WiSPformer: A Wiener-informed Signal-Preserving Spectral Transformer for Multivariate Time Series Forecasting

Anonymous authors
Paper under double-blind review

Abstract

Frequency-domain time-series forecasting has emerged as an effective alternative to time-domain modelling for multivariate time-series forecasting, because spectral representations compactly capture global trends and periodic patterns. However, the difficulty of frequency-based forecasting is the global spectral shrinkage which suppresses not only nuisance noise but also predictive local transients. In this paper, we propose *WiSPformer*, a Wiener-informed Signal-Preserving Transformer that combines a learnable Wiener Fourier filter, an adaptive signal-preserving mechanism, and a dual-channel complex spectral Transformer. Our learnable Wiener filter infers a frequency-wise spectral gain from the estimated signal and noise spectra. To overcome over-smoothing while retaining the benefits of spectral denoising, our signal-preserving method imposes a learnable protection on attenuation for channels with informative local dynamics. Lastly, we incorporate a dual-branch complex spectral Transformer to leverage synergistic dependencies between channels and frequency components. Extensive experiments on nine benchmark datasets show that WiSPformer consistently outperforms Wiener-only spectral filtering and achieves competitive performance compared with strong state-of-the-art baselines.

1 Introduction

Multivariate time-series forecasting is a fundamental problem in applications, such as climate prediction, energy management, industrial monitoring, and critical infrastructure management (Lai et al., 2018; Lim & Zohren, 2021; Mehdiyev et al., 2017; Wu et al., 2023b). These tasks not only require models to characterize temporal information (Zeng et al., 2023), but also to effectively model the spatial dependencies among variables (Cao et al., 2020a). In recent years, deep learning has made significant progress, with powerful forecasting models based on CNN, GNNs, and Transformer architectures (Liu et al., 2024a; Wu et al., 2023a; Nie et al., 2023; Wang et al., 2024a; Chen et al., 2023; donghao & wang xue, 2024a). These models improve forecasting by learning temporal patterns, channel dependencies, or multi-scale representations from historical observations. Among these methods, frequency-based approaches are effective solutions in time-series analysis because they offer a more concise and expressive representation of trends and periodicities (Zhou et al., 2022a; Fei et al., 2025).

In Fourier analysis, time-series sequences are decomposed across frequency bins. It makes the frequency domain a natural space for denoising and forecasting (Yi et al., 2023c; 2024; Fei et al., 2025). However, direct spectral suppression is not always beneficial for prediction, because real-world time-series often contain non-stationary distributions, local changes, noisy observations, and instance-level variations (Kim et al., 2022; Liu et al., 2022; Fan et al., 2023; Liu et al., 2025; Fu et al., 2025). Since the Fourier transform is a global representation of the signal, local changes in the time domain are dispersed across multiple frequency bins. Therefore, suppressing frequency components may remove not only noise but also informative local events. This creates a conflict where denoising improves robustness, but over-smoothing can erase useful transient patterns (Mallat, 2008).

Wiener filtering (Vaseghi, 1996) provides a principled starting point to this problem by frequency-wise shrinkage according to signal-to-noise ratio (SNR) rather than fixed frequency ranges. Components with higher SNR are preserved, while those with lower SNR are attenuated. However, classic

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.

Wiener filtering still relies on global spectral statistics, and informative local transients may be over-suppressed when their global SNR is low (Oppenheim & Schafer, 2009).

We propose *WiSPformer*, a Wiener-informed Signal-Preserving Transformer for multivariate time-series forecasting. It introduces a learnable Wiener filter with an adaptive signal-preserving method that limits excessive attenuation for channels with informative local dynamics. Then it utilises a dual-branch complex spectral Transformer for cross-channel and cross-frequency relationship learning in the real and imaginary spectra, followed by a lightweight calibration layer before inverse Fourier reconstruction. Our approach retains the advantages of Wiener-style spectral denoising and preserves informative transient changes for prediction. Figure 1 illustrates the filtering behaviors on a synthetic pulse signal. The orange-box region shows that the Wiener filter (green line) suppresses the informative frequencies of the input X while decreasing its noise. The blue-box regions also indicate that the pulse amplitude is attenuated by the filter during forecasting. Our WiSPformer retains more of the pulse structure and stays closer to the clean signal (X Clean) because it preserves more spectral energy removed by the Wiener filter.

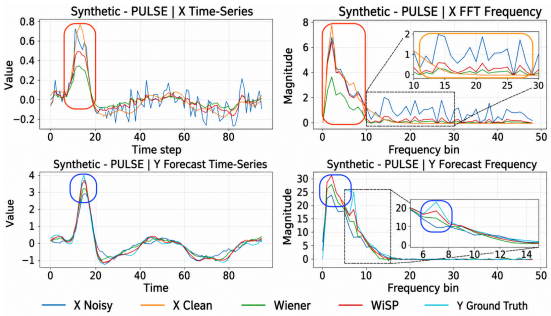


Figure 1: Signal-preserving keeps relevant local transients. The synthetic PULSE signal shows that WiSPformer retains more energy in the low- and mid-frequency range (WiSP), while suppressing high-frequency noise. The Wiener filter attenuates spectral components beyond the low- and mid-frequency bands during denoising. So WiSPformer better recovers the forecasting pulse signal (ground truth). More about the synthetic dataset in the Appendix H.

Our main contributions are summarized as follows:

- We propose a signal-preserving learnable Wiener spectral filter that integrates a sigmoidized Log-SNR Wiener filter with the signal-preserving mechanism. It enables the Wiener filter to suppress background noise while avoiding the over-smoothing of informative local transitions.
- We further develop WiSPformer, which combines the signal-preserving learnable Wiener filter and a dual-branch complex spectral Transformer with a complex-consistent calibration layer. It can effectively model both temporal dynamics and spatial dependencies.
- Extensive experiments show that preserving such local structures substantially mitigates the degradation caused by spectral denoising. Our proposed method consistently outperforms standalone Wiener-style filtering and achieves superior performance against 12 state-of-the-art baselines on 9 benchmark datasets.

2 PRELIMINARIES

Frequency filtering (Márquez & Zaman, 2013) provides a general way to manipulate signals in the spectral domain by selectively preserving or suppressing frequency components through amplitudes and phases. **Wiener filtering** (Wiener, 1964) performs linear frequency-domain denoising by estimating the underlying clean signal from noisy observations.

Lemma 1 (Classical Wiener Filter). *Let the observed signal be $x[t] = s[t] + n[t]$, where $s[t]$ is the true signal and $n[t]$ is additive noise. After applying Fast Fourier transform (FFT), we have $Z[k] = S[k] + N[k]$. $k \in \{0, \dots, F-1\}$ denotes the frequency-bin index. Consider the linear spectral estimator $\hat{S}[k] = H[k]Z[k]$, which minimizes the mean squared error*

$$\mathcal{L}_k(H) = \mathbb{E}\big[|H[k]Z[k] - S[k]|^2\big]. \tag{1}$$

If the true signal and the noise are zero-mean and independent, then the optimal Wiener frequency gain is

$$H^\star[k] = \frac{P_s[k]}{P_s[k] + P_n[k]} = \frac{\rho[k]}{1+\rho[k]} \quad \rho[k] = \frac{P_s[k]}{P_n[k]}, \tag{2}$$

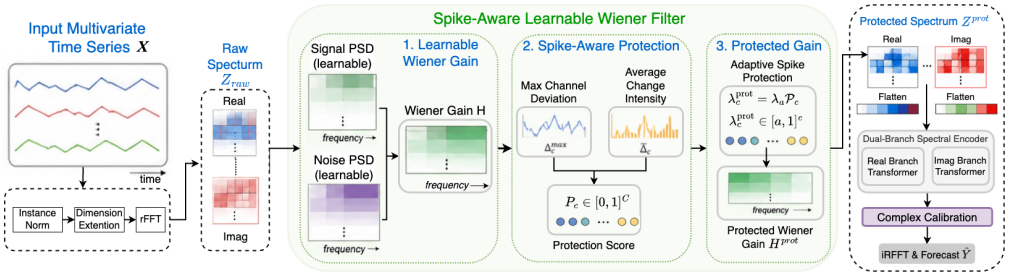


Figure 2: The overall structure of the WiSPformer.

where $P_s[k]$ and $P_n[k]$ are the signal and noise power spectral densities, respectively. $\rho[k]$ is the spectral signal-to-noise ratio (SNR).

Wiener filtering acts as a frequency-wise SNR-based selector: high-SNR components are mostly preserved with $H^\star(k) \approx 1$, while low-SNR components are suppressed with $H^\star(k) \approx 0$. See the Appendix D for the full derivation.

3 Methodology

Figure 2 illustrates the architecture of WiSPformer, which consists of three stages. 1. The input multivariate time-series is transformed into the Fourier domain and denoised by a signal-preserving learnable Wiener filter with adaptive transient protection. 2. The filtered spectra are decomposed into real and imaginary parts and processed by a dual-branch complex spectral Transformer to model cross-channel dependencies. 3. The two branches are calibrated, recombined, transformed back to the time domain via inverse Real Fast Fourier transform (iRFFT), and passed to a head for prediction.

3.1 Signal-Preserving Learnable Wiener Filter

The classic Wiener filter performs frequency-wise spectral denoising, but it can over-smooth non-stationary local transients. When their estimated SNR is a low Wiener gain $H(k)$, these transient patterns are suppressed together with noise. To preserve informative local variations, we introduce a signal-preserving learnable Wiener filter that lifts the Wiener gain for these channels with informative transient activity, reducing excessive shrinkage.

3.1.1 Our Sigmoidized Log-SNR Wiener Gain

Proposition 1 (Learnable Wiener gain and its log-SNR form)**.** *Following the classical Wiener filter, let $\rho_c \in \mathbb{R}^{F\times N}$ be learnable signal and noise power spectral density maps over the F frequency bins and N channels. For the channel c and the frequency bin k, we can define our learnable Wiener spectral gain as*

$$H_c(k) = \frac{\rho_c(k)}{1+\rho_c(k)}, \tag{3}$$

where $\rho_c(k)$ denotes the SNR at frequency bin k and channel c. Then it can be rewritten into a sigmoid transformation of the log-SNR.

$$H_c(k) = \sigma\big(\log \rho_c(k)\big), \quad 0 \le H(k) \le 1. \tag{4}$$

The log-SNR $log\rho_c(k)$ is represented as a set of learnable latent spectral parameters optimized through the mean square error between the forecasting signal and the true signal. As shown in Appendix E.2, the optimal latent log-SNR minimizes the reconstruction objective. In Appendix H.2, We further demonstrate that the learned log-SNR exhibits the strong corelation with the true log-SNR of the synthetic dataset.

3.1.2 SIGNAL-PRESERVING PROTECTION

Since the transient characteristics of time-series signals and the magnitude of changes between adjacent time steps can provide effective indicators of anomalous behavior (Blázquez-García et al., 2021), we customise them by using two complementary measures: the maximum deviation from the local mean and the average change intensity. The protection weight is then defined as a combination of these two measures, which reflects the filter's sensitivity to local patterns and transient changes.

Let $\mathbf{X} \in \mathbb{R}^{N \times T}$ denote a multivariate time-series sequence, where N is the number of channels and T is the number of time windows. We use $c \in \{1, \ldots, N\}$ and $t \in \{1, \ldots, T\}$ to index the channel and time windows, respectively. The entry $\mathbf{X}_{c,t}$ represents the observation of channel c at time window t. If $\mathbf{X}_{c,t}$ contains transient changes, then its channel c will show a larger **Maximum Channel Deviation** ($\Delta_{ct}^{\max}$) within the time window t:

$$\mu_c = \mathrm{mean}(\mathbf{X}_{c,\{1,\ldots,T\}}), \sigma_c = \mathrm{std}(\mathbf{X}_{c,\{1,\ldots,T\}}), \tag{5}$$

$$\Delta_c^{\max} = \frac{\max_t |\mathbf{X}_{c,t} - \mu_c|}{\sigma_c + \epsilon}, \tag{6}$$

where μ_c and σ_c denote the mean and standard deviation of the channel c over the input time windows ($\{1, \ldots, T\}$), respectively. ϵ is a small constant for numerical stability.

Average Change Intensity ($\bar{\Delta}_c$) characterizes the average magnitude of step-to-step changes over time on the channel c:

$$\bar{\Delta}_c = \frac{1}{T-1} \sum_{t=1}^{T-1} \frac{|\mathbf{X}_{c,t+1} - \mathbf{X}_{c,t}|}{\sigma_c + \epsilon}. \tag{7}$$

These two statistics characterise local dynamic patterns. $\Delta_c^{\max}$ is more sensitive to large-amplitude deviations or pulse-like peaks, whereas $\bar{\Delta}_c$ emphasizes local changes, jitter, and the sharpness of temporal transitions. For example, a **hill-shaped** signal has a large peak compared with its normal level. It exhibits high $\Delta_c^{\max}$ but low $\bar{\Delta}_c$. A **jagged** signal changes rapidly and irregularly step by step. It could show only moderate $\Delta_c^{\max}$ but high $\bar{\Delta}_c$.

Signal-Preserving Protection Score ($\mathcal{P}_c$) integrates these two measures into a unified channel protection score.

$$\mathcal{P}_c = \sigma([\alpha\, \Delta_c^{\max} + (1-\alpha)\, \bar{\Delta}_c] - \tau_c), \tag{8}$$

where $\alpha \in [0, 1]$ is a learnable mixing coefficient, σ is a sigmoid function. τ_c denotes a learned transient threshold for protection on channel c. It allows channels with weak transient activity to remain close to Wiener filtering, while stronger transient channels receive increased protection.

3.1.3 LEARNABLE CHANNEL-WISE ADAPTIVE PROTECTION

To alleviate the over-smoothing introduced by our learnable Wiener filter and ease the absolute protection mechanism in Eq. 8, we introduce a learnable channel-wise adaptive protection adjustment. Let $\lambda_c^{\mathrm{filter}} \in [0, 1]$ be a learnable base filtering level for channel c. We define the adaptive protection function as:

$$\lambda_c^{\mathrm{prot}} = \lambda_c^{\mathrm{filter}} + (1 - \lambda_c^{\mathrm{filter}})\, \mathcal{P}_c \tag{9}$$

The Signal-Preserving Wiener Gain on the channel c is defined as:

$$W_c(k) = \lambda_c^{\mathrm{prot}} + \left(1 - \lambda_c^{\mathrm{prot}}\right) H_c(k), \tag{10}$$

where $H_c(k)$ denotes the learnable Wiener Gain on channel c. Therefore, protection adjusts the overall preservation strength of transient-rich channels, while frequency-wise spectral weighting for denoising is estimated by the learnable Wiener filter.

When $\mathcal{P}_c = 0$, the protection coefficient reduces to its filtering baseline, i.e., $\lambda_c^{\mathrm{prot}} = \lambda_c^{\mathrm{filter}}$. In this case, the effective gain becomes

$$W_c(k) = \lambda_c^{\mathrm{filter}} + (1 - \lambda_c^{\mathrm{filter}}) H_c(k), \tag{11}$$

so the filter behaves like a scaled Wiener denoiser with its default attenuation capacity. When $\mathcal{P}_c = 1$, $\lambda_c^{\mathrm{prot}} = 1$, which preserves the channel from suppressing attenuation. For $0 < \mathcal{P}_c < 1$, $\lambda_c^{\mathrm{prot}}$

smoothly interpolates between $\lambda_c^{\text{filter}}$ and 1, thereby adaptively balancing spectral denoising and transient protection.

Importantly, in the Appendix F, we clarify that the protect gain does not replace the learned frequency-wise Wiener filter. Instead, it performs a channel-wise adjustment that relaxes Wiener attenuation while preserving its frequency-selective ordering. Consequently, informative spectral components continue to receive larger retained gains than noise-dominated components, allowing WiSPformer to preferentially recover predictive spectral structures while maintaining Wiener-style noise suppression.

3.2 Dual-branch Complex Spectral Transformer

Spectral-aware Wiener filtering enables frequency-wise selection and spectral denoising, but it does not explicitly model cross-channel dependencies. In multivariate time-series, different channels may interact dynamically, with relationships that vary over time or follow periodic patterns (Yue et al., 2025; Zhang et al., 2026). Prior frequency-domain forecasting models such as FreTS (Yi et al., 2023b) and FreEformer (Yue et al., 2025) have shown the effectiveness of learning over complex spectral representations with the frequency spectra with the real and imaginary parts processed independently. As Fig. 2 of the overall structure illustrates, we also adopt a dual-branch spectral Transformer, where the real and imaginary parts are learned by two distinct Transformer blocks before complexing for the final forecasting. Keeping the two parts separate during encoding helps preserve this complex structure, instead of prematurely mixing them into a single real-valued representation.

3.2.1 Attention Modelling of Cross-Channel Relationships

Let the input complex spectrum be represented as $Z^{\text{raw}} \in \mathbb{C}^{N\times F}$, and the complex spectrum after Wiener filtering be represented as $Z^{\text{flt}} \in \mathbb{C}^{N\times F}$. We apply a dimension extension block (Yi et al., 2023b) on Z^{raw} and concatenate the spectrum along the spectral dimension to $Z^{\text{cat}} \in \mathbb{C}^{N\times 2F\times d}$.

We then split Z^{cat} into its real and imaginary parts, flatten each channel-wise spectral vector, and feed them into two separate encoding branches with output dimension D:

$$E_{\Re} = \mathcal{E}_{\Re}(Z_{\Re}^{\text{flat}}), \qquad E_{\Im} = \mathcal{E}_{\Im}(Z_{\Im}^{\text{flat}}). \tag{12}$$

Here, $Z_{\Re}^{\text{flat}}$ and $Z_{\Im}^{\text{flat}}$ are the flattened real and imaginary parts of Z^{cat}. For each channel, the encoders $\mathcal{E}_{\Re}$ and $\mathcal{E}_{\Im}$ map an input spectral feature in $\mathbb{R}^{N\times 2dF}$ to an encoded representation in $\mathbb{R}^{N\times D}$, yielding $E_{\Re}$ and $E_{\Im}$, respectively.

The corresponding self-attention with d key/query dimension is:

$$Q = E_{\Re}W_Q, \qquad K = E_{\Re}W_K, \qquad V = E_{\Re}W_V, \tag{13}$$

$$\text{Attn}(E_{\Re}) = \text{Softmax}\left(\frac{QK^\top}{\sqrt{d}}\right)V. \tag{14}$$

The attention matrix is of size $N \times N$, which explicitly models relationships between channels. Moreover, since $Z_{\Re}^{\text{flat}}$ and $Z_{\Im}^{\text{flat}}$ are channel-based tokens that contain the full frequency bins, the linear projection and FFN within each token allow the model to recombine cross-frequency features.

3.2.2 Complex-Consistent Calibration Layer

Because the dual-branch transformer learns the real and imaginary spectra parts **separately**, their re-combined representation is not guaranteed to be the most suitable complex spectrum for inverse Real Fast Fourier Transform (iRFFT). As Eq. 15, iRFFT depends on the joint magnitude-phase geometry of each complex coefficient.

$$\hat{Z}[k] = \hat{Z}_{\Re}[k] + j\hat{Z}_{\Im}[k], \tag{15}$$

$$|\hat{Z}[k]| = \sqrt{\hat{Z}_{\Re}[k]^2 + \hat{Z}_{\Im}[k]^2}, \tag{16}$$

$$\phi(\hat{Z}[k]) = \text{atan2}\big(\hat{Z}_{\Im}[k], \hat{Z}_{\Re}[k]\big), \tag{17}$$

where $|\hat{Z}[k]|$ and $\phi(\hat{Z}[k])$ represents the magnitude and phase. To solve this problem, we implement a lightweight uniform mixing layer that projects them into a coherent complex space, as in Eq. 18.

$$Y_{\Re} = a \odot H_{\Re} - b \odot H_{\Im}, \qquad Y_{\Im} = a \odot H_{\Im} + b \odot H_{\Re}, \tag{18}$$

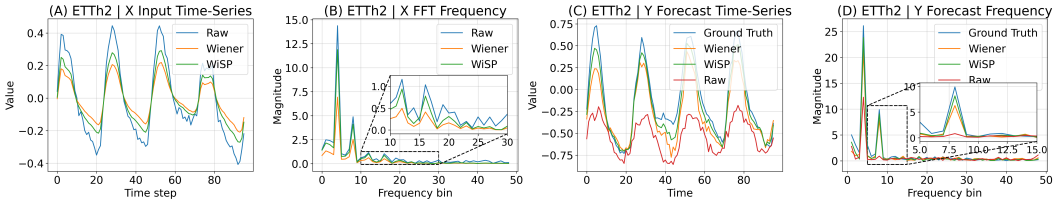


Figure 3: Signal-preserving retains informative local transients in WiSPformer. **Raw** is for original input signal, **Wiener** for the signal after Wiener filtering, and **WiSP** for the signal after WiSPformer.

where $H_{\Re}$ and $H_{\Im}$ are the real and imaginary representations from the two branches, and $\odot$ represents element-wise multiplication. Here, a and b are learnable parameters that denote the joint magnitude-phase correction operator in the complex domain.

4 Experiments

We evaluate the proposed WiSPformer on nine real-world time-series benchmarks and conduct detailed analyses of its frequency filter to validate the effectiveness of our framework. We report the **experimental setup**, **lookback length**, and **efficiency analysis** in the Appendix B, J and K.

Baselines. We extensively select **8** competitive baselines for comparison in Table 1 and extend a comprehensive comparison with a total of **12** baselines in Table 7. The spectral-based methods include Leddam (Yu et al., 2024), Amplifier (Fei et al., 2025), FilterNet (Yi et al., 2024), xPatch (Stitsyuk & Choi, 2025) and Alio (Park et al., 2025). The Transformer-based baselines include iTransformer (Liu et al., 2024a), Ada-MSHyper (Shang et al., 2024), CARD (Xue et al., 2024) and EMAFormer (Zhang et al., 2026). In addition, we consider three more spectral-based Transformer methods, including Fredformer (Piao et al., 2024), FreEformer (Yue et al., 2025), and FreDF (Wang et al., 2025).

4.1 Main Results

Table 1 presents the forecasting results of WiSPformer on 9 real-world datasets. Overall, WiSPformer achieves the best performance on 14 of 18 evaluation cases. xPatch (the second-best method) shows a competitive performance on ETTh1 and ETTh2. But its results are less consistent than WiSPformer on Weather, Traffic, Exchange, Solar and ECL. Comparing WiSPformer with xPatch, WiSPformer achieves average error reductions of 6.7% in MSE and 3.0% in MAE. Moreover, WiSPformer outperforms xPatch by reducing MSE and MAE by 18.1% and 7.4% on Exchange, 16.0% and 10.8% on Traffic, 12.8% and 6.4% on ECL, and 20.0% and 9.3% on Solar, respectively. Compared with spectral-based Transformer baselines, WiSPformer shows stronger overall performance and more consistent generalisation across datasets and outperforms FreDF, FreEformer, and Fredformer on most datasets.

4.2 Ablation Studies

We conduct ablation studies to quantify the contribution of each component in WiSPformer, including the learnable Wiener filter, the signal-preserving protection, and the dual-branch complex-consistent calibration Transformer.

Effect of Wiener filtering and Signal Preserving Protection. In Figure 8 and Appendix H, We create a synthetic dataset with three channels sharing a common seasonal backbone and distinct transient patterns: Step, Pulse, and Jagged signals. White Gaussian noise is added to each channel.

Since the transient components are part of the clean signal rather than noise, this dataset allows us to examine the learned Wiener log-SNR and the trade-off between denoising and signal preservation. Figure 1 reveals that the learned Wiener filter reduces noise while also over-attenuating transient-related spectral components. WiSPformer preserves more transient-related spectral energy, producing a forecast spectrum that more closely matches the ground truth and consequently a more accurate reconstruction of the pulse in the time domain. Table 3 shows that the WiSPformer consistently outperforms the learned Wiener filter and the raw noisy input, confirming the benefit of spectral

Models	WiSPformer [Ours]		EMAFormer [2026]		Amplifier [2025]		xPatch [2025]		FilterNet [2024]		FreDF [2025]		FreEformer [2025]		AdaMSHyp [2024]		Fredformer [2024]	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	**0.375**	**0.379**	0.381	**0.380**	0.381	0.394	**0.377**	0.384	0.384	0.398	0.392	0.399	0.379	0.381	0.378	0.391	0.384	0.395
ETTm2	**0.264**	**0.310**	0.270	**0.313**	0.276	0.323	**0.267**	**0.313**	0.276	0.322	0.278	0.319	0.272	0.313	0.274	0.324	0.279	0.324
ETTh1	**0.427**	**0.421**	0.432	0.425	0.430	0.428	0.428	**0.419**	0.440	0.432	0.437	0.435	0.433	0.431	**0.418**	0.426	0.435	0.426
ETTh2	**0.355**	**0.384**	0.374	0.396	0.359	0.391	**0.319**	**0.361**	0.378	0.404	0.371	0.396	0.372	0.393	0.371	0.385	0.365	0.393
Weather	**0.225**	**0.255**	0.240	0.262	0.243	0.271	**0.232**	0.261	0.245	0.272	0.254	0.274	0.239	**0.260**	0.233	0.266	0.246	0.272
Solar	**0.191**	**0.214**	**0.198**	0.224	0.275	0.298	0.239	0.236	0.263	0.288	0.225	0.252	0.217	**0.219**	0.225	0.253	0.226	0.262
Electricity	**0.156**	**0.247**	**0.158**	**0.247**	0.171	0.265	0.179	0.264	0.172	0.268	0.170	0.259	0.162	**0.251**	0.167	0.252	0.176	0.269
Traffic	**0.419**	**0.249**	0.431	0.282	0.482	0.315	0.499	0.279	0.463	0.310	**0.421**	0.279	0.435	**0.251**	0.428	0.271	0.433	0.291
Exchange	**0.307**	**0.378**	0.355	0.402	0.361	0.402	0.375	0.408	0.356	0.395	0.359	0.401	0.354	0.399	0.353	0.397	**0.333**	**0.391**
Count	**14**		1		0		**3**		0		0		0		1		0	

Table 1: Multivariate long-term time-series forecasting results compared with **8** methods using a unified lookback window $L = 96$ across all datasets. All results are averaged over four prediction lengths: $H \in \{96, 192, 336, 720\}$. The best results are highlighted in **RED**, and the second-best results are **BLUE**. Lower values indicate better performance. Our model achieves the best results in 11 cases. See Table 6 in the Appendix for the **full results**, including comparisons with **12** state-of-the-art methods.

Methods		FreTS	+ WiSP	T.KAN	+ WiSP	iTrans.	+ WiSP
ETTh1	MSE	0.470	**0.453**	0.437	**0.423**	0.454	**0.443**
	MAE	0.448	**0.430**	0.431	**0.426**	0.448	**0.439**
ETTh2	MSE	0.384	**0.369**	0.393	**0.376**	0.383	**0.377**
	MAE	0.429	**0.391**	0.411	**0.398**	0.407	**0.403**
ETTm1	MSE	0.406	**0.392**	0.383	**0.377**	0.407	**0.400**
	MAE	0.409	**0.387**	0.399	**0.397**	0.410	**0.405**
ETTm2	MSE	0.280	**0.277**	0.287	**0.275**	0.292	**0.291**
	MAE	0.328	**0.323**	0.333	**0.317**	0.335	**0.334**
Electricity	MSE	0.200	**0.183**	0.195	**0.184**	0.178	**0.171**
	MAE	0.288	**0.273**	0.275	**0.270**	0.270	**0.263**
Solar	MSE	0.285	**0.266**	0.253	**0.244**	0.253	**0.244**
	MAE	0.280	**0.266**	0.268	**0.255**	0.284	**0.278**
Weather	MSE	0.247	**0.244**	0.244	**0.232**	0.260	**0.253**
	MAE	0.294	**0.264**	0.273	**0.266**	0.282	**0.279**
Traffic	MSE	0.542	**0.507**	0.431	**0.423**	0.429	**0.426**
	MAE	0.298	**0.291**	0.286	**0.279**	0.284	**0.283**

Table 2: Average performance of WiSP on different forecasting backbones. Results are averaged over all prediction lengths. Full results are provided in the Appendix Table 7.

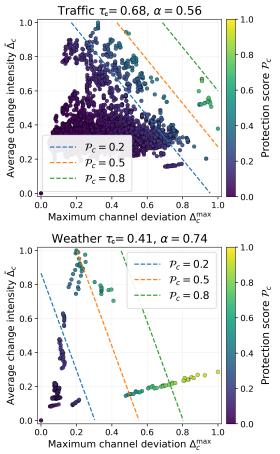


Figure 4: Protection score $\mathcal{P}_c$ versus $\Delta_c^{\max}$ and $\bar{\Delta}_c$ (left), and $\Delta W = \mathrm{Mean}(W_{\mathrm{wisp}} - W_{\mathrm{wiener}})$ (right), on Traffic and Weather.

protection and denoising. In Figure 9, the learned Wiener log-SNR shows a positive correlation with the ground-truth log-SNR and achieve a Pearson correlation of 0.803 and a Spearman rank correlation of 0.782, which exhibits the learned log-SNR can capture the ground-truth SNR. Figure 10 further demonstrates that the learned signal follows the true signal (X Clean).

In Figure 3[B], shows that the learned Wiener filter suppresses a range of spectral components. WiSPformer retains larger magnitudes in the zoomed region. It indicates that the protection mechanism prevents informative components from being overly attenuated. In Figure 3[C], WiSPformer makes the forecast follow the ground-truth more closely because it restores the spectrum removed by the learned Wiener filter in Figure 3[D]. Additional protection examples are provided in the Appendix I.

Figure 6 illustrates the relationship between the protection score $\mathcal{P}_c$ and the gain changes $\Delta W = \mathrm{Mean}(W_{\mathrm{wisp}} - W_{\mathrm{wiener}})$. ΔW increases as $\mathcal{P}_c$ becomes larger. It satisfies that the protection can increase the intended spectral gain.

Figure 4 shows how the protection $\mathcal{P}_c$ changes with the maximum channel deviation $\Delta_c^{\max}$ and the average change intensity $\bar{\Delta}_c$. Each point represents one channel c and is colored according to its protection score $\mathcal{P}_c$. The dashed lines denote protection score boundaries ($\mathcal{P}_c$ =0.2, 0.5, and 0.8). The boundary is linear because the protection score is computed from a weighted combination of $\Delta_c^{\max}$

Signal	WiSP		Wiener		Raw	
	MSE	MAE	MSE	MAE	MSE	MAE
Step	**0.736**	**0.509**	0.750	0.533	0.765	0.543
Jagged	**0.670**	**0.460**	0.675	0.461	0.677	0.464
Pulse	**0.065**	**0.196**	0.075	0.211	0.091	0.232

Table 3: Forecasting performance ($96 \rightarrow 96$) on the synthetic dataset. Lower is better.

Dataset	WiSPformer		Uncalibrated		Wiener		Raw		Share	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	**0.305**	**0.338**	0.309	0.343	0.317	0.340	0.319	0.342	0.315	0.344
ETTm2	**0.166**	**0.246**	0.168	**0.246**	0.168	0.249	0.170	0.249	0.169	0.248
ETTh1	**0.372**	**0.385**	0.375	0.387	0.376	0.388	0.382	0.390	0.384	0.394
ETTh2	**0.283**	**0.330**	0.284	**0.330**	0.284	**0.330**	0.289	0.332	0.298	0.338
Weather	**0.150**	**0.186**	0.151	0.187	0.153	0.189	0.152	0.188	0.153	0.189
Solar	**0.161**	**0.191**	0.166	0.195	0.169	0.197	0.180	0.225	0.172	0.207
Electricity	**0.132**	**0.222**	0.133	0.224	0.142	0.230	0.150	0.239	0.135	0.225
Traffic	**0.390**	**0.232**	0.391	0.234	0.397	0.239	0.401	0.238	0.395	0.238
Exchange	**0.079**	**0.199**	0.081	0.201	0.082	0.200	0.086	0.205	0.083	0.202

Table 4: Forecasting performance (96→96) under different configurations. **Share** denotes the shared single-branch Transformer architecture. **Wiener** denotes WiSPformer without signal preservation. **Raw** uses only raw signals without the WiSP filter. **Uncalibrated** removes the calibration layer. Lower MSE and MAE are better, highlighted in **bold**.

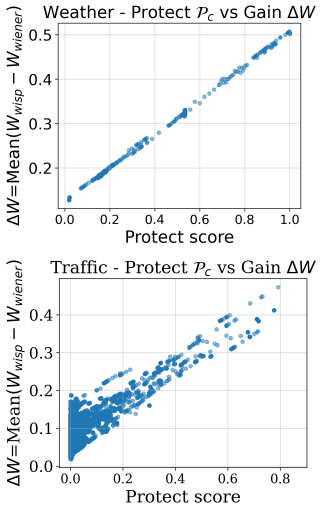


Figure 6: Relationship between $\Delta W = \mathrm{Mean}(W_{\mathrm{wisp}} - W_{\mathrm{wiener}})$ and protection score $\mathcal{P}_c$ on Weather and Traffic.

and $\bar{\Delta}_c$, where α controls the relative importance of the two values and τ_c determines each channel transient threshold. When both values remain small, the protection remain low, indicating relatively smooth channels. When either $\Delta_c^{\max}$ or $\bar{\Delta}_c$ increases, the $\mathcal{P}_c$ rises accordingly. For example, the points on the right side of the Traffic, $\Delta_c^{max} \approx 1.0$ and $\Delta_c^{\max} \approx 0.6$, reach high P_c scores. However, in Traffic, some channels reach very high $\bar{\Delta}_c$ values at the top-left corner, from 0.8 to 1.0, while their $\Delta_c^{\max}$ are less than 0.2. Further analysis of the protection mechanism is provided in the Appendix I.

Shared vs Dual-Branch Spectral Transformer. We then evaluate whether the dual-branch spectral Transformer performs better than a single-branch spectral Transformer that concatenates the real and imaginary components before a single Transformer block. As shown in Figure 5, the single-branch model Y_Share performs noticeably worse than Y_WiSP. In the time domain, Y_WiSP more closely follows Y_GTH. In the frequency domain of Figure 5, yFFT_WiSP is consistently closer to yFFT_GTH than yFFT_Share, particularly in the zoom-in frequency ranges. This demonstrates that the dual-branch structure better preserves the target spectral distribution and improves the forecasting performance in the time domain.

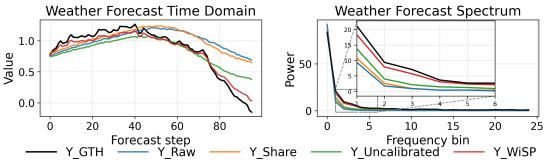


Figure 5: Comparison of forecasting outputs across configurations. Y_WiSP and yFFT_WiSP denote WiSPformer outputs in the time and frequency domains, respectively. Y_Share and yFFT_Share use the shared single-branch architecture. Y_GTH and yFFT_GTH denote the ground truth. Y_Raw and yFFT_Raw use raw signals only. Y_Uncalibrated and yFFT_Uncalibrated remove the calibration layer.

Effect of Complex-Consistent Calibration. We compare our model with calibration (Y_WiSP and yFFT_WiSP) and without calibration (Y_Uncalibrated and yFFT_Uncalibrated), as shown in Figure 5. The forecasting without calibration (Y_Uncalibrated) generally follows the temporal trend of Y_GTH. However, it still performs worse than the full model Y_WiSP in the zoomed window.

Transferability of WiSP to Other Forecasting Models.

Table 2 studies the transferability of WiSP by integrating it into three different forecasting backbones: FreTS (Yi et al., 2023b), TimeKAN (Huang et al., 2025), and iTransformer (Liu et al., 2024a). For each backbone, we report the performance before and after adding WiSP. Across most datasets,

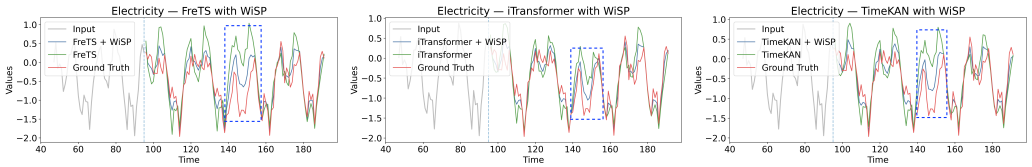


Figure 7: Visualisation for FreTS, iTransformer, and TimeKAN with/without WiSP on Electricity and ETTm2 datasets (96 → 96).

adding WiSP consistently reduces both MSE and MAE. Our WiSP is not architecture-specific and can serve as a general plug-in improvement.

The results suggest that WiSP is more compatible with frequency-oriented models (e.g., FreTS and TimeKAN) than the general sequence models (e.g., iTransformer). After comparing them, the improvements are often larger on FreTS and TimeKAN, particularly on ETTh2, Solar, Electricity and Weather. FreTS and TimeKAN perform representation learning in the frequency domain, so WiSP can exploit directly on the spectral features without an additional domain transformation. In contrast with iTransformer, the spectrum must first be transformed back to the time domain through iRFFT. So iTransformer no longer sees which frequency bins are explicitly amplified or which spectral components are removed. It only receives the reconstructed time-domain signal. Thus, the spectral prior introduced by WiSP is less directly accessible to the backbone, which may lead to smaller improvements than those achieved by explicitly spectral models.

Based on the visual comparisons in Figure 7, WiSP consistently improves different forecasting backbones, highlighted in the blue boxes. The predictions with WiSP better follow the ground-truth trends on both ETTm2 and Electricity, especially around sharp fluctuations and locally varying regions.

5 Related Work

Frequency-domain analysis has become an important direction in time-series forecasting because it captures diverse temporal patterns through spectral representations. Existing studies exploit frequency information in different ways. FiLM (Zhou et al., 2022b) uses Fourier projection for denoising, and FreTS (Yi et al., 2023c) learns representation directly with MLPs in the frequency domain. FilterNet (Yi et al., 2023c) applies frequency-domain filters to preserve informative components while suppressing high-frequency noise. Amplifier (Fei et al., 2025) enhances low-energy frequency components. TimeKAN (Huang et al., 2025) decomposes the input series into frequency bands with multi-order Kolmogorov-Arnold Networks. Recently, Transformer-based methods have also incorporated the frequency domain. FEDformer (Zhou et al., 2022a) introduces frequency-enhanced decomposition blocks within a Transformer framework for long-term forecasting. ETSformer (Woo et al., 2022) combines exponential smoothing and frequency attention. FreEformer (Yue et al., 2025) improves spectral representation learning through an enhanced attention transformer to produce more expressive attention patterns.

6 Conclusion

We presented *WiSPformer*, a signal-preserving Wiener spectral Transformer for multivariate time series forecasting. Although spectral denoising can improve robustness, spectral shrinkage may over-smooth informative local transients in non-stationary time series. WiSPformer combines a learnable Wiener spectral filter with an adaptive signal-preserving mechanism that limits attenuation of informative local changes. It further incorporates a dual-branch complex spectral Transformer with a calibration layer to model cross-channel dependencies. Experimental results on nine benchmark datasets show that WiSPformer consistently achieves competitive performance against 12 strong baselines. These results suggest that preserving predictive local transients can help improve spectral forecasting under non-stationary dynamics.

REFERENCES

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Syndar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=gerNCVqqtR.

Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A. Lozano. A review on outlier/anomaly detection in time series data. *ACM Comput. Surv.*, 54(3), April 2021. ISSN 0360-0300. doi: 10.1145/3444690. URL https://doi.org/10.1145/3444690.

Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. Spectral temporal graph neural network for multivariate time-series forecasting. In *Advances in Neural Information Processing Systems*, volume 33, pp. 17766–17778, 2020a. URL https://proceedings.neurips.cc/paper/2020/hash/cdf6581cb7aca4b7e19ef136c6e601a5-Abstract.html.

Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Conguri Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. Spectral temporal graph neural network for multivariate time-series forecasting. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020b. Curran Associates Inc. ISBN 9781713829546.

Mouxiang Chen, Lefei Shen, Han Fu, Zhuo Li, Jianling Sun, and Chenghao Liu. Calibration of time-series forecasting: Detecting and adapting context-driven distribution shift. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, pp. 341–352, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704901. doi: 10.1145/3637528.3671926. URL https://doi.org/10.1145/3637528.3671926.

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.

James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19(90):297–301, 1965. ISSN 00255718, 10886842. URL http://www.jstor.org/stable/2003354.

Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan K Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with tiDE: Time-series dense encoder. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=pCbC3aQB5W.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

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

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

Wei Fan, Pengyang Wang, Dongkun Wang, Dongjie Wang, Yuanchun Zhou, and Yanjie Fu. Dish-ts: A general paradigm for alleviating distribution shift in time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 7522–7529, 2023.

Jingru Fei, Kun Yi, Wei Fan, Qi Zhang, and Zhendong Niu. Amplifier: Bringing attention to neglected low-energy components in time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 11645–11653. AAAI Press, 2025. doi: 10.1609/aaai.v39i11.33267.

Yisong Fu, Zezhi Shao, Chengqing Yu, Yujie Li, Zhulin An, Cheems Wang, Yongjun Xu, and Fei Wang. Selective learning for deep time series forecasting. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=kgzRy6nD6D.

Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. Softs: efficient multivariate time series forecasting with series-core fusion. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

Sungwon Han, Seungeon Lee, Meeyoung Cha, Sercan O Arik, and Jinsung Yoon. Retrieval augmented time series forecasting. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=GUDnecJdJU.

Songtao Huang, Zhen Zhao, Can Li, and LEI BAI. TimeKAN: KAN-based frequency decomposition learning architecture for long-term time series forecasting. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=wTLc79YNbh.

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. In *International Conference on Learning Representations (ICLR)*, 2024.

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.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. URL https://arxiv.org/abs/1412.6980.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, pp. 95–104, New York, NY, USA, 2018. Association for Computing Machinery. doi: 10.1145/3209978.3210006.

Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379 (2194):20200209, February 2021. ISSN 1471-2962. doi: 10.1098/rsta.2020.0209. URL http://dx.doi.org/10.1098/rsta.2020.0209.

Bryan Lim, Sercan Ö. Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2021.03.012. URL https://www.sciencedirect.com/science/article/pii/S0169207021000637.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Haojun Chen, and Junjie Yang. Sparsetsf: modeling long-term time series forecasting with 1k parameters. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=ucNDIDRNjjv.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *ICLR*, 2024a.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: generative pre-trained transformers are large time series models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024b.

Zhiding Liu, Mingyue Cheng, Zhi Li, Zhenya Huang, Qi Liu, Yanhu Xie, and Enhong Chen. Adaptive normalization for non-stationary time series forecasting: A temporal slice perspective. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=5BqDSw8r5j.

Zhiding Liu, Mingyue Cheng, GuanHao Zhao, Jiqian Yang, Qi Liu, and Enhong Chen. Improving time series forecasting via instance-aware post-hoc revision. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=H7e5RpeIi4.

Jiecheng Lu, Xu Han, Yan Sun, and Shihao Yang. CATS: Enhancing multivariate time series forecasting by constructing auxiliary time series as exogenous variables. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=1lDAGDe0UR.

Stéphane Mallat. *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*. Academic Press, Inc., USA, 3rd edition, 2008. ISBN 0123743702.

Fausto Pedro García Márquez and Noor Zaman. *Digital Filters and Signal Processing*. IntechOpen, London, January 2013. doi: 10.5772/45654. URL https://doi.org/10.5772/45654.

Nijat Mehdiyev, Johannes Lahann, Andreas Emrich, David Enke, Peter Fettke, and Peter Loos. Time series classification using deep learning for process planning: A case from the process industry. *Procedia Computer Science*, 114:242–249, 2017. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2017.09.066. URL https://www.sciencedirect.com/science/article/pii/S1877050917318707. Complex Adaptive Systems Conference with Theme: Engineering Cyber Physical Systems, CAS October 30 – November 1, 2017, Chicago, Illinois, USA.

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

Alan V. Oppenheim and Ronald W. Schafer. *Discrete-Time Signal Processing*. Prentice Hall Press, USA, 3rd edition, 2009. ISBN 0131988425.

S. Palani. *Signals and Systems*. Springer International Publishing, 2021. ISBN 9783030757427. URL https://books.google.com.au/books?id=b3c_EAAAQBAJ.

Kwangryeol Park, Jaeho Kim, and Seulki Lee. Alio: Output alignment matters in long-term time series forecasting. In *Thirty-ninth Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=AuOZDp4gy7.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. *PyTorch: an imperative style, high-performance deep learning library*. Curran Associates Inc., Red Hook, NY, USA, 2019.

Xihao Piao, Zheng Chen, Taichi Murayama, Yasuko Matsubara, and Yasushi Sakurai. Fredformer: Frequency debiased transformer for time series forecasting. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, pp. 2400–2410, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704901. doi: 10.1145/3637528.3671928. URL https://doi.org/10.1145/3637528.3671928.

Zongjiang Shang, Ling Chen, Binqing Wu, and Dongliang Cui. Ada-mshyper: adaptive multi-scale hypergraph transformer for time series forecasting. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

Artyom Stitsyuk and Jaesik Choi. xpatch: Dual-stream time series forecasting with exponential seasonal-trend decomposition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 20601–20609, 2025. doi: 10.1609/aaai.v39i19.34270.

Saeed V. Vaseghi. *Wiener Filters*, pp. 140–163. Vieweg+Teubner Verlag, Wiesbaden, 1996. ISBN 978-3-322-92773-6. doi: 10.1007/978-3-322-92773-6_5. URL https://doi.org/10.1007/978-3-322-92773-6_5.

Hao Wang, Licheng Pan, Zhichao Chen, Degui Yang, Sen Zhang, Yifei Yang, Xinggao Liu, Haoxuan Li, and Dacheng Tao. Fredf: Learning to forecast in the frequency domain. In *ICLR*, 2025.

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 *International Conference on Learning Representations (ICLR)*, 2024a.

Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Yong Liu, Yunzhong Qiu, Haoran Zhang, Jianmin Wang, and Mingsheng Long. Timexer: Empowering transformers for time series forecasting with exogenous variables. *Advances in Neural Information Processing Systems*, 2024b.

Norbert Wiener. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. The MIT Press, 1964. ISBN 0262730057.

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. In *Advances in Neural Information Processing Systems*, 2021.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *ICLR*, 2023a.

Haixu Wu, Hang Zhou, Mingsheng Long, and Jianmin Wang. Interpretable weather forecasting for worldwide stations with a unified deep model. *Nature Machine Intelligence*, 5:602–611, 06 2023b. doi: 10.1038/s42256-023-00667-9.

Wang Xue, Tian Zhou, QingSong Wen, Jinyang Gao, Bolin Ding, and Rong Jin. Card: Channel aligned robust blend transformer for time series forecasting. In *International Conference on Learning Representations (ICLR)*, 2024.

Kun Yi, Qi Zhang, Wei Fan, Hui He, Liang Hu, Pengyang Wang, Ning An, Longbing Cao, and Zhendong Niu. FourierGNN: Rethinking multivariate time series forecasting from a pure graph perspective. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a.

Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Ning An, Defu Lian, Longbing Cao, and Zhendong Niu. Frequency-domain MLPs are more effective learners in time series forecasting. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b.

Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Ning An, Defu Lian, Longbing Cao, and Zhendong Niu. Frequency-domain MLPs are more effective learners in time series forecasting. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023c.

Kun Yi, Jingru Fei, Qi Zhang, Hui He, Shufeng Hao, Defu Lian, and Wei Fan. Filternet: Harnessing frequency filters for time series forecasting. In *Advances in Neural Information Processing Systems*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/6323d96f79d5d49e0d3fc88835c082cd-Abstract-Conference.html.

Guoqi Yu, Jing Zou, Xiaowei Hu, Angelica I Aviles-Rivero, Jing Qin, and Shujun Wang. Revitalizing multivariate time series forecasting: Learnable decomposition with inter-series dependencies and intra-series variations modeling. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=87CYNyCGOo.

Wenzhen Yue, Yong Liu, Xianghua Ying, Bowei Xing, Ruohao Guo, and Ji Shi. Freeformer: frequency enhanced transformer for multivariate time series forecasting. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, IJCAI '25, 2025. ISBN 978-1-956792-06-5. doi: 10.24963/ijcai.2025/401. URL https://doi.org/10.24963/ijcai.2025/401.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *AAAI*, 2023.

Zhiwei Zhang, Xinyi Du, Xuanchi Guo, Weihao Wang, and Wenjuan Han. Emaformer: Enhancing transformer through embedding armor for time series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(34):28636–28644, Mar. 2026. doi: 10.1609/aaai.v40i34.40095. URL https://ojs.aaai.org/index.php/AAAI/article/view/40095.

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 *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*, volume 35, pp. 11106–11115. AAAI Press, 2021.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. *ICML*, 2022a.

Tian Zhou, Ziqing Ma, xue wang, Qingsong Wen, Liang Sun, Tao Yao, Wotao Yin, and Rong Jin. FiLM: Frequency improved legendre memory model for long-term time series forecasting. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022b. URL https://openreview.net/forum?id=zTQdHSQUQWc.

AI USE STATEMENT

In this work, we used generative AI tools for polishing writing only. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Supplementary Material

A Extended Related Work

A.1 Robust and non-stationary time-series forecasting

Real-world time-series often exhibit non-stationarity, distribution shifts, abrupt local changes, noisy observations, and instance-level variations. These factors can make forecasting models unreliable under practical deployment conditions (Kim et al., 2022; Liu et al., 2022; Fan et al., 2023; Blázquez-García et al., 2021; Liu et al., 2025; Fu et al., 2025).

Normalization-based approaches aim to alleviate distribution shift between training and test data. RevIN (Kim et al., 2022) applies reversible instance normalization to reduce the effect of distribution shift in time-series forecasting. Non-stationary Transformer (Liu et al., 2022) explicitly studies the stationarity problem in Transformer-based forecasting and designs mechanisms to improve modelling under non-stationary distributions. DISH-TS (Fan et al., 2023) mitigates distribution shift in time-series forecasting. Further works consider adaptation and calibration under changing temporal contexts (Liu et al., 2023; Chen et al., 2024).

A.2 General forecasting architectures

Recent forecasting architectures have improved long-term time-series forecasting through different modelling designs. Transformer-based methods have been widely used because of their ability to model long-range dependencies (Zhou et al., 2021; Wu et al., 2021; Zhou et al., 2022a; Nie et al., 2023; Liu et al., 2024a). Informer (Zhou et al., 2021) improves the efficiency of self-attention for long sequence forecasting. Autoformer (Wu et al., 2021) introduces decomposition and auto-correlation mechanisms. FEDformer (Zhou et al., 2022a) further incorporates frequency-domain decomposition. PatchTST (Nie et al., 2023) segments time-series into patches and applies Transformer modelling over patch tokens, showing strong performance on long-term forecasting benchmarks. iTransformer (Liu et al., 2024a) inverts the tokenization scheme by treating variables as tokens, improving multivariate dependency modelling. Autoformer (Wu et al., 2021) captures spectral and periodic patterns through an auto-correlation mechanism with progressive frequency decomposition. Leddam (Yu et al., 2024) adopts learnable decomposition to separate trend and seasonal components and employs a dual-attention module to capture both inter-series dependencies and intra-series variations.

MLP-based and lightweight architectures have also shown competitive forecasting performance. DLinear (Zeng et al., 2023) demonstrates that simple linear models with decomposition can be strong baselines for long-term forecasting. TSMixer (Chen et al., 2023) adopts an all-MLP architecture to mix temporal and channel information. TiDE (Das et al., 2023) uses a dense encoder-decoder design for long-term forecasting. TimeMixer (Wang et al., 2024a) performs decomposable multi-scale mixing to capture temporal patterns at different resolutions. SparseTSF (Lin et al., 2024) explores lightweight long-term forecasting with very few parameters.

Convolutional and hybrid models provide another direction. TimesNet (Wu et al., 2023a) represents time-series as temporal 2D variations and uses convolutional structures for general time-series analysis. ModernTCN (donghao & wang xue, 2024b) proposes a modern convolution architecture for time-series tasks. SOFTS (Han et al., 2024) introduces series-core fusion for efficient multivariate forecasting. Graph-based models also explicitly capture relationships among variables, such as spectral temporal graph neural networks for multivariate forecasting (Yi et al., 2023a; Cao et al., 2020b).

A.3 Context-aware and retrieval-based forecasting

Another line of work improves forecasting by incorporating contextual information, exogenous variables, or retrieved historical patterns. Temporal Fusion Transformer (Lim et al., 2021) combines recurrent layers, attention, gating, and variable selection for interpretable multi-horizon forecasting. TimeXer (Wang et al., 2024b) empowers Transformer-based forecasting with exogenous variables. CATS (Lu et al., 2024) constructs auxiliary time-series as exogenous variables to enhance multivariate

forecasting. Retrieval-augmented forecasting methods further exploit similar historical patterns to improve predictions (Han et al., 2025).

A.4 TIME-SERIES FOUNDATION MODELS AND FUTURE DIRECTIONS

Recently, time-series foundation models have attracted increasing attention. These models aim to learn general-purpose temporal representations from large-scale time-series data and transfer them to downstream forecasting tasks. Chronos (Ansari et al., 2024) formulates time-series forecasting as a language modelling problem by discretizing time-series values into tokens. TimesFM (Das et al., 2024) proposes a decoder-only foundation model for time-series forecasting. Timer (Liu et al., 2024b) studies generative pre-trained Transformers as large time-series models. Time-LLM (Jin et al., 2024) adapts large language models to time-series forecasting through reprogramming.

These foundation models demonstrate the potential of large-scale pretraining for time-series forecasting. However, incorporating spectral denoising and local transient protection into foundation models remains underexplored. Since foundation models are often trained across heterogeneous datasets with different temporal resolutions, noise levels, and periodic structures, adaptive spectral protection may be useful for improving robustness. Extending WiSPformer to pretrained time-series foundation models is therefore a promising direction for future work.

B EXPERIMENTS

B.1 EXPERIMENTAL SETUP

All experiments in this study were carried out using PyTorch (Paszke et al., 2019) on a single NVIDIA RTX A6000 GPU with 48GB. MSE and MAE are used as evaluation metrics, and lower values mean better performance. For a fair comparison, the long-term forecasting benchmarks follow the common setting (Liu et al., 2024a; Wu et al., 2021) with the lookback window length L set to 96 and the prediction horizon H to 96, 192, 336, 720 for all datasets. Following the previous methods (Nie et al., 2023; Yue et al., 2025), we use RevIN (Yu et al., 2024) as our instance normalization block. We optimize the model using the ADAM optimizer (Kingma & Ba, 2015) with an initial learning rate selected from $\{10^{-4}, 5 \times 10^{-4}\}$. The Transformer model dimension D is chosen from $\{256, 512\}$, while the embedding dimension d is fixed at 16. The batch size is determined based on the dataset size and selected from $\{16, 32\}$. The training is conducted for a maximum of 50 epochs with an early stopping mechanism.

We provide the links to our code and the data in https://github.com/secretpaperauthor/WiSPformer-iclr/ .

B.2 DATASETS

We conduct extensive experiments on nine widely used real-world datasets for long-term time-series forecasting, including four ETT subsets (ETTh1, ETTh2, ETTm1, and ETTm2), Electricity, Exchange, Weather, Solar-Energy, and Traffic (Liu et al., 2024a; Shang et al., 2024; Wu et al., 2021). These datasets cover diverse domains such as energy, finance, weather, and transportation, providing a comprehensive benchmark for evaluating forecasting performance. Following prior work, we preprocess all datasets using standard procedures. The datasets are split chronologically into training, validation, and test sets.

Brief descriptions of the datasets are provided below.

- **ETT**: The Electricity Transformer Temperature (ETT) dataset records oil temperature and load metrics from electricity transformers between July 2016 and July 2018. It contains four subsets: ETTh1 and ETTh2 sampled hourly, and ETTm1 and ETTm2 sampled every 15 minutes.
- **Electricity (ECL)**: The Electricity dataset contains the hourly electricity consumption of 321 clients, measured in kWh, from July 2016 to July 2019.
- **Exchange**: This dataset contains daily exchange rates from 1990 to 2016 for eight countries, providing a benchmark for financial time-series forecasting.

Table 5: Overview of the datasets used in the experiments.

Dataset	Variables	Frequency	Length	Scope
ETTh1/ETTh2	7	1 Hour	17,420	Energy
ETTm1/ETTm2	7	15 Minutes	69,680	Energy
Electricity (ECL)	321	1 Hour	26,304	Energy
Exchange	8	Daily	7,588	Economy
Solar	137	10 Minutes	52,560	Nature
Weather	21	10 Minutes	52,696	Nature
Traffic	862	1 Hour	17,544	Transportation

- **Solar**: The Solar dataset records solar power production from 137 photovoltaic plants in Alabama State in 2006, sampled every 10 minutes.
- **Weather**: The Weather dataset contains 21 meteorological indicators, including air temperature and humidity, recorded every 10 minutes at the Weather Station of the Max Planck Institute for Biogeochemistry in 2020.
- **Traffic**: The Traffic dataset contains hourly road occupancy rates collected from 862 sensors on San Francisco freeways.

C Fourier Representation and Spectral Filtering

Let $\mathbf{X} \in \mathbb{R}^{T\times N}$ denote a multivariate time-series segment, where T is the number of time steps and N is the number of channels. For a univariate sequence $x[t]$, the discrete Fourier transform (DFT) (Palani, 2021) and inverse discrete Fourier transform (iDFT) (Palani, 2021) are defined as

$$\begin{aligned} Z[k] &= \sum_{t=0}^{T-1} x[t] e^{-j2\pi kt/T}, \qquad k = 0, 1, \ldots, T-1, \\ x[t] &= \frac{1}{T} \sum_{k=0}^{T-1} Z[k] e^{j2\pi kt/T}, \qquad t = 0, 1, \ldots, T-1, \end{aligned} \tag{19}$$

where j denotes the imaginary unit.

For real-valued time-series signals, the Fourier spectrum satisfies Hermitian symmetry (Palani, 2021), which means that the negative-frequency components are redundant and are determined by the positive-frequency components. Therefore, when using the real-valued fast Fourier transform (rFFT) (Cooley & Tukey, 1965), only the non-redundant frequency bins are retained. The number of such bins is

$$F = \left\lfloor \frac{T}{2} \right\rfloor + 1. \tag{20}$$

Applying rFFT to each channel of $\mathbf{X}$ along the temporal dimension yields the frequency-domain representation

$$\mathbf{Z} = \mathrm{rFFT}(\mathbf{X}) \in \mathbb{C}^{F\times N}. \tag{21}$$

D Derivation of the Wiener Filter

Lemma 2 (Classical Wiener Filter)**.** *Let the observed signal be*

$$x(t) = s(t) + n(t), \tag{22}$$

and let its rFFT representation be

$$Z(k) = S(k) + N(k), \tag{23}$$

where $Z(k)$, $S(k)$, and $N(k)$ denote the spectra of the observed signal, the true signal, and the noise, respectively. Consider the linear spectral estimator

$$\hat{S}(k) = H(k)Z(k), \tag{24}$$

with the mean squared error objective

$$\mathcal{L}_k(H) = \mathbb{E}\big[|H(k)Z(k) - S(k)|^2\big]. \tag{25}$$

If $S(k)$ and $N(k)$ are uncorrelated and $H(k) \in \mathbb{R}$, then the optimal Wiener response is

$$H^\star(k) = \frac{P_s(k)}{P_s(k) + P_n(k)}, \tag{26}$$

where

$$P_s(k) = \mathbb{E}[|S(k)|^2], \qquad P_n(k) = \mathbb{E}[|N(k)|^2]. \tag{27}$$

Proof. Substituting $Z(k) = S(k) + N(k)$ into the objective gives

$$\begin{aligned}\mathcal{L}_k(H) = \mathbb{E}\Big[&|(H(k) - 1)S(k)|^2 + |H(k)N(k)|^2 \\ &+ (H(k) - 1)S(k)H^*(k)N^*(k) \\ &+ (H^*(k) - 1)S^*(k)H(k)N(k)\Big].\end{aligned} \tag{28}$$

Since $S(k)$ and $N(k)$ are uncorrelated, we have $\mathbb{E}[S(k)N^*(k)] = 0$, and thus

$$\mathcal{L}_k(H) = |H(k) - 1|^2 \, P_s(k) + |H(k)|^2 \, P_n(k). \tag{29}$$

Assuming $H(k)$ is real-valued, this reduces to

$$\mathcal{L}_k(H) = (H(k) - 1)^2 P_s(k) + H^2(k) P_n(k). \tag{30}$$

Taking the derivative with respect to $H(k)$ and setting it to zero yields

$$\frac{\partial \mathcal{L}_k(H)}{\partial H(k)} = 2(H(k) - 1)P_s(k) + 2H(k)P_n(k) = 0, \tag{31}$$

which gives

$$H^\star(k) = \frac{P_s(k)}{P_s(k) + P_n(k)}. \tag{32}$$

□

Proposition 2 (SNR form of Wiener gain)**.** *Define the spectral signal-to-noise ratio at frequency k as*

$$\rho(k) = \frac{P_s(k)}{P_n(k)}. \tag{33}$$

Then the optimal Wiener gain can be equivalently written as

$$H^\star(k) = \frac{\rho(k)}{1 + \rho(k)}. \tag{34}$$

Proof. Starting from the Wiener gain in Lemma 1,

$$H^\star(k) = \frac{P_s(k)}{P_s(k) + P_n(k)}, \tag{35}$$

dividing the numerator and denominator by $P_n(k)$ gives

$$H^\star(k) = \frac{P_s(k)/P_n(k)}{P_s(k)/P_n(k) + 1} = \frac{\rho(k)}{1 + \rho(k)}. \tag{36}$$

□

E Our Sigmoidized Log-SNR Wiener Gain

E.1 Wiener Parameterization of the Spectral Gain

Given a multivariate time-series $\mathbf{X} \in \mathbb{R}^{T\times N}$, where T is the number of time steps and N is the number of channels, let $P_s, P_n \in \mathbb{R}^{F\times N}$ denote the learnable signal and noise power spectral density. For channel $c \in \{1,\dots,N\}$ and frequency bin $k \in \{0,\dots,F-1\}$, we define the Wiener spectral gain as

$$H_c(k) = \frac{P_s^c(k)}{P_s^c(k) + P_n^c(k)} = \frac{\rho_c(k)}{1+\rho_c(k)}, \tag{37}$$

where $P_s^c(k)$ and $P_n^c(k)$ are the entries of P_s and P_n at frequency bin k and channel c, respectively.

Proposition 3 (Log-SNR form of learnable Wiener gain)**.** ***Our learnable Wiener gain*** *can be equivalently written as*

$$\begin{aligned} H_c(k) &= \sigma\big(\log P_s^c(k) - \log P_n^c(k)\big) \\ &= \sigma\big(\log \rho_c(k)\big) \end{aligned} \tag{38}$$

where $\sigma(x) = \frac{e^x}{1+e^x}$ *is the* ***sigmoid function*** *and* $0 \le H_c(k) \le 1$.

Proof. From the definition of the spectral signal-to-noise ratio,

$$\rho_c(k) = \frac{P_s^c(k)}{P_n^c(k)}, \tag{39}$$

Taking the logarithm gives

$$\log \rho_c(k) = \log P_s^c(k) - \log P_n^c(k). \tag{40}$$

$$\rho_c(k) = \exp\big(\log P_s^c(k) - \log P_n^c(k)\big). \tag{41}$$

Substituting this into the Wiener gain in eq. 37, we obtain

$$\begin{aligned} H_c(k) &= \frac{\exp\big(\log P_s^c(k) - \log P_n^c(k)\big)}{1+\exp\big(\log P_s^c(k) - \log P_n^c(k)\big)} \\ &= \sigma\big(\log P_s^c(k) - \log P_n^c(k)\big) \\ &= \sigma\big(\log \rho_c(k)\big) \end{aligned} \tag{42}$$

Since the sigmoid function maps $\mathbb{R}$ to $[0,1]$, it follows $0 \le H_c(k) \le 1$. □

E.2 Learning the Latent Log-SNR $\rho_c(k)$

The derivation above illustrates the Wiener form of the spectral gain, but it does not yet explain why the SNR-related quantity can itself be learned. Classical Wiener filtering assumes that the signal and noise power spectra are known or estimated from observations. In WiSPformer, we instead parameterize their ratio through a learnable latent variable and optimize it end-to-end.

Latent log-SNR parameterization. For a channel c and frequency bin k, define a corresponding spectral gain as

$$H_c(k) = \sigma(\log \hat{\rho}_c(k)) = \sigma\big(z_c^k\big). \tag{43}$$

Here, $\hat{\rho}_c(k)$ denotes the learned latent SNR representation. To simplify notation, we omit the channel and frequency indices below. Let the observed spectral coefficient be

$$Z = S + N, \tag{44}$$

where S and N denote the clean signal and additive noise, respectively. Assume that S and N are zero-mean and uncorrelated, with

$$P_s = \mathbb{E}[|S|^2], \qquad P_n = \mathbb{E}[|N|^2]. \tag{45}$$

Consider the Wiener reconstruction objective

$$\mathcal{L}(z) = \mathbb{E}\left[|\sigma(z)Z - S|^2\right]. \tag{46}$$

Let $H = \sigma(z)$. Since S and N are uncorrelated, Eq. equation 46 becomes

$$\mathcal{L}(H) = (H-1)^2 P_s + H^2 P_n. \tag{47}$$

Taking the derivative with respect to H gives

$$\frac{\partial \mathcal{L}}{\partial H} = 2(H-1)P_s + 2HP_n. \tag{48}$$

Setting the derivative to zero yields

$$H^\star = \frac{P_s}{P_s + P_n}. \tag{49}$$

Because the sigmoid function is strictly monotonic and invertible, the corresponding optimal latent variable is

$$\begin{aligned} z^\star &= \operatorname{logit}(H^\star) \\ &= \log \frac{H^\star}{1-H^\star} \\ &= \log \frac{P_s/(P_s+P_n)}{P_n/(P_s+P_n)} \\ &= \log \frac{P_s}{P_n}. \end{aligned} \tag{50}$$

Therefore,

$$\boxed{z^\star = \log \rho} \tag{51}$$

where $\rho = P_s/P_n$ is the Wiener filter spectral SNR.

This result shows that the log-SNR does not need to be explicitly provided as a training target. Under the Wiener parameterization, it appears directly as the optimal latent variable of the reconstruction objective.

Gradient-based learnability. Since the spectral gain is parameterized by the sigmoid function,

$$H = \sigma(z) = \frac{1}{1+e^{-z}}, \tag{52}$$

its derivative with respect to the latent log-SNR is

$$\begin{aligned} \frac{\partial H}{\partial z} &= \frac{\partial}{\partial z}\left(\frac{1}{1+e^{-z}}\right) \\ &= \frac{e^{-z}}{(1+e^{-z})^2} \\ &= \frac{1}{1+e^{-z}}\left(1 - \frac{1}{1+e^{-z}}\right) \\ &= H(1-H). \end{aligned} \tag{53}$$

the gradient of the reconstruction objective with respect to the latent log-SNR is

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial z} &= \frac{\partial \mathcal{L}}{\partial H}\frac{\partial H}{\partial z} \\ &= 2\left[H(P_s+P_n) - P_s\right] H(1-H). \end{aligned} \tag{54}$$

Because $H(1-H) > 0$ for $0 < H < 1$, and $H < H^\star$, then

$$H(P_s + P_n) - P_s < 0, \tag{55}$$

and therefore

$$\frac{\partial \mathcal{L}}{\partial z} < 0. \tag{56}$$

Gradient descent consequently increases z, which increases H.

Conversely, if $H > H^\star$, then

$$\frac{\partial \mathcal{L}}{\partial z} > 0, \tag{57}$$

so gradient descent decreases z and consequently decreases H.

Thus, gradient-based optimization drives the latent variable toward

$$z^\star = \log \frac{P_s}{P_n}. \tag{58}$$

This provides a direct optimization interpretation of why the latent SNR representation is learnable.

Hence, the forecasting loss can directly optimize the latent log-SNR through back-propagation. Frequencies whose preservation reduces forecasting error receive gradients that increase their latent log-SNR and their spectral gain, whereas frequencies whose preservation is detrimental can receive lower latent log-SNR values.

F Interpretation of the Learnable Channel-Wise Adaptive Protection

This section provides a detailed interpretation of how the channel-wise protection mechanism interacts with the frequency-wise Wiener gain in WiSPformer.

F.1 Channel-wise Adaptive Protection

The protection mechanism intentionally operates at the channel level rather than the frequency level. The learned Wiener gain $H_c(k)$ already performs frequency-wise reliability estimation by assigning an individual attenuation factor to each spectral bin according to its estimated spectral SNR. Introducing another frequency-wise protection term would therefore largely duplicate the role of the Wiener filter.

Rather than determining which individual frequency bins should be retained, it identifies channels containing informative transient patterns that are important for forecasting. Such temporal transients are generally represented by coordinated energy distributed across multiple frequency bins. Although Wiener filter may appropriately attenuate each frequency bin according to its estimated SNR, the cumulative attenuation across many transient-related frequency bins may substantially weaken the underlying transient after inverse transformation, resulting in the loss of predictive information. Therefore, protection is estimated at the channel level to preserve the overall transient structure rather than individual spectral coefficients.

The protection mechanism therefore operates at a different granularity. As Eq. 10, it can be rewritten as

$$W_c(k) = H_c(k) + \lambda_c^{\mathrm{prot}} \left(1 - H_c(k)\right), \tag{59}$$

The first term, $H_c(k)$ retains the original frequency-wise shrinkage, while the second term restores part of the spectral magnitude that would otherwise be attenuated. Thus, $\lambda_c^{\mathrm{prot}}$ does not replace the Wiener filter gain. Instead, determines the overall protection strength for a transient-rich channel, whereas the actual restoration of each frequency bin remains adaptively controlled by its corresponding Wiener gain $H_c(k)$.

The Wiener filter answers which frequency bins are reliable, whereas the protection mechanism answers whether the current channel contains predictive transient information that should not be excessively suppressed.

F.2 Analysis of Signal and Noise Preservation

A common concern is whether the proposed channel-wise protection may also restore the noise removed by the Wiener filter. In Eq. 59, the proposed protection relaxes the attenuation while

preserving its frequency-selective structure. The signal components are preferentially recovered, whereas noise components remain attenuated relative to the raw spectrum.

According to the Wiener gain Eq. 43 and WiSP gain Eq. 59, we define the Wiener attenuation factor as:

$$A_c^{\mathrm{W}}(k) = 1 - H_c(k), \tag{60}$$

and the WiSPformer attenuation factor as:

$$A_c^{\mathrm{WiSP}}(k) = 1 - W_c(k). \tag{61}$$

Substituting Eq. (59) gives

$$\begin{aligned} A_c^{\mathrm{WiSP}}(k) &= 1 - \left[\lambda_c^{\mathrm{prot}} + (1 - \lambda_c^{\mathrm{prot}})H_c(k)\right] \\ &= (1 - \lambda_c^{\mathrm{prot}})\left(1 - H_c(k)\right) \\ &= (1 - \lambda_c^{\mathrm{prot}})A_c^{\mathrm{W}}(k). \end{aligned} \tag{62}$$

Eq. (62) shows that the proposed protection does not construct a second frequency-wise spectral mask. Instead, it scales the attenuation produced by the Wiener filter while preserving its spectral attenuation pattern.

For example, given two frequency bins k_i and k_j in the same channel satisfying $H_c(k_i) > H_c(k_j)$, the larger Wiener gain indicates a frequency component with higher estimated spectral reliability and $A_c^{\mathrm{W}}(k_j) > A_c^{\mathrm{W}}(k_i)$. Their attenuation factors satisfy:

$$A_c^{\mathrm{WiSP}}(k_i) = (1 - \lambda_c^{\mathrm{prot}})A_c^{\mathrm{W}}(k_i), \tag{63}$$

$$A_c^{\mathrm{WiSP}}(k_j) = (1 - \lambda_c^{\mathrm{prot}})A_c^{\mathrm{W}}(k_j). \tag{64}$$

For $0 \le \lambda_c^{\mathrm{prot}} < 1$, the scaling factor $(1 - \lambda_c^{\mathrm{prot}})$ is positive, and therefore

$$\boxed{A_c^{\mathrm{WiSP}}(k_i) < A_c^{\mathrm{WiSP}}(k_j).} \tag{65}$$

Therefore, although channel-wise protection relaxes the overall attenuation strength, it preserves the attenuation ordering established by the Wiener filter. Frequency components with higher Wiener gains continue to experience smaller attenuation, whereas components with lower Wiener gains remain more strongly suppressed. The Figure 11 of the Appendix I also indicates that our WiSPformer adaptively balances transient preservation and spectral denoising by retaining informative components while still suppressing part of the high-frequency magnitude.

G Complex-Consistent Calibration Layer

Let the outputs of real transformer and imaginary transformer be $H_{\Re}$ and $H_{\Im}$ so that the complex spectrum can be defined as:

$$H = H_{\Re} + jH_{\Im}, \qquad C = a + jb \tag{66}$$

where a and b are learnable parameters. The output of the Mixing Layer Y is

$$\begin{aligned} Y &= CH \\ &= (a + jb)(H_{\Re} + jH_{\Im}) \\ &= aH_{\Re} + jaH_{\Im} + jbH_{\Re} + j^2 bH_{\Im} \\ &= (aH_{\Re} - bH_{\Im}) + j(aH_{\Im} + bH_{\Re}) \\ &= Y_{\Re} + jY_{\Im}. \end{aligned}$$

where $Y_{\Re} = aH_{\Re} - bH_{\Im}$ and $Y_{\Im} = aH_{\Im} + bH_{\Re}$.

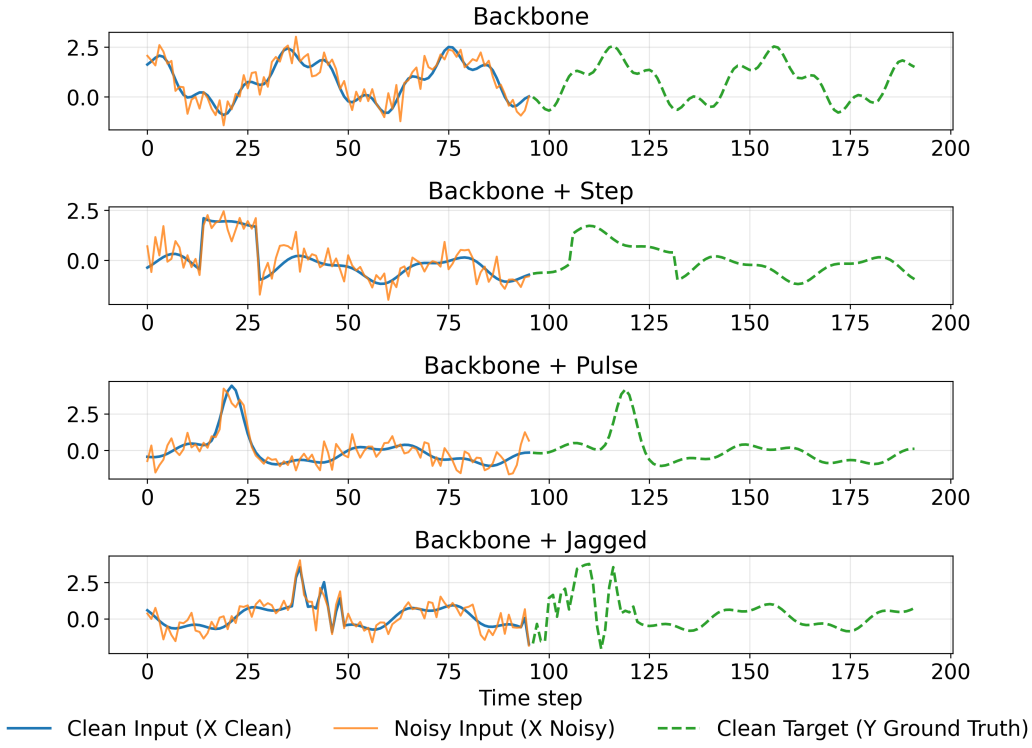


Figure 8: Synthetic dataset demonstration.

This shows that the mixing layer is exactly equivalent to a complex multiplication, rather than an arbitrary real-valued fusion of the two branches.

Furthermore, we can rewrite the complex coefficient $C = a + jb$ in polar form,

$$C = \gamma e^{j\theta} \tag{67}$$

where $\gamma = \sqrt{a^2 + b^2}$ represents the modulus of the complex number and $\theta = \text{atan2}(b, a)$ represents the phase of the complex number.

$$Y = CH = \gamma e^{j\theta} H \tag{68}$$

Therefore, it clearly explicates the role of the mixing layer:

1. **Magnitude scaling** $|Y| = \gamma |H|$, which shows that the mixing layer performs a frequency-wise rescaling of the spectral magnitude.
2. **Phase rotation** $\phi(Y) = \phi(H) + \theta$, which shows that the mixing layer applies a learnable phase rotation to the complex spectrum.

H Synthetic Dataset

The synthetic dataset is designed to evaluate the forecasting and denoising of the WiSPformer on multivariate time-series with both smooth background dynamics and structured local events. As Fig. 8 shows, this dataset contains 3 channels. Each channel signal is generated from a common framework: a slowly varying backbone signal is first created to represent the underlying seasonal trend, and a transient component is added to introduce temporal patterns. These transient components include **step**, **pulse**, and **jagged** signals, which model sudden level shifts, localized smooth peaks, and irregular short-term fluctuations, respectively. Finally, white Gaussian noise is added to each signal to produce the observed noisy input.

H.1 SYNTHETIC DATASET CONSTRUCTION

To evaluate the ability of WiSPformer to preserve predictive local transients while suppressing background noise, we construct a synthetic multivariate dataset consisting of a smooth backbone signal and three representative transient patterns: step changes, pulse peaks, and jagged fluctuations.

Backbone Signal. For a sequence of length T, the backbone signal b_t is generated as the sum of two sinusoidal components with random periods and random phases, together with a mild linear trend. Specifically, the two phases satisfy $\phi_1, \phi_2 \sim \mathcal{U}(0, 2\pi)$. The first sinusoid has a relatively long period $P_1 \sim \mathcal{U}(24, 48)$ and amplitude 0.4, while the second has a shorter period $P_2 \sim \mathcal{U}(8, 16)$ and amplitude 0.15. A small linear trend is added with slope $\beta \sim \mathcal{U}(-0.5, 0.5)$. The resulting backbone represents a smooth seasonal signal with a slight long-term drift:

$$b_t = 0.4 \sin\left(\frac{2\pi t}{P_1} + \phi_1\right) + 0.15 \sin\left(\frac{2\pi t}{P_2} + \phi_2\right) + \beta \frac{t}{T}. \tag{69}$$

Step Signal. The first transient component introduces piecewise constant jumps. Each step lasts for a random duration $d_k \sim \mathcal{U}\{10, 30\}$, with amplitude $a_k \sim \mathcal{U}(1.0, 2.0)$ and random sign $r_k \in \{-1, +1\}$. This produces short intervals where the signal abruptly shifts upward or downward before returning to the baseline.

$$s_t = \sum_{k=1}^{K_s} r_k a_k \mathbf{1}(d_k). \tag{70}$$

Pulse Signal. The second transient component generates localized Gaussian peaks. Each pulse has a fixed amplitude $A_p = 3.0$ and width of 10 time steps. Let c_k denote the pulse center and $\sigma = 10/4$ be the standard deviation. The pulse signal is

$$p_t = \sum_{k=1}^{K_p} A_p \exp\left(-\frac{(t - c_k)^2}{2\sigma^2}\right). \tag{71}$$

This produces sharp but smooth transient peaks distributed across the sequence.

Jagged Signal. The third transient component models irregular high-frequency fluctuations. Each jagged segment has length

$$\ell_k \sim \mathcal{U}\{10, 30\},$$

with amplitude

$$a_k \sim \mathcal{U}(1.0, 2.0),$$

random sign

$$r_k \in \{-1, +1\},$$

and random fluctuations

$$\epsilon^{(k)} \sim \mathcal{U}(-1, 1).$$

A short smoothing kernel $h = [0.25, 0.5, 0.25]$ is applied to preserve irregular local variations while avoiding unrealistically sharp discontinuities:

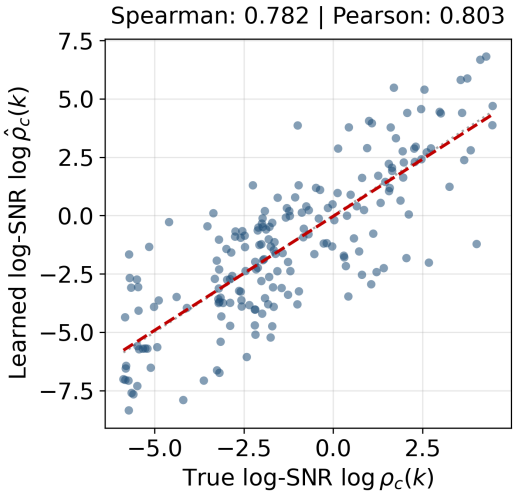


Figure 9: Synthetic log-SNR $\log \rho_c(k)$ and learned $\log \hat{\rho}_c(k)$ Correlation. The synthetic log-SNR is computed from the known clean signal and injected noise.

$$j_t = \left(h * \sum_{k=1}^{K_j} r_k a_k \epsilon_{t-\eta_k}^{(k)} \mathbf{1}(\eta_k \le t < \eta_k + \ell_k) \right)_t . \tag{72}$$

Clean Multivariate Signal and Noise. The three transient types are combined with the same backbone to generate three channels,

$$y_t^{(1)} = b_t + s_t, \qquad y_t^{(2)} = b_t + p_t, \qquad y_t^{(3)} = b_t + j_t. \tag{73}$$

Each clean channel is normalized by its standard deviation to ensure comparable scales despite different transient amplitudes. Finally, independent white Gaussian noise,

$$\varepsilon_t^{(c)} \sim \mathcal{N}(0, \sigma_\varepsilon^2),$$

is added to obtain the noisy observations,

$$\hat{y}_t = y_t + \varepsilon_t, \tag{74}$$

while the normalized clean signals are retained as the forecasting targets.

H.2 Synthetic Validation of the Learned Wiener Spectrum

To further validate our signal-preserving protection filter, we conduct additional analyses on the synthetic dataset, where the clean signal and injected noise are both available. This enables us to examine the qualitative filtering behaviour.

Across the three cases in Figure 8, WiSPformer consistently preserves the informative spectral components that are critical for forecasting while suppressing noise-dominated frequencies. To quantitatively examine whether the learned log-SNR parameters follow the Wiener principle, we compute the frequency-wise SNR using the ground-truth clean signal and injected noise. We therefore compare the ground-truth log-SNR, $\log \rho_c(k)$, with the learned spectral logits, $\log \hat{\rho}_c(k)$.

Figure 9 shows the comparison over all channel-frequency pairs. The learned spectral logits exhibit a strong positive correlation with the ground-truth log-SNR and achieve a Pearson correlation of 0.803 and a Spearman rank correlation of 0.782. This result quantitatively confirms the qualitative

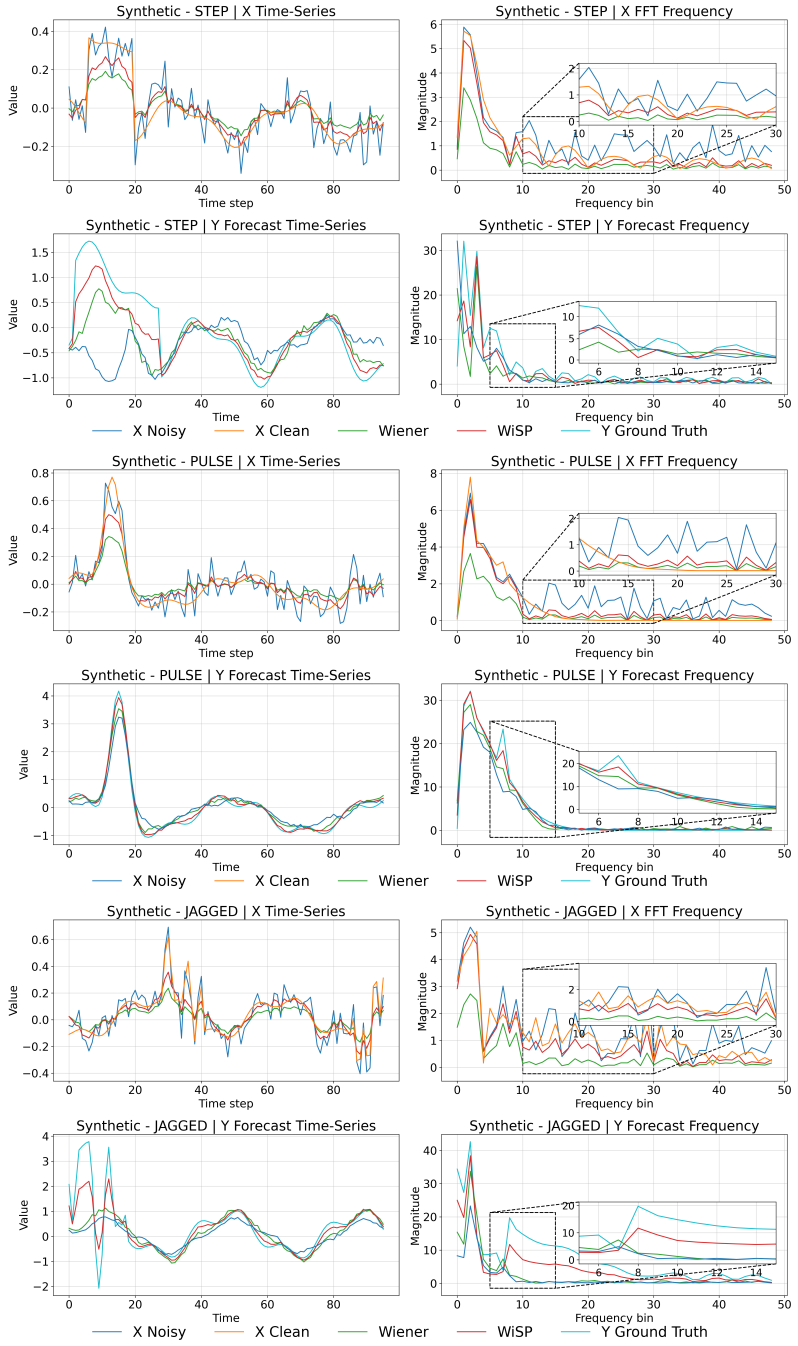


Figure 10: Comparison on synthetic Step, Jagged, and Pulse signals. Each example shows the input time series, input frequency spectrum, forecasted output time series, and forecasted output frequency spectrum. The dashed boxes and zoomed-in views highlight the high-frequency regions where different methods exhibit different spectral behaviors.

observations in Figure 10. Frequency bins with higher SNR tend to receive larger learned spectral logits, while lower-SNR bins generally receive smaller values. Figure 10 shows that WiSPformer retains more signal-related spectral components than the Wiener ones (red vs. green lines), while lower-SNR, noise-dominated frequencies receive smaller logits and remain more strongly attenuated.

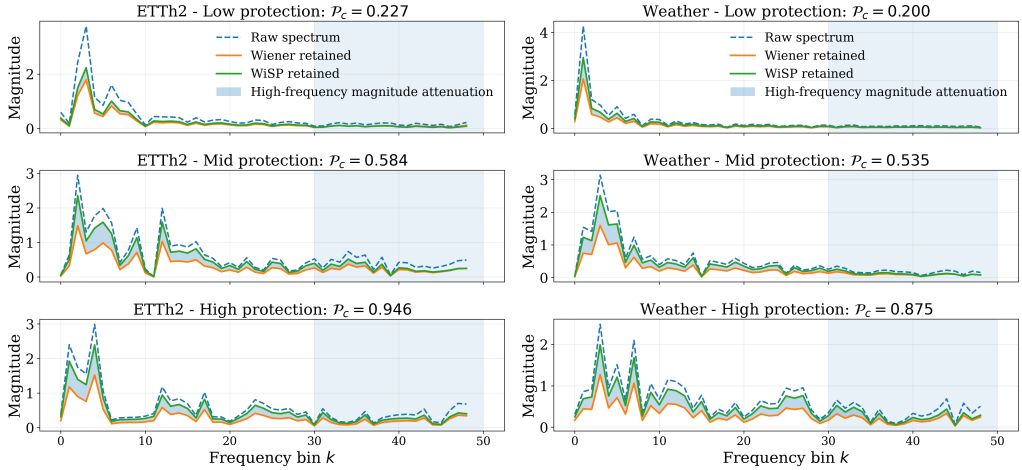


Figure 11: Comparison of signal-preserving protection under different protection scores. It shows the **Raw** spectral magnitude, spectrum filtered by our learned **Wiener** filter, and the spectrum retained by our full **WiSP**former. The shaded region highlights high-frequency magnitude attenuation.

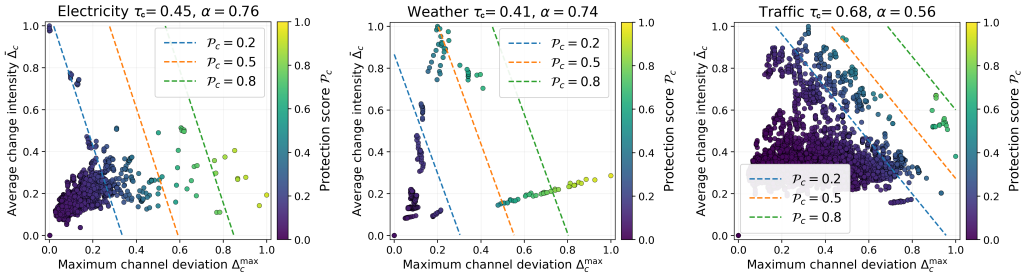


Figure 12: Relationship between $\Delta_c^{\max}$, $\bar{\Delta}_c$, and protection score P_c

Overall, these qualitative and quantitative results demonstrate that the proposed spectral weighting is not an arbitrary frequency transformation, but learns a meaningful approximation of the Wiener filtering principle through the forecasting objective.

I Signal-Preserving Learnable Wiener Filter Studies

Figure 11 illustrates how the signal-preserving mechanism affects spectral retention. For channels with higher protection scores, WiSP retains more spectral magnitude than the Wiener filter, as reflected by the larger grep area between the yellow and green lines. This indicates that the protection mechanism reduces over-attenuation and preserves more transient-related spectral information. Meanwhile, in the high-frequency region, the WiSP-retained spectrum remains below the raw spectrum, showing that WiSP adaptively balances transient preservation and spectral denoising by retaining informative components while still suppressing part of the high-frequency magnitude.

Figure 12 shows how the protection score P_c changes with the maximum channel deviation $\Delta_c^{\max}$ and the average change intensity $\bar{\Delta}_c$ on the datasets of Traffic, Electricity and Weather. Across the three datasets, channels with small $\Delta_c^{\max}$ and small $\bar{\Delta}_c$ generally receive low protection scores. When either the peak deviation or the average change intensity increases, the protection score tends to rise. Channels with simultaneously large $\Delta_c^{\max}$ and large $\bar{\Delta}_c$ usually obtain the highest protection scores. It shows that the proposed mechanism captures both peak excursions and persistent local variations.

In Electricity, several channels reach very high $\bar{\Delta}_c$ values, around 0.8 to 1.0, while their $\Delta_c^{\max}$ remains close to 0. Their protection scores are still low. Particularly, the points with $\bar{\Delta}_c \approx 1.0$ receive a lower protection score than the points with $\bar{\Delta}_c \approx 0.8$ because the latter points have $\Delta_c^{\max} \approx 0.2$, whereas the formers only have $\Delta_c^{\max} \approx 0$. In Weather, channels with larger deviations and stronger average

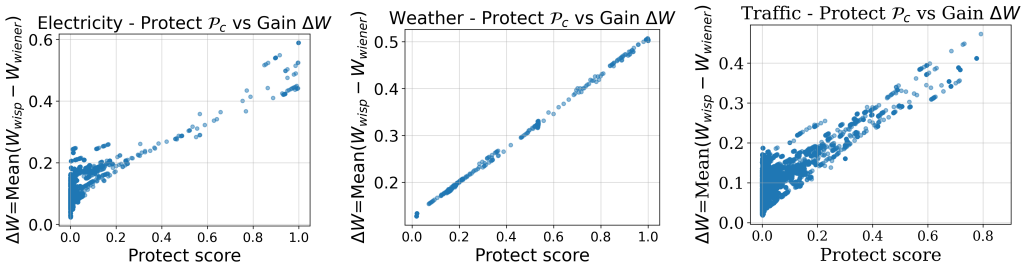
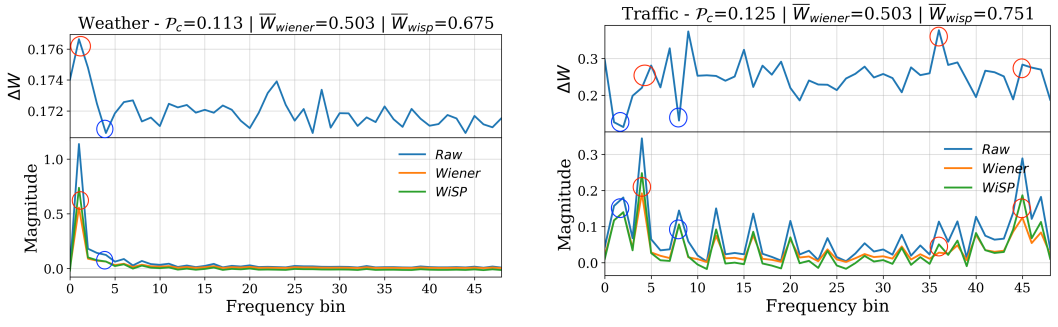
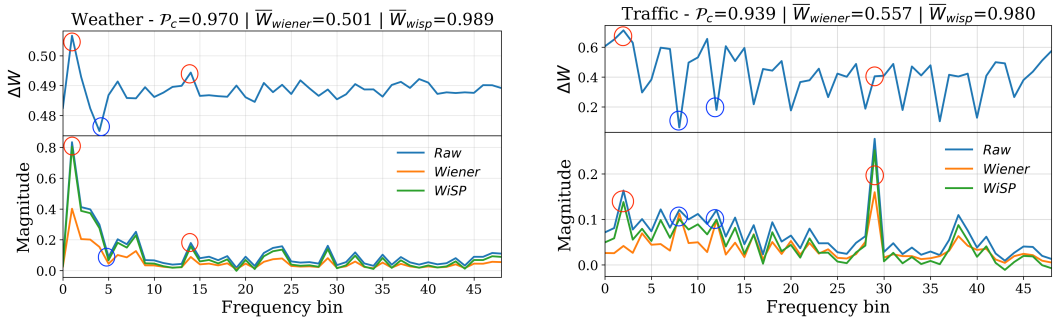


Figure 13: Relationship between $\Delta W = Mean(W_{wisp} - W_{wiener})$ and protection score P_c



(a) Weather (Low Protection $\mathcal{P}_c = 0.113$)

(b) Traffic (Low Protection $\mathcal{P}_c = 0.125$)



(c) Weather (High Protection $\mathcal{P}_c = 0.970$)

(d) Traffic (Low Protection $\mathcal{P}_c = 0.939$)

Figure 14: Examples of the gain adjustment ΔW to shape the retained frequency spectrum. The upper panel plots $\Delta W = W_{\text{wisp}} - W_{\text{wiener}}$, and the lower panel compares the Raw, Wiener-filtered, and WiSP-retained spectra.

changes are assigned higher protection scores. In Traffic, many channels have moderate-to-large $\bar{\Delta}_c$, but high protection scores mainly appear when $\Delta_c^{\max}$ is also relatively large. Overall, the three figures demonstrate that the protection score is not determined by a single statistic, but by the combined effect of peak deviation and sustained local variation.

Figure 14 represents low- and high-protection cases from Weather and Traffic to show the gain adjustment $\Delta W = W_{\text{wisp}} - W_{\text{wiener}}$ shapes the retained spectrum. In all four examples, the upper panel shows the gain difference ΔW across frequency bins. It demonstrates that the gain lift introduced by WiSP, accompanied by the protection score $\mathcal{P}_c$. In (a) and (b) where $\mathcal{P}_c$ is relatively low, the corresponding ΔW remains small. In contrast, in (c) and (d), higher protection scores lead to substantially larger ΔW and $\overline{W}_{\text{wisp}}$ is much closer to 1. The lower panels further show how the gain adjustment affects spectral retention. In the low-protection cases, WiSP raises the retained spectrum slightly above the Wiener baseline. In contrast, under high protection, WiSP preserves stronger spectral components. The red circles highlight that the gain lift produces a clear increase in retained magnitude, and makes the WiSP spectrum visibly closer to the raw signal than the Wiener-filtered one. By comparison, the blue circles mark shows the smaller gain lift, where the spectra under Wiener

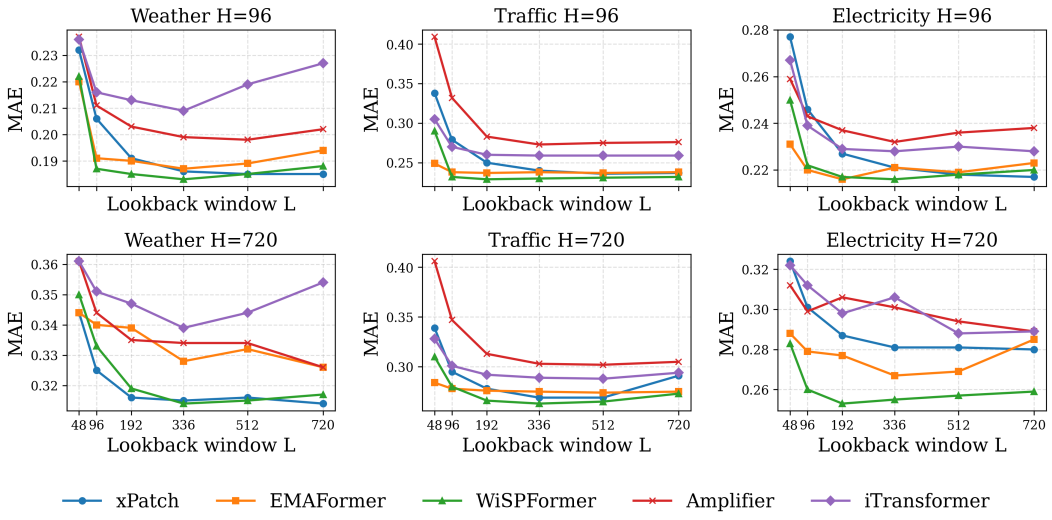


Figure 15: Forecasting performance (MAE), lookback window L = 48, 96, 192, 336, 512, 720 and prediction horizon H = 96 and 720.

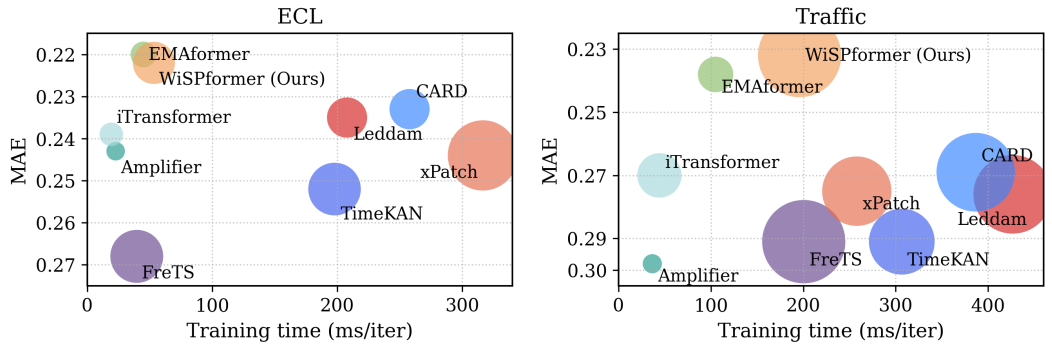


Figure 16: Comparison of model efficiency. Lookback and prediction lengths are both 96, with a fixed batch size of 32. Bubble area is proportional to training GPU memory usage. For improved visual presentation, bubbles within each subfigure are scaled by a uniform factor.

filtering and WiSP remain much closer and may even nearly overlap. Overall, low protection scores usually lead to smaller magnitude gaps between Wiener and WiSP, whereas higher protection scores produce larger recovery, although the final effect also depends on the strength of the raw spectral component itself.

J LOOKBACK LENGTH

In theory, a longer lookback window L should lead to better forecasting accuracy. This is because a longer sequence improves frequency resolution due to the inverse proportionality to sequence length (specifically $1/L$) (Palani, 2021). Figure 15 illustrates the MAE changes under different lookback lengths. Most models improve the forecasting when L increases from 48 to 512. However, the performance slightly degrades when L is at 720. This suggests that very long input sequences may introduce redundant or noisy information. WiSPformer benefits from signal-preserving protection and achieves stable performance across different lookback windows. It consistently obtains lower MAE, especially on Traffic and Electricity.

K Efficiency Analysis

The complexity of WiSPformer is $\mathcal{O}(NL + N^2)$, where N is the number of channels and L is the look-back window length. The $\mathcal{O}(NL)$ comes from the signal-preserving Wiener filtering and protection operations, while the $\mathcal{O}(N^2)$ term comes from the dual-branch spectral Transformer. Figure 16 compares the practical efficiency of WiSPformer with forecasting performance on the ECL and Traffic datasets, using training time per iteration and MAE. On ECL, WiSPformer attains one of the lowest MAE values while requiring substantially less training time than heavier baselines such as xPatch, CARD, Leddam, and TimeKAN. On Traffic, WiSPformer again achieves the best MAE among the compared methods, while maintaining a moderate training cost. Overall, WiSPformer achieves a good balance between accuracy and efficiency.

L Full Experimental Results

L.1 Complete Results for Table 1 of Main Paper

We compare WiSPformer with 12 state-of-the-art baselines across 9 dataset to evaluate long-range forecasting for multivariate time-series. For a fair comparison, we evaluate WiSPformer and baselines under unified experimental settings of each task, such as the unified lookback length $L = 96$ and the predict length $H \in \{96, 192.336.720\}$. We ran these baselines using their publicly available official implementation and hyperparameter settings. The average results from all prediction lengths are presented in Table 1 of the main paper, and Table 6 lists the full results of long-range time-series forecasting, where the best results are Red and the second best results are Blue.

L.2 Complete Results for Table 2 of Main Paper

Transferability of WiSP across forecasting backbones across 8 datasets. Table 7 lists the full results that compare each backbone model with its WiSP-enhanced one. Lower MSE and MAE indicate better performance.

M Limitations and Future Work

Although WiSP demonstrates promising performance on multivariate time-series forecasting benchmarks, its current formulation is mainly designed for regularly sampled time-series where temporal patterns can be effectively represented in the frequency domain. Its applicability to irregularly sampled data, streaming scenarios, or tasks with substantially different temporal structures remains to be further investigated.

In addition, WiSP currently focuses on forecasting under standard benchmark settings. Extending the proposed spectral protection and dependency modelling ideas to other time-series analysis tasks, such as imputation, anomaly detection, and classification, is a promising direction for future work. Further improving scalability to extremely long sequences and large-scale channel dimensions may also broaden its practical applicability.

N More Showcases (Fig. 18, Fig. 17)

Models		WiSPFormer [Ours]		EMAFormer [2026]		Amplifier [2025]		xPatch [2025]		Alio [2025]		FilterNet [2024]		Leddam [2024]		FreDF [2025]		FreEformer [2025]		AdaMSHyp [2024]		Fredformer [2024]		iTrans. [2024a]		CARD [2024]	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	0.303	0.335	0.305	0.336	0.316	0.355	0.311	0.346	0.330	0.361	0.318	0.358	0.319	0.359	0.324	0.362	0.306	0.340	0.305	0.354	0.326	0.361	0.334	0.368	0.316	0.347
	192	0.351	0.365	0.359	0.365	0.361	0.381	0.348	0.368	0.375	0.383	0.364	0.383	0.369	0.383	0.373	0.385	0.359	0.367	0.356	0.375	0.363	0.380	0.377	0.391	0.363	0.370
	336	0.392	0.390	0.394	0.388	0.393	0.402	0.388	0.391	0.408	0.406	0.396	0.406	0.394	0.402	0.402	0.404	0.392	0.391	0.387	0.398	0.395	0.403	0.426	0.420	0.392	0.390
	720	0.453	0.426	0.467	0.431	0.455	0.440	0.461	0.430	0.472	0.441	0.456	0.444	0.460	0.442	0.469	0.444	0.458	0.428	0.457	0.437	0.453	0.438	0.491	0.459	0.458	0.435
	Avg	0.375	0.379	0.381	0.380	0.381	0.394	0.377	0.384	0.396	0.398	0.384	0.398	0.386	0.397	0.392	0.399	0.379	0.381	0.378	0.391	0.384	0.395	0.407	0.410	0.382	0.383
ETTm2	96	0.160	0.244	0.167	0.246	0.176	0.258	0.164	0.248	0.184	0.267	0.174	0.257	0.176	0.257	0.173	0.252	0.169	0.248	0.175	0.257	0.177	0.259	0.180	0.264	0.169	0.248
	192	0.225	0.286	0.233	0.290	0.239	0.300	0.230	0.291	0.250	0.310	0.240	0.300	0.243	0.303	0.241	0.298	0.233	0.289	0.231	0.309	0.243	0.301	0.250	0.309	0.234	0.292
	336	0.291	0.328	0.293	0.329	0.297	0.338	0.292	0.331	0.312	0.348	0.297	0.339	0.303	0.341	0.298	0.334	0.292	0.328	0.293	0.332	0.302	0.340	0.311	0.348	0.294	0.339
	720	0.381	0.384	0.390	0.388	0.393	0.396	0.381	0.383	0.411	0.404	0.392	0.393	0.400	0.398	0.398	0.393	0.395	0.389	0.395	0.397	0.397	0.396	0.412	0.407	0.390	0.388
	Avg	0.264	0.310	0.270	0.313	0.276	0.323	0.267	0.313	0.289	0.332	0.276	0.322	0.281	0.325	0.278	0.319	0.272	0.313	0.274	0.324	0.279	0.324	0.288	0.332	0.272	0.317
ETTh1	96	0.371	0.385	0.375	0.390	0.371	0.392	0.376	0.386	0.380	0.397	0.375	0.394	0.377	0.394	0.382	0.400	0.373	0.390	0.372	0.393	0.373	0.392	0.386	0.405	0.383	0.391
	192	0.427	0.417	0.428	0.419	0.426	0.422	0.417	0.407	0.431	0.428	0.436	0.422	0.424	0.422	0.430	0.427	0.424	0.420	0.433	0.417	0.433	0.420	0.441	0.436	0.435	0.420
	336	0.448	0.428	0.469	0.439	0.448	0.434	0.449	0.425	0.471	0.445	0.476	0.443	0.459	0.442	0.474	0.451	0.466	0.443	0.422	0.433	0.470	0.437	0.487	0.458	0.479	0.442
	720	0.463	0.453	0.457	0.451	0.476	0.464	0.470	0.456	0.471	0.469	0.474	0.469	0.463	0.459	0.463	0.462	0.471	0.470	0.445	0.459	0.467	0.456	0.503	0.491	0.471	0.461
	Avg	0.427	0.421	0.432	0.425	0.430	0.428	0.428	0.419	0.438	0.435	0.440	0.432	0.431	0.429	0.437	0.435	0.433	0.431	0.418	0.426	0.435	0.426	0.454	0.448	0.442	0.429
ETTh2	96	0.283	0.330	0.291	0.336	0.279	0.337	0.233	0.300	0.300	0.348	0.292	0.343	0.292	0.343	0.289	0.337	0.286	0.335	0.283	0.332	0.293	0.342	0.297	0.349	0.281	0.330
	192	0.357	0.375	0.367	0.385	0.359	0.389	0.291	0.338	0.375	0.396	0.369	0.395	0.367	0.389	0.363	0.385	0.363	0.381	0.358	0.374	0.371	0.389	0.380	0.400	0.363	0.381
	336	0.366	0.403	0.415	0.422	0.377	0.406	0.344	0.377	0.413	0.427	0.420	0.432	0.412	0.424	0.419	0.426	0.416	0.420	0.428	0.437	0.382	0.409	0.428	0.432	0.411	0.418
	720	0.415	0.431	0.424	0.439	0.420	0.432	0.407	0.427	0.419	0.441	0.430	0.446	0.419	0.438	0.415	0.437	0.422	0.436	0.413	0.432	0.415	0.434	0.427	0.445	0.416	0.431
	Avg	0.355	0.384	0.374	0.396	0.359	0.391	0.319	0.361	0.377	0.403	0.378	0.404	0.373	0.399	0.371	0.396	0.372	0.393	0.371	0.385	0.365	0.393	0.383	0.407	0.368	0.390
Weather	96	0.148	0.186	0.154	0.190	0.156	0.204	0.168	0.203	0.175	0.215	0.162	0.207	0.156	0.202	0.164	0.202	0.153	0.189	0.157	0.195	0.163	0.207	0.174	0.214	0.150	0.188
	192	0.197	0.232	0.204	0.239	0.209	0.249	0.214	0.245	0.223	0.257	0.210	0.250	0.207	0.250	0.220	0.253	0.201	0.236	0.218	0.259	0.211	0.251	0.221	0.254	0.202	0.238
	336	0.238	0.278	0.261	0.282	0.264	0.290	0.236	0.273	0.281	0.298	0.265	0.290	0.262	0.291	0.275	0.294	0.261	0.282	0.251	0.282	0.267	0.292	0.278	0.296	0.260	0.282
	720	0.316	0.323	0.343	0.337	0.343	0.342	0.309	0.321	0.357	0.347	0.342	0.340	0.343	0.343	0.356	0.347	0.341	0.334	0.304	0.328	0.343	0.341	0.358	0.349	0.343	0.353
	Avg	0.225	0.255	0.240	0.262	0.243	0.271	0.232	0.261	0.259	0.279	0.245	0.272	0.242	0.272	0.254	0.274	0.239	0.260	0.233	0.266	0.246	0.272	0.258	0.278	0.239	0.265
Solar	96	0.161	0.190	0.169	0.200	0.235	0.274	0.201	0.215	0.192	0.217	0.212	0.246	0.197	0.241	0.193	0.224	0.180	0.191	0.187	0.234	0.185	0.233	0.203	0.237	0.194	0.209
	192	0.185	0.210	0.189	0.220	0.270	0.296	0.235	0.234	0.225	0.241	0.274	0.294	0.231	0.264	0.226	0.253	0.213	0.215	0.224	0.245	0.227	0.253	0.233	0.261	0.234	0.235
	336	0.203	0.225	0.224	0.232	0.298	0.312	0.258	0.249	0.242	0.257	0.289	0.305	0.241	0.268	0.239	0.264	0.233	0.232	0.244	0.268	0.246	0.284	0.248	0.273	0.256	0.253
	720	0.213	0.228	0.209	0.243	0.298	0.262	0.260	0.247	0.247	0.261	0.278	0.307	0.250	0.281	0.241	0.265	0.241	0.237	0.245	0.264	0.247	0.276	0.249	0.275	0.262	0.257
	Avg	0.191	0.214	0.198	0.224	0.275	0.298	0.239	0.236	0.227	0.244	0.263	0.288	0.230	0.264	0.225	0.252	0.217	0.219	0.225	0.253	0.226	0.262	0.233	0.262	0.237	0.239
Electricity	96	0.128	0.220	0.130	0.220	0.147	0.243	0.159	0.244	0.147	0.238	0.147	0.245	0.141	0.235	0.144	0.233	0.133	0.223	0.135	0.238	0.147	0.241	0.148	0.240	0.141	0.233
	192	0.146	0.236	0.150	0.238	0.157	0.251	0.160	0.248	0.160	0.250	0.160	0.250	0.159	0.252	0.159	0.247	0.152	0.240	0.152	0.239	0.165	0.258	0.162	0.253	0.160	0.250
	336	0.166	0.255	0.166	0.255	0.174	0.269	0.182	0.267	0.175	0.267	0.173	0.267	0.173	0.268	0.172	0.263	0.165	0.256	0.168	0.266	0.177	0.273	0.178	0.269	0.173	0.263
	720	0.185	0.277	0.188	0.274	0.206	0.296	0.216	0.298	0.207	0.294	0.210	0.309	0.201	0.295	0.204	0.294	0.198	0.286	0.212	0.293	0.213	0.304	0.225	0.317	0.197	0.284
	Avg	0.156	0.247	0.158	0.247	0.171	0.265	0.179	0.264	0.172	0.262	0.172	0.268	0.169	0.263	0.170	0.259	0.162	0.251	0.167	0.252	0.176	0.269	0.178	0.270	0.168	0.258
Traffic	96	0.387	0.231	0.400	0.238	0.455	0.298	0.471	0.275	0.393	0.264	0.430	0.294	0.426	0.276	0.391	0.265	0.395	0.234	0.395	0.258	0.406	0.277	0.395	0.268	0.419	0.269
	192	0.408	0.245	0.418	0.249	0.470	0.316	0.478	0.272	0.412	0.271	0.452	0.307	0.458	0.289	0.410	0.273	0.423	0.245	0.414	0.259	0.426	0.290	0.417	0.276	0.443	0.276
	336	0.421	0.254	0.436	0.256	0.479	0.316	0.501	0.276	0.428	0.278	0.470	0.316	0.486	0.297	0.424	0.280	0.443	0.255	0.434	0.274	0.437	0.292	0.433	0.283	0.460	0.283
	720	0.460	0.267	0.472	0.276	0.523	0.328	0.547	0.291	0.460	0.296	0.498	0.323	0.498	0.313	0.460	0.298	0.480	0.274	0.467	0.292	0.462	0.305	0.467	0.302	0.490	0.299
	Avg	0.419	0.249	0.431	0.282	0.482	0.315	0.499	0.279	0.423	0.255	0.463	0.310	0.467	0.294	0.421	0.279	0.435	0.251	0.428	0.271	0.433	0.291	0.428	0.282	0.453	0.282
Exchange	96	0.079	0.199	0.083	0.203	0.083	0.202	0.082	0.199	0.086	0.205	0.083	0.202	0.086	0.207	0.084	0.203	0.083	0.200	0.086	0.204	0.084	0.202	0.086	0.206	0.084	0.202
	192	0.163	0.289	0.175	0.298	0.175	0.297	0.177	0.298	0.178	0.300	0.174	0.296	0.175	0.301	0.176	0.297	0.174	0.296	0.177	0.298	0.178	0.302	0.177	0.299	0.174	0.295
	336	0.295	0.397	0.336	0.419	0.328	0.414	0.349	0.425	0.331	0.417	0.326	0.413	0.325	0.415	0.329	0.415	0.325	0.411	0.322	0.406	0.319	0.408	0.331	0.417	0.342	0.421
	720	0.691	0.627	0.826	0.688	0.858	0.696	0.891	0.711	0.857	0.699	0.840	0.670	0.831	0.686	0.846	0.689	0.833	0.687	0.827	0.678	0.749	0.651	0.847	0.691	0.841	0.689
	Avg	0.307	0.378	0.355	0.402	0.361	0.402	0.375	0.408	0.363	0.405	0.356	0.395	0.354	0.402	0.359	0.401	0.354	0.399	0.353	0.397	0.333	0.391	0.360	0.403	0.360	0.402
1st Count		31	32	0	7	1	0	10	12	1	0	0	0	0	0	1	0	1	2	4	0	0	0	0	0	0	0

Table 6: Full long-term multivariate time-series forecasting results with unified lookback window $L = 96$ for all datasets. The best method is Red and the second best is Blue. Average results are reported across four prediction lengths.

Methods		FreTS		+WiSP		TimeKAN		+WiSP		iTrans.		+WiSP	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	0.340	0.375	**0.322**	**0.350**	0.332	0.367	**0.320**	**0.366**	0.343	0.375	**0.334**	**0.369**
	192	0.383	0.382	**0.373**	**0.373**	0.367	0.384	**0.358**	**0.383**	0.383	0.393	**0.376**	**0.390**
	336	0.417	0.419	**0.408**	**0.395**	0.387	0.404	**0.386**	**0.402**	0.418	0.416	**0.410**	**0.412**
	720	0.483	0.458	**0.466**	**0.429**	0.447	0.440	**0.440**	**0.437**	0.481	0.451	0.481	0.451
	Avg	0.406	0.409	**0.392**	**0.387**	0.383	0.399	**0.377**	**0.397**	0.407	0.410	**0.400**	**0.405**
ETTm2	96	0.176	0.262	**0.174**	**0.255**	0.177	0.258	**0.173**	**0.250**	0.185	0.268	**0.183**	**0.267**
	192	0.241	0.304	**0.240**	**0.300**	0.243	0.303	**0.236**	**0.293**	0.256	0.314	**0.254**	0.314
	336	0.304	0.346	**0.299**	**0.339**	0.309	0.349	**0.296**	**0.333**	0.315	0.352	0.315	0.352
	720	0.399	0.400	**0.393**	**0.396**	0.418	0.422	**0.392**	**0.390**	**0.412**	**0.405**	0.413	0.406
	Avg	0.280	0.328	**0.277**	**0.323**	0.287	0.333	**0.275**	**0.317**	0.292	0.335	**0.291**	**0.334**
ETTh1	96	0.392	0.398	**0.385**	**0.392**	0.375	0.392	**0.367**	**0.387**	0.385	0.404	**0.383**	**0.403**
	192	0.438	0.426	0.438	**0.422**	0.424	0.421	**0.413**	**0.416**	0.435	0.432	**0.433**	**0.431**
	336	0.500	0.460	**0.484**	**0.445**	0.460	0.441	**0.446**	**0.436**	0.482	0.456	**0.478**	**0.454**
	720	0.548	0.509	**0.505**	**0.474**	0.491	0.472	**0.464**	**0.466**	0.483	0.478	**0.479**	**0.473**
	Avg	0.470	0.448	**0.453**	**0.430**	0.437	0.431	**0.423**	**0.426**	0.454	0.448	**0.443**	**0.439**
ETTh2	96	0.292	0.339	**0.285**	**0.332**	0.300	0.347	**0.287**	**0.336**	0.297	0.349	**0.292**	**0.346**
	192	0.369	0.388	**0.360**	**0.380**	0.378	0.394	**0.362**	**0.381**	0.380	0.400	**0.374**	**0.396**
	336	0.423	0.426	**0.411**	**0.418**	0.444	0.447	**0.425**	**0.433**	0.428	0.432	**0.421**	**0.428**
	720	0.451	0.561	**0.425**	**0.437**	0.449	0.456	**0.430**	**0.441**	0.427	0.445	**0.420**	**0.441**
	Avg	0.384	0.429	**0.369**	**0.391**	0.393	0.411	**0.376**	**0.398**	0.383	0.407	**0.377**	**0.403**
Electricity	96	0.176	0.268	**0.156**	**0.248**	0.175	0.252	**0.154**	**0.242**	0.148	0.239	**0.138**	**0.233**
	192	0.184	0.275	**0.168**	**0.259**	0.180	0.260	**0.168**	**0.256**	0.162	0.252	**0.155**	**0.249**
	336	0.198	0.287	**0.185**	**0.276**	0.194	0.277	**0.185**	**0.275**	0.175	0.269	**0.171**	**0.266**
	720	0.241	0.322	**0.224**	**0.309**	0.231	0.310	**0.229**	**0.307**	0.228	0.312	**0.218**	**0.307**
	Avg	0.200	0.288	**0.183**	**0.273**	0.195	0.275	**0.184**	**0.270**	0.178	0.270	**0.171**	**0.263**
Solar	96	0.259	0.264	**0.230**	**0.253**	0.226	0.243	**0.214**	**0.229**	0.216	0.256	**0.205**	**0.248**
	192	0.273	0.280	**0.264**	**0.268**	0.245	0.257	**0.232**	**0.242**	0.249	0.281	**0.237**	**0.273**
	336	0.303	0.289	**0.283**	**0.273**	0.261	0.265	**0.257**	**0.259**	0.272	0.298	**0.259**	**0.290**
	720	0.305	0.285	**0.289**	**0.271**	0.278	0.307	**0.273**	**0.289**	0.276	**0.302**	0.276	0.303
	Avg	0.285	0.280	**0.266**	**0.266**	0.253	0.268	**0.244**	**0.255**	0.253	0.284	**0.244**	**0.278**
Weather	96	0.168	0.229	**0.163**	**0.199**	0.164	0.211	**0.159**	**0.204**	0.176	0.216	**0.167**	**0.211**
	192	0.210	0.269	**0.206**	**0.239**	0.207	0.249	**0.202**	**0.246**	0.227	0.260	**0.219**	**0.257**
	336	0.264	0.310	**0.263**	**0.282**	0.264	0.291	**0.250**	**0.282**	0.280	0.300	**0.276**	**0.299**
	720	0.347	0.367	**0.341**	**0.334**	0.340	0.341	**0.320**	**0.331**	0.359	0.350	**0.351**	**0.349**
	Avg	0.247	0.294	**0.244**	**0.264**	0.244	0.273	**0.232**	**0.266**	0.260	0.282	**0.253**	**0.279**
Traffic	96	0.519	0.291	**0.488**	**0.281**	0.519	0.291	**0.488**	**0.281**	0.397	0.270	**0.395**	0.270
	192	0.521	0.293	**0.492**	**0.284**	0.521	0.293	**0.492**	**0.284**	0.418	0.279	**0.417**	**0.278**
	336	0.545	0.295	**0.507**	**0.291**	0.545	0.295	**0.507**	**0.291**	0.433	0.285	**0.430**	**0.284**
	720	0.583	0.314	**0.541**	**0.307**	0.583	0.314	**0.541**	**0.307**	0.470	0.303	**0.462**	**0.301**
	Avg	0.542	0.298	**0.507**	**0.291**	0.431	0.286	**0.423**	**0.279**	0.429	0.284	**0.426**	**0.283**

Table 7: Transferability of WiSP across forecasting backbones across eight datasets. Results compare each backbone model with its WiSP-enhanced one. Lower MSE and MAE indicate better performance.

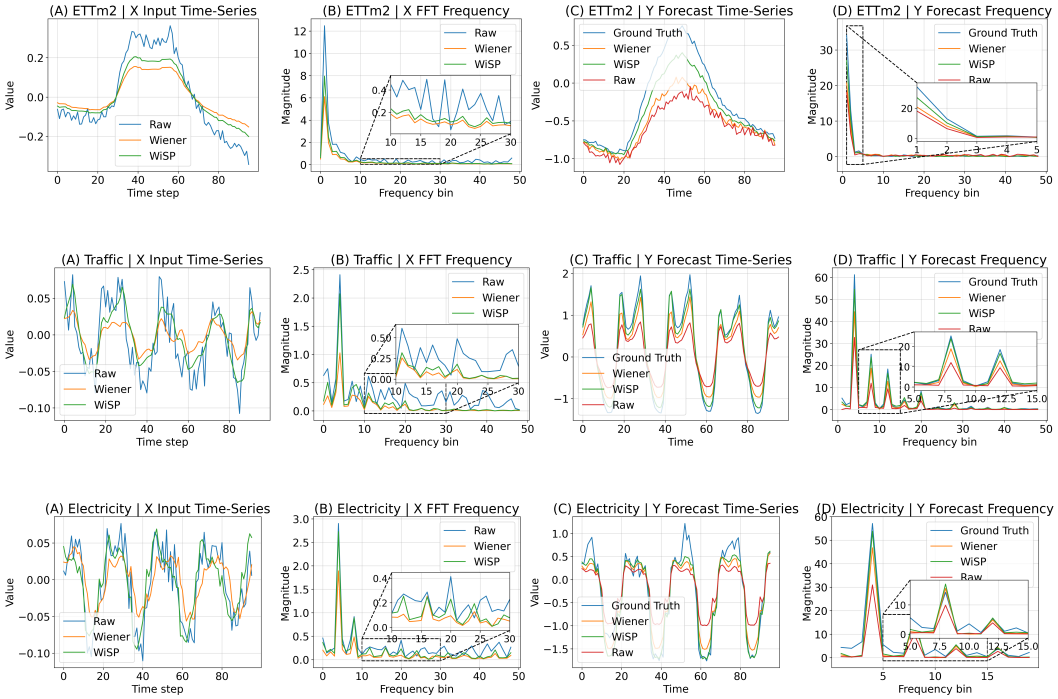


Figure 17: Showcase: Signal-preserving retains informative local transients in WiSPformer. 'Raw' is for the original input signal, 'Wiener' for the signal after Wiener filtering, and 'WiSP' for the signal after WiSPformer.

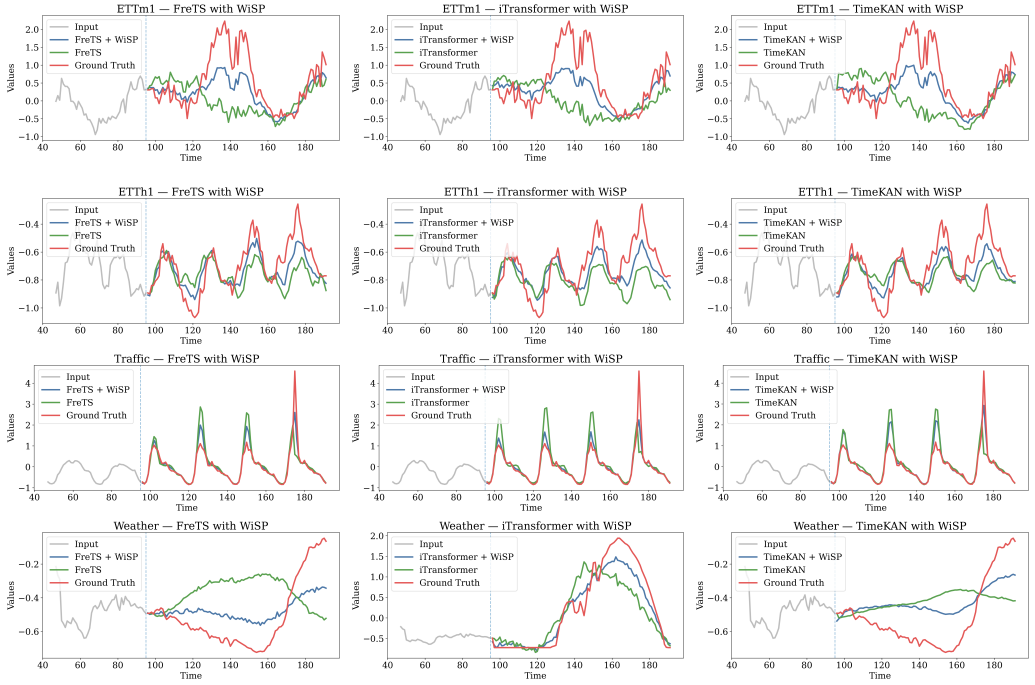


Figure 18: Showcase: Prediction visualisation for FreTS, iTransformer, and TimeKAN with and without WiSP on ETTm1, ETTh1, Traffic and Weather datasets (96 → 96).