	0.298	0.413	0.419	0.540	0.373	0.885	0.529	0.553	0.375	0.974	0.582
Electricity	96	**0.181**	**0.266**	0.253	0.365	0.223	0.299	0.798	0.654	0.565	0.376	1.034	0.863	1.177	0.848
	192	**0.188**	**0.274**	0.266	0.373	0.263	0.334	0.796	0.651	0.592	0.416	1.167	0.956	1.270	0.880
	336	**0.198**	**0.282**	0.281	0.388	0.285	0.364	0.798	0.654	1.275	0.950	0.977	0.764	1.318	0.896
	720	**0.216**	**0.296**	0.327	0.395	0.399	0.447	0.805	0.666	1.362	1.037	0.890	0.715	1.315	0.895

Table 13: MAE of different methods in fine-tuning study.

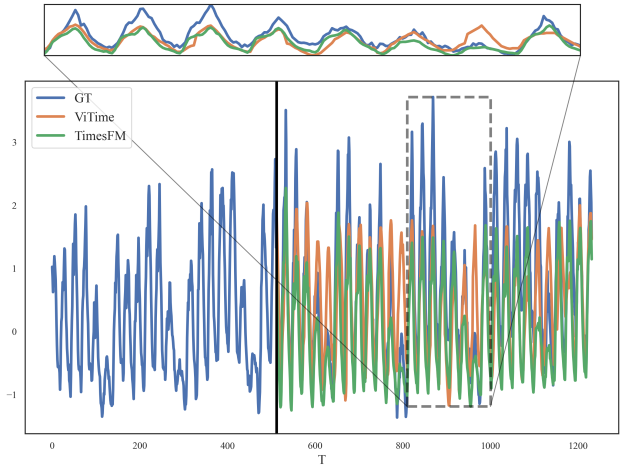
Dataset	H	VITIME (FT)		TimesFM (FT)	GPT4TS (FT)	TIME-LLM (FT)	PatchTST	SiMBA	TIMESNET	PatchTST	iTransformer	TimeMixer
		10%	100%	10%	10%	10%	10%	100%	100%	100%	100%	100%
ETTh1	96	0.397	**0.383**	0.398	0.485	0.460	0.485	0.395	0.402	0.400	0.405	0.390
	192	0.414	**0.401**	0.424	0.524	0.483	0.524	0.424	0.429	0.429	0.436	0.414
	336	0.427	**0.410**	0.436	0.550	0.540	0.550	0.443	0.469	0.440	0.458	0.429
	720	0.460	**0.438**	0.445	0.610	0.604	0.610	0.469	0.500	0.468	0.491	0.460
	Average	0.424	**0.408**	0.426	0.542	0.522	0.542	0.433	0.450	0.434	0.448	0.423
ETTh2	96	0.324	**0.302**	0.356	0.389	0.326	0.389	0.339	0.374	0.337	0.349	0.330
	192	0.354	**0.331**	0.400	0.414	0.373	0.414	0.390	0.414	0.382	0.400	0.402
	336	0.377	**0.352**	0.428	0.441	0.429	0.441	0.406	0.452	0.384	0.432	0.396
	720	0.431	0.411	0.457	0.480	0.449	0.480	0.431	0.468	0.422	0.445	**0.408**
	Average	0.371	**0.349**	0.410	0.431	0.394	0.431	0.392	0.427	0.381	0.406	0.384
ETTm2	96	0.260	**0.237**	0.263	0.274	0.261	0.274	0.263	0.267	0.256	0.264	0.254
	192	0.293	**0.278**	0.309	0.317	0.314	0.317	0.306	0.309	0.296	0.309	0.295
	336	0.325	**0.313**	0.349	0.353	0.327	0.353	0.343	0.351	0.329	0.348	0.330
	720	0.382	**0.371**	0.415	0.427	0.390	0.427	0.399	0.403	0.385	0.407	0.383
	Average	0.315	**0.300**	0.334	0.343	0.323	0.343	0.328	0.333	0.317	0.332	0.316
ETTm1	96	0.341	**0.333**	0.345	0.419	0.388	0.419	0.360	0.375	0.346	0.368	0.340
	192	0.364	**0.353**	0.374	0.434	0.416	0.434	0.382	0.387	0.370	0.391	0.365
	336	0.386	**0.373**	0.397	0.454	0.426	0.454	0.405	0.411	0.392	0.420	0.381
	720	0.420	**0.410**	0.436	0.556	0.476	0.556	0.437	0.450	0.420	0.459	0.417
	Average	0.378	**0.367**	0.388	0.466	0.426	0.466	0.396	0.406	0.382	0.409	0.376
Traffic	96	0.247	**0.237**		Not Reported		0.268	0.268	0.321	0.249	0.268	0.249
	192	0.247	**0.239**		Not Reported		0.274	0.317	0.336	0.256	0.276	0.250
	336	0.251	**0.246**		Not Reported		0.282	0.284	0.336	0.264	0.283	0.270
	720	**0.271**	0.279		Not Reported		0.319	0.297	0.350	0.286	0.302	0.281
	Average	0.254	**0.250**		Not Reported		0.286	0.291	0.336	0.264	0.282	0.263
Weather	96	0.188	**0.186**		Not Reported		0.221	0.219	0.220	0.198	0.214	0.197
	192	0.230	**0.228**		Not Reported		0.261	0.260	0.261	0.241	0.254	0.239
	336	0.271	**0.270**		Not Reported		0.300	0.297	0.306	0.282	0.296	0.280
	720	**0.317**	0.322		Not Reported		0.351	0.349	0.359	0.334	0.347	0.330
	Average	**0.252**	**0.252**		Not Reported		0.283	0.281	0.286	0.264	0.278	0.262
Electricity	96	0.231	0.226		Not Reported		0.235	0.253	0.272	**0.222**	0.240	0.224
	192	0.239	0.235		Not Reported		0.250	0.262	0.289	0.240	0.253	**0.220**
	336	0.256	**0.249**		Not Reported		0.270	0.277	0.300	0.259	0.269	0.255
	720	0.289	**0.279**		Not Reported		0.315	0.305	0.320	0.290	0.317	0.287
	Average	0.254	0.247		Not Reported		0.268	0.274	0.295	0.253	0.270	**0.246**

Table 14: Performance comparison of TimesFM and ViTime across different robust inference scenarios

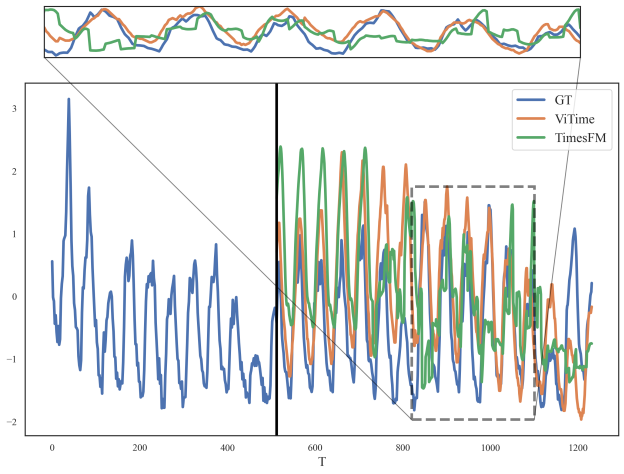
Dataset	H	TimesFM GN(0.1)		ViTime GN(0.1)		TimesFM GN(0.3)		ViTime GN(0.3)		ViTime DM(0.3)	
		ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE	ReMSE	ReMAE
ETTh1	96	0.410	0.414	0.431	0.416	0.427	0.423	0.444	0.436	0.433	0.414
	192	0.460	0.445	0.460	0.434	0.473	0.452	0.469	0.452	0.463	0.433
	336	0.513	0.478	0.493	0.458	0.525	0.485	0.501	0.474	0.496	0.456
	720	0.617	0.545	0.572	0.509	0.632	0.553	0.585	0.525	0.574	0.507
	Average	0.500	0.471	**0.489**	**0.454**	0.514	0.478	**0.500**	**0.472**	**0.492**	**0.453**
ETTh2	96	0.269	0.326	0.256	0.321	0.265	0.329	0.268	0.335	0.255	0.319
	192	0.326	0.367	0.301	0.356	0.314	0.366	0.308	0.367	0.297	0.352
	336	0.378	0.406	0.347	0.391	0.366	0.404	0.354	0.401	0.343	0.387
	720	0.480	0.475	0.449	0.460	0.464	0.470	0.450	0.465	0.441	0.455
	Average	0.363	0.394	**0.338**	**0.382**	0.352	0.392	**0.345**	**0.391**	**0.334**	**0.378**
ETTm1	96	0.521	0.444	0.428	0.404	0.532	0.442	0.436	0.425	0.408	0.392
	192	0.572	0.476	0.468	0.425	0.575	0.469	0.467	0.440	0.451	0.413
	336	0.642	0.508	0.517	0.450	0.640	0.501	0.509	0.462	0.507	0.440
	720	0.719	0.551	0.594	0.490	0.688	0.538	0.587	0.501	0.595	0.484
	Average	0.614	0.495	**0.502**	**0.442**	0.609	0.488	**0.500**	**0.457**	**0.490**	**0.432**
ETTm2	96	0.212	0.282	0.200	0.277	0.200	0.280	0.209	0.290	0.199	0.274
	192	0.291	0.329	0.260	0.319	0.262	0.321	0.266	0.327	0.262	0.317
	336	0.358	0.370	0.322	0.358	0.323	0.360	0.320	0.360	0.320	0.354
	720	0.459	0.430	0.403	0.407	0.420	0.418	0.392	0.406	0.398	0.403
	Average	0.330	0.353	**0.296**	**0.340**	0.301	0.345	**0.297**	**0.344**	**0.295**	**0.337**
Electricity	96	0.266	0.338	0.225	0.317	0.285	0.363	0.252	0.354	0.222	0.311
	192	0.320	0.375	0.240	0.328	0.333	0.398	0.262	0.361	0.235	0.323
	336	0.377	0.413	0.263	0.345	0.390	0.440	0.285	0.377	0.260	0.341
	720	0.488	0.486	0.348	0.402	0.527	0.532	0.368	0.432	0.343	0.397
	Average	0.363	0.403	**0.269**	**0.348**	0.384	0.433	**0.292**	**0.381**	**0.265**	**0.343**
Traffic	96	0.770	0.457	0.724	0.398	0.765	0.478	0.778	0.463	0.782	0.406
	192	0.830	0.489	0.723	0.395	0.815	0.505	0.776	0.461	0.772	0.400
	336	0.850	0.516	0.743	0.404	0.873	0.534	0.799	0.470	0.792	0.409
	720	0.979	0.582	0.844	0.450	0.988	0.600	0.896	0.514	0.886	0.454
	Average	0.857	0.511	**0.759**	**0.412**	0.860	0.529	**0.812**	**0.477**	**0.808**	**0.417**
Weather	96	0.157	0.202	0.178	0.209	0.170	0.222	0.178	0.230	0.170	0.209
	192	0.214	0.253	0.229	0.257	0.228	0.272	0.232	0.273	0.231	0.261
	336	0.272	0.299	0.289	0.300	0.283	0.314	0.284	0.309	0.288	0.302
	720	0.378	0.369	0.372	0.353	0.380	0.376	0.361	0.355	0.366	0.353
	Average	0.255	0.281	**0.267**	**0.280**	0.265	0.296	**0.264**	**0.292**	**0.264**	**0.281**



(a) Rescale factor=0.5

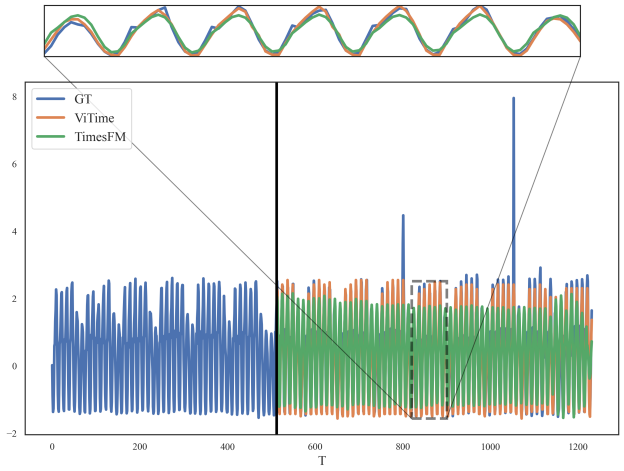


(b) Rescale factor=1

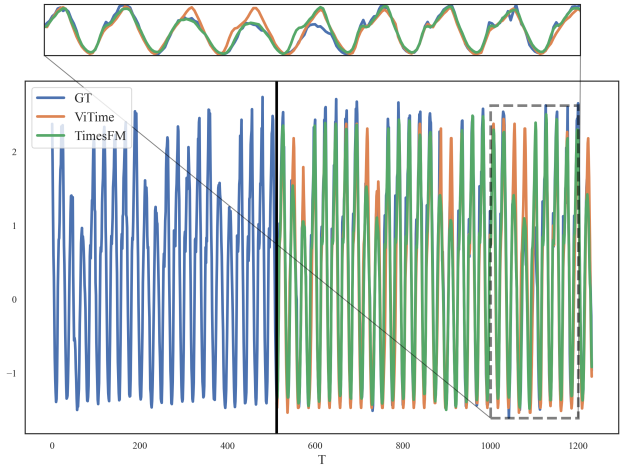


(c) Rescale factor=2

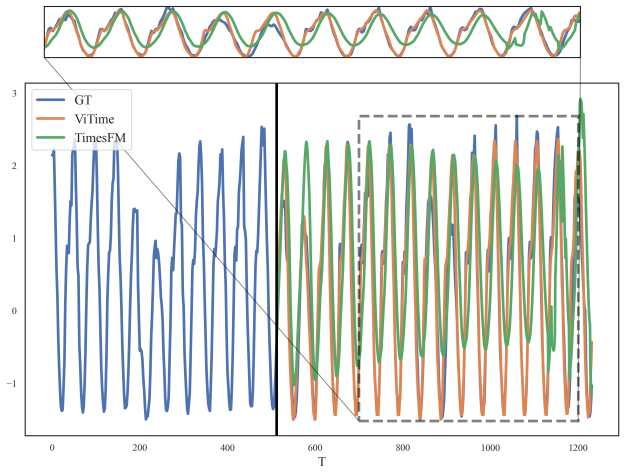
Figure 16: Illustrative example of Electricity dataset.



(a) Rescale factor=0.5

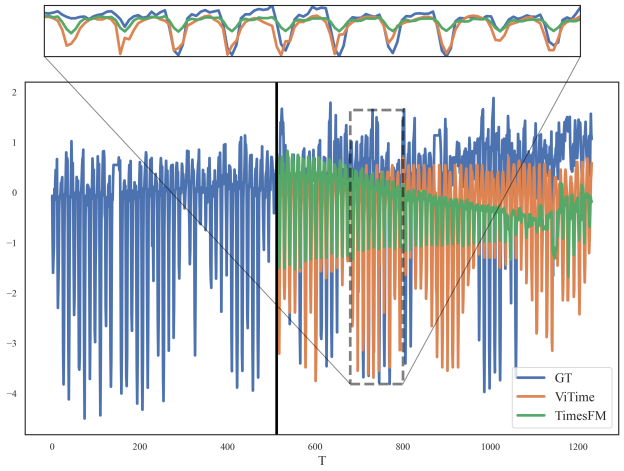


(b) Rescale factor=1

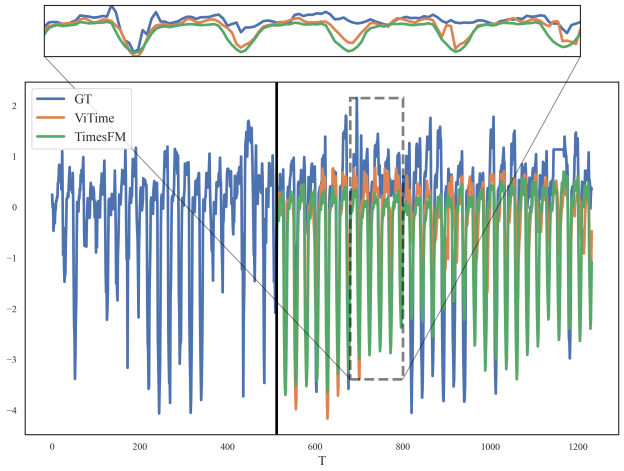


(c) Rescale factor=2

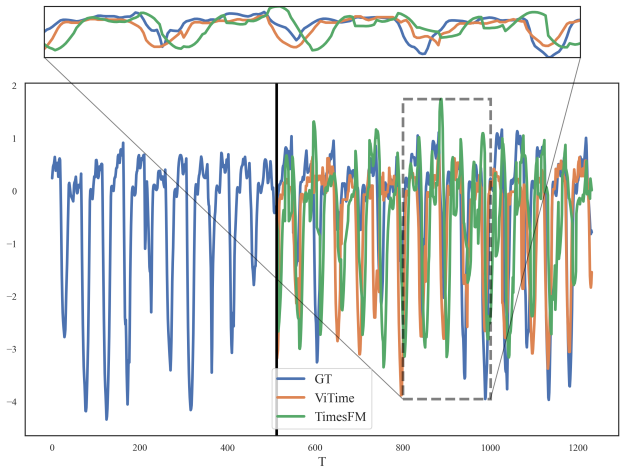
Figure 17: Illustrative example of Traffic dataset.



(a) Rescale factor=0.5

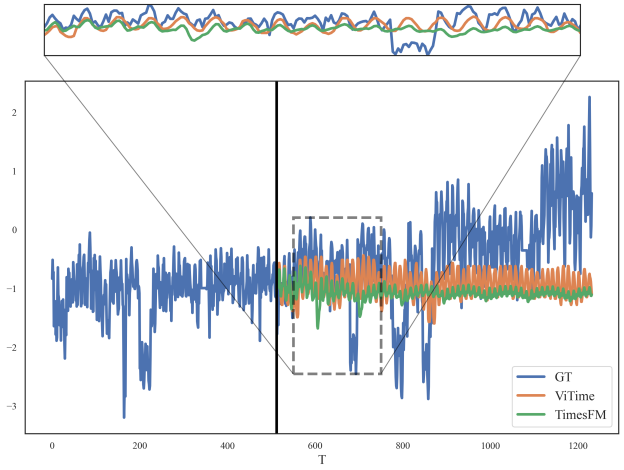


(b) Rescale factor=1

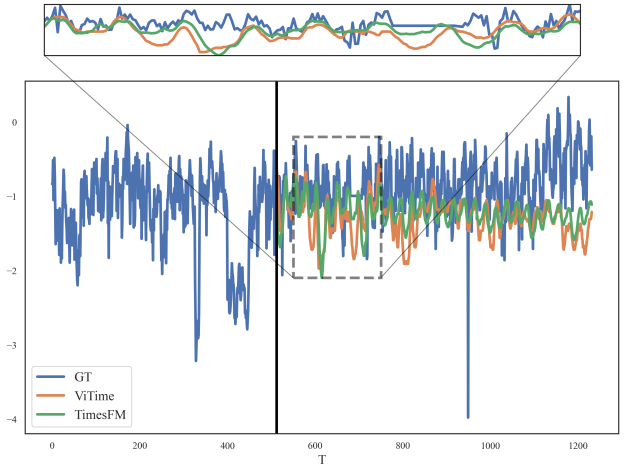


(c) Rescale factor=2

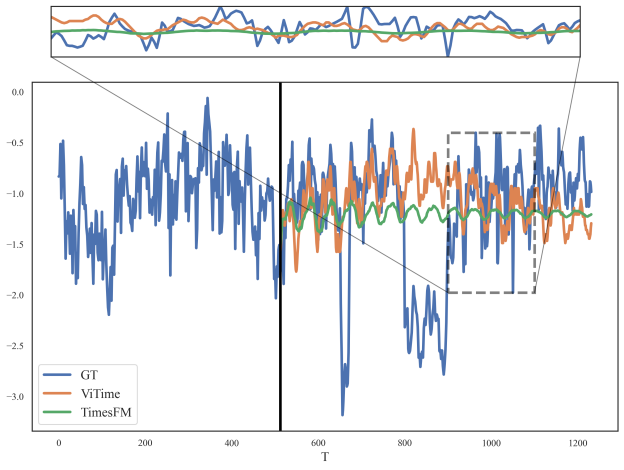
Figure 18: Illustrative example of ETTh1 dataset.



(a) Rescale factor=0.5

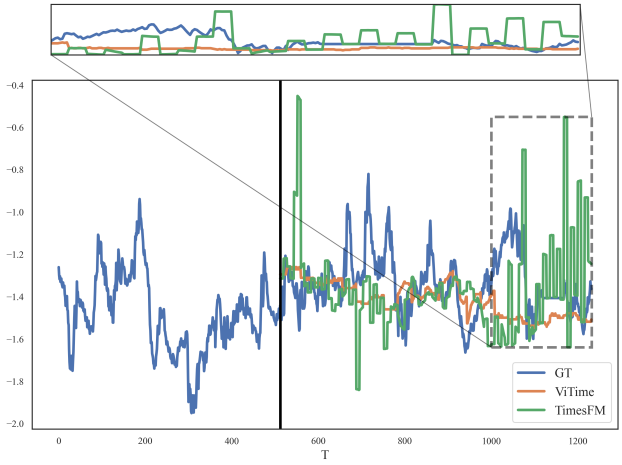


(b) Rescale factor=1

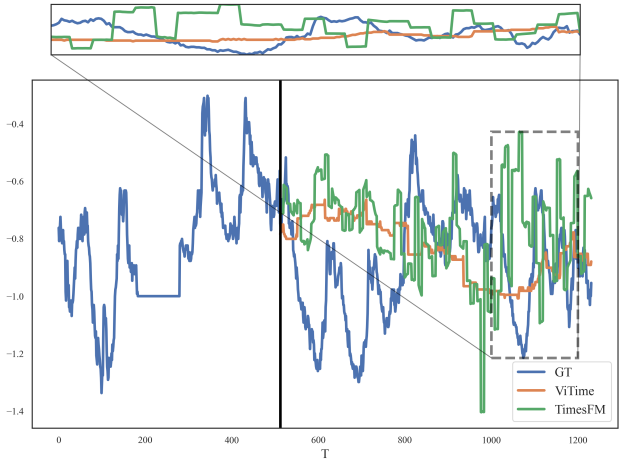


(c) Rescale factor=2

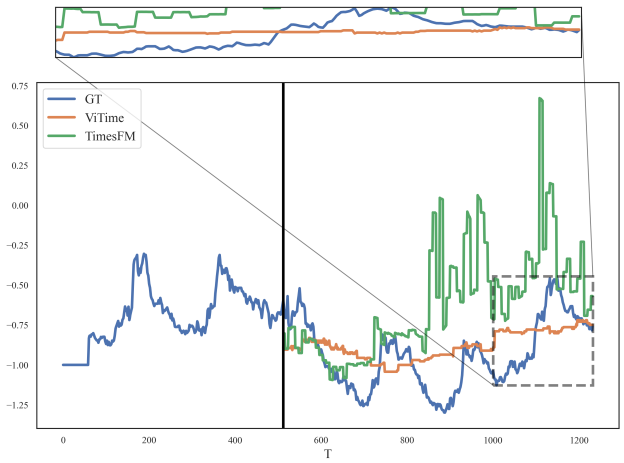
Figure 19: Illustrative example of ETTh2 dataset.



(a) Rescale factor=0.5

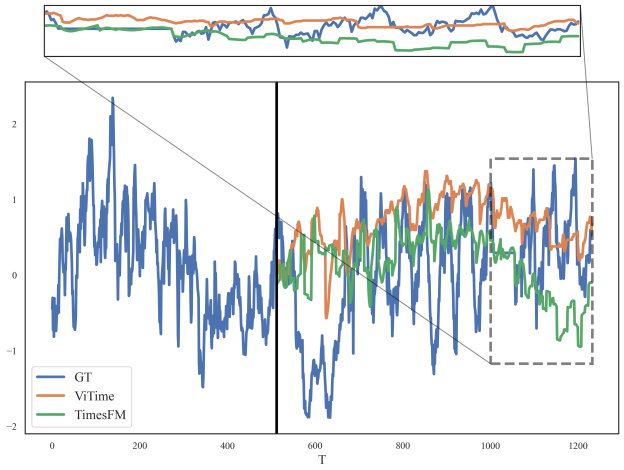


(b) Rescale factor=1

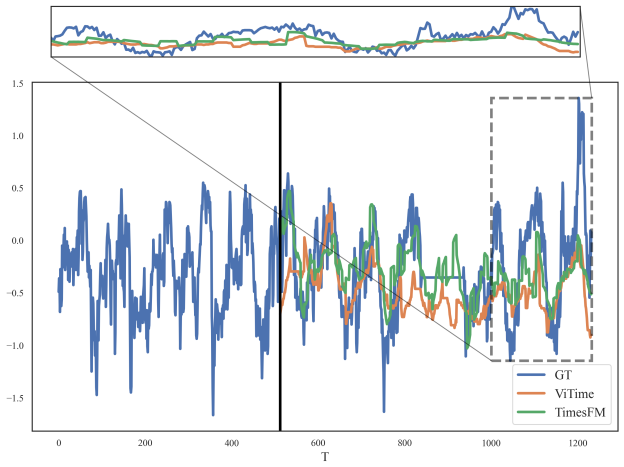


(c) Rescale factor=2

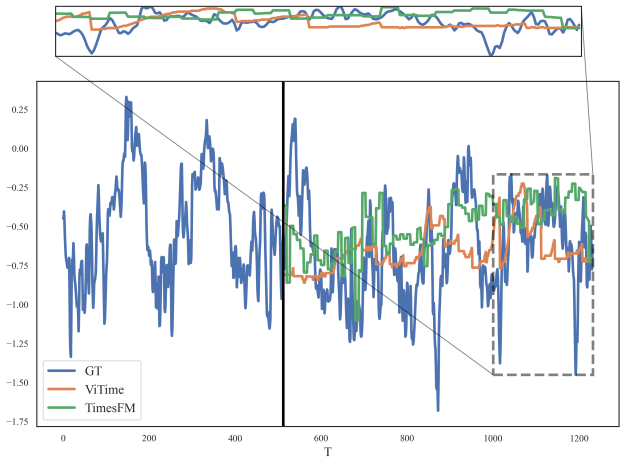
Figure 20: Illustrative example of ETTm1 dataset.



(a) Rescale factor=0.5



(b) Rescale factor=1



(c) Rescale factor=2

Figure 21: Illustrative example of ETTm2 dataset.