SPECTRAL DYNAMICS OR STRUCTURED LINEAR MAPS? TINY SOURCE-ONLY TIME-SERIES TRANSFER

Anonymous authors
Paper under double-blind review

ABSTRACT

Can a model trained on one time-series dataset forecast an unseen dataset without target fitting? We study this question under a strict source-trained, target-free protocol using DynaFK, a compact forecaster that combines a seasonal anchor with a phase-dependent residual. At any fixed horizon, its encoder, diagonal recurrence, and decoder compose exactly into a phase-specific linear map. This motivates a matched direct control: with 397 real scalars it matches DynaFK at $H = 96$, while larger linear and spectral models can improve accuracy at higher storage cost. The recurrence's clearest benefits are horizon-independent parameter storage and execution across different channel counts. Source choice and horizon still change the ranking, so the results describe an accuracy–storage–portability trade-off rather than a universal advantage for spectral dynamics.

1 INTRODUCTION

Forecasting systems often face a deployment domain that differs from the training domain. The attractive promise is to train once on a source dataset and execute directly on a related target, without target labels or fine-tuning. We call this setting *source-trained, target-free transfer*. It is narrower than "zero-shot" use in foundation models: source task training is allowed, but target fitting and target model selection are not.

Three issues make evidence in this setting fragile. First, fitting a scaler on target training rows exposes target distributional information, even if no gradient update is performed. Second, a method may be selected by target validation while still being described as target-free. Third, models with channel-specific input/output layers cannot execute when the target channel count changes. These distinctions are especially important when an extremely small model is compared with large general forecasters.

Compact linear and frequency models already perform surprisingly well on forecasting benchmarks. Spectral and Koopman models add a different idea: they factor prediction through a latent dynamical evolution (Koopman, 1931; Mezić, 2005; Lusch et al., 2018; Liu et al., 2023; Wang et al., 2023). This creates a useful tension. Does the recurrence itself explain transfer performance, or would a direct structured map work just as well? The question is especially relevant for small models, where every parameter and interface constraint is visible.

We answer this question through a controlled study of DynaFK. The model combines a parameter-free seasonal forecast with a small channel-shared phase-spectral residual. Its source-period parameters can be interpolated on normalized physical phase, allowing execution at another sampling frequency; channel sharing permits execution when the number of variables changes. We study these properties as separate, measurable trade-offs.

Our contributions are:

- We define a strict single-source, target-free protocol that separates source learning, target inference, and evaluator-only target reference statistics.

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.

- We give an exact fixed-horizon reduction of DynaFK to a phase-specific linear map. This result motivates matched controls and limits what may be inferred from a complex latent recurrence.
- Across heterogeneous sources, channel counts, frequencies, and horizons, we identify an accuracy–storage–portability trade-off: direct structured maps explain the short-horizon accuracy, while the recurrence keeps parameter storage fixed and executes across channel counts.

Appendix Figure 4 gives a visual taxonomy of two common information-flow patterns under distribution shift.

2 Problem Setup

Task and Information Boundary

Let source data be split chronologically into $D_s^{\mathrm{tr}}, D_s^{\mathrm{val}}, D_s^{\mathrm{te}}$, and let D_t^{te} denote the target test interval. A valid learning algorithm may use only D_s^{tr} to fit parameters and only D_s^{val} for early stopping and hyperparameter selection. After the source checkpoint is fixed, inference receives each target input window $x_{1:L}^t$ and produces $\hat{x}_{L+1:L+H}^t$. No target labels, validation losses, gradient updates, or fine-tuning are available.

Input-window statistics are not target training information: they are functions of the observed inference window. We therefore allow per-window mean and variance, as in distribution-shift normalization methods (Kim et al., 2023; Fan et al., 2023). For comparable reporting across channels, we divide the *post-inference error in original units* by the standard deviation of each target channel fitted on the conventional target training split. This target reference scale is used only to report error; inference and checkpoint selection never see it.

Prediction versus report scale. Let $e_{h,c} = y_{h,c} - \hat{y}_{h,c}$ be an error already returned to original units and let $s_c > 0$ be the target reference standard deviation. The benchmark reports

$$\mathrm{MSE}_{\mathrm{ref}} = \frac{1}{HC} \sum_{h,c} \left(e_{h,c}/s_c \right)^2 . \tag{1}$$

Changing s_c cannot change a prediction, checkpoint, or early-stopping decision; it only changes the evaluator's channel weighting. We consequently store both original-unit and reference-scaled per-window errors. This is analogous to an evaluator using held-out labels to score a submission: the metric may know the target, while the submitted model does not.

3 Related Work

Forecasting under Shift

Forecasting architectures address non-stationarity through normalization, decomposition, or adaptive representations (Wang et al., 2022; Kim et al., 2023; Fan et al., 2023). The phrase "transfer forecasting," however, covers incompatible information regimes (Table 1). DAF jointly trains source and target forecasters (Jin et al., 2022), whereas feature/label-shift adaptation likewise observes the target domain during learning (He et al., 2023). They are valuable adaptation methods but are not admissible source-only baselines. AdaRNN matches distributions across temporal segments (Du et al., 2021), and FOIL learns invariance across environments inferred within training data (Liu et al., 2024); both primarily address future temporal shift rather than a frozen checkpoint executed on an external dataset. Domain-generalized forecasting learns from multiple observed source domains (Deng et al., 2024), an additional information budget absent in our one-source track. FreKoo combines spectral dynamics and Koopman evolution for temporal domain generalization, but likewise studies a different domain-information regime (Yu et al., 2025). Lightweight general and foundation-style forecasters may also use multi-source pretraining; LightGTS is a recent compact example (Wang et al., 2025). We therefore use *single-source, target-free transfer* rather than treating these regimes as one zero-shot leaderboard.

Table 1: Forecasting under shift is not one protocol. "Target in fit" means that samples from the deployment domain affect training or selection.

Regime	Fit domains	Target in fit	Examples
Source-only	one source	no	this study
Multi-source DG	≥ 2 sources	no	DGTSF
Target adaptation	source+target	yes	DAF
Temporal OOD	past segments	no future	AdaRNN, FOIL

Compact and Koopman Forecasting

DLinear challenges the assumption that more elaborate sequence models are always necessary (Zeng et al., 2023); FITS and SparseTSF further compress forecasting using frequency interpolation or sparse periodic structure (Xu et al., 2024; Lin et al., 2024b). CycleNet learns a recurrent periodic pattern, FilterNet learns frequency filters, TimeEmb separates static and dynamic components, and MixLinear targets extreme parameter efficiency (Lin et al., 2024a; Yi et al., 2024; Xia et al., 2025; Ma et al., 2026). Koopman predictors instead learn linear latent evolution (Liu et al., 2023; Wang et al., 2023). DynaFK's narrower object is a channel-shared, phase-indexed stable diagonal evolution whose parameter storage is independent of forecast horizon. This combination allows heterogeneous channel counts but removes cross-channel capacity. We therefore test the trade-off directly instead of attributing performance to any one ingredient.

Evaluation Methodology

Unified benchmarking exposes how preprocessing and evaluation choices change forecasting rankings (Qiu et al., 2024); large module-level studies likewise raise the standard for attributing gains to one component (Zhao et al., 2026). Our contribution is not benchmarking by itself. We use the exact fixed-horizon reduction to construct a direct containment control under one explicit deployment contract.

4 Source-Only Evaluation Protocol

We separate source training, checkpoint selection, target inference, and evaluator-side normalization as follows. A run follows five ordered stages:

1. fit the source scaler on D_s^{tr} and construct only source train/validation loaders;
2. train candidate checkpoints on D_s^{tr} and select one using D_s^{val};
3. freeze the selected state and record the source scaler and training history;
4. construct the target test loader and predict from observed target windows without enabling gradients; and
5. return predictions to original units, then apply the report-only target reference scale to the error.

This ordering separates model-visible information from evaluation metadata. Configuration, hashes, environment details, and per-window errors are listed in Appendix A. Hyperparameters shared by all target domains are fixed before stage 4. In particular, K is selected from source validation, the daily period is read from timestamps, and the optimization schedule is constant across methods.

5 Method: DynaFK

DynaFK forecasts in two parts. A parameter-free seasonal anchor repeats the latest observed period. A small learned residual model groups normalized history by physical phase, compresses each phase trace into K complex coefficients, evolves those coefficients forward, and decodes the residual forecast. The learned maps are shared across channels, so one checkpoint can execute on targets with different numbers of variables.

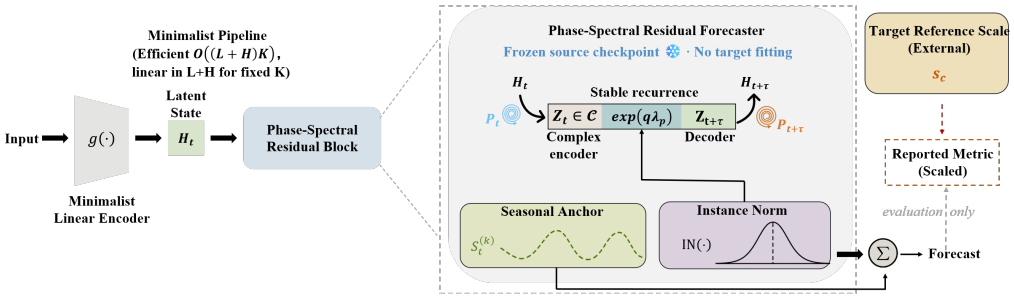


Figure 1: DynaFK inference and reporting flow. A source-selected frozen checkpoint applies a channel-shared phase-spectral residual forecaster without target fitting or alignment loss. Instance normalization uses only the observed input window, while the seasonal anchor bypasses the residual path and is added to its decoded output. The target reference scale s_c is external metadata applied only to post-inference errors when computing the reported metric. For fixed K, inference costs $\mathcal{O}((L+H)K)$.

Seasonal Residual Formulation

For a window $x \in \mathbb{R}^{L\times C}$, let μ and σ be its per-channel temporal mean and standard deviation. We normalize $\tilde{x} = (x-\mu)/(\sigma+\epsilon)$. Given a physical period P, the parameter-free anchor $a_{1:H}$ repeats the last observed period. The final forecast is

$$\hat{x}_{1:H} = a_{1:H} + \sigma \odot \hat{r}_{1:H}, \tag{2}$$

where $\hat{r}$ is a learned normalized residual. The anchor prevents a tiny model from spending capacity on copying the dominant daily cycle and makes seasonal naive the explicit zero-residual reference. Figure 1 shows the forecast path and separates model-visible window normalization from the external target reference scale used only for reporting.

Phase-Spectral Residual

We rearrange the normalized source input by absolute phase. If $I_s = L_s/P_s$, then $s_p \in \mathbb{R}^{I_s}$ contains the observed source cycles at phase p. A channel-shared complex encoder maps each channel independently:

$$z_{p,k} = \sum_{i=1}^{I} s_{p,i} E_{p,i,k}, \qquad E \in \mathbb{C}^{P\times I\times K}. \tag{3}$$

The continuous diagonal evolution is constrained to be stable,

$$\lambda_{p,k} = -\operatorname{softplus}(\alpha_{p,k}) + \mathrm{i}\omega_{p,k}, \tag{4}$$

and the residual for future cycle $q \geq 1$ is

$$\hat{r}_{p,q} = \Re\left[\sum_{k=1}^{K} z_{p,k} \exp(q\lambda_{p,k}) d_{p,k}\right], \qquad d \in \mathbb{C}^{P\times K}. \tag{5}$$

A short trainable one-dimensional smoothing kernel precedes the phase rearrangement. This construction is Floquet-inspired because its transition parameters depend periodically on physical phase; the learned representation need not satisfy the assumptions of the classical Floquet theorem.

What Is Spectral about a Fixed-Horizon Predictor?

Proposition 1 (fixed-horizon residual collapse). Once the forecast horizon is fixed, the encoder, diagonal recurrence, and decoder compose into one linear map for each physical phase. Fix P, the encoder width I_s, $Q = \lceil H/P \rceil$, and all DynaFK parameters. There exists a real tensor $A \in \mathbb{R}^{P\times Q\times I_s}$ such that, for every normalized, smoothed phase input,

$$\hat{r}_{p,q} = \sum_{i=1}^{I_s} A_{p,q,i} s_{p,i}. \tag{6}$$

Specifically, $A_{p,q,i} = \Re[\sum_k E_{p,i,k} \exp(q\lambda_{p,k}) d_{p,k}]$. This follows by substituting the encoder into Equation 5 and exchanging the two finite sums. Chronological decoding selects the first H phase/cycle coordinates, so the statement also covers a final incomplete cycle such as $H < P$. The result is for the normalized, smoothed residual path; the seasonal anchor and inverse scaling remain outside this linear map.

At a fixed horizon, the spectral recurrence is therefore a factorization of a phase-specific linear operator. Our phase-linear control directly learns A and contains every fixed-horizon map expressible by DynaFK under the same normalization and smoothing. DynaFK remains structurally distinct because its recurrence can be evaluated at additional q without adding coefficients. That can save storage across horizons, but need not improve fixed-horizon accuracy. The exact collapse motivates the mechanism control in Section 6. As a numerical check, constructing A directly from a trained checkpoint reproduces its predictions with maximum absolute error 4.8×10^{-7} on 32 random seven-channel windows; no parameters are refitted for this check.

The default $L = 96$, hourly $P = 24$, and $K = 4$ give 685 trainable tensor elements: PI_sK complex encoder elements, PK complex decoder elements, $2PK$ real spectral elements, and 13 real smoothing elements. Because each complex element stores two real numbers, the fair count is 1,165 real scalar degrees of freedom (4.55 KiB in single precision), versus 9,312 real scalars (36.38 KiB) for the $96 \to 96$ Linear baseline. We report both counts because a complex element stores two real values; complete accounting appears in the supplement.

Frequency and Dimension Portability

The physical daily period is derived from timestamps: $P = 24$ for hourly data, $P = 96$ for 15-minute data, and $P = 144$ for ten-minute data. When the target period differs, E, d, and λ are linearly interpolated on the circular coordinate p/P, including wrap-around. If the target window has $I_t = L_t/P_t \neq I_s$ observed cycles, each target phase trace is linearly resampled from length I_t to the source encoder width I_s before applying E. The model generates $\lceil H/P_t \rceil$ future cycles and chronological decoding retains the first H samples. This specifies execution but not a frequency-invariant observation or forecast duration. We therefore predefine and report both a fixed 96-hour history (384 target steps on ETTm) and a fixed 96-step history; in both cases $H = 96$ means target samples.

All learned maps act independently and identically on channels, so a checkpoint can execute for any target channel count. We isolate this portability trade-off with a dimension-specific channel-mixing Linear baseline.

Training

We minimize prediction MSE on source training windows and select the checkpoint by source validation MSE. There is no alignment term. We investigated Sinkhorn alignment between adjacent phase spectra and an L2 smoothing control, but neither changed transfer error (Section 6); retaining alignment in the objective would therefore add an unsupported mechanism.

6 Experiments

Protocol

Data. ETTh1 and ETTh2 contain seven hourly transformer-temperature variables; ETTm1 and ETTm2 are sampled every 15 minutes (Zhou et al., 2021). We additionally use hourly Electricity (ECL, 321 variables) and Traffic (862 variables). The ETT pair isolates related same-channel transfer, ECL/Traffic test heterogeneous dimensions, and ETTm/Weather test changed sampling intervals. Chronological train/validation/test lengths follow the standard benchmark splits. ETTh1, ETTh2, or ECL serves as the sole source in a given experiment. The ETTh1-source $H = 96$ submatrix was rerun from the current implementation; additional source directions and horizons are documented in the appendix.

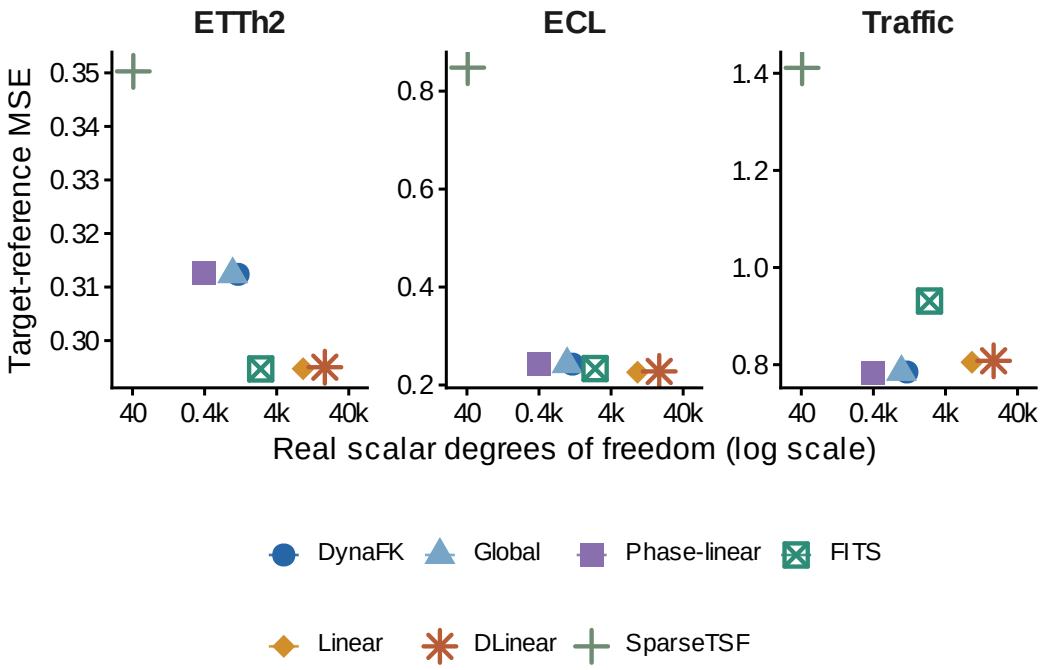


Figure 2: Accuracy–size trade-off at $H = 96$ from an ETTh1-only source. Points show mean target-reference MSE over three training seeds and vertical ranges show sample standard deviation (usually smaller than the marker). The horizontal axis counts real scalar degrees of freedom, including both real components of complex parameters. Target-train standard deviations are used only after inference to express the metric. The direct phase-linear control overlaps DynaFK with fewer scalars. FITS and Linear improve two targets for more storage, while their Traffic ranking reverses between MSE and MAE.

Normalization and metrics. For matching ETT channel names, a source-train scaler is applied before input-window normalization. For heterogeneous targets, the model operates on the raw target window followed by the same instance normalization. Predictions are returned to original units before target-reference normalized MSE and MAE are computed. Thus the target reference scale changes only the units of the reported error. We also retain original-unit errors, which read no target training rows, and evaluate whether the evaluator scale changes method rankings (Appendix Table 11).

Optimization and baselines. All learned models use Adam (Kingma & Ba, 2014), learning rate 10^{-3}, batch size 128, at most 40 epochs, and patience 8. We run seeds 2023–2025. Portable Linear, DLinear, and SparseTSF share the seasonal anchor, window normalization, splits, and early stopping. FITS follows its published mean-centering and complex frequency interpolation; its low-pass cutoff is the source daily frequency and its complex map is shared across channels (Xu et al., 2024). This configuration has 1,200 complex tensor elements, or 2,400 real scalars. The official Koopa implementation uses its published architecture and is evaluated only when source and target both have seven channels. The channel-mixing Linear applies a learned 7×7 mixer after its temporal map and is likewise restricted to ETTh1→ETTh2. N/A therefore denotes structural incompatibility, not an omitted result. $K = 4$ is selected by the source-validation elbow; $P = 24/96/144$ follows the timestamp interval; alignment is absent; and no rule is selected separately for a target.

Can Tiny Source-Only Models Transfer?

Figure 2 shows three conclusions; exact MSE/MAE and sample SD appear in the supplementary transfer tables. First, DynaFK is not the most accurate or the smallest method. FITS and Linear re-

Table 2: A non-ETT source: models are trained and selected only on ECL, then evaluated on three unseen targets at $H = 96$. Values are three-seed mean target-reference MSE.

Method	ETTh1	ETTh2	Traffic
Seasonal naive	.5122	.3905	1.2171
DynaFK	.4087	.3351	.7260
Linear	.4068	**.3206**	.7157
DLinear	**.4065**	.3207	**.7155**

Table 3: Paired DynaFK minus Linear difference at $H = 96$ with 95% moving-block bootstrap intervals (block length 192). Positive favors Linear; negative favors DynaFK.

Target	MSE Δ [95% CI]	MAE Δ [95% CI]
ETTh2	.0176 [.0090, .0279]	.0096 [.0052, .0146]
ECL	.0166 [.0146, .0189]	.0171 [.0154, .0188]
Traffic	$-.0201$ [$-.0266$, $-.0133$]	.0018 [$-.0003$, .0039]

duce MSE by about 5.7% on ETTh2; Linear also reduces ECL MSE by 6.8%. FITS uses 2.1 times, Linear 8.0 times, and Koopa about 37 times DynaFK's real-scalar storage. Second, DynaFK's own fixed-horizon parameterization is inefficient: the 397-scalar phase-linear control preserves its accuracy. Extreme compression still has a cost, because 41-parameter SparseTSF is worse by 11% on ETTh2 and much more on ECL/Traffic. Third, ranking is target dependent. DynaFK has 2.5% lower Traffic MSE than Linear and 15.7% lower than FITS, but their Traffic MAE difference includes zero under the block bootstrap. These results reveal an accuracy–storage trade-off with no uniformly best model. Evaluator scaling also matters: on Traffic, target-reference MAE ranks Linear before DynaFK, whereas original-unit MAE reverses this order by a small margin. The two metrics therefore give a small, metric-dependent ranking on Traffic.

Changing the source changes the result. With ETTh2 as source, DynaFK obtains .4328/.3330/.8402 MSE on ETTh1/ECL/Traffic, versus .4276/.2777/.8993 for Linear. Koopa fails to beat seasonal naive on ETTh2→ETTh1 (.6143 versus .5122). Full source-direction results appear in the supplement. The asymmetry shows that transfer quality depends on source suitability.

Table 2 reverses the heterogeneous-domain transfer: the source now has 321 channels and is not an ETT dataset. Every learned model improves on seasonal naive after the checkpoint is frozen, so useful transfer is not confined to an ETT source. The compact direct maps again match or beat DynaFK, however. The experiment therefore supports a limited source portability while exposing the same accuracy–capacity trade-off.

Uncertainty with Overlapping Windows

Seed-level SD alone does not reflect temporal dependence because adjacent test windows overlap. We therefore compare DynaFK and seasonal Linear using paired per-window errors. For each target, we average the paired difference over the three matched seeds and apply a circular moving-block bootstrap with 20,000 replicates. The primary block length is $L + H = 192$ windows; lengths 96 and 384 give the same qualitative conclusions. This analysis is conditional on the three trained checkpoints and therefore does not describe uncertainty across all possible training sets.

Table 3 confirms the target-dependent trade-off. Linear is reliably better on both metrics for ETTh2 and ECL. DynaFK is reliably better on Traffic MSE, whereas its Traffic MAE difference includes zero once temporal dependence is respected, supporting a target-dependent interpretation.

Do Spectral Dynamics Explain Fixed-Horizon Accuracy?

Phase-spectral structure. At $H = 96$, a global-spectrum version (981 real scalars) and a relaxed phase-linear residual (397 scalars) match DynaFK within 2.4×10^{-4} MSE on every target (Figure 3(b)). The exponential diagonal constraint therefore does not explain the accuracy in these

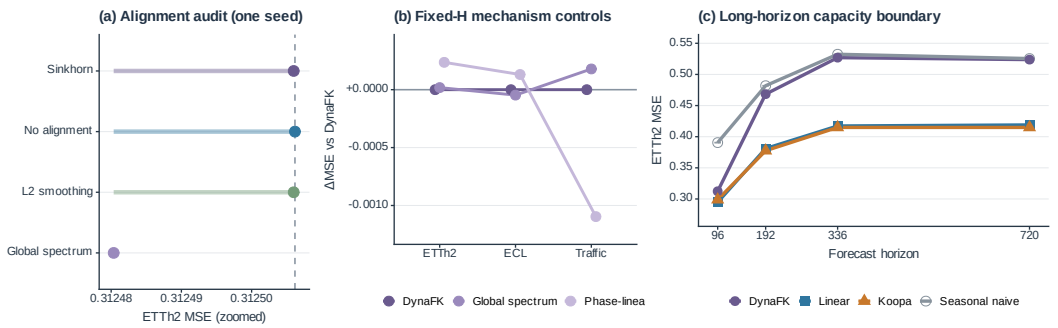


Figure 3: Mechanism and capacity study. (a) In a matched one-seed ETTh1→ETTh2 diagnostic, Sinkhorn, L2 smoothing, and no alignment agree at the reported precision; the axis is deliberately zoomed. (b) Three-seed fixed-horizon controls are shown as MSE difference from DynaFK on each target. (c) ETTh1→ETTh2 mean MSE over three seeds across horizons; vertical ranges are sample SD. DynaFK keeps 1,165 real scalars in each separately trained horizon configuration, whereas Linear and Koopa add horizon-specific output parameters. This is parameter-shape evidence, not a test of one frozen multi-horizon checkpoint.

data. Its concrete benefit is horizon-independent parameter count; whether that constraint helps on systems with stronger time-varying spectra remains open.

Alignment. In a matched exploratory ETTh1→ETTh2 run, Sinkhorn spectral alignment, no alignment, and L2 phase smoothing yield MSE 0.3125060, 0.3125062, and 0.3125060, respectively; a global spectrum yields 0.3124804. A λ sweep is similarly flat. These differences are far below between-seed variation, so alignment is omitted from the final method.

Selecting K without target validation. For $K = 1, 2, 4, 8$, source-validation MSE is 0.8164 ± 0.0002, 0.8162 ± 0.0002, 0.7383 ± 0.0003, and 0.7380 ± 0.0001. The corresponding ETTh2 test MSE is 0.3811, 0.3809, 0.3124, and 0.3123. Thus $K = 4$ is the smallest source-supported value after the large validation improvement; $K = 8$ nearly doubles size without material source or target improvement. P is derived from timestamp frequency rather than selected on a target.

What Does the Recurrence Provide?

Figure 3(c) reveals a capacity boundary in the archived horizon study. Each horizon uses a separately source-trained checkpoint. DynaFK's parameter count does not grow with H, but its gain over seasonal naive contracts from 20% at $H = 96$ to 1.0%, 1.0%, and 0.3% at $H = 192, 336, 720$. Linear and Koopa add horizon-specific output parameters and retain substantially lower error. A fixed $K = 4$ diagonal recurrence is consequently unsuitable when long-horizon accuracy is more important than fixed checkpoint size.

The fixed-horizon relationship also persists on daily Exchange Rate data, where the target period is $P = 1$. The independently trained 397-scalar phase-linear model remains close to DynaFK (0.0968 versus 0.0952 target-reference MSE), and algebraic collapse reproduces DynaFK's predictions within 1.19×10^{-7} maximum absolute error. Persistence is more accurate than either learned model (0.0863 MSE), separating the fixed-horizon explanation from transfer accuracy on this target (Appendix E).

For the archived cross-frequency ETTh1→ETTm transfer at $H = 96$, the predeclared 96-hour history gives ETTm1/ETTm2 MSE 0.3880 ± 0.0085/0.2578 ± 0.0058, while a fixed 96-step history gives 0.3923 ± 0.0019/0.2404 ± 0.0005. Their MAE rankings also differ. Physical phase interpolation and lookback duration are separate design choices, and neither history convention dominates. We report both history conventions as a pre-specified sensitivity.

Table 4 shows why a single aggregate "cross-frequency win" would be misleading. The physical-history protocol has the best ETTm1 MSE but worse ETTm1 MAE than seasonal naive, while fixed-

Table 4: Cross-frequency ETTh1→ETTm results at $H = 96$, averaged over three seeds. Both history conventions were specified independently of target test results.

History	m1 MSE	m1 MAE	m2 MSE	m2 MAE
96 h	**.3880**	.3967	.2578	.3175
96 steps	.3923	**.3844**	**.2404**	**.2909**
Seasonal naive	.4227	.3865	.2631	.3005

step history is best on both ETTm2 metrics. A deployer must specify whether the observation budget is measured in elapsed time or samples before target evaluation.

Appendix Table 15 compares direct compact and Koopman controls under the 96-hour input protocol. No method wins all targets, and $H = 96$ denotes target samples, so the experiment establishes executability and sensitivity for the stated history conventions.

7 Discussion

The fixed-horizon result changes how we read DynaFK. Its encoder, recurrence, and decoder form a structured factorization of a phase-specific linear predictor. The 397-scalar phase-linear control matches DynaFK at $H = 96$, showing that the observed short-horizon accuracy is explained by this direct map under the same preprocessing. The experiments therefore separate the predictive map from the factorization used to parameterize it.

The factorization is still useful when the deployment constraint is structural. Its parameter count stays fixed as the forecast horizon changes, and channel sharing lets the same interface run from seven source variables to targets with 321 or 862 variables. These properties favor the recurrent model when storage or channel compatibility matters; FITS or Linear is often preferable when fixed-horizon accuracy justifies additional parameters. Compact storage alone does not guarantee lower latency or memory use (Appendix G).

More broadly, source-only transfer is shaped by the source, target, metric, and horizon. Linear is more accurate on ETTh2 and ECL, whereas DynaFK has lower Traffic MSE, and long-horizon gains approach seasonal naive. Cross-frequency results likewise depend on whether the observation budget is measured in hours or samples. A useful comparison should therefore state these choices before evaluating the target.

Most of our benchmarks are strongly seasonal sensor datasets, and the number of source domains remains limited. We do not study probabilistic forecasts, missing observations, abrupt semantic changes, or general multivariate interfaces. Cross-frequency execution uses phase interpolation and resampling, H counts target samples, and three seeds with block bootstrap do not describe all future deployment domains. The appendix records the reproduced and earlier-run results separately.

8 Conclusion

Under a strict source-trained, target-free protocol, DynaFK is a portable factorization. Its clearest benefit is horizon-independent storage; the direct 397-scalar control retains its short-horizon accuracy, while DynaFK approaches seasonal naive at long horizons. Spectral structure therefore does not remove the accuracy–capacity trade-off.

Reproducibility Statement

Appendix A records the information boundary, configurations, dataset splits, parameter accounting, and additional results. The released artifacts contain run configurations, training histories, hashes, and per-window errors. Using the current frozen runners and identical data hashes, CPU reruns reproduce the ETTh1-source $H = 96$ DynaFK, phase-linear, Linear, and FITS target-reference cells within 2.98×10^{-8}. A follow-up GPU rerun also reproduces the channel-compatible Koopa cells exactly; the earlier CPU Koopa rerun is retained as a device-sensitivity diagnostic. The appendix

identifies which additional tables come from earlier runs; no missing value is inferred. Table 7 summarizes this evidence boundary.

AI USE STATEMENT

Generative AI tools assisted literature search, experimental planning and review, code development and debugging, reproducibility checks, and manuscript editing. Citation metadata and all scientific text, numerical results, and claims were reviewed and verified by the authors. Results were obtained from recorded executable pipelines and manifests; language models did not generate the numerical results or synthetic data. The authors take responsibility for the final manuscript and artifacts.

REFERENCES

Songgaojun Deng, Olivier Sprangers, Ming Li, Sebastian Schelter, and Maarten de Rijke. Domain generalization in time series forecasting. *ACM Transactions on Knowledge Discovery from Data*, 18(5):113:1–113:24, 2024. doi: 10.1145/3643035.

Yuntao Du, Jindong Wang, Wenjie Feng, Sinno Pan, Tao Qin, Renjun Xu, and Chongjun Wang. AdaRNN: Adaptive learning and forecasting of time series. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, pp. 402–411, 2021. doi: 10.1145/3459637.3482315.

Wei Fan, Pengyang Wang, Pengfei Li, Songcan Li, Lei Meng, Zhaoyang Wang, and Jian Sun. DishTS: A general paradigm for alleviating distribution shift in time series forecasting. In *International Conference on Learning Representations*, 2023.

Huan He, Owen Queen, Teddy Koker, Consuelo Cuevas, Theodoros Tsiligkaridis, and Marinka Zitnik. Domain adaptation for time series under feature and label shifts. In *International Conference on Machine Learning*, 2023.

Xiaoyong Jin, Youngsuk Park, Danielle Maddix, Hao Wang, and Yuyang Wang. Domain adaptation for time series forecasting via attention sharing. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 10280–10297, 2022.

Taesung Kim, Jinhee Kim, Yunwon Tae, Johannes Park, and Jaegul Choi. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2023.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Bernard O. Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931.

Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. CycleNet: Enhancing time series forecasting through modeling periodic patterns. In *Advances in Neural Information Processing Systems*, volume 37, 2024a. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/bfe7998398779dde03cad7a73b1f81b6-Abstract-Conference.html.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Haojun Chen, and Junjie Yang. SparseTSF: Modeling long-term time series forecasting with 1k parameters. In *International Conference on Machine Learning*, 2024b.

Haoxin Liu, Harshavardhan Kamarthi, Lingkai Kong, Zhiyuan Zhao, Chao Zhang, and B. Aditya Prakash. Time-series forecasting for out-of-distribution generalization using invariant learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 31312–31325, 2024.

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*, volume 36, 2023.

Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1):4950, 2018.

Aitian Ma, Dongsheng Luo, and Mo Sha. MixLinear: Extreme low resource multivariate time series forecasting with 0.1k parameters. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/7db2348b5bfeca620aa7327df815adcc-Abstract-Conference.html.

Igor Mezić. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41(1–3):309–325, 2005.

Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. TFB: Towards comprehensive and fair benchmarking of time series forecasting methods. *Proceedings of the VLDB Endowment*, 17 (9):2363–2377, 2024. doi: 10.14778/3665844.3665863.

Rui Wang, Yihe Dong, Sercan Ö. Arik, and Rose Yu. Koopman neural forecaster for time series with temporal distribution shifts. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=kUmdmHxK5N.

Yihang Wang, Yuying Qiu, Peng Chen, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Chenjuan Guo. LightGTS: A lightweight general time series forecasting model. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 64109–64126, 2025. URL https://proceedings.mlr.press/v267/wang25ch.html.

Yuxuan Wang, Haixu Wu, and Mingsheng Long. Non-stationary transformers: Exploring the lens of non-stationarity in time series forecasting. In *Advances in Neural Information Processing Systems*, volume 35, pp. 26898–26910, 2022.

Mingyuan Xia, Chunxu Zhang, Zijian Zhang, Hao Miao, Qidong Liu, Yuanshao Zhu, and Bo Yang. TimeEmb: A lightweight static-dynamic disentanglement framework for time series forecasting. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/6e79ae23fe8a8bb15a2b2972053f4fc3-Abstract-Conference.html.

Zhijian Xu, Ailing Zeng, and Qiang Xu. FITS: Modeling time series with 10k parameters. In *International Conference on Learning Representations*, 2024.

Kun Yi, Jingru Fei, Qi Zhang, Hui He, Shufeng Hao, Defu Lian, and Wei Fan. FilterNet: Harnessing frequency filters for time series forecasting. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/6323d96f79d5d49e0d3fc88835c082cd-Abstract-Conference.html.

En Yu, Jie Lu, Xiaoyu Yang, Guangquan Zhang, and Zhen Fang. Learning robust spectral dynamics for temporal domain generalization. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/da9390db30d0c9b38efd85360e38acc4-Paper-Conference.pdf.

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.

Zhiyuan Zhao, Juntong Ni, Shangqing Xu, Haoxin Liu, Wei Jin, and B. Aditya Prakash. TimeRecipe: A time-series forecasting recipe via benchmarking module level effectiveness. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/f46ddea413df86832418c5e04e59644f-Abstract-Conference.html.

Haoyi Zhou, Shanghang Zhang, Jianxin Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11106–11115, 2021.

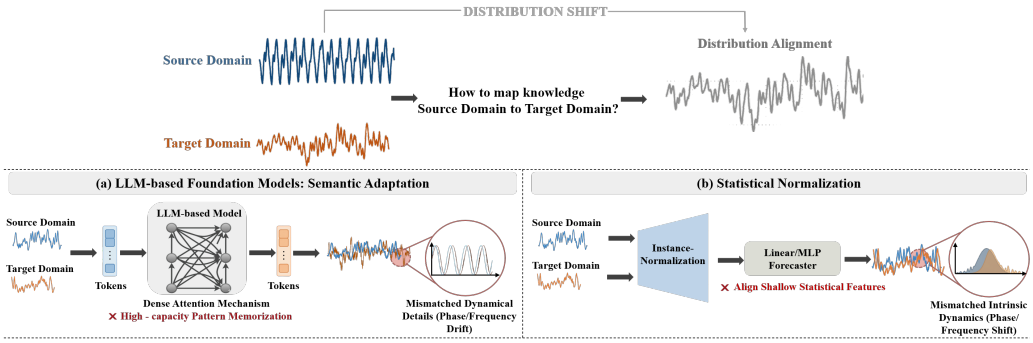


Figure 4: Conceptual forecasting paradigms under distribution shift. Panel (a) represents high-capacity pretrained adaptation, and panel (b) represents observed-window normalization with a compact forecaster. Both can leave phase- or frequency-dependent residual mismatch. The diagram is a taxonomy of information flow, not an empirical ranking.

A Protocol and Provenance

Evidence boundary. The current-runner reproduction covers the ETTh1-source $H = 96$ DynaFK, phase-linear, Linear, faithful FITS, and GPU Koopa cells listed in Table 7. All other numerical extensions in this appendix are archived historical artifacts from the earlier experiment campaign. They are retained for transparency, but are not claimed to have been regenerated from the current visible runner. Experiments with no artifact—including CycleNet, MixLinear, and the complete component matrix— have no reported value.

Information Boundary

Every reported local experiment satisfies the following executable contract. Model parameters are fitted on the source training split. Early stopping and the choice of K use the source validation split. Target loaders are constructed only after a source checkpoint is fixed and hashed. Target labels, target validation, gradients, and fine-tuning are unavailable. The observed target input window supplies only its own mean, variance, timestamps, and recent seasonal cycle.

The target-training standard deviation appears only in the evaluator. After a prediction is returned to original physical units, the per-channel error is divided by this reference standard deviation and then averaged. It neither enters the model nor changes checkpoint selection. For heterogeneous targets, the target values are not transformed by a source-channel scaler because the channel sets differ; input-window instance normalization remains available.

Table 5: Chronological split sizes and sampling intervals.

Dataset	Interval	Train	Validation	Test
ETTh1	60 min	8,640	2,880	2,880
ETTh2	60 min	8,640	2,880	2,880
ETTm1	15 min	34,560	11,520	11,520
ETTm2	15 min	34,560	11,520	11,520
ECL	60 min	18,412	2,632	5,260
Traffic	60 min	12,280	1,756	3,508
Weather	10 min	36,887	5,270	10,539

Final Configurations

Run Artifacts

Each run directory contains: (i) a JSON configuration including the leakage contract; (ii) the source scaler; (iii) training history and source-validation loss; (iv) the selected checkpoint and SHA-256

Table 6: Final source-training configurations. Values are fixed across targets; "daily" means the period computed from timestamps.

Component	Final configuration
Common	$L = H = 96$; Adam; learning rate 10^{-3}; batch 128; 40 epochs maximum; patience 8; stride 1; seeds 2023–2025
DynaFK	$K = 4$; daily P; smoothing width 13 on hourly sources; seasonal anchor; instance normalization; no alignment loss
Phase-linear	direct $A \in \mathbb{R}^{P\times(H/P)\times(L/P)}$; otherwise the DynaFK adapter
Linear	shared real $96 \to H$ map; seasonal anchor; instance normalization
DLinear	two shared $96 \to H$ maps; moving-average width 25; seasonal anchor; instance normalization
SparseTSF	daily period; smoothing width 25 on hourly sources; one cycle-to-cycle shared map
FITS	mean-centering and mean anchor; cutoff 24; complex $24 \to 48$ frequency map at $L = H = 96$
Koopa	official code; segment 24; one block; dynamic width 32; hidden width 64; two hidden layers; source spectral-mask fraction 0.2

Table 7: Frozen-runner reproduction check for the ETTh1-source, $H = 96$ claim-bearing submatrix. Δ is the maximum absolute difference in target-reference MSE/MAE against the historical manifest over three seeds. CPU/GPU checkpoint bytes are not expected to match.

Method	Re-evaluated targets	max $\lvert\Delta\rvert$	Status
DynaFK	ETTh2, ECL, Traffic	0	reproduced
phase-linear	ETTh2, ECL, Traffic	2.98×10^{-8}	reproduced
Linear	ETTh2, ECL, Traffic	2.98×10^{-8}	reproduced
FITS-faithful	ETTh2, ECL, Traffic	2.98×10^{-8}	reproduced
Koopa	ETTh2	0	reproduced

digest; (v) source and target data hashes; (vi) GPU, CUDA, PyTorch, Python, and operating-system versions; and (vii) compressed per-window arrays containing start index, MSE, MAE, seasonal-naive error, and original-unit error. The aggregate tables are generated from these manifests rather than copied from console output.

The historical principal environment is Ubuntu Linux, Python 3.12.13, PyTorch 2.5.1 with CUDA 12.4 and cuDNN 9.1, and one NVIDIA RTX 3090 Ti (24 GiB). Deterministic seeds are 2023, 2024, and 2025. cuDNN benchmarking is disabled and deterministic mode is enabled. The post-migration provenance audit reran the ETTh1-source $H = 96$ claim-bearing submatrix on CPU from the current runner and exact recorded data hashes. Table 7 records the numerically reproduced cells; a separate CPU Koopa run is retained as a device-sensitivity diagnostic.

B Complete H=96 Results

Table 8: ETTh1-source results at $H = 96$. Each cell is target-reference MSE/MAE, mean±sample SD over three seeds. Parameter elements follow PyTorch `numel`; DynaFK has 685 elements but 1,165 real scalar degrees of freedom, while FITS has 1,200 complex elements and 2,400 real scalar degrees of freedom. N/A indicates structural channel-dimension incompatibility.

	Params	ETTh2		ECL		Traffic	
Method	elems.	MSE	MAE	MSE	MAE	MSE	MAE
SparseTSF	41	.3503±.0007	.3861±.0003	.8479±.0005	.7622±.0001	1.4112±.0010	.8051±.0003
Phase-linear	397	.3126±.0001	.3506±.0001	.2427±.0004	.3240±.0007	.7838±.0001	.4620±.0006
Global spectrum	501	.3124±.0001	.3505±.0000	.2426±.0003	.3233±.0007	.7851±.0008	.4619±.0007
DynaFK	685	.3124±.0001	.3505±.0000	.2426±.0004	.3234±.0006	.7849±.0012	.4619±.0007
FITS	1,200	.2947±.0000	.3419±.0001	.2335±.0003	.3185±.0005	.9310±.0010	.5655±.0005
Linear	9,312	.2947±.0004	.3409±.0004	.2260±.0007	.3063±.0010	.8050±.0019	.4601±.0010
Channel Linear	9,361	.2958±.0004	.3423±.0002	N/A	N/A	N/A	N/A
DLinear	18,624	.2950±.0010	.3408±.0005	.2277±.0020	.3076±.0016	.8079±.0051	.4619±.0020
Official Koopa	43,592	.2989±.0021	.3446±.0011	N/A	N/A	N/A	N/A

Table 9: ETTh2-source results at $H = 96$, mean±sample SD over three seeds.

	ETTh1		ECL		Traffic	
Method	MSE	MAE	MSE	MAE	MSE	MAE
DynaFK	.4328±.0012	.4244±.0008	.3330±.0013	.4084±.0011	.8402±.0068	.5211±.0037
Linear	.4276±.0015	.4248±.0007	.2777±.0025	.3704±.0012	.8993±.0086	.5357±.0022
DLinear	.4274±.0011	.4250±.0003	.2791±.0037	.3721±.0025	.8929±.0049	.5352±.0016
Official Koopa	.6143±.0118	.5234±.0051	N/A	N/A	N/A	N/A

Table 10: ECL-source results at $H = 96$, mean±sample SD over three seeds. The source has 321 variables; the targets have 7, 7, and 862 variables. FITS-residual denotes the frequency interpolator placed inside the shared seasonal/instance-normalization adapter, and is therefore a diagnostic rather than the faithful FITS configuration in Table 8.

	ETTh1		ETTh2		Traffic	
Method	MSE	MAE	MSE	MAE	MSE	MAE
DynaFK	.4087±.0002	.3986±.0001	.3351±.0006	.3640±.0003	.7260±.0006	.4011±.0003
FITS-residual	.4303±.0005	.4155±.0002	.3395±.0003	.3696±.0002	.7850±.0002	.4423±.0001
Linear	.4068±.0007	.4000±.0002	.3206±.0003	.3551±.0002	.7157±.0007	.4088±.0002
DLinear	.4065±.0008	.3999±.0003	.3207±.0008	.3550±.0006	.7155±.0002	.4090±.0008

C METRIC SENSITIVITY WITHOUT TARGET-TRAINING STATISTICS

Table 11 keeps checkpoints and predictions fixed and changes only evaluator-side aggregation. ETTh2 and ECL preserve the DynaFK versus Linear ordering under target-reference and original-unit errors. On Traffic, MSE favors DynaFK under both aggregations, but MAE changes order: target-reference MAE is .460073 for Linear versus .461911 for DynaFK, whereas original-unit MAE is .0207616 versus .0207368. The difference is small and no new uncertainty test is attached to this ranking reversal, so we infer only that channel weighting can affect a win/loss label. Original-unit errors read no target-training rows but are dimensionful and cannot be compared across datasets. Existing per-window artifacts contain aggregate errors rather than predictions or per-channel errors; an observed-input-normalized per-series metric therefore requires reevaluation and is not reconstructed here.

Table 11: Metric sensitivity for DynaFK versus seasonal Linear at $H = 96$. Values are three-seed means from identical checkpoints. Target-reference scaling is evaluator-only; original-unit errors read no target-training statistics. Original-unit magnitudes are not comparable across datasets.

		Target-reference		Original unit	
Target	Method	MSE	MAE	MSE	MAE
ETTh2	DynaFK	0.312372	0.350510	19.5597	2.68674
	Linear	0.294748	0.340906	18.4716	2.61053
ECL	DynaFK	0.242602	0.323390	7.40286e+06	302.688
	Linear	0.226047	0.306298	6.31066e+06	271.18
Traffic	DynaFK	0.784894	0.461911	0.00157971	0.0207368
	Linear	0.804977	0.460073	0.00162027	0.0207616

D HORIZON, FREQUENCY, AND CAPACITY SENSITIVITY

For completeness, replacing faithful FITS mean-centering and mean anchor with the common instance-normalized seasonal-residual adapter gives ETTh2/ECL/Traffic MSE .3183/.2450/.8784 and ETTm1/ETTm2/Weather MSE .4723/.2728/.3427. We report this matched-adapter diagnostic separately because it is not the published FITS input/output convention and should not replace the faithful baseline.

Table 12: ETTh1→ETTh2 across horizons. MSE/MAE is mean±sample SD over three seeds. Params are PyTorch elements at that horizon.

H	Method (params)	MSE	MAE
96	DynaFK (685)	.3124±.0001	.3505±.0000
	Linear (9,312)	.2947±.0004	.3409±.0004
	Koopa (43,592)	.2989±.0021	.3446±.0011
192	DynaFK (685)	.4684±.0017	.4274±.0007
	Linear (18,624)	.3810±.0021	.3926±.0013
	Koopa (49,832)	.3775±.0007	.3914±.0009
336	DynaFK (685)	.5269±.0003	.4661±.0001
	Linear (32,592)	.4172±.0005	.4256±.0001
	Koopa (59,192)	.4149±.0026	.4236±.0020
720	DynaFK (685)	.5238±.0002	.4750±.0000
	Linear (69,840)	.4189±.0006	.4373±.0001
	Koopa (84,152)	.4149±.0025	.4341±.0021

Table 13: Choosing K from ETTh1 source validation. Mean±sample SD over three seeds.

K	Tensor elems.	Source-val MSE	ETTh2 MSE
1	181	.8164±.0002	.3811±.0001
2	349	.8162±.0002	.3809±.0001
4	685	.7383±.0003	.3124±.0001
8	1,357	.7380±.0001	.3123±.0001

Table 14: ETTh1→ETTm cross-frequency sensitivity. Mean±sample SD over three seeds.

Target	History	MSE	MAE
ETTm1	96 hours	.3880±.0085	.3967±.0044
	96 steps	.3923±.0019	.3844±.0005
ETTm2	96 hours	.2578±.0058	.3175±.0035
	96 steps	.2404±.0005	.2909±.0004

Table 15: Direct ETTh1-source baselines with a 96-hour target history and $H = 96$ target samples. Entries are target-reference MSE/MAE, mean±sample SD over three seeds. Fixed-width baselines resample the input history to 96 source steps; Koopa is incompatible with Weather's channel count.

	ETTm1		ETTm2		Weather	
Method	MSE	MAE	MSE	MAE	MSE	MAE
Seasonal naive	.4227	.3865	.2631	.3005	.3167	.2880
DynaFK	.3880±.0085	.3967±.0044	.2578±.0058	.3175±.0035	.3450±.0156	.3324±.0060
FITS	1.0010±.0010	.6259±.0005	.2581±.0006	.3415±.0004	.2496±.0007	.3062±.0006
Linear	.4955±.0010	.4522±.0007	.2872±.0006	.3379±.0004	.3560±.0011	.3231±.0004
DLinear	.4960±.0019	.4524±.0011	.2873±.0015	.3376±.0015	.3557±.0026	.3225±.0009
SparseTSF	.4219±.0001	.3891±.0003	.2629±.0002	.3048±.0003	.3206±.0003	.2974±.0006
Official Koopa	1.0168±.0142	.6298±.0035	.2533±.0049	.3375±.0030	N/A	N/A

E Mechanism Audits and Negative Results

Exact Fixed-Horizon Collapse

Substituting the complex encoder into the spectral evolution gives

$$\hat{r}_{p,q} = \Re \sum_k \left(\sum_i s_{p,i} E_{p,i,k} \right) e^{q\lambda_{p,k}} d_{p,k} \tag{7}$$

$$= \sum_i s_{p,i} \underbrace{\Re \sum_k E_{p,i,k} e^{q\lambda_{p,k}} d_{p,k}}_{A_{p,q,i}} . \tag{8}$$

The exchange of finite sums is exact. Thus, conditional on the same input normalization and smoothing, every DynaFK predictor at fixed P and H is a phase-linear predictor. The reverse containment is not asserted because the exponential factorization constrains A across output cycles.

For a numerical implementation check, we load the seed-2023 ETTh1 checkpoint, construct A algebraically, copy its smoothing kernel into the direct phase-linear model, and evaluate 32 random windows with seven channels and all 24 start phases represented. The maximum absolute prediction difference is 4.7684×10^{-7} and the root-mean-square difference is 7.0272×10^{-8}. The phase-linear coefficients are not fitted in this check.

Exchange Rate Mechanism Check

Setup. We train and select all models on hourly ETTh1 and evaluate them on daily Exchange Rate data. The source and target periods are $P_{\mathrm{s}} = 24$ and $P_{\mathrm{t}} = 1$, respectively; the source history is 96 samples, while the target history and forecast horizon are each 96 target steps. Results use seeds 2023, 2024, and 2025, with the source-only training and selection protocol unchanged from Section 4.

Results. Table 16 reports mean±sample SD over the three source training seeds. With $P_{\mathrm{t}} = 1$, seasonal naive reduces to last-value persistence.

Table 16: ETTh1→Exchange Rate at $H = 96$ target steps. Errors are mean±sample SD over three seeds.

Method	Real scalars	Ref. MSE	Ref. MAE	Original MSE	Original MAE
Persistence	0	.086308±.000000	.203845±.000000	.00073975±.00000000	.01692682±.00000000
DynaFK	1,165	.095223±.000699	.218356±.000868	.00081549±.00000618	.01811741±.00007118
Phase-linear	397	.096829±.000260	.220023±.000372	.00080427±.00000196	.01808427±.00002964
Global-spectrum	981	.096473±.001020	.219816±.000924	.00082608±.00000881	.01823664±.00007637
Linear	9,312	.102718±.000171	.225872±.000216	.00087260±.00000098	.01867634±.00001715
DLinear	18,624	.102074±.000472	.225134±.000461	.00086739±.00000291	.01862206±.00002907
FITS	2,400	4.073903±2.386284	.768566±.192966	.00082450±.00001217	.01927148±.00053189

FITS's large reference-scaled MSE is driven by one Exchange Rate channel with a very small reference standard deviation; recomputing the metric from the stored predictions gives the same value.

Interpretation. The phase-linear model is close to DynaFK: its target-reference MSE and MAE are 0.096829 and 0.220023, compared with 0.095223 and 0.218356 for DynaFK, relative differences of 1.69% and 0.76%. For the paired DynaFK minus phase-linear difference with block length 192, the MSE is -0.001606 [-0.013321, 0.009386] and the MAE is -0.001667 [-0.015596, 0.011049]; both intervals include zero. Algebraic collapse over the same 1,688 test windows per seed has maximum absolute prediction error 1.19×10^{-7} and RMSE approximately 5.6×10^{-9}. The fixed-H relationship therefore survives the change in target period. Persistence remains more accurate, with MSE 0.086308 and MAE 0.203845 versus DynaFK's 0.095223 and 0.218356, separating this mechanism check from transfer accuracy on Exchange Rate.

Table 17: 95% intervals for DynaFK minus Linear. B is the block length.

Target	Metric	$B = 96$	$B = 192$	$B = 384$
ETTh2	MSE	[.0098,.0268]	[.0090,.0279]	[.0086,.0275]
	MAE	[.0056,.0141]	[.0052,.0146]	[.0049,.0146]
ECL	MSE	[.0147,.0188]	[.0146,.0189]	[.0146,.0187]
	MAE	[.0156,.0186]	[.0154,.0188]	[.0153,.0189]
Traffic	MSE	[−.0261,−.0140]	[−.0266,−.0133]	[−.0265,−.0135]
	MAE	[−.0004,.0041]	[−.0003,.0039]	[−.0005,.0040]

Matched Alignment Audit

The main model excludes alignment because a matched one-seed audit gives ETTh1→ETTh2 MSE 0.312505990 with adjacent-phase Sinkhorn alignment, 0.312506169 without alignment, 0.312505990 with L2 phase smoothing, and 0.312480360 with one global spectrum. The largest difference is below 3×10^{-5}. A separate mean-anchor sweep over alignment weights 0.05, 0.1, 0.5, 1, 5, 10, and 50 gives MSE 0.315112, 0.315098, 0.315143, 0.315168, 0.315296, 0.315408, and 0.315567, respectively. It is flat at small weights and slightly worsens at large weights; it is not U-shaped.

We also implemented an input-conditioned local spectral DMD control so that alignment acts on spectra estimated from input windows rather than on a learned phase table. Its no-alignment MSE is 0.386611; Sinkhorn weights 0.01 and 0.1 give 0.387108 and 0.386906, and L2 smoothing gives 0.387380. The control is worse than learned DynaFK and again shows no alignment gain. These results rule out a positive alignment statement for the evaluated protocol.

Phase and Channel Controls

The global-spectrum and relaxed phase-linear controls in Table 8 are practically indistinguishable from DynaFK. Consequently, phase-conditioned exponential evolution is a compact parameterization but is not empirically identified as the accuracy mechanism. The channel-mixing Linear adds only 49 real weights to independent Linear and obtains 0.2958 rather than 0.2947 MSE on ETTh2. Since its mixer is dimension-specific, it cannot be applied to ECL or Traffic without changing the checkpoint.

F Paired Moving-Block Bootstrap

For each seed and target, DynaFK and Linear errors are paired by exact test window start index. We average the paired difference over the three seeds, compute circular contiguous block sums, and resample 20,000 sets of blocks. Positive difference favors Linear. This procedure handles dependence from overlapping windows but is conditional on the three learned checkpoints.

The sign is stable across block choices for ETTh2/ECL MSE and MAE and for Traffic MSE. Traffic MAE crosses zero in all cases. We therefore report a Traffic MSE advantage but no reliable MAE advantage for DynaFK.

G Efficiency Details

Inference benchmarks use the selected checkpoint, a fixed source test batch, 20 warm-up iterations, 100 timed iterations, and CUDA synchronization around the timed region. Peak memory is reset after warm-up. DynaFK uses 2.353 microseconds per series window and 24.48 MiB peak allocation; phase-linear 1.653/20.53; FITS 0.984/20.90; Linear 0.516/17.64; DLinear 1.030/18.37; SparseTSF 0.765/18.56; and Koopa 10.629/21.20. The batch contains 128 windows and seven channels.

Mean epoch times in milliseconds are 472.4 (DynaFK), 498.6 (phase-linear), 229.8 (FITS), 305.7 (Linear), 311.8 (DLinear), 481.7 (SparseTSF), and 647.2 (Koopa). These times include training batches and one source-validation pass. They are not intended as cross-hardware claims.

H ARTIFACT-TO-CLAIM MAP

The tables combine historical audited local runs with the explicitly marked post-migration reproduction in Table 7. DynaFK, phase-linear, Linear, faithful FITS, and GPU Koopa reproduce their compatible ETTh1-source $H = 96$ target-reference cells within numerical tolerance using the current runner and matching data hashes. The CPU Koopa rerun differed by up to 1.79×10^{-3} and is retained only as a device-sensitivity diagnostic. Other source directions and horizons remain tied to their historical manifests and are not claimed as rerun from the current snapshot.

Table 18: Artifact-to-claim audit.

Claim	Required evidence and audit field
No target fitting	Run configuration and training history; leakage contract and target-gradients flag.
Source-only selection	Source-validation history and checkpoint; best source-validation MSE and checkpoint hash.
Reported mean/SD	Three manifests per method; seeds 2023–2025 and target metrics.
Paired intervals	Per-window NPZ arrays; exact start-index equality and block-bootstrap JSON.
Fair model size	Checkpoint tensor dtypes and shapes; tensor-element and real scalar accounting.
Efficiency	Environment and benchmark manifest; GPU, batch, warm-up/timed iterations, and peak memory.
Data identity	Dataset files and their SHA-256 digests.