PAMod: Modeling Cyclical Shifts via Phase-Amplitude Modulation for Non-stationary Time Series Forecasting

Yingbo Zhou[1], **Yutong Ye**[2], **Shuhao Li**[1], **Rui Qian**[1], **Qiang Huang**[1], **Lemao Liu**[1], **Li Sun**[3], **Dejing Dou**[1,4*]
[1]College of Computer Science and Artificial Intelligence, Fudan University
[2]School of Computer Science and Technology, Zhejiang Sci-Tech University
[3]School of Computer Science, Beijing University of Posts and Telecommunications, [4]BEDI Cloud
yingbo_zhou@fudan.edu.cn, dejingdou@fudan.edu.cn

Abstract

Real-world time series forecasting faces the fundamental challenge of non-stationary statistical properties, including shifts in mean and variance over time. While reversible instance normalization (RevIN) has shown promise by stationarizing inputs and denormalizing outputs, it relies on the strong assumption that historical and future distributions remain identical. We observe that in many practical applications, distribution shifts follow cyclical patterns that correlate with periodic positions (e.g., seasonal and holiday volatility). To this end, we propose **PAMod**, a lightweight yet powerful framework that models cyclical distribution shifts via **P**hase-**A**mplitude **Mod**ulation in the normalized feature space. PAMod learns periodic embeddings to modulate representations: phase modulation captures mean shifts, while amplitude modulation adapts to variance changes. Crucially, we prove mathematically that modulating in normalized space is equivalent to applying dynamic denormalization, offering an elegant unification of distribution adaptation and representation learning. Extensive experiments on twelve real-world benchmarks demonstrate that PAMod achieves state-of-the-art performance with fewer computational resources. Furthermore, our modulation mechanism, as a novel plug-and-play technique, can improve existing time-series forecasting methods with simple integration. Code is available at https://github.com/ZYBOBO/PAMod_NeurIPS.

1 Introduction

Time series forecasting underpins critical decision-making across domains, including energy management [1, 2], finance analysis [3, 4], healthcare monitoring [5, 6], and traffic optimization [7, 8]. Despite significant advances in deep learning architectures, real-world time series remain particularly challenging due to their inherent non-stationary statistical properties (mean, variance, and higher moments) evolve [9, 10], violating the stationarity assumptions of many models.

Empirical evidence of the non-stationary is shown in Figure 1(a), where the training and test sets of real-world time series (e.g., the OT channel in ETTh1) exhibit distinct probability distributions with diverging means and variances. Notably, distribution shift, as a key manifestation of non-stationarity, refers to the dynamic evolution of statistical properties over time, not to mismatch across different domains. The dashed lines marking the respective means illustrate that statistical properties of the test set often systematically diverge from those of the training set. To bridge this gap, two

*Corresponding author.

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

dominant research paradigms have emerged. The first, the normalization paradigm [9, 11, 12], seeks to eliminate non-stationarity by coercing series into a stationary space for modeling. The second paradigm relies on complex architectures [13, 14, 10], which attempt to implicitly model and absorb the full spectrum of distributional shifts in their vast parameter space.

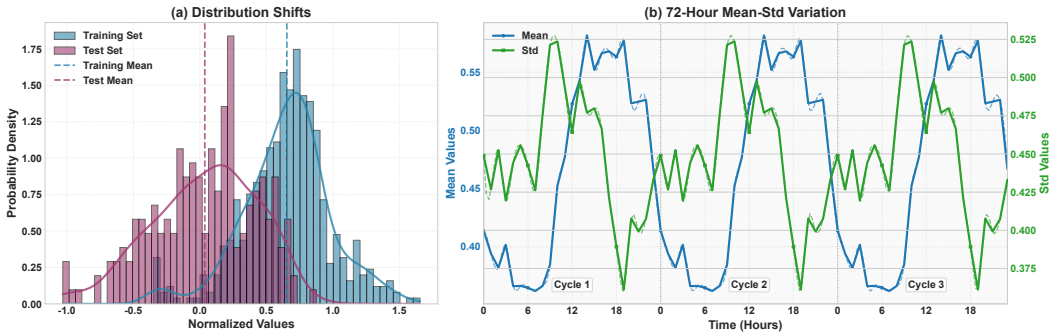


Figure 1: Distribution shift and periodic mean-std variation in the OT channel of ETTh1. (a) histograms and density curves illustrate distinct probability distributions, with dashed lines marking the respective means in training and test sets. (b) 72-hour cyclic variations exhibit stable 24-hour repetition across three consecutive cycles (Cycle 1–3), highlighting inherent diurnal periodicity. Notably, distribution shifts originate from decoupled mean-variance variations within these cycles.

We identify a critical, shared limitation in both approaches: they fail to explicitly model the structured, predictable component of non-stationarity that arises from cyclical dependencies. Normalization-based methods discard this structure by design, while complex architectures inefficiently attempt to rediscover it from data, often conflating it with noise. This leads to two key shortcomings: (1) **suboptimal parameter efficiency**, as either useful structure is removed or must be laboriously re-learned, and (2) **vulnerable generalization**, particularly when predictable cyclical shifts dominate, as models lack an inductive bias to capture them robustly.

We argue that rather than indiscriminately eliminating or preserving non-stationarity, we should distinguish between random fluctuations and structured cyclical shifts. As seen in Figure 1(b), over 72 consecutive hours (three 24-hour cycles), the OT channel in ETTh1 exhibits stable diurnal periodicity, where mean variations and variance fluctuations maintain cyclical regularity. Crucially, these components evolve independently yet synchronously across cycles, demonstrating that distribution shifts originate from structured rather than arbitrary variations. This observation motivates a paradigm shift: **distribution shifts in time series can decompose into cyclical components with correlated positions**. By modeling these cyclical patterns explicitly, we can extend the normalization paradigm to adapt to cyclical non-stationarity while capturing significant distribution shifts.

To explicitly model the cyclical shifts, we propose **P**hase-**A**mplitude **Mod**ulation (**PAMod**), a novel framework that learns cyclical distribution shifts through periodic modulation design. Inspired by the phase-amplitude modulation in communication systems [15], PAMod introduces two complementary mechanisms: phase modulation captures systematic mean shifts via additive transformations, while amplitude modulation adapts to variance changes via multiplicative scaling. These mechanisms operate in the normalized feature space, achieving equivalent distribution adaptation through a more elegant and unified approach.

In summary, our main contributions are three folds:

- We formalize cyclical distribution shifts as mean-variance decoupled variations, providing new analytical insight into periodic non-stationarity.
- We propose PAMod with the phase-amplitude modulation mechanism that learns to adjust means (phase) and variances (amplitude) based on cyclical positions.
- We establish new state-of-the-art performance across multiple real-world benchmarks, reducing MSE by 4%–16% with superior memory efficiency (5–10× fewer parameters than transformer-based methods).

2 Related Work

Time Series Forecasting Architectures. Deep learning has revolutionized time series forecasting through diverse architectures. Recurrent Neural Networks (RNNs) model sequential dependencies through recurrent states [16, 17], while Temporal Convolutional Networks (TCNs) employ dilated convolutions for efficient long-range modeling [18, 19]. Transformer-based approaches [20–22] leverage self-attention or frequency-domain mechanisms, and MLP-based methods [23, 24] demonstrate competitive performance with minimal complexity. Graph Neural Networks (GNNs) capture multivariate correlations via spatiotemporal graphs [25, 26], and state space models offer linear-time sequence modeling via selective scanning [27]. Despite their advances, these architectures often assume stationarity or treat periodicity as the auxiliary factor, leaving a gap for designs that explicitly unify periodic modeling with distribution shift mitigation.

Handing Non-stationary in Time Series. Non-stationarity is a core challenge in time series forecasting. Classical statistical methods [28] address this by applying temporal differencing to enforce stationarity. Recently, normalization techniques [9, 11, 12] have emerged as the de facto standardfor eliminating non-stationarity. Conversely, some specific designs for certain network architectures [13, 14, 10] have been introduced to capture distribution shifts. Additionally, decomposition-based methods [29, 30] separate time series into trend and seasonal components to mitigate non-stationary effects. A key limitation of these approaches is that they treat non-stationarity as a monolithic disturbance to be eliminated, rather than structured and periodic variations that can be explicitly modeled for targeted correction.

Periodic Modeling in Time Series. Periodicity is a ubiquitous property of real-world time series (e.g., diurnal cycles in sensor data), and modeling it effectively can enhance forecasting performance. Classical methods [31, 28] explicitly capture periodic patterns via predefined functions. To better leverage periodic information, current approaches [32–34] primarily focus on explicitly identifying, representing, or transforming repeating temporal patterns in the data. For instance, SparseTSF [33] utilizes a cross-period sparse technique to decouple cycles and trends, while CycleNet [34] introduces learnable recurrent cycles to model periodic patterns. Crucially, existing periodic modeling works rarely connect periodicity to distribution shifts, which treat periodic patterns as predictive features rather than the root cause of structured drift.

Unlike the above methods, PAMod bridges these gaps through a cohesive design: i) a lightweight framework that avoids computation inefficiency; ii) an explicit modeling approach to the non-stationary that preserves informative cyclical structures; and iii) a novel mechanism that directly ties periodicity to distribution shifts via the phase-amplitude modulation. To the best of our knowledge, PAMod pioneers the utilization of **explicit cyclical modeling to directly adapt to the non-stationary nature of time series data**.

3 Methodology

3.1 Preliminaries

Problem Definition. Given historical time series $X = \{x_1, x_2, \cdots, x_T\} \in \mathbb{R}^{T\times C}$ with C variates, the goal of time series forecasting is to predict next H-step values $\hat{Y} = \{\hat{x}_{T+1}, \hat{x}_{T+2}, \cdots, \hat{x}_{T+H}\} \in \mathbb{R}^{H\times C}$ with learnable forecasting function $f_\theta(\cdot)$:

$$\hat{x}_{T+1:T+H} = f_\theta(x_{1:T}). \tag{1}$$

RevIN Formulation. To mitigate the non-stationarity of time series, Reversible Instance Normalization (RevIN), as the normalization paradigm, is adopted by almost all the mainstream forecasters. Concretely, RevIN [9] normalizes the inputs as follows:

$$\begin{aligned} \overline{x}_t =& \frac{x_t - \mu_X}{\sigma_X}, \quad \mu_X = \frac{1}{T}\sum_{t=1}^{T} x_t, \\ \sigma_X =& \sqrt{\frac{1}{T}\sum_{t=1}^{T}(x_t - \mu_X)^2}. \end{aligned} \tag{2}$$

After forecasting $\overline{y}_m = f_\theta(\overline{x}_t)$, it denormalizes:

$$\hat{y}_m = \overline{y}_m \cdot \sigma_X + \mu_X, \tag{3}$$

where $m \in \{T+1, \cdots, T+H\}$. This assumes consistent distributions between historical inputs X and future ground truth Y, i.e., $\mu_X = \mu_Y, \sigma_X = \sigma_Y$, which fails in real-world applications as distribution shifts occur.

3.2 Overview of PAMod

In Figure 1(b), we can observe that distribution shifts can be decomposed into mean shifts (phase) and variance changes (amplitude), both functions of periodic positions. To this end, we model cyclical shifts via phase-amplitude modulation in the normalized feature space. Our PAMod consists of instance-wise normalization, learnable cyclical embedding-guided phase-amplitude modulation, MLP and denormalization, illustrated in Figure 2. While normalization removes arbitrary, non-cyclical distribution shifts to stabilize learning, our approach explicitly models and reintroduces the structured, predictable non-stationarity inherent in cyclical patterns to enhance forecasting accuracy.

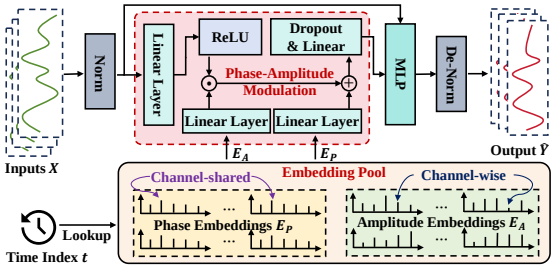


Figure 2: Overview of PAMod, which consists of normalization, phase-amplitude modulation with the embedding pool, MLP and denormalization for non-stationary time series forecasting.

3.3 Core Components of PAMod

Intuition. The design of PAMod is based on the generalized expression of amplitude-phase modulation in communication theory [35, 15].

Consider a carrier wave $\cos(2\pi f_c t)$ whose amplitude $A(t)$ and phase $P(t)$ are simultaneously modulated by information signals $M_A(t)$ and $M_P(t)$:

$$x_M(t) = \underbrace{[A_c + M_A(t)]}_{A(t)} \cdot \cos(2\pi f_c t + \underbrace{k \cdot M_P(t)}_{P(t)}), \tag{4}$$

where A_c is the constant carrier amplitude, and k is the phase modulation index. To elucidate the combined effect, we apply a first-order Taylor expansion under the assumption of a small phase deviation (i.e., $kM_P(t) \ll 1$), which corresponds to modeling smooth distribution shifts:

$$\begin{aligned} x(t) &\approx A(t)[\cos(2\pi f_c t) - kM_P(t)\sin(2\pi f_c t)] \\ &= \underbrace{A(t)\cos(2\pi f_c t)}_{\text{In-phase Component}} - \underbrace{A(t)kM_P(t)\sin(2\pi f_c t)}_{\text{Quadrature Component}}. \end{aligned} \tag{5}$$

Since $kM_P(t) \ll 1$, the magnitude of the quadrature term satisfies $A(t)kM_P(t) \ll A(t)$. Consequently, the quadrature term can be treated as a weak additive perturbation $\Phi(t)$, and the primary signal satisfies:

$$x(t) \approx A(t)\cos(2\pi f_c t) + \Phi(t). \tag{6}$$

Recalling Equation (3), we can find that the denormalization process and the phase-amplitude modulation share a perfectly aligned mathematical structure. Therefore, instead of using static scaling factors, we can modulate the normalized features with learnable cyclical signals to guide the learning of dynamic distribution shifts.

Learnable Cyclical Embedding. To obtain modulation signals that resemble $A(t)$ and $\Phi(t)$ within the time series modeling framework, we propose learnable cyclical embeddings that, unlike the ideal sine waves real-world trends rarely follow, serve as data-driven proxies parameterizing the structured phase (mean offset) and amplitude (variance fluctuation) variations.

Considering that real-world time series typically exhibit prominent and explicit cyclic patterns [34, 36], the specific cycle length L is available and straightforward via autocorrelation functions

[37]. Given a learnable mean matrix $W_m \in \mathbb{R}^{L\times T}$, a shared embedding vector is retrieved based on the cyclical position of the absolute time t for all the channels:

$$E_P(t) = \text{Lookup}(W_m, t \bmod L) \in \mathbb{R}^{1\times T}, \tag{7}$$

where the Lookup operation conditions the model on the absolute phase within the cycle ($t \bmod L$), enabling it to dynamically adapt to systematic mean offsets that correlate with periodic patterns.

Since the heterogeneous variance shifts observed across different channels, we define a channel-wise embedding tensor $W_v \in \mathbb{R}^{L\times C\times T}$ to capture cycle-dependent variance fluctuations for each channel $i \in [0, C-1]$:

$$E_A^i(t) = \text{Lookup}(W_v, i, t \bmod L) \in \mathbb{R}^{C\times T}. \tag{8}$$

The amplitude embedding $E_a^i(t)$ allows each variate to independently learn to amplify or suppress variance fluctuations based on its position within the periodic cycle.

Phase-Amplitude Modulation. With learnable $E_P(t)$ and $E_A(t)$, we implement our phase-amplitude modulation to transform the normalized features X_{norm}. First, we use linear projection U_1 with ReLU activation to extract nonlinear temporal patterns as carrier waves:

$$X_1 = \text{ReLU}(U_1 X_{\text{norm}}), \;\; U_1 \in \mathbb{R}^{T\times S}, \tag{9}$$

where $S = 4T$ in the hidden dimension. After that, we project the learnable embeddings to bridge the gap between the embedding and feature space, and then model cyclical shifts via element-wise multiplication and additive fusion:

$$X_2 = X_1 \odot U_2 E_A(t) + U_3 E_P(t), \;\; U_2, U_3 \in \mathbb{R}^{T\times S}. \tag{10}$$

Finally, the modulated features X_{mod} can be obtained with linear projection and Dropout operation:

$$X_{\text{mod}} = \text{Dropout}(U_4 X_2), \;\; U_4 \in \mathbb{R}^{S\times T}. \tag{11}$$

MLP-based Prediction Head. We adopt a two-layer MLP architecture, optionally with Dropout operation, to project the modulated features into the expected prediction horizon:

$$\overline{Y} = U_6(\text{Dropout}(\text{GeLU}(U_5(X_{\text{norm}} + X_{\text{mod}}))), \tag{12}$$

where $U_5 \in \mathbb{R}^{d\times T}$ and $U_6 \in \mathbb{R}^{H\times d}$.

3.4 Theoretical Analysis

We provide a concise theoretical argument explaining why explicitly modeling cycle-dependent distribution shift via phase-amplitude modulation improves generalization, especially under periodic non-stationarity. Notably, although the argument builds on periodic signals, PAMod relies on fully learnable cyclical embeddings, which can capture latent recurrent patterns beyond rigid cycles.

Periodic distribution shift setting. Let $X_t \in \mathbb{R}^{C\times T}$ be an input window ending at absolute time t, and let $\tau = t \bmod L$ denote the cyclical position. Real-world series (e.g., load/traffic) often exhibit *periodic distribution shift* $\mathcal{D}$: the conditional distribution changes with τ while remaining relatively stable across different cycles:

$$\mathcal{D}_t \equiv \mathcal{D}_\tau, \quad \tau = t \bmod L, \tag{13}$$

i.e., samples with the same τ are drawn from the same distribution regardless of the time index.

Cycle-dependent location-scale shift. A common and practically relevant family of shifts is the channel-wise location-scale form:

$$X_t = \mu(\tau) + \sigma(\tau) \odot Z_t, \qquad \tau = t \bmod L, \tag{14}$$

where Z_t is an approximately cycle-invariant process, and $\mu(\tau)$ and $\sigma(\tau)$ encode structured *mean offsets* and *variance fluctuations* within each cycle. Based on Equation (14), the optimal predictor is inherently τ-conditional:

$$f_\theta^\star(X_t, \tau) = \mathbb{E}[Y_t \mid X_t, \tau], \tag{15}$$

and ignoring τ forces a single hypothesis to fit a mixture of heterogeneous distributions $\{\mathcal{D}_\tau\}_{\tau=0}^{L-1}$, which typically increases approximation error and harms out-of-distribution performance across time.

PAMod as cycle-aware mean and variance adaptation. According to Equation (3) and (12), the modulated prediction can be expressed as:

$$\begin{aligned}\hat{Y}_{\text{PAMod}} &= \sigma_X \cdot f_\theta(X_{\text{norm},\tau}) + \mu_X \\ &= \sigma_X \cdot f_\theta(X_{\text{norm}} + X_{\text{mod}}) + \mu_X.\end{aligned} \tag{16}$$

Assuming the neural network is approximately linear and the modulation is small relative to the normalized input, PAMod's prediction can be approximated as:

$$\begin{aligned}\hat{Y}_{\text{PAMod}} = \sigma_X \cdot f_\theta(X_{\text{norm}}) \odot (1 + \alpha(\tau)) \\ + (\sigma_X \cdot \phi(\tau) + \mu_X),\end{aligned} \tag{17}$$

where $\alpha(\tau)$ and $\phi(\tau)$ represent the learned amplitude $U_2 E_A(t)$ and phase $U_3 E_P(t)$, respectively. This formulation shows that our PAMod can adaptively adjust both the mean and variance of the base prediction $f_\theta(X_{\text{norm}})$ based on the cyclical position τ. Moreover, we provide the performance guarantee of PAMod under periodic location-scale shifts in Appendix A and D.

Table 1: Multivariate forecasting performance. The lookback length is set to $L = 96$ and all the results are averaged from all predictions $H \in \{12, 24, 48, 96\}$ for PEMS and $H \in \{96, 192, 336, 720\}$ for other benchmarks. See Table 8 in the Appendix for the full results.

Model	**PAMod**		TQNet		TimeEmb		FilterTS		Amplifier		TimeXer		CycleNet		TimeMixer		iTransformer		PatchTST	
	(Ours)		[2025]		[2025]		[2025]		[2025]		[2024]		[2024]		[2024]		[2024]		[2023]	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	**0.363**	**0.378**	0.377	0.393	0.368	0.385	0.385	0.396	0.382	0.395	0.382	0.397	0.379	0.396	0.381	0.395	0.407	0.410	0.387	0.400
ETTm2	**0.263**	**0.307**	0.277	0.323	0.265	0.308	0.277	0.322	0.280	0.326	0.274	0.322	0.266	0.314	0.275	0.323	0.288	0.332	0.281	0.326
ETTh1	**0.413**	**0.420**	0.441	0.434	0.425	0.425	0.434	0.430	0.430	0.428	0.437	0.437	0.457	0.441	0.447	0.440	0.454	0.448	0.469	0.455
ETTh2	**0.360**	**0.387**	0.378	0.402	0.362	0.390	0.375	0.398	0.381	0.405	0.368	0.396	0.388	0.409	0.364	0.395	0.383	0.407	0.387	0.407
ECL	0.165	**0.254**	**0.164**	0.259	0.168	0.261	0.180	0.272	0.172	0.266	0.171	0.270	0.168	0.259	0.182	0.272	0.178	0.270	0.205	0.290
Traffic	0.434	**0.272**	0.445	0.276	0.453	0.295	0.470	0.315	0.483	0.317	0.466	0.287	0.472	0.314	0.484	0.297	**0.428**	0.282	0.481	0.300
Weather	0.239	**0.261**	0.242	0.269	**0.237**	0.262	0.245	0.274	0.253	0.275	0.241	0.271	0.243	0.271	0.240	0.271	0.258	0.278	0.259	0.273
Solar	0.206	**0.226**	**0.198**	0.256	0.248	0.270	0.215	0.277	0.241	0.270	0.237	0.302	0.210	0.261	0.216	0.280	0.233	0.262	0.270	0.307
PEMS03	**0.091**	**0.189**	0.097	0.203	0.104	0.207	0.134	0.246	0.131	0.239	0.112	0.214	0.118	0.226	0.167	0.267	0.113	0.222	0.180	0.291
PEMS04	**0.086**	**0.185**	0.091	0.197	0.096	0.200	0.125	0.241	0.135	0.249	0.105	0.209	0.119	0.232	0.185	0.287	0.111	0.221	0.195	0.307
PEMS07	0.080	**0.168**	**0.075**	0.171	0.097	0.188	0.120	0.220	0.122	0.226	0.085	0.182	0.113	0.214	0.181	0.271	0.101	0.204	0.211	0.303
PEMS08	**0.119**	**0.190**	0.142	0.229	0.132	0.221	0.180	0.266	0.183	0.271	0.175	0.250	0.150	0.246	0.226	0.299	0.150	0.226	0.280	0.321
1st Count	**7**	**12**	3	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

4 Experiments

In this section, we conduct comprehensive experiments with real-world time series benchmarks to sufficiently assess the performance of our proposed PAMod, including comparison with SOTA baselines on performance and efficiency (Section 4.2), ablation studies (Section 4.3), compatibility analysis (Section 4.4), and interpretable cases (Section 4.5).

4.1 Experimental Settings

Datasets. Following established evaluation protocols [42, 36] in time series forecasting literature, we perform experiments on twelve real-world benchmark datasets. These include four ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2), Electricity (ECL), Traffic, Solar-Energy, Weather, and four PEMS datasets (PEMS03, PEMS04, PEMS07, PEMS08) [20, 44, 18]. In line with common practice, each dataset is partitioned into training, validation, and test sets. For the ETT series, we adopt a 6:2:2 split ratio, while a 7:1:2 ratio is used for all other datasets. The prediction horizons H are set to $\{12, 24, 48, 96\}$ for the PEMS datasets and $\{96, 192, 336, 720\}$ for the others.

Baselines. We compare PAMod against nine state-of-the-art models in recent years, including TQNet [36], TimeEmb [38], FilterTS [39], Amplifier [40], TimeXer [41], CycleNet [34], TimeMixer [30], iTransformer [42], and PatchTST [43]. In line with standard evaluation protocols, a uniform look-back length of 96 is adopted for all models to ensure a fair comparison.

Implementation Details. All experiments are implemented in PyTorch [45] and trained on a single NVIDIA GeForce RTX 4090 GPU (24GB), using the Adam optimizer [46]. To ensure stable training, we follow prior work [47, 10] in employing a hybrid loss function that combines Mean Absolute Error (MAE) in both the time and frequency domains. The key hyperparameter τ, which defines the periodicity length, is set for each dataset following the established guidelines from CycleNet and TQNet. Model performance is evaluated using the standard metrics of Mean Squared Error (MSE) and Mean Absolute Error (MAE).

4.2 Main Results

Forecasting Performance. Table 1 shows the comparison results of PAMod with 9 recent methods across 12 real-world time series benchmarks. Lower MSE and MAE values indicate better forecasting performance. Overall, PAMod consistently achieves Top 2 performance across all datasets, with the smallest error metrics in 19 out of 24 evaluation cases, demonstrating new state-of-the-art accuracy. Notably, an increasing number of variables in a dataset tends to amplify cross-series non-stationarity, which in turn increases the difficulty of forecasting. Nevertheless, PAMod still delivers competitive and even better performance. For instance, while TQNet employs cross-attention mechanisms to capture temporal dependencies and achieves an average MSE of 0.445 on the Traffic dataset, PAMod, which leverages lightweight phase-amplitude modulation to explicitly model non-stationary dynamics, attains a lower average MSE of 0.434 on the same dataset. Those results confirm that explicitly modeling non-stationarity is a more effective strategy for capturing complex temporal patterns.

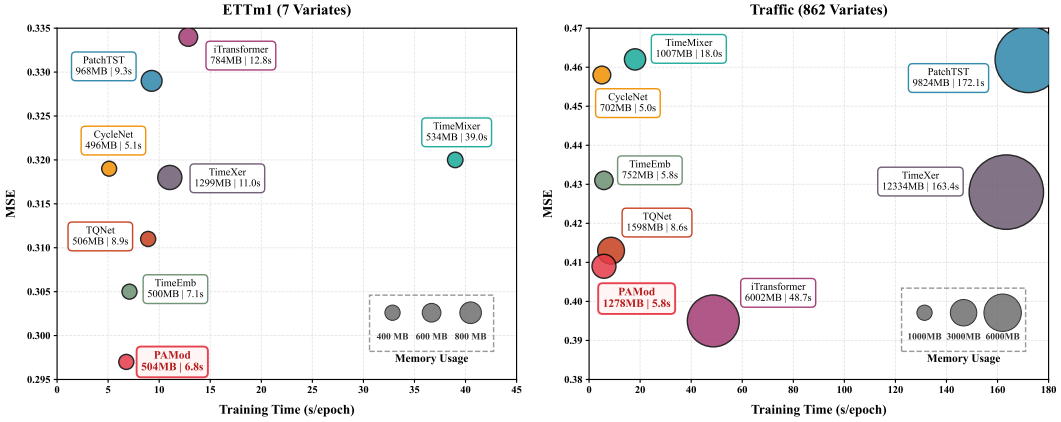


Figure 3: Computational efficiency under the input-96-predict-96 setting. Comparison of forecasting performance (MSE), memory usage, and training time on the ETTm1 and Traffic datasets. We set the batch sizes to 32 and 16 for the ETTm1 and Traffic datasets, respectively.

Model Efficiency. To evaluate the efficiency of PAMod, we record the training time and memory footprint of PAMod with seven baseline models under the same batch size for a fair comparison. As shown in Figure 3, PAMod demonstrates a superior balance of prediction accuracy, training speed, and memory efficiency across both small- and large-scale forecasting scenarios. On the ETTm1 dataset (7 Variates), it achieves the lowest MSE (0.297) while maintaining training time second only to CycleNet and moderate memory consumption. When evaluated in the Traffic dataset (862 variates), PAMod not only retains competitive accuracy but also exhibits near-optimal training speed and drastically reduced memory usage—surpassing strong baselines by more than an order of magnitude in both time and memory efficiency. These results underscore PAMod's scalable and resource-efficient design, making it well-suited for real-world deployment where computational constraints are as critical as predictive performance.

4.3 Ablation Studies

To investigate the role of each component in PAMod, we perform comprehensive ablation studies on the modulation mechanism, training loss, and cycle length.

Table 2: Ablation studies. The lookback length and prediction horizon are set to 96 for all the datasets.

Type	**PAMod** (**Ours**)		Modulation Variants												Training loss			
			w/o $\phi(\tau)$		w/o $\alpha(\tau)$		r/w LRC		r/w $\sin(\tau)$		s/w $\phi(\tau)$ & $\alpha(\tau)$		w/o modulation		MAE		MSE	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	**0.297**	**0.337**	0.302	0.341	0.304	0.342	0.305	0.344	0.308	0.347	0.301	0.340	0.324	0.355	0.294	0.331	0.296	0.340
ETTm2	**0.162**	**0.240**	0.168	0.245	0.168	0.246	0.163	0.241	0.168	0.245	0.163	0.241	0.175	0.253	0.164	0.240	0.164	0.242
ETTh1	**0.357**	**0.382**	0.361	0.384	0.365	0.385	0.359	0.383	0.363	0.386	0.359	0.383	0.371	0.387	0.362	0.379	0.360	0.386
ETTh2	**0.279**	**0.328**	0.284	0.330	0.287	0.335	0.280	0.327	0.284	0.330	0.281	0.328	0.298	0.341	0.284	0.330	0.283	0.330
ECL	**0.136**	**0.226**	0.139	0.228	0.156	0.245	0.141	0.229	0.145	0.232	0.142	0.230	0.181	0.256	0.138	0.225	0.137	0.229
Traffic	**0.409**	**0.256**	0.413	0.259	0.418	0.265	0.417	0.256	0.425	0.262	0.420	0.257	0.497	0.310	0.412	0.250	0.410	0.258
Weather	**0.153**	**0.191**	0.156	0.194	0.168	0.203	0.153	0.191	0.158	0.194	0.156	0.193	0.194	0.224	0.154	0.191	0.155	0.193
Solar	**0.184**	**0.209**	0.188	0.212	0.188	0.205	0.197	0.216	0.199	0.220	0.185	0.210	0.239	0.261	0.191	0.207	0.184	0.210
PEMS03	**0.133**	**0.228**	0.139	0.234	0.182	0.280	0.133	0.228	0.142	0.238	0.135	0.230	0.247	0.346	0.132	0.226	0.132	0.230
PEMS04	**0.111**	**0.212**	0.114	0.216	0.188	0.287	0.115	0.215	0.123	0.219	0.113	0.214	0.278	0.365	0.112	0.212	0.113	0.213
PEMS07	**0.116**	**0.199**	0.121	0.203	0.172	0.268	0.122	0.207	0.132	0.211	0.128	0.212	0.317	0.373	0.114	0.198	0.110	0.201
PEMS08	**0.181**	**0.223**	0.186	0.228	0.210	0.283	0.189	0.226	0.205	0.247	0.195	0.262	0.348	0.378	0.179	0.224	0.175	0.226
Avg	**0.210**	**0.253**	0.214	0.256	0.234	0.279	0.215	0.255	0.221	0.261	0.215	0.258	0.289	0.321	0.212	0.251	0.210	0.255

Components. As shown in Table 2, $\phi(\tau)$ and $\alpha(\tau)$ denote cycle-based phase bias and amplitude modulation, respectively. 'r/w LRC' introduces the learnable recurrent cycle as CycleNet's to replace $\phi(\tau)$ and $\alpha(\tau)$. Meanwhile, we also design the τ-based Sinusoidal function to replace $\phi(\tau)$ and $\alpha(\tau)$ as 'r/w $\sin(\tau)$', and swap $\phi(\tau)$ and $\alpha(\tau)$ as 's/w $\phi(\tau)$ & $\alpha(\tau)$'. The results of Modulation Variants in Table 2 are inferior to those of PAMod, indicating the effectiveness of our phase-amplitude modulation mechanism. Moreover, the pure MLP model (i.e., w/o modulation) obtains the worst results, confirming the importance of explicitly modeling cyclical shifts. Additionally, adopting either MAE or MSE loss has a negligible impact on the forecasting performance, which further highlights that the accuracy improvements of PAMod originate primarily from its explicit modeling of cycle-dependent phase and amplitude dynamics.

Table 3: Performance comparison of integrating our PAMod mechanism with different backbone models on the ETTh2 and ETTm1 datasets. The best results are highlighted in **bold**, and "Imp." denotes the performance improvement achieved by incorporating PAMod.

Dataset	ETTh2								ETTm1							
Horizon	96		192		336		720		96		192		336		720	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
DLinear [2023]	0.333	0.387	0.477	0.476	0.594	0.541	0.831	0.657	0.345	0.372	0.380	0.389	0.413	0.413	0.474	0.453
+PAMod	**0.312**	**0.342**	**0.426**	**0.433**	**0.445**	**0.472**	**0.484**	**0.489**	**0.317**	**0.348**	**0.355**	**0.372**	**0.395**	**0.392**	**0.448**	**0.432**
Imp.	6.3%	11.6%	10.7%	9.0%	25.1%	12.8%	41.8%	25.6%	8.1%	6.5%	6.6%	4.4%	4.4%	5.1%	5.5%	4.6%
PatchTST [2023]	0.302	0.348	0.388	0.400	0.426	0.433	0.431	0.466	0.329	0.367	0.367	0.385	0.399	0.410	0.545	0.439
+PAMod	**0.293**	**0.334**	**0.365**	**0.388**	**0.401**	**0.418**	**0.419**	**0.438**	**0.312**	**0.345**	**0.352**	**0.370**	**0.380**	**0.394**	**0.478**	**0.428**
Imp.	3.0%	4.0%	5.9%	3.0%	5.9%	3.5%	2.8%	6.0%	5.2%	6.0%	4.1%	3.9%	4.8%	3.9%	12.3%	2.5%
iTransformer [2024]	0.297	0.349	0.380	0.400	0.428	0.432	0.427	0.445	0.334	0.368	0.377	0.391	0.426	0.420	0.491	0.459
+PAMod	**0.284**	**0.332**	**0.363**	**0.385**	**0.406**	**0.414**	**0.410**	**0.431**	**0.312**	**0.349**	**0.354**	**0.376**	**0.393**	**0.399**	**0.446**	**0.434**
Imp.	4.4%	4.9%	4.5%	3.8%	5.1%	4.2%	4.0%	3.1%	6.6%	5.2%	6.1%	3.8%	7.7%	5.0%	9.2%	5.4%
TQNet [2025]	0.295	0.343	0.367	0.393	0.417	0.427	0.433	0.446	0.311	0.353	0.356	0.378	0.390	0.401	0.452	0.440
+PAMod	**0.280**	**0.329**	**0.358**	**0.378**	**0.399**	**0.413**	**0.412**	**0.433**	**0.299**	**0.337**	**0.347**	**0.369**	**0.375**	**0.388**	**0.435**	**0.425**
Imp.	5.1%	4.1%	2.5%	3.8%	4.3%	3.3%	4.8%	2.9%	3.9%	4.5%	2.5%	2.4%	3.8%	3.2%	3.8%	3.4%

Cycle Lengths. The learnable phase and amplitude embeddings $\phi(\tau)$ and $\alpha(\tau)$ in our PAMod mechanism are explicitly conditioned on the cycle length τ. To investigate its critical role, we conduct a systematic study of τ across the ECL and PEMS07 datasets. As shown in Figure 4, performance peaks when τ aligns with the inherent periodicity defined by the dataset's recording frequency. For instance, in the hourly-sampled ECL dataset, the daily cycle ($\tau = 24$) substantially outperforms non-periodic lengths, and the weekly cycle ($\tau = 24 \times 7 = 168$) achieves optimal results. These findings confirm that distribution shifts are

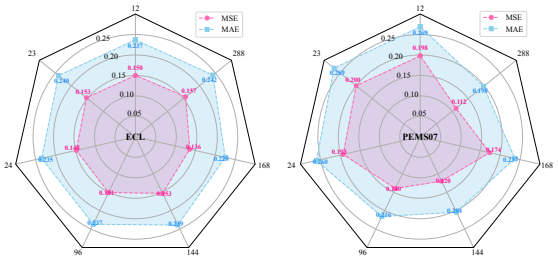


Figure 4: Performance of PAMod on the ECL and PEMS07 with varying cycle length $\tau \in \{12, 23, 24, 96, 144, 168, 288\}$.

structurally aligned with the maximum periodicity [34, 36]. Therefore, aligning τ accordingly is essential: it allows our model to explicitly capture these structured cyclical variations, which is the key to improving forecasting performance under non-stationarity.

4.4 Compatibility Analysis

To evaluate the compatibility and versatility of our phase-amplitude modulation (PAMod) mechanism, we seamlessly integrate it as a plug-and-play module into a diverse set of backbones, including both MLP-based and Transformer-based architectures. Crucially, PAMod is designed to be architecture-agnostic, capable of functioning as a complementary learnable embedding module or as a direct replacement for existing normalization techniques.

To evaluate the compatibility of our phase-amplitude modulation (PAMod) mechanism, we seamlessly integrate it as a plug-and-play module into diverse backbones and normalizations. As shown in Table 3, PAMod consistently improves performance when integrated with distinct backbones on ETTh2 and ETTm1. In Table 4, PAMod also outperforms specialized normalizations, i.e., RevIN [9], SAN [11], and DDN [12], on the Weather and ECL datasets. These results validate that PAMod acts on general feature representations, making it a lightweight, plug-in module for improving non-stationary time series forecasting.

Table 4: MSE comparison with normalization methods.

Backbone		Autoformer [2021]				SparseTSF [2024]			
Norm Type		RevIN	SAN	DDN	**PAMod**	RevIN	SAN	DDN	PAMod
Weather	96	0.212	0.194	0.190	**0.188**	0.186	0.182	0.178	**0.164**
	192	0.264	0.258	0.231	**0.227**	0.231	0.229	0.224	**0.216**
	336	0.309	0.329	0.289	**0.281**	0.285	0.283	0.279	**0.273**
	720	0.377	0.440	0.369	**0.354**	0.360	0.362	0.358	**0.351**
	Avg	0.291	0.305	0.270	**0.263**	0.266	0.264	0.260	**0.251**
ECL	96	0.179	0.172	0.150	**0.148**	0.202	0.189	0.185	**0.137**
	192	0.216	0.195	0.173	**0.169**	0.199	0.192	0.188	**0.154**
	336	0.233	0.211	0.185	**0.180**	0.212	0.201	0.194	**0.170**
	720	0.246	0.236	0.201	**0.199**	0.253	0.248	0.235	**0.208**
	Avg	0.219	0.204	0.177	**0.174**	0.217	0.208	0.201	**0.167**

To determine how the richness of historical information shapes model behavior, we evaluate sensitivity across different lookback window lengths. As shown in Figure 5, PAMod's performance scales positively with the lookback window length while maintaining a consistent lead over baselines. This scaling effect demonstrates the efficacy of using learnable embeddings to model distribution shifts. By dynamically capturing the evolving phase (mean) and amplitude (variance) of cyclical patterns, these embeddings provide a stable representational basis that makes long historical sequences interpretable, enabling the model to leverage extended contexts and capture genuine long-term dependencies.

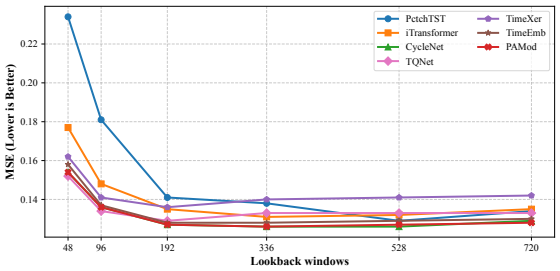


Figure 5: Performance under varying lookback windows on ECL. The prediction horizon is fixed as 96.

4.5 Interpretable Cases

To gain an intuitive understanding of PAMod's capability to model cyclical shifts, we visualize both the learned embedding weights and the corresponding prediction outcomes.

Phase-Amplitude Weights. As shown in Figure 6, the t-SNE projections of the phase and amplitude weights exhibit a clear structural disentanglement, directly reflecting their designed roles. The phase weight cluster is strictly aligned according to the cycle index, demonstrating its primary function in capturing temporal mean shifts. Conversely, the amplitude weight forms distinct clusters based on channel identity, indicating that it learns to modulate channel-specific variance. Such clean separation confirms that our phase-amplitude modulation successfully

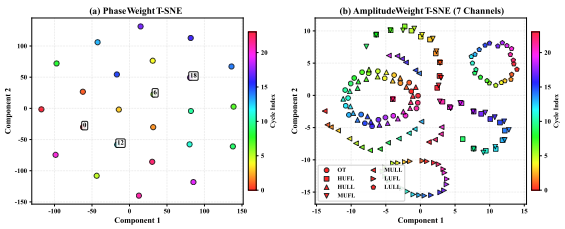
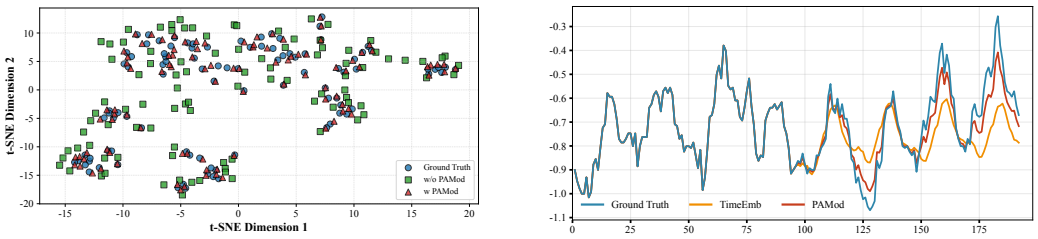


Figure 6: t-SNE [48] visualization of learned phase and amplitude weights on the ETTh1 dataset.

decouples the temporal (phase) and channel-wise (amplitude) factors of distribution shifts, providing an explicit and interpretable mechanism to model non-stationarity in multivariate time series.



(a) The predicted and true distributions

(b) Visualization of forecasting performance

Figure 7: Visualizations of the learned distributions and forecasting performance on ETTh1.

Distribution Alignment. Figure 7(a) provides geometric evidence that PAMod aligns predictions with the ground truth under distribution shift. The ground truth forms a compact, structured cluster, representing the intrinsic data manifold. Predictions with PAMod closely overlap this cluster, maintaining its structure and continuity, indicating high distributional consistency. In contrast, the predictions without PAMod are widely scattered and deviate markedly from the true manifold. This visual evidence validates that our phase-amplitude mechanism explicitly models temporal distribution drift, therefore enabling predictions to reside on the true data manifold. Furthermore, Figure 7(b) shows that PAMod achieves a closer fit to the ground truth than TimeEmb, accurately replicating both its long-term trend and local variations. This confirms its enhanced capacity for modeling complex temporal dynamics and achieving fine-grained temporal alignment.

5 Conclusion

Real-world time series exhibit inherent non-stationarity, which often manifests as structured, cyclical distribution shifts over time. To address this challenge, we propose **PAMod**, a **P**hase-**A**mplitude **Mod**ulation framework that explicitly models cyclical shifts in mean (phase) and variance (amplitude) through learnable periodic embeddings. Operating in normalized feature space, PAMod performs dynamic distribution adaptation equivalent to learnable denormalization, unifying shift modeling with representation learning in a lightweight, plug-and-play module. Extensive experiments validate that PAMod achieves state-of-the-art forecasting accuracy across diverse real-world benchmarks while maintaining high parameter efficiency, demonstrating that explicitly modeling cyclical non-stationarity is both effective and essential for robust time series forecasting. Future work may explore extending PAMod to model multi-scale periodicities simultaneously and adapt the modulation mechanism to other time series tasks.

Acknowledgements and Disclosures

We would like to thank the anonymous reviewers and area chairs for their valuable comments and suggestions. Funding in direct support of this work: None. Additional revenues related to this work: None. The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper. No third-party support was received during the last 36 months for the work presented, and the authors have no financial relationships outside the submitted work that could be perceived as influencing it.

References

[1] Diyar Akay and Mehmet Atak. Grey prediction with rolling mechanism for electricity demand forecasting of turkey. *energy*, 32(9):1670–1675, 2007.

[2] Wei Fan, Yanjie Fu, Shun Zheng, Jiang Bian, Yuanchun Zhou, and Hui Xiong. DEWP: deep expansion learning for wind power forecasting. *ACM Trans. Knowl. Discov. Data*, 18(3): 71:1–71:21, 2024.

[3] Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Özbayoglu. Financial time series forecasting with deep learning : A systematic literature review: 2005-2019. *Appl. Soft Comput.*, 90:106181, 2020.

[4] Sida Lin, Yankai Chen, Yiyan Qi, Chenhao Ma, Bokai Cao, Yifei Zhang, Xue Liu, and Jian Guo. CSPO: cross-market synergistic stock price movement forecasting with pseudo-volatility optimization. In *Companion Proceedings of the ACM on Web Conference (WWW)*, pages 354–363. ACM, 2025.

[5] Andrea L Bertozzi, Elisa Franco, George Mohler, Martin B Short, and Daniel Sledge. The challenges of modeling and forecasting the spread of covid-19. *Proceedings of the National Academy of Sciences*, 117(29):16732–16738, 2020.

[6] Yihe Wang, Yu Han, Haishuai Wang, and Xiang Zhang. Contrast everything: A hierarchical contrastive framework for medical time-series. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[7] Wanneng Shu, Ken Cai, and Neal Naixue Xiong. A short-term traffic flow prediction model based on an improved gate recurrent unit neural network. *IEEE Trans. Intell. Transp. Syst.*, 23 (9):16654–16665, 2022.

[8] Hao Miao, Yan Zhao, Chenjuan Guo, Bin Yang, Kai Zheng, Feiteng Huang, Jiandong Xie, and Christian S. Jensen. A unified replay-based continuous learning framework for spatio-temporal prediction on streaming data. In *40th IEEE International Conference on Data Engineering (ICDE)*, pages 1050–1062, 2024.

[9] 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 *The Tenth International Conference on Learning Representations (ICLR)*, 2022.

[10] Peiyuan Liu, Beiliang Wu, Yifan Hu, Naiqi Li, Tao Dai, Jigang Bao, and Shu-Tao Xia. Timebridge: Non-stationarity matters for long-term time series forecasting. In *Forty-second International Conference on Machine Learning (ICML)*, 2025.

[11] Zhiding Liu, Mingyue Cheng, Zhi Li, Zhenya Huang, Qi Liu, Yanhu Xie, and Enhong Chen. Adaptive normalization for non-stationary time series forecasting: A temporal slice perspective. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[12] Tao Dai, Beiliang Wu, Peiyuan Liu, Naiqi Li, Xue Yuerong, Shu-Tao Xia, and Zexuan Zhu. DDN: dual-domain dynamic normalization for non-stationary time series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[13] Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[14] Rui Wang, Yihe Dong, Sercan Ö. Arik, and Rose Yu. Koopman neural operator forecaster for time-series with temporal distributional shifts. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

[15] Hans Roder. Amplitude, phase, and frequency modulation. *Proceedings of the Institute of Radio Engineers*, 19(12):2145–2176, 2006.

[16] 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 (SIGIR)*, pages 95–104, 2018.

[17] Shengsheng Lin, Weiwei Lin, Wentai Wu, Feiyu Zhao, Ruichao Mo, and Haotong Zhang. Segrnn: Segment recurrent neural network for long-term time series forecasting. *arXiv*, abs/2308.11200, 2023.

[18] 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. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[19] Donghao Luo and Xue Wang. Moderntcn: A modern pure convolution structure for general time series analysis. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[20] 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 *Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, pages 11106–11115, 2021.

[21] 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 (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pages 27268–27286, 2022.

[22] 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 (KDD)*, pages 2400–2410, 2024.

[23] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI)*, pages 11121–11128, 2023.

[24] 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. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[25] 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. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[26] Wanlin Cai, Yuxuan Liang, Xianggen Liu, Jianshuai Feng, and Yuankai Wu. Msgnet: Learning multi-scale inter-series correlations for multivariate time series forecasting. In *Thirty-Eighth AAAI Conference on Artificial Intelligence (AAAI)*, pages 11141–11149, 2024.

[27] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv*, abs/2312.00752, 2023.

[28] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[29] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

[30] 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. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[31] Robert B Cleveland, William S Cleveland, Jean E McRae, Irma Terpenning, et al. Stl: A seasonal-trend decomposition. *J. off. Stat*, 6(1):3–73, 1990.

[32] Tao Dai, Beiliang Wu, Peiyuan Liu, Naiqi Li, Jigang Bao, Yong Jiang, and Shu-Tao Xia. Periodicity decoupling framework for long-term series forecasting. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[33] Shengsheng Lin, Weiwei Lin, Wentai Wu, Haojun Chen, and Junjie Yang. Sparsetsf: Modeling long-term time series forecasting with *1k* parameters. In *Forty-first International Conference on Machine Learning (ICML)*, 2024.

[34] Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. Cyclenet: Enhancing time series forecasting through modeling periodic patterns. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[35] A Bloch. Modulation theory. *Journal of the Institution of Electrical Engineers-Part III: Communication Engineering*, 91(13):31–42, 1944.

[36] Shengsheng Lin, Haojun Chen, Haijie Wu, Chunyun Qiu, and Weiwei Lin. Temporal query network for efficient multivariate time series forecasting. In *Forty-second International Conference on Machine Learning (ICML)*, 2025.

[37] Henrik Madsen. *Time series analysis*. Chapman and Hall/CRC, 2007.

[38] 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*, abs/2510.00461, 2025.

[39] Yulong Wang, Yushuo Liu, Xiaoyi Duan, and Kai Wang. Filterts: Comprehensive frequency filtering for multivariate time series forecasting. In *The Association for the Advancement of Artificial Intelligence (AAAI)*, pages 21375–21383, 2025.

[40] Jingru Fei, Kun Yi, Wei Fan, Qi Zhang, and Zhendong Niu. Amplifier: Bringing attention to neglected low-energy components in time series forecasting. In *The Association for the Advancement of Artificial Intelligence (AAAI)*, pages 11645–11653, 2025.

[41] 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. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[42] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[43] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

[44] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 22419–22430, 2021.

[45] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, and et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8024–8035, 2019.

[46] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR)*, 2015.

[47] Hao Wang, Lichen Pan, Yuan Shen, Zhichao Chen, Degui Yang, Yifei Yang, Sen Zhang, Xinggao Liu, Haoxuan Li, and Dacheng Tao. Fredf: Learning to forecast in the frequency domain. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

[48] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[49] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *The 22nd Conference on Learning Theory (COLT)*, 2009.

[50] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Mach. Learn.*, 79(1-2): 151–175, 2010.

A Performance Guarantee of PAMod

In this section, we provide a formal theoretical justification for the Phase-Amplitude Modulation (PAMod) framework. We demonstrate that by conditioning the model on cyclical indices, PAMod minimizes the generalization error bound under periodic non-stationarity.

A.1 Discrepancy under Domain Shift

To analyze the shift between different cyclical distributions, we adopt the Discrepancy Distance (d_{disc}), which generalizes the $\mathcal{H}\Delta\mathcal{H}$-divergence from binary classification to continuous output spaces and general loss functions [49].

Definition A.1. (Discrepancy Distance). Let $\mathcal{H}$ be a hypothesis class of functions mapping $\mathcal{X} \to \mathcal{Y}$. The discrepancy between two distributions $\mathcal{D}_i$ and $\mathcal{D}_j$ is defined as:

$$d_{\text{disc}}(\mathcal{D}_i, \mathcal{D}_j) = \sup_{h,h' \in \mathcal{H}} \left| \mathbb{E}_{x \sim \mathcal{D}_i} \mathcal{L}(h(x), h'(x)) - \mathbb{E}_{x \sim \mathcal{D}_j} \mathcal{L}(h(x), h'(x)) \right|, \tag{18}$$

where $\mathcal{L}$ is the loss function (e.g., Mean Squared Error). This metric is particularly suited for our framework as it captures the variations in both mean (phase) and variance (amplitude) within the hypothesis space.

A.2 Generalization Bound under Periodic Shift

We assume the time series follows the periodic location-scale shift defined in Equation (14): $X_t = \mu(\tau) + \sigma(\tau) \odot Z_t$, where $\tau = t \bmod L$ and Z_t is a cycle-invariant latent process. Let $\psi(X, \tau)$ denote the PAMod operator, and $\hat{\mathcal{D}}_\tau$ be the distribution of the modulated features.

Theorem A.2. ***(Generalization Bound [49, 50])****. For any cyclical position $\tau \in \{0, \ldots, L-1\}$, the expected risk $\mathcal{R}_\tau(h)$ of the predictor is bounded by:*

$$\mathcal{R}_\tau(h) \leq \hat{\mathcal{R}}_S(h) + d_{\textit{disc}}(\hat{\mathcal{D}}_\tau, \mathcal{D}_{ref}) + \lambda, \tag{19}$$

where $\hat{\mathcal{R}}_S(h)$ is the empirical risk on the training set, $d_{\textit{disc}}(\hat{\mathcal{D}}_\tau, \mathcal{D}_{ref})$ is the discrepancy between the modulated distribution $\hat{\mathcal{D}}_\tau$ and a stationary reference distribution $\mathcal{D}_{ref}$ (the distribution of Z), and λ is the error of the ideal joint hypothesis.

Proof Sketch: By the triangle inequality of the discrepancy distance, the error on a target distribution $\mathcal{D}_\tau$ can be decomposed into the source error and the distance between the two distributions. PAMod implements a transformation $\psi(X, \tau) = X \odot E_A(\tau) + E_P(\tau)$. Under the assumption in Eq. (14), if the model learns $E_A(\tau) \approx \sigma(\tau)^{-1}$ and $E_P(\tau) \approx -\mu(\tau)\sigma(\tau)^{-1}$, the transformed distribution $\hat{\mathcal{D}}_\tau$ converges to the stationary distribution $\mathcal{D}_{ref}$. Consequently:

$$\inf_{E_A, E_P} d_{\text{disc}}(\psi(\mathcal{D}_\tau), \mathcal{D}_{ref}) \to 0. \tag{20}$$

Thus, PAMod minimizes the upper bound of the generalization error by explicitly neutralizing the cycle-dependent distribution discrepancy, which remains a large constant in models without cycle-awareness.

A.3 Lipschitz Continuity and Stability

To ensure that the modulation does not amplify input perturbations, we provide a stability guarantee.

Lemma A.3. ***(Lipschitz Continuity of PAMod)****. The modulation operator $\psi(X, \tau)$ is Lipschitz continuous with respect to X. Specifically, if the linear projections U_i have bounded weights $\|U\| \leq W_{\max}$, then:*

$$\|\psi(X_1, \tau) - \psi(X_2, \tau)\| \leq K \|X_1 - X_2\|, \tag{21}$$

where K depends on $W_{\max}$ and the norm of the embeddings.

Discussion: This continuity ensures that the learnable cyclical embeddings do not introduce high-frequency noise or instability into the latent space. By combining the Discrepancy Minimization (Theorem A.2.) and Lipschitz Stability (Lemma A.3.), PAMod provides a robust performance guarantee for long-term time series forecasting even under severe periodic distribution drifts.

B More Details of PAMod

B.1 Framework Implementation

The pseudocode of PAMod is presented in Algorithm 1, where the key design lies in our phase-amplitude modulation.

Algorithm 1 Pseudocode of PAMod.

Input: Lookback length $\mathbf{X} \in \mathbb{R}^{T \times C}$, cycle length L
Output: The prediction horizon $\hat{\mathbf{X}} \in \mathbb{R}^{H \times C}$

Initialize learnable embedding matrix $\Omega_p \in \mathbb{R}^{c \times T}$ and $\Omega_a \in \mathbb{R}^{L \times CT}$ with the Xavier normal distribution
if Instance Normalization is True **then**
  $\mu, \sigma \leftarrow Mean(X), STD(X)$
  $X_{\text{norm}} \leftarrow \frac{X-\mu}{\sqrt{\sigma^2+\epsilon}}$
end if
▷ Cycle Index
$T_{end} \leftarrow$ time stamp of the last observed step
$\tau = T_{end} \mod L$
▷ Phase Embedding
for $i \in \{1, \cdots, C\}$ **do**
  $E_P^i(t) \leftarrow \text{Lookup}(W_m, t)$ ▷ $E_P^i(t) \in \mathbb{R}^{1 \times T}$
end for
$E_P(t) \leftarrow \text{contact}(E_p^1, \cdots, E_p^C)$ ▷ $E_P(t) \in \mathbb{R}^{C \times T}$
▷ Amplitude Embedding
$E_A(t) \leftarrow \text{Reshape}(\text{Lookup}(W_v, t)$ ▷ $E_A(t) \in \mathbb{R}^{C \times T}$
▷ Phase-Amplitude Modulation
$X_1 = \text{ReLU}(U_1 X_{\text{norm}})$ ▷ $X_1 \in \mathbb{R}^{C \times T}$
$X_{\text{mod}} = \text{Dropout}(U_4(X_1 \odot U_2 E_A(t) + U_3 E_P(t)))$ ▷ $X_{\text{mod}} \in \mathbb{R}^{C \times T}$
▷ Forecasting Head
$\hat{Y} = \text{MLP}(X_{\text{mod}} + X_{\text{norm}})$
$\hat{\mathbf{Y}} = \hat{\mathbf{Y}}.transpose(-1, -2)$ ▷ $\hat{\mathbf{Y}} \in \mathbb{R}^{H \times C}$
if Instance Normalization is True **then**
  $\hat{Y} \leftarrow \hat{Y} \times \sqrt{\sigma^2 + \epsilon} + \mu$
end if

B.2 Benchmarks details

We evaluate the performance of PAMod compared with various baselines on 12 well-established benchmarks [2], which are detailed in Table 5.

- **ETT (Electricity Transformer Temperature)** comprises two hourly-level datasets (i.e., ETTh1 and ETTh2) and two 15-minute-level datasets (i.e., ETTm1 and ETTm2). Each dataset contains seven oil and load features of electricity transformers from July 2016 to July 2018.
- **Electricity** encompasses the hourly electricity consumption data of 321 customers from 2012 to 2014.
- **Traffic** describes the road occupancy rates from the California Department of Transportation. It contains the hourly data recorded by the sensors of San Francisco freeways from 2015 to 2016.
- **Solar** records the solar power production of 137 PV plants in 2006, which is sampled every 10 minutes.
- **Weather** includes 21 indicators of weather, such as air temperature, and humidity. Its data is recorded every 10 min for 2020 in Germany.
- **PEMS** contains public traffic network data in California collected by 5-minute windows.

[2] All the datasets are publicly available at `https://github.com/thuml/iTransformer`

We follow the same data processing and train-validation-test set split protocol of TimesNet [29], where each part is strictly divided according to chronological order without data leakage issues.

Table 5: Detailed descriptions of datasets. Channel denotes the number of variates in each dataset. Prediction length points out four prediction settings. The dataset size is split into (Train, Validation, Test). Frequency denotes the sampling interval of time points.

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

B.3 Experimental Details

PAMod is trained for 30 epochs with early stopping based on a patience of 5 on the validation set. During the training process, the learning rate is set to $[1 \times 10^{-3}, 5 \times 10^{-3}]$, the embedding dimension is set to 512, and the dropout rate is set to 0.5 by default. To improve training stability, we adopt a hybrid MAE loss [47] that operates in both the time and frequency domains:

$$\begin{gathered} \mathcal{L}_t = \frac{1}{H}\sum_{i=1}^{H}|Y - \hat{Y}|,\ \mathcal{L}_f = \frac{1}{H}\sum_{i=1}^{H}|\mathcal{F}(Y) - \mathcal{F}(\hat{Y})|, \\ \mathcal{L} = (1-\alpha) \times \mathcal{L}_t + \alpha \times \mathcal{L}_f, \end{gathered} \tag{22}$$

where $\mathcal{F}$ denotes the Fast Fourier Transform, and the hyperparameter α is set to $[0.05, 0.35]$ for different datasets. We use Mean Squared Error (MSE) and Mean Absolute Error (MAE) as evaluation metrics. Given the ground truth values $Y \in \mathbb{R}^{H \times C}$ and the predicted values $\hat{Y} \in \mathbb{R}^{H \times C}$, these metrics are defined as:

$$\text{MSE} = \frac{1}{H}\sum_{i=1}^{H}(Y - \hat{Y})^2, \quad \text{MAE} = \frac{1}{H}\sum_{i=1}^{H}|Y - \hat{Y}|. \tag{23}$$

C More Results of PAMod

C.1 Error Bars

We obtain the standard deviation of PAMod performance by training the model with 5 different random seeds over 12 datasets. As seen in Table 6, the error bars of all the results are tiny, indicating our PAMod is robust and reliable.

C.2 Extra Quantitative Results

Table 7 shows more compatibility analysis on the ETTh1 and ETTm2 datasets. Table 8 presents the full comparison results of PAMod against nine baseline methods across 12 real-world multivariate datasets. The results demonstrate that PAMod consistently achieves state-of-the-art forecasting performance under most experimental settings, underscoring the effectiveness of the proposed approach. In Table 9, we randomly mask the input values using zeros or Gaussian noise $\mathcal{N}(0, \sigma^2)$ with probability $p \in \{0.1, 0.3\}$ to evaluate the robustness of PAMod, where PAMod outperforms all the baselines under both zero-imputation and noise-corruption scenarios.

C.3 Extra Visual Cases

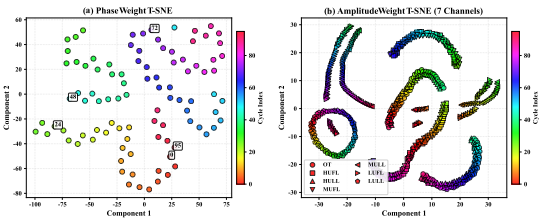
Here, we provide more showcases for visualization in Figure 8. In subfigure 9(a), we display the phase and amplitude embedding weights to illustrate their respective roles in modeling mean and variance

Table 6: Robustness of PAMod performance obtained from 5 random seeds on 12 benchmarks. D and M denote Dataset and Metric, respectively.

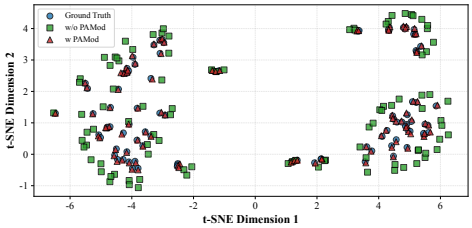
D	ETTm1		ETTm2		ETTh1		ETTh2	
M	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
96	0.297 ± 0.001	0.337 ± 0.001	0.162 ± 0.000	0.240 ± 0.001	0.357 ± 0.001	0.382 ± 0.000	0.279 ± 0.000	0.328 ± 0.000
192	0.346 ± 0.001	0.366 ± 0.001	0.226 ± 0.001	0.283 ± 0.000	0.403 ± 0.002	0.413 ± 0.001	0.349 ± 0.001	0.374 ± 0.001
336	0.375 ± 0.001	0.387 ± 0.002	0.283 ± 0.001	0.321 ± 0.001	0.443 ± 0.004	0.434 ± 0.002	0.403 ± 0.002	0.414 ± 0.001
720	0.432 ± 0.002	0.422 ± 0.001	0.382 ± 0.002	0.382 ± 0.001	0.447 ± 0.003	0.452 ± 0.001	0.409 ± 0.002	0.432 ± 0.002
D	ECL		Traffic		Weather		Solar	
M	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
96	0.136 ± 0.001	0.226 ± 0.000	0.409 ± 0.001	0.256 ± 0.001	0.153 ± 0.002	0.191 ± 0.002	0.184 ± 0.000	0.209 ± 0.000
192	0.151 ± 0.002	0.240 ± 0.001	0.424 ± 0.002	0.268 ± 0.001	0.201 ± 0.001	0.237 ± 0.001	0.205 ± 0.001	0.226 ± 0.001
336	0.168 ± 0.003	0.258 ± 0.003	0.436 ± 0.002	0.274 ± 0.001	0.259 ± 0.000	0.281 ± 0.001	0.215 ± 0.002	0.232 ± 0.001
720	0.205 ± 0.001	0.290 ± 0.002	0.465 ± 0.002	0.291 ± 0.003	0.341 ± 0.001	0.336 ± 0.002	0.216 ± 0.002	0.238 ± 0.002
D	PEMS03		PEMS04		PEMS07		PEMS08	
M	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
12	0.059 ± 0.000	0.157 ± 0.000	0.066 ± 0.001	0.162 ± 0.001	0.053 ± 0.000	0.141 ± 0.000	0.074 ± 0.000	0.161 ± 0.001
24	0.072 ± 0.000	0.173 ± 0.000	0.075 ± 0.001	0.174 ± 0.000	0.065 ± 0.000	0.156 ± 0.000	0.094 ± 0.000	0.177 ± 0.000
48	0.098 ± 0.001	0.199 ± 0.001	0.091 ± 0.002	0.192 ± 0.001	0.086 ± 0.001	0.176 ± 0.001	0.126 ± 0.001	0.199 ± 0.001
96	0.133 ± 0.002	0.228 ± 0.001	0.111 ± 0.001	0.212 ± 0.001	0.116 ± 0.001	0.199 ± 0.001	0.181 ± 0.002	0.223 ± 0.001

Table 7: Performance comparison of integrating our PAMod mechanism with different backbone models on the ETTh1 and ETTm2 datasets. The best results are highlighted in **bold**, and "Imp." denotes the performance improvement achieved by incorporating PAMod.

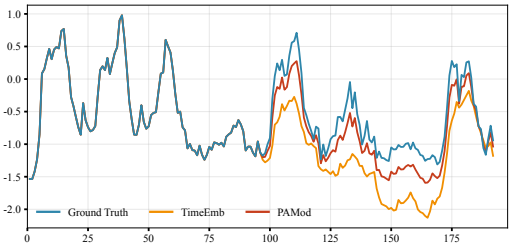
Dataset	ETTh1								ETTm2							
Horizon	96		192		336		720		96		192		336		720	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
DLinear [2023]	0.386	0.400	0.437	0.432	0.481	0.459	0.519	0.516	0.193	0.292	0.284	0.362	0.369	0.427	0.554	0.522
+PAMod	**0.375**	**0.389**	**0.423**	**0.425**	**0.462**	**0.442**	**0.476**	**0.458**	**0.177**	**0.255**	**0.239**	**0.308**	**0.308**	**0.346**	**0.422**	**0.413**
Imp.	2.8%	2.8%	3.2%	1.6%	4.0%	3.7%	8.3%	11.2%	8.3%	12.7%	15.8%	14.9%	16.5%	19.0%	23.8%	20.9%
PatchTST [2023]	0.414	0.419	0.460	0.445	0.501	0.466	0.500	0.488	0.175	0.259	0.241	0.302	0.305	0.343	0.402	0.400
+PAMod	**0.396**	**0.392**	**0.438**	**0.427**	**0.469**	**0.445**	**0.473**	**0.461**	**0.170**	**0.252**	**0.235**	**0.296**	**0.299**	**0.337**	**0.389**	**0.388**
Imp.	4.3%	6.4%	4.8%	4.0%	6.4%	4.5%	5.4%	5.5%	2.9%	2.7%	2.5%	2.0%	2.0%	1.7%	3.2%	3.0%
iTransformer [2024]	0.386	0.405	0.441	0.436	0.487	0.458	0.503	0.491	0.180	0.264	0.250	0.309	0.311	0.348	0.412	0.407
+PAMod	**0.372**	**0.393**	**0.425**	**0.421**	**0.464**	**0.443**	**0.484**	**0.469**	**0.169**	**0.252**	**0.238**	**0.292**	**0.294**	**0.332**	**0.385**	**0.384**
Imp.	3.6%	3.0%	3.6%	3.4%	4.7%	3.3%	3.8%	4.5%	6.1%	4.5%	4.8%	5.5%	5.5%	4.6%	6.6%	5.7%
TQNet [2025]	0.371	0.393	0.428	0.426	0.476	0.446	0.487	0.470	0.173	0.256	0.238	0.298	0.301	0.340	0.397	0.396
+PAMod	**0.369**	**0.387**	**0.418**	**0.417**	**0.457**	**0.436**	**0.453**	**0.455**	**0.165**	**0.242**	**0.226**	**0.284**	**0.285**	**0.323**	**0.377**	**0.380**
Imp.	0.5%	1.5%	2.3%	2.1%	4.0%	2.2%	7.0%	3.2%	4.6%	5.5%	5.0%	4.7%	5.3%	5.6%	5.0%	4.0%



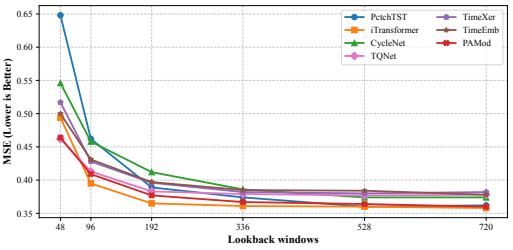
(a) Embedding weights on ETTm1



(b) Distribution alignment on ECL



(c) Forecasting performance on ECL

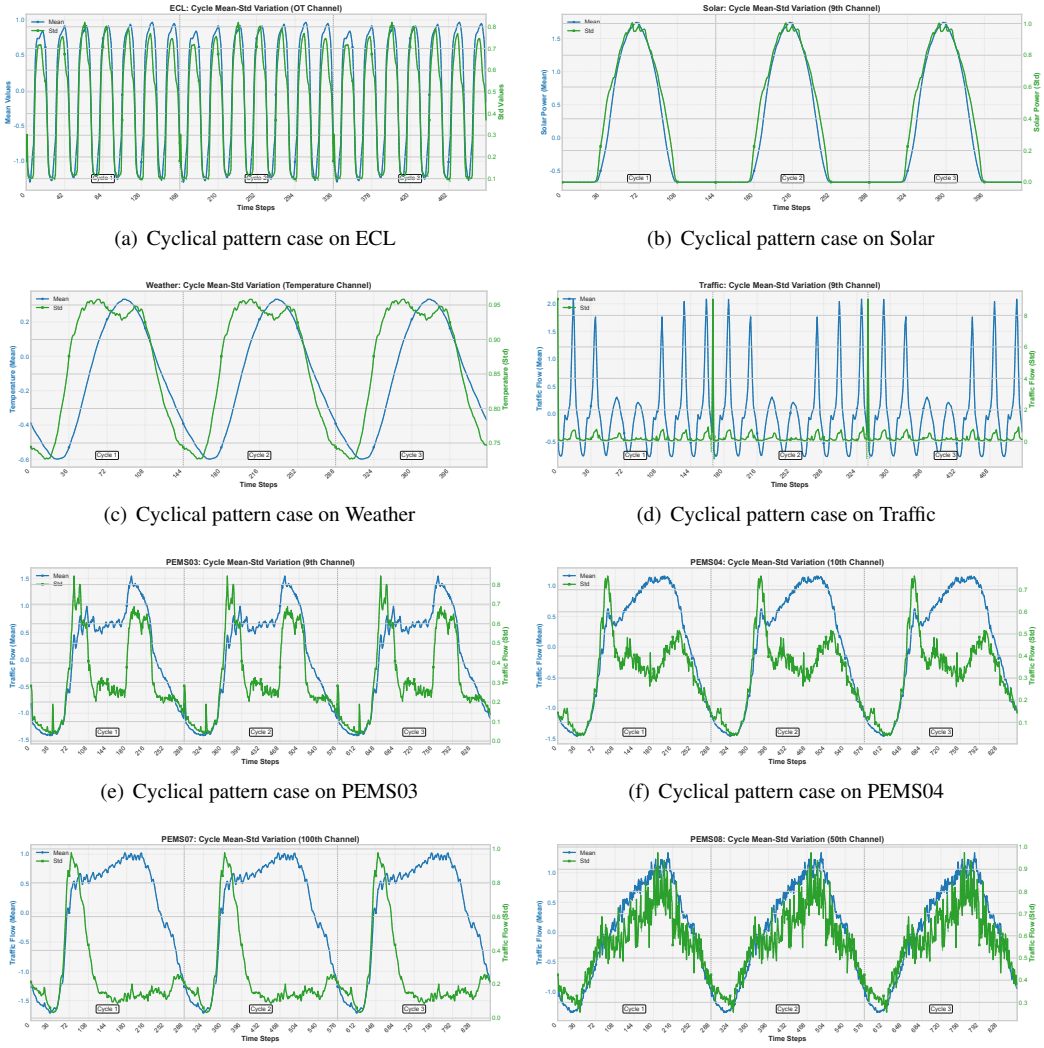


(d) Lookback sensitivity on Traffic

Figure 8: Extra visual cases for PAMod.

shifts. In subfigure 9(b), we show that predictions with PAMod closely overlap the ground truth on ECL, reflecting strong distributional alignment. In subfigure 9(c), we exhibit that PAMod achieves a closer fit to the ground truth curve than TimeEmb on ECL, further demonstrating its superior capacity for modeling temporal dynamics. In subfigure 9(d), as the lookback window increases, PAMod consistently outperforms most baselines, confirming that modeling cyclical shifts enables effective use of longer historical context.

C.4 Cyclical Patterns in Different Datasets



(a) Cyclical pattern case on ECL

(b) Cyclical pattern case on Solar

(c) Cyclical pattern case on Weather

(d) Cyclical pattern case on Traffic

(e) Cyclical pattern case on PEMS03

(f) Cyclical pattern case on PEMS04

(g) Cyclical pattern case on PEMS07

(h) Cyclical pattern case on PEMS08

Figure 9: Different cyclical patterns among different datasets.

As shown in Figure 9, we visualize the cyclical patterns to the other datasets. Similar cyclical patterns emerge consistently: over multiple cycles, both mean and variance components in different datasets exhibit structured periodicity, with their fluctuations evolving synchronously across cycles. These results reinforce our observation that distribution shifts in real-world time series often decompose into cyclical components with correlated positions, rather than arising from random perturbations. The persistence of such patterns across diverse datasets further supports the generality of modeling cyclical non-stationarity via explicit phase-amplitude modulation.

Table 8: Full results of multivariate time series forecasting across 12 benchmarks. The best results are highlighted in **bold**, the second best are underlined, and the **Count** row counts the number of times each model ranks in the top 1.

Model		**PAMod (Ours)**		TQNet [2025]		TimeEmb [2025]		FilterTS [2025]		Amplifier [2025]		TimeXer [2024]		CycleNet [2024]		TimeMixer [2024]		iTransformer [2024]		PatchTST [2023]	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	**0.297**	**0.337**	0.311	0.353	0.305	0.344	0.321	0.360	0.316	0.355	0.318	0.356	0.319	0.360	0.320	0.357	0.334	0.368	0.329	0.367
	192	**0.346**	**0.366**	0.356	0.378	0.354	0.374	0.363	0.382	0.361	0.381	0.362	0.383	0.360	0.381	0.361	0.381	0.377	0.391	0.367	0.385
	336	**0.375**	**0.387**	0.390	0.401	0.380	0.393	0.396	0.404	0.393	0.404	0.395	0.407	0.389	0.403	0.390	0.404	0.426	0.420	0.399	0.410
	720	**0.432**	**0.422**	0.452	0.440	0.434	0.427	0.462	0.438	0.456	0.440	0.452	0.441	0.447	0.441	0.454	0.441	0.491	0.459	0.454	0.439
	Avg	**0.363**	**0.378**	0.377	0.393	0.368	0.385	0.385	0.396	0.382	0.395	0.382	0.397	0.379	0.396	0.381	0.395	0.407	0.410	0.387	0.400
ETTm2	96	**0.162**	**0.240**	0.173	0.256	0.164	0.243	0.173	0.256	0.178	0.261	0.171	0.256	0.163	0.246	0.175	0.258	0.180	0.264	0.175	0.259
	192	**0.226**	**0.283**	0.238	0.298	0.227	0.285	0.238	0.299	0.244	0.304	0.237	0.299	0.229	0.290	0.237	0.299	0.250	0.309	0.241	0.302
	336	**0.283**	**0.321**	0.301	0.340	0.285	0.323	0.300	0.338	0.309	0.346	0.296	0.338	0.284	0.327	0.298	0.340	0.311	0.348	0.305	0.343
	720	**0.382**	**0.382**	0.397	0.396	0.383	**0.382**	0.399	0.395	0.390	0.394	0.392	0.394	0.389	0.391	0.391	0.396	0.412	0.407	0.402	0.400
	Avg	**0.263**	**0.307**	0.277	0.323	0.265	0.308	0.277	0.322	0.280	0.326	0.274	0.322	0.266	0.314	0.275	0.323	0.288	0.332	0.281	0.326
ETTh1	96	**0.357**	**0.382**	0.371	0.393	0.366	0.387	0.375	0.391	0.371	0.392	0.382	0.403	0.375	0.395	0.375	0.400	0.386	0.405	0.414	0.419
	192	**0.403**	**0.413**	0.428	0.426	0.417	0.416	0.424	0.421	0.425	0.422	0.429	0.435	0.436	0.428	0.429	0.421	0.441	0.436	0.460	0.445
	336	**0.443**	**0.434**	0.476	0.446	0.457	0.436	0.465	0.442	0.448	**0.434**	0.468	0.448	0.496	0.455	0.484	0.458	0.487	0.458	0.501	0.466
	720	**0.447**	**0.452**	0.487	0.470	0.459	0.460	0.472	0.466	0.476	0.464	0.469	0.461	0.520	0.484	0.498	0.482	0.503	0.491	0.500	0.488
	Avg	**0.413**	**0.420**	0.441	0.434	0.425	0.425	0.434	0.430	0.430	0.428	0.437	0.437	0.457	0.441	0.447	0.440	0.454	0.448	0.469	0.455
ETTh2	96	0.279	**0.328**	0.295	0.343	**0.277**	**0.328**	0.290	0.338	0.290	0.341	0.286	0.338	0.298	0.344	0.289	0.341	0.297	0.349	0.302	0.348
	192	**0.349**	**0.374**	0.367	0.393	0.354	0.378	0.374	0.390	0.369	0.390	0.363	0.389	0.372	0.396	0.372	0.392	0.380	0.400	0.388	0.400
	336	0.403	**0.414**	0.417	0.427	0.400	0.417	0.415	0.424	0.419	0.431	0.414	0.423	0.431	0.439	**0.386**	**0.414**	0.428	0.432	0.426	0.433
	720	0.409	**0.432**	0.433	0.446	0.416	0.437	0.420	0.438	0.446	0.456	**0.408**	**0.432**	0.450	0.458	0.412	0.434	0.427	0.445	0.431	0.446
	Avg	**0.360**	**0.387**	0.378	0.402	0.362	0.390	0.375	0.398	0.381	0.405	0.368	0.396	0.388	0.409	0.364	0.395	0.383	0.407	0.387	0.407
ECL	96	0.136	**0.226**	**0.134**	0.229	0.137	0.232	0.151	0.245	0.147	0.242	0.140	0.242	0.136	0.229	0.153	0.247	0.148	0.240	0.181	0.270
	192	**0.151**	**0.240**	0.154	0.247	0.154	0.248	0.164	0.256	0.158	0.251	0.157	0.256	0.152	0.244	0.166	0.256	0.162	0.253	0.188	0.274
	336	**0.168**	**0.258**	0.169	0.264	0.171	0.265	0.181	0.274	0.175	0.271	0.176	0.275	0.170	0.264	0.185	0.277	0.178	0.269	0.204	0.293
	720	0.205	**0.290**	**0.201**	0.294	0.209	0.298	0.225	0.311	0.206	0.298	0.211	0.306	0.212	0.299	0.225	0.310	0.225	0.317	0.246	0.324
	Avg	0.165	**0.254**	**0.164**	0.259	0.168	0.261	0.180	0.272	0.172	0.266	0.171	0.270	0.168	0.259	0.182	0.272	0.178	0.270	0.205	0.290
Traffic	96	0.409	**0.256**	0.413	0.261	0.431	0.280	0.446	0.308	0.456	0.299	0.428	0.271	0.458	0.296	0.462	0.285	**0.395**	0.268	0.462	0.290
	192	0.424	**0.268**	0.432	0.271	0.442	0.291	0.456	0.308	0.472	0.318	0.448	0.282	0.457	0.294	0.473	0.296	**0.417**	0.276	0.466	0.290
	336	0.436	**0.274**	0.450	0.277	0.455	0.297	0.472	0.313	0.487	0.320	0.473	0.289	0.470	0.299	0.498	0.296	**0.433**	0.283	0.482	0.300
	720	**0.465**	**0.291**	0.486	0.295	0.484	0.313	0.508	0.332	0.517	0.332	0.516	0.307	0.502	0.314	0.506	0.313	0.467	0.302	0.514	0.320
	Avg	0.434	**0.272**	0.445	0.276	0.453	0.295	0.470	0.315	0.483	0.317	0.466	0.287	0.472	0.314	0.484	0.297	**0.428**	0.282	0.481	0.300
Weather	96	0.153	0.191	0.157	0.200	**0.150**	**0.190**	0.162	0.207	0.167	0.212	0.157	0.205	0.158	0.203	0.163	0.209	0.174	0.214	0.177	0.210
	192	**0.201**	**0.237**	0.206	0.245	**0.201**	0.238	0.209	0.252	0.215	0.251	0.204	0.247	0.207	0.247	0.208	0.250	0.221	0.254	0.225	0.250
	336	**0.259**	**0.281**	0.262	0.287	**0.259**	0.282	0.263	0.292	0.276	0.292	0.261	0.290	0.262	0.289	0.251	0.287	0.278	0.296	0.278	0.290
	720	0.341	**0.336**	0.344	0.342	**0.339**	**0.336**	0.345	0.344	0.352	0.346	0.340	0.341	0.344	0.344	0.339	0.341	0.358	0.349	0.354	0.340
	Avg	0.239	**0.261**	0.242	0.269	**0.237**	0.262	0.245	0.274	0.253	0.275	0.241	0.271	0.243	0.271	0.240	0.271	0.258	0.278	0.259	0.273
Solar	96	0.184	**0.209**	**0.173**	0.233	0.206	0.241	0.196	0.264	0.234	0.283	0.215	0.295	0.190	0.247	0.189	0.259	0.203	0.237	0.234	0.286
	192	0.205	**0.226**	**0.199**	0.257	0.241	0.262	0.211	0.278	0.237	0.259	0.236	0.301	0.210	0.266	0.222	0.283	0.233	0.261	0.267	0.310
	336	0.215	**0.232**	**0.211**	0.263	0.268	0.284	0.226	0.284	0.247	0.269	0.252	0.307	0.217	0.266	0.231	0.292	0.248	0.273	0.290	0.315
	720	0.216	**0.238**	**0.209**	0.270	0.278	0.291	0.227	0.282	0.246	0.270	0.244	0.305	0.223	0.266	0.223	0.285	0.249	0.275	0.289	0.317
	Avg	0.206	**0.226**	**0.198**	0.256	0.248	0.270	0.215	0.277	0.241	0.270	0.237	0.302	0.210	0.261	0.216	0.280	0.233	0.262	0.270	0.307
PEMS03	12	**0.059**	**0.157**	0.060	0.161	0.066	0.168	0.072	0.181	0.070	0.172	0.070	0.173	0.066	0.172	0.076	0.188	0.071	0.174	0.099	0.216
	24	**0.072**	**0.173**	0.077	0.182	0.083	0.189	0.104	0.219	0.090	0.200	0.092	0.194	0.089	0.201	0.113	0.226	0.093	0.201	0.142	0.259
	48	**0.098**	**0.199**	0.104	0.215	0.116	0.220	0.155	0.269	0.147	0.260	0.129	0.229	0.136	0.247	0.191	0.292	0.125	0.236	0.211	0.319
	96	**0.133**	**0.228**	0.148	0.253	0.151	0.251	0.203	0.315	0.217	0.323	0.157	0.261	0.182	0.282	0.288	0.363	0.164	0.275	0.269	0.370
	Avg	**0.091**	**0.189**	0.097	0.203	0.104	0.207	0.134	0.246	0.131	0.239	0.112	0.214	0.118	0.226	0.167	0.267	0.113	0.222	0.180	0.291
PEMS04	12	**0.066**	**0.162**	0.067	0.166	0.071	0.172	0.087	0.199	0.082	0.190	0.074	0.178	0.078	0.186	0.092	0.204	0.078	0.183	0.105	0.224
	24	**0.075**	**0.174**	0.077	0.181	0.083	0.187	0.107	0.223	0.102	0.215	0.087	0.195	0.099	0.212	0.128	0.243	0.095	0.205	0.153	0.275
	48	**0.091**	**0.192**	0.097	0.206	0.104	0.210	0.138	0.255	0.151	0.269	0.110	0.214	0.133	0.248	0.213	0.315	0.120	0.233	0.229	0.339
	96	**0.111**	**0.212**	0.123	0.233	0.126	0.231	0.166	0.285	0.205	0.320	0.148	0.251	0.167	0.281	0.307	0.384	0.150	0.262	0.291	0.389
	Avg	**0.086**	**0.185**	0.091	0.197	0.096	0.200	0.125	0.241	0.135	0.249	0.105	0.209	0.119	0.232	0.185	0.287	0.111	0.221	0.195	0.307
PEMS07	12	0.053	**0.141**	**0.051**	0.143	0.057	0.151	0.067	0.170	0.079	0.179	0.057	0.152	0.062	0.162	0.073	0.184	0.067	0.265	0.095	0.207
	24	0.065	**0.156**	**0.063**	0.159	0.075	0.171	0.095	0.203	0.094	0.196	0.079	0.179	0.086	0.192	0.111	0.219	0.088	0.190	0.150	0.262
	48	0.086	**0.176**	**0.081**	0.179	0.106	0.200	0.146	0.239	0.129	0.237	0.099	0.191	0.128	0.234	0.237	0.328	0.110	0.215	0.253	0.340
	96	0.116	**0.199**	**0.103**	0.203	0.148	0.231	0.172	0.266	0.185	0.291	0.107	0.205	0.176	0.268	0.303	0.354	0.139	0.245	0.346	0.404
	Avg	0.080	**0.168**	**0.075**	0.171	0.097	0.188	0.120	0.220	0.122	0.226	0.085	0.182	0.113	0.214	0.181	0.271	0.101	0.204	0.211	0.303
PEMS08	12	0.074	**0.161**	**0.071**	0.170	0.087	0.188	0.086	0.194	0.079	0.182	0.075	0.176	0.082	0.185	0.091	0.201	0.079	0.182	0.168	0.232
	24	**0.094**	**0.177**	0.096	0.196	0.106	0.198	0.126	0.236	0.115	0.218	0.102	0.201	0.117	0.226	0.137	0.246	0.115	0.219	0.224	0.281
	48	**0.126**	**0.199**	0.149	0.244	0.158	0.253	0.223	0.320	0.192	0.294	0.158	0.248	0.169	0.268	0.265	0.343	0.186	0.235	0.321	0.354
	96	0.181	**0.223**	0.253	0.309	**0.177**	0.246	0.284	0.315	0.346	0.390	0.366	0.377	0.233	0.306	0.410	0.407	0.221	0.267	0.408	0.417
	Avg	**0.119**	**0.190**	0.142	0.229	0.132	0.221	0.180	0.266	0.183	0.271	0.175	0.250	0.150	0.246	0.226	0.299	0.150	0.226	0.280	0.321
1st Count		**35**	**59**	14	0	7	4	0	0	0	1	1	1	0	0	1	1	4	0	0	0

Table 9: Performance under missing value scenarios on ECL.

ECL		Standard		Zeros				Noise			
				p=0.1		p=0.3		p=0.1		p=0.3	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
iTransformer	96	0.148	0.240	0.157	0.253	0.195	0.298	0.157	0.254	0.193	0.295
	192	0.162	0.253	0.171	0.265	0.208	0.308	0.171	0.267	0.203	0.305
	336	0.178	0.269	0.183	0.278	0.221	0.319	0.190	0.289	0.221	0.325
	720	0.225	0.317	0.228	0.319	0.256	0.344	0.227	0.322	0.262	0.355
	Avg	0.178	0.270	0.185	0.279	0.220	0.317	0.186	0.283	0.220	0.320
TimeMixer	96	0.153	0.247	0.155	0.259	0.252	0.370	0.157	0.261	0.250	0.354
	192	0.166	0.256	0.172	0.273	0.240	0.346	0.170	0.271	0.278	0.375
	336	0.185	0.277	0.189	0.290	0.327	0.420	0.191	0.293	0.237	0.342
	720	0.225	0.310	0.234	0.325	0.323	0.416	0.236	0.328	0.279	0.375
	Avg	0.182	0.272	0.188	0.287	0.286	0.388	0.189	0.288	0.261	0.362
TQNet	96	0.134	0.229	0.138	0.235	0.155	0.257	0.137	0.233	0.156	0.256
	192	0.154	0.247	0.156	0.254	0.172	0.271	0.158	0.255	0.176	0.274
	336	0.169	0.264	0.172	0.269	0.189	0.289	0.174	0.269	0.196	0.294
	720	0.201	0.294	0.214	0.306	0.235	0.328	0.210	0.301	0.242	0.333
	Avg	0.164	0.259	0.170	0.266	0.188	0.286	0.170	0.265	0.193	0.289
PAMod	96	0.136	0.226	0.139	0.231	0.142	0.236	0.138	0.231	0.147	0.243
	192	0.151	0.240	0.154	0.247	0.158	0.253	0.155	0.248	0.165	0.261
	336	0.168	0.258	0.170	0.263	0.175	0.270	0.172	0.266	0.182	0.279
	720	0.205	0.290	0.208	0.295	0.213	0.303	0.210	0.298	0.220	0.311
	Avg	0.165	0.254	0.168	0.259	0.172	0.266	0.169	0.261	0.179	0.274

D Discussion

PAMod as Dynamic Denormalization. We clarify the transition from Equation 16 to Equation 17 via Effective Feature Re-parameterization and Scale-Equivariant Propagation. Recall that intermediate amplitude modulation is $X_1 \odot \alpha(\tau)$, where $X_1 = \mathrm{ReLU}(U_1 X_{\mathrm{norm}})$. Since $\alpha(\tau)$ is an input-independent parameter vector conditioned strictly on temporal index τ, its scale-modulating effect across base sequence X_{norm} can be effectively re-parameterized via implicit function $g_\theta(\alpha(\tau))$, subsuming average projection and non-linear scaling across feature dimensions: $X_1 \odot \alpha(\tau) = X_{\mathrm{norm}} \odot g_\theta(\alpha(\tau))$. Thus, total modulated input $X_{\mathrm{input}} = X_{\mathrm{norm}} + X_{\mathrm{Mod}}$ become:

$$X_{\mathrm{norm}} + X_{\mathrm{mod}} = X_{\mathrm{norm}} \odot (1 + g_\theta(\alpha(\tau))) + \phi(\tau). \tag{24}$$

Let $\alpha_\theta(\tau) = g_\theta(\alpha(\tau))$, the forecasting output before baseline shift addition is:

$$\hat{Y} = \sigma_X \cdot f_\theta(X_{\mathrm{norm}} \odot (1 + \alpha_\theta(\tau)) + \phi(\tau)). \tag{25}$$

Since $\alpha_\theta(\tau)$ and $\phi(\tau)$ represent input-independent temporal modulations, their spatial propagation through the feature extractor f_θ naturally induces multiplicative scaling and additive shifting on $f_\theta(X_{\mathrm{norm}})$, yielding Equation 17. Then, the dynamic denormalization can be formalized as (μ_d, σ_d), where $\mu_d = \sigma_X \cdot \phi(\tau) + \mu_X$ and $\sigma_d = \sigma_X \cdot (1 + \alpha_\theta(\tau))$. The theoretical rationale is as follows:

- **Effective Re-parameterization**. Subsuming layer-wise non-linear projections into an effective gain function $g_\theta(\alpha(\tau))$ aligns with continuous factor re-parameterization techniques in feature learning.
- **Local Scale-Equivariance**. Deep networks operating on RevIN-normalized inputs exhibit local smoothness along feature trajectories. Under bounded input shifts, the primary response to scaling factor $(1 + \alpha_\theta(\tau))$ is dominated by output-side multiplicative scaling on $f_\theta(X_{\mathrm{norm}})$, while non-linearities (e.g., GeLU) refine high-frequency spatial details.
- **Shift Equivariance**. Input-independent bias $\phi(\tau)$ propagates through f_θ as a baseline offset accumulator, resulting in an additive output shift.

Differentiation from CycleNet [34], SAN [11], and DDN [12]. **CycleNet** learns a recurrent cycle prototype as an additive template at the time-step level, which treats periodicity as a whole and does not decouple mean and variance shifts. **SAN** divides the input into local time slices to model the evolution of statistical properties, while **DDN** performs normalization via sliding windows in time and frequency domains. Both adapt to general non-stationarity but do not explicitly exploit

periodic structure. **PAMod** differs in both "what" and "where": It operates within the normalized feature space, inserted between RevIN normalization and the prediction head. Rather than estimating statistics directly, PAMod structurally modulates the normalized features X_{norm} via cycle-indexed phase (mean) and amplitude (variance) embeddings.

Potential limitations. While PAMod demonstrates strong performance in forecasting tasks with cyclical distribution shifts, several limitations merit discussion:

- **Fixed cycle assumption**. PAMod relies on a predefined cycle length τ, which may not adapt well to series with varying or unknown periodicities (e.g., irregular event-driven patterns or multi-frequency cycles without a dominant period).
- **Channel-common cycle**. The current design applies the same cycle length to all channels, which could be suboptimal when different variables exhibit distinct periodic behaviors (e.g., temperature with daily cycles and electricity load with weekly cycles). A channel-adaptive cycle mechanism could offer a more tailored solution.
- **Sensitivity to cycle misalignment**. If the assumed cycle length significantly deviates from the true data periodicity, the phase and amplitude embeddings may fail to capture the actual distribution structure, leading to reduced robustness in shift modeling.
- **Limited to cyclical shifts**. PAMod is designed to model cyclically structured distribution shifts, but may not effectively handle abrupt, non-cyclical regime changes or anomalies that fall outside learned periodic patterns.

Future Directions. Several directions remain open for further extending PAMod:

- **Adaptive cycle learning**. Instead of using a fixed τ, future versions could learn or dynamically adjust the cycle length from data, e.g., via periodicity estimation modules or multi-scale cycle banks.
- **Channel-wise cycle modeling**. Incorporating variable-specific cycle embeddings would allow PAMod to better handle multivariate series where different channels follow distinct periodic regimes. One potential extension is to assign each channel its own learnable cycle length $[c_1, c_2, \cdots, c_N]$, then compute the least common multiple (LCM) across all channels to expand each channel's embeddings to a unified temporal resolution, and finally merge them along the channel dimension. This allows the model to accommodate heterogeneous periodicities while maintaining computational regularity.
- **higher-order dynamics modeling**. (1) phase coupling via inter-harmonic embedding interactions $E_P^{(k)} \otimes E_P^{(m)}$ to capture non-linear phase coupling associated with 3rd-order cumulants (bispectrum); (2) polynomial envelopes via higher-order expansions $\sum_{w_k}(X \odot E_A)^k$ to model heavy-tailed distributions across cycles.
- **Integration with structural dependencies**. Although PAMod focuses on temporal cyclicity, combining it with inter-variable relationship modeling (e.g., graph-based or attention-based channel interaction) could further improve performance in spatio-temporal or highly correlated multivariate settings.
- **Generalization to other non-stationary tasks**. Exploring PAMod's adaptation to other domains with structured distribution shifts—such as financial volatility, energy load forecasting with weather shocks, or healthcare monitoring with seasonal effects—would test its broader utility.

In summary, while PAMod provides an effective framework for cyclical distribution modeling, further advances in adaptive cycle inference, channel-aware design, and integration with relational modeling could enhance its flexibility and scope.

E Broader Impacts

This work improves time series forecasting via learnable cyclical distribution shift modeling, with potential benefits for energy, transportation, and climate applications. We foresee no specific negative societal impacts beyond those common to forecasting models, and the method incurs negligible computational overhead.

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 in the abstract accurately reflect our contributions.

   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 limitations of our work in Appendix C.

   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: [N/A]

Justification: The paper does not include theoretical results.

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 provide complete experimental details in Section 4 and Appendix A. Additionally, we have shared the full reproducible code in the 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 shared the full reproducible code in the Supplementary Material.

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: We describe the complete experimental details and hyperparameter choices in Section 4 and Appendix A.

   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 report the error bars of PAMNet in Table 6.

   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 report the computational resource requirements of our proposed method in Section 4.

   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: [Yes]

    Justification: We have discussed the broader impacts in Appendix D.

    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 poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: 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: The paper does not release new assets.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - 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: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

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.