From Tables to Time: Extending TabPFN-v2 to Time Series Forecasting

Anonymous authors
Paper under double-blind review

Abstract

Recent progress in foundation models has enabled strong zero-shot performance for time series forecasting. In this work, we show that such capabilities can also emerge from tabular foundation models. We introduce TabPFN-TS, a simple method that treats forecasting as a tabular regression problem by combining lightweight temporal featurization with the pretrained TabPFN-v2. This formulation requires no time-series–specific pretraining and naturally supports both univariate and covariate-informed forecasting. Despite its compact size (11M parameters), TabPFN-TS achieves state-of-the-art performance on covariate-informed forecasting and competitive accuracy on univariate forecasting across the GIFT-Eval and `fev-bench` benchmarks. We further provide controlled analyses examining how the model interprets temporal structure, how featurization choices affect accuracy, and how forecasts change under alternative tabular backbones. Together, our results demonstrate that tabular foundation models—when paired with suitable temporal features—offer an efficient and versatile alternative for forecasting, bridging tabular and time-series learning within a unified framework.

1 Introduction

Recent advances in foundation models have reshaped time-series forecasting by enabling zero-shot prediction across a wide variety of domains. Large pretrained architectures can now generalize to unseen series and horizons without task-specific fine-tuning, setting a new paradigm for forecasting research. Comprehensive benchmarks (Aksu et al., 2024; Shchur et al., 2025) have further standardized evaluation, enabling fair and reproducible comparison.

While most foundation models focus on univariate forecasting, real-world systems are often influenced by additional factors. Incorporating such covariates—e.g., weather, control inputs, or economic indicators—can make forecasts far more informative and actionable. Yet leveraging covariates remains challenging for current time-series foundation models, which are typically designed around purely sequential inputs. A unified and data-efficient framework capable of handling both univariate and covariate-informed forecasting would therefore be highly desirable.

In parallel, foundation models for *tabular learning* have made remarkable progress. Recent tabular models exhibit strong zero-shot predictive performance across diverse supervised tasks (Erickson et al., 2025). Among them, TabPFN-v2 (Hollmann et al., 2025) is notable as the first tabular foundation model, supporting many tasks, including classification, regression, outlier detection, density estimation, synthetic data generation, embeddings that are useful for downstream tasks, and fine-tunability. This raises a natural question: *Can a tabular foundation model also perform probabilistic time-series forecasting if the temporal structure is expressed through features?* If so, tabular models would offer a unified representation for both tabular and time series forecasting, sidestepping the need for specialized sequence architectures.

In this work, we propose to treat time series forecasting as a *tabular regression problem*. We introduce TabPFN-TS, a simple method that combines lightweight temporal featurization with the pretrained TabPFN-v2, enabling zero-shot forecasting without any time-series–specific pretraining or fine-tuning. Each time step is represented as a row of temporal features (and optional covariates) paired with its observed value.

Forecasting then reduces to predicting future rows—whose temporal features are known in advance—using TabPFN-v2 as a tabular regressor. This formulation allows all future points within the prediction horizon to be predicted in a single forward pass.

Despite its simplicity and compact size (11M parameters), TabPFN-TS achieves state-of-the-art performance on covariate-informed forecasting and competitive accuracy on univariate benchmark. Our analyses further show how the model interprets temporal structure, how featurization affects accuracy, and how forecasts change under different tabular backbones.

Our main contributions are:

1. **Forecasting as Tabular Regression.** We introduce a formulation that enables a pretrained tabular foundation model (TabPFN-v2) to perform both univariate and covariate-informed forecasting in a zero-shot manner.

2. **Lightweight Temporal Featurization.** We design a compact feature scheme that encodes time progression, multi-scale seasonality, and covariates, allowing tabular models to act on temporal data effectively.

3. **Strong Empirical Performance.** TabPFN-TS achieves state-of-the-art performance on covariate-aware forecasting and competitive accuracy on univariate forecasting across GIFT-Eval and `fev-bench`.

4. **Mechanistic and Ablation Studies.** We provide analyses explaining how TabPFN-TS exploits temporal structure, how featurization choices affect performance, and how forecasting quality varies across tabular foundation model backbones.

Taken together, these results demonstrate that tabular foundation models, paired with suitable temporal features, form a simple, efficient, and extensible alternative for general-purpose forecasting, bridging tabular and time-series learning within a unified framework.

2 Background and Related Work

Time Series Forecasting. The goal is to predict future observations of a sequence based on its historical context and, optionally, a set of covariates—external variables that provide additional information about the system's dynamics (e.g. holidays, known prices, or control inputs). Incorporating such signals can improve forecast accuracy by capturing influences beyond the intrinsic temporal patterns of the target series. Formally, the task can be expressed as modeling the conditional distribution

$$P(y_{C+1:H} \mid y_{1:C}, Z_{1:C+H}),$$

where $y_{1:C} = [y_1, \ldots, y_C]$ denotes the observed history of the target series, $y_{C+1:H} = [y_{C+1}, \ldots, y_H]$ are the future values to be predicted, and $Z_{1:C+H} = [\mathbf{z}_1, \ldots, \mathbf{z}_{C+H}]$ represents the associated covariates. When no covariates are provided, the task reduces to *univariate forecasting*; otherwise, it corresponds to *covariate-informed forecasting*.

Time Series Foundation Models. Time series foundation models (TSFMs) aim to provide zero-shot forecasting across domains by pretraining (potentially) large sequence models on broad collections of real-world time series. Early models such as Chronos (Ansari et al., 2024) and TimesFM (Das et al., 2024) adopt transformer architectures trained at scale for univariate forecasting. Chronos-Bolt improves efficiency beyond Chronos through patch-based encoding, while Toto (Cohen et al., 2024) targets multivariate forecasting with proportional factorized attention and a Student-t mixture output layer. TiRex (Auer et al., 2025b) leverages an xLSTM backbone with strong in-context learning, and Sundial (Liu et al., 2025) introduces a flow-matching objective for flexible probabilistic forecasting.

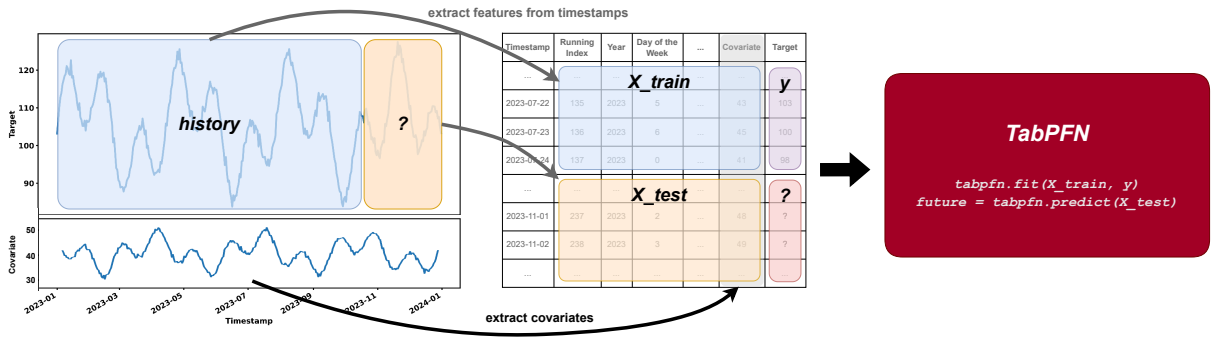


Figure 3.1: Overview of TabPFN-TS. From a given time series, we construct feature matrices $\mathbf{X}_{\text{train}}$ and $\mathbf{X}_{\text{test}}$, optionally appending exogenous covariates when available. The two matrices represent the historical and future windows, respectively. The observed historical targets form $\mathbf{y}_{\text{train}}$. TabPFN-v2 then conditions on the context $(\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}})$ and predicts the corresponding targets at $\mathbf{X}_{\text{test}}$.

Several TSFMs incorporate covariates. ChronosX (Arango et al., 2025) injects exogenous variables through modular adapters; and COSMIC (Auer et al., 2025a) augments forecasting with in-context covariate conditioning.[1]

TabPFN-v2. TabPFN (Hollmann et al., 2023) is a foundation model pretrained to perform tabular prediction directly from examples. Instead of being optimized on a single dataset, it is pretrained on millions of synthetic regression and classification tasks. Each task consists of input–output pairs (X, y) as the context and query points X' for which the model must predict corresponding targets y'. Through this large-scale meta-training, TabPFN learns general patterns of how features relate to targets that transfer across unseen problems.

At inference, the model receives a table of feature-target pairs (X, y) as context and outputs the posterior predictive distribution $p(y' \mid X, y, x')$ for new inputs X'. TabPFN represents this distribution as a *Riemann distribution*—a discretized probability density over possible target values—and is trained directly on this probabilistic representation. Having the model output a distribution allows to handle downstream cases which requires a distribution, for instance to account for the model uncertainty.

TabPFN-v2 (Hollmann et al., 2025) extends its predecessor TabPFN with architectural improvements, a richer synthetic data generator, while adding native support for handling missing values, outliers, and uninformative feature. Together, these properties make TabPFN-v2 a powerful probabilistic tabular regressor.

Our approach to time series forecasting is complementary to standard TSFMs: rather than pretraining a new sequence model, we use TabPFN-v2 and reformulate forecasting as tabular regression via temporal featurization. Unlike most TSFMs—which are designed around autoregressive or `seq2seq` architectures—TabPFN-TS predicts the full horizon in a single non-autoregressive forward pass, offering a strong alternative to sequence-based foundation models.

3 Methodology

In this section, we present TabPFN-TS, a novel approach to using TabPFN-v2 for multi-step, univariate time series forecasting. We recast time series forecasting as a tabular regression problem, where a time series sequence is treated as a table, as shown in Figure 3.1.

[1] ChronosX and COSMIC are not publicly available at the time of writing and are therefore not included in our comparison.

3.1 From Time Series to Tabular Data

Given a time series $\mathbf{y}_{1:C+H} = [y_1, \ldots, y_{C+H}]$, the first C observations define the historical context, and the remaining H correspond to the forecast horizon. As illustrated in Figure 3.1, we construct feature matrices $\mathbf{X}_{\text{train}} \in \mathbb{R}^{C \times D}$ and $\mathbf{X}_{\text{test}} \in \mathbb{R}^{H \times D}$, where D denotes the number of features. The corresponding historical targets form $\mathbf{y}_{\text{train}} \in \mathbb{R}^{C}$. This representation, consisting of $(\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}})$ and $\mathbf{X}_{\text{test}}$, allows the use of any standard supervised tabular models, including TabPFN-v2.

Running Index. To introduce a temporal reference within the timeline, we include the index of each time step as a feature (e.g., 0 for the first time step in the time series, 4 for the fifth):

$$\Phi_{\text{index}}(t) = t \in \mathbb{R}^1.$$

This provides a straightforward and effective way to track the progression of time across the observations and allows the model to extrapolate.

Calendar Features. From each timestamp, we encode eight cyclic calendar components: second of minute, minute of hour, hour of day, day of week, day of month, day of year, week of year, and month of year. We additionally include the calendar year as a separate (non-cyclic) feature. Let $\{P_i\}_{i=1}^{8}$ be the periods associated with these cyclic components. The corresponding features for each timestamp t are encoded as

$$\Phi_{\text{cal}}(t) = (\cos\Big(\frac{2\pi t}{P_1}\Big), \sin\Big(\frac{2\pi t}{P_1}\Big), \ldots, \cos\Big(\frac{2\pi t}{P_8}\Big), \sin\Big(\frac{2\pi t}{P_8}\Big), \text{year}(t)) \in \mathbb{R}^{17}$$

For full implementation details, see Appendix A.1.

Automatic Seasonal Features. Beyond the standard calendar periodicities, time series often have domain-specific cycles that calendar-based encodings fail to capture, e.g., depending on non-Gregorian calendars (Chinese birthdays) or the moon cycle (tides). To address this, we apply an automatic extraction process to identify the top-k periodicities and encode them as features, thereby enriching the seasonality inputs to the model.

Algorithm 1 Automatically extract top-k Seasonalities, see Appendix A.2 for a detailed algorithm

Require: univariate series series, the number of periods to obtain k, the smoothing window size L

Preprocess series
Detrend linearly: $\text{series}[t] = \text{series}[t] - (\alpha t + \beta)$, where α and β are found using least squares
Apply Hann window: $\text{series} = \text{conv}(\text{series}, w_{\text{Hann}}(L))$
Double length by symm. zero-padding: $\text{series} = [0, \ldots, 0, \text{series}[0], \ldots, \text{series}[N], 0, \ldots, 0]$

Fourier Transform
Compute FFT magnitudes mags and frequencies freqs based on the preprocessed series
Set $\text{mags}[0] = 0$ (remove the zero-frequency component)

Select Peaks
Find all peak indices peaks in mags (all (groups of) points larger than their neighbors)

Convert & Clean
Invert frequencies to periods and round $\text{periods} = \lfloor 1/\text{freqs} \rceil$
Remove duplicate and zero periods from the peak indices peaks
Keep only top k peaks: $\text{peaks} = [i \text{ for } i \text{ in peaks if } i \text{ in topk}(\text{mags}[\text{peaks}])]$

return periods[peaks]

We first detrend each series via a simple linear least-squares regression. To reduce spectral leakage and improve frequency resolution, we apply a Hann window (Harris, 1978) and zero-pad the windowed signal by

a factor of two (Oppenheim & Schafer, 1989). We then compute the real-valued discrete Fourier transform and remove the zero-frequency (DC) component of the spectrum, which corresponds to the mean of the time series and does not represent a seasonal oscillation. Finally, we select the k largest spectral peaks by magnitude. Algorithm 1 provides high-level pseudo-code for this extraction process.

Given the detected frequencies $f_1, \ldots, f_k$, we then build the following features:

$$\Phi_{\text{auto}}(t) = (\cos(2\pi f_1 t),\ \sin(2\pi f_1 t),\ \ldots,\ \cos(2\pi f_k t),\ \sin(2\pi f_k t)) \in \mathbb{R}^{2k}.$$

Time-varying Covariates (when available). TabPFN-TS also supports time-varying covariates (although only those that are known for future points in time, such as holidays, not unknown ones like weather). We denote them by $\mathbf{z}_{1:C+H} = [\mathbf{z}_1, \ldots, \mathbf{z}_{C+H}]$, where each $\mathbf{z}_t \in \mathbb{R}^{D_z}$ represents D_z external variables observed (or known) at time t.

For each timestamp, the full feature vector becomes:

$$\mathbf{x_t} = \Phi_{\text{index}}(t) \oplus \Phi_{\text{cal}}(t) \oplus \Phi_{\text{auto}}(t) \oplus \mathbf{z}_t \in \mathbb{R}^{(18+2k+D_z)}.$$

Combining all Features. Stacking these feature vectors across all timestamps yields the full feature matrix that describes the time series

$$\mathbf{X} = [\,\mathbf{x_1}; \ldots; \mathbf{x_{C+H}}\,] \in \mathbb{R}^{(C+H)\times D},$$

where $D = (18 + 2k + D_z)$ is the total number of features. Note that we do not rely on lagged or auto-regressive features (e.g., moving averages and lag terms), since these require past predictions and conflict with non-auto-regressive, multi-step forecasting making the inference much slower.

As described in Section 3.1, we split $\mathbf{X}$ into $\mathbf{X}_{\text{train}} = X_{1:C,:}$ and $\mathbf{X}_{\text{test}} = X_{C+1:C+H}$, corresponding to the historical and forecast windows.

3.2 Point and Probabilistic Forecasting with TabPFN-v2

We treat $(\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}})$ as a classical regression dataset and feed it into TabPFN-v2. For each test input x in $\mathbf{X}_{\text{test}}$, TabPFN-v2 outputs an approximate posterior predictive distribution $p(y \mid \mathbf{X}_{\text{train}},\, \mathbf{y}_{\text{train}},\, x)$.

In TabPFN-v2, this distribution is represented on a fixed numerical grid over the target space by assigning probabilities to bins (a discretized density). From this representation, we compute point predictions and probability summaries in the following way: the mean for squared-error evaluations, the median for absolute-error evaluations, and arbitrary quantiles (e.g. 5%, 50%, 95%) for probabilistic metrics and uncertainty bands.

4 Experiments

We present quantitative comparisons of TabPFN-TS against state-of-the-art forecasting models on both univariate (Section 4.1) and covariate-aware forecasting (Section 4.2) tasks. For all evaluations, we use a fixed configuration of TabPFN-v2 described in Appendix A.3.

4.1 Univariate Forecasting

We evaluate TabPFN-TS on GIFT-Eval (Aksu et al., 2024), a comprehensive benchmark developed to evaluate general time series forecasting models.

Datasets. GIFT-Eval comprises 23 datasets with diverse characteristics, encompassing over 144,000 time series and 177 million data points across seven application domains and ten different sampling frequencies. It covers both univariate and multivariate forecasting settings, as well as a wide range of prediction horizons, from short- to long-term forecasts. Considering all valid combinations of datasets, sampling frequencies, and prediction horizons, GIFT-Eval contains a total of 97 distinct benchmarking tasks. An overview of the datasets and their corresponding statistics is provided in Appendix A.4.

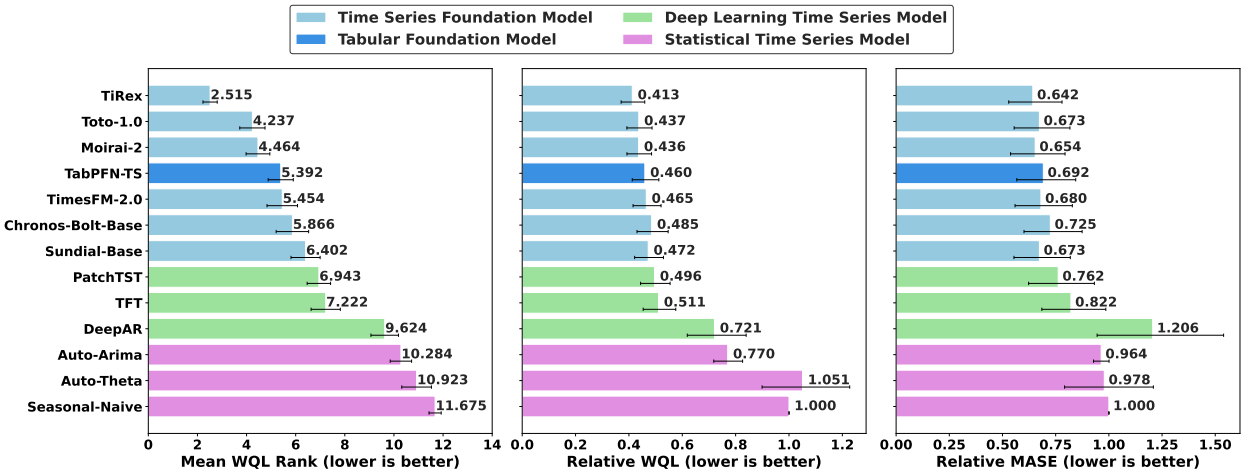


Figure 4.1: Univariate forecasting performance of TabPFN-TS and baseline models on GIFT-Eval benchmark. Although TabPFN-TS is not the top-ranked model, it achieves competitive performance on par with other time series foundation models (e.g. TimesFM2.0, Chronos-Bolt) while being a much smaller model (11M) and pretrained solely on tabular data. Scores are normalized by Seasonal Naive and aggregated across datasets; error bars show 95% confidence intervals.

Baselines. We benchmark TabPFN-TS against a comprehensive set of baselines spanning statistical methods, deep learning models, and time series foundation models[2]. For classical statistical methods, we include Seasonal Naive, AutoETS, AutoARIMA, and AutoTheta (Garza et al., 2022). Among deep learning approaches, we evaluate DeepAR (Salinas et al., 2020) and Temporal Fusion Transformer (TFT) (Lim et al., 2021). For foundation models baselines, we include TiRex (Auer et al., 2025b), Toto-1.0 (Cohen et al., 2024), TimesFM-2.0 (Das et al., 2024), Chronos-Bolt (Ansari et al., 2024), and Moirai-2 (Woo et al., 2024).

Evaluation Metrics. Following standard practice (Ansari et al., 2024; Shchur et al., 2023; Aksu et al., 2024), we assess point forecast accuracy using the Mean Absolute Scaled Error (MASE) and probabilistic forecast accuracy using the Weighted Quantile Loss (WQL). MASE scales the absolute forecast error by the historical seasonal error of each series, yielding a scale-invariant measure comparable across datasets. WQL evaluates the discrepancy between the predictive distribution and the observed value across quantile levels, providing a good proxy assessment of probabilistic calibration. Consistent with GIFT-Eval, we compute WQL at uniformly spaced quantiles $\{0.1, 0.2, \ldots, 0.9\}$. Following Ansari et al. (2024), we aggregate relative scores across datasets using the geometric mean and additionally report the mean rank of WQL to provide an alternative view of overall performance.

Results. Figure 4.1 summarizes the forecasting performance of TabPFN-TS on the GIFT-Eval benchmark. In both probabilistic forecasting (WQL) and point forecasting (MASE), TabPFN-TS ranks closely behind the leading foundation models TiRex, Toto-1.0, and Moirai-2, and performs on par with substantially larger architectures such as TimesFM-2.0 and Chronos-Bolt, despite being over two orders of magnitude smaller and trained without any time-series–specific pretraining. Qualitative visualizations are provided in Appendix A.5 for reference.

These results demonstrate that TabPFN-TS is capable of producing accurate and well-calibrated forecasts even though its backbone, TabPFN-v2, is pretrained solely on synthetic tabular data and relies on a simple featurization scheme (see Section 3.1). This finding is two-fold: it suggests that knowledge learned from generic tabular data can transfer effectively to temporal prediction, and that further improvements are likely if tabular foundation models are exposed to real-world time-series data during pretraining. Together, these results position TabPFN-TS as a compact yet competitive alternative to large time-series foundation models.

[2]All baselines included were publicly available before September 2025, our literature cutoff date.

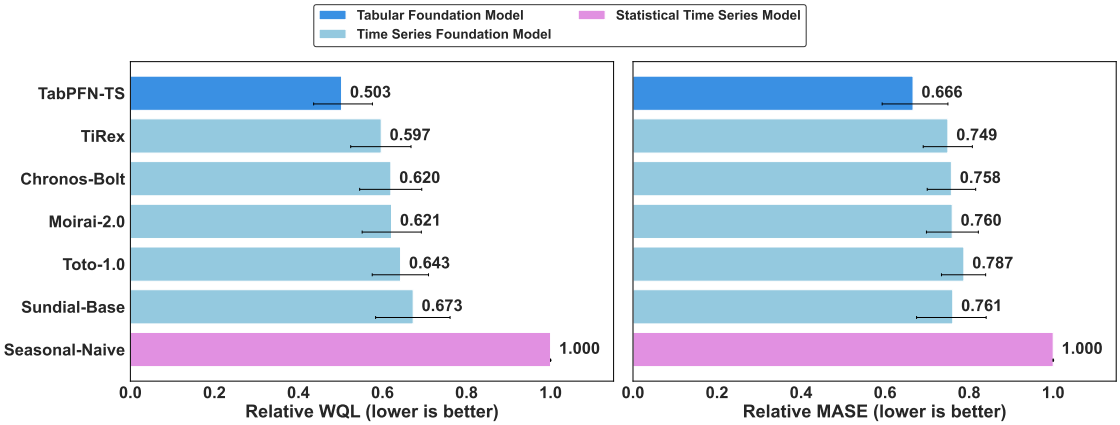


Figure 4.2: Covariate-informed forecasting performance of TabPFN-TS and baseline models on 28 `fev-bench` tasks with known dynamic covariates. As the only model that can directly incorporates covariate inputs, TabPFN-TS achieves the strongest results, outperforming all other models, including multivariate approaches such as Toto-1.0.

4.2 Covariate-Informed Forecasting

For this task, we evaluate TabPFN-TS on `fev-bench` (Shchur et al., 2025), a general time-series forecasting benchmark that includes forecasting tasks where covariates are available.

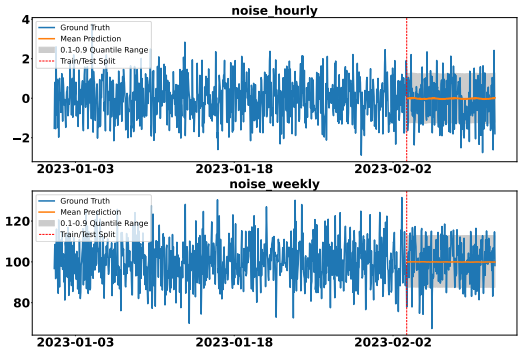
Datasets. `fev-bench` comprises 100 forecasting tasks across multiple domains, of which 42 include covariate information. We focus on the 30 tasks where covariates are available in both the historical and future horizons. This configuration is required by TabPFN-TS, which models the dependency between the target series and known covariates and relies on having those covariate values available at prediction time. When future covariates are missing, the model cannot properly condition its forecasts on relevant contextual signals. Two of these tasks are excluded because TabPFN-TS exceeded the 6-hour evaluation limit, leaving 28 tasks for the final evaluation. Appendix A.6 provides an overview of the selected tasks.

Baselines. We follow the standardized evaluation protocol of `fev-bench`, which reports results only on pretrained time series foundation models. Deep learning baselines are not included here as `fev-bench` does not include dataset-specific supervised deep learning baselines. Therefore, like Section 4.1, TabPFN-TS compares against the following pretrained models: TiRex (Auer et al., 2025b), Chronos-Bolt (Base) (Ansari et al., 2024), Toto-1.0 (Cohen et al., 2024), Moirai-2.0 (Woo et al., 2024), and Sundial (Liu et al., 2025).

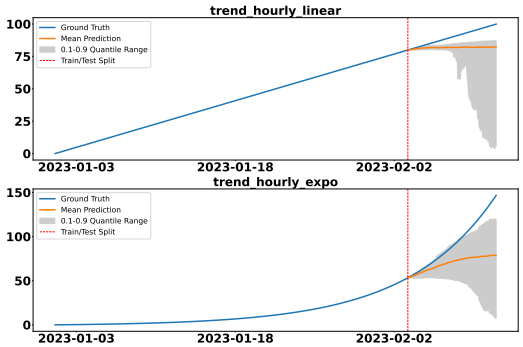
Evaluation Metrics. Consistently with Section 4.1, we assess point forecasting accuracy using the Mean Absolute Scaled Error (MASE) and probabilistic forecast accuracy using the Weighted Quantile Loss (WQL).

Results. Across the 28 `fev-bench` tasks with known dynamic covariates, TabPFN-TS achieves the strongest overall performance. While strong foundation models, such as TiRex and Chronos-Bolt, remain highly competitive in the univarate setting (see Section 4.1 and Shchur et al. (2025)), TabPFN-TS outperforms them when covariates are available. Notably, Toto-1.0, which is designed for the multivariate setting, doesn not fully recover the benefit obtained when covariates are treated as first-class predictive features like in TabPFN-TS.

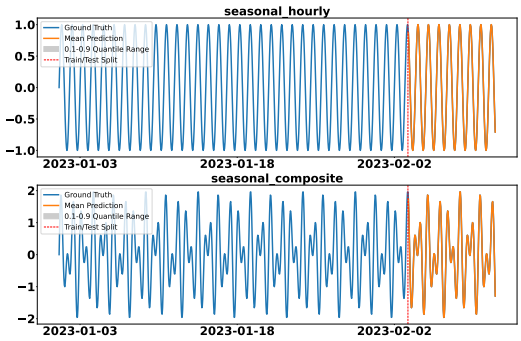
These results provide strong evidence for the forecasting-as-tabular-regression formulation (Section 3): it allows TabPFN-TS to integrate covariates directly and without architectural changes or fine-tuning. Together, this positions TabPFN-TS as a practical and competitive alternative to current time series foundation models, especially in settings where contextual or external drivers are known and informative.



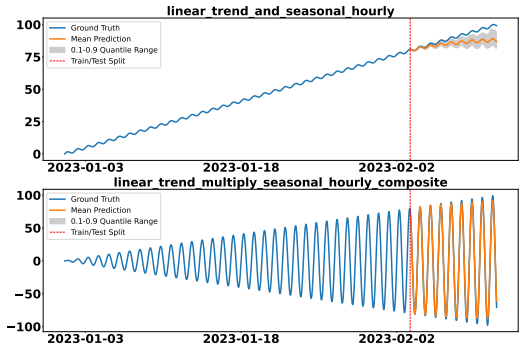
(a) On i.i.d noise



(b) On linear and exponential trends



(c) On seasonal patterns



(d) On additive and multiplicative combinations

Figure 5.1: Qualitative performance of TabPFN-v2 across noise, trend, and seasonal structures. The model handles noise and periodicity well but cannot extrapolate linear or exponential trend beyond the observed target range.

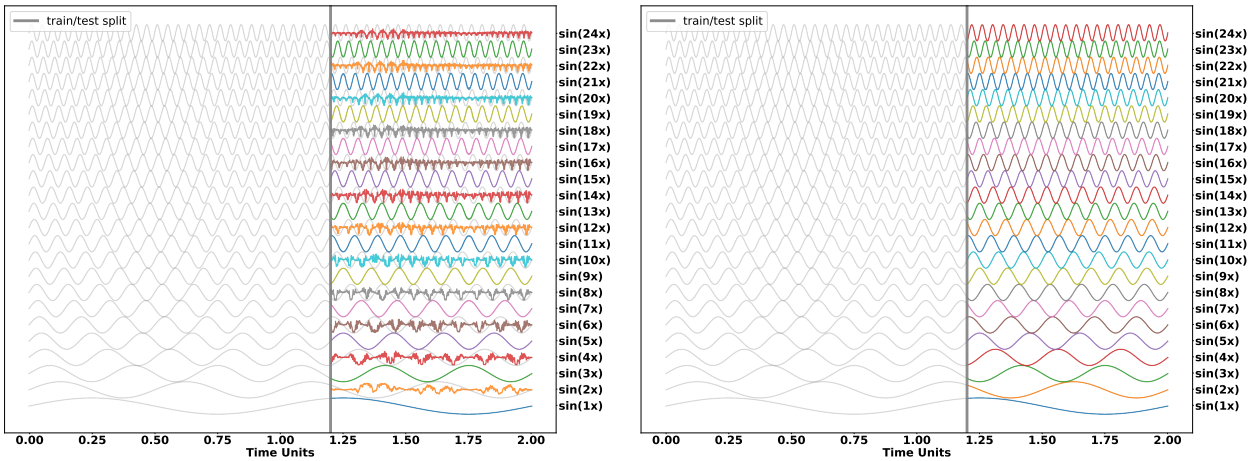
5 Ablations

We conduct a series of ablation studies to understand how TabPFN-TS performs forecasting and to assess the contribution of its key components. Section 5.1 provides qualitative examples illustrating the model's forecasting behavior. Section 5.2 examines how the model interprets temporal structure within the tabular regression framework to perform forecasting. Section 5.3 studies how TabPFN-TS integrates covariate information under controlled synthetic setups. Section 5.4 evaluates alternative tabular regressors to assess the role of the pretrained backbone. Finally, Section 5.5 analyzes the impact of the temporal featurization design on overall performance.

5.1 Qualitative Analysis

We complement the quantitative benchmarks (Section 4) with qualitative evaluations on controlled synthetic series, following the setups proposed by Ansari et al. (2024). In all cases, the first 800 points are provided as context and the model forecasts the next 200 steps. These experiments illustrate the characteristic strengths and limitations of TabPFN-TS in isolation from dataset-specific confounders.

I.I.D. Noise. Figure 5.1a shows the model's predictions when the input is i.i.d. noise sampled from $\mathcal{N}(0, 1)$ and $\mathcal{N}(100, 10)$ at hourly and weekly resolution. TabPFN-TS does not overfit: it predicts the sample mean and produces predictive intervals that closely align with the corresponding Gaussian quantiles. This demonstrates stable behavior and well-calibrated uncertainty when no temporal structure is present.



(a) Conditioning on $\sin(t)$ only

(b) Conditioning on $\sin(t)$ and $\cos(t)$

Figure 5.2: Reconstruction of sinusoidal signals under different temporal embeddings. TabPFN-v2 is evaluated on $\sin(nt)$ for $n = 1, \ldots, 24$. (a) With $\sin(t)$ as the sole temporal feature, the non-injective embedding causes different time points to collapse to the same feature representation, and the model therefore reconstructs odd but not even harmonics. (b) With $\{\sin(t), \cos(t)\}$, the injective embedding allows the model to interpolate reliably in this phase space and reconstruct all harmonics. These results suggest that periodic generalization arises from interpolation in an expressive temporal embedding.

Seasonality. TabPFN-TS models seasonal structure exceptionally well (Figures 5.1c–d). When periodic patterns are represented in the features, the model not only reconstructs the signal accurately but also produces very tight predictive intervals that closely follow the ground truth. This strong behaviour is further discussed in Section 5.2, where we show that the sinusoidal embedding provides an expressive phase representation that enables precise interpolation of periodic components.

Trend. Figure 5.1b illustrates a clear limitation: TabPFN-TS does not extrapolate trends when future values fall outside the range of the conditioning set (i.e. beyond the observed target range). This behavior appears both for linear and exponential trends. Since the model operates via interpolation in feature space (discussed later in Section 5.2), it cannot extend predictions beyond the observed target domain—an important limitation for time series with sustained growth or decay.

Takeaway. These qualitative results reveal a consistent pattern: TabPFN-TS behaves reliably when the future targets lie within the range of the conditioning set—capturing noise and seasonal patterns with accurate means and tight, well-calibrated uncertainty—but fails to extrapolate when forecasting requires moving beyond the observed target domain, as in linear or exponential trends. This contrast highlights that the model's strengths and limitations depend heavily on how temporal structure is represented, motivating the deeper analysis in Section 5.2 and Section 6.1.

5.2 How TabPFN-TS Interprets Temporal Structure

To understand why TabPFN-v2 handles seasonality effectively (see Section 5.1) despite being pretrained only on tabular data, we study how its backbone (TabPFN-v2) when combined with minimal temporal featurization, performs forecasting on periodic signals. This reveals what information the temporal featurization provides and how the model uses it.

Setup. We generate periodic signals that allow us to probe how TabPFN-v2 responds to temporal structure under different featurization settings. To examine its ability to recover harmonic content from basic periodic cues, we generate target series of the form $y_t = \sin(nt)$ for $n = 1, \ldots, 24,$. The model is evaluated under two featurization settings: (i) using $\sin(t)$ alone, and (ii) the pair $(\sin(t), \cos(t))$. This setup enables us to test

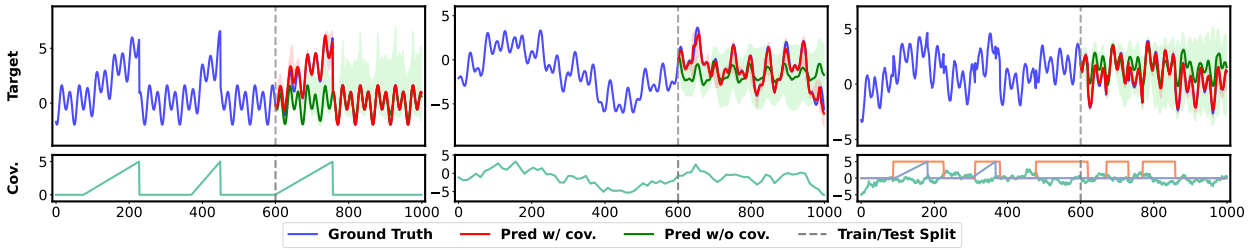


Figure 5.3: Forecasts of TabPFN-TS on synthetic time series augmented by time-varying covariates. Median predictions are shown in red (conditioned on covariates) and green (unconditioned), with shaded bands indicating the 10th - 90th percentiles.

whether the model can recover higher-order harmonics from low-frequency temporal coordinates. We then extend the analysis to composite signals formed by summing multiple sinusoids with random frequencies, amplitudes, and phases.

Findings. With only $\sin(t)$, the model reconstructs odd harmonics but fails on even ones (Figure 5.2a). This arises because the map $t \mapsto \sin(t)$ is **not injective**: multiple time points share the same feature value, making local interpolation inherently ambiguous.

When both $\sin(t)$ and $\cos(t)$ are provided, the embedding becomes injective on the unit circle and uniquely identifies phase. Under this representation, TabPFN-v2 accurately approximates harmonics across a wide range of frequencies (Figure 5.2b). However, we observe a gradual deterioration in accuracy at higher frequencies (Figure A.4 and A.5). This is consistent with the finite resolution of the temporal embedding: as frequency increases, adjacent phases become harder to distinguish in the $\{\sin t, \cos t\}$ coordinate system at the given sampling density, consistent with classical sampling and aliasing effects (Oppenheim, 1999). Thus, the model interpolates smoothly in this feature space, but only up to the point where the embedding retains sufficient phase resolution.

For composite signals, TabPFN-v2 again reconstructs the patterns accurately (Figure A.6). Nearby phases map to nearby feature vectors, enabling the model to retrieve appropriate patterns without explicit frequency inputs.

Takeaway. These experiments suggest that when supplied with an injective sinusoidal embedding of time, TabPFN-TS behaves like a learned kernel or nearest-neighbor regressor, interpolating effectively in feature space. This explains its strong performance whenever the featurization exposes relevant seasonal components.

5.3 How TabPFN-TS Utilizes Covariates

In Section 4.2, we showed that TabPFN-v2 achieves the strongest performance on covariate-informed forecasting tasks. To better understand this effect, we conduct controlled synthetic experiments where the informativeness and interaction structure of each covariate are known.

Synthetic Setup. Similar to Arango et al. (2025) and Auer et al. (2025a), we generate synthetic time series where the relationship between the target and the covariates is fully controlled. Each series consists of a stationary stochastic process (Brockwell & Davis, 2016)—implemented here as a sum of sinusoids with random amplitudes, phases, and periods—augmented with one or more covariates that follow common temporal patterns such as linear trends, pulses, ramps, or stochastic drifts. Covariates influence the target through either additive or multiplicative coupling, resulting in ten interpretable synthetic classes (Figure A.7).

Evaluation Protocol. For each class, we evaluate TabPFN-TS under two conditions: (i) using temporal features only, and (ii) using both temporal features and covariates. This isolates the incremental contribution of covariate information.

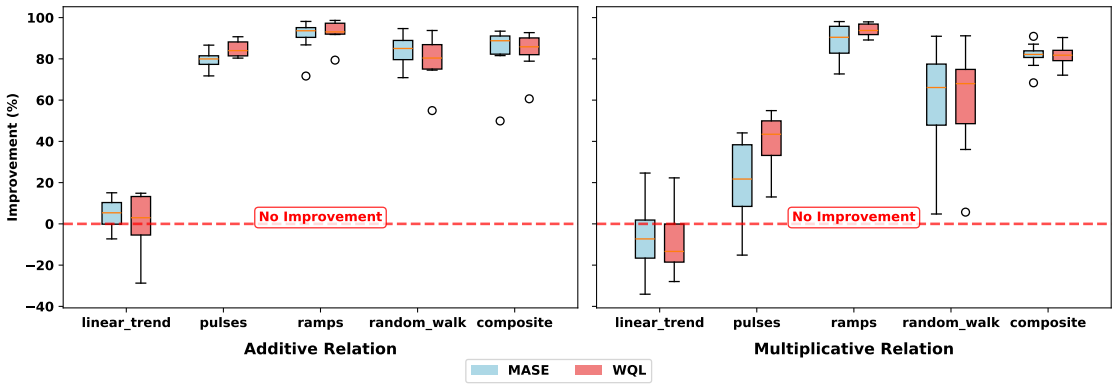


Figure 5.4: Improvement when incorporating covariates across synthetic classes. Boxplots show relative improvement (in percentage, %) in MASE (blue) and WQL (red). Most covariate types lead to substantial improvement, particularly for pulses, ramps, random walks, and composite signals. In contrast, for both additive and multiplicative linear trend series, covariates fail to improve forecasts, reflecting the model's limitation in extrapolating linear growth.

Findings. Figure 5.4 shows that incorporating covariates substantially improves forecasting accuracy for most synthetic classes—pulses, ramps, random walks, and composite signals—reducing both MASE and WQL and producing sharper uncertainty estimates (Figure A.7). These gains are consistent with our benchmark results and indicate that TabPFN-TS can reliably exploit informative auxiliary signals when their effects lie within the range covered by the conditioning set (detailed in Section 6.1).

The main exception is the *linear-trend* setting, where covariates provide little benefit because the future targets fall outside the distribution observed in the context; TabPFN-TS does not extrapolate linear growth (discussed later in Section 6.1). Overall, these controlled experiments show that TabPFN-TS effectively leverages covariates across a wide range of scenarios, with extrapolation remaining the primary limitation.

5.4 Choice of Tabular Backbone

To assess whether our forecasting formulation depends on the choice of tabular backbone, we substitute TabPFN-v2 with TabDPT (Ma et al., 2025), one of the few open-source tabular foundation models that supports regression (Erickson et al., 2025). We refer to this variant as TabDPT-TS. As TabDPT produces only point predictions, we evaluate its accuracy on point forecasting on 78 of the 97 GIFT-Eval tasks (the remainder exceeded our compute budget).

Figure 5.5 shows that TabDPT-TS achieves meaningful forecasting performance but is not competitive with TabPFN-TS. Its accuracy is comparable to Chronos-Bolt-Tiny, a smaller variant of Chronos-Bolt, indicating that while our temporal featurization enables TabDPT to make zero-shot forecasts, the backbone model itself significantly influences overall performance.

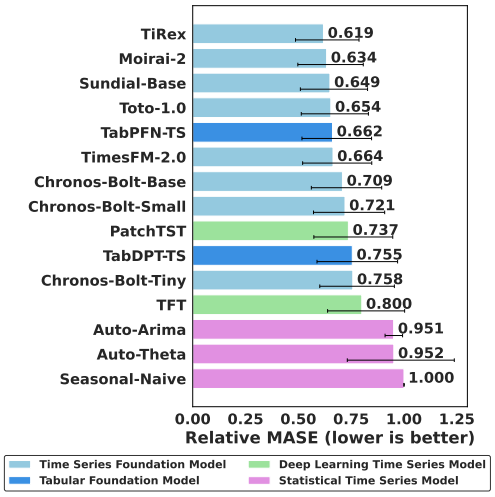


Figure 5.5: Accuracy of TabDPT-TS versues baselines. It achieves reasonable accuracy but remains well below TabPFN-TS.

We attribute the gap between TabPFN-TS and TabDPT-TS to the difference in their pretraining corpora: TabPFN-v2 is trained on millions of synthetically generated tasks drawn from a wide variety of functional relationships and noise regimes, while TabDPT is pretrained exclusively on real-world tabular datasets. We hypothesize that the broader task diversity present in TabPFN-v2 's synthetic meta-training in-

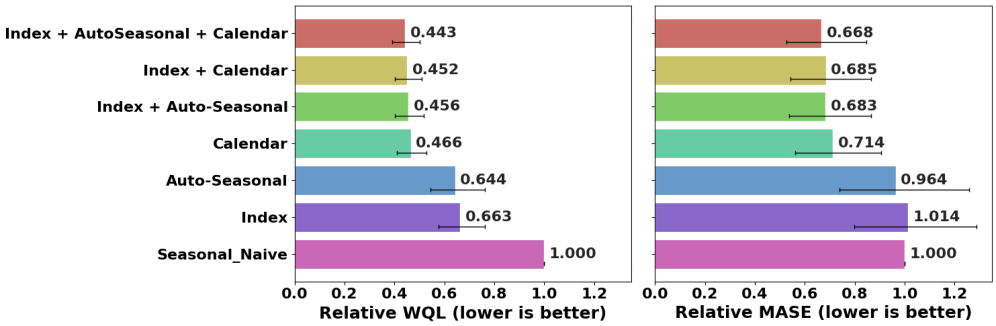


Figure 5.6: Ablation of temporal featurization components in TabPFN-TS. Relative WQL (left) and MASE (right) show that all components contribute meaningfully to performance. Using only the index or only the auto-seasonal features yields substantially worse results, while combining all feature types achieves the best accuracy, underscoring the value of the proposed featurization scheme.

duces inductive biases that align more closely with the forecasting-as-tabular-regression formulation, though a more systematic analysis is needed to verify this.

Taken together, this experiment suggests that: (i) the **temporal featurization generalizes across tabular foundation models**—different tabular foundation models can perform zero-shot forecasting under the same representation; and (ii) the **regression backbone matters**—models pretrained on broader and more diverse task distributions (such as TabPFN-v2) appear to transfer more effectively to the time series forecasting.

5.5 Impact of Temporal Featurization

To assess the contribution of the temporal features introduced in Section 3.1, we evaluate TabPFN-TS under different combinations of these features on the 81 smallest tasks (out of 97) in GIFT-Eval.

Figure 5.6 reports the relative WQL and MASE across feature subsets. Using only the index or only the automatic seasonal features performs poorly, whereas calendar-only features achieve substantially better results. Once the index is included, the gap between calendar (`index+calendar`) and automatic seasonality (`index+auto-seasonal`) narrows considerably.

This behavior highlights the importance of **encoding time progression**. Without an explicit index, calendar features provide both periodic structure and a weak notion of temporal progression, since some calendar attributes present monotonically within the timeline. Automatic seasonal features, by contrast, encode oscillatory components but offer no information about global temporal progression. When the index is present, it provides the primary temporal axis, while both calendar and automatic seasonal features contribute the complementary periodic structure.

The best results are obtained when all feature types are combined. This indicates that calendar and automatic seasonal features capture different—and complementary—seasonal patterns. Together with the index, they provide a board and expressive temporal representation.

Overall, these findings show that TabPFN-TS benefits from temporal features that jointly encode **monotonic progression** and **multi-scale seasonality**. The framework is naturally extensible, allowing practitioners to incorporate domain-specific periodic features to further improve forecasting accuracy when such knowledge is available.

6 Limitation

6.1 Extrapolation Beyond the Conditioning Range

The analyses in Sections 5.1 and 5.3 reveal a persistent limitation of TabPFN-TS: it struggles in extrapolating trends, where the future target values lie outside the range observed in the conditioning set.

This behavior is a direct consequence of how TabPFN-v2 is pretrained. During meta-training, each synthetic task is generated by sampling a tabular dataset and randomly partitioning it into a context set (X, y) and a query set X'. The model is then trained to predict the corresponding targets y'. Because the query points are always drawn from the same generating distribution as the context—and therefore lie within the convex hull of the observed targets—TabPFN-v2 is repeatedly optimized to solve interpolation tasks rather than extrapolation tasks.

Consequently, in time-series forecasting, TabPFN-TS inherits this inductive bias. When future targets lie outside the range of observed target range, it has no basis—under its interpolation-oriented objective—to project the trajectory further. Predictions therefore remain within or close to the span of the observed targets.

In summary, TabPFN-v2 is a strong conditional interpolator but lacks a mechanism for systematic trend extrapolation. This limitation is structural, stemming from the meta-training objective, rather than from the tabular regression formulation used for forecasting.

6.2 Inference Time

A second practical limitation of TabPFN-TS is its higher computational cost at inference. Although the model supports data-parallel evaluation across multiple GPUs, this only increases throughput but does not reduce the per-time-series cost. Figure 6.1 reports single-GPU-equivalent inference times, normalized by GPU count to allow direct comparison with Chronos-Bolt-Base and TimesFM-2.0. Under equal hardware assumptions, TabPFN-TS requires roughly 30× more inference time.

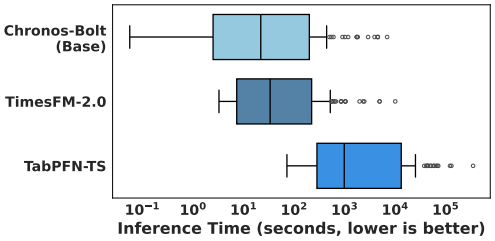


Figure 6.1: Dataset-level inference time across 97 GIFT-Eval tasks (log scale; lower is better). Chronos-Bolt-Base and TimesFM-2.0 were evaluated on one NVIDIA T4. TabPFN-TS distributes the time series of each dataset across four T4 GPUs; we report its runtime normalized to a single GPU for fair comparison.

This overhead arises from how TabPFN-v2 performs prediction. In tabular settings, one forward pass produces the prediction for an entire dataset. In time series forecasting, however, a dataset contains many individual series—often hundreds or thousands—and each series becomes an independent query instance under our formulation. TabPFN-TS must therefore execute a separate forward pass for every series.

Moreover, TabPFN-TS retains the full-attention architecture of TabPFN-v2, which incurs $O(L^2)$ computational complexity in sequence length L. Recent time-series foundation models mitigate this cost by processing large batches of series and by using architectural techniques such as patching or lightweight decoders that reduce dependence on sequence length. Consequently, they achieve substantially faster inference.

In summary, the higher inference cost of TabPFN-TS arises structurally from the combination of (i) its one-forward-pass-per-series formulation and (ii) its quadratic attention mechanism. Improving inference efficiency—through batching, architectural refinement, or sequence-length optimization—remains an important direction for future work.

7 Conclusion

Summary of Contributions. This work demonstrates that a tabular foundation model—pretrained solely on synthetic, non-temporal data—can be adapted into a competitive zero-shot forecaster through a simple reformulation of time-series prediction as tabular regression. Combining TabPFN-v2 with a lightweight temporal featurization yields TabPFN-TS, a compact (11M) model that requires no time-series–specific pretraining or fine-tuning, yet achieves strong performance on univariate forecasting and state-of-the-art accuracy on covariate-informed tasks.

Key Empirical Insights. Our ablations highlight three central findings. (1) Temporal featurization is essential: performance improves substantially when monotonic progression and complementary periodicities are jointly encoded. (2) Pretraining induces distinct inductive biases: TabPFN-TS excels at interpolation inside the observed target range, but—consistent with its meta-training objective—fails when forecasting requires systematic extrapolation. (3) Forecasting capability is model-agnostic: substituting the backbone with TabDPT still produces meaningful zero-shot forecasts, indicating that the overall formulation generalizes across tabular foundation models.

Implications. These results indicate that tabular foundation models constitute a surprisingly effective and data-efficient alternative to specialized time-series architectures. The tabular formulation naturally unifies univariate and covariate-aware forecasting, treats all inputs as general features, and enables probabilistic forecasting without architectural modifications to the backbone.

Vision and Future Directions. More broadly, this work suggests a promising direction toward unified foundation models for structured data that operate seamlessly over tabular, temporal, and hybrid inputs. Addressing the core limitation—the inability to extrapolate trends beyond the conditioning range—may require new meta-training objectives that blend interpolation, controlled extrapolation, and explicit handling of out-of-support targets. Such advances open a path toward compact, domain-agnostic predictors that bridge long-standing gaps between tabular learning and time series forecasting.

References

Taha Aksu, Gerald Woo, Juncheng Liu, Xu Liu, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. Gift-eval: A benchmark for general time series forecasting model evaluation. *arxiv preprint arxiv:2410.10393*, 2024.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

Sebastian Pineda Arango, Pedro Mercado, Shubham Kapoor, Abdul Fatir Ansari, Lorenzo Stella, Huibin Shen, Hugo Senetaire, Caner Turkmen, Oleksandr Shchur, Danielle C. Maddix, Michael Bohlke-Schneider, Yuyang Wang, and Syama Sundar Rangapuram. Chronosx: Adapting pretrained time series models with exogenous variables, 2025. URL `https://arxiv.org/abs/2503.12107`.

Andreas Auer, Raghul Parthipan, Pedro Mercado, Abdul Fatir Ansari, Lorenzo Stella, Bernie Wang, Michael Bohlke-Schneider, and Syama Sundar Rangapuram. Zero-shot time series forecasting with covariates via in-context learning. *arXiv preprint arXiv:2506.03128*, 2025a.

Andreas Auer, Patrick Podest, Daniel Klotz, Sebastian Böck, Günter Klambauer, and Sepp Hochreiter. Tirex: Zero-shot forecasting across long and short horizons with enhanced in-context learning, 2025b. URL `https://arxiv.org/abs/2505.23719`.

Peter J. Brockwell and Richard A. Davis. Stationary Processes. In Peter J. Brockwell and Richard A. Davis (eds.), *Introduction to Time Series and Forecasting*, pp. 39–71. Springer International Publishing, Cham, 2016. ISBN 978-3-319-29854-2. doi: 10.1007/978-3-319-29854-2_2.

Ben Cohen, Emaad Khwaja, Kan Wang, Charles Masson, Elise Ramé, Youssef Doubli, and Othmane Abou-Amal. Toto: Time series optimized transformer for observability, 2024. URL `https://arxiv.org/abs/2407.07874`.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

Nick Erickson, Lennart Purucker, Andrej Tschalzev, David Holzmüller, Prateek Mutalik Desai, David Salinas, and Frank Hutter. Tabarena: A living benchmark for machine learning on tabular data, 2025. URL `https://arxiv.org/abs/2506.16791`.

Federico Garza, Max Mergenthaler Canseco, Cristian Challú, and Kin G. Olivares. StatsForecast: Lightning fast forecasting with statistical and econometric models. PyCon Salt Lake City, Utah, US 2022, 2022. URL `https://github.com/Nixtla/statsforecast`.

Fredric J. Harris. On the use of windows for harmonic analysis with the discrete fourier transform. *Proceedings of the IEEE*, 66(1):51–83, 1978.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations*, 2023.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025.

Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.

Yong Liu, Guo Qin, Zhiyuan Shi, Zhi Chen, Caiyin Yang, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Sundial: A family of highly capable time series foundation models, 2025. URL `https://arxiv.org/abs/2502.00816`.

Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Hamidreza Kamkari, Alex Labach, Jesse C. Cresswell, Keyvan Golestan, Guangwei Yu, Anthony L. Caterini, and Maksims Volkovs. Tabdpt: Scaling tabular foundation models on real data, 2025. URL https://arxiv.org/abs/2410.18164.

Alan V. Oppenheim and Ronald W. Schafer. *Discrete-Time Signal Processing*. Prentice Hall, 1989.

A.V. Oppenheim. *Discrete-Time Signal Processing*. Pearson education signal processing series. Pearson Education, 1999. ISBN 9788131704929. URL https://books.google.de/books?id=geTn5W47KEsC.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3):1181–1191, 2020.

Oleksandr Shchur, Caner Turkmen, Nick Erickson, Huibin Shen, Alexander Shirkov, Tony Hu, and Yuyang Wang. AutoGluon-TimeSeries: AutoML for probabilistic time series forecasting. In *International Conference on Automated Machine Learning*, 2023.

Oleksandr Shchur, Abdul Fatir Ansari, Caner Turkmen, Lorenzo Stella, Nick Erickson, Pablo Guerron, Michael Bohlke-Schneider, and Yuyang Wang. fev-bench: A realistic benchmark for time series forecasting, 2025. URL https://arxiv.org/abs/2509.26468.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers, 2024. URL https://arxiv.org/abs/2402.02592.

A Appendix

A.1 Implementation of Calendar Features

Algorithm 2 Detailed Calendar Features Implementation

Require:

[nosep,leftmargin=*,topsep=0pt]

Time-indexed table $\mathcal{D}$ with index level `timestamp`.

Ensure: $\mathcal{D}$ augmented with

[nosep,leftmargin=*,topsep=0pt]

- `year` column;
- sin and cos embeddings for each of: (`second_of_minute`, 60), (`minute_of_hour`, 60), (`hour_of_day`, 24), (`day_of_week`, 7), (`day_of_month`, 30.5), (`day_of_year`, 365), (`week_of_year`, 52), (`month_of_year`, 12).

$\mathcal{D} \leftarrow \mathcal{D}$.copy()
$\mathbf{T} \leftarrow \mathcal{D}$.index.get_level_values(`"timestamp"`)

Extract year component
$\mathcal{D}$[`"year"`] $\leftarrow$ $\mathbf{T}$.`year`

Extract calendar-based seasonality
$\mathcal{S} \leftarrow \{$
(`"second_of_minute"`, 60),
(`"minute_of_hour"`, 60),
(`"hour_of_day"`, 24),
(`"day_of_week"`, 7),
(`"day_of_month"`, 30.5),
(`"day_of_year"`, 365),
(`"week_of_year"`, 52),
(`"month_of_year"`, 12)
$\}$ ▷ List of seasonal features with their natural periods

for all $(name, P)$ in $\mathcal{S}$ **do**
  $\mathbf{f} \leftarrow$ time_feature($name$).index($\mathbf{T}$) ▷ integer cycle index
  $\tilde{P} \leftarrow P - 1$
  $\mathcal{D}[name\|_\texttt{sin}] \leftarrow \sin\left(2\pi\, \mathbf{f}/\tilde{P}\right)$
  $\mathcal{D}[name\|_\texttt{cos}] \leftarrow \cos\left(2\pi\, \mathbf{f}/\tilde{P}\right)$
end for
return $\mathcal{D}$

A.2 Implementation of Automatic Seasonal Features

Algorithm 3 Detailed Extract top-k Seasonalities Algorithm

Require:

[nosep,leftmargin=*,topsep=0pt]

Time series $\mathbf{x}_t = \{x_1, x_2, \ldots, x_N\}$

- Integer k (max number of periods)
- Hann window length L

Ensure: Set $\mathcal{P}$ of up to k dominant periods

Preprocessing:

1: Detrend $\mathbf{x}_t$ via linear regression

$$\tilde{x}_t = x_t - (\alpha t + \beta) \quad \text{, where } \alpha \text{ and } \beta \text{ are found using least squares}$$

2: Apply Hann window:

$$w'_t = 0.5\left(1 - \cos\left(\frac{2\pi t'}{L}\right)\right) \quad \text{for} \quad t \in \{0, \ldots, L\}$$

$$\breve{x} = conv(\tilde{x}, w)$$

3: Symmetrically zero-pad to length $2N$:

$$\mathbf{y} = [0, \ldots, 0, \breve{x}_1, \ldots, \breve{x}_N, 0, \ldots, 0]$$

Spectral Analysis:

4: Compute fast fourier transform:

$$Y_k = \sum_{t=1}^{2N} y_t e^{-i2\pi(k-1)t/(2N)} \quad \text{for } k = 1, \ldots, N$$

$$\text{Magnitudes: } A_k = |Y_k|$$

$$\text{Frequencies: } f_k = \frac{k-1}{2N} \quad \text{(normalized to Nyquist)}$$

5: Remove DC component:

$$A_1 \leftarrow 0$$

Peak Selection:

6: Identify local maxima (peaks larger than immediate neighbors, taking midpoint of multi-point peaks in practice):

$$\mathcal{L} = \left\{ i \in \{2, \ldots, N-1\} \,\middle|\, A_i > A_{i-1} \text{ and } A_i > A_{i+1} \right\}$$

Period Conversion:

7: Convert frequencies to periods and round to integers:

$$p_i = \left\lfloor \frac{1}{f_i} \right\rceil \text{, for } i \in L$$

8: Remove duplicates and 0. periods, yielding a new set of indexes $\mathcal{I}$

9: Build top k index set $\mathcal{T} = \{i \in \mathcal{I} | A_i \in \text{topk}_k(\{A_i | i \in \mathcal{I}\})\}$

10: **return** $\{p_i | i \in \mathcal{T}\}$

A.3 TabPFN-TS Configuration

We describe the configuration used for all TabPFN-TS experiments. Unless otherwise noted, all settings are fixed across benchmarks.

Data Preprocessing. For time series with missing observations, we remove the affected time points from the conditioning set. For efficiency, we restrict the conditioning window to the most recent 4096 observations (after removing missing points), which we found to provide a good balance between accuracy and computational cost (see Appendix A.9). No additional normalization is applied to the target values, as TabPFN-v2 performs internal z-normalization during inference.

TabPFN-v2 Model Configuration. TabPFN-v2 offers several publicly available regression checkpoints. For all experiments, we use the `tabpfn-v2-regression-2noar4o2.ckpt`[3], which showed stable behavior across both point and probabilistic forecasting in preliminary evaluations. We do not perform hyperparameter tuning or checkpoint selection beyond this fixed choice, as our focus is on evaluating the utility of the forecasting formulation rather than optimizing the backbone.

Temporal Featurization. We apply the full featurization pipeline described in Section 3.1. For Automatic Seasonal Features, we retain the top $k = 5$ periodicities identified by the spectral analysis described in Section 3.1. This value provides a good balance between expressiveness and computational efficiency: increasing k adds additional feature dimensions, but we found in preliminary checks that the model's forecasting accuracy is relatively stable across reasonable values of k. A more extensive study of this parameter is left for future work.

[3] https://huggingface.co/Prior-Labs/TabPFN-v2-reg/blob/9ff6a7be140b2663cdd21387a5decc3a8018ad6b/tabpfn-v2-regression-2noar4o2.ckpt

A.4 GIFT-Eval Benchmark Datasets and Corresponding Statistics

Each benchmarking task in GIFT-Eval corresponds to a unique combination of dataset, prediction horizon (short-, medium-, or long-term), and sampling frequency (where applicable). For a given dataset, a benchmarking task is defined only if sufficient historical data is available to support the specified window size and forecast length, as shown in the short-, medium-, and long-term columns of Table 1. In total, GIFT-Eval comprises 97 such tasks that span diverse domains, temporal resolutions, and forecasting lengths.

These 97 tasks are used in the main experimental evaluation. For the ablation studies, we exclude datasets marked with an asterisk (*) due to their relatively large size and higher resource requirements.

Table 1: Statistics of datasets from the GIFT-Eval benchmark (reproduced from Aksu et al. (2024) under a CC BY 4.0 license). Datasets marked with an asterisk (*) are excluded from ablation studies due to the large size.

Dataset	Domain	Frequency	# Series	Series Length			# Obs	# Target Variates	Short-term		Med-term		Long-term	
				Avg	Min	Max			Pred Length	Windows	Pred Length	Windows	Pred Length	Windows
Jena Weather	Nature	10T	1	52,704	52,704	52,704	52,704	21	48	20	480	11	720	8
Jena Weather	Nature	H	1	8,784	8,784	8,784	8,784	21	48	19	480	2	720	2
Jena Weather	Nature	D	1	366	366	366	366	21	30	2				
BizITObs - Application	Web/CloudOps	10S	1	8,834	8,834	8,834	8,834	2	60	15	600	2	900	1
BizITObs - Service	Web/CloudOps	10S	21	8,835	8,835	8,835	185,535	2	60	15	600	2	900	1
BizITObs - L2C	Web/CloudOps	5T	1	31,968	31,968	31,968	31,968	7	48	20	480	7	720	5
BizITObs - L2C	Web/CloudOps	H	1	2,664	2,664	2,664	2,664	7	48	6	480	1	720	1
Bitbrains - Fast Storage	Web/CloudOps	5T*	1,250	8,640	8,640	8,640	10,800,000	2	48	18	480	2	720	2
Bitbrains - Fast Storage	Web/CloudOps	H	1,250	721	721	721	901,250	2	48	2				
Bitbrains - rnd*	Web/CloudOps	5T	500	8,640	8,640	8,640	4,320,000	2	48	18	480	2	720	2
Bitbrains - rnd	Web/CloudOps	H	500	720	720	720	360,000	2	48	2				
Restaurant	Sales	D	807	358	67	478	289,303	1	30	1				
ETT1	Energy	15T	1	69,680	69,680	69,680	69,680	7	48	20	480	15	720	10
ETT1	Energy	H	1	17,420	17,420	17,420	17,420	7	48	20	480	4	720	3
ETT1	Energy	D	1	725	725	725	725	7	30	3				
ETT1	Energy	W-THU	1	103	103	103	103	7	8	2				
ETT2	Energy	15T	1	69,680	69,680	69,680	69,680	7	48	20	480	15	720	10
ETT2	Energy	H	1	17,420	17,420	17,420	17,420	7	48	20	480	4	720	3
ETT2	Energy	D	1	725	725	725	725	7	30	3				
ETT2	Energy	W-THU	1	103	103	103	103	7	8	2				
Loop Seattle*	Transport	5T	323	105,120	105,120	105,120	33,953,760	1	48	20	480	20	720	15
Loop Seattle*	Transport	H	323	8,760	8,760	8,760	2,829,480	1	48	19	480	2	720	2
Loop Seattle	Transport	D	323	365	365	365	117,895	1	30	2				
SZ-Taxi	Transport	15T	156	2,976	2,976	2,976	464,256	1	48	7	480	1	720	1
SZ-Taxi	Transport	H	156	744	744	744	116,064	1	48	2				
M_DENSE	Transport	H	30	17,520	17,520	17,520	525,600	1	48	20	480	4	720	3
M_DENSE	Transport	D	30	730	730	730	21,900	1	30	3				
Solar	Energy	10T	137	52,560	52,560	52,560	7,200,720	1	48	20	480	11	720	8
Solar	Energy	H	137	8,760	8,760	8,760	1,200,120	1	48	19	480	2	720	2
Solar	Energy	D	137	365	365	365	50,005	1	30	2				
Solar	Energy	W-FRI	137	52	52	52	7,124	1	8	1				
Hierarchical Sales	Sales	D	118	1,825	1,825	1,825	215,350	1	30	7				
Hierarchical Sales	Sales	W-WED	118	260	260	260	30,680	1	8	4				
M4 Yearly	Econ/Fin	A-DEC	22,974	37	19	284	845,109	1	6	1				
M4 Quarterly	Econ/Fin	Q-DEC	24,000	100	24	874	2,406,108	1	8	1				
M4 Monthly	Econ/Fin	M	48,000	234	60	2,812	11,246,411	1	18	1				
M4 Weekly	Econ/Fin	W-SUN	359	1,035	93	2,610	371,579	1	13	1				
M4 Daily	Econ/Fin	D	4,227	2,371	107	9,933	10,023,836	1	14	1				
M4 Hourly	Econ/Fin	H	414	902	748	1,008	373,372	1	48	2				
Hospital	Healthcare	M	767	84	84	84	64,428	1	12	1				
COVID Deaths	Healthcare	D	266	212	212	212	56,392	1	30	1				
US Births	Healthcare	D	1	7,305	7,305	7,305	7,305	1	30	20				
US Births	Healthcare	W-TUE	1	1,043	1,043	1,043	1,043	1	8	14				
US Births	Healthcare	M	1	240	240	240	240	1	12	2				
Saugeen	Nature	D	1	23,741	23,741	23,741	23,741	1	30	20				
Saugeen	Nature	W-THU	1	3,391	3,391	3,391	3,391	1	8	20				
Saugeen	Nature	M	1	780	780	780		1	12	7				
Temperature Rain*	Nature	D	32,072	725	725	725	780	1	30	3				
KDD Cup 2018	Nature	H	270	10,898	9,504	10,920	2,942,364	1	48	20	480	2	720	2
KDD Cup 2018	Nature	D	270	455	396	455	122,791	1	30	2				
Car Parts	Sales	M	2,674	51	51	51	136,374	1	12	1				
Electricity*	Energy	15T	370	140,256	140,256	140,256	51,894,720	1	48	20	480	20	720	20
Electricity	Energy	H	370	35,064	35,064	35,064	12,973,680	1	48	20	480	8	720	5
Electricity	Energy	D	370	1,461	1,461	1,461	540,570	1	30	5				
Electricity	Energy	W-FRI	370	208	208	208	76,960	1	8	3				

A.5 Additional Results on GIFT-Eval

Figures A.1–A.3 present example predictions from TabPFN-TS on randomly selected samples from short-, medium-, and long-term forecasting tasks, respectively.

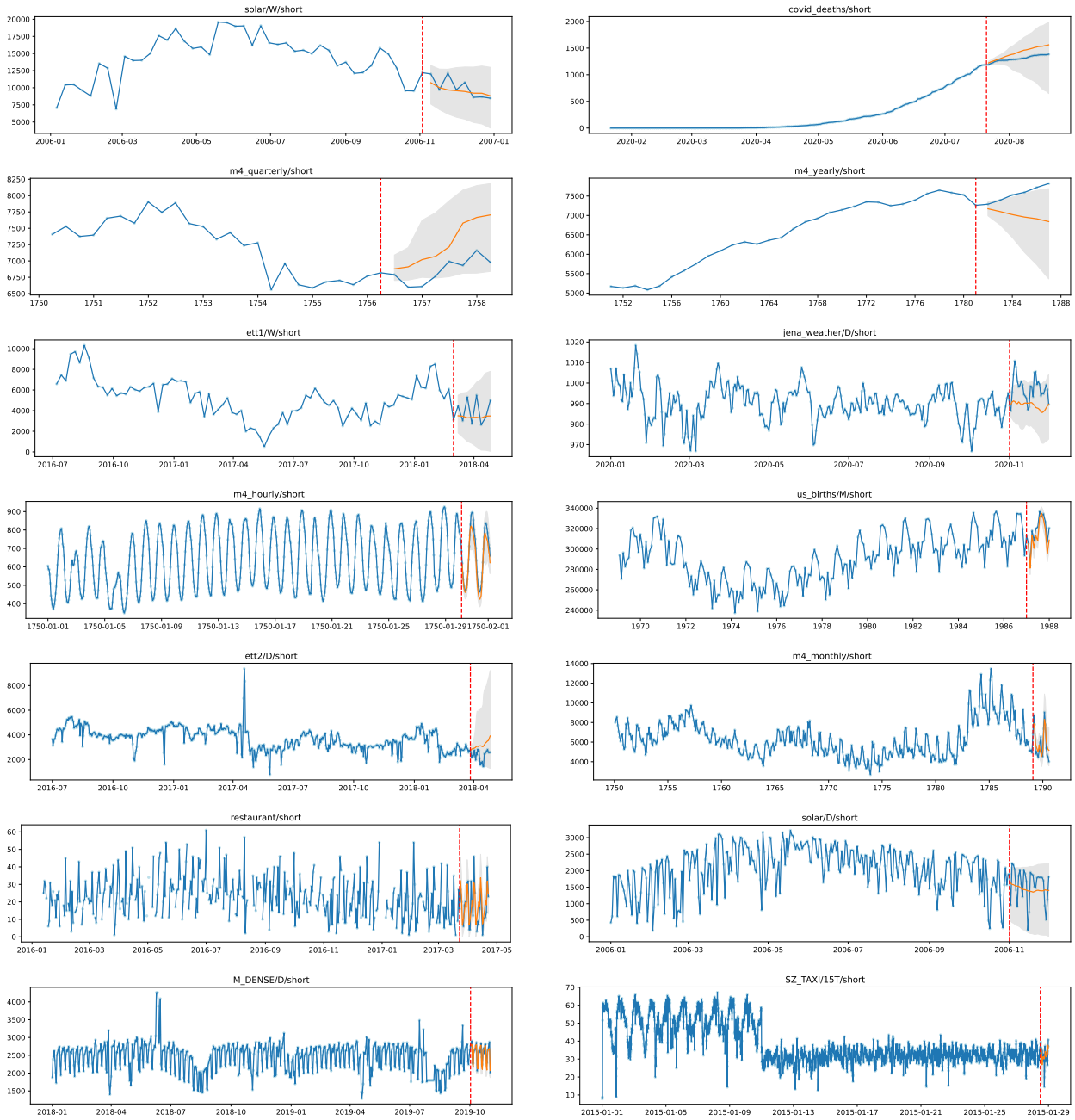


Figure A.1: Visualization of the TabPFN-TS' predictions on some of the **short-term** benchmarking tasks.

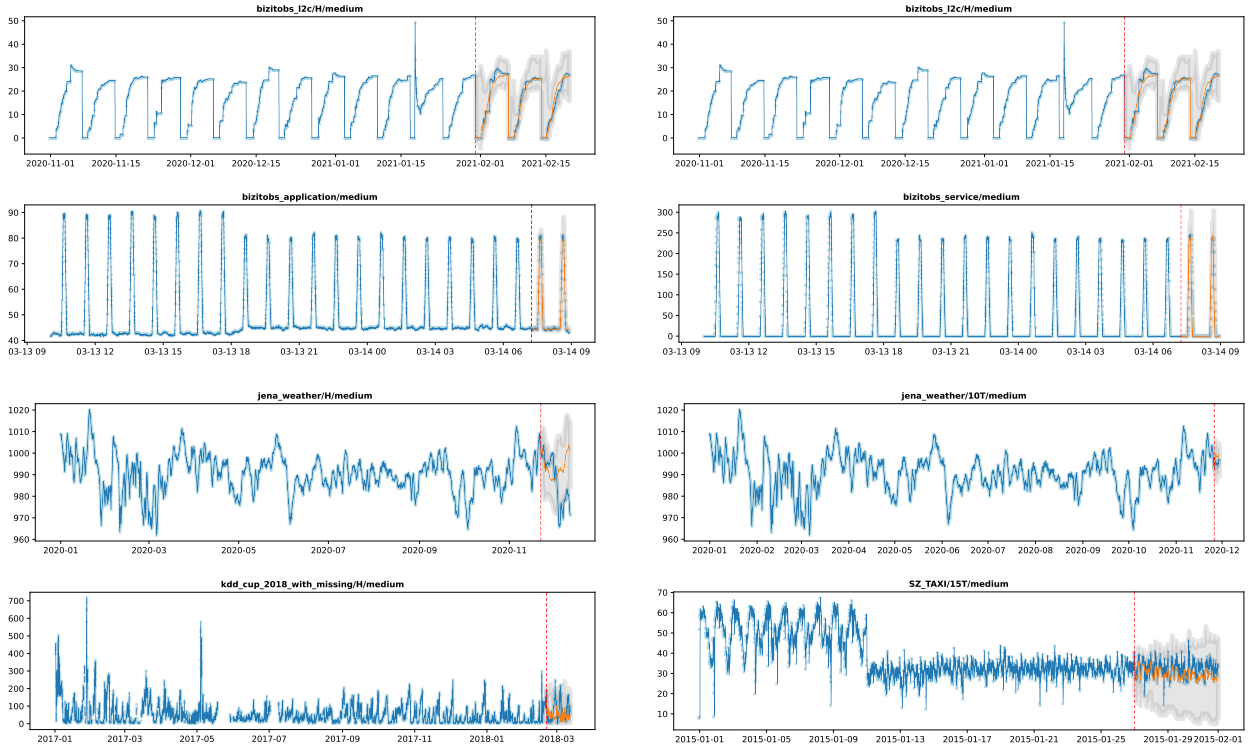


Figure A.2: Visualization of the TabPFN-TS' predictions on some of the **medium-term** benchmarking tasks.

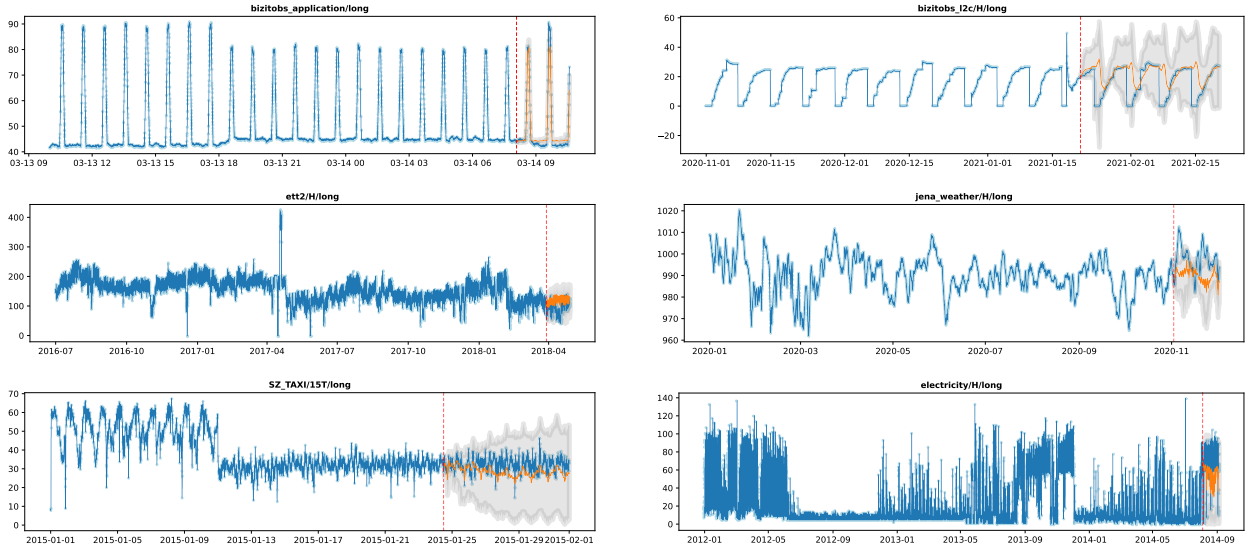


Figure A.3: Visualization of the TabPFN-TS' predictions on some of the **long-term** benchmarking tasks.

A.6 Covariate-Informed Forecasting Evaluation on `fev-bench`

We evaluate on the subset of `fev-bench` tasks that include covariates in both the conditioning and prediction windows (commonly referred to in the time-series literature as past and known future covariates). The tasks used are listed in Table 2. Detailed metadata are provided in the original benchmark paper (Shchur et al., 2025). As these details are already standardized and publicly available, we refer readers to the benchmark documentation rather than replicate the full tables here.

#	`fev-bench` Tasks
1	proenfo_gfc12
2	proenfo_gfc14
3	proenfo_gfc17
4	rohlik_orders_1D
5	rohlik_sales_1W
6	rohlik_orders_1W
7	entsoe_15T
8	entsoe_30T
9	entsoe_1H
10	epf_be
11	epf_de
12	epf_fr
13	epf_np
14	epf_pjm
15	rossmann_1D
16	rossmann_1W
17	hermes
18	walmart
19	m5_1W
20	m5_1M
21	favorita_stores_1D
22	favorita_stores_1W
23	favorita_stores_1M
24	favorita_transactions_1D
25	solar_with_weather_15T
26	solar_with_weather_1H
27	uci_air_quality_1H
28	uci_air_quality_1D

Table 2: Subset of `fev-bench` tasks used for covariate-informed forecasting. Detailed task characteristics are available in the original benchmark paper (Shchur et al., 2025).

A.7 Additional Results for Harmonic Reconstruction Experiments

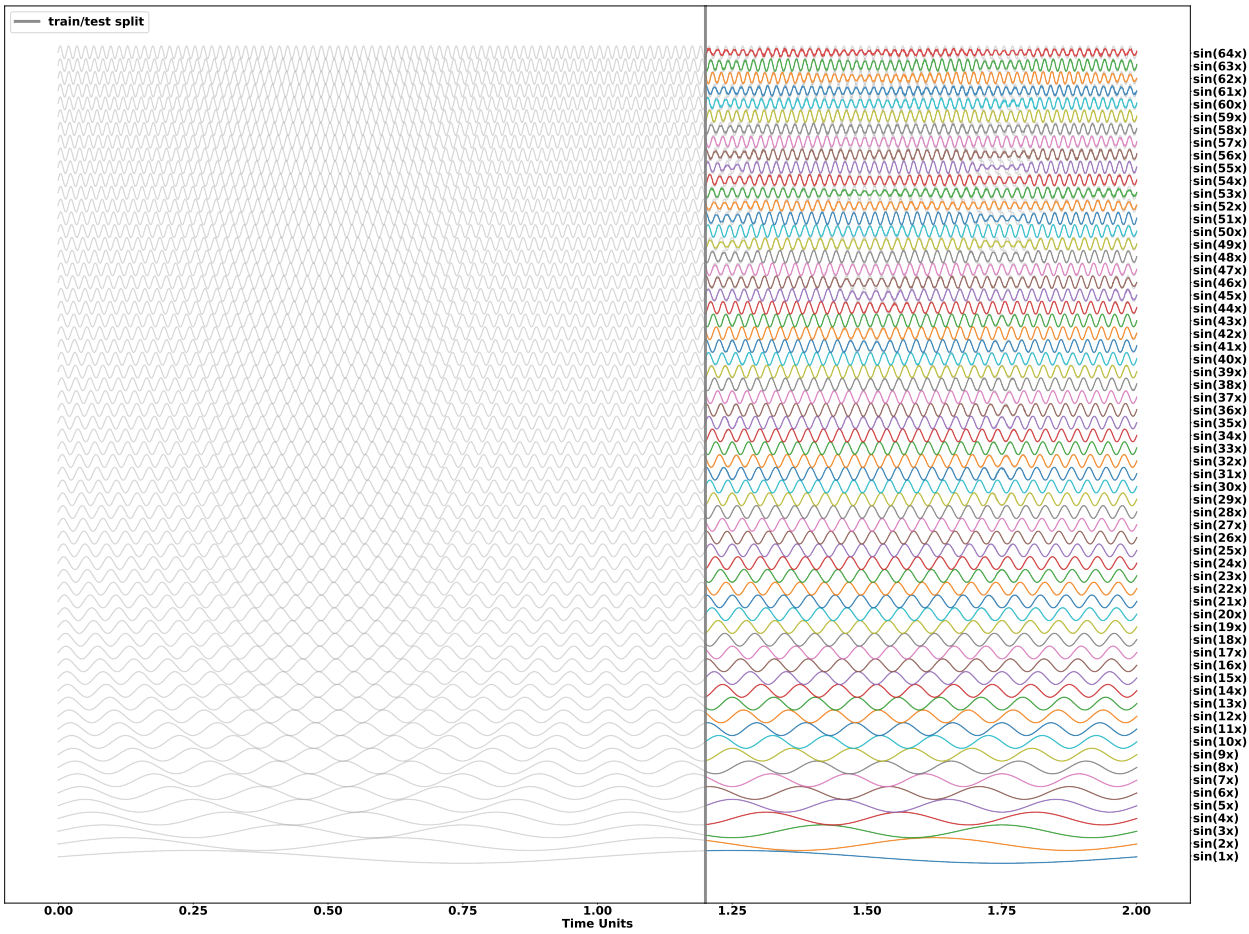


Figure A.4: Reconstruction of $\sin(nt)$ under the injective temporal embedding $\{\sin(t), \cos(t)\}$. TabPFN-v2 is evaluated on $\sin(nt)$ for $n = 1, \ldots, 64$. The model accurately reconstructs low and mid-range frequencies, but accuracy gradually degrades at higher n, consistent with the finite phase resolution of the sinusoidal embedding given the sampling density.

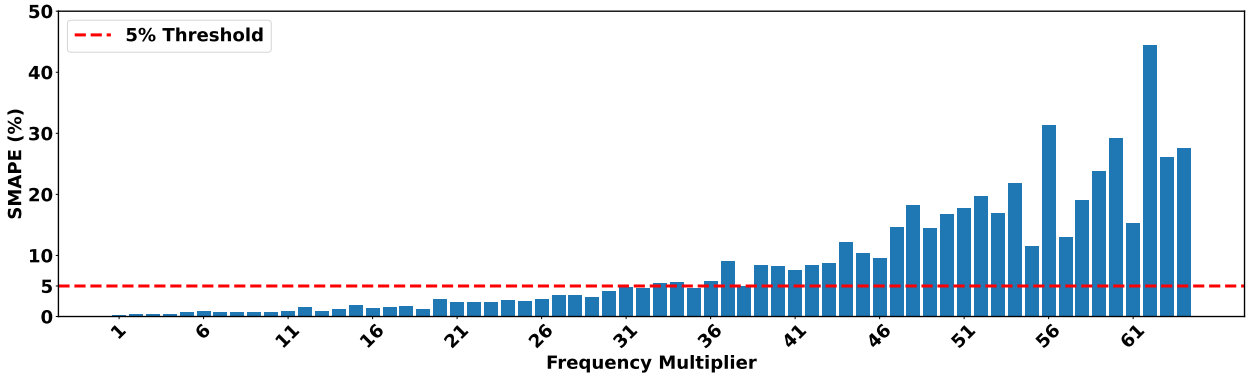


Figure A.5: Symmetric mean absolute percentage error (sMAPE) for predicting $\sin(nt)$ using $\{\sin(t), \cos(t)\}$ as input features. Errors remain low for moderate frequencies but increase as n grows, reflecting the reduced separability of high-frequency phases in the sinusoidal coordinate system.

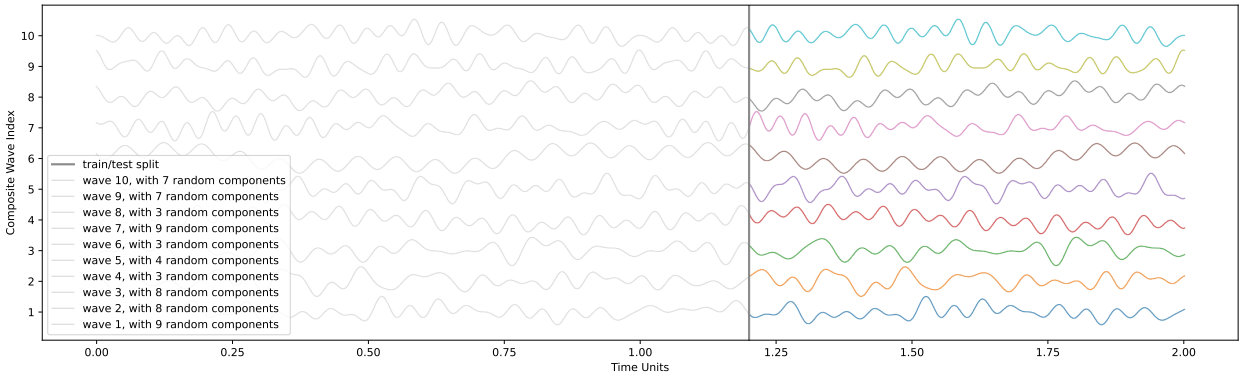


Figure A.6: Reconstruction of composite sinusoidal signals under the injective temporal embedding $\{\sin(t), \cos(t)\}$. Each composite signal is the sum of 3–10 sinusoids with random frequencies, amplitudes, and phases. TabPFN-v2 accurately recovers these multi-frequency patterns, consistent with interpolation in a phase-preserving temporal embedding.

A.8 Synthetic Data Generation for Time Series with Covariates

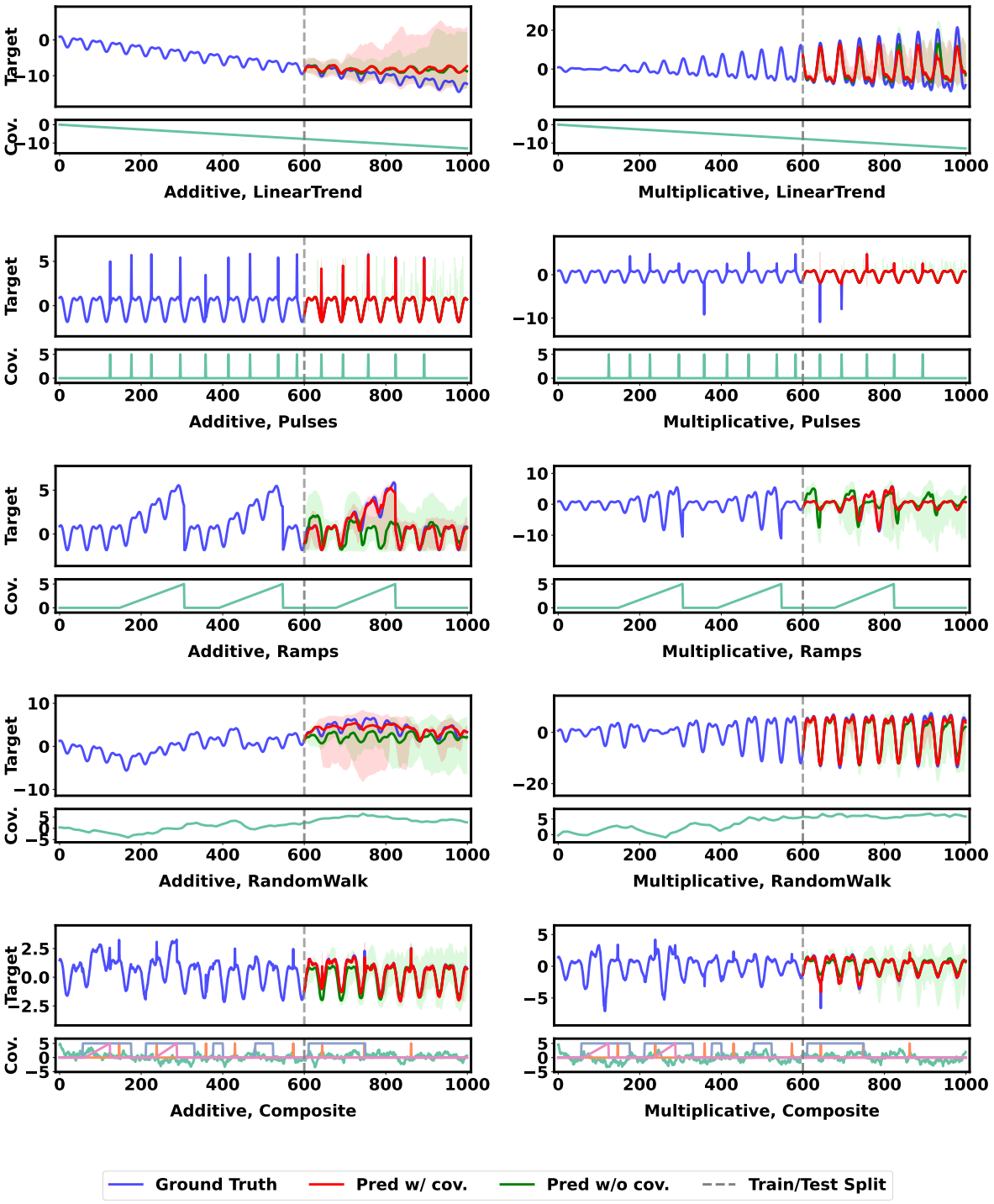


Figure A.7: Forecasting visualization on synthetic covariate-augmented time series. Each subplot corresponds to one synthetic class from $\mathcal{C} = \text{additive}, \text{multiplicative} \times \text{LinearTrend}, \text{Pulses}, \text{Ramps}, \text{RandomWalk}, \text{Composite}$. Blue lines denote ground truth, red lines predictions with covariates, and green lines predictions without covariates. Shaded regions indicate predictive uncertainty, and the vertical dashed line marks the train/test split.

A.9 Ablation: Context Length vs. Accuracy

In this ablation, we investigate how the amount of available context affects the performance of TabPFN-TS. We experiment with four context lengths: 1024, 2048, 4096, and 10,000. The maximum length of 10,000 is chosen to match the largest dataset size used during the pretraining of TabPFN-v2.

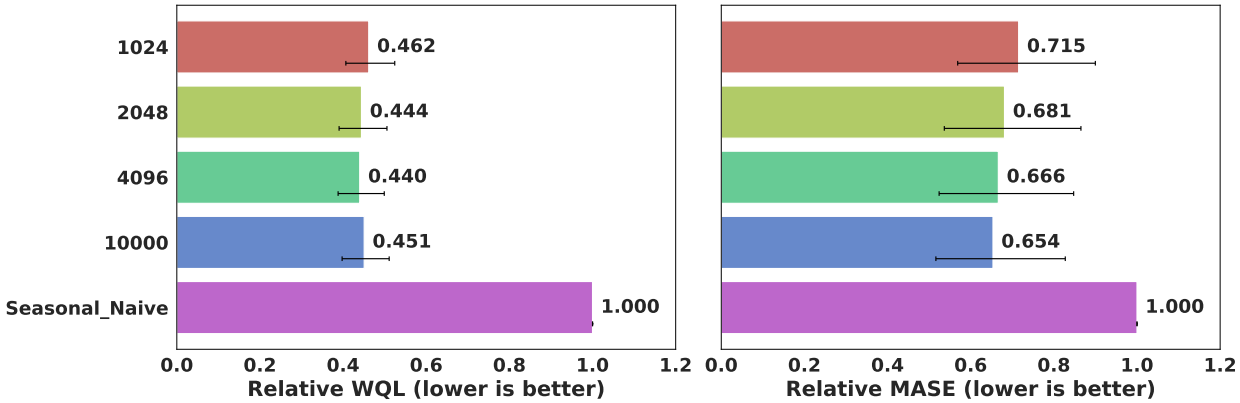


Figure A.8: Effect of context length on forecasting performance of TabPFN-TS.

As shown in Figure A.8, increasing the context length leads to improved performance overall, though the gains diminish beyond 4096 points. While MASE continues to improve with longer context, WQL shows a slight increase at the longest length. These results suggest that moderate-length contexts are often sufficient, but the impact of longer contexts may vary depending on the forecasting objective.