

# Battling the Non-stationarity in Time Series Forecasting via Test-time Adaptation

HyunGi Kim<sup>1</sup>, Siwon Kim<sup>1</sup>, Jisoo Mok<sup>1</sup>, Sungroh Yoon<sup>1, 2, 3 †</sup>

<sup>1</sup>Department of Electrical and Computer Engineering, Seoul National University

<sup>2</sup>Interdisciplinary Program in Artificial Intelligence, Seoul National University

<sup>3</sup>AIIS, ASRI, and INMC, Seoul National University

† Corresponding Author

rlagusrl0128@snu.ac.kr, tuslkk17@gmail.com, magicshop1118@snu.ac.kr, sryoon@snu.ac.kr

## Abstract

Deep Neural Networks have spearheaded remarkable advancements in time series forecasting (TSF), one of the major tasks in time series modeling. Nonetheless, the non-stationarity of time series undermines the reliability of pre-trained source time series forecasters in mission-critical deployment settings. In this study, we introduce a pioneering test-time adaptation framework tailored for TSF (TSF-TTA). TAFAS, the proposed approach to TSF-TTA, flexibly adapts source forecasters to continuously shifting test distributions while preserving the core semantic information learned during pre-training. The novel utilization of partially-observed ground truth and gated calibration module enables proactive, robust, and model-agnostic adaptation of source forecasters. Experiments on diverse benchmark datasets and cutting-edge architectures demonstrate the efficacy and generality of TAFAS, especially in long-term forecasting scenarios that suffer from significant distribution shifts. The code is available at <https://github.com/kimanki/TAFAS>.

## Introduction

Time series forecasting (TSF), which is one of the most core tasks in time series modeling, aims to predict future values based on historical data points. The widespread applications of TSF across various industries include but are not limited to: weather prediction (Verma, Heinonen, and Garg 2024), traffic forecasting (Liu et al. 2023a), stock market prediction (Li et al. 2023a), and supply chain management (Hosseinnia Shavaki and Ebrahimi Ghahnavieh 2023). Such a broad and over-arching impact of TSF results highlights the importance of developing a dependable time series forecaster, whose predictions maintain reliability despite changes in external factors.

A critical bottleneck in the reliable deployment of pre-trained time series forecasters is created by the non-stationary nature of real-world time series data that leads to continuous data distribution shifts (Petropoulos et al. 2022). Previous works on alleviating the effect of non-stationarity aim to improve the robustness of time series forecasters through advancements in the pre-training process (Kim et al. 2021; Fan et al. 2023; Liu et al. 2024). Unfortunately, as the non-stationarity worsens the distributional discrepancy between training and test data over time, the pre-trained forecaster becomes increasingly unreliable, even

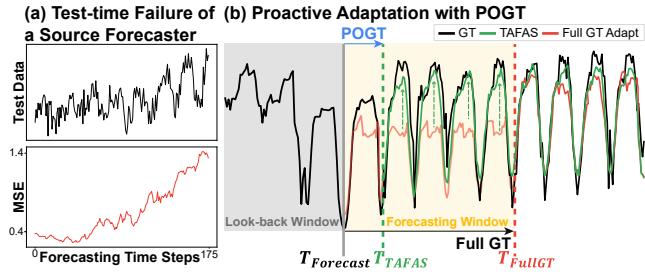


Figure 1: (a) The performance of a pre-trained source forecaster degrades when it encounters distribution-shifted inputs at test-time. In the figure, the distribution shift occurs through the gradual increase of mean value. (b) The sequential nature of time series provides the opportunity to proactively adapt the forecaster with partially-observed ground truth (POGT) before acquiring full ground truth (GT).

if it has learned meaningful temporal semantics from training data (Kuznetsov and Mohri 2014). In Figure 1(a), we visualize how constantly evolving, non-stationary test data negatively affect the forecasting results of a pre-trained time series forecaster.

This shortcoming of existing approaches underscores the need to continuously adapt the pre-trained source forecaster on shifted test-time inputs while preserving its core semantics. Adapting the source forecaster to incorporate new time-variant semantics within test-time inputs allows it to reflect the ever-changing test distributions. In this regard, we pioneer a test-time adaptation (TTA) framework tailored for TSF (TSF-TTA). TTA, which has been primarily studied in the computer vision domain under classification settings (Wang et al. 2021; Niu et al. 2022, 2023; Lee et al. 2024), dynamically adjusts a pre-trained classifier on test inputs; this objective of TTA makes it well-aligned with the aforementioned motivation of adapting the source forecaster to newly arriving test data. Traditionally, TTA has operated under two main assumptions on the nature of test inputs. First, TTA assumes a complete absence of test labels because it is infeasible to hand annotate inputs at test-time. Second, because most of the image data are assumed to be Independent and Identically Distributed (IID), TTA generally operates under the same IID assumption. In TSF-TTA, however, these assumptions no longer hold due to the intrin-

sic characteristics of time series data.

Unlike the first assumption of TTA, in TSF-TTA, ground truth for predicted time steps eventually becomes accessible, albeit in a delayed manner. For instance, when predicting electricity consumption for the next 30 days, the actual amount of electricity consumption becomes known to us after 30 days. Interestingly, as depicted in Figure 1(b), the sequential nature of time series makes ground truth values partially observable before acquiring the full ground truth. In the aforementioned example, the partial ground truth for the first 7 days is obtainable only after a week. Utilizing this partially-observable ground truth offers an invaluable opportunity to preemptively perform TSF-TTA prior to the arrival of full ground truth. Moreover, the second assumption is violated in TSF-TTA because temporal dependency exists in time series. This necessitates a technique for addressing the non-IIDness of time series on local (within window) and global (throughout the entire test-time) levels.

By considering these challenges and opportunities presented by properties of time series, we propose a **T**est-time **A**daptive **F**orec**A**sting for non-stationary time Series (**TAFAS**) that is extensible to various TSF architectures. TAFAS consists of periodicity-aware adaptation scheduling (PAAS) and a gated calibration module (GCM). PAAS adaptively obtains partially-observed ground truth of sufficient length to represent semantically meaningful periodic patterns. After then, model-agnostic GCMs are adapted to calibrate test-time inputs such that they conform to the distribution the source forecaster effectively handles. The gating mechanism in GCMs controls how much the calibrated results should be utilized by considering global distribution shifts. Together, PAAS and GCM allow the source forecaster to be proactively adapted on non-stationary test-time inputs. Throughout the adaptation, the source forecaster remains frozen to preserve the core semantics it has learned from the extensive historical data. With the proactively adapted forecaster, TAFAS adjusts the latter part of the original predictions, where ground truths are yet to be observed, with the adapted predictions reflecting the distribution shift.

Comprehensive experimental results demonstrate that the TAFAS consistently enhances forecasting capabilities across source forecasters of various architectures. TAFAS leads to particularly large performance gains in long-term forecasting scenarios where distribution shifts become more pronounced. Moreover, the seamless integration of TAFAS with methods addressing the non-stationarity in pre-training stage and time series foundation models further enhances their ability to navigate test-time distribution shifts. Notably, TAFAS improves the forecasting error of Chronos (Ansari et al. 2024) on unseen test data streams by up to 45%.

Our contributions are summarized as follows:

- We pioneer test-time adaptation in time series forecasting (TSF-TTA) to address the non-stationarity in time series. Our examination of the properties of time series reveals the challenges in extending existing TTA frameworks to TSF, necessitating a new avenue to enable TSF-TTA.
- We introduce TAFAS, a model-agnostic TSF-TTA framework that consists of periodicity-aware adapta-

tion scheduling (PAAS) and Gated Calibration Module (GCM). These two technical components collectively enable proactive adaptation of the source forecaster on test-time inputs while preserving its core semantics.

- TAFAS consistently excels in test-time adaptation across various TSF benchmark datasets and architectures, significantly improving test errors on highly non-stationary data and in long-term forecasting scenarios.

## Related Works

As TSF has become a pivotal application in various industries, diverse TSF architectures have been developed. Due to the page limit, an exhaustive discussion on TSF architectures and TTA is included in Appendix. Here, we focus on studies that improve the time series forecaster by mitigating distribution shifts caused by the non-stationarity of time series. A line of studies introduces normalization and de-normalization modules before and after the forecaster to remove and restore the non-stationary statistics (Kim et al. 2021; Liu et al. 2022; Fan et al. 2023; Liu et al. 2024). RevIN (Kim et al. 2021) performs instance normalization with learnable scale and bias factors, whereas NST (Liu et al. 2022) uses a non-parametric approach without learnable transformations. Dish-TS (Fan et al. 2023) and SAN (Liu et al. 2024) perform statistical prediction both within the look-back window and between the look-back and prediction windows to appropriately execute normalization and denormalization. However, their generalization capability to the continuously evolving test data distribution is inherently limited as they address non-stationarity only within training distributions. Although TSF-TTA and online TSF share a common ground in learning from streaming inputs (Wen et al. 2024; Ao and Faye 2023; Guo et al. 2016; Pham et al. 2023), their primary objectives are fundamentally different. Online TSF aims to train a time series forecaster from scratch with streaming data, but TSF-TTA aims to adapt a pre-trained source forecaster.

## Challenges and Opportunities in TSF-TTA

Generally, TTA leverages unlabeled test-time inputs to adapt classifiers on shifted distributions. In this study, we aim to extend TTA to TSF to improve the ability of the pre-trained source time series forecaster to handle newly arriving test data. While the task characteristics of TSF and the properties of time series make applying the existing TTA frameworks to TSF a non-trivial problem, they also open new doors to develop a tailored approach to TSF-TTA. In this section, we highlight the challenges (C) and opportunities (O) unique to TSF-TTA after introducing task definition and formulations.

**Task definition & formulations.** TSF is the task of predicting the future horizon window of  $H$  time steps ( $\{\mathbf{x}_{t+1}, \dots, \mathbf{x}_{t+H}\}$ ) given the past look-back window of  $L$  time steps ( $\{\mathbf{x}_{t-L+1}, \dots, \mathbf{x}_t\}$ ).  $\mathbf{x}_t \in \mathbb{R}^C$  denotes  $C$  number of variables observed at time  $t$ . To perform TSF, a time series forecaster  $\mathcal{F}_\theta: \mathbb{R}^{L \times C} \rightarrow \mathbb{R}^{H \times C}$  is trained to predict subsequent future  $H$  time steps given  $L$  past time steps.

The train, validation, and test sets of TSF datasets are obtained by splitting a single continuous time se-

ries  $\{\mathbf{x}_1, \dots, \mathbf{x}_T\}$  in chronological order. Then, the (past, future) pairs are obtained using a sliding window, *i.e.*,  $(\mathbf{X}_t, \mathbf{Y}_t) = (\{\mathbf{x}_{t-L+1}, \dots, \mathbf{x}_t\}, \{\mathbf{x}_{t+1}, \dots, \mathbf{x}_{t+H}\})$ . In non-stationary time series, data distribution continuously changes over time, resulting in distribution shifts between data splits and also within each split.

**C1. Entropy-based TTA losses are infeasible for regression-based TSF.** TTA fundamentally assumes that ground truth labels are not available. Therefore, existing TTA methods utilize the entropy of predicted class probability distributions to formulate objective functions for the adaptation (Wang et al. 2021; Niu et al. 2022; Lee et al. 2024). However, TSF is a regression task, where the entropy of class probabilities is ill-defined, rendering the straightforward extension of entropy-based losses impossible.

**O1. Ground truth is accessible in TSF.** In TSF, ground truth values become accessible as the predicted future time steps eventually arrive. Therefore, instead of entropy-based losses, the Mean Squared Error (MSE) loss, the de facto objective function used for regression tasks, can be used as a learning signal to perform TSF-TTA.

**C2. Full ground truth is accessible in a delayed time, resulting in a delayed adaptation.** However, computing the MSE loss after observing full ground truth results in a delay of  $H$  time steps between the point of forecasting ( $t$ ) and obtaining the full ground truth ( $t' = t + H$ ). Thus, naively waiting for the arrival of full ground truth to perform TSF-TTA implies that none of the forecasted predictions in  $H$  time steps can be adapted. As the length of the forecasting window increases, the point at which full ground truth becomes obtainable is further delayed. This further delay in the point of adaptation inhibits performing TSF-TTA in a timely manner to reflect the adjacent distribution shifts.

**O2. Utilizing partially-observed ground truth enables proactive TSF-TTA.** The sequential nature of test-time inputs makes ground truth partially observable before acquiring the full ground truth. After  $p$  time steps ( $p < H$ ) from the forecasting time step  $t$  of  $\mathbf{X}_t$ , the first  $p$  time steps out of the full ground truth (*i.e.*,  $\{\mathbf{x}_{t+1}, \dots, \mathbf{x}_{t+p}\} \in \mathbb{R}^{p \times C}$ ) are observable. Replacing the full ground truth in the MSE loss with its partially-observed counterpart reduces the adaptation delay and thus enables proactive adaptation.

## TAFAS: Test-time Adaptive Forecasting for Non-stationary Time Series

In this section, we introduce TAFAS, a novel framework that considers the challenges and opportunities of TSF-TTA. The overall pipeline of TAFAS is provided in Figure 2.

### Periodicity-Aware Adaptation Scheduling (PAAS)

To enable a proactive adaptation of the pre-trained source forecaster by reducing the adaptation delay, in TAFAS, we utilize partially-observed ground truth (POGT). However, as stated in **Section O2.**, POGT does not eliminate the adaptation delay because to obtain POGT of length  $p$ , we must wait for  $p$  time steps. Therefore, choosing the appropriate value of  $p$  is important for balancing the trade-off between

the amount of semantic information in POGT and the adaptation delay. When  $p$  is large, the POGT contains copious semantic information, but the adaptation delay increases, offsetting the advantage of employing the POGT. Conversely, when  $p$  is small, the forecaster can be adapted more proactively, but the POGT may contain meaningless patterns.

To ensure that the POGT incorporates semantically meaningful temporal patterns while preventing an excessive time delay, we introduce a periodicity-aware adaptation scheduling (PAAS) scheme that reflects the inherent periodic patterns of the test-time inputs. Several studies have demonstrated that the look-back window contains meaningful periodic patterns (Wu et al. 2023, 2021). PAAS thus extracts these patterns from the look-back window to determine  $p$ . From here on,  $t_0$  denotes the time step at which the first test-time look-back window is obtained. PAAS applies variable-wise Fast Fourier Transform (FFT) on the first look-back window  $\mathbf{X}_{t_0}$  to identify the variable with the highest signal power (Eq. 1). Before FFT, the mean of  $\mathbf{X}_{t_0}$  is set to zero to remove the influence of bias. For the identified variable  $c^*$ , PAAS calculates the amplitude of each frequency component to determine the dominant frequency (Eq. 2).

$$c^* = \arg \max_c \sum_f \|\text{FFT}(\mathbf{X}_{t_0}^c)\|^2 \quad (1)$$

$$f^* = \arg \max_f \|\text{FFT}(\mathbf{X}_{t_0}^{c^*})\|^2 \quad (2)$$

Based on the relationship between the frequency and period, PAAS derives the period of the dominant periodic patterns of  $\mathbf{X}_{t_0}$  and set it to the length of POGT as  $p_{t_0} = \lceil \frac{L}{f^*} \rceil$ . The resulting periodicity-aware POGT  $\mathbf{Y}_{t_0}[: p_{t_0}]$  includes relevant semantics embodied in the dominant periodic patterns.

Once  $p_{t_0}$  is determined,  $p_{t_0} + 1$  instances are aggregated into a test mini-batch:  $\{\mathbf{X}\}_{t_0}^{t_0 + p_{t_0}} = \{\mathbf{X}_{t_0}, \dots, \mathbf{X}_{t_0 + p_{t_0}}\}$ . When the subsequent look-back window arrives at time step  $t_0 + p_{t_0} + 1$ , PAAS is repeated to calculate the subsequent length of POGT adaptively. Considering the inherent periodic patterns vary across datasets and instances, the adaptive characteristic of PAAS assures data-agnostic TSF-TTA.

### Gated Calibration Module (GCM)

Selecting the module to adapt is a critical design choice in TTA. Existing TTA methods generally adapt normalization layers, *e.g.*, Batch Normalization (Ioffe and Szegedy 2015), to adjust the distributions of the intermediate features. However, state-of-the-art TSF methods adopt various forms of architectures, many of which are missing such normalization layers. Hence, we introduce a model-agnostic Gated Calibration Module (GCM) to guarantee that TAFAS can be applied generally to diverse pre-trained source forecasters.

TAFAS adapts the GCM with the source forecaster frozen to preserve the core temporal semantics that the forecaster has learned from extensive historical data. GCM is attached to both the front and tail ends of the source forecaster, referred to as input and output GCMs. The input GCM maps the distribution-shifted test input  $\mathbf{X}_t$  to a calibrated input  $\mathbf{X}_t^{\text{cali}}$  that belong in a distribution the source forecaster can

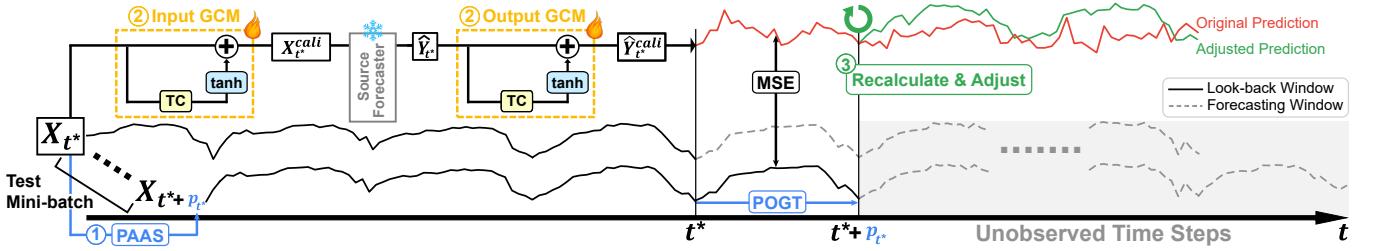


Figure 2: An overview of TAFAS. (1: Blue) By computing the periodicity of dominant patterns for  $X_{t^*}$ , PAAS determines the length of partially-observed ground truth (POGT)  $p_{t^*}$ . (2: Yellow) Then input and output GCMs are proactively adapted on  $X_{t^*}$  at  $t^* + p_{t^*}$  to mitigate local and global distribution shifts through Temporal Calibration (TC) and gating ( $\tanh$ ) mechanisms, by minimizing MSE between the POGT and corresponding prediction. The source forecaster is frozen to preserve its core semantic information learned. (3: Green) Following the proactive adaptation, predictions for test mini-batch  $\{X\}_{t^*}^{t^*+p_{t^*}}$  are recalculated to reflect the distribution shift and the unobserved part of the original predictions is adjusted with the adapted predictions.

handle. The output GCM remaps the source forecaster’s prediction  $\hat{Y}_t$  to  $\hat{Y}_t^{\text{cali}}$  in order to calibrate  $\hat{Y}_t$  back to the continuously changing test distribution.

GCM consists of variable-wise temporal calibration and gating mechanisms to handle both the local (within the look-back window) and global (throughout the entire test-time) non-stationarity. The temporal calibration handles local distribution shifts by transforming the given window to calculate calibrated results. The gating mechanism handles global distribution shifts by updating how much to reflect calibrated results over time. Both temporal calibration and gating mechanisms are applied variable-wise, since each variable can have a different degree of non-stationarity. The two operations in the input GCM are expressed as the following:

$$\begin{aligned} \text{GCM}(X_t) = X_t + \\ \text{Tile}(\tanh(\alpha)) \circ \left( \text{Concat}(\{\mathbf{W}^c X_t^c\}_{c=1}^C) + \mathbf{b} \right), \end{aligned} \quad (3)$$

where  $\mathbf{W}^c \in \mathbb{R}^{L \times L}$ ,  $\mathbf{b} \in \mathbb{R}^{L \times C}$ , and  $\alpha \in \mathbb{R}^C$ .  $\text{Tile}(\cdot) : \mathbb{R}^C \rightarrow \mathbb{R}^{L \times C}$  broadcasts the gating vector in a temporal dimension,  $\text{Concat}(\cdot)$  concatenates the calibrated signals along variable dimension, and  $\circ$  denotes a Hadamard product.  $\mathbf{W}^c$  and  $\mathbf{b}$  are initialized to be zero so that at the first test time step when the test data diverge little from the training distribution, the original input is passed without calibration. The mechanism of the output GCM follows Eq. 3 with  $X_t$  replaced with  $\hat{Y}_t$ , and the dimensions of  $\mathbf{W}^c$  and  $\mathbf{b}$  adjusted accordingly to  $H \times H$  and  $H \times C$ .

Let  $t^*$  denote a time step at which PAAS calculates the POGT length  $(t_0, t_0 + p_{t_0} + 1, \dots)$  and  $p_{t^*}$  denote the POGT length computed at  $t^*$ . After a test mini-batch  $\{X\}_{t^*}^{t^*+p_{t^*}}$  is obtained at  $t^* + p_{t^*}$ , GCMs are adapted by minimizing the TAFAS loss defined as the following:

$$\mathcal{L}^{\text{partial}} = \text{MSE}(\hat{Y}_{t^*}^{\text{cali}}[:, p_{t^*}], Y_{t^*}[:, p_{t^*}]) \quad (4)$$

$$\mathcal{L}^{\text{full}} = \text{MSE}(\{\hat{Y}_{t^*}^{\text{cali}}\}_{\tilde{t}^*}^{t^*+p_{t^*}}, \{Y\}_{\tilde{t}^*}^{t^*+p_{t^*}}) \quad (5)$$

$$\mathcal{L}^{\text{TAFAS}} = \mathcal{L}^{\text{partial}} + \mathcal{L}^{\text{full}}, \quad (6)$$

where  $\tilde{t}^*$  represents the most recent time step among the past POGT-computing steps whose corresponding mini-batches have now observed their full ground truths.  $\mathcal{L}^{\text{partial}}$  is computed between the first  $p_{t^*}$  time steps of the calibrated prediction for  $X_{t^*}$  whose periodicity-aware POGT is available

( $\hat{Y}_{t^*}^{\text{cali}}[:, p_{t^*}]$ ) and the corresponding POGT ( $Y_{t^*}[:, p_{t^*}]$ ).  $\mathcal{L}^{\text{full}}$  is calculated between the calibrated predictions of all look-back windows within the past mini-batch constructed at  $\tilde{t}^* + p_{\tilde{t}^*}$  and the associated full ground truths to use their longer semantic information for adaptation. This adaption process of TAFAS enables proactive adaptation right when the semantically meaningful POGT is obtained while utilizing the extended semantic details from the full ground truths of the past mini-batch. When no past mini-batch with full ground truths is available, e.g., at the beginning of the adaptation phase, TAFAS uses only  $\mathcal{L}^{\text{partial}}$  for adaptation.

### Prediction Adjustment (PA)

Because the forecaster is proactively adapted, TAFAS can replace the latter part of original predictions, whose ground truths are yet to be observed, with adjusted predictions that reflect the distribution shift. After the forecaster is adapted at  $t^* + p_{t^*}$ , TAFAS recalculates the predictions for all look-back windows in  $\{X\}_{t^*}^{t^*+p_{t^*}}$  and then substitutes the original predictions for time steps after  $t^* + p_{t^*}$  with the adapted predictions. Specifically, for the look-back window  $X_{t^*+k}$ , where  $k \in \{0, \dots, p_{t^*}\}$ , the corresponding prediction  $\hat{Y}_{t^*+k}^{\text{cali}}$  predicts time steps  $\{(t^* + k + 1), \dots, (t^* + k + H)\}$ . For the time steps  $\{(t^* + p_{t^*} + 1), \dots, (t^* + k + H)\}$  which are yet to be observed, TAFAS substitutes the original prediction  $\hat{Y}_{t^*+k}^{\text{cali}}$  with the adapted prediction  $\hat{Y}_{t^*+k}^{\text{cali}, \text{adapted}}$  that reflects distribution shifts as the following:

$$\hat{Y}_{t^*+k, i}^{\text{adjust}} = \begin{cases} \hat{Y}_{t^*+k, i}^{\text{cali}} & \text{if } i \leq (t^* + p_{t^*}) \\ \hat{Y}_{t^*+k, i}^{\text{cali}, \text{adapted}} & \text{if } i > (t^* + p_{t^*}) \end{cases} \quad (7)$$

$\hat{Y}_{t^*+k, i}^{\text{cali}, \text{adapted}}$  denotes the adapted prediction values for the time step  $i$  of  $\hat{Y}_{t^*+k}^{\text{cali}}$ . We summarize the overall pipeline of TAFAS in Appendix.

## Experiments

### Experimental Setup

**Datasets.** We demonstrate the effectiveness of TAFAS using the seven widely used multivariate TSF benchmark datasets: ETTh1, ETTh1, ETTh2, ETTm1, ETTm2, Exchange, Illness, and

Table 1: Test MSE on multivariate time series forecasting datasets with and without TAFAS across various TSF architectures: Transformer-based (iTransformer, PatchTST), Linear-based (DLinear, OLS), and MLP-based (FreTS, MICN). +TAFAS denotes whether the TAFAS framework is applied to the corresponding source forecaster. The lower MSE is marked in bold.

Models + TAFAS	Transformer-based				Linear-based				MLP-based				
	iTransformer		PatchTST		DLinear		OLS		FreTS		MICN		
	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	
ETTh1	96	0.444	<b>0.438</b>	0.436	<b>0.429</b>	0.451	<b>0.442</b>	0.451	<b>0.441</b>	0.441	<b>0.437</b>	0.455	<b>0.446</b>
	192	0.503	<b>0.492</b>	0.492	<b>0.481</b>	0.504	<b>0.493</b>	0.504	<b>0.494</b>	0.498	<b>0.491</b>	0.513	<b>0.501</b>
	336	0.562	<b>0.554</b>	0.539	<b>0.529</b>	0.551	<b>0.541</b>	0.551	<b>0.540</b>	0.563	<b>0.555</b>	0.574	<b>0.561</b>
	720	0.786	<b>0.704</b>	0.713	<b>0.690</b>	0.700	<b>0.669</b>	0.700	<b>0.662</b>	0.715	<b>0.684</b>	0.736	<b>0.702</b>
ETTm1	96	0.388	<b>0.362</b>	0.386	<b>0.377</b>	0.371	<b>0.351</b>	0.371	<b>0.353</b>	0.367	<b>0.355</b>	0.398	<b>0.372</b>
	192	0.448	<b>0.429</b>	0.440	<b>0.429</b>	0.443	<b>0.418</b>	0.444	<b>0.417</b>	0.429	<b>0.418</b>	0.448	<b>0.428</b>
	336	0.519	<b>0.493</b>	0.500	<b>0.487</b>	0.518	<b>0.481</b>	0.518	<b>0.479</b>	0.493	<b>0.476</b>	0.524	<b>0.493</b>
	720	0.592	<b>0.560</b>	0.562	<b>0.542</b>	0.592	<b>0.549</b>	0.592	<b>0.549</b>	0.560	<b>0.539</b>	0.602	<b>0.567</b>
ETTh2	96	0.241	<b>0.239</b>	0.233	<b>0.232</b>	0.229	<b>0.227</b>	0.231	<b>0.229</b>	0.234	<b>0.232</b>	0.234	<b>0.231</b>
	192	0.291	<b>0.287</b>	0.282	<b>0.277</b>	0.283	<b>0.281</b>	0.284	<b>0.281</b>	0.286	<b>0.282</b>	0.285	<b>0.282</b>
	336	0.333	<b>0.326</b>	0.328	<b>0.318</b>	0.325	<b>0.317</b>	0.326	<b>0.317</b>	0.328	<b>0.318</b>	0.330	<b>0.320</b>
	720	0.415	<b>0.393</b>	0.416	<b>0.396</b>	0.415	<b>0.391</b>	0.416	<b>0.386</b>	0.420	<b>0.390</b>	0.414	<b>0.398</b>
ETTm2	96	0.159	<b>0.157</b>	0.157	<b>0.156</b>	0.159	<b>0.158</b>	0.160	<b>0.159</b>	0.158	<b>0.156</b>	0.161	<b>0.160</b>
	192	0.197	<b>0.192</b>	0.194	<b>0.194</b>	0.193	<b>0.191</b>	0.194	<b>0.192</b>	0.193	<b>0.192</b>	0.195	<b>0.193</b>
	336	0.244	<b>0.235</b>	0.234	<b>0.232</b>	0.232	<b>0.229</b>	0.233	<b>0.230</b>	0.233	<b>0.230</b>	0.235	<b>0.232</b>
	720	0.312	<b>0.301</b>	0.307	<b>0.299</b>	0.306	<b>0.297</b>	0.307	<b>0.298</b>	0.302	<b>0.293</b>	0.308	<b>0.299</b>
Exchange	96	0.086	<b>0.084</b>	0.084	<b>0.081</b>	<b>0.078</b>	0.079	0.081	<b>0.079</b>	0.084	<b>0.079</b>	0.083	<b>0.079</b>
	192	0.175	<b>0.165</b>	0.179	<b>0.168</b>	0.171	<b>0.161</b>	0.172	<b>0.162</b>	0.176	<b>0.163</b>	0.183	<b>0.170</b>
	336	0.329	<b>0.280</b>	0.350	<b>0.286</b>	0.321	<b>0.280</b>	0.323	<b>0.265</b>	0.326	<b>0.283</b>	0.342	<b>0.296</b>
	720	0.844	<b>0.773</b>	0.845	<b>0.844</b>	0.837	<b>0.711</b>	0.836	<b>0.583</b>	0.840	<b>0.772</b>	1.276	<b>0.942</b>
Illness	24	<b>2.119</b>	2.124	<b>2.078</b>	<b>2.078</b>	2.642	<b>2.631</b>	2.509	<b>2.506</b>	<b>2.516</b>	<b>2.516</b>	<b>3.280</b>	3.306
	36	1.989	<b>1.988</b>	<b>2.095</b>	<b>2.095</b>	2.501	<b>2.450</b>	2.435	<b>2.384</b>	<b>2.441</b>	<b>2.441</b>	<b>3.503</b>	3.524
	48	2.173	<b>2.155</b>	<b>1.964</b>	<b>1.964</b>	2.487	<b>2.403</b>	2.417	<b>2.336</b>	2.257	<b>2.198</b>	2.066	<b>2.033</b>
	60	1.925	<b>1.876</b>	<b>1.833</b>	<b>1.833</b>	2.530	<b>2.444</b>	2.490	<b>2.415</b>	2.074	<b>2.056</b>	1.915	<b>1.881</b>
Weather	96	0.179	<b>0.170</b>	0.174	<b>0.171</b>	0.195	<b>0.179</b>	0.196	<b>0.179</b>	0.181	<b>0.169</b>	0.176	<b>0.175</b>
	192	0.227	<b>0.214</b>	0.221	<b>0.215</b>	0.240	<b>0.223</b>	0.240	<b>0.223</b>	0.225	<b>0.212</b>	0.224	<b>0.222</b>
	336	0.284	<b>0.265</b>	0.276	<b>0.265</b>	0.292	<b>0.270</b>	0.292	<b>0.270</b>	0.280	<b>0.260</b>	0.280	<b>0.274</b>
	720	0.360	<b>0.343</b>	0.352	<b>0.334</b>	0.364	<b>0.345</b>	0.364	<b>0.343</b>	0.356	<b>0.336</b>	0.350	<b>0.349</b>

Weather (Wu et al. 2021). In Appendix, we report the results of the ADF test (Elliott, Rothenberg, and Stock 1992) to demonstrate that a substantial degree of non-stationarity exists in all these datasets.

**Time series forecasters.** For the source time series forecasters, we adopt six state-of-the-art forecasters across various architectures: Transformer-based (iTransformer (Liu et al. 2023b), PatchTST (Nie et al. 2022)), Linear-based (DLinear (Zeng et al. 2023), OLS (Toner and Darlow 2024)), and MLP-based (FreTS (Yi et al. 2024), MICN (Wang et al. 2023)). Moreover, we verify the effectiveness of TAFAS on TSF foundation model (Ansari et al. 2024) pre-trained on a large corpus of time series data. TAFAS can be applied in combination with all of these forecasters regardless of the architecture design due to its fully model-agnostic design.

**Implementation details.** Unless stated otherwise, we follow the standard protocol in TSF evaluation (Wu et al. 2023). We use the look-back window length  $L = 36$  for Illness and  $L = 96$  for the other datasets. For forecasting window length  $H$ , we evaluate on 4 different lengths,  $H \in \{24, 36, 48, 60\}$  for Illness and  $H \in \{96, 192, 336, 720\}$  for the other datasets. We split datasets in chronological order

with the ratio of (0.6, 0.2, 0.2) for ETTh1, ETTm1, ETTh2, and ETTm2 and (0.7, 0.1, 0.2) for Exchange, Illness, and Weather to construct train, validation, and test sets. We repeat each pre-training run over three different seeds and select the pre-trained source forecaster with the lowest average validation MSE. More details on training processes are provided in Appendix.

## TAFAS on Various TSF Architectures

Table 1 presents the MSE of forecasting results with and without TAFAS across various source TSF architectures and multiple forecasting windows. The full results, including Mean Absolute Error (MAE) and standard deviations, are reported in Appendix due to the space limit. TAFAS consistently reduces the forecasting error at test-time, effectively handling the test-time non-stationarity of time series. We highlight that the effectiveness of TAFAS at mitigating test-time distribution shifts remains strong across various architectures and datasets, consolidating its broad model- and data-agnostic applicability. Furthermore, when  $H = 336$ , TAFAS improves the average MSE of iTransformer and DLinear by **4.95%** and **5.20%**, respectively, and on an even

Table 2: Compatability of TAFAS with various normalization modules addressing non-stationarity in the pre-training stage. We report test MSE with and without TAFAS where each source forecaster (iTransformer, DLinear, and FreTS) is pre-trained with the normalization modules of RevIN, Dish-TS, or SAN.

Models	iTransformer						DLinear						FreTS						
	Norm. + TAFAS	RevIN		Dish-TS		SAN		RevIN		Dish-TS		SAN		RevIN		Dish-TS		SAN	
		$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$	$\times$	$\checkmark$
ETTh1	96	0.444	<b>0.437</b>	0.457	<b>0.441</b>	0.451	<b>0.439</b>	0.451	<b>0.444</b>	0.452	<b>0.439</b>	0.438	<b>0.433</b>	0.448	<b>0.436</b>	0.440	<b>0.421</b>	0.438	<b>0.434</b>
	192	0.503	<b>0.490</b>	0.507	<b>0.499</b>	0.511	<b>0.497</b>	0.504	<b>0.497</b>	0.507	<b>0.488</b>	0.489	<b>0.480</b>	0.511	<b>0.491</b>	0.494	<b>0.472</b>	0.485	<b>0.478</b>
	336	0.561	<b>0.548</b>	0.560	<b>0.549</b>	0.596	<b>0.564</b>	0.550	<b>0.548</b>	0.555	<b>0.531</b>	0.534	<b>0.527</b>	0.567	<b>0.551</b>	0.547	<b>0.521</b>	0.529	<b>0.523</b>
	720	0.785	<b>0.702</b>	0.720	<b>0.669</b>	0.727	<b>0.689</b>	0.699	<b>0.669</b>	0.708	<b>0.642</b>	0.664	<b>0.652</b>	0.729	<b>0.682</b>	0.706	<b>0.641</b>	0.665	<b>0.640</b>
ETTm1	96	0.754	<b>0.652</b>	0.418	<b>0.367</b>	0.360	<b>0.351</b>	0.371	<b>0.354</b>	0.371	<b>0.350</b>	0.353	<b>0.345</b>	1.071	<b>0.360</b>	0.375	<b>0.354</b>	0.355	<b>0.347</b>
	192	0.817	<b>0.791</b>	0.487	<b>0.430</b>	0.418	<b>0.408</b>	0.445	<b>0.422</b>	0.445	<b>0.416</b>	0.415	<b>0.405</b>	0.893	<b>0.421</b>	0.447	<b>0.418</b>	0.415	<b>0.405</b>
	336	0.870	<b>0.833</b>	0.543	<b>0.493</b>	0.480	<b>0.463</b>	0.520	<b>0.484</b>	0.520	<b>0.477</b>	0.474	<b>0.460</b>	0.556	<b>0.478</b>	0.522	<b>0.478</b>	0.472	<b>0.460</b>
	720	0.940	<b>0.897</b>	0.606	<b>0.554</b>	0.530	<b>0.517</b>	0.593	<b>0.553</b>	0.597	<b>0.539</b>	0.526	<b>0.514</b>	0.599	<b>0.535</b>	0.598	<b>0.541</b>	0.525	<b>0.512</b>
ETTh2	96	0.241	<b>0.240</b>	0.263	<b>0.259</b>	0.243	<b>0.242</b>	0.230	<b>0.228</b>	0.232	<b>0.231</b>	0.228	<b>0.228</b>	0.244	<b>0.241</b>	0.247	<b>0.246</b>	0.239	<b>0.239</b>
	192	0.295	<b>0.292</b>	0.308	<b>0.335</b>	0.303	<b>0.300</b>	0.284	<b>0.279</b>	0.286	<b>0.276</b>	0.277	<b>0.276</b>	0.290	<b>0.283</b>	0.304	<b>0.288</b>	0.282	<b>0.281</b>
	336	0.333	<b>0.325</b>	0.354	<b>0.367</b>	0.336	<b>0.328</b>	0.325	<b>0.314</b>	0.328	<b>0.298</b>	0.308	<b>0.305</b>	0.333	<b>0.318</b>	0.336	<b>0.303</b>	0.307	<b>0.305</b>
	720	0.415	<b>0.392</b>	0.469	<b>0.424</b>	0.434	<b>0.405</b>	0.409	<b>0.383</b>	0.424	<b>0.358</b>	0.381	<b>0.364</b>	0.422	<b>0.384</b>	0.435	<b>0.366</b>	0.368	<b>0.356</b>
ETTm2	96	0.179	<b>0.178</b>	0.165	<b>0.162</b>	0.156	<b>0.156</b>	0.160	<b>0.159</b>	0.160	<b>0.159</b>	0.161	<b>0.155</b>	0.173	<b>0.154</b>	0.159	<b>0.158</b>	0.161	<b>0.155</b>
	192	0.204	<b>0.202</b>	0.196	<b>0.201</b>	0.190	<b>0.189</b>	0.193	<b>0.192</b>	0.195	<b>0.193</b>	0.197	<b>0.190</b>	0.210	<b>0.189</b>	0.193	0.196	0.201	<b>0.192</b>
	336	0.245	<b>0.242</b>	0.258	<b>0.257</b>	0.228	<b>0.225</b>	0.232	<b>0.232</b>	0.234	<b>0.233</b>	0.237	<b>0.229</b>	0.247	<b>0.228</b>	0.235	0.241	0.238	<b>0.230</b>
	720	0.316	<b>0.309</b>	0.315	<b>0.328</b>	0.299	<b>0.292</b>	0.306	<b>0.301</b>	0.306	<b>0.298</b>	0.296	<b>0.291</b>	0.306	<b>0.297</b>	0.312	<b>0.308</b>	0.297	<b>0.289</b>
Exchange	96	0.086	<b>0.084</b>	0.091	0.119	0.079	<b>0.078</b>	0.081	<b>0.078</b>	0.081	<b>0.076</b>	0.079	<b>0.078</b>	0.084	<b>0.079</b>	0.084	<b>0.080</b>	0.079	<b>0.078</b>
	192	0.175	<b>0.164</b>	0.199	0.310	0.161	<b>0.155</b>	0.169	<b>0.164</b>	0.167	<b>0.149</b>	0.163	<b>0.158</b>	0.177	<b>0.164</b>	0.187	<b>0.162</b>	0.161	<b>0.158</b>
	336	0.329	<b>0.282</b>	0.366	0.546	0.307	<b>0.275</b>	0.317	<b>0.293</b>	0.307	<b>0.254</b>	0.300	<b>0.275</b>	0.328	<b>0.302</b>	0.336	<b>0.295</b>	0.298	<b>0.276</b>
	720	0.844	<b>0.557</b>	0.919	1.529	1.110	<b>0.645</b>	0.834	<b>0.815</b>	0.929	<b>0.546</b>	0.845	<b>0.802</b>	0.837	<b>0.738</b>	0.812	<b>0.675</b>	0.846	<b>0.704</b>
Illness	24	<b>2.118</b>	2.121	2.765	<b>2.571</b>	<b>2.587</b>	2.589	2.654	<b>2.613</b>	<b>2.778</b>	2.865	2.460	<b>2.423</b>	2.480	<b>2.438</b>	2.577	<b>2.542</b>	2.536	<b>2.500</b>
	36	1.989	<b>1.984</b>	2.701	<b>2.494</b>	2.492	<b>2.465</b>	2.503	<b>2.441</b>	<b>2.606</b>	2.702	2.513	<b>2.472</b>	2.420	<b>2.386</b>	<b>2.530</b>	2.661	2.477	<b>2.431</b>
	48	2.183	<b>2.132</b>	2.527	<b>2.409</b>	2.386	<b>2.292</b>	2.487	<b>2.406</b>	<b>2.525</b>	2.702	2.443	<b>2.392</b>	2.247	<b>2.153</b>	<b>2.246</b>	2.434	2.416	<b>2.340</b>
	60	2.030	<b>2.029</b>	3.372	<b>2.738</b>	2.363	<b>2.298</b>	2.529	<b>2.452</b>	<b>2.549</b>	2.785	2.423	<b>2.383</b>	2.081	<b>1.997</b>	<b>2.068</b>	2.453	2.408	<b>2.354</b>
Weather	96	0.205	<b>0.199</b>	0.183	<b>0.164</b>	0.168	<b>0.163</b>	0.198	<b>0.184</b>	0.195	<b>0.178</b>	0.171	<b>0.165</b>	0.195	<b>0.165</b>	0.193	<b>0.166</b>	0.169	<b>0.164</b>
	192	0.256	<b>0.246</b>	0.231	<b>0.209</b>	0.213	<b>0.203</b>	0.243	<b>0.225</b>	0.240	<b>0.219</b>	0.214	<b>0.207</b>	0.251	<b>0.204</b>	0.240	<b>0.206</b>	0.212	<b>0.204</b>
	336	0.306	<b>0.286</b>	0.286	<b>0.257</b>	0.266	<b>0.253</b>	0.295	<b>0.269</b>	0.292	<b>0.267</b>	0.269	<b>0.258</b>	0.304	<b>0.253</b>	0.293	<b>0.253</b>	0.266	<b>0.256</b>
	720	0.376	<b>0.356</b>	0.364	<b>0.326</b>	0.340	<b>0.325</b>	0.367	<b>0.342</b>	0.366	<b>0.337</b>	0.342	<b>0.335</b>	0.379	<b>0.331</b>	0.367	<b>0.321</b>	0.338	<b>0.328</b>

longer forecasting window ( $H = 720$ ), the performance improvement brought upon by TAFAS reaches **5.76%** and **6.30%**. These results indicate that TAFAS is particularly advantageous in long-term forecasting scenarios with more severe distribution shifts.

To further demonstrate the effectiveness of TAFAS at mitigating extreme test-time distribution shifts in long-term forecasting scenarios, we verify TAFAS under the forecasting lengths of  $H \in \{780, 840, 900\}$  on DLinear. In Appendix, we plot the percentage of improvement in MSE achieved by TAFAS. Applying TAFAS exhibits a noticeable jump in performance improvement when compared to  $H = 336$ ; we note that on the ETTh2 dataset, the degree of improvement increases by more than 8%.

### Compatibility with Methods Addressing Non-stationarity in Pre-training time

One of the mainstream approaches to mitigating non-stationarity in TSF is to employ normalization and denormalization modules (Kim et al. 2021; Fan et al. 2023; Liu et al. 2024). However, they address non-stationarity only in the pre-training stage using training distributions, and thus, they may not generalize to consistently changing test distributions. As TAFAS is pluggable to any source fore-

casters in test-time, this compatibility can further enhance the robustness of source forecasters pre-trained with these widely adopted normalization modules. Table 2 presents the results of applying TAFAS on source forecasters equipped with RevIN (Kim et al. 2021), Dish-TS (Fan et al. 2023), or SAN (Liu et al. 2024). Across all datasets and architectures, TAFAS further improves the forecasting capability of these advanced source forecasters. The strength of TAFAS in long-range time series forecasting is again demonstrated here. For iTransformer with  $H = 720$ , TAFAS reduces the MSE of RevIN and SAN by 8.90% and 9.39% on average. Likewise, for DLinear with  $H = 720$ , TAFAS improves RevIN and SAN by 4.45% and 2.72% on average.

Interestingly, the normalization approaches significantly increase the test MSE of the source forecaster in some experimental settings, e.g., from 0.367 to 1.071 in FreTS + RevIN on ETTm1 with  $H = 96$ . This observation highlights that addressing non-stationarity only in the pre-training phase can fail to generalize on the changing test distributions. In the above-mentioned setting, TAFAS improves the performance of FreTS + RevIN by **66.39%**, indicating that it can overcome this limitation of pre-training-based approaches.

Table 3: Test MSE with and without TAFAS on Chronos, a foundation model pre-trained on a massive corpus of time series data. We report test MSE for  $H = 96$ .

Models	TAFAS	ETTh1	ETTm1	ETTh2	ETTm2
Chronos-small	$\times$	0.795	1.317	0.904	0.609
	$\checkmark$	<b>0.624</b>	<b>0.858</b>	<b>0.611</b>	<b>0.492</b>
Chronos-base	$\times$	0.770	1.385	1.357	0.934
	$\checkmark$	<b>0.611</b>	<b>0.761</b>	<b>0.708</b>	<b>0.668</b>
Chronos-large	$\times$	0.838	1.397	0.485	0.459
	$\checkmark$	<b>0.635</b>	<b>0.772</b>	<b>0.477</b>	<b>0.431</b>

Table 4: Comparison of test MSE with the state-of-the-art online TSF methods: FSNet and OneNet for  $H = 720$ .

	ETTh1	ETTm1	ETTh2	ETTm2	Exchange
FSNet	0.615	1.641	0.788	0.431	1.464
OneNet	<b>0.620</b>	1.593	0.543	0.410	0.977
TAFAS	0.640	<b>0.512</b>	<b>0.356</b>	<b>0.289</b>	<b>0.704</b>

## Unleashing the Knowledge of Foundation Models

Recently, TSF foundation models pre-trained on up to billions of time steps (Garza and Mergenthaler-Cansceto 2023; Das et al. 2024; Ansari et al. 2024; Goswami et al. 2024) have shown promising forecasting performance. Here, we demonstrate that TAFAS can further improve the performance of such a powerful source forecaster. According to Table 3, TAFAS significantly improves the test MSE of Chronos (Ansari et al. 2024) on ETT datasets by up to 45%. The performance improvement is observed consistently on varying model sizes: small, base, and large. We highlight that ETT datasets were not included in pre-training data, demonstrating that TAFAS effectively adapts TSF foundation models to unseen time series data streams. The compatibility of TAFAS with foundation models alludes that it can adapt their predictions effectively without overwriting the rich semantic information encoded in these models.

## Comparison with Online TSF Methods

As mentioned in **Related Works**, online TSF is another line of research that leverages sequentially arriving data to update forecasters, but it differs from of TSF-TTA in that online TSF trains forecasters from scratch, whereas TSF-TTA adapts a pre-trained source forecaster. Here, we compare TAFAS with online TSF methods to corroborate that TAFAS has tangible advantages over online TSF. Table 4 presents test MSE of the state-of-the-art online TSF methods: FSNet (Pham et al. 2023) and OneNet (Wen et al. 2024), in the long-term forecasting scenario of  $H = 720$ . In most experimental settings, TAFAS significantly outperforms online TSF methods. The empirical superiority of TAFAS to online TSF can be attributed to the following factors: 1) proactive adaptation of forecaster using periodicity-aware POGT, 2) the timely adjustment of predictions to reflect adjacent distribution shifts, and 3) preservation of the knowledge in the source forecaster through the use of auxiliary non-stationarity-aware GCM modules.

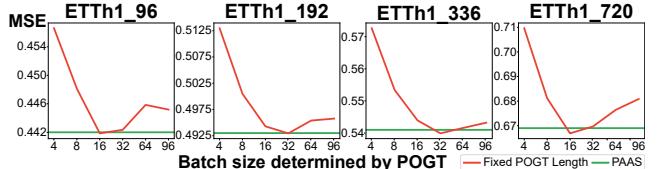


Figure 3: Comparison of PAAS against using a fixed POGT length on the ETTh1 dataset.

Table 5: Forecasting errors as different modules are adapted in iTransformer. “Norm.” denotes the layer normalization.

Modules to Adapt	ETTh1		ETTh2	
	MSE	MAE	MSE	MAE
None (baseline)	0.786	0.657	0.415	0.441
Norm.	0.776	0.657	0.409	0.436
Encoder	0.840	0.678	0.414	0.438
Decoder	0.808	0.664	0.410	0.436
All	0.832	0.673	0.414	0.438
GCM (TAFAS)	<b>0.704</b>	<b>0.627</b>	<b>0.393</b>	<b>0.425</b>

## Analysis of Each Technical Component in TAFAS

We explore alternative design choices for two main technical components in TAFAS. First, we study the effect of replacing PAAS with a fixed POGT length. Even though experiments are conducted with DLinear on all seven datasets, due to the page limit, Figure 3 only shows the results on the ETTh1 dataset. Extended results and the range of POGT lengths computed by PAAS are in Appendix. Figure 3 shows that too short or long POGT curtails the effect of TAFAS, supporting the motivation behind dynamically adjusting its length. Second, we demonstrate the effectiveness of introducing GCM. Table 5 presents forecasting errors for iTransformer with  $H = 720$  when other modules inside iTransformer are adapted. Modifying internal modules results in reduced performance improvement. In some cases, the forecasting performance drops below the baseline, likely due to the overwriting of core semantics in the source forecaster. The effectiveness of each component in TAFAS is shown through ablation studies in Appendix. Lastly, to validate that TAFAS can be deployed at test-time without significant hyper-parameter tuning, we demonstrate its robustness to changes in hyper-parameters in Appendix.

## Conclusion

To address the ever-changing distributions in non-stationary time series, we propose TAFAS, a pioneering TSF-TTA framework that dynamically adapts the source forecaster at test-time while preserving the knowledge of the source forecaster. Using PAAS, the partially-observed ground truth with semantically meaningful patterns is acquired for proactive adaptation. Then GCM, which considers both local and global temporal distribution shifts is adapted to address changing distributions. TAFAS serves as a dataset- and model-agnostic framework, demonstrated by thorough experimental results and analyses. The TSF-TTA framework pioneered in our work paves a new avenue toward sustainable deployment of state-of-the-art time series forecasters.

## Acknowledgments

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [No.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University)], the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2022R1A3B1077720, 2022R1A5A708390811), the BK21 FOUR program of the Education and the Research Program for Future ICT Pioneers, Seoul National University in 2024, Hyundai Motor Company, and Samsung Electronics Co., Ltd (IO240124-08661-01).

## References

- Ansari, A. F.; Stella, L.; Turkmen, C.; Zhang, X.; Mercado, P.; Shen, H.; Shchur, O.; Rangapuram, S. S.; Arango, S. P.; Kapoor, S.; et al. 2024. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*.
- Ao, S.-I.; and Fayek, H. 2023. Continual Deep Learning for Time Series Modeling. *Sensors*, 23(16): 7167.
- Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer Normalization. <https://arxiv.org/pdf/1607.06450.pdf>.
- Challu, C.; Olivares, K. G.; Oreshkin, B. N.; Garza, F.; Mergenthaler, M.; and Dubrawski, A. 2022. N-HiTS: Neural Hierarchical Interpolation for Time Series Forecasting. *arXiv preprint arXiv:2201.12886*.
- Das, A.; Kong, W.; Leach, A.; Mathur, S.; Sen, R.; and Yu, R. 2023. Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint arXiv:2304.08424*.
- Das, A.; Kong, W.; Sen, R.; and Zhou, Y. 2024. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*.
- Ekambaram, V.; Jati, A.; Nguyen, N.; Sinthong, P.; and Kalagnanam, J. 2023. Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 459–469.
- Elliott, G.; Rothenberg, T. J.; and Stock, J. H. 1992. Efficient tests for an autoregressive unit root.
- Fan, W.; Wang, P.; Wang, D.; Wang, D.; Zhou, Y.; and Fu, Y. 2023. Dish-ts: a general paradigm for alleviating distribution shift in time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37(6), 7522–7529.
- Garza, A.; and Mergenthaler-Canseco, M. 2023. TimeGPT-1. *arXiv preprint arXiv:2310.03589*.
- Girard, A.; Rasmussen, C.; Candela, J. Q.; and Murray-Smith, R. 2002. Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting. *Advances in neural information processing systems*, 15.
- Gong, T.; Jeong, J.; Kim, T.; Kim, Y.; Shin, J.; and Lee, S.-J. 2022. NOTE: Robust Continual Test-time Adaptation Against Temporal Correlation. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Goswami, M.; Szafer, K.; Choudhry, A.; Cai, Y.; Li, S.; and Dubrawski, A. 2024. MOMENT: A Family of Open Time-series Foundation Models. In *International Conference on Machine Learning*.
- Guo, T.; Xu, Z.; Yao, X.; Chen, H.; Aberer, K.; and Funaya, K. 2016. Robust online time series prediction with recurrent neural networks. In *2016 IEEE international conference on data science and advanced analytics (DSAA)*, 816–825. Ieee.
- Hosseini Shavaki, F.; and Ebrahimi Ghahnavieh, A. 2023. Applications of deep learning into supply chain management: a systematic literature review and a framework for future research. *Artificial Intelligence Review*, 56(5): 4447–4489.
- Hyndman, R. J.; and Athanasopoulos, G. 2018. *Forecasting: principles and practice*. OTexts.
- Ioffe, S.; and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, 448–456. pmlr.
- Kim, T.; Kim, J.; Tae, Y.; Park, C.; Choi, J.-H.; and Choo, J. 2021. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*.
- Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Kitaev, N.; Kaiser, L.; and Levskaya, A. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.
- Kuznetsov, V.; and Mohri, M. 2014. Generalization bounds for time series prediction with non-stationary processes. In *Algorithmic Learning Theory: 25th International Conference, ALT 2014, Bled, Slovenia, October 8-10, 2014. Proceedings* 25, 260–274. Springer.
- Lee, J.; Jung, D.; Lee, S.; Park, J.; Shin, J.; Hwang, U.; and Yoon, S. 2024. Entropy is not Enough for Test-Time Adaptation: From the Perspective of Disentangled Factors. In *The Twelfth International Conference on Learning Representations*.
- Li, M.; Zhu, Y.; Shen, Y.; and Angelova, M. 2023a. Clustering-enhanced stock price prediction using deep learning. *World wide web*, 26(1): 207–232.
- Li, S.; Jin, X.; Xuan, Y.; Zhou, X.; Chen, W.; Wang, Y.-X.; and Yan, X. 2019. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems*, 32.
- Li, Z.; Qi, S.; Li, Y.; and Xu, Z. 2023b. Revisiting long-term time series forecasting: An investigation on linear mapping. *arXiv preprint arXiv:2305.10721*.
- Liu, H.; Dong, Z.; Jiang, R.; Deng, J.; Deng, J.; Chen, Q.; and Song, X. 2023a. Spatio-temporal adaptive embedding makes vanilla transformer sota for traffic forecasting. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, 4125–4129.
- Liu, S.; Yu, H.; Liao, C.; Li, J.; Lin, W.; Liu, A. X.; and Dustdar, S. 2021. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. *International conference on learning representations*.

- Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; and Long, M. 2023b. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*.
- Liu, Y.; Wu, H.; Wang, J.; and Long, M. 2022. Non-stationary Transformers: Rethinking the Stationarity in Time Series Forecasting. *NeurIPS*.
- Liu, Z.; Cheng, M.; Li, Z.; Huang, Z.; Liu, Q.; Xie, Y.; and Chen, E. 2024. Adaptive normalization for non-stationary time series forecasting: A temporal slice perspective. *Advances in Neural Information Processing Systems*, 36.
- Loshchilov, I.; and Hutter, F. 2016. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.
- Masini, R. P.; Medeiros, M. C.; and Mendes, E. F. 2023. Machine learning advances for time series forecasting. *Journal of economic surveys*, 37(1): 76–111.
- Nelson, B. K. 1998. Time series analysis using autoregressive integrated moving average (ARIMA) models. *Academic emergency medicine*, 5(7): 739–744.
- Nie, Y.; Nguyen, N. H.; Sindhong, P.; and Kalagnanam, J. 2022. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*.
- Niu, S.; Wu, J.; Zhang, Y.; Chen, Y.; Zheng, S.; Zhao, P.; and Tan, M. 2022. Efficient Test-Time Model Adaptation without Forgetting. In Chaudhuri, K.; Jegelka, S.; Song, L.; Szepesvari, C.; Niu, G.; and Sabato, S., eds., *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 16888–16905. PMLR.
- Niu, S.; Wu, J.; Zhang, Y.; Wen, Z.; Chen, Y.; Zhao, P.; and Tan, M. 2023. Towards Stable Test-time Adaptation in Dynamic Wild World. In *The Eleventh International Conference on Learning Representations*.
- Petropoulos, F.; Apiletti, D.; Assimakopoulos, V.; Babai, M. Z.; Barrow, D. K.; Taieb, S. B.; Bergmeir, C.; Bessa, R. J.; Bijak, J.; Boylan, J. E.; et al. 2022. Forecasting: theory and practice. *International Journal of Forecasting*, 38(3): 705–871.
- Pham, Q.; Liu, C.; Sahoo, D.; and Hoi, S. 2023. Learning Fast and Slow for Online Time Series Forecasting. In *The Eleventh International Conference on Learning Representations*.
- Toner, W.; and Darlow, L. 2024. An Analysis of Linear Time Series Forecasting Models. *arXiv preprint arXiv:2403.14587*.
- Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.
- Verma, Y.; Heinonen, M.; and Garg, V. 2024. ClimODE: Climate and Weather Forecasting with Physics-informed Neural ODEs. In *The Twelfth International Conference on Learning Representations*.
- Wang, D.; Shelhamer, E.; Liu, S.; Olshausen, B.; and Darrell, T. 2021. Tent: Fully Test-Time Adaptation by Entropy Minimization. In *International Conference on Learning Representations*.
- Wang, H.; Peng, J.; Huang, F.; Wang, J.; Chen, J.; and Xiao, Y. 2023. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *The eleventh international conference on learning representations*.
- Wang, S.; Wu, H.; Shi, X.; Hu, T.; Luo, H.; Ma, L.; Zhang, J. Y.; and ZHOU, J. 2024. TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting. In *The Twelfth International Conference on Learning Representations*.
- Wen, Q.; Chen, W.; Sun, L.; Zhang, Z.; Wang, L.; Jin, R.; Tan, T.; et al. 2024. Onenet: Enhancing time series forecasting models under concept drift by online ensembling. *Advances in Neural Information Processing Systems*, 36.
- Wen, Q.; Zhou, T.; Zhang, C.; Chen, W.; Ma, Z.; Yan, J.; and Sun, L. 2022. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*.
- Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; and Long, M. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. *ICLR*.
- Wu, H.; Wu, J.; Xu, J.; Wang, J.; and Long, M. 2022. Flowformer: Linearizing transformers with conservation flows. *ICML*.
- Wu, H.; Xu, J.; Wang, J.; and Long, M. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34: 22419–22430.
- Xu, Z.; Zeng, A.; and Xu, Q. 2024. FITS: Modeling Time Series with \$10k\$ Parameters. In *The Twelfth International Conference on Learning Representations*.
- Yi, K.; Zhang, Q.; Fan, W.; Wang, S.; Wang, P.; He, H.; An, N.; Lian, D.; Cao, L.; and Niu, Z. 2024. Frequency-domain MLPs are more effective learners in time series forecasting. *Advances in Neural Information Processing Systems*, 36.
- Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2023. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37(9), 11121–11128.
- Zhang, Y.; and Yan, J. 2022. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*.
- Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35(12), 11106–11115.
- Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; and Jin, R. 2022. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, 27268–27286. PMLR.

## Appendix

### Algorithm of TAFAS

Algorithm 1 summarizes the overall pipeline of TAFAS.

### Detailed Explanations on Datasets

Table A1 shows the characteristics of the seven widely used public TSF benchmark datasets used throughout experiments. (Train, Validation, Test) shows the number of time steps in the train, validation, and test set, respectively. ADF shows the results of the Augmented Dickey-Fuller Test (Elliott, Rothenberg, and Stock 1992), which quantify the non-stationarity of each dataset. A higher ADF test statistic suggests greater non-stationarity, indicating prevalent distribution shifts.

Table A1: Characteristics of the seven widely used public TSF benchmark datasets.

Dataset	Var.	Time Steps	Sampling Rate	ADF
ETTh1	7	17420	1 hour	-5.91
ETTm1	7	69680	1 hour	-14.98
ETTh2	7	17420	15 minutes	-4.13
ETTm2	7	69680	15 minutes	-5.66
Exchange	8	7588	1 day	-1.90
Illness	7	966	1 week	-5.33
Weather	21	52696	10 minutes	-26.68

### Training Details

A hyper-parameter search was conducted for each source forecaster, exploring learning rates of 1e-3, 1e-4, and 1e-5, and weight decay values of 0.0, 1e-3, and 1e-4. For each combination, models were trained using three different seeds. The hyper-parameter combination that resulted in the lowest validation MSE loss was selected for use. Pre-training was performed for 30 epochs with a batch size of 64, ensuring that the validation MSE loss had sufficiently saturated. We employed the Adam optimizer (Kingma and Ba 2014) and adjusted the learning rate using cosine learning rate scheduling (Loshchilov and Hutter 2016). All experiments were conducted using a single NVIDIA A40 GPU. We referred the configuration (e.g., number of layers) of each source forecaster to the Time Series Library (Wu et al. 2023). When incorporating the normalization-based approaches (RevIN (Kim et al. 2021), Dish-TS (Fan et al. 2023), and SAN (Liu et al. 2024)) for pre-training the source forecaster, the hyper-parameter searches for RevIN and Dish-TS were conducted in the same manner as for training the baseline source forecaster. For SAN, a two-stage pre-training process is required, involving the training of the statistics prediction module. According to the settings in the original paper, the statistics prediction module was trained for 10 epochs, with a hyper-parameter search for learning rates of 1e-3 and 1e-4. When performing TSF-TTA with TAFAS, the learning rate was searched among 5e-3, 3e-3, 1e-3, 5e-4, and 1e-4. The initial value of the gating parameter  $\alpha$  was searched among 0.01, 0.05, 0.1, and 0.3. For reproducibility, we will make our code publicly available in the final version.

Algorithm 1: PyTorch-style Pseudocode for TAFAS

```

1 # L: look-back window length
2 # forecaster: source forecaster
3 # in_GCM: input GCM
4 # out_GCM: output GCM
5
6 forecaster.requires_grad_(False)
7 do_PAAS = True
8 test_batch = []
9
10 for test_input in test_loader:
11     # PAAS
12     if do_PAAS:
13         period = PAAS(test_input)
14         bsz = 0
15         do_PAAS = False
16
17     # get mini-batch based on PAAS
18     if bsz < period + 1:
19         test_batch.append(test_input)
20         bsz += 1
21     if bsz == period + 1:
22         do_PAAS = True
23     else:
24         continue
25
26     # get POGT
27     test_batch = torch.stack(test_batch)
28     POGT = test_batch[-1] [-:period]
29
30     # adapt GCM
31     input_cali = in_GCM(test_input)
32     pred = forecaster(input_cali)
33     pred_cali = out_GCM(pred)
34     l_p = MSE(pred_cali[:period], POGT)
35     if full_gt_available:
36         l_f = MSE(full_pred, full_GT)
37     else:
38         l_f = 0.0
39     l_tafas = l_p + l_f
40     l_tafas.backward()
41     optimizer.zero_grad()
42     optimizer.step()
43
44     # PA
45     with torch.no_grad():
46         pred_adapted = out_GCM(forecaster(
47             in_GCM(test_batch)))
48         for i in range(bsz - 1):
49             pred_cali[i, period-i:] =
50                 pred_adapted[i, period-i:]
51
52         test_batch = []
53
54     def PAAS(test_input):
55         # test_input: (L, C)
56         test_input -= test_input.mean(dim=0)
57         amplitude = abs(fft(test_input))
58         var_idx = (amplitude ** 2).sum(dim=0)
59             .argmax()
60         freq = amplitude[:, var_idx].argmax()
61         p = test_input.shape[0] // freq
62
63     return p

```

## Additional Related Works

### Time Series Forecasting Models

As time series forecasting has become a pivotal application in various industries, diverse time series forecasting architectures have been developed to accurately predict future time steps (Wen et al. 2022; Masini, Medeiros, and Mendes 2023). They range from traditional statistical methods (Nelson 1998; Girard et al. 2002; Hyndman and Athanasopoulos 2018) to deep neural network-based architectures, including Transformers (Vaswani et al. 2017; Zhou et al. 2021; Li et al. 2019; Liu et al. 2021, 2023b; Wu et al. 2021; Nie et al. 2022; Zhang and Yan 2022; Kitaev, Kaiser, and Levskaya 2020; Zhou et al. 2022; Wu et al. 2022), Linear layers (Zeng et al. 2023; Li et al. 2023b; Toner and Darlow 2024), and MLPs (Yi et al. 2024; Wang et al. 2023; Ekambaram et al. 2023; Das et al. 2023; Xu, Zeng, and Xu 2024; Wang et al. 2024; Challu et al. 2022). Transformer-based models aim to capture temporal dependencies as well as inter-variable dependencies utilizing the attention mechanism. To address the quadratic time and memory complexity of self-attention operation, a line of work has been proposed to modify the self-attention module to be more efficient, facilitating long-term forecasting (Zhou et al. 2021; Kitaev, Kaiser, and Levskaya 2020; Li et al. 2019). Some works have revised the Transformer architecture to better exploit the properties of time series, such as sub-series periodicity or frequency information (Wu et al. 2021; Zhou et al. 2022). Other works remain the architecture untouched, but consider how to input time series by patching or inverting (Nie et al. 2022; Liu et al. 2023b). On the other hand, in response to a recent work that raised questions to the modeling capability of Transformer-based TSF models (Zeng et al. 2023), a series of Linear (Zeng et al. 2023; Li et al. 2023b; Toner and Darlow 2024) and MLP-based architectures (Yi et al. 2024; Wang et al. 2023; Ekambaram et al. 2023; Das et al. 2023; Xu, Zeng, and Xu 2024; Wang et al. 2024; Challu et al. 2022) have been developed, achieving comparable or outperforming TSF capabilities compared to Transformer-based models. Still, there is no consensus on the most representative TSF architecture, making the model-agnosticism of the TSF-TTA framework more desirable. More recently, time series foundation models, pre-trained on up to billions of time steps, have shown promising forecasting capabilities (Ansari et al. 2024; Das et al. 2024; Goswami et al. 2024; Garza and Mergenthaler-Canseco 2023). The emergence of foundation models highlighted exploiting rich semantic information encoded in pre-trained models on unseen data streams.

### Test-Time Adaptation (TTA)

The goal of test-time adaptation is to adapt a pre-trained source model to distributionally shifted inputs encountered during testing, thereby improving generalization performance on unseen test distributions. Test-time adaptation has primarily evolved in classification tasks, leveraging the characteristic components of classification tasks (Wang et al. 2021; Niu et al. 2022, 2023; Lee et al. 2024; Gong et al. 2022). TTA methods commonly rely on the entropy of

class probabilities, which is only applicable to classification tasks and infeasible to TSF, which is a regression task. TENT (Wang et al. 2021) minimizes the entropy of the target class probability by updating the channel-wise affine transformation of the normalization layer. EATA (Niu et al. 2022) builds on the entropy minimization framework by proposing a sample filtering method for sample-efficient entropy minimization. SAR (Niu et al. 2023) considers additional scenarios in classification tasks, such as label imbalance, and proposes sharpness-aware entropy minimization. DEYO (Lee et al. 2024) combines entropy minimization with image-specific data augmentation, utilizing pseudo label probability for the transformed test input. Moreover, they do not consider the properties of time series data, such as instance-wise distribution shifts within a window input and the global temporal dependency across streaming window data. Although NOTE (Gong et al. 2022) considers the temporal correlation between streaming images, it still relies on an entropy-based approach and does not account for the distribution shifts that can occur within a single instance window in time series data. Additionally, they are not fully model-agnostic because they adapt specific types of normalization layers (e.g., Batch Norm (Ioffe and Szegedy 2015) or Layer Norm (Ba, Kiros, and Hinton 2016)) and are inapplicable to models without normalization layers. Given that TSF has been developed based on various architectural advances, model dependence significantly limits the generality of TSF-TTA framework.

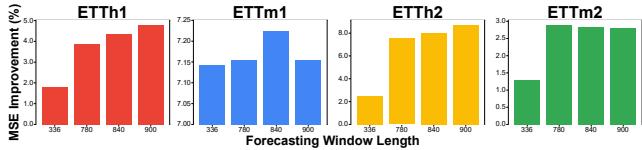


Figure A1: Improvements of MSE (%) as the forecasting window length increases.

Table A2: Comparison of average test MSE between TAFAS and baseline models. Both TAFAS and the baselines are adapted using the same input with sequentially arriving test data. Baselines include DLinear and DLinear pre-trained with Dish-TS.

	ETTh1	ETTm1	ETTh2	ETTm2	Exchange
DLinear	0.553	0.476	0.313	0.222	0.350
+TAFAS	0.538	0.450	0.304	0.219	0.337
DLinear w/ Dish-TS	0.550	0.452	0.308	0.234	0.300
+TAFAS	0.525	0.446	0.291	0.221	0.256

## Additional Experimental Results

Figure A1 presents the effectiveness of TAFAS in long-term forecasting scenarios that we stated in the Experiments Section in the manuscript.

### Comparison with Baselines using POGT

To further verify the effectiveness of TAFAS, we have conducted experiments where the baselines are also adapted using the same input as TAFAS. The baselines include DLinear and DLinear with Dish-TS. All results are reported in terms of MSE averaged over  $H \in \{96, 192, 336, 720\}$ . According to Table A2, the baselines fall short of TAFAS even when they use POGT for further training. This demonstrates that TAFAS consistently outperforms the baselines by effectively leveraging sequentially arriving test data.

### Hyperparameter Robustness Analysis

To validate that TAFAS can be deployed at test-time without significant hyper-parameter tuning, we show its robustness to changes in hyper-parameters. Table A3 presents test MSE and MAE for different combinations of the two hyper-parameters involved in TAFAS: the test-time learning rate  $\mu$  and initialization value for gating parameter  $\alpha$ . The experiments were conducted using DLinear on the ETTh1 dataset with  $H = 720$ . TAFAS constantly achieves low MSE and MAE across different parameter settings with consistently low standard deviations, indicating that it is reasonably robust against changes in hyper-parameters.

### Component-wise Ablation Studies

Table A4 presents ablative experiments aimed at demonstrating the necessity of various components of TAFAS for successfully performing TSF-TTA. When GCM is excluded, the entire source forecaster is adapted to maintain model-agnosticity of TSF-TTA. In the absence of PAAS,

Table A3: Forecasting errors for different  $(\mu, \alpha)$ .

$(\mu, \alpha)$	MSE	MAE
(1e-3, 0.01)	$0.669 \pm 0.002$	$0.600 \pm 0.002$
(1e-3, 0.05)	$0.669 \pm 0.001$	$0.598 \pm 0.000$
(1e-3, 0.1)	$0.691 \pm 0.001$	$0.609 \pm 0.001$
(5e-4, 0.01)	$0.682 \pm 0.000$	$0.602 \pm 0.000$
(5e-4, 0.05)	$0.671 \pm 0.001$	$0.599 \pm 0.000$
(5e-4, 0.1)	$0.670 \pm 0.001$	$0.599 \pm 0.000$
(1e-4, 0.05)	$0.683 \pm 0.001$	$0.601 \pm 0.001$
(1e-4, 0.1)	$0.676 \pm 0.001$	$0.599 \pm 0.000$
(1e-4, 0.3)	$0.669 \pm 0.001$	$0.599 \pm 0.000$

Table A4: Ablation study on each component of TAFAS. The “-” denotes the excluded component from TAFAS.

	Exchange	ETTh1	ETTh2			
	MSE	MAE	MSE	MAE	MSE	MAE
Baseline	0.844	0.692	0.786	0.662	0.415	0.441
- PAAS	0.795	0.672	0.740	0.645	0.393	0.425
TAFAS - GCM	0.979	0.756	1.320	0.839	0.491	0.480
- PA	0.809	0.679	0.706	0.628	0.392	0.425
TAFAS	<b>0.773</b>	<b>0.665</b>	<b>0.704</b>	<b>0.627</b>	<b>0.393</b>	<b>0.425</b>

a test batch size of 64 is arbitrarily selected. The experiments are conducted for iTransformer with  $H = 720$ . Each row in Table A4 indicates which component was removed for ablation; besides the removed component, the rest of TAFAS remains unchanged. The results show that removing each component of TAFAS subsequently degrades the performance of TAFAS, validating its effectiveness. Figure A3 illustrates the MSE values for each dataset when test batches are adaptively composed using PAAS compared to using fixed batch sizes. In most instances, PAAS consistently achieves superior performance. The variation in effective batch sizes across different datasets underscores the importance of identifying an optimal batch size for each dataset. PAAS demonstrates its capability to configure these batch sizes at test time effectively. Table A5 shows the range of test batch sizes derived through PAAS for each dataset.

The results show that introducing each component of TAFAS subsequently reduces the MSE, validating its effectiveness.

### Qualitative Analysis

Figure A2 shows the qualitative comparison of forecasting results with and without TAFAS for iTransformer. In the top row, the baseline produces repetitive patterns that resemble the latest periodic pattern within the look-back window. In contrast, TAFAS preemptively adapts the source forecaster to changing distributions and outputs significantly more accurate predictions. Furthermore, TAFAS effectively adapts to both low-frequency (top left) and high-frequency (top right) dominant patterns. The two panels on the bottom row highlight the particular advantages of TAFAS in more chal-

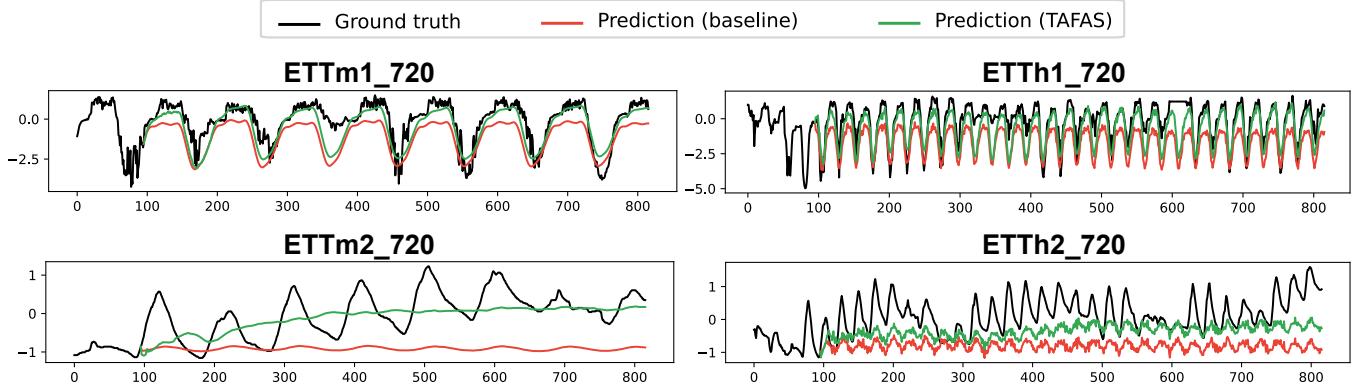


Figure A2: Visualization of forecasting results with and without TAFAS. The top row illustrates that TAFAS effectively adapts to both low-frequency (top left) and high-frequency (top right) dominant patterns within the look-back window. The bottom row highlights the promising aspect of TAFAS in significantly more challenging scenarios characterized by pronounced global distribution shifts.

Table A5: Range of POGT length  $p$  constructed via PAAS.

Dataset	Range of $p$
ETTh1	12-96
ETTm1	32-96
ETTh2	8-96
ETTm2	6-96
Exchange	48-96
Illness	36-36
Weather	2-96

lenging long-term forecasting scenarios where test data di-  
gress noticeably from historical data, resulting in a more ex-  
treme distribution shift. In the examples in these two panels,  
the time series in the prediction window visible differs from  
that in the look-back window, which is indicative of a global  
shift in distribution. Here, the baseline merely repeats the  
pattern within the look-back window, while TAFAS effec-  
tively captures the global distribution shift.

### MAE and Standard Deviations of TAFAS

Table A6 presents the MAE and standard deviations for the three different runs. The standard deviations were rounded to the fourth decimal place. When TAFAS is applied, most standard deviations are 0.000, demonstrating the stability of TAFAS in test-time adaptation.

Table A6: TestMAE and standard deviations on multivariate time series forecasting datasets with and without TAFAS across various TSF architectures: Transformer-based (iTTransformer (Liu et al. 2023b), PatchTST (Nie et al. 2022)), Linear-based (DLinear (Zeng et al. 2023), OLS (Toner and Darlow 2024)), and MLP-based (fReTS (Yi et al. 2024), MICN (Wang et al. 2023)). +TAFAS denotes whether the TAFAS framework is applied to the corresponding source forecaster.

Models + TAFAS	Transformer-based						Linear-based						MLP-based					
	iTTransformer			PatchTST			DLinear			OLS			fReTS			MICN		
	$\times$	$\checkmark$	$\times$	$\times$	$\checkmark$	$\times$	$\times$	$\checkmark$	$\times$	$\times$	$\checkmark$	$\times$	$\times$	$\checkmark$	$\times$	$\times$	$\checkmark$	
96	0.448 (0.000)	0.443 (0.000)	0.450 (0.001)	0.444 (0.000)	0.446 (0.000)	0.442 (0.000)	0.446 (0.000)	0.444 (0.000)	0.447 (0.000)	0.444 (0.000)	0.459 (0.000)	0.453 (0.000)						
192	0.494 (0.000)	0.489 (0.000)	0.483 (0.001)	0.489 (0.001)	0.483 (0.000)	0.480 (0.000)	0.483 (0.000)	0.482 (0.000)	0.486 (0.000)	0.483 (0.000)	0.503 (0.004)	0.497 (0.002)						
336	0.532 (0.001)	0.532 (0.000)	0.520 (0.001)	0.519 (0.001)	0.515 (0.000)	0.516 (0.000)	0.514 (0.000)	0.512 (0.000)	0.527 (0.000)	0.525 (0.000)	0.546 (0.002)	0.544 (0.001)						
720	0.662 (0.006)	0.627 (0.000)	0.621 (0.006)	0.621 (0.005)	0.605 (0.000)	0.598 (0.002)	0.605 (0.000)	0.599 (0.000)	0.619 (0.000)	0.610 (0.000)	0.642 (0.006)	0.633 (0.005)						
96	0.405 (0.003)	0.387 (0.001)	0.403 (0.002)	0.397 (0.001)	0.393 (0.000)	0.381 (0.000)	0.393 (0.000)	0.382 (0.000)	0.391 (0.000)	0.386 (0.001)	0.412 (0.009)	0.397 (0.004)						
192	0.438 (0.001)	0.430 (0.000)	0.435 (0.000)	0.428 (0.000)	0.428 (0.000)	0.416 (0.000)	0.429 (0.000)	0.416 (0.000)	0.426 (0.001)	0.423 (0.001)	0.431 (0.000)	0.423 (0.000)						
336	0.475 (0.002)	0.466 (0.001)	0.468 (0.000)	0.461 (0.000)	0.467 (0.000)	0.452 (0.000)	0.467 (0.000)	0.452 (0.000)	0.463 (0.001)	0.458 (0.000)	0.471 (0.001)	0.461 (0.001)						
720	0.523 (0.001)	0.511 (0.001)	0.517 (0.001)	0.509 (0.000)	0.515 (0.000)	0.497 (0.000)	0.515 (0.000)	0.499 (0.000)	0.510 (0.001)	0.504 (0.001)	0.532 (0.002)	0.516 (0.001)						
96	0.330 (0.001)	0.329 (0.001)	0.321 (0.001)	0.320 (0.001)	0.315 (0.000)	0.314 (0.000)	0.315 (0.000)	0.314 (0.000)	0.317 (0.000)	0.316 (0.000)	0.322 (0.000)	0.321 (0.001)	0.321 (0.001)	0.320 (0.001)				
192	0.365 (0.001)	0.362 (0.001)	0.357 (0.002)	0.353 (0.002)	0.352 (0.000)	0.350 (0.000)	0.352 (0.000)	0.349 (0.000)	0.359 (0.000)	0.355 (0.000)	0.357 (0.001)	0.354 (0.001)						
336	0.392 (0.001)	0.386 (0.001)	0.386 (0.002)	0.382 (0.001)	0.379 (0.000)	0.374 (0.000)	0.379 (0.000)	0.372 (0.000)	0.386 (0.000)	0.378 (0.000)	0.386 (0.004)	0.378 (0.001)						
720	0.441 (0.001)	0.425 (0.001)	0.438 (0.004)	0.427 (0.003)	0.433 (0.000)	0.417 (0.000)	0.434 (0.000)	0.412 (0.000)	0.441 (0.000)	0.419 (0.000)	0.434 (0.003)	0.421 (0.003)						
96	0.265 (0.000)	0.263 (0.001)	0.262 (0.000)	0.262 (0.000)	0.264 (0.000)	0.262 (0.000)	0.264 (0.000)	0.262 (0.000)	0.263 (0.000)	0.260 (0.001)	0.259 (0.000)	0.265 (0.000)	0.264 (0.000)					
192	0.296 (0.000)	0.292 (0.000)	0.295 (0.000)	0.294 (0.000)	0.290 (0.000)	0.289 (0.000)	0.291 (0.000)	0.290 (0.000)	0.291 (0.000)	0.290 (0.000)	0.293 (0.000)	0.291 (0.000)						
336	0.331 (0.001)	0.324 (0.000)	0.325 (0.001)	0.323 (0.000)	0.319 (0.000)	0.317 (0.000)	0.320 (0.000)	0.317 (0.000)	0.321 (0.000)	0.319 (0.000)	0.322 (0.000)	0.320 (0.000)						
720	0.373 (0.001)	0.366 (0.000)	0.371 (0.001)	0.367 (0.001)	0.365 (0.000)	0.359 (0.000)	0.365 (0.000)	0.359 (0.000)	0.366 (0.001)	0.359 (0.001)	0.366 (0.000)	0.361 (0.000)						
96	0.207 (0.000)	0.208 (0.000)	0.202 (0.001)	0.200 (0.001)	0.197 (0.000)	0.197 (0.000)	0.197 (0.000)	0.197 (0.000)	0.195 (0.000)	0.201 (0.000)	0.198 (0.000)	0.200 (0.000)	0.198 (0.000)					
192	0.298 (0.000)	0.293 (0.000)	0.301 (0.001)	0.296 (0.001)	0.292 (0.000)	0.289 (0.000)	0.293 (0.000)	0.289 (0.000)	0.298 (0.000)	0.297 (0.001)	0.291 (0.001)	0.293 (0.001)						
336	0.415 (0.001)	0.389 (0.003)	0.431 (0.010)	0.395 (0.005)	0.408 (0.000)	0.387 (0.001)	0.409 (0.000)	0.384 (0.000)	0.412 (0.000)	0.393 (0.002)	0.420 (0.004)	0.403 (0.005)						
720	0.692 (0.001)	0.665 (0.000)	0.691 (0.004)	0.690 (0.004)	0.688 (0.001)	0.684 (0.000)	0.687 (0.000)	0.577 (0.000)	0.689 (0.004)	0.668 (0.018)	0.839 (0.015)	0.726 (0.031)						
24	0.906 (0.008)	0.907 (0.008)	0.841 (0.003)	0.841 (0.003)	1.055 (0.002)	1.042 (0.002)	1.008 (0.002)	0.993 (0.000)	0.946 (0.002)	0.946 (0.002)	1.029 (0.002)	1.030 (0.002)						
36	0.922 (0.006)	0.922 (0.006)	0.860 (0.007)	0.860 (0.007)	1.026 (0.001)	1.001 (0.005)	1.000 (0.001)	0.948 (0.000)	0.961 (0.007)	0.961 (0.007)	1.062 (0.002)	1.063 (0.001)						
48	0.939 (0.001)	0.933 (0.002)	0.854 (0.018)	0.854 (0.018)	1.030 (0.000)	0.015 (0.000)	1.005 (0.002)	0.957 (0.000)	0.947 (0.004)	0.913 (0.004)	0.923 (0.004)	0.913 (0.005)						
60	0.895 (0.007)	0.872 (0.008)	0.862 (0.005)	0.862 (0.005)	1.046 (0.001)	1.033 (0.001)	1.029 (0.000)	0.975 (0.000)	0.934 (0.004)	0.915 (0.004)	0.914 (0.005)	0.906 (0.003)						
96	0.220 (0.002)	0.219 (0.002)	0.215 (0.001)	0.219 (0.001)	0.235 (0.000)	0.235 (0.000)	0.234 (0.000)	0.237 (0.000)	0.220 (0.000)	0.220 (0.000)	0.219 (0.001)	0.219 (0.000)						
192	0.260 (0.000)	0.257 (0.000)	0.257 (0.000)	0.256 (0.000)	0.270 (0.000)	0.267 (0.000)	0.270 (0.000)	0.273 (0.000)	0.259 (0.000)	0.261 (0.001)	0.262 (0.001)	0.264 (0.001)						
336	0.301 (0.000)	0.300 (0.000)	0.297 (0.000)	0.300 (0.000)	0.306 (0.000)	0.303 (0.000)	0.306 (0.000)	0.305 (0.000)	0.298 (0.000)	0.295 (0.000)	0.302 (0.001)	0.303 (0.001)						
720	0.352 (0.001)	0.352 (0.001)	0.346 (0.000)	0.346 (0.000)	0.356 (0.000)	0.353 (0.000)	0.348 (0.000)	0.353 (0.000)	0.347 (0.000)	0.344 (0.000)	0.346 (0.001)	0.356 (0.003)						

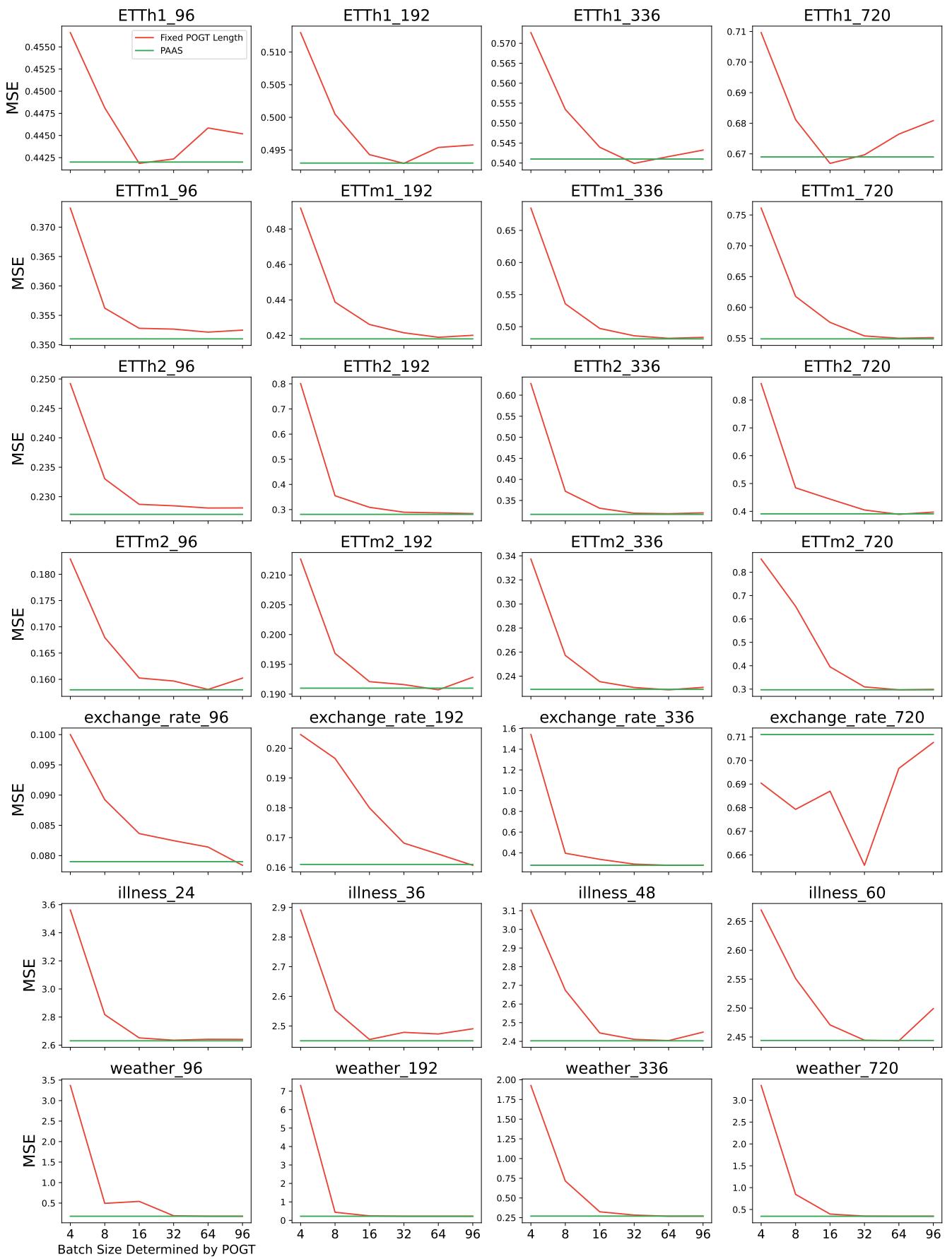


Figure A3: Comparison of test MSE when using PAAS against fixed length POGT.