Paloma: Phase-Conditioned Residual Modulation for Time Series Forecasting

Jingru Fei[1], **Wei Fan**[2], **Kun Yi**[3], **Qi Zhang**[4], **Zhendong Niu**[1*]
[1]Beijing Institute of Technology, [2]University of Auckland,
[3]State Information Center, [4]Tongji University,
{jingrufei, zniu}@bit.edu.cn, wei.fan@auckland.ac.nz,
kunyi.cn@gmail.com, zhangqi_cs@tongji.edu.cn

Abstract

Explicitly modeling periodicity has proven to be an efficient approach for time series forecasting, where the residual, obtained after removing the periodic component from the time series, is typically treated as unstructured noise. However, in this work, we show that such residuals still exhibit structured, *phase-dependent* deviations, as supported by both empirical observations and theoretical analysis. This suggests that residuals should not be treated merely as unstructured noise, but instead as *phase-conditioned signals* whose variations arise in both direction and magnitude. Motivated by this perspective and inspired by phase and amplitude modulation in classical signal processing, we propose *Paloma*, a simple yet effective framework for modeling residual deviations. Specifically, Paloma comprises two complementary modules: *a Phase Rotation module*, which performs phase-conditioned rotations to capture directional changes in the feature space, and *an Amplitude Scaling module*, which applies phase-conditioned affine transformations to model magnitude variations. We further abstract the two modules into a general-purpose plugin, termed the *Paloma technique*, which can be readily integrated into existing forecasting architectures to modulate input sequences in a phase-aware manner, enabling backbone models to capture phase-dependent variations in both direction and magnitude. Extensive experiments on twelve datasets demonstrate that Paloma achieves state-of-the-art and robust forecasting performance. Additional results show that the Paloma technique consistently improves a wide range of baseline models, validating its effectiveness as a general-purpose component. Our Code is available at `https://github.com/TSTS13/Paloma`.

1 Introduction

Time series forecasting plays a fundamental role in many real-world applications, including energy systems [1, 2, 3], urban transportation planning [4, 5], and financial markets [6, 7]. In these scenarios, accurately capturing the temporal dynamics is critical for effective decision-making, resource allocation, and system reliability. Recent advances in deep learning have developed various forecasting architectures for modeling temporal dynamics, including Transformer-based [8, 9, 10], MLP-based [11, 12], CNN-based [13, 14], and RNN-based [15] models. Built upon different modeling principles, these methods have substantially enriched the methodological landscape of time series forecasting.

Among the diverse temporal structures in real-world time series, periodicity is one of the most prominent and informative patterns [16, 17]. Explicitly modeling periodicity has therefore attracted increasing attention [18, 19, 20], as it provides a principled and parameter-efficient way to capture

*Corresponding author

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

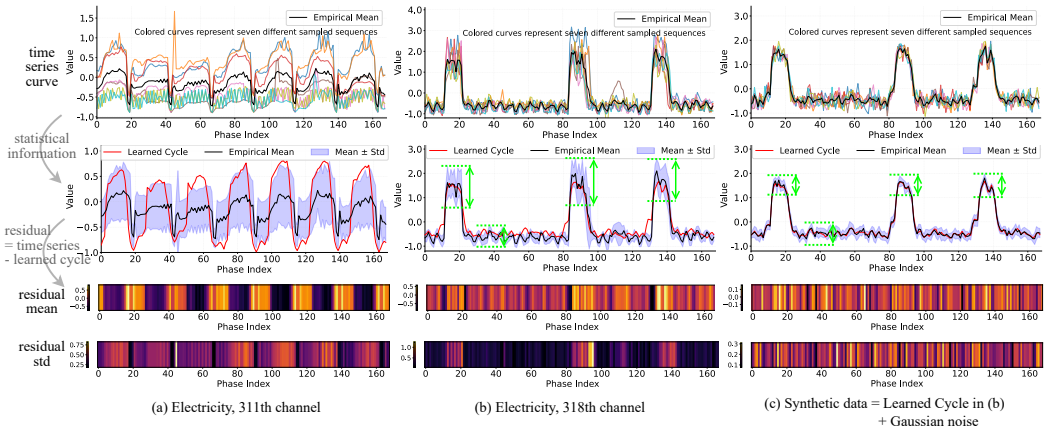


Figure 1: Visualization of structured residual deviations. (a) and (b) present two representative channels from the Electricity dataset. From top to bottom, the rows show sampled sequences, learned cycles from CycleNet, and the residual mean and standard deviation visualized as one-dimensional heatmaps under a cycle length of 168. (c) is constructed by combining the learned cycle from (b) with Gaussian noise, with the row-wise definitions consistent with those in (a) and (b). Additional experiments for (c) and visualizations on the Weather dataset are provided in Appendix A.

recurring temporal structures while achieving competitive performance with substantially lower computational cost than Transformer-based models. For example, CycleNet [18] represents periodic components as learnable recurrent cycles, decomposing a time series into a phase-aligned periodic component and a residual term. The residual is then modeled by a simple predictor, such as a single-layer linear model or a shallow MLP. This design implicitly assumes that, once the periodic component is removed, the remaining residual is largely unstructured noise and can be adequately captured by a lightweight predictor. *However, it remains unclear whether such residuals are truly unstructured, or whether they still contain systematic, phase-dependent patterns*.

To investigate this question, we visualize the learned cycles produced by CycleNet [18] and the corresponding residual terms on the real-world Electricity dataset, as shown in Figure 1(a) and (b), which present two representative yet distinct cases. In Figure 1(a), the first row shows multiple cycle instances with significantly varying peak magnitudes, leading to inconsistent periodic patterns and hindering the learning of a unified periodic structure; consequently, the learned cycle in the second row (red curve) deviates substantially from the empirical mean (black curve). This clearly indicates that the residual term is not unstructured noise, but instead exhibits structured deviations, as evidenced by the heatmap visualizations of its mean and standard deviation, which reveal clear phase-dependent patterns. In contrast to Figure 1(a), Figure 1(b) presents a markedly different scenario. The multiple cycle instances in the first row follow a consistent periodic pattern, and the learned cycle in the second row closely matches the empirical mean, indicating that the periodic component is well captured. Nevertheless, further analysis still reveals persistent phase-dependent structures in the residual, evident in both its mean and standard deviation, particularly around phase indices near 20, 90, and 140, with more pronounced effects observed in the standard deviation.

We further examine this phenomenon from a counterfactual perspective by assuming that the residual term is purely random noise, and analyzing the resulting time series and its statistical properties. To this end, we conduct a controlled experiment, as illustrated in Figure 1(c), where synthetic sequences are constructed by combining the learned cycle from Figure 1(b) with Gaussian noise as the residual component. A clear contrast can be observed between Figure 1(b) and (c) at the locations highlighted by the green arrows: in (c), the standard deviation remains nearly invariant across phases, whereas in (b), it consistently increases around peak regions. In summary, Figure 1(a) and (b) show that *the residual consistently exhibits phase-dependent deviations even after the periodic component is effectively removed*, while the comparison between Figure 1(b) and (c) further suggests that *the residual cannot be regarded as purely random noise*. These observations are theoretically supported in Section 3.2, where we show that such structured deviations inherently persist in the residual.

To this end, we interpret the residual as a phase-conditioned signal with structured variations across phases. In classical signal processing [21, 22, 23, 24], phase refers to the position of a periodic signal within its cycle at a specific moment and is usually represented by an angle, where different

phase angles correspond to different directional states, while amplitude controls the strength of signal variations. Motivated by this perspective, we propose *Paloma*, a simple yet effective framework that models residual deviations via phase–amplitude modulation through two complementary modules: (i) a *Phase Rotation module* that applies phase-conditioned rotations to model directional transformations in the feature space, and (ii) an *Amplitude Scaling module* that performs phase-conditioned affine transformations to capture variation intensity across phases. Moreover, the two modules can be combined into a general-purpose plugin, termed the *Paloma technique*. Beyond modeling residuals, Paloma technique can be readily integrated into existing forecasting architectures and directly applied to input sequences, enabling them to capture phase-dependent variations in both direction and magnitude that are otherwise overlooked. Extensive experiments on twelve datasets demonstrate that Paloma achieves state-of-the-art forecasting performance, while additional plug-in experiments further verify the effectiveness and generality of the *Paloma technique* across diverse forecasting architectures. In summary, our contributions are as follows:

- We identify that the residual, obtained after removing the periodic component in time series, exhibits structured, phase-dependent deviations rather than unstructured noise, and provide theoretical analysis explaining their persistence.
- We propose *Paloma*, a simple framework that models residual deviations via phase–amplitude modulation, featuring a Phase Rotation module and an Amplitude Scaling module, and further abstract the two modules as a general-purpose plugin, termed the *Paloma technique*.
- Extensive experiments demonstrate that Paloma achieves state-of-the-art and robust performance, and validate the effectiveness of Paloma technique across diverse forecasting models.

2 Related Work

Temporal Periodicity Modeling. Temporal periodicity has long been recognized as a key factor in improving forecasting accuracy by capturing recurring patterns in time series data. Representative models, including Autoformer [25], FEDformer [1], and DLinear [11], leverage seasonal–trend decomposition (STD) to disentangle the original time series into seasonal and trend components, followed by independent modeling of each component. Building on these approaches, Leddam [26] replaces the fixed moving-average kernel in STD with a learnable decomposition kernel, providing greater flexibility and yielding improved performance. Recent advances have explored leveraging the intrinsic periodicity of time series as prior knowledge for efficient forecasting. SparseTSF [16] introduces a cross-period sparse forecasting technique to decouple cycles and trends, achieving competitive performance with low computational cost. Along this line of research, CycleNet [18] introduces Residual Cycle Forecasting (RCF), which models periodic patterns using learnable recurrent cycles and predicts the resulting residuals. While effective and efficient, the residual component is typically modeled using a simple predictor, such as a linear layer or a shallow MLP, implicitly assuming that it mainly consists of unstructured noise. However, our analysis suggests that the residual may still contain structured patterns beyond the modeled periodic component. This motivates a more refined modeling of the residual component.

Modulation–Inspired Time Series Modeling. Recent advances in time series modeling have increasingly drawn inspiration from classical signal processing [23, 24, 27, 28]. FilterNet [29] introduces learnable frequency filters to enhance the utilization of full-spectrum information, effectively performing frequency-domain shaping. In contrast, TimeEmb [30] applies frequency filtering specifically to the time-varying component, targeting the modeling of non-stationary temporal dynamics. Subsequently, FreqCycle [19] enhances mid-to-high frequency components via learnable filters and adaptive weighting, improving the modeling of short-term variations. These filtering processes can be viewed as a form of implicit spectral modulation. Unlike these approaches, which implicitly realize modulation through frequency filtering, our method explicitly modulates the signal by separately modeling its phase and amplitude variations. Beyond frequency filtering and spectral modulation, PhaseFormer [31] explores the role of phase information in time series forecasting by replacing conventional patch-based tokenization with phase-based tokens, emphasizing phase as a more compact and coherent representation of temporal dynamics. Different from PhaseFormer, Paloma treats phase as a conditional variable for controlling residual deviations.

3 Preliminaries

3.1 Problem Formulation

Time Series Forecasting. Given a time series input $\mathbf{X} = \{\mathbf{x}_l\}_{l=0}^{T-1} \in \mathbb{R}^{T\times N}$ and its corresponding future sequence $\mathbf{Y} = \{\mathbf{x}_l\}_{l=T}^{T+\tau-1} \in \mathbb{R}^{\tau\times N}$, where N denotes the number of channels; T and τ denote the lookback window length and prediction horizon, respectively. The time series forecasting task is to learn a parametric model $f_\theta(\cdot)$ that maps the historical observations $\mathbf{X}$ to the predicted future sequence $\hat{\mathbf{Y}}$, i.e., $\hat{\mathbf{Y}} = f_\theta(\mathbf{X})$, with θ being the model parameters.

Phase Index Construction. Given a time series input $\mathbf{X} = \{\mathbf{x}_l\}_{l=0}^{T-1} \in \mathbb{R}^{T\times N}$, we define a corresponding phase index sequence $\mathbf{i} = \{i_l\}_{l=0}^{T-1}$, where $i_l \in \{0, 1, \ldots, W-1\}$, and W denotes the cycle length. For a sequence starting at absolute time t_0, the phase index at relative step l is defined as $i_l = [(t_0 \bmod W) + l] \bmod W, \quad l = 0, 1, \ldots, T-1$. This formulation assigns each time step to a specific phase within the cycle, enabling alignment of temporal observations across different periods and facilitating phase-conditioned modeling of temporal dynamics.

3.2 Understanding Phase-Dependent Deviations in Residuals

Given a time series input $\mathbf{X} \in \mathbb{R}^{T\times N}$ with phase index sequence $\mathbf{i} = \{i_l\}_{l=0}^{T-1}$, a common strategy for explicitly modeling periodic components, adopted by methods such as CycleNet [18], is to employ a learnable prototype matrix $\mathbf{Q} \in \mathbb{R}^{W\times N}$, where the k-th row $\mathbf{Q}_k \in \mathbb{R}^N$ represents the reference pattern associated with phase index k. To better understand the role of the phase-dependent prototype matrix $\mathbf{Q}$, we consider a simplified prediction setting in which the prediction depends solely on $\mathbf{Q}$, i.e., $\hat{\mathbf{x}}_l = \mathbf{Q}_{i_l}$, where $\hat{\mathbf{x}}_l \in \mathbb{R}^N$ denotes the predicted value at time step l (with $\mathbf{x}_l \in \mathbb{R}^N$ being the corresponding ground-truth).

Proposition 1 (Optimal Phase Prototype under MSE). *Under the above setting, consider the widely used mean squared error (MSE) objective* $\mathcal{L}(\mathbf{Q}) = \mathbb{E}\left[\|\mathbf{x}_l - \mathbf{Q}_{i_l}\|_2^2\right]$. *Then the optimal solution satisfies* $\mathbf{Q}_k^* = \mathbb{E}\left[\mathbf{x}_l \mid i_l = k\right], \quad \forall k \in \{0, 1, \ldots, W-1\}$.

Proposition 1 shows that, under the MSE objective, the prototype matrix $\mathbf{Q}$ learns the phase-conditioned mean of the time series, i.e., the dominant first-order periodic structure associated with each phase. However, removing this structure alone is insufficient to guarantee statistical invariance of the residual across phases. In particular, higher-order statistics, such as variance, may still depend on the phase index, as illustrated in Figure 1. As a result, the residual may retain structured, phase-dependent deviations beyond the mean, reflecting intrinsic heterogeneity in the underlying data distribution. We formalize this phenomenon in the following proposition.

Proposition 2 (Exact Preservation of Phase-Conditional Second-Order Structure). *Let* $\mathbf{Q}^*$ *be the optimal solution given in Proposition 1, namely,* $\mathbf{Q}_k^* = \mathbb{E}[\mathbf{x}_l \mid i_l = k]$ *for all* $k \in \{0, 1, \ldots, W-1\}$ *such that* $\mathbb{P}(i_l = k) > 0$. *Define the residual as* $\mathbf{r}_l = \mathbf{x}_l - \mathbf{Q}_{i_l}^*$. *Assume that the conditional second moments of* $\mathbf{x}_l$ *given* i_l *exist. Then, for any* $k \in \{0, 1, \ldots, W-1\}$ *such that* $\mathbb{P}(i_l = k) > 0$, *the residual satisfies* $\mathbb{E}[\mathbf{r}_l \mid i_l = k] = \mathbf{0}$, *and* $\mathrm{Cov}(\mathbf{r}_l \mid i_l = k) = \mathrm{Cov}(\mathbf{x}_l \mid i_l = k)$.

Propositions 1 and 2 together imply that, even when the phase-conditioned mean is optimally eliminated under the MSE objective (with the MAE counterpart provided in Appendix B.3), the residual inherently preserves higher-order phase-dependent structures. Consequently, when the conditional covariance $\mathrm{Cov}(\mathbf{x}_l \mid i_l = k)$ varies across phases, this variability is reflected in the residual, giving rise to the structured deviations observed in Figure 1. In practice, such phase-dependent effects are often further amplified, as the learned periodic pattern $\mathbf{Q}$ may deviate from the optimal solution due to finite data, optimization challenges, and model capacity limitations.

4 Method

In this section, we elaborate on our proposed method, *Paloma*, which is inspired by phase and amplitude modulation in signal processing [21, 22, 23, 24] and improves time series forecasting by explicitly modeling phase- and amplitude-dependent deviations in the residual.

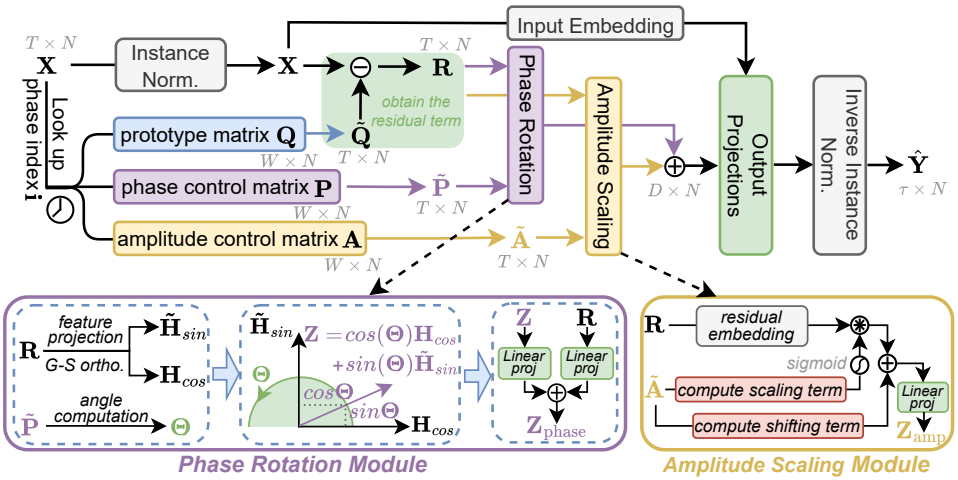


Figure 2: Overall architecture of Paloma. Paloma is a lightweight framework for modeling residual deviations, centered on (i) the Phase Rotation module, which captures phase-conditioned directional variations via rotation in the feature space, where G-S ortho. denotes Gram–Schmidt orthogonalization [34, 35], and (ii) the Amplitude Scaling module, which models phase-dependent magnitude variations through affine modulation.

4.1 Overview of Paloma

As illustrated in Figure 2, given a time series input $\mathbf{X} \in \mathbb{R}^{T\times N}$, we first apply instance normalization [32] to mitigate non-stationarity [32, 33] in time series data, and construct the phase indices $\mathbf{i}$ based on the cycle length W, following the formulation described in Section 3.1. Based on $\mathbf{i}$, we retrieve phase-aligned rows from three learnable matrices: the prototype matrix $\mathbf{Q}$, the phase control matrix $\mathbf{P}$, and the amplitude control matrix $\mathbf{A}$, obtaining $\tilde{\mathbf{Q}}$, $\tilde{\mathbf{P}}$, and $\tilde{\mathbf{A}}$, respectively, as described in Sections 4.1, 4.2, 4.3. We then remove the periodic component to obtain the residual term $\mathbf{R} = \mathbf{X} - \tilde{\mathbf{Q}}$. The residual is processed by two complementary modules: the *Phase Rotation Module*, which uses $\tilde{\mathbf{P}}$ to model directional variations via phase-conditioned rotations, and the *Amplitude Scaling Module*, which uses $\tilde{\mathbf{A}}$ to capture magnitude variations through adaptive scaling and shifting. Finally, the outputs of the two modules are fused and projected, combined with the input embedding via a residual connection, and mapped to the prediction horizon, followed by inverse instance normalization to produce the final forecasting result $\hat{\mathbf{Y}} \in \mathbb{R}^{\tau\times N}$.

Obtaining the Residual Term. Before introducing the two proposed modules designed to model the residual, we first provide additional details on how the residual is obtained. Given N channels and a priori cycle length W, we first define a learnable prototype matrix $\mathbf{Q} \in \mathbb{R}^{W\times N}$, where the k-th row $\mathbf{Q}_k \in \mathbb{R}^N$ represents the reference pattern associated with phase index k. The matrix $\mathbf{Q}$ is initialized with a standard Gaussian distribution and updated via backpropagation. Then the residual term $\mathbf{R} \in \mathbb{R}^{T\times N}$ is defined as $\mathbf{R} = \mathbf{X} - \tilde{\mathbf{Q}}$, where $\tilde{\mathbf{Q}} = [\mathbf{Q}_{i_0}, \mathbf{Q}_{i_1}, \ldots, \mathbf{Q}_{i_{T-1}}]^\top \in \mathbb{R}^{T\times N}$ denotes the phase-aligned prototype matrix.

4.2 Phase Rotation Module

Motivated by Proposition 2, the residual term exhibits phase-dependent second-order structure, suggesting that the residual may undergo phase-dependent geometric variations in the feature space. To capture such variations, we design a phase rotation module that performs a phase-conditioned transformation of residual features. Given the residual term $\mathbf{R} \in \mathbb{R}^{T\times N}$, we first map it into two latent feature representations along the temporal dimension:

$$\mathbf{H}_{\cos} = \mathrm{Linear}(\mathbf{R}) \in \mathbb{R}^{D\times N}, \quad \mathbf{H}_{\sin} = \mathrm{Linear}(\mathbf{R}) \in \mathbb{R}^{D\times N}, \tag{1}$$

where D denotes the hidden dimension. The subscripts cos and sin are used to emphasize that these two components will be combined through a trigonometric rotation, forming a phase-dependent transformation. To ensure a well-defined rotation, we construct an orthogonal basis from $(\mathbf{H}_{\cos}, \mathbf{H}_{\sin})$ using Gram–Schmidt orthogonalization [34, 35]:

$$\mathbf{H}_{\sin}^{\perp} = \mathbf{H}_{\sin} - \frac{\langle \mathbf{H}_{\sin}, \mathbf{H}_{\cos} \rangle}{\|\mathbf{H}_{\cos}\|^2} \mathbf{H}_{\cos} \in \mathbb{R}^{D\times N}, \tag{2}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product, and the resulting $\mathbf{H}_{\text{sin}}^{\perp}$ is orthogonal to $\mathbf{H}_{\cos}$. This orthogonalization stabilizes the subsequent rotation and avoids degenerate solutions where the two directions become collinear [34, 35].

To generate the phase-conditioned rotation angle, we introduce a learnable phase control matrix $\mathbf{P} \in \mathbb{R}^{W \times N}$, where the k-th row $\mathbf{P}_k \in \mathbb{R}^N$ encodes phase-specific information that modulates the rotation angle for phase k. The matrix $\mathbf{P}$ is defined and learned in the same manner as $\mathbf{Q}$. Accordingly, the phase-aligned control matrix is constructed as $\tilde{\mathbf{P}} = [\mathbf{P}_{i_0}, \mathbf{P}_{i_1}, \ldots, \mathbf{P}_{i_{T-1}}]^\top \in \mathbb{R}^{T \times N}$. We then compute the phase-dependent rotation angle $\mathbf{\Theta}$ and obtain the rotated representation $\mathbf{Z}$ as follows:

$$\mathbf{\Theta} = \pi \cdot \tanh(\text{Linear}(\tilde{\mathbf{P}})) \in \mathbb{R}^{D \times N}, \tag{3}$$

$$\mathbf{Z} = \cos(\mathbf{\Theta}) \odot \mathbf{H}_{\cos} + \sin(\mathbf{\Theta}) \odot \mathbf{H}_{\text{sin}}^{\perp} \in \mathbb{R}^{D \times N}. \tag{4}$$

This formulation performs a phase-conditioned rotation of residual features within a learned orthogonal subspace, enabling the model to capture phase-dependent geometric variations. The rotated representation $\mathbf{Z}$ is further processed and integrated with $\mathbf{R}$ through a residual connection:

$$\mathbf{Z}_{\text{phase}} = \text{LayerNorm}\left(\text{Linear}(\mathbf{Z}) + \text{Linear}(\mathbf{R})\right) \in \mathbb{R}^{D \times N}, \tag{5}$$

where $\text{LayerNorm}(\cdot)$ denotes layer normalization applied along the hidden dimension D.

4.3 Amplitude Scaling Module

Proposition 2 indicates that the residual exhibits phase-dependent second-order structure. From a covariance perspective, this structure is characterized by both principal directions and variance magnitudes, corresponding to the eigenvectors and eigenvalues of the covariance matrix, respectively. While the phase rotation module models phase-dependent variations in feature direction, variations in residual magnitude also play an important role in characterizing such structure. To capture this complementary aspect, we introduce an amplitude scaling module based on phase-conditioned affine modulation. Specifically, we introduce a learnable amplitude control matrix $\mathbf{A} \in \mathbb{R}^{W \times N}$, which parameterizes phase-conditioned scaling and shifting of residual features, where the k-th row $\mathbf{A}_k \in \mathbb{R}^N$ encodes the amplitude modulation parameters for phase k. The matrix $\mathbf{A}$ shares the same learning mechanism as the prototype matrix $\mathbf{Q}$ and the phase control matrix $\mathbf{P}$, ensuring a consistent parameterization across modules. Accordingly, the phase-aligned amplitude control matrix is constructed as $\tilde{\mathbf{A}} = [\mathbf{A}_{i_0}, \mathbf{A}_{i_1}, \ldots, \mathbf{A}_{i_{T-1}}]^\top \in \mathbb{R}^{T \times N}$.

We first embed the residual $\mathbf{R}$ into a hidden space as $\mathbf{R}_e = \text{Embedding}(\mathbf{R}) \in \mathbb{R}^{D \times N}$, where $\text{Embedding}(\cdot)$ denotes a linear projection, and $\mathbf{R}_e$ serves as the base representation for subsequent phase-conditioned modulation. We then compute phase-dependent scaling and shifting terms from the phase-aligned amplitude control matrix $\tilde{\mathbf{A}}$ as follows:

$$\mathbf{S} = \text{Sigmoid}(\text{Linear}(\tilde{\mathbf{A}})) \in \mathbb{R}^{D \times N}, \quad \mathbf{B} = \text{Linear}(\tilde{\mathbf{A}}) \in \mathbb{R}^{D \times N}. \tag{6}$$

The amplitude-modulated representation is obtained via a phase-conditioned affine transformation followed by a linear output projection:

$$\mathbf{Z}_{\text{amp}} = \text{Linear}(\mathbf{S} \odot \mathbf{R}_e + \mathbf{B}) \in \mathbb{R}^{D \times N}, \tag{7}$$

where $\odot$ denotes element-wise multiplication. The amplitude scaling module captures phase-dependent magnitude variations in the residual representation via affine modulation, complementing the phase rotation module to jointly characterize residual deviations in terms of directional and magnitude components.

4.4 Output Projections

We fuse the outputs of the phase rotation and amplitude scaling modules as $\mathbf{H} = \mathbf{Z}_{\text{phase}} + \mathbf{Z}_{\text{amp}} \in \mathbb{R}^{D \times N}$, and further process $\mathbf{H}$ with a lightweight projection to introduce nonlinearity and enable feature interaction, yielding $\mathbf{H}' = \text{GELU}(\text{Linear}(\mathbf{H})) \in \mathbb{R}^{D \times N}$. Meanwhile, the original input sequence $\mathbf{X}$ is projected into the same hidden space via a linear embedding, yielding $\mathbf{X}_e = \text{Embedding}(\mathbf{X}) \in \mathbb{R}^{D \times N}$, where $\mathbf{X}_e$ acts as a shortcut pathway that preserves the original input information and facilitates stable optimization. Finally, the hidden representation $\mathbf{H}'$ is combined with the input embedding $\mathbf{X}_e$ via a residual connection and projected to the prediction horizon:

$$\hat{\mathbf{Y}} = \text{Linear}(\mathbf{H}' + \mathbf{X}_e) \in \mathbb{R}^{\tau \times N}. \tag{8}$$

Table 1: Performance comparison with a fixed lookback window $T = 96$. The prediction lengths are set as $\tau \in \{12, 24, 48, 96\}$ for PEMS datasets for short-term forecasting, and $\tau \in \{96, 192, 336, 720\}$ for the others for long-term forecasting. Results are averaged over the four prediction lengths. The best and second-best results are highlighted in **red** and **blue**, respectively. Full results are provided in Table 6, and additional baseline results are reported in Table 7.

Models	**Paloma** [**ours**]		DistDF [ICLR'26]		GTR [ICLR'26]		RAFT [ICML'25]		FreDF [ICLR'25]		TimeXer [NeurIPS'24]		CycleNet [NeurIPS'24]		FilterNet [NeurIPS'24]		iTransformer [ICLR'24]		PatchTST [ICLR'23]		DLinear [AAAI'23]	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	**0.363**	**0.372**	0.378	0.394	**0.367**	**0.389**	0.381	0.400	0.392	0.399	0.382	0.397	0.379	0.396	0.392	0.401	0.407	0.410	0.387	0.400	0.403	0.407
ETTm2	**0.267**	**0.308**	0.277	0.321	0.268	0.315	0.281	0.330	0.278	0.319	0.274	0.322	**0.266**	**0.314**	0.285	0.328	0.288	0.332	0.281	0.326	0.350	0.401
ETTh1	**0.421**	**0.416**	0.430	**0.429**	0.439	0.434	**0.428**	0.433	0.437	0.435	0.437	0.437	0.457	0.441	0.441	0.439	0.454	0.448	0.469	0.455	0.456	0.452
ETTh2	**0.361**	**0.386**	**0.367**	**0.393**	0.372	0.400	0.382	0.410	0.371	0.396	0.368	0.396	0.388	0.409	0.383	0.407	0.383	0.407	0.387	0.407	0.559	0.515
Electricity	**0.166**	**0.251**	0.172	0.267	**0.166**	0.260	0.175	0.272	0.170	**0.259**	0.171	0.270	**0.168**	**0.259**	0.173	0.268	0.178	0.270	0.205	0.290	0.212	0.300
Traffic	0.461	**0.260**	**0.417**	**0.279**	0.470	0.280	**0.414**	0.284	0.421	**0.279**	0.466	0.287	0.472	0.301	0.463	0.310	0.428	0.282	0.481	0.300	0.625	0.383
Weather	**0.231**	**0.262**	0.248	0.275	**0.239**	**0.268**	0.270	0.309	0.254	0.274	0.241	0.271	0.243	0.271	0.245	0.272	0.258	0.278	0.259	0.273	0.265	0.317
Solar-Energy	**0.197**	**0.234**	0.207	0.255	**0.194**	**0.245**	0.301	0.303	0.211	0.262	0.237	0.302	0.210	0.261	0.235	0.266	0.233	0.262	0.270	0.307	0.330	0.401
PEMS03	**0.094**	**0.194**	0.123	0.233	**0.087**	**0.189**	0.144	0.230	0.113	0.219	0.112	0.214	0.118	0.226	0.145	0.251	0.113	0.222	0.180	0.291	0.278	0.375
PEMS04	**0.085**	**0.182**	0.092	0.195	**0.087**	**0.189**	0.104	0.210	0.096	0.202	0.105	0.209	0.119	0.232	0.146	0.258	0.111	0.221	0.195	0.307	0.295	0.388
PEMS07	**0.078**	**0.164**	0.084	0.177	**0.076**	**0.169**	0.094	0.193	0.089	0.182	0.085	0.182	0.113	0.214	0.123	0.229	0.101	0.204	0.211	0.303	0.329	0.396
PEMS08	**0.111**	**0.194**	0.151	0.244	**0.142**	**0.222**	0.151	0.234	0.173	0.255	0.175	0.250	0.150	0.246	0.172	0.260	0.150	0.226	0.280	0.321	0.379	0.416
1st Count	7	11	0	0	4	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

4.5 Complexity Analysis

Paloma adopts a channel-independent [8, 11] linear architecture, where the main computations are performed by linear projections along the temporal dimension and lightweight projections in the hidden space. The temporal projections in the Phase Rotation and Amplitude Scaling modules incur a cost of $\mathcal{O}(NTD)$, while the hidden-space projections introduce an additional cost of $\mathcal{O}(ND^2)$. The final prediction head contributes $\mathcal{O}(ND\tau)$. Other operations, including residual construction, phase-aligned prototype lookup, element-wise modulation, normalization, and the Gram–Schmidt orthogonalization step, are linear in the input or hidden size and are dominated by the projection operations. Therefore, the overall computational complexity of Paloma is $\mathcal{O}\big(N(TD + D^2 + D\tau)\big)$. Importantly, Paloma maintains linear complexity with respect to both the lookback length T and the number of channels N, making it suitable for efficient multivariate time series forecasting.

5 Experiments

5.1 Experimental Setup

Baselines and Datasets We consider 17 representative and widely recognized state-of-the-art forecasting models as baselines, including (i) Linear-based: GTR [20], RAFT [36], CycleNet [18], DLinear [11], SparseTSF [16], OLinear [37], TimeMixer [38]; (ii) Transformer-based: TimeXer [39], iTransformer [9], PatchTST [8]; (iii) Frequency-based: FilterNet [29], FreTS [40], FITS [41], Fredformer [42]; (iv) TCN-based: TimesNet [43]; (v) Two different training frameworks: DistDF [44] and FreDF [45]. To evaluate the effectiveness of our model, we conduct extensive experiments on eight benchmark datasets for long-term forecasting, including ETT (four subsets) [46], Electricity [25], Traffic [25], Solar-Energy [46], and Weather [25], and PEMS (four subsets) [14] for short-term forecasting. Detailed information on the baselines and datasets can be found in Appendix D.1 and Appendix D.2. And the implementation details are provided in Appendix D.3.

5.2 Main Results

Table 1 reports the forecasting performance for both long-term and short-term settings, where Paloma consistently outperforms a wide range of state-of-the-art methods across diverse datasets. We further report the standard deviation of Paloma across five runs with different random seeds in Table 7, highlighting its stable and robust performance. We attribute this performance gain to the proposed phase–amplitude modulation mechanism for modeling structured residual deviations. The phase rotation module captures phase-dependent variations by rotating feature representations in a learned feature space according to the cycle position, while the amplitude scaling module models the corresponding magnitude changes. Together, they enable Paloma to capture phase-dependent residual deviations, resulting in improved accuracy and robustness.

We also observe relatively higher MSE value on the Traffic dataset, which is likely related to the complex spatiotemporal correlations and temporal lag effects inherent in traffic data, particularly affecting long-term forecasting performance. While Paloma adopts a channel-independent modeling strategy that effectively captures per-channel temporal patterns, it may be less suited for modeling

Table 3: Generalization performance of Paloma Tech. on different models. The lookback window is fixed at 96 for all the experiments.

LookBack		iTransformer						PatchTST						DLinear						FilterNet					
Setup		Original		+ Paloma Tech.		Improve		Original		+ Paloma Tech.		Improve		Original		+ Paloma Tech.		Improve		Original		+ Paloma Tech.		Improve	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Electricity	96	0.150	0.241	0.137	0.233	8.67%	3.32%	0.164	0.254	0.137	0.234	16.46%	7.87%	0.195	0.277	0.139	0.239	28.72%	13.72%	0.147	0.245	0.137	0.239	6.80%	2.45%
	192	0.164	0.255	0.157	0.250	4.27%	1.96%	0.174	0.265	0.155	0.250	10.92%	5.66%	0.194	0.279	0.156	0.256	19.59%	8.24%	0.160	0.250	0.151	0.245	5.63%	2.00%
	336	0.178	0.272	0.172	0.267	3.37%	1.84%	0.193	0.285	0.176	0.272	8.81%	4.56%	0.207	0.295	0.172	0.275	16.91%	6.78%	0.173	0.267	0.164	0.261	5.20%	2.25%
	720	0.210	0.300	0.192	0.290	8.57%	3.33%	0.232	0.320	0.220	0.309	5.17%	3.44%	0.244	0.330	0.207	0.309	15.16%	6.36%	0.210	0.309	0.201	0.302	4.29%	2.27%
	Avg	0.176	0.267	0.165	0.260	6.25%	2.62%	0.191	0.281	0.172	0.266	9.85%	5.23%	0.210	0.295	0.169	0.270	19.76%	8.63%	0.173	0.268	0.163	0.262	5.78%	2.24%
PEMS03	12	0.078	0.185	0.062	0.164	20.51%	11.35%	0.072	0.179	0.065	0.172	9.72%	3.91%	0.105	0.221	0.061	0.165	41.90%	25.34%	0.071	0.177	0.062	0.165	12.68%	6.78%
	24	0.089	0.197	0.076	0.180	14.61%	8.63%	0.102	0.213	0.079	0.189	22.55%	11.27%	0.183	0.299	0.079	0.186	56.83%	37.79%	0.102	0.213	0.086	0.198	15.69%	7.04%
	48	0.176	0.288	0.110	0.213	37.50%	26.04%	0.155	0.263	0.107	0.215	30.97%	18.25%	0.315	0.407	0.110	0.214	65.08%	47.42%	0.162	0.272	0.134	0.255	17.28%	6.25%
	96	0.228	0.335	0.178	0.280	21.93%	16.42%	0.204	0.305	0.145	0.246	28.92%	19.34%	0.455	0.508	0.146	0.245	67.91%	51.77%	0.244	0.340	0.208	0.318	14.75%	6.47%
	Avg	0.143	0.251	0.107	0.209	25.42%	16.72%	0.133	0.240	0.099	0.206	25.73%	14.38%	0.265	0.359	0.099	0.203	62.57%	43.56%	0.145	0.251	0.123	0.234	15.17%	6.77%
PEMS04	12	0.081	0.189	0.067	0.168	17.28%	11.11%	0.087	0.195	0.080	0.190	8.05%	2.56%	0.114	0.228	0.076	0.184	33.33%	19.30%	0.082	0.190	0.066	0.178	19.51%	6.32%
	24	0.097	0.207	0.078	0.183	19.59%	11.59%	0.119	0.231	0.092	0.205	22.69%	11.26%	0.187	0.297	0.091	0.203	51.34%	31.65%	0.110	0.224	0.083	0.195	24.55%	12.95%
	48	0.128	0.241	0.100	0.208	21.88%	13.69%	0.172	0.279	0.113	0.226	34.30%	19.00%	0.319	0.402	0.119	0.235	62.70%	41.54%	0.160	0.276	0.116	0.238	27.50%	13.77%
	96	0.176	0.284	0.124	0.236	29.55%	16.90%	0.220	0.323	0.140	0.253	36.36%	21.67%	0.424	0.481	0.155	0.269	63.44%	44.07%	0.234	0.343	0.186	0.304	20.51%	11.37%
	Avg	0.121	0.230	0.092	0.199	23.40%	13.68%	0.150	0.257	0.106	0.219	28.90%	14.98%	0.261	0.352	0.110	0.223	57.74%	36.70%	0.146	0.258	0.113	0.229	22.60%	11.24%

complex channel interactions compared to methods explicitly designed for such dependencies, such as iTransformer [9]. Nevertheless, Paloma remains highly competitive in short-term traffic forecasting, achieving 5 top-1 and 3 top-2 results across 8 evaluation metrics on the four PEMS datasets.

5.3 Model Analysis

Ablation Studies. We conduct ablation studies to evaluate the contributions of the Phase Rotation (PR) and Amplitude Scaling (AS) modules across six datasets, as shown in Table 2. Specifically, removing either the PR or AS module leads to noticeable performance drops, underscoring the importance of modeling phase-dependent variations via directional transformations (PR) and magnitude modulation (AS). When both modules are removed, the performance further deteriorates, revealing their complementary roles in modeling structured residual deviations. These results validate the effectiveness of the proposed phase–amplitude modulation mechanism and emphasize the necessity of explicitly modeling structured residual deviations.

Table 2: Ablation study of the Phase Rotation (PR) and Amplitude Scaling (AS) modules on six datasets. The reported results are averaged over all prediction horizons. Detailed results are provided in Table 9.

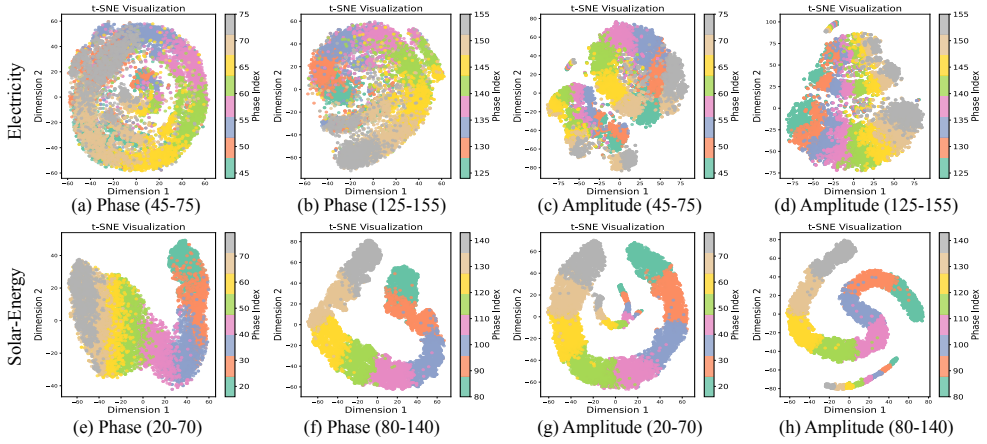
Design	ETTm1		ETTh1		Electricity		Weather		Traffic		Solar-Energy	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Paloma	**0.363**	**0.372**	**0.421**	**0.416**	**0.166**	**0.251**	**0.231**	**0.262**	**0.461**	**0.260**	**0.197**	**0.234**
w/o PR	0.368	0.374	0.427	0.420	0.170	0.258	0.238	0.270	0.471	0.267	0.206	0.247
w/o AS	0.369	0.374	0.430	0.424	0.170	0.259	0.238	0.270	0.468	0.269	0.201	0.243
w/o PR,AS	0.376	0.380	0.428	0.425	0.174	0.262	0.243	0.280	0.474	0.283	0.209	0.248

Paloma Tech. as plugin. Beyond the ablation study within the Paloma model, we further evaluate the generalizability of the proposed phase–amplitude modulation mechanism by applying it as a plugin module to other baselines. Specifically, we integrate the combination of the Phase Rotation and Amplitude Scaling modules, referred to as *Paloma Tech.*, into several representative forecasting architectures, including Transformer-based models (iTransformer [9] and PatchTST [8]), the Linear-based model DLinear [11], and the frequency-based model FilterNet [29]. As shown in Table 3, Paloma Tech. consistently improves the performance of all baseline models across different datasets and prediction horizons, demonstrating that the proposed modulation mechanism is architecture-agnostic and effective for capturing phase-dependent variations, whether applied to residuals or directly to input sequences. Notably, the performance gains are particularly significant when integrating Paloma Tech. into DLinear. Although DLinear benefits from seasonal–trend decomposition to separate coarse temporal components, its forecasting process still mainly relies on simple linear transformations and lacks an explicit mechanism for modeling phase-dependent temporal variations. By introducing the proposed phase–amplitude modulation mechanism, Paloma Tech. effectively complements this limitation, enabling DLinear to capture more structured phase-aware temporal dynamics.

5.4 Visualization Analysis

To better understand the roles of the proposed Phase Rotation and Amplitude Scaling modules, we conduct t-SNE visualization analysis of the learned representations $\mathbf{Z}_{\text{phase}}$ and $\mathbf{Z}_{\text{amp}}$ on the Electricity and Solar-Energy datasets, as shown in Figures 3. This analysis aims to qualitatively examine how the proposed phase–amplitude modulation mechanism organizes the representation space in a structured manner, where phase-aware structures and magnitude-related variations are modeled by complementary components

Phase representations. As illustrated in Figures 3(a)–(b) and (e)–(f), the phase representations $\mathbf{Z}_{\text{phase}}$ exhibit clear phase-aware organization in the hidden space. Samples with nearby phase indices are arranged along continuous trajectories, forming distinct geometric patterns, such as a spiral-



(a) Phase (45-75) (b) Phase (125-155) (c) Amplitude (45-75) (d) Amplitude (125-155)

(e) Phase (20-70) (f) Phase (80-140) (g) Amplitude (20-70) (h) Amplitude (80-140)

Figure 3: t-SNE visualizations of $\mathbf{Z}_{\text{phase}}$ and $\mathbf{Z}_{\text{amp}}$ on Electricity and Solar-Energy datasets under the setting of lookback window $T = 96$ and prediction length $\tau = 96$. The inherent periodicities of the Electricity and Solar-Energy datasets are 168 and 144, respectively, and the numbers in parentheses denote the corresponding ranges of phase indices.

like structure on the Electricity dataset and an irregular U-shaped structure on the Solar-Energy dataset. The color progression along the trajectories indicates that the learned representations follow the intrinsic periodic ordering of the data, suggesting that the Phase Rotation module captures phase-dependent variations and arranges features according to their cycle positions.

Amplitude representations. As shown in Figures 3(c)–(d) and (g)–(h), the amplitude representations $\mathbf{Z}_{\text{amp}}$ exhibit organization patterns that are clearly distinct from those of $\mathbf{Z}_{\text{phase}}$. On the Electricity dataset (Figures 3(c)–(d)), the amplitude representations appear more scattered and intermixed, reflecting the more complex and heterogeneous consumption patterns. In contrast, On the Solar-Energy dataset (Figures 3(g)–(h)), $\mathbf{Z}_{\text{amp}}$ exhibits a structured yet non-uniform geometry, forming a dominant continuous manifold alongside a thinner secondary branch. The main manifold captures the majority of samples and reflects the principal variation in residual magnitudes, while the secondary branch corresponds to a smaller subset with distinct amplitude characteristics. Taken together, these patterns suggest that the Amplitude Scaling module captures heterogeneous amplitude variations conditioned on phase.

5.5 Efficiency Analysis

To comprehensively assess both effectiveness and efficiency, we further conduct efficiency experiments on the Weather dataset (21 channels), as reported in Table 4. To ensure a fair comparison, all models are evaluated under a unified training framework with a fixed batch size of 32. From the results, Paloma achieves a favorable balance between predictive performance and computational cost.

Table 4: Efficiency comparison between Paloma and other models on the Weather dataset with lookback window T = 96 and forecast horizon τ = 720. Training Time denotes the average time required per epoch for the model.

Model	Parameters	MACs	Training Time	Memory Usage	MSE	MAE
GTR	150.71K	3.21M	9.4s	1754MB	0.341	0.342
CycleNet	108.32K	2.22M	8.2s	1414MB	0.344	0.344
DLinear	139.68K	2.94M	7.4s	1420MB	0.345	0.381
FilterNet	369.40K	7.19M	12.1s	1434MB	0.342	0.340
iTransformer	1.29M	27.11M	16.6s	1858MB	0.358	0.349
PatchTST	2.30M	326.33M	21.2s	2130MB	0.354	0.340
TimeXer	1.98M	191.15M	32.6s	2102MB	0.340	0.341
Paloma(ours)	260.38K	5.08M	12.9s	1424MB	0.324	0.336

Compared to lightweight linear models such as GTR, CycleNet, and DLinear, Paloma introduces only a moderate increase in parameters and MACs, while delivering notably improved forecasting accuracy. In contrast to Transformer-based models (e.g., iTransformer and PatchTST), Paloma demonstrates advantages in both efficiency and effectiveness, requiring substantially lower computational resources while achieving better predictive performance. Efficiency results on the ETTh1 dataset (7 channels) and the Traffic dataset (862 channels) are provided in Tables 10 and 11, further confirming the efficiency of Paloma across datasets with varying channel sizes.

6 Conclusion and Limitation

In this work, we show that the residual term in cycle–residual decomposition retains intrinsic phase-dependent structures rather than behaving as unstructured noise. Motivated by this observation, we propose *Paloma*, a simple yet effective framework inspired by phase and amplitude modulation in classical signal processing. By introducing a Phase Rotation module for modeling directional variations and an Amplitude Scaling module for capturing magnitude variations, Paloma explicitly models structured temporal dynamics conditioned on phase. Beyond the proposed architecture, we further abstract the two modules into a general-purpose plugin, termed *Paloma technique*, which can be readily integrated into existing models to enhance phase-aware temporal modeling. Extensive experiments demonstrate the effectiveness, robustness, and generalizability of the proposed approach.

Nevertheless, the proposed method has several limitations. First, its performance depends on the availability of a reasonably accurate prior for the cycle length W. As shown in the sensitivity analysis in Appendix E.4.2, Paloma is sensitive to periodicity misalignment. Interestingly, this sensitivity may also serve as a diagnostic signal for estimating the underlying periodicity of a dataset, since forecasting performance tends to deteriorate when the assumed cycle length becomes closer to, yet remains misaligned with, the intrinsic period (see Appendix E.4.2 for detailed analysis). Second, the current formulation is primarily designed for periodic or quasi-periodic signals, and extending the framework to more general non-periodic scenarios remains an important direction for future work.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (62272048); and in part by the National Key Research and Development Program of China (2019YFB1406302).

References

[1] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pages 27268–27286. PMLR, 2022.

[2] Wei Fan, Pengyang Wang, Dongkun Wang, Dongjie Wang, Yuanchun Zhou, and Yanjie Fu. Dish-ts: a general paradigm for alleviating distribution shift in time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 7522–7529, 2023.

[3] Jingru Fei, Kun Yi, Alex Xing Wang, Qingsong Wen, Xiangxiang Zhu, and Wei Fan. Olivia: Harmonizing time series foundation models with power spectral density. In *Forty-third International Conference on Machine Learning*, 2026.

[4] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. Adaptive graph convolutional recurrent network for traffic forecasting. *Advances in neural information processing systems*, 33:17804–17815, 2020.

[5] Chengxin Wang, Yiran Zhao, Shaofeng Cai, and Gary Tan. Investigating pattern neurons in urban time series forecasting. In *The Thirteenth International Conference on Learning Representations*, 2025.

[6] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International conference on learning representations*, 2021.

[7] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. Temporal relational ranking for stock prediction. *ACM Transactions on Information Systems (TOIS)*, 37(2):1–30, 2019.

[8] 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.

[9] 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.

[10] Xiangfei Qiu, Yuhan Zhu, Zhengyu Li, Xingjian Wu, Bin Yang, and Jilin Hu. DAG: A dual correlation network for time series forecasting with exogenous variables. In *ICML*, 2026.

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

[12] Jingru Fei, Kun Yi, Wei Fan, Qi Zhang, and Zhendong Niu. Amplifier: Bringing attention to neglected low-energy components in time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pages 11645–11653, 2025.

[13] Donghao Luo and Xue Wang. Moderntcn: A modern pure convolution structure for general time series analysis. In *ICLR*, 2024.

[14] Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in neural information processing systems*, 35:5816–5828, 2022.

[15] Shengsheng Lin, Weiwei Lin, Wentai Wu, Feiyu Zhao, Ruichao Mo, and Haotong Zhang. Segrnn: Segment recurrent neural network for long-term time series forecasting. *IEEE Internet of Things Journal*, 2025.

[16] Shengsheng Lin, Weiwei Lin, Wentai Wu, Haojun Chen, and Junjie Yang. Sparsetsf: Modeling long-term time series forecasting with 1k parameters. *arXiv preprint arXiv:2405.00946*, 2024.

[17] Wei Fan, Shun Zheng, Xiaohan Yi, Wei Cao, Yanjie Fu, Jiang Bian, and Tie-Yan Liu. DEPTS: Deep expansion learning for periodic time series forecasting. In *International Conference on Learning Representations*, 2022.

[18] Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. Cyclenet: Enhancing time series forecasting through modeling periodic patterns. *Advances in Neural Information Processing Systems*, 37:106315–106345, 2024.

[19] Boya Zhang, Shuaijie Yin, Huiwen Zhu, and Xing He. Freqcycle: A multi-scale time-frequency analysis method for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 28159–28167, 2026.

[20] Fanpu Cao, Lu Dai, Jindong Han, and Hui Xiong. Enhancing multivariate time series forecasting with global temporal retrieval. *arXiv preprint arXiv:2602.10847*, 2026.

[21] John B Anderson, Tor Aulin, and Carl-Erik Sundberg. *Digital phase modulation*. Springer Science & Business Media, 2013.

[22] Philip X Joris, Christoph E Schreiner, and Adrian Rees. Neural processing of amplitude-modulated sounds. *Physiological reviews*, 84(2):541–577, 2004.

[23] John G Proakis and Masoud Salehi. *Digital communications*, volume 4. McGraw-hill New York, 2001.

[24] Alan V Oppenheim. *Discrete-time signal processing*. Pearson Education India, 1999.

[25] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.

[26] Guoqi Yu, Jing Zou, Xiaowei Hu, Angelica I Aviles-Rivero, Jing Qin, and Shujun Wang. Revitalizing multivariate time series forecasting: Learnable decomposition with inter-series dependencies and intra-series variations modeling. *arXiv preprint arXiv:2402.12694*, 2024.

[27] Richard Asselin. Frequency filter for time integrations. *Monthly Weather Review*, 100(6):487–490, 1972.

[28] Kun Yi, Qi Zhang, Wei Fan, Longbing Cao, Shoujin Wang, Hui He, Guodong Long, Liang Hu, Qingsong Wen, and Hui Xiong. A survey on deep learning based time series analysis with frequency transformation. In *Proceedings of the 31st ACM SIGKDD conference on knowledge discovery and data mining v. 2*, pages 6206–6215, 2025.

[29] Kun Yi, Jingru Fei, Qi Zhang, Hui He, Shufeng Hao, Defu Lian, and Wei Fan. Filternet: Harnessing frequency filters for time series forecasting. *Advances in Neural Information Processing Systems*, 37:55115–55140, 2024.

[30] Mingyuan Xia, Chunxu Zhang, Zijian Zhang, Hao Miao, Qidong Liu, Yuanshao Zhu, and Bo Yang. Timeemb: A lightweight static-dynamic disentanglement framework for time series forecasting. *arXiv preprint arXiv:2510.00461*, 2025.

[31] Yiming Niu, Jinliang Deng, and Yongxin Tong. Phaseformer: From patches to phases for efficient and effective time series forecasting. *arXiv preprint arXiv:2510.04134*, 2025.

[32] Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International conference on learning representations*, 2021.

[33] Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in neural information processing systems*, 35:9881–9893, 2022.

[34] Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

[35] Werner H Greub. *Linear algebra*. Springer Science & Business Media, 2012.

[36] Sungwon Han, Seungeon Lee, Meeyoung Cha, Sercan O Arik, and Jinsung Yoon. Retrieval augmented time series forecasting. *arXiv preprint arXiv:2505.04163*, 2025.

[37] Wenzhen Yue, Yong Liu, Haoxuan Li, Hao Wang, Xianghua Ying, Ruohao Guo, Bowei Xing, and Ji Shi. Olinear: A linear model for time series forecasting in orthogonally transformed domain. *arXiv preprint arXiv:2505.08550*, 2025.

[38] 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.

[39] Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Guo Qin, Haoran Zhang, Yong Liu, Yunzhong Qiu, Jianmin Wang, and Mingsheng Long. Timexer: Empowering transformers for time series forecasting with exogenous variables. *Advances in Neural Information Processing Systems*, 37:469–498, 2024.

[40] Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Ning An, Defu Lian, Longbing Cao, and Zhendong Niu. Frequency-domain mlps are more effective learners in time series forecasting. *Advances in Neural Information Processing Systems*, 36:76656–76679, 2023.

[41] Zhijian Xu, Ailing Zeng, and Qiang Xu. Fits: Modeling time series with $10k$ parameters. *arXiv preprint arXiv:2307.03756*, 2023.

[42] Xihao Piao, Zheng Chen, Taichi Murayama, Yasuko Matsubara, and Yasushi Sakurai. Fredformer: Frequency debiased transformer for time series forecasting. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 2400–2410, 2024.

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

[44] Hao Wang, Licheng Pan, Yuan Lu, Zhixuan Chu, Xiaoxi Li, Shuting He, Zhichao Chen, Haoxuan Li, Qingsong Wen, and Zhouchen Lin. Distdf: Time-series forecasting needs joint-distribution wasserstein alignment. *arXiv preprint arXiv:2510.24574*, 2025.

[45] Hao Wang, Licheng Pan, Zhichao Chen, Degui Yang, Sen Zhang, Yifei Yang, Xinggao Liu, Haoxuan Li, and Dacheng Tao. Fredf: Learning to forecast in the frequency domain. *arXiv preprint arXiv:2402.02399*, 2024.

[46] 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, pages 11106–11115, 2021.

[47] Trevor Hastie, Robert Tibshirani, Jerome Friedman, et al. The elements of statistical learning, 2009.

[48] Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical transactions of the royal society a: mathematical, physical and engineering sciences*, 379(2194), 2021.

[49] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 95–104, 2018.

[50] Qihe Huang, Lei Shen, Ruixin Zhang, Shouhong Ding, Binwu Wang, Zhengyang Zhou, and Yang Wang. Crossgnn: Confronting noisy multivariate time series via cross interaction refinement. *Advances in Neural Information Processing Systems*, 36:46885–46902, 2023.

[51] Kun Yi, Qi Zhang, Wei Fan, Hui He, Liang Hu, Pengyang Wang, Ning An, Longbing Cao, and Zhendong Niu. Fouriergnn: Rethinking multivariate time series forecasting from a pure graph perspective. *Advances in neural information processing systems*, 36:69638–69660, 2023.

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[53] Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*, 2023.

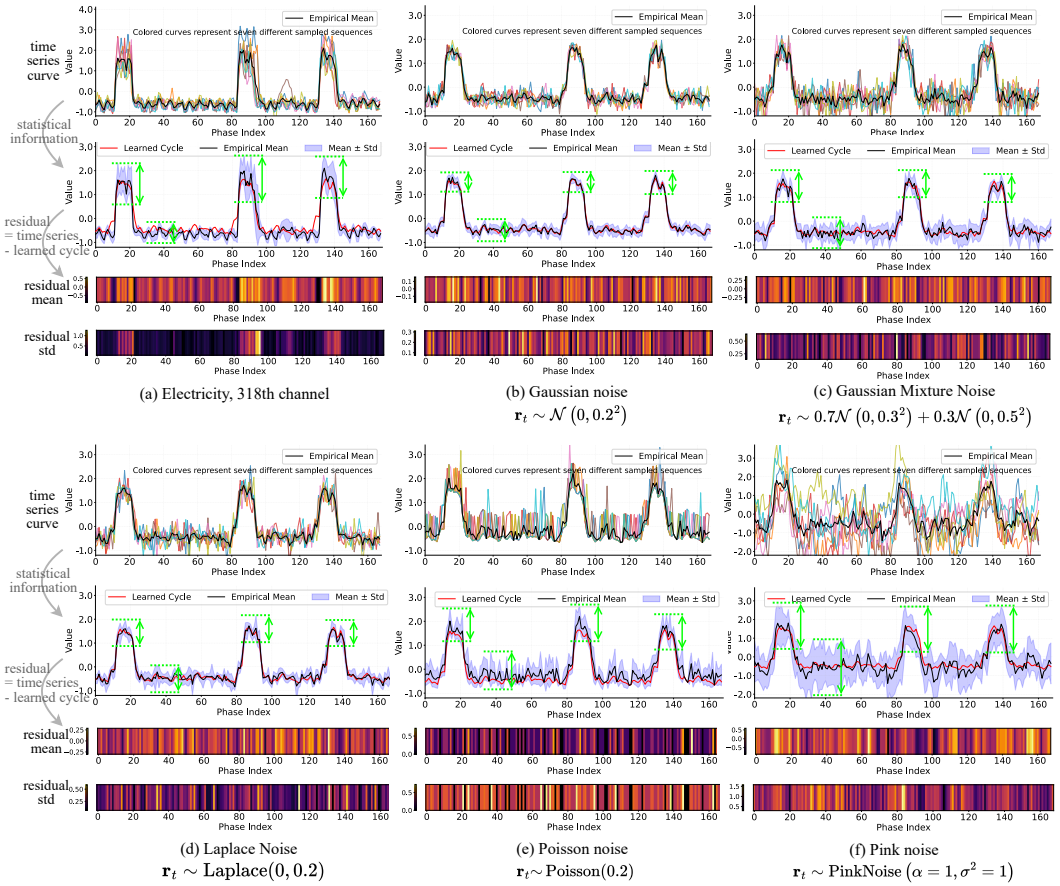
[54] Zhe Li, Shiyi Qi, Yiduo Li, and Zenglin Xu. Revisiting long-term time series forecasting: An investigation on linear mapping. *arXiv preprint arXiv:2305.10721*, 2023.

[55] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[56] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[57] Wang Xue, Tian Zhou, Qingsong Wen, Jinyang Gao, Bolin Ding, and Rong Jin. Card: Channel aligned robust blend transformer for time series forecasting. *arXiv preprint arXiv:2305.12095*, 2023.

A More Explanations about Motivation



(a) Electricity, 318th channel

(b) Gaussian noise $\mathbf{r}_t \sim \mathcal{N}\left(0, 0.2^2\right)$

(c) Gaussian Mixture Noise $\mathbf{r}_t \sim 0.7\mathcal{N}\left(0, 0.3^2\right) + 0.3\mathcal{N}\left(0, 0.5^2\right)$

(d) Laplace Noise $\mathbf{r}_t \sim \text{Laplace}(0, 0.2)$

(e) Poisson noise $\mathbf{r}_t \sim \text{Poisson}(0.2)$

(f) Pink noise $\mathbf{r}_t \sim \text{PinkNoise}\left(\alpha = 1, \sigma^2 = 1\right)$

Figure 4: Comparison between real-world residuals and synthetic noise-based counterparts under different noise types. Subfigure (a) shows the real-world case from the Electricity dataset (318th channel). Subfigures (b)–(f) present synthetic sequences constructed by combining the learned cycle in (a) with different noise types, including Gaussian, Gaussian mixture, Laplace, Poisson, and pink noise. The meaning of each row is consistent with that in Figure 1. Unlike real-world data, where residual statistics exhibit clear phase-dependent patterns, all noise-based counterparts fail to produce consistent phase-aligned structure, regardless of distributional complexity or variability characteristics.

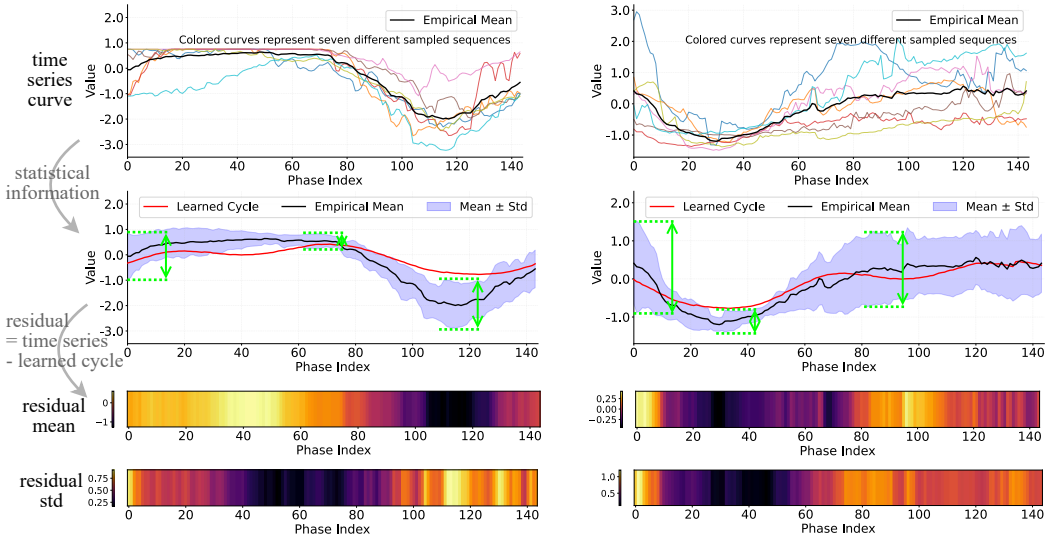
To further validate the observations in Figure 1(c), we construct additional synthetic examples by combining the learned cycle from the Electricity dataset (318th channel) with various types of noise. The corresponding visualizations are shown in Figure 4. Specifically, we consider a diverse set of noise distributions, including Gaussian noise, Gaussian mixture noise, Laplace noise, Poisson noise, and Pink noise. For each case, the synthetic time series is generated as $\mathbf{x}_t = \mathbf{Q}_{i_t} + \mathbf{r}_t$, where $\mathbf{Q}_{i_t}$ denotes the learned cycle and $\mathbf{r}_t$ is sampled from the corresponding noise distribution. The results reveal several important observations:

(1) Independently sampled noise does not induce phase-dependent variance patterns. For noise types such as Gaussian (Figure 4(b)) and Laplace (Figure 4(d)) distributions, the residual standard deviation remains approximately homogeneous across phase indices. The residual mean is close to zero mainly by construction, as the sampled noise is zero-mean. Therefore, the key observation is not the mean behavior, but the absence of systematic phase-dependent variation in the residual standard deviation. This suggests that independently sampled noise alone cannot reproduce the phase-dependent residual variability observed in real-world data.

(2) More flexible noise distributions still fail to produce phase-aligned structure. For more flexible noise models, such as Gaussian mixture noise (Figure 4(c)) and Poisson noise (Figure 4(e)), the

residual variance may exhibit increased local variability due to the underlying distributional properties. However, these variations remain unstructured with respect to the phase index, indicating that distributional complexity alone is insufficient to induce phase-dependent patterns. A similar observation holds for colored noise (e.g., pink noise in Figure 4(f)), where the residual exhibits smoother and more continuous fluctuations compared to independently sampled noise. Nevertheless, these variations still do not exhibit consistent dependence on the phase index, and thus fail to form structured, phase-dependent patterns. Overall, these results demonstrate that diverse noise characteristics, including heterogeneous distributions and temporally smoother fluctuations, are insufficient to reproduce the stable, phase-dependent variance structures observed in real-world data.

Taken together, these results demonstrate that the structured residual deviations observed in real-world time series (Figure 4(a)) cannot be simply explained as noise. Instead, they reflect intrinsic, phase-dependent statistical properties of the underlying data distribution. This provides strong empirical motivation for explicitly modeling phase-dependent residual dynamics, as proposed in our method.



(a) Weather, 5th channel

(b) Weather, 21th channel

Figure 5: Visualization of structured residual deviations on the Weather dataset. The row definitions are consistent with those in Figure 1.

Figure 5 provides additional visualizations of residual deviations on the Weather dataset. Consistent with the observations in Figure 1, the residual statistics exhibit clear phase-dependent patterns across different channels. In particular, both the residual mean and residual standard deviation vary systematically along the phase index, indicating that the residual cannot be regarded as unstructured noise after cycle removal. These results further support our motivation that residual dynamics retain structured phase-dependent variations beyond the learned periodic component.

B Theoretical Supplement

B.1 Proof of Proposition 1

We begin by restating the setting used in Proposition 1. Let $\mathbf{X} = \{\mathbf{x}_l\}_{l=0}^{T-1} \in \mathbb{R}^{T\times N}$ be a sequence with phase index sequence $\mathbf{i} = \{i_l\}_{l=0}^{T-1}$, where $i_l \in \{0, 1, \ldots, W-1\}$. We consider a simplified prediction setting in which the prediction depends solely on the learnable prototype matrix $\mathbf{Q} \in \mathbb{R}^{W\times N}$, i.e., $\hat{\mathbf{x}}_l = \mathbf{Q}_{i_l}$, where $\hat{\mathbf{x}}_l \in \mathbb{R}^N$ and $\mathbf{x}_l \in \mathbb{R}^N$ denote the predicted and corresponding ground-truth values at time step l, respectively.

Proposition 1 (Optimal Phase Prototype under MSE). *Under the above setting, consider the widely used mean squared error (MSE) objective* $\mathcal{L}(\mathbf{Q}) = \mathbb{E}\left[\|\mathbf{x}_l - \mathbf{Q}_{i_l}\|_2^2\right]$. *Then the optimal*

solution satisfies

$$\mathbf{Q}_k^* = \mathbb{E}\left[\mathbf{x}_l \mid i_l = k\right], \quad \forall k \in \{0, 1, \ldots, W-1\}. \tag{9}$$

Proof. Assume that $\mathbb{E}\|\mathbf{x}_l\|_2^2 < \infty$. By the law of total expectation, the objective can be decomposed according to the phase index:

$$\mathcal{L}(\mathbf{Q}) = \mathbb{E}\left[\|\mathbf{x}_l - \mathbf{Q}_{i_l}\|_2^2\right] = \sum_{k=0}^{W-1} \mathbb{P}(i_l = k)\, \mathbb{E}\left[\|\mathbf{x}_l - \mathbf{Q}_{i_l}\|_2^2 \mid i_l = k\right]. \tag{10}$$

Conditioned on the event $i_l = k$, we have $\mathbf{Q}_{i_l} = \mathbf{Q}_k$, and hence

$$\mathcal{L}(\mathbf{Q}) = \sum_{k=0}^{W-1} \mathbb{P}(i_l = k)\, \mathbb{E}\left[\|\mathbf{x}_l - \mathbf{Q}_k\|_2^2 \mid i_l = k\right]. \tag{11}$$

Therefore, for any k such that $\mathbb{P}(i_l = k) > 0$, minimizing $\mathcal{L}(\mathbf{Q})$ with respect to $\mathbf{Q}_k$ is equivalent to minimizing the phase-wise term

$$\mathcal{L}_k(\mathbf{Q}_k) := \mathbb{E}\left[\|\mathbf{x}_l - \mathbf{Q}_k\|_2^2 \mid i_l = k\right]. \tag{12}$$

For a fixed $k \in \{0, 1, \ldots, W-1\}$, let

$$\boldsymbol{\mu}_k := \mathbb{E}\left[\mathbf{x}_l \mid i_l = k\right]. \tag{13}$$

Then

$$\mathbf{x}_l - \mathbf{Q}_k = (\mathbf{x}_l - \boldsymbol{\mu}_k) + (\boldsymbol{\mu}_k - \mathbf{Q}_k). \tag{14}$$

Substituting this into $\mathcal{L}_k(\mathbf{Q}_k)$ yields

$$\mathcal{L}_k(\mathbf{Q}_k) = \mathbb{E}\left[\|(\mathbf{x}_l - \boldsymbol{\mu}_k) + (\boldsymbol{\mu}_k - \mathbf{Q}_k)\|_2^2 \mid i_l = k\right] \tag{15}$$

$$= \mathbb{E}\left[\|\mathbf{x}_l - \boldsymbol{\mu}_k\|_2^2 \mid i_l = k\right] + 2\,\mathbb{E}\left[(\mathbf{x}_l - \boldsymbol{\mu}_k)^\top (\boldsymbol{\mu}_k - \mathbf{Q}_k) \mid i_l = k\right] + \|\boldsymbol{\mu}_k - \mathbf{Q}_k\|_2^2. \tag{16}$$

Since $\boldsymbol{\mu}_k - \mathbf{Q}_k$ is deterministic under the conditional distribution given $i_l = k$, the cross term becomes

$$\mathbb{E}\left[(\mathbf{x}_l - \boldsymbol{\mu}_k)^\top (\boldsymbol{\mu}_k - \mathbf{Q}_k) \mid i_l = k\right] = \mathbb{E}\left[\mathbf{x}_l - \boldsymbol{\mu}_k \mid i_l = k\right]^\top (\boldsymbol{\mu}_k - \mathbf{Q}_k). \tag{17}$$

By the definition of $\boldsymbol{\mu}_k$,

$$\mathbb{E}\left[\mathbf{x}_l - \boldsymbol{\mu}_k \mid i_l = k\right] = \mathbb{E}\left[\mathbf{x}_l \mid i_l = k\right] - \boldsymbol{\mu}_k = \mathbf{0}. \tag{18}$$

Hence the cross term vanishes, and we obtain

$$\mathcal{L}_k(\mathbf{Q}_k) = \mathbb{E}\left[\|\mathbf{x}_l - \boldsymbol{\mu}_k\|_2^2 \mid i_l = k\right] + \|\boldsymbol{\mu}_k - \mathbf{Q}_k\|_2^2. \tag{19}$$

The first term is independent of $\mathbf{Q}_k$, while the second term is nonnegative and is minimized if and only if

$$\mathbf{Q}_k = \boldsymbol{\mu}_k = \mathbb{E}\left[\mathbf{x}_l \mid i_l = k\right]. \tag{20}$$

Therefore, for every $k \in \{0, 1, \ldots, W-1\}$ such that $\mathbb{P}(i_l = k) > 0$, the optimal solution satisfies

$$\mathbf{Q}_k^* = \mathbb{E}\left[\mathbf{x}_l \mid i_l = k\right]. \tag{21}$$

This completes the proof. □

B.2 Proof of Proposition 2

Proposition 2 (Exact Preservation of Phase-Conditional Second-Order Structure). *Let $\mathbf{Q}^*$ be the optimal solution given in Proposition 1, namely, $\mathbf{Q}_k^* = \mathbb{E}[\mathbf{x}_l \mid i_l = k]$ for all $k \in \{0, 1, \ldots, W-1\}$ such that $\mathbb{P}(i_l = k) > 0$. Define the residual as $\mathbf{r}_l = \mathbf{x}_l - \mathbf{Q}_{i_l}^*$. Assume that the conditional second moments of $\mathbf{x}_l$ given i_l exist. Then, for any $k \in \{0, 1, \ldots, W-1\}$ such that $\mathbb{P}(i_l = k) > 0$, the residual satisfies*

$$\mathbb{E}[\mathbf{r}_l \mid i_l = k] = \mathbf{0}, \tag{22}$$

and

$$\mathrm{Cov}(\mathbf{r}_l \mid i_l = k) = \mathrm{Cov}(\mathbf{x}_l \mid i_l = k). \tag{23}$$

Proof. Fix any $k \in \{0, 1, \ldots, W-1\}$ such that $\mathbb{P}(i_l = k) > 0$. Define

$$\boldsymbol{\mu}_k := \mathbb{E}[\mathbf{x}_l \mid i_l = k]. \tag{24}$$

By Proposition 1, we have

$$\mathbf{Q}_k^* = \boldsymbol{\mu}_k. \tag{25}$$

Conditioned on $i_l = k$, the residual becomes

$$\mathbf{r}_l = \mathbf{x}_l - \mathbf{Q}_{i_l}^* = \mathbf{x}_l - \mathbf{Q}_k^* = \mathbf{x}_l - \boldsymbol{\mu}_k. \tag{26}$$

We first compute its conditional mean:

$$\mathbb{E}[\mathbf{r}_l \mid i_l = k] = \mathbb{E}[\mathbf{x}_l - \boldsymbol{\mu}_k \mid i_l = k] = \mathbb{E}[\mathbf{x}_l \mid i_l = k] - \boldsymbol{\mu}_k = \boldsymbol{\mu}_k - \boldsymbol{\mu}_k = \mathbf{0}. \tag{27}$$

Next, by the definition of conditional covariance,

$$\mathrm{Cov}(\mathbf{r}_l \mid i_l = k) = \mathbb{E}\big[(\mathbf{r}_l - \mathbb{E}[\mathbf{r}_l \mid i_l = k])(\mathbf{r}_l - \mathbb{E}[\mathbf{r}_l \mid i_l = k])^\top \mid i_l = k\big]. \tag{28}$$

Since $\mathbb{E}[\mathbf{r}_l \mid i_l = k] = \mathbf{0}$, it follows that

$$\mathrm{Cov}(\mathbf{r}_l \mid i_l = k) = \mathbb{E}[\mathbf{r}_l \mathbf{r}_l^\top \mid i_l = k]. \tag{29}$$

Substituting $\mathbf{r}_l = \mathbf{x}_l - \boldsymbol{\mu}_k$, we obtain

$$\mathrm{Cov}(\mathbf{r}_l \mid i_l = k) = \mathbb{E}\big[(\mathbf{x}_l - \boldsymbol{\mu}_k)(\mathbf{x}_l - \boldsymbol{\mu}_k)^\top \mid i_l = k\big]. \tag{30}$$

By the definition of conditional covariance and $\boldsymbol{\mu}_k = \mathbb{E}[\mathbf{x}_l \mid i_l = k]$, the right-hand side is exactly $\mathrm{Cov}(\mathbf{x}_l \mid i_l = k)$. Therefore,

$$\mathrm{Cov}(\mathbf{r}_l \mid i_l = k) = \mathrm{Cov}(\mathbf{x}_l \mid i_l = k). \tag{31}$$

This completes the proof. □

B.3 Proposition 3 (Optimal Phase Prototype under MAE)

Proposition 3 (Optimal Phase Prototype under MAE). *Under the same simplified prediction setting, i.e., $\hat{\mathbf{x}}_l = \mathbf{Q}_{i_l}$, consider the mean absolute error (MAE) objective $\mathcal{L}_{\mathrm{MAE}}(\mathbf{Q}) = \mathbb{E}\left[\|\mathbf{x}_l - \mathbf{Q}_{i_l}\|_1\right]$. Assume that the conditional first absolute moments exist. Then, for any $k \in \{0, 1, \ldots, W-1\}$ such that $\mathbb{P}(i_l = k) > 0$, the optimal solution satisfies*

$$\mathbf{Q}_{k,n}^* \in \mathrm{Median}\left(x_{l,n} \mid i_l = k\right), \quad \forall n \in \{1, \ldots, N\}, \tag{32}$$

where $x_{l,n}$ and $\mathbf{Q}_{k,n}$ denote the n-th channel of $\mathbf{x}_l$ and $\mathbf{Q}_k$, respectively.

Proof. By the law of total expectation, the MAE objective can be decomposed according to the phase index:

$$\mathcal{L}_{\mathrm{MAE}}(\mathbf{Q}) = \sum_{k=0}^{W-1} \mathbb{P}(i_l = k)\mathbb{E}\left[\|\mathbf{x}_l - \mathbf{Q}_{i_l}\|_1 \mid i_l = k\right]. \tag{33}$$

Conditioned on $i_l = k$, we have $\mathbf{Q}_{i_l} = \mathbf{Q}_k$, and hence

$$\mathcal{L}_{\mathrm{MAE}}(\mathbf{Q}) = \sum_{k=0}^{W-1} \mathbb{P}(i_l = k)\mathbb{E}\left[\|\mathbf{x}_l - \mathbf{Q}_k\|_1 \mid i_l = k\right]. \tag{34}$$

For any k such that $\mathbb{P}(i_l = k) > 0$, minimizing $\mathcal{L}_{\mathrm{MAE}}(\mathbf{Q})$ with respect to $\mathbf{Q}_k$ is equivalent to minimizing

$$\mathcal{L}_k^{\mathrm{MAE}}(\mathbf{Q}_k) := \mathbb{E}\left[\|\mathbf{x}_l - \mathbf{Q}_k\|_1 \mid i_l = k\right]. \tag{35}$$

Since the ℓ_1 norm is separable across variables, we have

$$\mathcal{L}_k^{\mathrm{MAE}}(\mathbf{Q}_k) = \sum_{n=1}^{N} \mathbb{E}\left[|x_{l,n} - \mathbf{Q}_{k,n}| \mid i_l = k\right]. \tag{36}$$

Thus, each variable can be optimized independently.

For a fixed phase k and variable n, let $q = \mathbf{Q}_{k,n}$ and define

$$\phi(q) = \mathbb{E}\left[|x_{l,n} - q| \mid i_l = k\right]. \tag{37}$$

A standard property of the absolute loss is that any minimizer of the expected absolute deviation is a median of the underlying distribution [47]. Therefore,

$$q^* \in \mathrm{Median}(x_{l,n} \mid i_l = k). \tag{38}$$

Applying this result to every variable n yields

$$\mathbf{Q}_{k,n}^* \in \mathrm{Median}(x_{l,n} \mid i_l = k), \quad \forall n \in \{1, \ldots, N\}. \tag{39}$$

This completes the proof. □

Proposition 3 shows that replacing the MSE objective with the MAE objective changes the optimal prototype from the phase-conditioned mean to the phase-conditioned median. However, phase-dependent deviations typically persist in the residual, since subtracting a phase-dependent prototype (such as the mean or median) primarily shifts the center of the distribution, without eliminating differences in variability or overall shape across phases. Consequently, the residual continues to reflect the underlying phase-dependent deviations.

C More Related Work

Deep Learning-based Time Series Forecasting Deep learning approaches have emerged as powerful tools for time series forecasting, driven by their ability to model complex nonlinear temporal dependencies [48]. These approaches can be broadly classified into RNN-based, TCN-based, GNN-based, Transformer-based, and Linear-based architectures. RNN-based models [49, 15] capture temporal dependencies by recursively updating hidden states, where each state encodes historical information up to the current time step. While effective for sequential modeling, they often suffer from difficulties in capturing long-term dependencies and limited parallelization. TCN-based models [43, 13] capture temporal dependencies using causal and dilated convolutions, providing large receptive fields while respecting temporal causality. They allow efficient parallel computation and stable training, but may struggle with very long-range dependencies and do not explicitly model temporal dynamics in a step-by-step manner. GNN-based models [50, 51] represent time series as nodes in a graph and capture dependencies by propagating information through graph structures, enabling the modeling of complex spatial–temporal relationships. They are effective for capturing structured and long-range interactions, but often rely on predefined or learned graph structures and may incur higher computational cost and sensitivity to graph quality. Transformer-based models [9, 8] capture temporal dependencies using attention mechanisms, such as self-attention [52] and cross-attention [53], enabling flexible modeling of interactions across embedding tokens. They typically exhibit strong expressiveness, but incur high computational cost as the lookback window length increases. Linear-based methods [11, 54, 38, 40] model time series using simple linear mappings, typically capturing temporal dependencies via linear projections of historical observations. They are computationally efficient and easy to train, often achieving strong performance in practice despite their simplicity. Recent studies have increasingly emphasized Transformer-based and linear-based approaches, both of which achieve state-of-the-art performance while exhibiting distinct characteristics. Transformer-based models are highly expressive, whereas linear-based methods provide superior computational efficiency. Paloma, proposed in this work, is a linear-based model that effectively captures phase-dependent residual deviations while maintaining the simplicity and parameter efficiency inherent to linear architectures.

D Experimental Details

D.1 Baselines

To comprehensively evaluate the effectiveness of the proposed Paloma for time series forecasting, we consider 17 representative and state-of-the-art methods as baselines. These methods span diverse architectural paradigms, including MLP-based, Transformer-based, frequency-based, and TCN-based approaches, along with two recently proposed training frameworks. A brief overview of these methods is provided as follows:

Global Temporal Retriever (GTR) [20] is a lightweight, plug-and-play module that addresses the limited historical context of existing multivariate time series forecasting models by enabling access to global temporal information across entire cycles. It adaptively retrieves and integrates relevant global patterns with local inputs, thereby enhancing forecasting performance without modifying the underlying model architecture. The detailed implementation is available at `https://github.com/macovaseas/GTR`. The performance of GTR reported in Table 1 (Table 6) and Table 8 follows the results presented in Table 7 and Table 11 of the original GTR paper.

RAFT [36] is a retrieval-augmented time series forecasting method that enhances model capacity by retrieving historically similar patterns from the training data and incorporating their future values into prediction. By providing external pattern-based information as an inductive bias, it reduces the burden on model parameters and consistently improves forecasting performance. The detailed implementation is available at `https://github.com/archon159/RAFT`. The performance of RAFT reported in Table 1 (Table 6) follows the results presented in Table 7 of the GTR [20] paper.

CycleNet [18] is a simple yet efficient time series forecasting method that models periodic patterns via Residual Cycle Forecasting (RCF) and predicts residuals using lightweight backbones, achieving state-of-the-art performance while remaining flexible for integration with existing models. The detailed implementation is available at `https://github.com/ACAT-SCUT/CycleNet`. The performance of CycleNet reported in Table 1 (Table 6) and Table 8 follows the results presented in Table 7 and Table 11 of the GTR [20] paper.

SparseTSF [16] is an extremely lightweight long-term time series forecasting model that leverages cross-period sparse forecasting to efficiently extract periodic patterns, achieving competitive performance with fewer than 1k parameters while maintaining strong generalization in resource-constrained scenarios. The detailed implementation is available at `https://github.com/lss-1138/SparseTSF`. The performance of SparseTSF reported in Table 8 follows the results presented in Table 11 of the GTR [20] paper.

DLinear [11] decomposes the input series into trend and seasonal components and applies separate linear layers to each, thereby improving performance by explicitly modeling trend dynamics. The detailed implementation is available at `https://github.com/cure-lab/LTSF-Linear`. The performance of DLinear reported in Table 1 (Table 6) follows the results presented in Table 7 of the GTR [20] paper.

OLinear [37] is a linear-based forecasting model that leverages an orthogonal transformation (OrthoTrans) to decorrelate temporal dependencies and a normalized linear layer (NormLin) to capture multivariate relationships, achieving state-of-the-art performance while also enhancing existing models as a plug-in. The detailed implementation is available at `https://github.com/jackyue1994/OLinear`. The performance of OLinear reported in Table 7 follows the results presented in Table 15 of the original OLinear paper.

TimeMixer [38] is a fully MLP-based forecasting model that captures multiscale temporal variations through decomposable mixing of seasonal and trend components and multi-predictor fusion, enabling effective modeling of complex time series with high efficiency. The detailed implementation is available at `https://github.com/kwuking/TimeMixer`. The performance of TimeMixer reported in Table 7 follows the results presented in Table 7 of the GTR [20] paper.

TimeXer [39] is a Transformer-based forecasting model that incorporates exogenous variables through patch-wise self-attention, variate-wise cross-attention, and global tokens to effectively model both temporal dependencies and cross-variable relationships for improved prediction of endogenous series. The detailed implementation is available at `https://github.com/thuml/TimeXer`. The

performance of TimeXer reported in Table 1 (Table 6) follows the results presented in Table 7 of the GTR [20] paper.

iTransformer [9] is a Transformer-based forecasting model that inverts input dimensions to treat variates as tokens, enabling effective modeling of multivariate correlations without modifying the original Transformer architecture. The detailed implementation is available at `https://github.com/thuml/iTransformer`. The performance of iTransformer reported in Table 1 (Table 6) follows the results presented in Table 7 of the GTR [20] paper.

PatchTST [8] employs a patch-based, channel-independent Transformer design to efficiently model local semantics and long-range dependencies in time series forecasting. The detailed implementation is available at `https://github.com/yuqinie98/patchtst`. The performance of PatchTST reported in Table 1 (Table 6) and Table 8 follows the results presented in Table 7 and Table 11 of the GTR [20] paper.

FilterNet [29] adopts learnable frequency filters to model time series by selectively passing or attenuating signal components, thereby effectively capturing informative patterns across the full frequency spectrum. The detailed implementation is available at `https://github.com/aikunyi/FilterNet`. The performance of FilterNet reported in Table 1 (Table 6) follows the results presented in Table 15 of the OLinear [37] paper.

FreTS [40] adopts frequency-domain MLPs to model time series by transforming signals into the frequency domain and learning real and imaginary components, enabling effective capture of global dependencies and compact signal patterns. The detailed implementation is available at `https://github.com/aikunyi/FreTS`. The performance of FreTS reported in Table 7 follows the results presented in Table 5 of the FreDF [45] paper, except for the Solar-Energy and four PEMS datasets, for which the results are obtained by running the official code five times and reporting the average values.

FITS [41] operates in the complex frequency domain to perform time series analysis via interpolation, achieving competitive performance with only about 10k parameters. The detailed implementation is available at `https://github.com/VEWOXIC/FITS`. The performance of FITS reported in Table 7 follows the results presented in Table 15 of the OLinear [37] paper.

Fredformer [42] addresses frequency bias in Transformer-based forecasting by promoting balanced learning across frequency bands, enabling more effective capture of both high- and low-frequency patterns in time series. The detailed implementation is available at `https://github.com/chenzRG/Fredformer`. The performance of Fredformer reported in Table 7 follows the results presented in Table 15 of the OLinear [37] paper.

TimesNet [43] models time series by transforming 1D sequences into 2D representations based on multi-periodicity, enabling effective capture of intra- and inter-period temporal variations through parameter-efficient convolutional blocks. The detailed implementation is available at `https://github.com/thuml/TimesNet`. The performance of TimesNet reported in Table 7 follows the results presented in Table 5 of the FreDF [45] paper, except for the Solar-Energy and four PEMS datasets, for which the results are obtained by running the official code five times and reporting the average values.

DistDF [44] trains forecasting models by minimizing a tractable joint-distribution Wasserstein discrepancy that upper bounds conditional distribution differences, thereby aligning forecast and label distributions without relying on biased likelihood estimation. The detailed implementation is available at `https://github.com/Master-PLC/DistDF`. The performance of DistDF reported in Table 1 (Table 6) follows the results presented in Table 6 of the original DistDF [44] paper, except for the Solar-Energy and four PEMS datasets, for which the results are obtained by running the official code five times and reporting the average values.

FreDF [45] mitigates the bias of the Direct Forecast paradigm caused by label autocorrelation by transforming forecasting into the frequency domain, enabling more accurate and unbiased time series prediction. The detailed implementation is available at `https://github.com/Master-PLC/FreDF`. The performance of FreDF reported in Table 1 (Table 6) follows the results presented in Table 5 of the original FreDF [45] paper, except for the Solar-Energy and four PEMS datasets, for which the results are obtained by running the official code five times and reporting the average values.

D.2 Datasets

To comprehensively evaluate the performance of our model, we conduct extensive experiments on eight benchmark datasets for long-term forecasting and the PEMS datasets for short-term forecasting. Detailed information of the datasets is provided in Table 5, and further descriptions are given below:

ETT (Electricity Transformer Temperature) dataset [46] records seven variables related to electricity transformers from July 2016 to July 2018. It consists of four subsets with two temporal granularities: ETTh1 and ETTh2 are sampled hourly, while ETTm1 and ETTm2 are sampled every 15 minutes.

Weather dataset [25] records 21 meteorological variables, including air temperature and humidity, collected at 10-minute intervals throughout 2020 from the Weather Station of the Max Planck Institute for Biogeochemistry in Germany.

Electricity dataset [25] records hourly electricity consumption (in kWh) for 321 clients, covering a multi-year period.

Traffic dataset [25] records hourly road occupancy rates collected by 862 sensors on the freeways of the San Francisco Bay Area, covering the period from 2015 to 2016.

Solar-Energy dataset [46] records solar power production from 137 photovoltaic plants in Alabama, sampled every 10 minutes in 2006.

PEMS dataset [14] provides public traffic data collected from the Caltrans Performance Measurement System (PeMS) in California, aggregated at 5-minute intervals. In this study, we use four subsets: PEMS03, PEMS04, PEMS07, and PEMS08.

Table 5: Detailed descriptions of the datasets. *Dim* indicates the number of variates in each dataset. *Prediction Length* refers to the number of future time steps to be predicted. *Dataset Size* represents the total number of time points in the (Train, Validation, Test) splits, respectively. *Frequency* indicates the sampling interval of the time series. *Cycle Length* denotes the length of the global cycle.

Dataset	Dim	Prediction Length	Dataset Size	Frequency	Cycle Length	Domain
ETTh1, ETTh2	7	{96, 192, 336, 720}	(8545, 2881, 2881)	Hourly	24	Electricity
ETTm1, ETTm2	7	{96, 192, 336, 720}	(34465, 11521, 11521)	15min	96	Electricity
Weather	21	{96, 192, 336, 720}	(36792, 5271, 10540)	10min	144	Weather
Electricity	321	{96, 192, 336, 720}	(18317, 2633, 5261)	Hourly	168	Electricity
Traffic	862	{96, 192, 336, 720}	(12185, 1757, 3509)	Hourly	168	Transportation
Solar-Energy	137	{96, 192, 336, 720}	(36601, 5161, 10417)	10min	144	Energy
PEMS03	358	{12, 24, 48, 96}	(15617, 5135, 5135)	5min	288	Transportation
PEMS04	307	{12, 24, 48, 96}	(10172, 3375, 3375)	5min	288	Transportation
PEMS07	883	{12, 24, 48, 96}	(16911, 5622, 5622)	5min	288	Transportation
PEMS08	170	{12, 24, 48, 96}	(10690, 3548, 3548)	5min	288	Transportation

D.3 Implementation Details

All experiments are implemented in PyTorch [55] and conducted on a single NVIDIA RTX 4090 GPU with 24GB memory. We use the Adam optimizer [56], with the learning rate selected from $\{5\text{e}{-4}, 1\text{e}{-3}, 2\text{e}{-3}, 3\text{e}{-3}, 5\text{e}{-3}\}$. Paloma is trained for up to 30 epochs, with early stopping based on validation performance using a patience of 5. Following prior works (e.g., CARD [57] and OLinear [37]), we adopt a weighted loss function. The hidden dimension of the linear projection, d_{model}, is chosen from $\{64, 128, 256, 512\}$, and the batch size is selected from $\{16, 64, 128, 256\}$.

E Additional Results

E.1 Full Forecasting Results

Lookback window $T = 96$. Table 6 provides the complete results corresponding to Table 1 in the main text. In addition, Table 7 (under the same experimental setting as Table 6) includes results from more representative baselines, enabling a more comprehensive comparison with the proposed Paloma. Table 7 also reports the standard deviation of Paloma over five runs with different random seeds. Overall, the results in Tables 6 and Table 7 demonstrate that Paloma achieves state-of-the-art performance while maintaining strong robustness.

Increasing Lookback Length. We further examine how model performance evolves as the lookback length increases, as shown in Table 8 and Figure 6. Paloma exhibits a generally improving and stable performance trend across datasets, indicating its ability to effectively leverage extended temporal context. In contrast, competing methods often show diminishing or inconsistent gains with longer input lengths, reflecting difficulty in reliably utilizing long-range historical information. This behavior highlights the advantage of explicitly modeling phase-dependent variations when handling longer input sequences.

E.2 Full Ablation Results

Table 9 provides the complete results corresponding to Table 2 in the main text. Table 9 presents a comprehensive ablation study of the proposed Phase Rotation (PR) and Amplitude Scaling (AS) modules under the long-term forecasting setting, including variants without PR ("w/o PR"), without AS ("w/o AS"), and without both modules ("w/o PR, AS"). The experiments are conducted on six representative datasets, including ETTm1, ETTh1, Electricity, Weather, Traffic, and Solar-Energy.

Table 6: Performance comparison with prediction lengths $\tau \in \{12, 24, 48, 96\}$ for PEMS and $\tau \in \{96, 192, 336, 720\}$ for others and fixed lookback window $T = 96$. Results are averaged over four prediction lengths. Highlight the best results in **red** and the second-best results in **blue**.

Models		**Paloma** [**ours**]		DistDF [ICLR'26]		GTR [ICLR'26]		RAFT [ICML'25]		FreDF [ICLR'25]		TimeXer [NeurIPS'24]		CycleNet [NeurIPS'24]		FilterNet [NeurIPS'24]		iTransformer [ICLR'24]		PatchTST [ICLR'23]		DLinear [AAAI'23]	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	**0.295**	**0.332**	0.316	0.357	**0.305**	**0.349**	0.325	0.369	0.324	0.362	0.318	0.356	0.319	0.360	0.321	0.361	0.334	0.368	0.329	0.367	0.345	0.372
	192	**0.343**	**0.358**	0.358	0.380	**0.349**	**0.375**	0.364	0.388	0.373	0.385	0.362	0.383	0.360	0.381	0.367	0.387	0.377	0.391	0.367	0.385	0.380	0.389
	336	**0.376**	**0.381**	0.392	0.404	**0.380**	**0.398**	0.391	0.407	0.402	0.404	0.395	0.407	0.389	0.403	0.401	0.409	0.426	0.420	0.399	0.410	0.413	0.413
	720	**0.439**	**0.417**	0.448	0.437	**0.435**	**0.436**	0.444	0.438	0.469	0.444	0.452	0.441	0.447	0.441	0.477	0.448	0.491	0.459	0.454	0.439	0.474	0.453
	Avg	**0.363**	**0.372**	0.378	0.394	**0.367**	**0.389**	0.381	0.400	0.392	0.399	0.382	0.397	0.379	0.396	0.392	0.401	0.407	0.410	0.387	0.400	0.403	0.407
ETTm2	96	**0.164**	**0.242**	0.174	0.256	0.168	0.249	0.176	0.264	0.173	0.252	0.171	0.256	**0.163**	**0.246**	0.175	0.258	0.180	0.264	0.175	0.259	0.193	0.292
	192	**0.229**	**0.284**	0.239	0.298	**0.232**	0.292	0.241	0.306	0.241	0.298	0.237	0.299	**0.229**	**0.290**	0.240	0.301	0.250	0.309	0.241	0.302	0.284	0.362
	336	0.289	**0.323**	0.300	0.338	**0.287**	0.330	0.302	0.345	0.298	0.334	0.296	0.338	**0.284**	**0.327**	0.311	0.347	0.311	0.348	0.305	0.343	0.369	0.427
	720	**0.387**	**0.383**	0.397	0.394	**0.386**	**0.390**	0.404	0.405	0.398	0.393	0.392	0.394	0.389	0.391	0.414	0.405	0.412	0.407	0.402	0.400	0.554	0.522
	Avg	**0.267**	**0.308**	0.277	0.321	0.268	0.315	0.281	0.330	0.278	0.319	0.274	0.322	**0.266**	**0.314**	0.285	0.328	0.288	0.332	0.281	0.326	0.350	0.401
ETTh1	96	**0.360**	**0.378**	0.373	0.393	**0.367**	**0.391**	0.376	0.401	0.382	0.400	0.382	0.403	0.375	0.395	0.382	0.402	0.386	0.405	0.414	0.419	0.386	0.400
	192	**0.414**	**0.409**	0.428	0.425	0.420	**0.421**	**0.417**	0.424	0.430	0.427	0.429	0.435	0.436	0.428	0.430	0.429	0.441	0.436	0.460	0.445	0.437	0.432
	336	**0.452**	**0.428**	0.466	0.445	0.476	0.447	**0.454**	**0.441**	0.474	0.451	0.468	0.448	0.496	0.455	0.472	0.451	0.487	0.458	0.501	0.466	0.481	0.459
	720	**0.458**	**0.449**	**0.453**	**0.453**	0.493	0.476	0.465	0.465	0.463	0.462	0.469	0.461	0.520	0.484	0.481	0.473	0.503	0.491	0.500	0.488	0.519	0.516
	Avg	**0.421**	**0.416**	0.430	**0.429**	0.439	0.434	**0.428**	0.433	0.437	0.435	0.437	0.437	0.457	0.441	0.441	0.439	0.454	0.448	0.469	0.455	0.456	0.452
ETTh2	96	**0.278**	**0.325**	0.287	**0.336**	0.290	0.342	0.289	0.342	0.289	0.337	**0.286**	0.338	0.298	0.344	0.293	0.343	0.297	0.349	0.302	0.348	0.333	0.387
	192	**0.353**	**0.376**	**0.358**	**0.381**	0.362	0.389	0.382	0.400	0.363	0.385	0.363	0.389	0.372	0.396	0.374	0.396	0.380	0.400	0.388	0.400	0.477	0.476
	336	**0.401**	**0.413**	**0.408**	0.421	0.414	0.429	0.424	0.439	0.419	0.426	0.414	**0.423**	0.431	0.439	0.417	0.430	0.428	0.432	0.426	0.433	0.594	0.541
	720	**0.412**	**0.429**	0.416	0.435	0.423	0.442	0.433	0.457	0.415	0.437	**0.408**	**0.432**	0.450	0.458	0.449	0.460	0.427	0.445	0.431	0.446	0.831	0.657
	Avg	**0.361**	**0.386**	**0.367**	**0.393**	0.372	0.400	0.382	0.410	0.371	0.396	0.368	0.396	0.388	0.409	0.383	0.407	0.383	0.407	0.387	0.407	0.559	0.515
Electricity	96	**0.134**	**0.222**	0.137	0.235	**0.134**	**0.229**	0.152	0.257	0.144	0.233	0.140	0.242	**0.136**	**0.229**	0.147	0.245	0.148	0.240	0.181	0.270	0.197	0.282
	192	**0.151**	**0.238**	0.159	0.257	**0.152**	0.245	0.156	0.257	0.159	0.247	0.157	0.256	**0.152**	**0.244**	0.160	0.250	0.162	0.253	0.188	0.274	0.196	0.285
	336	**0.170**	**0.256**	0.178	0.272	**0.171**	0.264	**0.171**	0.272	0.172	**0.263**	0.176	0.275	**0.170**	0.264	0.173	0.267	0.178	0.269	0.204	0.293	0.209	0.301
	720	**0.208**	**0.289**	0.212	0.302	**0.208**	0.300	0.221	0.300	**0.204**	**0.294**	0.211	0.306	0.212	0.299	0.210	0.309	0.225	0.317	0.246	0.324	0.245	0.333
	Avg	**0.166**	**0.251**	0.172	0.267	**0.166**	0.260	0.175	0.272	0.170	**0.259**	0.171	0.270	**0.168**	**0.259**	0.173	0.268	0.178	0.270	0.205	0.290	0.212	0.300
Traffic	96	0.425	**0.243**	**0.380**	**0.262**	0.440	0.263	**0.388**	0.265	0.391	0.265	0.428	0.271	0.458	0.296	0.430	0.294	0.395	0.268	0.462	0.290	0.650	0.396
	192	0.442	**0.254**	**0.407**	0.275	0.454	0.274	**0.400**	0.278	0.410	**0.273**	0.448	0.282	0.457	0.294	0.452	0.307	0.417	0.276	0.466	0.290	0.598	0.370
	336	0.462	**0.262**	0.429	0.284	0.472	0.282	**0.411**	0.287	**0.424**	**0.280**	0.473	0.289	0.470	0.299	0.470	0.316	0.433	0.283	0.482	0.300	0.605	0.373
	720	0.514	**0.279**	**0.452**	**0.297**	0.514	0.301	**0.456**	0.305	0.460	0.298	0.516	0.307	0.502	0.314	0.498	0.323	0.467	0.302	0.514	0.320	0.645	0.394
	Avg	0.461	**0.260**	**0.417**	**0.279**	0.470	0.280	**0.414**	0.284	0.421	**0.279**	0.466	0.287	0.472	0.301	0.463	0.310	0.428	0.282	0.481	0.300	0.625	0.383
Weather	96	**0.154**	**0.190**	0.164	0.209	**0.154**	**0.200**	0.185	0.241	0.164	0.202	**0.157**	0.205	0.158	0.203	0.162	0.207	0.174	0.214	0.177	0.210	0.196	0.255
	192	**0.195**	**0.238**	0.212	0.252	**0.202**	**0.244**	0.237	0.287	0.220	0.253	0.204	0.247	0.207	0.247	0.210	0.250	0.221	0.254	0.225	0.250	0.237	0.296
	336	**0.249**	**0.284**	0.270	0.295	**0.259**	**0.287**	0.290	0.327	0.275	0.294	0.261	0.290	0.262	0.289	0.265	0.290	0.278	0.296	0.278	0.290	0.283	0.335
	720	**0.324**	**0.336**	0.348	0.345	0.341	0.342	0.367	0.380	0.356	0.347	**0.340**	0.341	0.344	0.344	0.342	**0.340**	0.358	0.349	0.354	**0.340**	0.345	0.381
	Avg	**0.231**	**0.262**	0.248	0.275	**0.239**	**0.268**	0.270	0.309	0.254	0.274	0.241	0.271	0.243	0.271	0.245	0.272	0.258	0.278	0.259	0.273	0.265	0.317
Solar-Energy	96	**0.175**	**0.224**	0.185	0.237	**0.176**	**0.235**	0.278	0.278	0.188	0.240	0.215	0.295	0.190	0.247	0.205	0.242	0.203	0.237	0.234	0.286	0.290	0.378
	192	**0.192**	**0.240**	0.202	0.250	**0.193**	**0.244**	0.297	0.297	0.206	0.255	0.236	0.301	0.210	0.266	0.233	0.265	0.233	0.261	0.267	0.310	0.320	0.398
	336	**0.204**	**0.235**	0.212	0.265	**0.201**	**0.250**	0.311	0.311	0.217	0.272	0.252	0.307	0.217	0.266	0.249	0.278	0.248	0.273	0.290	0.315	0.353	0.415
	720	**0.215**	**0.238**	0.230	0.268	**0.205**	**0.251**	0.318	0.325	0.231	0.282	0.244	0.305	0.223	0.266	0.253	0.281	0.249	0.275	0.289	0.317	0.356	0.413
	Avg	**0.197**	**0.234**	0.207	0.255	**0.194**	**0.245**	0.301	0.303	0.211	0.262	0.237	0.302	0.210	0.261	0.235	0.266	0.233	0.262	0.270	0.307	0.330	0.401
PEMS03	12	**0.058**	**0.158**	0.068	0.174	**0.057**	**0.156**	0.071	0.175	0.068	0.172	0.070	0.173	0.066	0.172	0.071	0.177	0.071	0.174	0.099	0.216	0.122	0.243
	24	**0.074**	**0.175**	0.094	0.205	**0.070**	**0.172**	0.099	0.201	0.096	0.205	0.092	0.194	0.089	0.201	0.102	0.213	0.093	0.201	0.142	0.259	0.201	0.317
	48	**0.101**	**0.205**	0.138	0.252	**0.094**	**0.198**	0.153	0.243	0.128	0.240	0.129	0.229	0.136	0.247	0.162	0.272	0.125	0.236	0.211	0.319	0.333	0.425
	96	**0.142**	**0.236**	0.191	0.301	**0.127**	**0.228**	0.253	0.301	0.161	0.269	0.157	0.261	0.182	0.282	0.244	0.340	0.164	0.275	0.269	0.370	0.457	0.515
	Avg	**0.094**	**0.194**	0.123	0.233	**0.087**	**0.189**	0.144	0.230	0.113	0.219	0.112	0.214	0.118	0.226	0.145	0.251	0.113	0.222	0.180	0.291	0.278	0.375
PEMS04	12	**0.065**	**0.160**	**0.071**	0.167	**0.065**	**0.164**	0.077	0.179	0.073	0.174	0.074	0.178	0.078	0.186	0.082	0.190	0.078	0.183	0.105	0.224	0.148	0.272
	24	**0.074**	**0.171**	0.080	0.181	**0.075**	**0.177**	0.091	0.195	0.085	0.186	0.087	0.195	0.099	0.212	0.110	0.224	0.095	0.205	0.153	0.275	0.224	0.340
	48	**0.090**	**0.189**	0.097	0.205	**0.093**	**0.197**	0.111	0.218	0.101	0.212	0.110	0.214	0.133	0.248	0.160	0.276	0.120	0.233	0.229	0.339	0.355	0.437
	96	**0.110**	**0.207**	0.120	0.228	**0.114**	**0.219**	0.137	0.247	0.125	0.235	0.148	0.251	0.167	0.281	0.234	0.343	0.150	0.262	0.291	0.389	0.452	0.504
	Avg	**0.085**	**0.182**	0.092	0.195	**0.087**	**0.189**	0.104	0.210	0.096	0.202	0.105	0.209	0.119	0.232	0.146	0.258	0.111	0.221	0.195	0.307	0.295	0.388
PEMS07	12	**0.051**	**0.138**	**0.056**	0.149	**0.051**	**0.142**	0.062	0.159	0.060	0.156	0.057	0.152	0.062	0.162	0.064	0.163	0.067	0.165	0.095	0.207	0.115	0.242
	24	**0.064**	**0.152**	0.073	0.168	**0.063**	**0.156**	0.078	0.178	0.077	0.170	0.079	0.179	0.086	0.192	0.093	0.200	0.088	0.190	0.150	0.262	0.210	0.329
	48	**0.084**	**0.172**	0.089	0.183	**0.082**	**0.178**	0.102	0.203	0.093	0.188	0.099	0.191	0.128	0.234	0.137	0.248	0.110	0.215	0.253	0.340	0.398	0.458
	96	**0.112**	**0.194**	0.119	0.209	**0.110**	**0.202**	0.135	0.231	0.124	0.215	0.107	0.205	0.176	0.268	0.198	0.306	0.139	0.245	0.346	0.404	0.594	0.553
	Avg	**0.078**	**0.164**	0.084	0.177	**0.076**	**0.169**	0.094	0.193	0.089	0.182	0.085	0.182	0.113	0.214	0.123	0.229	0.101	0.204	0.211	0.303	0.329	0.396
PEMS08	12	**0.069**	**0.163**	0.076	0.177	**0.071**	**0.168**	0.078	0.181	0.080	0.182	0.075	0.176	0.082	0.185	0.080	0.182	0.079	0.182	0.168	0.232	0.154	0.276
	24	**0.086**	**0.181**	0.107	0.210	**0.095**	**0.192**	0.102	0.205	0.118	0.220	0.102	0.201	0.117	0.226	0.114	0.219	0.115	0.219	0.224	0.281	0.248	0.353
	48	**0.118**	**0.203**	0.171	0.265	**0.148**	**0.234**	**0.148**	0.244	0.206	0.293	0.158	0.248	0.169	0.268	0.184	0.284	0.186	0.235	0.321	0.354	0.440	0.470
	96	**0.171**	**0.229**	0.250	0.325	0.256	0.293	0.277	0.307	0.289	0.325	0.366	0.377	0.233	0.306	0.309	0.356	**0.221**	**0.267**	0.408	0.417	0.674	0.565
	Avg	**0.111**	**0.194**	0.151	0.244	**0.142**	**0.222**	0.151	0.234	0.173	0.255	0.175	0.250	0.150	0.246	0.172	0.260	0.150	0.226	0.280	0.321	0.379	0.416
1st Count		35	55	3	0	19	5	3	0	1	0	1	0	5	0	0	0	0	0	0	0	0	0

Table 7: Additional performance comparison with a fixed lookback window $T = 96$. The prediction lengths are set as $\tau \in \{12, 24, 48, 96\}$ for PEMS datasets for short-term forecasting, and $\tau \in \{96, 192, 336, 720\}$ for the others for long-term forecasting. The best and second-best results are highlighted in **red** and **blue**, respectively. The reported results with standard deviation of Paloma are averaged from 5 runs (with different random seeds of 2024, 2025, 2026, 2027, 2028).

Models		**Paloma** [**ours**]		OLinear [NeurIPS'25]		TimeMixer [ICLR'24]		FITS [ICLR'24]		Fredformer [KDD'24]		FreTS [NeurIPS'23]		TimesNet [NeurIPS'23]	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	**0.295** ± 0.001	**0.332** ± 0.001	**0.302**	**0.334**	0.320	0.357	0.353	0.375	0.326	0.361	0.339	0.374	0.338	0.379
	192	**0.343** ± 0.001	**0.358** ± 0.001	**0.357**	**0.363**	0.361	0.381	0.486	0.445	0.363	0.380	0.382	0.397	0.389	0.400
	336	**0.376** ± 0.001	**0.381** ± 0.001	**0.387**	**0.385**	0.390	0.404	0.531	0.475	0.395	0.403	0.421	0.426	0.429	0.428
	720	**0.439** ± 0.001	**0.417** ± 0.002	**0.452**	**0.426**	0.454	0.441	0.600	0.513	0.453	0.438	0.485	0.462	0.495	0.464
	Avg	**0.363** ± 0.001	**0.372** ± 0.001	**0.374**	**0.377**	0.381	0.395	0.493	0.452	0.384	0.395	0.407	0.415	0.413	0.418
ETTm2	96	**0.164** ± 0.001	**0.242** ± 0.001	**0.169**	**0.249**	0.175	0.258	0.182	0.266	0.177	0.259	0.190	0.282	0.185	0.264
	192	**0.229** ± 0.002	**0.284** ± 0.001	**0.232**	**0.290**	0.237	0.299	0.253	0.312	0.243	0.301	0.260	0.329	0.254	0.307
	336	**0.289** ± 0.001	**0.323** ± 0.003	**0.291**	**0.328**	0.298	0.340	0.313	0.349	0.302	0.340	0.373	0.405	0.314	0.345
	720	**0.387** ± 0.001	**0.383** ± 0.001	**0.389**	**0.387**	0.391	0.396	0.416	0.406	0.397	0.396	0.517	0.499	0.434	0.413
	Avg	**0.267** ± 0.001	**0.308** ± 0.001	**0.270**	**0.313**	0.275	0.323	0.291	0.333	0.279	0.324	0.335	0.379	0.297	0.332
ETTh1	96	**0.360** ± 0.003	**0.378** ± 0.002	**0.360**	**0.382**	0.375	0.400	0.385	0.394	**0.373**	0.392	0.399	0.412	0.422	0.433
	192	**0.414** ± 0.004	**0.409** ± 0.003	**0.416**	**0.414**	0.429	0.421	0.434	0.422	0.433	0.420	0.453	0.443	0.465	0.457
	336	**0.452** ± 0.002	**0.428** ± 0.005	**0.457**	**0.438**	0.484	0.458	0.476	0.444	0.470	0.437	0.503	0.475	0.492	0.470
	720	**0.458** ± 0.006	**0.449** ± 0.009	**0.463**	0.462	0.498	0.482	0.465	0.462	0.467	**0.456**	0.596	0.565	0.532	0.502
	Avg	**0.421** ± 0.004	**0.416** ± 0.005	**0.424**	**0.424**	0.447	0.440	0.440	0.431	0.435	0.426	0.488	0.474	0.478	0.466
ETTh2	96	**0.278** ± 0.002	**0.325** ± 0.003	**0.284**	**0.329**	0.289	0.341	0.292	0.340	0.293	0.342	0.350	0.403	0.320	0.364
	192	**0.353** ± 0.001	**0.376** ± 0.002	**0.360**	**0.379**	0.372	0.392	0.377	0.391	0.371	0.389	0.472	0.475	0.409	0.417
	336	0.401 ± 0.003	**0.413** ± 0.002	0.409	0.415	**0.386**	0.414	0.416	0.425	**0.382**	**0.409**	0.564	0.528	0.449	0.451
	720	**0.412** ± 0.003	**0.429** ± 0.004	**0.415**	**0.431**	**0.412**	0.434	0.418	0.437	**0.415**	0.434	0.815	0.654	0.473	0.474
	Avg	**0.361** ± 0.002	**0.386** ± 0.003	0.367	**0.388**	**0.365**	0.395	0.376	0.398	**0.365**	0.393	0.550	0.515	0.413	0.426
Electricity	96	**0.134** ± 0.001	**0.222** ± 0.001	**0.131**	**0.221**	0.153	0.247	0.198	0.274	0.147	0.241	0.189	0.277	0.171	0.273
	192	**0.151** ± 0.001	**0.238** ± 0.001	**0.150**	**0.238**	0.166	**0.256**	0.363	0.422	0.165	0.258	0.193	0.282	0.188	0.289
	336	**0.170** ± 0.001	**0.256** ± 0.001	**0.165**	**0.254**	0.185	0.277	0.444	0.490	0.177	0.273	0.207	0.296	0.208	0.304
	720	**0.208** ± 0.001	**0.289** ± 0.002	**0.191**	**0.279**	0.225	0.310	0.532	0.551	0.213	0.304	0.245	0.332	0.289	0.363
	Avg	**0.166** ± 0.001	**0.251** ± 0.001	**0.159**	**0.248**	0.182	0.273	0.384	0.434	0.176	0.269	0.209	0.297	0.214	0.307
Traffic	96	0.425 ± 0.001	**0.243** ± 0.001	**0.398**	**0.226**	0.462	0.285	0.601	0.361	**0.406**	0.277	0.528	0.341	0.504	0.298
	192	0.442 ± 0.002	**0.254** ± 0.001	**0.439**	**0.241**	0.473	0.296	0.603	0.365	**0.426**	0.290	0.531	0.338	0.526	0.305
	336	**0.462** ± 0.001	**0.262** ± 0.001	0.464	**0.250**	0.498	0.296	0.609	0.366	**0.437**	0.292	0.551	0.345	0.540	0.310
	720	0.514 ± 0.000	**0.279** ± 0.001	**0.502**	**0.270**	0.506	0.313	0.648	0.387	**0.462**	0.305	0.598	0.367	0.570	0.324
	Avg	0.461 ± 0.001	**0.260** ± 0.001	**0.451**	**0.247**	0.485	0.298	0.615	0.370	**0.433**	0.291	0.552	0.348	0.535	0.309
Weather	96	**0.154** ± 0.001	**0.190** ± 0.001	**0.153**	**0.190**	0.163	0.209	0.196	0.236	0.163	**0.207**	0.184	0.239	0.178	0.226
	192	**0.195** ± 0.001	**0.238** ± 0.001	**0.200**	**0.235**	0.208	0.250	0.240	0.271	0.211	0.251	0.223	0.275	0.227	0.266
	336	**0.249** ± 0.001	**0.284** ± 0.001	0.258	**0.280**	**0.251**	0.287	0.292	0.307	0.267	0.292	0.272	0.316	0.283	0.305
	720	**0.324** ± 0.001	**0.336** ± 0.001	**0.337**	**0.333**	0.339	0.341	0.365	0.354	0.343	0.341	0.340	0.363	0.359	0.355
	Avg	**0.231** ± 0.001	**0.262** ± 0.001	**0.237**	**0.260**	0.240	0.272	0.273	0.292	0.246	0.272	0.255	0.299	0.262	0.288
Solar-Energy	96	**0.175** ± 0.003	**0.224** ± 0.002	**0.179**	**0.191**	0.189	0.259	0.319	0.353	0.185	0.233	0.237	0.300	0.250	0.292
	192	**0.192** ± 0.002	**0.240** ± 0.002	**0.209**	**0.213**	0.222	0.283	0.367	0.387	0.227	0.253	0.265	0.321	0.296	0.318
	336	**0.204** ± 0.005	**0.235** ± 0.008	**0.231**	**0.229**	**0.231**	0.292	0.408	0.403	0.246	0.284	0.283	0.330	0.319	0.330
	720	**0.215** ± 0.006	**0.238** ± 0.004	**0.241**	**0.236**	0.223	0.285	0.411	0.395	0.247	0.276	0.286	0.326	0.338	0.337
	Avg	**0.197** ± 0.004	**0.234** ± 0.004	**0.215**	**0.217**	0.216	0.280	0.376	0.384	0.226	0.262	0.268	0.322	0.301	0.319
PEMS03	12	**0.058** ± 0.001	**0.158** ± 0.001	**0.060**	**0.159**	0.076	0.188	0.117	0.226	0.068	0.174	0.083	0.194	0.085	0.192
	24	**0.074** ± 0.001	**0.175** ± 0.001	**0.078**	**0.179**	0.113	0.226	0.235	0.324	0.093	0.202	0.127	0.241	0.118	0.223
	48	**0.101** ± 0.002	**0.205** ± 0.001	**0.104**	**0.210**	0.191	0.292	0.541	0.521	0.146	0.258	0.204	0.311	0.155	0.260
	96	**0.142** ± 0.001	**0.236** ± 0.003	**0.140**	**0.247**	0.288	0.363	1.062	0.790	0.228	0.330	0.255	0.345	0.228	0.317
	Avg	**0.094** ± 0.001	**0.194** ± 0.002	**0.095**	**0.199**	0.167	0.267	0.489	0.465	0.135	0.243	0.167	0.273	0.147	0.248
PEMS04	12	**0.065** ± 0.001	**0.160** ± 0.001	**0.068**	**0.163**	0.092	0.204	0.129	0.239	0.085	0.189	0.098	0.210	0.087	0.195
	24	**0.074** ± 0.001	**0.171** ± 0.001	**0.079**	**0.176**	0.128	0.243	0.246	0.337	0.117	0.224	0.148	0.263	0.103	0.215
	48	**0.090** ± 0.001	**0.189** ± 0.001	**0.095**	**0.197**	0.213	0.315	0.568	0.539	0.174	0.276	0.235	0.341	0.136	0.250
	96	**0.110** ± 0.001	**0.207** ± 0.001	**0.122**	**0.226**	0.307	0.384	1.181	0.843	0.273	0.354	0.304	0.393	0.190	0.303
	Avg	**0.085** ± 0.001	**0.182** ± 0.001	**0.091**	**0.190**	0.185	0.287	0.531	0.489	0.162	0.261	0.196	0.302	0.129	0.241
PEMS07	12	**0.051** ± 0.002	**0.138** ± 0.001	**0.052**	**0.138**	0.073	0.184	0.109	0.222	0.063	**0.158**	0.080	0.190	0.082	0.181
	24	**0.064** ± 0.002	**0.152** ± 0.002	**0.065**	**0.151**	0.111	0.219	0.230	0.327	0.089	0.192	0.136	0.250	0.101	0.204
	48	**0.084** ± 0.001	**0.172** ± 0.001	**0.084**	**0.171**	0.237	0.328	0.551	0.531	0.136	0.241	0.248	0.340	**0.134**	0.238
	96	**0.112** ± 0.001	**0.194** ± 0.001	**0.108**	**0.196**	0.303	0.354	1.112	0.809	0.197	0.298	0.352	0.413	0.181	0.279
	Avg	**0.078** ± 0.002	**0.164** ± 0.001	**0.077**	**0.164**	0.181	0.271	0.500	0.472	0.121	**0.222**	0.204	0.298	0.124	0.225
PEMS08	12	**0.069** ± 0.000	**0.163** ± 0.000	**0.068**	**0.159**	0.091	0.201	0.122	0.233	0.081	0.185	0.095	0.204	0.112	0.212
	24	**0.086** ± 0.001	**0.181** ± 0.001	**0.089**	**0.178**	0.137	0.246	0.236	0.330	0.112	0.214	0.150	0.259	0.141	0.238
	48	**0.118** ± 0.000	**0.203** ± 0.001	**0.123**	**0.204**	0.265	0.343	0.562	0.540	0.174	0.267	0.250	0.341	0.198	0.283
	96	**0.171** ± 0.001	**0.229** ± 0.001	**0.173**	**0.236**	0.410	0.407	1.216	0.846	0.277	0.335	0.323	0.376	0.320	0.351
	Avg	**0.111** ± 0.001	**0.194** ± 0.001	**0.113**	**0.194**	0.226	0.299	0.534	0.487	0.161	**0.250**	0.205	0.295	0.193	0.271
1st Count		45	37	12	27	1	0	0	0	5	1	0	0	0	0

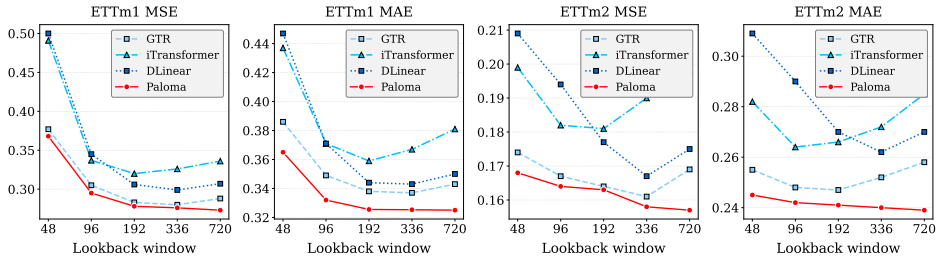


Figure 6: Forecasting performance with the lookback window $T \in \{48, 96, 192, 336, 720\}$ and fixed prediction length $\tau = 96$.

Table 8: Performance comparison with longer lookback windows $T \in \{336, 720\}$. Highlight the best results in **red** and the second-best results in **blue**. The baseline results follow Table 11 in GTR [20].

LookBack		$T = 336$								$T = 720$							
Models		PatchTST [ICLR'23]		CycleNet [NeurIPS'24]		GTR [ICLR'26]		**Paloma [ours]**		SparseTSF [ICML'25]		CycleNet [NeurIPS'24]		GTR [ICLR'26]		**Paloma [ours]**	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.385	0.405	0.374	0.396	**0.369**	**0.395**	**0.355**	**0.380**	**0.362**	**0.388**	0.379	0.403	0.368	0.400	**0.359**	**0.389**
	192	0.414	0.421	**0.406**	**0.415**	0.409	0.420	**0.399**	**0.408**	**0.403**	**0.411**	0.416	0.425	0.409	0.424	**0.406**	**0.418**
	336	0.440	0.440	**0.431**	**0.430**	0.432	0.434	**0.428**	**0.425**	**0.434**	**0.428**	0.447	0.445	0.443	0.449	**0.439**	**0.438**
	720	0.456	0.470	**0.450**	0.464	0.451	**0.463**	**0.434**	**0.449**	**0.426**	**0.447**	0.477	0.483	0.464	0.476	**0.442**	**0.460**
	Avg	0.424	0.434	**0.415**	**0.426**	**0.415**	0.428	**0.404**	**0.416**	**0.406**	**0.419**	0.430	0.439	0.421	0.437	**0.412**	**0.426**
ETTh2	96	0.275	**0.337**	0.279	0.341	**0.274**	0.341	**0.273**	**0.331**	0.294	0.346	**0.271**	**0.337**	0.275	0.343	**0.269**	**0.328**
	192	**0.338**	**0.379**	0.342	0.385	**0.340**	0.385	**0.340**	**0.374**	0.339	**0.377**	**0.332**	0.380	0.340	0.388	**0.336**	**0.372**
	336	**0.365**	**0.398**	**0.371**	0.413	0.363	0.409	0.374	**0.404**	**0.359**	**0.397**	**0.362**	0.408	0.377	0.416	0.367	**0.398**
	720	**0.391**	**0.429**	0.426	0.451	0.405	0.439	**0.397**	**0.427**	**0.383**	**0.424**	0.415	0.449	0.406	0.442	**0.386**	**0.420**
	Avg	**0.342**	**0.386**	0.355	0.398	**0.345**	0.393	0.346	**0.384**	**0.344**	**0.386**	0.345	0.394	0.349	0.397	**0.340**	**0.380**
ETTm1	96	0.291	0.343	0.299	0.348	**0.283**	**0.337**	**0.276**	**0.325**	0.312	0.354	0.307	0.353	**0.290**	**0.345**	**0.273**	**0.325**
	192	0.334	0.370	0.334	0.367	**0.323**	**0.365**	**0.313**	**0.349**	0.347	0.376	0.337	**0.371**	**0.328**	**0.371**	**0.313**	**0.352**
	336	0.367	0.392	0.368	0.386	**0.355**	**0.384**	**0.345**	**0.370**	0.367	**0.386**	0.364	0.387	**0.357**	0.388	**0.344**	**0.374**
	720	0.422	0.426	**0.417**	**0.414**	0.420	0.422	**0.404**	**0.403**	0.419	0.413	**0.410**	**0.411**	0.417	0.420	**0.407**	**0.406**
	Avg	0.354	0.383	0.355	0.379	**0.345**	**0.377**	**0.335**	**0.362**	0.361	0.382	0.355	**0.381**	**0.348**	**0.381**	**0.334**	**0.364**
ETTm2	96	0.164	0.254	**0.159**	**0.247**	0.161	0.252	**0.158**	**0.240**	0.163	0.252	**0.159**	**0.249**	0.169	0.258	**0.157**	**0.239**
	192	0.221	0.293	**0.214**	**0.286**	0.221	0.292	**0.216**	**0.281**	**0.217**	0.290	**0.214**	**0.289**	0.220	0.295	**0.214**	**0.282**
	336	0.276	0.328	**0.269**	**0.322**	0.278	0.326	**0.270**	**0.316**	0.270	0.327	**0.268**	**0.326**	0.270	0.328	**0.263**	**0.316**
	720	0.366	0.383	0.363	**0.382**	**0.359**	0.388	**0.356**	**0.372**	**0.352**	**0.379**	**0.353**	0.384	0.355	0.389	**0.352**	**0.375**
	Avg	0.257	0.315	**0.251**	**0.309**	0.254	0.314	**0.250**	**0.303**	0.251	**0.312**	**0.249**	**0.312**	0.253	0.317	**0.247**	**0.304**
Electricity	96	0.131	0.225	**0.128**	0.223	**0.127**	**0.222**	**0.127**	**0.216**	**0.138**	0.233	**0.128**	**0.223**	**0.128**	0.226	**0.128**	**0.220**
	192	0.148	0.240	**0.144**	**0.237**	**0.147**	0.242	**0.147**	**0.234**	0.151	**0.244**	**0.143**	**0.237**	**0.146**	**0.244**	0.148	**0.237**
	336	**0.165**	0.259	**0.160**	**0.254**	**0.165**	0.260	0.166	**0.252**	0.166	**0.260**	**0.159**	**0.254**	**0.162**	0.261	0.163	**0.254**
	720	0.202	0.291	**0.198**	**0.287**	**0.199**	0.292	0.202	**0.284**	0.205	0.293	**0.197**	**0.287**	**0.197**	0.295	**0.199**	**0.285**
	Avg	0.162	0.254	**0.158**	**0.250**	**0.159**	0.254	0.161	**0.247**	0.165	0.258	**0.157**	**0.250**	**0.158**	0.256	0.160	**0.249**
Solar-Energy	96	**0.190**	0.278	0.200	0.250	0.180	**0.237**	**0.174**	**0.214**	0.195	0.243	0.194	0.255	**0.178**	**0.242**	**0.175**	**0.206**
	192	0.206	**0.252**	0.221	0.261	**0.196**	0.254	**0.190**	**0.230**	0.215	0.254	0.205	**0.251**	**0.198**	0.260	**0.188**	**0.216**
	336	0.217	**0.254**	0.236	0.272	**0.201**	0.257	**0.198**	**0.225**	0.232	0.262	0.218	**0.257**	**0.202**	0.264	**0.201**	**0.225**
	720	0.219	**0.255**	0.245	0.277	**0.205**	0.258	**0.208**	**0.221**	0.237	**0.263**	0.239	0.278	**0.209**	0.269	**0.207**	**0.226**
	Avg	0.208	0.260	0.226	0.265	**0.196**	**0.252**	**0.193**	**0.223**	0.220	**0.256**	0.214	0.260	**0.197**	0.259	**0.193**	**0.218**
Weather	96	0.155	0.204	0.167	0.221	**0.147**	**0.199**	**0.146**	**0.185**	0.169	0.223	0.164	0.220	**0.146**	**0.200**	**0.144**	**0.189**
	192	0.195	0.242	0.212	0.258	**0.192**	**0.240**	**0.188**	**0.233**	0.214	0.262	0.209	0.258	**0.192**	**0.244**	**0.190**	**0.236**
	336	0.249	0.283	0.260	0.293	**0.243**	**0.281**	**0.237**	**0.276**	0.257	0.293	0.255	0.292	**0.243**	**0.283**	**0.238**	**0.275**
	720	**0.321**	**0.334**	0.328	0.339	0.324	0.337	**0.313**	**0.329**	0.321	0.340	0.320	0.338	**0.317**	**0.336**	**0.302**	**0.327**
	Avg	0.230	0.266	0.242	0.278	**0.227**	**0.264**	**0.221**	**0.256**	0.240	0.280	0.237	0.277	**0.225**	**0.266**	**0.219**	**0.257**
1st Count		4	1	6	0	2	0	24	34	7	6	7	2	2	0	23	28

Among them, ETTm1 and ETTh1 are selected as representative subsets from the four ETT datasets, as they cover the two temporal granularities (ETTm and ETTh). The ablation results in Table 9 show that removing either the PR or AS module consistently degrades performance across all datasets, indicating that both components contribute positively to the model. Notably, removing the PR module leads to slightly larger performance drops compared to removing the AS module, suggesting that phase modeling plays a more dominant role in capturing temporal dynamics. When both modules are removed, the performance further deteriorates, demonstrating that PR and AS provide complementary benefits. Overall, these results validate the effectiveness of the proposed phase–amplitude modulation mechanism in modeling structured residual variations.

Table 9: Full ablation study of the Phase Rotation (PR) and Amplitude Scaling (AS) modules on six datasets. The lookback window is fixed at 96, and results are averaged over four prediction lengths.

Design		ETTm1		ETTh1		Electricity		Weather		Traffic		Solar-Energy	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Paloma	96	0.295	0.332	0.360	0.378	0.134	0.222	0.154	0.190	0.425	0.243	0.175	0.224
	192	0.343	0.358	0.414	0.409	0.151	0.238	0.195	0.238	0.442	0.254	0.192	0.240
	336	0.376	0.381	0.452	0.428	0.170	0.256	0.249	0.284	0.462	0.262	0.204	0.235
	720	0.439	0.417	0.458	0.449	0.208	0.289	0.324	0.336	0.514	0.279	0.215	0.238
	Avg	0.363	0.372	0.421	0.416	0.166	0.251	0.231	0.262	0.461	0.260	0.197	0.234
w/o PR	96	0.300	0.335	0.365	0.383	0.140	0.231	0.160	0.197	0.434	0.250	0.179	0.232
	192	0.346	0.360	0.420	0.414	0.156	0.245	0.203	0.246	0.452	0.262	0.205	0.252
	336	0.382	0.385	0.458	0.432	0.172	0.261	0.256	0.290	0.474	0.269	0.220	0.258
	720	0.445	0.417	0.464	0.451	0.211	0.294	0.332	0.345	0.522	0.288	0.221	0.244
	Avg	0.368	0.374	0.427	0.420	0.170	0.258	0.238	0.270	0.471	0.267	0.206	0.247
w/o AS	96	0.300	0.336	0.366	0.381	0.139	0.230	0.163	0.202	0.430	0.249	0.178	0.231
	192	0.348	0.359	0.423	0.420	0.154	0.243	0.205	0.245	0.450	0.261	0.198	0.246
	336	0.378	0.384	0.465	0.440	0.174	0.264	0.254	0.291	0.471	0.275	0.207	0.251
	720	0.448	0.418	0.464	0.454	0.214	0.297	0.330	0.342	0.522	0.290	0.219	0.242
	Avg	0.369	0.374	0.430	0.424	0.170	0.259	0.238	0.270	0.468	0.269	0.201	0.243
w/o PR,AS	96	0.314	0.346	0.368	0.382	0.142	0.232	0.166	0.207	0.443	0.264	0.185	0.237
	192	0.351	0.362	0.420	0.421	0.159	0.248	0.213	0.259	0.453	0.281	0.202	0.249
	336	0.385	0.388	0.461	0.441	0.178	0.266	0.259	0.300	0.473	0.288	0.225	0.261
	720	0.454	0.423	0.463	0.455	0.218	0.301	0.335	0.353	0.526	0.298	0.225	0.244
	Avg	0.376	0.380	0.428	0.425	0.174	0.262	0.243	0.280	0.474	0.283	0.209	0.248

E.3 More Efficiency Analysis

To comprehensively evaluate the efficiency of the proposed model Paloma, we conduct three efficiency experiments on datasets with varying numbers of channels, including ETTh1 (7 channels, Table 10), Weather (21 channels, Table 4), and Traffic (862 channels, Table 11) datasets, covering small-, medium-, and large-scale scenarios. All experiments are performed under a unified setting with a lookback window $T = 96$ and a prediction length $\tau = 720$. The batch size is fixed at 32 for ETTh1 and Weather datasets, and set to 16 for Traffic dataset due to its significantly larger scale.

The efficiency results across Tables 4, 10, and 11 demonstrate that Paloma maintains a consistent balance between computational cost and predictive performance across datasets of varying scales. On the small-scale ETTh1 dataset (7 channels), Paloma achieves superior accuracy with comparable training time and moderate computational overhead. On the medium-scale Weather dataset (21 channels), Paloma continues to deliver better forecasting performance while requiring substantially fewer MACs and memory than Transformer-based models. On the large-scale Traffic dataset (862 channels), although the overall computational cost increases for all methods, Paloma remains significantly more efficient than Transformer-based approaches, with markedly lower MACs and training time, while incurring only a moderate increase in computational cost compared to lightweight linear models such as GTR, CycleNet, and DLinear. These results indicate that Paloma maintains a favorable balance between efficiency and accuracy as the number of channels increases, demonstrating strong scalability to high-dimensional time series.

Table 10: Efficiency comparison between Paloma and other models on the ETTh1 dataset with lookback window T = 96 and forecast horizon τ = 720. Training Time denotes the average time required per epoch for the model.

Model	Parameters	MACs	Training Time	Memory Usage	MSE	MAE
GTR	350.99K	2.5M	2.4s	1738MB	0.493	0.476
CycleNet	210.04K	1.47M	2.0s	1400MB	0.520	0.484
DLinear	139.68K	979.44K	1.8s	1398MB	0.519	0.516
FilterNet	369.37K	2.4M	2.6s	1412MB	0.481	0.473
iTransformer	737.49K	5.16M	2.7s	2016MB	0.503	0.491
PatchTST	2.75M	60.74M	2.9s	1600MB	0.500	0.488
TimeXer	2.11M	37.04M	4.3s	2046MB	0.469	0.461
Paloma(ours)	591.4K	4.07M	2.5s	1406MB	0.458	0.449

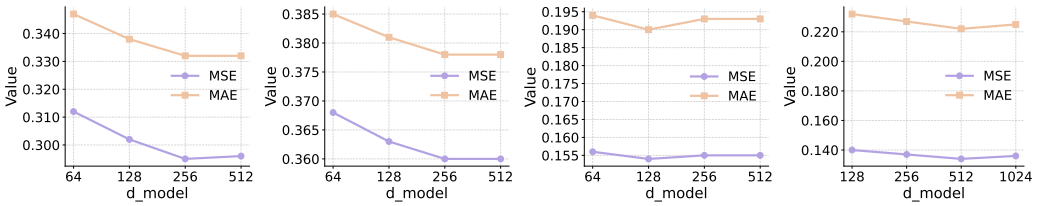
Table 11: Efficiency comparison between Paloma and other models on the Traffic dataset with lookback window T = 96 and forecast horizon τ = 720. Training Time denotes the average time required per epoch for the model.

Model	Parameters	MACs	Training Time	Memory Usage	MSE	MAE
GTR	1.1M	827.95M	30.3s	2336MB	0.514	0.301
CycleNet	563.84K	362.08M	25.5s	1764MB	0.502	0.314
DLinear	139.7K	120.61M	22.6s	1696MB	0.645	0.394
FilterNet	502.41K	408.64M	28.5s	2078MB	0.498	0.323
iTransformer	6.72M	5.79G	42.3s	5860MB	0.467	0.302
PatchTST	10.74M	69.46G	205.8s	16614MB	0.514	0.320
TimeXer	11.49M	42.16G	257.9s	13644MB	0.516	0.307
Paloma(ours)	2.0M	1.34G	29.9s	2422MB	0.514	0.279

E.4 Parameter Sensitivity Analysis

E.4.1 Hidden Dimension d_{model}

We analyze the sensitivity of Paloma to the hidden dimension d_{model}, as shown in Figure 7. As d_{model} increases from small to moderate values, the performance consistently improves, indicating that a larger representation capacity helps capture richer temporal patterns. However, when d_{model} becomes sufficiently large, the performance tends to saturate, with only marginal gains or slight fluctuations observed. Notably, this trend is consistent across all datasets, indicating that while performance improves with increasing d_{model} at smaller scales, it becomes stable beyond a moderate range. This suggests that Paloma is not highly sensitive to the choice of d_{model} once sufficient model capacity is reached, and can achieve strong performance without requiring excessively large dimensions.



(a) ETTm1 (b) ETTh1 (c) Weather (d) Electricity

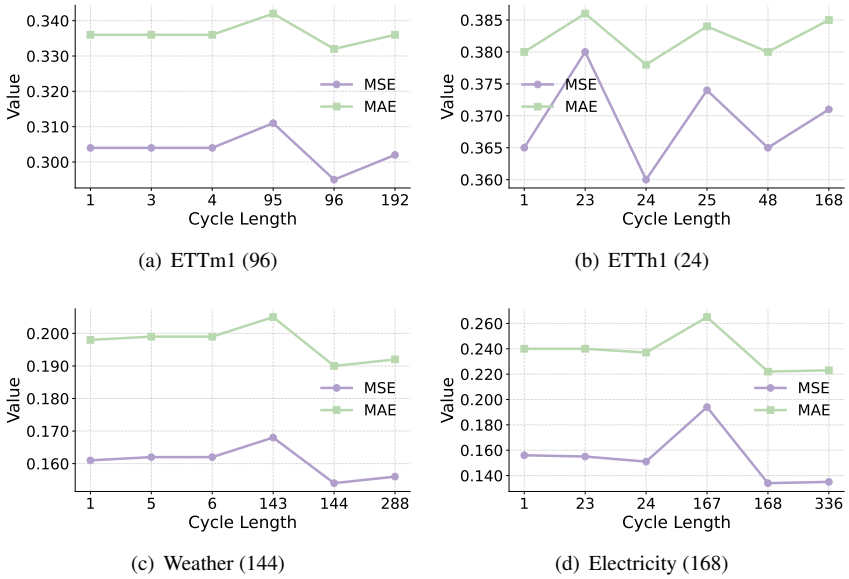
Figure 7: Sensitivity analysis of the hidden dimension d_{model} under the setting of lookback window $T = 96$ and prediction length $\tau = 96$.

E.4.2 Cycle Length W

We further investigate the sensitivity of Paloma to the cycle length W, as shown in Figure 8. From Figure 8, two key observations can be drawn. First, the best performance is consistently achieved when the cycle length W matches the inherent periodicity of the dataset. Second, when W deviates from the true cycle length, the performance degrades to varying degrees, indicating that accurate cycle specification is important for effective phase modeling. Notably, the degradation is more pronounced when W is close to but misaligned with the true periodicity (e.g., $W = 95$ for ETTm1, $W = 23$ or 25 for ETTh1, $W = 143$ for Weather, and $W = 167$ for Electricity), suggesting that small mismatches may introduce phase misalignment and disrupt the learned periodic structure.

These observations also reveal a limitation of Paloma, namely that its performance depends on the availability of a reasonably accurate prior for the cycle length W. When the assumed cycle length is misaligned with the true periodicity, the resulting phase inconsistency can negatively affect model performance. From another perspective, this sensitivity can be exploited as a diagnostic signal. When the true periodicity is unknown, one can evaluate the model under a set of candidate cycle lengths. *Empirically, the model tends to exhibit worse predictive performance when W is closer to but not aligned with the true cycle length, which may indicate that the assumed W is approaching*

the intrinsic periodicity of the dataset. This observation provides a practical heuristic for exploring the underlying periodic structure in real-world scenarios.



(a) ETTm1 (96)

(b) ETTh1 (24)

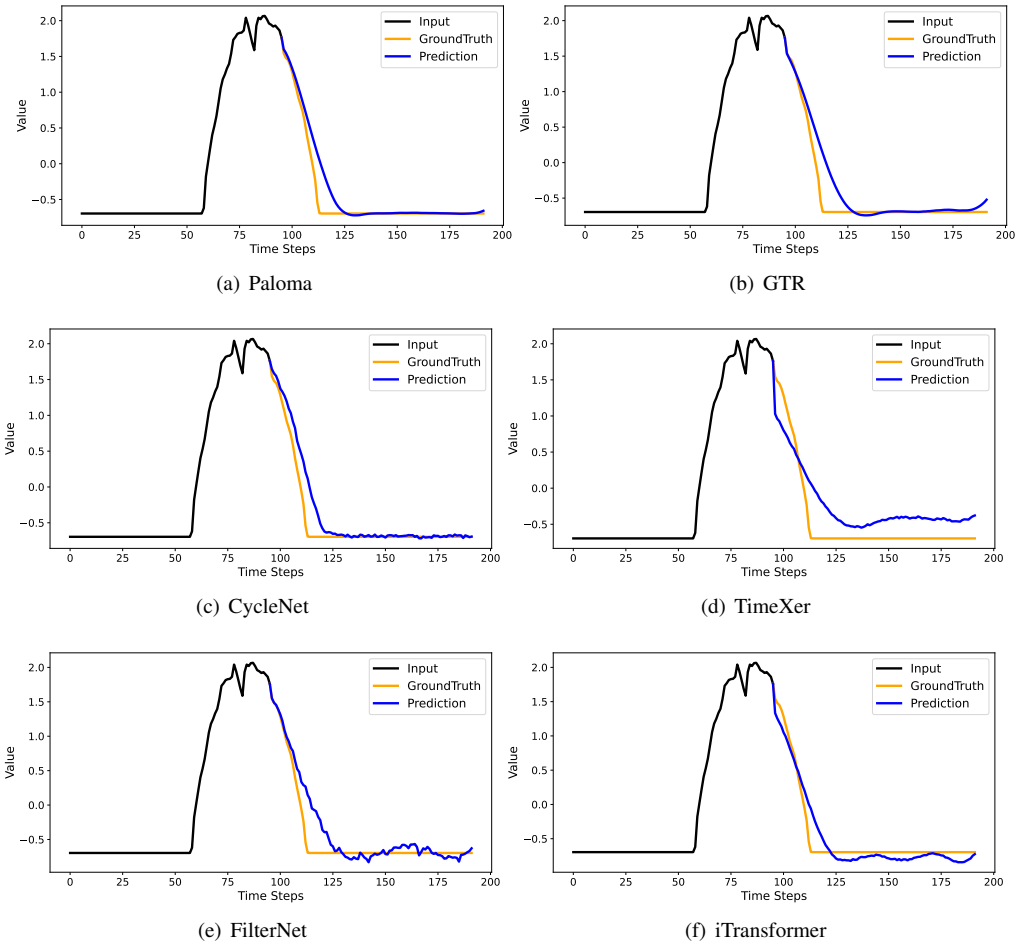
(c) Weather (144)

(d) Electricity (168)

Figure 8: Sensitivity analysis of the cycle length W under the setting of lookback window $T = 96$ and prediction length $\tau = 96$. The values in parentheses indicate the inherent periodicity of each dataset.

F Visualizations

To provide a more intuitive comparison with state-of-the-art methods, we present qualitative prediction results on the Solar-Energy and ETTm1 datasets, as shown in Figures 9 and 10. We select several representative models as baselines, including GTR [20], CycleNet [18], TimeXer [39], FilterNet [29], and iTransformer [9], covering diverse modeling paradigms. Compared with these approaches, Paloma produces more accurate and stable predictions of future temporal variations, demonstrating its superior ability to capture complex dynamics.

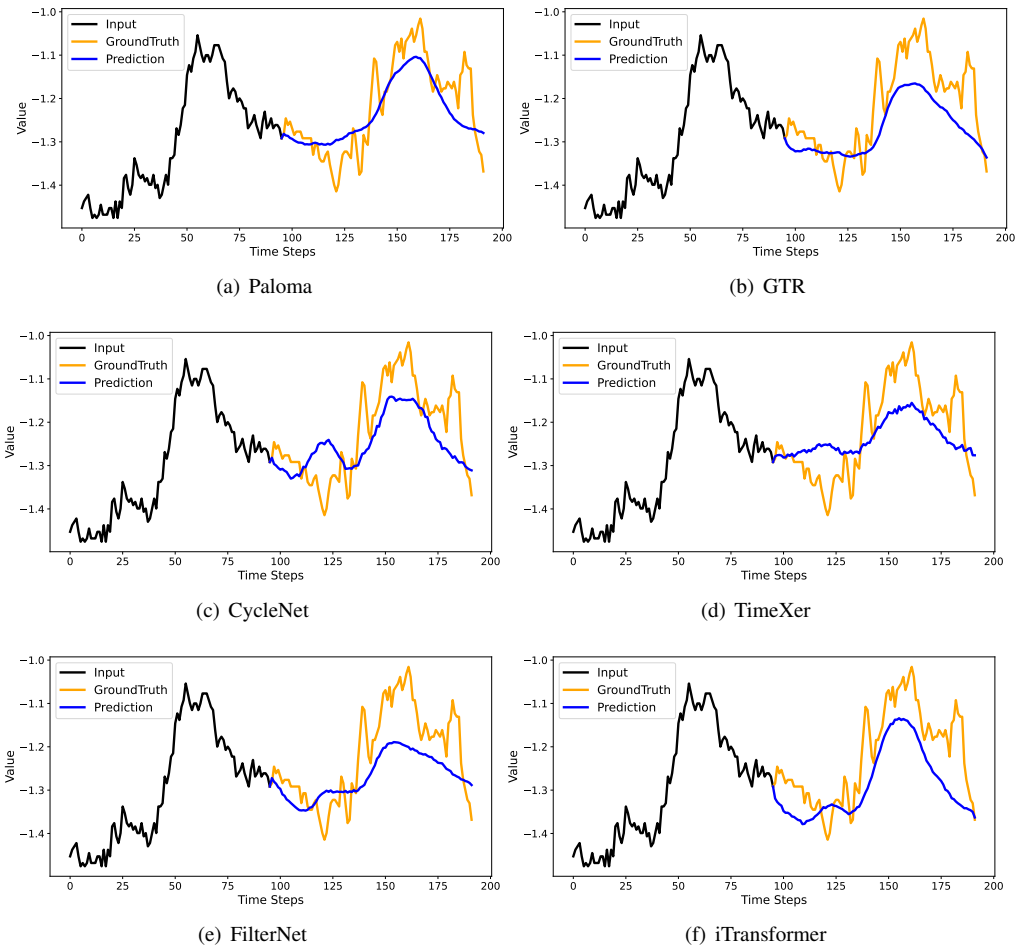


(a) Paloma (b) GTR

(c) CycleNet (d) TimeXer

(e) FilterNet (f) iTransformer

Figure 9: Visualization of forecasting results on the Solar-Energy dataset with lookback window $T = 96$ and prediction length $\tau = 96$.



(a) Paloma (b) GTR

(c) CycleNet (d) TimeXer

(e) FilterNet (f) iTransformer

Figure 10: Visualization of forecasting results on the ETTm1 dataset with lookback window $T = 96$ and prediction length $\tau = 96$.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The main claims are clearly written in the abstract and introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We have discussed the limitation of our method in the Conclusion and Limitation section.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: The complete proofs of the propositions are provided in Appendix B.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We have included the implementation details in Appendix D.3 and provided the code as supplementary material.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We have provided the code as supplementary material, which also includes the sources of the datasets.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Please see Appendix D.3 for implementation details.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We have reported the standard deviations of the results for our method in Table 7.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We have reported the computational resource requirements of our method in Table 4, Table 10, and Table 11.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: Our research aligns with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [N/A]

    Justification: This work aims to improve the accuracy of time series forecasting. To the best of our knowledge, it is not expected to introduce significant negative societal impacts.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: This paper does not have this risk.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: The code and datasets used in the paper are publicly available and properly credited.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [N/A]

    Justification: N/A.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: N/A.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: N/A.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: LLMs are used only for grammar and spell checking.

Guidelines:

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