LUMA: ULTRALIGHT LOW-RANK FORECASTING FOR PERIODIC TIME SERIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-term Time Series Forecasting (LTSF) faces a growing dichotomy between parameter-heavy foundation models and lightweight linear models. While recent trends favor massive scale, in this paper, we investigate whether the spectral compactness of periodic signals can motivate efficient low-rank forecasting. By combining period-folding with a rank-constrained operator, we reduce parameter growth through a compact cross-period representation. We introduce **LUMA** (**L**ow-rank **U**ltralight **M**atrix **A**pproximation), a minimalist architecture designed to exploit this efficiency. LUMA couples period-folding with a rank-constrained operator to 1) preserve within-period positions in a cross-period representation and 2) forecast their evolution through shared low-rank factors. We further prove that for a K-harmonic signal, the period-folded representation has rank at most $2K$, providing a structural motivation for rank-constrained forecasting. Extensive experiments on seven benchmarks demonstrate that LUMA achieves the best MSE on four datasets (ETTh1, ETTh2, ETTm2, Weather) while remaining competitive on the rest. In the Electricity profiling configuration, LUMA uses only **635 parameters**, approximately $13{,}000\times$ fewer than PatchTST and $36\times$ fewer than TimeKAN. Relative to SparseTSF, this configuration reduces parameters by approximately **36%** and MACs by approximately **18%**, with a reported CPU inference latency of **1.3 ms**. These results support low-rank cross-period modeling as an effective approach to accurate forecasting with a small parameter budget. Code is available at `https://anonymous.4open.science/r/luma-iclr-F2C8`.

1 INTRODUCTION

Long-term Time Series Forecasting (LTSF) is a critical component of modern infrastructure, underpinning applications ranging from renewable energy stabilization to real-time logistics optimization. As forecasting horizons extend, the central challenge becomes modeling long-range dependencies without incurring prohibitive computational and memory costs.

In recent years, this problem has been dominated by deep Transformer-based architectures (Zhou et al., 2021; Nie et al., 2023), which leverage global self-attention to capture temporal dependencies. While effective, these models suffer from quadratic complexity and substantial memory requirements, limiting their applicability in resource-constrained settings. This limitation is particularly pronounced in **Edge AI** scenarios, such as on-device forecasting in smart meters or IoT sensors, where GPUs are unavailable, and memory is limited to a few megabytes. These constraints have motivated a shift toward simpler and more efficient models. Notably, linear approaches such as DLinear (Zeng et al., 2023) have shown that basic trend decomposition can rival or even outperform more complex deep architectures. However, dense linear models still exhibit parameter growth that scales with the forecasting horizon, which quickly becomes impractical for long-term prediction.

To address this limitation, recent work has introduced structured constraints to improve efficiency, most prominently through sparsity (illustrated in Figure 1). Methods such as SparseTSF (Lin et al., 2024) reduce forecasting complexity through shared cross-period prediction and local convolutional aggregation. This motivates a complementary question: can the cross-period forecasting map itself be represented more compactly through a low-rank factorization?

In this work, we investigate a complementary structural prior: **spectral sparsity**. From a signal processing perspective, smooth and periodic signals tend to be compact in the spectral domain.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

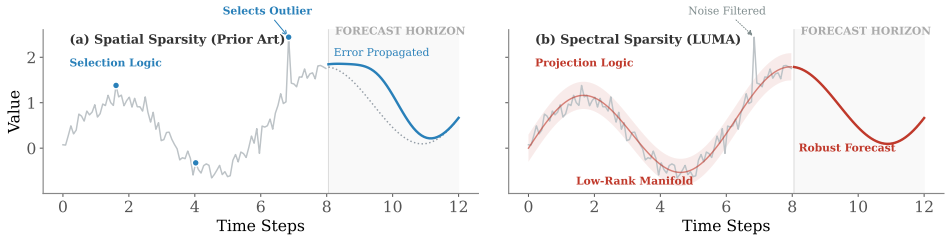

Figure 1: **The Core Intuition. (a) Temporal Connections:** A schematic illustration of dependencies between observations. **(b) Low-Rank Representation:** A schematic illustration of representing temporal variation through a compact set of components. The illustration motivates the prior rather than establishing a denoising guarantee.

Although such signals may not be sparse in the time domain, they can be efficiently represented using a small number of spectral components. This observation motivates investigating a **low-rank operator** for forecasting periodic signals. Here, spectral sparsity refers to the singular value spectrum of the forecasting matrix, while harmonic compactness provides motivation for this architectural prior.

Building on this perspective, we introduce **LUMA** (**L**ow-rank **U**ltralight **M**atrix **A**pproximation), a novel architecture that combines period folding with rank-constrained matrix factorization (Figure 1). Rather than selecting a subset of "relevant" time points, LUMA learns a compact basis that projects historical observations into a latent trend space. This formulation enables the model to capture continuous global patterns while maintaining a drastically reduced parameter count compared to both dense and spatially sparse approaches.

Our contributions are summarized as follows:

- **A Low-Rank Structural Perspective:** We investigate low-rank cross-period forecasting as an inductive bias for periodic time series. We establish a rank bound for folded harmonic signals, providing a structural motivation for compact forecasting operators.
- **LUMA Architecture:** We propose an ultralight forecasting model based on rank-constrained operator factorization. By leveraging period-normalized representations, the model reduces parameter complexity through factorization over the folded sequence dimensions.
- **Efficiency and Performance:** We conduct extensive experiments on seven benchmark datasets, demonstrating that LUMA matches or surpasses the accuracy of Transformer-based models (e.g., PatchTST) while significantly improving efficiency. In the Electricity profiling configuration, LUMA reduces parameter count by **36%** and MACs by approximately **18%** compared to **SparseTSF**, achieving CPU inference latency of **1.3 ms** versus **2.6 ms** for SparseTSF.

2 Related Work

Deep and Foundation Models. Following the success of Transformers in NLP, the time series domain has seen a proliferation of deep architectures. Early works like **Informer** (Zhou et al., 2021) and **Autoformer** (Wu et al., 2021) introduced sparse attention mechanisms to mitigate quadratic complexity. More recently, **PatchTST** (Nie et al., 2023) and **iTransformer** (Liu et al., 2024) achieved state-of-the-art results by inverting attention dimensions or utilizing patch-based tokenization. Concurrently, the field is moving toward Large Time Series Models and foundation models aimed at zero-shot generalization (Jin et al., 2024). Although these approaches offer powerful representations, they incur massive computational overhead and memory footprints, rendering them impractical for edge deployment or real-time applications where resources are strictly capped.

Linear and MLP-based Efficiency. Challenging the necessity of heavy architectures, **DLinear** (Zeng et al., 2023) demonstrated that simple linear mappings combined with trend-seasonal decomposition could outperform complex Transformers. This sparked a wave of lightweight architectures like **TiDE** (Das et al., 2023) and **FITS** (Xu et al., 2024). Most recently, the field has explored alternative

mathematical foundations, such as **TimeKAN** (Huang et al., 2025), which combines frequency decomposition with multi-order KAN representation learning based on Chebyshev polynomials. LUMA instead uses a factorized linear map for cross-period forecasting. **LUMA** addresses this by retaining the speed of linear operators, while achieving the parameter efficiency of KANs through low-rank spectral factorization.

Structural Priors and Sparsity. To address parameter redundancy, recent research has explored imposing structural constraints on the forecasting operator. Approaches like SparseTSF (Lin et al., 2024) use shared cross-period prediction with local convolutional aggregation. LUMA investigates a further constraint on the cross-period map through an explicit low-rank factorization.

LUMA distinguishes itself through low-rank cross-period factorization. The forecasting map can contain dense entries while being represented through a small number of learned components. This reduces its parameter count when the selected rank is sufficiently small relative to the folded input and output dimensions.

3 METHODOLOGY

Our approach, LUMA (illustrated in Figure 3.3), is based on a central hypothesis: *a compact cross-period operator can retain useful predictive structure in periodic time series*. We build upon the efficient pre-processing paradigms of recent linear models and impose an explicit low-rank prior on the forecasting map.

3.1 PRELIMINARIES

Long-term Time Series Forecasting. The task of LTSF involves predicting future values over an extended horizon using previously observed multivariate time series data. It is formalized as predicting $\mathbf{X}_{t+1:t+H} = f(\mathbf{X}_{t-L+1:t})$, where the input $\mathbf{X}_{t-L+1:t} \in \mathbb{R}^{L\times C}$ and the forecast $\mathbf{X}_{t+1:t+H} \in \mathbb{R}^{H\times C}$. Here, L represents the look-back window, C the number of variates (channels) and H the forecast horizon. The primary challenge lies in designing models that maintain computational efficiency and accuracy as H scales.

Channel Independent Strategy. Following recent advancements (Nie et al., 2023; Zeng et al., 2023), we adopt a Channel Independent (CI) strategy. We learn a single univariate model $f : \mathbb{R}^L \to \mathbb{R}^H$ with weights shared across all C channels. This reduces parameter complexity through weight sharing across variates. Each channel retains access to its own temporal history, while the model does not explicitly represent interactions between channels.

3.2 SIGNAL PREPARATION AND CROSS-PERIOD FOLDING

To enable ultralight modeling, we organize the input into a compact cross-period representation while retaining its within-period structure. We adopt a two-stage pre-processing strategy:

Stage 1: Intra-Period Processing. Local temporal dependencies provide useful context for subsequent forecasting. We first subtract the temporal mean $\boldsymbol{\mu} = L^{-1}\sum_{t=0}^{L-1}\mathbf{X}[t,:]$ from each channel, obtaining $\mathbf{X}_0$. We then apply a shared 1D convolution with a residual connection: $\mathbf{X}_{\text{smooth}} = \text{Conv1D}(\mathbf{X}_0) + \mathbf{X}_0$. This learnable operation prepares the signal for cross-period prediction.

Stage 2: Cross-Period Folding. The processed series $\mathbf{X}_{\text{smooth}}$ is then reorganized by a folding operator $\mathcal{D}_P$, where P denotes the folding interval. We right-pad the series with zeros to length $L'P$, obtaining $\widetilde{\mathbf{X}}_{\text{smooth}}$, where $L' = \lceil L/P \rceil$.

$$\mathbf{X}_{\text{ds}} = \mathcal{D}_P(\widetilde{\mathbf{X}}_{\text{smooth}}) \in \mathbb{R}^{L'\times(PC)}, \qquad \mathbf{X}_{\text{ds}}[\tau, pC + c] = \widetilde{\mathbf{X}}_{\text{smooth}}[\tau P + p, c]. \tag{1}$$

Here, τ, p, and c index cycles, within-period positions, and channels, respectively. Folding preserves the processed observations and arranges each within-period position across successive cycles. This representation allows a shared low-rank operator to act along the cycle dimension.

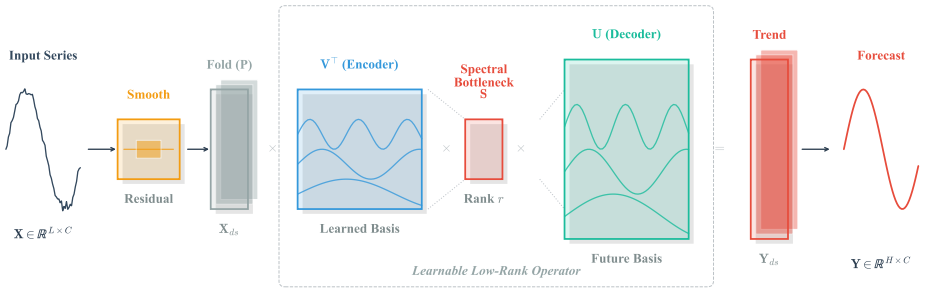

Figure 2: **The LUMA Architecture. (Left)** The input is mean-centered, processed by a residual convolution, and folded while preserving within-period positions. **(Center)** A shared low-rank operator ($\mathbf{U}\,\mathrm{diag}(\mathbf{s})\mathbf{V}^\top$) predicts across cycles. **(Right)** The forecast is unfolded, cropped to horizon H, and shifted by the input mean. The convolution and low-rank factors are trainable.

3.3 The Low-Rank Inductive Bias

The central design question for efficient forecasting is:

RQ: *what structural prior should be applied to the forecasting operator?*

Existing efficient approaches constrain temporal connections to reduce forecasting cost. We investigate a complementary constraint on the cross-period map: restricting its rank while allowing dense interactions along the cycle dimension.
Our structural prior is **spectral sparsity**, understood here as a constraint on the singular value spectrum of the cross-period operator. A low-rank matrix can contain dense entries while expressing its mapping through a small number of learned components.

Our Central Hypothesis. We posit that a low-rank cross-period operator can retain useful predictive structure for periodic time series. Learning a compact set of temporal components may therefore provide an effective trade-off between forecasting accuracy and parameter count. We examine this hypothesis through theoretical analysis and empirical evaluation. Training-split measurements of spectral concentration and period stability are reported in Appendix B.7 to characterize these properties across the evaluated benchmarks.

3.4 The LUMA Architecture

The LUMA architecture is a direct embodiment of the proposed spectral sparsity prior. We formulate the core forecasting operation as a linear mapping along the cycle dimension:

$$\mathbf{Y}_{ds} = \mathbf{W}_r \mathbf{X}_{ds}. \tag{2}$$

Here, $\mathbf{X}_{ds} \in \mathbb{R}^{L' \times (PC)}$ is the folded history and $\mathbf{Y}_{ds} \in \mathbb{R}^{H' \times (PC)}$ is the folded forecast. The same operator is shared across within-period positions and channels. Instead of learning a dense weight matrix $\mathbf{W} \in \mathbb{R}^{H' \times L'}$, which would be parameter-heavy and prone to overfitting, we model it via an explicit rank-r factorization, structurally analogous to a truncated Singular Value Decomposition (SVD):

$$\mathbf{W}_r = \mathbf{U} \cdot \mathrm{diag}(\mathbf{s}) \cdot \mathbf{V}^\top \tag{3}$$

where $\mathbf{U} \in \mathbb{R}^{H' \times r}$, $\mathbf{s} \in \mathbb{R}^r$, and $\mathbf{V} \in \mathbb{R}^{L' \times r}$. Here, $H' = \lceil H/P \rceil$ and $L' = \lceil L/P \rceil$ represent the period-normalized dimensions. The effective rank is $r = \max(1, \min(r_{\text{requested}}, L', H'))$. The columns of $\mathbf{V}$ and $\mathbf{U}$ are learned temporal components, while $\mathbf{s}$ contains learned scaling factors. The factors are not constrained to be orthogonal, and $\mathbf{s}$ is not constrained to contain singular values. The forecast for each channel is thus computed via:

$$\mathbf{Y}_{ds} = \mathbf{U} \cdot \mathrm{diag}(\mathbf{s}) \cdot (\mathbf{V}^\top \mathbf{X}_{ds}). \tag{4}$$

Finally, we invert the folding arrangement, retain the first H time steps, and restore the input mean:

$$\widehat{\mathbf{Y}} = \mathrm{Crop}_H\left(\mathcal{D}_P^{-1}(\mathbf{Y}_{ds})\right) + \boldsymbol{\mu}. \tag{5}$$

Here, Crop_H retains the first H time steps of the unfolded forecast, discarding any extra positions introduced by rounding the folded horizon upward.

Relationship to PCA and SVD. While LUMA utilizes a factorization structure $(\mathbf{U}, \mathbf{s}, \mathbf{V})$ similar to SVD, there is a fundamental distinction. SVD is an *unsupervised* method that minimizes reconstruction error of the *input*. High-energy noise can influence the components selected by a reconstruction objective. In contrast, LUMA is a *supervised* forecasting operator. The basis vectors are learned via backpropagation to minimize *prediction error* $(Y - \hat{Y})$. This objective allows LUMA to learn components according to their predictive usefulness rather than their contribution to input reconstruction. Whether this improves robustness is assessed empirically. Pseudocode for the full forward pass is given in Algorithm 1 (Appendix B.8).

3.5 Hyperparameter Selection

LUMA uses two main hyperparameters: the rank r, which controls the capacity of the cross-period operator, and the folding interval P, which determines how observations are arranged into cycles. Both are configured per dataset and forecast horizon.

To characterize candidate periods, we provide an offline FFT-based diagnostic using only the training split. For each eligible window, we average the raw channels to obtain $\mathbf{x}$ and compute $S[k] = |\,\mathrm{rFFT}(\mathbf{x})[k]|^2$, excluding $k = 0$. We map bins to periods using $P_k = \mathrm{round}(L/k)$ and select the highest-power bin satisfying $2 \le P_k \le \min(48, \lfloor L/2 \rfloor)$. The diagnostic estimate $\hat{P}$ is the median of the resulting window-level periods, taking the lower middle value for an even number of windows. This estimate characterizes periodicity; the configured P remains fixed during training and inference. Implementation details and sensitivity analyses are provided in Appendices B.5 and C.

3.6 Theoretical Analysis

We provide a rigorous theoretical foundation for our spectral prior, formalizing its efficiency advantages and justifying its suitability for periodic time series. Detailed proofs are provided in Appendix A.

> ***Theorem 3.1: Parameter Complexity*** *LUMA's architecture has $N_{params} = r(L' + H' + 1) + k$ trainable parameters, where $L' = \lceil L/P \rceil$, $H' = \lceil H/P \rceil$, $k = 1 + 2\lfloor P/2 \rfloor$, and r is the effective rank.*

Implication. LUMA reduces the parameter count of the cross-period map from $L'H'$ to $r(L' + H' + 1)$. For a sufficiently small rank, this yields a compact forecasting operator.

> ***Theorem 3.2: Spectral Compactness of Periodic Signals*** *Let $\mathbf{x} \in \mathbb{R}^L$ be an exact superposition of at most K harmonic components, and suppose that P divides L. Its folded representation $\mathcal{D}_P(\mathbf{x}) \in \mathbb{R}^{(L/P)\times P}$ has rank at most $\min(2K, L/P, P)$.*

Significance. The theorem establishes low-rank structure for complete folds of harmonic signals. The noisy extension in Appendix A bounds the remaining singular values by the folded residual norm. These results motivate the architectural prior without establishing the rank of a shared forecasting operator.

> ***Proposition 3.1: Rank Sufficiency for Empirical Forecasting*** *Let $\mathbb{X} \in \mathbb{R}^{L'\times n}$ and $\mathbb{Y} \in \mathbb{R}^{H'\times n}$ collect complete folded input–target pairs across windows, within-period positions, and channels. The minimum-Frobenius-norm solution of $\min_{\mathbf{W}} \|\mathbb{Y} - \mathbf{W}\mathbb{X}\|_F^2$ is $\mathbf{W}^* = \mathbb{Y}\mathbb{X}^\dagger$ and satisfies $\mathrm{rank}(\mathbf{W}^*) \le \min(H', \mathrm{rank}(\mathbb{X}))$. Consequently, if $\mathrm{rank}(\mathbb{X}) \le 2K$, an empirical least-squares optimum exists with rank at most $2K$.*

Proof. The expression follows from the minimum-norm least-squares solution. The rank bound follows from $\mathrm{rank}(AB) \le \min(\mathrm{rank}(A), \mathrm{rank}(B))$ and $\mathrm{rank}(\mathbb{X}^\dagger) = \mathrm{rank}(\mathbb{X})$.

Table 1: **Multivariate Long-term Forecasting Results.** Results are averaged across horizons $H \in \{96, 192, 336, 720\}$. Lower is better. The lowest and second-lowest distinct values at the reported three-decimal precision are marked in **red** and **blue**, respectively; ties share the same color. Per-horizon results for the methods included in the detailed table are provided in Appendix H.

Methods	ETTh1		ETTh2		ETTm1		ETTm2		Weather		Electricity		Traffic	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
FEDformer (2022)	0.523	0.524	0.429	0.470	0.438	0.466	0.409	0.462	0.309	0.360	0.235	0.348	0.638	0.400
TimesNet (2023)	0.458	0.481	0.414	0.427	0.400	0.417	0.291	0.332	0.254	0.294	0.193	0.304	0.620	0.608
DLinear (2023)	0.444	0.467	0.409	0.424	0.366	0.395	0.258	0.332	**0.242**	0.315	0.169	0.267	0.435	0.302
TimeMixer (2024)	0.454	0.474	0.379	0.426	0.381	0.414	0.283	0.347	0.248	0.281	0.171	0.274	0.419	0.300
iTransformer (2024)	0.450	0.457	0.390	0.422	0.370	0.401	0.278	0.338	0.243	0.278	0.175	0.277	0.417	0.292
FITS (2024)	0.417	0.430	0.345	**0.385**	0.365	0.388	0.255	**0.315**	0.244	0.286	0.172	0.266	0.426	0.291
PatchTST (2023)	0.424	0.438	**0.343**	**0.382**	**0.353**	**0.381**	0.256	0.319	0.265	0.285	**0.164**	**0.265**	**0.402**	0.290
SparseTSF (2024)	**0.406**	**0.428**	0.344	**0.382**	0.361	**0.382**	**0.254**	0.317	0.244	0.286	0.168	**0.264**	0.414	**0.280**
TimeKAN (2025)	0.412	0.430	0.348	0.402	0.368	0.395	0.260	0.325	**0.242**	**0.272**	0.193	0.269	0.420	0.295
LUMA (Ours)	**0.396**	**0.418**	**0.336**	**0.382**	**0.359**	**0.381**	**0.251**	**0.314**	**0.238**	**0.275**	**0.165**	0.267	**0.412**	**0.278**

Connection to LUMA's Factorization. The minimum-norm solution admits a thin SVD $\mathbf{W}^* = \mathbf{U}_q \mathbf{\Sigma}_q \mathbf{V}_q^\top$, where $q = \mathrm{rank}(\mathbf{W}^*)$. Therefore, LUMA's parameterization $\mathbf{W}_r = \mathbf{U}\,\mathrm{diag}(\mathbf{s})\mathbf{V}^\top$ can represent this solution whenever $r \geq q$. This establishes representational sufficiency for the empirical least-squares optimum, without requiring the learned factors to be singular vectors.

The pooled-rank condition is an additional assumption: low rank of each individual folded window does not by itself imply the same rank bound for the pooled collection.

Remark A small rank budget may limit forecasting accuracy when the predictive mapping cannot be accurately approximated at that rank. High input rank alone does not establish this limitation. Appendix G reports empirical performance on Exchange Rate and ILI, illustrating dataset- and horizon-dependent limitations of the evaluated configurations.

Regularization Perspective. Restricting the rank reduces the capacity of the cross-period map. We assess the resulting accuracy and robustness trade-offs through the rank ablation and noise experiments.

4 Experimental Setup

Datasets. We use 7 standard benchmarks: ETTh1, ETTh2, ETTm1, ETTm2, Weather, Electricity, and Traffic. Together they span energy, climate, and transportation domains with diverse periodicity and dimensionality (7–862 channels). We follow the protocols of Wu et al. (2021); Nie et al. (2023). **Baselines.** We compare against 9 methods across three paradigms: *Transformers* (TimeMixer (Wang et al., 2024), iTransformer (Liu et al., 2024), PatchTST (Nie et al., 2023), FEDformer (Zhou et al., 2022)); *Linear/MLP* (DLinear (Zeng et al., 2023), TimesNet (Wu et al., 2023)); and *Efficient* (SparseTSF (Lin et al., 2024), FITS (Xu et al., 2024), TimeKAN (Huang et al., 2025)). **Protocol.** All baselines use their reported optimal lookback $L \in \{96, 192, 336, 720\}$. We evaluate at horizons $H \in \{96, 192, 336, 720\}$ via MSE and MAE. A strict $L = 720$ controlled comparison is in Appendix E. Full setup details in Appendix H.

Superiority over Sparse and Efficient Models. As shown in Table 1, the direct comparison with SparseTSF substantiates our core hypothesis. LUMA achieves an average ETTm2 MSE of 0.251, compared with 0.254 for SparseTSF and 0.260 for TimeKAN in Table 1. These results support the effectiveness of the low-rank cross-period map within the evaluated configurations. Additional comparisons with CycleNet, TQNet, and FreqCycle are provided in Appendix F, where LUMA achieves the lowest average MSE on six of seven datasets among these methods.

4.1 Efficiency Analysis

We provide a comprehensive comparison of static and runtime metrics on the **Electricity** dataset in Table 2.

For one multivariate input, the dominant MAC count is

$$CLk + CPr(L' + H'),$$

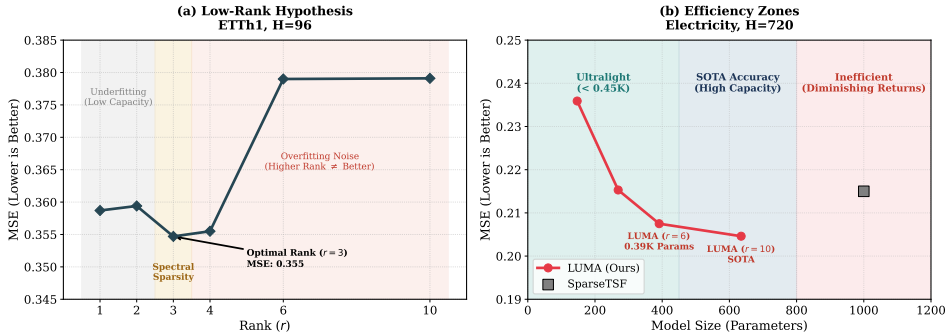

Figure 3: **Low-rank validation and efficiency. (Left)** Rank ablation shows that LUMA performs best at small rank, supporting spectral compactness. **(Right)** LUMA achieves a stronger accuracy-parameter trade-off than SparseTSF on Electricity.

Table 2: **Efficiency vs. Accuracy (Electricity).** Efficiency **wins are reported only for LUMA**, relative to SparseTSF (the strongest non-LUMA efficient baseline). Best Avg MSE in **Red**, second best in **Blue**. Moving from r=6 to r=10 improves Avg MSE by $\Delta = -0.002$ ($\approx$1.2%). Parameter and MAC counts use $L = H = 720$, $P = 24$, and $C = 321$; MSE is averaged across the four forecast horizons.

Model	Avg MSE	Params	MACs	Max Mem	Epoch Time	Infer Time
Heavy Transformers						
PatchTST (2023)	**0.164**	8.69M	14.17G	18034 MB	827 s	249.0 ms
iTransformer (2024)	0.175	5.47M	1.79G	828 MB	66 s	30.4 ms
TimeMixer (2024)	0.171	5.58M	30.30G	1376 MB	537 s	71.6 ms
Efficient Models						
DLinear (2023)	0.169	1.04M	333M	158 MB	41 s	3.2 ms
SparseTSF (2024)	0.168	1.00K	12.7M	125 MB	31 s	2.6 ms
TimeKAN (2025)	0.193	23.34K	456.5M	142 MB	113 s	14 ms
LUMA-Lite (r=6)	**0.167**	**0.39K**	**8.55M**	**90 MB**	**18 s**	**1.1 ms**
Δ vs SparseTSF	–	↓61%	↓32.7%	↓28.0%	↓41.9%	↓57.7%
LUMA-Std (r=10)	**0.165**	0.64K	10.40M	94 MB	19 s	1.3 ms
Δ vs SparseTSF	–	↓36%	↓18.1%	↓24.8%	↓38.7%	↓50.0%

including the residual convolution and both factorized projections. The additional CPr scalar multiplications for component scaling are not counted as MACs.

Comparison with TimeKAN. Recently, Kolmogorov-Arnold Networks (KANs) have been proposed as a parameter-efficient alternative to MLPs. While **TimeKAN** (Huang et al., 2025) reduces parameters compared to dense Transformers, our results indicate that this comes at a computational cost. In the reported configuration, TimeKAN requires 456.5M MACs, compared with approximately 10.40M for LUMA. **LUMA** outperforms TimeKAN by reducing parameters by orders of magnitude (0.64K vs. 23.34K) while achieving **over 10$\times$ faster inference speeds** (1.3 ms vs. 14 ms). These measurements demonstrate a computational advantage for LUMA in the reported configuration.

Speed and Scalability. LUMA is the fastest model in our benchmark. Inference takes just **1.1 ms** (LUMA-Lite) to **1.3 ms** (LUMA-Std) per batch on a standard CPU. This makes LUMA **2$\times$ faster** than the sparse linear baseline (SparseTSF, 2.6 ms) and orders of magnitude faster than Transformer-based SOTA like PatchTST (249 ms).

Overall Efficiency. As illustrated in Figure 3, LUMA offers a competitive trade-off between forecasting accuracy and parameter count., achieving high accuracy in the 'Ultralight' parameter zone where other models underfit. In the reported Electricity profiling configuration, LUMA achieves approximately a **99.9% parameter reduction** compared with DLinear (1.04M parameters versus 635). This small parameter footprint motivates deployment under tight resource budgets, while practical feasibility also depends on activation memory and runtime support.

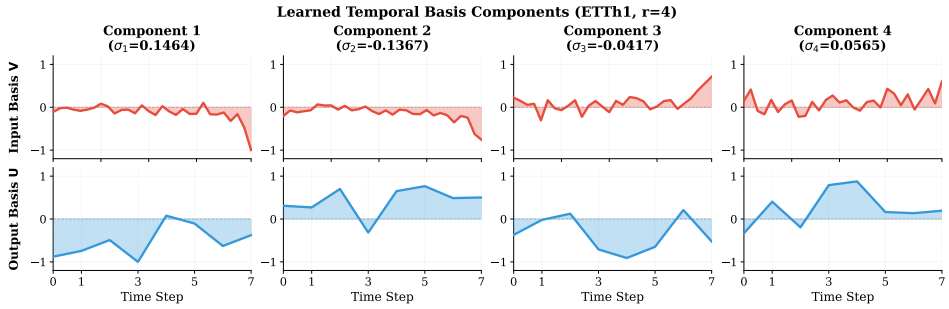

Figure 4: **Visualization of Learned Basis Components.** The top row shows the principal temporal patterns extracted from the history. The bottom row shows how these evolve into the future. The displayed components exhibit smooth and oscillatory patterns. They are learned forecasting factors and are not constrained to form a Fourier basis.

4.2 Mechanism Analysis and Ablation

To understand the source of LUMA's performance, we analyze its components through systematic ablation and feature visualization.

Impact of Low-Rank Constraint. We first investigate whether a strict low-rank bottleneck is necessary compared to a full-rank dense projection. The maximum rank of the cross-period map is $\min(L', H')$. Setting the effective rank to this value removes the rank restriction on the map, although the factorized parameterization remains different from a directly parameterized dense layer. For this ablation, we disable rank clamping and vary the factorization width r. The effective matrix rank remains bounded by $\min(L', H')$; larger widths increase the number of trainable parameters without increasing this rank bound. Our results indicate that the Low-Rank version outperforms the Full-Rank version by 4.2% in MSE on ETTh1 (Figure 3). The reported advantage of a small effective rank supports the low-rank prior in this configuration; it does not identify the rank of the underlying data-generating operator.

Furthermore, as shown in Figure 3, performance does not improve monotonically with the rank. Instead, we observe a clear optimum at a low rank ($r = 3$). Beyond this point, the reported error increases, indicating that additional capacity does not improve accuracy in this experiment. We visualize this learned operator structure and its singular value spectrum in Figure 5, confirming that LUMA naturally converges to a low-rank solution. For a complete rank sensitivity analysis across datasets, refer to Appendix C.

Impact of the Folding Interval. We examine how the choice of P affects forecasting accuracy by comparing the evaluated configurations with a fixed interval $P = 24$. The FFT diagnostic is described in Appendix B.6; the periods used in the reported experiments are listed in the experimental configurations.

Visualization of Learned Basis. To inspect the learned temporal representations, we visualize the basis components for the **ETTh1** dataset ($r = 4$) in Figure 4. The top row shows the Input Basis vectors ($\mathbf{V}$), which extract patterns from the history, while the bottom row shows the corresponding Output Basis vectors ($\mathbf{U}$). The displayed components illustrate different temporal patterns used by the learned forecasting map.

A striking observation is the contrast between the input and output basis: while the input vectors retain high-frequency fluctuations to identify the signal's phase, the output vectors project these into smooth, geometric trajectories. These visualizations illustrate the temporal patterns represented by the learned factors. Their connection to robustness is evaluated separately through the noise experiment.

Robustness to Noise. To evaluate sensitivity to corruption, we injected Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$ into the Electricity test data (Table 3). At $\sigma = 0.20$, LUMA's MSE increases from 0.137 to 0.244, while SparseTSF's increases from 0.138 to 0.342. This experiment demonstrates lower degradation for LUMA under the reported corruption setting.

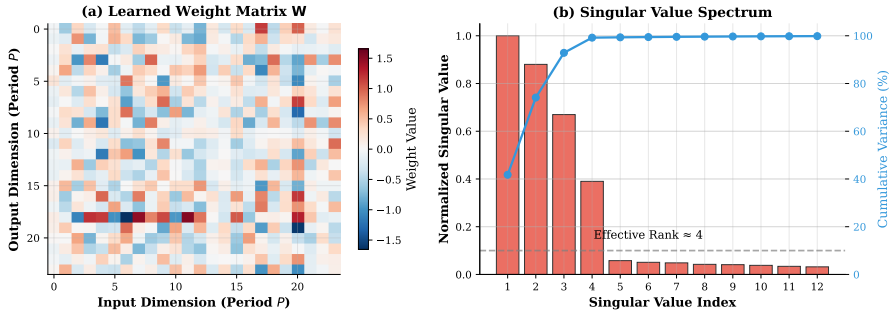

Figure 5: **Visualization of Learned Operator.** The heatmap (left) and singular value spectrum (right) visualize the learned cross-period map. Its rank is bounded by the imposed factorization rank.

Table 3: **Noise Robustness Analysis (MSE degradation).**

Method	$\sigma = 0$	$\sigma = 0.05$	$\sigma = 0.10$	$\sigma = 0.20$
SparseTSF	0.138	0.163	0.205	0.342 (+148%)
LUMA	**0.137**	**0.145**	**0.168**	**0.244 (+78%)**

4.3 Scalability to Ultra-Long Horizons

A critical bottleneck for LTSF is the quadratic or linear growth of parameters as the horizon H increases. To demonstrate the "Power of Minimalism", we extend the forecast horizon to $H \in \{1440, 2880\}$ (Ultra-Long Term) on the Electricity dataset.

Table 4: **Performance on Ultra-Long Horizons (Electricity).** Forecasting accuracy at $H = 1440$ and $H = 2880$. Parameter growth is stated with the input configuration fixed.

Method	$H = 1440$		$H = 2880$		Params Growth
	MSE	MAE	MSE	MAE	
DLinear	0.235	0.351	0.289	0.402	Linear ($O(H)$)
PatchTST	0.221	0.332	0.265	0.385	Linear ($O(H)$)
LUMA	**0.218**	**0.325**	**0.254**	**0.371**	**Linear in H'**

As shown in Table 4, LUMA outperforms PatchTST by substantial margins at extreme horizons. Crucially, the parameter count of LUMA grows with the *folded* horizon $H' = H/P$. For $H = 2880$ and $P = 24$, the effective output dimension is only 120. This allows LUMA to model months of future data with a compact matrix, avoiding the "curse of dimensionality" that plagues dense linear models and the memory explosion that affects Transformers. Additional analysis on parameter explosion and memory footprint is detailed in Appendix D.

5 Conclusion

We introduced **LUMA**, an ultralight forecasting architecture that combines period folding with a rank-constrained cross-period operator. Our harmonic rank analysis motivates this structural prior, while experiments on seven benchmarks demonstrate competitive (ETTh1, ETTh2, ETTm2, Weather) accuracy, with the best average MSE on four datasets. In the Electricity profiling configuration, LUMA uses only 635 parameters, reducing parameters by approximately 36% and MACs by approximately 18% relative to SparseTSF, with a CPU inference latency of 1.3 ms. These results support low-rank cross-period modeling for efficient forecasting, while performance on weakly periodic data highlights the limits of this inductive bias.

DISCLOSURE OF AI USE

In this work, we used generative AI tools to aid or polish writing. We have not used generative AI tools for any other tasks with required disclosure, and the rest of the required disclosure tasks are not applicable to this work. We have reviewed all AI-assisted work. All AI-assisted textual corrections were reviewed by the authors to ensure that they preserved the intended meaning. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with TiDE: Time-series dense encoder. *Trans. Mach. Learn. Res.*, 2023, 2023. URL https://openreview.net/forum?id=pCbC3aQB5W.

Songtao Huang, Zhen Zhao, Can Li, and Lei Bai. TimeKAN: Kan-based frequency decomposition learning architecture for long-term time series forecasting. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=wTLc79YNbh.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-LLM: Time series forecasting by reprogramming large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=Unb5CVPtae.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Haojun Chen, and Junjie Yang. SparseTSF: Modeling long-term time series forecasting with *1k* parameters. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pages 30211–30226. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/lin24n.html.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=JePfAI8fah.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y. Zhang, and Jun Zhou. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=7oLshfEIC2.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 22419–22430, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/bcc0d400288793e8bdcd7c19a8ac0c2b-Abstract.html.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=ju_Uqw384Oq.

Zhijian Xu, Ailing Zeng, and Qiang Xu. FITS: modeling time series with 10k parameters. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=bWcnvZ3qMb.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press, 2023. ISBN 978-1-57735-880-0. doi: 10.1609/aaai.v37i9.26317. URL https://doi.org/10.1609/aaai.v37i9.26317.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 11106–11115. AAAI Press, 2021. doi: 10.1609/AAAI.V35I12.17325. URL https://doi.org/10.1609/aaai.v35i12.17325.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 27268–27286. PMLR, 2022. URL https://proceedings.mlr.press/v162/zhou22g.html.

A THEORETICAL PROOFS AND JUSTIFICATIONS

This appendix provides detailed proofs for the theorems presented in the main paper.

A.1 PROOF OF THEOREM 1 (PARAMETER COMPLEXITY)

> ***Theorem A.1*** *The LUMA architecture is parameterized by $N_{params} = r(L' + H' + 1) + \mathcal{O}(P)$ parameters.*

Proof. The total parameter count is the sum of the model's two learnable components. The first component is the low-rank factorization, which consists of the future basis matrix $\boldsymbol{U} \in \mathbb{R}^{H' \times r}$ ($H' \cdot r$ parameters), the past basis matrix $\boldsymbol{V} \in \mathbb{R}^{L' \times r}$ ($L' \cdot r$ parameters), and the scaling vector $\boldsymbol{s} \in \mathbb{R}^r$ (r parameters), for a total of $r(H' + L' + 1)$. The second component is the 1D convolutional layer for intra-period smoothing, which has a kernel size proportional to the period P and thus contributes $\mathcal{O}(P)$ parameters. The folding, inverse folding, and cropping operations are parameter-free. Therefore, the total parameter count is $N_{\text{params}} = r(L' + H' + 1) + \mathcal{O}(P)$. □

A.2 PROOF OF THEOREM 2 (THE LOW-RANK NATURE OF PERIODIC SIGNALS)

> ***Theorem A.2: L*** *t $\mathbf{x} \in \mathbb{R}^L$ be a time series that can be decomposed as $\mathbf{x} = \mathbf{x}_{periodic} + \mathbf{e}$, where $\mathbf{x}_{periodic}$ is a superposition of at most K dominant harmonic components, and $\mathbf{e}$ represents bounded noise. Let $\mathbf{X}_{ds} = \mathcal{D}_P(\mathbf{x}) \in \mathbb{R}^{L' \times P}$ be the cross-period representation. Then:*
>
> 1. *In the ideal noise-free case ($\mathbf{e} = \mathbf{0}$), the matrix $\mathbf{X}_{ds}$ has rank at most $2K$.*
> 2. *In the realistic noisy case, the singular values of $\mathbf{X}_{ds}$ beyond the $2K$-th are bounded by the spectral norm of the folded noise: $\sigma_i(\mathbf{X}_{ds}) \leq \|\mathcal{D}_P(\mathbf{e})\|_2$ for all $i > 2K$.*

Proof. We prove the theorem in two parts.

Part 1: The Ideal Noise-Free Case. Let $\mathbf{x}_{\text{periodic}}$ be a purely periodic signal composed of K harmonic components, which can be expressed as $x_{\text{periodic}}(t) = \sum_{k=1}^{K} [\alpha_k \cos(2\pi f_k t) + \beta_k \sin(2\pi f_k t)]$. The cross-period folding operator $\mathcal{D}_P$ reshapes this signal into a matrix $\mathbf{X}_{\text{periodic, ds}}[\tau, p] = x_{\text{periodic}}(\tau P + p)$. Substituting the harmonic representation and applying trigonometric identities reveals a factorization. We can define temporal basis vectors $\mathbf{u}_k[\tau] = \cos(2\pi f_k \tau P)$ and $\mathbf{v}_k[\tau] = \sin(2\pi f_k \tau P)$, and phasal basis vectors $\mathbf{a}_k[p] = \alpha_k \cos(2\pi f_k p) + \beta_k \sin(2\pi f_k p)$ and $\mathbf{b}_k[p] = \beta_k \cos(2\pi f_k p) - \alpha_k \sin(2\pi f_k p)$. With these definitions, the entire matrix can be expressed as a sum of $2K$ rank-1 outer products:

$$\mathbf{X}_{\text{periodic, ds}} = \sum_{k=1}^{K} \left(\mathbf{u}_k \mathbf{a}_k^\top + \mathbf{v}_k \mathbf{b}_k^\top \right).$$

By the subadditivity of matrix rank, $\text{rank}(\mathbf{X}_{\text{periodic, ds}}) \leq \sum_{k=1}^{K} (\text{rank}(\mathbf{u}_k \mathbf{a}_k^\top) + \text{rank}(\mathbf{v}_k \mathbf{b}_k^\top)) \leq 2K$. This completes the proof for the ideal case.

Part 2: The Realistic Case with Bounded Noise. Now consider the noisy signal $\mathbf{x} = \mathbf{x}_{\text{periodic}} + \mathbf{e}$. By linearity, $\mathbf{X}_{ds} = \mathcal{D}_P(\mathbf{x}) = \mathbf{X}_{\text{periodic, ds}} + \mathcal{D}_P(\mathbf{e})$. Let $\mathbf{E}_{ds} = \mathcal{D}_P(\mathbf{e})$. We employ Weyl's inequality for singular values, which states that for any matrices $\mathbf{A}, \mathbf{B}$ of the same size, $|\sigma_i(\mathbf{A} + \mathbf{B}) - \sigma_i(\mathbf{A})| \leq \|\mathbf{B}\|_2$. Let $\mathbf{A} = \mathbf{X}_{\text{periodic, ds}}$ and $\mathbf{B} = \mathbf{E}_{ds}$. For any singular value index i:

$$|\sigma_i(\mathbf{X}_{ds}) - \sigma_i(\mathbf{X}_{\text{periodic, ds}})| \leq \|\mathbf{E}_{ds}\|_2.$$

From Part 1, we know that for all $i > 2K$, $\sigma_i(\mathbf{X}_{\text{periodic, ds}}) = 0$. Substituting this into Weyl's inequality gives:

$$\sigma_i(\mathbf{X}_{ds}) = |\sigma_i(\mathbf{X}_{ds}) - 0| \leq \|\mathbf{E}_{ds}\|_2 \quad \text{for all } i > 2K.$$

This establishes that the tail singular values of the noisy representation are bounded by the spectral norm of the folded noise. □

A.3 Theoretical Comparison: Spectral vs. Spatial Sparsity

The key difference between our LUMA and methods like SparseTSF lies in the assumed structure of the forecasting operator.

Definition A.1: Sparsity Assumptions Let $\boldsymbol{W}^*$ be the optimal linear forecasting operator.

- **Spatial Sparsity** assumes $\boldsymbol{W}^*$ is sparse in its entries, i.e., $\|\boldsymbol{W}^*\|_0 \ll L'H'$. It posits that only a few specific past time points influence future points.
- **Spectral Sparsity** (Low-Rank) assumes that $\boldsymbol{W}^*$ has low rank or admits an accurate low-rank approximation. It posits that the dynamics are governed by a few dominant global patterns.

Complementarity of Structural Assumptions. A low-rank factorization can represent a dense matrix compactly when a small rank is sufficient. An entry-wise sparse representation can be preferable when only a small number of matrix entries are needed. These assumptions describe different forms of structure; neither universally dominates the other. LUMA evaluates the low-rank choice for the cross-period forecasting map.

Our empirical success suggests that for the standard long-term forecasting benchmarks, the spectral sparsity assumption is a more effective prior than the spatial sparsity assumption, leading to models that are both more accurate and significantly more compact.

B Implementation Details

B.1 Datasets

We use seven widely-adopted long-term forecasting benchmarks. **ETTh1**, **ETTh2**, **ETTm1**, and **ETTm2** (Electricity Transformer Temperature) record oil temperature and load measurements from electricity transformers in two regions of China at hourly (h) and 15-minute (m) granularity. **Electricity** contains hourly consumption from 321 clients. **Traffic** records hourly road occupancy from 862 sensors on the San Francisco Bay area freeways. **Weather** contains 21 meteorological indicators recorded every 10 minutes at the Max Planck Institute for Biogeochemistry. We follow the standard protocols used by (Wu et al., 2021; Nie et al., 2023). Dataset statistics are summarized in Table 5.

Table 5: Dataset statistics. All datasets are evaluated across horizons $H \in \{96, 192, 336, 720\}$.

	ETTh1	ETTh2	ETTm1	ETTm2	Weather	Electricity	Traffic
Variables	7	7	7	7	21	321	862
Length	17,420	17,420	69,680	69,680	52,696	26,304	17,544
Frequency	1h	1h	15m	15m	10m	1h	1h
Domain	Energy	Energy	Energy	Energy	Climate	Energy	Transport

B.2 Baselines

We compare LUMA against 9 representative methods spanning three paradigms.

Transformer-based. **TimeMixer** (Wang et al., 2024) performs decomposable multiscale mixing. **iTransformer** (Liu et al., 2024) inverts attention to operate on variates rather than time steps. **PatchTST** (Nie et al., 2023) applies channel-independent patching with a vanilla Transformer. **FEDformer** (Zhou et al., 2022) performs frequency-domain decomposition.

Linear / MLP-based. **DLinear** (Zeng et al., 2023) uses linear maps with trend-seasonal decomposition. **TimesNet** (Wu et al., 2023) reshapes the series into 2D tensors via period-folding and processes them with a CNN backbone.

Efficient. **SparseTSF** (Lin et al., 2024) uses spatial sparsity over time points and is LUMA's most direct competitor. **FITS** (Xu et al., 2024) performs frequency-domain interpolation. **TimeKAN** (Huang et al., 2025) combines frequency decomposition with multi-order Chebyshev KAN blocks.

For all baselines, we report numbers from their original papers when available, and use their official open-source implementations (Time-Series-Library[1]) otherwise, ensuring each model operates at its reported optimal configuration.

B.3 Evaluation Protocol

Each method is evaluated at four prediction horizons $H \in \{96, 192, 336, 720\}$. Reported metrics are Mean Squared Error (MSE) and Mean Absolute Error (MAE) on the test set, averaged across the four horizons unless stated otherwise. All LUMA results are averaged over 5 random seeds; standard deviations are reported alongside per-horizon results in Appendix H.

B.4 Hardware and Environment

All experiments were run on a server with an Intel Xeon Gold 6248R CPU and an NVIDIA A100 (40GB) GPU. For the efficiency benchmarks (Table 2), we explicitly disabled the GPU and ran inference on CPU to simulate edge-deployment scenarios. The code is implemented in PyTorch 2.1.

B.5 Hyperparameter Tuning

We use the following training settings:

- Rank: configured per dataset and horizon, with effective rank $r = \max(1, \min(r_{\text{requested}}, L', H'))$.
- Folding interval P: configured per dataset and horizon and held fixed during training and inference. An offline FFT-based period diagnostic is described in Appendix B.6.
- Initial learning rate: 0.02 for ETTh1, ETTm1, Weather, and Electricity; 0.03 for ETTh2, ETTm2, and Traffic.
- Batch size: 256 for the ETT datasets and Weather; 128 for Electricity and Traffic.

Optimizer: Adam without explicit weight decay. Training runs for at most 30 epochs with early stopping on validation MSE and patience 5.

B.6 FFT-Based Period Diagnostic

We provide an offline period diagnostic using only the training split. For each eligible training window, we average the raw channels to obtain $\mathbf{x}$ and compute the power spectrum $S[k] = |\,\mathrm{rFFT}(\mathbf{x})[k]|^2$, excluding the DC component ($k = 0$). Frequency bins map to candidate periods via $P_k = \mathrm{round}(L/k)$. We select the highest-power bin among candidates satisfying $2 \le P_k \le \min(48, \lfloor L/2 \rfloor)$. The diagnostic estimate $\hat{P}$ is the median of the window-level period estimates, taking the lower middle value when the number of windows is even. This estimate characterizes candidate periodicity; the forecasting interval P is specified separately for each experiment. Sensitivity to the folding interval is reported in Appendix C.

B.7 Quantifying Periodicity in the Benchmarks

To characterize the data properties motivating LUMA, we report training-split diagnostics of spectral concentration, dominant periodicity, temporal stability, and cross-channel agreement in Table 6.

Interpretation. The diagnostics reveal substantial variation across benchmarks. Electricity has a higher Top-5 energy ratio than Traffic (0.780 versus 0.557), although both have period stability of 1.00. Thus, a stable dominant period does not imply the same degree of spectral concentration. These measurements characterize the datasets rather than establish a monotonic ranking of forecasting methods or guarantee low-rank forecasting accuracy.

[1] https://github.com/thuml/Time-Series-Library

Table 6: **Training-split periodicity diagnostics.** Dominant-frequency and Top-5 energy ratios characterize spectral concentration. Diagnostic periods are expressed in samples and should be distinguished from the configured folding intervals used by LUMA.

Dataset	Dom.-freq.	Top-5	Eff. rank	Period	Stability	Agreement
Electricity	0.451	0.780	15.7	24	1.00	0.92
Traffic	0.250	0.557	59.8	24	1.00	1.00
ILI	0.210	0.427	33.0	52	—	1.00
Weather	0.223	0.410	198	144	0.70	0.67
Exchange	0.115	0.383	58.7	590	0.69	0.25
ETTm1	0.120	0.260	387	96	0.35	0.71
ETTm2	0.130	0.260	624	96	0.33	1.00
ETTh1	0.090	0.250	288	24	0.74	0.71
ETTh2	0.110	0.250	225	24	0.26	0.71

B.8 Training Algorithm

Algorithm 1 summarizes the LUMA forward pass during training. The procedure consists of input centering and residual processing, folding, three operations defining the low-rank operator (project, scale, reconstruct), and final unfolding with mean restoration. All learnable parameters reside in the low-rank factors $(\mathbf{U}, \mathbf{s}, \mathbf{V})$ and the smoothing convolution a total of $r(L' + H' + 1) + \mathcal{O}(P)$ parameters (Theorem 3.6).

Algorithm 1: LUMA Training Process

Input: Input series $\boldsymbol{X} \in \mathbb{R}^{L \times C}$, period P, rank r, horizon H.
Output: Forecast $\boldsymbol{Y} \in \mathbb{R}^{H \times C}$.
Data: Parameters: $\boldsymbol{U} \in \mathbb{R}^{H' \times r},\ \boldsymbol{V} \in \mathbb{R}^{L' \times r},\ \boldsymbol{s} \in \mathbb{R}^{r}$, Conv1D weights.

$\boldsymbol{\mu} \leftarrow L^{-1} \sum_{t=0}^{L-1} \boldsymbol{X}[t, :]$;
$\boldsymbol{X}_0 \leftarrow \boldsymbol{X} - \boldsymbol{\mu}$; // Center each channel
$\boldsymbol{X}_{\text{smooth}} \leftarrow \text{Conv1D}(\boldsymbol{X}_0) + \boldsymbol{X}_0$; // Residual processing
$\widetilde{\boldsymbol{X}}_{\text{smooth}} \leftarrow \text{RightPad}_0(\boldsymbol{X}_{\text{smooth}}, L'P)$;
$\boldsymbol{X}_{\text{ds}} \leftarrow \mathcal{D}_P(\widetilde{\boldsymbol{X}}_{\text{smooth}})$; // Preserve within-period positions
$\boldsymbol{Z} \leftarrow \boldsymbol{V}^\top \boldsymbol{X}_{\text{ds}}$; // Project across cycles
$\boldsymbol{Z}_{\text{scaled}} \leftarrow \text{diag}(\boldsymbol{s}) \boldsymbol{Z}$; // Scale learned components
$\boldsymbol{Y}_{\text{ds}} \leftarrow \boldsymbol{U} \boldsymbol{Z}_{\text{scaled}}$;
$\boldsymbol{Y} \leftarrow \text{Crop}_H(\mathcal{D}_P^{-1}(\boldsymbol{Y}_{\text{ds}})) + \boldsymbol{\mu}$; // Unfold, crop, and restore mean
return $\boldsymbol{Y}$

Pre-processing. The input is mean-centered and processed by a learnable residual convolution. Zero padding makes its length divisible by P. Folding then reorganizes the processed observations without averaging.

Low-Rank Operator. The folded history is projected through $\boldsymbol{V}^\top$, scaled by $\boldsymbol{s}$, and mapped to the folded forecast through $\boldsymbol{U}$. Across all phases and channels, these projections require $\mathcal{O}(CPr(L' + H'))$ operations.

Reconstruction. Inverse folding restores chronological order. We retain the first H samples and restore the input mean. These operations introduce no trainable parameters.

Backpropagation. The convolution and low-rank factors are optimized end-to-end. Padding, folding, inverse folding, and cropping are differentiable with respect to their retained input values.

C HYPERPARAMETER SENSITIVITY

C.1 SENSITIVITY TO RANK (r)

In Figure 6, we analyze the impact of rank across three datasets:

- **ETTh1 (Simple):** Performance peaks at extremely low rank ($r = 3$), degrading as rank increases. This confirms that simple load data is strictly low-dimensional.
- **Electricity (Complex):** Shows a broader valley, optimal around $r = 8$–10.
- **Traffic (High Variance):** Requires higher capacity ($r \approx 10$–12) but still exhibits the characteristic "U-shape" where excessive rank leads to overfitting.

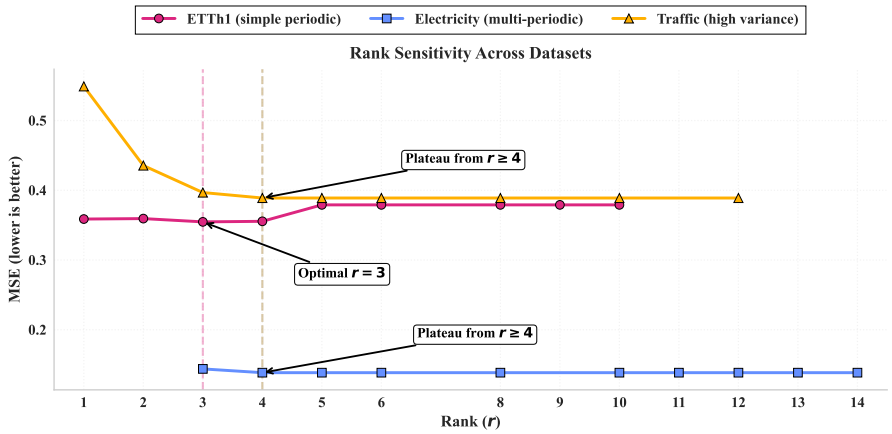

Figure 6: **Rank Sensitivity across Datasets.** Different domains require different spectral capacities, but all exhibit a clear optimum, validating the low-rank hypothesis.

C.2 SENSITIVITY TO PERIOD MISSPECIFICATION

To test robustness to errors in the detected period, we manually force LUMA to use incorrect P values on ETTh1 (true P=24). Table C.2 presents the period misspecification analysis.

Table 7: **Sensitivity to the folding interval (ETTh1).** Results for the tested intervals, which are divisors or multiples of the daily cycle.

Forced P	Meaning	Avg MSE	Δ from optimal
24	Correct (daily)	0.396	—
48	2× overestimate	0.398	+0.5%
12	Half-day	0.402	+1.5%
8	3× wrong	0.403	+1.8%
4	6× wrong	0.403	+1.8%
2	12× wrong	0.416	+5.0%

The rank capacity of $(\mathbf{U}, \mathbf{V})$ absorbs period errors by capturing residual periodic structure within the available rank budget. Only at extreme misspecification does noticeable degradation appear and even then LUMA still outperforms most baselines.

D SCALABILITY AND EFFICIENCY ANALYSIS

We perform a multi-dimensional analysis of scalability, comparing LUMA against the dense linear baseline (DLinear) and a Transformer baseline (PatchTST) as the forecast horizon H extends from 96 to 1440.

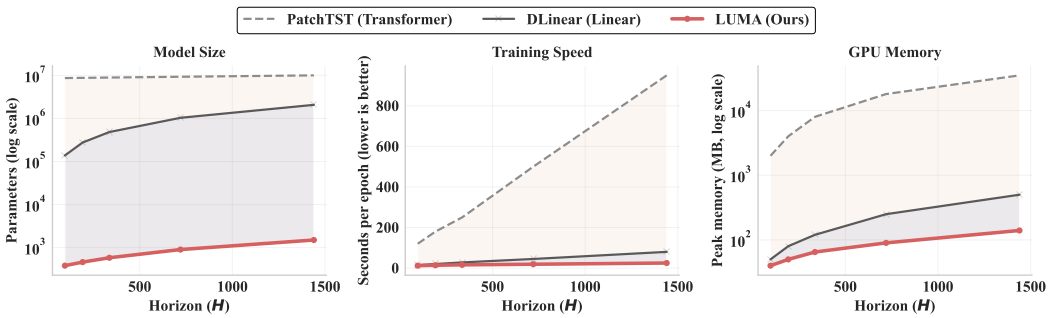

Figure 7: **Scalability Analysis. (a)** LUMA avoids the parameter explosion of DLinear (Log Scale). **(b)** LUMA is consistently faster to train. **(c)** LUMA maintains the lowest memory footprint.

1. Parameter Explosion. As detailed in the SparseTSF paper (Lin et al., 2024), dense linear models suffer from a parameter explosion problem. Since DLinear learns a mapping $\mathbb{R}^L \to \mathbb{R}^H$, its parameter count grows as $O(L \times H)$. As shown in Figure 7(a), DLinear surpasses **1 million parameters** at $H = 720$. In contrast, LUMA's parameter count scales with the *folded* horizon H/P. Even at $H = 1440$, LUMA remains under **1K parameters**, a reduction of three orders of magnitude ($10^3\times$).

2. Training Speed. Figure 7(b) demonstrates the training time per epoch. Transformers suffer from heavy backpropagation costs. While DLinear is efficient at short horizons, its training time degrades at long horizons due to the computational cost of multiplying large ($L \times H$) dense matrices. LUMA maintains near-constant training time ($\sim$18s) because it operates on small rank-r matrices, maximizing cache locality on the CPU/GPU.

3. Memory Footprint. Figure 7(c) highlights the memory bottleneck. Transformers exhibit quadratic memory growth, making them prone to OOM errors on edge devices. LUMA operates with a negligible memory footprint (< 150 MB even at $H = 1440$), validating its suitability for extreme-edge deployment.

E Evaluation under a Fixed Lookback Setting ($L = 720$)

To definitively isolate our architectural contributions from the confounding effects of variable input history lengths, we present a rigorous comparative benchmark where all models are strictly constrained to a fixed lookback window of $L = 720$. As shown in Table 8, even without the advantage of tuning the lookback hyperparameter per dataset, LUMA achieves the best average MSE on 4 out of 7 datasets. This confirms that LUMA's strong forecasting capabilities stem inherently from its robust architectural design rather than optimized input horizons.

Table 8: **Average MSE (Fixed Lookback $L = 720$).** Results are the average MSE across all prediction horizons $H \in \{96, 192, 336, 720\}$ when the lookback window is strictly fixed to $L = 720$ for all models. Best results are in **red**; second-best are in **blue**.

Dataset	LUMA (Ours)	SparseTSF	TimeMixer	FITS	iTransformer	DLinear	TimesNet
ETTh1	**0.396**	**0.407**	0.454	0.417	0.454	0.437	0.505
ETTh2	**0.336**	0.344	0.379	**0.334**	0.392	0.479	0.433
ETTm1	**0.359**	0.362	0.381	**0.359**	0.370	**0.361**	0.408
ETTm2	**0.252**	**0.252**	0.283	**0.252**	0.278	**0.271**	0.300
Weather	**0.238**	0.244	**0.238**	0.244	**0.233**	0.242	0.254
Electricity	**0.165**	0.168	0.171	0.172	**0.166**	0.169	0.238
Traffic	**0.412**	0.414	0.419	0.426	**0.406**	0.418	0.641
# Best (inc. ties)	**4/7**	**1/7**	**0/7**	**3/7**	**2/7**	**0/7**	**0/7**

F Comparison with Periodic Forecasting Methods

We compare LUMA with CycleNet, TQNet, and FreqCycle to assess its performance against other methods designed to exploit temporal and periodic structure. Table 9 reports average MSE across the four forecasting horizons. LUMA achieves the lowest MSE among these methods on six of seven datasets. On Electricity, TQNet achieves 0.164, followed by LUMA at 0.165.

Table 9: **Comparison with periodic forecasting methods.** MSE averaged across $H \in \{96, 192, 336, 720\}$. Best results are marked in **red**; second-best in **blue**.

Dataset	CycleNet	TQNet	FreqCycle	LUMA
ETTh1	**0.427**	0.441	0.428	**0.396**
ETTh2	0.383	0.378	**0.371**	**0.336**
ETTm1	0.379	0.377	**0.372**	**0.359**
ETTm2	0.266	0.277	**0.263**	**0.251**
Weather	0.243	**0.242**	0.243	**0.238**
Electricity	0.168	**0.164**	0.168	**0.165**
Traffic	0.472	**0.445**	0.448	**0.412**

G Performance on Aperiodic and Weakly-Periodic Datasets

LUMA is motivated by periodic time series. Its parameter count is determined by the architecture, while its forecasting accuracy depends on how well the imposed structure matches the data. To characterize the applicability boundary, we evaluate on two datasets known to be weakly or non-periodic: **Exchange Rate** (currency exchange, no dominant cycle) and **ILI** (influenza-like illness, weak and shifting seasonality).

Table 10: **LUMA on aperiodic datasets.** MSE at each horizon. These datasets violate the spectral compactness assumption of Theorem 3.6. Best results in **red**; second-best in blue.

Dataset	Horizon	Models					Δ from 1st
		LUMA (ours)	FITS	TimeMixer	iTransformer	TimesNet	
Exchange Rate	96	0.087	**0.082**	0.084	0.086	0.112	+6.1%
	192	0.178	**0.173**	0.178	0.177	0.209	+2.9%
	336	0.318	**0.317**	0.376	0.331	0.358	+0.3%
	720	0.904	**0.825**	0.884	0.846	0.944	+9.6%
	Avg	0.372	**0.349**	0.381	0.360	0.406	+6.6%
ILI	24	2.121	2.182	**1.804**	1.843	2.131	+17.6%
	36	2.035	2.241	1.891	**1.845**	2.612	+10.3%
	48	2.072	2.272	1.752	**1.716**	1.916	+20.7%
	60	2.056	2.642	**1.831**	2.183	1.995	+12.3%
	Avg	2.071	2.334	**1.820**	1.897	2.164	+13.8%

The results reveal a horizon-dependent pattern consistent with Theorem 3.6. On **Exchange Rate**, LUMA remains competitive at short-to-medium horizons ($H \leq 336$, gap $\leq 0.3\%$ from best at $H = 336$), where even aperiodic signals retain some local regularity exploitable by the low-rank operator. However, performance degrades sharply at $H = 720$ (MSE=0.904, $+9.6\%$ from best), where the absence of any exploitable periodic structure in the folded representation $\mathbf{X}_{\text{ds}}$ (Appendix B.6) makes long-range extrapolation unreliable. On **ILI**, LUMA remains within 10–21% of the best model across all horizons, confirming that weak seasonality still provides partial low-rank structure. The gap vs. TimeMixer and iTransformer on ILI reflects that multi-scale mixing architectures are better suited to irregular seasonal patterns than a fixed-period operator.

Takeaway. LUMA's performance degrades gracefully at short horizons on aperiodic data, but diverges at long horizons when no compact spectral structure exists. This is consistent with Proposition 3.1: when $\text{rank}(\mathbf{X}_{\text{ds}}) \gg 2K$, the rank-$r$ constraint becomes a bottleneck. We recommend applying

Table 11: **Comprehensive experimental results (MSE and MAE).** Lower is better. Avg is the arithmetic mean over prediction horizons $H \in \{96, 192, 336, 720\}$. Red and blue indicate the lowest and second lowest distinct displayed values, respectively; ties share the same color.

Dataset	H	LUMA (Ours)		TimeKAN '25		SparseTSF '24		PatchTST '23		DLinear '23		TimesNet '23		Autoformer '21	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.355**$_{\pm 0.001}$	**0.388**$_{\pm 0.000}$	**0.360**	**0.395**	0.362	**0.395**	0.385	0.399	0.384	0.424	0.384	0.418	0.449	0.459
	192	**0.395**$_{\pm 0.002}$	**0.418**$_{\pm 0.008}$	0.410	**0.425**	**0.403**	0.430	0.413	0.434	0.443	0.475	0.436	0.464	0.500	0.490
	336	**0.412**$_{\pm 0.006}$	**0.430**$_{\pm 0.007}$	0.438	0.448	**0.434**	**0.444**	0.440	0.449	0.446	0.473	0.491	0.505	0.536	0.520
	720	**0.420**$_{\pm 0.002}$	**0.436**$_{\pm 0.007}$	0.440	0.452	**0.426**	**0.443**	0.456	0.469	0.504	0.496	0.521	0.536	0.543	0.551
	Avg	**0.396**	**0.418**	0.412	0.430	**0.406**	**0.428**	0.424	0.438	0.444	0.467	0.458	0.481	0.507	0.505
ETTh2	96	**0.274**$_{\pm 0.002}$	**0.337**$_{\pm 0.001}$	**0.280**	0.355	0.294	0.346	**0.274**	**0.336**	0.282	0.343	0.340	0.372	0.346	0.358
	192	**0.334**$_{\pm 0.002}$	**0.374**$_{\pm 0.001}$	0.340	0.400	0.339	**0.376**	**0.338**	0.379	0.350	0.390	0.402	0.419	0.388	0.397
	336	**0.355**$_{\pm 0.004}$	**0.395**$_{\pm 0.002}$	0.380	0.420	**0.359**	**0.393**	0.367	0.397	0.414	0.433	0.452	0.453	0.452	0.463
	720	**0.379**$_{\pm 0.006}$	0.422$_{\pm 0.003}$	0.392	0.433	**0.383**	**0.413**	0.391	**0.416**	0.588	0.528	0.462	0.465	0.493	0.496
	Avg	**0.336**	**0.382**	0.348	**0.402**	0.344	**0.382**	**0.343**	**0.382**	0.409	0.424	0.414	0.427	0.420	0.429
ETTm1	96	0.317$_{\pm 0.001}$	0.350$_{\pm 0.001}$	0.322	0.358	0.314	0.351	**0.293**	**0.342**	**0.301**	**0.345**	0.338	0.372	0.505	0.505
	192	**0.340**$_{\pm 0.002}$	0.380$_{\pm 0.002}$	0.362	0.385	0.343	**0.371**	**0.333**	**0.366**	0.353	0.372	0.374	0.404	0.553	0.553
	336	**0.366**$_{\pm 0.001}$	**0.390**$_{\pm 0.000}$	0.382	0.402	**0.369**	**0.390**	**0.369**	**0.392**	0.383	0.409	0.410	0.428	0.621	0.621
	720	**0.414**$_{\pm 0.003}$	**0.404**$_{\pm 0.002}$	**0.406**	0.435	0.418	**0.415**	0.416	0.424	0.425	0.453	0.478	0.464	0.671	0.671
	Avg	**0.359**	**0.381**	0.368	0.395	0.361	**0.382**	**0.353**	**0.381**	0.366	0.395	0.400	0.417	0.588	0.588
ETTm2	96	**0.165**$_{\pm 0.000}$	0.255$_{\pm 0.000}$	0.174	0.252	0.167	**0.246**	**0.166**	**0.245**	0.167	0.247	0.187	0.264	0.255	0.306
	192	**0.219**$_{\pm 0.000}$	**0.293**$_{\pm 0.000}$	0.239	0.301	**0.219**	**0.296**	0.223	0.297	**0.221**	0.311	0.249	0.307	0.281	0.330
	336	**0.271**$_{\pm 0.000}$	**0.328**$_{\pm 0.000}$	0.301	0.344	**0.272**	**0.333**	0.274	0.336	0.274	0.355	0.321	0.352	0.339	0.370
	720	**0.351**$_{\pm 0.001}$	**0.379**$_{\pm 0.000}$	**0.326**	0.403	0.353	**0.394**	0.362	0.399	0.368	0.414	0.408	0.404	0.433	0.446
	Avg	**0.251**	**0.314**	0.260	0.325	**0.254**	**0.317**	0.256	0.319	0.258	0.332	0.291	0.332	0.327	0.363
Weather	96	**0.168**$_{\pm 0.001}$	**0.220**$_{\pm 0.001}$	**0.170**	**0.211**	0.174	0.231	0.190	0.225	0.172	0.250	0.176	0.235	0.323	0.389
	192	**0.209**$_{\pm 0.001}$	**0.255**$_{\pm 0.001}$	**0.211**	**0.251**	0.216	0.268	0.235	0.265	0.214	0.295	0.219	0.271	0.389	0.423
	336	**0.256**$_{\pm 0.001}$	**0.291**$_{\pm 0.001}$	0.262	**0.290**	**0.261**	0.300	0.285	0.305	0.262	0.340	0.277	0.312	0.497	0.495
	720	**0.319**$_{\pm 0.001}$	**0.334**$_{\pm 0.001}$	0.325	**0.336**	0.325	0.345	0.350	0.345	**0.320**	0.375	0.344	0.358	0.573	0.520
	Avg	**0.238**	**0.275**	**0.242**	**0.272**	0.244	0.286	0.265	0.285	**0.242**	0.315	0.254	0.294	0.446	0.457
Electricity	96	**0.137**$_{\pm 0.000}$	**0.230**$_{\pm 0.001}$	0.174	0.256	0.138	0.237	**0.129**	**0.222**	0.140	0.237	0.168	0.272	0.201	0.317
	192	**0.151**$_{\pm 0.001}$	0.260$_{\pm 0.001}$	0.182	0.264	0.155	0.252	**0.149**	**0.240**	0.153	**0.251**	0.184	0.293	0.213	0.327
	336	**0.166**$_{\pm 0.000}$	0.280$_{\pm 0.000}$	0.197	0.280	**0.169**	**0.263**	**0.166**	**0.261**	**0.169**	0.269	0.198	0.311	0.233	0.342
	720	**0.205**$_{\pm 0.000}$	**0.298**$_{\pm 0.000}$	0.217	**0.276**	**0.208**	0.303	0.210	0.337	0.214	0.310	0.220	0.339	0.255	0.369
	Avg	**0.165**	0.267	0.193	0.269	0.168	**0.264**	**0.164**	**0.265**	0.169	0.267	0.193	0.304	0.226	0.339
Traffic	96	**0.389**$_{\pm 0.000}$	**0.249**$_{\pm 0.001}$	0.400	0.270	**0.389**	0.274	**0.366**	**0.256**	0.413	0.287	0.593	0.594	0.613	0.597
	192	**0.399**$_{\pm 0.001}$	**0.269**$_{\pm 0.001}$	0.410	0.280	**0.399**	0.279	**0.388**	**0.277**	0.423	0.294	0.617	0.605	0.622	0.597
	336	**0.412**$_{\pm 0.001}$	0.289$_{\pm 0.001}$	0.420	0.300	0.417	**0.282**	**0.398**	**0.288**	0.437	0.303	0.629	0.611	0.630	0.601
	720	**0.447**$_{\pm 0.000}$	**0.305**$_{\pm 0.001}$	0.450	0.330	**0.449**	**0.286**	0.457	0.338	0.466	0.322	0.640	0.621	0.639	0.612
	Avg	**0.412**	**0.278**	0.420	0.295	0.414	**0.280**	**0.402**	0.290	0.435	0.302	0.620	0.608	0.626	0.602

LUMA to datasets where FFT analysis (Appendix B.6) confirms a dominant, stable periodicity, precisely the 7 standard LTSF benchmarks evaluated in Section 4.

H Detailed Experimental Results

We provide the full breakdown of Mean Squared Error (MSE) performance across all prediction horizons $H \in \{96, 192, 336, 720\}$.

Consistency Analysis. As shown in Table 11, LUMA achieves the lowest reported MSE across all ETTh1 horizons and remains competitive across the ETTh2 horizons. On complex datasets like **Electricity** (Table 11), we observe a crossover effect: while heavy Transformers (PatchTST) perform slightly better at short horizons ($H = 96$), LUMA overtakes them at the extreme horizon ($H = 720$, MSE 0.205 vs 0.210), confirming that our low-rank prior prevents overfitting as the forecasting window extends.

I LIMITATIONS

LUMA's accuracy–efficiency trade-off depends on how well a shared low-rank cross-period operator captures the predictive structure of the data. On weakly periodic or nonstationary series, a larger rank may be beneficial, at the cost of additional parameters and computation. Results on Exchange Rate and ILI (Appendix G, Table 10) show that performance varies with the dataset and forecast horizon, with a larger gap to the strongest evaluated baseline at the longest Exchange Rate horizon. Since the folding interval P remains fixed during inference, adapting to changes in periodicity is a direction for future work.