AMPTS: AMPLITUDE-GUIDED PROGRESSIVE RESTORATION METHOD FOR TIME-SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Time-series forecasting (TSF) plays an important role in a wide range of real-world applications. Existing TSF methods typically either generate the complete future sequence through a single direct mapping or formulate forecasting as a noise-conditioned generative task that progressively recovers the future through multiple denoising stages. Although the latter provides a progressive forecasting perspective, diffusion-based approaches generally rely on stochastic perturbations to construct intermediate states, with their progressive processes constrained by predefined noise-evolution paths. To address this limitation, we propose **AmpTS**, an amplitude-guided progressive restoration method that uses structure-preserving reference amplitude levels to organize iterative transitions among self-generated restoration states. Starting from a zero state, AmpTS repeatedly re-estimates the complete future sequence at each restoration stage by conditioning on historical observations and the current state, enabling multi-stage progressive forecasting without relying on stochastic perturbations to construct restoration states. Extensive experiments on real-world datasets show that AmpTS achieves competitive and consistent performance against representative direct forecasting and diffusion-based forecasting methods, demonstrating the effectiveness of amplitude-guided progressive restoration for time-series point forecasting.

1 INTRODUCTION

Time-series forecasting plays a fundamental role in a wide range of real-world applications, including energy, finance, weather, and transportation (Liu & Wang, 2024; Kong et al., 2025a; Kim et al., 2025). In recent years, substantial research efforts have focused on developing increasingly effective forecasting architectures to improve predictive performance (Qiu et al., 2025a; Qi et al., 2026). Despite these advances, most existing methods follow a similar forecasting paradigm: historical observations are first encoded and then directly mapped to the complete future sequence through a single prediction process. Consequently, prior research has largely concentrated on improving the mapping from history to future, while comparatively less attention has been paid to how the forecasting process itself should be organized. This motivates a more fundamental question: *Must the future sequence be inferred directly and entirely in a single prediction process?*

Diffusion models provide an important perspective for rethinking this one-shot forecasting paradigm. Diffusion models formulate complex target generation as a noise-conditioned progressive process, constructing states at different corruption levels through gradual noise injection and learning a reverse process to progressively recover the target (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2020). This class of methods has demonstrated strong modeling capability in image and video generation (Dhariwal & Nichol, 2021; Rombach et al., 2022; Ho et al., 2022).

This idea has subsequently been introduced into time-series forecasting and other temporal tasks, allowing future sequences to be progressively recovered through multiple restoration stages (Rasul et al., 2021; Shen & Kwok, 2023; Shen et al., 2024; Yuan & Qiao, 2024). However, existing diffusion-based forecasting methods typically rely on stochastic noise to construct intermediate states and organize progressive prediction through reverse denoising. For deterministic time-series point forecasting, such stochastic perturbations may not provide the most natural state construction,

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.

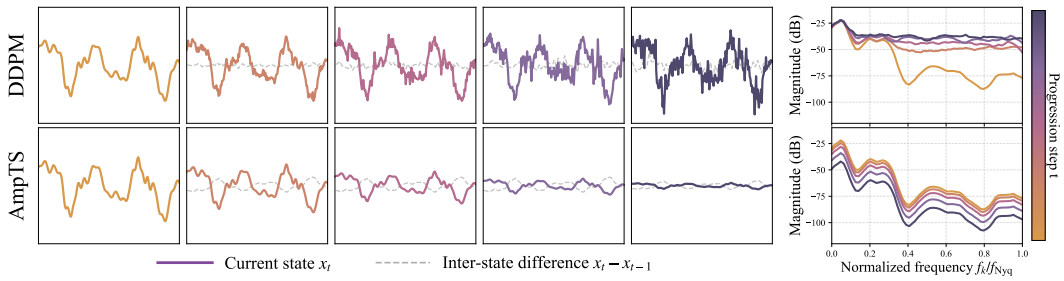


Figure 1: Comparison between Gaussian diffusion states in DDPM and the reference amplitude levels used in AmpTS. Left: time-domain comparison. Right: frequency-domain comparison.

as continuously injected noise can gradually weaken predictive temporal structures in the future sequence. This raises a more relevant question: *How can we construct a more effective progressive restoration process for time-series point forecasting beyond relying solely on noise perturbation and reverse denoising?*

To address this issue, recent time-series studies have begun to redesign state construction and transition mechanisms in diffusion processes. For example, ARMD replaces the random-noise endpoint with sequence-window shifting along the temporal axis, constructing continuous intermediate states between historical and future sequences (Gao et al., 2025). MA-TSD redesigns diffusion transitions using moving-average operators to preserve more low-frequency temporal information across different stages (Wang et al., 2025). Although these methods introduce state-evolution processes that are better tailored to time series, their progressive forecasting procedures are still built upon noise perturbations or predefined intermediate-state paths. Meanwhile, Cold Diffusion shows that progressive restoration does not inherently depend on stochastic noise, but can instead be formulated through more general deterministic state transformations and iterative recovery (Bansal et al., 2023). This further motivates us to ask: *For time-series point forecasting, can restoration states be recursively generated by the model itself, rather than relying on predefined intermediate target states, and progressively improve the complete-future prediction through iterative state transitions?*

To this end, we propose **AmpTS**, an amplitude-guided progressive restoration method for deterministic time-series point forecasting. AmpTS defines a set of structure-preserving reference restoration levels through amplitude scaling and uses them to guide transitions between self-generated restoration states. Unlike autoregressive forecasting along the temporal axis, AmpTS performs iterative inference over an independent restoration-state dimension while predicting the entire forecasting horizon at each stage.

- We revisit time-series point forecasting from the perspective of the forecasting process and formulate complete-future prediction as a progressive restoration process. Unlike one-shot forecasting and restoration along predefined intermediate-state paths, we explore a progressive forecasting paradigm driven by self-generated restoration states, allowing the complete future sequence to be repeatedly estimated and progressively refined across multiple restoration stages.

- We propose AmpTS, an amplitude-guided progressive restoration method for time-series forecasting. AmpTS uses structure-preserving reference amplitude levels to organize the iterative updates of self-generated restoration states, enabling the model to repeatedly estimate and progressively refine the complete future sequence across multiple restoration stages.

- We conduct extensive experiments on multiple real-world time-series datasets. The results show that AmpTS achieves competitive and consistent performance against representative recent forecasting methods and diffusion-based time-series forecasting approaches, providing an effective alternative for organizing deterministic progressive time-series forecasting.

2 RELATED WORK

2.1 TIME-SERIES FORECASTING

Time-series forecasting aims to extract temporal dependencies and structural patterns from historical observations to infer future dynamics. Recent studies have improved forecasting performance from different perspectives, including local pattern modeling, multi-scale representation, variable dependency modeling, convolutional architectures, and high-dimensional temporal representations (Nie et al., 2023; Wu et al., 2023; Liu et al., 2024; Wang et al., 2024; Luo & Wang, 2024; Qiu et al., 2025b; Nematirad et al., 2025; Wang et al., 2026). Despite continuous advances in modeling mechanisms, these methods generally formulate forecasting as a direct mapping from historical observations to future sequences. In contrast, AmpTS focuses on the forecasting process itself and organizes complete-future prediction as a multi-stage progressive forecasting process.

2.2 DIFFUSION AND PROGRESSIVE RESTORATION FOR TIME-SERIES FORECASTING

Diffusion models decompose target generation into a continuous progressive restoration process through multiple intermediate variables at different state levels. In recent years, this idea has been widely applied to time-series forecasting. One line of research mainly improves the reverse prediction network, conditional information modeling, and denoising learning strategies (Rasul et al., 2021; Tashiro et al., 2021; Shen & Kwok, 2023; Yuan & Qiao, 2024). Some studies further exploit multi-scale or structural information in time series to guide different diffusion stages, thereby improving temporal modeling throughout the progressive restoration process (Fan et al., 2024).

Another line of research attempts to redesign the state construction and transition mechanisms of the diffusion process itself. For example, ARMD constructs continuous intermediate states between historical and future sequences by shifting sequence windows along the temporal axis (Gao et al., 2025); MA-TSD redesigns state transitions using moving-average operators to better preserve low-frequency temporal information across different stages (Wang et al., 2025); more recent work reorganizes the perturbation of different components from a frequency-domain perspective to delay the loss of important temporal structures during state evolution (Caldas et al., 2026).

Unlike these methods, AmpTS neither relies on stochastic perturbations to define restoration levels nor requires the actual restoration states to follow a predefined intermediate-state path. Instead, it uses deterministic reference amplitude levels to organize transitions among recursively self-generated restoration states, while directly re-estimating the complete future sequence at each restoration stage.

3 METHOD

3.1 REFERENCE AMPLITUDE LEVELS

Conventional time-series forecasting models typically learn a direct mapping from historical observations to future sequences and generate the complete forecast through a single forward pass. In contrast, AmpTS first defines a set of *reference amplitude levels* based on the target future sequence to characterize different restoration stages from the complete future to the zero state, while the actual restoration states are recursively generated by the model during inference.

Given the ground-truth future sequence $\mathbf{Y}_0 \in \mathbb{R}^{C\times H}$, where H denotes the prediction length and C denotes the number of variables, we define a cumulative amplitude schedule over T restoration steps as $\bar{\alpha}_t = 1 - \frac{t}{T}$. Based on this schedule, the ideal reference amplitude state at restoration level t is defined as

$$\mathbf{Y}_t = \sqrt{\bar{\alpha}_t}\mathbf{Y}_0 = \sqrt{1 - \frac{t}{T}}\mathbf{Y}_0, \qquad t \in \{0, 1, \ldots, T\}. \tag{1}$$

Each non-terminal reference state is simply an amplitude-scaled version of the complete future sequence. Therefore, within this set of ideal reference states, different restoration levels only modify the overall signal magnitude without randomly perturbing the temporal alignment, relative variation patterns, or inter-variable structure of the original future sequence. In this way, the amplitude scaling

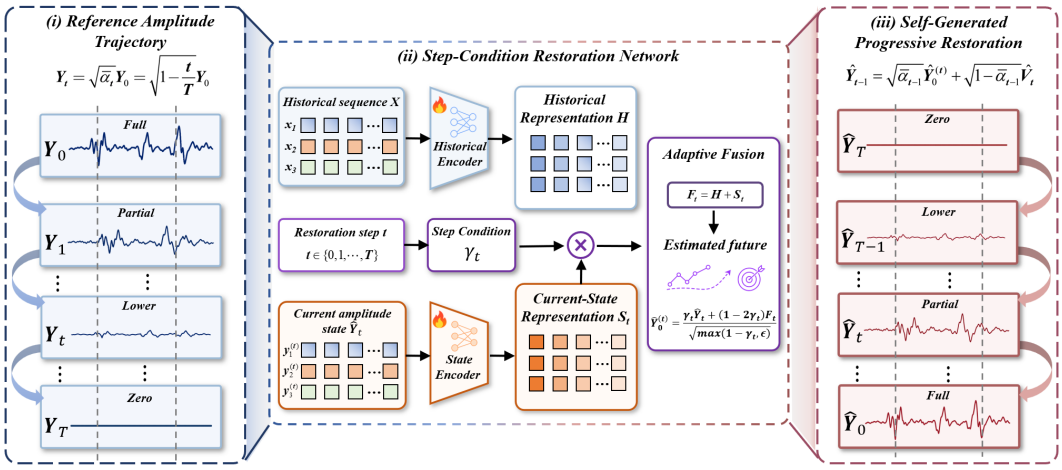


Figure 2: Overview of AmpTS.

defines a set of structure-preserving reference states at explicitly ordered restoration levels, which are used to organize transitions among subsequent self-generated restoration states.

We further define the reference-level transition coefficient between two consecutive restoration levels as

$$\alpha_t = \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}} = \frac{T-t}{T-t+1}, \qquad t \in \{1, \dots, T\}. \tag{2}$$

Accordingly, two adjacent ideal reference states satisfy $\mathbf{Y}_t = \sqrt{\alpha_t}\mathbf{Y}_{t-1}$. Unlike diffusion processes that construct states at different corruption levels through Gaussian perturbations, AmpTS defines its reference restoration levels solely through amplitude scaling of the target future sequence. The resulting reference states preserve the same temporal structure across restoration levels and form an explicitly ordered reference coordinate system. These reference states are used to define the restoration stages and provide transition references for the subsequent self-generated restoration states.

3.2 Step-Conditioned Restoration Network

Given the historical observations and the current self-generated restoration state at step t, AmpTS estimates the complete future sequence at each restoration stage and uses this estimate to progressively update the restoration state. To this end, we introduce the Step-Conditioned Restoration Network (SCR-Net). SCR-Net adopts a lightweight architecture composed of convolutional and linear layers. Conditioned on the restoration step t, it integrates the future prior inferred from the historical observations with the predictive information carried by the current self-generated restoration state, and outputs an estimate of the complete future sequence.

Historical representation extraction. To extract predictive information from the historical observations, AmpTS employs a lightweight historical encoder composed of temporal convolution, linear projection, and decomposition modules. Given the historical sequence $\mathbf{X} \in \mathbb{R}^{C \times L}$, we treat the variable dimension as convolutional channels and apply one-dimensional convolution along the temporal axis. The feature extraction process is defined as $\overline{\mathbf{X}} = \mathrm{Conv}_k(\mathbf{X})$, where $\mathrm{Conv}_k(\cdot)$ denotes a one-dimensional convolution with kernel size k. The convolutional output yields the enhanced historical representation $\overline{\mathbf{X}} \in \mathbb{R}^{C \times L}$. AmpTS subsequently maps the enhanced historical representation into the prediction space through a temporal linear projection: $\mathbf{H}^{(0)} = \overline{\mathbf{X}}\mathbf{W}_p$, where $\mathbf{W}_p \in \mathbb{R}^{L \times H}$ maps the historical dimension of length L to the prediction dimension of length H.

We then decompose the preliminary future representation into trend and fluctuation components:

$$\mathbf{H}^{\mathrm{tr}} = \mathcal{M}_k\left(\mathbf{H}^{(0)}\right), \qquad \mathbf{H}^{\mathrm{fl}} = \mathbf{H}^{(0)} - \mathbf{H}^{\mathrm{tr}}, \tag{3}$$

where $\mathcal{M}_k(\cdot)$ denotes a moving-average operator with kernel size k. The trend and fluctuation components are independently refined through linear mappings and then combined to form the final historical representation:

$$\widetilde{\mathbf{H}}^{\mathrm{tr}} = \mathbf{H}^{\mathrm{tr}}\mathbf{W}_{\mathrm{tr}}, \qquad \widetilde{\mathbf{H}}^{\mathrm{fl}} = \mathbf{H}^{\mathrm{fl}}\mathbf{W}_{\mathrm{fl}}, \qquad \mathbf{H} = \widetilde{\mathbf{H}}^{\mathrm{tr}} + \widetilde{\mathbf{H}}^{\mathrm{fl}}. \tag{4}$$

The resulting representation $\mathbf{H} \in \mathbb{R}^{C\times H}$ serves as the future prior inferred from the historical observations and is shared across all restoration steps.

Current-state representation extraction. At restoration step t, the current self-generated restoration state $\hat{\mathbf{Y}}_t \in \mathbb{R}^{C\times H}$ contains the predictive information progressively accumulated from previous restoration steps. AmpTS employs a lightweight temporal linear mapping to extract the current-state representation: $\mathbf{S}_t = \hat{\mathbf{Y}}_t\mathbf{W}_s$, where $\mathbf{W}_s \in \mathbb{R}^{H\times H}$ performs projection along the prediction-time dimension. The historical and current-state representations are then fused as $\mathbf{F}_t = \mathbf{H} + \mathbf{S}_t$. Here, $\mathbf{H}$ provides a future prior inferred from the historical observations, while $\mathbf{S}_t$ represents the predictive information carried by the current self-generated restoration state.

Step-conditioned restoration. Different restoration steps correspond to different restoration levels, and thus the reliability of the current self-generated state varies throughout the progressive restoration process. Following the step-adaptive balancing mechanism introduced by Gao et al. (2025), we introduce an independent learnable coefficient γ_t for each restoration step to regulate the interaction between the current restoration state and the fused future representation.

Specifically, at restoration step t, the estimate of the complete future sequence is formulated as

$$\widehat{\mathbf{Y}}_0^{(t)} = \frac{\gamma_t\widehat{\mathbf{Y}}_t + (1-2\gamma_t)\mathbf{F}_t}{\sqrt{\max(1-\gamma_t, \varepsilon)}}. \tag{5}$$

Here, $\varepsilon > 0$ is a small constant introduced for numerical stability. $\widehat{\mathbf{Y}}_t$ denotes the current self-generated restoration state, while $\mathbf{F}_t$ represents an auxiliary future representation obtained by integrating the historical prior with the current-state representation. Unlike conventional convex weighted fusion, γ_t does not simply represent a normalized weight between the two terms. Instead, it serves as a restoration-step-dependent modulation coefficient that adaptively adjusts the interaction between the current state and the auxiliary future representation across different restoration levels.

3.3 Self-Generated Progressive Restoration

During both training and inference, AmpTS starts from a zero state in the normalized space and progressively generates and updates the restoration state through state transitions parameterized by the predefined reference amplitude levels using the step-conditioned restoration network, ultimately producing the complete future prediction.

At restoration step t, given the historical representation $\mathbf{H} \in \mathbb{R}^{C\times H}$ and the current self-generated restoration state $\hat{\mathbf{Y}}_t \in \mathbb{R}^{C\times H}$, SCR-Net estimates the complete future sequence as $\hat{\mathbf{Y}}_0^{(t)} = R_\theta\left(\mathbf{H}, \hat{\mathbf{Y}}_t, t\right)$. $\hat{\mathbf{Y}}_0^{(t)}$ denotes the estimate of the entire forecasting horizon at restoration step t, rather than a next-step prediction generated sequentially along the temporal axis.

Inspired by the deterministic sampling formulation of DDIM (Song et al., 2021), we parameterize the transition between adjacent restoration levels using a complete-future reference component and a residual coordinate. Importantly, this formulation is used to construct transitions between self-generated forecasting states rather than to invert a Gaussian noising process.

To construct the next restoration state based on the current complete-future estimate, we define the complementary cumulative coefficient as $\bar{\beta}_t = 1 - \bar{\alpha}_t = \frac{t}{T}$. The trajectory residual is then computed from the discrepancy between the current self-generated state and the amplitude-scaled reference component constructed from the current complete-future estimate:

$$\hat{\mathbf{V}}_t = \frac{\hat{\mathbf{Y}}_t - \sqrt{\bar{\alpha}_t}\hat{\mathbf{Y}}_0^{(t)}}{\sqrt{\bar{\beta}_t}}, \qquad \hat{\mathbf{V}}_t \in \mathbb{R}^{C\times H}. \tag{6}$$

Here, $\hat{\mathbf{V}}_t$ characterizes the deviation of the current self-generated restoration state from the amplitude-scaled reference component constructed from $\hat{\mathbf{Y}}_0^{(t)}$, and preserves the state information accumulated during restoration across adjacent restoration levels.

AmpTS constructs the next self-generated restoration state by combining the current complete-future estimate and trajectory residual according to the preceding reference amplitude level:

$$\hat{\mathbf{Y}}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{Y}}_0^{(t)} + \sqrt{1-\bar{\alpha}_{t-1}}\hat{\mathbf{V}}_t. \tag{7}$$

3.4 Training Objective

AmpTS is trained through an end-to-end progressive restoration process. We use the mean squared error between the final prediction and the ground-truth future sequence as the training objective:

$$\mathcal{L} = \frac{1}{HC}\left\|\hat{\mathbf{Y}}_0 - \mathbf{Y}_0\right\|_F^2. \tag{8}$$

4 EXPERIMENTS

4.1 EXPERIMENT SETUP

Datasets. For long-term time-series forecasting, we conduct extensive experiments on eight real-world datasets from diverse application domains, including ETT (ETTh1, ETTh2, ETTm1, and ETTm2) (Zhou et al., 2021), Weather (Wu et al., 2021), Exchange (Lai et al., 2018), Stock (Yoon et al., 2019), and Energy (Candanedo et al., 2017). Detailed dataset statistics are provided in Appendix A.1.

Baselines. We compare AmpTS with a diverse set of representative and state-of-the-art time-series forecasting methods, including alignment-based methods: TimeAlign (Hu et al., 2026); convolution-based methods: TimesNet (Wu et al., 2023) and FACT (Wang et al., 2026); Transformer-based methods: TQNet (Lin et al., 2025), iTransformer (Liu et al., 2024), and PatchTST (Nie et al., 2023); MLP- and linear-based methods: TimeMixer (Wang et al., 2024) and DLinear (Zeng et al., 2023); recurrent methods: P-sLSTM (Kong et al., 2025b); and state-space and frequency-domain methods: Affirm (Wu et al., 2025). We further compare AmpTS with representative diffusion-based time-series methods applicable to forecasting, including CSDI (Tashiro et al., 2021), TimeDiff (Shen & Kwok, 2023), Diffusion-TS (Yuan & Qiao, 2024), CN-Diff (Rishi et al., 2025), MA-TSD (Wang et al., 2025), SSOL (Chen & Singh, 2025), and SimDiff (Ding et al., 2026).

Implementation Details. All experiments are implemented in Python 3.11 and PyTorch 2.5.1, and conducted on a single NVIDIA GeForce RTX 4060 Laptop GPU with CUDA 12.1. For long-term forecasting, the input length is fixed to $L = 96$ for all methods. In comparison with non-diffusion forecasting methods, the forecasting horizons are set to $F \in \{24, 48, 96, 192\}$ for Stock and $F \in \{96, 192, 336, 720\}$ for the remaining datasets, and we report the average performance over the four horizons. In comparison with diffusion-based forecasting methods, the forecasting horizon is set to $F = 192$ for Stock and $F = 720$ for all other datasets. We adopt Mean Squared Error (MSE) and Mean Absolute Error (MAE) as the evaluation metrics. Additional model configurations and hyperparameter settings are provided in Appendix A.4.

4.2 LONG-TERM FORECASTING

We evaluate AmpTS against both representative non-diffusion forecasting methods and diffusion-based time-series forecasting methods. The former assesses its overall forecasting capability, while the latter examines the effectiveness of deterministic progressive restoration in comparison with stochastic diffusion-based forecasting.

Comparison with Non-Diffusion Forecasting Methods. Table 1 reports the long-term forecasting performance of AmpTS against a diverse set of non-diffusion baselines. The results show that AmpTS achieves competitive forecasting performance across datasets with diverse temporal dynamics. Among the 16 dataset–metric combinations, AmpTS obtains the best or tied-best performance

in 11 cases, demonstrating a clear overall advantage in forecasting accuracy. Even on datasets where AmpTS does not achieve the best result, it remains highly competitive, with forecasting errors comparable to those of recent SOTA methods.

Overall, AmpTS achieves strong performance across a diverse set of non-diffusion time-series forecasting methods and outperforms the compared approaches in many settings. These results suggest that organizing time-series forecasting as a deterministic progressive restoration process can provide an effective alternative to conventional direct forecasting.

Table 1: Long-term forecasting results. The input length L is 96. All results are averaged across four forecasting lengths: $F \in \{24, 48, 96, 192\}$ for Stock and $F \in \{96, 192, 336, 720\}$ for the others. The best and second-best results are highlighted in **bold** and underlined, respectively.

Models	AmpTS (Ours)		TimeAlign (2026)		FACT (2026)		TQNet (2025)		P-sLSTM (2025b)		Affirm (2025)		TimeMixer (2024)		iTransformer (2024)		PatchTST (2023)		DLinear (2023)		TimesNet (2023)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.440**	**0.431**	0.453	0.435	0.469	0.450	0.441	0.434	0.450	0.450	0.444	0.436	0.444	0.439	0.459	0.451	0.446	0.446	0.463	0.459	0.460	0.455
ETTh2	**0.373**	**0.396**	0.375	0.399	0.379	0.402	0.378	0.403	0.613	0.537	0.374	0.400	0.393	0.409	0.383	0.406	0.381	0.409	0.561	0.519	0.409	0.425
ETTm1	0.398	0.395	0.389	**0.382**	0.389	0.402	**0.377**	0.394	0.409	0.417	0.396	0.403	0.383	0.398	0.415	0.415	0.388	0.403	0.404	0.408	0.412	0.417
ETTm2	0.285	0.326	0.278	**0.320**	**0.276**	0.321	0.277	0.322	0.405	0.416	0.280	0.327	0.277	0.323	0.294	0.335	0.289	0.334	0.355	0.402	0.295	0.332
Exchange	**0.347**	**0.395**	0.489	0.410	0.414	0.427	0.385	0.413	0.894	0.667	0.365	0.405	0.370	0.408	0.360	0.403	0.383	0.411	0.388	0.412	0.436	0.455
Weather	**0.240**	0.269	0.244	**0.264**	0.243	0.270	0.242	0.269	0.302	0.343	0.260	0.280	0.244	0.275	0.262	0.282	0.258	0.280	0.265	0.316	0.259	0.285
Energy	**0.112**	**0.232**	0.117	0.236	0.117	0.237	0.114	0.235	0.181	0.270	0.118	0.240	0.115	0.236	0.118	0.241	0.119	0.241	0.125	0.263	0.124	0.248
Stock	**0.048**	**0.152**	0.050	0.155	0.060	0.169	0.057	0.160	1.832	1.148	0.059	0.164	0.049	0.156	0.060	0.170	0.085	0.187	0.178	0.305	0.058	0.168

Comparison with Diffusion-Based Forecasting Methods. Table 2 presents a direct comparison between AmpTS and representative diffusion-based time-series forecasting methods. AmpTS achieves either the best or the second-best performance across all 16 dataset–metric combinations. These results provide evidence that stochastic Gaussian corruption is not necessary for achieving competitive progressive time-series point forecasting. Instead of constructing intermediate states through random perturbations, AmpTS organizes forecasting through deterministic reference amplitude levels and recursively generated restoration states, while retaining the multi-stage refinement characteristic of diffusion-style forecasting. This comparison supports the effectiveness of deterministic progressive restoration as an alternative way to organize time-series point forecasting.

Table 2: Long-term forecasting results. The input length L is 96. The forecasting length is $F = 192$ for Stock and $F = 720$ for all other datasets. The best and second-best results are highlighted in **bold** and underlined, respectively.

Models	AmpTS (Ours)		SimDiff (2026)		SSOL (2025)		CN-Diff (2025)		MA-TSD (2025)		Diffusion-TS (2024)		TimeDiff (2023)		CSDI (2021)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.468**	0.463	0.471	**0.454**	0.599	0.570	0.514	0.520	0.566	0.636	0.555	0.498	0.506	0.505	1.067	1.801
ETTh2	0.418	0.437	**0.406**	**0.426**	1.819	0.989	0.748	0.648	0.570	0.609	0.462	0.468	0.475	0.478	1.376	0.801
ETTm1	0.471	**0.435**	0.481	0.439	0.580	0.532	**0.461**	0.450	0.489	0.501	0.547	0.530	0.577	0.519	0.639	0.551
ETTm2	**0.406**	**0.398**	0.407	0.399	0.864	0.680	0.411	0.403	0.474	0.465	0.439	0.431	0.448	0.425	1.444	0.804
Exchange	0.818	0.678	1.019	0.746	1.368	0.885	2.144	1.166	**0.660**	**0.646**	1.102	1.008	1.094	0.809	3.977	1.712
Weather	**0.341**	**0.339**	0.351	0.345	0.355	0.361	0.417	0.382	0.362	0.372	0.388	0.387	0.399	0.391	0.352	0.346
Energy	**0.175**	**0.310**	0.178	0.314	2.185	1.161	0.182	0.319	0.192	0.350	0.190	0.347	0.210	0.344	0.195	0.337
Stock	**0.077**	**0.206**	0.101	0.238	8.441	2.563	0.086	0.218	0.194	0.267	0.122	0.144	0.092	0.221	1.438	1.010

4.3 ABLATION STUDY

To evaluate the effectiveness of the individual components in AmpTS, we conduct a comprehensive ablation study with the following variants. **w/o Iterative Restoration** removes the complete multi-

step progressive restoration process and performs prediction with a single call to SCR-Net. **w/o Residual Coordinate** discards the residual coordinate in the state transition and directly constructs the preceding state from the current complete-future estimate: $\hat{\mathbf{Y}}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{Y}}_0^{(t)}$. **Generic Iterative Refinement** retains the same number of restoration steps, but removes the amplitude-guided state transition and directly uses the current complete-future estimate as the state for the next iteration: $\hat{\mathbf{Y}}_{t-1} = \hat{\mathbf{Y}}_0^{(t)}$. **Gaussian Diffusion** replaces the reference amplitude-level construction with a Gaussian corruption-and-denoising process while retaining SCR-Net as the restoration backbone. **w/o Step Condition** removes the step-specific coefficients γ_t and instead shares a single coefficient across all restoration steps. **w/o State Representation** removes the linear transformation in the current-state branch and directly uses the current restoration state $\hat{\mathbf{Y}}_t$ in the subsequent fusion. **Simple Weighted Fusion** replaces the original restoration formulation in Eq. 5 with a simpler weighted combination: $\hat{\mathbf{Y}}_0^{(t)} = \gamma_t\hat{\mathbf{Y}}_t + (1-\gamma_t)\mathbf{F}_t$.

As shown in Table 3, AmpTS achieves better overall performance than all ablation variants, validating the effectiveness of the proposed progressive restoration framework. In particular, removing iterative restoration, trajectory residual, or amplitude-guided state transitions leads to clear performance degradation, indicating that the gains of AmpTS arise from the structured progressive restoration process rather than simply applying the forecasting network repeatedly. The other variants also generally result in reduced forecasting accuracy. More detailed analyses of the individual components are provided in Appendix B.1.

Table 3: Ablation study of AmpTS. The input length L is 96. All results are averaged across four forecasting lengths: $F \in \{24, 48, 96, 192\}$ for Stock and $F \in \{96, 192, 336, 720\}$ for the others. The best and second-best results are highlighted in **bold** and underlined, respectively.

Models	AmpTS (Full)		w/o Iterative Restoration		w/o Residual Coordinate		Generic Iterative Refinement		Gaussian Diffusion		w/o Step Condition		w/o State Representation		Simple Weighted Fusion	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh2	**0.373**	**0.396**	0.389	0.410	0.377	0.400	0.378	0.401	0.391	0.409	0.386	0.407	0.386	0.407	0.374	0.398
ETTm1	**0.398**	**0.395**	0.414	0.412	0.453	0.438	0.449	0.432	0.402	0.399	0.400	0.397	**0.398**	0.398	**0.398**	0.396
Exchange	**0.347**	**0.395**	0.375	0.414	0.348	0.397	**0.347**	**0.395**	0.364	0.413	0.375	0.415	0.387	0.422	0.351	0.401
Stock	**0.048**	**0.152**	0.053	0.160	0.051	0.157	0.051	0.157	0.051	0.157	0.050	0.156	0.051	0.157	0.051	0.156

4.4 Analysis of Progressive Restoration

Effect of Restoration Steps. The number of restoration steps T determines the granularity of the progressive restoration process in AmpTS. A larger T introduces finer intermediate transitions between the initial state and the final prediction, but does not necessarily yield better forecasting performance. As shown in Figure 3, increasing T initially improves both MSE and MAE, with the best performance achieved at a moderate number of restoration steps. Further increasing T provides little additional benefit and can even lead to slight performance degradation. Interestingly, the training and validation losses continue to decrease overall as T increases, suggesting that finer-grained restoration facilitates optimization of the training objective, while the resulting optimization gains do not necessarily translate into better forecasting generalization.

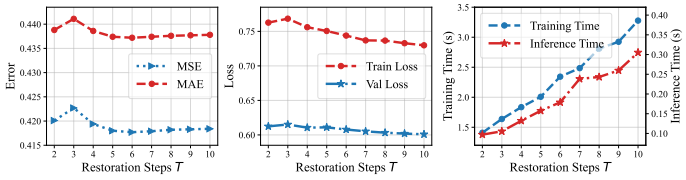


Figure 3: Effect of the number of restoration steps T on ETTh2 with prediction length $F = 720$.

Restoration Dynamics. To analyze how AmpTS progressively improves the prediction of the complete future sequence through self-generated restoration states, we take ETTm1 as an example, visualize the intermediate predictions at different restoration stages, and report the test-set prediction errors at each step. As shown in Figure 4, as restoration proceeds, AmpTS progressively captures the overall trends and local fluctuations of the future sequence, making the complete-future prediction gradually approach the ground truth. The results on the right show that both MSE and MAE exhibit

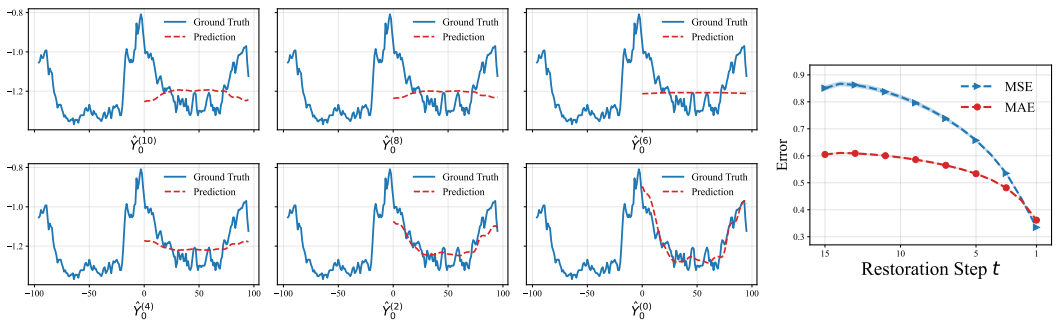


Figure 4: Progressive restoration process of AmpTS. Left: intermediate predictions at different restoration steps. Right: test-set MSE and MAE across restoration steps.

an overall decreasing trend throughout the restoration process, an overall improvement trend, despite small fluctuations at early restoration stages.

These results further illustrate the progressive forecasting behavior of AmpTS. Unlike generating the complete future sequence through a single forward pass, AmpTS recursively utilizes self-generated restoration states across multiple restoration stages and re-estimates the complete future sequence at each stage, thereby progressively improving the prediction.

4.5 Effectiveness of Progressive Restoration

To examine whether the progressive restoration process of AmpTS can operate beyond a specific historical encoder, we replace the original historical encoder with DLinear and iTransformer, respectively, while keeping the remaining restoration process unchanged. We then compare the original forecasting models with their counterparts integrated into AmpTS.

As shown in Table 4, introducing the progressive restoration process improves both DLinear and iTransformer under the majority of forecasting settings. These results do not establish compatibility with arbitrary forecasting models, but they suggest that the self-generated progressive restoration process can operate beyond a specific historical encoder, highlighting its potential as a flexible way to organize time-series forecasting.

Table 4: Compatibility of AmpTS with different historical encoders. The input length L is 96. The better result is highlighted in **bold**.

Models	Method	Metric	Stock				ETTh2				Energy				Exchange			
			24	48	96	192	96	192	336	720	96	192	336	720	96	192	336	720
DLinear	Ori.	MSE	0.091	0.123	0.166	0.331	0.351	0.481	0.594	0.819	0.087	0.104	0.129	**0.180**	0.095	0.182	0.327	**0.749**
		MAE	0.218	0.266	0.299	0.436	0.403	0.479	0.544	0.652	0.209	0.238	0.272	0.334	0.228	0.321	0.434	**0.664**
	+ AmpTS	MSE	**0.021**	**0.038**	**0.067**	**0.109**	**0.289**	**0.374**	**0.416**	**0.420**	**0.084**	**0.101**	**0.127**	0.181	**0.081**	**0.172**	**0.321**	0.828
		MAE	**0.106**	**0.139**	**0.188**	**0.250**	**0.337**	**0.388**	**0.426**	**0.439**	**0.195**	**0.223**	**0.255**	**0.317**	**0.197**	**0.293**	**0.408**	0.685
iTrans.	Ori.	MSE	0.022	0.039	0.072	0.108	0.295	**0.375**	0.416	0.432	**0.075**	0.097	**0.122**	**0.177**	0.086	0.177	0.330	0.846
		MAE	0.105	0.139	0.192	0.244	0.348	0.397	0.430	0.450	**0.182**	0.217	0.249	0.314	0.205	0.299	0.417	0.693
	+ AmpTS	MSE	**0.021**	**0.036**	**0.067**	**0.089**	**0.292**	0.376	**0.413**	**0.421**	**0.075**	**0.095**	**0.122**	**0.177**	**0.083**	**0.172**	**0.323**	**0.840**
		MAE	**0.102**	**0.136**	**0.186**	**0.222**	**0.344**	**0.393**	**0.428**	**0.440**	**0.182**	**0.213**	**0.248**	**0.313**	**0.202**	**0.295**	**0.412**	**0.692**

5 Conclusion

In this work, we propose AmpTS, an amplitude-guided progressive restoration method for time-series point forecasting. AmpTS uses structure-preserving reference amplitude levels to organize transitions among self-generated restoration states, enabling the model to repeatedly estimate and progressively refine the complete future sequence across multiple restoration stages. Extensive experiments demonstrate competitive and consistent forecasting performance against representative direct and diffusion-based forecasting methods. These results support self-generated progressive restoration as an effective alternative for organizing time-series point forecasting.

AI USE STATEMENT

In this work, we used generative AI tools to assist with methodological discussions, feedback on experimental design, code implementation and debugging, literature search, language polishing, and translation. All AI-assisted content was reviewed by the authors. All experiments reported in this paper were conducted by the authors, and the reported results were obtained from actual experimental runs and subsequently verified. Any code modified with the assistance of generative AI was tested in practice, and all cited references were verified against their original sources. All final decisions regarding the methodology, experimental design, interpretation of results, and conclusions of the paper were made by the authors. The authors take full responsibility for all content and claims presented in this work.

ETHICS STATEMENT

This work focuses on methodological research for time-series forecasting and does not involve human subjects, personally identifiable information, or the collection of private data. The proposed method is evaluated on standard time-series forecasting benchmarks. We are not aware of any significant ethical, privacy, security, fairness, or societal concerns arising from this work.

REPRODUCIBILITY STATEMENT

We have made efforts to ensure the reproducibility of all experimental results reported in this work. Detailed descriptions of the datasets, data splits, evaluation metrics, training settings, hyperparameters, and baseline implementations are provided in Appendix A. The complete training and inference procedures of AmpTS are summarized with pseudocode in Appendix C. We will also provide the source code, configuration files, and experiment scripts as anonymous supplementary materials to facilitate reproduction of our results.

REFERENCES

Arpit Bansal, Eitan Borgnia, Hong-Min Chu, Jie Li, Hamid Kazemi, Furong Huang, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Cold diffusion: Inverting arbitrary image transforms without noise. *Advances in Neural Information Processing Systems*, 36:41259–41282, 2023.

Francisco Caldas, Sahil Kumar, and Cláudia Soares. A decomposable forward process in diffusion models for time-series forecasting. *arXiv preprint arXiv:2601.21812*, 2026.

Luis M Candanedo, Véronique Feldheim, and Dominique Deramaix. Data driven prediction models of energy use of appliances in a low-energy house. *Energy and buildings*, 140:81–97, 2017.

Hui Chen and Vikas Singh. Single-step operator learning for conditioned time-series diffusion models. *Advances in Neural Information Processing Systems*, 38:163991–164021, 2025.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Hang Ding, Xue Wang, Tian Zhou, and Tao Yao. Simdiff: Simpler yet better diffusion model for time series point forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 20781–20789, 2026.

Xinyao Fan, Yueying Wu, Chang Xu, Yu-Hao Huang, Weiqing Liu, and Jiang Bian. Mg-tsd: Multi-granularity time series diffusion models with guided learning process. In *International Conference on Learning Representations*, volume 2024, pp. 12916–12934, 2024.

Jiaxin Gao, Qinglong Cao, and Yuntian Chen. Auto-regressive moving diffusion models for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 16727–16735, 2025.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in neural information processing systems*, 35:8633–8646, 2022.

Yifan Hu, Jie Yang, Tian Zhou, Peiyuan Liu, Yujin Tang, Rong Jin, and Liang Sun. Bridging past and future: Distribution-aware alignment for time series forecasting. In *International Conference on Learning Representations*, volume 2026, pp. 126148–126172, 2026.

Jongseon Kim, Hyungjoon Kim, HyunGi Kim, Dongjun Lee, and Sungroh Yoon. A comprehensive survey of deep learning for time series forecasting: Architectural diversity and open challenges. *Artificial Intelligence Review*, 58(7):216, 2025. doi: 10.1007/s10462-025-11223-9.

Xiangjie Kong, Zhenghao Chen, Weiyao Liu, Kaili Ning, Lechao Zhang, Syauqie Muhammad Marier, Yichen Liu, Yuhao Chen, and Feng Xia. Deep learning for time series forecasting: A survey. *International Journal of Machine Learning and Cybernetics*, 16:5079–5112, 2025a. doi: 10.1007/s13042-025-02560-w.

Yaxuan Kong, Zepu Wang, Yuqi Nie, Tian Zhou, Stefan Zohren, Yuxuan Liang, Peng Sun, and Qingsong Wen. Unlocking the power of lstm for long term time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 11968–11976, 2025b.

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*, pp. 95–104, 2018.

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

Xinhe Liu and Wenmin Wang. Deep time series forecasting models: A comprehensive survey. *Mathematics*, 12(10):1504, 2024. doi: 10.3390/math12101504.

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

Donghao Luo and Xue Wang. Moderntcn: A modern pure convolution structure for general time series analysis. In *International conference on learning representations*, volume 2024, pp. 31728–31770, 2024.

Reza Nematirad, Anil Pahwa, and Balasubramaniam Natarajan. Times2d: Multi-period decomposition and derivative mapping for general time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 19651–19658, 2025.

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.

Sibo Qi, Yuejing Zhai, Peng Chen, and Wuman Luo. Efficient and explainable multivariate time series forecasting: a survey of architectures, taxonomies, and open challenges. *Artificial Intelligence Review*, 2026.

Xiangfei Qiu, Hanyin Cheng, Xingjian Wu, Junkai Lu, Jilin Hu, Chenjuan Guo, Christian S Jensen, and Bin Yang. A comprehensive survey of deep learning for multivariate time series forecasting: A channel strategy perspective. *arXiv preprint arXiv:2502.10721*, 2025a.

Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. Duet: Dual clustering enhanced multivariate time series forecasting. In *Proceedings of the 31st ACM SIGKDD conference on knowledge discovery and data mining v. 1*, pp. 1185–1196, 2025b.

Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International conference on machine learning*, pp. 8857–8868. PMLR, 2021.

J Rishi, GVS Mothish, and Deepak Subramani. Conditional diffusion model with nonlinear data transformation for time series forecasting. In *ICML*, 2025.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Lifeng Shen and James Kwok. Non-autoregressive conditional diffusion models for time series prediction. In *International Conference on Machine Learning*, pp. 31016–31029. PMLR, 2023.

Lifeng Shen, Weiyu Chen, and James Kwok. Multi-resolution diffusion models for time series forecasting. In *International conference on learning representations*, volume 2024, pp. 49042–49060, 2024.

Jingzhe Shi, Qinwei Ma, Huan Ma, and Lei Li. Scaling law for time series forecasting. In *Advances in Neural Information Processing Systems*, volume 37, pp. 83314–83344, 2024.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in neural information processing systems*, 34:24804–24816, 2021.

Chenxi Wang, Linxiao Yang, Zhixian Wang, Liang Sun, and Yi Wang. A non-isotropic time series diffusion model with moving average transitions. In *Forty-second International Conference on Machine Learning*, 2025.

Huiqiang Wang, Jieming Shi, and Qing Li. Fact: Fine-grained across-variable convolution for multivariate time-series forecasting. In *International Conference on Learning Representations (ICLR)*, 2026.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. In *International conference on learning representations*, volume 2024, pp. 38626–38652, 2024.

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

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

Yuhan Wu, Xiyu Meng, Huajin Hu, Junru Zhang, Yabo Dong, and Dongming Lu. Affirm: Interactive mamba with adaptive fourier filters for long-term time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 21599–21607, 2025.

Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

Xinyu Yuan and Yan Qiao. Diffusion-ts: Interpretable diffusion for general time series generation. In *International Conference on Learning Representations*, volume 2024, pp. 41582–41610, 2024.

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

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

A Implementation Details

A.1 Datasets

We conduct experiments on eight widely used real-world time-series datasets for long-term forecasting, covering electricity, weather, finance, and energy domains. Specifically, the datasets include ETTh1, ETTh2, ETTm1, ETTm2, Weather, Exchange, Stock, and Energy. Detailed descriptions are provided as follows:

1. **ETT (Electricity Transformer Temperature) dataset** (Zhou et al., 2021) contains electricity transformer measurements collected from two stations in China from July 2016 to July 2018. It contains seven variables and is divided into four subsets: ETTh1 and ETTh2 are sampled hourly, while ETTm1 and ETTm2 are sampled every 15 minutes.
2. **Weather dataset** contains 21 meteorological variables collected by the Weather Station of the Max Planck Institute for Biogeochemistry in Germany. The data are recorded every 10 minutes and include measurements such as temperature, pressure, humidity, and wind-related variables.
3. **Exchange dataset** (Lai et al., 2018) contains daily exchange rates from eight countries or regions between 1990 and 2016. The dataset consists of eight correlated financial time series and exhibits strong non-stationarity and irregular temporal variations.
4. **Energy dataset** (Candanedo et al., 2017) contains energy-consumption and environmental measurements collected from a residential building at 10-minute intervals over approximately 4.5 months. It contains 28 numerical variables, including appliance energy consumption, indoor temperature and humidity, and outdoor weather conditions.
5. **Stock dataset** (Yoon et al., 2019) contains daily historical Google stock data from 2004 to 2019. It consists of six financial variables, including Open, High, Low, Close, Adjusted Close, and Volume.

Table 5: Statistics of the datasets used in our experiments. "Dim" denotes the number of variables. "Dataset Size" reports the numbers of forecasting samples in the training, validation, and test sets, respectively. The input length is fixed to $L = 96$.

Dataset	Dim	Prediction Length	Dataset Size	Frequency
ETTm1	7	$\{96, 192, 336, 720\}$	$(34465, 11521, 11521)$	15 min
ETTm2	7	$\{96, 192, 336, 720\}$	$(34465, 11521, 11521)$	15 min
ETTh1	7	$\{96, 192, 336, 720\}$	$(8545, 2881, 2881)$	1 hour
ETTh2	7	$\{96, 192, 336, 720\}$	$(8545, 2881, 2881)$	1 hour
Weather	21	$\{96, 192, 336, 720\}$	$(36792, 5271, 10540)$	10 min
Exchange	8	$\{96, 192, 336, 720\}$	$(5120, 665, 1422)$	1 day
Energy	28	$\{96, 192, 336, 720\}$	$(13623, 1879, 3852)$	10 min
Stock	6	$\{24, 48, 96, 192\}$	$(2460, 346, 714)$	1 day

A.2 Metric Details

We evaluate the forecasting accuracy using two commonly adopted metrics: Mean Squared Error (MSE) and Mean Absolute Error (MAE). Given the ground-truth values $\{y_i\}_{i=1}^{N}$ and the corresponding predictions $\{\hat{y}_i\}_{i=1}^{N}$, the two metrics are defined as

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2, \qquad \mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|, \tag{9}$$

where N denotes the total number of evaluated prediction elements. Lower values of MSE and MAE indicate better forecasting performance.

A.3 Baseline Details

Non-Diffusion Forecasting Methods. We provide a brief description of the non-diffusion forecasting baselines considered in our experiments, together with their publicly available implementations.

TimeAlign (Hu et al., 2026) is a distribution-aware forecasting framework that bridges the discrepancy between historical and future representations through a reconstruction-based alignment mechanism. Code: `https://github.com/TROUBADOUR000/TimeAlign`.

FACT (Wang et al., 2026) models fine-grained and dynamic inter-variable dependencies in both the time and frequency domains using depth-wise and multi-dilated convolutions. Code: `https://github.com/wanghq21/FACT`.

TQNet (Lin et al., 2025) introduces periodically shifted learnable temporal queries into the attention mechanism to capture global inter-variable patterns while retaining sample-specific correlations from the input sequence. Code: `https://github.com/ACAT-SCUT/TQNet`.

P-sLSTM (Kong et al., 2025b) adapts sLSTM to long-term time-series forecasting by incorporating patching and channel-independent modeling to improve long-range sequence representation. Code: `https://github.com/Eleanorkong/P-sLSTM`.

Affirm (Wu et al., 2025) combines adaptive Fourier filtering with interactive Mamba blocks to capture frequency-domain patterns and temporal dependencies at different granularities. Code: `https://github.com/zjuml/Affirm`.

TimeMixer (Wang et al., 2024) is an MLP-based forecasting architecture that decomposes and mixes seasonal and trend patterns across multiple temporal scales and integrates predictions from different scales. Code: `https://github.com/kwuking/TimeMixer`.

iTransformer (Liu et al., 2024) treats individual time-series variables as tokens and applies self-attention across variates to explicitly model multivariate correlations. Code: `https://github.com/thuml/iTransformer`.

PatchTST (Nie et al., 2023) segments each time series into temporal patches and applies a channel-independent Transformer to model long-range temporal dependencies efficiently. Code: `https://github.com/yuqinie98/PatchTST`.

DLinear (Zeng et al., 2023) decomposes the input sequence into trend and seasonal components and independently projects them into the forecasting horizon using lightweight linear layers. Code: `https://github.com/cure-lab/LTSF-Linear`.

TimesNet (Wu et al., 2023) transforms one-dimensional time series into two-dimensional representations according to multiple temporal periods and models both intra-period and inter-period variations using convolutional blocks. Code: `https://github.com/thuml/TimesNet`.

Diffusion-Based Forecasting Methods. We further consider representative diffusion-based time-series forecasting methods to compare deterministic progressive restoration with conventional stochastic diffusion processes.

SimDiff (Ding et al., 2026) develops an end-to-end diffusion framework for time-series point forecasting, using a unified Transformer as both denoiser and predictor together with normalization independence and a median-of-means ensemble for stable point estimation. Code: `https://github.com/Dear-Sloth/SimDiff`.

SSOL (Chen & Singh, 2025) formulates conditioned time-series diffusion from an operator-learning perspective and learns a frequency-aware inverse semigroup operator that directly maps highly cor-

rupted states to clean sequences in a single generation step. Code: https://github.com/vsingh-group/SSOL-timeseries.

CN-Diff (Rishi et al., 2025) introduces nonlinear time-dependent transformations and learnable conditions into the diffusion forward process, together with a corresponding conditional reverse process for time-series forecasting. Code: https://github.com/quest-lab-iisc/CNDiff.

MA-TSD (Wang et al., 2025) replaces isotropic Gaussian diffusion transitions with moving-average-based transitions to preserve low-frequency temporal information throughout the diffusion process. Code: https://github.com/WillWang1113/Moving-Average-Diffusion.

Diffusion-TS (Yuan & Qiao, 2024) employs an encoder–decoder Transformer with disentangled temporal representations and directly reconstructs clean time series at each diffusion step, while incorporating decomposition and frequency-domain objectives. Code: https://github.com/Y-debug-sys/Diffusion-TS.

TimeDiff (Shen & Kwok, 2023) is a non-autoregressive conditional diffusion model that introduces future mixup and autoregressive initialization to provide time-series-specific conditioning for the denoising process. Code: not publicly available.

CSDI (Tashiro et al., 2021) is a conditional score-based diffusion model that learns the conditional distribution of missing or future values given observed time-series measurements and can be applied to probabilistic forecasting. Code: https://github.com/ermongroup/CSDI.

A.4 Experiment Details

All experiments are implemented in Python 3.11 and PyTorch 2.5.1, and conducted on a single NVIDIA GeForce RTX 4060 Laptop GPU with CUDA 12.1. We use the Adam optimizer for model training and apply early stopping to mitigate overfitting. The input length is fixed to $L = 96$ for all datasets.

Table 6 summarizes the main hyperparameter settings of AmpTS on each dataset. In particular, *History Mode* specifies the processing configuration used in the historical representation branch. Depending on the dataset, different combinations of convolution, temporal linear projection, and decomposition-based refinement can be adopted instead of always using the complete historical branch described in the main text. *Conv. Kernel* denotes the kernel size of the one-dimensional convolution. We additionally report the number of restoration steps T, the learning rate and its scheduling strategy, and the maximum number of training epochs.

Table 6: Hyperparameter settings of AmpTS on different datasets.

Dataset	History Mode	Conv. Kernel	Restoration Steps T	LR	LR Schedule	Epochs
ETTh1	Conv.+Linear+Decomp.	7	15	0.001	Step Decay	20
ETTh2	Conv.+Decomp	1	5	0.0001	Step Decay	20
ETTm1	Conv.+Decomp	1	15	0.0001	Step Decay	20
ETTm2	Conv.+Decomp	1	18	0.00005	Step Decay	20
Weather	Conv.+Linear+Decomp.	7	4	0.001	Step Decay	20
Exchange	Conv.+Linear	5	3	0.0001	Step Decay	20
Energy	Conv.+Linear+Decomp.	3	5	0.001	Step Decay	20
Stock	Conv.+Linear	3	8	0.001	Step Decay	20

B ADDITIONAL EXPERIMENTAL RESULTS

B.1 ABLATION ANALYSIS

To further examine the role of each component in AmpTS, we provide a detailed analysis of the ablation variants reported in Table 3. Overall, the full AmpTS model achieves the most stable forecasting performance across the four datasets, while removing different components leads to varying degrees of performance degradation. These results suggest that the iterative mechanism, state transition, and step-conditioned modeling jointly contribute to the effectiveness of the progressive restoration process.

w/o Iterative Restoration. This variant completely removes the multi-step progressive restoration process and directly produces the final prediction through a single forward pass of SCR-Net. Compared with the full model, the forecasting errors increase across all four datasets. This result indicates that the performance gains of AmpTS do not arise solely from the historical encoder or a single complete-future estimation, but also benefit from the iterative use of self-generated restoration states. By decomposing one-shot forecasting into multiple restoration stages, the model can use states generated at preceding steps as conditions for subsequent predictions, thereby progressively refining its estimate of the complete future sequence.

w/o Residual Coordinate. This variant removes the residual coordinate during state transitions and constructs the preceding restoration state solely from the complete-future estimate obtained at the current step, thereby discarding the state information not represented by its amplitude-scaled complete-future component. Removing the residual coordinate degrades the overall performance on all datasets, with the most pronounced degradation observed on ETTm1. This suggests that, in addition to the amplitude-scaled component determined by the current complete-future estimate, the self-generated restoration state contains predictive information inherited from preceding restoration steps. The residual coordinate allows such information to be carried across adjacent restoration levels rather than reconstructing each state solely from the current prediction.

Generic Iterative Refinement. To determine whether the performance improvement simply results from repeatedly applying the forecasting network, we further remove the amplitude-guided state transition while retaining the same number of iterations and directly use the complete-future estimate from the current step as the state for the next iteration. This variant shows a clear degradation on ETTm1, where MSE increases from 0.398 to 0.449 and MAE from 0.395 to 0.432, while different degrees of degradation are also observed on ETTh2 and Stock. Overall, simply repeating the forecasting process cannot consistently reproduce the performance of AmpTS. This indicates that multi-step computation alone is insufficient and that retaining information from preceding self-generated states is beneficial. AmpTS organizes this state propagation through reference restoration levels and the corresponding transition rule, allowing preceding states to continuously influence subsequent complete-future estimation.

Gaussian Diffusion. This variant replaces the reference amplitude-level construction with a Gaussian corruption-and-denoising process while retaining the same restoration network architecture as much as possible. The results show that Gaussian diffusion does not outperform the full AmpTS model on any of the four datasets. This suggests that, under the current unified forecasting framework, introducing stochastic Gaussian perturbations is not necessary for competitive progressive point forecasting. In contrast, AmpTS uses amplitude scaling to define structure-preserving reference restoration levels and uses these levels to organize transitions among self-generated restoration states without constructing intermediate states through stochastic corruption.

w/o Step Condition. This variant removes the independent modulation coefficients associated with different restoration steps and instead shares a single coefficient across all steps. Its overall performance is inferior to that of the full model across the four datasets. This indicates that different restoration stages do not necessarily require the same interaction between the current-state representation and the auxiliary future representation. Step-specific parameters allow SCR-Net to adapt this interaction according to the current restoration level.

w/o State Representation. This variant removes the linear transformation in the current-state branch and directly uses the current restoration state for subsequent fusion. The degradation is particularly evident on the Exchange dataset, while performance also decreases to varying degrees

on ETTh2 and Stock. These results suggest that the current restoration state itself may not lie in a representation space that is well suited for fusion with the historical future prior. Through the state representation mapping, the model can reorganize the temporal information contained in the self-generated state and extract predictive features that are more suitable for complete-future estimation, thereby improving the interaction between historical information and current-state information.

Simple Weighted Fusion. This variant replaces the original step-adaptive interaction in Eq. (5) with a conventional convex weighted combination: $\widehat{\mathbf{Y}}_0^{(t)} = \gamma_t \widehat{\mathbf{Y}}_t + (1-\gamma_t)\mathbf{F}_t$. Its performance remains generally close to that of the full model, but slight degradation is still observed across the datasets. This suggests that, as the restoration state continuously evolves throughout the iterative process, allowing more flexible state–representation interactions across different restoration levels can further improve progressive restoration compared with a fixed-form convex combination.

Overall, these ablation results demonstrate from different perspectives that the performance of AmpTS cannot be attributed to any single component or to simply performing prediction multiple times. Multi-step iteration establishes the basic progressive forecasting process, the amplitude-guided state transition determines how states evolve across different restoration stages, and the trajectory residual preserves and propagates the information accumulated during restoration. Meanwhile, step-conditioned modulation and state representation further improve the model's ability to exploit the current state at different restoration levels. Together, these components constitute the progressive restoration mechanism of AmpTS and enable more stable forecasting performance across different time-series datasets.

B.2 Detailed Restoration Dynamics

To provide a more detailed presentation of the progressive restoration process of AmpTS and its corresponding results, we report the test-set MSE and MAE at each restoration step under different forecasting horizons on the ETTm1 and Energy datasets, as shown in Tables 7 and 8. Each result is computed over the entire test set, together with the corresponding 95% confidence interval obtained by bootstrap resampling over forecasting windows. Across both datasets and different forecasting horizons, the restoration processes exhibit similar trends: the prediction error may show slight fluctuations during the early stages, while both MSE and MAE generally decrease as restoration proceeds. These results provide a more complete numerical presentation of the restoration dynamics shown in the main text.

Table 7: Restoration dynamics of AmpTS on ETTm1 under different forecasting horizons. Each entry reports the mean error and its 95% bootstrap confidence interval.

Restoration Step	$F=96$		$F=192$		$F=336$		$F=720$	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$t=15$	0.850 [0.842, 0.859]	0.605 [0.602, 0.608]	0.857 [0.849, 0.865]	0.612 [0.609, 0.615]	0.856 [0.848, 0.863]	0.616 [0.614, 0.619]	0.874 [0.865, 0.891]	0.629 [0.626, 0.635]
$t=14$	0.866 [0.858, 0.876]	0.610 [0.607, 0.614]	0.867 [0.859, 0.875]	0.615 [0.613, 0.618]	0.857 [0.850, 0.864]	0.617 [0.615, 0.619]	0.853 [0.849, 0.861]	0.622 [0.620, 0.625]
$t=13$	0.862 [0.853, 0.871]	0.609 [0.605, 0.612]	0.861 [0.853, 0.869]	0.613 [0.610, 0.616]	0.850 [0.843, 0.856]	0.614 [0.612, 0.617]	0.842 [0.838, 0.847]	0.618 [0.617, 0.620]
$t=12$	0.851 [0.842, 0.860]	0.605 [0.602, 0.609]	0.851 [0.842, 0.858]	0.610 [0.607, 0.612]	0.840 [0.833, 0.847]	0.611 [0.608, 0.614]	0.832 [0.829, 0.836]	0.614 [0.613, 0.616]
$t=11$	0.837 [0.827, 0.846]	0.600 [0.597, 0.603]	0.838 [0.829, 0.846]	0.605 [0.602, 0.608]	0.829 [0.822, 0.836]	0.607 [0.605, 0.609]	0.823 [0.820, 0.827]	0.611 [0.610, 0.613]
$t=10$	0.818 [0.810, 0.827]	0.594 [0.591, 0.597]	0.822 [0.814, 0.829]	0.599 [0.597, 0.602]	0.816 [0.809, 0.822]	0.602 [0.600, 0.604]	0.814 [0.812, 0.818]	0.608 [0.607, 0.609]
$t=9$	0.796 [0.787, 0.805]	0.586 [0.583, 0.589]	0.803 [0.796, 0.811]	0.593 [0.590, 0.595]	0.801 [0.794, 0.808]	0.597 [0.594, 0.599]	0.805 [0.803, 0.809]	0.604 [0.603, 0.606]
$t=8$	0.770 [0.761, 0.778]	0.576 [0.574, 0.579]	0.781 [0.775, 0.789]	0.585 [0.582, 0.587]	0.784 [0.778, 0.790]	0.590 [0.588, 0.593]	0.795 [0.793, 0.800]	0.600 [0.600, 0.602]
$t=7$	0.739 [0.730, 0.746]	0.565 [0.562, 0.568]	0.756 [0.749, 0.763]	0.575 [0.572, 0.577]	0.764 [0.759, 0.770]	0.583 [0.580, 0.585]	0.784 [0.782, 0.789]	0.596 [0.595, 0.598]
$t=6$	0.702 [0.694, 0.710]	0.551 [0.548, 0.554]	0.726 [0.719, 0.732]	0.563 [0.561, 0.566]	0.741 [0.734, 0.747]	0.574 [0.571, 0.576]	0.771 [0.768, 0.776]	0.591 [0.590, 0.593]
$t=5$	0.658 [0.651, 0.665]	0.534 [0.531, 0.537]	0.689 [0.683, 0.695]	0.549 [0.547, 0.551]	0.711 [0.706, 0.718]	0.562 [0.560, 0.564]	0.754 [0.751, 0.759]	0.584 [0.583, 0.586]
$t=4$	0.605 [0.598, 0.611]	0.512 [0.509, 0.515]	0.643 [0.637, 0.649]	0.530 [0.528, 0.532]	0.673 [0.668, 0.678]	0.546 [0.544, 0.548]	0.728 [0.725, 0.734]	0.574 [0.572, 0.576]
$t=3$	0.536 [0.530, 0.543]	0.482 [0.479, 0.484]	0.581 [0.575, 0.586]	0.503 [0.501, 0.505]	0.618 [0.613, 0.623]	0.523 [0.521, 0.525]	0.686 [0.683, 0.692]	0.556 [0.555, 0.558]
$t=2$	0.444 [0.438, 0.449]	0.436 [0.434, 0.438]	0.491 [0.486, 0.496]	0.459 [0.457, 0.461]	0.532 [0.527, 0.536]	0.482 [0.480, 0.484]	0.605 [0.603, 0.609]	0.520 [0.519, 0.522]
$t=1$	0.335 [0.330, 0.339]	0.362 [0.359, 0.363]	0.376 [0.371, 0.381]	0.381 [0.379, 0.383]	0.407 [0.403, 0.412]	0.401 [0.399, 0.402]	0.472 [0.471, 0.472]	0.435 [0.435, 0.436]

B.3 Computational Efficiency

To further evaluate the computational efficiency of AmpTS, we compare the forecasting accuracy and computational cost of different methods on the Weather and ETTh1 datasets. As shown in the

Table 8: Restoration dynamics of AmpTS on Energy under different forecasting horizons. Each entry reports the mean error and its 95% bootstrap confidence interval.

Restoration Step	$F = 96$		$F = 192$		$F = 336$		$F = 720$	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$t = 5$	0.153 [0.151, 0.155]	0.273 [0.271, 0.274]	0.175 [0.173, 0.177]	0.297 [0.295, 0.299]	0.197 [0.195, 0.199]	0.320 [0.318, 0.322]	0.244 [0.242, 0.246]	0.369 [0.368, 0.371]
$t = 4$	0.170 [0.168, 0.172]	0.287 [0.286, 0.289]	0.194 [0.192, 0.196]	0.312 [0.310, 0.314]	0.218 [0.215, 0.220]	0.336 [0.334, 0.338]	0.270 [0.268, 0.273]	0.386 [0.384, 0.388]
$t = 3$	0.155 [0.153, 0.157]	0.275 [0.274, 0.277]	0.176 [0.174, 0.178]	0.298 [0.297, 0.300]	0.199 [0.196, 0.201]	0.322 [0.320, 0.324]	0.248 [0.246, 0.251]	0.372 [0.370, 0.374]
$t = 2$	0.117 [0.116, 0.118]	0.240 [0.238, 0.241]	0.133 [0.131, 0.134]	0.260 [0.259, 0.262]	0.152 [0.150, 0.154]	0.283 [0.281, 0.285]	0.195 [0.193, 0.197]	0.332 [0.330, 0.334]
$t = 1$	0.070 [0.069, 0.070]	0.172 [0.171, 0.173]	0.089 [0.088, 0.091]	0.204 [0.203, 0.206]	0.116 [0.114, 0.118]	0.240 [0.238, 0.242]	0.175 [0.173, 0.176]	0.310 [0.308, 0.312]

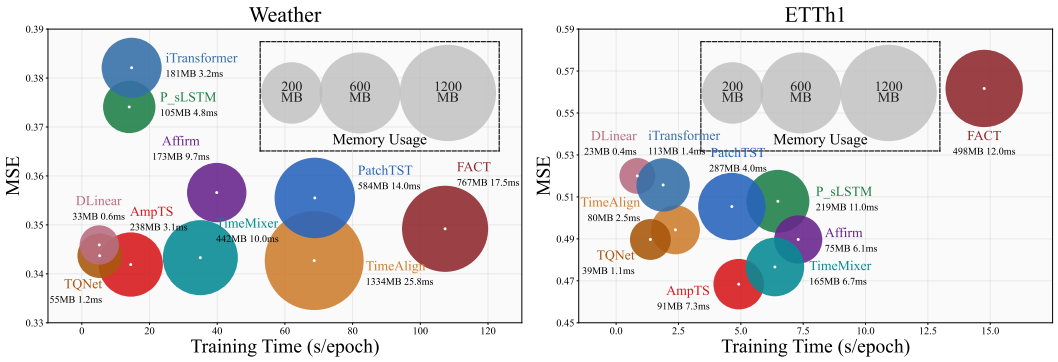


Figure 5: Computational efficiency comparison on the Weather and ETTh1 datasets with an input length of $L = 96$ and a forecasting horizon of $F = 720$.

Figure 5, the horizontal axis denotes the average training time per epoch, the vertical axis reports the forecasting MSE, and the bubble size represents memory usage. The memory consumption and inference time of each model are additionally annotated next to the corresponding point.

On the Weather dataset, AmpTS requires substantially less training time than PatchTST, TimeMixer, TimeAlign, FACT, and Affirm. Although lightweight models such as DLinear and TQNet incur lower computational costs, AmpTS achieves more competitive forecasting performance while maintaining relatively low training and inference overhead. In contrast, several more complex models achieve comparable forecasting accuracy at the cost of considerably higher training time and memory consumption; for example, TimeAlign requires more than 1 GB of memory.

The trend is even more evident on ETTh1. AmpTS requires only approximately 91 MB of memory and maintains a relatively low training cost, while achieving the lowest MSE among the compared methods. Compared with TimeMixer, PatchTST, P-sLSTM, and FACT, AmpTS provides a clear advantage in forecasting accuracy while also maintaining relatively low training time and memory usage. Although lightweight models such as DLinear and TQNet achieve faster inference, their forecasting errors remain higher than those of AmpTS.

Overall, benefiting from the lightweight SCR-Net and a small number of restoration steps, AmpTS achieves competitive or superior forecasting accuracy while maintaining relatively low training time, memory consumption, and inference overhead, demonstrating a favorable accuracy–efficiency trade-off.

B.4 Robustness and Statistical Significance

To evaluate the robustness of AmpTS to random initialization, we repeat the experiments with three random seeds and compare AmpTS with TQNet and SimDiff, which are the strongest non-diffusion and diffusion-based baselines in our main experiments, respectively. For each dataset and each random seed, we first average the results over the four forecasting horizons, and then report the mean and standard deviation across the three runs. As shown in Table 9, AmpTS exhibits consistently small standard deviations across different datasets, indicating stable forecasting performance under different random initializations.

We further perform two-sided Wilcoxon signed-rank tests following prior work (Shi et al., 2024). For the comparison with TQNet, rather than using the dataset-level averages reported in Table 9, we retain all dataset–horizon combinations for statistical testing. Specifically, the results are first averaged across the three random seeds for each dataset–horizon pair, resulting in $8 \times 4 = 32$ paired forecasting tasks. The Wilcoxon test yields $p_{\mathrm{MSE}} = 0.6916$ and $p_{\mathrm{MAE}} = 0.0028$. Therefore, the difference in MSE is not statistically significant, whereas the difference in MAE is significant at the 1% significance level. AmpTS achieves lower MAE on 23 out of the 32 paired forecasting tasks, indicating a more consistent advantage in absolute forecasting error.

For SimDiff, based on the dataset-level averages currently reported in Table 9, we conduct the Wilcoxon test over the eight paired datasets. The resulting p-values are $p_{\mathrm{MSE}} = 0.0078$ and $p_{\mathrm{MAE}} = 0.0078$, indicating statistically significant differences in both metrics at the 1% significance level.

Table 9: Robustness comparison of AmpTS, TQNet, and SimDiff over three random seeds. For each random seed, the results are averaged across four forecasting horizons: $F \in \{24, 48, 96, 192\}$ for Stock and $F \in \{96, 192, 336, 720\}$ for the other datasets. Results are reported as mean $\pm$ standard deviation across three runs.

Model	AmpTS		TQNet (2025)		SimDiff (2026)	
Dataset	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	0.440 ± 0.0010	0.431 ± 0.0006	0.442 ± 0.0011	0.436 ± 0.0018	0.478 ± 0.0073	0.460 ± 0.0064
ETTh2	0.373 ± 0.0007	0.397 ± 0.0004	0.376 ± 0.0020	0.402 ± 0.0010	0.407 ± 0.0016	0.427 ± 0.0008
ETTm1	0.398 ± 0.0002	0.395 ± 0.0001	0.379 ± 0.0017	0.394 ± 0.0014	0.477 ± 0.0042	0.437 ± 0.0031
ETTm2	0.285 ± 0.0001	0.326 ± 0.0001	0.277 ± 0.0001	0.322 ± 0.0005	0.403 ± 0.0014	0.393 ± 0.0004
Weather	0.242 ± 0.0005	0.270 ± 0.0002	0.250 ± 0.0064	0.275 ± 0.0053	0.351 ± 0.0013	0.345 ± 0.0007
Exchange	0.348 ± 0.0003	0.396 ± 0.0002	0.372 ± 0.0141	0.408 ± 0.0058	1.096 ± 0.0667	0.780 ± 0.0308
Stock	0.049 ± 0.0004	0.152 ± 0.0003	0.056 ± 0.0027	0.162 ± 0.0046	0.107 ± 0.0054	0.246 ± 0.0066
Energy	0.113 ± 0.0001	0.232 ± 0.0001	0.113 ± 0.0005	0.232 ± 0.0009	0.179 ± 0.0018	0.315 ± 0.0018

C Training and Inference Procedures

Algorithm 1 and Algorithm 2 summarize the training and inference procedures of AmpTS, respectively. Both training and inference start from the terminal zero state and proceed through the same multi-step progressive restoration process.

Algorithm 1: AmpTS Training

Input: Training set $\mathcal{D}$; restoration steps T
Output: Trained AmpTS parameters (ϕ, θ)
Predefine $\bar{\alpha}_t = 1 - \frac{t}{T}$ and $\bar{\beta}_t = 1 - \bar{\alpha}_t$ for $t = 0, \ldots, T$;
repeat
  Sample a batch $(\mathbf{X}, \mathbf{Y}_0) \sim \mathcal{D}$;
  Normalize $\mathbf{X}$ and $\mathbf{Y}_0$ using statistics computed from $\mathbf{X}$;
  Obtain the historical future prior: $\mathbf{H} \leftarrow \mathrm{HistEnc}_\phi(\mathbf{X})$;
  Initialize $\widehat{\mathbf{Y}}_T \leftarrow \mathbf{0}$;
  for $s = T, T-1, \ldots, 1$ **do**
    Estimate the complete future sequence: $\widehat{\mathbf{Y}}_0^{(s)} \leftarrow R_\theta(\mathbf{H}, \widehat{\mathbf{Y}}_s, s)$;
    Estimate the trajectory residual: $\widehat{\mathbf{V}}_s \leftarrow \dfrac{\widehat{\mathbf{Y}}_s - \sqrt{\bar{\alpha}_s}\widehat{\mathbf{Y}}_0^{(s)}}{\sqrt{\bar{\beta}_s}}$;
    Update the preceding restoration state: $\widehat{\mathbf{Y}}_{s-1} \leftarrow \sqrt{\bar{\alpha}_{s-1}}\widehat{\mathbf{Y}}_0^{(s)} + \sqrt{\bar{\beta}_{s-1}}\widehat{\mathbf{V}}_s$;
  end
  Compute the training objective: $\mathcal{L} \leftarrow \mathrm{MSE}\left(\widehat{\mathbf{Y}}_0, \mathbf{Y}_0\right)$;
  Update (ϕ, θ) using $\nabla_{\phi,\theta}\mathcal{L}$;
until *converged*;

Algorithm 2: AmpTS Inference

Input: Historical sequence $\mathbf{X}$; restoration steps T; trained parameters (ϕ, θ)
Output: Forecast $\widehat{\mathbf{Y}}_0$
Predefine $\bar{\alpha}_t = 1 - \frac{t}{T}$ and $\bar{\beta}_t = 1 - \bar{\alpha}_t$ for $t = 0, \ldots, T$;
Normalize $\mathbf{X}$ using its historical statistics;
Obtain the historical future prior: $\mathbf{H} \leftarrow \mathrm{HistEnc}_\phi(\mathbf{X})$;
Initialize the terminal restoration state: $\widehat{\mathbf{Y}}_T \leftarrow \mathbf{0}$;
for $t = T, T-1, \ldots, 1$ **do**
    Estimate the complete future sequence: $\widehat{\mathbf{Y}}_0^{(t)} \leftarrow R_\theta(\mathbf{H}, \widehat{\mathbf{Y}}_t, t)$;
    Estimate the trajectory residual: $\widehat{\mathbf{V}}_t \leftarrow \dfrac{\widehat{\mathbf{Y}}_t - \sqrt{\bar{\alpha}_t}\widehat{\mathbf{Y}}_0^{(t)}}{\sqrt{\bar{\beta}_t}}$;
    Update the preceding restoration state: $\widehat{\mathbf{Y}}_{t-1} \leftarrow \sqrt{\bar{\alpha}_{t-1}}\widehat{\mathbf{Y}}_0^{(t)} + \sqrt{\bar{\beta}_{t-1}}\widehat{\mathbf{V}}_t$;
end
Inverse-normalize $\widehat{\mathbf{Y}}_0$;
return $\widehat{\mathbf{Y}}_0$;

D CODE AVAILABILITY

The code for AmpTS is publicly available at: `https://anonymous.4open.science/r/ICLR2027-AmpTS-AF11/`.