TS-ICL: A Flexible Time-Indexed Foundation Model for Time Series via In-Context Learning

Etienne Le Naour*
EDF R&D
Palaiseau, France
etiennelenaour09@gmail.com

Tahar Nabil*
EDF R&D
Palaiseau, France
tahar.nabil@edf.fr

Adrien Petralia
EDF R&D
Palaiseau, France
adrien.petralia@gmail.com

Abstract

Foundation models mark a profound paradigm shift in time series modeling, with task-specific models being superseded by general-purpose zero-shot models. Yet, current approaches primarily focus on forecasting, while real-world time series are often irregularly and partially observed, requiring models that can jointly forecast, impute missing values, and handle degraded sampling conditions. To address these challenges, we introduce `TS-ICL`, a novel probabilistic In-Context Learning encoder–regressor Transformer that unifies forecasting and imputation. `TS-ICL` formulates time series tasks as timestamp-aligned regression and naturally incorporates covariates by training on synthetic dependency structures generated from a novel causal data prior. Empirically, `TS-ICL` achieves a new state-of-the-art in imputation, while remaining competitive with leading forecasting foundation models across both univariate and covariate-aware benchmarks. It shows particularly strong performance in forecasting with partially observed look-back windows. *Code page*: https://github.com/EDF-Lab/ts-icl.

1 Introduction

The recent advent of Time Series Foundation Models (TSFMs) drastically changes time series modeling, shifting from task-specific to transferable models that leverage large-scale pretraining and adaptation mechanisms to provide zero-shot inference on unseen data [4, 28, 10, 29]. TSFMs address limited data regimes and distribution shifts [45], but still face fundamental limitations. First, many tasks cannot be solved in practice without leveraging external information, requiring covariate-aware inference [41]. Second, real-world observations are often incomplete, with missing values and asynchronous measurements [9]. This challenges standard modeling assumptions and motivates unified frameworks that can jointly handle forecasting and imputation in flexible observation settings.

To address these challenges, (1) recent TSFMs such as `Chronos-2` [3] build on In-Context Learning [5] (ICL) to support covariate-informed inference and handle missing values, while remaining highly efficient. Nevertheless, they do not natively address imputation, thus hindering their practical use. (2) In a remarkably different approach, `TabPFN` [17] and `TabICLv2` [35] have revolutionized the tabular domain with strong few-shot regression capabilities via Transformer-based ICL and synthetic data priors. When adapted to time series (e.g., `TabPFN-TS` [19]), these Tabular Foundation Models (TFMs) naturally support covariates and enable both zero-shot imputation [27] and forecasting [39]. Yet, they lack temporal inductive bias and rely on handcrafted time features. As a result, TFMs lag behind pure TSFMs in forecasting benchmarks, while also incurring high inference cost [39].

Consequently, existing approaches fail to jointly provide (i) unified forecasting and imputation, (ii) covariate-aware inference, and (iii) efficient zero-shot performance (see Table 1). In this paper, we tackle this challenge and introduce `TS-ICL`, a unified probabilistic Transformer foundation

*Equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Table 1: Capabilities of recent time series foundation models. Only TS-ICL supports forecasting and imputation, while enabling efficient covariate-aware inference and supporting irregular sampling.

Method	Handles Forecasting	Handles Imputation	Covariate Integration	Probabilistic Predictions	Designed for Time Series	Irregular Sampling	Fast Inference
TiRex [4], Toto [10], TimesFMv2.5 [11]	✓	✗	✗	✓	✓	✗	✓
Chronos-2 [3]	✓	✗	✓	✓	✓	✗	✓
TabPFNv2.5-TS [19], TabICLv2-TS [35]	✓	✓	✓	✓	✗	✓	✗
TS-ICL **(ours)**	✓	✓	✓	✓	✓	✓	✓

model for imputation and forecasting. (i) TS-ICL casts time series modeling as an in-context regression problem, where observations are represented as timestamp-aligned inputs and encoded into contextual representations that enable forecasting or imputation in a single forward pass. (ii) To enable effective covariate-aware inference, a structured synthetic prior over target–covariate relationships is introduced using Directed Acyclic Graphs (DAGs) to define dependency structures, with node-level mechanisms inspired by structural causal models [33, 17]. (iii) Unlike existing TSFMs, TS-ICL operates directly on timestamped observations rather than fixed grids, allowing flexible handling of missing or irregularly sampled data in practice.

Contributions. The main contributions are as follows:

- **A unified and flexible TSFM architecture.** We introduce TS-ICL, a novel probabilistic TSFM that casts time series modeling as a time-indexed in-context regression problem, unifying forecasting and imputation with native support for covariates.
- **Structured synthetic prior.** We design a novel DAG-based causal prior over target–covariate time series, enabling robust zero-shot generalization to unseen dependency structures.
- **State-of-the-art imputation performance.** TS-ICL sets a new state-of-the-art on zero-shot imputation benchmarks, consistently outperforming both task-specific models and TFMs, while being up to $50\times$ faster than TFMs at inference.
- **Competitive forecasting performance.** On forecasting benchmarks, TS-ICL matches state-of-the-art TSFMs while supporting covariate-aware inference, and remains particularly robust to missing observations due to its time-indexed formulation.

2 Related Work

Time series foundation models. Time Series Foundation Models (TSFMs) are pretrained general-purpose models for time series, typically trained on large mixtures of real and synthetic data and often based on patch-based architectures [28, 4, 10, 11]. While they achieve strong zero-shot forecasting performance on standard benchmarks [1, 34], they exhibit several limitations in practical settings: they are typically not designed for covariate-aware inference, and primarily focus on forecasting without addressing imputation. Chronos-2 [3] partially mitigates the covariate limitation by enabling inference-time conditioning on exogenous time series, leading to improved performance when informative covariates are available. However, its patch-based formulation assumes regularly sampled inputs and does not natively support imputation, which is critical in many real-world time series applications [37, 9, 6, 13].

Tabular foundation models for time series. Tabular Foundation Models (TFMs) such as TabPFN [17] and TabICLv2 [35] leverage in-context learning over synthetic task distributions to achieve strong few-shot regression performance. Extensions to time series [19] demonstrate competitive results for both forecasting [39] and imputation [27], while naturally supporting covariates at inference time. However, these approaches typically rely on handcrafted temporal features, such as Fourier features operating at predefined frequencies, and incur higher inference costs compared to dedicated TSFMs [39], limiting their scalability in practice.

Supervised imputation models. Classical supervised imputation methods such as BRITS [6] and SAITS [13] achieve strong performance in in-domain settings but require task-specific training and generalize poorly across domains. Recent large-scale evaluations [27] suggest that TFM-based approaches can significantly outperform these methods in realistic scenarios.

Time-indexed and Neural Field models. Continuous-time modeling of time series (also referred to as time-indexed modeling [43]) has been explored through Neural Ordinary Differential Equations [8] (ODE) and latent ODE frameworks [36]. More recently, Neural Field-based representations [26, 38] encode irregular observations into continuous latent functions without requiring explicit temporal alignment. These approaches provide a principled foundation for flexible time representations, but are not designed for zero-shot inference in forecasting or imputation settings.

Despite these advances, existing approaches remain fragmented across forecasting, imputation, and representation learning. No framework jointly provides efficient zero-shot inference, covariate-aware modeling, and a unified treatment of forecasting and imputation within a single architecture.

3 TS-ICL Architecture

This section introduces the proposed TS-ICL architecture, designed for efficient probabilistic zero-shot time series forecasting and imputation of univariate targets, while maintaining flexibility in handling covariates and irregularly sampled observations. A high-level overview of the architecture and its main components is provided below, while general intuition and detailed descriptions of each module are deferred to Appendix A.

Problem setting. Let $\boldsymbol{x} = (x_t)_{t\in\mathcal{T}}$ denote a time series defined over a (possibly irregular) set of timestamps $\mathcal{T}$. The index set $\mathcal{T}$ is partitioned into two disjoint subsets: (i) the context timestamps $\mathcal{T}^{\text{ctxt}}$, corresponding to observed values, and (ii) the target timestamps $\mathcal{T}^{\text{tgt}}$, corresponding to values to be predicted. Accordingly, $\boldsymbol{x}^{\text{ctxt}} = (x_t)_{t\in\mathcal{T}^{\text{ctxt}}}$ denotes the observed context values, and $\boldsymbol{x}^{\text{tgt}} = (x_t)_{t\in\mathcal{T}^{\text{tgt}}}$ the target values.

Additionally, an optional set of exogenous covariate time series is considered:

$$\boldsymbol{X}^{\text{covar}} = \left\{\boldsymbol{x}_c^{\text{covar}} = (x_{c,t}^{\text{covar}})_{t\in\mathcal{T}_c^{\text{covar}}}\right\}_{c=1}^{C-1},$$

where C refers to the channel dimension, with one channel corresponding to the time series of interest and the remaining $C-1$ channels corresponding to exogenous covariates.

Each covariate is defined over its own set of timestamps $\mathcal{T}_c^{\text{covar}} \subseteq \mathcal{T}$. Covariates may be observed over arbitrary subsets of timestamps, including only the context (e.g. the look-back window in forecasting) or both context and target (e.g. look-back and horizon windows), and may be sparsely observed. The TS-ICL framework can handle such heterogeneous availability without requiring any imputation preprocessing.

The objective, common to both forecasting and imputation, is to infer the target values conditioned on the observed context and, when available, the covariates:

$$p\big(\boldsymbol{x}^{\text{tgt}} \mid \boldsymbol{x}^{\text{ctxt}}, \boldsymbol{X}^{\text{covar}}\big).$$

This formulation unifies forecasting and imputation as conditional inference problems. For clarity, the next section omits batch and timestamp indices whenever no ambiguity arises.

3.1 TS-ICL Overview

The key idea behind TS-ICL is to reformulate time series prediction as an in-context regression problem over learned temporal representations. Thus, the architecture is composed of four successive modules that transform raw observations into global and local context-aware representations used for prediction. The pipeline first encodes each time series, then aggregates information across covariates, and finally produces timestamp-specific representations that enable in-context regression. The overall architecture is illustrated in Figure 1.

(i) Time Series Encoder $\mathcal{E}$. This module extracts representations from the context time series and optional $C-1$ covariates in a channel-independent manner. It follows a Perceiver encoder design [21, 38] based on a fixed set of M learnable tokens that sequentially cross-attend to timestamp–value pairs. This allows compressing an arbitrary-length input into a compact latent sequence.

$$(\boldsymbol{x}^{\text{ctxt}}, \mathcal{T}^{\text{ctxt}}, \boldsymbol{X}^{\text{covar}}, \mathcal{T}^{\text{covar}}) \xrightarrow{\mathcal{E}} \boldsymbol{Z}^{\text{val}} \in \mathbb{R}^{C\times M\times d}.$$

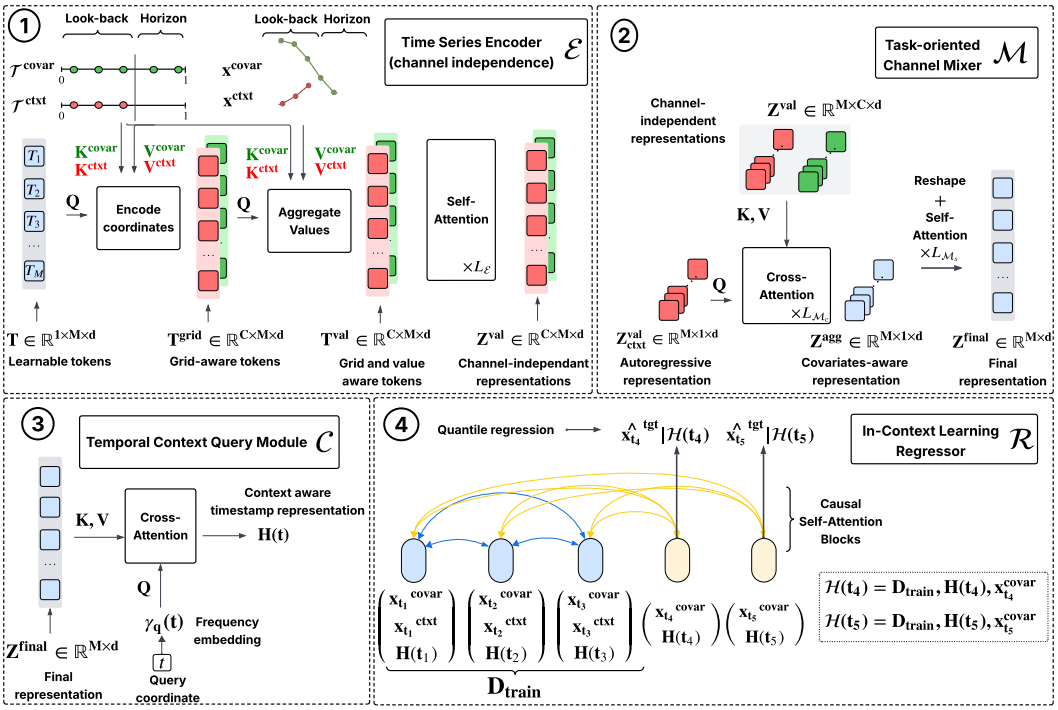


Figure 1: The TS-ICL pipeline. From temporal encoding to in-context regression. The diagram illustrates the four-module transformation for forecasting with one covariate observed on the horizon.

(ii) Channel Mixer $\mathcal{M}$. This module aggregates information across the C channels to produce a unified representation. For each of the M latent positions, the token corresponding to the time series of interest queries the tokens of the $C-1$ covariates via cross-attention. This mechanism collapses the channel dimension by selectively integrating exogenous information into the main series' representation. A subsequent stack of self-attention layers models global dependencies among the resulting M tokens, yielding a compact task-oriented representation. This step is critical to capture inter-channel dependencies, which are not modeled by the channel-independent encoder.

$$\boldsymbol{Z}^{\text{val}} \in \mathbb{R}^{C\times M\times d} \xrightarrow{\mathcal{M}} \boldsymbol{Z}^{\text{final}} \in \mathbb{R}^{M\times d}.$$

(iii) Temporal Context Query Module $\mathcal{C}$. This modules aims at bridging the gap between the discrete latent tokens $\boldsymbol{Z}^{\text{final}}$ and the continuous time domain. Any timestamp $t \in \mathcal{T}$ is first embedded using a frequency-based positional encoding [30]. This local encoding then cross-attends to the time series representation $\boldsymbol{Z}^{\text{final}}$, providing a single local and global context-aware representation of arbitrary timestamps. This design enables querying at arbitrary timestamps, supporting irregular sampling and unifying forecasting and imputation.

$$(t, \boldsymbol{Z}^{\text{final}}) \xrightarrow{\mathcal{C}} \boldsymbol{H}(t) \in \mathbb{R}^d.$$

(iv) In-Context Learning Regressor $\mathcal{R}$. Given the representations $\boldsymbol{H}(t)$, prediction is formulated as an in-context regression problem [15, 42], where the observed context defines a training set:

$$\mathcal{D}_{\text{train}} = \{\boldsymbol{H}(t), \boldsymbol{x}_t^{\text{ctxt}}, \boldsymbol{X}_t^{\text{covar}}\}_{t\in\mathcal{T}^{\text{ctxt}}},$$

used to infer target values at unseen target timestamps t^{tgt}. The regressor is implemented as a Transformer that performs causal self-attention over the training (context) set, effectively learning to map representations to values in an in-context manner. When available, covariates are incorporated in the training set via cross-attention, allowing the model to condition on exogenous time series under arbitrary availability patterns (e.g., context-only, context and target, or sparse observations). TS-ICL outputs a dense set of quantile estimates of the target distribution, trained using a smoothed pinball loss [40].

$$\boldsymbol{H}(t^{\text{tgt}}), \boldsymbol{X}_{t^{\text{tgt}}}^{\text{covar}} \xrightarrow{\mathcal{R}} p\big(\boldsymbol{x}^{\text{tgt}} \mid \boldsymbol{H}(t^{\text{tgt}}), \boldsymbol{X}_{t^{\text{tgt}}}^{\text{covar}}, \mathcal{D}_{\text{train}}\big).$$

This four-step formulation unifies time series representation learning and in-context regression, enabling TS-ICL to perform forecasting and imputation while flexibly operating over timestamped observations and optionally observed covariates. Additional architectural details and hyperparameters are provided in Appendix A and Appendix B.2, respectively.

4 Data Prior and Training Procedure

TS-ICL is trained on a structured data prior combining real-world and synthetic time series, spanning both univariate signals and multivariate covariate–target structures. This prior is designed to jointly capture temporal dynamics and inter-variable dependencies within a unified training distribution. Concretely, training samples consist of either univariate time series or multivariate problems where target–covariate relationships are generated via structured transformations of base signals.

Univariate time series. In the univariate setting, a large and highly heterogeneous pretraining prior is constructed by combining real-world and synthetic time series. This mixture is designed to expose TS-ICL to a broad spectrum of temporal dynamics. For *real-world data*, the prior leverages a collection of 31 datasets spanning multiple domains, as detailed in Appendix B.1.1. These datasets cover diverse temporal phenomena, including trends, seasonal patterns, regime shifts, and varying levels of non-stationarity. The training distribution is further augmented with *synthetic data* sampled from the TempoPFN univariate time series generator [31], which induces controlled but diverse stochastic processes. Overall, this yields a large-scale pretraining distribution over univariate time series, comprising 40 datasets and approximately 2M time series with lengths ranging from ~ 100 to $\sim$ 600k time steps.

Covariates generator. To enable TS-ICL to learn under structured covariates, synthetic multivariate time series are constructed from base univariate signals (either real or generated as described above). Specifically, (i) A Directed Acyclic Graph (DAG) is generated over univariate signals where nodes correspond to time series and edges encode causal dependencies. Following [35], each non-root node is produced by applying a transformation sampled from a Structural Causal Model (SCM) registry, including both linear and non–linear operators (e.g., linear mappings, MLP, RNN). This yields heterogeneous dependency structures while preserving temporal coherence. (ii) Given the resulting graph, one node is selected as the prediction target and a subset of the remaining nodes is sampled as covariates, producing multivariate problems with varying numbers of covariates and controllable dependency strength. This design ensures that covariates can be causally informative, redundant, or entirely independent of the target, preventing reliance on spurious correlations. Figure 2 illustrates the synthetic target–covariate generation process. Additional details on the data prior are provided in Appendix B.1.2.

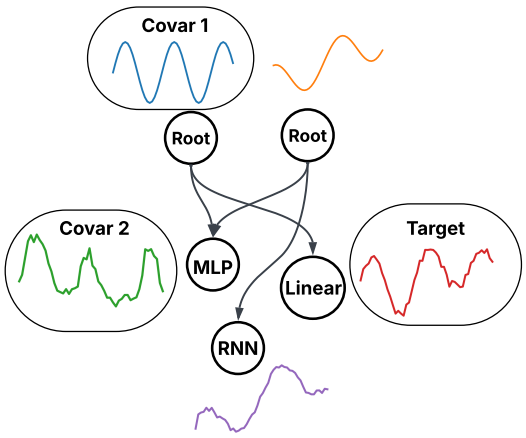


Figure 2: Synthetic target–covariate generation. Multivariate structures are constructed from a sampled DAG, where base signals are transformed via linear and non-linear SCMs. One node is selected as the target, while others serve as informative or redundant covariates.

Whole procedure. Overall, each training sample is constructed by first sampling base time series (real or synthetic), which may either be used directly or serve as building blocks for multivariate structures via the DAG-based generator. This yields a unified training distribution over univariate and multivariate time series. The training task is then defined dynamically. • For imputation, observations are masked either point-wise or in contiguous segments with randomly sampled masking ratios. • For forecasting, a future horizon of random length is masked. The full data generation pipeline is summarized in Algorithm 1 in Appendix B.1.3, while hyperparameters and training details are reported in Appendix B.2.

5 Experiments

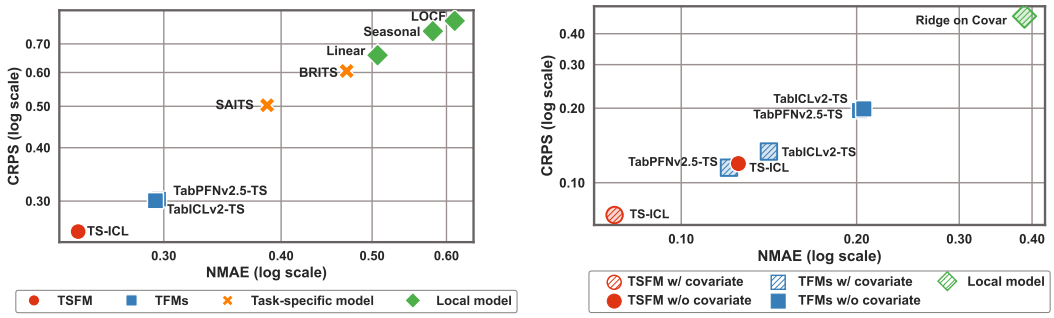
TS-ICL is evaluated on zero-shot imputation (Section 5.1) and forecasting (Section 5.2) under two settings: (i) *univariate time series* and (ii) *time series with known covariates* available at inference time. All experiments use a 27M-parameter version of TS-ICL (see Appendix B.2 for architectural details). Imputation experiments rely on `fm-impute-bench` [27], while forecasting is evaluated on `fev-bench` [39], following recent protocols for time series foundation models. Ablation studies on TS-ICL components are provided in Appendix C and additional results on the TIME benchmark [34] are presented in Appendices E and F.

5.1 Imputation Experiments

The zero-shot imputation capability of TS-ICL is evaluated on `fm-impute-bench` [27], covering both *univariate* settings and scenarios with *covariates* available at inference time. The benchmark spans diverse missingness patterns, sequence lengths, and application domains.

Setting. (i) In the *univariate setting*, `fm-impute-bench` comprises 33 datasets across multiple domains (e.g., energy, climate, etc.) with varying lengths and frequencies (see Table 10, Appendix E.1). Each sample corresponds to a four-week window. TS-ICL is evaluated under four masking scenarios, namely: • 50% or • 70% pointwise missingness, and • two or • four disjoint one-day gaps. This results in 132 tasks and about 1.3M windows to impute. (ii) The same four masking scenarios are applied to the *known-covariates* setting on six datasets providing informative covariates (see Table 11, Appendix E.1). This results in 24 tasks and approximately 1K windows to impute.

Baselines. (i) *Univariate comparisons* include Tabular Foundation Models (TFMs) adapted to time series: `TabPFNv2.5-TS` [19], `TabICLv2-TS` [35], along with standard local methods: `Linear Interpolation`, `Seasonal Naive`, and `Last Observation Carried Forward (LOCF)`. In addition, supervised imputation models `SAITS` [13] and `BRITS` [6], trained per dataset, serve as strong task-specific baselines. (ii) In the *known-covariates* setting, the same foundation models are considered, together with ridge regression on covariates to quantify the predictive signal of exogenous variables. Variants of foundation models without covariates are also included for comparison. Following `fm-impute-bench`, pointwise performance is evaluated with Normalized Mean Absolute Error (NMAE) and probabilistic performance with Continuous Ranked Probability Score (CRPS). Metric definitions are provided in Appendix D. Figure 3 illustrates the trade-off between pointwise and probabilistic performance, while Table 2 reports the median inference time.

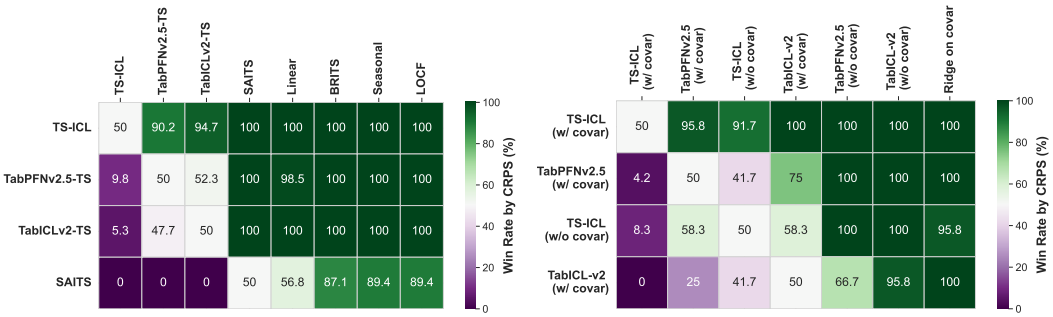


(a) *Univariate* imputation across 132 tasks. (b) Imputation with *known covariates* across 24 tasks.

Figure 3: NMAE-CRPS (lower is better) on the `fm-impute` benchmark. Each point corresponds to a method, averaged across tasks.

Results. TS-ICL sets a new state-of-the-art for zero-shot imputation, improving both pointwise and probabilistic scores over TFMs while being up to two orders of magnitude faster at inference.

(i) *Univariate setting.* As shown in Figure 3a, TS-ICL achieves lower NMAE and CRPS than competing TFMs, improving over `TabICLv2-TS` by 17% and 15%, respectively, while being $\sim 50\times$ faster at inference (Table 2). `TabPFNv2.5-TS` and `TabICLv2-TS` perform similarly and outperform



(a) *Univariate* imputation across 132 tasks. (b) Imputation with *known covariates* across 24 tasks.

Figure 4: Pairwise win rates of the top-4 models on the `fm-impute` benchmark. Each entry indicates the percentage of tasks where a method outperforms another according to the CRPS.

task-specific and local baselines by a wide margin. Pairwise win rates (Figure 4a) further show that `TS-ICL` dominates on the vast majority of tasks, indicating robustness across tasks.

(ii) *Covariate-aware setting.* Results in Figure 3b show similar trends when covariates are available. `TS-ICL` improves over `TabPFNv2.5-TS` by 36% in NMAE and 35% in CRPS, and gains 39% (NMAE) and 38% (CRPS) over its variant without covariates. While all TFMs benefit from covariates, they remain below `TS-ICL`. Ridge regression indicates limited predictive power of covariates alone, and pairwise comparisons (Figure 4b) demonstrate the clear dominance of `TS-ICL`.

Table 2: Median imputation inference time on `fm-impute-bench univar` with a H100 GPU.

	TSFM	Tabular Foundation models		Task-Specific Models		Local models		
	`TS-ICL`	`TabPFNv2.5-TS`	`TabICLv2-TS`	`SAITS`	`BRITS`	`Linear interpolation`	`Seasonal Naive`	`LOCF`
Inference time (s per window)	6.51×10^{-3}	2.80×10^{-1}	3.07×10^{-1}	1.33×10^{-2}	4.52×10^{-1}	1.61×10^{-4}	7.54×10^{-4}	5.58×10^{-4}

Overall, `TS-ICL` outperforms state-of-the-art TFMs for zero-shot imputation in both *univariate* and *covariate-informed* settings, while being two orders of magnitude faster at inference. Additional qualitative results in Appendix E, including Figures 18 and 19, as well as results on the `TIME` benchmark [34], further support these findings.

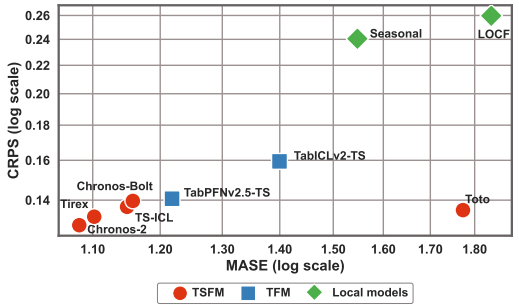
5.2 Forecasting Experiments

The zero-shot forecasting capability of `TS-ICL` is evaluated on `fev-bench` [39], a comprehensive benchmark covering diverse datasets, horizons, and sampling frequencies, with controlled evaluations in both *univariate* and *known-covariates* settings. An additional univariate setting with *missing values in the look-back window* is also considered across the entire `fev-bench` benchmark.

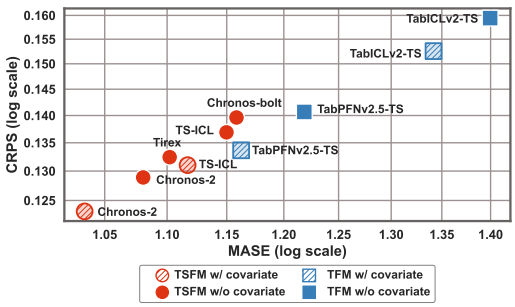
Setting. Evaluation considers two forecasting regimes. (i) In the *univariate setting*, the benchmark comprises 100 tasks and ∼235k forecasting windows (see Table 19, Appendix F.1). Models rely solely on past observations, with no access to covariates or cross-series information. (ii) In the *known-covariate setting*, we evaluate all methods on the same 100 tasks. Among these, 30 datasets include meaningful exogenous time series, denoted as "known dynamics" covariates in Table 19. For these datasets, methods that support covariates are evaluated both with and without covariate inputs, while methods that do not are kept unchanged. This protocol enables a controlled assessment of the effect of covariate information while preserving comparability across tasks.

Baselines. `TS-ICL` is compared against a broad set of baselines covering foundation models and local methods. (i) *Univariate comparisons* include state-of-the-art TSFMs (`Chronos-2` [3], `TiRex` [4], `Chronos-bolt` and `Toto` [10]), TFMs adapted to time series (`TabPFNv2.5-TS`, `TabICLv2-TS`) and local baselines (`Seasonal Naive`, `LOCF`). Supervised forecasting models are omitted, as prior large-scale studies indicate that TSFMs generally outperform them on established benchmarks [1]. `TimesFM 2.5` [11] and `Moirai2` [28] are excluded from evaluations due to substantial data leakage with `fev-bench`. (ii) In the *known-covariate* setting, we evaluate both TSFMs and TFMs un-

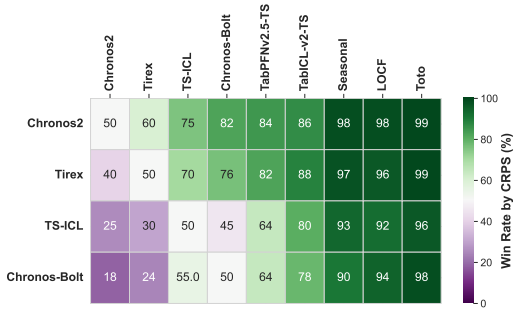
der the unified protocol described above. Methods supporting covariates (`TS-ICL`, `Chronos-2`, `TabPFNv2.5-TS`, `TabICLv2-TS`) are evaluated with and without covariates on the 30 relevant datasets, while covariate-agnostic TSFMs are included unchanged to quantify the benefit of incorporating exogenous information. Figure 5 reports the results following the `fev-bench` protocol, using Mean Absolute Scaled Error (MASE) and CRPS (metric definitions provided in Appendix D) while Table 3 reports the median inference time.



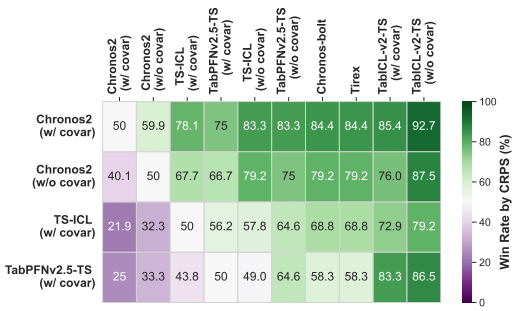
(a) *Univariate* forecasting (100 tasks).



(b) Forecasting on 100 tasks (30 *covariate-aware*).



(c) `Fev-bench`. *Univariate* forecasting (100 tasks)



(d) Forecasting on 100 tasks (30 *covariate-aware*).

Figure 5: `Fev-bench` forecasting benchmark. (a)-(b) MASE-CRPS trade-off (lower is better). Points correspond to per-method scores averaged across tasks. (c)-(d) Pairwise win rates. Each entry indicates the percentage of tasks where a method outperforms another according to the CRPS.

Results. `TS-ICL` achieves strong zero-shot performance on `fev-bench`, remaining competitive with leading TSFMs while consistently outperforming TFMs and local baselines on both point and probabilistic metrics.

(i) In the *univariate setting*, `TS-ICL` ranks among the top-performing methods (Figure 5a), remaining within $\sim$6% of `Chronos-2` and $\sim$3% of `TiRex`, while outperforming all other baselines, including TFMs and local methods. Pairwise comparisons (Figure 5c) confirm consistent majority wins across tasks against tabular and local approaches. Finally, it offers a strong accuracy–efficiency trade-off (Table 3), with inference time on the order of 10^{-2} seconds per window, around $4\times$ slower than `Chronos2` but still about $40\times$ faster than TFMs.

(ii) In the *known-covariate setting*, performance improves with exogenous information (Figure 5b). `Chronos2` remains the strongest overall method, while `TS-ICL` benefits from covariates, as illustrated qualitatively in Figure 6, and consistently outperforms TFMs under identical inputs. It also improves its relative ranking among TSFMs, surpassing `TiRex` in CRPS and achieving $\sim$70% pairwise win rates (Figure 5d), indicating stable gains on `fev-bench` when leveraging covariates.

Table 3: Median forecasting inference time on `fev-bench` (*univariate setting*) with a H100 GPU.

	Time Series Foundation Models				Tabular Foundation Models		Local Methods	
	`TS-ICL`	`Chronos-2`	`TiRex`	`TOTO`	`TabPFNv2.5-TS`	`TabICLv2-TS`	`S-Naive`	`LOCF`
Median Inference time (s / window)	1.54×10^{-2}	3.53×10^{-3}	5.20×10^{-2}	1.28×10^{-1}	4.33×10^{-1}	3.76×10^{-1}	3.09×10^{-4}	1.73×10^{-4}

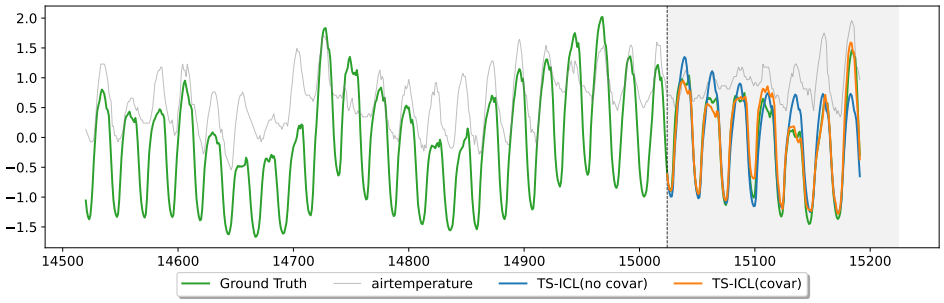


Figure 6: TS-ICL forecast on an horizon of length 168, with one additional covariate, on *GFC17*.

Forecasting with missing values. Beyond controlled evaluation settings on fev-bench, evaluating TS-ICL robustness in realistic scenarios with partially missing historical observations is crucial. Table 4 compares TS-ICL and Chronos-2 forecasting performance under increasing levels of look-back window missingness (30%–90%) across the entire fev-bench benchmark. While both models degrade as the context becomes sparser, TS-ICL consistently outperforms Chronos-2, with significant gaps at all missingness levels. As a result, TS-ICL remains substantially more robust for forecasting under missing historical observations.

Table 4: Zero-shot forecasting MASE under increasing missingness in the look-back window for TS-ICL and Chronos-2 on fev-bench (100 univariate tasks, look-back = 4092, arithmetic mean). Relative degradation (%) is reported in parentheses. Seasonal Naive serves as a simple baseline for evaluating forecasting performance under degraded conditions. Best results are in **bold**.

	0 % missing	30 % missing	50 % missing	70 % missing	90 % missing
Chronos-2	**1.62 (0%)**	2.16 (-33%)	2.44 (-50%)	2.54 (-56%)	3.97 (-144%)
TS-ICL	1.70 (0%)	**1.77 (-4%)**	**1.89** (-11%)	**2.16** (-27%)	3.63 (-113%)
Seasonal(0% missing)	2.48	2.48	2.48	2.48	**2.48**

Overall, TS-ICL is thus a competitive non patch-based TSFM for zero-shot forecasting. It remains close to state-of-the art TSFMs such as Chronos-2 and TiRex across standard benchmarks, while effectively leveraging covariates when available. Its most notable advantage lies in its robustness to missing history, where it consistently outperforms Chronos-2, highlighting the sensitivity of patch-based models to incomplete contexts. In contrast, TS-ICL benefits from its time-indexed formulation to tackle realistic forecasting settings. Additional analyses and forecasting plots are provided in Appendix F.1, while Appendix F.2 reports leakage-free comparisons against 12 TSFMs on the TIME benchmark [34] across 98 zero-shot tasks.

6 Conclusion

TS-ICL is a flexible probabilistic time series foundation model based on an in-context regression formulation, unifying forecasting and imputation within a continuous-time framework. It sets new state-of-the-art performance on zero-shot imputation, while remaining competitive with leading TSFMs on forecasting benchmarks and serving as a strong alternative to patch-based models. A key advantage of TS-ICL lies in its robustness to incomplete historical observations: it consistently outperforms Chronos-2 under partially observed look-back windows, highlighting the benefits of its time-indexed formulation in realistic forecasting settings. It also enables efficient covariate-aware inference, further improving performance when exogenous information is available. Despite these strengths, TS-ICL exhibits higher inference cost than highly optimized models such as Chronos-2 (up to $4\times$ slower despite having $4\times$ fewer parameters), mainly due to its pointwise regression formulation, which increases computational cost during both training and inference. These limitations may be mitigated through architectural optimizations such as mixed-precision training. In addition, further increasing the diversity of the training data prior may improve the zero-shot generalization of TS-ICL, as observed in tabular foundation models [18, 35]. Finally, the flexibility of the TS-ICL encoder–regressor framework makes it a natural foundation for extending beyond forecasting and imputation to tasks such as zero-shot anomaly detection and time series classification.

References

[1] 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. In *NeurIPS Workshop on Time Series in the Age of Large Models*, 2024. URL https://openreview.net/forum?id=Z2cMOOANFX.

[2] Abdul Fatir Ansari, Lorenzo Stella, Ali Caner Türkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda-Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Bernie Wang. Chronos: Learning the language of time series. *Trans. Mach. Learn. Res.*, 2024.

[3] Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, et al. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*, 2025.

[4] 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. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=v7UqniC9pF.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[6] Wei Cao, Dong Wang, Jian Li, Hao Zhou, Yitan Li, and Lei Li. BRITS: bidirectional recurrent imputation for time series. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[7] Mouxiang Chen, Lefei Shen, Zhuo Li, Xiaoyun Joy Wang, Jianling Sun, and Chenghao Liu. VisionTS: Visual masked autoencoders are free-lunch zero-shot time series forecasters. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=5DSj3MfWrB.

[8] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[9] James S Clark and Ottar N Bjørnstad. Population time series: process variability, observation errors, missing values, lags, and hidden states. *Ecology*, 85(11):3140–3150, 2004.

[10] Ben Cohen, Emaad Khwaja, Youssef Doubli, Salahidine Lemaachi, Chris Lettieri, Charles Masson, Hugo Miccinilli, Elise Ramé, Qiqi Ren, Afshin Rostamizadeh, Jean Ogier du Terrail, Anna-Monica Toon, Kan Wang, Stephan Xie, Zongzhe Xu, Viktoriya Zhukova, David Asker, Ameet Talwalkar, and Othmane Abou-Amal. This time is different: An observability perspective on time series foundation models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=1jDAYXfcS2.

[11] 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, ICML 2024, Vienna, Austria, July 21-27, 2024*, Proceedings of Machine Learning Research, 2024.

[12] Samuel Dooley, Gurnoor Singh Khurana, Chirag Mohapatra, Siddartha V Naidu, and Colin White. ForecastPFN: Synthetically-trained zero-shot forecasting. In *Advances in Neural Information Processing Systems*, volume 36, pp. 2403–2426, 2023.

[13] Wenjie Du, David Côté, and Yan Liu. SAITS: Self-attention-based imputation for time series. *Expert Systems with Applications*, 219:119619, 2023. doi: https://doi.org/10.1016/j.eswa.2023.119619.

[14] Wenjie Du, Yiyuan Yang, Linglong Qian, Jun Wang, and Qingsong Wen. PyPOTS: A Python Toolkit for Machine Learning on Partially-Observed Time Series. *arXiv:2305.18811*, 2023.

[15] Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 30583–30598. Curran Associates, Inc., 2022.

[16] Tilmann Gneiting, Fadoua Balabdaoui, and Adrian E Raftery. Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 69(2):243–268, 2007.

[17] 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*, 2022.

[18] 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.

[19] Shi Bin Hoo, Samuel Müller, David Salinas, and Frank Hutter. From tables to time: How TabPFN-v2 outperforms specialized time series forecasting models. *arXiv preprint arXiv:2501.02945*, 2025.

[20] Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[21] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pp. 4651–4664. PMLR, 2021.

[22] Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. URL https://kellerjordan.github.io/posts/muon/.

[23] Diederik P Kingma and Jimmy Lei Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2015.

[24] Louis Kirsch, James Harrison, Jascha Sohl-Dickstein, and Luke Metz. General-purpose in-context learning by meta-learning transformers. In *Sixth Workshop on Meta-Learning at the Conference on Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=t6tA-KB4dO.

[25] Roger Koenker and Kevin F Hallock. Quantile Regression. *Journal of Economic Perspectives*, 15(4):143–156, 2001. doi: 10.1257/jep.15.4.143.

[26] Etienne Le Naour, Louis Serrano, Léon Migus, Yuan Yin, Ghislain Agoua, Nicolas Baskiotis, Patrick Gallinari, and Vincent Guigue. Time series continuous modeling for imputation and forecasting with implicit neural representations. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=P1vzXDklar.

[27] Etienne Le Naour, Tahar Nabil, Adrien Petralia, and Ghislain Agoua. Are time-indexed foundation models the future of time series imputation? *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=cTk56KpsP5.

[28] Chenghao Liu, Taha Aksu, Juncheng Liu, Xu Liu, Hanshu Yan, Quang Pham, Silvio Savarese, Doyen Sahoo, Caiming Xiong, and Junnan Li. Moirai 2.0: When less is more for time series forecasting. *arXiv preprint arXiv:2511.11698*, 2025.

[29] 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. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=LO7ciRpjI5.

[30] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[31] Vladyslav Moroshan, Julien Siems, Arber Zela, Timur Carstensen, and Frank Hutter. TempoPFN: Towards synthetic pre-training of linear RNNs for zero-shot time series forecasting. In *EurIPS 2025 Workshop: AI for Tabular Data*, 2025. URL https://openreview.net/forum?id=Iqex1gfnvc.

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

[33] Jonas Peters, Dominik Janzing, and Bernhard Scholkopf. *Elements of causal inference: foundations and learning algorithms*. MIT press, 2017.

[34] Zhongzheng Qiao, Sheng Pan, Anni Wang, Viktoriya Zhukova, Yong Liu, Xudong Jiang, Qingsong Wen, Mingsheng Long, Ming Jin, and Chenghao Liu. It's TIME: Towards the next generation of time series forecasting benchmarks. *arXiv preprint arXiv:2602.12147*, 2026.

[35] Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. *arXiv preprint arXiv:2602.11139*, 2026.

[36] Yulia Rubanova, Ricky T. Q. Chen, and David Duvenaud. Latent odes for irregularly-sampled time series. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2019. Curran Associates Inc.

[37] Michael Schulz and Karl Stattegger. Spectrum: Spectral analysis of unevenly spaced paleoclimatic time series. *Computers & Geosciences*, 23(9):929–945, 1997.

[38] Louis Serrano, Thomas X Wang, Etienne Le Naour, Jean-Noël Vittaut, and Patrick Gallinari. Aroma: Preserving spatial structure for latent pde modeling with local neural fields. *Advances in Neural Information Processing Systems*, 37:13489–13521, 2024.

[39] 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. *arXiv preprint arXiv:2509.26468*, 2025.

[40] Ingo Steinwart and Andreas Christmann. Estimating conditional quantiles with the help of the pinball loss. *Bernoulli*, 17(1), 2011. doi: 10.3150/10-BEJ267.

[41] Sean J Taylor and Benjamin Letham. Forecasting at scale. *The American Statistician*, 72(1): 37–45, 2018.

[42] Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[43] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Learning deep time-index models for time series forecasting. In *International Conference on Machine Learning*, pp. 37217–37237. PMLR, 2023.

[44] Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Forty-first International Conference on Machine Learning*, 2024.

[45] Xin Wu, Fei Teng, Xingwang Li, Ji Zhang, Qiang Duan, and Tianrui Li. Out-of-distribution generalization in time series: A survey. *Information Fusion*, pp. 104336, 2026.

[46] Shifeng Xie, Vasilii Feofanov, Jianfeng Zhang, Themis Palpanas, and Ievgen Redko. Cauker: Classification time series foundation models can be pretrained on synthetic data. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=xBW2FIfswU.

Appendices

A TS-ICL Architecture Details **14**

A.1 TS-ICL overview . . . 14

A.2 The Time Series Encoder $\mathcal{E}$ . . . 15

A.3 The Channel Mixer $\mathcal{M}$ . . . 16

A.4 The Temporal Context Query Module $\mathcal{C}$ . . . 17

A.5 The In-Context Learning Regressor Module $\mathcal{R}$ . . . 18

B Training Details **20**

B.1 Training Prior . . . 20

B.2 Architecture Hyperparameters and Implementation Details . . . 26

C Ablation Studies **29**

C.1 Architecture Scaling . . . 29

C.2 Component Ablations: Synergy between Encoder and Regressor . . . 29

C.3 Covariate Management Strategies . . . 30

D Evaluation Metrics **31**

D.1 Metrics Definition . . . 31

E Extended Imputation Experiments **33**

E.1 `Fm-impute-bench` Benchmark . . . 33

E.2 `TIME` Benchmark . . . 39

F Extended Forecasting Experiments **41**

F.1 `Fev-bench` Benchmark . . . 41

F.2 `TIME` Benchmark . . . 48

F.3 Extended qualitative analysis on time series with strong trends . . . 51

G Extended Discussions **52**

G.1 Limitations . . . 52

G.2 Broader Impacts . . . 53

A TS-ICL Architecture Details

This section provides a detailed breakdown of the TS-ICL framework introduced in Section 3. Casting both imputation and forecasting tasks as in-context regression problems over learned temporal representation, TS-ICL consists in four successive modules that transform raw observations into global and local context-aware representations used for prediction. Section A.1 develops the intuition behind TS-ICL whereas the subsequent sections focus on the four modules.

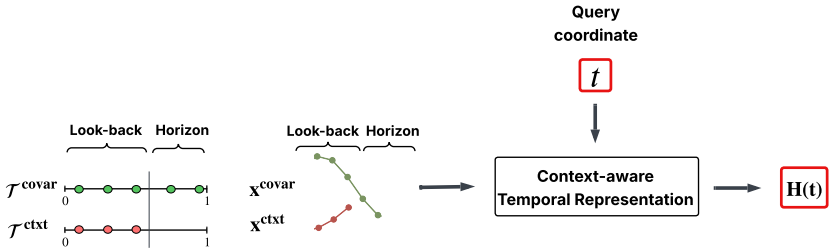
A.1 TS-ICL overview

The core intuition behind TS-ICL is to learn an in-context time series regressor. In general, In-Context Learning (ICL) may be seen as a metalearning algorithm, providing adaptation at inference time through supervised input-output pairs rather than parameter udpates [42, 15, 24, 17]. More explicitly, given a labeled training set $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^N$ with features x and labels y, ICL aims to approximate $p(y_{query}|x_{query}, \mathcal{D})$.

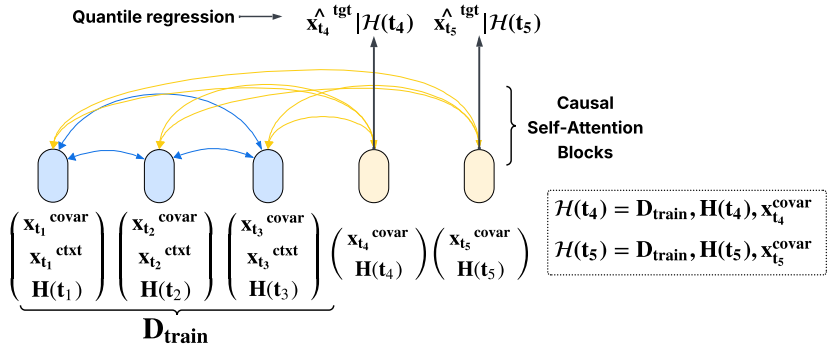
In a time series setting, TS-ICL considers a single time series as one supervised dataset, with timesteps as features and observed values as labels. Assuming (univariate) observations $x(t)$ at timesteps $\{t_i\}_{i=1}^N$, the goal is thus to approximate $p\left(x(t_{query})|t_{query}, \{(t_1, x(t_1)), \ldots, (t_N, x(t_N))\}\right)$. To this end, TS-ICL follows an encoder-regressor architecture, shown in Figure 7. (i) The encoder enriches the representation of each local timestep t with a shared *global* time series representation Z^{final}, providing per-timestep features $H(t, Z^{final})$. (ii) The regressor solves the in-context *pointwise* regression problem:

$$p\left(x(t_{query})|H(t_{query}), \{(H(t_1), x(t_1)), \ldots, (H(t_N), x(t_N))\}\right), \tag{1}$$

where we write $H(t)$ for $H(t, Z^{final})$.



(a) Encoder.



(b) ICL regressor.

Figure 7: Overview of the (a) encoder - (b) ICL regressor architecture of TS-ICL.

A few observations stem from Eq. (1). First, the query timestep must attend to every timestep in the observed context examples: TS-ICL follows the *pointwise* ICL formulation inherited from tabular foundation models [19]. Second, (1) does not depend on a specific ordering of the timesteps, and applies thus equally to both the imputation and the forecasting tasks, with partially or irregularly observed contexts. Third, Z^{final} must encode *any* context: with or without covariates, with irregular

observations, etc. The encoder leverages cross-attention and Perceiver-like mechanisms to progressively learn such flexible temporal representations in three stages, detailed in Sections A.2 to A.4. Finally, not shown in (1), the regressor also incorporates covariate information to $H(t)$, as described in greater length in Section A.5.

A.2 The Time Series Encoder $\mathcal{E}$

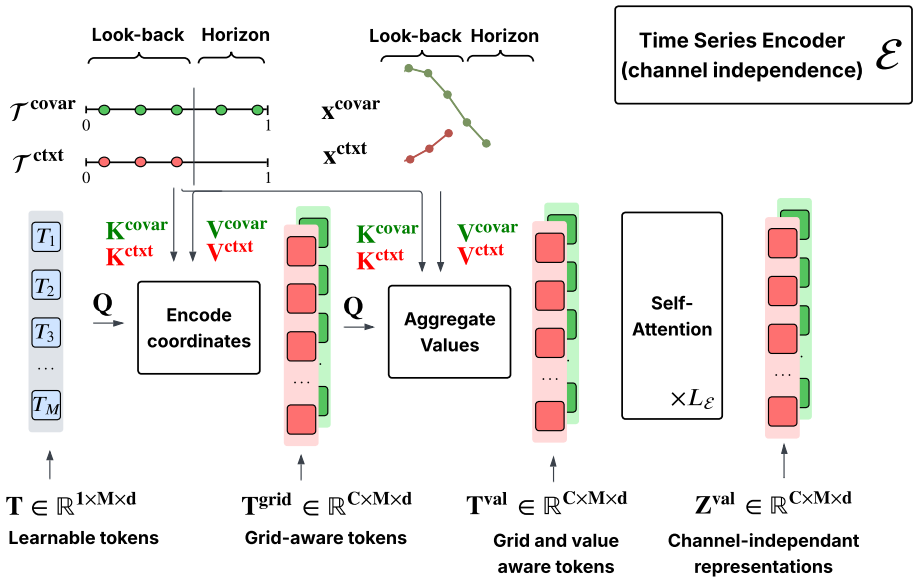


Figure 8: Overview of the Time Series Encoder $\mathcal{E}$. Forecasting task shown for illustration.

Encoder Overview. The encoder $\mathcal{E}$ maps the observed context $(\mathcal{T}^{\text{ctxt}}, \boldsymbol{x}^{\text{ctxt}})$ and $C-1$ optional covariates $(\mathcal{T}^{\text{covar}}, \boldsymbol{X}^{\text{covar}})$, where $C \geq 1$, jointly into a channel-independent latent representation $\boldsymbol{Z}^{\text{val}} \in \mathbb{R}^{C\times M\times d}$. Shown in Figure 8, the module operates through the following steps:

(i) **Temporal Encoding.** The timestamps from both the context $\mathcal{T}^{\text{ctxt}}$ and covariates $\mathcal{T}^{\text{covar}}$ are independently mapped into higher-dimensional representations using Fourier features [30] followed by a linear projection:

$$\mathcal{T} \xrightarrow{\text{Fourier + Linear}} \gamma(\mathcal{T}) \in \mathbb{R}^{T\times d}.$$

(ii) **Coordinate Encoding.** A set of M learnable latent tokens $\boldsymbol{T} \in \mathbb{R}^{1\times M\times d}$ serves as a query (Q) and attends to the temporal embeddings of all channels (context and covariates) through cross-attention:

$$\boldsymbol{T}^{\text{grid}} = \text{CrossAttn}(Q = \boldsymbol{T},\ K = V = \gamma(\mathcal{T})) \in \mathbb{R}^{C\times M\times d}.$$

This step produces *grid-aware tokens* that capture the geometric structure of the sampling grid for each channel.

(iii) **Value Aggregation.** The observed values $(\boldsymbol{x}^{\text{ctxt}}, \boldsymbol{X}^{\text{covar}})$ are first projected into the latent dimension (*value lifting*). The grid-aware tokens $\boldsymbol{T}^{\text{grid}}$ then attend to these value embeddings (see Figure 9 for a temporal interpretation):

$$\boldsymbol{T}^{\text{val}} = \text{CrossAttn}\Big(Q = \boldsymbol{T}^{\text{grid}},\ K = \gamma(\mathcal{T}),\ V = \text{lift}(\boldsymbol{X})\Big) \in \mathbb{R}^{C\times M\times d}.$$

This step integrates the specific observed values into the latent representations, resulting in *grid and value aware tokens*.

(iv) **Latent Refinement.** Finally, $L_{\mathcal{E}}$ layers of channel-independent self-attention are applied to refine the representations:

$$\boldsymbol{Z}^{\text{val}} = \text{Transformer}_L(\boldsymbol{T}^{\text{val}}) \in \mathbb{R}^{C \times M \times d}.$$

This produces the final latent representations $\boldsymbol{Z}^{\text{val}}$, where each channel has been compressed into M informative tokens.

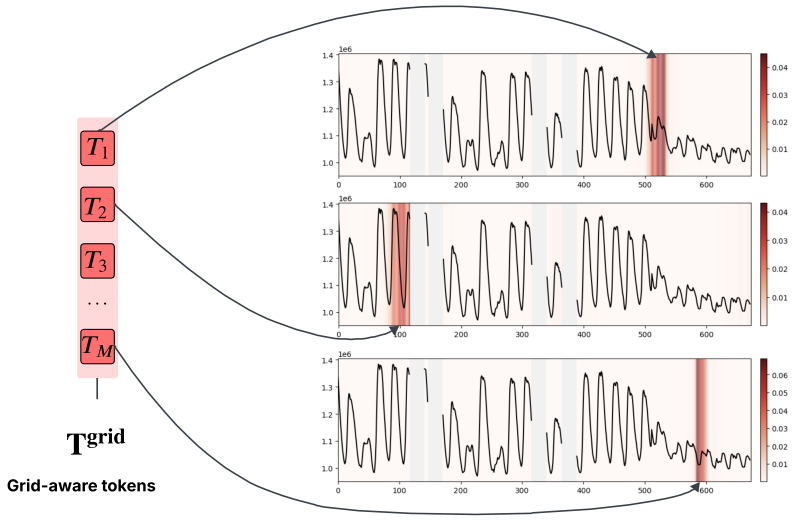


Figure 9: Temporal interpretation of tokens via cross-attention between $\mathbf{T}^{grid}$ and $\gamma(t)$ for each $t \in \mathcal{T}^{obs}$. Cross-attention maps for three distinct tokens are shown for a given attention head.

A.3 The Channel Mixer $\mathcal{M}$

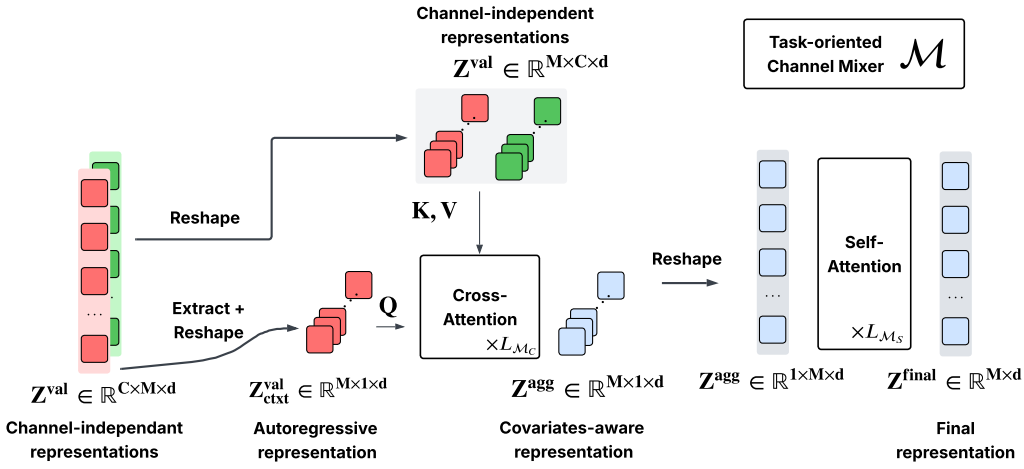


Figure 10: Overview of the Channel Mixer $\mathcal{M}$.

Module $\mathcal{M}$ Overview. The Channel Mixer $\mathcal{M}$ is designed to aggregate information across multiple channels (time series of interest and covariates representations) by conditioning the channel-independent features on the target series context. It transforms a set of independent representations $\boldsymbol{Z}^{\text{val}}$ into a unified, covariate-aware representation $\boldsymbol{Z}^{\text{final}}$.

Shown in Figure 10, the module follows a three-step process:

(i) **Cross-Channel Attention.** The autoregressive context representation $\boldsymbol{Z}^{\text{val}}_{\text{ctxt}} \in \mathbb{R}^{M\times 1\times d}$, which represents the specific temporal dynamics of the target time series, acts as a query (Q). It attends to the channel-independent representations $\boldsymbol{Z}^{\text{val}} \in \mathbb{R}^{M\times C\times d}$ (where C is the number of channels), which serve as keys (K) and values (V):

$$\boldsymbol{Z}^{\text{agg}} = \text{CrossAttn}_L\Big(Q = \boldsymbol{Z}^{\text{val}}_{\text{ctxt}},\ K = V = \boldsymbol{Z}^{\text{val}}\Big) \in \mathbb{R}^{M\times 1\times d}.$$

This operation, repeated over $L_{\mathcal{M}_c}$ layers, compresses the multi-channel information into a single "covariates-aware" latent representation, effectively selecting the most relevant features from the covariates for the given context.

(ii) **Latent Reshaping.** To prepare the aggregated representation for global sequence processing, the tensor is reshaped to treat the M latent tokens as a sequence:

$$\boldsymbol{Z}^{\text{agg}} \in \mathbb{R}^{M\times 1\times d} \quad \xrightarrow{\text{reshape}} \quad \boldsymbol{Z}^{\text{agg}} \in \mathbb{R}^{1\times M\times d}.$$

(iii) **Global Latent Refinement.** Finally, $L_{\mathcal{M}_s}$ self-attention blocks are applied to the sequence of tokens. This allows the model to capture global dependencies across the aggregated latent space:

$$\boldsymbol{Z}^{\text{final}} = \text{Transformer}_L(\boldsymbol{Z}^{\text{agg}}) \in \mathbb{R}^{1\times M\times d}.$$

The resulting $\boldsymbol{Z}^{\text{final}}$ is the final time series representation, integrating both the local channel information and the global context necessary for the downstream task.

A.4 The Temporal Context Query Module $\mathcal{C}$

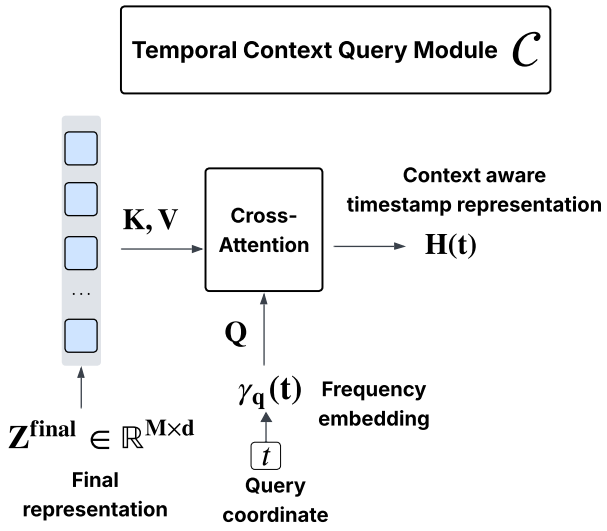


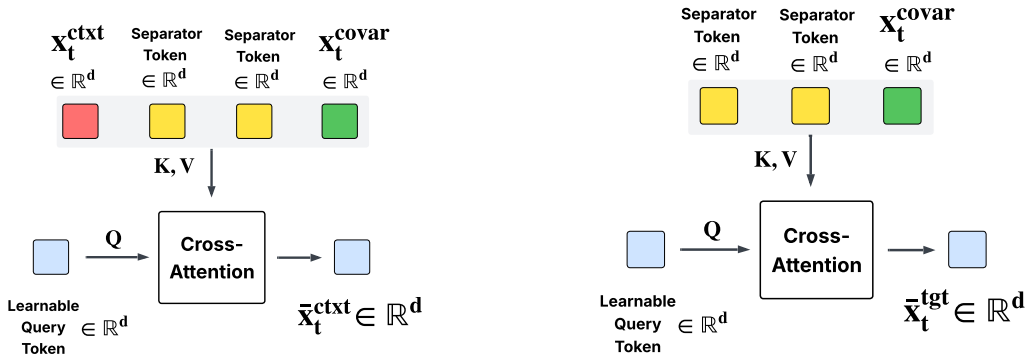
Figure 11: Overview of the Temporal Context Query Module $\mathcal{C}$.

Module $\mathcal{C}$ Overview. The Temporal Context Query Module $\mathcal{C}$ maps the final latent representation $\boldsymbol{Z}^{\text{final}} \in \mathbb{R}^{M\times d}$ and a query coordinate t to a time-series-aware representation $H(t) \in \mathbb{R}^d$. This module enables continuous-time querying of the encoded context by bridging the gap between the discrete latent tokens and the continuous time domain.

Shown in Figure 11, the module operates as follows:

(i) **Frequency Encoding.** A target timestamp $t \in \mathbb{R}$ is mapped into a higher-dimensional frequency embedding $\gamma_q(t)$ (Fourier features, [30]). This encoding uses sinusoidal functions at multiple scales to capture both coarse and fine-grained temporal patterns:

$$t \xrightarrow{\text{Frequency Encoding}} \gamma_q(t) \in \mathbb{R}^d.$$



(a) Build context tokens $\bar{\mathbf{x}}_t^{\mathbf{ctxt}}$ for $t \in \mathcal{T}^{cxt}$.

(b) Build target tokens $\bar{\mathbf{x}}_t^{\mathbf{tgt}}$ for $t \in \mathcal{T}^{tgt}$.

Figure 12: Cross-Attention Mechanism for Context and Target Token Construction.

(ii) **Contextual Querying via Cross-Attention.** The frequency embedding $\gamma_q(t)$ serves as the query (Q) in a cross-attention mechanism. It attends to the final time series representation $\boldsymbol{Z}^{\text{final}}$, which provides the keys (K) and values (V):

$$H(t) = \text{CrossAttn}\Big(Q = \gamma_q(t),\ K = V = \boldsymbol{Z}^{\text{final}}\Big).$$

This operation extracts a localized, time-series-aware representation from the global latent context, specifically conditioned on the query coordinate t.

(iii) **Representation Output.** The resulting vector $H(t)$ constitutes the "time-series-aware timestamp representation". It integrates the global context stored in the latent tokens with the specific temporal information of the query, serving as the primary input for the downstream in-context regressor.

A.5 The In-Context Learning Regressor Module $\mathcal{R}$

Module $\mathcal{R}$ Overview. The In-Context Learning Regressor $\mathcal{R}$ is the final component of the architecture. It treats both forecasting and imputation as in-context regression tasks, where the model learns to map target representations to values by conditioning on observed "input-output" pairs [5, 15, 42]. $\mathcal{R}$ leverages a specific token construction mechanism to align context-aware embeddings with raw covariates.

Input Projection and Token Construction via Cross-Attention. As depicted in Figure 12, prior to token construction, all raw inputs (including the observed values $\mathbf{x}_t^{\text{ctxt}}$ and covariates $\mathbf{x}_t^{\text{covar}}$) are linearly projected into a common d-dimensional latent space $\mathbb{R}^d$. A Cross-Attention mechanism then fuses these projected representations with a learnable query token $Q \in \mathbb{R}^d$.

Notably, the covariate input $\mathbf{x}_t^{\text{covar}}$ is strictly optional and can consist of zero, one, or multiple distinct covariates. The Cross-Attention mechanism accommodates this variability: since attention operates over sets, varying the number of covariates simply changes the sequence length of the Keys and Values, requiring no architectural modifications. This allows the regressor $\mathcal{R}$, and thus `TS-ICL`, to operate on covariate grids $\mathcal{T}_c^{\text{covar}}$ unaligned with the context grid $\mathcal{T}^{\text{ctxt}}$.

- **Context Tokens ($\bar{\mathbf{x}}_t^{\mathbf{ctxt}}$):** For $t \in \mathcal{T}^{\text{ctxt}}$, the model groups the projected observed value $\mathbf{x}_t^{\text{ctxt}}$ with a set of learned separator tokens and the projected covariates $\mathbf{x}_t^{\text{covar}}$ (if any). These act as Keys (K) and Values (V) for the learnable Query token Q, resulting in a unified context representation $\bar{\mathbf{x}}_t^{\text{ctxt}} \in \mathbb{R}^d$.
- **Target Tokens ($\bar{\mathbf{x}}_t^{\mathbf{tgt}}$):** For $t \in \mathcal{T}^{\text{tgt}}$, the ground truth value is unknown. The target token $\bar{\mathbf{x}}_t^{\text{tgt}} \in \mathbb{R}^d$ is similarly constructed by applying Cross-Attention between the learnable Query Q and the available information: the covariate embeddings $\mathbf{x}_t^{\text{covar}}$ and the separator tokens.

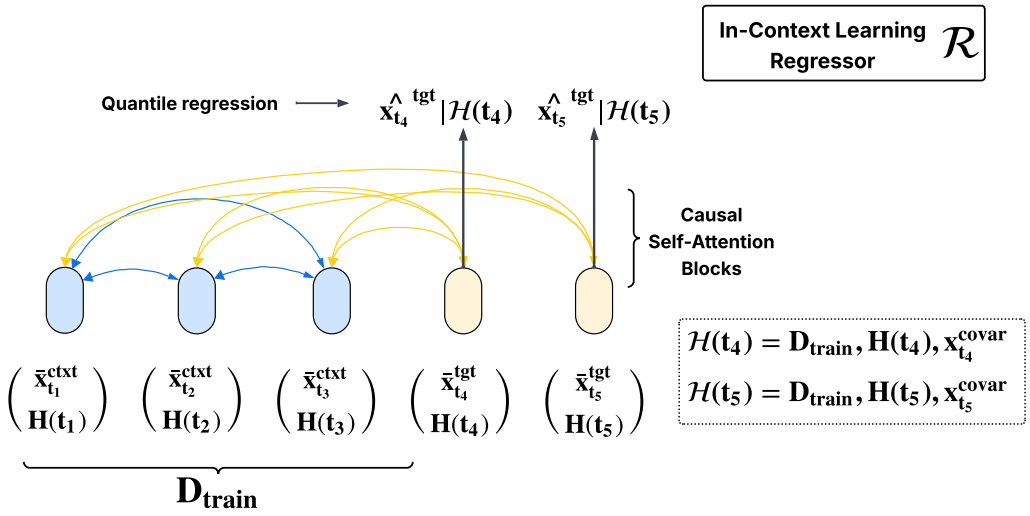


Figure 13: Overview of the In-Context Learning Regressor Module $\mathcal{R}$. Forecasting task shown for illustration.

In-Context Input Sequence. The regressor processes a sequence $\mathbf{S}$ organized to facilitate relational learning (Figure 13). The sequence consists of paired context tokens followed by target queries:

$$\mathbf{S} = \big[\underbrace{(\bar{\mathbf{x}}_{t_1}^{\text{ctxt}}, \mathbf{H}(t_1)), \ldots, (\bar{\mathbf{x}}_{t_n}^{\text{ctxt}}, \mathbf{H}(t_n))}_{\mathcal{D}_{\text{train}}}, (\bar{\mathbf{x}}_{t_{n+1}}^{\text{tgt}}, \mathbf{H}(t_{n+1})), \ldots \big]$$

where $\mathbf{H}(t)$ represents the context-aware temporal embedding. All $(\bar{\mathbf{x}}_t^{\text{ctxt}}, \mathbf{H}(t)$ pairs are summed to form the final regressor input sequence in the $d-$dimensional latent space.

Causal In-Context Regression. The sequence $\mathbf{S}$ is processed by L layers of causal self-attention blocks. The causal mask is critical as it ensures a specific information flow:

- Each target token $(\bar{\mathbf{x}}_{t_j}^{\text{tgt}}, \mathbf{H}(t_j))$ attends to all previous pairs in $\mathcal{D}_{\text{train}}$ to infer the underlying mapping $\mathbf{H}(t) \to \mathbf{x}(t)$.
- The attention mechanism allows the model to dynamically weigh past observations based on their similarity to the current target query in the representation space, without attending to future target values.

Quantile Prediction and Loss. To capture predictive uncertainty, for each target timestamp $t_j \in \mathcal{T}^{\text{tgt}}$, the model outputs 99 quantiles $\hat{\boldsymbol{q}}(t_j) = (\hat{q}_{\alpha_k}(t_j))_{k=1}^{99}$ via a linear projection of the final hidden states. The model is trained by minimizing a Smooth Pinball Loss:

$$\mathcal{L} = \sum_{t_j \in \mathcal{T}^{\text{tgt}}} \sum_{k=1}^{99} \Big[\alpha_k e_{j,k} + \beta \log \Big(1 + \exp(-e_{j,k}/\beta) \Big) \Big]$$

where $e_{j,k} = x(t_j) - \hat{q}_{\alpha_k}(t_j)$, $\alpha_k \in (0,1)$ is the quantile level, and $\beta > 0$ is a smoothing parameter. In practice, we set $\beta = 0.01$. During training, gradients are only backpropagated through the target predictions; the context set $\mathcal{D}_{\text{train}}$ is treated strictly as conditioning data and does not contribute to the loss.

B Training Details

This appendix provides detailed information about the training procedure of TS-ICL. Section B.1 covers the data aspect, from pretraining corpus to prior generation, whereas Section B.2 describes the set of hyperparameters used to instantiate and train TS-ICL.

B.1 Training Prior

B.1.1 Univariate Time Series Datasets.

The univariate pretraining datasets of TS-ICL originate from three main sources, namely: (i) LOTSA [44]; (ii) Chronos training data [2] and (iii) TempoPFN synthetic data [31]. The latter includes in particular the synthetic generators from ForecastPFN [12], Chronos [2] and CauKer [46]. Overall, the pretraining corpus comprises 40 datasets, listed in Table 5 with their key features.

Table 5: All 40 univariate time series datasets used to pretrain TS-ICL and their key properties. The weight column reports the down- or upsampling coefficient applied dynamically at each epoch. †: offline downsampling from the original dataset.

Dataset	Release Platform	Domain	Freq	Num. Series	Num. Variates	Max Length	Weight
Australian Electricity	Chronos	Energy	30T	5	1	232,272	220
BDG-2 Bull	LOTSA	Energy	H	41	1	12,280	25
BDG-2 Fox	LOTSA	Energy	H	135	1	12,280	5
BDG-2 Panther	LOTSA	Energy	H	105	1	6,132	2.5
BuildingsBench900k	LOTSA	Energy	H	100,000†	1	8,759	0.02048
Residential Load Power	LOTSA	Energy	1T	271	3	614,880	1.2
Residential PV Power	LOTSA	Energy	1T	233	3	614,880	1.5
Wind Farms H	Chronos	Energy	H	337	1	6,148	4
Wind Farms D	Chronos	Energy	D	337	1	366	2
China Air Quality	LOTSA	Climate	H	437	6	397,335	0.3
CMIP6 2000	LOTSA	Climate	6H	8,192	22†	7,300	0.057
ERA5 1989	LOTSA	Climate	H	8,192	15†	8,736	0.085
ERA5 1990	LOTSA	Climate	H	8,192	15†	8,736	0.085
ERA5 1991	LOTSA	Climate	H	8,192	15†	8,736	0.085
Spanish Weather	Kaggle	Climate	H	5	3	24,544	105
Subseasonal	LOTSA	Climate	1D	862	4	16,470	0.3
Subseasonal Precipitation	LOTSA	Climate	1D	862	1	11,323	1.2
Weatherbench daily	Chronos	Climate	1D	10,000†	1	14,610	0.1024
Mexico City Bikes	Chronos	Traffic	H	494	1	104,449	2.5
PEMS04	LOTSA	Traffic	5T	307	3	16,992	1.2
PEMS07	LOTSA	Traffic	5T	883	1	28,224	1.2
PEMS08	LOTSA	Traffic	5T	170	3	17,856	2.1
Q-TRAFFIC	LOTSA	Traffic	15T	45,148	1	5,856	0.024
Taxi (30 Min.)	Chronos	Traffic	30T	2,428	1	1,488	0.88
Taxi (Hourly)	Chronos	Traffic	H	2,428	1	744	0.88
Uber TLC (Hourly)	Chronos	Traffic	H	262	1	4,344	4
Alibaba Cluster Trace 2018	LOTSA	Cloud	5T	58,409	2	1,728	0.009
Wiki Daily	Chronos	Web	D	100,000	1	2,741	0.00512
Monash M3 Monthly	LOTSA	Econ./Fin.	M	1,428	1	126	0.72
NN5 Weekly	LOTSA	Econ./Fin.	W	111	1	105	5
Project Tycho	LOTSA	Health	W	1,258	1	3,854	0.21
Anomaly	TempoPFN	Synthetic	-	5,000	1	10,000	0.0256
ForecastPFN	TempoPFN	Synthetic	-	5,000	1	10,000	1
GP	TempoPFN	Synthetic	-	5,000	1	10,000	0.4096
Kernel Synth 1M	Chronos	Synthetic	-	1,000,000	1	1,024	0.001024
Sawtooth	TempoPFN	Synthetic	-	5,000	1	10,000	0.0512
Sinewave	TempoPFN	Synthetic	-	5,000	1	10,000	0.1024
Spikes	TempoPFN	Synthetic	-	5,000	1	10,000	0.0256
Step	TempoPFN	Synthetic	-	5,000	1	10,000	0.0512
OU	TempoPFN	Synthetic	-	5,000	1	10,000	0.4096

Table 5 highlights that our selection (i) spans *multiple domains*, including energy, nature/climate, transport, cloud, health/economics; (ii) covers a broad range of *frequencies*, from minutely- to weekly- and monthly-sampled time series; (iii) includes series of *varying context length*, from 126 timesteps to 614k. In total, this forms a corpus of about 2M series, strictly non-overlapping with the different benchmarks on which `TS-ICL` is evaluated (`fm-imputation`, `fev-bench` and `TIME`).

Sampling strategy. We adopt a simple three-step stratified sampling strategy to encourage training on maximally diverse patterns. (i) Very large hourly datasets are downsampled offline, thereby reducing memory footprint: we use 100k random samples from *BuildingsBench900k*, 10k random samples from *Weatherbench daily* and 15 (resp., 22) representative channels out of 45 (resp. 53) for the *ERA5* (resp., *CMIP6*) datasets. (ii) Similarly to `Moirai` [44], a random subsample or upsample is drawn from each dataset at each training epoch, to avoid biases towards hourly datasets in the energy and climate domains. The sampling coefficients are reported in Table 5. (iii) For each sample, we select a single window drawn at random from the available series length.

B.1.2 Covariates Problem Generators.

In this section, we provide the technical specifications for the synthetic target–covariate(s) generation process described in Section 4.

Graph Construction Logic The generator constructs a fixed Directed Acyclic Graph (DAG) for each time series batch. The construction follows a topological ordering to ensure acyclicity:

(i) **Node Initialization.** We start with R root nodes, where R is randomly drawn from a geometric distribution (favoring smaller values), and capped at a maximum of $R_{\max} = 3$. Each root corresponds to one of the base univariate time series in the pretraining corpus in Table 5 (e.g., signals from real-world datasets or `TempoPFN`).

(ii) **Graph Size.** The total number of generated series C (channels) corresponds to the number of time series that will define the final covariates–target problem. Specifically, the constructed problem will consist of $C - 1$ covariate time series and 1 target time series. The value of C is sampled from a shifted exponential distribution to favor smaller, more manageable graphs while allowing for high complexity:

$$C = \min\left(C_{\max}, \lfloor \text{Exp}(\lambda) \rfloor + C_{\min}\right),$$

where we set $C_{\min} = 2$, $C_{\max} = 20$, and $\lambda = 0.8$ by default. The total number of nodes in the graph is $V = 2C + R$.

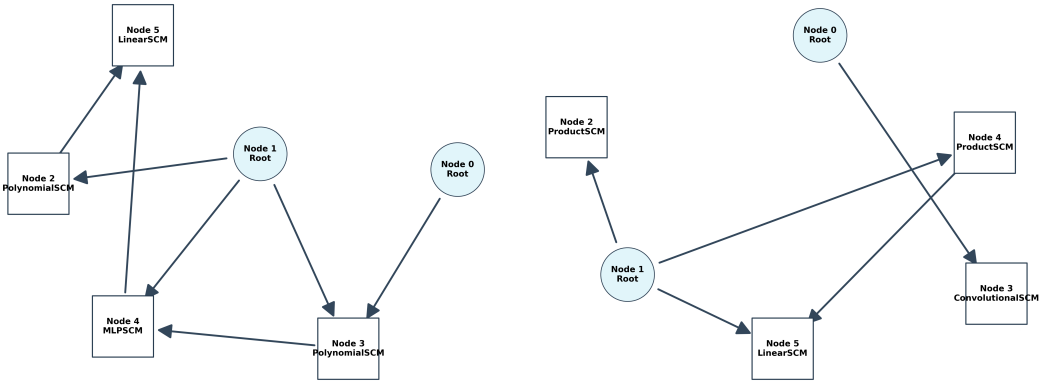
(iii) **Edge Sampling.** For each non-root node $i \in \{R, \ldots, V-1\}$, the number of parents k_i is sampled from a geometric distribution $k_i \sim \text{Geometric}(p)$, clipped to the number of available ancestors. Parents are then sampled uniformly without replacement from the set of all preceding nodes $\{0, \ldots, i-1\}$.

(iv) **Operator Assignment.** Each non-root node is assigned a Structural Causal Model (SCM) sampled from the Structural Causal Model registry.

This DAG structure allows for the generation of both dependent and independent child nodes, which is essential for constructing realistic target–covariate time series relationships. This design also ensures that covariates are not always informative, thereby encouraging models to learn to ignore irrelevant inputs when appropriate. As an illustration, Figure 14 displays two randomly generated graphs with 6 nodes each, including 2 channels (for one target and one covariate).

Structural Causal Model (SCM) Registry. To ensure a wide variety of functional relationships, we implement a set of diverse SCMs inspired by [35]. Let $\mathbf{X}pa(i)$ denote the collection of parent time series for node i. The child node Y_i is generated as $Y_i = \text{Normalize}(f(\mathbf{X}pa(i)))$, where $f \in$ *Registry*. The registry includes:

- **LinearSCM.** A simple linear combination $Y = \mathbf{W}\mathbf{X} + b$.
- **MLPSCM.** A multi-layer perceptron with random depth (2–10 layers) and hidden dimensions (8–128). Activations are randomly selected for each layer (ReLU, Tanh, ELU, etc.). We use a sparsity-inducing "block-wise dropout" initialization to create specific feature-group dependencies.



(a) Fully dependent structure (all nodes are connected).

(b) Partially independent structure (not all nodes are connected).

Figure 14: Examples of randomly generated DAG structures with six nodes. The left graph enforces strong dependencies between nodes, while the right graph allows for partial independence, leading to more diverse causal structures.

- **ConvolutionalSCM.** Models local temporal dependencies using 1D convolutions with random kernel sizes (3–8) and random channel depths.
- **RNNSCM.** Captures deep temporal dependencies using a GRU architecture. These are strictly causal, ensuring the value at time t only depends on $t' \leq t$.
- **PolynomialSCM.** Each input is raised to a random power $d \in \{1, 2, 3, 4\}$ before being linearly combined, inducing symmetric non-linearities.
- **DiscretizeSCM.** Simulates quantization effects by mapping a linear mixture of inputs into a fixed number of discrete bins (2 to 15).
- **ProductSCM.** Computes the element-wise product of all parent signals, representing multiplicative interactions.

For visual illustrations of the types of dependencies induced by each SCM, we refer the reader to Figure 15.

Normalization and Stability. To prevent numerical instability and exploding values across deep DAGs, every node's output is z-normalized. This ensures that every generated time series Y_i maintains a mean of 0 and a standard deviation of 1 before it is used as an input for further child nodes.

Problem Formulation (Target and Covariates). Once the DAG is computed, we finalize the target–covariate problem by:

(i) Sampling a subset of C nodes from the graph to be "visible" to the model.

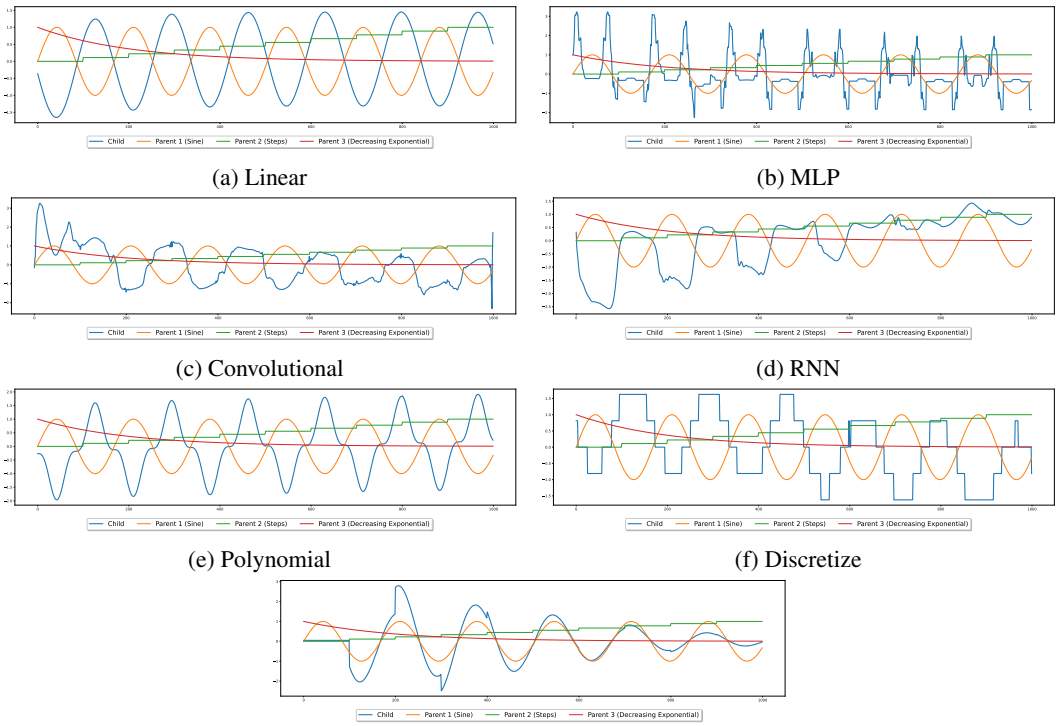
(ii) Randomly designating one of these nodes as the Target (y).

(iii) Designating the remaining $C - 1$ nodes as Covariates ($\mathbf{x}$).

To encourage learning of informative covariate-target relationships, we first seek to fulfill steps (ii) and (iii) by looking for connected components of the DAG. We thus sample the C channels from the subset of nodes that have all, or at least one, root node as common ancestor.

- This approach ensures that covariates are not merely "noise" but share a common underlying causal structure with the target, sometimes acting as direct causes, sometimes as effects, and sometimes as siblings sharing a latent root cause.
- If this subset is empty or contains less than C nodes, the remaining nodes are drawn uniformly at random within the entire graph, allowing for independent or unrelated (covariate, target) pairs.

Examples of such multivariate problems drawn from the data prior are shown in Figure 16.



(a) Linear (b) MLP

(c) Convolutional (d) RNN

(e) Polynomial (f) Discretize

(g) Product

Figure 15: Examples of time series generated by different Structural Causal Models (SCMs) from the registry. Each plot shows the transformation of three root signals (sinusoidal, linear trend, and exponential decay) into a child time series through the corresponding operator.



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Figure 16: Examples of different covariate-target time series sampled from the data prior.

B.1.3 The whole procedure

We summarize the full data generation pipeline in Algorithm 1. The procedure samples heterogeneous training tasks combining real-world time series, synthetic time series and target–covariates time series problems. This enables the model to learn both standard univariate forecasting/imputation tasks and more complex multivariate settings generated via structural causal models.

A Bernoulli switch controlled by probability π determines whether a task is sampled from the univariate or covariate (SCM-based) mode.

Algorithm 1 Data Prior Sampling Procedure for `TS-ICL`

Require: Collection of base univariate time series $\mathcal{D}$ (real + synthetic)
Require: SCM registry Υ
Require: Probability π of sampling a univariate task
Sample $u \sim \mathcal{U}(0, 1)$
if $u < \pi$ **then**
    Univariate mode:
    Sample time series $x \sim \mathcal{D}$
    Apply task masking (imputation or forecasting)
    return x
else
    Covariate mode:
    Sample number of task channels:
        $C \sim \text{Exp}(\lambda)$, clipped to $[C_{\min}, C_{\max}]$
    Sample number of root signals:
        $R \sim \text{Geometric}(q)$, clipped to $R_{\max}$
    Define latent DAG size:
        $V = 2C + R$
    Sample root time series:
        $S_1, \ldots, S_R \sim \mathcal{D}$
    Initialize DAG nodes:
        $\{X_i\}_{i=1}^{R} \leftarrow \{S_i\}_{i=1}^{R}$
    DAG construction
    for $i = R + 1, \ldots, V$ **do**
        Sample number of parents $k_i \sim \text{Geometric}(p)$
        Sample parent set $\mathcal{P}_i \subset \{1, \ldots, i-1\}$
        Sample SCM $f_i \sim \Upsilon$
        Compute:
            $X_i \leftarrow \text{Normalize}\big(f_i(\mathbf{X}_{\mathcal{P}_i})\big)$
    end for
    Observation step
    Sample observed set $\mathcal{O} \subset \{1, \ldots, V\}$ such that $|\mathcal{O}| = C$
    Sample target node $y \sim \mathcal{O}$
    Define covariates $\mathbf{x} = \mathcal{O} \setminus \{y\}$
    Task masking
        - imputation: random point/block masking
        - forecasting: future horizon masking
    return $(\mathbf{x}, y)$
end if

B.2 Architecture Hyperparameters and Implementation Details

The TS-ICL architecture is governed by a set of hyperparameters controlling model capacity, tokenization granularity, and depth across modules. All latent vectors across the four modules share a common dimensionality d, ensuring seamless information flow.

B.2.1 Architectural Hyperparameters.

Time Series Encoder $\mathcal{E}$ hyperparameters. The encoder serves as the primary feature extractor, compressing multi-channel time series into a fixed-size latent bottleneck - see Figure 8.

- **Latent dimension (d):** The feature dimensionality used throughout the encoder is set to $d = 256$.
- **Temporal Encoding:** Timestamps are mapped using Fourier features in logarithmic scale with base 2, with $L_{\text{fourier}} = 128$ frequencies up to maximum frequency 2^{10}, followed by a linear projection layer to $\mathbb{R}^d$.
- **Number of latent tokens (M):** The number of learnable tokens per channel is $M = 32$. This parameter controls the resolution of the latent representation.
- **Refinement block:** The refinement stage consists of $L_{\mathcal{E}} = 3$ channel-independent Transformer blocks, each with $n_h^{\mathcal{E}} = 8$ self-attention heads of dimension 64.

Channel Mixer Module $\mathcal{M}$ hyperparameters. The mixer facilitates task-oriented channel mixing by conditioning the target series on covariates - see Figure 10.

- **Latent dimension (d):** The feature dimensionality used throughout the channel mixer is set to $d = 256$.
- **Cross-Channel Attention:** The number of cross-attention layers used to aggregate channel-independent tokens into a covariate-aware representation is $L_{\mathcal{M}}^{\text{cross}} = 3$. Each layer contains $n_h^{\mathcal{M}} = 8$ attention heads of dimension 64.
- **Global Latent Refinement:** After reshaping, the latent sequence is processed by $L_{\mathcal{M}}^{\text{self}} = 3$ self-attention layers to capture dependencies across the aggregated latent space. Each layer contains $n_h^{\mathcal{M}} = 8$ self-attention heads of dimension 64.

Temporal Context Query Module $\mathcal{C}$ hyperparameters. This module acts as a continuous-time interface between the latent tokens and the regressor - see Figure 11.

- **Latent dimension (d):** The feature dimensionality used throughout the context query module is set to $d = 256$.
- **Frequency Encoding:** Query coordinates t are encoded using high-frequency sinusoidal features (NeRF-style) to preserve fine-grained temporal localizations before being projected to $\mathbb{R}^d$. We use three frequency bands each with 128 frequencies and respective maximum frequencies 2^6, 2^7, 2^{10}.
- **Query mechanism:** A single 8×64 cross-attention layer is used to extract $H(t)$ from $\boldsymbol{Z}^{\text{final}}$. This layer operates in parallel on the three frequency bands. After concatenating, the context-aware time representation $H(t)$ has dimension $3 \times d = 768$.

In-Context Learning Regressor $\mathcal{R}$ hyperparameters. The regressor performs the final predictive task using a causal Transformer architecture - see Figures 12 & 13.

- **Latent dimension (d):** The feature dimensionality used throughout the in-context learning regressor is set to $d = 512$.
- **Input Value Projection:** Before entering the cross-attention block for token construction, all available observed values x_t are projected to $\mathbb{R}^d$ via a linear layer. Similarly, if available, covariates X_t are projected to $\mathbb{R}^d$ via another linear layer, shared by all covariates.
- **ICL Transformer:** The causal sequence processing is performed by $L_{\mathcal{R}} = 12$ Transformer layers, each with $n_h^{\mathcal{R}} = 8$ heads of dimension 64.

- **Quantile Head:** A final linear layer maps the d-dimensional hidden states of target tokens to a 99-dimensional vector representing the equidistant predicted quantiles.

B.2.2 Training hyperparameters and task-specific strategies.

This section provides additional details about the training strategy.

Input scaling. Following common practice in time series forecasting, raw time series are pre-processed by an instance-normalization layer, scaling each sample to zero mean and unit variance. We then apply a pointwise $\sinh^{-1}$ transform to the standardized inputs, to stabilize training against outlier values [3].

Task mixing and zero-shot robustness. We employ a task mixing probability $\pi = 0.8$: 20% of the training batches are drawn from the univariate pretraining corpus in Table 5, whereas the remaining 80% are further processed by the SCM prior in Algorithm 1 to create target–covariates tasks (with $C - 1$ covariates, $C \geq 2$) or new univariate tasks ($C = 1$). This mixture ensures the model remains a robust univariate predictor while learning to leverage exogenous signals.

- **Covariate Complexity Sampling:** For covariate tasks, the number of channels C is (i) either $C = 1$ (univariate task) with probability 0.2 (ii) or sampled from a truncated exponential distribution:

$$P(K = k) \propto e^{-\lambda k}, \quad k \in \{2, \ldots, 20\}.$$

  We set $\lambda = 0.5$ to favor simpler tasks with few covariates while maintaining significant exposure to high-dimensional inputs (up to 20 covariates).

Specialized Checkpoints. While the architecture $\mathcal{R}$ is identical for all tasks, we provide two specialized checkpoints. The *Forecasting* checkpoint is trained strictly with right causal masking, whereas the *Imputation* checkpoint is trained to reconstruct missing values using both preceding and succeeding context.

Imputation Training. We enforce task diversity while training the imputation checkpoint with a two-step procedure.

1. **Window sampling:** a per-batch context length T sampled from multiple regimes to handle varying context lengths:

$$T \sim \begin{cases} \mathcal{U}(128,\, 336) & \text{with probability } p_1 = 0.15 \\ \mathcal{U}(512,\, 1024) & \text{with probability } p_2 = 0.6 \\ \mathcal{U}(1300,\, 1400) & \text{with probability } p_3 = 0.05 \\ \mathcal{U}(2000,\, 2100) & \text{with probability } p_4 = 0.1 \\ \mathcal{U}(4000,\, 4096) & \text{with probability } p_5 = 0.1 \end{cases},$$

   where $p_1, \ldots, p_5$ are dataset balancing probabilities. The maximum supported window length for imputation is $T = 4096$.
2. **Masking Procedure:** once a context window has been sampled, we then apply a random masking strategy. A fraction ρ of the observations are held out as target queries $\mathcal{T}^{\text{tgt}}$. The remaining points form $\mathcal{D}_{\text{train}}$.
   - ρ is either a pointwise missingness rate, sampled randomly from $\{0.05,\, 0.06, \ldots, 0.95\}$;
   - or corresponds to up to four missing blocks of random length between 12 and 168 (adjusted depending on available context length) and at most 50% pointwise missing values.

Forecasting training. A similar procedure is applied to the forecasting checkoint;

1. **Window sampling:** the lookback and horizon pairs are sampled jointly:

$$(L,\, H) \sim \begin{cases} (\mathcal{U}(50,\, 200),\, \mathcal{U}(8,\, 20)) & \text{with probability } p_1 = 0.2 \\ (\mathcal{U}(200,\, 672),\, \mathcal{U}(18,\, 36)) & \text{with probability } p_2 = 0.15 \\ (\mathcal{U}(1000,\, 1100),\, \mathcal{U}(48,\, 336)) & \text{with probability } p_3 = 0.5 \\ (\mathcal{U}(2000,\, 2100),\, \mathcal{U}(48,\, 336)) & \text{with probability } p_4 = 0.1 \\ (\mathcal{U}(3062,\, 4096),\, \mathcal{U}(48,\, 672)) & \text{with probability } p_5 = 0.05 \end{cases}.$$

The maximum supported configuration is:

$$T_{\text{look-back}} \leq 4096, \quad T_{\text{horizon}} \leq 672.$$

2. **Masking Procedure:** We enforce robustness to irregularly sampled time series by removing a fraction ρ of the observations in the lookback window, with probability 0.15. We draw ρ from the set $\{0.1,\, 0.25,\, 0.5,\, 0.75\}$.

Quantile head. The model predicts a set of 99 quantiles $\{\alpha_k\}_{k=1}^{99}$, uniformly spaced in $(0, 1)$, allowing for full density estimation. We set the smoothing coefficient of the pinball loss to $\beta = 0.01$ to ensure differentiability.

B.2.3 Optimization Hyperparameters.

For both imputation and forecasting checkpoints, the optimization is configured as follows:

- **Optimizer:** *Muon* [22] for 2D parameters and AdamW for 1D parameters (embeddings, scales, biases).
- **Learning rate:** Max $lr = 4e-4$.
- **Scheduler:** Cosine decay down to $5e-4$.
- **Hardware:** $4\times$ Nvidia H100 GPUs (92GB).
- **Batch size:** the global batch size is $B = 256$, obtained through a mini-batch size of 32 and two steps of gradient accumulation.
- **Training Budget:** The imputation checkpoint is trained for $500k$ optimization steps, representing approximately 5 training days. The forecasting checkpoint is trained for $650k$ optimization steps, representing approximately 9 training days.

C Ablation Studies

In this section, we provide a comprehensive analysis of the architectural choices and scaling properties of `TS-ICL`. All ablation models were trained for a fixed budget of 100k steps on a H100 GPU to ensure fair comparison. Evaluation is performed on a subset of 11 datasets (44 tasks) of `fm-impute-bench`, namely `fm-impute-mini`, commonly used for ablations [27] (see Table 6).

Table 6: `fm-impute-mini` subset for zero-shot imputation ablation studies.

Dataset	Domain	Freq	Num. Series	Series Length	Num. Test Windows	Window Size
BDG2-Bear	Energy	1H	91	17544	7522	672
BDG2-Rat	Energy	1H	280	17544	24915	672
Covid19 Energy	Energy	1H	1	31912	195	672
GFC12 Load	Energy	1H	20	39414	4960	672
Hog	Energy	1H	24	17544	2310	672
Jena Weather 10T	Climate	10min	21	52704	1428	4032
Jena Weather 1H	Climate	1H	21	8784	1344	672
Oikolab Weather	Climate	1H	8	100057	5288	672
PDB	Energy	1H	1	17520	96	672
Pedestrian Counts	Transport	1H	66	96400	7733	672
Weather	Climate	1H	11	35064	2398	672

C.1 Architecture Scaling

We evaluate the impact of model capacity by varying (i) the number of attention heads $n_h = n_h^{\mathcal{E}} = n_h^{\mathcal{M}} = n_h^{\mathcal{C}}$ in the three Transformers of the encoder; (ii) the number $\mathcal{L}_{\mathcal{R}}$ of self-attention layers in the in-context regressor $\mathcal{R}$; (iii) the number $n_h^{\mathcal{R}}$ of self-attention heads in $\mathcal{R}$. We define three model configurations: *Small*, *Medium*, and *Large*, with respectively **8.5M**, **12M** and **27M** parameters.

Table 7: Model Capacity Ablation. Average CRPS over 44 univariate imputation tasks from the `fm-impute-mini` benchmark (lower is better).

Model	n_h	$\mathcal{L}_{\mathcal{R}}$	$n_h^{\mathcal{R}}$	Params	`fm-impute-mini`
`Small`	4	8	4	∼8.5M	0.201
`Medium`	8	12	4	∼12M	0.197
`Large`	8	12	8	∼27M	**0.194**

Results. Table 7 shows that increasing model capacity consistently improves imputation accuracy, with the *Large* configuration obtaining the best average CRPS. In particular, moving from *Small* to *Large* reduces the average CRPS from 0.201 to 0.194, suggesting that additional attention capacity in both the encoder and the in-context regressor improves the model's ability to exploit contextual information across tasks. The gains, however, are relatively smooth and exhibit diminishing returns: the *Medium* model already closes a substantial fraction of the gap to the *Large* model, while using less than half the number of parameters. Moreover, the *Small* model remains competitive despite its lower capacity. Overall, this suggests a favorable accuracy–efficiency trade-off, with smaller variants remaining attractive under computational constraints.

C.2 Component Ablations: Synergy between Encoder and Regressor

To justify the hybrid structure of `TS-ICL`, we compare the full architecture against two baseline configurations:

- **Encoder-Only.** We remove the In-Context Regressor $\mathcal{R}$. The context-aware representation $H(t)$ from module $\mathcal{C}$ is passed directly through a dense MLP network - with 5×256 hidden layers - to project onto the target values.
- **Regressor-Only (Pure ICL).** We remove the Encoder $\mathcal{E}$ and the Mixer $\mathcal{M}$. The Regressor $\mathcal{R}$ receives only raw temporal Fourier features [30] as representation $H(t)$, performing pure in-context regression without the benefit of the refined latent context.

Table 8: **Model Capacity Ablation.** Average CRPS over 44 univariate imputation tasks from the `fm-impute-mini` benchmark (lower is better).

Configuration	Architecture Change	`fm-impute-mini`
Encoder-Only	$H(t) \to$ MLP Head	0.297
Regressor-Only	No $\mathcal{E}$, Raw Fourier Features	0.204
Full TS-ICL	**Encoder $\mathcal{E}$ + Regressor $\mathcal{R}$**	**0.194**

Results. Table 8 highlights the complementarity between the encoder and the in-context regressor. The *Encoder-Only* variant performs substantially worse, indicating that the latent context representation alone is not sufficient without an expressive regression mechanism. Conversely, the *Regressor-Only* baseline is much stronger, but still underperforms the full model, showing that raw Fourier features provide a competitive ICL baseline but lack the refined context produced by the encoder. The full `TS-ICL` architecture achieves the best CRPS, suggesting that the encoder and regressor act synergistically: the encoder builds informative context-aware representations, while $\mathcal{R}$ effectively uses them for in-context prediction.

C.3 Covariate Management Strategies

We investigate the optimal placement of the covariate mixing mechanism. Since `TS-ICL` allows for multi-stage conditioning, we compare three strategies for handling exogenous signals:

- **Early Mixing (Encoder Only):** Covariates are processed in the Encoder $\mathcal{E}$ and mixed in the Mixer $\mathcal{M}$. The Regressor $\mathcal{R}$ only receives $H(t)$ and context-target pairs of the main series, with no cross-attention on covariates during token construction.
- **Late Mixing (Regressor Only):** The Encoder $\mathcal{E}$ is univariate (no covariates). All covariate information is provided directly to the Regressor $\mathcal{R}$ via the Cross-Attention mechanism during input token construction.
- **Dual Mixing (Full Architecture):** Covariates are leveraged both in the global representation (Encoder/Mixer) and for local token conditioning (Regressor).

Table 9: Covariate Mixing Strategy. Average CRPS over 6 covariate-aware imputation tasks from `fm-impute-covars` (lower is better).

Mixing Strategy	Mixing Stage	TMLR (Covar)
Early Mixing	$\mathcal{E} + \mathcal{M}$ only	0.123
Late Mixing	$\mathcal{R}$ only	**0.085**
Dual Mixing	$\mathcal{E} + \mathcal{M}$ and $\mathcal{R}$	**0.085**

Results. The results in Table 9 show that late covariate mixing is already sufficient to obtain strong performance, with *Late Mixing* matching the *Dual Mixing* variant. This suggests that injecting covariate information directly in the regressor $\mathcal{R}$ provides effective pointwise conditioning at prediction time. In contrast, *Early Mixing* alone performs worse, indicating that global covariate information in the encoder is not sufficient without late-stage conditioning. We nevertheless retain the dual strategy, as early mixing may provide useful global context about covariate structure in harder settings, while late mixing supplies local, pointwise covariate information to the regressor.

D Evaluation Metrics

This section provides formal definitions for the evaluation metrics employed across the various experimental benchmarks presented in Section 5 and Appendices E, F. These metrics are designed to provide a comprehensive assessment of model performance, covering (i) computational efficiency during inference, (ii) the calibration and quality of the predicted probability distributions, and (iii / iv) the accuracy of point predictions via scale-independent error measures.

D.1 Metrics Definition

(i) Inference efficiency score definition. The computational efficiency is measured by the median inference time per window (as in [39]), expressed in milliseconds (ms). The calculation follows a two-step aggregation: first, the mean inference time is computed for each individual dataset to account for domain-specific variations; second, the median of these mean values is taken. For a collection of D datasets, where each dataset d contains M_d windows with individual inference times $\Delta t_{m,d}$, the efficiency score is defined as:

$$\mu_d = \frac{1}{M_d} \sum_{m=1}^{M_d} \Delta t_{m,d},$$

$$\text{Efficiency} = \text{median}(\{\mu_1, \ldots, \mu_D\}).$$

This procedure ensures that the final metric is representative of typical performance while remaining robust to outliers across different data distributions.

(ii) Weighted Quantile Loss (WQL) and Continuous Ranked Probability Score (CRPS) definitions. To evaluate the quality of the predicted distribution, the Continuous Ranked Probability Score (CRPS) is employed, which measures the compatibility between the predicted cumulative distribution function $\hat{F}$ and the observed ground truth x. The CRPS can be expressed in its integral form as:

$$\text{CRPS}(\hat{F}, x) = \int_0^1 2 \cdot \text{QL}_\alpha(\hat{F}^{-1}(\alpha), x)\, \mathrm{d}\alpha,$$

where $\text{QL}_\alpha(q, x)$ represents the quantile loss (or pinball loss) at level α:

$$\text{QL}_\alpha(q, x) = (\alpha - \mathbb{I}_{\{x<q\}})(x - q).$$

To ensure computational tractability and provide a normalized metric for cross-dataset comparison, a normalized discrete approximation of the CRPS is utilized, known as the Weighted Quantile Loss (WQL) [25, 16]. For a set of K discrete quantiles $\{\alpha_1, \ldots, \alpha_K\}$, the WQL is calculated as:

$$\text{WQL} = \frac{1}{K} \sum_{j=1}^{K} \text{WQL}_{\alpha_j},$$

where each individual WQL_α is normalized by the absolute scale of the targets:

$$\text{WQL}_\alpha = \frac{2 \sum_{i,t \in \mathcal{T}^{\text{tgt}}} \text{QL}_\alpha(q_{i,t}^{(\alpha)}, x_{i,t})}{\sum_{i,t \in \mathcal{T}^{\text{tgt}}} |x_{i,t}|}.$$

In the evaluation, $K = 9$ equidistant quantiles $\alpha \in \{0.1, 0.2, \ldots, 0.9\}$ are used, following standard pratice, e.g. [39, 34]. This formulation allows the WQL to serve as a robust, scale-invariant proxy for the CRPS, capturing the accuracy of the entire predicted distribution.

Important note. For deterministic baselines such as `Linear`, `Seasonal`, or `LOCF`, the same point prediction is used across all quantile levels when computing the WQL.

(iii) Normalized Mean Absolute Error (NMAE) definition. To assess point prediction accuracy while accounting for differing scales across series, the Normalized Mean Absolute Error (NMAE) is used (as in [32]). This metric rescales the standard Mean Absolute Error (MAE) by the standard deviation of the observations in the context set.

For a series i and a target horizon $\mathcal{T}^{\text{tgt}}$, let $x_{i,t}$ be the ground truth and $\hat{x}_{i,t}$ the predicted median (quantile 0.5 for `TS-ICL`). The NMAE for series i is defined as:

$$\text{NMAE}_i = \frac{\frac{1}{|\mathcal{T}^{\text{tgt}}|}\sum_{t\in\mathcal{T}^{\text{tgt}}}|x_{i,t}-\hat{x}_{i,t}|}{\sigma_{i,\text{ctxt}}},$$

where $\sigma_{i,\text{ctxt}}$ is the standard deviation of the series i calculated over the observed context set $\mathcal{T}^{\text{ctxt}}$:

$$\sigma_{i,\text{ctxt}} = \sqrt{\frac{1}{|\mathcal{T}^{\text{ctxt}}|}\sum_{t\in\mathcal{T}^{\text{ctxt}}}(x_{i,t}-\bar{x}_{i,\text{ctxt}})^2}.$$

This normalization provides a scale-independent measure of the error relative to the inherent volatility of the series. The global NMAE is obtained by averaging across all N series:

$$\text{NMAE} = \frac{1}{N}\sum_{i=1}^{N}\text{NMAE}_i.$$

(iv) Mean Absolute Scaled Error (MASE) definition. To evaluate point prediction accuracy across datasets with varying scales, the Mean Absolute Scaled Error (MASE) is used [20]. MASE normalizes the Mean Absolute Error (MAE) of the model by the mean absolute error of a seasonal naïve baseline.

For a series i, let $x_{i,t}$ be the ground truth and $\hat{x}_{i,t}$ the predicted median. The MASE for series i is defined as:

$$\text{MASE}_i = \frac{1}{|\mathcal{T}^{\text{tgt}}|}\sum_{t\in\mathcal{T}^{\text{tgt}}}\frac{|x_{i,t}-\hat{x}_{i,t}|}{a_i},$$

where a_i is the seasonal normalization factor calculated over the target set $\mathcal{T}^{\text{tgt}}$:

$$a_i = \frac{1}{|\mathcal{T}^{\text{tgt}}|-s}\sum_{t\in\mathcal{T}^{\text{tgt}},t>s}|x_{i,t}-x_{i,t-s}|,$$

and s is the seasonal periodicity. This normalization ensures that the metric is scale-independent by comparing the model's error to the typical seasonal variations within the same target horizon.

The global metric is obtained by averaging across all N series:

$$\text{MASE} = \frac{1}{N}\sum_{i=1}^{N}\text{MASE}_i.$$

E Extended Imputation Experiments

This section provides broader insights into TS-ICL imputation performances. A detailed description of the fm-impute-bench benchmark used in the main text (Section 5.1) is given in Section E.1, together with complementary results and qualitative visualizations. Section E.2 further extends the evaluation to a second benchmark, TIME [34], which we adapt to the univariate imputation setting.

E.1 Fm-impute-bench Benchmark

E.1.1 Datasets and Baselines

Univariate inference datasets. Table 10 details the *univariate datasets* used for the zero-shot imputation experiments in Section 5.1. These datasets cover a diverse range of domains, including energy, transport, and climate science, with sampling frequencies varying from 5 minutes to 1 hour (specifically 5, 10, 15, 30, and 60 minutes). The imputation tasks are performed on four-week windows. When considering the four distinct missingness scenarios, this benchmark represents a large-scale evaluation involving approximately 1.3 million windows to be imputed.

Table 10: All datasets used for zero-shot imputation in the fm-impute-bench benchmark under the *univariate setting*.

Dataset	Release Platform	Domain	Freq	Num. Series	Series Length	Num. Test Windows	Window Size
BDG2-Bear	LOTSA	Energy	1H	91	17544	7522	672
BDG2-Rat	LOTSA	Energy	1H	280	17544	24915	672
Borealis	LOTSA	Energy	1H	15	7447	77	672
Covid19 Energy	LOTSA	Energy	1H	1	31912	195	672
GFC12 Load	LOTSA	Energy	1H	20	39414	4960	672
Hog	LOTSA	Energy	1H	24	17544	2310	672
Ideal	LOTSA	Energy	1H	217	16167	156	672
PDB	LOTSA	Energy	1H	1	17520	96	672
KDD Cup2022	LOTSA	Energy	10min	134	35279	2546	4032
ERA5 geopotential	LOTSA	Climate	1H	500	8736	19000	672
ERA5 humidity	LOTSA	Climate	1H	500	8736	19000	672
ERA5 temperature	LOTSA	Climate	1H	500	8736	19000	672
ERA5 wind speed	LOTSA	Climate	1H	500	8736	19000	672
Oikolab Weather	LOTSA	Climate	1H	8	100057	5288	672
Pedestrian Counts	LOTSA	Transport	1H	66	96400	7733	672
Traffic	LOTSA	Transport	1H	861	17544	83479	672
PEMS BAY	LOTSA	Transport	5min	325	52128	2275	8064
PEMS 03	LOTSA	Transport	5min	358	26208	358	8064
SHMETRO	LOTSA	Transport	15min	576	8809	576	2688
ETT1-15T	GIFT-eval	Energy	15min	7	69680	1050	2688
ETT1-1H	GIFT-eval	Energy	1H	7	17420	1092	672
ETT2-15T	GIFT-eval	Energy	15min	7	69680	1050	2688
ETT2-1H	GIFT-eval	Energy	1H	7	17420	1092	672
Solar-1H	GIFT-eval	Energy	1H	137	8760	8768	672
Jena Weather 10T	GIFT-eval	Climate	10min	21	52704	1428	4032
Jena Weather 1H	GIFT-eval	Climate	1H	21	8784	1344	672
Loop Seattle 5T	GIFT-eval	Transport	5min	323	105120	21964	8064
Loop Seattle 1H	GIFT-eval	Transport	1H	323	8760	20672	672
MDense	GIFT-eval	Transport	1H	30	17520	4710	672
Enedis LDM Small	Zenodo	Energy	30min	500	17424	20500	1344
London Smart Meters Small	Chronos	Energy	30min	500	22000	25779	1344
Spanish Energy	Kaggle	Energy	1H	9	35064	1962	672
Weather	Informer	Climate	1H	11	35064	2398	672

Inference datasets with covariates. Table 11 details the six datasets used to evaluate zero-shot imputation with *exogenous covariates*. Following the protocol in Table 10, four-week windows are generated for these experiments. As described in [27], the PV and Wind datasets map regional renewable energy production in 2021 to solar irradiance and wind speed, respectively. In contrast, Load-France tracks national electricity demand using average temperature as the primary covari-

ate to model consumption patterns. When considering the four distinct missingness scenarios, the covariate benchmark represents an evaluation involving approximately 1k windows to be imputed.

Table 11: All datasets used for zero-shot imputation in the `fm-impute-bench` benchmark under the *known-covariate setting.*

Dataset	Release Platform	Domain	Freq	Target / Covariate	Series Length	Num. Test Windows	Window Size
PV-OCC	RTE / Meteo	Energy	1H	1 / 1	8760	38	672
PV-PACA	RTE / Meteo	Energy	1H	1 / 1	8760	38	672
Wind-HDF	RTE / Meteo	Energy	1H	1 / 1	8760	38	672
Wind-GE	RTE / Meteo	Energy	1H	1 / 1	8760	38	672
Load-France 21	RTE / Enedis	Energy	30min	1 / 1	17520	41	1344
Load-France 22	RTE / Enedis	Energy	30min	1 / 1	17520	41	1344

Baselines details. A brief description of the baselines used in the benchmark is provided below.

- `TabPFNv2.5-TS` [19] is a time series foundation model that adapts the tabular foundation model `TabPFN` - a transformer-based model pretrained on synthetic supervised-learning tasks for in-context prediction of unseen tabular datasets [17] - to temporal data. `TabPFN-TS` leverages a `TabPFN` regression backbone and reformulates time-series prediction as an in-context tabular regression problem. Originally proposed for zero-shot forecasting, `TabPFN-TS` still naturally applies to imputation: observed target values form the in-context training set, while missing timestamps are treated as query points, enabling the model to impute gaps using all available non-missing observations. For consistency with `TabICLv2-TS`, we use the same feature-generation pipeline as in the `TabICLv2-TS` package: each timestamp is converted into a tabular row with temporal features, automatically extracted seasonal features, and, when available, exogenous covariates [35]. The pretrained `TabPFN` regressor is then queried in zero-shot to obtain point predictions. Our experiments use the `TabPFNv2.5` regression checkpoint as the backbone, together with the official implementation: `https://github.com/PriorLabs/tabpfn-time-series`.
- Similarly to `TabPFNv2.5-TS`, `TabICLv2-TS` is a foundation model for time series analysis adapted from the `TabICLv2` [35] tabular foundation model, designed for scalable in-context learning on regression and classification tasks.We use the package-provided time-series transformation pipeline to construct the tabular representation. `TabICLv2` then performs regression by conditioning on the resulting table in a single in-context inference procedure. In our experiments, we use the `TabICLv2` implementation and forecasting utilities from the official release: `https://github.com/soda-inria/tabicl`.
- `Linear` imputes missing values by linearly interpolating between the closest observed neighbors surrounding the gap. If a gap has no future (resp., past) anchor, it falls fack to `NOCB`, next-observation-carried-backward (resp., `LOCF`, last-observation-carried-forward).
- `Seasonal Naive` imputes a missing value at timestamp t by repeating the observation from the previous seasonal period, i.e., the value at $t - S$. S is pre-defined for each dataset based on its dominant frequency (e.g., daily). If the value at $t - S$ is also missing, the method sequentially searches for an available observation at other seasonal timestamps (e.g., $t + S$, then $t - 2S$, etc.). The method falls back to `LOCF` in case this search fails.
- `LOCF` imputes a missing value by copying the most recent available past value.
- `SAITS` [13] is a supervised Transformer-based imputation model designed for partially observed multivariate time series. It uses two diagonally-masked self-attention blocks to capture temporal and cross-variable dependencies, whose outputs are combined through a learned gating mechanism to reconstruct missing values.
- `BRITS` [6] is a supervised recurrent imputation model based on bidirectional RNNs with learned temporal decay. It processes each window forward and backward to account for irregular gaps, jointly estimating hidden states and missing values while encouraging consistency between directions.

Task specific baseline detailed training. The two supervised baselines, `SAITS` and `BRITS`, are trained on the training split of `fm-impute-bench` (see [27] for more details). Both implementations are taken from the `PyPOTS` toolbox [14] and trained on fixed-length windows with a masked-reconstruction objective: a subset of observed entries is randomly masked, and the model reconstructs these values from the remaining observations and the corresponding binary observation mask. Inputs are z-score normalized per variable, training minimizes MAE on the artificially masked positions, and model selection is performed using validation MSE with early stopping. We use the default hyperparameter configurations recommended by the original authors, the Adam optimizer [23], a batch size of 64, at most 50 training epochs, and early stopping with patience 5. Since `SAITS` and `BRITS` produce pointwise imputations, we adapt them to quantile-based evaluation by replicating each point prediction across all requested quantile levels.

E.1.2 Extended Results

This section extends the empirical evaluation in Section 5.1 with a more detailed analysis of imputation performance across all experimental settings. Specifically, we provide:

- **Aggregated detailed performance tables:** We report the average NMAE and CRPS (metrics definition in Section D) across the 132 *univariate* tasks and 24 *covariate-aware* tasks of `fm-impute-bench`. These results, detailed in Table 12 and Table 13, provide a detailed view of both point-wise and probabilistic performance. It is important to note that metrics are aggregated across tasks using the arithmetic mean, following the evaluation protocol established in `fm-impute-bench` [27].
- **NMAE pairwise win rates:** To complement the CRPS-based win rate diagrams presented in the main text Figure 4, we include the corresponding pairwise win rate visualizations in terms of NMAE for both *univariate* and *known-covariate* experiments in Figure 17. The NMAE-based pairwise comparisons provide additional evidence of the robustness of `TS-ICL`, showing consistent superiority regardless of the chosen accuracy metric.

Table 12: Aggregated imputation metrics on the 132 tasks of the *univariate setting* in `fm-impute-bench` (mean ± std). Best in **bold**.

	TSFM	Tabular Foundation Models		Task Specific Models		Local Models		
	`TS-ICL`	`TabPFNv2.5-TS`	`TabICLv2-TS`	`SAITS`	`BRITS`	`Linear interp.`	`Seasonal Naive`	`LOCF`
NMAE (↓)	**0.243 ± 0.118**	0.296 ± 0.145	0.294 ± 0.136	0.386 ± 0.140	0.470 ± 0.181	0.507 ± 0.287	0.580 ± 0.177	0.612 ± 0.255
CRPS (↓)	**0.255 ± 0.137**	0.303 ± 0.156	0.301 ± 0.148	0.503 ± 0.193	0.605 ± 0.227	0.658 ± 0.377	0.750 ± 0.230	0.793 ± 0.335

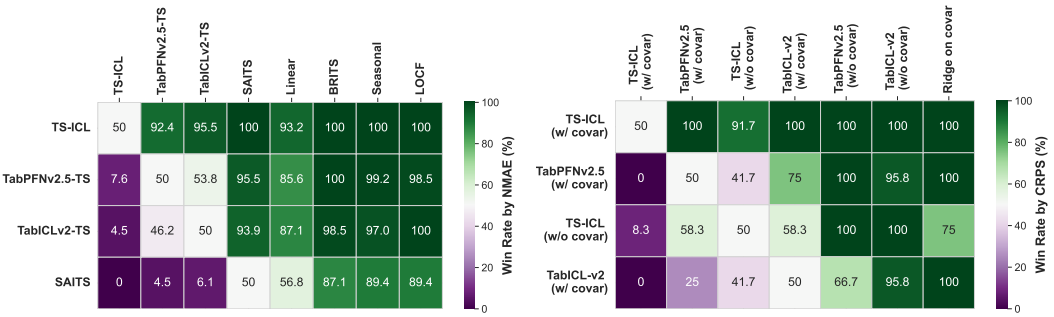
Table 13: Aggregated imputation performance metrics across the 24 tasks of the *known covariates* setting in `fm-impute-bench` (mean ± std). Best in **bold**.

	TSFM		Tabular Foundation Models				Local Model
	`TS-ICL`		`TabPFNv2.5-TS`		`TabICLv2-TS`		`Ridge on Covar`
	`w/ covar`	`w/o covar`	`w/ covar`	`w/o covar`	`w/ covar`	`w/o covar`	`w/ covar`
NMAE (↓)	**0.077 ± 0.053**	0.125 ± 0.113	0.121 ± 0.081	0.202 ± 0.158	0.141 ± 0.099	0.206 ± 0.136	0.388 ± 0.253
CRPS (↓)	**0.074 ± 0.047**	0.119 ± 0.100	0.115 ± 0.074	0.196 ± 0.147	0.134 ± 0.092	0.199 ± 0.125	0.471 ± 0.306

E.1.3 Additional settings

As shown in Section E.1.2, `TS-ICL` delivers strong zero-shot imputation performance, both univariate and with auxiliary covariates. To better understand the regimes under which `TS-ICL` provides real value, this section complements the 50% and 70% pointwise missingness levels of `fm-impute-bench` with low (10%, 25%) and severe (90%) levels. We also add the `Cubic Spline` interpolation, representative of polynomial-based methods typically used at low missingness levels.

Results. Table 14 shows the mean absolute errors aggregated per missingness levels, and confirms the trends observed in Section E.1.2. `TS-ICL` achieves the best aggregated MAE at every missingness level, outperforming both Tabular Foundation models and local baselines. Specifically, `TS-ICL`



(a) *Univariate* imputation across 132 tasks. (b) Imputation with *known covariates* across 24 tasks.

Figure 17: Pairwise win rates of the top-4 models on the `fm-impute` benchmark. Each entry indicates the percentage of tasks where a method outperforms another according to the NMAE.

wins 165/165 task-level comparison against the interpolation baselines. These results directly challenge the standard assumption that classical interpolation may be sufficient under moderate missingness: even at 10% and 25% missingness, `TS-ICL` improves upon `Linear Interpolation` by about 30%. As missingness increases, the gap widens further, demonstrating the robustness of `TS-ICL` in sparse regimes. Consistently with the findings of `fm-impute-bench`, `Cubic Spline Interpolation` is prone to overfitting under high missingness rates and is less robust compared to `Linear` [27].

Overall, the results make a strong case for the widespread use of `TS-ICL` across a wide spectrum of practical imputation scenarios, from low to severe sparsity, with or without observed covariates.

Table 14: Aggregated imputation MAE (↓) on the pointwise tasks of the *univariate setting* in `fm-impute-bench` (arithmetic mean, 33 tasks). Best in **bold**.

	TSFM	Tabular Foundation Models		Local Models	
Missingness	`TS-ICL`	`TabPFNv2.5-TS`	`TabICLv2-TS`	`Linear`	`Cubic`
10%	**0.125**	0.141	0.145	0.176	0.200
25%	**0.133**	0.149	0.154	0.192	0.223
50%	**0.155**	0.179	0.179	0.237	0.294
70%	**0.188**	0.225	0.224	0.314	0.446
90%	**0.293**	0.365	0.357	0.529	2.140

E.1.4 Qualitative Analysis and Visualizations

This section presents visual examples of `TS-ICL` imputations for both *univariate* and *known-covariate* settings. We illustrate in Figure 18 and Figure 19 the model imputation capabilities across various missingness patterns covering pointwise and blockwise scenarios.

Results. Several observations emerge from these plots.

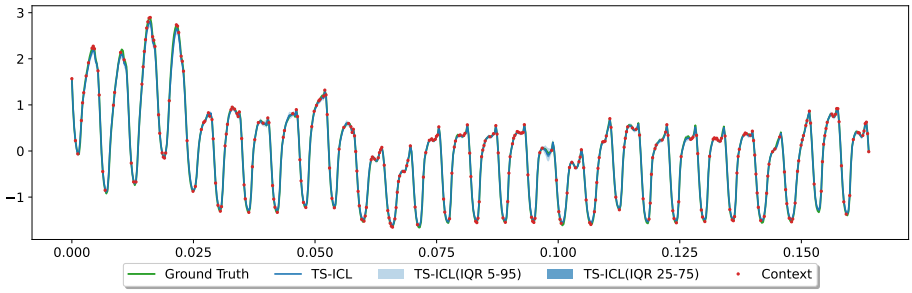
(i) With rich context, `TS-ICL` provides accurate reconstructions of smooth patterns, with tight inter-quantile ranges (Figure 18a). On the contrary, `TS-ICL` adjusts its uncertainty estimation of sparsely observed yet regular patterns (Figures 19b and 19e).

(ii) `TS-ICL` adapts well to distribution shifts (Figures 18b and 19c).

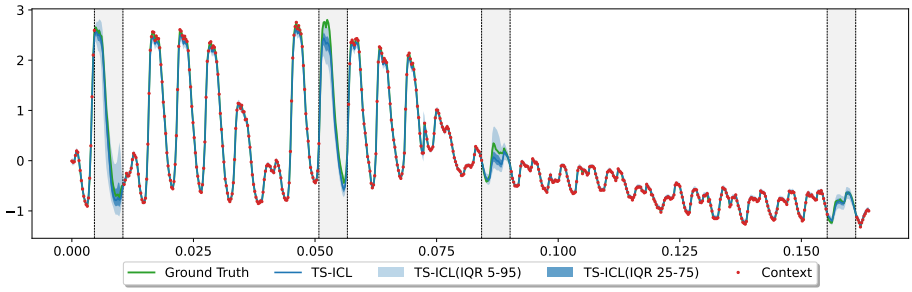
(iii) `TS-ICL` tends to provide smooth reconstructions of sparse high-frequency signals, with higher interquantile ranges accomodating their stronger variability (Figures 18c to 18e).

(iv) Figure 19a (third block) suggests that `TS-ICL` can serve as a counterfactual estimator, replacing unusual sequences with expected ones under regular conditions.

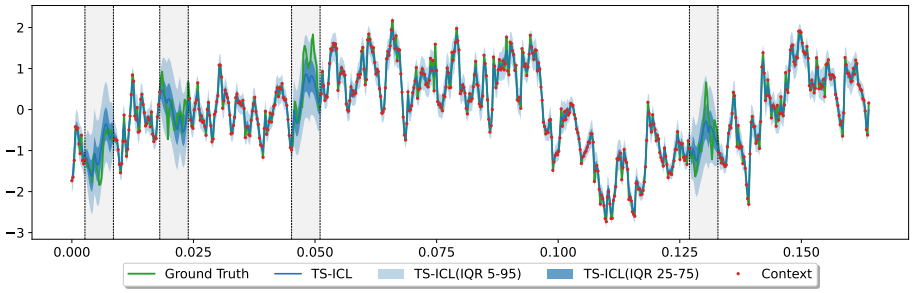
(v) Finally, the ability of `TS-ICL` to incorporate covariate information at inference is key to produce meaningful imputations in challenging scenarios (Figure 19d).



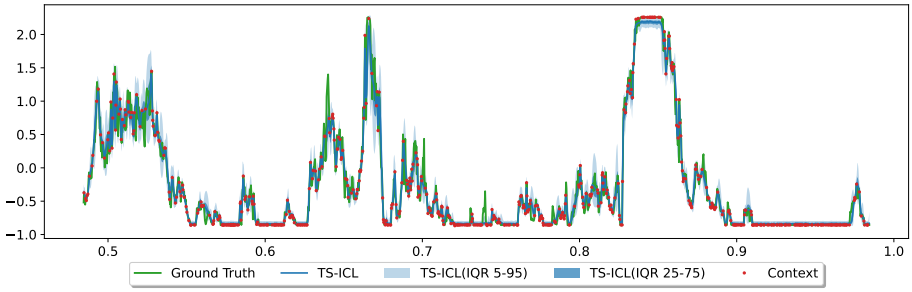
(a) *PDB*, 50% missing values.



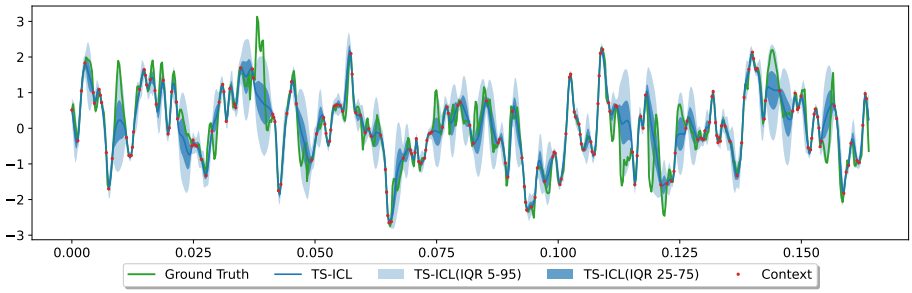
(b) *Covid19 Energy*, four one-day missing blocks.



(c) *BDG2-Rat*, four one-day missing blocks.

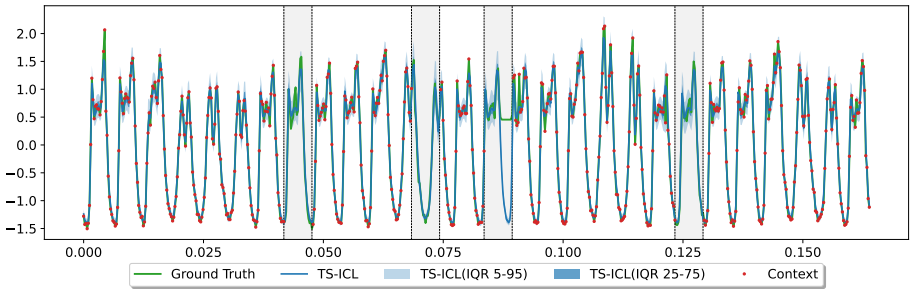


(d) *KDD Cup2022*, 70% missing values.

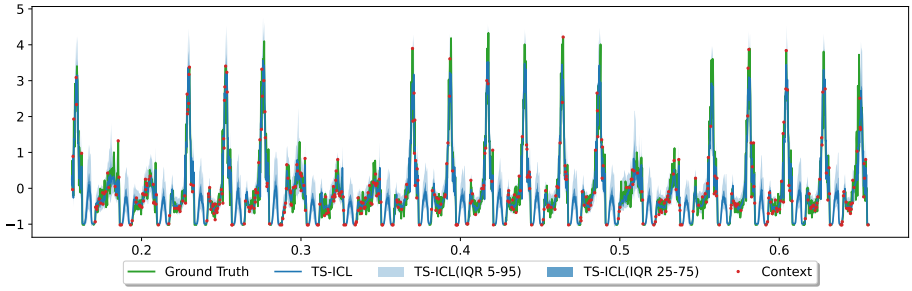


(e) *ERA5 wind speed*, 70% missing values.

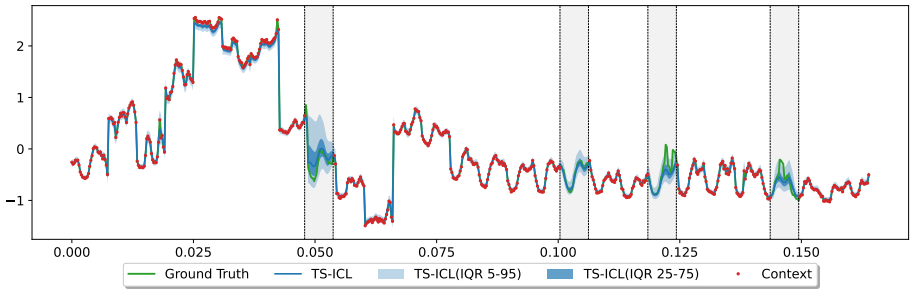
Figure 18: Qualitative assessment of `TS-ICL` imputations on the `fm-impute-bench` benchmark.



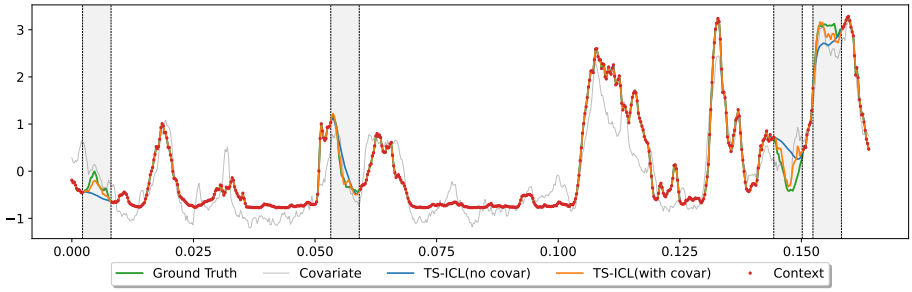
(a) *MDense*, four one-day missing blocks.



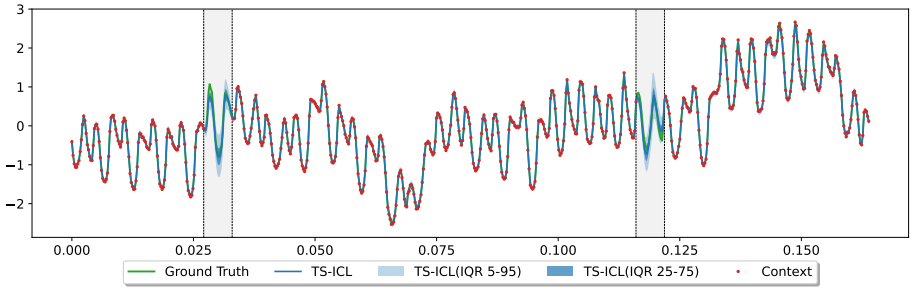
(b) *SHMETRO*, 70% missing values.



(c) *Spanish Energy*, four one-day missing blocks.



(d) *Wind-GE*, four one-day missing blocks.



(e) *GFC12 Load*, two one-day missing blocks.

Figure 19: Qualitative assessment of `TS-ICL` imputations on the `fm-impute-bench` benchmark (continued).

E.2 TIME Benchmark

In this section, we evaluate the zero-shot imputation capability of TS-ICL on TIME [34] a recently introduced benchmark originally designed for TSFM forecasting. We adapt it to cover imputation for the *univariate setting*. Performance is assessed across diverse missingness patterns, sequence lengths, and application domains (details in Table 15).

Table 15: All datasets used for zero-shot imputation in the TIME benchmark.

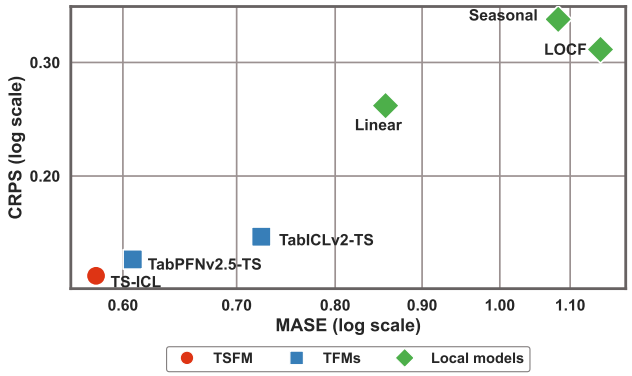
Dataset	Release Platform	Domain	Freq	Num. Series	Num. Variate	Avg Series Length	Short-term		Med-term		Long-term	
							Window Size	Num. Test Windows	Window Size	Num. Test Windows	Window Size	Num. Test Windows
Water Quality-Darwin	IMOS	Nature	15T	7	6	15,229	256	3,780	1024	630	4096	210
Current Velocity	IMOS	Nature	5T	1	6	26,486	256	720	1024	90	4096	30
Current Velocity	IMOS	Nature	10T	10	6	20,669	256	7,200	1024	900	4096	300
Current Velocity	IMOS	Nature	15T	5	6	8,503	256	3,600	1024	450	4096	150
Current Velocity	IMOS	Nature	20T	27	6	6,460	256	19,440	1024	2,430	4096	810
Current Velocity	IMOS	Nature	H	21	6	3,502	256	3,528	1024	504	4096	252
CPHL	IMOS	Nature	15T	2	1	10,831	256	240	1024	30	4096	10
CPHL	IMOS	Nature	30T	2	1	14,687	256	240	1024	60	4096	20
CPHL	IMOS	Nature	H	4	1	4,971	256	112	1024	16	4096	8
Coastal T-S	IMOS	Nature	5T	18	3	68,604	256	6,480	1024	810	4096	270
Coastal T-S	IMOS	Nature	15T	5	3	20,870	256	1,800	1024	225	4096	75
Coastal T-S	IMOS	Nature	20T	1	3	8,198	256	360	1024	45	4096	15
Coastal T-S	IMOS	Nature	H	24	3	5,489	256	2,016	1024	288	4096	144
SG Weather	data.gov.sg	Nature	D	6	4	2,953	256	2,928	1024	1,272	4096	648
SG PM 2.5	data.gov.sg	Nature	H	1	5	38,688	256	460	1024	150	4096	65
NE China Wind	Github	Nature	H	1	4	8,764	256	120	1024	40	4096	16
Australia Solar	Pvoutput	Energy	H	1	3	35,064	256	315	1024	105	4096	45
EPF Electricity	Academic	Energy	H	5	1	52,416	256	525	1024	175	4096	75
OpenElectricity	OpenElec	Energy	5T	1	10	43,488	256	1,680	1024	420	4096	140
EWELD Load	Academic	Energy	15T	1	10	20,544	256	560	1024	140	4096	20
SG Carpark	data.gov.sg	Transport	15T	354	1	14,332	256	14,868	1024	2,478	4096	354
Finland Traffic	Digitraffic	Transport	15T	1	1	35,136	256	186	1024	31	4096	4
Port Activity	Competition	Transport	D	99	2	2,127	256	2,376				
Port Activity	Competition	Transport	W	99	2	304	256	792				
ECDC COVID	ECDC	Healthcare	D	9	1	1,117	256	45				
ECDC COVID	ECDC	Healthcare	W	16	1	165	256	64				
Global Influenza	WHO	Healthcare	W	15	4	205	256	240				
Crypto	FRED	Finance	D	1	4	2,842	256	36				
US Term Structure	FRED	Finance	B	1	40	9,327	256	1,400				
Oil Price	FRED	Finance	B	1	12	5,035	256	420				
Job Claims	FRED	Finance	W	1	2	196	256	8				
Uncertainty-1M	FRED	Economics	M	1	3	780	256	21				
Housing Inventory	FRED	Economics	M	1	4	114	256	12				
JOLTS	FRED	Economics	M	1	6	297	256	30				
US Labor	FRED	Economics	M	1	14	380	256	70				
Vehicle Supply	FRED	Economics	M	1	6	391	256	30				
Auto Production-SF	FRED	Economics	M	1	1	367	256	5				
Commodity Prod.	FRED	Economics	M	32	1	325	256	160				
Commodity Import	FRED	Economics	M	8	1	697	256	40				
WUI-Global	FRED	Economics	Q	1	15	294	256	75				
Global Price	FRED	Economics	Q	1	60	142	256	300				
Vehicle Sales	FRED	Sales	M	1	10	596	256	50				
Online Retail II	Competition	Sales	D	1	1	739	256	6				
Supply Chain-Cust.	Competition	Sales	D	1	36	2,007	256	432				
Supply Chain-Loc.	Competition	Sales	D	1	51	2,007	256	612				
Azure2019-D	Github	CloudOPS	5T	989	3	8,627	256	8,901				
Azure2019-I	Github	CloudOPS	5T	492	3	8,630	256	4,428				
Azure2019-U	Github	CloudOPS	5T	78	3	1,406	256	1,404				
Smart Mfg.	Competition	Industry	H	34	5	1,666	256	2,380	1024	340	4096	170
MetroPT-3	Competition	Industry	5T	1	6	17,809	256	216	1024	36	4096	18

Setting. We reuse the datasets and windowing protocol originally designed for forecasting, treating each lookback window as a partially observed sequence with synthetically introduced missing values. The benchmark spans multiple domains (e.g., energy, finance, healthcare, transportation) and covers diverse lengths and frequencies. Considering short, medium, and long window size configurations yields 98 imputation datasets, each composed of multiple samples to impute. The per-task window sizes can be found in Table 15. To evaluate robustness under different missingness patterns, we define four masking scenarios, namely:

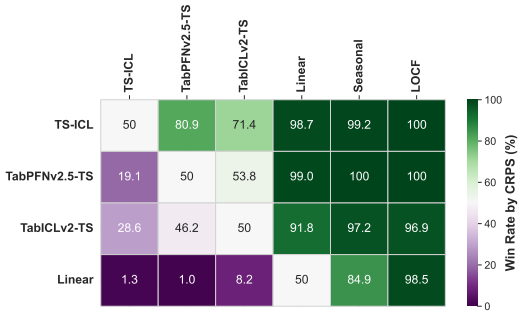
- pointwise masking with 50% missing values;
- pointwise masking with 70% missing values;
- a single contiguous missing block of size $1/15$ of the window; and
- two disjoint missing blocks, each of size $1/15$ of the window.

This setup captures both random and structured missingness, reflecting realistic data corruption patterns. Overall, this results in 98×4 imputation tasks and approximately 440k windows to reconstruct.

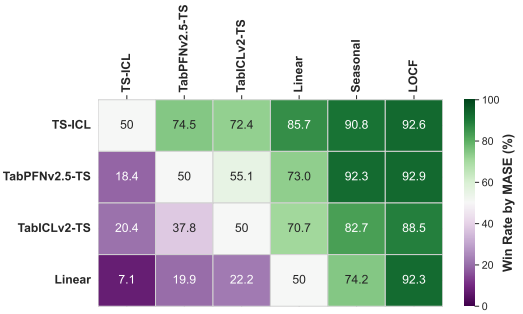
Baselines. TS-ICL is evaluated against the same tabular foundation model and local baselines as those described in Section 5.1. Final aggregated results (arithmetic mean) are reported in Figure 20 (and Table 16), using both probabilistic (CRPS) and scale-normalized pointwise metrics (MASE; see Section D for a definition).



(a) Aggregated scores across 392 tasks - TIME benchmark.



(b) Win rates across 392 tasks (CRPS).



(c) Win rates across 392 tasks (MASE).

Figure 20: Agregated metrics on the TIME univariate time series imputation benchmark. (a) MASE-CRPS (lower is better). Each point corresponds to a method, averaged across 392 tasks. (b-c) Pairwise win rates of the top-4 models. Each entry indicates the percentage of tasks where a method outperforms another according to the (b) CRPS or (c) the MASE.

Results. Results consistently highlight the strong performance of TS-ICL. As illustrated in Figure 20a, TS-ICL achieves the lowest overall error, yielding relative improvements of 5.4% in CRPS and 4.8% in MASE over TabPFNv2.5-TS, the current state-of-the-art tabular foundation model (TFM). Compared to TabICLv2-TS, these gains extend to 13.0% (CRPS) and 20.0% (MASE). Beyond aggregate metrics, TS-ICL demonstrates a dominant pairwise win rate against TabPFNv2.5-TS, outperforming it on 80.9% of tasks for CRPS and 74.5% for MASE (Figure 20c). Notably, TS-ICL achieves this peak accuracy with significant efficiency gains: it maintains an average inference runtime two orders of magnitude faster compared to the most competitive TFMs. This combination positions TS-ICL as a highly scalable solution for large-scale time series imputation.

Table 16: Detailed performance metrics (mean ± std) for the univariate time series on the TIME imputation benchmark (392 tasks). Best in **bold**.

	TSFM	Tabular FMs		Local models		
	TS-ICL	TabPFNv2.5	TabICLv2	Linear	Seasonal	LOCF
MASE (↓)	**0.579 ± 0.323**	0.608 ± 0.281	0.724 ± 0.662	0.857 ± 0.539	1.082 ± 0.243	1.146 ± 0.700
CRPS (↓)	**0.140 ± 0.118**	0.148 ± 0.123	0.161 ± 0.134	0.257 ± 0.228	0.350 ± 0.398	0.314 ± 0.256

F Extended Forecasting Experiments

This section provides broader insights into TS-ICL forecasting performances. A detailed description of the fev-bench datasets used in the main benchmark in Section 5.2 is given in Section F.1, together with complementary results and qualitative visualizations. Section F.2 further extends the zero-shot evaluation in the *univariate setting* to a second benchmark, TIME [34], across 98 tasks and against 12 foundation models. Section F.3 complements the qualitative analysis on time series with strong trends, a known limitation of tabular foundation models.

F.1 Fev-bench Benchmark

Inference datasets. Table 19 details the datasets used for zero-shot forecasting in Section 5.2. As described by [39], these datasets cover a diverse range of domains, with frequencies ranging from 5 minutes to quarterly data. Each forecasting task has its own prediction horizon. In Section 5.2, we distinguish two scenarios: *univariate zero-shot forecasting* and *zero-shot forecasting with known covariates* when available.

Baseline details. We consider the strongest time-series foundation model baselines reported in fev-bench at the time of writing, while excluding models with substantial training-data overlap with the benchmark. In particular, we do not include Moirai-2.0 and TimesFm2.5 among main baselines due to their high reported leakage rates of 28% and 10%, respectively, on fev-bench. All baselines are evaluated in the zero-shot setting, without task-specific fine-tuning. We also report the leakage indicator from [39], defined as the fraction of model–task pairs for which the model pretraining data overlaps with the benchmark data.

- Chronos-2 [3] is a 120M-parameter, patch-based encoder-only transformer closely following the T5 encoder design, with alternating time and group attention layers for in-context learning across related series and covariates. It is the only TSFM baseline in fev-bench that natively supports known-future covariates and handles missing values in the look-back window. Its reported leakage rate is 0%, making it a clean zero-shot baseline.
- TiRex [4] is a 35M-parameter decoder-only xLSTM model for zero-shot probabilistic forecasting. It predicts quantiles directly and does not use covariates in the fev-bench setup. Its reported leakage rate is 1%.
- TimesFM-2.5 [11] is a 200M-parameter patched decoder-only transformer designed for long-context forecasting and direct quantile prediction. It is evaluated as a univariate forecaster in fev-bench. Its reported leakage rate is 10%, so its aggregate score should be interpreted with some caution.
- Toto-1.0 [10] is a 151M-parameter decoder-only transformer optimized for multivariate observability time series. It supports multivariate inputs, but does not use known future covariates in the fev-bench setting. Its reported leakage rate is 8%, which is non-negligible but substantially lower than that of Moirai-2.0.
- Chronos-Bolt [2] is a 205M-parameter T5 encoder–decoder model and a patch-based variant of Chronos. It chunks the historical context into patches and produces multi-step quantile forecasts. It is evaluated as a univariate forecaster in fev-bench. Its reported leakage rate is 0%.

F.1.1 Extended Results

This section extends the empirical evaluation in Section 5.2 with a more detailed analysis of forecasting performance across all experimental settings. Specifically, we provide:

- **Aggregated detailed performance tables:** We report the average MASE and CRPS (metrics definition in Section D) across the *univariate* tasks and *covariates-aware* tasks of fm-impute-bench. These results, detailed in Table 17 and Table 18, providing a detailed view of both point-wise and probabilistic performance. Note that metrics are aggregated across tasks using the geometric mean, following the evaluation protocol established in fev-bench.

- **MASE pairwise win rates:** To complement the CRPS-based win rate diagrams presented in the main text (Figures 5c and 5d), we include the corresponding pairwise win rate visualizations in terms of MASE for both *univariate* and *known-covariate* experiments in Figure 21.

Table 17: `Fev-bench` *univariate* forecasting (100 tasks). Performance metrics aggregated (geometric mean ± geometric std). Best in **bold**.

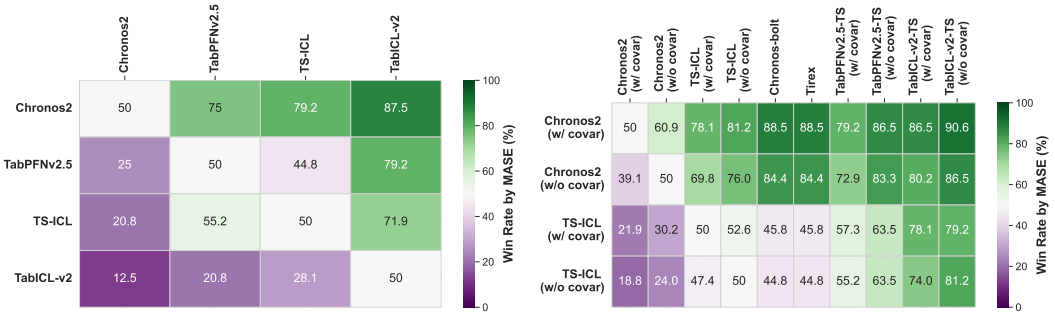
	TSFM				
	`TS-ICL`	`Chronos-2`	`Chronos-Bolt`	`TiRex`	`Toto`
MASE (↓)	1.150 ± 2.175	**1.081** ± **2.201**	1.158 ± 2.154	1.102 ± 2.134	1.773 ± 2.148
CRPS (↓)	0.137 ± 2.995	**0.129** ± **3.009**	0.140 ± 3.027	0.132 ± 3.064	0.135 ± 2.958

	Tabular Foundation models		Local models	
	`TabPFNv2.5`	`TabICLv2`	`Seasonal`	`LOCF`
MASE (↓)	1.218 ± 2.173	1.400 ± 2.129	1.547 ± 2.062	1.839 ± 2.243
CRPS (↓)	0.141 ± 3.028	0.159 ± 3.252	0.241 ± 2.954	0.260 ± 2.682

Table 18: `Fev-bench` forecasting on 100 tasks (30 *covariate-aware*) forecasting (100 tasks). Performance metrics aggregated (geometric mean ± geometric std). Best in **bold**.

	TSFM			
	`TS-ICL`		`Chronos-2`	
	w/ covar	*w/o covar*	*w/ covar*	*w/o covar*
MASE (↓)	1.117 ± 2.232	1.150 ± 2.175	**1.034** ± **2.250**	1.081 ± 2.202
CRPS (↓)	0.131 ± 3.000	0.137 ± 2.995	**0.123** ± **3.062**	0.129 ± 3.018

	Tabular Foundation Models		Univariate FMs	
	`TabPFNv2.5-TS` (w/ covar)	`TabICLv2-TS` (w/ covar)	`Chronos-Bolt` (univar.)	`TiRex` (univar.)
MASE (↓)	1.162 ± 2.235	1.342 ± 2.194	1.158 ± 2.154	1.102 ± 2.134
CRPS (↓)	0.134 ± 3.060	0.153 ± 3.279	0.140 ± 3.027	0.132 ± 3.064



(a) *Univariate* forecasting across 100 tasks. (b) Forecasting with *known covariates* across 30 tasks.

Figure 21: Pairwise win rates for forecating on the `fev-bench` benchmark. Each entry indicates the percentage of tasks where a method outperforms another according to the MASE.

F.1.2 Qualitative Analysis and Visualizations

This section presents visual examples of `TS-ICL` imputations for both *univariate* and *known-covariates* settings. We illustrate model forecasting capabilities across various *fev-bench* tasks.

Results. Several observations emerge from the forecasting plots in Figure 22 (*known covariate setting*) and Figures 23 to 25(*univariate setting*), where the median forecast is shown together with the corresponding 25-75 and 5-95 inter-quantile ranges.

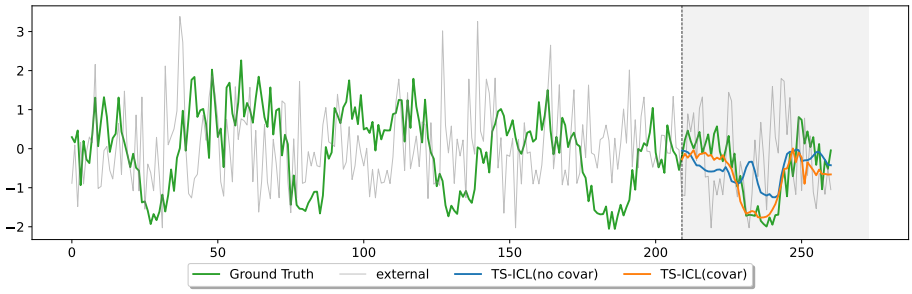
(i) The plots highlight the general ability of `TS-ICL` to extrapolate from long context windows (with a maximum lookback length of 4096) and regular patterns in heterogeneous sampling rates, domains and seasonalities (e.g. Figures 23a, 23b, 24a and 25c).

(ii) Similarly to the imputation setting, `TS-ICL` tends to provide smooth forecasts of high-frequency phenomena and adjusts its inter-quantile range accordingly (e.g. Figures 23d, 24c and 25e).

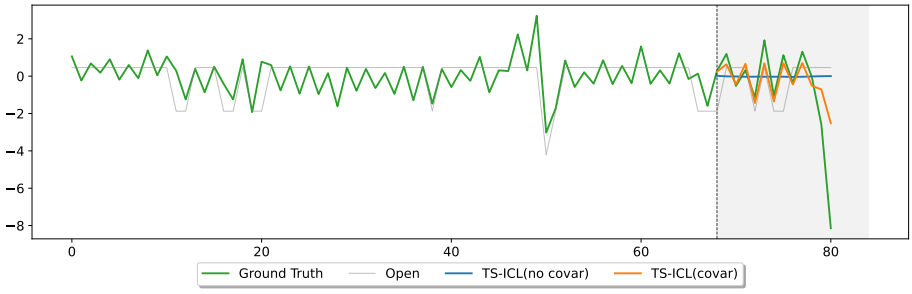
(iii) In the *univariate setting*, extrapolating from very short contexts is particularly challenging. `TS-ICL` compensates with wider inter-quantile ranges, with mixed success depending on the regularity of the underlying phenomena (Figures 23e, 24e, 25a and 25b).

(iv) In the *known covariate setting*, `TS-ICL` manages to leverage additional covariate, when the latter informs about the target, while mostly ignoring it otherwise (Figure 22).

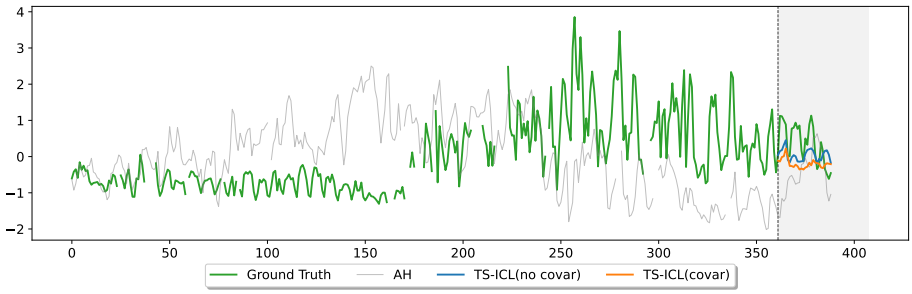
(v) Figure 25d gives an example of forecasting with missing values, with `TS-ICL` providing adequate uncertainty estimates.



(a) *Hermes/W* - $H = 52$.

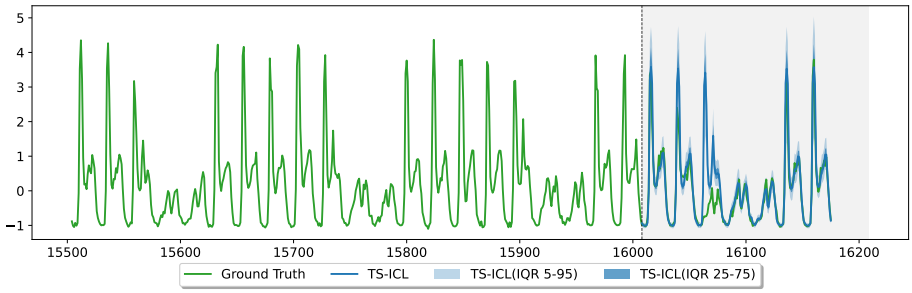


(b) *Rossmann/W* - $H = 13$.

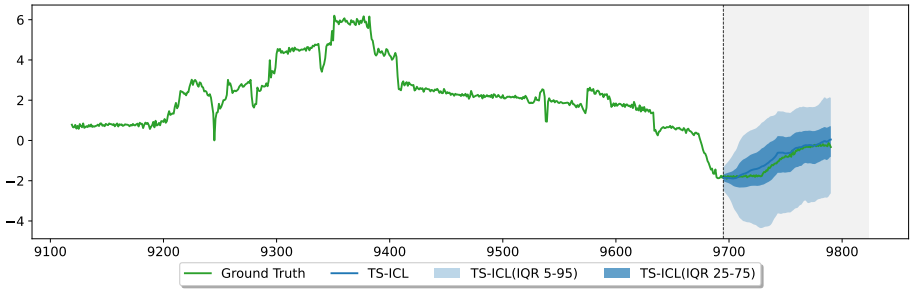


(c) *UCI Air Quality/D* - $H = 28$.

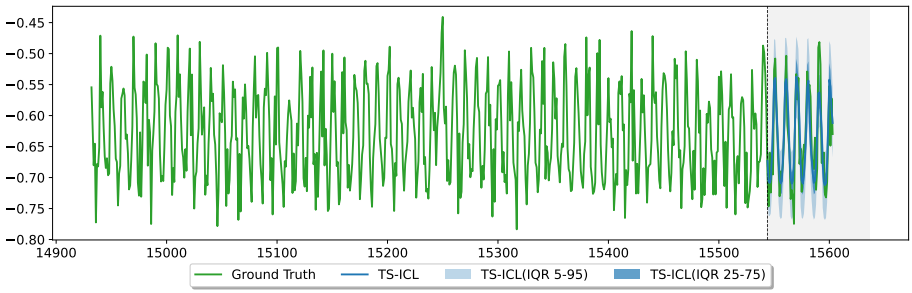
Figure 22: Qualitative assessment of `TS-ICL` forecasts on the `fev-bench` benchmark, in the *known covariate setting*. Covariates are shown in light gray.



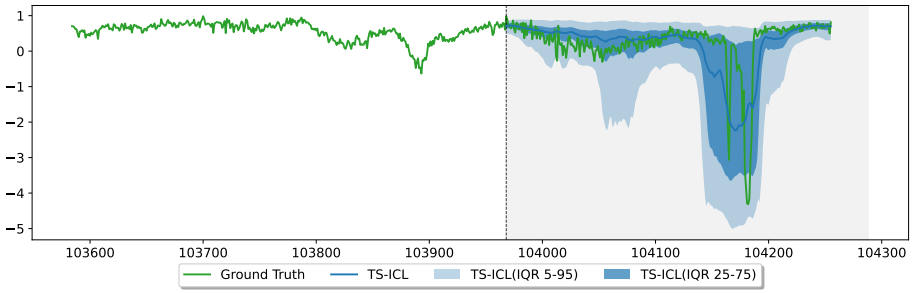
(a) *M-DENSE/1H* - $H = 168$.



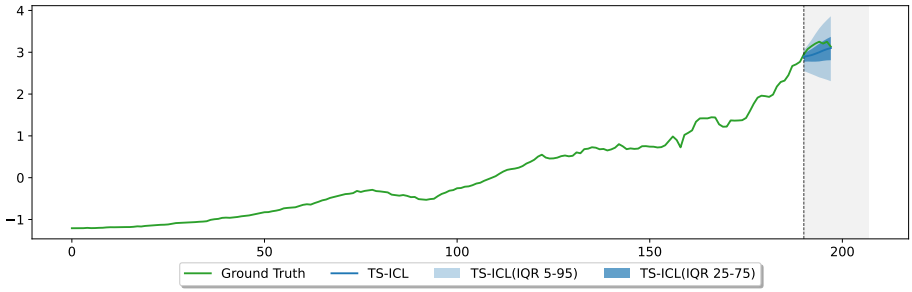
(b) *BOOMLET - 1631/30T* - $H = 96$.



(c) *BOOMLET - 1225/1T* - $H = 96$.

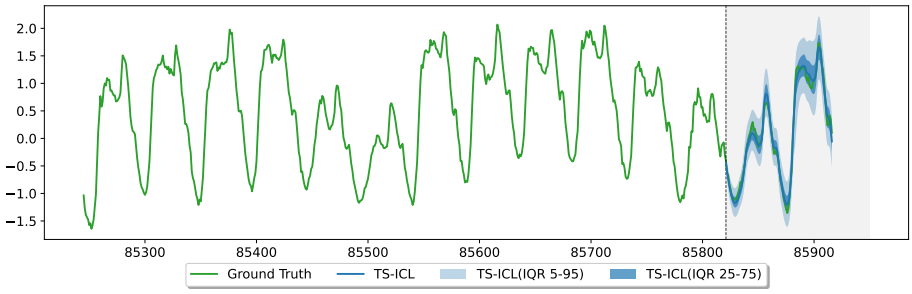


(d) *Loop Seattle/5T* - $H = 288$.

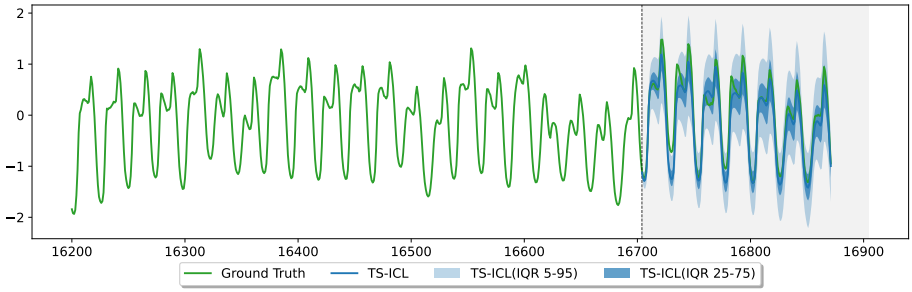


(e) *US Consumption/1Q* - $H = 8$.

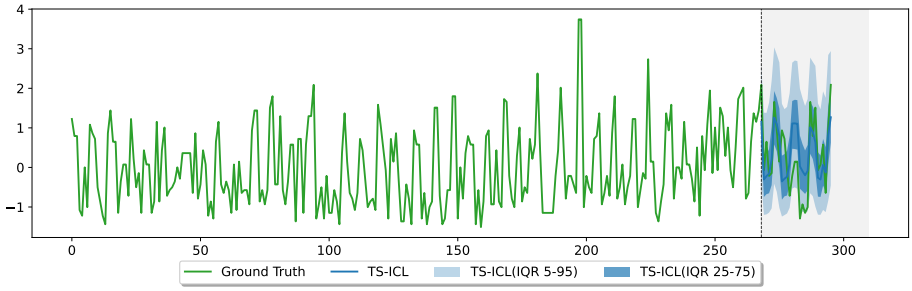
Figure 23: Qualitative assessment of `TS-ICL` forecasts on the `fev-bench` benchmark.



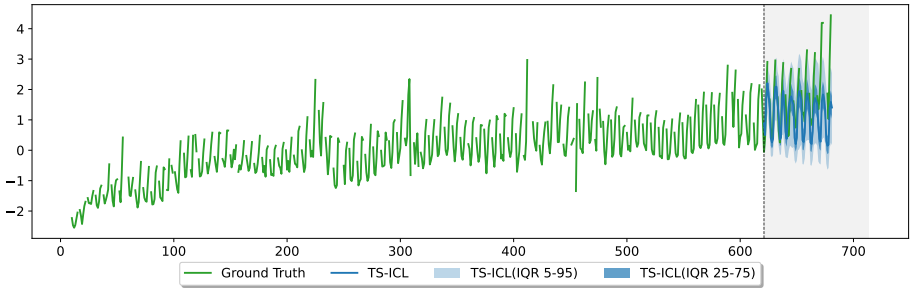
(a) *ENTSOE-e Load/30T* - $H = 96$.



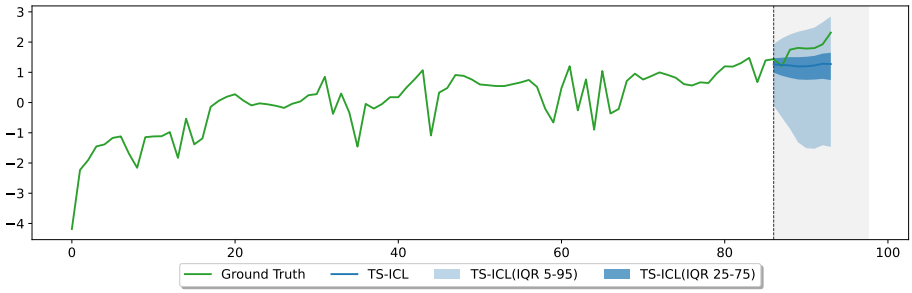
(b) *GFC17/1H* - $H = 168$.



(c) *Restaurant/1D* - $H = 28$.

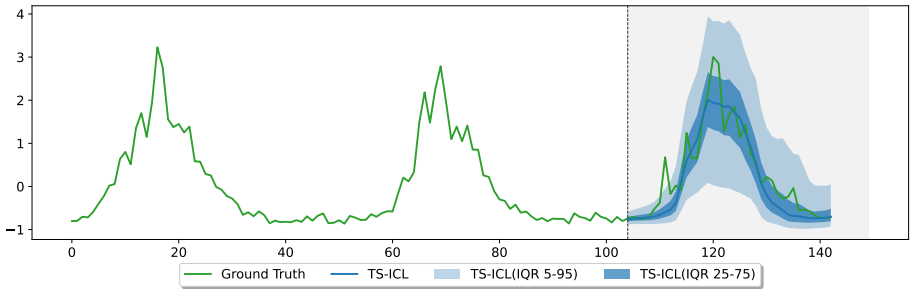


(d) *Rohlik Orders/1D* - $H = 61$.

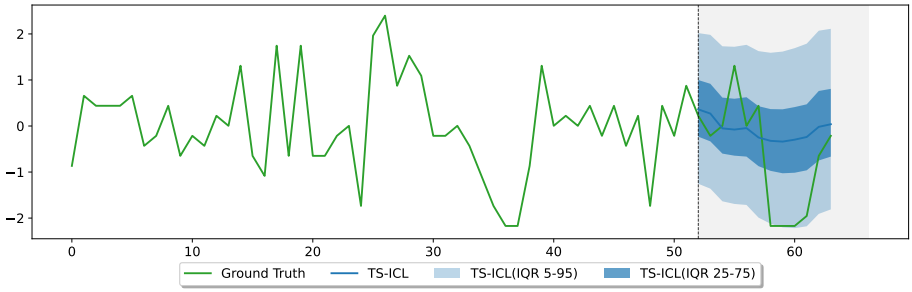


(e) *Rohlik Orders/1W* - $H = 8$.

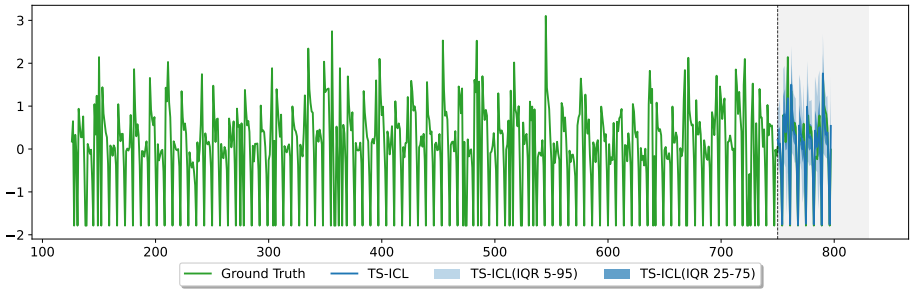
Figure 24: Qualitative assessment of `TS-ICL` forecasts on the `fev-bench` benchmark (continued).



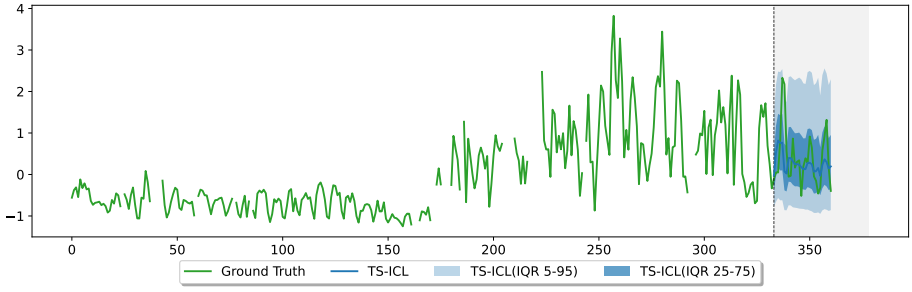
(a) *Walmart/1W* - $H = 39$.



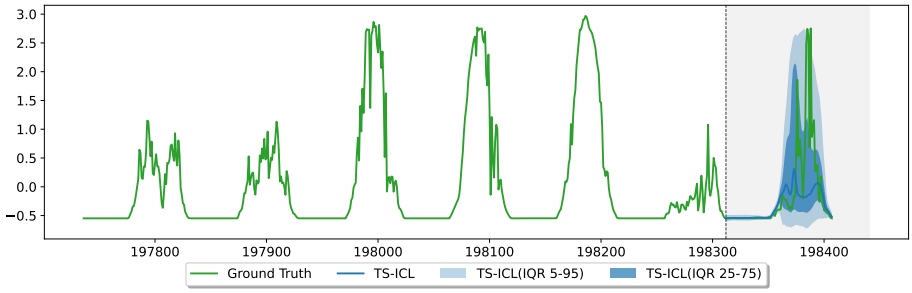
(b) *M5/1M* - $H = 12$.



(c) *Rossmann/1D* - $H = 48$.



(d) *UCI Air Quality/1D* - $H = 28$.



(e) *Solar with Weather/15T* - $H = 96$.

Figure 25: Qualitative assessment of `TS-ICL` forecasts on the `fev-bench` benchmark (continued).

Table 19: Full statistics of `fev-bench` tasks with data sources. Covariates: P (Past), K (Known), S (Static).

Task / Dataset	Source	Domain	Freq	Horizons	Num. Series	Num. Target	Median Length	Covariates			Num. Test Windows
								P	K	S	
BizITObs - L2C	LOTSA	cloud	5T	288	1	7	31,968	0	0	0	140
BizITObs - L2C	LOTSA	cloud	H	24	1	7	2,664	0	0	0	140
ETT	GitHub	energy	15T	96	2	7	69,680	0	0	0	280
ETT	GitHub	energy	H	168	2	7	17,420	0	0	0	280
ETT	GitHub	energy	D	28	2	7	724	0	0	0	280
ETT	GitHub	energy	W	13	2	7	103	0	0	0	70
Hierarchical Sales	LOTSA	retail	D	28	118	1	1,825	0	0	0	1,180
Hierarchical Sales	LOTSA	retail	W	13	118	1	260	0	0	0	1,180
Hospital	LOTSA	healthcare	M	12	767	1	84	0	0	0	3,068
Jena Weather	MPI Jena	nature	10T	144	1	21	52,704	0	0	0	420
Jena Weather	MPI Jena	nature	D	28	1	21	366	0	0	0	231
Jena Weather	MPI Jena	nature	H	24	1	21	8,784	0	0	0	420
Loop Seattle	LOTSA	mobility	D	28	323	1	365	0	0	0	3,230
Loop Seattle	LOTSA	mobility	5T	288	323	1	105,120	0	0	0	3,230
Loop Seattle	LOTSA	mobility	H	168	323	1	8,760	0	0	0	3,230
M-DENSE	LOTSA	mobility	D	28	30	1	730	0	0	0	300
M-DENSE	LOTSA	mobility	H	168	30	1	17,520	0	0	0	300
SZ Taxi	LOTSA	mobility	15T	96	156	1	2,976	0	0	0	1,560
SZ Taxi	LOTSA	mobility	H	168	156	1	744	0	0	0	312
Solar	LOTSA	energy	W	13	137	1	52	0	0	0	137
Solar	LOTSA	energy	D	28	137	1	365	0	0	0	1,370
Australian Tourism	Monash	econ	Q	8	89	1	36	0	0	0	178
FRED-MD - CEE	Fed	econ	M	12	1	3	798	4	0	0	60
FRED-MD - Macro	Fed	econ	M	12	1	51	798	0	0	0	1,020
FRED-QD - CEE	Fed	econ	Q	8	1	3	266	4	0	0	60
FRED-QD - Macro	Fed	econ	Q	8	1	51	266	0	0	0	1,020
GVAR	Mohaddes	econ	Q	8	33	6	178	3	0	0	1,980
US Consumption	FPP3	econ	M	12	31	1	792	0	0	0	310
US Consumption	FPP3	econ	Q	8	31	1	262	0	0	0	310
US Consumption	FPP3	econ	Y	5	31	1	64	0	0	0	310
World CO2 Emissions	WorldBank	econ	Y	5	191	1	60	0	0	0	1,719
World Life Expectancy	WorldBank	econ	Y	5	237	1	74	0	0	0	2,370
World Tourism	WorldBank	econ	Y	5	178	1	21	0	0	0	356
ENTSO-e Load	ENTSO-E	energy	15T	96	6	1	175,292	0	3	0	120
ENTSO-e Load	ENTSO-E	energy	30T	96	6	1	87,645	0	3	0	120
ENTSO-e Load	ENTSO-E	energy	H	168	6	1	43,822	0	3	0	120
EPF-BE	GitHub	energy	H	24	1	1	52,416	0	2	0	20
EPF-DE	GitHub	energy	H	24	1	1	52,416	0	2	0	20
EPF-FR	GitHub	energy	H	24	1	1	52,416	0	2	0	20
EPF-NP	GitHub	energy	H	24	1	1	52,416	0	2	0	20
EPF-PJM	GitHub	energy	H	24	1	1	52,416	0	2	0	20
ERCOT	ERCOT	energy	D	28	8	1	6,452	0	0	0	160
ERCOT	ERCOT	energy	H	168	8	1	154,872	0	0	0	160
ERCOT	ERCOT	energy	M	12	8	1	211	0	0	0	120
ERCOT	ERCOT	energy	W	13	8	1	921	0	0	0	160
GFC12	LOTSA	energy	H	168	11	1	39,414	0	1	0	110
GFC14	LOTSA	energy	H	168	1	1	17,520	0	1	0	20
GFC17	LOTSA	energy	H	168	8	1	17,544	0	1	0	160
Solar with Weather	Kaggle	energy	15T	96	1	1	198,600	2	7	0	20
Solar with Weather	Kaggle	energy	H	24	1	1	49,648	2	7	0	20
BOOMLET - 1062	BOOM	cloud	5T	288	1	21	16,384	0	0	0	420
BOOMLET - 1209	BOOM	cloud	5T	288	1	53	16,384	0	0	0	1,060
BOOMLET - 1225	BOOM	cloud	T	60	1	49	16,384	0	0	0	980
BOOMLET - 1230	BOOM	cloud	5T	288	1	23	16,384	0	0	0	460
BOOMLET - 1282	BOOM	cloud	T	60	1	35	16,384	0	0	0	700
BOOMLET - 1487	BOOM	cloud	5T	288	1	54	16,384	0	0	0	1,080
BOOMLET - 1631	BOOM	cloud	30T	96	1	40	10,463	0	0	0	800
BOOMLET - 1676	BOOM	cloud	30T	96	1	100	10,463	0	0	0	2,000
BOOMLET - 1855	BOOM	cloud	H	24	1	52	5,231	0	0	0	1,040
BOOMLET - 1975	BOOM	cloud	H	24	1	75	5,231	0	0	0	1,500
BOOMLET - 2187	BOOM	cloud	H	24	1	100	5,231	0	0	0	2,000
BOOMLET - 285	BOOM	cloud	T	60	1	75	16,384	0	0	0	1,500
BOOMLET - 619	BOOM	cloud	T	60	1	52	16,384	0	0	0	1,040
BOOMLET - 772	BOOM	cloud	T	60	1	67	16,384	0	0	0	1,340
BOOMLET - 963	BOOM	cloud	T	60	1	28	16,384	0	0	0	560
Favorita Store Sales	Kaggle	retail	M	12	1,579	1	54	1	1	6	3,158
Favorita Store Sales	Kaggle	retail	W	13	1,579	1	240	1	1	6	15,790
Favorita Store Sales	Kaggle	retail	D	28	1,579	1	1,688	1	2	6	15,790
Favorita Transactions	Kaggle	retail	M	12	51	1	54	1	0	5	102
Favorita Transactions	Kaggle	retail	W	13	51	1	240	1	0	5	510
Favorita Transactions	Kaggle	retail	D	28	51	1	1,688	1	1	5	510
KDD Cup 2022	Kaggle	energy	D	14	134	1	243	9	0	0	1,340
KDD Cup 2022	Kaggle	energy	10T	288	134	1	35,279	9	0	0	1,340
KDD Cup 2022	Kaggle	energy	30T	96	134	1	11,758	9	0	0	1,340
M5	Kaggle	retail	M	12	30,490	1	58	0	8	5	30,490
M5	Kaggle	retail	W	13	30,490	1	257	0	8	5	30,490
M5	Kaggle	retail	D	28	30,490	1	1,810	0	8	5	30,490
Restaurant	Kaggle	retail	D	28	817	1	296	0	0	4	6,536
Rohlik Orders	Kaggle	retail	W	8	7	1	170	9	4	0	35
Rohlik Orders	Kaggle	retail	D	61	7	1	1,197	9	4	0	35
Rohlik Sales	Kaggle	retail	W	8	5,243	1	150	1	13	7	5,243
Rohlik Sales	Kaggle	retail	D	14	5,390	1	1,046	1	13	7	5,390
Rossmann	Kaggle	retail	W	13	1,115	1	133	1	4	10	8,920
Rossmann	Kaggle	retail	D	48	1,115	1	942	1	5	10	11,150
Walmart	Kaggle	retail	W	39	2,936	1	143	0	10	4	2,936
ECDC ILI	ECDC	healthcare	W	13	25	1	201	0	0	0	250
Hermes	LOTSA	retail	W	52	10,000	1	261	0	1	2	10,000
Hospital Admissions	Gov.UK	healthcare	D	28	8	1	1,731	0	0	0	160
Hospital Admissions	Gov.UK	healthcare	W	13	8	1	246	0	0	0	128
Redset	GitHub	cloud	5T	288	118	1	25,920	0	0	1	1,180
Redset	GitHub	cloud	15T	96	126	1	8,640	0	0	1	1,260
Redset	GitHub	cloud	H	24	138	1	2,160	0	0	1	1,380
UCI Air Quality	UCI	nature	H	168	1	4	9,357	0	3	0	80
UCI Air Quality	UCI	nature	D	28	1	4	389	0	3	0	44
UK COVID - Nation - Cumul.	Gov.UK	healthcare	D	28	4	3	729	5	0	0	240
UK COVID - Nation - Cumul.	Gov.UK	healthcare	W	8	4	3	105	5	0	0	48
UK COVID - Nation - New	Gov.UK	healthcare	D	28	4	3	729	5	0	0	240
UK COVID - Nation - New	Gov.UK	healthcare	W	8	4	3	105	5	0	0	48
UK COVID - UTLA - Cumul.	Gov.UK	healthcare	W	13	214	1	104	0	0	0	1,070
UK COVID - UTLA - New	Gov.UK	healthcare	D	28	214	1	721	0	0	0	2,140

F.2 TIME Benchmark

In this section, we evaluate the zero-shot forecasting capabilities of TS-ICL on the TIME benchmark [34], covering univariate settings. This benchmark is particularly valuable as it provides a rigorous framework for zero-shot evaluation, ensuring a total absence of data leakage across all compared foundation models. Performance is assessed across a wide range of missingness patterns, sequence lengths, and application domains (see Table 20 for details).

Table 20: Individual statistics of forecasting tasks across all datasets. Freq denotes the sampling frequency.

Dataset	Release Platform	Domain	Freq	Num. Series	Num. Variate	Avg Series Length	Short-term		Med-term		Long-term	
							Horizon	Num. Test Windows	Horizon	Num. Test Windows	Horizon	Num. Test Windows
Water Quality-Darwin	IMOS	Nature	15T	7	6	15,229	16 (4H)	3,780	96 (D)	630	288 (3D)	210
Current Velocity	IMOS	Nature	5T	1	6	26,486	36 (3H)	720	288 (D)	90	864 (3D)	30
Current Velocity	IMOS	Nature	10T	10	6	20,669	18 (3H)	7,200	144 (D)	900	432 (3D)	300
Current Velocity	IMOS	Nature	15T	5	6	8,503	12 (3H)	3,600	96 (D)	450	288 (3D)	150
Current Velocity	IMOS	Nature	20T	27	6	6,460	9 (3H)	19,440	72 (D)	2,430	216 (3D)	810
Current Velocity	IMOS	Nature	H	21	6	3,502	24 (D)	3,528	168 (W)	504	336 (2W)	252
CPHL	IMOS	Nature	15T	2	1	10,831	12 (3H)	240	96 (D)	30	288 (3D)	10
CPHL	IMOS	Nature	30T	2	1	14,687	12 (3H)	240	48 (D)	60	144 (3D)	20
CPHL	IMOS	Nature	H	4	1	4,971	24 (D)	112	168 (W)	16	336 (2W)	8
Coastal T-S	IMOS	Nature	5T	18	3	68,604	36 (3H)	6,480	288 (D)	810	864 (3D)	270
Coastal T-S	IMOS	Nature	15T	5	3	20,870	12 (3H)	1,800	96 (D)	225	288 (3D)	75
Coastal T-S	IMOS	Nature	20T	1	3	8,198	9 (3H)	360	72 (D)	45	216 (3D)	15
Coastal T-S	IMOS	Nature	H	24	3	5,489	24 (D)	2,016	168 (W)	288	336 (2W)	144
SG Weather	data.gov.sg	Nature	D	6	4	2,953	3 (3D)	2,928	7 (W)	1,272	14 (2W)	648
SG PM 2.5	data.gov.sg	Nature	H	1	5	38,688	24 (D)	460	72 (3D)	150	168 (W)	65
NE China Wind	Github	Nature	H	1	4	8,764	24 (D)	120	72 (3D)	40	168 (W)	16
Australia Solar	Pvoutput	Energy	H	1	3	35,064	24 (D)	315	72 (3D)	105	168 (W)	45
EPF Electricity	Academic	Energy	H	5	1	52,416	24 (D)	525	72 (3D)	175	168 (W)	75
OpenElectricity	OpenElec	Energy	5T	1	10	43,488	24 (2H)	1,680	96 (8H)	420	288 (D)	140
EWELD Load	Academic	Energy	15T	1	10	20,544	24 (6H)	560	96 (D)	140	672 (W)	20
SG Carpark	data.gov.sg	Transport	15T	354	1	14,332	16 (4H)	14,868	96 (D)	2,478	672 (W)	354
Finland Traffic	Digitraffic	Transport	15T	1	1	35,136	16 (4H)	186	96 (D)	31	672 (W)	4
Port Activity	Competition	Transport	D	99	2	2,127	30 (M)	2,376				
Port Activity	Competition	Transport	W	99	2	304	13 (Q)	792				
ECDC COVID	ECDC	Healthcare	D	9	1	1,117	30 (30D)	45				
ECDC COVID	ECDC	Healthcare	W	16	1	165	13 (Q)	64				
Global Influenza	WHO	Healthcare	W	15	4	205	13 (Q)	240				
Crypto	FRED	Finance	D	1	4	2,842	30 (M)	36				
US Term Structure	FRED	Finance	B	1	40	9,327	20 (4W)	1,400				
Oil Price	FRED	Finance	B	1	12	5,035	20 (4W)	420				
Job Claims	FRED	Finance	W	1	2	196	13 (Q)	8				
Uncertainty-1M	FRED	Economics	M	1	3	780	6 (6M)	21				
Housing Inventory	FRED	Economics	M	1	4	114	12 (A)	12				
JOLTS	FRED	Economics	M	1	6	297	12 (A)	30				
US Labor	FRED	Economics	M	1	14	380	12 (A)	70				
Vehicle Supply	FRED	Economics	M	1	6	391	12 (A)	30				
Auto Production-SF	FRED	Economics	M	1	1	367	12 (A)	5				
Commodity Prod.	FRED	Economics	M	32	1	325	12 (A)	160				
Commodity Import	FRED	Economics	M	8	1	697	12 (A)	40				
WUI-Global	FRED	Economics	Q	1	15	294	4 (A)	75				
Global Price	FRED	Economics	Q	1	60	142	4 (A)	300				
Vehicle Sales	FRED	Sales	M	1	10	596	12	50				
Online Retail II	Competition	Sales	D	1	1	739	30	6				
Supply Chain-Cust.	Competition	Sales	D	1	36	2,007	30	432				
Supply Chain-Loc.	Competition	Sales	D	1	51	2,007	30	612				
Azure2019-D	Github	CloudOPS	5T	989	3	8,627	288 (D)	8,901				
Azure2019-I	Github	CloudOPS	5T	492	3	8,630	288 (D)	4,428				
Azure2019-U	Github	CloudOPS	5T	78	3	1,406	48 (4H)	1,404				
Smart Mfg.	Competition	Industry	H	34	5	1,666	24 (D)	2,380	168 (W)	340	336 (2W)	170
MetroPT-3	Competition	Industry	5T	1	6	17,809	48 (4H)	216	288 (D)	36	576 (2D)	18

Setting. The evaluation is conducted under a *primarily univariate forecasting* setting, where Time considers short-, medium-, and long-context window sizes, resulting in 98 forecasting tasks and approximately 110k windows to predict. Note, however, that some baselines (namely Chronos-2, Toto, Moirai, and VisionTS) leverage look-back windows from others series at inference time, operating in a multivariate forecasting regime [34].

Baselines. The guaranteed absence of data leakage in the TIME benchmark allows us to expand the comparison to a broader set of state-of-the-art foundation models that were previously excluded in Section 5.2. Final aggregated results are reported in Figure 26 (and Table 21), using both probabilistic (CRPS) and scale-normalized pointwise metrics (MASE; see Section D for a definition). We compare TS-ICL against:

- Time Series Foundation Models (TSFMs): Chronos-2 [3], Timesfm2_5 [11], TiRex [4], Moirai2 [28], Toto [10], Chronos-bolt, Sundial [29], TimesFm [11], Vision_ts [7], and Moirai [44].

- Tabular Foundation Models (TFMs): `TabPFNv2.5-TS` [17] and `TabICLv2-TS` [35].

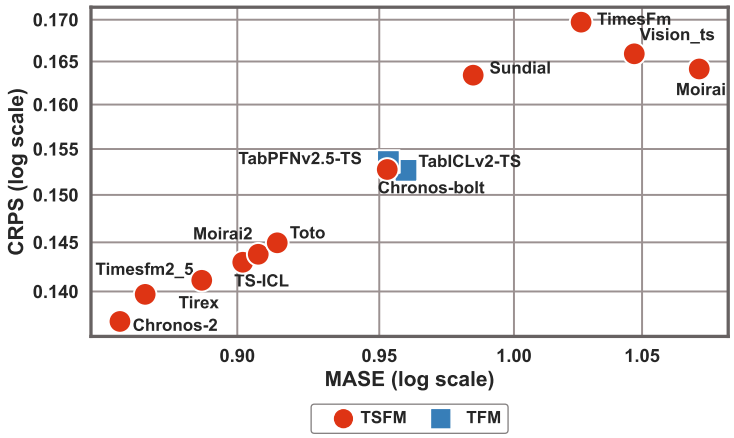


Figure 26: MASE-CRPS time trade-off (lower is better) on the `TIME` benchmark. The x-axis reports task-averaged MASE for each method, while the y-axis shows task-averaged CRPS for each method. Each point corresponds to a method, averaged across tasks.

Table 21: Detailed forecasting performance metrics aggregated (geometric mean ± geometric std) across the 98 tasks of the *univariate setting* in the `TIME` benchmark. Best in **bold**.

	Time Series Foundation Models (TSFMs)						
	`Chronos-2`	`TimesFM2_5`	`TiRex`	`TS-ICL`	`Toto`	`Moirai2`	`Bolt`
MASE (↓)	**0.861±1.621**	0.869±1.613	0.888±1.585	0.902±1.597	0.907±1.597	0.914±1.604	0.953±1.629
CRPS (↓)	**0.137±2.885**	0.140±2.902	0.141±2.866	0.143±2.807	0.144±2.821	0.145±2.833	0.153±2.688

	Tabular FMs		Other TSFMs			
	`TabPFNv2.5-TS`	`TabICLv2-TS`	`Sundial`	`TimesFM`	`Moirai`	`Vision_ts`
MASE (↓)	0.953±1.601	0.960±1.590	0.985±1.675	1.026±1.593	1.073±1.598	1.047±1.603
CRPS (↓)	0.154±2.852	0.153±2.872	0.163±2.666	0.170±2.840	0.164±2.573	0.166±2.753

Results. `TS-ICL` demonstrates strong competitive performance on the `Time` benchmark, consistently ranking among the top-tier Time Series Foundation Models (TSFMs). As detailed in Table 21, while `Chronos-2` maintains its position as the SOTA leader, `TS-ICL` achieves a highly comparable MASE of 0.902 and a CRPS of 0.143. Notably, the performance gap between `TS-ICL` model and `Chronos-2` is minimal, with `TS-ICL` trailing by only 4.7% in MASE and 4.3% in CRPS in relative terms. This narrow margin is further evidenced by the pairwise win rate analysis (Figures 27 and 28), where `TS-ICL` successfully outperforms `Chronos-2` on 29.6% of tasks for CRPS and 22.4% for MASE.

Beyond this head-to-head comparison, `TS-ICL` shows a clear superiority over the entire category of Tabular Foundation Models (TFMs), significantly outperforming both `TabPFN` and `TabICL`. It also surpasses several established TSFMs, including `TimesFM`, `Moirai`, and `Sundial`. By matching the performance of much larger, specialized models within such a tight margin while offering a more efficient architecture, `TS-ICL` establishes itself as a robust and scalable alternative for high-performance zero-shot forecasting.

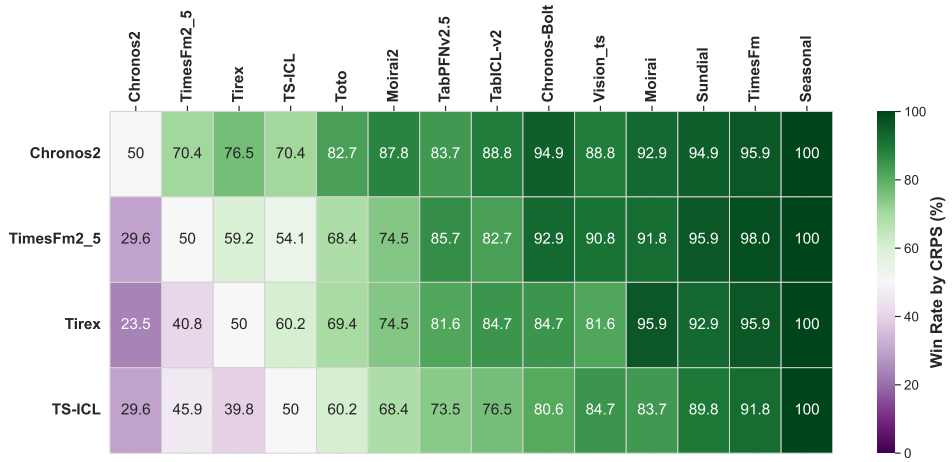


Figure 27: Pairwise win rates for the top-4 models against all other forecasters on the TIME benchmark. Each entry indicates the percentage of tasks where a method outperforms another according to the CRPS.

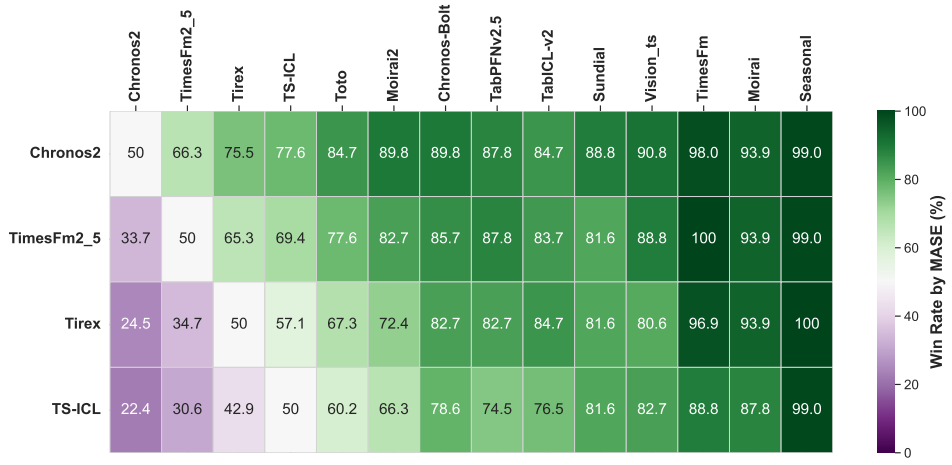


Figure 28: Pairwise win rates for the top-4 models against all other forecasters on the TIME benchmark. Each entry indicates the percentage of tasks where a method outperforms another according to the MASE.

F.3 Extended qualitative analysis on time series with strong trends

Forecasting time series with strong trends is a known limitation of tabular foundation models, as reported by the authors of `TabPFN-TS` [19]. In this section, we extend our zero-shot evaluation and investigate how `TS-ICL` performs on time series with controlled trends.

Setting. Following the protocol of TabPFN-TS, we generate deterministic series with linear, quadratic and exponential trends, using a context of $L = 2048$ points and a forecasting horizon of $H = 336$ points. Each setting corresponds to a different random seed, which controls the amplitude of the trend. Both models are used zero-shot, only the first L points are given as context, and the last H points are hidden for evaluation. Since trend amplitudes vary across seeds, we compute the MAE after min-max normalization by the context range. Table 22 reports the relative improvement of `TS-ICL` over TabPFN-TS in this normalized MAE, averaged over 5 seeds.

Table 22: Zero-shot forecasting on synthetic series with deterministic trends (context length 2048, horizon 336). Relative improvement of `TS-ICL` over `TabPFN-TS` in min-max normalized MAE (w.r.t. the context range), averaged over 5 seeds with different trend amplitudes. Wins: number of seeds on which `TS-ICL` achieves a lower MAE than `TabPFN-TS`.

Trend	Rel. improvement	Wins
Linear	+83.3%	5/5
Quadratic	+94.1%	5/5
Exponential	+63.7%	5/5

Results. `TS-ICL` outperforms `TabPFN-TS` in all 15 settings (3 trend types $\times$ 5 seeds), with relative improvements of 83.3%, 94.1% and 63.7% for linear, quadratic and exponential trends, respectively. Qualitatively, `TabPFN-TS` tends to saturate near the maximum value observed in the context window, whereas `TS-ICL` correctly extrapolates the underlying trend. This is consistent with the limitation acknowledged by the `TabPFN-TS` authors, namely that interpolation-based inference struggles to extrapolate beyond the observed target range [19].

We hypothesize that the improvement of `TS-ICL` stems from (i) its forecasting-oriented pre-training strategy, which explicitly exposes the model to extrapolation beyond the context range, and (ii) learned time-series representations that better preserve long-term trend information than Fourier-feature representations.

G Extended Discussions

G.1 Limitations

Although TS-ICL provides a unified framework for zero-shot forecasting and imputation, several limitations remain. We discuss them below, along with directions for future work.

G.1.1 Scalability and Efficiency

Inference efficiency. The main computational limitation of TS-ICL is its pointwise in-context regressor. Unlike recent patch-based time-series foundation models [3, 4], which reduce computation by processing temporal patches, TS-ICL operates on timestamp-level representations and applies causal self-attention over context and target tokens. This design is key to its flexibility, especially for imputation and irregularly sampled observations, but increases inference cost for long contexts, dense target grids, and large batches. Thus, while TS-ICL is substantially faster than tabular foundation model adaptations, it remains slower than highly optimized patch-based forecasters such as Chronos-2. Future work could improve efficiency through context caching, target batching, mixed precision, sparse or linear attention, or hybrid patch–pointwise architectures.

Training budget and model scale. Our experiments use a fixed-size version of TS-ICL. The results demonstrate strong zero-shot performance, especially for imputation, but we have not exhaustively scaled parameter count, data volume, context length, or training compute. The reported results should therefore be viewed as evidence for the effectiveness of the encoder–regressor formulation, rather than as a final estimate of the model family's best attainable performance. Larger models, longer training schedules, and broader pretraining mixtures may further improve accuracy, but could also affect the trade-off between robustness and inference cost.

G.1.2 Pretraining Prior and Generalization

Dependence on the pretraining prior. As other foundation models, TS-ICL zero-shot generalization depends on the diversity and representativeness of the pretraining distribution. Our prior, introduced in Section 4, combines real-world time series, synthetic generators, and a DAG-based covariate mechanism to expose the model to diverse temporal and cross-variable structures. However, real deployments may involve domains, frequencies, non-stationarities, interventions, noise processes, or missingness mechanisms that are underrepresented in this prior. Performance may therefore degrade under substantial domain shift. Extending the prior with additional domains, more realistic missingness, richer multivariate structure, and domain-specific dynamics is an important direction for future work.

G.1.3 Covariate Assumptions

Synthetic covariate generation. The DAG/SCM-based covariate generator creates diverse target–covariate dependencies during training. While useful for learning robust covariate conditioning, it should not be interpreted as causal discovery or as evidence that the model performs valid causal reasoning. The synthetic graph defines only a training prior over dependencies: it does not identify causal relationships in real data or justify counterfactual or interventional conclusions at test time. In practice, covariates may also be noisy, redundant, delayed, partially observed, spuriously correlated, or unavailable at the forecast horizon. Although TS-ICL can ingest such covariates, its predictions may degrade when their quality or availability differs from the training prior.

Multivariate applicability. Unlike certain patch-based foundation models such as Chronos-2, TS-ICL does not natively support multivariate (cross-variate) prediction. Instead, TS-ICL is built around the assumption that practical gains live primarily in correctly exploiting information available on *query* timesteps; hence a *univariate-plus-known-covariate* design. This design choice is empirically grounded: on fev-bench, Chronos-2 improves approximately ten times more by incorporating covariate support (13.5% on geometrically aggregated MASE) against cross-variate support (1.4% improvement) [3, 39]. [3] hypothesizes that strong univariate models with sufficiently long contexts *"may capture much of the same structure as a multivariate model"*. Nevertheless, settings requiring jointly coherent multivariate outputs, e.g. in the observability domain, remain a valuable direction for future work.

G.1.4 Evaluation and Deployment

Evaluation scope. Our evaluation covers large-scale zero-shot imputation and forecasting benchmarks, including univariate and covariate-aware settings. Nevertheless, these benchmarks do not cover all deployment scenarios. In particular, we focus on standard pointwise and block-missing imputation regimes, fixed benchmark protocols, and established forecasting horizons. Further evaluation is needed for extreme sparsity, very long-range forecasting, streaming inference, online adaptation, high-dimensional multivariate systems, event-based observations, and domain-specific operational constraints.

Robustness and deployment. `TS-ICL` is intended as a general-purpose decision-support model, not a substitute for domain-specific validation. Predictions may be unreliable under strong distribution shift, abnormal sensor behavior, adversarial or corrupted inputs, or unseen missingness patterns. In high-stakes domains such as healthcare, finance, energy systems, or critical infrastructure, erroneous forecasts or imputations could lead to harmful downstream decisions if used without oversight. Users should validate the model on representative data, compare it with task-specific baselines, inspect uncertainty estimates, and monitor performance after deployment.

G.2 Broader Impacts

`TS-ICL` is a general-purpose time-series foundation model for zero-shot forecasting and imputation. It may benefit applications where incomplete, irregular, or covariate-informed time series are used for planning, monitoring, or decision support, including energy, climate, healthcare, transportation, industry, and scientific measurement. By providing a unified framework, `TS-ICL` can reduce the need to train task-specific models for each dataset, potentially improving accessibility for practitioners with limited labeled data or machine-learning expertise. Its efficient inference relative to tabular foundation model adaptations may also lower computational costs.

However, `TS-ICL` is a general modeling tool, and its impact depends on the deployment context. Forecasts and imputations may be inaccurate or miscalibrated under severe distribution shifts, extreme missingness, weak covariates, or domains far from the pretraining prior. In high-stakes settings such as healthcare, finance, infrastructure, or safety-critical monitoring, such errors could lead to harmful decisions if used without expert oversight. Because time-series data can contain sensitive individual- or organization-level information, applications may also raise privacy, surveillance, fairness, and governance concerns.

We therefore view `TS-ICL` as a decision-support component, not a substitute for domain-specific validation or expert judgment. Before use in consequential settings, practitioners should evaluate performance, calibration, and robustness on representative data, compare against domain-specific baselines, and assess privacy, fairness, and security risks. Applications involving sensitive or regulated data should follow applicable legal and institutional requirements, and important decisions should include uncertainty estimates and human review.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately summarize the scope and contributions of the paper. TS-ICL is presented as a time-index foundation model for time series that unifies zero-shot forecasting and imputation through an encoder–regressor in-context learning architecture, with native support for irregular timestamps and optional covariates. These claims are directly supported by the model formulation and architecture described in Section 3, which casts forecasting and imputation as timestamp-aligned conditional inference problems, and by the structured synthetic data prior introduced in Section 4, which is designed to expose the model to diverse target–covariate dependency structures. The empirical claims are also appropriately scoped: the paper reports state-of-the-art or competitive zero-shot imputation results, competitive forecasting performance against time-series foundation model baselines, and robustness under missing observations, as evaluated in Section 5. Additional architectural details are provided in Appendix A, training and prior-generation details in Appendix B, ablation studies in Appendix C, and extended imputation and forecasting evaluations in Appendices E and F. Overall, the claims in the abstract and introduction are consistent with the methods, experiments, and limitations presented in the paper, and do not substantially exceed the demonstrated empirical scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: The paper discusses limitations in the conclusion and in a dedicated section in Appendix G.1. The conclusion notes that, although `TS-ICL` is faster than tabular foundation model adaptations, it remains slower at inference than optimized patch-based forecasters such as `Chronos2`, mainly due to the pointwise in-context regressor. Appendix G.1 broadens this discussion to inference scalability, training budget and model scale, dependence on the pretraining prior, assumptions behind the synthetic DAG/SCM covariate generator, evaluation scope, and robustness under distribution shift. It also outlines mitigation directions such as caching, automatic mixed precision, sparse or linear attention, broader pretraining priors, and additional domain-specific validation.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.

- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not introduce formal theoretical results requiring proofs. The mathematical content is used to define the conditional forecasting/imputation problem, the TS-ICL architecture, the data-generation prior, and the evaluation metrics.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: The paper provides the information needed to reproduce the main experimental results supporting its claims. The model architecture is described in Section 3 in the main text and Appendix B, with the corresponding architectural hyperparameters and implementation details reported in Appendix B.2.3. The training prior is specified in Appendix B.1.2, including the real and synthetic pretraining sources, the DAG/SCM-based covariate generator, and the full data prior sampling procedure. The experimental protocols are also described for both main tasks: zero-shot imputation on fm-impute-bench and zero-shot forecasting on fev-bench, including the evaluation settings, masking scenarios, baselines, metrics, and inference-time reporting. Additional ablations and extended experiments are provided in the appendices. The paper also includes a GitHub repository, providing an additional path for reproducing or verifying the reported results.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes].

   Justification: The paper provides open access to the code through a Github repository at https://github.com/EDF-Lab/ts-icl. The repository contains the implementation of TS-ICL, configuration files, environment specifications, and scripts to reproduce the main experimental results reported in the paper. It also provides instructions for accessing and preparing the data used in the experiments, including the public benchmark datasets used for zero-shot imputation and forecasting, as well as the procedure for generating the synthetic components of the training prior. The supplemental material further documents the architecture, training prior, hyperparameters, evaluation metrics, benchmark protocols, and experimental settings, so that the repository and appendix together provide a faithful path to reproducing the main claims of the paper.

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.

- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental settings are described in Sections 5.1 and 5.2. These sections specify the imputation and forecasting benchmarks, the univariate and known-covariate evaluation settings, the number of tasks and windows, the masking regimes used for imputation, the forecasting protocols, the baselines, and the evaluation metrics. Training and implementation details are further provided in Appendix B, including the pretraining corpus, the DAG-based covariate generator, the dynamic construction of imputation and forecasting training tasks, the architecture hyperparameters, the optimizer, learning-rate schedule, hardware, and batch size. Additional ablations in Appendix C document the effect of architectural choices and model scaling.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The main experimental claims reported in Section 5 are supported by both aggregate performance summaries and task-level robustness analyses. Imputation results are reported over 132 univariate tasks with approximately 1.3M windows and 24 known-covariate tasks with approximately 1K windows, while forecasting results are reported over 100 tasks with approximately 235K windows. In addition to averaged NMAE/MASE and CRPS scores, the paper reports pairwise win-rate matrices for the main imputation and forecasting comparisons; each entry gives the fraction of tasks on which one method outperforms another according to CRPS. Appendix E and Appendix F further reports detailed aggregated tables with standard deviations across tasks: mean $\pm$ standard deviation for imputation metrics, and geometric mean $\pm$ geometric standard deviation for forecasting metrics. Together, these results provide appropriate information about variability and consistency across datasets, missingness regimes, covariate settings, and forecasting tasks, rather than relying only on single aggregate scores. The evaluation metrics used for these comparisons are explicitly defined in Appendix D.

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes].

   Justification: The paper reports the compute resources required to reproduce the experiments. Appendix B, and in particular the paragraph on optimization hyperparameters, specifies the training hardware, namely 4 Nvidia H100 GPUs with 92GB memory, together with the global batch size, gradient accumulation strategy, optimizer, learning-rate schedule, and total training budget. The appendix also reports the compute budget used for the ablation studies, where all ablation models are trained for a fixed budget of 100k steps on H100 GPUs to ensure fair comparison. In addition, the main experimental sections report inference-time measurements on H100 GPUs for both imputation and forecasting, including per-window median inference times for TS-ICL and all baselines. Together, these details specify the compute workers, memory requirements, training budget, and evaluation-time costs needed to reproduce the reported experiments.

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research evaluates a time-series foundation model using existing benchmark datasets, synthetic data generators, and standard zero-shot forecasting/imputation protocols. The paper does not involve human-subject experiments, sensitive data collection, or high-risk generative content, and no deviation from the NeurIPS Code of Ethics is apparent.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes].

Justification: The paper includes a dedicated broader impact discussion in Appendix G.2. This section discusses potential positive impacts of TS-ICL, including improved forecasting and imputation for incomplete or irregular time series in domains such as energy, climate, healthcare, transportation, industrial monitoring, and scientific measurement, as well as improved accessibility through zero-shot inference and reduced task-specific training. It also discusses potential negative impacts, including incorrect or miscalibrated predictions in high-stakes settings, risks under distribution shift or severe missingness, privacy concerns for sensitive time-series data, and possible use in surveillance or monitoring contexts. The section further recommends mitigation strategies such as domain-specific validation, uncertainty evaluation, human oversight, privacy and governance safeguards, and careful assessment before deployment in consequential applications.

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release a high-risk generative model, scraped dataset, or other asset with an obvious high risk of misuse. TS-ICL is a time-series forecasting and imputation model evaluated on existing benchmarks and synthetic priors.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes].

    Justification: The paper uses existing datasets, benchmark suites, baseline models, and software packages, and credits their creators through the corresponding citations. The pretraining corpus, evaluation benchmarks, and baselines are documented in the main paper and further detailed in Appendix E and F, including LOTSA, Chronos training data, TempoPFN synthetic generators, fm-impute-bench, fev-bench, TIME, Chronos-2, TiRex, TOTO, TabPFNv2.5-TS, TabICLv2-TS, and local baselines. The supplemental material and Github repository at `https://github.com/EDF-Lab/ts-icl` include an asset listing the relevant datasets, codebases, models, versions or access URLs when available, licenses, and terms of use. We use all existing assets in accordance with their stated licenses and terms, and we do not relicense or redistribute third-party assets beyond what is permitted. For assets released with this work, the repository provides the corresponding license and usage terms.

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes].

    Justification: The paper introduces new assets associated with TS-ICL, including the model implementation, training and evaluation code, configuration files, and the DAG/SCM-based synthetic covariate-generation procedure. These assets are documented in the paper and appendices, with Appendix B describing the training prior, synthetic data-generation process, architecture hyperparameters, and implementation details. The assets are provided through the Githube repository at `https://github.com/EDF-Lab/ts-icl`, which includes documentation, usage instructions, environment specifications, experiment scripts, and licensing information. The paper does not introduce a new dataset collected from human participants; the released assets rely on existing public datasets and synthetic data-generation procedures, so consent collection for newly collected personal data is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing experiments or research with human subjects. Participant instructions, screenshots, and compensation details are therefore not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or human-subject research. IRB approval or equivalent review is therefore not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method does not use LLMs as an important, original, or non-standard component. TS-ICL uses transformer-based in-context learning for time-series regression, but no language model is part of the model architecture, training prior, or experimental protocol.

    Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.