How Good Can Linear Models Be for Time-Series Forecasting?

Anonymous authors
Paper under double-blind review

Abstract

Time-series forecasting research has been moving steadily toward larger architectures, from specialized transformers to general-purpose foundation models, on the assumption that capacity is what unlocks accuracy. We take the opposite position: most of the gap can be closed at far lower cost by tuning preprocessing rather than scaling models. We use Ridge regression as the testbed, since it has a closed-form solution and interpretable weights, which let the optimal hyperparameters be read off the search directly. We search over context length, local normalization, regularization, and augmentation on eight standard benchmarks and find three patterns. (1) Optimal lookback is strongly series-specific and often non-monotonic in forecast horizon, with fitted power-law exponents ranging from $+0.46$ on ETTm2 to -0.19 on Exchange and Traffic, challenging the convention that longer horizons need longer history. (2) Normalizing over a learned trailing fraction of the context, rather than its entirety, is almost universally preferred. (3) Series within the same dataset often disagree on hyperparameters; the optimal degree of cross-series sharing varies from fully shared to fully per-series. The resulting models beat prior linear forecasters on 7 of 8 benchmarks (a near-tie on the 8th), and attain lower average MSE than Transformer and MLP baselines under matched settings on seven of eight. The optimized hyperparameters also serve as a diagnostic on the data itself, revealing structures that larger models absorb silently into their learned parameters.

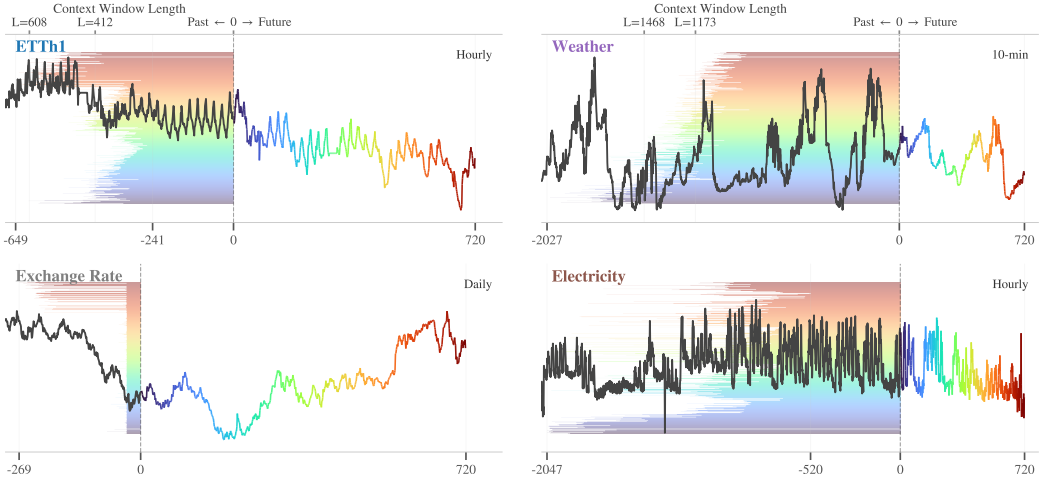

Figure 1: Per-horizon optimal context-horizon relationships for four time series. The context lengths were obtained via hyperparameter search and are shown as color-matched bars.

1 Introduction

The long-term time series forecasting literature has followed a familiar arc over the past several years. Transformers (Vaswani et al., 2017) were adapted to the task with progressively more sophisticated attention and patching mechanisms (Wu et al., 2021; Zhou et al., 2022; Nie et al., 2022; Liu et al., 2023), Multi-Layer Perceptron (MLP) and Convolutional Neural Network (CNN) alternatives demonstrated that comparable accuracy could be achieved with simpler inductive biases (Wu et al., 2022; Wang et al., 2024; Chen et al., 2023; Das et al., 2023), and linear models entered the conversation

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

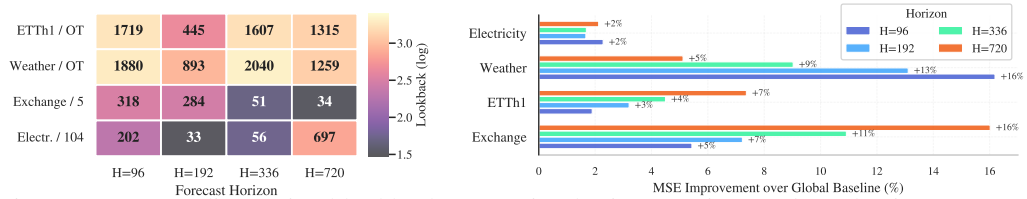

Figure 2: *Left*: Median optimal lookback per (series, horizon) varies. *Right*: Adapting context per horizon yields up to +16% MSE improvement over a global baseline across these four datasets.

when Zeng et al. (2023) showed that a single linear layer, applied after trend-seasonal decomposition, could outperform several modern transformers. The resulting proliferation of linear variants, *e.g.*, DLinear, NLinear (Zeng et al., 2023), RLinear (Li et al., 2023), SparseTSF (Lin et al., 2024), gave the impression of a rich design space, until Toner & Darlow (2024) proved that these models are functionally equivalent to unconstrained linear regression over suitably augmented features, and that closed-form ordinary least squares solutions can match or exceed their SGD-trained counterparts. This collapse of seemingly architectural diversity into a single model class raises a natural question: if the model is effectively fixed, where should the remaining degrees of freedom be spent?

We argue that the answer is preprocessing. The standard evaluation protocol in time series forecasting fixes context length, normalization strategy, and data augmentation to a single setting per benchmark and then compares architectures. This convention makes sense when the goal is to isolate the effect of model design, but it systematically disadvantages models whose capacity is too limited to absorb suboptimal input representations through learned parameters. Transformers with millions of parameters can partially compensate for a poorly chosen lookback window or an uninformative normalization scheme; a linear model cannot. The result is that linear methods appear weaker than they are, not because they lack expressive power for the task, but because they are more sensitive to choices that are rarely tuned. We test this hypothesis with Ridge regression (Hoerl & Kennard, 1970), applied to a systematic search over context length, local normalization windows, regularization strength, and augmentation in both time and frequency domains across eight standard benchmarks at per-horizon and per-series granularity. Ridge has a closed-form solution, no hidden nonlinearity, runs a trial in a few milliseconds on a GPU, and produces weights that can be inspected directly. The same transparency that makes it a competitive forecaster also makes it a diagnostic instrument: the structure of the optimal hyperparameters reveals properties of the data that deeper models would absorb silently into their learned representations.

The search reveals three intriguing observations. First, the relationship between optimal lookback and forecast horizon is strongly dataset-specific (Figures 1&2) and often non-monotonic (Figure 3). When fitting a power-law $L^* = a \cdot H^b$ for the searched optimal lookback L^* and the prediction horizon H, the exponent b ranges from $+0.46$ on ETTm2 to -0.19 on Exchange and Traffic, contradicting the common assumption that longer horizons demand longer history (Oreshkin et al., 2020; Challu et al., 2023). Second, normalizing over a learned trailing fraction of the context, rather than its entirety as in prior work, consistently improves accuracy, indicating that recent local statistics carry more signal than global ones (Figure 7). Third, series within the same dataset often prefer different hyperparameters (Figure 4), and the optimal degree of cross-series sharing varies from fully shared on ETTh1 to fully per-series on Weather, suggesting the heterogeneity across channels as an underexplored axis in forecasting.

Our contributions are as follows. First, we show that carefully tuned Ridge regression outperforms prior linear forecasters across 7 of 8 datasets, and matches or exceeds Transformer- and MLP-based architectures on seven of eight benchmarks, while being orders of magnitude cheaper to train. Second, we demonstrate that the optimal hyperparameter landscape of a transparent linear model encodes structural properties of time series data (scaling behavior, normalization preferences, series heterogeneity) that are informative beyond the model itself and can guide the design of more complex forecasters. Third, we release SearchCast (URL), a reproducible pipeline that supports per-horizon and per-series search with configurable ablation controls, facilitating future studies to diagnose their own datasets with the same methodology.

2 Related Work

Transformer-based forecasters. Attention-based architectures dominated long-term forecasting for several years. Autoformer (Wu et al., 2021) and FEDformer (Zhou et al., 2022) replaced standard

self-attention with frequency-domain operators to capture seasonal structure. PatchTST (Nie et al., 2022) segmented inputs into patches and processed each channel independently, and remains the strongest transformer baseline on long-term benchmarks. iTransformer (Liu et al., 2023) inverted the attention axis to model cross-channel dependencies. Recent benchmarks (Qiu et al., 2024; Toner & Darlow, 2024) find that no single transformer wins across datasets and horizons, which raises the question of where the remaining gains should come from.

MLP and CNN alternatives. Another line of work explored simpler architectures. TimesNet (Wu et al., 2022) reshapes 1D series into 2D tensors and applies convolutions. TiDE (Das et al., 2023) and TimeMixer (Wang et al., 2024) are encoder-decoder MLPs, the latter mixing across temporal axes. TSMixer (Chen et al., 2023) reaches competitive accuracy with an attention-free MLP mixer.

Linear models. DLinear (Zeng et al., 2023) showed that a trend-seasonal decomposition followed by two linear layers could outperform mordern transformers; NLinear, RLinear (Li et al., 2023), and SparseTSF (Lin et al., 2024) extended the recipe with last-value normalization, reversible instance normalization (Kim et al., 2022), and parameter-sparse forecasting, respectively. Toner & Darlow (2024) then proved that DLinear, NLinear, and several related variants are functionally equivalent to unconstrained linear regression over augmented features, and that closed-form ordinary least squares (OLS) solutions match or beat the same models trained with SGD. This collapses the diverse linear forecasters into a single model class.

Foundation models for time series. A more recent line proposes general-purpose forecasters trained on large corpora and applied zero-shot, including Chronos (Ansari et al., 2024), TimesFM (Abhimanyu, 2024), Moirai (Woo et al., 2024), and DAM (Darlow et al., 2025). These models target broad coverage rather than per-dataset accuracy, and on standard supervised benchmarks. However, reaching this generality requires large pretraining corpora and substantially more compute than per-dataset fitting, while accuracy on standard supervised benchmarks is often comparable to tuned per-dataset baselines, a small marginal return on the added cost.

Positioning. We build on the unification of Toner & Darlow (2024). Instead of proposing another linear variant, we fix the model to Ridge regression (Hoerl & Kennard, 1970) and spend the remaining budget on preprocessing: context length, local normalization, regularization, and augmentation, searched per-horizon and per-series across eight benchmarks. This closes most of the gap to deeper baselines and turns the optimized hyperparameters into a diagnostic: how optimal context scales with horizon, whether a trailing fraction of the window beats the full window for normalization, and how much variates within a dataset disagree, all readable off the search and mostly invisible in the parameters of deeper models.

3 METHOD

3.1 PRELIMINARY: RIDGE REGRESSION

Toner & Darlow (2024) showed that recent linear forecasting variants are functionally equivalent to unconstrained linear regression over appropriately augmented feature sets. We take this unification as our starting point and build on the simple and efficient setup: Ridge regression (Hoerl & Kennard, 1970) with a closed-form solution. Given a context window $\mathbf{x} \in \mathbb{R}^L$ of L past observations for a single variate and a forecast horizon H, the model predicts $\hat{\mathbf{y}} = \mathbf{W}\mathbf{x} + \mathbf{b}$ where $\mathbf{W} \in \mathbb{R}^{H \times L}$ and $\mathbf{b} \in \mathbb{R}^H$. The weight matrix is obtained by solving

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} \|\mathbf{Y} - \mathbf{W}\mathbf{X}\|_F^2 + \alpha\|\mathbf{W}\|_F^2 \tag{1}$$

where $\mathbf{X} \in \mathbb{R}^{L \times N}$ and $\mathbf{Y} \in \mathbb{R}^{H \times N}$ are the training context and target matrices assembled from N sliding windows, respectively; and α is the regularization strength. The solution $\mathbf{W}^* = \mathbf{Y}\mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top + \alpha\mathbf{I})^{-1}$ is computed in closed form without iterative optimization, making training effectively instantaneous on a single GPU and highly amenable to large-scale hyperparameter search. Each variate is modeled independently following the channel-independent paradigm that has proven effective across both linear (Zeng et al., 2023) and Transformer-based (Nie et al., 2022) forecasters.

3.2 PREPROCESSING PIPELINE

The preprocessing applied before the simple Ridge regression is where most of the modeling flexibility resides. We parameterize it along four axes.

Normalization. We consider two normalization scopes. Global normalization computes statistics over the entire training set for each variate; local normalization computes them over the most recent $r \cdot L$ time steps of each input window, where $r \in (0, 1]$ is the *local ratio*. In either case we support two methods: standardization and robust normalization. Local normalization with $r = 1$ recovers full-window instance normalization as used in prior work (Toner & Darlow, 2024); values $r < 1$ restrict the statistics to a trailing fraction of the context, allowing the model to adapt to the most recent distributional regime. The local ratio is searched on a log scale between 0.001 and 1.0. Under local normalization we append the per-window scale σ as an additional feature in place of the intercept (which is uninformative once the context has zero local mean), letting the regression learn a volatility-dependent shift; a zero coefficient on σ recovers pure instance normalization.

Context length. The lookback L determines how much history the model sees and directly controls the dimensionality of the regression problem. We search $L \in [32, 2048]$ on a log scale, with upper bounds capped by the data size and forecast horizon.

Augmentation. We optionally perturb training windows with additive noise scaled by σ in either the time (Gaussian noise) or the frequency domain (Gaussian perturbation of Fourier coefficients), where σ is searched on a log scale between 0.001 and 0.5. A third option applies no augmentation at all.

Regularization. Rather than selecting a single α, we evaluate a grid of 21 logarithmically spaced values (in range $[10^{-6}, 10^{3}]$) and retain the one that minimizes validation loss, effectively treating regularization as an inner optimization loop that imposes no additional search cost.

3.3 Grouped Hyperparameter Search

Prior work almost always fixes a single set of preprocessing hyperparameters across every benchmark, horizon step, and variate. This is computationally convenient but assumes a degree of homogeneity within the datasets that rarely holds in practice: different horizons/variates can demand different context lengths. The opposite extreme, an independent search per horizon step and per variate, would capture this heterogeneity but is rarely attempted because the search budget grows combinatorially. We introduce a grouped search scheme that moves continuously between these two endpoints.

Horizon grouping. We partition the H forecast steps into contiguous blocks of size g_h and share hyperparameters within each block. Setting $g_h = 1$ recovers fully per-step tuning; setting $g_h = H$ recovers the global baseline. Intermediate values allow hyperparameters to vary smoothly across the forecast horizon without requiring a separate search per step.

Series grouping. Similarly, the C variates are partitioned into groups of size g_s. Setting $g_s = 1$ yields a fully per-series search that can capture heterogeneous dynamics across channels; setting $g_s = C$ shares *weights and hyperparameters* across all variates.

Unified view. The pair (g_h, g_s) defines a grid of hyperparameter search cells over the horizon$\times$series space. Each cell runs an independent Optuna search with a Tree-structured Parzen Estimator (Akiba et al., 2019) over the joint preprocessing space: lookback L, normalization scope and method, local ratio r, augmentation type and intensity σ, with α selected by inner grid search per trial. The global model ($g_h = H, g_s = C$) and fully local model ($g_h = 1, g_s = 1$) are special cases of this framework.

Cross-validation. Each Optuna trial evaluates candidate hyperparameters using k-fold expanding-window cross-validation on the training set, with folds constructed chronologically to respect temporal ordering. We use $k = 3$ in all main experiments. The best hyperparameters are then applied to the held-out test set, which is never used during the search.

4 Experiments

4.1 Experimental Setup

Datasets We evaluate on eight widely used multivariate forecasting benchmarks spanning diverse domains and temporal granularities: ETTh1 and ETTh2 (electricity transformer temperature, hourly), ETTm1 and ETTm2 (same source, 15-minute intervals), Weather (21 meteorological indicators, 10-minute), Electricity (321 clients, hourly), Traffic (862 road sensors, hourly), and Exchange (daily exchange rates of 8 countries). Following standard protocol (Wu et al., 2021), each dataset is split chronologically into training, validation, and test sets at a 6:2:2 ratio for ETT and 7:1:2 for the others.

Table 1: Long-term multivariate forecasting results (MSE). Left panel: linear models; right panel: nonlinear models. OLS, FITS, and DLinear report the results with instance normalization from Toner & Darlow (2024). **Bold** : best per row. Underline : second best.

		Linear Models				Nonlinear Models		
Dataset	H	**Ours**	OLS	FITS	DLinear	PatchTST	TimeMixer	PatchMLP
ETTm1	96	0.296	0.307	0.309	0.312	**0.290**	0.293	0.309
	192	0.332	0.336	0.338	0.341	**0.329**	0.335	0.355
	336	**0.361**	0.365	0.367	0.372	0.362	0.368	0.393
	720	**0.399**	0.415	0.417	0.422	0.415	0.426	0.450
	Avg	***0.347***	*0.356*	*0.358*	*0.362*	*0.349*	*0.356*	*0.377*
ETTm2	96	**0.161**	0.162	0.162	0.163	0.165	0.164	0.173
	192	**0.213**	0.216	0.217	0.217	0.221	0.225	0.232
	336	**0.257**	0.268	0.269	0.269	0.276	0.277	0.294
	720	**0.324**	0.349	0.350	0.354	0.362	0.360	0.380
	Avg	***0.239***	*0.249*	*0.250*	*0.251*	*0.256*	*0.257*	*0.270*
ETTh1	96	**0.365**	0.375	0.377	0.379	0.376	0.374	0.396
	192	**0.399**	0.413	0.413	0.419	0.409	0.417	0.430
	336	**0.424**	0.445	0.432	0.451	0.431	0.438	0.456
	720	**0.427**	0.460	0.428	0.470	0.458	0.487	0.496
	Avg	***0.404***	*0.423*	*0.412*	*0.430*	*0.419*	*0.429*	*0.445*
ETTh2	96	**0.269**	0.270	0.270	0.275	0.274	0.291	0.302
	192	**0.329**	0.331	0.331	0.342	0.348	0.349	0.381
	336	**0.353**	0.354	0.354	0.359	0.377	0.403	0.416
	720	0.384	0.380	**0.377**	0.384	0.406	0.401	0.433
	Avg	*0.334*	*0.334*	***0.333***	*0.340*	*0.351*	*0.361*	*0.383*
Electricity	96	**0.131**	0.133	0.133	0.134	0.143	0.153	0.140
	192	**0.146**	0.148	0.148	0.149	0.158	0.168	0.160
	336	**0.162**	0.164	0.164	0.165	0.168	0.188	0.179
	720	**0.200**	0.203	0.203	0.205	0.214	0.228	0.210
	Avg	***0.160***	*0.162*	*0.162*	*0.163*	*0.171*	*0.184*	*0.172*
Traffic	96	0.382	0.385	0.386	0.387	**0.370**	**0.370**	0.415
	192	0.395	0.397	0.398	0.399	**0.386**	0.399	0.433
	336	0.408	0.410	0.411	0.412	**0.396**	0.420	0.448
	720	0.443	0.448	0.449	0.450	**0.435**	0.445	0.480
	Avg	*0.407*	*0.410*	*0.411*	*0.412*	***0.397***	*0.409*	*0.444*
Weather	96	**0.141**	**0.141**	0.142	0.142	0.148	0.147	0.153
	192	**0.183**	0.184	0.185	0.185	0.191	0.194	0.196
	336	**0.233**	0.234	0.236	0.235	0.243	0.244	0.245
	720	**0.303**	0.307	0.307	0.310	0.313	0.319	0.321
	Avg	***0.215***	*0.217*	*0.218*	*0.218*	*0.224*	*0.226*	*0.229*
Exchange	96	**0.084**	0.086	0.087	0.085	0.087	0.086	0.090
	192	**0.171**	0.180	0.183	0.178	0.176	0.173	0.186
	336	0.308	0.343	0.344	0.335	**0.296**	0.376	0.337
	720	**0.824**	0.992	0.965	0.920	0.853	0.874	0.893
	Avg	***0.347***	*0.400*	*0.395*	*0.380*	*0.353*	*0.377*	*0.377*

Evaluation Protocol. We report MSE at horizons $H \in \{96, 192, 336, 720\}$ and per-dataset averages. Search uses 20 trials per cell, and 3-fold cross-validation (sensitivity in Appendix F). Because our method searches the context length L, each fold includes the additional pre-validation history needed to form the input window, but keeps the number of validation target points fixed across all candidate L. Thus longer contexts do not receive more validation samples nor a different validation interval.

Baselines. Linear: OLS, FITS (with instance normalization, from Toner & Darlow (2024)), and DLinear (Zeng et al., 2023). These isolate the gain attributable to preprocessing search rather than model capacity. **Nonlinear: Nonlinear:** PatchTST (Nie et al., 2022) (Transformer), and

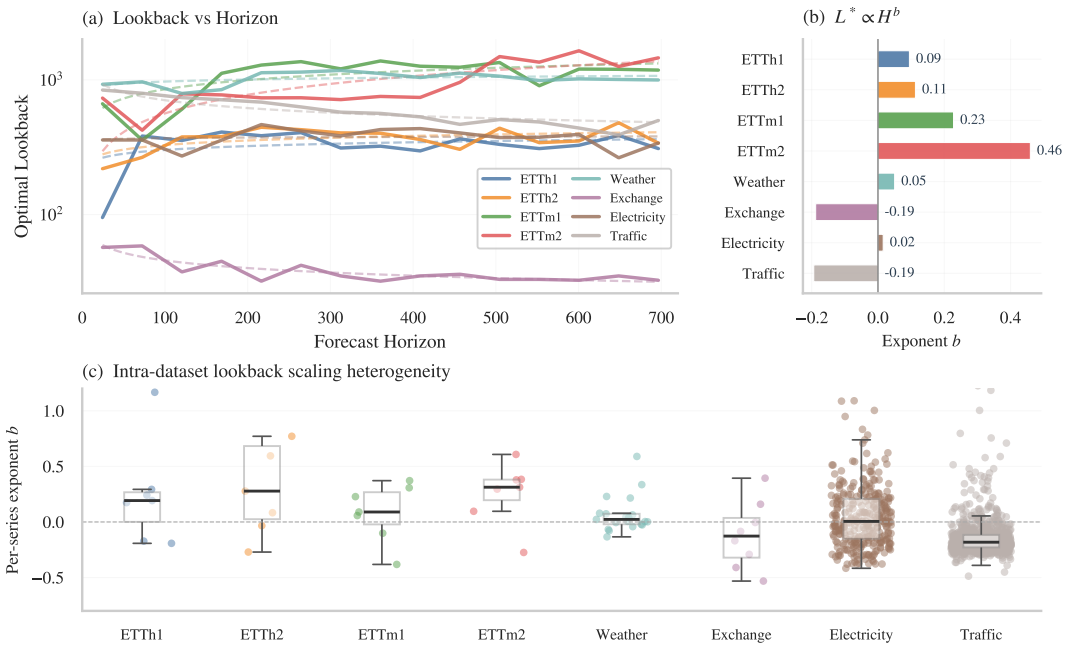

Figure 3: Optimal lookback L^* vs. forecast horizon H across 8 datasets. (a) Median lookback (log-log) with IQR bands and power-law fits $L^* \propto H^b$. (b) Fitted exponent b per dataset. (c) Per-series exponent distribution within each dataset.

TimeMixer (Wang et al., 2024) and PatchMLP (Tang & Zhang, 2025) (MLP). Nonlinear baselines are rerun using the best configurations selected by TFB (Qiu et al., 2024) to match test data points and ensure fair comparisons; linear baselines from Toner & Darlow (2024) use $L = 720$. Our method searches over L (while keeping the validation samples fixed for fair comparisons), a degree of freedom we argue is underexplored.

4.2 Main Results

Table 1 reports MSE across the eight benchmarks. **Within the linear class**, our pipeline SearchCast achieves the best average MSE on seven of eight datasets and ties FITS on ETTh2 (0.334 vs. 0.333). Reductions over OLS reach 4.0% on ETTm2, 4.5% on ETTh1, and 13.3% on Exchange. The gap is consistent across H rather than driven by a single horizon, indicating the gain comes from preprocessing choices that generalize. **Against nonlinear baselines**, ours attains the lowest average MSE on seven of eight benchmarks despite having no nonlinearity and no learned representation. The only loss is to PatchTST on Traffic (0.407 vs. 0.397), the largest dataset (862 channels), where shared representations across similar series plausibly help Transformers more than simple Ridge. On the remaining seven datasets, the margin over PatchTST ranges from 0.6% (ETTm1) to 6.6% (ETTm2). The linear comparison isolates the contribution of preprocessing search; the nonlinear comparison shows it is large enough to close the gap on most benchmarks.

5 Analysis

We use the search output as a diagnostic of the data. We first study how optimal lookback changes with horizon and series (§5.1), then how much hyperparameter sharing is safe across series and horizons (§5.2). Forecasts and refit Ridge weights connect these choices to model behavior (§5.3). The remaining subsections isolate the main preprocessing choices: local normalization (§5.4), series-level variation in r and α (§5.5), and augmentation type/intensity (§5.6).

5.1 Lookback *v.s.* Horizon Across Datasets

Figure 3(a,b) fits $L^* = a \cdot H^b$ to the searched optimal context at each horizon group **per dataset**. Exponents span both signs and an order of magnitude: from $b = +0.46$ on ETTm2 down to $b = -0.19$ on Exchange and Traffic. ETTm2 is the only dataset where the conventional intuition

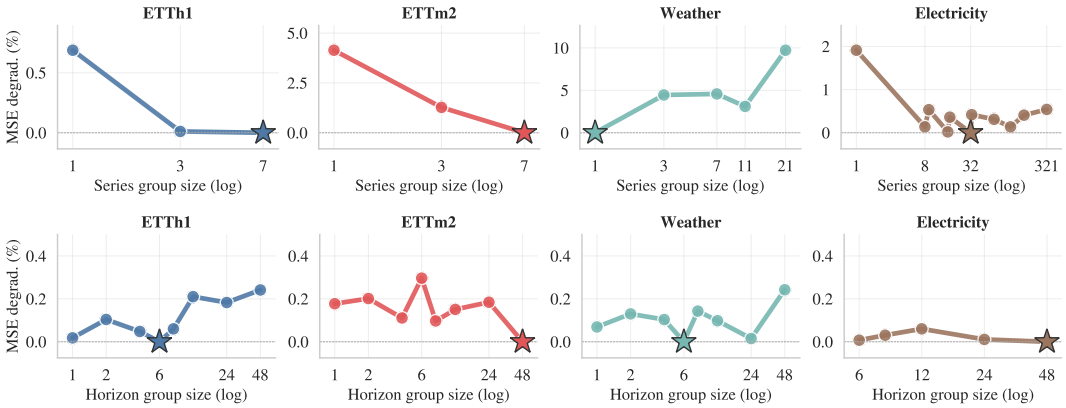

Figure 4: Effect of series (*Top*) and horizon (*Bottom*) grouping on forecasting accuracy across 4 datasets. Each panel shows MSE degradation (%) relative to the best group size (marked with $\star$) as a function of group size, from per-series/horizon to fully shared.

holds strongly, there longer horizons ask for more history, though sublinearly. ETTm1 and ETTh2 sit in a mild positive regime ($b \approx 0.1$–0.2); Weather and Electricity are nearly flat ($|b| < 0.1$), settling on $\sim 10^3$ steps regardless of horizon. The negative exponents on Exchange and Traffic are the most informative: longer horizons prefer *shorter* context, consistent with non-stationarity where distant history actively misleads the model. The standard $L = 96$ default is therefore wrong in two opposite directions at once: it underserves Weather and Electricity by roughly an order of magnitude, and overserves Exchange and Traffic at long horizons, where the optimum drops below 96.

Figure 3(c) repeats the fit **per variate**. Dataset-level exponents are aggregates over considerable intra-dataset spread. Weather is the only benchmark whose per-series exponents cluster tightly around the median; ETTh2, Electricity, and Traffic straddle zero, meaning some channels prefer growing context with horizon while others prefer shrinking it. This highlights that a single shared lookback is insufficient even within one dataset, which motivates the series/horizon grouping analysis below.

5.2 Series and Horizon Grouping

Figure 4 (top) sweeps the series group size g_s from per-series ($g_s = 1$) to fully shared (C), reporting MSE degradation relative to the per-dataset best. The picture is non-monotone. **Fully shared** is best on both ETT benchmarks, where per-series search degrades MSE by 0.5–4%, suggesting physically related variates regularize each other. **Fully per-series** is best on Weather, where degradation grows monotonically with g_s to 10% at $g_s = 21$, suggesting its 21 channels (pressure, humidity, wind, rainfall) live on incompatible scales. Electricity prefers **moderate** sizes ($g_s = 32$). The result indicates that the optimal degree of cross-series sharing is a property of the dataset.

The bottom row sweeps $g_h \in \{1, 2, 3, 4, 6, 8, 12, 16, 24, 48\}$. g_h groups *consecutive* horizons into bins of width g_h that share one HP setting: at $g_h = 1$, horizons $H = 95$ and $H = 96$ are tuned separately; at $g_h = 48$, all of $H = 49, \ldots, 96$ share a single setting. The four evaluation cutoffs $\{96, 192, 336, 720\}$ are far enough apart that they land in different bins for every valid g_h. So as g_h grows, the four reported cutoffs keep their own tuned HPs; only the horizons *near* each cutoff are progressively absorbed into its bin. The curves are flat ($\leq 0.4\%$ degradation), with optima at $g_h = 48$ on ETTm2 and Electricity, and $g_h = 6$ on ETTh1 and Weather. This says HPs vary smoothly with H within a $\sim$48-step neighborhood, consistent with the smooth $L^*(H)$ trends in §5.1. We adopt $g_h = 48$ by default for its efficiency and spend the trial budget on searching g_s instead.

5.3 Forecasts and Learned Weights Visualization

Figure 5 shows the visible effect of the search. With fixed dataset-level defaults, Ridge drifts toward the mean as the horizon grows, especially on Weather and Exchange. The tuned model keeps following the slow drift because these datasets select short effective context, small local-normalization windows, or both (Figures 3b and 7).

Figure 6 shows what the selected lookbacks are used for. Exchange puts almost all weight on recent lags, matching its short-memory behavior. Traffic and ETTh2 use calendar lags: Traffic has strong

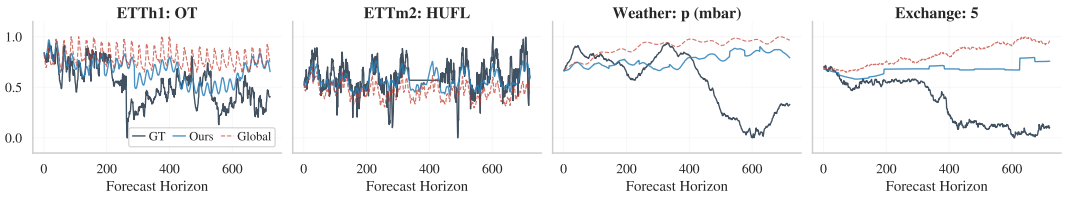

Figure 5: Forecast comparison. Our method (blue) closely tracks the ground truth (black), while the global baseline (red) reverts to the mean, particularly on non-stationary series (Weather, Exchange). The choppiness in Weather and especially Exchange is an artifact of switching hyperparameters across forecast settings, which can create discontinuities between segments.

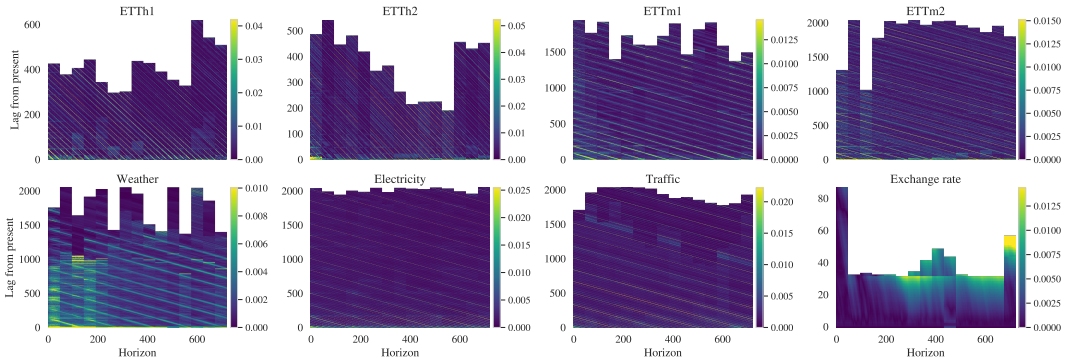

Figure 6: Weight magnitude $|w|$ over lag and forecast horizon. Lighter shades indicate larger magnitudes. Lag is measured from the most recent input (bottom row); white regions lie beyond each model's chosen lookback.

weekly bands near $168, 336, \ldots$, while ETTh2 shows a shorter daily comb. Electricity mixes lag-$0/1$ dependence with remote anchors. Weather and ETTm datasets often place their largest weights hundreds of steps back. Thus a long L^* mainly gives the model access to a few phase-matched observations, not to the whole history. The jumps at $g_h{=}48$ bin boundaries show the tradeoff: horizon grouping barely changes MSE ($\leq 0.4\%$; §5.2), but it can change the weights abruptly.

5.4 Global *v.s.* Local Normalization

Prior linear forecasters normalize either globally (training-set statistics) or locally (full input window) (Toner & Darlow, 2024; Zeng et al., 2023). We relax the local case to a learned trailing fraction $r \in (0, 1]$: $r = 1$ recovers full-window normalization, $r < 1$ restricts statistics to the last $r \cdot L$ steps. Across ETTh1, ETTm2, Weather, and Exchange, we found that local is selected in 62–100% of dataset–horizon cells; the local-ratio column of Figure 7 shows that the optimal $\log_{10} r$ is almost always strictly negative, clustered in $[-2.5, -0.5]$ — trailing fractions between 0.3% and 30% of the window. **Full-window normalization is essentially never selected.** The effect is strongest on ETTm2 and Exchange, where many cells sit near $\log_{10} r = -2$: only the final few percent of the lookback set the scale. These series are nonstationary on the scale of L, and full-window statistics blur regimes the model is better off keeping distinct. We also tested a robust median/IQR variant of local normalization, but it underperformed mean/std in all tested settings, suggesting that tail variation and large deviations carry useful scale information in these benchmarks. We therefore use local standardization in Table 1 and search only the trailing fraction r.

5.5 Hyperparameter Variations Across Series

Figures 3 and 7 identify *which* hyperparameters drive the Weather *vs.* ETTh1 contrast. Lookback is uniform across series on ETTh1 and dispersed on Weather (Figure 3c). The local-ratio and α heatmaps in Figure 7 sharpen the picture. **ETTh1:** both rows are visibly uniform across the four variates and vary more with horizon than with series, consistent with variates from one physical system. **Weather:** the two rows diverge — local ratio varies widely (OT at $\log_{10} r \approx -2$ vs. LULL near 0), and α shows a large gap between OT ($\log_{10} \alpha \approx 1$–$2$) and the other channels (near 4). The per-series gain on Weather (Figure 4) therefore comes from local ratio and α, not lookback. ETTm2

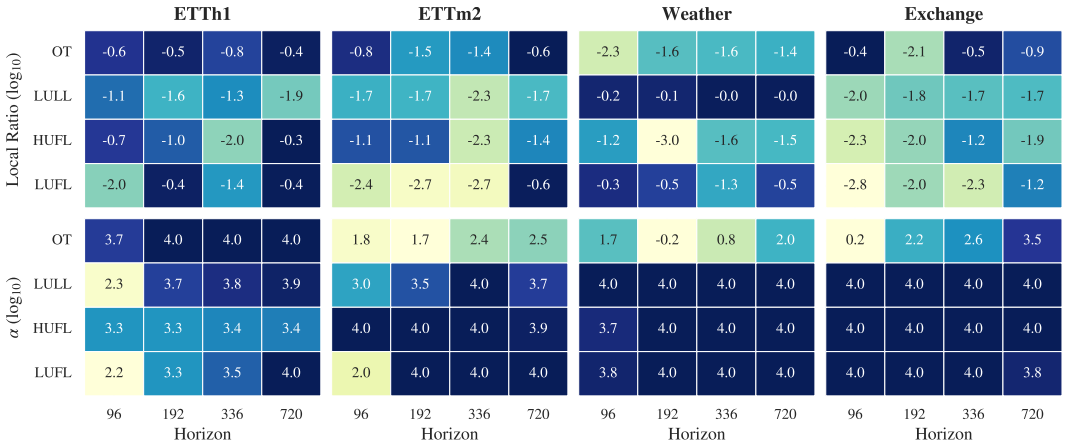

Figure 7: Per-series hyperparameter (local ratio r and regularization α) heatmaps, with $g_h = 48$.

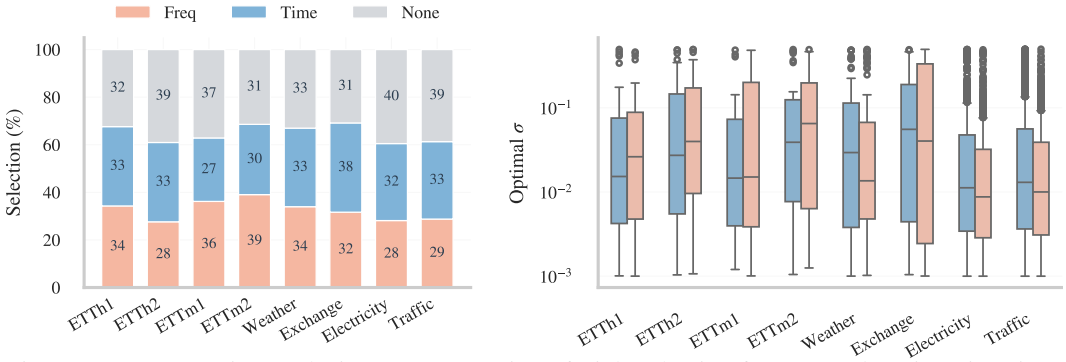

Figure 8: Augmentation analysis. *Left*: Proportion of trials selecting frequency-domain noise, time-domain noise, or no augmentation. *Right*: Distribution of optimal σ conditional on augmentation being selected, broken down by domain.

and Exchange sit between these extremes; grouping by measurement type or a few clusters would likely capture most of the heterogeneity at a fraction of full per-series cost.

5.6 Augmentation Selection and Intensity

Figure 8(a): augmentation is selected in 60–70% of horizon groups on every benchmark, with time- and frequency-domain noise split roughly evenly (time slightly more common). Figure 8(b) displays *how* they are selected. The within-dataset spread of optimal σ is itself dataset-dependent: Weather and Exchange are widest (near- and far-horizon groups demand different noise levels); ETTm1, Electricity, and Traffic are tighter.

6 Conclusion

The reputation of linear models as uncompetitive baselines in long-term forecasting reflects under-tuned preprocessing, not limited model capacity. A Ridge regression searched over context length, local normalization, regularization, and augmentation matches or exceeds prior linear baselines as well as Transformer, MLP, and CNN baselines on most standard benchmarks, at a fraction of the training cost. Beyond accuracy, the optimized hyperparameters serve as a diagnostic lens on the data: optimal lookback can grow, plateau, or shrink with the forecast horizon depending on dataset stationarity; normalization is almost always local, restricted to a trailing fraction of the window; cross-series sharing is dataset-specific, ranging from fully shared to fully per-series, and the heterogeneity is driven by normalization and regularization rather than lookback; and the same locality and seasonality the search recovers in the hyperparameters are visible directly in the trained weights. These preprocessing choices transfer to any model class, and the released SearchCast pipeline makes the same diagnostic cheap to run on new datasets.

AI USE STATEMENT

We used generative AI tools to assist with code implementation and literature checks, obtain reviewer-style research feedback, and refine the writing. The authors take full responsibility for the final text, code, figures, methods, results, analyses, claims, and references, including material prepared with AI assistance.

ETHICS STATEMENT

This study evaluates linear models on several public, real-world time-series benchmarks and involves no human participants or personal data. We are not aware of any specific ethical concerns that require disclosure beyond those generally associated with research on linear time-series forecasting.

REPRODUCIBILITY STATEMENT

Section 3 defines the objective and the evaluation, the benchmarks and comparison protocol. we release SearchCast (URL), a reproducible pipeline that supports per-horizon and per-series search with configurable ablation controls, facilitating future studies to diagnose their own datasets with the same methodology.

REFERENCES

Das Abhimanyu. A decoder-only foundation model for time-series forecasting. In *International Conference on Machine Learning*, 2024.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.

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

William A Brock, W Davis Dechert, José A Scheinkman, and Blake LeBaron. A test for independence based on the correlation dimension. *Econometric Reviews*, 15(3):197–235, 1996.

Colin Catlin. Adaptive forecasting in dynamic markets: An evaluation of autots within the m6 competition. *International Journal of Forecasting*, 2025.

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

Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O Arik, and Tomas Pfister. Tsmixer: An all-mlp architecture for time series forecasting. *arXiv preprint arXiv:2303.06053*, 2023.

Robert B Cleveland, William S Cleveland, Jean E McRae, and Irma Terpenning. STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, 6(1):3–73, 1990.

Luke Darlow, Qiwen Deng, Ahmed Hassan, Martin Asenov, Rajkarn Singh, Artjom Joosen, Adam Barker, and Amos Storkey. Dam: Towards a foundation model for forecasting. In *International Conference on Learning Representations*, 2025.

Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint arXiv:2304.08424*, 2023.

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970.

Rob J Hyndman and George Athanasopoulos. *Forecasting: Principles and Practice*. OTexts, Melbourne, Australia, 3rd edition, 2021.

Rob J Hyndman, George Athanasopoulos, et al. *Forecasting: principles and practice*, volume 2. OTexts Melbourne, Australia, 2014.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2022.

Zhe Li, Shiyi Qi, Yiduo Li, and Zenglin Xu. Revisiting long-term time series forecasting: An investigation on linear mapping. *arXiv preprint arXiv:2305.10721*, 2023.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Haojun Chen, and Junjie Yang. Sparsetsf: Modeling long-term time series forecasting with 1k parameters. *arXiv preprint arXiv:2405.00946*, 2024.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*, 2020.

Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S Jensen, Zhenli Sheng, et al. Tfb: Towards comprehensive and fair benchmarking of time series forecasting methods. *arXiv preprint arXiv:2403.20150*, 2024.

Skipper Seabold and Josef Perktold. statsmodels: Econometric and statistical modeling with python. In *Proceedings of the 9th Python in Science Conference (SciPy)*, pp. 92–96, 2010.

Oleksandr Shchur, Ali Caner Turkmen, Nick Erickson, Huibin Shen, Alexander Shirkov, Tony Hu, and Bernie Wang. Autogluon–timeseries: Automl for probabilistic time series forecasting. In *International Conference on Automated Machine Learning*. PMLR, 2023.

Peiwang Tang and Weitai Zhang. Unlocking the power of patch: Patch-based mlp for long-term time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 12640–12648, 2025.

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

William Toner and Luke Darlow. An analysis of linear time series forecasting models. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 48404–48427, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv:2405.14616*, 2024.

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

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34: 22419–22430, 2021.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 11121–11128, 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*, pp. 27268–27286. PMLR, 2022.

A LONG-RANGE LINEAR AUTOCORRELATION IN THE BENCHMARK SERIES

The per-dataset search of Section 5.1 selects context lengths that differ by two orders of magnitude, from a few dozen steps on Exchange to roughly a thousand on Weather and Electricity (Figure 3). This appendix examines a basic question behind that result. When the search assigns the model a long context, is there genuinely useful information that far in the past, or is the model only re-representing an obvious daily or weekly cycle that a much simpler seasonal method (Zeng et al., 2023; Zhou et al., 2022) could capture without a long context? We address it by measuring how strongly each series is linearly related to its own past, both on the raw series and after *deseasonalization*. Deseasonalization here means removing a series' known periodic cycles (for instance the daily and weekly ones), so that any remaining correlation cannot be attributed to those cycles alone. We apply four deseasonalization methods of increasing strength (defined below) and check whether the long-range correlation survives all of them.

The *autocorrelation* at lag k is the Pearson correlation between the series at time t and the same series k steps earlier, $\rho(y_t, y_{t-k})$, computed over all t and then averaged across the channels of a dataset (Hyndman & Athanasopoulos, 2021). It ranges from -1 to 1. A value near 0 means the value k steps ago carries essentially no *linear* information about the present, while a large magnitude means it carries substantial such information. Because we focus on the linear relationship, autocorrelation is precisely the kind of structure our forecaster can exploit. Before measuring, each series is z-scored using training-set statistics only, that is, shifted and scaled to mean 0 and variance 1 using quantities computed on the training portion alone, so that the measurement is comparable across series and does not use test data.

From weakest to strongest, we apply four deseasonalization methods. The *raw* variant removes nothing. The *per-position-in-period* variant subtracts the average value at each position of the dominant cycle (for example the mean for each hour-of-day). The *harmonic* variant subtracts fitted sine and cosine waves at all known periods such as daily and weekly. The *STL* variant applies a standard procedure that splits a series into a trend, a seasonal part, and a remainder, keeping only the remainder (Cleveland et al., 1990). The rationale for trying all four is that if a long-lag correlation were entirely a periodic cycle, a model that explicitly represents cycles, such as the trend-seasonal decomposition inside DLinear (Zeng et al., 2023) or the frequency-domain operators of Autoformer and FEDformer (Wu et al., 2021; Zhou et al., 2022), could reproduce it at low cost, and a long context would offer a purely linear model no genuine advantage. Each reported correlation comes with a 95% confidence interval from a pair-resampling bootstrap. We resample the observed (y_t, y_{t-k}) pairs with replacement ($B = 200$ times), recompute the correlation each time, and report the middle 95% of those values. A narrow interval means the estimate is statistically reliable.

Figure 9 traces the autocorrelation across all measured lags for the four variants, and Table 2 reports the slice at lag $k = 720$, the longest forecast horizon evaluated in Section 4. All six datasets retain a substantial correlation at $k = 720$, between 0.39 and 0.73, with confidence intervals far above 0. The four removal methods agree closely at long lags, so the surviving correlation is not an artifact of any single deseasonalization choice. On ETTm1 and ETTm2 the deseasonalized correlation is in fact *higher* than the raw one, because removing the strong daily cycle uncovers slower structure that the cycle had been masking.

Table 2: Long-range linear autocorrelation in the six standard benchmarks, computed at lag $k = 720$ after harmonic deseasonalization at all configured periods. Values are channel-averaged Pearson correlations on the z-scored concatenation of the train, validation and test splits. Confidence intervals are 95% pair-resampling bootstraps with $B = 200$.

Dataset	$\hat{\rho}(y_t, y_{t-720})$	95% CI
ETTh1	0.387	[0.381, 0.394]
ETTh2	0.526	[0.521, 0.531]
ETTm1	0.435	[0.433, 0.438]
ETTm2	0.725	[0.723, 0.728]
Weather	0.428	[0.400, 0.451]
Exchange	0.532	[0.527, 0.538]

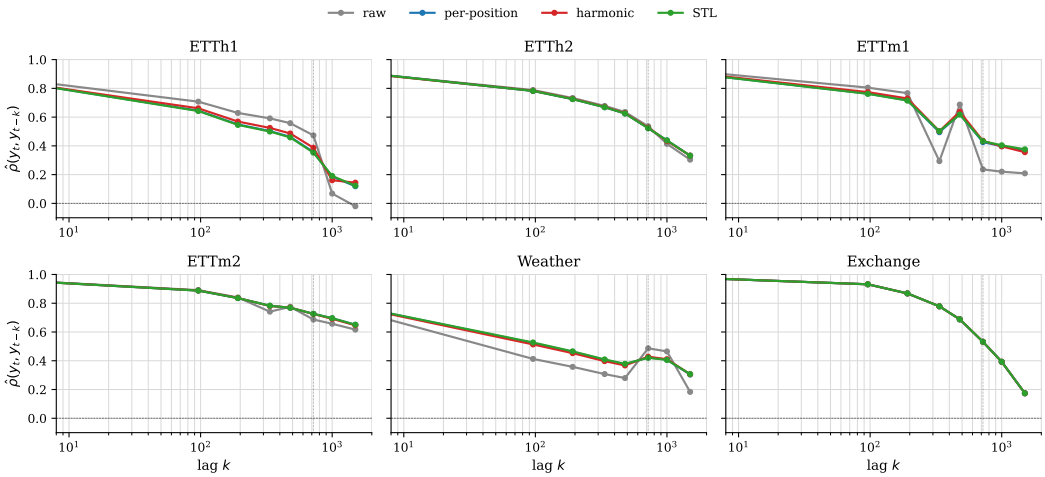

Figure 9: Channel-averaged Pearson autocorrelation $\hat{\rho}(y_t, y_{t-k})$ as a function of lag k, for the six standard benchmarks under four deseasonalization schemes. Dotted vertical line marks $k = 720$. The confidence intervals are narrow (typical 95% bootstrap half-width below 0.01, given in Table 2 for $k = 720$). The four schemes agree at long lags, with all six datasets retaining $\hat{\rho} \geq 0.39$ at $k = 720$ after harmonic deseasonalization.

Even after the obvious daily and weekly cycles are removed, a value from 720 steps in the past still carries a real and moderately strong linear association with the current value (correlation 0.39 to 0.73). Therefore, when the search provides the linear model with a long context on these datasets, the model is genuinely exploiting long-range information rather than re-representing a cycle that a simpler seasonal method could capture.

B How much accuracy comes from context length alone

Section 5.1 shows *which* context length the full search prefers on each dataset. This appendix isolates a narrower question. Of the accuracy our method gains, how much is due to the context length L *by itself*, as opposed to the other preprocessing choices (normalization, regularization, augmentation)? To answer it we hold every other preprocessing setting fixed and vary only L.

We use a single fixed preprocessing configuration everywhere, which we call the universal default. This default uses local mean normalization, the standard regularization grid described in Section 3, and no augmentation. Keeping this configuration identical across datasets means that any change in error can be attributed to L alone. We sweep $L \in \{24, 48, 96, 192, 336, 480, 720, 1000, 1500, 2000\}$, forecast at the longest horizon $H = 720$, and report the test mean squared error (MSE, the held-out squared prediction error used throughout our evaluation in Section 4, where lower is better). For each L we report the median MSE across channels and random seeds, the median being robust to a few unusually hard channels.

Figure 10 plots the full error-versus-L curves and Table 3 lists the endpoints. The datasets fall into three clear regimes. In the *strong* regime (ETTh1, ETTh2, Weather) the error continues to decrease as L grows to 2000, for a total reduction between 36% and 53%. In the *plateau* regime (ETTm2) most of the reduction is achieved by $L \approx 192$–336, after which the curve saturates, for a 16% reduction. In the *flat* regime (ETTm1) additional context yields almost no improvement. Exchange is shown only partially because its test split is too short to form windows with both $L = 2000$ and $H = 720$. Two points follow. First, on the strong-regime datasets a fixed short default such as $L = 96$ forfeits most of the attainable accuracy before any other preprocessing setting is tuned. Second, which regime a dataset falls into cannot be predicted from its metadata in advance and must instead be measured, which is why our method (Section 3) searches L per dataset rather than fixing it. We also note that re-running the full preprocessing search while holding L fixed at a short value does not recover the

long-context gain, which indicates that context length itself, rather than the other settings, is the source of the improvement.

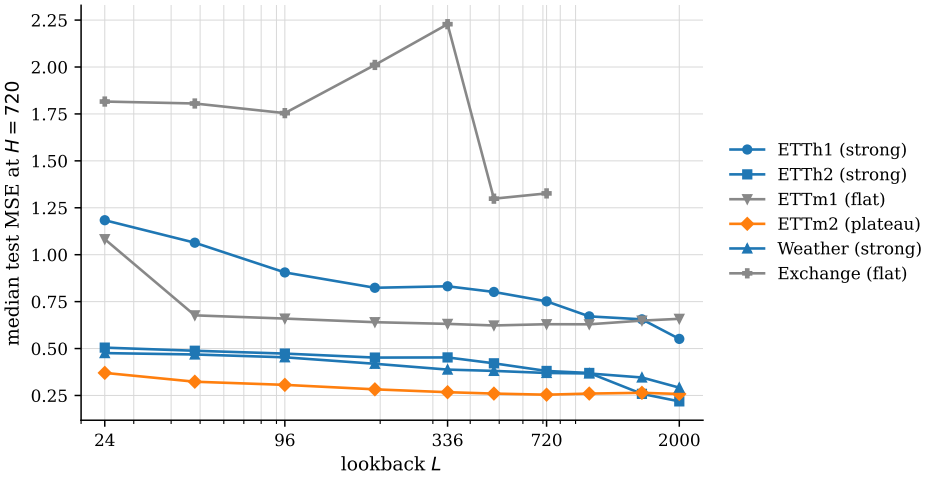

Figure 10: Median per-channel test MSE on Ridge with universal-default preprocessing at $H = 720$ as a function of the lookback L. Lines are color-coded by regime. Strong (blue) continues to improve up to $L = 2000$, plateau (orange) saturates around $L = 192$–336, and flat (gray) is essentially insensitive to L.

Table 3: Median per-channel test MSE on Ridge with universal-default preprocessing at $H = 720$ on the six standard benchmarks, as a function of the lookback L. The total-drop column is the percent change between $L = 96$ and $L = 2000$, with negative values indicating improvement. Exchange is flagged because its test split has insufficient samples to form $L = 2000$, $H = 720$ windows.

Dataset	$L = 96$	$L = 720$	$L = 2000$	Total drop
ETTh1	0.91	0.75	0.55	-39%
ETTh2	0.47	0.38	0.22	-53%
Weather	0.45	0.37	0.29	-36%
ETTm2	0.31	0.25	0.26	-16%
ETTm1	0.66	0.63	0.66	flat
Exchange	1.75	1.33	–	partial

To confirm that the model performs substantive forecasting rather than merely repeating recent values, we compare it against a *persistence* forecast, the naive rule that predicts the future to equal the last observed value, $\hat{y}_{t+h} = y_t$ (also called a no-change or random-walk forecast) (Hyndman & Athanasopoulos, 2021). Table 4 reports the ratio of the persistence error to the Ridge error at $H = 720$. Ridge is between 1.4 and 5.9 times more accurate on every dataset.

Table 4: Persistence (last-value) baseline at $H = 720$. The persistence forecast predicts $\hat{y}_{t+h} = y_t$. Ridge MSE is the universal-default Ridge from Table 3 at the per-dataset best $L \in \{24, \ldots, 2000\}$. The ratio is how many times smaller the Ridge error is.

Dataset	Persistence MSE	Ridge MSE	Ratio (Persistence / Ridge)
ETTh1	1.10	0.43	2.56
ETTh2	0.60	0.38	1.56
ETTm1	2.33	0.40	5.87
ETTm2	0.46	0.32	1.44
Weather	0.55	0.30	1.81
Exchange	1.42	0.81	1.75

For some datasets (ETTh1, ETTh2, Weather), extending the linear model's context, with no other change, reduces its error by 36 to 53%. For others (ETTm1) it yields almost no improvement. No single context length is best for every dataset, so the only reliable way to identify each dataset's optimal value is to search for it (Section 3). The persistence comparison confirms that the model performs genuine forecasting rather than reproducing the most recent value.

C Does a nonlinear model capture structure the linear model misses?

Our main results (Section 4, Table 1) show that the tuned linear model attains lower average MSE than the strongest nonlinear baseline, PatchTST, on seven of the eight benchmarks. A natural concern is that MSE may fail to reflect everything a model captures. Perhaps a nonlinear model captures genuine structure that does not register in the error value, so that the linear model would only appear competitive. This appendix addresses that concern directly by examining the prediction errors that each model produces.

The *residuals* of a model are its prediction errors at the forecast step, $y_{\text{true}} - \hat{y}$. If a model has captured everything predictable in a series, its residuals should resemble random noise with no remaining pattern. A linear model can only represent linear relationships (weighted sums of past values), so any *nonlinear* structure (a pattern in which the influence of the past is not a fixed linear weight) remains in its residuals. We therefore assess how closely each model's residuals resemble pure noise, using the Brock–Dechert–Scheinkman (BDS) test (Brock et al., 1996), a standard statistical test of whether a sequence is independent random noise or still contains residual dependence, as implemented in `statsmodels` (Seabold & Perktold, 2010). We apply it at embedding dimension two, meaning it inspects how often pairs of points fall close together relative to the frequency expected under pure randomness. A larger BDS statistic indicates more residual non-random structure. We compare the statistic itself rather than its p-value, because with this many points every p-value is driven to 0 and can no longer distinguish the models.

For each model we form the per-cell difference $\Delta = \text{BDS}_{\text{arch}} - \text{BDS}_{\text{Ridge}}$, matched on the same (dataset, context length, horizon, seed, channel). A *negative* Δ means that model's residuals are closer to noise than the linear model's, that is, it captured nonlinear structure that the linear model did not. As in Appendix A, each comparison carries a 95% confidence interval from resampling, and we call a cell *significant* when that interval excludes 0 (for example, $[-3.0, -1.2]$ does, whereas $[-1.5, +0.8]$ does not). Table 5 aggregates the comparison over the six standard benchmarks (ETTh1, ETTh2, ETTm1, ETTm2, Weather, Exchange) and, separately, over the two large many-channel datasets (Electricity with 321 channels and Traffic with 862), which include Traffic, the one benchmark where Section 4 reports that PatchTST achieves lower error (Table 1). DLinear, itself a linear model, lies near $\Delta \approx 0$, as expected, and serves as a control confirming that the test does not report differences where none exist.

Table 5: Paired BDS comparison against Ridge. $\Delta = \text{BDS}_{\text{arch}} - \text{BDS}_{\text{Ridge}}$ is the per-cell difference of the BDS statistic at embedding dimension 2, paired by (dataset, L, H, seed, channel) and bootstrapped with $B = 5{,}000$ resamples. A cell is significant when its paired 95% confidence interval excludes zero, and a negative mean Δ means the architecture leaves less non-random structure than Ridge. The *standard* group is the six benchmarks ETTh1, ETTh2, ETTm1, ETTm2, Weather and Exchange. The *large* group is Electricity and Traffic.

Comparison	Group	n cells	Mean Δ	Significant cells	Mean fraction $\Delta < 0$
DLinear − Ridge	standard	24	−0.56	13/24	0.59
DLinear − Ridge	large	16	−0.77	9/16	0.61
Transformer − Ridge	standard	24	−2.86	10/24	0.55
Transformer − Ridge	large	16	+0.18	9/16	0.38
iTransformer − Ridge	standard	24	+2.40	14/24	0.44
iTransformer − Ridge	large	16	−0.31	9/16	0.45
PatchTST − Ridge	standard	24	−7.83	18/24	0.70
PatchTST − Ridge	large	8	−0.63	1/8	0.62

On the six standard benchmarks PatchTST shows a clear negative difference (mean $\Delta = -7.83$, significant in 18 of 24 cells). It does leave less non-random structure in its residuals than the linear model, indicating that it captures nonlinear patterns the linear model cannot. However, on those same six datasets this additional structure does not change the accuracy ranking, on which the tuned linear model still matches or outperforms PatchTST in Table 1. Two further observations qualify a simple "more capacity captures more structure" interpretation. On Weather the difference is *positive* and significant at three of four horizons, indicating that PatchTST's residuals are *more* structured than the linear model's on that dataset. On the two large datasets, which include Traffic where PatchTST does achieve lower error, only one of eight comparisons is significant, so its advantage there is not explained by the capture of nonlinear structure.

In summary, the strongest nonlinear model does extract some patterns that the linear model does not capture, and the test detects them clearly on the six standard datasets. However, on those same datasets this additional structure does not translate into lower error. The tuned linear model still matches or outperforms the nonlinear one (Table 1). In other words, most of what determines accuracy on these benchmarks is the long-range linear signal documented in Appendix A, exploited through an appropriate context length as documented in Appendix B. The additional nonlinear structure that a higher-capacity model captures is real but yields little or no accuracy benefit here. On Traffic, where a higher-capacity model does achieve lower error, this test indicates that its advantage arises from a source other than the capture of nonlinear structure.

D Multi-run statistics, MAE, and cost

Table 6 reports the average MSE over the four horizons as mean $\pm$ standard deviation over three runs: three independent searches for SearchCast and three seeds for the rerun nonlinear baselines (TimeMixer on Electricity and Traffic has a single run). On the six datasets where SearchCast leads with full repeated runs for every method (the four ETT benchmarks, Exchange, and Weather), the intervals do not overlap. Table 7 reports per-horizon MSE and MAE for SearchCast and the nonlinear baselines of Table 1; MAE yields the same conclusion as MSE. Table 8 reports wall-clock cost. Our cost covers the entire pipeline: the full hyperparameter search (20 TPE trials per cell, each performing one closed-form fit per cross-validation fold and per candidate ridge coefficient), the final fit, and inference; each baseline is timed for a single training run with its already tuned configuration, whose tuning cost is not counted. All experiments run on the same node with one H200 GPU and a 72-core Grace CPU.

Table 6: Average MSE over horizons $\{96, 192, 336, 720\}$, mean $\pm$ standard deviation over three runs. †Single run.

Dataset	SearchCast	PatchTST	TimeMixer	PatchMLP
ETTh1	$\mathbf{0.4038 \pm 0.0026}$	0.4273 ± 0.0125	0.4370 ± 0.0146	0.4513 ± 0.0065
ETTh2	$\mathbf{0.3337 \pm 0.0008}$	0.3526 ± 0.0021	0.3591 ± 0.0038	0.3881 ± 0.0083
ETTm1	$\mathbf{0.3470 \pm 0.0004}$	0.3488 ± 0.0001	0.3562 ± 0.0008	0.3761 ± 0.0033
ETTm2	$\mathbf{0.2387 \pm 0.0018}$	0.2580 ± 0.0019	0.2593 ± 0.0025	0.2777 ± 0.0100
Exchange	$\mathbf{0.3467 \pm 0.0035}$	0.3685 ± 0.0133	0.3932 ± 0.0174	0.3868 ± 0.0128
Weather	$\mathbf{0.2150 \pm 0.0005}$	0.2239 ± 0.0007	0.2272 ± 0.0017	0.2299 ± 0.0009
Electricity	$\mathbf{0.1596 \pm 0.0003}$	0.1704 ± 0.0003	$0.1843^{\dagger}$	0.1730 ± 0.0017
Traffic	0.4070 ± 0.0001	$\mathbf{0.3975 \pm 0.0005}$	$0.4087^{\dagger}$	0.4455 ± 0.0062

E Details of the grouped search

Each dataset is partitioned into cells: horizon-step groups of size g_h crossed with series groups of size g_s. Each cell runs an independent search of 20 TPE trials over the preprocessing choices. Within a trial, α is swept over the fixed 21-point grid in closed form under 3-fold expanding-window cross-validation, with validation targets fixed across candidate L. The selected configuration is then refit in closed form on the full training split. Series in a group share both the configuration and the weights, because the closed-form fit pools their windows as samples; $g_s = 1$ recovers per-series weights. Horizon steps in a group share only the configuration, and each step has its own weight

Table 7: Per-horizon MSE / MAE under the matched evaluation protocol. SearchCast is the mean of three search runs; baselines are the runs reported in Table 1.

Dataset	H	SearchCast	PatchTST	TimeMixer	PatchMLP
ETTh1	96	0.365 / 0.390	0.376 / 0.397	0.374 / 0.402	0.396 / 0.417
	192	0.399 / 0.410	0.409 / 0.425	0.417 / 0.435	0.430 / 0.441
	336	0.424 / 0.425	0.431 / 0.444	0.438 / 0.450	0.456 / 0.457
	720	0.427 / 0.449	0.458 / 0.478	0.487 / 0.488	0.496 / 0.490
	Avg	0.404 / 0.419	0.419 / 0.436	0.429 / 0.444	0.445 / 0.451
ETTh2	96	0.269 / 0.332	0.274 / 0.337	0.291 / 0.359	0.302 / 0.351
	192	0.329 / 0.371	0.348 / 0.384	0.349 / 0.387	0.381 / 0.415
	336	0.353 / 0.394	0.377 / 0.415	0.403 / 0.430	0.416 / 0.440
	720	0.384 / 0.424	0.406 / 0.441	0.401 / 0.435	0.433 / 0.451
	Avg	0.334 / 0.381	0.351 / 0.394	0.361 / 0.403	0.383 / 0.414
ETTm1	96	0.296 / 0.344	0.290 / 0.343	0.293 / 0.345	0.309 / 0.357
	192	0.332 / 0.367	0.329 / 0.369	0.335 / 0.372	0.355 / 0.387
	336	0.361 / 0.386	0.362 / 0.390	0.368 / 0.386	0.393 / 0.409
	720	0.399 / 0.411	0.415 / 0.421	0.426 / 0.417	0.450 / 0.439
	Avg	0.347 / 0.377	0.349 / 0.381	0.356 / 0.380	0.377 / 0.398
ETTm2	96	0.161 / 0.254	0.165 / 0.255	0.164 / 0.256	0.173 / 0.264
	192	0.213 / 0.291	0.221 / 0.293	0.225 / 0.298	0.232 / 0.302
	336	0.257 / 0.322	0.276 / 0.328	0.277 / 0.332	0.294 / 0.345
	720	0.324 / 0.367	0.362 / 0.382	0.360 / 0.387	0.380 / 0.401
	Avg	0.239 / 0.308	0.256 / 0.314	0.257 / 0.318	0.270 / 0.328
Exchange	96	0.084 / 0.199	0.087 / 0.205	0.086 / 0.208	0.090 / 0.214
	192	0.171 / 0.292	0.176 / 0.300	0.173 / 0.295	0.186 / 0.311
	336	0.308 / 0.399	0.296 / 0.399	0.376 / 0.452	0.337 / 0.425
	720	0.824 / 0.683	0.853 / 0.696	0.874 / 0.703	0.893 / 0.711
	Avg	0.347 / 0.393	0.353 / 0.400	0.377 / 0.414	0.377 / 0.415
Weather	96	0.141 / 0.189	0.148 / 0.197	0.147 / 0.200	0.153 / 0.206
	192	0.183 / 0.232	0.191 / 0.240	0.194 / 0.245	0.196 / 0.244
	336	0.233 / 0.274	0.243 / 0.281	0.244 / 0.284	0.245 / 0.284
	720	0.303 / 0.324	0.313 / 0.333	0.319 / 0.332	0.321 / 0.338
	Avg	0.215 / 0.255	0.224 / 0.263	0.226 / 0.265	0.229 / 0.268
Electricity	96	0.131 / 0.226	0.143 / 0.247	0.153 / 0.256	0.140 / 0.240
	192	0.146 / 0.240	0.158 / 0.260	0.168 / 0.269	0.160 / 0.258
	336	0.162 / 0.257	0.168 / 0.266	0.188 / 0.290	0.179 / 0.278
	720	0.200 / 0.289	0.214 / 0.307	0.228 / 0.320	0.210 / 0.309
	Avg	0.160 / 0.253	0.171 / 0.270	0.184 / 0.284	0.172 / 0.271
Traffic	96	0.382 / 0.265	0.370 / 0.261	0.370 / 0.253	0.415 / 0.308
	192	0.395 / 0.270	0.386 / 0.269	0.399 / 0.270	0.433 / 0.315
	336	0.408 / 0.276	0.396 / 0.275	0.420 / 0.274	0.448 / 0.312
	720	0.443 / 0.295	0.435 / 0.295	0.445 / 0.295	0.480 / 0.338
	Avg	0.407 / 0.276	0.397 / 0.275	0.409 / 0.273	0.444 / 0.318

vector. Each cell thus fits one matrix $\mathbf{W} \in \mathbb{R}^{L \times g_h}$ applied to every series in its group. Nothing is shared across cells.

F Sensitivity to the search settings

We vary the number of cross-validation folds f and the number of trials per cell t around the default ($f = 3$, $t = 20$); Tables 9 and 10 report the average MSE (mean $\pm$ std over three runs). Accuracy saturates at about 15 trials. The fold count matters little except on Exchange, the smallest dataset,

Table 8: Wall-clock time in minutes. SearchCast: entire pipeline (search, final fit, and inference). Baselines: a single training run with the TFB-tuned configuration.

Dataset	SearchCast	PatchTST	TimeMixer	PatchMLP
ETTh1	0.3	1.6	10.1	1.5
ETTh2	0.2	2.2	13.7	2.1
ETTm1	0.7	9.7	35.7	6.0
ETTm2	0.8	7.2	26.2	6.0
Exchange	0.9	1.1	9.7	1.1
Weather	6.8	11.0	98.7	11.6
Electricity	5.6	90.2	1268.2	17.4
Traffic	8.9	283.1	2410.1	18.1

where $f = 3$ is clearly best; on the large Electricity and Traffic datasets, a single fold is as good and cheaper.

Table 9: Average MSE vs. number of cross-validation folds f ($t = 20$). **Bold**: best per row.

Dataset	$f = 1$	$f = 2$	$f = 3$ (default)	$f = 5$
ETTh1	$\mathbf{0.4033 \pm .0017}$	$0.4054 \pm .0014$	$0.4038 \pm .0026$	$0.4042 \pm .0023$
ETTh2	$0.3378 \pm .0040$	$0.3352 \pm .0002$	$\mathbf{0.3337 \pm .0008}$	$0.3353 \pm .0010$
ETTm1	$0.3474 \pm .0004$	$0.3472 \pm .0005$	$\mathbf{0.3470 \pm .0004}$	$0.3494 \pm .0008$
ETTm2	$0.2411 \pm .0016$	$0.2401 \pm .0008$	$\mathbf{0.2387 \pm .0018}$	$0.2389 \pm .0007$
Exchange	$0.3691 \pm .0045$	$0.3754 \pm .0021$	$\mathbf{0.3467 \pm .0035}$	$0.3867 \pm .0056$
Weather	$0.2163 \pm .0004$	$0.2155 \pm .0002$	$\mathbf{0.2150 \pm .0005}$	$0.2155 \pm .0008$
Electricity	$\mathbf{0.1587 \pm .0001}$	$0.1589 \pm .0001$	$0.1596 \pm .0003$	$0.1592 \pm .0001$
Traffic	$\mathbf{0.4039 \pm .0000}$	$0.4056 \pm .0003$	$0.4070 \pm .0001$	$0.4083 \pm .0004$

Table 10: Average MSE vs. number of trials per cell t ($f = 3$). **Bold**: best per row.

Dataset	$t = 10$	$t = 15$	$t = 20$ (default)	$t = 30$	$t = 50$
ETTh1	$0.4035 \pm .0024$	$0.4036 \pm .0024$	$0.4038 \pm .0026$	$0.4038 \pm .0027$	$\mathbf{0.4025 \pm .0011}$
ETTh2	$0.3336 \pm .0014$	$0.3334 \pm .0009$	$0.3337 \pm .0008$	$\mathbf{0.3333 \pm .0002}$	$0.3335 \pm .0002$
ETTm1	$0.3528 \pm .0054$	$0.3469 \pm .0002$	$0.3470 \pm .0004$	$0.3471 \pm .0003$	$\mathbf{0.3468 \pm .0004}$
ETTm2	$0.2441 \pm .0081$	$\mathbf{0.2381 \pm .0013}$	$0.2387 \pm .0018$	$0.2383 \pm .0015$	$0.2382 \pm .0004$
Exchange	$\mathbf{0.3447 \pm .0013}$	$0.3468 \pm .0040$	$0.3467 \pm .0035$	$0.3465 \pm .0031$	$0.3475 \pm .0024$
Weather	$0.2165 \pm .0010$	$0.2150 \pm .0008$	$0.2150 \pm .0005$	$0.2149 \pm .0005$	$\mathbf{0.2147 \pm .0005}$
Electricity	$0.1612 \pm .0020$	$0.1596 \pm .0004$	$0.1596 \pm .0003$	$\mathbf{0.1595 \pm .0002}$	$\mathbf{0.1595 \pm .0001}$
Traffic	$0.4097 \pm .0064$	$\mathbf{0.4066 \pm .0005}$	$0.4070 \pm .0001$	$\mathbf{0.4066 \pm .0001}$	$\mathbf{0.4066 \pm .0000}$

G Baseline configurations and searching the baselines

TFB (Qiu et al., 2024) tunes each (dataset, horizon) setting of each baseline separately. Table 11 shows ETTh2 as an example. Over the 32 settings, the tuned lookback is 336 or 512 in 28 of PatchTST's, 30 of TimeMixer's, and 26 of PatchMLP's. The cost of arriving at these configurations is not counted in Table 8.

To test whether the same search helps deep models, we give PatchTST and TimeMixer our 20-trial TPE budget over context length and model width, with depth fixed at 1 (Table 12). The searched configurations underperform the TFB ones at two to three orders of magnitude more wall-clock time than our full pipeline.

H Connection to Automated and hierarchical forecasting

Automated forecasting systems such as AutoGluon–TimeSeries (Shchur et al., 2023) and AutoTS (Catlin, 2025) search and ensemble across model families under a time budget, and Prophet (Taylor & Letham, 2018) automates the fitting of a fixed decomposable model. Our setting is narrower:

Table 11: TFB configurations on ETTh2. *Package default, used where TFB does not set the field; unmarked values are tuned per setting.

Model	H	Lookback	Batch	lr	Patience	Layers / d_{model}
PatchTST	96, 192, 336	512	32*	0.0001*	3*	2* / 32
PatchTST	720	336	32*	0.0001*	3*	2* / 32
TimeMixer	96	512	16	0.01	10	2 / 16
TimeMixer	192	336	64	0.0001	5	1 / 16
TimeMixer	336	512	16	0.01	10	2 / 16
TimeMixer	720	512	64	0.0001	5	1 / 64
PatchMLP	96	96	64	0.001	5*	1 / 1024
PatchMLP	192	512	64	0.001	5*	1 / 1024
PatchMLP	336	336	64	0.0001	5*	3 / 1024
PatchMLP	720	96	64	0.0001	5*	3 / 1024

Table 12: Searching the nonlinear baselines on ETTh1 and ETTh2. Average MSE and wall-clock minutes per dataset.

Method	Fits / trials	ETTh1 MSE	Cost	ETTh2 MSE	Cost
SearchCast (20 trials $\times$ 15 cells)	300 trials	**0.4038**	0.3	**0.3337**	0.2
PatchTST, TFB config	4 fits	0.4187	1.6	0.3511	2.2
PatchTST + 20-trial search	80 fits	0.4243	38.6	0.3656	37.7
TimeMixer, TFB config	4 fits	0.4292	10.1	0.3609	13.7
TimeMixer + 20-trial search	80 fits	0.4340	303.2	0.3782	315.9

the model class is fixed to closed-form Ridge regression and the search covers only preprocessing and regularization, which keeps each trial exact and cheap and makes the searched variables interpretable. The appropriate comparison is therefore with individually tuned forecasters under a shared protocol, rather than with AutoML systems whose search spaces and budgets are larger. Hierarchical forecasting (Hyndman et al., 2014; Hyndman & Athanasopoulos, 2021) enforces aggregation consistency across related series; our grouped search also exploits structure across series, but uses it to share hyperparameters rather than to impose coherence constraints.

I Limitations

Our study is limited to standard numeric long-horizon forecasting benchmarks, so the lookback-scaling, locality, and heterogeneity patterns we report should be read as findings for this regime rather than universal claims. Table 1 reports the mean of three search runs for our method and single runs for the baselines (multi-seed statistics in Appendix D); very small margins should therefore be treated as ties. Nonlinear baselines are rerun with the configurations tuned by TFB (Qiu et al., 2024), so comparisons reflect those configurations rather than a jointly retuned preprocessing protocol. Per-series search scales with channel count, which motivates our practical choice of g_h=48 and moderate g_s; finer-grained search may yield further gains. Finally, our claims are specific to Ridge with searched preprocessing. Applying the same preprocessing search to deeper models, and comparing all model classes under jointly tuned preprocessing, remains future work.

J Details of the Benchmark Datasets

We use existing public benchmark datasets, baseline implementations, and credit the original creators below. All assets are used unmodified and within their stated terms of use. Table 13 summarizes their statistics.

Benchmark datasets. The eight multivariate forecasting benchmarks used in Table 1 are obtained from the public benchmark distribution accompanying Wu et al. (2021) (`https://github.com/thuml/Autoformer`, MIT License). Their original sources and terms are:

Table 13: Statistics of the benchmark datasets. Each dataset is split chronologically into training, validation, and test sets at 6:2:2 for the ETT datasets and 7:1:2 for the others (Wu et al., 2021; Toner & Darlow, 2024). All methods are evaluated on every test window, without batch truncation.

Dataset	Variates	Timesteps	Forecast horizons	Frequency
ETTh1 / ETTh2	7	17,420	96, 192, 336, 720	1 hour
ETTm1 / ETTm2	7	69,680	96, 192, 336, 720	15 min
Electricity	321	26,304	96, 192, 336, 720	1 hour
Traffic	862	17,544	96, 192, 336, 720	1 hour
Weather	21	52,696	96, 192, 336, 720	10 min
Exchange	8	7,588	96, 192, 336, 720	1 day

- **ETTh1, ETTh2, ETTm1, ETTm2** (Electricity Transformer Temperature). Originally released by the authors of Informer at https://github.com/zhouhaoyi/ETDataset. Refer to the repository for the dataset's terms of use.
- **Electricity** (*ElectricityLoadDiagrams20112014*). UCI Machine Learning Repository, https://archive.ics.uci.edu/dataset/321/electricityloaddiagrams20112014, distributed under the Creative Commons Attribution 4.0 International (CC BY 4.0) license.
- **Traffic** (Caltrans PeMS road-occupancy data). California Department of Transportation, https://pems.dot.ca.gov/, made publicly available by Caltrans for research and operational use.
- **Weather**. Max Planck Institute for Biogeochemistry, https://www.bgc-jena.mpg.de/wetter/, publicly distributed by the institute.
- **Exchange Rate**. Released by the authors of LSTNet at https://github.com/laiguokun/multivariate-time-series-data under the MIT License.