PHIMIXER: FORECAST-ORIGIN PHASE CONDITIONING FOR LONG-TERM TIME-SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multivariate time-series forecasting involves temporal and cross-channel dependencies whose dynamics can vary across recurring cycles. We investigate whether *forecast-origin phase* can serve as a conditioning signal for adapting forecasting transformations to the underlying rhythmic state. We introduce **PhiMixer**, a lightweight phase-conditioned architecture that separates learnable multi-period cyclic components from the input and uses forecast-origin phase to modulate temporal and cross-channel processing. Its rhythm-aware relative attention (RRF) captures periodic temporal structure, while the *Phase-Informed Low-Rank Channel Mixer* (PILCM) generates phase-dependent low-rank channel transformations without dense $\mathcal{O}(C^2)$ interactions. PhiMixer achieves the best performance on 6/8 datasets on MSE/MAE, respectively, among the compared methods. Ablation studies show that active phase conditioning provides consistent gains, particularly on periodic traffic data, with conditioning consistently outperforming phase-as-feature concatenation. With approximately 126K parameters and linear channel scaling, PhiMixer offers an efficient approach to phase-aware long-term multivariate forecasting.

1 INTRODUCTION

Long-term multivariate time-series forecasting requires a model to capture dependencies across both time and variables. This is challenging because real-world time series often exhibit heterogeneous temporal dynamics, cross-variable dependencies, trends, and recurring patterns at multiple time scales. For example, electricity demand and traffic measurements contain strong daily and weekly rhythms, while individual variables may respond differently to the same underlying temporal conditions. An effective forecasting model must therefore capture temporal structure and cross-channel interactions while remaining computationally efficient, particularly for long prediction horizons and high-dimensional time series.

Recent forecasting architectures have explored different ways of addressing these challenges. PatchTST (Nie et al., 2023) uses local temporal patches together with channel-independent processing, while TSMixer (Chen et al., 2023) demonstrates that lightweight mixing operations can provide an effective alternative to attention. iTransformer (Liu et al., 2024) explicitly models cross-variable dependencies by treating variables as tokens, and approaches such as Autoformer (Wu et al., 2022), TimesNet (Wu et al., 2023), and TimeMixer++ (Wang et al., 2025b) exploit decomposition or multi-scale temporal structure. More recent, efficient architectures, e.g. SOFTS (Han et al., 2024a) and OLinear (Yue et al., 2025) have shown that competitive forecast accuracy can be achieved with substantially simpler transformations. Despite these advances, periodicity and temporal position are typically used to construct representations or biases, while the transformations applied to those representations remain largely shared across different positions within a recurring cycle.

We argue that the position at which a forecast begins provides a useful but underexploited source of information. Consider two forecasting instances with similar historical observations, but whose forecast origins occur at different positions of a daily or weekly cycle. For example, in traffic forecasting, two windows may have similar recent traffic patterns, but one forecast may begin at 6 PM, near the evening rush hour, while another begins at 12 AM, when traffic is typically much lower. Although the recent values may provide substantial information about the future, these two instances correspond to different phases, and hence potentially different states, of the underlying

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

recurring process. Consequently, treating both instances with the same forecasting transformation may be suboptimal, motivating the use of forecast-origin phase as a conditioning signal. A model that is explicitly informed of the forecast-origin phase can potentially adapt its computation to these different states rather than relying entirely on the observed values to infer the temporal position. This leads to the central question of this work: *can forecast-origin phase be used as a conditioning signal to adapt how temporal and cross-channel information is processed?*

We introduce **PhiMixer**, a lightweight phase-conditioned architecture for long-term multivariate time-series forecasting. The central idea is to represent the position of the forecast origin within a dominant recurring cycle and use this representation to condition the forecasting computation. Rather than treating phase solely as an additional temporal feature, PhiMixer uses it to modulate the transformations performed by the model. Consequently, forecasting instances originating at different positions of a recurring cycle can induce different effective transformations while sharing the same underlying model parameters.

The main architectural component of PhiMixer is the **Phase-Informed Low-Rank Channel Mixer (PILCM)**. PILCM uses the forecast-origin phase to generate a phase-dependent family of low-rank cross-channel transformations. This allows the model to learn different cross-variable interaction patterns at different positions within a recurring cycle without constructing a dense channel-by-channel interaction matrix. The low-rank formulation also preserves the computational advantages of lightweight channel mixing, making the approach suitable for high- dimensional multivariate forecasting.

PhiMixer further extends phase conditioning to temporal processing and rhythmic interactions. We use phase-conditioned temporal modulation to allow the temporal mixer to adapt to the forecast origin, and introduce a circular relative-attention mechanism, **RRF**, to provide an explicit inductive bias for interactions between rhythmically related temporal representations. The architecture combines these mechanisms with channel-independent temporal processing and lightweight cross-channel aggregation. In particular, global aggregation and redistribution provide cross-variable information without requiring dense pairwise channel attention. Several components of the architecture build upon established techniques, including FiLM (Perez et al., 2017), patch-based representations, and STAR-style aggregate–redistribute processing (Han et al., 2024a). Our focus is therefore not on introducing each of these individual mechanisms, but on using forecast-origin phase to condition the transformations within a lightweight forecasting architecture.

We evaluate PhiMixer on 12 standard long-term forecasting benchmarks across various prediction horizons. It achieves the best performance among the compared methods on 6/8 datasets on MSE/MAE, respectively, while using approximately 126K trainable parameters and maintaining millisecond-level inference latency. Our contributions are summarized as follows:

- We introduce *forecast-origin phase conditioning* for long-term multivariate time-series forecasting, using the phase of the forecast origin to adapt both temporal and cross-channel transformations.
- We develop a Phase-Informed Low-rank Channel Mixer (PILCM), which parameterizes a low-rank cross-channel transformation as a function of forecast-origin phase, enabling phase-dependent channel interaction without explicit $C \times C$ channel mixing.
- We demonstrate the effectiveness and efficiency of the proposed framework on 12 standard long-term forecasting benchmarks, supported by component and phase-conditioning ablations and scalability analysis showing a compact parameter footprint and efficient scaling with the number of channels.

2 Related Work

2.1 Long-Term Time-Series Forecasting

Recent work on long-term time-series forecasting has explored Transformer, MLP, convolutional, linear, and multi-scale architectures. PatchTST (Nie et al., 2023) uses channel-independent patch-based representations to improve efficiency and capture local temporal patterns, while TSMixer (Chen et al., 2023) demonstrates that simple temporal and feature mixing can be competitive with attention-based models. In contrast, iTransformer (Liu et al., 2024) treats vari-

ables as tokens to explicitly model cross-variable dependencies. Decomposition- and multi-scale-based approaches, including Autoformer (Wu et al., 2022), TimesNet (Wu et al., 2023), and TimeMixer++ (Wang et al., 2025b), explicitly exploit trend, seasonal, or multi-resolution structure. More recent efficient approaches such as SOFTS (Han et al., 2024a) and OLinear (Yue et al., 2025) further demonstrate the effectiveness of lightweight cross-series and linear transformations. Other directions have explored frequency-domain filtering (Yi et al., 2024) and segment-wise mixture-of-experts architectures (Ortigossa & Segal, 2026).

These methods differ substantially in their treatment of temporal and cross-channel structure, but they generally use fixed architectural transformations once the input representation has been constructed. In particular, periodic information is commonly incorporated through decomposition, frequency representations, or multi-scale features. PhiMixer instead uses the forecast-origin phase as an explicit conditioning signal, allowing the temporal and cross-channel transformations themselves to adapt to the position of the forecast within a recurring cycle.

A separate line of work performs frequency-phase separation for time-series forecasting, including PAMod (Zhou et al., 2026), TimeAPN (Hu et al., 2026), FreDN (An et al., 2026), TimeKAN (Cao et al., 2026), and FIRE (He et al., 2025). These approaches decompose or modulate signals in the frequency domain and learn amplitude- or phase-aware representations. PhiMixer differs from them in two respects. First, the phase used here is defined relative to the forecast origin and conditions the model's transformations directly, rather than serving as a learned frequency-domain representation of the input. Second, phase modulates both the temporal and cross-channel operators, whereas frequency-phase methods typically operate on the input representation alone.

2.2 Channel Interaction and Periodic Modeling

Multivariate forecasting methods can broadly be viewed through Channel Independence (CI), Channel Dependence (CD), and Channel Partiality (CP) strategies (Qiu et al., 2025a). CI approaches, such as DLinear (Zeng et al., 2022) and PatchTST, process variables independently and therefore scale efficiently with the number of channels. CD approaches, including iTransformer (Liu et al., 2024) and OLinear (Yue et al., 2025), explicitly model inter-variable dependencies, but their interaction mechanisms can become costly for high-dimensional series. CP approaches such as MCformer (Han et al., 2024b) reduce this cost by projecting channel representations into a lower-dimensional space before interaction. SOFTS (Han et al., 2024a) provides another efficient alternative through global aggregation and redistribution of channel information. PhiMixer follows a hybrid CI+CD design, retaining independent temporal processing while introducing lightweight cross-channel coupling through global aggregation and phase-conditioned low-rank mixing.

Periodicity has also been incorporated into forecasting models through seasonal decomposition (Wu et al., 2022), frequency-domain representations (Zhou et al., 2022), multi-period modeling (Wu et al., 2023), and learnable cyclic representations (Lin et al., 2024). These approaches primarily use periodicity to construct or separate temporal representations. In contrast, PhiMixer uses the position of the forecast origin within a cycle to condition the transformations applied to the representation. This distinction enables the effective temporal and cross-channel operators to vary with forecast phase while retaining a compact computational structure.

3 Methodology

We consider the long-term multivariate forecasting problem of predicting $\hat{Y} \in \mathbb{R}^{H \times C}$ from an input window $X \in \mathbb{R}^{L \times C}$, where L is the look-back length, H is the forecast horizon, and C is the number of variables. PhiMixer is designed around a simple principle: the position of the forecast origin within a recurring cycle can provide useful context for both temporal processing and cross-channel interaction. Figure 1 summarizes the architecture.

3.1 Architecture Overview

PhiMixer consists of five stages. First, RevIN (Kim et al., 2022) normalizes the input, followed by a learnable multi-period cyclic decomposition that separates recurring components from the residual signal. Second, the forecast-origin timestamp is converted into a deterministic harmonic phase

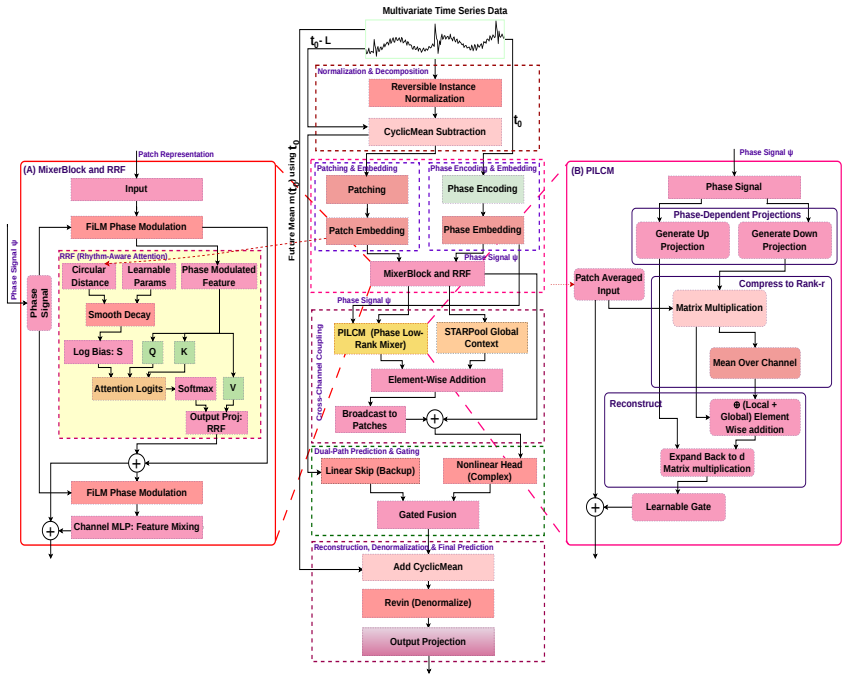


Figure 1: Forecast-origin phase conditioning in PhiMixer. The input is decomposed into cyclic and residual components. The forecast-origin phase ψ dynamically conditions the temporal and cross-channel transformations of the residual through FiLM and PILCM, respectively, while the cyclic component follows a direct forecasting path.

representation, while the residual is divided into temporal patches following PatchTST (Nie et al., 2023). Third, the patch representations are processed by phase-conditioned mixer blocks, where the phase modulates both temporal and feature transformations. Fourth, cross-channel information is introduced through a STAR-style aggregate–redistribute module inspired by SOFTS (Han et al., 2024a) and our Phase-Informed Low-Rank Channel Mixer (PILCM). Finally, a dual-path prediction head combines a direct linear forecast with a nonlinear forecast. The periodic component and the inverse RevIN transformation are then applied to obtain the final prediction.

3.2 Cyclic Decomposition and Phase Representation

Multi-period cyclic decomposition. To capture recurring patterns at multiple scales, we use learnable cyclic tables following CycleNet (Lin et al., 2024). For periods $\{P_k\}_{k=1}^{K_p}$, each table $T^{(k)} \in \mathbb{R}^{P_k \times C}$ defines

$$m_s[\tau] = \sum_{k=1}^{K_p} T^{(k)}[(s+\tau) \bmod P_k], \qquad \tau = 0, \dots, W-1. \tag{1}$$

After RevIN normalization, the look-back input is de-seasonalized as $\tilde{X} = X_{\text{norm}} - m_{t_0-L}$, while the future cyclic component m_{t_0} is generated solely from the forecast timestamp and added back to the prediction.

Forecast-origin phase. We additionally encode the forecast-origin position within the dominant cycle. For forecast index t_0 and period $\mathcal{P}$,

$$\phi = 2\pi \frac{t_0 \bmod \mathcal{P}}{\mathcal{P}}, \qquad \psi(t_0) = [1, \cos\phi, \sin\phi, \dots, \cos K\phi, \sin K\phi], \tag{2}$$

Here K is the number of harmonics and $P_{\text{dim}} = 2K + 1$ the resulting phase-feature dimension; we set $K{=}4$ across all datasets, capturing the primary cycle and its first three overtones. The phase

period $\mathcal{P}$ is derived from each dataset's sampling frequency, and the cyclic-mean periods $\{P_k\}_{k=1}^{K_p}$ in Eq. 1 extend $\mathcal{P}$ with a weekly cycle where the training window spans multiple full weeks. Per-dataset period configurations and detailed derivations are provided in Appendix B.2.

Patch embedding. The residual $\tilde{X} \in \mathbb{R}^{B\times L\times C}$ is divided into $N = L/p$ non-overlapping patches of length p and projected by a shared linear layer $\mathrm{Embed} : \mathbb{R}^p \to \mathbb{R}^d$:

$$h = \mathrm{Embed}(\mathrm{Patch}(\tilde{X})) \in \mathbb{R}^{B\times C\times N\times d}. \tag{3}$$

3.3 Phase-Conditioned Temporal Mixing

The temporal representation is obtained using M lightweight mixer blocks that alternate temporal mixing across patches and feature mixing within each patch, following TSMixer (Chen et al., 2023). Unlike standard mixers, both transformations are conditioned on the forecast-origin phase ψ using FiLM (Perez et al., 2017):

$$\mathrm{FiLM}(Z, \psi) = Z \odot (1 + \gamma(\psi)) + \beta(\psi), \qquad [\gamma; \beta] = \mathrm{MLP}(\psi). \tag{4}$$

The modulation parameters are zero-initialized, making the initial transformation an identity. Each block is then

$$h \leftarrow h + \mathrm{RRF}(\mathrm{FiLM}(\mathrm{LN}(h), \psi)), \tag{5}$$

$$h \leftarrow h + \mathrm{MLP}_{\mathrm{ch}}(\mathrm{FiLM}(\mathrm{LN}(h), \psi)). \tag{6}$$

Thus, parameters are shared across forecasting instances while their effective transformations vary with forecast-origin phase.

Rhythm-aware relative attention (RRF). RRF provides a circular relative-position prior over the N patches. For the FiLM-modulated representation Z, multi-head queries, keys, and values are obtained by a shared linear projection. For patches i and j, the circular distance is

$$\gamma_{ij} = \min\big((i-j) \bmod N,\ (j-i) \bmod N\big), \tag{7}$$

which is converted into a learnable smooth decay:

$$S_{ij} = \frac{1}{1 + \exp(a(\gamma_{ij} - b))} + \frac{\exp(-\gamma_{ij})}{1 + \exp(ab)}. \tag{8}$$

The resulting bias is incorporated into the attention logits:

$$A = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_h}} + \log S\right), \qquad \mathrm{RRF}(Z) = \mathrm{Proj}(AV), \tag{9}$$

where a and b are learnable and d_h is the per-head dimension. RRF operates over patches within the look-back window, while longer-period seasonality is handled by the cyclic decomposition.

3.4 Efficient Cross-Channel Coupling

The mixer blocks process each channel independently. To recover cross-variable dependencies without the quadratic cost of dense channel attention, PhiMixer introduces two complementary coupling mechanisms: STARPool and PILCM. Both operate on a compact channel representation obtained by averaging the patch dimension,

$$\bar{h} = \frac{1}{N}\sum_{n=1}^{N} h_{:,:,n,:} \in \mathbb{R}^{B\times C\times d}. \tag{10}$$

STAR-style aggregation. We adopt the aggregate–redistribute principle of STAR in SOFTS (Han et al., 2024a). A global representation is formed by aggregating information across channels and is subsequently fused with each channel representation:

$$\mathrm{STAR}(\bar{h}) = \bar{h} + \sigma(g_s) \cdot \mathrm{MLP}_{\mathrm{fuse}}\left(\left[\bar{h}, \mathrm{mean}_c\left(\mathrm{MLP}_{\mathrm{core}}(\bar{h})\right)\right]\right). \tag{11}$$

Here, mean_c averages over the channel dimension, $\mathrm{MLP}_{\mathrm{core}}$ is a two-layer MLP producing a shared global context, $\mathrm{MLP}_{\mathrm{fuse}}$ is a single linear layer fusing channel and global features, and $\sigma(g_s)$ is a learnable scalar gate. The resulting representation is broadcast across the patch dimension and added to h.

Phase-Informed Low-Rank Channel Mixer. PILCM is the central cross-channel component of PhiMixer. Given the forecast-origin phase ψ, $W_p^{\downarrow} \in \mathbb{R}^{d\times r}$, $W_p^{\uparrow} \in \mathbb{R}^{r\times d}$, and $r \ll d$, it generates phase-dependent low-rank projection operators

$$D(\psi) = \sum_p \psi_p W_p^{\downarrow}, \qquad U(\psi) = \sum_p \psi_p W_p^{\uparrow}. \tag{12}$$

For the patch-averaged representation $\bar{h}$, we compute

$$Z = \bar{h}D(\psi), \qquad \bar{Z} = \frac{1}{C}\sum_c Z_{:,c,:}, \tag{13}$$

and

$$\mathrm{PILCM}(\bar{h}, \psi) = \bar{h} + \sigma(g_p)(Z + \bar{Z})U(\psi). \tag{14}$$

Thus, phase dynamically controls cross-channel information sharing while the rank-r bottleneck avoids explicit $C \times C$ interactions, with $O(Cdr + CNd)$ computation.

3.5 Dual-Path Prediction

The processed representation is flattened and passed through dropout before two complementary prediction paths. The nonlinear path uses the learned representation,

$$y_{\mathrm{nl}} = W_{\mathrm{head}}\,\mathrm{vec}(h), \tag{15}$$

while the linear path directly forecasts from the de-seasonalized input,

$$y_{\mathrm{lin}} = W_{\mathrm{skip}}\tilde{X}_c. \tag{16}$$

Their outputs are combined through a learnable gate:

$$\hat{Y} = \mathrm{RevIN}^{-1}\left(\left[y_{\mathrm{lin}} + d_{\mathrm{nl}}\sigma(\alpha)y_{\mathrm{nl}}\right]^{\top} + m_{t_0}\right), \tag{17}$$

where d_{nl} is a stochastic path-drop mask during training and equals one at inference. The linear path provides a direct forecasting route, while the nonlinear path captures higher-order dependencies from the phase-conditioned representation.

3.6 Training Objective

PhiMixer is trained end-to-end with a composite objective combining pointwise forecasting accuracy with decomposition, spectral, and cross-channel consistency:

$$\mathcal{L} = \mathcal{L}_{\mathrm{bal}} + \lambda_{\mathrm{DB}}\mathcal{L}_{\mathrm{DB}} + \lambda_{\mathrm{F}}\mathcal{L}_{\mathrm{FreDF}} + \lambda_{\mathrm{C}}\mathcal{L}_{\mathrm{CCC}}. \tag{18}$$

Here, $\mathcal{L}_{\mathrm{bal}}$ is the balanced forecasting loss, while the auxiliary terms enforce decomposition, frequency-domain, and cross-channel consistency. The frequency-domain term follows FreDF (Wang et al., 2025a). Full definitions are provided in Appendix B.6.1.

4 Experiments

This section presents the evaluation of PhiMixer on 12 LTSF benchmark datasets that span a variety of domains, such as electricity, weather, finance, and traffic. More details of the datasets, baselines, and implementation are given in Appendix B.

4.1 Long-Term Forecasting Results

We evaluate PhiMixer on 12 standard long-term forecasting benchmarks using $L = 96$. Tables 1 and 2 report results averaged across prediction horizons. Across the eight main benchmarks, PhiMixer achieves its strongest performance on ETTm1, ETTm2, and ECL, obtaining the best average MSE/MAE on these datasets. It remains competitive on ETTh1, ETTh2, Weather, and Exchange, while Traffic remains challenging due to its strong spatial cross-channel dependencies. On the four PEMS benchmarks, PhiMixer achieves the best average results on PEMS03, PEMS07, and PEMS08 and remains competitive on PEMS04. Averaged across PEMS03/04/07/08, PhiMixer achieves an aggregate MSE/MAE of 0.139/0.242. Overall, the results demonstrate that PhiMixer provides competitive long-term forecasting accuracy.

Table 1: Long-term forecasting results averaged over four prediction horizons $H \in \{96, 192, 336, 720\}$. Best results are shown in red and second-best in blue. Full results can be found in Appendix C.

	PhiMixer		OLinear		TimeM.++		Time-o1		GCMNet		SEG-MOE		SOFTS		TexFilter		PatchTST	
Dataset	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	0.435	**0.427**	**0.424**	**0.424**	**0.418**	0.432	0.431	0.429	0.445	0.443	0.473	0.443	0.449	0.443	0.441	0.439	0.453	0.446
ETTh2	**0.356**	**0.384**	0.367	0.389	**0.339**	**0.380**	0.359	0.388	0.376	0.401	0.401	0.416	0.373	0.400	0.383	0.407	0.385	0.410
ETTm1	**0.359**	**0.377**	0.374	**0.377**	**0.368**	**0.378**	0.379	0.393	0.389	0.403	0.438	0.419	0.393	0.402	0.391	0.401	0.396	0.406
ETTm2	**0.254**	**0.302**	0.270	**0.314**	**0.269**	0.320	0.272	0.317	0.274	0.319	0.312	0.348	0.287	0.323	0.285	0.328	0.287	0.330
ECL	**0.152**	**0.243**	**0.159**	**0.248**	0.165	0.253	0.170	0.260	0.170	0.265	0.250	0.323	0.174	0.264	0.172	0.268	0.189	0.276
Weather	**0.231**	**0.255**	0.237	**0.260**	**0.226**	0.262	0.241	0.280	0.248	0.278	0.266	0.283	0.255	0.278	0.239	0.272	0.256	0.279
Exchange	0.369	**0.399**	**0.355**	**0.399**	**0.357**	0.409	**0.357**	0.405	0.358	**0.403**	–	–	–	–	0.388	0.419	0.366	0.404
Traffic	0.467	0.293	0.451	**0.247**	**0.416**	**0.264**	0.419	0.280	0.447	0.281	0.531	0.330	**0.409**	**0.267**	0.462	0.310	0.454	0.286

Table 2: Average PEMS forecasting performance over $H \in \{12, 24, 48, 96\}$. Best results are shown in red and second-best in blue. Full results can be found in Appendix C.

	PhiMixer		TimeM.++		MSTN-Tra.		MSTN-BiL.		M.TCN		TSMixer		TimeMixer		FITS		DLinear		PatchTST		Autoformer	
Dataset	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	**0.117**	**0.227**	0.165	0.263	**0.156**	**0.257**	0.167	0.272	0.408	0.419	0.162	0.281	0.167	0.267	0.489	0.465	0.278	0.375	0.180	0.291	0.554	0.542
PEMS04	0.127	0.242	0.136	0.251	**0.115**	**0.223**	**0.125**	**0.234**	0.488	0.471	0.136	0.254	0.185	0.287	0.531	0.489	0.295	0.388	0.195	0.307	0.764	0.669
PEMS07	**0.133**	**0.233**	0.152	0.258	**0.149**	**0.237**	0.187	0.263	0.304	0.374	0.150	0.251	0.181	0.271	0.500	0.472	0.329	0.395	0.211	0.303	0.362	0.438
PEMS08	**0.180**	**0.265**	**0.200**	**0.279**	0.318	0.322	0.348	0.348	0.510	0.479	0.225	0.305	0.226	0.299	0.534	0.487	0.379	0.416	0.280	0.321	0.739	0.627

4.2 Component and Phase Ablation Study

We systematically evaluated eight ablation variants across ETTh1, ETTh2, PEMS03, and PEMS04. In each *w/o* variant, the indicated **component or phase-conditioning** input is removed: PILCM, FiLM, STARPool, RRF, CyclicMean, phase in PILCM, phase in FiLM, or phase in both PILCM and FiLM. Table 3 reports results averaged across forecasting horizons.

On the ETTh datasets, RRF and PILCM show the most consistent contributions. Removing RRF increases average MSE from 0.433 to 0.455 on ETTh1 (+5.1%) and from 0.356 to 0.371 on ETTh2 (+4.2%). Removing PILCM increases MSE to 0.449 (+3.7%) and 0.374 (+5.1%), respectively. The detailed per-horizon results further show that RRF can produce larger degradations at individual horizons, reaching a 7.3% increase on ETTh1 at $H = 720$. Removing phase conditioning from PILCM and FiLM also degrades performance, with ETTh1 MSE increasing from 0.433 to 0.445 (+2.8%) and 0.443 (+2.3%), respectively. Removing phase from both modules increases MSE to 0.446 (+3.0%) on ETTh1 and 0.373 (+4.8%) on ETTh2. These results indicate that phase conditioning provides additional benefits beyond the individual module architectures.

Table 3: Ablation results averaged across forecasting horizons ($H \in \{96, 192, 336, 720\}$ for ETTh, $H \in \{12, 24, 48, 96\}$ for PEMS). **Red**: 1st rank, **Blue**: 2nd rank. STARP: STARPool, CyclicM: CyclicMean, P.: Phase. Full results are given in Appendix D.

Dataset	Full Model		w/o PILCM		w/o FiLM		w/o STARP		w/o RRF		w/o CyclicM		w/o P.in PILCM		w/o P.in FiLM		w/o Phase	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.433**	**0.427**	0.449	0.441	0.444	0.437	0.443	0.436	0.455	0.440	**0.438**	**0.434**	0.445	0.438	0.443	0.437	0.446	0.440
ETTh2	**0.356**	**0.384**	0.374	0.401	0.372	0.399	**0.369**	**0.395**	0.371	0.398	**0.369**	0.397	0.371	0.398	0.370	0.397	0.373	0.398
PEMS03	**0.117**	**0.227**	0.121	0.232	0.131	0.241	**0.118**	**0.228**	0.123	0.233	0.128	0.238	0.119	0.230	0.124	0.234	0.160	0.265
PEMS04	**0.127**	**0.242**	0.136	0.251	0.148	0.262	**0.131**	**0.244**	0.133	0.246	0.148	0.262	0.132	0.246	0.140	0.253	0.196	0.296

Table 4: Effect of progressively adding loss components on ETTh1, ETTm2, PEMS03, and PEMS04 at H=96.

	ETTh1		ETTm2		PEMS03		PEMS04	
Loss Components	**MSE**	**Imp.**	**MSE**	**Imp.**	**MSE**	**Imp.**	**MSE**	**Imp.**
MSE only (baseline)	0.371	—	0.151	—	0.193	—	0.201	—
+ DB	0.368	0.81	0.149	1.32	0.191	1.04	0.199	0.99
+ FreDF	0.367	0.27	0.148	0.67	0.190	0.52	0.198	0.50
+ CCC (Full loss)	**0.365**	**0.54**	**0.147**	**0.68**	**0.188**	**1.05**	**0.196**	**1.01**
Total improvement (%)		**1.62**		**2.65**		**2.59**		**2.49**

On the PEMS datasets, the relative contribution of the components becomes more pronounced and dataset-dependent. RRF removal increases average MSE from 0.117 to 0.123 (+5.1%) on PEMS03 and from 0.127 to 0.133 (+4.7%) on PEMS04. CyclicMean has a larger effect on the traffic datasets: removing it increases average MSE by 9.4% on PEMS03 and 16.5% on PEMS04. FiLM is also important, with average MSE increases of 12.0% and 16.5%, respectively, when it is removed. The detailed per-horizon results show even larger degradations at $H = 96$, reaching 16.0% on PEMS03 and 23.5% on PEMS04 for FiLM removal.

Phase conditioning has a particularly strong effect on the PEMS datasets. Removing phase from both PILCM and FiLM increases average MSE from 0.117 to 0.160 (+36.8%) on PEMS03 and from 0.127 to 0.196 (+54.3%) on PEMS04. In contrast, removing phase from only PILCM or only FiLM produces smaller degradations: on PEMS04, average MSE increases to 0.132 (+3.9%) and 0.140 (+10.2%), respectively. This indicates that the *phase signal is most effective when it is jointly used to condition multiple forecasting transformations rather than being confined to a single module*.

Overall, the ablations support a complementary role for rhythm-aware attention, phase-conditioned transformation, and periodic decomposition. RRF and PILCM provide consistent gains across the evaluated datasets, while FiLM and CyclicMean become particularly important for the PEMS benchmarks. STARPool has a comparatively smaller and more variable effect. Most importantly, the large degradation caused by removing phase conditioning from both FiLM and PILCM demonstrates that the benefit of phase is not limited to representing periodic structure; using forecast-origin phase to actively condition the forecasting transformations is an important part of PhiMixer. To isolate the mechanism, we compare against a variant that concatenates $\psi(t_0)$ to the patch embeddings (FiLM off, PILCM phase-free). Conditioning gives lower MSE than concatenation across all tested datasets, with the gap growing with horizon on PEMS (Appendix D.1).

4.3 Sensitivity Analysis

We examine how sensitive PhiMixer is to the phase period $\mathcal{P}$. We vary $\mathcal{P}$ by ± 2 and ± 4 steps around its sampling-derived value on ETTh1, ETTh2, Weather, PEMS03, and PEMS04. Sensitivity scales with the strength of daily periodicity: on ETTh1 (weak periodicity), varying $\mathcal{P}$ by ± 4 steps changes MSE by less than 0.4%; on ETTh2 (moderate periodicity), a U-shaped trend appears with minimum at the default value and ± 4 steps cost 1.1–1.5% MSE; on Weather (strong periodicity), ± 4 steps cost 5.2–5.7% MSE. In all five datasets, the sampling-derived $\mathcal{P}$ is optimal or within noise of the optimum. On the main benchmarks (ETTh1, ETTh2, and Weather), even a ± 4-step shift costs at most 6% MSE; on the PEMS benchmarks, where the daily cycle is sharpest, sensitivity is much larger. We also examined sensitivity to K, r, and p across all five datasets; differences across values were small on ETTh1, ETTh2, and Weather, and somewhat larger on the PEMS benchmarks, with the default settings providing a favorable balance of accuracy and parameter count. Appendix D.1 provides the full analysis.

4.4 Effect of Loss Components

We further examine the contribution of the composite training objective by progressively adding its loss components on ETTh1, ETTm2, PEMS03, and PEMS04 at H=96. Starting from the standard MSE loss, we sequentially introduce the decomposition-balancing (DB), frequency-domain (FreDF), and cross-channel correlation consistency (CCC) terms. DB encourages consistency between the predicted and decomposed components, FreDF promotes agreement in the frequency

Table 5: Complexity and parameter comparison between PhiMixer and representative SOTA forecasters, at the common ETTh1 configuration (C=7, L=96, H=96). Ratio is relative to PhiMixer (∼126K). Red: more parameters; Blue: fewer parameters. Full details in Appendix G.1

Model	Complexity	H Scaling	Params	Ratio
CycleNet	$\mathcal{O}(L \cdot C)$	$\mathcal{O}(H)$	∼65K	↓ 0.52×
ModernTCN	$\mathcal{O}(L \cdot C)$	$\mathcal{O}(H)$	∼100K	↓ 0.79×
PatchTST	$\mathcal{O}(CL^2 + CH)$	$\mathcal{O}(H)$	∼300K	↑ 2.38×
Time-o1	$\mathcal{O}(m \cdot H^2)$	$\mathcal{O}(H^2)$	∼290K	↑ 2.30×
TimeMixer++	$\mathcal{O}(M \cdot C \cdot K \cdot (Lf + pf))$	$\mathcal{O}(H)$	∼647K	↑ 5.13×
OLinear	$\mathcal{O}(T \cdot C + C \cdot d)$	$\mathcal{O}(H)$	∼1.3M	↑ 10.32×
PhiMixer	$\mathcal{O}(CLd + CN^2d + CNd(d + H + 1) + Cd^2 + Cdr + CLH)$	$\mathcal{O}(H)$	∼126K	1.00×

domain, and CCC preserves cross-channel dependencies. As shown in Table 4, each loss component provides a consistent incremental improvement across datasets. On ETTh1, adding DB reduces MSE by 0.81%, followed by another 0.27% improvement with FreDF and 0.54% with CCC, resulting in a total reduction of 1.62% relative to the MSE-only baseline. On ETTm2, the corresponding incremental improvements are 1.32%, 0.67%, and 0.68%, yielding a total improvement of 2.65%. On the PEMS traffic benchmarks, the auxiliary losses provide larger gains: total reductions of 2.59% on PEMS03 and 2.49% on PEMS04. Notably, on PEMS03 and PEMS04 the CCC loss contributes a single-step improvement comparable to the first auxiliary loss (DB), consistent with the strong cross-sensor correlations present in urban traffic data. These results indicate that the auxiliary objectives provide complementary supervision beyond the pointwise MSE objective. The ablation ("Architecture vs. composite loss") in Appendix D.1 shows that both the architecture and the loss contribute, with the architecture contributing more, and combining them is best.

4.5 Efficiency Analysis

PhiMixer is designed to scale efficiently with both sequence length and the number of channels. Patching reduces the temporal dimension from L to $N = L/p$, so rhythm-aware attention operates on N patch tokens with $\mathcal{O}(CN^2d)$ complexity rather than on the full sequence. With $p = 16$, the quadratic temporal term is reduced by a factor of $p^2 = 256$. More importantly, PILCM avoids explicit $C \times C$ channel interactions by using a rank-r bottleneck, requiring $\mathcal{O}(Cdr + CNd)$ computation. Thus, the channel-dependent computation of PhiMixer scales linearly with C. To validate this across datasets, increasing the number of channels from $C = 7$ to $C = 883$ increases GPU latency by only $1.32\times$ (1.91–2.52 ms), empirically supporting the intended near-linear channel scaling (Refer to Appendix G.3).

The parameter footprint is also compact. Most mixer parameters are shared across channels, while only the cyclic tables and RevIN parameters introduce additional channel-dependent parameters. At the common ETTh1 configuration ($C = 7$, $L = H = 96$), PhiMixer requires approximately 126K parameters. Table 5 compares its computational scaling and parameter footprint with representative forecasting models. PhiMixer is larger than the simplest linear and decomposition baselines, but substantially smaller than several recent models while retaining phase-conditioned temporal and cross-channel modeling (Refer to Appendix G.1).

5 Conclusion

We presented PhiMixer, a lightweight multivariate forecasting framework built around forecast-origin phase conditioning. By using phase to modulate temporal and cross-channel transformations, PhiMixer combines periodic decomposition, rhythm-aware attention, phase-conditioned low-rank channel mixing, and dual-path prediction within a compact architecture. Experiments on 12 standard benchmarks demonstrate competitive forecasting performance, with particularly strong results on several ETT, ECL, and PEMS datasets while maintaining a lightweight computational footprint. Ablation studies further show that phase conditioning provides meaningful gains, particularly on PEMS, while RRF and PILCM contribute consistently across datasets. Overall, the results demonstrate that forecast-origin phase can serve as an effective conditioning signal for long-term time-series forecasting without substantial computational overhead. Future work will explore phase-conditioned modeling for imputation, classification, and cross-domain time-series pre-training.

Data and Code Availability

The datasets employed in this study are all publicly available from their original sources. They are referenced throughout the paper and described in Appendix B.1. The source code, dataset preprocessing scripts, and evaluation scripts are released in the anonymized repository at https://anonymous.4open.science/r/Phimixer-51E7.

Competing Interests

The authors declare no conflict of interest.

AI Use Statement

AI tools were used for grammar checking and formatting only.

Ethics Statement

This work uses only publicly available, non-personal time-series datasets and does not involve human subjects, personal data, or sensitive information. The proposed method is intended for general time-series forecasting and does not present foreseeable ethical concerns.

Reproducibility Statement

Dataset splits, hyperparameters, and evaluation protocols are described in Appendix B. Full per-horizon results are provided in Appendix C.

References

Zhongde An, Jinhong You, Jiyanglin Li, Yiming Tang, Wen Li, Heming Du, and Shouguo Du. FreDN: Spectral disentanglement for time series forecasting via learnable frequency decomposition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026. URL https://arxiv.org/abs/2511.11817.

Jinfei Cao, Zheru Dong, Haoyi Xu, Xiaoou Liu, and You Chen. TimeKAN: an adaptive frequency-decomposed kolmogorov–arnold network for long-term stock forecasting. *Scientific Reports*, 2026. doi: 10.1038/s41598-026-59107-4. URL https://doi.org/10.1038/s41598-026-59107-4.

Ching Chang, Wei-Yao Wang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Aligning pre-trained llms as data-efficient time-series forecasters. *ACM Trans. Intell. Syst. Technol.*, 16(3), April 2025. ISSN 2157-6904. doi: 10.1145/3719207. URL https://doi.org/10.1145/3719207.

Si-An Chen, Chun-Liang Li, Sercan O Arik, Nathanael Christian Yoder, and Tomas Pfister. TSMixer: An all-MLP architecture for time series forecast-ing. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=wbpxTuXgm0.

Yu Chen and PeMS data. Pems, 2025. URL https://test.ieee-dataport.org/documents/pems-3. California Traffic Network Datasets: PEMS03, PEMS04, PEMS07 and PEMS08 for Traffic Flow Forecasting.

Luo donghao and wang xue. ModernTCN: A modern pure convolution structure for general time series analysis. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=vpJMJerXHU.

Vijay Ekambaram, Arindam Jati, Pankaj Dayama, Sumanta Mukherjee, Nam H. Nguyen, Wesley M. Gifford, Chandra Reddy, and Jayant Kalagnanam. Tiny time mixers (ttms): Fast pre-trained

models for enhanced zero/few-shot forecasting of multivariate time series, 2024. URL https://arxiv.org/abs/2401.03955.

Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. Softs: Efficient multivariate time series forecasting with series-core fusion, 2024a. URL https://arxiv.org/abs/2404.14197.

Wenyong Han, Tao Zhu, Liming Chen, Huansheng Ning, Yang Luo, and Yaping Wan. Mcformer: Multivariate time series forecasting with mixed-channels transformer. *IEEE Internet of Things Journal*, 11(17):28320–28329, 2024b. doi: 10.1109/JIOT.2024.3401697.

Cheng He, Xijie Liang, Zengrong Zheng, Patrick P. C. Lee, Xu Huang, Zhaoyi Li, Hong Xie, Defu Lian, and Enhong Chen. A unified frequency domain decomposition framework for interpretable and robust time series forecasting, 2025. URL https://arxiv.org/abs/2510.10145.

Yue Hu, Jialiang Tang, Siwei Yu, Baosheng Yu, Jing Zhang, and Dacheng Tao. TimeAPN: Adaptive amplitude-phase non-stationarity normalization for time series forecasting, 2026. URL https://arxiv.org/abs/2603.17436.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-llm: Time series forecasting by reprogramming large language models, 2024. URL https://arxiv.org/abs/2310.01728.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=cGDAkQo1C0p.

Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. Cyclenet: Enhancing time series forecasting through modeling periodic patterns. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 106315–106345. Curran Associates, Inc., 2024. doi: 10.52202/079017-3373. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/bfe7998398779dde03cad7a73b1f81b6-Paper-Conference.pdf.

Wenxie Lin. California traffic network datasets, 2025. PEMS03, PEMS04, PEMS07, PEMS08 traffic flow datasets from Caltrans PeMS.

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, 2022. URL https://arxiv.org/abs/2106.09305.

Xiangsen Liu, Jinchang Ren, Hongming Zhang, and Erlei Zhang. Gcmnet: A global context mamba network for long-term time series forecasting. *Pattern Recognition*, 177:113287, 2026. ISSN 0031-3203.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting, 2024. URL https://arxiv.org/abs/2310.06625.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers, 2023. URL https://arxiv.org/abs/2211.14730.

Evandro S. Ortigossa and Eran Segal. Seg-moe: Multi-resolution segment-wise mixture-of-experts for time series forecasting transformers, 2026. URL https://arxiv.org/abs/2601.21641.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer, 2017. URL https://arxiv.org/abs/1709.07871.

Xiangfei Qiu, Hanyin Cheng, Xingjian Wu, Jilin Hu, Chenjuan Guo, and Bin Yang. A comprehensive survey of deep learning for multivariate time series forecasting: A channel strategy perspective, 2025a. URL https://arxiv.org/abs/2502.10721.

Xiangfei Qiu, Xingjian Wu, Hanyin Cheng, Xvyuan Liu, Chenjuan Guo, Jilin Hu, and Bin Yang. Dbloss: Decomposition-based loss function for time series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pp. 27741–27768. Curran Associates, Inc., 2025b.

Chao Shang, Jie Chen, and Jinbo Bi. Discrete graph structure learning for forecasting multiple time series, 2021. URL https://arxiv.org/abs/2101.06861.

Sumit S Shevtekar and Chandresh Kumar Maurya. MSTN: A lightweight and fast model for general time-series analysis. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856.

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*, 2025a.

Hao Wang, Licheng Pan, Zhichao Chen, Xu Chen, Qingyang Dai, Lei Wang, Haoxuan Li, and Zhouchen Lin. Time-o1: Time-series forecasting needs transformed label alignment. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026.

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*, 2024. URL https://openreview.net/forum?id=7oLshfEIC2.

Shiyu Wang, Jiawei Li, Xiaoming Shi, Zhou Ye, Baichuan Mo, Wenze Lin, Shengtong Ju, Zhixuan Chu, and Ming Jin. Timemixer++: A general time series pattern machine for universal predictive analysis, 2025b.

Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Etsformer: Exponential smoothing transformers for time-series forecasting, 2022. URL https://arxiv.org/abs/2202.01381.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting, 2022. URL https://arxiv.org/abs/2106.13008.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis, 2023. URL https://arxiv.org/abs/2210.02186.

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, 2023. URL https://arxiv.org/abs/2311.06190.

Kun Yi, Jingru Fei, Qi Zhang, Hui He, Shufeng Hao, Defu Lian, and Wei Fan. Filternet: Harnessing frequency filters for time series forecasting. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Wenzhen Yue, Yong Liu, Hao Wang, Haoxuan Li, Xianghua Ying, Ruohao Guo, Bowei Xing, and Ji Shi. OLinear: A linear model for time series forecasting in orthogonally transformed domain. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=DAyKP1tvwI.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting?, 2022. URL https://arxiv.org/abs/2205.13504.

Tianping Zhang, Yizhuo Zhang, Wei Cao, Jiang Bian, Xiaohan Yi, Shun Zheng, and Jian Li. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures, 2022. URL https://arxiv.org/abs/2207.01186.

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

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting, 2022. URL https://arxiv.org/abs/2201.12740.

Yingbo Zhou, Yutong Ye, Shuhao Li, Rui Qian, Qiang Huang, Lemao Liu, Li Sun, and Dejing Dou. PAMod: Modeling cyclical shifts via phase-amplitude modulation for non-stationary time series forecasting, 2026. URL https://arxiv.org/abs/2605.00466.

A Architectural Design and Rationale

A.1 How is PhiMixer Different from Prior Works?

The field of multivariate time series forecasting is conventionally organized around three channel interaction strategies—Channel Independence (CI), Channel Dependence (CD), and Channel Partiality (CP) approaches (Qiu et al., 2025a)—as summarized in Table 6. These strategies specify how a model encodes and exploits inter-variable dependencies across the C channels of a multivariate signal.

CI. CI methods such as DLinear (Zeng et al., 2022) and PatchTST (Nie et al., 2023) process each variable independently using lightweight temporal operators such as linear layers or multilayer perceptrons. Their advantage lies in linear channel cost $\mathcal{O}(C)$ and efficient temporal complexity $\mathcal{O}(L)$, though this comes at the expense of discarding cross-channel structure.

CD. CD approaches such as iTransformer (Liu et al., 2024) and OLinear (Yue et al., 2025), whose Cross-Series Learner explicitly models multivariate dependencies through a normalized linear layer, capture inter-variable correlation directly. While attention- and graph-based CD variants offer comprehensive feature coupling, they typically incur high computational cost—$\mathcal{O}(C^2)$ or $\mathcal{O}(L^2)$—which limits scalability and latency.

CP. CP models such as MCformer (Han et al., 2024b) map the channel dimension into a low-rank latent space before applying interaction, thereby avoiding the full $\mathcal{O}(C^2)$ cost of CD models. Although channel-efficient, these models often impose simplified temporal assumptions, reducing their ability to capture multi-scale dynamics.

Existing state-of-the-art systems illustrate these trade-offs. CI-based PatchTST (Nie et al., 2023) relies on fixed patch-based tokenization; LLM-derived approaches such as LLM4TS (Chang et al., 2025) and TIME-LLM (Jin et al., 2024) depend on frozen backbone architectures; CD-based TimesNet (Wu et al., 2023) employs fixed frequency priors; and CP-based MCformer (Han et al., 2024b) attains efficiency through partial channel blending. TimeMixer++ (Wang et al., 2025b) adopts multi-resolution imaging, and OLinear (Yue et al., 2025) achieves efficiency through orthogonal transformations combined with an explicit, though static, cross-series dependency layer. These approaches use fixed or architecture-specific mechanisms for incorporating temporal structure, whereas PhiMixer makes forecast-origin phase an explicit conditioning variable. By combining the complementary strengths of the CI and CD strategies with a novel phase-conditioned processing mechanism, PhiMixer delivers improved performance across long-term forecasting benchmarks while maintaining a compact footprint and low-latency characteristics.

A.1.1 Core Innovation: Phase-Conditioned Processing and Hybrid Channel Design

PhiMixer is defined by its phase-conditioned processing paradigm combined with a hybrid channel strategy that integrates CI and CD approaches. This fusion of temporal strategies aims to balance the trade-offs among CI, CD, and CP modeling approaches. Specifically, the two core architectural ideas of PhiMixer are:

1. **Phase-Conditioned Processing.**

Table 6: Taxonomy of model architectures by channel-interaction strategy, adapted from (Qiu et al., 2025a). Different state-of-the-art models align with the established Channel Interaction (CI, CD, CP) strategies. PhiMixer is positioned as a hybrid design (CI+CD) via phase-conditioned processing and low-rank channel mixing to capture periodic, phase-adaptive dynamics.

Strategy	**Mechanism**	**Characteristic**						**Method**
		Asym.	**Lag.**	**Pol.**	**Gw.**	**Dyn.**	**Ms.**	
CI	-	-	-	-	-	-	-	PatchTST
CI	-	-	-	-	-	-	-	CycleNet
CI	-	-	-	-	-	-	-	DLinear
CI	-	-	-	-	-	-	-	Timer
CI	-	-	-	-	-	-	-	Chronos
CI	-	-	-	-	-	-	-	LLM4TS
CI	-	-	-	-	-	-	-	Time-LLM
CI	-	-	-	-	-	-	-	RevIN
CD	CNN-based	✓	-	-	-	-	-	Informer
CD	CNN-based	✓	-	-	-	-	-	Autoformer
CD	CNN-based	✓	-	-	-	-	-	FEDformer
CD	CNN-based	✓	-	-	-	-	-	TimesNet
CD	MLP-based	✓	-	-	-	-	-	TSMixer
CD	MLP-based	✓	-	-	-	-	-	TTM
CD	Transformer-based	✓	-	-	-	✓	-	iTransformer
CD	Transformer-based	✓	-	-	-	✓	-	Crossformer
CD	Transformer-based	✓	✓	-	-	-	-	VCformer
CD	Transformer-based	✓	✓	-	-	✓	-	MOIRAI
CD	Transformer-based	✓	-	-	-	✓	-	UniTS
CD	GNN-based	-	-	-	✓	-	-	GTS
CD	GNN-based	✓	-	-	-	-	✓	MSGNet
CD	GNN-based	-	✓	-	-	-	-	FourierGNN
CD	GNN-based	-	✓	-	-	-	-	FC-STGNN
CD	GNN-based	✓	-	-	-	✓	-	TPGNN
CD	Linear Channel Mixing	✓	-	-	-	-	-	SOFTS
CD	Others	✓	-	-	-	✓	-	C-LoRA
CP	CNN-based	✓	-	-	-	-	-	ModernTCN
CP	Transformer-based	✓	-	-	✓	-	-	DUET
CP	Transformer-based	✓	-	-	-	✓	-	MCformer
CP	Transformer-based	✓	-	-	✓	✓	-	DGCformer
CP	Transformer-based	-	-	-	-	✓	-	CM
CP	GNN-based	✓	-	-	-	-	-	MTGNN
CP	GNN-based	✓	-	✓	-	-	-	CrossGNN
CP	GNN-based	✓	-	-	✓	-	-	WaveForM
CP	GNN-based	-	-	-	-	✓	-	MTSF-DG
CP	GNN-based	✓	-	-	✓	-	-	ReMo
CP	GNN-based	✓	-	-	✓	✓	✓	Ada-MSHyper
CP	Others	✓	✓	-	-	-	-	LIFT
CP	Others	✓	-	-	✓	-	-	CCM
CI + CD	Parallel CNN (CI) and BiLSTM/Transformer(CD) based Dual Encoder	✓	-	-	-	✓	✓	MSTN
CI + CD	**Phase-conditioned MixerBlocks (CI) + STARPool and PILCM (CD)**	✓	-	-	-	✓	✓	**PhiMixer**

In contrast to approaches that treat periodic structure as a secondary consideration, PhiMixer treats phase as a primary conditioning variable. The phase ϕ of the forecast origin—defined as $\phi = 2\pi(t_0 \bmod \mathcal{P})/\mathcal{P}$—serves as a shared conditioning signal for the principal temporal and cross-channel transformations. This phase information modulates each mixer block via FiLM, letting the network adjust its internal transformations based on the current position within the cycle. It also generates the projection matrices of a low-rank cross-channel mixing module (PILCM), letting channel interactions adapt to the temporal phase at cost $O(C \cdot d \cdot r + C \cdot N \cdot d)$—linear in the number of channels for fixed d and r. This gives the model an explicit inductive bias for periodic structure.

2. **Hybrid CI+CD Channel Strategy with Linear Complexity.** PhiMixer combines two complementary channel strategies. The MixerBlocks operate under the CI paradigm, processing each channel independently to maintain efficiency. STARPool and PILCM jointly introduce CD: STARPool pools information across all channels to form a global context broadcast back to every channel, while PILCM converts forecast-origin phase into a phase-conditioned low-rank channel operator, performing coupling within a rank-r compressed feature space ($r \ll d$) at cost $O(C \cdot d \cdot r + C \cdot N \cdot d)$, without pairwise $O(C^2)$ cross-channel

Table 7: Comparison of architectural characteristics between the CI, CD, and CP strategies and PhiMixer (CI+CD). Ratings are directional, following the trade-offs described above.

Dimension	CI	CD	CP	(CI+CD) PhiMixer	Rationale
Efficiency	High	Low	Moderate	High	PILCM keeps channel mixing linear, $\mathcal{O}(C)$; patching reduces temporal cost.
Phase-Awareness	Low	Low	Low	High	Phase embedding conditions the FiLM and PILCM modules directly.
Cross-Channel Information	Low	High	Moderate	High	STARPool and PILCM provide global channel context via mean-pooling across all channels.
Parameter Footprint	Low	High	Moderate	Low	$\sim$126K parameters (≈ 0.48 MB) (Section G.1).

modeling. This hybrid CI+CD design lets PhiMixer capture cross-channel dependencies while keeping complexity linear in C.

Unlike hybrid approaches that simply stack fixed CI and CD components, PILCM generates its channel-interaction matrices from the phase embedding at each step, at the small additional cost noted above ($O(Cdr)$), rather than learning one static interaction pattern. This means the strength of cross-channel coupling can vary with the position in the cycle—for example, denser interaction during rush hour and weaker interaction late at night—though we present this as a motivating intuition rather than a directly measured effect. This phase-adaptive cross-channel coupling is a distinction from prior work, where channel interaction patterns are typically static.

The PhiMixer framework integrates phase-conditioned processing with a hybrid CI+CD channel strategy, aiming to reconcile the core trade-offs among CI-, CD-, and CP-based modeling strategies as summarized in Table 7. Its design centers on two ideas that jointly target representational fidelity and computational efficiency:

1. **Phase-conditioned processing.** Unlike models that treat periodic structure as an afterthought—using fixed sinusoidal positional encodings or shallow seasonal decomposition—PhiMixer uses the phase ϕ as a shared conditioning signal for the principal temporal and cross-channel transformations. This modulates each mixer block via FiLM, letting the network adjust its behavior based on the current position within the cycle. PILCM also generates its channel-interaction matrices directly from the phase embedding, letting cross-channel dependencies adapt to the temporal phase at the same $O(Cdr)$ cost.
2. **Efficiency through linear channel mixing: avoiding $\mathcal{O}(C^2)$.** PhiMixer applies PILCM with a rank-r bottleneck ($r \ll d$) to couple channels at cost $O(Cdr)$, avoiding the quadratic $\mathcal{O}(C^2)$ complexity of dense channel attention. By performing cross-channel interaction in a compressed feature space and using STARPool for global context sharing, PhiMixer aims for representational capacity closer to CD architectures while keeping the computational profile close to CI. Unlike channel-attention mechanisms that compute pairwise interactions between all channels, PILCM combines phase-conditioned low-rank projections with mean-pooling to capture cross-channel dependencies without the quadratic cost.

A.2 Discussion of Design Choices

The circular distance γ_{ij} is used as a soft bias added to the attention logits, not as a hard constraint. The learned query–key product can override this prior whenever non-cyclic structure is present, and the decay parameters a and b are learned from data. Our RRF ablation (Table 3) shows that removing RRF degrades performance across datasets, indicating that the prior provides net benefit even when the look-back window is only approximately cyclic.

PILCM compresses channel information via a rank-r projection before aggregating with a channel-mean. This is a deliberate complexity trade-off: replacing the mean with learned channel attention would incur $\mathcal{O}(C^2)$ cost, defeating the linear-scaling design. The asymmetry lost by uniform mean-pooling is partially recovered by the phase-conditioned projection operators $D(\psi)$ and $U(\psi)$, which vary with cycle position and can modulate which channels contribute most strongly at each phase. Extending PILCM with learned asymmetric aggregation is a natural direction for future work, though it may trade off the linear-scaling design that motivates PILCM.

Table 8: Benchmark Datasets for Long-Term Forecasting.

Task	Datasets	Features	Horizons	Dataset Size	Information
Forecasting	ETTh1	7	{96, 192, 336, 720}	(8,545, 2,881, 2,881)	Electricity (Hourly)
	ETTh2	7	{96, 192, 336, 720}	(8,545, 2,881, 2,881)	Electricity (Hourly)
	ETTm1	7	{96, 192, 336, 720}	(34,465, 11,521, 11,521)	Electricity (15 mins)
	ETTm2	7	{96, 192, 336, 720}	(34,465, 11,521, 11,521)	Electricity (15 mins)
	Electricity (ECL)	321	{96, 192, 336, 720}	(18,317, 2,633, 5,261)	Electricity (Hourly)
	Traffic	862	{96, 192, 336, 720}	(12,185, 1,757, 3,509)	Transportation (Hourly)
	Weather	21	{96, 192, 336, 720}	(36,792, 5,271, 10,540)	Weather (10 mins)
	Exchange	8	{96, 192, 336, 720}	(5,120, 665, 1,422)	Exchange Rate (Daily)
	PEMS03	358	{12, 24, 48, 96}	(15,724, 5,241, 5,243)	Transportation (5 mins)
	PEMS04	307	{12, 24, 48, 96}	(10,195, 3,398, 3,399)	Transportation (5 mins)
	PEMS07	883	{12, 24, 48, 96}	(16,934, 5,644, 5,646)	Transportation (5 mins)
	PEMS08	170	{12, 24, 48, 96}	(10,713, 3,571, 3,572)	Transportation (5 mins)

Table 9: Period configuration per dataset family. The phase period $\mathcal{P}$ is used in Eq. equation 2; the cyclic-mean periods $\{P_k\}_{k=1}^{K_p}$ in Eq. equation 1.

Dataset	**Sampling**	$\mathcal{P}$	$\{P_k\}_{k=1}^{K_p}$
ETTh1/ETTh2, ECL, Traffic	Hourly	24	$\{24, 168\}$
ETTm1/ETTm2	15-min	96	$\{96, 672\}$
Weather	10-min	144	$\{144\}$
PEMS03/04/07/08	5-min	288	$\{288\}$
Exchange	Daily	7	$\{7\}$

B Implementation Details

B.1 Benchmark Dataset Details

We evaluated PhiMixer on twelve publicly available benchmark datasets (Table 8). These comprise four ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2) for electricity transformer temperature, the ECL dataset covering electricity consumption from 321 clients, the Traffic dataset with 862 sensors, the Weather dataset with 21 climate variables, the Exchange dataset with 8 currency exchange rates, and four PEMS traffic datasets (PEMS03, PEMS04, PEMS07, PEMS08). All datasets are preprocessed and evaluated following standard protocols (Liu et al., 2024; Nie et al., 2023; Wu et al., 2023) and are publicly available (Lin, 2025; Liu et al., 2022; Chen & data, 2025; Wu et al., 2022).

B.2 Period Configuration Details

The phase embedding uses K harmonics of the dominant period, producing a $P_{\text{dim}} = 2K + 1$ dimensional phase-feature vector. Here K is a global hyperparameter independent of the periods $\mathcal{P}$ and $\{P_k\}_{k=1}^{K_p}$: it controls only the frequency resolution of the phase embedding. This deterministic, parameter-free embedding provides a continuous conditioning signal consumed by the learnable FiLM and PILCM modules. The phase period $\mathcal{P}$ is derived from each dataset's sampling frequency as the number of samples per day, $\mathcal{P} = \lfloor 1440/\Delta t \rfloor$, where Δt is the sampling interval in minutes and 1440 is the number of minutes per day; for the daily Exchange dataset, where this yields $\mathcal{P} = 1$, we instead use the weekly period $\mathcal{P} = 7$. The phase period $\mathcal{P}$ is used in Eq. equation 2; the cyclic-mean periods $\{P_k\}_{k=1}^{K_p}$ in Eq. equation 1 extend $\mathcal{P}$ with a weekly cycle $7\mathcal{P}$ where the training window spans multiple full weeks; otherwise, $\{P_k\}_{k=1}^{K_p}$ reduces to the daily cycle alone, as for Weather ($\{144\}$) and PEMS ($\{288\}$). We use $K = 4$ ($P_{\text{dim}} = 9$) throughout, comprising four harmonics of the dominant period and capturing the primary cycle and its first three overtones while keeping the FiLM and PILCM conditioning modules compact. The resulting configuration is summarized in Table 9.

B.3 Baseline Details

We compare PhiMixer against a comprehensive set of state-of-the-art forecasting models spanning diverse architectural paradigms, including linear models, MLP-based models, Transformer-based models, TCN-based models, and recent efficient forecasting architectures.

For the ETT, ECL, Traffic, Weather, and Exchange datasets, we compare with recent baselines including Time-o1 (Wang et al., 2026), TimeMixer++ (Wang et al., 2025b), OLinear (Yue et al., 2025), GCMNet (Liu et al., 2026), SEG-MOE (Ortigossa & Segal, 2026), SOFTS (Han et al., 2024a), FilterNet (TexFilter) (Yi et al., 2024), PatchTST (Nie et al., 2023), iTransformer (Liu et al., 2024), TimesNet (Wu et al., 2023), and TSMixer (Chen et al., 2023).

For PEMS traffic datasets, we compare with MSTN-Transformer (Shevtekar & Maurya, 2026), MSTN-BiLSTM (Shevtekar & Maurya, 2026), iTransformer (Liu et al., 2024), ModernTCN (donghao & wang xue, 2024), PatchTST (Nie et al., 2023), TSMixer (Chen et al., 2023), SOFTS (Han et al., 2024a), and Autoformer (Wu et al., 2022).

B.4 Metric Details

We evaluate long-term forecasting performance using Mean Squared Error (MSE) and Mean Absolute Error (MAE), following the standard protocols of TimesNet (Wu et al., 2023), iTransformer (Liu et al., 2024), SOFTS (Han et al., 2024a), and TimeMixer++ (Wang et al., 2025b).

Let $Y \in \mathbb{R}^{H\times C}$ denote the ground truth and $\hat{Y} \in \mathbb{R}^{H\times C}$ denote the predictions, where H is the forecast horizon and C is the number of channels. The metrics are defined as:

MSE measures the average squared difference between predictions and ground truth:

$$\text{MSE} = \frac{1}{H \cdot C} \sum_{i=1}^{H} \sum_{c=1}^{C} \left(Y_{i,c} - \hat{Y}_{i,c} \right)^2 . \tag{19}$$

MAE measures the average absolute difference between predictions and ground truth:

$$\text{MAE} = \frac{1}{H \cdot C} \sum_{i=1}^{H} \sum_{c=1}^{C} \left| Y_{i,c} - \hat{Y}_{i,c} \right| . \tag{20}$$

Lower values of both MSE and MAE indicate better forecasting performance.

B.5 Experimental Details

B.5.1 Data Preprocessing and Evaluation Protocol

We adopt standard preprocessing and evaluation protocols consistent with prior works (Liu et al., 2024; Wu et al., 2023; Shevtekar & Maurya, 2026; Nie et al., 2023). To maintain temporal causality, all datasets are divided chronologically into training, validation, and test sets. For the ETT datasets, we employ a 60%/20%/20% split ratio. For ECL, Traffic, Weather, and Exchange, we use a 70%/10%/20% split. The PEMS datasets (PEMS03, PEMS04, PEMS07, PEMS08) follow a 60%/20%/20% split as in iTransformer (Liu et al., 2024) and MSTN (Shevtekar & Maurya, 2026). The raw time series is first partitioned into the three splits, after which sliding windows are independently constructed within each segment to avoid data leakage. Normalization statistics are estimated solely from the training portion and then applied to both validation and test sets. Following established benchmark protocols (Wu et al., 2023; Liu et al., 2024), we fix the look-back window length at $L = 96$ for all datasets. The forecast horizons are set to $H \in \{96, 192, 336, 720\}$ for ETT, ECL, Traffic, Weather, and Exchange, and $H \in \{12, 24, 48, 96\}$ for the PEMS datasets. All results are reported using MSE and MAE on the normalized scale, following standard evaluation practices in the literature.

B.5.2 Training and Model Configuration

PhiMixer is implemented in Python 3.13.1 using PyTorch 2.7.1 and trained on an NVIDIA DGX A100 server equipped with 8 NVIDIA A100-SXM4 GPUs (40 GB VRAM each). The architec-

Table 10: PhiMixer model architecture hyperparameters. Configuration shown for the standard ETTh1 setting (L=96, H=96). Period values are specified in time steps according to the sampling frequency of each dataset.

Parameter	**Value**
Hidden dimension d	64
MixerBlocks M	1
Patch length p	16
Number of patches $N = L/p$	$96/16 = 6$
Cyclic harmonics K	4
Phase dimension $P_{\text{dim}} = 2K+1$	9
PILCM rank r	8
RRF attention heads	4
Dominant period $\mathcal{P}$ (hourly)	24
Weekly period (hourly time steps)	168

ture includes a Cyclic Mean path with periods $\{24, 168\}$ for seasonal decomposition, and a nonlinear patch-mixing path comprising patch embedding, phase-conditioned MixerBlocks with RRF, STARPool for global context aggregation, and the PILCM for efficient cross-channel coupling. A dual-path prediction head combines a linear skip connection with a gated nonlinear branch. The model uses a hidden dimension of $d = 64$, a single MixerBlock ($M = 1$), patch length $p = 16$ (giving $N = 6$ patches for $L = 96$), $K = 4$ phase harmonics (phase feature dimension $P_{\text{dim}} = 9$), and PILCM rank $r = 8$. Dropout is set to 0.3 for regularization. The model inference latency is measured using `torch.profiler` with batch size 1. Training is conducted using the AdamW optimizer with an initial learning rate of 5×10^{-4} and weight decay of 1×10^{-4}, along with gradient clipping at a maximum norm of 5.0. The learning rate is decayed using a cosine annealing schedule from 5×10^{-4} to 1×10^{-6} over the course of training. We use a batch size of 64 for all experiments, consistent with the standard practice in prior work (Wang et al., 2024). Models are trained for up to 50 epochs with early stopping applied after 6 epochs of no improvement on validation MSE, using a composite loss function comprising balanced MSE, decomposition-balancing (DB), frequency-domain (FreDF), and correlation consistency (CCC) terms. All experiments use a fixed look-back window of $L = 96$. Forecast horizons are $H \in \{96, 192, 336, 720\}$ for ETT, ECL, Traffic, Weather, and Exchange, and $H \in \{12, 24, 48, 96\}$ for PEMS datasets. Hyperparameters are selected following standard practices in the literature. To ensure statistical robustness, all experiments are repeated with 5 different random seeds, and results are reported as the mean $\pm$ standard deviation. The loss weights are $\lambda_{\text{DB}} = 1.0$, $\lambda_{\text{F}} = 0.5$, and $\lambda_{\text{CCC}} = 0.05$. The DB loss uses smoothing rates $\mathcal{A} = \{0.1, 0.3, 0.5\}$ with $\beta = 0.6$ to balance trend and residual components.

B.6 Hyperparameters

This section provides the hyperparameter configuration used for PhiMixer. The architectural and optimization hyperparameters are fixed across datasets and forecast horizons. Dataset-specific settings are limited to the periodic configuration used by the cyclic decomposition and the prediction horizons, with period values specified in time steps according to the sampling frequency of each dataset. Table 10 details the architectural hyperparameters, Table 11 lists the optimization settings, and Table 12 specifies the composite loss weights.

B.6.1 Loss Function

This section provides the mathematical definitions of the loss terms introduced in Section 3. The total objective is

$$\mathcal{L} = \mathcal{L}_{\text{bal}} + \lambda_{\text{DB}}\mathcal{L}_{\text{DB}} + \lambda_{\text{F}}\mathcal{L}_{\text{FreDF}} + \lambda_{\text{C}}\mathcal{L}_{\text{CCC}}, \tag{21}$$

Channel-balanced MSE. Let $a = \hat{Y} - Y$ denote the prediction error, with per-channel mean squared error $\overline{a_c^2} = \frac{1}{BH}\sum_{i,t} a_{i,t,c}^2$. The channel-balanced term is

Table 11: PhiMixer training hyperparameters used across the evaluated datasets and forecast horizons.

Parameter	Value
Optimizer	AdamW
Learning rate (initial)	5×10^{-4}
Learning rate (final)	1×10^{-6}
Learning rate schedule	Cosine annealing
Weight decay	1×10^{-4}
Gradient clipping norm	5.0
Maximum epochs	50
Early-stopping patience	6
Batch size	64
Dropout rate	0.3
Path-drop probability	0.1

Table 12: PhiMixer composite loss hyperparameters.

Loss Component	Weight / Value
$\mathcal{L}_{\text{bal}}$ (balanced MSE)	Base loss
$\mathcal{L}_{\text{DB}}$ (decomposition-balancing)	$\lambda_{\text{DB}} = 1.0$
$\mathcal{L}_{\text{FreDF}}$ (frequency-domain)	$\lambda_{\text{F}} = 0.5$
$\mathcal{L}_{\text{CCC}}$ (correlation consistency)	$\lambda_{\text{C}} = 0.05$
DB smoothing rates	$\mathcal{A} = \{0.1, 0.3, 0.5\}$
DB trend-seasonal balance	$\beta = 0.6$
DB frequency weight	$\lambda_{\text{freq}} = 0.3$

$$\mathcal{L}_{\text{bal}} = \frac{1}{C} \sum_{c=1}^{C} w_c \, \overline{a_c^2}, \qquad w_c = \frac{\max_c \overline{a_c^2}}{\overline{a_c^2} + \epsilon}, \tag{22}$$

where the weights w_c are detached from the gradient computation and $\epsilon = 10^{-8}$ ensures numerical stability.

Decomposition-balancing loss (DB). We adopt the decomposition-aware loss design of DBLoss (Qiu et al., 2025b). Let $T_\alpha(Y) = \text{EMA}_\alpha(Y)$ denote the exponential moving average of Y at smoothing rate α, and let $R_\alpha(Y) = Y - T_\alpha(Y)$ denote the corresponding residual. For a set of smoothing rates $\mathcal{A} = \{0.1, 0.3, 0.5\}$, the residual loss is

$$\mathcal{L}^{\alpha}_{\text{residual}} = \text{MSE}(R_\alpha(\hat{Y}), R_\alpha(Y)) + \lambda_{\text{freq}} \cdot \left\| \mathcal{F}(R_\alpha(\hat{Y})) - \mathcal{F}(R_\alpha(Y)) \right\|_1, \tag{23}$$

where $\mathcal{F}$ denotes the real fast Fourier transform. The trend loss is scaled by a stop-gradient factor,

$$\mathcal{L}^{\alpha}_{\text{trend}} = \text{MAE}(T_\alpha(\hat{Y}), T_\alpha(Y)) \cdot \text{sg}\left(\frac{\mathcal{L}^{\alpha}_{\text{residual}}}{\text{MAE}(T_\alpha(\hat{Y}), T_\alpha(Y)) + \epsilon} \right), \tag{24}$$

and the decomposition-balancing loss averages over smoothing rates:

$$\mathcal{L}_{\text{DB}} = \frac{1}{|\mathcal{A}|} \sum_{\alpha \in \mathcal{A}} \left[\beta \, \mathcal{L}^{\alpha}_{\text{residual}} + (1 - \beta) \, \mathcal{L}^{\alpha}_{\text{trend}} \right], \tag{25}$$

with $\beta = 0.6$.

Frequency-domain loss (FreDF). We adopt the frequency-domain formulation of FreDF (Wang et al., 2025a). Let $\mathcal{F}(\cdot)$ denote the real FFT and let $F = \lfloor H/2 \rfloor + 1$ denote the number of frequency bins. The frequency-domain loss is

$$\mathcal{L}_{\text{FreDF}} = \frac{1}{BFC} \sum_{b=1}^{B} \sum_{f=1}^{F} \sum_{c=1}^{C} \left| \mathcal{F}(\hat{Y})_{b,f,c} - \mathcal{F}(Y)_{b,f,c} \right|. \tag{26}$$

Cross-channel correlation consistency (CCC). For any tensor $Z \in \mathbb{R}^{B\times H\times C}$, define its correlation matrix as $\mathrm{Corr}(Z) = \frac{\tilde{Z}^\top \tilde{Z}}{BH-1}$, where $\tilde{Z}$ is obtained by reshaping Z to $\mathbb{R}^{(BH)\times C}$ and standardizing each channel to zero mean and unit variance over the pooled (BH) observations. The CCC loss is

$$\mathcal{L}_{\text{CCC}} = \frac{1}{C^2} \sum_{p=1}^{C} \sum_{q=1}^{C} \left| \mathrm{Corr}(\hat{Y})_{pq} - \mathrm{Corr}(Y)_{pq} \right|. \tag{27}$$

We set the loss weights $\lambda_{\text{DB}} = 1$, $\lambda_{\text{F}} = 0.5$, $\lambda_{\text{freq}} = 0.3$, and $\lambda_{\text{C}} = 0.05$.

C Full Results

Tables 13 and 14 present the complete per-horizon results on the eight main benchmarks and the four PEMS benchmarks, respectively. On the main benchmarks, PhiMixer attains the lowest average MSE on ETTm1, ETTm2, and ECL, and the lowest average MAE on ETTm1, ETTm2, ECL, and Weather. On datasets where PhiMixer is not the lowest, the gap remains modest across all cases. The per-horizon results show that PhiMixer's relative performance varies across datasets and prediction horizons, with gains observed at shorter horizons on ECL and at longer horizons on ETTm2. On the PEMS benchmarks, PhiMixer attains the lowest average MSE and MAE on PEMS03, PEMS07, and PEMS08, while remaining close to the best-performing baseline on PEMS04. This pattern indicates that PhiMixer performs well on the evaluated high-dimensional traffic forecasting tasks, although its relative performance varies across prediction horizons.

When averaged across the eight main benchmarks, PhiMixer achieves the lowest average MAE and the second-lowest average MSE, while it achieves both the lowest average MSE and MAE across the four PEMS benchmarks. In terms of first-place counts, PhiMixer secures the highest number of first-place MSE and MAE rankings on both the main and PEMS benchmarks, across all compared methods and prediction horizons. Relative to the best-competing baselines, PhiMixer shows improvements in average MAE on the main benchmarks and in both average MSE and MAE on the PEMS benchmarks. These gains are modest and are obtained with a parameter footprint of approximately 126K parameters. Taken together, these results indicate that PhiMixer provides competitive forecasting accuracy across the evaluated benchmarks, with particularly strong average performance on ETTm1, ETTm2, ECL, PEMS03, PEMS07, and PEMS08, while remaining close to the best-performing baseline on the other datasets. The per-horizon results also show that the model's relative performance varies across datasets and forecast horizons.

D Detailed Ablation Results

To isolate the contribution of each architectural component, we systematically remove individual modules and evaluate performance across ETTh1, ETTh2, PEMS03, and PEMS04. We consider eight variants: *w/o PILCM* removes the phase-informed low-rank channel mixer entirely; *w/o FiLM* removes the phase-aware Feature-wise Linear Modulation; *w/o STARPool* removes the star aggregate–dispatch pool; *w/o RRF* removes the rhythm-aware relative attention; *w/o CyclicMean* removes the multi-period cyclic-mean decomposition; *w/o P. in PILCM* keeps PILCM but removes its phase conditioning input; *w/o P. in FiLM* keeps FiLM but removes its phase conditioning input; and *w/o Phase* removes phase conditioning from both FiLM and PILCM. The phase-related variants isolate the effect of phase conditioning within each module, distinguishing phase-aware computation from the module's core functionality.

Table 15 presents the complete per-horizon ablation results on ETTh1, ETTh2, PEMS03, and PEMS04. RRF removal causes the largest degradation on ETTh1 at $H = 720$ ($0.481 \rightarrow 0.516$, +7.3%), with consistent 5–8% increases across datasets, indicating its role in capturing periodic dependencies. PILCM removal also consistently degrades performance, with 2–6% increases on

Table 13: Multivariate long-term forecasting results. Input look-back window $L = 96$. Prediction horizons $H \in \{96, 192, 336, 720\}$. Avg is averaged from all four prediction lengths. **Red**: 1st rank, **Blue**: 2nd rank. We evaluate each model five times and report the mean (standard deviation); results marked with underline are not significantly worse than the first rank (Wilcoxon signed-rank test with Holm–Bonferroni correction, $\alpha = 0.05$). Example: 0.010(0.001) means 0.010 ± 0.001. TimeM.++: TimeMixer++, Impr.: Improvement %.

Dataset	H	PhiMixer		OLinear		TimeM.++		Time-o1		GCMNet		SEG-MOE		SOFTS		TexFilter		PatchTST	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.365(0.01)	**0.383(0.01)**	**0.360**	**0.382**	**0.361**	0.403	0.368	0.391	0.390	0.406	0.370	0.401	0.381	0.399	0.382	0.402	0.394	0.406
	192	**0.420(0.01)**	**0.413(0.02)**	**0.416**	**0.414**	**0.416**	0.441	0.424	0.422	0.438	0.434	0.435	0.430	0.435	0.431	0.430	0.429	0.440	0.435
	336	0.466(0.02)	0.439(0.02)	**0.457**	**0.438**	**0.430**	**0.434**	0.467	0.441	0.474	0.454	0.507	0.456	0.480	0.452	0.472	0.451	0.491	0.462
	720	0.481(0.02)	0.471(0.02)	**0.463**	**0.462**	0.467	**0.451**	**0.465**	0.463	0.477	0.478	0.579	0.485	0.499	0.488	0.481	0.473	0.487	0.479
	Avg	0.433	**0.427**	**0.424**	**0.424**	**0.419**	**0.432**	0.431	0.429	0.445	0.443	0.473	0.443	0.449	0.443	0.441	0.439	0.453	0.446
ETTh2	96	**0.271(0.01)**	**0.324(0.01)**	0.284	0.329	**0.276**	**0.328**	0.282	0.330	0.290	0.341	0.307	0.347	0.297	0.347	0.293	0.343	0.288	0.340
	192	**0.348(0.01)**	**0.371(0.01)**	0.360	**0.379**	**0.342**	**0.379**	0.359	0.381	0.375	0.395	0.390	0.402	0.373	0.394	0.374	0.396	0.376	0.395
	336	**0.389(0.02)**	**0.410(0.02)**	0.409	0.415	**0.346**	**0.398**	0.394	0.414	0.414	0.428	0.434	0.441	0.410	0.426	0.417	0.430	0.440	0.451
	720	0.414(0.02)	0.429(0.02)	0.415	0.431	**0.392**	**0.415**	**0.400**	**0.427**	0.423	0.440	0.474	0.474	0.411	0.433	0.449	0.460	0.436	0.453
	Avg	**0.356**	**0.384**	0.367	0.389	**0.339**	**0.380**	0.359	0.388	0.376	0.401	0.401	0.416	0.373	0.400	0.383	0.407	0.385	0.410
ETTm1	96	**0.295(0.01)**	**0.336(0.01)**	**0.302**	**0.334**	0.310	**0.334**	0.321	0.357	0.330	0.370	0.323	0.363	0.325	0.361	0.321	0.361	0.329	0.365
	192	**0.338(0.01)**	**0.361(0.01)**	0.357	0.363	**0.348**	**0.362**	0.360	0.378	0.369	0.389	0.392	0.395	0.375	0.389	0.367	0.387	0.380	0.394
	336	**0.370(0.02)**	**0.387(0.02)**	0.387	**0.385**	**0.376**	0.391	0.389	0.400	0.398	0.410	0.455	0.431	0.405	0.412	0.401	0.409	0.400	0.410
	720	**0.432(0.02)**	**0.422(0.02)**	0.452	0.426	**0.440**	**0.423**	0.447	0.435	0.457	0.442	0.583	0.486	0.466	0.447	0.477	0.448	0.475	0.453
	Avg	**0.359**	**0.377**	0.375	**0.377**	**0.368**	**0.378**	0.379	0.393	0.389	0.403	0.438	0.419	0.393	0.402	0.392	0.401	0.396	0.406
ETTm2	96	**0.147(0.01)**	**0.232(0.01)**	**0.169**	0.249	0.170	0.245	0.172	0.251	0.170	0.254	0.191	0.273	0.180	**0.233**	0.175	0.258	0.184	0.264
	192	**0.213(0.01)**	**0.278(0.01)**	0.232	**0.290**	**0.229**	0.291	0.235	0.294	0.236	0.296	0.266	0.323	0.246	0.306	0.240	0.301	0.246	0.306
	336	**0.275(0.01)**	**0.318(0.02)**	**0.291**	**0.328**	0.303	0.343	0.293	0.333	0.296	0.335	0.337	0.367	0.319	0.352	0.311	0.347	0.308	0.346
	720	**0.380(0.02)**	**0.380(0.02)**	0.389	**0.387**	**0.373**	0.399	0.388	0.389	0.396	0.392	0.453	0.431	0.405	0.401	0.414	0.405	0.409	0.402
	Avg	**0.254**	**0.302**	0.270	**0.314**	**0.269**	0.320	0.272	0.317	0.275	0.319	0.312	0.348	0.288	0.323	0.285	0.328	0.287	0.330
ECL	96	**0.125(0.01)**	**0.217(0.01)**	**0.131**	**0.221**	0.135	0.222	0.145	0.235	0.142	0.241	0.199	0.274	0.143	0.233	0.147	0.245	0.164	0.251
	192	**0.140(0.01)**	**0.231(0.01)**	0.150	0.238	**0.147**	**0.235**	0.159	0.249	0.159	0.254	0.223	0.298	0.158	0.248	0.160	0.250	0.173	0.262
	336	**0.158(0.02)**	**0.249(0.02)**	0.165	0.254	**0.164**	**0.245**	0.173	0.264	0.176	0.270	0.254	0.329	0.178	0.269	0.173	0.267	0.190	0.279
	720	**0.186(0.02)**	**0.273(0.02)**	**0.191**	**0.279**	0.212	0.310	0.203	0.292	0.203	0.294	0.326	0.390	0.218	0.305	0.210	0.309	0.230	0.313
	Avg	**0.152**	**0.243**	**0.159**	**0.248**	0.165	0.253	0.170	0.260	0.170	0.265	0.250	0.323	0.174	0.264	0.173	0.268	0.189	0.276
Weather	96	**0.145(0.01)**	**0.186(0.01)**	**0.153**	**0.190**	0.155	0.205	0.169	0.219	0.165	0.216	0.178	0.213	0.166	0.208	0.162	0.207	0.176	0.217
	192	**0.195(0.01)**	**0.233(0.01)**	**0.200**	**0.235**	0.201	0.245	0.210	0.258	0.212	0.255	0.227	0.256	0.217	0.253	0.210	0.250	0.221	0.256
	336	**0.252(0.02)**	**0.274(0.02)**	0.258	0.280	**0.237**	**0.265**	0.259	0.297	0.271	0.297	0.285	0.300	0.282	0.300	0.266	0.290	0.275	0.296
	720	0.331(0.02)	**0.327(0.02)**	0.337	**0.333**	**0.312**	0.334	0.327	0.349	0.345	0.346	0.373	0.361	0.356	0.351	**0.320**	0.340	0.352	0.346
	Avg	**0.231**	**0.255**	0.237	**0.260**	**0.226**	0.262	0.241	0.281	0.248	0.278	0.266	0.283	0.255	0.278	0.240	0.272	0.256	0.279
Exchange	96	**0.070(0.01)**	**0.188(0.01)**	**0.082**	**0.200**	0.085	0.214	0.086	0.206	0.087	0.206	–	–	–	–	0.091	0.211	0.088	0.205
	192	**0.169(0.02)**	**0.289(0.01)**	**0.171**	**0.293**	0.175	0.313	0.182	0.305	0.181	0.302	–	–	–	–	0.186	0.305	0.176	0.299
	336	0.332(0.02)	**0.412(0.01)**	0.331	0.414	**0.316**	0.420	0.331	0.417	0.333	0.418	–	–	–	–	0.380	0.449	**0.301**	**0.397**
	720	0.904(0.02)	0.706(0.02)	0.837	**0.688**	0.851	0.689	**0.831**	0.691	**0.830**	**0.685**	–	–	–	–	0.896	0.712	0.901	0.714
	Avg	0.369	**0.399**	**0.355**	**0.399**	**0.357**	0.409	**0.358**	0.405	0.358	**0.403**	–	–	–	–	0.388	0.419	0.367	0.404
Traffic	96	0.451(0.00)	0.285(0.01)	0.398	**0.226**	**0.392**	0.253	0.393	0.265	0.405	0.263	0.503	0.322	**0.376**	**0.251**	0.430	0.294	0.427	0.272
	192	0.456(0.01)	0.287(0.01)	0.439	**0.241**	**0.402**	**0.258**	0.410	0.275	0.424	0.271	0.501	0.317	**0.398**	**0.261**	0.452	0.307	0.454	0.289
	336	0.467(0.02)	0.292(0.02)	0.464	**0.250**	0.428	**0.263**	**0.421**	0.280	0.449	0.283	0.517	0.316	**0.415**	0.269	0.470	0.316	0.450	0.282
	720	0.494(0.02)	0.308(0.02)	0.502	**0.270**	**0.441**	**0.282**	0.451	0.298	0.511	0.307	0.602	0.364	**0.447**	0.287	0.498	0.323	0.484	0.301
	Avg	0.467	0.293	0.451	**0.247**	**0.416**	**0.264**	0.419	0.280	0.447	0.281	0.531	0.330	**0.409**	**0.267**	0.463	0.310	0.454	0.286
1st Count		**19**	**22**	**4**	**12**	**12**	**6**	**0**	**0**	**1**	**1**	**0**	**0**	**4**	**0**	**0**	**0**	**1**	**1**
Average		0.328	0.335	0.330	0.337	0.322	0.337	0.360	0.382	0.415	0.372	0.336	0.352	0.369	0.397	0.351	0.376	0.348	0.355
Impr. %		–	–	0.61	0.59	-1.86	0.59	8.89	12.30	20.96	9.95	2.38	4.83	11.11	15.62	6.55	10.90	5.75	5.63

ETTh and up to 10.2% on PEMS04, suggesting that phase-informed cross-channel coupling contributes to forecasting accuracy. FiLM removal shows notable degradation on PEMS (PEMS03: 0.188 → 0.218, +16.0%; PEMS04: 0.196 → 0.242, +23.5%) and moderate degradation on ETTh (ETTh1: 0.365 → 0.381, +4.4%; ETTh2: 0.271 → 0.286, +5.5%), indicating that phase-aware modulation is particularly relevant for high-dimensional traffic forecasting. CyclicMean removal also produces larger effects on PEMS datasets (PEMS04: 0.196 → 0.237, +20.9%), reflecting the strong daily periodicities in traffic data. STARPool removal yields mixed results, with a slight improvement on PEMS03 at $H = 96$ (0.188 → 0.187), while PILCM removal shows smaller and dataset-dependent effects (0.188 → 0.191 on PEMS03; 0.196 → 0.216 on PEMS04).

Phase conditioning enhances both modules: Removing phase from PILCM or FiLM is consistently less severe than removing the entire module (e.g., ETTh1 H=96: w/o Phase in PILCM 0.378 vs w/o

Table 14: Long-term forecasting evaluation results on PEMS datasets. Input look-back window $L = 96$. Prediction horizons $H \in \{12, 24, 48, 96\}$. Red/Blue: First/Second ranks. We evaluate each model five times and report the mean (standard deviation); results marked with underline are not significantly worse than the first rank (Wilcoxon signed-rank test with Holm–Bonferroni correction, $\alpha = 0.05$). Example: 0.010(0.001) means 0.010 ± 0.001. TimeM.++: TimeMixer++, Tra.: Transformer, BiL.: BiLSTM, M.TCN: ModernTCN, Impr.: Improvement %.

Dataset	H	PhiMixer		TimeM.++		MSTN-Tra.		MSTN-BiL.		M.TCN		TSMixer		TimeMixer		FITS		DLinear		PatchTST		Autoformer	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	12	**0.064(0.00)**	**0.169(0.00)**	0.097	0.208	0.131	0.237	0.141	0.249	0.112	0.221	**0.075**	**0.187**	0.076	0.188	0.117	0.226	0.122	0.243	0.099	0.216	0.277	0.387
	24	**0.086(0.01)**	**0.196(0.01)**	0.120	0.230	0.135	0.241	0.151	0.259	0.173	0.281	**0.113**	0.238	**0.113**	**0.226**	0.235	0.324	0.201	0.317	0.142	0.259	0.422	0.466
	48	**0.128(0.02)**	**0.241(0.02)**	**0.170**	0.272	0.174	**0.268**	0.178	0.280	0.307	0.395	0.195	0.320	0.191	0.292	0.541	0.521	0.333	0.425	0.211	0.319	0.806	0.679
	96	**0.188(0.02)**	0.301(0.02)	0.274	0.342	**0.182**	**0.281**	0.197	**0.299**	1.041	0.779	0.266	0.380	0.288	0.363	1.062	0.790	0.457	0.515	0.269	0.370	0.710	0.634
	Avg	**0.117**	**0.227**	0.165	0.263	**0.156**	**0.257**	0.167	0.272	0.408	0.419	0.162	0.281	0.167	0.267	0.489	0.465	0.278	0.375	0.180	0.291	0.554	0.542
PEMS04	12	**0.076(0.00)**	**0.182(0.00)**	0.099	0.214	0.105	0.210	0.114	0.222	0.132	0.245	**0.085**	**0.195**	0.092	0.204	0.129	0.239	0.148	0.272	0.105	0.224	0.562	0.577
	24	**0.097(0.01)**	**0.210(0.01)**	0.115	0.231	**0.111**	**0.217**	0.120	0.228	0.244	0.338	0.112	0.228	0.128	0.243	0.246	0.337	0.224	0.340	0.153	0.275	0.637	0.617
	48	0.137(0.01)	0.257(0.02)	0.144	0.261	**0.119**	**0.229**	**0.127**	**0.239**	0.452	0.482	0.159	0.278	0.213	0.315	0.568	0.539	0.355	0.437	0.229	0.339	1.002	0.775
	96	0.196(0.02)	0.317(0.02)	0.185	0.297	**0.126**	**0.235**	**0.137**	**0.247**	1.127	0.818	0.190	0.313	0.307	0.384	1.181	0.843	0.452	0.504	0.291	0.389	0.853	0.708
	Avg	0.127	0.242	0.136	0.251	**0.115**	**0.223**	**0.125**	**0.234**	0.488	0.471	0.136	0.254	0.185	0.287	0.531	0.489	0.295	0.388	0.195	0.307	0.764	0.669
PEMS07	12	**0.063(0.00)**	**0.159(0.00)**	0.090	0.197	0.131	0.221	0.179	0.254	0.085	0.196	**0.070**	**0.177**	0.073	0.184	0.109	0.222	0.115	0.242	0.095	0.207	0.201	0.330
	24	**0.094(0.01)**	**0.199(0.02)**	0.110	**0.219**	0.142	0.232	0.183	0.260	0.127	0.245	**0.070**	0.221	0.111	**0.219**	0.230	0.327	0.210	0.329	0.150	0.262	0.304	0.402
	48	**0.144(0.02)**	**0.239(0.02)**	**0.149**	0.256	0.155	**0.241**	0.189	0.265	0.267	0.380	0.157	0.265	0.237	0.328	0.551	0.531	0.398	0.458	0.253	0.340	0.422	0.472
	96	0.229(0.02)	0.335(0.02)	0.258	0.359	**0.168**	**0.252**	**0.196**	**0.273**	0.736	0.673	0.268	0.342	0.303	0.354	1.112	0.809	0.594	0.553	0.346	0.404	0.519	0.546
	Avg	**0.133**	**0.233**	0.152	0.258	**0.149**	**0.237**	0.187	0.263	0.304	0.374	0.150	0.251	0.181	0.271	0.500	0.472	0.329	0.395	0.211	0.303	0.362	0.438
PEMS08	12	**0.078(0.00)**	**0.179(0.00)**	0.119	0.222	0.291	0.297	0.309	0.314	0.125	0.239	0.095	0.203	**0.091**	**0.201**	0.122	0.233	0.154	0.276	0.168	0.232	0.467	0.503
	24	**0.115(0.01)**	**0.218(0.01)**	0.149	0.249	0.287	0.299	0.342	0.340	0.238	0.336	0.150	0.257	**0.137**	**0.246**	0.236	0.330	0.248	0.353	0.224	0.281	0.503	0.512
	48	**0.189(0.02)**	**0.284(0.02)**	**0.206**	**0.292**	0.333	0.328	0.344	0.346	0.528	0.534	0.256	0.344	0.265	0.343	0.562	0.540	0.440	0.470	0.321	0.354	0.964	0.729
	96	**0.336(0.02)**	0.370(0.02)	**0.329**	**0.355**	0.362	**0.364**	0.398	0.393	1.150	0.808	0.399	0.415	0.410	0.407	1.216	0.846	0.674	0.565	0.408	0.417	1.021	0.763
	Avg	**0.180**	**0.265**	**0.200**	**0.279**	0.318	0.322	0.348	0.348	0.510	0.479	0.225	0.305	0.226	0.299	0.534	0.487	0.379	0.416	0.280	0.321	0.739	0.627
1st Count		**14**	**14**	**1**	**1**	**5**	**5**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Average		0.139	0.242	0.163	0.263	0.185	0.260	0.207	0.279	0.428	0.436	0.168	0.273	0.190	0.281	0.514	0.478	0.320	0.394	0.217	0.306	0.605	0.569
Impr. %		–	–	14.72	7.98	24.86	6.92	32.85	13.26	67.52	44.50	17.26	11.36	26.84	13.88	72.96	49.37	56.56	38.58	35.94	20.92	77.02	57.47

PILCM 0.382 MSE; w/o Phase in FiLM 0.378 vs w/o FiLM 0.381 MSE). Removing phase from both modules further degrades performance, particularly on PEMS datasets: on PEMS03, average MSE increases from 0.117 to 0.160 (+36.8%), and on PEMS04, from 0.127 to 0.196 (+54.3%). This indicates that phase conditioning enhances both modules, with a more pronounced effect on high-dimensional traffic data.

Overall, RRF and PILCM are the most consistently beneficial components, while FiLM, STARPool, and CyclicMean provide complementary gains that vary with dataset and horizon. Phase conditioning serves as an effective enabler, enhancing both FiLM and PILCM without being the sole driver of performance.

D.1 Phase Period and Hyperparameter Sensitivity

The phase period $\mathcal{P}$ plays a central role in PhiMixer: it determines the conditioning signal $\psi(t_0)$ that modulates the temporal mixer through FiLM and the cross-channel mixer through PILCM. We examine how sensitive performance is to variations of $\mathcal{P}$ around its sampling-derived value.

For each dataset, we train a model with $\mathcal{P}$ set to its sampling-derived value (24 for ETTh1 and ETTh2; 144 for Weather) and to ± 2- and ± 4-step shifts. A "step" corresponds to one sampling interval: 1 hour for ETTh1/ETTh2 and 10 minutes for Weather. Thus, the tested values are $\mathcal{P} \in \{20, 22, 24, 26, 28\}$ for ETTh1/ETTh2 and $\mathcal{P} \in \{140, 142, 144, 146, 148\}$ for Weather. All other hyperparameters are held fixed (L=H=96), and the same random seed is used across all settings so that differences reflect only the phase period.

Table 16 reports the test MSE and MAE for each setting. How sensitive the model is depends on the strength of daily periodicity in the underlying dataset:

1. **ETTh1 (weak daily periodicity).** Performance is essentially insensitive to $\mathcal{P}$: shifting $\mathcal{P}$ by ± 4 steps changes MSE by less than 0.4%. The maximum-to-minimum spread across all five settings is approximately 0.4%, which lies within the run-to-run variation observed across seeds (Table 1). This suggests that phase conditioning is largely inactive on ETTh1, consistent with the relatively weak daily structure of transformer temperature signals at the hourly sampling rate.

Table 15: Complete ablation study on ETTh and PEMS datasets ($H \in \{96, 192, 336, 720\}$ for ETTh, $H \in \{12, 24, 48, 96\}$ for PEMS). **Red** and **Blue** indicate first and second ranks. STARP: STARPool, CyclicM: CyclicMean, P.: Phase.

Dataset	H	Full Model MSE	Full Model MAE	w/o PILCM MSE	w/o PILCM MAE	w/o FiLM MSE	w/o FiLM MAE	w/o STARP. MSE	w/o STARP. MAE	w/o RRF MSE	w/o RRF MAE	w/o CyclicM. MSE	w/o CyclicM. MAE	w/o P.in PILCM MSE	w/o P.in PILCM MAE	w/o P.in FiLM MSE	w/o P.in FiLM MAE	w/o Phase MSE	w/o Phase MAE
ETTh1	96	**0.365**	**0.383**	0.382	0.401	0.381	0.399	0.380	0.399	0.386	0.397	**0.373**	**0.393**	0.379	0.396	0.378	0.395	0.381	0.397
	192	**0.420**	**0.413**	0.436	0.430	0.433	0.428	0.433	0.428	0.437	**0.424**	**0.427**	**0.424**	0.431	0.426	0.430	0.425	0.436	0.429
	336	**0.466**	**0.439**	0.478	0.447	0.473	0.446	0.475	**0.445**	0.479	0.446	**0.470**	**0.445**	0.474	0.446	0.473	0.446	0.484	0.457
	720	**0.481**	**0.471**	0.499	0.485	0.488	0.476	**0.482**	0.473	0.516	0.493	**0.481**	**0.472**	0.495	0.483	0.490	0.480	0.483	0.475
	Avg	**0.433**	**0.427**	0.449	0.441	0.444	0.437	0.443	0.436	0.455	0.440	**0.438**	**0.434**	0.445	0.438	0.443	0.437	0.446	0.440
ETTh2	96	**0.271**	**0.324**	0.288	0.339	0.286	0.339	**0.283**	**0.334**	0.287	0.337	0.284	0.336	0.286	0.336	0.284	0.336	0.286	0.336
	192	**0.348**	**0.371**	0.366	0.390	0.365	0.387	**0.361**	**0.384**	0.365	0.385	0.364	0.386	0.364	0.387	0.363	0.385	0.369	0.388
	336	**0.389**	**0.410**	0.412	0.428	0.409	0.425	0.406	**0.421**	0.409	0.426	**0.403**	0.422	0.408	0.424	0.407	0.423	0.410	0.424
	720	**0.414**	**0.430**	0.429	0.447	0.428	0.446	**0.425**	**0.442**	**0.425**	0.443	0.426	0.443	0.427	0.444	0.426	0.444	0.428	0.444
	Avg	**0.356**	**0.384**	0.374	0.401	0.372	0.399	**0.369**	**0.395**	0.371	0.398	**0.369**	0.397	0.371	0.398	0.370	0.397	0.373	0.398
PEMS03	12	**0.064**	**0.169**	0.069	0.175	0.067	0.174	**0.066**	**0.170**	0.067	0.173	0.069	0.176	0.067	0.173	0.066	0.172	0.072	0.173
	24	**0.086**	**0.196**	0.092	0.202	0.093	0.204	**0.089**	**0.199**	0.090	0.200	0.094	0.206	0.090	0.200	0.090	0.200	0.108	0.217
	48	**0.128**	**0.241**	0.131	0.244	0.144	0.259	**0.129**	**0.242**	0.133	0.245	0.141	0.255	0.130	0.243	0.138	0.249	0.176	0.286
	96	**0.188**	**0.301**	0.191	0.308	0.218	0.328	**0.187**	**0.302**	0.204	0.315	0.208	0.315	0.189	0.305	0.200	0.315	0.290	0.385
	Avg	**0.117**	**0.227**	0.121	0.232	0.131	0.241	**0.118**	**0.228**	0.123	0.233	0.128	0.238	0.119	0.230	0.124	0.234	0.160	0.265
PEMS04	12	**0.076**	**0.182**	0.079	0.185	0.081	0.190	**0.077**	**0.183**	0.079	0.184	0.085	0.194	0.078	0.184	0.079	0.186	0.094	0.201
	24	**0.097**	**0.210**	0.102	0.216	0.108	0.222	**0.098**	**0.211**	0.099	0.213	0.111	0.226	0.099	0.213	0.104	0.217	0.137	0.245
	48	**0.137**	**0.257**	0.146	0.265	0.160	0.280	**0.139**	**0.258**	0.143	0.261	0.160	0.279	0.141	0.260	0.150	0.270	0.216	0.317
	96	**0.196**	**0.317**	0.216	0.338	0.242	0.354	**0.210**	**0.324**	0.212	0.326	0.237	0.348	0.211	0.328	0.225	0.338	0.338	0.419
	Avg	**0.127**	**0.242**	0.136	0.251	0.148	0.262	**0.131**	**0.244**	0.133	0.246	0.148	0.262	0.132	0.246	0.140	0.253	0.196	0.296

Table 16: Sensitivity to phase-period shifts at $L{=}H{=}96$. Shifts are measured in time steps relative to the default (sampling-derived) period. "-4" indicates that $\mathcal{P}$ is set 4 steps below the default (ETTh1/ETTh2: 20 instead of 24; Weather: 140 instead of 144). "$+4$" indicates 4 steps above (ETTh1/ETTh2: 28; Weather: 148).

	ETTh1		ETTh2		Weather	
$\mathcal{P}$ shift	MSE	MAE	MSE	MAE	MSE	MAE
-4	0.365	0.382	0.275	0.326	0.153	0.195
-2	0.365	0.382	0.273	0.325	0.152	0.193
default	**0.365**	**0.383**	**0.271**	**0.324**	**0.145**	**0.186**
$+2$	0.366	0.383	0.275	0.326	0.153	0.193
$+4$	0.366	0.382	0.274	0.326	0.153	0.194

2. **ETTh2 (moderate daily periodicity).** A clear U-shaped trend appears with minimum at the default value $\mathcal{P}{=}24$. Shifting $\mathcal{P}$ by ± 2 steps increases MSE by 0.7–1.5%, while ± 4 steps increase MSE by 1.1–1.5%. This indicates that phase conditioning is active on ETTh2 and that the default period provides measurable benefit.
3. **Weather (strong 10-minute periodicity).** Sensitivity is substantially larger. Shifting $\mathcal{P}$ by ± 2 steps increases MSE by approximately 5%, and by ± 4 steps by approximately 5.2–5.7%. The default value $\mathcal{P}{=}144$ produces the lowest error across the entire sweep, confirming that the sampling-derived value is optimal on this dataset.

These results support four conclusions.

1. **Phase conditioning is adaptive.** The effect of $\mathcal{P}$ scales with the periodicity of the underlying signal: negligible where daily structure is weak (ETTh1), moderate where it is present (ETTh2), substantial where it dominates (Weather), and largest on the PEMS benchmarks (PEMS03 and PEMS04), where the daily traffic cycle is sharpest. This is consistent with the design of PhiMixer, where the phase embedding conditions the temporal and cross-channel transformations rather than directly contributing a prediction term, so its influence is naturally proportional to how much periodic information is available in the signal.
2. **Multiple periods.** When more than one period is active (e.g., daily and weekly), phase is defined with respect to the dominant (shortest non-trivial) cycle, while longer periods are handled independently by the cyclic-mean tables $\{P_j\}$. This separation ensures that phase

Table 17: Sensitivity to phase-period shifts on the PEMS benchmarks. The default $\mathcal{P}$=288 corresponds to one daily cycle at 5-minute sampling. Best per row in **bold**.

Dataset	H	$\mathcal{P}$=284	$\mathcal{P}$=286	$\mathcal{P}$=288	$\mathcal{P}$=290	$\mathcal{P}$=292
PEMS03	12	0.081	0.110	**0.064**	0.117	0.081
	24	0.123	0.222	**0.086**	0.231	0.124
	48	0.232	0.433	**0.128**	0.473	0.224
	96	0.364	1.094	**0.188**	0.893	0.333
PEMS04	12	0.122	0.132	**0.076**	0.163	0.128
	24	0.219	0.254	**0.097**	0.279	0.219
	48	0.419	0.466	**0.137**	0.590	0.420
	96	0.699	0.848	**0.196**	1.011	0.783

Table 18: Sensitivity to harmonic count K, PILCM rank r, and patch length p, evaluated at L=H=96. **Bold** marks the default configuration used throughout the main paper (K=4, r=8, p=16).

Parameter	**Value**	**ETTh1** (MSE)	**ETTh2** (MSE)	**Weather** (MSE)
K (harmonics)	2	0.366	0.269	0.146
	4	**0.365**	**0.271**	**0.145**
	6	0.364	0.269	0.143
	8	0.364	0.270	0.142
r (PILCM rank)	4	0.365	0.274	0.144
	8	**0.365**	**0.271**	**0.145**
	16	0.363	0.268	0.143
	32	0.363	0.269	0.142
p (patch length)	8	0.367	0.273	0.144
	16	**0.365**	**0.271**	**0.145**
	32	0.367	0.272	0.148

conditioning remains well-defined even when multiple periodicities coexist, and avoids phase-wrapping ambiguity that would arise from using a longer period for ψ.

3. **The sampling-derived $\mathcal{P}$ is optimal or within noise of the optimum.** On all five datasets, the default value produces the lowest or joint-lowest MSE. No shift of $\mathcal{P}$ improves on the sampling-derived value, confirming that $\lfloor 1440/\Delta t \rfloor$ is a reliable default.
4. **Sensitivity scales with periodicity strength.** On ETTh1/ETTh2, a ± 4-step shift changes MSE by at most 6%. On the PEMS benchmarks, where the daily cycle is sharp, a two-step shift increases MSE by up to $5.8\times$ at long horizons. This range of sensitivities reflects how much periodic structure each dataset contains.

Note on $\mathcal{P}$ selection. The phase period is computed from each dataset's sampling frequency as $\mathcal{P}$=$\lfloor 1440/\Delta t \rfloor$, where Δt is the sampling interval in minutes (Eq. 2). This makes $\mathcal{P}$ a fixed function of the physical sampling rate rather than a tuned hyperparameter. The present study shows that this sampling-derived value is optimal or near-optimal on all five evaluated datasets (ETTh1, ETTh2, Weather, PEMS03, and PEMS04). Extending PhiMixer to datasets without a clear dominant cycle (via DFT-based period detection or a learned period parameter) is left as future work.

Phase-period sensitivity on PEMS. We additionally examine phase-period sensitivity on the PEMS traffic benchmarks. Table 17 reports PEMS03 and PEMS04 results for $\mathcal{P}$ varied by ± 2 and ± 4 steps around its sampling-derived value of 288 (5 min sampling; $288 \times 5 = 1440$ min = one day). The default $\mathcal{P}$=288 is optimal at every horizon on both datasets, and performance degrades sharply for all perturbations. At H=96, a four-step error ($\mathcal{P} \in \{284, 292\}$) increases MSE by $1.9\times$ on PEMS03 and 3.4–3.8$\times$ on PEMS04, and a two-step error ($\mathcal{P} \in \{286, 290\}$) produces the largest errors in the sweep. This non-monotonic relationship between shift size and error – where the two-step shift produces larger degradation than the four-step shift at several horizons – indicates that the

Table 19: Sensitivity to harmonic count K, PILCM rank r, and patch length p on the PEMS benchmarks (L=96, MSE). **Bold** marks the default configuration used in the main paper (K=4, r=8, p=16).

		PEMS03				PEMS04			
Parameter	**Value**	H=12	H=24	H=48	H=96	H=12	H=24	H=48	H=96
K	2	0.066	0.089	0.133	0.196	0.079	0.101	0.146	0.212
	4	**0.064**	**0.086**	**0.128**	**0.188**	**0.076**	**0.097**	**0.137**	**0.197**
	6	0.065	0.089	0.134	0.198	0.080	0.101	0.147	0.214
	8	0.065	0.089	0.132	0.196	0.081	0.102	0.146	0.213
r	4	0.067	0.091	0.134	0.195	0.080	0.103	0.146	0.211
	8	**0.064**	**0.086**	**0.128**	**0.188**	**0.076**	**0.097**	**0.137**	**0.197**
	16	0.066	0.090	0.135	0.197	0.081	0.103	0.146	0.213
	32	0.065	0.089	0.134	0.198	0.080	0.102	0.145	0.211
p	8	0.066	0.090	0.133	0.196	0.081	0.102	0.147	0.213
	16	**0.064**	**0.086**	**0.128**	**0.188**	**0.076**	**0.097**	**0.137**	**0.197**
	32	0.067	0.089	0.134	0.196	0.081	0.101	0.148	0.214

Table 20: Architecture vs. composite loss at L=H=96. **full**: full PhiMixer. **stripped**: modules disabled. **full loss**: balanced MSE + DB + FreDF + CCC. **MSE**: MSE-only. Best per row in **bold**.

	Full model		Stripped model	
Dataset	full loss	MSE	full loss	MSE
ETTh1	**0.365**	0.371	0.376	0.382
ETTh2	**0.271**	0.275	0.280	0.283
Weather	**0.145**	0.149	0.156	0.161

sensitivity surface is not simply increasing in $|\Delta\mathcal{P}|$. We leave a full characterization of this behavior to future work.

Sensitivity to $\mathcal{P}$ thus scales with the strength of periodicity in the underlying signal: ETTh1 and ETTh2 exhibit little dependence on $\mathcal{P}$, while PEMS03 and PEMS04, with sharp daily traffic cycles, are highly sensitive to it. This is consistent with the design of PhiMixer, where phase conditioning adapts the model's transformations to the underlying rhythmic state, so its influence is naturally proportional to how much periodic structure the data exhibits.

Sensitivity to K, r, and p. We additionally examined sensitivity to the harmonic count K, PILCM rank r, and patch length p across all five evaluated datasets (Tables 18 and 19). Across all tested values of $K \in \{2, 4, 6, 8\}$, $r \in \{4, 8, 16, 32\}$, and $p \in \{8, 16, 32\}$, MSE differences were small: within 0.001–0.004 on ETTh1, within 0.001–0.006 on ETTh2 and Weather, and within 3–9% on PEMS03 and PEMS04, with the largest differences at the longest horizon (H=96). The defaults (K=4, r=8, p=16) were selected to balance accuracy against parameter count: increasing K or r yields only marginal MSE gains while adding parameters, and reducing p from 16 to 8 increases parameters substantially with no consistent accuracy benefit.

Architecture vs. composite loss. Table 20 isolates the contributions of the architecture and the composite loss on ETTh1, ETTh2, and Weather at L=H=96, using a 2×2 grid that crosses the full model against a stripped variant (FiLM, PILCM, and RRF disabled) and the full composite loss against MSE-only training. The full model with the full loss achieves the lowest MSE on all three datasets. Removing either factor increases MSE on all three datasets: switching the composite loss to plain MSE increases MSE by roughly 1–3%, and disabling FiLM, PILCM, and RRF increases MSE by 3–8%. The two effects are complementary—the full architecture with the full loss achieves the lowest MSE in all three cases.

Phase as feature vs. conditioning. A natural alternative to conditioning is to concatenate the phase embedding $\psi(t_0)$ to the patch representations as an ordinary input feature, letting the model

Table 21: Phase-mode ablation on ETTh1, ETTh2, and Weather at L=H=96 (MSE). **condition**: $\psi(t_0)$ conditions FiLM and the PILCM projections (the full model). **concat**: $\psi(t_0)$ is concatenated to the patch embeddings; FiLM is disabled and PILCM is phase-free. Best result per dataset in **bold**.

Dataset	condition (MSE)	concat (MSE)
ETTh1	**0.365**	0.371
ETTh2	**0.271**	0.276
Weather	**0.145**	0.153

Table 22: Phase-mode ablation on PEMS at L=96 (MSE). **condition**: full model, where $\psi(t_0)$ conditions FiLM and the PILCM projections. **concat:**: $\psi(t_0)$ concatenated to patch embeddings with FiLM disabled and PILCM phase-free. Best per row in **bold**.

H	PEMS03		PEMS04	
	condition (MSE)	concat (MSE)	condition (MSE)	concat (MSE)
12	**0.064**	0.066	**0.076**	0.081
24	**0.086**	0.092	**0.097**	0.103
48	**0.128**	0.135	**0.137**	0.148
96	**0.188**	0.197	**0.196**	0.208

learn any phase dependency through its usual parameters. Tables 21 and 22 compare this variant against the full model, showing that conditioning yields lower MSE than concatenation on both the main and PEMS benchmarks. On the PEMS benchmarks, the advantage grows monotonically with the forecast horizon, which is consistent with the strong daily periodicity of 5-minute traffic flow ($\mathcal{P}$=288): as the horizon lengthens, the forecast origin's position within the cycle becomes increasingly informative, so actively conditioning the model's transformations on phase provides more benefit than supplying it as a static input feature.

E Impact of Lookback Window Size

To assess the sensitivity of PhiMixer to the lookback window length, we evaluate the model with $L \in \{96, 192, 336, 720\}$ on the ETTh1, ETTm1, Weather, and ECL datasets. Fig. 2 visualizes the MSE across different input lengths for each dataset and prediction horizon $H \in \{96, 192, 336, 720\}$. Longer lookback windows generally reduce error. To be consistent with prior works, all our experiments are conducted with $L = 96$.

F Additional Representation Analysis

F.1 prediction Visualization

Figures 3 and 4 visualize forecasting on ETTm1 and ETTm2 datasets under the input-96-predict-336 setting. On ETTm1 (Fig. 3), PhiMixer achieves MSE 0.370 and MAE 0.387 at $H = 336$, effectively capturing the multi-scale temporal patterns with stable forecasts that closely follow the ground truth. The model demonstrates improved performance in tracking both the overall trend and fine-grained fluctuations, particularly during periods of rapid change.

On ETTm2 (Fig. 4), PhiMixer achieves MSE 0.275 and MAE 0.318 at $H = 336$, delivering predictions that closely align with the observed seasonal patterns. The model captures the periodic structure of the data, with peaks and troughs at close temporal positions, while maintaining stable predictions over the extended 336-step horizon. Compared to other baselines, PhiMixer exhibits improved tracking of both low-frequency trends and high-frequency variations, indicating its capability to model the dynamics of time-series datasets.

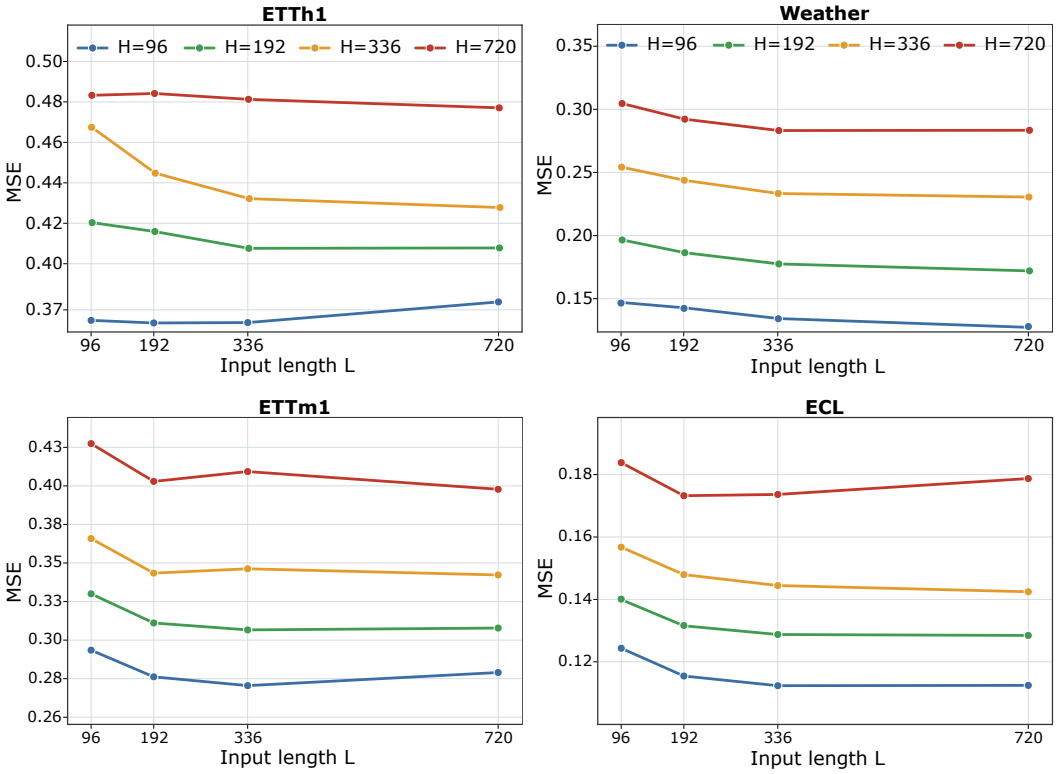


Figure 2: Impact of input length L on MSE across ETTh1, ETTm1, Weather and ECL dataset.

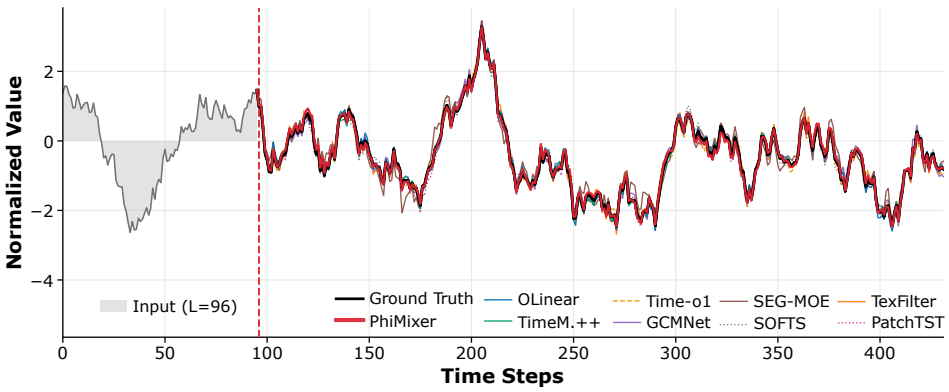


Figure 3: Visualization of ETTm1 dataset predictions by different models under the input-96-predict-336 setting.

F.2 Visualization of PhiMixer Signal Flow

Fig. 5 visualizes the information flow through PhiMixer on ETTm1 ($C = 7$). The raw input (a) exhibits diverse temporal characteristics across channels. RevIN normalization (b) removes instance-wise channel mean and scale, followed by learnable affine modulation. (c) CyclicMean subtraction estimates periodic components associated with the selected periods and produces the residual representation. Patch embedding (d) transforms the residual into tokenized representations. Phase-conditioned MixerBlocks (e) refine these representations through RRF and channel MLPs, with FiLM modulation via the phase embedding. PILCM (f) performs the proposed phase-informed low-rank channel coupling at cost $O(C \cdot d \cdot r + C \cdot N \cdot d)$. Pre-head features (g) are flattened before

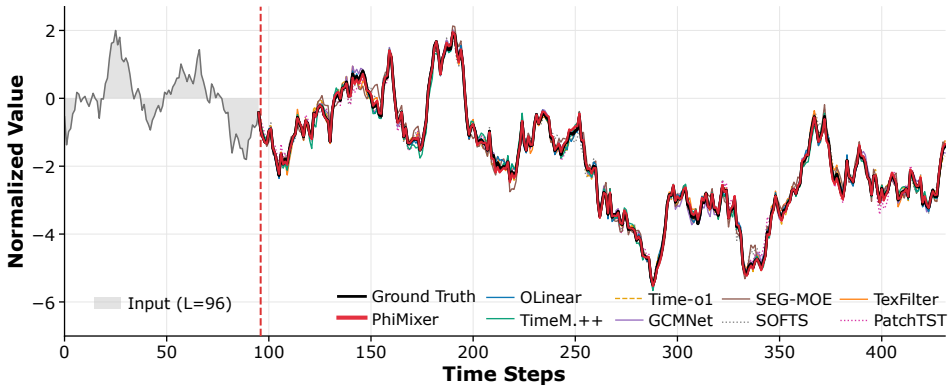


Figure 4: Visualization of ETTm2 dataset predictions by different models under the input-96-predict-336 setting.

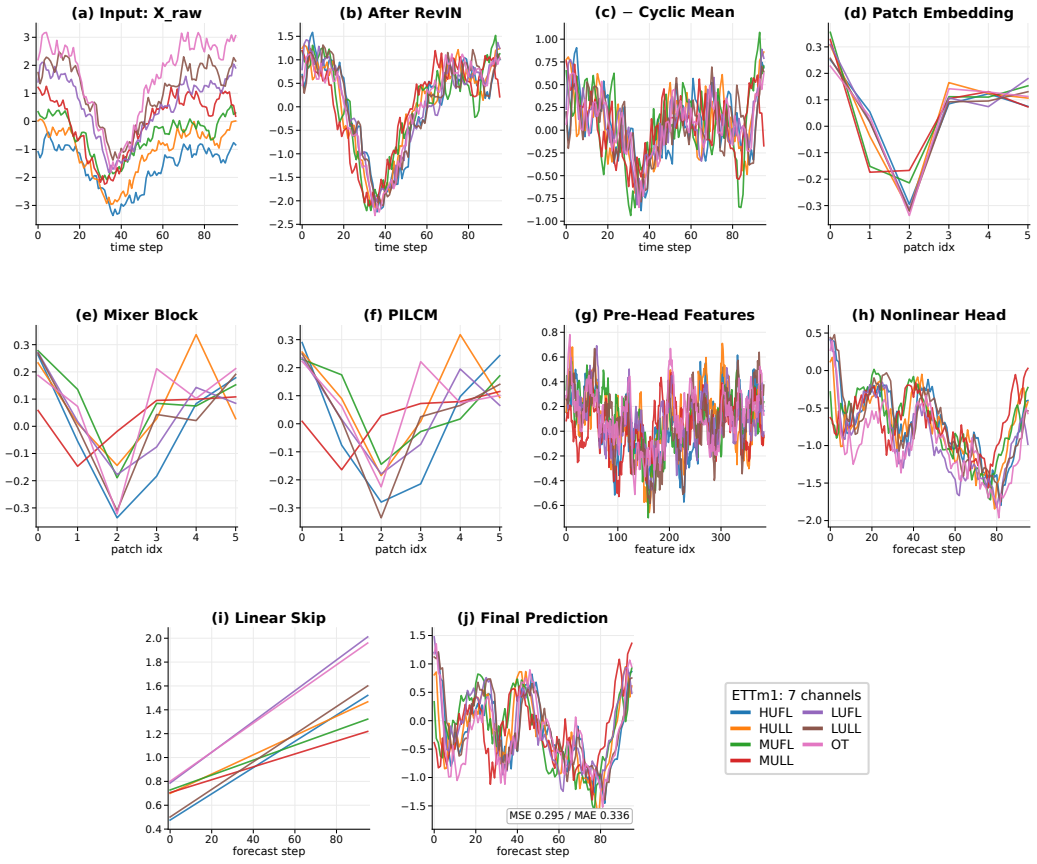


Figure 5: Key stage visualizations of PhiMixer signal flow on ETTm1 ($C = 7$).

the prediction head, and the nonlinear head (h) produces the nonlinear forecast. The linear skip path (i) provides a direct projection from the input to the horizon. Finally, gated fusion (j) combines the linear and nonlinear paths to produce the final forecast, achieving MSE 0.295 and MAE 0.336. The progression demonstrates how PhiMixer refines the input through phase-aware decomposition, rhythmic attention, and efficient channel mixing to produce the final forecast.

G Model Complexity and Edge-AI Deployability

G.1 Complexity Analysis

We denote the look-back window length as L, the number of variate/channel features as C, the forecast horizon as H, the model (hidden) dimension as d, the patch length as p (so the number of patches is $N = L/p$), the number of cyclic harmonics as K (giving a phase-feature dimension $P_{\text{dim}} = 2K + 1$), the low-rank bottleneck of the phase-informed channel mixer (PILCM) as r, and the number of stacked Mixer blocks as $\mathcal{B}$. PhiMixer adopts a dual-path architecture. The first is a linear skip connection that directly maps the raw look-back window to the target horizon. The second is a nonlinear patch-mixing path, which begins by removing both instance-level (RevIN) and periodic (CyclicMean) biases from the input. This path then segments each channel into $N = L/p$ non-overlapping patches, refines these patch representations through $\mathcal{B}$ phase-conditioned Mixer blocks, and projects the refined features to H forecasting steps. A learnable scalar gate $\sigma(\alpha)$ adaptively fuses the outputs of both paths prior to the final de-normalization step. We next provide a detailed breakdown of the computational cost associated with each module in the proposed architecture.

1. **Normalization (RevIN + CyclicMean).** Instance normalization is $\mathcal{O}(L \cdot C)$ forward and $\mathcal{O}(H \cdot C)$ on de-normalization; the periodic-mean lookup table costs $\mathcal{O}((L+H) \cdot C \cdot |\mathcal{P}|)$ for $|\mathcal{P}|$ registered periods (typically 1–2). Both are linear in L, H, and C.
2. **Patch embedding.** A shared $\text{Linear}(P, d)$ applied to every patch: $\mathcal{O}(C \cdot N \cdot P \cdot d) = \mathcal{O}(C \cdot L \cdot d)$.
3. **Token mixing (period-aware attention).** Attention is computed over the $N = L/p$ *patch tokens*, not raw time steps: $\mathcal{O}(C \cdot N^2 \cdot d)$ for the score/aggregation step and $\mathcal{O}(C \cdot N \cdot d^2)$ for the shared QKV/output projections. Because $N = L/p \ll L$, the quadratic term is effectively $\mathcal{O}(C \cdot L^2 \cdot d/p^2)$ — quadratic in L but suppressed by the squared patch length ($p^2 = 256$ at $p = 16$).
4. **Channel mixing (per-token MLP).** $\mathcal{O}(C \cdot N \cdot d^2)$.
5. **FiLM conditioning ($\times 2$ per block).** $\mathcal{O}(P_{\text{dim}} \cdot d)$ — independent of L, negligible relative to the above.
6. **PILCM.** Operating on the patch-averaged representation $\bar{h} \in \mathbb{R}^{B \times C \times d}$, PILCM's phase-conditioned low-rank projection costs $\mathcal{O}(C \cdot d \cdot r)$, with an additional $\mathcal{O}(C \cdot N \cdot d)$ for broadcasting the result back across the N patches — together $\mathcal{O}(C \cdot d \cdot r + C \cdot N \cdot d)$, linear in C for fixed d, r, avoiding the $\mathcal{O}(C^2)$ cost of pairwise cross-channel modeling.
7. **Prediction head + linear skip path.** $\text{Linear}(N \cdot d, H)$ costs $\mathcal{O}(C \cdot N \cdot d \cdot H)$; the parallel linear skip $\text{Linear}(L, H)$ costs $\mathcal{O}(C \cdot L \cdot H)$.

With the default configuration (one MixerBlock, RRF on, PILCM on), the total forward-pass complexity is

$$\mathcal{O}\Big(\underbrace{CLd}_{\text{embed}} + \underbrace{CN^2d}_{\text{attn.scores}} + \underbrace{CNd^2}_{\text{attn.proj.+ channel MLP}} + \underbrace{Cd^2}_{\text{STARPool}} + \underbrace{CNdH}_{\text{head}} + \underbrace{CLH}_{\text{skip path}} + \underbrace{Cdr + CNd}_{\text{PILCM}} \Big), \quad N = L/p. \tag{28}$$

When RRF is active, the temporal attention component scales as $O(C \cdot N^2 \cdot d)$ with $N = L/p$. Disabling RRF in favor of the fallback MLP token mixer does not remove this quadratic term: the fallback applies a linear layer $\text{Linear}(N, 2N)$ along the patch dimension, which involves an $N \times 2N$ weight matrix, resulting in $O(N^2 \cdot d)$ multiply-adds per batch-channel. Both RRF and the fallback MLP therefore incur $O(C \cdot N^2 \cdot d)$ cost. This quadratic term is suppressed by the patch length: with $p = 16$, $N = 6$ for $L = 96$, representing a $256\times$ reduction compared to full-resolution $O(L^2)$ attention. The $O(N^2)$ term is intrinsic to mixing across N patch tokens regardless of the specific mixer chosen.

Clarification: The $\mathcal{O}(N^2)$ term is quadratic in the number of patches $N = L/p$, not in the raw sequence length L — with $p = 16$, this is a $256\times$ reduction ($N = 6$ for $L = 96$ versus $L = 96$). Every term in Eq. equation 28 is linear in C (no $\mathcal{O}(C^2)$), where C is the number of channels (ranging

from 7 to 883), $N = L/p$ is the number of patches, $d = 64$ is the hidden dimension, H is the forecast horizon, and $r = 8$ is the PILCM rank.

Parameter count. Because the patch embedding, Mixer blocks, PILCM, and prediction heads are *weight-shared across channels* (applied per-token/per-channel via broadcasting), only the RevIN affine parameters ($2C$) and the CyclicMean lookup tables ($C \cdot \sum_{p\in\mathcal{P}} p$) scale with the number of channels C. Every other module's parameter count is *independent of* C, making PhiMixer's core footprint invariant to the number of channels. This is a deliberate edge-AI design choice: moving from a 7-channel dataset (ETTh1) to a 358-sensor dataset (PEMS03) adds only a few hundred extra scalars beyond the core footprint.

Collecting all terms for the default single-block configuration ($d = 64$, $p = 16$, $K = 4 \Rightarrow P_{\text{dim}} = 9$, $r = 8$, with RRF, FiLM, and PILCM enabled, and STARPool all enabled, we obtain the exact, code-verified expression:

$$\#\text{params}(L, H) \;=\; \underbrace{79{,}635}_{L,\,H\text{-independent core}} \;+\; 5\,L\,H \;+\; 2H, \tag{29}$$

for $C = 7$, $p = 16$ (so $N = L/16$), with a single MixerBlock. The 79,635 core parameters are independent of both the look-back length L and the forecast horizon H, and include the patch embedding, MixerBlocks, STARPool, PILCM, FiLM modulation networks, LayerNorm layers, and all associated biases and gates. The $5LH$ term arises from the prediction head, which has $N \cdot d \cdot H = (L/16) \cdot 64 \cdot H = 4LH$ parameters, and the linear skip path, which contributes $L \cdot H$ parameters. The $2H$ term accounts for the biases of the prediction head (H) and the skip path (H).At the standard ETTh1 configuration ($L = 96$, $H = 96$, $C = 7$), Eq. equation 29 evaluates to:

$$79{,}635 + 5 \cdot 96 \cdot 96 + 2 \cdot 96 = 79{,}635 + 46{,}080 + 192 = 125{,}907, \tag{30}$$

yielding approximately $\mathbf{126K}$ parameters (≈ 0.48 MB in FP32). This matches a direct enumeration of every Linear, $\mathrm{LayerNorm}$, and gate tensor in the implementation. Across the standard horizon sweep $H \in \{96, 192, 336, 720\}$ at fixed $L = 96$, the model scales *linearly* with H, yielding $\mathbf{126K}$, $\mathbf{172K}$, $\mathbf{242K}$, and $\mathbf{427K}$ parameters respectively — a slope of only 482 parameters per unit horizon. This linear scaling contrasts sharply with models using full-horizon projection matrices, which incur quadratic growth $\mathcal{O}(H^2)$.

For $C > 7$, additional parameters are added only for RevIN ($2C$ parameters for the affine scale and shift) and CyclicMean ($C \cdot \sum_{p\in\mathcal{P}} p$ parameters for the periodic lookup tables, where $\mathcal{P}$ is the set of periods). For example, on PEMS03 with $C = 358$ and $\mathcal{P} = \{288\}$, these add approximately $2 \cdot 358 + 358 \cdot 288 \approx 103{,}000$ parameters — still modest compared to the core footprint and linear in C. Importantly, the $\mathcal{O}(C^2)$ scaling of dense channel attention is entirely avoided, as the core 79,635 parameters are independent of C and the C-dependent terms grow only linearly with the number of channels (Table 24).

Assumptions. Eq. equation 29 assumes: (i) $C = 7$ (the default ETTh1 configuration); (ii) patch length $p = 16$; (iii) hidden dimension $d = 64$; (iv) $K = 4$ harmonics ($P_{\text{dim}} = 9$); (v) bottleneck rank $r = 8$; (vi) STARPool enabled; (vii) single MixerBlock ($\mathcal{B} = 1$); and (viii) RRF, FiLM, and PILCM all enabled. For different configurations, the core constant 79,635 changes, but the $5LH + 2H$ dependence on L and H remains identical.

G.2 Comparative Analysis and Edge-AI Deployability

Table 23 orders baselines by parameter count relative to PhiMixer ($\sim$126K at $L{=}H{=}96$). DLinear, CycleNet, and ModernTCN sit at 0.40–0.79$\times$ our size, but none combines an explicit periodic-phase inductive bias (CyclicMean + FiLM conditioning) with a low-rank cross-channel mixer (PILCM) the way PhiMixer does. PatchTST and SOFTS are 2.38$\times$ larger; Time-o1 is 2.30$\times$ larger *and* incurs quadratic horizon scaling $\mathcal{O}(H^2)$ from its projection-matrix pre-computation, so its relative overhead grows further as H increases while PhiMixer's grows only linearly (Eq. equation 29). TimesNet and FilterNet/TexFilter reach 2.78$\times$ via FFT-based $\mathcal{O}(CL \log L)$ period discovery; iTransformer

Table 23: Complexity and parameter comparison between PhiMixer and representative SOTA forecasters, at the common ETTh1 configuration (C=7, L=96, H=96). Ratio is relative to PhiMixer (∼126K). Red: more parameters; Blue: fewer parameters.

Model	Complexity	H Scaling	Params	Ratio
DLinear	$\mathcal{O}(CL)$	Independent	∼50K	↓ 0.40×
CycleNet	$\mathcal{O}(L \cdot C)$	$\mathcal{O}(H)$	∼65K	↓ 0.52×
ModernTCN	$\mathcal{O}(L \cdot C)$	$\mathcal{O}(H)$	∼100K	↓ 0.79×
PatchTST	$\mathcal{O}(CL^2 + CH)$	$\mathcal{O}(H)$	∼300K	↑ 2.38×
SOFTS	$\mathcal{O}(CL + CH)$	$\mathcal{O}(H)$	∼300K	↑ 2.38×
Time-o1	$\mathcal{O}(m \cdot H^2)$	$\mathcal{O}(H^2)$	∼290K	↑ 2.30×
TimesNet	$\mathcal{O}(CL \log L + CH)$	$\mathcal{O}(H)$	∼350K	↑ 2.78×
TexFilter	$\mathcal{O}(CL \log L + CH)$	$\mathcal{O}(H)$	∼350K	↑ 2.78×
iTransformer	$\mathcal{O}(C^2 + CL + CH)$	$\mathcal{O}(H)$	∼400K	↑ 3.17×
SEG-MOE	$\mathcal{O}(CL + CH + C^2)$	$\mathcal{O}(H)$	∼500K	↑ 3.97×
TimeMixer++	$\mathcal{O}(M \cdot C \cdot K \cdot (Lf + pf))$	$\mathcal{O}(H)$	∼647K	↑ 5.13×
OLinear	$\mathcal{O}(T \cdot C + C \cdot d)$	$\mathcal{O}(H)$	∼1.3M	↑ 10.32×
PhiMixer	$\mathcal{O}\big(CLd + CN^2d + CNd(d + H + 1) + Cd^2 + Cdr + CLH\big)$	$\mathcal{O}(H)$	∼126K	1.00×

and SEG-MOE reach 3.17–3.97× due to dense $\mathcal{O}(C^2)$ cross-channel attention, which makes their cost grow superlinearly with the number of channels — a particular liability for high-dimensional multivariate datasets. TimeMixer++'s multi-resolution imaging pushes it to 5.13×, and OLinear's large hidden width (D=512) and eigendecomposition-based orthogonal transform push it to 10.32× our footprint. Because PhiMixer's Mixer blocks, PILCM, and heads are channel-shared (Eq. equation 29 has no C-dependence beyond RevIN/CyclicMean), this gap widens further, not narrower, as C grows — the opposite trend to $\mathcal{O}(C^2)$ attention-based models.

Edge-AI Deployability. At the standard configuration (C=7, L=96, H=96, d=64, single block, STARPool enabled), PhiMixer requires exactly 125,907 parameters, i.e. a model size of ≈ 0.48 MB (parameter storage in FP32), well within the storage budget of resource-constrained edge and embedded systems. Using `torch.profiler` with batch size 1, we measure inference latency of 1.37 ms on a CPU and 1.91 ms on an NVIDIA A100-SXM4-40GB, with 9.64 MB peak inference memory and 18.38 MB training memory. Excluding negligible elementwise normalization ops, the forward pass performs ≈ 1.82 million multiply–accumulate operations (≈ 3.64 MFLOPs) per sample, dominated by the patch-token attention/channel-mixer stack (≈ 75% of total MACs) and the prediction head/skip path (≈ 15%). Parameter count (Eq. equation 29) scales linearly with the forecast horizon H. Computational cost (Eq. equation 28) scales linearly with C and H for fixed L, p, d, r, with the temporal token-mixing term scaling quadratically in the number of patches $N = L/p$. No SVD, pre-computation, or offline calibration is required, so PhiMixer can be applied across heterogeneous sensor counts without architectural changes. The compact footprint (0.48 MB) and low inference latency (1.37 ms on CPU, 1.91 ms on GPU) suggest that PhiMixer could be considered for latency- and resource-constrained forecasting settings. A detailed per-dataset inference latency profile is provided in Table 24.

In summary, PhiMixer achieves linear channel scaling $\mathcal{O}(C)$ for fixed L, p, d, and r, with no quadratic $\mathcal{O}(C^2)$ channel-interaction term anywhere in the inference architecture. This is the key distinction from dense channel-attention models such as iTransformer and SEG-MOE, which scale $\mathcal{O}(C^2)$. Its temporal complexity is controlled by patching ($N = L/p \ll L$), with the $O(N^2)$ attention/token-mixing cost present regardless of which token mixer is used, but small in absolute terms because $N \ll L$. Thus, while for fixed L, p, d, and r, PhiMixer's channel-dependent terms scale linearly with C, its temporal attention cost is quadratic in the number of patches $N = L/p$—which is substantially smaller than quadratic in the raw sequence length L due to the patch-based tokenization ($N \ll L$). This patch-suppressed quadratic cost is a deliberate design choice that balances expressiveness and efficiency for long-sequence forecasting. The training objective includes an inter-channel correlation-consistency term whose computation depends on the channel correlation matrix; this term contributes only to training-time cost and is not part of the inference-time complexity analyzed above.

Parameter Count Breakdown. Table 25 provides a module-by-module breakdown of PhiMixer's parameter count at the standard ETTh1 configuration (C=7, L=96, H=96, d=64, one MixerBlock, STARPool enabled). The core parameters of the network (patch embedding, RRF attention, channel

Table 24: Inference cost of PhiMixer at L=96, H=96 across 12 datasets. CPU latency is measured on an Intel Core i7; GPU latency on an NVIDIA A100-SXM4-40GB.

Dataset	C	Params	Size (MB)	CPU (ms)	GPU (ms)
ETTh1	7	125,907	0.48	1.37	1.91
ETTh2	7	125,907	0.48	1.33	2.09
ETTm1	7	129,939	0.50	1.30	1.87
ETTm2	7	129,939	0.50	1.29	1.87
Exchange	8	124,621	0.48	1.29	1.85
Weather	21	127,615	0.49	1.57	2.17
PEMS08	170	173,849	0.66	3.14	1.97
PEMS04	307	213,579	0.81	3.69	2.01
ECL	321	186,823	0.71	3.96	2.04
PEMS03	358	228,369	0.87	4.20	2.01
Traffic	862	291,777	1.11	5.69	2.34
PEMS07	883	380,619	1.45	5.47	2.52

Table 25: Parameter count breakdown for ETTh1 (C=7, L=96, H=96, d=64).

Module	Parameters
RevIN	14
CyclicMean (24, 168)	1,344
Patch Embedding	1,088
LayerNorm ($\times$2)	256
RRF (qkv + proj + gates)	16,642
Channel Mixer (MLP)	16,576
FiLM (2 modules)	17,920
PILCM (W_down, W_up, gate)	9,217
STARPool (core + fuse + gate)	16,577
Nonlinear Head	36,960
Linear Skip	9,312
Gating parameter α	1
Total	**125,907**

mixer, FiLM modulation, PILCM, STARPool, and prediction heads) are shared across channels, so only the RevIN affine parameters ($2C$) and the CyclicMean lookup tables ($C \cdot \sum_{p\in\mathcal{P}} p$) scale with the number of channels.

G.3 Inference Latency Across Datasets

Table 24 reports single-sample inference latency of PhiMixer across 12 datasets at the common configuration (L=96, H=96, batch size 1). On the NVIDIA A100-SXM4-40GB, latency remains within 1.91–2.52 ms as the number of channels increases from C=7 (ETTh1) to C=883 (PEMS07), a 126$\times$ increase in input width associated with only a 1.32$\times$ increase in inference time. This corresponds to an effective throughput of approximately 397–541 samples/s. On the tested commodity CPU, latency varies more visibly with the number of channels, ranging from 1.29 ms (ETTm2 and Exchange) to 5.69 ms (Traffic), with an average of 2.86 ms across all datasets. These measurements demonstrate low single-sample inference latency for PhiMixer under the tested CPU and GPU configurations.

H Scalability and Deployment Considerations

We next examine representative time series models from the standpoint of computational efficiency, focusing on temporal modeling mechanisms, scalability, and the implications of their architectural choices for inference cost.

Models such as TimesNet (Wu et al., 2023), Autoformer (Wu et al., 2022), ETSFormer (Woo et al., 2022), and FEDformer (Zhou et al., 2022) explicitly incorporate periodic, seasonal, or frequency-

aware mechanisms for long-term forecasting. However, explicit periodic or seasonal inductive biases do not by themselves guarantee adaptation to aperiodic or event-driven dynamics such as sudden spikes or abrupt changes. Other efficient architectures, such as PatchTST (Nie et al., 2023), LightTS (Zhang et al., 2022), and DLinear (Zeng et al., 2022), improve efficiency through patching, lightweight temporal processing, or linear decomposition rather than explicit phase conditioning. Convolutional architectures such as ModernTCN (donghao & wang xue, 2024) efficiently capture multi-scale local temporal patterns, although convolution-based processing generally requires architectural mechanisms such as large kernels, dilation, or hierarchical scales to capture long-range dependencies. Graph-based methods such as GTS (Shang et al., 2021) and FourierGNN (Yi et al., 2023) explicitly model relationships among variables, but their effectiveness depends on how inter-variable structure is constructed and updated. Efficient attention variants like Informer (Zhou et al., 2021), FEDformer (Zhou et al., 2022), and Autoformer (Wu et al., 2022) employ sparse or frequency-domain attention mechanisms for long-range modeling. Despite these efficiency-oriented designs, attention-based and sequence-level architectures can still incur substantial memory and computational costs as input length grows, depending on their attention and representation mechanisms.

Recent MLP-based architectures such as TSMixer (Chen et al., 2023) and TTM (Ekambaram et al., 2024) achieve computational efficiency through temporal and channel mixing, demonstrating competitive performance at lower computational costs than attention-based models. However, their mixing operations are largely determined by the learned architecture rather than explicitly conditioned on the phase of the forecast origin, limiting their ability to adapt to varying temporal contexts. TimeMixer++ (Wang et al., 2025b) extends this line of work with multi-resolution imaging, while OLinear (Yue et al., 2025) achieves efficiency through orthogonal-domain transformations and explicit cross-series dependency modeling. Neither approach, however, explicitly conditions its principal transformations on forecast-origin phase.

A separate line of work leverages large language models for time series forecasting, with examples including TIME-LLM (Jin et al., 2024) and LLM4TS (Chang et al., 2025). These approaches exploit the representational capabilities of pretrained language models, but their substantially larger computational footprints can pose challenges for latency- and memory-constrained deployment. iTransformer (Liu et al., 2024) improves multivariate dependency modeling by treating variates as tokens and applying attention across channels. This design makes cross-variable interaction explicit, but its attention cost grows with the number of variates, which can become substantial for high-dimensional multivariate data.

These limitations highlight the need for architectures that deliver better forecasting performance without sacrificing efficiency. PhiMixer addresses this gap by combining phase-conditioned processing with a hybrid CI+CD design, maintaining low computational cost through patching ($N = L/p$) and phase-informed low-rank channel mixing (PILCM) at $O(Cdr)$, while adapting temporal and cross-channel transformations to the forecast-origin phase.

We provide implementation code, dataset preprocessing scripts, and evaluation scripts in the anonymous repository available at: `https://anonymous.4open.science/r/Phimixer-51E7`. The architecture, training procedure, dataset splits, optimization settings, and evaluation metrics are described in Sections B.5, and B.5.2

I Limitations, Failure Cases, and Future Work

Although PhiMixer achieves the best MSE on 6/12 and the best MAE on 8/12 of the benchmark datasets, it exhibits relative weaknesses on ETTh1, ETTh2, Traffic, and PEMS04, where certain competing methods achieve lower errors.

On ETTh1, PhiMixer attains an average MSE/MAE of 0.433/0.427, whereas TimeMixer++ achieves 0.419/0.432 and OLinear achieves 0.424/0.424. At $H = 96$, PhiMixer records 0.365/0.383, lagging behind OLinear (0.360/0.382) and TimeMixer++ (0.361/0.403). On ETTh2, PhiMixer yields 0.356/0.384, while TimeMixer++ attains 0.339/0.380 and OLinear achieves 0.367/0.389. The weaker performance on these datasets suggests that the cyclic-mean decomposition may not adequately capture the seasonal characteristics of ETT data, where hourly-sampled transformer temperature readings exhibit relatively subtle periodic structure.

On Traffic, which consists of 862 sensors with significant spatial interdependencies, PhiMixer achieves 0.467/0.293, while SOFTS (0.409/0.267), Time-o1 (0.419/0.280), and TimeMixer++ (0.416/0.264) obtain better results. The complex inter-sensor relationships in traffic data may demand more expressive cross-channel modeling than the mean-pooling-based coupling provided by STARPool and PILCM. Nevertheless, PhiMixer remains competitive with PatchTST, indicating that its hybrid CI+CD design remains competitive on spatially correlated datasets.

On PEMS04, PhiMixer achieves 0.127/0.242, trailing MSTN-Transformer (0.115/0.223) and MSTN-BiLSTM (0.125/0.234). At $H = 96$, PhiMixer records 0.196/0.317, while MSTN-Transformer achieves 0.126/0.235. The performance gap on this dataset suggests that the phase-informed low-rank channel mixing may not fully capture the complex spatial dependencies in traffic sensor networks, where more expressive channel attention mechanisms provide an advantage.

These observations highlight a fundamental trade-off in PhiMixer's architecture: it prioritizes general-purpose phase-aware processing with linear complexity over dataset-specific inductive biases. By emphasizing broad applicability across diverse domains with a compact footprint ($\sim$126K parameters) and linear channel scaling $\mathcal{O}(C)$, PhiMixer delivers competitive performance on most benchmarks, though it may be surpassed on datasets where heavily optimized specialized architectures hold an advantage. This aligns with our design objective of creating a lightweight, efficient model suitable for a wide range of real-world scenarios. The modular architecture enables practitioners to tailor it to specific applications—by selecting appropriate cyclic periods, adjusting the PILCM rank, or modifying RRF parameters—all promising directions for future work.