SELECTIVE ASSOCIATIVE MEMORY: RETHINKING THE REPRESENTATION LEARNER IN CONTINUOUS-TIME TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

We identify *selective associative memory* as the unifying principle behind gated RNNs, attention, gated CNNs/MLPs, selective SSMs, and MoE architectures. These models differ not in their fundamental operation, but in four distinct, largely independent design dimensions: operating space, branch count, gating function type, and fusion method. Based on this framework, we derive the minimal gating requirements for time series forecasting: feature and time dimension gating only. We instantiate this minimal design in **Alterum**, a lightweight continuous-time model with about 10K trainable parameters, combining an exponential-scale Fourier Feature Embedding with a Temporal Decomposition Gated Network and a closed-form ridge regression head. Across multivariate, univariate, and few-shot forecasting, Alterum attains average MSE of 0.276, 0.094, and 0.330, respectively, and remains competitive under 15% random input masking. These results indicate that a framework-driven minimal design can substantially reduce parameter count and improve inference efficiency while maintaining competitive accuracy. Code is available at https://anonymous.4open.science/r/ALTERUM-E23B/.

1 INTRODUCTION

Discrete models built on historical observations—recurrent neural networks (RNNs), multi-layer perceptrons (MLPs), and Transformers—have dominated research and practice. A central limitation is their heavy dependence on the sampling rate: inconsistent or suboptimal sampling introduces approximation errors and spurious correlations (Jenkins & Priestley, 1957; Shannon, 1949; Unser, 2000).

Continuous-time models capture the smooth evolution of signals in the continuous domain, but classical variants rely on smoothness priors and hand-specified function forms that fit nonlinear series poorly (Rasmussen & Williams, 2006; Garnelo et al., 2018). Recent meta-learning approaches (Woo et al., 2023; Le Naour et al., 2024) instead learn inductive biases from data, yet their designs are largely empirical, lacking a unifying principle for what makes a continuous representation effective.

Over the past decade, sequence modeling has witnessed an explosion of architectural innovations: gated RNNs (LSTM, GRU), attention-based Transformers, gated CNNs/MLPs, SSMs (Mamba, RWKV), and MoE models. These architectures were proposed independently, each motivated by different problems—gradient vanishing, parallelization, long-sequence efficiency, or linear complexity. Yet when we look past their surface differences, we observe a striking commonality: all of them consist of a *memory branch* that stores temporal patterns, and a *selection branch* that decides which patterns to retrieve.

We call this structure **selective associative memory**: store relevant information, selectively retrieve what is needed. Table 1 re-reads representative architectures through this lens. The differences between these models lie not in the fundamental operation, but in four distinct, largely independent design dimensions: (i) the *operating space* on which Memory and Selection each act (feature, sequence, model, or time), (ii) the *branch count* of parallel memory and selection branches (single or multiple), (iii) the *gating function type* that generates selection weights (sigmoid, softmax, or hardmax), and (iv) the *fusion method* that combines memory and selection branches (element-wise multiplication, weighted sum, sparse selection, or concatenation).

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.

This reframing has two implications. First, it reveals that gating is not a novel mechanism introduced by each new architecture—it is a recurring principle that prior architectures discovered empirically. Second, it suggests a *design principle*: for a given task, we can identify which operating space dimensions are necessary, and then build the minimal architecture that implements exactly those dimensions—no more, no less.

What operating space dimensions does time series forecasting require? We argue that sequence-dimension gating via full self-attention is not the most cost-effective choice for lightweight continuous-time forecasting: unlike language modeling which requires capturing long-range, complex token-level dependencies across thousands of steps, time series forecasting operates over a short lookback window, where local cross-step dependencies can be captured far more efficiently without the full quadratic attention overhead. Similarly, model-dimension gating via MoE is unnecessary for this well-structured task with inherent trend-seasonality decomposition, as a single lightweight expert suffices. What we do need is *feature-dimension gating*: our Fourier features encode a broad spectrum of frequencies, and not all are equally relevant to each prediction. We also need *time-scale gating*: real-world series exhibit multi-scale structure from slow long-term trends to fast periodic fluctuations, requiring gating across these characteristic timescales.

This analysis leads us to a minimal design: feature dimension gating + time dimension gating, with a single memory branch. We instantiate this minimal design in **Alterum**, a lightweight continuous-time model with about 10K trainable parameters. The Fourier Feature Embedding (FFE) provides multi-scale frequency coverage across the log-uniform spectrum commonly observed in real-world series. The Temporal Decomposition Gated Network (TDGN) implements feature dimension gating via a Sigmoid Linear Unit (SiLU), and time dimension gating via trend-seasonal decomposition. A closed-form ridge regression head provides stable prediction without end-to-end backpropagation through deep layers.

Our contributions are three-fold:

1. **Unifying framework.** We identify *selective associative memory* as the organizing principle behind gated RNNs, attention, gated CNNs/MLPs, selective SSMs, and MoE models (Table 1). We show that these seemingly diverse architectures are variations of the same motif, differing only in four distinct, largely independent design dimensions: operating space, branch count, gating function type, and fusion method (formalized in Appendix M).
2. **Minimal design principle.** Based on this framework, we derive the minimal gating requirements for a given task. For time series forecasting, this means feature and time dimension gating only—no attention, no MoE, no deep stacks.
3. **Empirical validation.** We instantiate this minimal design in Alterum. The fact that a 10K-parameter model (2–3 orders of magnitude less than typical baselines) achieves state-of-the-art or near-state-of-the-art performance across multivariate (MSE 0.276), univariate (MSE 0.094), and few-shot (MSE 0.330) forecasting—with graceful degradation under 15% random input masking—demonstrates that a framework-driven minimal design can substantially reduce parameter count and improve inference efficiency while maintaining competitive accuracy.

2 Related Work

2.1 Discrete Models for Time Series Forecasting

Discrete models operate directly on sampled observations and have dominated time series forecasting. Transformer-based models such as Autoformer (Wu et al., 2021), FEDformer (Zhou et al., 2022), PatchTST (Nie et al., 2023), and iTransformer (Liu et al., 2024b) achieve strong results through self-attention mechanisms. RNN-based approaches like SegRNN (Lin et al., 2026) and WITRAN (Jia et al., 2023) leverage recurrent connections to capture temporal dependencies. CNN-based methods including SCINet (Liu et al., 2022), MICN (Wang et al., 2023), TimesNet (Wu et al., 2023), and ModernTCN (Liu et al., 2024a) capture local patterns through convolutional kernels. MLP-based models such as N-HiTS (Challu et al., 2023), DLinear (Zeng et al., 2023), Koopa (Liu et al., 2023), and Frets (Yi et al., 2023) achieve competitive results with simpler archi-

tectures. Despite their strong performance, these models are tied to the fixed sampling frequency of continuous-time processes, and typically rely on deep network architectures.

2.2 Continuous-Time Models for Time Series Forecasting

Continuous-time models split into two largely non-overlapping lines. Early work either imposes a predefined function form or targets irregular observations: Gaussian Processes (Rasmussen & Williams, 2006) scale cubically in training size; Neural Processes (Garnelo et al., 2018) are evaluated on synthetic tasks rather than long-term forecasting; ordinary differential equation (ODE)-based models (De Brouwer et al., 2019) handle sporadically-observed medical signals; and process diffusion (Biloš et al., 2023) produces probabilistic forecasts under the continuous ranked probability score (CRPS) on irregular grids. None targets the deterministic long-term forecasting setting considered in this work. Time-index models (Woo et al., 2023; Le Naour et al., 2024) instead cast forecasting as meta-learning over continuous coordinates: DeepTime (Woo et al., 2023) concatenates random Fourier features at hand-specified scales and regresses through a deep ReLU MLP with a closed-form ridge head, while TimeFlow (Le Naour et al., 2024) focuses on imputation and irregular-sampling scenarios. Rather than proposing yet another custom continuous-time architecture, Alterum instantiates this lens minimally: FFE removes the manual grid search over concatenated Fourier scales that DeepTime requires, and TDGN realizes the memory–selection operation in a single forward pass whose parameter count is independent of the sequence length, in contrast to gated RNNs and selective SSMs whose recurrent hidden state is defined over discrete tokens and requires depth to reach long horizons.

2.3 Selective Associative Memory Across Architecture Families

The principle of *selective associative memory*—storing temporal patterns in a memory branch and selectively retrieving task-relevant components via a selection branch—recurs across diverse sequence architecture families. Table 1 re-reads representative models from RNNs, attention, gated CNNs/MLPs, SSMs, and MoE through this lens.

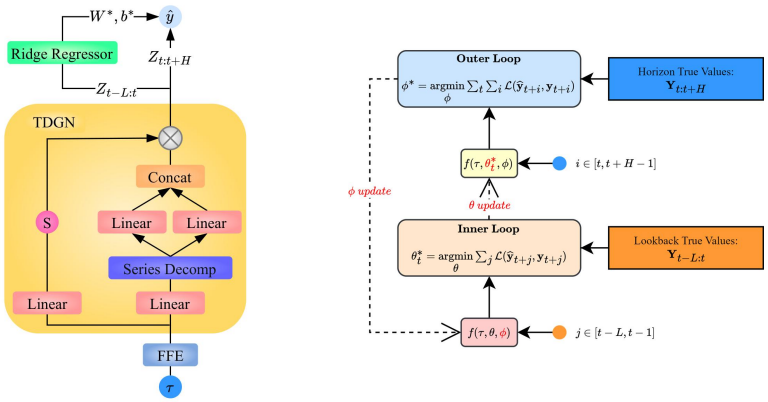
Three observations follow. First, the memory–selection structure is widespread across architecture families. Second, these architectures differ in four distinct, largely independent design dimensions: operating space, branch count, gating function type, and fusion method. Third, in most prior work, this structure emerges as a byproduct of other design goals. In contrast, we take selective associative memory as an explicit design objective, and derive the minimal gating requirements for time series forecasting: feature and time dimension gating only—instantiated in TDGN (Section 3.3). We further show that this unifying principle can be distilled into a compact mathematical paradigm $f(\mathbf{x}) = \text{Memory}(\mathbf{x}) \odot \text{Selection}(\mathbf{x})$, where the differences between architectures lie in four distinct, largely independent design dimensions: operating space, branch count, gating function type, and fusion method (see Appendix M).

Table 1: Representative sequence architectures re-read through the lens of selective associative memory. These architectures differ only in four distinct, largely independent design dimensions: operating space, branch count, gating function type, and fusion method.

Architecture	Memory branch	Selection branch	Operating space	Original motivation
LSTM (Hochreiter & Schmidhuber, 1997)	Cell and hidden states	Input / forget / output gates	Time	Learnable gated memory cell
GRU (Cho et al., 2014)	Hidden state	Update / reset gates	Time	LSTM simplification for seq2seq
Attention (Transformer) (Vaswani et al., 2017)	Value vectors	Query–key softmax weighting	Sequence	Parallelizable global dependency
Gated CNN / MLP (GLU/SwiGLU) (Dauphin et al., 2017; Shazeer, 2020)	Convolutional / linear features	Sigmoid / SiLU gating	Feature	Parallelizable non-recurrent modeling
Linear attention (Katharopoulos et al., 2020)	Kernel-projected KV accumulation	Kernel-projected query weighting	Sequence	Subquadratic attention
Mamba / selective SSM (Gu & Dao, 2024)	Recurrent hidden state	Input-dependent SSM parameters (B/C/Δ) + SiLU gating	Time	Content-based SSM
RWKV-TS (Hou & Yu, 2024)	Time-mixing state	Input-dependent time decay + receptance gating	Time	RNN–Transformer hybrid
MoE (Shazeer et al., 2017)	Multiple expert networks	Router top-k gating	Model	Sparse large-scale models
TDGN (Ours)	Trend + seasonal decomposed features	SiLU gate	Feature + Time	Selective associative memory, minimal

3 Alterum: Minimal Instantiation for Time Series Forecasting

Based on the selective associative memory framework, we now derive the minimal architecture for time series forecasting. As discussed in Section 1, we choose feature-dimension gating and time-scale gating as the two essential dimensions. Sequence-dimension gating via full self-attention is not the most cost-effective choice for lightweight continuous-time forecasting, and model-dimension gating via MoE is unnecessary for this well-structured task. This leads to three components: (1) a



(a) Alterum architecture (b) Meta-optimization framework

multi-scale frequency encoding to provide the feature foundation, (2) a gated network to implement both feature-dimension and time-scale gating, and (3) a stable prediction head.

As shown in Figure 1a, Alterum decouples representation learning from closed-form prediction. It uses relative time-index coordinates $\tau_{t+i} = \frac{i+L}{L+H-1}$, normalized to $[0,1]$ for $i = -L, \ldots, H-1$ (L lookback, H horizon), which reduces cross-scale time differences (e.g., 15-minute vs. hourly data). We first formalize the problem setting and overview the architecture, then detail the Fourier Feature Embedding (FFE), the Temporal Decomposition Gated Network (TDGN), and the closed-form ridge regression head, and finally provide theoretical analysis.

3.1 Problem Formulation and Overview

Given the lookback observations $\mathbf{X} \in \mathbb{R}^{L \times C}$ and normalized scalar time coordinates $\tau \in \mathbb{R}^{(L+H)\times 1}$ spanning both the lookback and forecast horizons, the goal is to learn a mapping f_θ from the lookback window to the next H steps $\mathbf{y} \in \mathbb{R}^{H \times C}$.

Alterum decomposes this function into two nested stages:

- **Representation learning**: A parameterized network h_ϕ maps the time coordinates τ to a high-dimensional latent representation $\mathbf{Z} = h_\phi(\tau) \in \mathbb{R}^{(L+H)\times D}$.
- **Closed-form prediction**: A differentiable closed-form ridge regressor fits a linear map between the lookback representations and the observations $\mathbf{X}$, then applies this map to the forecast representations to obtain $\hat{\mathbf{y}}$.

The final model couples a lightweight representation learner with an analytic solver (Table 8, Appendix B).

The inductive bias arises from two complementary sources: the multi-scale frequency coverage of the Fourier Feature Embedding (FFE), and the trend–seasonal decomposition with gated selection of the Temporal Decomposition Gated Network (TDGN); Section 4.2.4 and Appendix C isolate them. The closed-form ridge head further stabilizes few-shot estimates through L_2 regularization.

3.2 Fourier Feature Embedding

Early time-index models injected temporal information through Fourier analysis but struggled to extrapolate beyond the observed window (Godfrey & Gashler, 2018); random Fourier features alleviate the spectral bias of time-index modeling (Tancik et al., 2020; Rahaman et al., 2019), yet prior methods (e.g., DeepTime's Concatenated Fourier Features (CFF)) require a costly manual search over scale values (Woo et al., 2023). We propose the Fourier Feature Embedding (FFE) to remove this tuning burden.

Specifically, given a normalized scalar time coordinate $\tau \in [0,1]$, FFE maps it to a high-dimensional feature space through:

$$\gamma(\tau) = (\sin(\pi\tau\mathbf{B}_0), \ldots, \sin(\pi\tau\mathbf{B}_{N-1}), \cos(\pi\tau\mathbf{B}_0), \ldots, \cos(\pi\tau\mathbf{B}_{N-1})), \tag{1}$$

where $\mathbf{B}_n \in \mathbb{R}^{1\times d/2N}$ $(n = 0,1,\ldots,N-1)$ is sampled from a Gaussian distribution $\mathcal{N}(0,\sigma_n^2)$, d is the dimension of the FFE layer, and N is the number of scales. The standard deviations follow an exponential expansion $\sigma_n = \sigma_0 \cdot e^n$, equally spaced in the $\log_{10}$ space with interval $\log_{10} e \approx 0.434$. Smaller scales focus on overall trends and long-term dependencies, while larger scales capture short-term fluctuations and fine-grained details. FFE is parameter-free: it fixes how a continuous time point is embedded, but it does not decide which of the embedded frequencies the predictor should read. That selection is the role of TDGN, the learnable representation learner stacked on top of FFE.

3.3 Temporal Decomposition Gated Network

We design TDGN as a minimal instantiation of the storage-and-retrieval structure defined in Section 1: a linear projection forms the memory, a SiLU nonlinearity (Elfwing et al., 2018) implements the selection gate, and the standard Autoformer decomposition (Wu et al., 2021) organizes memory into the two dominant timescales of real-world series—slow long-term trend and fast periodic seasonal variation. The gate operates on these two scales independently, and their relative contribution varies across datasets (Appendix C).

Formally, given the FFE-encoded feature $\mathbf{z} = \gamma(\tau) \in \mathbb{R}^d$, TDGN first projects it through a linear layer and splits the result into an input branch and a gate branch:

$$\tilde{\mathbf{z}} = \mathrm{Linear}_0(\mathbf{z}) \in \mathbb{R}^{3h}, \quad [\mathbf{z}_{\text{inp}}, \mathbf{z}_{\text{gate}}] = \mathrm{split}(\tilde{\mathbf{z}}, [h, 2h]), \tag{2}$$

where h is the hidden dimension. The input branch is then decomposed into trend and seasonal components via moving averaging:

$$\mathbf{z}_{\text{trend}} = \mathrm{AvgPool}(\mathbf{z}_{\text{inp}}), \quad \mathbf{z}_{\text{seasonal}} = \mathbf{z}_{\text{inp}} - \mathbf{z}_{\text{trend}}. \tag{3}$$

The moving average uses a fixed window of size 25 (the standard Autoformer sequence decomposition (Wu et al., 2021)). Each component is passed through an independent linear projection; the outputs are concatenated and passed through the selection gate on the gate branch:

$$\mathbf{h} = [\mathrm{Linear}_s(\mathbf{z}_{\text{seasonal}}),\ \mathrm{Linear}_t(\mathbf{z}_{\text{trend}})] \odot \mathrm{SiLU}(\mathbf{z}_{\text{gate}}) \in \mathbb{R}^{2h}, \tag{4}$$

where the gate enables sparse activation, suppressing redundant features while retaining critical signal patterns.

3.4 Closed-Form Ridge Regression

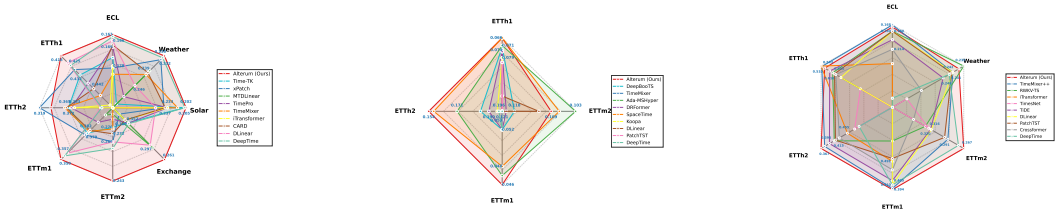
A prediction head must adapt quickly to local temporal patterns while providing effective gradients for learning global inductive biases; inner-loop gradient-based backpropagation is costly and unstable in the few-shot regime. We therefore adopt a differentiable closed-form ridge regressor (Bertinetto et al., 2019), turning the prediction layer into an analytic linear solution. Specifically, the prediction head treats the features output by the representation learner h_ϕ on the lookback window as the design matrix $\mathbf{Z}_L \in \mathbb{R}^{L\times D}$, and the ground-truth target values on the lookback window as the response $\mathbf{Y}_L \in \mathbb{R}^{L\times C}$. The inner-loop optimization problem is formulated as:

$$L(\boldsymbol{\theta}) = \|\tilde{\mathbf{Z}}_L\boldsymbol{\theta} - \mathbf{Y}_L\|_F^2 + \lambda\|\boldsymbol{\theta}\|_F^2 = \mathrm{tr}((\tilde{\mathbf{Z}}_L\boldsymbol{\theta} - \mathbf{Y}_L)^\top(\tilde{\mathbf{Z}}_L\boldsymbol{\theta} - \mathbf{Y}_L)) + \lambda\,\mathrm{tr}(\boldsymbol{\theta}^\top\boldsymbol{\theta}), \tag{5}$$

where $\tilde{\mathbf{Z}}_L = [\mathbf{Z}_L, \mathbf{1}]$ augments the lookback representation with a bias column, and λ is the regularization coefficient. Setting the gradient to zero yields the closed-form solution of the ridge regressor:

$$\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0 \Rightarrow \boldsymbol{\theta}^* = (\tilde{\mathbf{Z}}_L^\top\tilde{\mathbf{Z}}_L + \lambda\mathbf{I})^{-1}\tilde{\mathbf{Z}}_L^\top\mathbf{Y}_L. \tag{6}$$

The optimal base parameters $\boldsymbol{\theta}^*$ are applied to the horizon window: $\hat{\mathbf{y}}_H = \mathbf{Z}_H W_{\boldsymbol{\theta}^*} + b_{\boldsymbol{\theta}^*}$, with $\mathbf{Z}_H \in \mathbb{R}^{H\times D}$ the horizon feature output and $\hat{\mathbf{y}}_H \in \mathbb{R}^{H\times C}$ the prediction. The outer loop is optimized with Adam (Kingma & Ba, 2015) to obtain the meta-parameters ϕ^* of Alterum.



(a) Multivariate (b) Univariate (c) Few-shot

Figure 2: Performance comparison across three settings: (a) multivariate, (b) univariate, (c) few-shot average MSE. Alterum consistently achieves among the lowest MSE.

3.5 Theoretical Analysis

We analyze the spectral coverage property of FFE, and discuss its implication for multi-scale approximation; details are in Appendix K.

Proposition 1 (Log-uniform spectral coverage of FFE) *Let* $p_{\mathrm{FFE}}(\omega) = \frac{1}{N}\sum_{n=0}^{N-1}\mathcal{N}(0,\sigma_n^2)$ *with* $\sigma_n = \sigma_0 e^n$, *and let* $N \geq \ln(\omega_{\max}/\omega_{\min}) + c_0$ *for an absolute constant* c_0. *Then there exist constants* $c_1, c_2 > 0$ *such that for every octave band* $B = [a, 2a] \subset [\omega_{\min}, \omega_{\max}]$,

$$\frac{c_1}{N_{\mathrm{oct}}} \leq \int_B p_{\mathrm{FFE}}(\omega)\, d\omega \leq \frac{c_2}{N_{\mathrm{oct}}}, \tag{7}$$

where $N_{\mathrm{oct}} = \log_2(\omega_{\max}/\omega_{\min})$. *In particular, every octave band receives a comparable fraction of the total feature budget.*

This log-uniform coverage ensures that no frequency band is starved of features. Under the band-level assumption (A4, Appendix K) that each octave's error is bounded by $O(E_n/m_n)$, the total error is bounded by $O(E_{\mathrm{total}} \log(\omega_{\max}/\omega_{\min})/d)$, substantially better than linear. Further analysis of the standard RFF bound and empirical effective rank is in Appendix K.

4 Experiments

In this section, we empirically evaluate Alterum's effectiveness and efficiency on eight standard benchmarks (Liu et al., 2024b), comparing against representative baselines spanning all major architecture families (detailed setup in Appendix A). We present main results across multivariate, univariate, and few-shot settings, then provide controlled component analyses, ablate key design choices, examine lookback window sensitivity, and conclude with efficiency comparisons.

4.1 Main Results

4.1.1 Multivariate Forecasting

Overall performance. Alterum achieves the best average MSE (0.276), first on seven of eight datasets, second only on ETTh2 (Figure 2a, Table 2).

Comparison with baselines. Compared with the strongest baselines, Alterum improves over xPatch (0.302), the continuous-time model DeepTime (0.303), DLinear (0.305), and Time-TK (0.306) by 8.6%, 8.9%, 9.5%, and 9.8%, respectively, and over iTransformer (0.320) by 13.8% (Table 2).

Failure cases. xPatch's exponential moving-average decomposition suits ETTh2's short-term autocorrelation, the only case where Alterum ranks second.

Robustness to missing observations. To probe whether our continuous time-index representation is robust to incomplete observations, we randomly set a fraction of input values to zero in

the lookback window, retraining Alterum from scratch with the same hyperparameters. With 10% masking, Alterum ranks second overall (0.291), within 5.4% of its clean performance. Even at 15% masking it still ranks third (0.302), tied with xPatch and outperforming all other baselines, and at 20% masking it still beats iTransformer (0.320) and MTDLinear (0.330). This graceful degradation stems from the continuous frequency encoding: because FFE depends on continuous time indices rather than discrete positional embeddings, dropping individual time steps does not disrupt the spectral representation. This directly probes the core distinction between discrete and continuous-time representations—whether the model relies on exact discrete positions or on continuous time indices—rather than claiming robustness as an end in itself.

Table 2: Multivariate forecasting results (MSE), averaged over four horizons ($H \in \{96, 192, 336, 720\}$). Alterum results with and without random masking of input observations (10%, 15%, 20%) are separated from baseline models by a dashed line. Best results are **red bold**, second-best are blue underline. Full results with MAE and std in Appendix E.

Dataset	Alterum (Ours)	Alterum (mask=10%)	Alterum (mask=15%)	Alterum (mask=20%)	Time-TK 2026	xPatch 2025	MTDLinear 2025	TimePro 2025	TimeMixer 2024	iTransformer 2024b	CARD 2024	DLinear 2023	DeepTime 2023
ECL	**0.163**	0.172	0.176	0.179	0.174	0.179	0.198	0.169	0.182	0.178	0.168	0.166	0.164
ETTh1	**0.418**	0.422	0.426	0.439	0.432	0.428	0.456	0.438	0.447	0.454	0.442	0.423	0.425
ETTh2	0.340	0.376	0.408	0.458	0.372	**0.319**	0.453	0.377	0.365	0.383	0.368	0.431	0.433
ETTm1	**0.350**	0.358	0.362	0.369	0.379	0.377	0.399	0.391	0.381	0.407	0.382	0.357	0.354
ETTm2	**0.243**	0.274	0.301	0.338	0.276	0.267	0.284	0.281	0.275	0.288	0.272	0.267	0.263
Exchange	**0.261**	0.284	0.299	0.310	0.353	0.375	0.290	0.352	0.391	0.360	0.360	0.297	0.351
Solar	**0.202**	0.206	0.207	0.209	0.205	0.239	0.324	0.233	0.216	0.233	0.237	0.253	0.204
Weather	**0.230**	0.234	0.238	0.242	0.256	0.232	0.238	0.252	0.240	0.258	0.239	0.246	0.231
Avg	**0.276**	0.291	0.302	0.318	0.306	0.302	0.330	0.312	0.312	0.320	0.309	0.305	0.303

4.1.2 Univariate Forecasting

Overall performance. Alterum achieves the best average MSE (0.094), first or tied for first on three of four datasets (Figure 2b, Table 3).

Comparison with baselines. Compared with the strongest baselines, Alterum improves over SpaceTime (0.095) by 1.1% and over Ada-MSHyper (0.098) by 4.1% (Table 3), suggesting that Alterum's continuous-time representation is effective even when channel information is unavailable.

Table 3: Univariate forecasting results (MSE), averaged over four horizons ($H \in \{96, 192, 336, 720\}$). Best results are **red bold**, second-best are blue underline. Full results with MAE in Appendix F.

Dataset	Alterum (Ours)	DeepBoTS 2026	TimeMixer 2024	Ada-MSHyper 2024	DRFormer 2024	SpaceTime 2023	Koopa 2023	DLinear 2023	PatchTST 2023	DeepTime 2023
ETTh1	**0.066**	0.072	0.074	0.071	0.075	**0.066**	0.079	0.104	0.082	0.114
ETTh2	**0.154**	0.185	0.198	0.171	0.190	0.157	0.195	0.198	0.198	0.196
ETTm1	**0.046**	0.052	0.053	0.047	0.052	0.048	0.054	0.054	0.052	0.052
ETTm2	0.109	0.118	0.121	**0.103**	0.120	0.107	0.121	0.112	0.121	0.121
Avg	**0.094**	0.107	0.112	0.098	0.110	0.095	0.112	0.117	0.113	0.121

4.1.3 Few-Shot Forecasting

Overall performance. Alterum achieves the best average MSE (0.330), first on three of six datasets, second on ECL and Weather (Figure 2c, Table 4).

Failure cases. On ETTh1, Alterum ranks third (0.543) behind iTransformer (0.510) and TimeMixer++ (0.517), both of which explicitly model cross-variate dependencies via attention and are orders of magnitude larger than Alterum, suggesting that explicit inter-variate modeling with larger capacity retains an advantage under extreme data scarcity.

Comparison with baselines. Compared with mainstream baselines, Alterum is on par with the strongest baseline TimeMixer++ (0.332, 0.6% improvement), while improving over PatchTST (0.378) and DLinear (0.407) by 12.7% and 18.9% respectively (Table 4). Interestingly, the

continuous-time model DeepTime (0.482) underperforms Alterum by 31.5%, suggesting that selective associative memory design may be more data-efficient than hand-crafted Fourier features under extreme data scarcity.

Table 4: Few-shot multivariate forecasting results (MSE) with 10% training data, averaged over four horizons ($H \in \{96, 192, 336, 720\}$). Best results are **red bold**, second-best are blue underline. Full results with MAE and std in Appendix G.

Dataset	Alterum (Ours)	TimeMixer++ 2025	RWKV-TS 2024	iTransformer 2024b	TimesNet 2023	TiDE 2023	DLinear 2023	PatchTST 2023	Crossformer 2023	DeepTime 2023
ECL	0.172	**0.168**	0.180	0.241	0.323	0.196	0.180	0.180	0.214	0.306
ETTh1	0.543	0.517	0.616	**0.510**	0.869	0.589	0.691	0.633	0.645	1.160
ETTh2	**0.367**	0.379	0.411	0.455	0.479	0.395	0.605	0.415	0.428	0.494
ETTm1	**0.394**	0.398	0.561	0.491	0.677	0.425	0.411	0.501	0.462	0.410
ETTm2	**0.267**	0.291	0.325	0.375	0.320	0.317	0.316	0.296	0.343	0.275
Weather	0.236	0.241	**0.232**	0.291	0.279	0.249	0.241	0.242	0.267	0.244
Avg	**0.330**	0.332	0.388	0.394	0.491	0.362	0.407	0.378	0.393	0.482

4.2 Model Analysis

4.2.1 Inductive Bias vs. Model Scale

Inductive bias vs. parameter count. Figure 3 plots MSE against parameter count at $H = 720$. Alterum (9K parameters) achieves the lowest MSE (0.356), 13.2% below the next-best (0.410), while other models with larger parameter counts range from 0.41–0.44. Together, these results suggest that scale amplifies an already sound inductive bias.

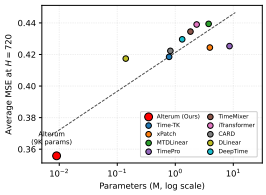


Figure 3: Average multivariate MSE at $H = 720$ vs. parameter count (log scale). Alterum (9K parameters) achieves the lowest MSE; accuracy does not improve monotonically with parameter count (dashed line: linear fit).

Controlled comparison at a matched parameter budget. To isolate inductive bias quality, we match three alternatives to Alterum's average trainable parameter count (10,329 over 32 dataset-horizon configurations) and compare four architectures on the same budget (Table 5): FFE + TDGN, CFF + TDGN, Positional Encoding + TDGN, and a Raw MLP. Replacing FFE with CFF, positional encoding, or the Raw MLP increases MSE by 4.3%, 96.0%, and 98.6%. The CFF row is particularly informative: DeepTime's CFF requires a manual grid search over eight scale ranges spanning four orders of magnitude, repeated per dataset and horizon, whereas FFE replaces this search with a single hyperparameter N and matches or slightly improves accuracy at the same capacity. The learned features retain energy across more independent principal components ($r_{90\%}$=8.9 vs. 1.3–5.8 for the alternatives), indicating that FFE does not compress temporal variation into a handful of dominant directions but preserves the full set of multi-scale frequency directions needed to capture both slow trends and fast seasonal patterns, confirming that the 10K capacity yields markedly different accuracy depending on how temporal information is encoded.

4.2.2 TDGN Ablation and Gating Analysis

TDGNś gating and decomposition are both essential, with gating the dominant bottleneck: removing it raises MSE by 27.6% on ETTh2 and 260.1% on ECL, and the full decomposition then underperforms even single-branch variants, showing that the gate is the prerequisite for the trend and seasonal branches to cooperate rather than interfere. Gating is sparsely activated (75% below 0.45 at $H = 96$), and decomposition contributes a stable 3–5% gain that adapts to dataset characteristics (Appendix C, Table 9, Figure 4).

Table 5: Controlled comparison of feature learners at a matched parameter budget (average multivariate MSE over eight datasets and four horizons; Params is the average trainable parameter count, 10,329).

Feature Learner	Inductive Bias	Params	$r_{90\%}$	$r_{95\%}$	Avg MSE	Δ vs. Ours
FFE + TDGN (Ours)	Multi-scale Fourier + decomp. + gating	10,329	8.9	12.6	0.276	–
CFF + TDGN	Manually-scaled Fourier + decomp. + gating	10,329	5.8	7.9	0.288	+4.3%
Pos. Enc. + TDGN	Sinusoidal enc. + decomp. + gating	10,329	1.4	1.6	0.541	+96.0%
Raw MLP	None	10,329	1.3	1.5	0.548	+98.6%

4.2.3 Lookback Window Sensitivity

Alterum's parameter count is independent of the lookback $L = \tau \times H$. On ETTm2, larger τ generally improves MSE up to an optimal point, with gains plateauing thereafter, while speed and memory grow linearly with τ; even at the largest setting ($\tau = 9, H = 720, L = 6480$), it uses 19 ms/iter and 0.070 GB with only 2K parameters — merely 0.05% of xPatch's 3.942M parameters — and consumes 6.2x less memory than xPatch at its standard $L = 96$. Whereas many baselines use a fixed lookback, Alterum benefits from far longer lookback sequences at a fraction of the cost (Appendix J).

4.2.4 Model Efficiency

Under the multivariate setting (Batch 32, Lookback 96, Horizon 720), Alterum uses 9K parameters (8,781), 0.038 GB GPU memory, and 15 ms/iter: 0.2% of xPatch's parameters with 88% less memory and 32% faster training, 8.9% lower MSE than DeepTime, and 0.1% of TimePro's parameters with 94% less memory and 2.9x faster training (Table 6). CARD is 83% slower with 38x more memory and 91x more parameters; DLinear is faster (11 ms/iter) but uses 2x more memory and 16x more parameters, while Alterum remains competitive on all three axes simultaneously.

Table 6: Training speed, GPU memory, parameters (Batch Size=32, Lookback=96, Horizon=720; averaged across the eight multivariate datasets). Best results are **red bold**, second-best are blue underline. Full results in Appendix H.

Model	Speed (ms/iter)	GPU Memory (GB)	Parameters (M)
Alterum (Ours)	<u>15</u>	**0.038**	**0.009**
Time-TK (Zhang et al., 2026)	33	0.389	0.783
xPatch (Stitsyuk & Choi, 2025)	22	0.316	3.942
MTDLinear (Nochumsohn et al., 2025)	43	0.139	3.754
TimePro (Ma et al., 2025)	43	0.592	8.603
TimeMixer (Wang et al., 2024)	64	1.064	1.826
iTransformer (Liu et al., 2024b)	27	0.404	2.335
CARD (Xue et al., 2024)	88	1.448	0.823
DLinear (Zeng et al., 2023)	**11**	<u>0.076</u>	<u>0.140</u>
DeepTime (Woo et al., 2023)	18	2.195	1.315

5 Limitations and Conclusion

This work presents *selective associative memory* as a unifying framework for sequence modeling. Gated RNNs, attention, gated CNNs/MLPs, selective SSMs, and MoE architectures all share the same memory–selection motif, differing only in four distinct, largely independent design dimensions: operating space, branch count, gating function type, and fusion method.

Limitations. The logarithmic error bound relies on a band-level approximation assumption; while FFE's spectral coverage holds unconditionally and empirical evidence supports the assumption, a fully rigorous bound remains future work. Second, the minimal design is derived for time series forecasting; its transferability to other modalities is untested.

Future work. We will extend this framework to NLP, CV, and Speech, and study how the minimal design scales with task complexity. This framework offers a principled path toward minimal, efficient, and interpretable sequence models.

AI USE STATEMENT

In this work, we used generative AI tools for language polishing and translation assistance. We have not used generative AI tools for generating research ideas, experimental design, or data analysis, and all scientific claims, experimental results, and mathematical derivations are the original work of the authors. We have reviewed all AI-assisted text for accuracy and take full responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI. This statement describes the present version as required by the ICLR 2027 policy.

REPRODUCIBILITY STATEMENT

All datasets used in this work are publicly available. The source code and configuration files are available at https://anonymous.4open.science/r/ALTERUM-E23B/. The full experimental results, including per-horizon breakdowns and standard deviations, are included in the appendix to enable independent verification.

REFERENCES

Haim Avron, Kenming Ng, and Vikas Sindhwani. Random fourier features for kernel ridge regression: Approximation bounds and statistical guarantees. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 183–191, 2017.

Luca Bertinetto, Joao F. Henriques, Philip H. S. Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *Advances in Neural Information Processing Systems*, 2019.

Marin Biloš, Kashif Rasul, Anderson Schneider, Yuriy Nevmyvaka, and Stephan Günnemann. Modeling temporal data as continuous functions with process diffusion. In *International Conference on Learning Representations*, 2023.

Cristian Challu, Kin G. Olivares, Boris N. Oreshkin, Federico Garza, Max Mergenthaler-Canseco, and Artur Dubrawski. N-HiTS: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 6989–6997, 2023.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

Abhimanyu Das, Weihao Kong, Adarsh Subbaswamy, Arslan Siawash, Tianyi Zhou, and Mahdi Hojat Fard. Long-term forecasting with TiDE: Time-series dense encoder. *Transactions on Machine Learning Research*, 2023.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *International Conference on Machine Learning*, 2017.

Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. Gru-ode-bayes: Continuous modeling of sporadically-observed time series. In *Advances in Neural Information Processing Systems*, 2019.

Ruixin Ding, Yuqi Chen, Yu-Ting Lan, and Wei Zhang. DRFormer: Multi-scale Transformer utilizing diverse receptive fields for long time-series forecasting. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 2024.

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J. Rezende, S. M. Ali Eslami, and Yee Whye Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018.

Luke B. Godfrey and Michael S. Gashler. Neural decomposition of time-series data for effective generalization. *IEEE Transactions on Neural Networks and Learning Systems*, 29(7):2973–2985, 2018. doi: 10.1109/TNNLS.2017.2712854.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Proceedings of the First Conference on Language Modeling (COLM)*, 2024.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997.

Haowen Hou and F. Richard Yu. RWKV-TS: Beyond traditional recurrent neural network for time series tasks. *arXiv preprint arXiv:2401.09093*, 2024.

G. M. Jenkins and M. B. Priestley. The spectral analysis of time-series. *Journal of the Royal Statistical Society: Series B (Methodological)*, 19(1):1–12, 1957. doi: 10.1111/j.2517-6161.1957.tb00240.x.

Yuxin Jia, Youfang Lin, Xinyan Hao, Yan Lin, Shengnan Guo, and Huaiyu Wan. Witran: Water-wave information transmission and recurrent acceleration network for long-range time series forecasting. In *Advances in Neural Information Processing Systems*, 2023.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pp. 5156–5165, 2020.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

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.

Zhu Li, Jean-Francois Ton, Dino Oglic, and Dino Sejdinovic. Towards a unified analysis of random fourier features. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pp. 3905–3914, 2019.

Daojun Liang, Jing Chen, Xiao Wang, Yinglong Wang, and Shuo Li. DeepBooTS: Dual-stream residual boosting for drift-resilient time-series forecasting. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence (AAAI)*, 2026.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Feiyu Zhao, Ruichao Mo, and Haotong Zhang. Segrnn: Segment recurrent neural network for long-term time series forecasting. *IEEE Internet of Things Journal*, 13(5), 2026. doi: 10.1109/JIOT.2025.3647705.

Min Liu, Zeting Cheng, Jianmin Wang, and Mingsheng Long. Scinet: Time series modeling and forecasting with sample convolution and interaction. In *Advances in Neural Information Processing Systems*, 2022.

Yong Liu, Chenyu Li, Jianmin Wang, and Mingsheng Long. Koopa: Learning non-stationary time series dynamics with koopman predictors. In *Advances in Neural Information Processing Systems*, 2023.

Yong Liu, Tengge Hu, Haoran Zhang, Shiyu Wang, and Mingsheng Long. Moderntcn: A modern pure convolution structure for general time series analysis. In *International Conference on Learning Representations*, 2024a.

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 *International Conference on Learning Representations*, 2024b.

Xiaowen Ma, Zhen-Liang Ni, Shuai Xiao, and Xinghao Chen. TimePro: Efficient multivariate long-term time series forecasting with variable- and time-aware hyper-state. In *International Conference on Machine Learning (ICML)*, 2025.

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

Liran Nochumsohn, Hedi Zisling, and Omri Azencot. A multi-task learning approach to linear multivariate forecasting. In *International Conference on Artificial Intelligence and Statistics*, 2025.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8026–8037, 2019.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, 2019.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Zongjiang Shang, Ling Chen, Binqing Wu, and Dongliang Cui. Ada-MSHyper: Adaptive multi-scale hypergraph Transformer for time series forecasting. In *Advances in Neural Information Processing Systems*, 2024.

Claude E. Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, 1949. doi: 10.1109/JRPROC.1949.232969.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

Bharath K. Sriperumbudur and Zoltán Szabó. Optimal rates for random fourier features. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, 2015.

Artyom Stitsyuk and Jaesik Choi. xPatch: Dual-stream time series forecasting with exponential seasonal-trend decomposition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 20601–20609, 2025.

Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems*, 2020.

Michael Unser. Sampling—50 years after shannon. *Proceedings of the IEEE*, 88(4):569–587, 2000. doi: 10.1109/5.843002.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *International Conference on Learning Representations*, 2023.

Shiyu Wang, Haixu Wu, Tengge Hu, Yong Liu, H. Zhou, Jianmin Wang, and Mingsheng Long. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *International Conference on Learning Representations*, 2024.

Shiyu Wang, Jiawei Li, Xiaoming Shi, Zhou Ye, Baichuan Mo, Wenze Lin, Shengtong Ju, Zhixuan Chu, and Ming Jin. TimeMixer++: A general time series pattern machine for universal predictive analysis. In *International Conference on Learning Representations*, 2025.

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*, 2023.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, 2021.

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

Wang Xue, Tian Zhou, Qingsong Wen, Jinyang Gao, Bolin Ding, and Rong Jin. CARD: Channel aligned robust blend Transformer for time series forecasting. In *International Conference on Learning Representations*, 2024.

Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Ning An, Defu Lian, Longbing Cao, and Zhendong Niu. Frequency-domain mlps are more effective learners in time series forecasting. In *Advances in Neural Information Processing Systems*, 2023.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI Conference on Artificial Intelligence*, 2023.

Fan Zhang, Shiming Fan, and Hua Wang. Time-TK: A multi-offset temporal interaction framework combining transformer and kolmogorov-arnold networks for time series forecasting. In *Proceedings of the ACM Web Conference 2026 (WWW)*, 2026.

Michael Zhang, Khaled Saab, Michael Poli, Tri Dao, Karan Goel, and Christopher Ré. Effectively modeling time series with simple discrete state spaces. In *International Conference on Learning Representations*, 2023.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations*, 2023.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed Transformer for long-term series forecasting. In *International Conference on Machine Learning*, 2022.

A Experimental Setup Details

Datasets. We evaluate Alterum on eight standard benchmark datasets (Liu et al., 2024b): ECL, four ETT subsets, Exchange, Solar-Energy, and Weather. Detailed statistics are in Table 7. (1) ECL contains electricity consumption measurements from 321 users from 2012 to 2014; (2) ETT (including ETTh1, ETTh2, ETTm1, ETTm2) contains seven influencing factors of electric power transformers from July 2016 to July 2018, where ETTh1 and ETTh2 are recorded hourly and ETTm1 and ETTm2 every 15 minutes; (3) Exchange contains daily exchange rate data for eight countries (Australia, UK, Canada, Switzerland, China, Japan, New Zealand, Singapore) from 1990 to 2016; (4) Solar-Energy records solar power generation from 137 photovoltaic power stations in 2006, sampled every 10 minutes; (5) Weather contains 21 meteorological features such as temperature and humidity, sampled every 10 minutes during 2020.

Baselines. We compare Alterum with representative baselines spanning all major architecture families:

- **Transformers**: TimeMixer++ (Wang et al., 2025), Ada-MSHyper (Shang et al., 2024), CARD (Xue et al., 2024), DRFormer (Ding et al., 2024), iTransformer (Liu et al., 2024b), Crossformer (Zhang & Yan, 2023), PatchTST (Nie et al., 2023)
- **CNNs**: xPatch (Stitsyuk & Choi, 2025), TimesNet (Wu et al., 2023)
- **RNNs**: RWKV-TS (Hou & Yu, 2024)

- **MLPs**: MTDLinear (Nochumsohn et al., 2025), DeepBoTS (Liang et al., 2026), TimeMixer (Wang et al., 2024), DLinear (Zeng et al., 2023), Koopa (Liu et al., 2023), TiDE (Das et al., 2023)
- **Kolmogorov-Arnold Networks (KANs)**: Time-TK (Zhang et al., 2026)
- **SSMs**: TimePro (Ma et al., 2025), SpaceTime (Zhang et al., 2023)
- **Continuous-time Models**: DeepTime (Woo et al., 2023)

For multivariate forecasting, baseline results are taken from their original papers; for univariate settings, we adopt results from Ada-MSHyper (Shang et al., 2024) and DRFormer (Ding et al., 2024) (except DeepBoTS (Liang et al., 2026) and SpaceTime (Zhang et al., 2023), whose univariate results are from their original papers); for few-shot settings, we adopt results from TimeMixer++ (Wang et al., 2025) and RWKV-TS (Hou & Yu, 2024). DeepTime (Woo et al., 2023) results for univariate, few-shot, and selected multivariate datasets (ETTh1, ETTh2, ETTm1, Solar) are obtained by running its official code under the same protocol, reporting the best achieved results.

Implementation Details. Following standard practice, we train Alterum with the MSE loss and Adam (Kingma & Ba, 2015) for at most 10 epochs. The lookback length is set to $L = \tau \times H$ with τ searched over $\{1,3,5,7,9\}$ per dataset and horizon (Section 4.2.3 analyzes the sensitivity; per-dataset configurations are in the released code). All experiments are implemented in PyTorch (Paszke et al., 2019), and run on a single NVIDIA GeForce RTX 3090 GPU (24 GB), and each configuration is repeated three times.

Each baseline uses its own standard lookback setting as reported in its original paper. Alterum searches over $\tau \in \{1,3,5,7,9\}$ to adapt L to the horizon, which helps exploit long history efficiently. Because the model maps continuous time coordinates to features, its parameter count is independent of L; even at the largest setting, efficiency remains competitive with baselines at their standard lookback (Sections 4.2.3 and 4.2.4).

Table 7: Summary statistics of benchmark datasets. Dataset Size is given as (train, validation, test) split lengths.

Datasets	Dim	Horizon Length	Dataset Size	Frequency	Information
Electricity	321	$\{96, 192, 336, 720\}$	(18317, 2633, 5261)	Hourly	Electricity
ETTh1, ETTh2	7	$\{96, 192, 336, 720\}$	(8545, 2881, 2881)	Hourly	Temperature
ETTm1, ETTm2	7	$\{96, 192, 336, 720\}$	(34465, 11521, 11521)	15min	Temperature
Exchange	8	$\{96, 192, 336, 720\}$	(5120, 665, 1422)	Daily	Economy
Solar-Energy	137	$\{96, 192, 336, 720\}$	(36601, 5161, 10417)	10min	Electricity
Weather	21	$\{96, 192, 336, 720\}$	(36792, 5271, 10540)	10min	Weather

B Bi-level Optimization Details

Table 8 provides an overview of the bi-level training loop.

Table 8: Overview of the bi-level training loop.

Dimension	Outer Loop	Inner Loop
Optimization target	Meta-parameters ϕ (e.g., TDGN layer parameters, ridge regularization coefficient λ)	Base parameters θ (e.g., weight W_θ and bias b_θ of the Alterum prediction layer)
Optimization objective	Minimize prediction loss on horizon window	Minimize reconstruction loss on lookback window
Optimization method	Adam optimizer	Closed-form ridge regression
Output	Optimal meta-parameters ϕ^*	Optimal base parameters θ^*

C TDGN Ablation and Gating Analysis

We conduct ablation experiments on ETTh2 (regular periodicity) and ECL (periodicity with complex perturbations) to dissect the contributions of TDGN's gating mechanism and series decomposition (Table 9).

Gating is the dominant bottleneck, with horizon-dependent impact. Removing the gating mechanism degrades performance substantially on both datasets: MSE increases from 0.163 to 0.587 (+260.1%) on ECL and from 0.340 to 0.434 (+27.6%) on ETTh2. The gap widens with prediction horizon: on ETTh2, the degradation grows from +17.5% at $H = 96$ to +47.3% at $H = 720$ (with a brief dip at $H = 192$); on ECL, it peaks at +326.7% at $H = 336$. This horizon-dependent effect indicates that irrelevant components accumulate over longer horizons, making selective suppression increasingly critical. The far larger impact on ECL reflects its 321-user electricity consumption data, where irregular and noisy components are abundant compared with the regular periodicity of ETTh2.

The gate is the prerequisite for branch cooperation. A particularly revealing finding is that without gating, the full decomposition (trend + seasonal) underperforms even single-branch variants: on ECL, w/o gating (0.587) is far worse than w/o trend (0.168) and w/o seasonal (0.170); on ETTh2, the same pattern holds (0.434 vs. 0.351/0.356). This demonstrates that the gating mechanism is not merely a noise filter but the coordination mechanism that prevents the trend and seasonal branches from producing conflicting or redundant representations. Without it, the two branches interfere with each other, degrading performance below what either branch alone achieves.

Decomposition provides stable, adaptive gains. The series decomposition contributes a consistent 3–5% improvement across both datasets. On ECL, the w/o trend variant is consistently at least as good as w/o seasonal across all four horizons, consistent with its strong daily periodicity. On ETTh2, the two branches alternate in dominance, with the w/o seasonal variant even edges out the w/o trend variant at $H = 336$ (0.373 vs. 0.376), reflecting its mixed trend-periodic characteristics and more balanced contribution. This dataset-adaptive behavior shows that decomposition allows TDGN to flexibly weight components based on data characteristics rather than relying on a fixed inductive bias.

Sparse gating activation. The gating value distribution (Figure 4) follows a unimodal pattern centered in the 0.01–0.15 range, with 75% of values below 0.45, indicating that the gate selectively activates only the most relevant components rather than uniformly scaling all features.

Table 9: TDGN ablation and component contribution analysis (MSE) across four horizons ($H \in \{96, 192, 336, 720\}$); Avg is the mean over horizons. Best results are **red bold**.

Dataset	H	Full	w/o gating	w/o seasonal	w/o trend
ETTh2	96	**0.275**	0.323	0.285	0.280
	192	**0.332**	0.384	0.356	0.342
	336	**0.362**	0.454	0.373	0.376
	720	**0.391**	0.576	0.410	0.406
	Avg	**0.340**	0.434	0.356	0.351
ECL	96	**0.134**	0.384	0.138	0.136
	192	**0.147**	0.575	0.150	0.150
	336	**0.165**	0.704	0.171	0.168
	720	**0.205**	0.683	0.222	0.218
	Avg	**0.163**	0.587	0.170	0.168

D Frequency Response Analysis

To reveal the mechanism of the FFE module, we compare the frequency response heatmaps of CFF (Figure 5a), FFE ($N = 16$) (Figure 5b), and FFE ($N = 32$) (Figure 5c). CFF uses eight fixed scales (0.01, 0.1, 1, 5, 10, 20, 50, 100) obtained through manual grid search (Woo et al., 2023), with uneven spacing in the $\log_{10}$ space (intervals of 1.0, 1.0, 0.7, 0.3, 0.3, 0.4, 0.3). This uneven arrangement causes near-zero responses in some frequency bands, producing response voids and fragmented frequency representation.

In FFE, the standard deviation of each Gaussian sampling group expands exponentially as e^n, making the scale parameters approximately equally spaced in the $\log_{10}$ parameter space with interval

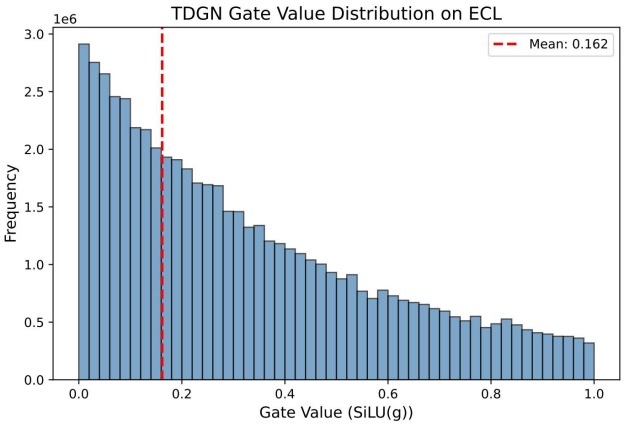
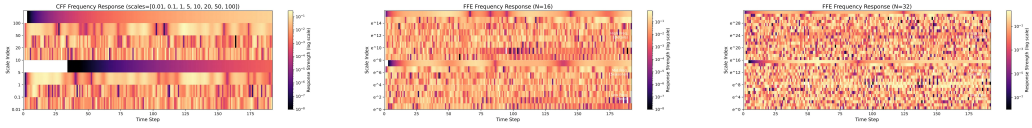


Figure 4: Distribution of TDGN gating values SiLU(g) on ECL (test phase, $H = 96$).

$\log_{10} e \approx 0.434$. This design aligns with a commonly observed temporal prior: typical periods of real-world time series (seconds, minutes, hours, days, weeks) tend to be approximately uniformly distributed in logarithmic space.

The heatmaps show that FFE ($N = 16$) possesses complete and continuous low-, mid-, and high-frequency response ranges without obvious voids, enabling simultaneous modeling of trend and periodic components.



(a) CFF (b) FFE ($N = 16$) (c) FFE ($N = 32$)

Figure 5: Comparison of frequency response heatmaps between CFF and FFE. (a) CFF; (b) FFE ($N = 16$); (c) FFE ($N = 32$). The horizontal axis represents time steps, the vertical axis represents scale index, and color indicates response intensity.

The random-masking experiment randomly replaces a fraction of input observations with zeros while keeping the sequence length and continuous time indices unchanged. All experiments reuse the exact same per-dataset hyperparameter configurations as the clean multivariate setting, with no additional tuning performed under masking. Four masking levels (0%, 10%, 15%, 20%) are evaluated.

E Full Multivariate Forecasting Results

This section reports the complete multivariate forecasting results for all baseline models (Table 10), together with the random-masking robustness experiment (Table 11).

F Full Univariate Forecasting Results

This section reports the complete univariate forecasting results for all baseline models. Table 12 summarizes these results, with Alterum's results reported as mean $\pm$ std.

Table 10: Complete multivariate forecasting results for all baselines across different horizon lengths. The first two columns show Alterum with and without 10% random masking; the dashed line separates Alterum results from baseline models. Best results are **red bold**, second-best are blue underline.

Dataset	H	Alterum (Ours)		Alterum (mask=10%)		Time-TK 2026		xPatch 2025		MTDLinear 2025		TimePro 2025		TimeMixer 2024		iTransformer 2024b		CARD 2024		DLinear 2023		DeepTime 2023	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ECL	96	**0.134**	**0.231**	0.145	0.250	0.147	0.240	0.159	0.244	0.183	0.268	0.139	0.234	0.153	0.247	0.148	0.240	0.141	0.233	0.140	0.237	0.137	0.238
	192	**0.147**	**0.245**	0.156	0.259	0.162	0.253	0.160	0.248	0.183	0.271	0.156	0.249	0.166	0.256	0.162	0.253	0.160	0.250	0.153	0.249	0.152	0.252
	336	**0.165**	0.266	0.173	0.278	0.177	0.269	0.182	0.267	0.196	0.286	0.172	0.267	0.185	0.277	0.178	0.269	0.173	**0.263**	0.169	0.267	0.166	0.268
	720	0.205	0.306	0.213	0.316	0.209	0.298	0.216	0.298	0.231	0.318	0.209	0.299	0.225	0.310	0.225	0.317	**0.197**	**0.284**	0.203	0.301	0.201	0.302
	Avg	**0.163**	0.262	0.172	0.276	0.174	0.265	0.179	0.264	0.198	0.286	0.169	0.262	0.182	0.273	0.178	0.270	0.168	**0.258**	0.166	0.264	0.164	0.265
ETTh1	96	**0.367**	0.396	0.374	0.404	0.370	0.393	0.376	**0.386**	0.386	0.396	0.375	0.398	0.375	0.400	0.386	0.405	0.383	0.391	0.375	0.399	0.374	0.399
	192	**0.403**	0.418	0.410	0.428	0.423	0.421	0.417	**0.407**	0.441	0.426	0.427	0.429	0.429	0.421	0.441	0.436	0.435	0.420	0.405	0.416	0.406	0.421
	336	**0.439**	0.447	0.444	0.455	0.465	0.444	0.449	**0.425**	0.490	0.455	0.472	0.450	0.484	0.458	0.487	0.458	0.479	0.442	**0.439**	0.443	0.442	0.449
	720	0.464	0.483	**0.460**	0.482	0.470	0.462	0.470	**0.456**	0.506	0.488	0.476	0.474	0.498	0.482	0.503	0.491	0.471	0.461	0.472	0.490	0.477	0.493
	Avg	**0.418**	0.436	0.422	0.442	0.432	0.430	0.428	**0.419**	0.456	0.441	0.438	0.438	0.447	0.440	0.454	0.448	0.442	0.429	0.423	0.437	0.425	0.441
ETTh2	96	0.275	0.338	0.315	0.376	0.293	0.340	**0.233**	**0.300**	0.300	0.345	0.293	0.345	0.289	0.341	0.297	0.349	0.281	0.330	0.289	0.353	0.290	0.349
	192	0.332	0.374	0.365	0.407	0.368	0.388	**0.291**	**0.338**	0.390	0.405	0.367	0.394	0.372	0.392	0.380	0.400	0.363	0.381	0.383	0.418	0.395	0.422
	336	0.362	0.404	0.391	0.431	0.410	0.423	**0.344**	**0.377**	0.500	0.482	0.419	0.431	0.386	0.414	0.428	0.432	0.411	0.418	0.448	0.465	0.458	0.464
	720	**0.391**	0.429	0.433	0.468	0.418	0.438	0.407	**0.427**	0.623	0.555	0.427	0.445	0.412	0.434	0.427	0.445	0.416	0.431	0.605	0.551	0.589	0.550
	Avg	0.340	0.386	0.376	0.421	0.372	0.397	**0.319**	**0.361**	0.453	0.447	0.377	0.404	0.365	0.395	0.383	0.407	0.368	0.390	0.431	0.447	0.433	0.446
ETTm1	96	0.301	0.344	0.311	0.356	0.315	0.354	0.311	0.346	0.337	0.363	0.326	0.364	0.320	0.357	0.334	0.368	0.316	0.347	**0.299**	**0.343**	0.305	0.347
	192	0.337	0.369	0.345	0.378	0.356	0.378	0.348	0.368	0.379	0.388	0.367	0.383	0.361	0.381	0.377	0.391	0.363	0.370	**0.335**	**0.365**	0.340	0.371
	336	**0.358**	0.387	0.367	0.397	0.393	0.402	0.388	0.391	0.412	0.414	0.402	0.409	0.390	0.404	0.426	0.420	0.392	0.390	0.369	**0.386**	0.366	0.389
	720	**0.405**	0.425	0.409	0.428	0.453	0.439	0.461	0.430	0.468	0.445	0.469	0.446	0.454	0.441	0.491	0.459	0.458	0.425	0.425	**0.421**	0.406	0.425
	Avg	**0.350**	0.381	0.358	0.390	0.379	0.393	0.377	0.384	0.399	0.403	0.391	0.401	0.381	0.396	0.407	0.410	0.382	0.383	0.357	**0.379**	0.354	0.383
ETTm2	96	**0.163**	0.252	0.199	0.296	0.173	0.253	0.164	**0.248**	0.179	0.264	0.178	0.260	0.175	0.258	0.180	0.264	0.169	**0.248**	0.167	0.260	0.166	0.257
	192	**0.217**	0.295	0.252	0.336	0.238	0.298	0.230	**0.291**	0.245	0.308	0.242	0.303	0.237	0.299	0.250	0.309	0.234	0.292	0.224	0.303	0.225	0.302
	336	**0.264**	**0.329**	0.293	0.364	0.298	0.339	0.292	0.331	0.306	0.350	0.303	0.342	0.298	0.340	0.311	0.348	0.294	0.339	0.281	0.342	0.277	0.336
	720	**0.329**	**0.372**	0.353	0.399	0.395	0.395	0.381	0.383	0.407	0.415	0.400	0.399	0.391	0.396	0.412	0.407	0.390	0.388	0.397	0.421	0.383	0.409
	Avg	**0.243**	**0.312**	0.274	0.349	0.276	0.321	0.267	0.313	0.284	0.334	0.281	0.326	0.275	0.323	0.288	0.332	0.272	0.317	0.267	0.332	0.263	0.326
Exchange	96	**0.078**	**0.199**	0.108	0.243	0.083	0.202	0.082	**0.199**	0.084	0.202	0.085	0.204	0.090	0.235	0.086	0.206	0.084	0.202	0.081	0.203	0.081	0.205
	192	0.158	0.293	0.193	0.329	0.168	0.292	0.177	0.298	0.173	0.300	0.178	0.299	0.187	0.343	0.177	0.299	0.174	0.295	0.157	0.293	**0.151**	**0.284**
	336	**0.269**	**0.393**	0.282	0.409	0.322	0.411	0.349	0.425	0.306	0.412	0.328	0.414	0.353	0.473	0.331	0.417	0.342	0.421	0.305	0.414	0.314	0.412
	720	**0.537**	**0.580**	0.551	0.594	0.838	0.684	0.891	0.711	0.595	0.595	0.817	0.679	0.934	0.761	0.847	0.691	0.841	0.689	0.643	0.601	0.856	0.663
	Avg	**0.261**	**0.366**	0.284	0.394	0.353	0.397	0.375	0.408	0.290	0.377	0.352	0.399	0.391	0.453	0.360	0.403	0.360	0.402	0.297	0.378	0.351	0.391
Solar	96	0.190	0.242	0.197	0.259	**0.187**	0.234	0.201	0.215	0.283	0.361	0.196	0.237	0.189	0.259	0.203	0.237	0.194	**0.209**	0.222	0.291	0.189	0.254
	192	**0.192**	0.249	0.197	0.260	0.205	0.256	0.235	**0.234**	0.313	0.383	0.231	0.263	0.222	0.283	0.233	0.261	0.234	0.235	0.250	0.309	0.196	0.257
	336	**0.206**	0.261	0.208	0.266	0.213	0.271	0.258	**0.249**	0.348	0.403	0.250	0.281	0.231	0.292	0.248	0.273	0.256	0.253	0.269	0.324	0.208	0.262
	720	0.219	0.283	0.222	0.288	**0.214**	0.267	0.260	**0.247**	0.353	0.401	0.253	0.285	0.223	0.285	0.249	0.275	0.262	0.257	0.271	0.328	0.224	0.267
	Avg	**0.202**	0.259	0.206	0.268	0.205	0.257	0.239	**0.236**	0.324	0.387	0.233	0.267	0.216	0.280	0.233	0.262	0.237	0.239	0.253	0.313	0.204	0.260
Weather	96	0.168	0.223	0.172	0.234	0.173	0.213	0.168	0.203	0.159	0.221	0.166	0.207	0.163	0.209	0.174	0.214	**0.150**	**0.188**	0.176	0.237	0.166	0.221
	192	0.210	0.265	0.214	0.272	0.222	0.257	0.214	0.245	**0.202**	0.268	0.216	0.254	0.208	0.250	0.221	0.254	**0.202**	**0.238**	0.220	0.282	0.207	0.261
	336	0.246	0.292	0.250	0.301	0.277	0.296	**0.236**	**0.273**	0.259	0.318	0.273	0.296	0.251	0.287	0.278	0.296	0.260	0.282	0.265	0.319	0.251	0.298
	720	**0.296**	0.333	0.299	0.338	0.351	0.346	0.309	**0.321**	0.332	0.373	0.351	0.346	0.339	0.341	0.358	0.347	0.343	0.353	0.323	0.362	0.301	0.338
	Avg	**0.230**	0.278	0.234	0.286	0.256	0.278	0.232	**0.261**	0.238	0.295	0.252	0.276	0.240	0.272	0.258	0.278	0.239	0.265	0.246	0.300	0.231	0.280
Avg		**0.276**	0.335	0.291	0.353	0.306	0.342	0.302	**0.331**	0.330	0.371	0.312	0.347	0.312	0.354	0.320	0.351	0.309	0.335	0.305	0.356	0.303	0.349

G Full Few-Shot Forecasting Results

This section reports the complete few-shot forecasting results. Table 13 summarizes the MSE and MAE results across all baselines, and Table 14 reports Alterum's detailed results across different horizon lengths as mean ± std.

H Full Model Efficiency Results

All efficiency metrics are measured on a single NVIDIA RTX 3090 GPU, with batch size fixed to 32, lookback window $L = 96$, and prediction horizon $H = 720$. Training speed is reported as the average real-world runtime per training step (ms/iter), measured after warmup over 10^3 iterations. GPU memory reports the peak allocated memory during training, and parameter counts report the number of trainable parameters in millions (M).

This section reports the complete model efficiency results for all baseline models under Batch Size=32, Lookback=96, Horizon=720, as shown in Table 15.

I Prediction Visualization

This section presents visualization of prediction results on the test set for each dataset under Lookback=96, Horizon=96 (Figures 6–13). We compare with xPatch, MTDLinear, TimeMixer, iTransformer, and CARD. Among these baselines, Alterum demonstrates competitive prediction performance.

Table 11: Random-masking robustness of Alterum (MSE & MAE ± std). Each dataset reports four horizons ($H \in \{96, 192, 336, 720\}$) and an average. The masking fraction indicates the percentage of input observations randomly set to zero. All experiments reuse the same per-dataset hyperparameters as the clean multivariate setting.

Dataset	H	mask=0%		mask=10%		mask=15%		mask=20%	
		MSE ± Std	MAE ± Std	MSE ± Std	MAE ± Std	MSE ± Std	MAE ± Std	MSE ± Std	MAE ± Std
ECL	96	0.134±0.000	0.231±0.001	0.145±0.000	0.250±0.001	0.150±0.000	0.257±0.001	0.154±0.001	0.263±0.001
	192	0.147±0.001	0.245±0.000	0.156±0.001	0.259±0.001	0.159±0.001	0.264±0.001	0.162±0.001	0.269±0.001
	336	0.165±0.002	0.266±0.003	0.173±0.002	0.278±0.002	0.177±0.002	0.283±0.003	0.181±0.002	0.289±0.003
	720	0.205±0.001	0.306±0.002	0.213±0.002	0.316±0.002	0.216±0.002	0.317±0.002	0.219±0.001	0.321±0.002
	Avg	0.163±0.001	0.262±0.002	0.172±0.001	0.276±0.002	0.176±0.001	0.280±0.002	0.179±0.001	0.286±0.002
ETTh1	96	0.367±0.003	0.396±0.002	0.374±0.002	0.404±0.002	0.380±0.002	0.411±0.001	0.388±0.002	0.420±0.002
	192	0.403±0.001	0.418±0.001	0.410±0.006	0.428±0.007	0.414±0.008	0.433±0.009	0.416±0.005	0.436±0.008
	336	0.439±0.006	0.447±0.005	0.444±0.001	0.455±0.001	0.447±0.002	0.460±0.002	0.448±0.004	0.461±0.004
	720	0.464±0.010	0.483±0.005	0.460±0.007	0.482±0.004	0.463±0.008	0.486±0.004	0.505±0.031	0.514±0.018
	Avg	0.418±0.005	0.436±0.003	0.422±0.004	0.442±0.004	0.426±0.005	0.448±0.004	0.439±0.010	0.458±0.008
ETTh2	96	0.275±0.003	0.338±0.002	0.315±0.002	0.376±0.002	0.351±0.001	0.405±0.001	0.402±0.003	0.440±0.002
	192	0.332±0.003	0.374±0.003	0.365±0.003	0.407±0.003	0.400±0.002	0.433±0.002	0.450±0.003	0.467±0.002
	336	0.362±0.013	0.404±0.012	0.391±0.008	0.431±0.007	0.425±0.006	0.455±0.005	0.474±0.004	0.487±0.005
	720	0.391±0.003	0.429±0.002	0.433±0.029	0.468±0.021	0.456±0.010	0.486±0.007	0.504±0.006	0.516±0.006
	Avg	0.340±0.006	0.386±0.005	0.376±0.011	0.421±0.008	0.408±0.005	0.445±0.004	0.458±0.004	0.478±0.004
ETTm1	96	0.301±0.001	0.344±0.001	0.311±0.003	0.356±0.004	0.314±0.003	0.359±0.003	0.321±0.008	0.367±0.012
	192	0.337±0.001	0.369±0.001	0.345±0.001	0.378±0.001	0.351±0.001	0.387±0.001	0.361±0.005	0.398±0.002
	336	0.358±0.002	0.387±0.002	0.367±0.006	0.397±0.005	0.371±0.002	0.403±0.002	0.379±0.002	0.413±0.002
	720	0.405±0.009	0.425±0.008	0.409±0.012	0.428±0.011	0.410±0.012	0.430±0.011	0.415±0.013	0.437±0.011
	Avg	0.350±0.003	0.381±0.003	0.358±0.006	0.390±0.005	0.362±0.005	0.395±0.004	0.369±0.007	0.404±0.007
ETTm2	96	0.163±0.001	0.252±0.001	0.199±0.003	0.296±0.004	0.202±0.007	0.300±0.006	0.204±0.002	0.300±0.002
	192	0.217±0.002	0.295±0.002	0.252±0.002	0.336±0.002	0.289±0.004	0.368±0.004	0.339±0.003	0.407±0.003
	336	0.264±0.002	0.329±0.001	0.293±0.001	0.364±0.002	0.329±0.001	0.394±0.001	0.378±0.001	0.430±0.001
	720	0.329±0.002	0.372±0.001	0.353±0.001	0.399±0.001	0.385±0.000	0.425±0.001	0.431±0.001	0.459±0.001
	Avg	0.243±0.002	0.312±0.001	0.274±0.002	0.349±0.002	0.301±0.003	0.372±0.003	0.338±0.002	0.399±0.002
Exchange	96	0.078±0.002	0.199±0.003	0.108±0.007	0.243±0.010	0.131±0.019	0.270±0.021	0.127±0.007	0.265±0.008
	192	0.158±0.002	0.293±0.002	0.193±0.003	0.329±0.004	0.199±0.002	0.335±0.001	0.204±0.006	0.340±0.006
	336	0.269±0.009	0.393±0.005	0.282±0.010	0.409±0.011	0.298±0.011	0.425±0.010	0.322±0.010	0.448±0.007
	720	0.537±0.029	0.580±0.020	0.551±0.044	0.594±0.030	0.568±0.056	0.606±0.036	0.586±0.063	0.619±0.037
	Avg	0.261±0.011	0.366±0.008	0.284±0.016	0.394±0.014	0.299±0.022	0.409±0.017	0.310±0.022	0.418±0.014
Solar	96	0.190±0.001	0.242±0.002	0.197±0.001	0.259±0.003	0.200±0.001	0.264±0.003	0.202±0.001	0.270±0.003
	192	0.192±0.002	0.249±0.009	0.197±0.002	0.260±0.008	0.197±0.002	0.263±0.007	0.200±0.002	0.266±0.007
	336	0.206±0.003	0.261±0.010	0.208±0.003	0.266±0.008	0.209±0.003	0.269±0.008	0.211±0.003	0.273±0.007
	720	0.219±0.003	0.283±0.009	0.222±0.004	0.288±0.011	0.222±0.007	0.290±0.017	0.222±0.003	0.290±0.010
	Avg	0.202±0.002	0.259±0.008	0.206±0.003	0.268±0.008	0.207±0.003	0.272±0.009	0.209±0.002	0.275±0.007
Weather	96	0.168±0.001	0.223±0.002	0.172±0.000	0.234±0.001	0.176±0.001	0.243±0.001	0.180±0.004	0.250±0.010
	192	0.210±0.001	0.265±0.001	0.214±0.002	0.272±0.001	0.218±0.002	0.280±0.001	0.223±0.001	0.289±0.001
	336	0.246±0.003	0.292±0.003	0.250±0.003	0.301±0.003	0.255±0.003	0.308±0.003	0.259±0.002	0.317±0.003
	720	0.296±0.002	0.333±0.003	0.299±0.002	0.338±0.004	0.302±0.002	0.345±0.003	0.307±0.002	0.353±0.002
	Avg	0.230±0.002	0.278±0.002	0.234±0.002	0.286±0.002	0.238±0.002	0.294±0.002	0.242±0.002	0.302±0.004
Avg		0.276±0.004	0.335±0.004	0.291±0.006	0.353±0.006	0.302±0.006	0.364±0.006	0.318±0.006	0.378±0.006

Table 12: Complete univariate forecasting results for all baselines across different horizon lengths. Best results are **red bold**, second-best are blue underline.

Dataset	H	Alterum (Ours)		DeepBoTS 2026		TimeMixer 2024		Ada-MSHyper 2024		DRFormer 2024		SpaceTime 2023		Koopa 2023		DLinear 2023		PatchTST 2023		DeepTime 2023	
		MSE ± Std	MAE ± Std	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.054±0.001**	0.176±0.001	0.055	0.177	0.057	0.181	0.057	**0.173**	0.056	0.183	**0.054**	0.181	0.058	0.184	0.056	0.180	0.056	0.181	0.055	0.176
	192	**0.065±0.001**	0.205±0.002	0.072	0.204	0.072	0.204	0.072	**0.198**	0.072	0.206	0.066	0.207	0.073	0.207	0.071	0.204	0.076	0.210	0.078	0.210
	336	**0.068±0.001**	**0.212±0.001**	0.080	0.219	0.085	0.227	0.070	0.213	0.084	0.230	0.069	**0.212**	0.089	0.232	0.098	0.244	0.094	0.242	0.115	0.264
	720	**0.075±0.001**	**0.220±0.001**	0.079	0.224	0.083	0.227	0.085	0.228	0.088	0.233	0.076	0.222	0.094	0.241	0.189	0.359	0.101	0.250	0.208	0.363
	Avg	**0.066±0.001**	**0.203±0.001**	0.072	0.206	0.074	0.210	0.071	**0.203**	0.075	0.213	**0.066**	0.206	0.079	0.216	0.104	0.247	0.082	0.221	0.114	0.253
ETTh2	96	0.128±0.000	0.272±0.001	0.129	0.275	0.133	0.283	**0.116**	**0.262**	0.133	0.279	0.119	0.268	0.139	0.290	0.131	0.279	0.130	0.276	0.130	0.275
	192	0.153±0.009	0.317±0.009	0.178	0.329	0.190	0.341	0.168	0.323	0.183	0.336	**0.151**	**0.306**	0.183	0.336	0.176	0.329	0.181	0.331	0.170	0.332
	336	**0.161±0.002**	**0.324±0.001**	0.211	0.365	0.226	0.379	0.177	0.350	0.218	0.373	0.169	0.332	0.219	0.385	0.209	0.367	0.226	0.379	0.183	0.350
	720	**0.174±0.006**	**0.336±0.006**	0.220	0.377	0.241	0.396	0.221	0.380	0.227	0.383	0.188	0.352	0.238	0.393	0.276	0.426	0.253	0.406	0.302	0.444
	Avg	**0.154±0.004**	**0.312±0.004**	0.185	0.337	0.198	0.350	0.171	0.329	0.190	0.343	0.157	0.315	0.195	0.351	0.198	0.350	0.198	0.348	0.196	0.350
ETTm1	96	**0.026±0.000**	0.121±0.000	0.029	0.126	0.029	0.128	0.027	**0.118**	0.028	0.125	**0.026**	0.121	0.030	0.130	0.028	0.123	0.029	0.126	0.029	0.125
	192	**0.038±0.001**	**0.148±0.000**	0.044	0.158	0.044	0.160	**0.038**	**0.148**	0.043	0.159	0.039	0.152	0.045	0.161	0.045	0.156	0.043	0.158	0.044	0.156
	336	**0.051±0.001**	0.170±0.001	0.057	0.185	0.058	0.185	0.052	**0.165**	0.057	0.184	**0.051**	0.173	0.060	0.188	0.061	0.182	0.056	0.183	0.057	0.178
	720	**0.070±0.001**	**0.201±0.002**	0.080	0.218	0.081	0.218	0.071	0.206	0.081	0.219	0.074	0.213	0.081	0.218	0.080	0.210	0.080	0.217	0.079	0.209
	Avg	**0.046±0.001**	0.160±0.001	0.052	0.172	0.053	0.173	0.047	**0.159**	0.052	0.172	0.048	0.165	0.054	0.174	0.054	0.168	0.052	0.171	0.052	0.167
ETTm2	96	0.063±0.001	0.182±0.001	0.064	0.180	0.068	0.187	**0.051**	**0.163**	0.064	0.182	0.060	0.179	0.067	0.186	0.063	0.183	0.071	0.192	0.065	0.186
	192	0.091±0.001	0.227±0.001	0.099	0.233	0.101	0.236	**0.089**	**0.207**	0.102	0.233	0.090	0.222	0.101	0.238	0.092	0.227	0.102	0.237	0.096	0.234
	336	0.118±0.001	0.261±0.001	0.129	0.273	0.133	0.278	0.114	**0.240**	0.129	0.273	**0.113**	0.255	0.134	0.279	0.119	0.261	0.130	0.274	0.138	0.285
	720	0.164±0.006	0.327±0.006	0.180	0.329	0.183	0.332	**0.156**	**0.310**	0.176	0.328	0.166	0.318	0.182	0.332	0.175	0.320	0.179	0.328	0.186	0.338
	Avg	0.109±0.002	0.249±0.002	0.118	0.254	0.121	0.258	**0.103**	**0.230**	0.120	0.258	0.107	0.244	0.121	0.259	0.112	0.248	0.121	0.258	0.121	0.261
Avg		**0.094±0.002**	0.231±0.002	0.107	0.242	0.112	0.248	0.098	**0.230**	0.110	0.252	0.095	0.233	0.112	0.250	0.117	0.250	0.113	0.250	0.121	0.258

Table 13: Few-shot multivariate forecasting results (MSE & MAE) based on 10% training data, averaged over four prediction horizons ($H \in \{96, 192, 336, 720\}$). Best results are **red bold**, second-best are blue underline.

Dataset	Alterum (Ours)		TimeMixer++ 2025		RWKV-TS 2024		iTransformer 2024b		TimesNet 2023		TiDE 2023		DLinear 2023		PatchTST 2023		Crossformer 2023		DeepTime 2023	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ECL	0.172	0.272	**0.168**	**0.271**	0.180	0.274	0.241	0.337	0.323	0.392	0.196	0.289	0.180	0.280	0.180	0.273	0.214	0.308	0.306	0.373
ETTh1	0.543	**0.504**	0.517	0.512	0.616	0.542	**0.510**	0.597	0.869	0.628	0.589	0.535	0.691	0.600	0.633	0.542	0.645	0.558	1.160	0.699
ETTh2	**0.367**	0.407	0.379	**0.391**	0.411	0.425	0.455	0.461	0.479	0.465	0.395	0.412	0.605	0.538	0.415	0.431	0.428	0.447	0.494	0.471
ETTm1	**0.394**	**0.406**	0.398	0.431	0.561	0.486	0.491	0.516	0.677	0.537	0.425	0.458	0.411	0.429	0.501	0.466	0.462	0.489	0.410	0.414
ETTm2	**0.267**	**0.321**	0.291	0.351	0.325	0.357	0.375	0.412	0.320	0.353	0.317	0.371	0.316	0.368	0.296	0.343	0.343	0.389	0.275	**0.321**
Weather	0.236	0.277	0.241	0.271	**0.232**	**0.268**	0.291	0.331	0.279	0.301	0.249	0.291	0.241	0.283	0.242	0.279	0.267	0.306	0.244	0.278
Avg	**0.330**	**0.365**	0.332	0.371	0.388	0.392	0.394	0.442	0.491	0.446	0.362	0.393	0.407	0.416	0.378	0.389	0.393	0.416	0.482	0.426

Table 14: Alterum few-shot forecasting results across different horizon lengths (MSE & MAE ± std).

Dataset	H	MSE ± Std	MAE ± Std
ECL	96	0.138±0.000	0.237±0.000
	192	0.153±0.001	0.252±0.001
	336	0.173±0.001	0.273±0.001
	720	0.222±0.002	0.324±0.002
	Avg	0.172±0.001	0.272±0.001
ETTh1	96	0.419±0.003	0.422±0.002
	192	0.463±0.004	0.463±0.004
	336	0.547±0.016	0.510±0.010
	720	0.742±0.005	0.622±0.004
	Avg	0.543±0.007	0.504±0.005
ETTh2	96	0.287±0.004	0.351±0.003
	192	0.375±0.001	0.394±0.001
	336	0.377±0.008	0.425±0.005
	720	0.428±0.003	0.459±0.002
	Avg	0.367±0.004	0.407±0.003
ETTm1	96	0.342±0.003	0.372±0.002
	192	0.375±0.002	0.390±0.001
	336	0.401±0.001	0.410±0.001
	720	0.459±0.001	0.451±0.001
	Avg	0.394±0.002	0.406±0.001
ETTm2	96	0.175±0.001	0.256±0.001
	192	0.235±0.001	0.297±0.000
	336	0.288±0.002	0.343±0.003
	720	0.371±0.003	0.389±0.001
	Avg	0.267±0.002	0.321±0.001
Weather	96	0.165±0.001	0.216±0.001
	192	0.208±0.001	0.256±0.002
	336	0.257±0.001	0.296±0.000
	720	0.314±0.001	0.339±0.001
	Avg	0.236±0.001	0.277±0.001
Avg		0.330±0.003	0.365±0.002

J Effect of Lookback Window Length

We analyze the effect of the lookback window length L (where $L = \tau \times H$) on the ETTm2 dataset, with batch size set to 32. Since our model architecture makes the parameter count independent of the lookback window, we only analyze the impact on model performance, training speed, and memory usage.

As shown in Figure 14a, increasing τ reduces MSE, with the optimal average performance achieved at $\tau = 7$. When $\tau > 7$, performance on short horizon ($H = 96$) plateaus, while performance on long

Table 15: Complete model efficiency results across all dataset configurations. Best results are **red bold**, second-best are blue underline.

Metric	Dataset	Alterum (Ours)	Time-TK 2026	xPatch 2025	MTDLinear 2025	TimePro 2025	TimeMixer 2024	iTransformer 2024b	CARD 2024	DLinear 2023	DeepTime 2023
Speed (ms/iter)	ECL	29	55	43	212	97	157	57	262	**28**	**28**
	ETTh1	12	25	17	17	27	50	23	55	**8**	15
	ETTh2	12	26	18	16	17	49	21	53	**8**	15
	ETTm1	11	23	18	14	28	42	19	51	**6**	16
	ETTm2	11	24	19	13	26	45	20	50	**6**	16
	Exchange	11	32	16	18	21	52	22	56	**10**	16
	Solar	20	43	26	24	88	56	28	118	**15**	18
	Weather	12	33	20	26	42	57	29	59	**7**	18
	Avg	15	33	22	43	43	64	27	88	**11**	18
GPU Memory (GB)	ECL	**0.177**	1.989	1.312	0.711	1.948	6.126	1.709	7.527	0.288	7.952
	ETTh1	**0.007**	0.028	0.095	0.038	0.061	0.138	0.205	0.046	0.030	0.121
	ETTh2	**0.007**	0.073	0.095	0.038	0.043	0.138	0.152	0.046	0.030	0.121
	ETTm1	**0.012**	0.045	0.095	0.038	0.153	0.138	0.152	0.046	0.030	0.427
	ETTm2	**0.005**	0.033	0.095	0.038	0.061	0.231	0.152	0.046	0.030	0.427
	Exchange	**0.007**	0.030	0.098	0.037	0.066	0.153	0.152	0.050	0.030	0.122
	Solar	**0.075**	0.765	0.598	0.149	2.177	1.136	0.442	3.277	0.134	7.254
	Weather	**0.012**	0.150	0.143	0.064	0.224	0.452	0.264	0.547	0.037	1.134
	Avg	**0.038**	0.389	0.316	0.139	0.592	1.064	0.404	1.448	0.076	2.195
Parameters (M)	ECL	**0.051**	1.291	3.942	24.444	7.887	0.220	5.154	1.939	0.140	1.315
	ETTh1	**0.006**	0.148	3.942	0.698	2.073	0.190	3.576	0.154	0.140	1.315
	ETTh2	**0.002**	1.291	3.942	0.698	1.135	0.190	0.305	0.154	0.140	1.315
	ETTm1	**0.004**	0.540	3.942	0.698	7.887	0.190	0.305	0.154	0.140	1.315
	ETTm2	**0.002**	0.262	3.942	0.698	2.073	0.190	0.305	0.154	0.140	1.315
	Exchange	**0.0004**	0.148	3.942	0.559	2.393	0.190	0.305	0.154	0.140	1.315
	Solar	**0.004**	1.291	3.942	0.559	38.758	13.220	3.576	1.939	0.140	1.315
	Weather	**0.002**	1.291	3.942	1.676	6.622	0.219	5.154	1.939	0.140	1.315
	Avg	**0.009**	0.783	3.942	3.754	8.603	1.826	2.335	0.823	0.140	1.315

horizon ($H = 720$) slightly decreases, indicating that excessively long lookback inputs introduce redundant noise. As shown in Figures 14b and 14c, training speed and GPU memory increase linearly with τ. Even at the maximum setting ($\tau = 9, H = 720$), the total overhead (19 ms/iter, 0.070 GB) remains comparable to or lower than most baselines: xPatch (19 ms/iter, 0.095 GB), TimeMixer (45 ms/iter, 0.231 GB), iTransformer (20 ms/iter, 0.152 GB), and DeepTime (106 ms/iter, 0.427 GB). While CARD uses less memory (0.046 GB), it is significantly slower (50 ms/iter). Therefore, we recommend selecting $\tau \in [5, 7]$ as the optimal trade-off.

K FFE Spectral Coverage and Approximation Analysis

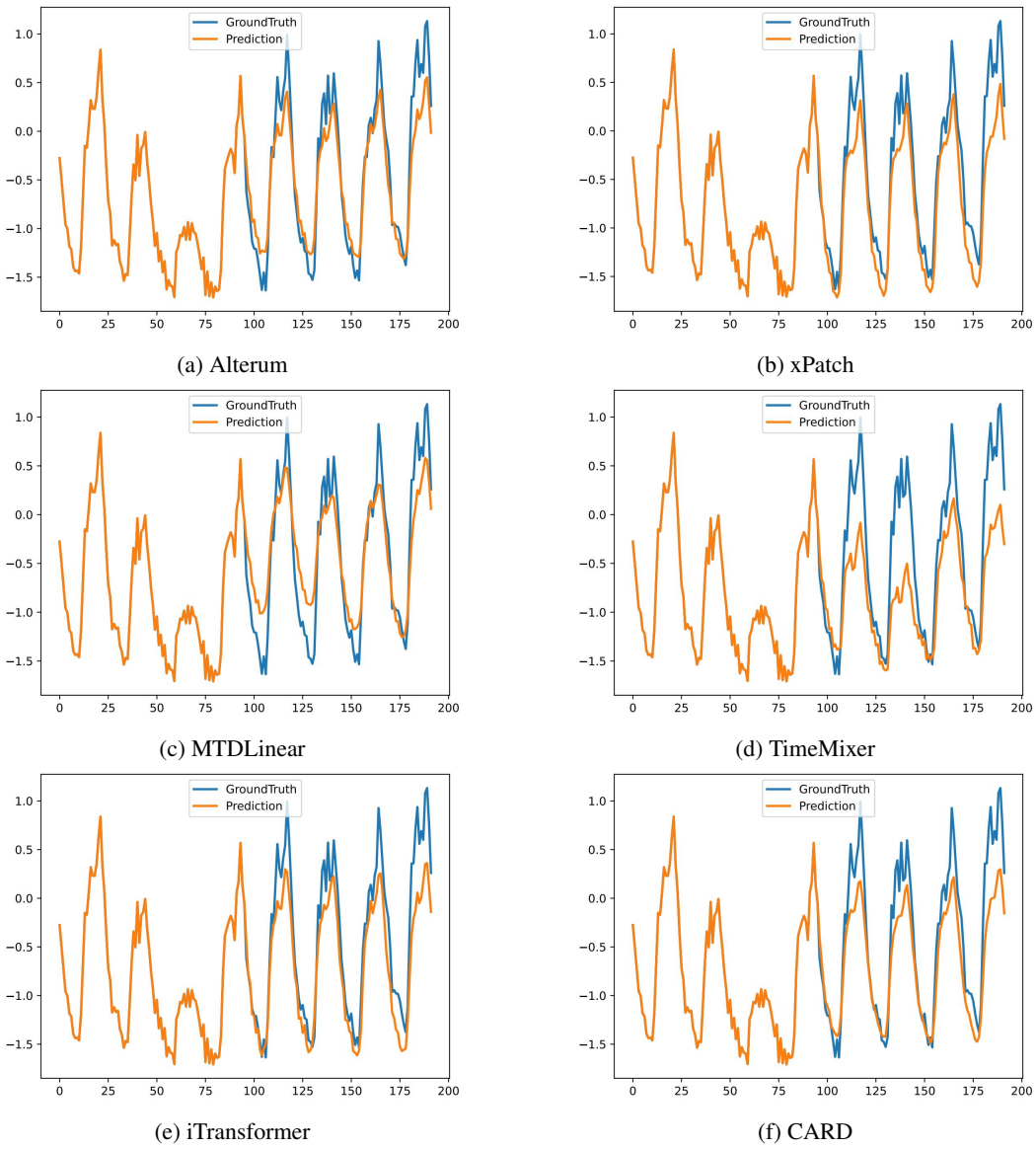
K.1 Proof of Proposition 1

We show that the Gaussian mixture $p_{\mathrm{FFE}}(\omega) = \frac{1}{N}\sum_{n=0}^{N-1} p_{\sigma_n}(\omega)$ with $p_{\sigma_n}(\omega) = \frac{1}{\sqrt{2\pi}\sigma_n} e^{-\omega^2/(2\sigma_n^2)}$ and $\sigma_n = \sigma_0 e^n$ allocates a comparable probability mass to every octave band in $[\omega_{\min}, \omega_{\max}]$, provided $N \geq \ln(\omega_{\max}/\omega_{\min}) + c_0$ for an absolute constant c_0.

Fix an octave band $B = [a, 2a] \subset [\omega_{\min}, \omega_{\max}]$. Let $n^* = \lfloor \ln(a/\sigma_0) \rfloor$, so that $\sigma_{n^*} \in [a/e, a]$ and hence $\sigma_{n^*+1} = e\sigma_{n^*} \in [a, ea]$. Choose $m = n^* + 1$, so that $t \triangleq a/\sigma_m \in [1/e, 1]$; such an $m \leq N-1$ always exists; for the highest band $\sigma_m \leq 2\omega_{\max}/\ln 2 \leq N-1$, and lower bands are easier. The component p_{σ_m} contributes

$$\int_a^{2a} p_{\sigma_m}(\omega) d\omega = \Phi\left(\frac{2a}{\sigma_m}\right) - \Phi\left(\frac{a}{\sigma_m}\right) = \Phi(2t) - \Phi(t), \tag{8}$$

where Φ is the standard normal CDF. The function $h(t) = \Phi(2t) - \Phi(t)$ is increasing on $[0, t_*]$ and decreasing on $[t_*, \infty)$ with $t_* = \sqrt{\frac{2}{3}\ln 2} \approx 0.68$, so on $t \in [1/e, 1]$ it attains its minimum at an endpoint: $\min\{h(1/e), h(1)\} = \Phi(2/e) - \Phi(1/e) \approx 0.12$. Hence the matched component contributes at least $0.12/N$ to $\int_B p_{\mathrm{FFE}}$. Components with $|n - m| \leq 2$ each contribute at most a constant fraction of the band mass and at least zero; components with $n < m - 2$ have $\sigma_n \leq \sigma_{m-3} = \sigma_m/e^3 < a/e^2$, so $p_{\sigma_n}(a) \leq \frac{1}{\sqrt{2\pi}\sigma_m} e^{-(2a/\sigma_m)^2}$, which is exponentially small; components with $n > m + 2$ have $\sigma_n \geq \sigma_{m+3} = e^3\sigma_m > 2e^2 a$, so p_{σ_n} is nearly constant over $[a, 2a]$ with value $\approx 1/(\sqrt{2\pi}\sigma_n)$, and their total contribution is bounded by a convergent geometric series $\frac{a}{\sqrt{2\pi}}\sum_{n>m+2} 1/\sigma_n$, which is of constant size relative to $1/N$. Therefore $\int_B p_{\mathrm{FFE}}(\omega) d\omega \approx 1/N$, with upper and lower constants independent of



(a) Alterum (b) xPatch

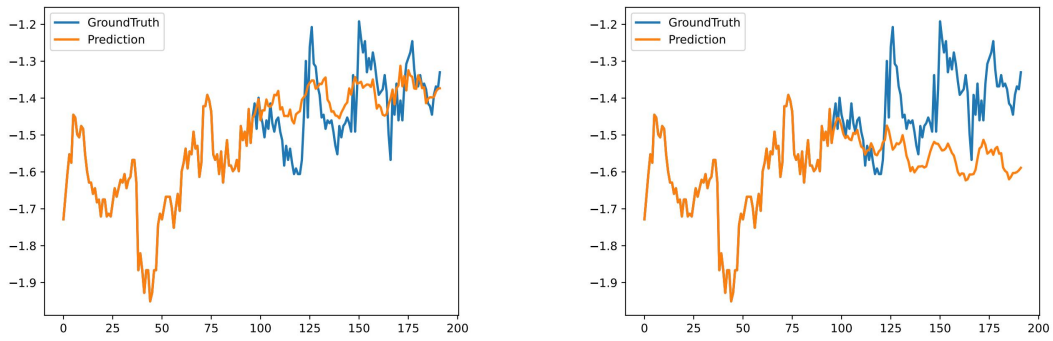
(c) MTDLinear (d) TimeMixer

(e) iTransformer (f) CARD

Figure 6: Visualization results on ECL.

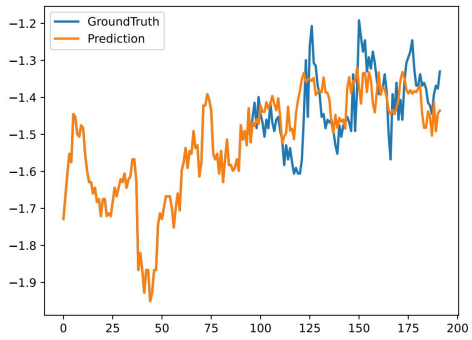
a. Since $N \geq \ln(\omega_{\max}/\omega_{\min}) = (\ln 2)N_{\text{oct}}$ (the assumed lower bound on N), we obtain $\int_B p_{\text{FFE}} d\omega \approx 1/N_{\text{oct}}$, with constants independent of the frequency range. ■

Numerical verification of Proposition 1. We verify the log-uniform coverage claim numerically for the FFE configuration used in our experiments: $N = 32$ scales, $\sigma_n = \sigma_0 e^n$ ($n = 0, \ldots, 31$), and frequency range $[\omega_{\min}, \omega_{\max}] = [1, e^{31}]$ with $N_{\text{oct}} = \log_2(\omega_{\max}/\omega_{\min}) \approx 44.7$. Table 16 reports, for every octave band $[2^k, 2^{k+1}]$, the rescaled mass $M_k = \int_{2^k}^{2^{k+1}} p_{\text{FFE}}(\omega) d\omega$ and its deviation from the rescaling $M_k \cdot N_{\text{oct}} = 1/2$ (half of the symmetric mixture total mass lies on the positive axis). Concrete values, $M_k = \frac{1}{N}\sum_{n=0}^{N-1}[\Phi(2^{k+1}/\sigma_n) - \Phi(2^k/\sigma_n)]$, evaluated in closed form via the standard normal CDF Φ. The rescaled mass lies in $[0.479, 0.488]$ across the 40 lowest bands, which is within $\pm 5\%$ of the ideal value, and decays smoothly to 0.387 at the highest band ($k = 43$) as the largest scale σ_{31} is approached; the max-to-min ratio of the rescaled mass over all 44 bands is 1.26. These numerical constants are substantially tighter than the analytical ones in the proof above (lower constant > 0.12, upper constant $O(1)$), confirming that every octave band receives a comparable fraction of the FFE feature budget.

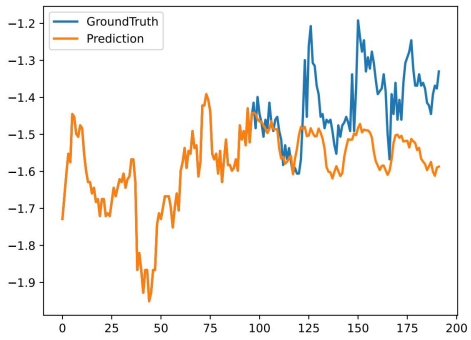


(a) Alterum

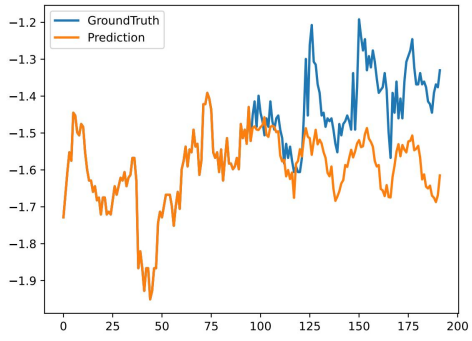
(b) xPatch



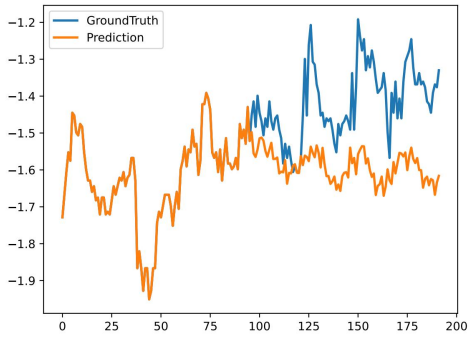
(c) MTDLinear



(d) TimeMixer



(e) iTransformer



(f) CARD

Figure 7: Visualization results on ETTh1.

K.2 Approximation Error: Conditional Bound and Discussion

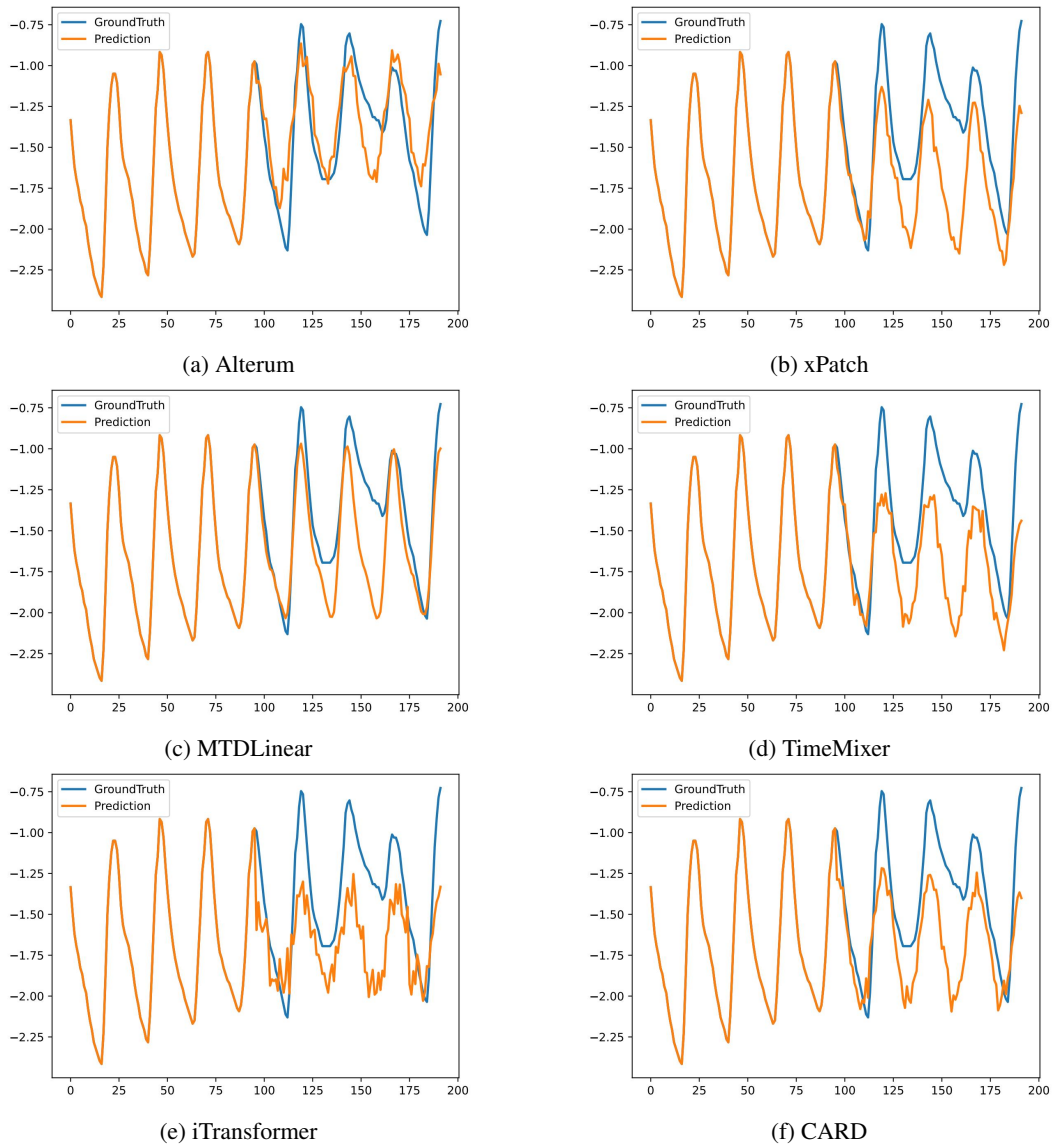
We now discuss the implication of log-uniform coverage for approximation error. Throughout, let f have support in $[\omega_{\min}, \omega_{\max}]$ and satisfy: (A1) finite total energy $E_{\text{total}} = \int_{\omega_{\min}}^{\omega_{\max}} |\hat{f}(\omega)|^2 d\omega$; (A2) equal energy per octave band, $E_n = E_{\text{total}}/N_{\text{oct}}$; (A3) smooth in-band spectra (the spectral density varies by at most a constant factor within each band).

Conditional logarithmic bound. If one additionally assumes that the per-octave-band approximation error with m_n features satisfies the bound

$$\varepsilon_n \leq C \cdot \frac{E_n}{m_n}, \tag{9}$$

then Proposition 1 implies $m_n = d/N_{\text{oct}}$ for every band, and the total error satisfies

$$\mathbb{E}_{\mathbf{B}}\left[\min_w \|f - w^\top \gamma_d\|_{L^2}^2\right] \leq \sum_{n=1}^{N_{\text{oct}}} \varepsilon_n \leq N_{\text{oct}} \cdot C \cdot \frac{E_{\text{total}}}{d/N_{\text{oct}}} = C' \cdot \frac{E_{\text{total}} \cdot N_{\text{oct}}}{d} = \frac{E_{\text{total}}}{d} \log\left(\frac{\omega_{\max}}{\omega_{\min}}\right), \tag{10}$$



(a) Alterum (b) xPatch

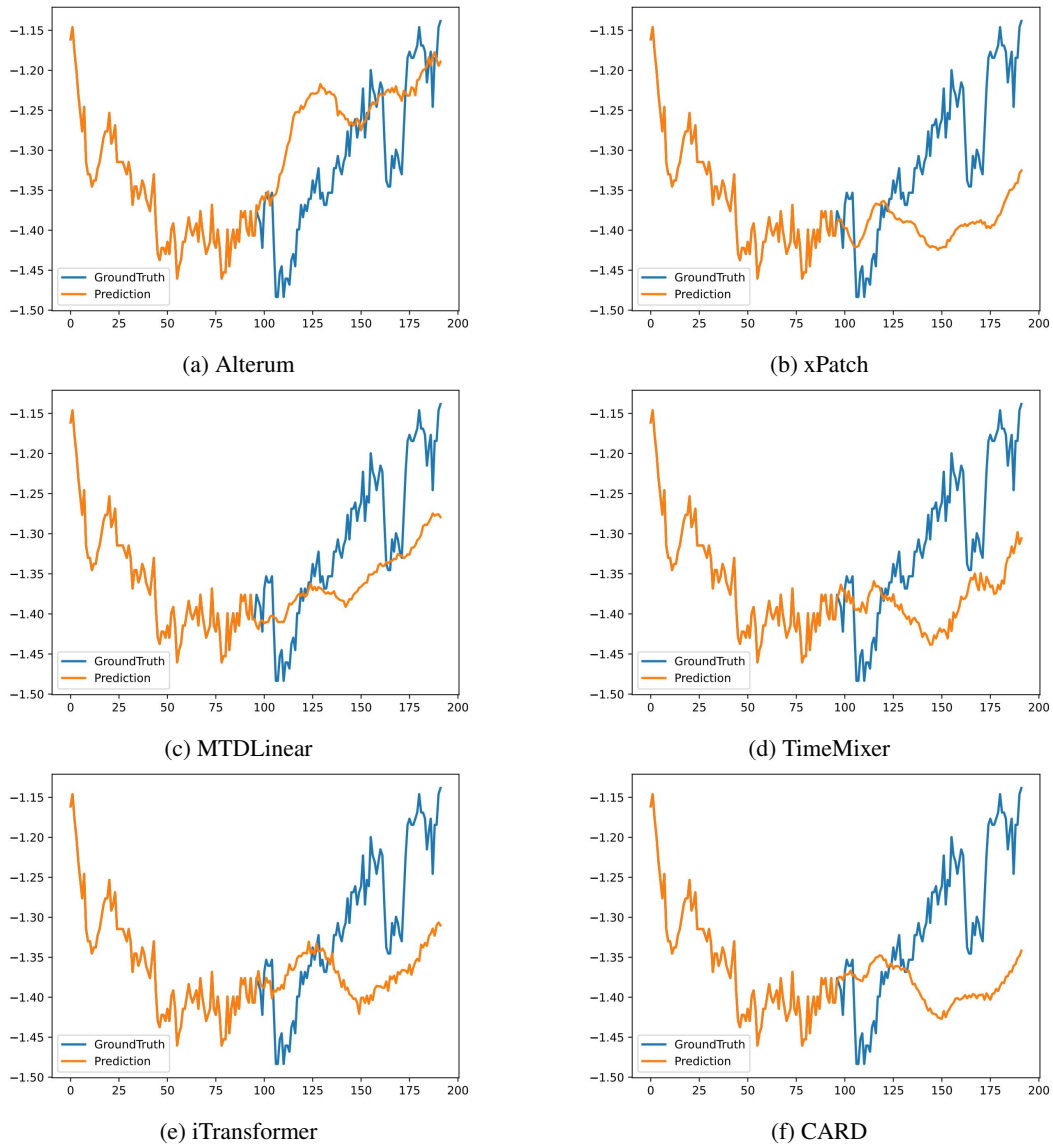
(c) MTDLinear (d) TimeMixer

(e) iTransformer (f) CARD

Figure 8: Visualization results on ETTh2.

where the first inequality uses the orthogonality of disjoint frequency bands (Parseval) and the factors w to allocate each scale's features to its corresponding band, and the last step uses $N_{\text{oct}} = \log_2(\omega_{\max}/\omega_{\min})$. The underband $B_1 = [\omega_{\min}, 2\omega_{\min})$ receives only $m_1^{\text{unif}} = d \cdot (\omega_{\max}/\omega_{\min})/(\omega_{\max} - \omega_{\min}) \approx 1/\omega_{\max}$ features; summing the per-band errors under (A4) gives an upper bound of $O\left(\frac{E_{\text{total}}}{d} \cdot \frac{\omega_{\max}/\omega_{\min}}{\log(\omega_{\max}/\omega_{\min})}\right)$, which is substantially worse than the logarithmic bound for large frequency ranges.

Why (A4) is stronger than the standard RFF bound. The classical RFF bound is $\mathbb{E}\left[\min_w \|f_n - w^\top \gamma_d\|^2_{H_{\omega_n}}\right] \le \|\hat{f}_n\|^2_{H_{\omega_n}}/m$. For the Gaussian kernel k_{σ_n}, the spectral density satisfies $p_{\sigma_n}(\omega) \asymp 1/\sigma_n$ within the matched band, so the RKHS norm is $\|\hat{f}_n\|^2_{H_{\omega_n}} = \int_\omega \hat{f}_n(\omega)^2/p_{\sigma_n}(\omega)d\omega \asymp \sigma_n E_n$. Since σ_n grows exponentially, the worst-case RFF bound is dominated by the highest-frequency bands and scales linearly with $\omega_{\max}$. Assumption (A4) is therefore strictly stronger than the standard worst-case bound; it is expected to hold in regimes where the feature distribution is well-matched to the target spectrum (importance-sampling-type regimes) and the in-band spectrum



(a) Alterum (b) xPatch

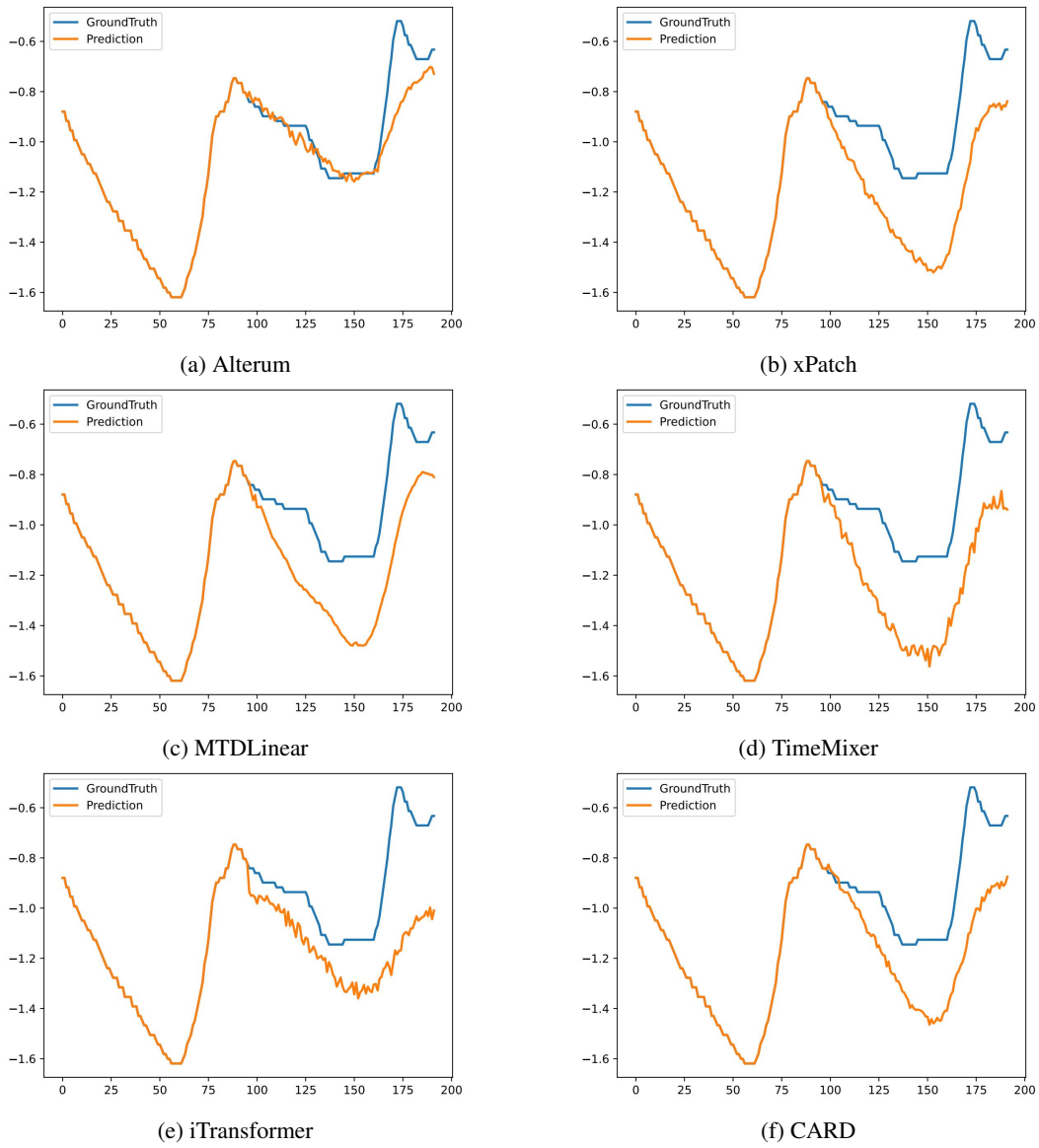
(c) MTDLinear (d) TimeMixer

(e) iTransformer (f) CARD

Figure 9: Visualization results on ETTm1.

is smooth (A3). We do not claim (A4) as a theorem; it is an idealized assumption under which the logarithmic rate is transparent.

Scale invariance of the standard RFF bound. The form $\|\hat{f}_n\|^2_{H_{\omega_n}}/m$ assumes a normalized kernel with $\kappa^2 = \sup_r k(r,r) = 1$; the general bound is $\kappa^2\|\hat{f}_n\|^2_{H_{\omega_n}}/m$. Scaling the features of scale n by any constant c (e.g., $c = \sqrt{\sigma_n/m_n}$) which turns the expected kernel into $\sigma_n k_{\sigma_n}$ rescales the kernel to $c^2 k_n$; this multiplies κ^2 by c^2 and divides $\|\hat{f}_n\|^2_{H_{\omega_n}}$ by c^2, leaving the product $\kappa^2\|\hat{f}_n\|^2_{H_{\omega_n}}$ and hence the bound unchanged. Consequently, no constant feature normalization can remove the frequency-scale factor σ_n from the RKHS norm: the worst-case bound is insensitive to the spectral coverage of the sampling distribution, which is precisely the property that distinguishes FFE from uniform sampling. A rigorous logarithmic bound therefore requires importance-sampling and spectral-matching analyses of random features that go beyond the standard Monte-Carlo bound (Avron et al., 2017; Li et al., 2019; Sriperumbudur & Szabó, 2015); we leave this as future work.



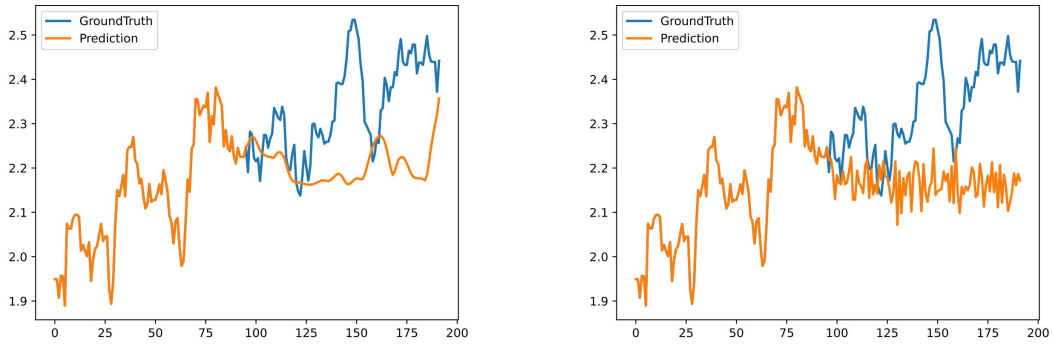
(a) Alterum (b) xPatch

(c) MTDLinear (d) TimeMixer

(e) iTransformer (f) CARD

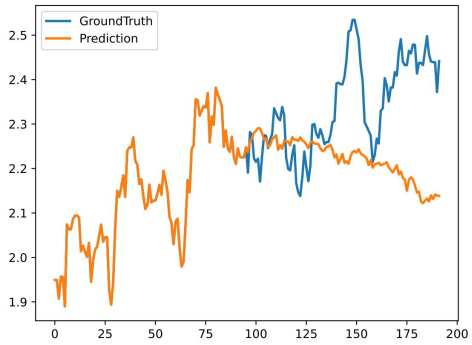
Figure 10: Visualization results on ETTm2.

Why practical performance exceeds the worst-case bound. Alterum does not rely on the worst-case RFF bound. The representation is trained end-to-end with a closed-form ridge regression head, and the effective rank analysis (Appendix L) shows that FFE+TDGN features concentrate energy in a far smaller number of principal components ($r_{90\%} = 8.9$ on average), so the effective dimension is far smaller than the nominal d. The log-uniform coverage (Proposition 1) ensures that this small effective budget is distributed across all frequency bands rather than concentrated in low frequencies, which is the key structural advantage over uniform sampling and manually-scaled features. We establish the necessary coverage condition under which (A4) can hold, and the low effective rank reported in this work indicates that the actual feature covariance is far more concentrated than the worst-case RFF bound assumes, consistent with (A4) being satisfied in practice. A direct empirical test of (A4), measuring per-band approximation errors on synthetic spectra, remains a natural next step; the spectral decay analysis in Appendix K.3 provides indirect empirical support. See Appendix K.3 for a mechanistic derivation of the band-level error decomposition and empirical validation on all eight datasets.

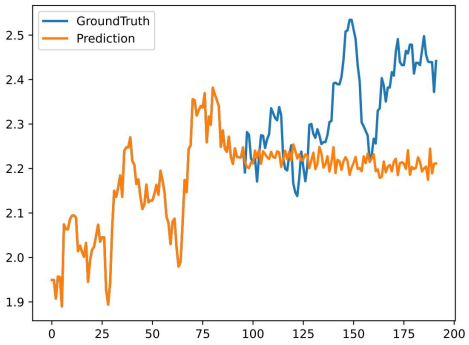


(a) Alterum

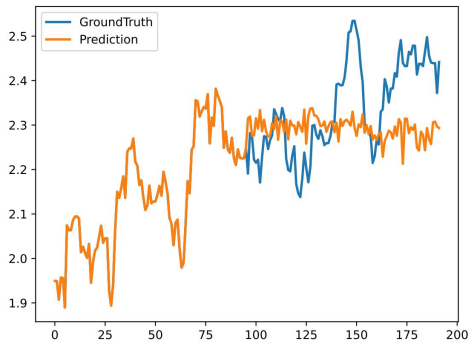
(b) xPatch



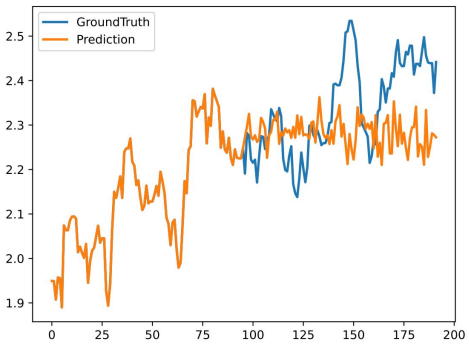
(c) MTDLinear



(d) TimeMixer



(e) iTransformer



(f) CARD

Figure 11: Visualization results on Exchange.

K.3 Empirical Support for Assumption (A4)

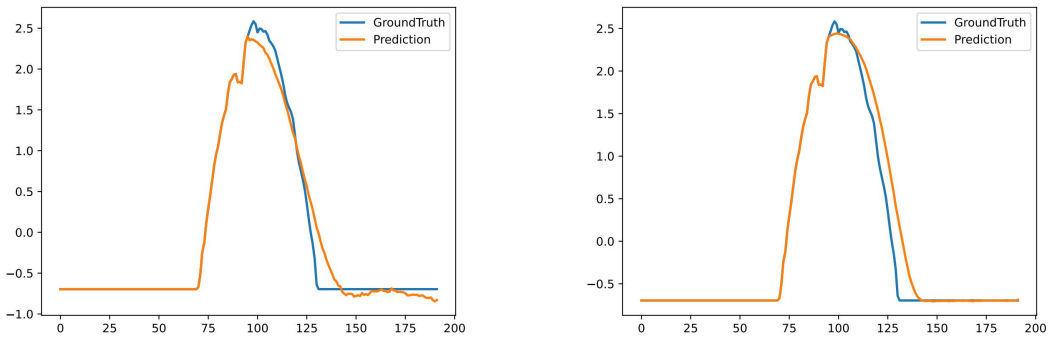
We provide a mechanistic derivation of the per-band error decomposition and an empirical estimate of the spectral decay exponent on all eight datasets.

Mechanistic error decomposition. In our implementation, each scale n contributes exactly $m_n = d_{\text{Fourier}}/(2N)$ frequencies drawn from $N(0, e^{2n})$. The best linear combination of the corresponding cosine/sine basis functions in octave band $B_n = [2^n, 2^{n+1}]$ has squared error

$$\varepsilon_n = E_n - \sum_{k=1}^{m_n} |\hat{f}(\omega_{n,k})|^2 \Delta_{n,k}, \tag{11}$$

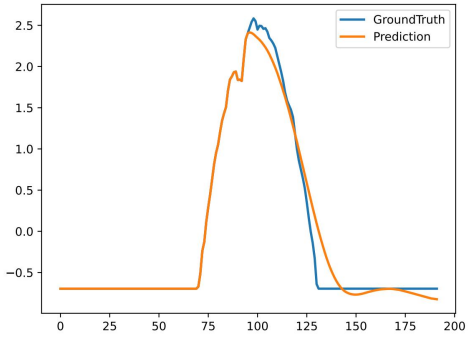
where $E_n = \int_{B_n} |\hat{f}(\omega)|^2 d\omega$. Taking expectation over random sampling, and using $p_{\text{FFE}}(\omega) \approx C/(Ne^n)$ constant within each band:

$$\varepsilon_n \approx E_n \cdot \min\left(1, \frac{m_n}{e^n}\right). \tag{12}$$

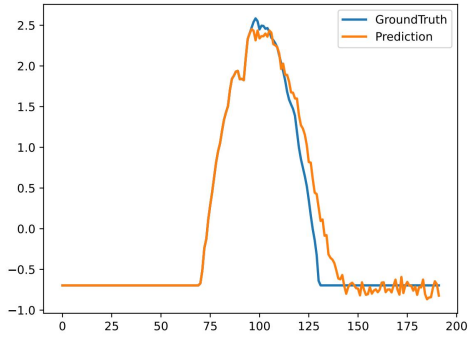


(a) Alterum

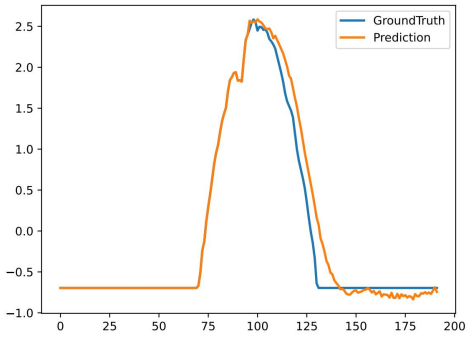
(b) xPatch



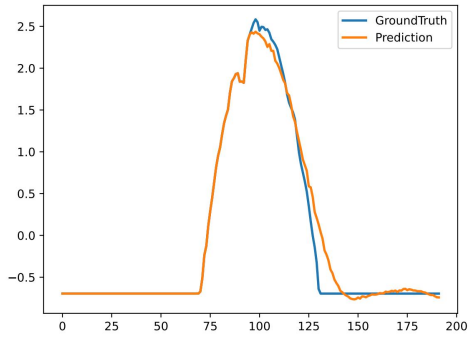
(c) MTDLinear



(d) TimeMixer



(e) iTransformer



(f) CARD

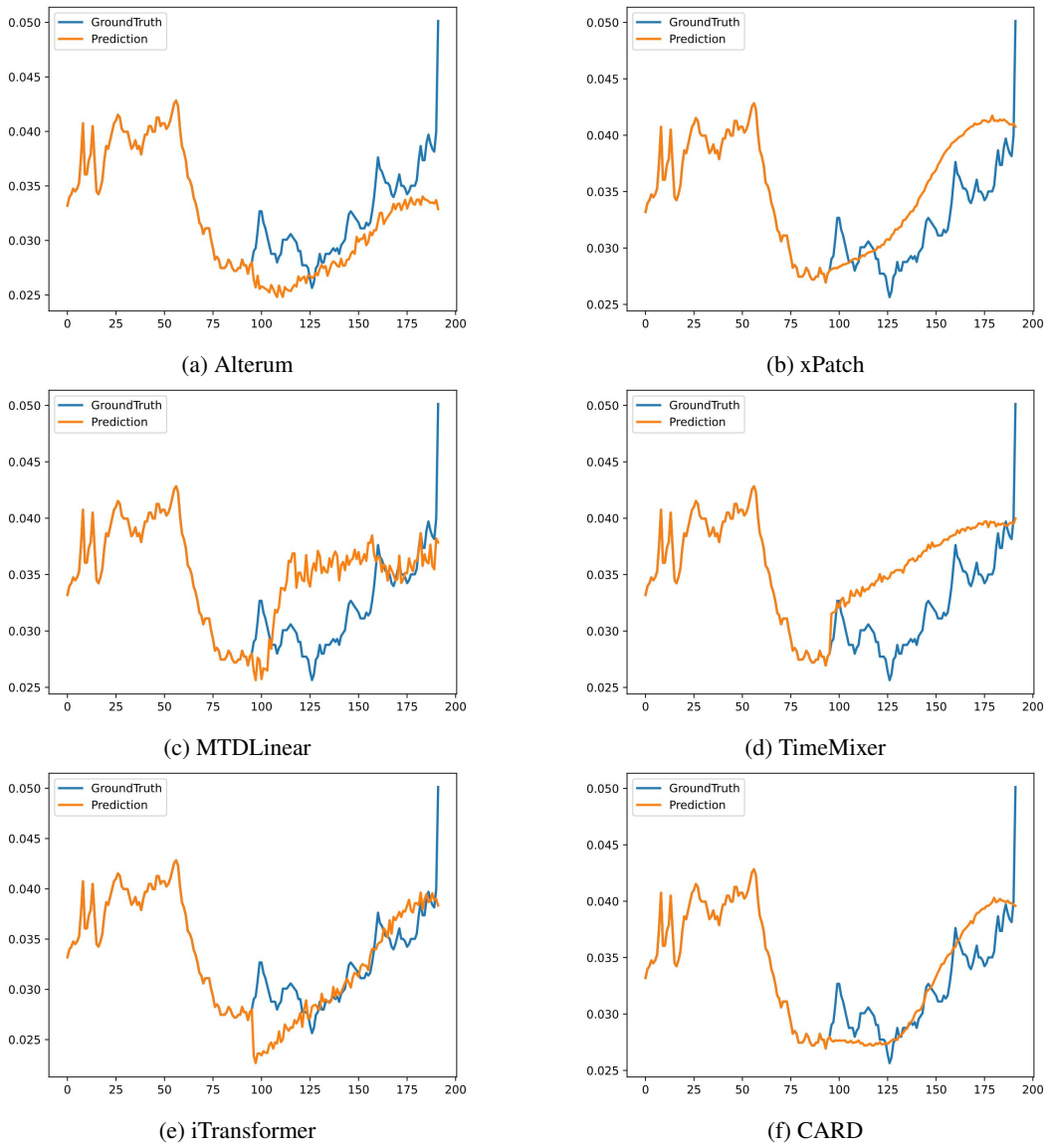
Figure 12: Visualization results on Solar.

Low-frequency bands are fully sampled ($\varepsilon_n \approx E_n$); high-frequency bands are undersampled ($\varepsilon_n \approx E_n \cdot m_n/e^n$). Summing over bands with $E_n \propto e^{-\gamma n}$:

$$\sum_n \varepsilon_n = O(1) + O\left(\frac{N_{\text{oct}}^{\gamma}}{d^{\gamma}}\right), \tag{13}$$

which grows slower than A4's $O(N_{\text{oct}}/d)$ when $\gamma < 1$.

Threshold effect and effective rank. The above decomposition assumes uniform spectral density within each band. In practice, band energy is concentrated in a small number of dominant frequencies. We verify this on synthetic signals with power-law spectra: for each octave band, we compute the minimum number of features m^* required to reduce the normalized error ε_n/E_n below 10%. Table 17 reports m^* alongside the band's effective rank r_{90} (number of dominant frequencies capturing 90% of band energy). The two are closely matched: once $m_n \geq r_{90}$, the error drops sharply rather than decaying linearly. This threshold behavior is tighter than A4's linear E_n/m_n upper bound; A4 is therefore a conservative sufficient condition, while the actual error decays as a step function of



(a) Alterum (b) xPatch

(c) MTDLinear (d) TimeMixer

(e) iTransformer (f) CARD

Figure 13: Visualization results on Weather.

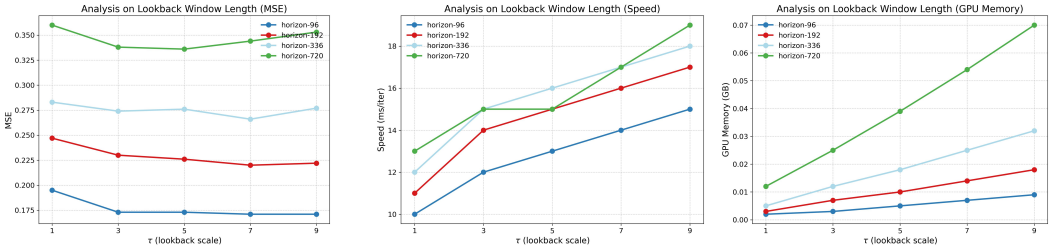
m_n/r_{90}. This is consistent with the low effective rank reported in Appendix L ($r_{90\%} = 8.9$ on real data).

Estimation of β. For each dataset, we estimate the power spectral density $S^{(c)}(\omega_j)$ of channel c using Welch's method with segment length L_{seg} and a Hann window. We then fit a power law in log-log coordinates via ordinary least squares:

$$\log S^{(c)}(\omega_j) = -\beta^{(c)} \log \omega_j + a^{(c)}, \tag{14}$$

where $\beta^{(c)} = \frac{\sum_j (\log \omega_j - \overline{\log \omega_j})(\log S^{(c)}(\omega_j) - \overline{\log S^{(c)}})}{\sum_j (\log \omega_j - \overline{\log \omega_j})^2}$. We take the median across channels: $\beta = \text{median}\{\beta^{(c)}\}$. The band-energy decay exponent is $\gamma = (\beta - 1)\ln 2$.

All eight datasets satisfy $\gamma < 1$, with β ranging from 0.99 (ECL, near $1/f$ noise) to 2.19 (Solar, near Brownian noise). The estimates stabilize for $L_{\text{seg}} \geq 512$; smaller windows yield slightly higher β due to limited low-frequency resolution (Table 18). Combined with the threshold effect in Table 17, where $m^* \approx r_{90}$ rather than decaying linearly, these results confirm that A4 is a conservative sufficient



(a) MSE (b) Training speed (c) GPU memory

Figure 14: Effect of lookback window length on ETTm2 dataset.

Table 16: Numerical verification of log-uniform coverage. For every octave band $[2^k, 2^{k+1}] \subset [1, e^{31}]$, we report the mass $M_k = \int_{2^k}^{2^{k+1}} p_{\mathrm{FFE}}(\omega) d\omega$ and the total mass $N_{\mathrm{oct}} \cdot M_k$. Under exact log-uniformity on the positive axis, $N_{\mathrm{oct}} \cdot M_k = 1/2$ (the total positive-axis mass of the symmetric, zero-mean Gaussian mixture is $1/2$); the measured values lie in $[0.479, 0.488]$ for the 40 lowest bands and decay smoothly to 0.387 at the highest band $k = 43$ as the largest scale σ_{31} is approached, an edge effect of the finite scale range, so every band receives a comparable fraction of the feature budget (max/min ratio ≈ 1.26). Settings: $N = 32$, $\sigma_n = e^n$, $N_{\mathrm{oct}} = \log_2 e^{31} \approx 44.7$.

k	M_k	$N_{\mathrm{oct}} M_k$	k	M_k	$N_{\mathrm{oct}} M_k$	k	M_k	$N_{\mathrm{oct}} M_k$
0	0.01080	0.483	15	0.01079	0.483	30	0.01081	0.484
1	0.01077	0.482	16	0.01091	0.488	31	0.01076	0.481
2	0.01080	0.483	17	0.01081	0.483	32	0.01089	0.487
3	0.01091	0.488	18	0.01076	0.481	33	0.01085	0.485
4	0.01080	0.483	19	0.01090	0.488	34	0.01074	0.480
5	0.01077	0.481	20	0.01085	0.485	35	0.01085	0.485
6	0.01090	0.488	21	0.01075	0.481	36	0.01087	0.486
7	0.01084	0.485	22	0.01087	0.486	37	0.01072	0.480
8	0.01075	0.481	23	0.01089	0.487	38	0.01075	0.481
9	0.01087	0.486	24	0.01075	0.481	39	0.01077	0.482
10	0.01088	0.487	25	0.01083	0.484	40	0.01051	0.470
11	0.01075	0.481	26	0.01091	0.488	41	0.01023	0.458
12	0.01083	0.485	27	0.01078	0.482	42	0.00982	0.439
13	0.01091	0.488	28	0.01079	0.482	43	0.00864	0.387
14	0.01077	0.482	29	0.01091	0.488			

condition: real-world series have $\gamma < 1$ and low effective rank, so their actual error decays faster than the logarithmic bound.

Table 17: Threshold effect on synthetic power-law signals. r_{90} is the band effective rank, N_{bin} is the number of frequency bins in the band, and m^* is the minimum features needed to reduce ε_n / E_n below 10%. $m^* \approx r_{90}$, confirming a sharp threshold rather than linear decay.

β	Band	r_{90}	N_{bin}	m^*
1.0	[4, 8]	4	4	6
1.0	[8, 16]	7	8	11
1.0	[16, 32]	14	16	21
2.0	[8, 16]	7	8	10
2.0	[16, 32]	14	16	18
3.0	[16, 32]	12	16	22

Table 18: Spectral decay exponent β, band-energy exponent $\gamma = (\beta - 1)\ln 2$, and robustness std across nine window lengths. All datasets satisfy $\gamma < 1$. See text for estimation details.

Dataset	β	R^2	γ	std(β)
ETTh1	1.33	0.75	0.23	0.054
ETTh2	1.22	0.81	0.15	0.028
ETTm1	1.30	0.86	0.21	0.148
ETTm2	1.29	0.92	0.20	0.043
ECL	0.99	0.59	-0.01	0.232
Exchange	1.64	0.79	0.44	0.022
Solar	2.19	0.93	0.83	0.216
Weather	2.13	0.97	0.78	0.045

L Effective Rank of the Alterum Features

Tables 20, 21, 22, 23, 24, 25, 26, and 27 report per-horizon results for all four feature learners on each dataset, with columns H, $D = 2h$, d, N, Params (Section L), λ^*, r, $r_{90\%}$, $r_{95\%}$, and MSE; $r_{90\%}$ and $r_{95\%}$ are defined below.

Model configurations and parameter counts. For each dataset and horizon we use the configuration (d, h, N) reported in the following tables, where d is the FFE output dimension (equivalently, the TDGN input dimension), h the TDGN hidden dimension, and N the number of FFE scales. The FFE is parameter-free; the number of trainable parameters is computed from the closed form $3dh + 2h^2 + 5h + 1$, comprising the TDGN linear layers and the single ridge-head regularization scalar. Because the configurations vary across horizons, the parameter count is horizon-specific (reported per row in the tables below). Averaged over the eight datasets and four horizons, the model has 10,329 trainable parameters (≈10K), the budget reported in Table 5: at the $H = 720$ configuration the mean over the eight datasets is 8,781 (≈9K), the budget used in the efficiency comparison (Table 6). The Raw MLP hidden width is matched per dataset and horizon to the corresponding parameter count.

A key structural property of Alterum is whether its representation learner concentrates its feature energy on a small number of principal components. We quantify this as follows. After training on a given dataset, we compute the centered feature covariance over the validation set, the distribution on which the ridge head is evaluated:

$$\hat{\Sigma} = \frac{1}{|\mathscr{D}_{\text{val}}|} \sum_{\tau \in \mathscr{D}_{\text{val}}} \big(h_\phi(\tau) - \bar{h}\big)\big(h_\phi(\tau) - \bar{h}\big)^\top, \quad \bar{h} = \frac{1}{|\mathscr{D}_{\text{val}}|} \sum_{\tau \in \mathscr{D}_{\text{val}}} h_\phi(\tau), \tag{15}$$

where h_ϕ is the FFE+TDGN representation learner and $\mathscr{D}_{\text{val}}$ the validation set (the feature dimension is denoted $D = 2h$ throughout). We then perform an eigencomposition $\hat{\Sigma} = \sum_{i=1}^{D} \hat{\mu}_i v_i v_i^\top$ with $\hat{\mu}_1 \geq \ldots \geq \hat{\mu}_D$, and report, for each dataset and each horizon, the following quantities: (i) the effective dimension of the regularized covariance, $r = \text{tr}(\hat{\Sigma} + \lambda^* I)^{-1}$, evaluated at the *trained* regularization strength λ^* of the closed-form ridge head (the softplus-activated scalar parameter of the ridge regressor, learned jointly with the TDGN); (ii) the number of principal components capturing 90% and 95% of the total feature energy, $r_{p\%} = \min\{r : E(r) \geq p/100\}$ with $E(r) = \sum_{i=1}^{r} \hat{\mu}_i / \sum_{i=1}^{D} \hat{\mu}_i$, which are scale-invariant and independent of λ^*; and (iii) the multivariate average MSE of Alterum for reference. Because the feature dimension $D = 2h$ varies with the horizon configuration, Tables 20, 21, 22, 23, 24, 25, 26, and 27 report, for each dataset and horizon, all four feature learners, including D, the trained regularization strength λ^*, r, $r_{90\%}$, $r_{95\%}$, and MSE. Table 5 reports the effective rank of the four feature learners alongside the controlled comparison of Section 4.2.4. Since the feature map is not normalized, the absolute scale of $\hat{\Sigma}$, and hence the magnitude of r, depends on the feature scale; the scale-invariant ratios $r_{90\%}/r_{95\%}$ are therefore the primary indicators of energy concentration, whereas r at the trained λ^* reflects the regularization balance used by the model.

Structural interpretation of the measured effective ranks. The measured effective ranks in the detailed variant tables (Tables 20, 21, 22, 23, 24, 25, 26, and 27) and Table 5 admit a qualitative

structural explanation. First, the TDGN output dimension is exactly $D = 2h$ (the concatenation of the h-dimensional seasonal and trend branches), so the effective rank is unconditionally bounded by

$$r = \mathrm{tr}\left(\hat{\Sigma}(\hat{\Sigma} + \lambda^* I)^{-1}\right) \leq D = 2h. \tag{16}$$

Second, the moving-average decomposition is a low-pass filter, so the trend branch is approximately low-rank and contributes at most one dominant direction; and the seasonal branch, which maps the FFE features through a linear layer of width h, contributes at most h directions. If in addition the trend branch is exactly rank-one, these constraints yield the heuristic upper bound $r \leq 1 + \min(h, 2N)$, where $2N$ is the number of sine/cosine features produced by the N FFE scales. We emphasize that this bound is a heuristic rather than a theorem: the exact effective rank also depends on the feature covariance spectrum, the learned TDGN weights, the data distribution, and the regularization strength λ^*, none of which can be derived analytically. Consistent with this expectation, the measured $r_{90\%}$ never exceeds 31 across all 32 dataset-horizon configurations (per-dataset mean $r_{90\%} \leq 16.3$), and the effective dimension at the trained λ^* is substantially below D on most configurations (e.g., $\bar{r} \leq 4.3$ on ETTh1, ETTh2, ETTm1, Exchange, and Weather), r approaches D because the trained λ^* is small relative to the spectrum, which is why we use the scale-invariant $r_{90\%}/r_{95\%}$ as the primary indicators. A rapid saturation of $E(r)$ implies that a few directions carry most of the predictive signal, which, together with the closed-form ridge head, is associated with stable few-shot estimates. Overall, the measurements corroborate the expectation that FFE+TDGN produces an energy-concentrated representation, a structural property inherited from the logarithmic spectral coverage of FFE and the decomposition of TDGN, rather than an arbitrary fixed dataset.

In the controlled comparison (Section 4.2.4), the Raw MLP's mean effective dimension $\bar{r} = 0.543$ is below one, indicating that the representation collapses to roughly a single effective direction.

Table 19: Robustness of β estimates across nine Welch window lengths. Smaller windows (64–256) yield slightly higher β due to limited low-frequency resolution; the estimates stabilize for $L_{\text{seg}} \geq 512$.

Dataset	64	128	256	512	1024	2048	4096	8192	16384
ETTh1	1.45	1.34	1.29	1.27	1.27	1.28	1.33	1.30	1.27
ETTh2	1.29	1.27	1.25	1.22	1.23	1.25	1.22	1.23	1.30
ETTm1	1.70	1.64	1.53	1.42	1.36	1.32	1.30	1.29	1.28
ETTm2	1.39	1.38	1.35	1.32	1.29	1.32	1.29	1.26	1.28
ECL	1.71	1.40	1.24	1.14	1.06	1.01	0.99	0.98	0.99
Exchange	1.70	1.69	1.69	1.69	1.66	1.70	1.64	–	–
Solar	2.75	2.65	2.56	2.41	2.31	2.24	2.19	2.15	2.13
Weather	2.25	2.22	2.18	2.16	2.16	2.14	2.13	2.11	2.10

M Unified Paradigm of Selective Associative Memory

In Section 2.3, we propose selective associative memory (SAM) as a unified framework that reveals the common principle behind gated RNNs, attention mechanisms, gated CNNs/MLPs, selective SSMs, and MoE architectures. This appendix derives the mathematical formalization of this framework, showing that the unifying principle can be distilled into a compact paradigm structure under which the above diverse architecture families can be viewed as instances.

M.1 From Selective Associative Memory to the Unified Paradigm

The core proposition of the selective associative memory framework is that all gating mechanisms contain two basic components: a memory component responsible for information storage and retrieval, and a selection component responsible for information selection and regulation. Based on this intuition, we can formalize it into the following unified paradigm:

$$f(\mathbf{x}) = \mathrm{Memory}(\mathbf{x}) \odot \mathrm{Selection}(\mathbf{x}) \tag{17}$$

where:

Table 20: Detailed per-horizon results for all four feature learners on ECL.

H	D	d	N	Params	Feature Learner	λ^*	r	$r_{90\%}$	$r_{95\%}$	MSE
96	128	256	32	57,665	FFE + TDGN (Ours)	1.5	12.2	31	52	0.134
					CFF + TDGN	1.3	31.6	16	24	0.138
					Pos. Enc. + TDGN	0.318	6.2	2	2	1.517
					Raw MLP	1.3	1.0	1	1	0.830
192	128	256	32	57,665	FFE + TDGN (Ours)	9.2	112.7	24	39	0.147
					CFF + TDGN	3.0	102.5	15	22	0.159
					Pos. Enc. + TDGN	0.791	8.6	2	2	1.040
					Raw MLP	0.746	1.2	1	1	0.877
336	128	256	32	57,665	FFE + TDGN (Ours)	10.6	79.7	4	12	0.165
					CFF + TDGN	3.5	98.2	15	22	0.171
					Pos. Enc. + TDGN	0.107	11.8	1	1	0.756
					Raw MLP	0.948	1.4	1	1	0.851
720	64	512	32	51,361	FFE + TDGN (Ours)	3.2	11.8	6	8	0.205
					CFF + TDGN	1.4	39.5	4	6	0.217
					Pos. Enc. + TDGN	1.1	3.8	2	2	0.864
					Raw MLP	2.8	0.701	1	1	0.873

Table 21: Detailed per-horizon results for all four feature learners on ETTh1.

H	D	d	N	Params	Feature Learner	λ^*	r	$r_{90\%}$	$r_{95\%}$	MSE
96	16	128	16	3,241	FFE + TDGN (Ours)	0.849	1.3	9	11	0.367
					CFF + TDGN	0.781	1.5	3	5	0.386
					Pos. Enc. + TDGN	0.764	0.781	1	1	0.692
					Raw MLP	1.1	0.196	1	2	0.698
192	16	128	16	3,241	FFE + TDGN (Ours)	4.3	5.9	6	9	0.403
					CFF + TDGN	1.9	5.8	3	3	0.459
					Pos. Enc. + TDGN	1.4	3.3	1	1	0.709
					Raw MLP	0.997	1.0	1	1	0.708
336	32	128	16	6,737	FFE + TDGN (Ours)	0.770	1.7	19	23	0.439
					CFF + TDGN	0.762	1.0	14	19	0.466
					Pos. Enc. + TDGN	1.0	1.1	1	2	0.716
					Raw MLP	0.956	0.174	2	2	0.703
720	16	256	16	6,313	FFE + TDGN (Ours)	0.863	0.793	8	10	0.464
					CFF + TDGN	1.1	1.9	5	6	0.556
					Pos. Enc. + TDGN	0.911	0.445	2	2	0.746
					Raw MLP	1.1	0.185	1	1	0.718

- Memory$(\cdot)$ is the memory branch, which performs information storage, transformation, and retrieval;
- Selection$(\cdot)$ is the selection branch, which generates gating weights to regulate the flow of information;
- $\odot$ is the fusion operation, which combines the memory branch and the selection branch.

Core Proposition: The differences between different architectures essentially lie in **four distinct, largely independent design dimensions**, rather than in the underlying principle. By adjusting the values of these four dimensions, we can derive various existing gating architectures from this unified paradigm.

Table 22: Detailed per-horizon results for all four feature learners on ETTh2.

H	D	d	N	Params	Feature Learner	λ^*	r	$r_{90\%}$	$r_{95\%}$	MSE
96	16	64	16	1,705	FFE + TDGN (Ours)	1.3	7.5	8	10	0.275
					CFF + TDGN	3.3	8.1	5	7	0.303
					Pos. Enc. + TDGN	1.0	2.8	2	2	0.359
					Raw MLP	1.6	0.435	1	2	0.356
192	16	64	16	1,705	FFE + TDGN (Ours)	1.1	1.5	9	11	0.332
					CFF + TDGN	1.1	2.1	6	7	0.337
					Pos. Enc. + TDGN	0.893	1.3	1	2	0.397
					Raw MLP	1.2	0.156	2	2	0.393
336	16	64	16	1,705	FFE + TDGN (Ours)	1.4	0.633	7	9	0.362
					CFF + TDGN	1.7	1.2	4	6	0.377
					Pos. Enc. + TDGN	1.2	1.1	1	1	0.487
					Raw MLP	1.5	0.184	2	2	0.401
720	16	64	16	1,705	FFE + TDGN (Ours)	1.8	1.2	8	10	0.391
					CFF + TDGN	1.1	0.849	6	8	0.398
					Pos. Enc. + TDGN	1.4	0.561	2	2	0.644
					Raw MLP	3.7	0.251	1	1	0.551

Table 23: Detailed per-horizon results for all four feature learners on ETTm1.

H	D	d	N	Params	Feature Learner	λ^*	r	$r_{90\%}$	$r_{95\%}$	MSE
96	32	128	16	6,737	FFE + TDGN (Ours)	6.9	23.7	11	15	0.301
					CFF + TDGN	6.4	27.5	5	8	0.302
					Pos. Enc. + TDGN	0.804	3.5	1	1	0.437
					Raw MLP	1.7	0.306	1	1	0.659
192	32	256	16	12,881	FFE + TDGN (Ours)	4.9	10.9	7	9	0.337
					CFF + TDGN	2.9	18.0	7	9	0.343
					Pos. Enc. + TDGN	0.794	1.5	1	1	0.664
					Raw MLP	1.8	0.337	1	1	0.683
336	32	256	32	12,881	FFE + TDGN (Ours)	2.4	6.6	9	16	0.358
					CFF + TDGN	2.6	14.1	6	8	0.362
					Pos. Enc. + TDGN	0.979	2.5	1	1	0.686
					Raw MLP	2.0	0.685	1	1	0.690
720	32	64	32	3,665	FFE + TDGN (Ours)	30.8	16.3	7	10	0.405
					CFF + TDGN	20.5	18.8	7	9	0.405
					Pos. Enc. + TDGN	6.8	1.6	1	1	0.709
					Raw MLP	0.676	1.3	1	1	0.781

M.2 Four Distinct Design Dimensions

According to the unified paradigm, we can characterize different gating mechanisms from four distinct, largely independent design dimensions.

M.2.1 Operating Space

The operating space refers to the dimension on which Memory and Selection each perform operations, determining the scope of the gating effect.

Operating space of the Selection branch:

Table 24: Detailed per-horizon results for all four feature learners on ETTm2.

H	D	d	N	Params	Feature Learner	λ^*	r	$r_{90\%}$	$r_{95\%}$	MSE
96	32	128	16	6,737	FFE + TDGN (Ours)	1.1	2.1	9	15	0.163
					CFF + TDGN	0.813	1.9	8	12	0.163
					Pos. Enc. + TDGN	0.789	2.2	2	2	0.211
					Raw MLP	1.1	0.077	1	1	0.237
192	16	64	16	1,705	FFE + TDGN (Ours)	0.902	1.2	9	11	0.217
					CFF + TDGN	1.0	0.714	6	9	0.220
					Pos. Enc. + TDGN	0.729	1.1	1	2	0.281
					Raw MLP	1.2	0.062	2	2	0.296
336	16	64	16	1,705	FFE + TDGN (Ours)	1.8	2.8	6	8	0.264
					CFF + TDGN	1.9	4.1	4	5	0.266
					Pos. Enc. + TDGN	1.0	1.7	2	2	0.321
					Raw MLP	1.9	0.125	2	2	0.329
720	8	128	16	1,589	FFE + TDGN (Ours)	1.0	0.915	3	4	0.329
					CFF + TDGN	0.867	1.7	2	2	0.332
					Pos. Enc. + TDGN	0.766	0.861	1	1	0.376
					Raw MLP	1.4	0.043	1	2	0.390

Table 25: Detailed per-horizon results for all four feature learners on Exchange.

H	D	d	N	Params	Feature Learner	λ^*	r	$r_{90\%}$	$r_{95\%}$	MSE
96	16	8	16	361	FFE + TDGN (Ours)	4.8	13.0	2	3	0.078
					CFF + TDGN	9.3	5.9	1	1	0.082
					Pos. Enc. + TDGN	12.5	4.9	1	2	0.080
					Raw MLP	2.3	3.1	1	1	0.468
192	16	8	16	361	FFE + TDGN (Ours)	0.390	3.5	3	4	0.158
					CFF + TDGN	0.914	1.2	2	2	0.159
					Pos. Enc. + TDGN	1.1	0.529	2	3	0.168
					Raw MLP	0.901	0.384	2	2	0.171
336	16	8	16	361	FFE + TDGN (Ours)	0.688	0.150	5	6	0.269
					CFF + TDGN	0.679	0.122	2	3	0.285
					Pos. Enc. + TDGN	0.678	0.113	1	2	0.307
					Raw MLP	0.713	0.074	1	1	0.268
720	16	8	16	361	FFE + TDGN (Ours)	0.705	0.232	4	5	0.537
					CFF + TDGN	0.675	0.185	2	2	0.614
					Pos. Enc. + TDGN	0.704	0.181	2	2	0.638
					Raw MLP	0.722	0.155	1	1	0.460

- **Feature dimension**: element-wise independent computation, gating each feature channel independently. Representative architectures: GLU variants (SwiGLU, GEGLU, ReGLU) (Shazeer, 2020).
- **Sequence dimension**: pairwise similarity computation, gating the correlation between each token and other tokens. Representative architectures: Attention, Linear Attention.
- **Model dimension**: expert-level sparse routing, selecting different experts for each input. Representative architectures: MoE, Switch Transformer.

Operating space of the Memory branch:

- **Feature dimension**: element-wise linear transformation, performing linear transformation in the feature space. Representative architectures: GLU variants (SwiGLU, GEGLU, ReGLU) (Shazeer, 2020).

Table 26: Detailed per-horizon results for all four feature learners on Solar.

H	D	d	N	Params	Feature Learner	λ^*	r	$r_{90\%}$	$r_{95\%}$	MSE
96	32	128	32	6,737	FFE + TDGN (Ours)	6.1	28.7	18	22	0.190
					CFF + TDGN	5.3	26.9	7	9	0.191
					Pos. Enc. + TDGN	0.283	6.2	1	1	0.328
					Raw MLP	2.8	0.049	1	2	0.729
192	32	128	16	6,737	FFE + TDGN (Ours)	5.0	30.0	13	16	0.192
					CFF + TDGN	5.1	26.5	3	4	0.192
					Pos. Enc. + TDGN	0.153	9.6	1	1	0.641
					Raw MLP	1.4	1.5	1	2	0.789
336	32	128	16	6,737	FFE + TDGN (Ours)	13.9	30.3	14	18	0.206
					CFF + TDGN	12.6	27.4	6	8	0.207
					Pos. Enc. + TDGN	0.031	5.4	1	1	0.716
					Raw MLP	3.9	0.160	1	2	0.752
720	32	64	16	3,665	FFE + TDGN (Ours)	14.7	27.6	8	11	0.219
					CFF + TDGN	13.1	23.2	8	10	0.218
					Pos. Enc. + TDGN	9.3	2.0	2	2	0.746
					Raw MLP	16.0	0.050	2	2	0.750

Table 27: Detailed per-horizon results for all four feature learners on Weather.

H	D	d	N	Params	Feature Learner	λ^*	r	$r_{90\%}$	$r_{95\%}$	MSE
96	16	64	16	1,705	FFE + TDGN (Ours)	1.3	1.5	8	10	0.168
					CFF + TDGN	1.3	1.8	5	6	0.168
					Pos. Enc. + TDGN	0.907	2.9	1	2	0.192
					Raw MLP	1.7	0.251	1	2	0.223
192	16	64	16	1,705	FFE + TDGN (Ours)	19.4	11.2	5	7	0.210
					CFF + TDGN	5.8	10.9	4	4	0.210
					Pos. Enc. + TDGN	0.871	4.5	2	2	0.240
					Raw MLP	2.7	0.975	1	2	0.262
336	8	128	32	1,589	FFE + TDGN (Ours)	19.3	2.2	2	3	0.246
					CFF + TDGN	14.5	4.2	2	3	0.245
					Pos. Enc. + TDGN	3.0	2.2	1	1	0.289
					Raw MLP	4.2	0.683	1	1	0.300
720	8	128	32	1,589	FFE + TDGN (Ours)	15.2	2.0	4	5	0.296
					CFF + TDGN	9.2	2.8	2	3	0.297
					Pos. Enc. + TDGN	3.5	1.9	1	1	0.343
					Raw MLP	3.9	0.144	2	2	0.345

- **Time dimension**: multi-timescale decomposition, decomposing the input into components of different timescales. Representative architectures: TDGN (our Temporal Decomposition Gated Network).
- **Sequence dimension**: value vector aggregation, aggregating information in the sequence space. Representative architectures: Attention, Linear Attention.
- **Model dimension**: multi-expert parallel computation, fusing multiple experts' parallel outputs. Representative architectures: MoE, Switch Transformer.

M.2.2 Branch Count

Branch count refers to how many parallel branches Memory and Selection each have, determining the parallel capacity of the model.

- **SwiGLU**: 1 memory branch, 1 selection branch (single-branch structure).
- **TDGN**: 2 memory branches (trend + seasonal), 1 selection branch (multi-memory-branch, single-selection-branch).
- **Multi-Head Attention**: H memory branches (H heads), H selection branches (H heads) (symmetric multi-branch structure).
- **MoE**: K memory branches (K experts), 1 selection branch (router) (multi-memory-branch, single-selection-branch).

M.2.3 Gating Function Type

The gating function type refers to what function the Selection branch uses to generate gating weights, determining the "softness" of the gating.

- **Element-wise gating**: continuous pointwise functions (e.g., sigmoid, SiLU, GELU, ReLU), soft gating applied independently to each feature channel. Representative architectures: GLU variants (SwiGLU, GEGLU, ReGLU).
- **Softmax gating**: $\mathrm{softmax}(x_i) = e^{x_i} / \sum_j e^{x_j}$, semi-hard gating, probability distribution. Representative architectures: Attention, MoE router.
- **Hardmax gating**: $\mathrm{hardmax}(x) = \mathbf{1}[i = \arg\max_j x_j]$, hard gating, one-hot selection. Representative architectures: original MoE (hard routing).

M.2.4 Fusion Method

The fusion method refers to how Memory and Selection are combined, determining the interaction mode between the two branches.

- **Element-wise multiplication**: $y_i = m_i \cdot s_i$, element-wise gating. Representative architectures: GLU variants (SwiGLU, GEGLU, ReGLU).
- **Weighted sum**: $y = \sum_i s_i \cdot m_i$, cross-element aggregation. Representative architectures: Attention, Linear Attention.
- **Sparse selection**: $y = \sum_{i \in \text{top-}k} s_i \cdot m_i$, sparse aggregation. Representative architectures: MoE, Switch Transformer.
- **Concatenation + projection**: $y = \mathrm{Proj}([m_1, ..., m_H])$, multi-branch concatenation followed by transformation. Representative architectures: Multi-Head Attention output projection.

M.3 Representative Architectures as Instances of the Unified Paradigm

We now show that several representative architectures are specific instances of this unified paradigm.

Instance 1: GLU

- Operating space: Feature dimension (Memory) + Feature dimension (Selection)
- Branch count: 1 memory branch, 1 selection branch
- Gating function: Element-wise gating
- Fusion method: Element-wise multiplication

$$f(\mathbf{x}) = (\mathbf{x}\mathbf{W}_1) \odot \sigma(\mathbf{x}\mathbf{W}_2), \tag{18}$$

where $\sigma(\cdot)$ may be any pointwise nonlinearity (e.g., Sigmoid, ReLU, GELU, Swish), yielding the GLU, ReGLU, GEGLU, and SwiGLU variants (Shazeer, 2020).

Instance 2: Attention

- Operating space: Sequence dimension (Memory) + Sequence dimension (Selection)
- Branch count: 1 memory branch (V), 1 selection branch ($\mathrm{QK}^\top$)
- Gating function: Softmax gating
- Fusion method: Weighted sum

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{19}$$

Instance 3: MoE

- Operating space: Model dimension (Memory) + Model dimension (Selection)
- Branch count: K memory branches (K experts), 1 selection branch (router)
- Gating function: Softmax gating
- Fusion method: Sparse selection (top-k)

$$f(\mathbf{x}) = \sum_{k\in\text{top-}k} r_k(\mathbf{x}) \cdot e_k(\mathbf{x}) \tag{20}$$

Instance 4: TDGN (Ours)

- Operating space: Time dimension (Memory) + Feature dimension (Selection)
- Branch count: 2 memory branches (trend + seasonal), 1 selection branch (gate)
- Gating function: Sigmoid gating (SiLU)
- Fusion method: Element-wise multiplication

$$f(\mathbf{x}) = [\text{season}(\mathbf{x}), \text{trend}(\mathbf{x})] \odot \sigma(\mathbf{x}) \tag{21}$$

M.4 Decoupling and Composability of Dimensions

It should be noted that the above four design dimensions are **decoupled** in the sense that they target different structural aspects of the computation (feature-level, sequence-level, model-level, and time-scale-level), and therefore exhibit **approximate orthogonality** in practice: different design decisions act on different aspects of the computation and do not directly alter each other's input distribution. Adjusting one dimension typically does not fundamentally alter the behavioral characteristics of another, although in deep stacks these aspects are coupled through layer-by-layer composition. Accordingly, design choices from different dimensions can be composed without significant mutual interference.

This decoupling means that we can stack design choices of different dimensions as building blocks to construct more complex architectures. For example, modern large language models combine feature-dimension operating space (SwiGLU in FFN), sequence-dimension operating space (self-attention), model-dimension operating space (MoE routing), and multi-branch structure (Multi-Head Attention).

M.5 Design Guidance from the Unified Paradigm

This unified paradigm reveals an important design principle: **architecture design can be formulated as making choices on four distinct, largely independent design dimensions, rather than inventing entirely new mechanisms**. For a given task, the minimal design principle is to select only the necessary dimension combinations, thereby achieving parameter efficiency while maintaining performance.

This principle also explains why different tasks require different architectural designs: for simple tasks, only feature dimension, single branch, element-wise gating, and element-wise multiplication are needed; for tasks requiring modeling long-range dependencies, sequence-dimension operating space needs to be added; for tasks requiring large-capacity models, model-dimension operating space and sparse fusion method need to be added; for tasks requiring modeling multi-timescale dependencies, time-dimension memory branches need to be added.

M.6 Relation to the Selective Associative Memory Framework

The unified paradigm proposed in this appendix provides a mathematical formalization for the selective associative memory framework introduced in Section 2.3. It further confirms our core proposition: different architecture families share a common underlying principle, and the differences between them only lie in the following four distinct, largely independent design dimensions:

1. **Operating space** (the computational structure of Memory and Selection);
2. **Branch count** (the number of parallel memory and selection branches);
3. **Gating function type** (the functional form that generates gating weights);
4. **Fusion method** (the way memory and selection branches are combined).

This formalization supports the unified framework we proposed and provides actionable design guidance for future architecture design.