THE HIDDEN OPTIMIZER IN SPECTRAL TIME-SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-branch architectures in spectral time-series forecasting are commonly presumed to expand representational capacity. This paper demonstrates that such structural additions often merely counteract a hidden optimization pathology rather than enrich expressive power. To isolate this phenomenon, we utilize a modular Phase-Structured Predictor (PSP) that decouples phase, frequency, and input-conditioned pathways, establishing a controlled diagnostic testbed. Through this architecture, we reveal that standard adaptive optimizers (e.g., AdamW) leave an acute c_b/n update-scale attenuation in forecast space due to the inverse Fourier transform, severely throttling terminal kernel adaptation and corrupting upstream representations. We prove that under *whole-model closure*, multi-branch dynamics can be rigorously collapsed into a single effective kernel. Grounded in this equivalence, we introduce **SpecScale** (*Spectral Scale Optimizer*): theoretically, its *Exact formulation* strictly reproduces multi-branch training dynamics step-for-step with zero trajectory divergence; practically, it yields an efficient, branch-free rule that applies an analytic $(1 + n/c_b)$ frequency multiplier directly to base-kernel AdamW updates. On a four-cell diagnostic panel under closure, SpecScale recovers 96.5% of the multi-branch empirical gain. On the full 32-cell weighted benchmark, SpecScale reduces test MSE by 3.158% (improving 27 of 32 settings under equal-budget selection) while matching a 15-setting hyperparameter grid search with $5\times$ fewer trials and zero inference overhead. Furthermore, readout corrections consistently transfer to three prominent external backbones, confirming that readout scale attenuation is an architecture-agnostic optimization bottleneck.

1 INTRODUCTION

Spectral time-series forecasters compactly capture multi-periodicity and global temporal dependencies by projecting temporal sequences into frequency representations (Zhou et al., 2022; Xu et al., 2024; Yi et al., 2023). In these architectures, a *spectral readout*—typically an inverse real Fast Fourier Transform (irfft)—maps learned frequency components back to future time-domain values. This reconstruction map serves an indispensable role: it forms the final operational bridge that converts abstract frequency-domain interactions into physical future trajectories, governing both the representable forecast family and how parameter updates alter predictions. However, this interface conceals a fundamental optimization pathology: while adaptive optimizers such as AdamW normalize gradient updates in *parameter space*, the discrete inverse transform inherently scales each frequency bin by c_b/n (where n is sequence length and c_b is frequency multiplicity). Consequently, parameter updates are severely attenuated when transmitted into *forecast space*. This scale mismatch progressively throttles physical prediction movement as sequence length grows, creating a pervasive optimization bottleneck at the output interface.

To establish fine-grained experimental control, we design the **Phase-Structured Predictor (PSP)**: a modular host architecture that cleanly decouples phase, frequency, and input-conditioned pathways, providing the structural transparency needed to isolate individual frequency dynamics. Within this controlled testbed, we expose the impact of update-scale attenuation through a capacity-preserving probe: **Quadrature Reparameterization (QR)**, which augments the base kernel with an unscaled orthogonal coordinate outside the inverse transform. Because standard parameter updates are suppressed by the transform's inverse scaling, single-kernel readouts adapt sluggishly and prematurely stagnate in high-loss plateaus. In contrast, QR restores forecast updates to their natural scale, enabling

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.

the readout to rapidly eliminate prediction errors. Empirically, this unthrottled adaptation yields decisive gains: QR lowers test MSE in 28 of 32 benchmark configurations with a 4.25% mean reduction across eight datasets, swiftly escaping the plateaus that trap the baseline (Figure 1a). Crucially, this terminal sluggishness is not an isolated defect: persistent prediction errors propagate corrupted, lagging gradients backward, severely derailing upstream representation learning across the network.

Importantly, these empirical gains require zero added expressive capacity: because QR folds algebraically into the original linear kernel, its representable forecast family is mathematically identical to the baseline. This equivalence recontextualizes recent spectral architectures that incorporate specialized output mechanisms—such as amplitude–phase normalizations (Hu et al., 2026), learned filters (Yi et al., 2024), and phase-aware tokens (Niu et al., 2026). While widely viewed as representational improvements, their gains often stem from unscaled coordinate paths that inadvertently bypass inverse-transform attenuation—relying on architectural complexity to patch an underlying optimization deficit. This insight raises a fundamental question: if auxiliary branches contribute no new capacity, *can we eliminate this structural overhead entirely and resolve the scale deficit directly within the optimizer?*

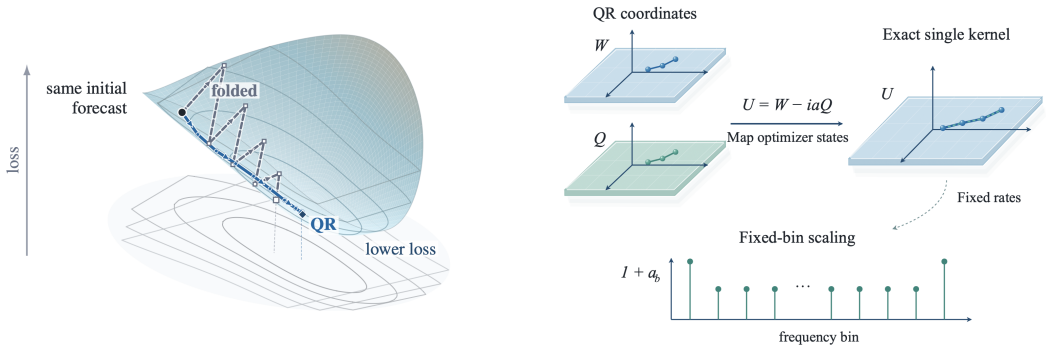
In this paper, we resolve this prediction-space scale deficit directly at the optimizer level, bypassing the need for redundant network branches (Ding et al., 2023; Everett et al., 2024). Grounded in a structural condition we formalize as *whole-model closure*—where the network interacts with its readout coordinates exclusively through their effective kernel—we introduce **SpecScale** (*Spectral Scale Optimizer*). Under closure, we prove that multi-branch coordinate dynamics can be analytically collapsed into a single effective kernel. SpecScale realizes this equivalence through two complementary tiers: (1) theoretically, an *exact single-kernel optimizer* that strictly reproduces multi-branch training step-for-step with zero trajectory divergence; and (2) practically, an efficient branch-free rule, *Fixed-Bin Scaling*, which applies the analytic geometric multiplier $(1 + n/c_b)$ directly to base-kernel AdamW updates (Figure 1b). Our contributions are:

- **Diagnosing prediction-space scale attenuation in spectral readouts.** We uncover that standard adaptive optimizers (e.g., AdamW) coupled with discrete inverse transforms induce severe c_b/n update shrinkage in forecast space. Using the controlled PSP testbed and our capacity-preserving QR probe, we prove that empirical gains commonly attributed to multi-branch architectures arise from counteracting this update shrinkage rather than expanding expressive capacity. Synchronized traces confirm that terminal update sluggishness directly corrupts upstream feature representations.
- **The SpecScale optimization framework.** Grounded in the whole-model closure criterion, we prove the mathematical redundancy of auxiliary branches during training and introduce SpecScale: (a) an *Exact formulation* accounting for coupled moments, finite-ϵ dynamics, and virtual gradient clipping to achieve strict step-by-step trajectory identity; and (b) a lightweight *Fixed-Bin rule* that distills the dominant dynamics into an analytic multiplier $(1 + n/c_b)$ applied directly after AdamW normalization, eliminating auxiliary branches, optimizer states, and inference overhead.
- **Comprehensive empirical validation and cross-backbone transfer.** On a four-cell plain-attention panel under closure, Exact matches multi-branch training to machine precision (10^{-7} in float32, 10^{-15} in float64), and Fixed-bin recovers 96.5% of QR's gain. On the full 32-cell weighted benchmark, Fixed-bin reduces mean MSE by 3.158% (improving 27 of 32 cells under equal-budget selection), matching a 15-setting hyperparameter grid search. Furthermore, zero-initialized readout corrections consistently transfer to prominent external spectral backbones, confirming that readout scale attenuation is an architecture-agnostic bottleneck.

2 Related Work

Our study of spectral readout dynamics connects three lines of work: spectral time-series forecasting, equivalent parameterization theory, and structural optimizer design.

Spectral representations for forecasting. Frequency-domain time-series models have evolved across several paradigms to capture temporal periodicities. Early approaches combined time-frequency decomposition with mode selection (Zhou et al., 2022; Wu et al., 2021); subsequent architectures shifted toward learning direct frequency-domain mappings or complex spectral filters (Yi et al., 2023; Xu et al., 2024; Yi et al., 2024); and recent designs introduce phase-aware tokens or amplitude–phase decoupling (Niu et al., 2026; Hu et al., 2026; Yang et al., 2025). Despite these varied



(a) Same forecast set, different optimization paths

(b) From coordinates to a branch-free rule

Figure 1: From spectral readout geometry to a branch-free optimizer rule. **(a)** Algebraically equivalent readouts follow divergent optimization paths: inverse-transform attenuation traps single-kernel baselines in high-loss plateaus, whereas unscaled coordinates (QR) restore natural update scale. **(b)** Grounded in whole-model closure, SpecScale collapses multi-branch dynamics into an exact single-kernel optimizer, and Fixed-bin scaling ($1 + a_b, a_b = n/c_b$) recovers 96.5% of the empirical gain with zero parameter or inference overhead.

modeling strategies, existing methods focus primarily on upstream spectral representations, leaving the optimization dynamics of the terminal reconstruction interface unexamined.

Equivalent parameterizations and optimization dynamics. Overparameterization and alternative coordinate systems can fundamentally alter training trajectories within a fixed function class (Arora et al., 2018; Yun et al., 2021). Methods such as weight normalization (Salimans & Kingma, 2016), ReZero (Bachlechner et al., 2021), and ExpandNets (Guo et al., 2020) leverage this property to accelerate convergence or stabilize training by restructuring parameter coordinates. Spectral coordinates have likewise been applied to convolutional filters (Rippel et al., 2015) and implicit networks (Shi et al., 2024) to modulate frequency learning biases. However, these analyses predominantly target real-valued matrices or spatial convolutions, rather than the discrete complex geometry and sequence-length scaling inherent in spectral inverse transforms.

Structural reparameterization and optimizer design. RepOptimizers (Ding et al., 2023) demonstrate that linear structural priors can be compiled into single-operator gradient multipliers for SGD, eliminating multi-branch inference overhead. Parallel scaling theories analyze how parameterization and Adam's ϵ affect asymptotic width transfer (Everett et al., 2024), while architectures like ModernTCN (Luo & Wang, 2024) merge multi-scale temporal convolutions at deployment. However, these techniques are formulated for real-valued linear branches under SGD or asymptotic width limits, leaving the finite-step AdamW dynamics, complex coordinate rotations, and gradient clipping of spectral readouts unaddressed.

3 Method

This section investigates how spectral readout parameterization shapes optimization, and how its benefits can be realized without maintaining auxiliary branches. We first formulate the discrete spectral readout and its function-space update scale (Section 3.1), introduce the PSP host and the QR probe (Section 3.2), derive the exact single-kernel optimizer under whole-model closure (Section 3.3), and distill the practical branch-free training rule (Section 3.4).

3.1 Spectral Readout Geometry and Update Scales

Let $X \in \mathbb{C}^{n_b}$ be a pre-readout spectrum of transform length n, with $n_b = \lfloor n/2 \rfloor + 1$ one-sided real-FFT (rFFT) frequency bins. For each forecast step, a complex parameter kernel $W \in \mathbb{C}^{n_b}$ reconstructs the scalar time-domain prediction z via the inverse real FFT (irfft):

$$z = \mathrm{irfft}(X \odot W, n)[0] = \frac{1}{n} \sum_{b=0}^{n_b-1} c_b \,\mathrm{Re}(X_b W_b), \tag{1}$$

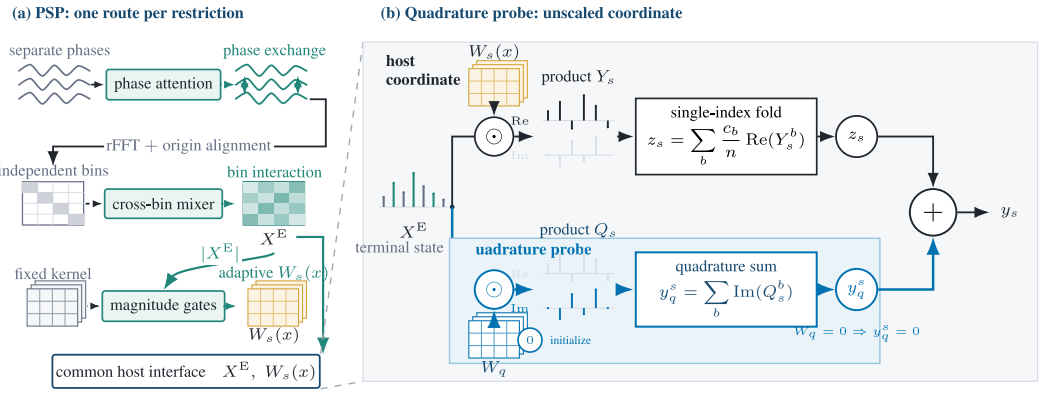


Figure 2: The PSP architecture and diagnostic probe. **(a) PSP:** a fully controllable phase backbone with three independently switchable modules. Details in Appendix A **(b) multi-branch QR:** General route and the route augmented with QR, mathematically eliminated with single-kernel optimization.

where $\odot$ denotes elementwise multiplication, and c_b accounts for conjugate-frequency multiplicity ($c_b = 1$ at DC and Nyquist; $c_b = 2$ elsewhere). This discrete inversion inherently scales each bin by c_b/n. For convenience, we define the reciprocal scale factor $a_b = n/c_b$, which evaluates to $n/2$ at interior frequencies and n at the boundaries.

Writing $w_b = (\mathrm{Re}\, W_b, \mathrm{Im}\, W_b)^\top$, the real-coordinate Jacobian is $J_b = \frac{\partial z}{\partial w_b} = \frac{c_b}{n}\,[\mathrm{Re}\, X_b \quad -\mathrm{Im}\, X_b]$. Standard adaptive optimizers such as AdamW (Loshchilov & Hutter, 2019) normalize parameter gradients coordinate-wise, enforcing a roughly uniform step size in *parameter space* ($\|\Delta w_b\| \approx \eta$, where η is the learning rate). However, transmitting these updates through the inverse transform attenuates the actual displacement in *forecast space*:

$$\Delta z = \sum_{b=0}^{n_b-1} J_b \Delta w_b \propto \sum_{b=0}^{n_b-1} \frac{1}{a_b}\eta. \tag{2}$$

Because $a_b \propto n$, physical prediction movement is scaled down by $O(1/n)$, progressively throttling function-space adaptation as sequence length increases.

3.2 Quadrature Coordinates and Function Equivalence

The controlled host architecture (PSP). To isolate readout optimization from upstream representation learning, we instantiate the modular **Phase-Structured Predictor (PSP)** as our host testbed (Figure 2a). Given a period p dividing lookback L and horizon H, PSP partitions inputs into p phase subseries of length $n = L/p$ and isolates feature interactions via three switchable routes (Appendix A): (1) *phase attention* mixing subseries across periods; (2) *cross-bin mixing* capturing inter-frequency dependencies; and (3) *magnitude gating* conditioning the kernel $W_s = W_{\mathrm{base},s} + G(X) \odot \Delta W_s$ on inputs. The single-kernel baseline trained with AdamW is designated as the **Host**, evaluated under two variants: a *Plain-attention Host* where W_{base} appears solely in the terminal readout (satisfying closure), and a *Weighted Host* whose attention reads base-kernel magnitudes.

Quadrature Reparameterization (QR) as a diagnostic probe. To test if $1/a_b$ scale attenuation causes premature stagnation, we introduce QR as an unscaled diagnostic probe (Figure 2b). Using the Fourier phase rotation identity $\mathrm{Im}(XQ) = \mathrm{Re}(X(-iQ))$, QR reparameterizes the effective kernel $U_b \in \mathbb{C}$ into an in-phase coordinate W_b and an unscaled quadrature coordinate Q_b:

$$U_b = W_b - i a_b Q_b, \qquad \text{where } a_b = \frac{n}{c_b}. \tag{3}$$

Setting $W'_b = U_b$ algebraically absorbs the quadrature coordinate back into the single base kernel:

$$\mathrm{irfft}(X \odot W', n)[0] = \underbrace{\frac{1}{n}\sum_{b=0}^{n_b-1} c_b\, \mathrm{Re}(X_b W_b)}_{\text{base forecast } z} + \underbrace{\sum_{b=0}^{n_b-1} \mathrm{Im}(X_b Q_b)}_{\text{unscaled quadrature forecast } y_q} = y. \tag{4}$$

The factor $a_b = n/c_b$ balances the transform's attenuation for Q_b. Setting $Q_b = 0$ initially preserves baseline predictions, while the algebraic fold into U_b guarantees that QR adds **zero representational capacity** at fixed features.

Under AdamW, however, the two parameterizations move differently. While parameter-space steps are standardized ($\|\Delta W_b\| \approx \|\Delta Q_b\| \approx \eta$), W_b is attenuated by $1/a_b$ while Q_b updates predictions directly. Combining both coordinates, the forecast-space step becomes:

$$\Delta y_b \propto \left(\frac{1}{a_b} + 1\right)\eta = \frac{1 + a_b}{a_b}\eta, \tag{5}$$

which is $(1 + a_b)$ times the Host update in Equation (2). While initial upstream gradients match at $Q = 0$, this enlarged terminal step alters prediction errors, causing upstream representations to diverge after the first update.

Controlled checks confirm that this $(1 + a_b)$ update scale—rather than added capacity—drives the empirical gains, as QR reaches lower minima on frozen representations (Appendix C.1) and improves benchmark forecasting (Section 4.2). Because auxiliary branches add no expressiveness, their optimization benefits should be achievable directly within a single kernel.

3.3 Exact Trajectory Equivalence via Single-Kernel Optimization

While the QR probe counteracts the $1/a_b$ attenuation, it achieves the desired prediction-space scale by introducing an auxiliary branch. Because Equation (4) proves that QR adds no representational capacity, a natural question arises: *can we train the single effective kernel U directly along the exact multi-branch trajectory, without ever instantiating the auxiliary coordinate Q?*

Whole-model closure. To collapse the two-branch model into a single stored kernel U, the forward prediction and loss must depend on (W, Q) exclusively through U. Let $F_{\theta,D}$ denote all forward operations entering the loss. Whole-model closure requires:

$$W - iAQ = W' - iAQ' \implies F_{\theta,D}(x; W, Q) = F_{\theta,D}(x; W', Q') \quad \forall x, \tag{6}$$

where $A = \mathrm{diag}(a_0, \ldots, a_{n_b-1})$ applies $a_b = n/c_b$ across frequency bins. Thus, features, gates, and losses may not read W or Q separately (Ding et al., 2023; Yun et al., 2021). When closure holds, the loss gradient with respect to U, $g_U = \partial\mathcal{L}/\partial U$, linearly determines the coordinate gradients:

$$g_W = g_U, \qquad g_Q = R^\top A g_U, \qquad \|g_W\|^2 + \|g_Q\|^2 = \sum_{b=0}^{n_b-1}(1 + a_b^2)\|g_{U,b}\|^2, \tag{7}$$

where R represents multiplication by $-i$ in real coordinates (mapping $(\mathrm{Re}\, z, \mathrm{Im}\, z)$ to $(\mathrm{Im}\, z, -\mathrm{Re}\, z)$), satisfying $R^\top R = I$ and $RA = AR$.

Resolving adaptive optimization dynamics. Under basic gradient descent, folding updates yields $U_b^+ = U_b - \eta(1 + a_b^2)g_{U,b}$. However, reproducing AdamW requires overcoming three obstacles:

- *Moment state mapping:* With identical hyperparameters (β_1, β_2), the moving averages of the virtual coordinate Q are exact algebraic transformations of W's states:

$$m_Q = R^\top A m_W, \qquad v_Q = S^\top A^2 v_W, \qquad S = |R|. \tag{8}$$

  Zero-initialized states satisfy Equation (8) naturally. Because elementwise squaring preserves the a_b^2 scaling and permutation S, both branch trajectories are tracked using a single moment pair (m_W, v_W) maintained for U.
- *Finite-ϵ state-dependent scaling:* Folding parameter steps into $\Delta U_b = \Delta W_b - i a_b \Delta Q_b$ yields:

$$\Delta U_b = -\hat{m}_{W,b}\left[\frac{\eta_W}{s_b + \epsilon_{W,b}} + \frac{a_b\eta_Q}{s_b + \epsilon_{Q,b}/a_b}\right], \tag{9}$$

  where $s_b = \sqrt{\hat{v}_{W,b}}$. Under finite ϵ, enforcing equal branch epsilons $\epsilon_W = \epsilon_Q$ induces an effective rate that interpolates between $\eta_W + a_b\eta_Q$ (when $s_b \gg \epsilon$) and $\eta_W + a_b^2\eta_Q$ (as $s_b \to 0$).

- *Augmented global gradient clipping:* Global clipping scales updates by $\rho = \min(1, C/N)$, where N is the total parameter gradient norm. To match this scaling without storing Q, the optimizer must include the virtual coordinate norm from Equation (7):

$$N^2 = \|g_\theta\|^2 + \|g_D\|^2 + \sum_{b=0}^{n_b-1} (1 + a_b^2)\|g_{U,b}\|^2, \tag{10}$$

  where g_θ and g_D denote upstream and content-conditioned residual gradients, ensuring that the clipping coefficient ρ matches the multi-branch run across the entire network.

Proposition 1 (Exact Trajectory Equivalence)**.** *Assume the network satisfies whole-model closure (Equation* (6)*), shares initial upstream parameters θ, gate residual D, and effective kernel $U_0 = W_0 - iAQ_0$, uses mapped optimizer states with equal step counters, applies identical mini-batch sequences, and sets decoupled weight decay $d_W = d_Q$. Then an optimizer storing only U and applying Equations* (9) *and* (10) *produces identical predictions, effective kernels, and upstream representations at every training step in exact arithmetic.*

Proposition 1 establishes that auxiliary branches are fundamentally replaceable during training: under closure, the single-kernel optimizer (**SpecScale-Exact**) follows the multi-branch trajectory with zero divergence, eliminating auxiliary tensors and multi-branch compute (proof in Appendix B.1; float64 checks confirm identity to 10^{-15} in Appendix B.2).

3.4 SpecScale: Practical Branch-Free Rule and Deployment

While the SpecScale-Exact optimizer proves that multi-branch dynamics can be collapsed into a single kernel, tracking virtual clipping norms and finite-ϵ adjustments adds implementation overhead. In standard training regimes where gradients exceed machine precision ($s_b \gg \epsilon$), Equation (9) shows that the core driver of the multi-branch advantage is simply the geometric multiplier $1 + a_b$ on the base-kernel update.

We distill this insight into **SpecScale**, an explicit, branch-free rule that applies this frequency-dependent scale factor directly after standard AdamW normalization:

$$\Delta U_b = -(1 + a_b)\eta \frac{\hat{m}_{U,b}}{\sqrt{\hat{v}_{U,b}} + \epsilon} = -\left(1 + \frac{n}{c_b}\right)\eta \frac{\hat{m}_{U,b}}{\sqrt{\hat{v}_{U,b}} + \epsilon}. \tag{11}$$

Specifically, updates for interior frequencies ($0 < b < n/2$) receive a multiplier of $1 + n/2$, while boundary frequencies ($b \in \{0, n/2\}$) receive $1 + n$. All non-readout parameters retain their default AdamW updates, and ordinary global gradient clipping is applied. In PyTorch, this is implemented simply as a parameter-group learning rate multiplier on the base kernel, requiring no auxiliary tensors, custom backward graphs, or optimizer state tracking.

Deployment scopes and efficiency. The modular host architecture allows evaluating two practical deployment regimes:

- *Strict closure (Plain-attention Host):* When the base kernel appears solely in the terminal readout, whole-model closure holds strictly. SpecScale trains the single-kernel model directly, and the resulting checkpoint is deployed with zero modification.
- *Cached deployment (Weighted Host):* When attention mechanisms consume base-kernel magnitudes during training, multi-branch models can still be collapsed for inference: precomputing and caching the magnitude vector $m_{k,b} = S_H^{-1} \sum_s |W^{k,b}_{\text{base},s}|$ allows folding the terminal weights via $U_b = W_b - ia_bQ_b$, matching two-branch forecasts within numerical precision ($< 6.68 \times 10^{-6}$, Appendix A.2).

Under both regimes, the final deployed model operates on a single linear complex kernel, incurring **zero additional parameters, zero extra FLOPs, and zero inference latency overhead**.

4 Experiments

We evaluate our theoretical insights and branch-free training rules across diverse forecasting benchmarks and controlled diagnostic panels, addressing three core questions: (1) Does rectifying the

readout scale bottleneck enhance forecasting accuracy across diverse real-world benchmarks compared to modern architectures (Section 4.2)? (2) Does the single-kernel optimizer strictly reproduce multi-branch dynamics, and does branch-free Fixed-bin scaling retain these gains without structural overhead (Section 4.3)? (3) Is the empirical advantage genuinely driven by update scale rather than expressiveness, and how does it propagate into upstream representation learning (Section 4.4)?

4.1 Experimental Setup

Datasets and evaluation protocol. We evaluate eight standard multivariate benchmarks (Wu et al., 2023; Liu et al., 2024) (ETTh1, ETTh2, ETTm1, ETTm2, ECL, Traffic, Weather, Solar) across four horizons $H \in \{96, 192, 336, 720\}$ with fixed lookback $L = 720$ (32 evaluation cells), adopting standard chronological splits and RevIN (Kim et al., 2022). Models are evaluated by test MSE and MAE. To isolate readout interventions from initialization drift, we measure the paired relative change $\Delta = 100 \times (\mathrm{MSE}_{\text{model}} - \mathrm{MSE}_{\text{Host}})/\,\mathrm{MSE}_{\text{Host}}$ across five seeds (42–46) sharing identical mini-batch sequences. Training uses AdamW (Loshchilov & Hutter, 2019) with unit global clipping ($C = 1.0$), zero weight decay, at most 30 epochs, and three-epoch early stopping (details in Appendix E.1).

Compared models and baselines. We examine four PSP training variants: the single-kernel baseline **Host** (standard AdamW); the dual-branch **QR probe**; **SpecScale-Exact** reproducing QR's trajectory under closure (Proposition 1); and our branch-free **SpecScale** rule $(1 + n/c_b)$. We benchmark against five competitive spectral and compact baselines: FITS (Xu et al., 2024), FilterNet (PaiFilter/TexFilter) (Yi et al., 2024), MixLinear (Ma et al., 2026), and SparseTSF (Lin et al., 2024).

4.2 Main Forecasting Benchmark

Table 1 evaluates forecasting performance across all 32 settings on weighted PSP, comparing the single-kernel Host against our dual-branch QR probe, our branch-free **SpecScale** optimizer, and five competitive external architectures.

Table 1: Main forecasting benchmark on weighted PSP ($L = 720$, averaged over four horizons and five seeds). Bold marks the best readout score; underline marks the best external baseline. Δ_{Spec} is the relative MSE change of branch-free **SpecScale** over Host (full results in Appendix E.3).

	PSP Readout Variants (Ours)				External Spectral and Compact Models				
Dataset	Host	QR (Probe)	**SpecScale**	Δ_{Spec} (%)	FITS	PaiFilter	MixLinear	TexFilter	SparseTSF
ETTh1	0.484	**0.417**	**0.417**	-11.44	0.513	0.433	<u>0.401</u>	0.441	0.405
ETTh2	0.383	0.354	**0.348**	-7.69	0.345	0.386	0.343	0.398	<u>0.342</u>
ETTm1	0.362	**0.357**	0.358	-0.71	0.370	<u>0.369</u>	0.397	0.383	0.373
ETTm2	0.276	**0.249**	0.253	-7.89	<u>0.254</u>	0.271	0.269	0.278	0.260
ECL	0.160	**0.159**	0.160	-0.02	0.298	<u>0.162</u>	0.185	0.175	0.169
Traffic	0.398	**0.394**	0.395	-1.04	0.746	0.404	0.450	<u>0.402</u>	0.412
Weather	0.222	0.222	**0.221**	-0.23	0.243	<u>0.226</u>	0.254	0.240	0.247
Solar	**0.188**	**0.188**	0.189	$+0.43$	0.224	<u>0.199</u>	0.337	0.206	0.229
Mean Δ	–	-4.25%	$\mathbf{-3.57\%}$	(27/32 wins)	*matches/beats best external model in 23/32 cells*				

Forecasting efficacy across benchmarks. As shown in Table 1, rectifying the readout scale deficit yields consistent improvements across diverse temporal dynamics. The multi-branch QR probe achieves a -4.25% benchmark-wide reduction, winning 28 of 32 cells. Directly matching this behavior within a single kernel, our branch-free **SpecScale** optimizer reduces mean MSE by $\mathbf{3.573\%}$ at default rates and $\mathbf{3.158\%}$ under equal-budget global-rate selection (Appendix H, Table 27), improving 27 of 32 cells. Across external baselines, SpecScale matches or outperforms the best competing spectral architecture on six of eight dataset averages (leading 23 of 32 cells on unrounded MSE). A fine-grained per-lead inspection confirms that this advantage is sustained across all horizons rather than localized (e.g., lowering test MSE at 719 of 720 forecast positions on ETTh2 $H = 720$; Appendix E.5, Figure 8).

Analytical insight: Geometric prior versus heuristic tuning. A natural question is whether Host could match these gains through empirical learning rate tuning alone. To test this, we compare SpecScale against a "Strong Host" baseline searching a $5\times$ larger candidate space (15 hyperparameter combinations per cell, jointly tuning global and base-kernel multipliers). Remarkably, SpecScale with only 3 candidate trials achieves statistical parity (-0.011% MSE difference, winning 17 of 32 cells; Table 27). This indicates that the analytic multiplier $(1 + n/c_b)$ acts as a principled geometric prior derived directly from the readout Jacobian, eliminating the need for expensive grid searches. Furthermore, by relieving terminal update stagnation, SpecScale accelerates optimization, attaining a median paired wall-clock training time ratio of $0.723\times$ across 96 runs on A100 GPUs (Table 32).

4.3 Trajectory Equivalence and Branch-Free Scaling

We next empirically verify Proposition 1: can multi-branch dynamics be reproduced within a single stored kernel? On the plain-attention host, the base kernel appears strictly in the terminal readout, satisfying whole-model closure and providing the exact testbed for single-kernel optimization.

Table 2: Trajectory reproduction and base-kernel scaling on plain-attention PSP (three seeds). **SpecScale-Exact** matches the multi-branch QR probe to numerical precision ($|\Delta| \leq 1.39 \times 10^{-7}$ across all 12 runs), while branch-free **SpecScale** $(1 + a_b)$ recovers 96.5% of the gain.

	Test MSE at the validation-best checkpoint				QR vs. Exact Agreement	
Cell	Host	QR (Probe)	SpecScale-Exact	SpecScale	Max $\lvert\Delta\rvert$	Max Rel. (%)
ETTh2 h720	0.453488	0.406197	0.406197	0.405802	6.5×10^{-9}	1.6×10^{-6}
ETTm2 h720	0.372987	0.347440	0.347440	0.348948	1.4×10^{-7}	4.0×10^{-5}
ECL h96	0.128256	0.128613	0.128613	0.128586	7.4×10^{-9}	5.8×10^{-6}
Solar h96	0.171378	0.169776	0.169776	0.170356	1.0×10^{-8}	6.2×10^{-6}
Mean (12 pairs)	cell means above (paired reduction: -4.48% vs. -4.32%)				1.389×10^{-7}	4.018×10^{-5}

As reported in Table 2, the SpecScale-Exact optimizer and the multi-branch QR probe achieve identical forecasting improvements across all 12 cell–seed pairs (both achieving a 4.4826% mean paired MSE reduction over Host). Across every step and validation checkpoint, their maximum absolute output discrepancy is 1.389×10^{-7} (maximum relative difference $4.018 \times 10^{-5}\%$), strictly within float32 numerical precision. This verifies Proposition 1: auxiliary coordinates can be completely eliminated during training without altering optimization trajectories.

We next evaluate branch-free SpecScale $(1 + a_b$, Equation 11), which discards virtual clipping and finite-ϵ tracking, applying the multiplier directly under standard AdamW. As shown in Table 2, SpecScale achieves a 4.3235% test MSE reduction across the diagnostic panel, recovering 96.5% of the empirical gain with zero auxiliary parameter, memory, or inference overhead.

Dissecting update scale versus frequency profile. To isolate whether the empirical gain originates from the frequency-dependent profile $(1 + a_b)$ or simply the enlarged base-kernel update magnitude, we evaluate two controls on the same panel (Table 15, Appendix D.2): applying a uniform multiplier $s = 1 + n/2$ across all frequencies yields a 4.3776% MSE reduction, while validation-tuning a scalar multiplier on the base kernel alone achieves 4.5700%, leaving upstream parameters untouched. These comparisons demonstrate that while SpecScale provides an explicit, hyperparameter-free default directly from the readout geometry, the primary empirical driver is restoring the base-kernel update scale in forecast space.

4.4 Mechanistic Diagnostics and Feature Propagation

We conduct four coordinated diagnostic experiments to decouple update scale from representational capacity and trace how terminal adjustments guide representation learning.

Frozen-feature controls. To separate terminal optimization from representation learning, we evaluate corrections on frozen representations across four cells ($n \in \{30, 60, 120, 360\}$). Using shared mini-batches and identical initial forecasts ($< 1.57 \times 10^{-6}$ discrepancy), rectifying the

terminal update scale consistently reaches lower validation minima and lowers test MSE by 0.97% across all 12 runs (Table 6). Matched controls further separate coordinate direction from scale: a duplicate folded branch and a fold-scaled quadrature branch follow identical training paths (ratio 2.000; halving rates recovers the single-fold baseline exactly; Table 7), confirming that coordinate scale, rather than orthogonal orientation, governs the optimization advantage.

Curvature and stability boundary. On synthetic spectra, measured Hessian eigenvalue ratios match the predicted $(n/2)^2$ curvature law within 0.04% across all lengths (Table 9, Figure 5). Full-batch gradient descent is stable at $0.9 \times \mathrm{lr}_{\mathrm{crit}}$ and diverges at $1.05 \times \mathrm{lr}_{\mathrm{crit}}$ across all lengths, verifying that coordinate geometry dictates both the update step scale and the theoretical stability boundary (Appendices C.2 and C.3).

Feature learning propagation. A synchronized, unclipped trace on ETTh2 h720 tracks when terminal adjustments begin altering upstream representations. At step 1, shared parameters remain identical while the auxiliary coordinate departs from zero ($\|W_q\| = 0.105$). By step 2, shared-parameter and phase-state gradient distances reach 10.4% and 12.3% relative to Host. By epoch 1, representation distance reaches 0.708, ultimately yielding a 10.55% lower test MSE across three seeds (Figure 7, Appendix D.1). This confirms that terminal update acceleration directly guides and improves upstream feature learning.

Causal rate interventions. Bi-directional rate interventions directly confirm the causal role of update scale. On four weighted-host cells, scaling Host's base-kernel rate alone reduces test MSE by 6.41%, recovering 79.9% of the multi-branch advantage (Table 10). Conversely, calibrating the initial update RMS to match Host diminishes the advantage to 0.84%. Increasing the base kernel's update scale reproduces the multi-branch gain, whereas attenuating the coordinate scale removes it (Appendix C.4).

4.5 Cross-Architecture Transfer to External Spectral Backbones

To test whether the prediction-space scale deficit represents an intrinsic bottleneck of discrete spectral readouts, we transfer our capacity-preserving intervention to three prominent external architectures: FITS (Xu et al., 2024) (complex-frequency interpolation), PaiFilter (Yi et al., 2024) (learned frequency filtering), and MixLinear (Ma et al., 2026) (time–frequency latents).

Across 96 evaluation pairs (8 datasets $\times$ 4 horizons $\times$ 3 seeds), adding a zero-initialized rank-four correction improves test MSE in 78 of 96 pairs for FITS and 60 of 96 pairs for PaiFilter, both lowering errors across seven of the eight dataset averages with mean paired MSE reductions of 0.191% and 0.184%, respectively (95% CIs: $[-0.360, -0.023]\%$ and $[-0.360, -0.007]\%$). On MixLinear, the intervention acts on its existing spectral latent and folds algebraically back into the original linear weights ($\widetilde{W}_F = W_F - i\,\mathrm{FFT}_{\mathrm{out}}(Q)$), reducing mean paired test MSE by 0.184% on the diagnostic panel without altering model parameters or deployment structure (Table 24). These consistent improvements across distinct model families confirm that inverse-transform scale attenuation is an architecture-agnostic bottleneck, and that rectifying update scale provides a general, portable optimization enhancement (Appendix E.4).

5 Conclusion

In this work, we show that empirical gains from specialized spectral readouts often reflect an optimization scale deficit rather than representational expansion: discrete inverse transforms inherently attenuate forecast-space updates under AdamW, throttling convergence. Under *whole-model closure*, multi-branch dynamics rigorously collapse into our single-kernel optimizer, **SpecScale**. Theoretically, its Exact formulation establishes when an architectural training effect can be represented entirely by an optimizer with zero trajectory divergence. Practically, its branch-free rule $(1 + n/c_b)$ makes this scale intervention directly usable, improving 27 of 32 weighted-PSP cells under equal-budget selection and transferring consistently to external spectral forecasters without structural overhead. Together, these findings turn readout geometry into a practical principle for training spectral forecasters. Future directions include dynamic scheduling of frequency multipliers across training phases and extending scale alignment to higher-dimensional spatio-temporal transforms.

AI USE STATEMENT

Generative AI was used for language polishing and writing assistance. The authors designed and conducted all experiments, verified all claims, and take full responsibility for the final content.

REFERENCES

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *ICML*, 2018.

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Huanru Henry Mao, Garrison W. Cottrell, and Julian McAuley. Rezero is all you need: Fast convergence at large depth. In *UAI*, 2021.

Xiaohan Ding, Honghao Chen, Xiangyu Zhang, Kaiqi Huang, Jungong Han, and Guiguang Ding. Re-parameterizing your optimizers rather than architectures. In *ICLR*, 2023. URL https://openreview.net/forum?id=B92TMCG_7rp.

Katie E Everett, Lechao Xiao, Mitchell Wortsman, Alexander A Alemi, Roman Novak, Peter J Liu, Izzeddin Gur, Jascha Sohl-Dickstein, Leslie Pack Kaelbling, Jaehoon Lee, and Jeffrey Pennington. Scaling exponents across parameterizations and optimizers. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12666–12700. PMLR, 2024. URL https://proceedings.mlr.press/v235/everett24a.html.

Shuxuan Guo, Jose M. Alvarez, and Mathieu Salzmann. ExpandNets: Linear over-parameterization to train compact convolutional networks. In *NeurIPS*, pp. 1298–1310, 2020. URL https://arxiv.org/abs/1811.10495.

Yue Hu, Jialiang Tang, Siwei Yu, Baosheng Yu, Jing Zhang, and Dacheng Tao. Timeapn: Adaptive amplitude-phase non-stationarity normalization for time series forecasting. *arXiv:2603.17436*, 2026.

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 *ICLR*, 2022.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Haojun Chen, and Junjie Yang. Sparsetsf: Modeling long-term time series forecasting with 1k parameters. In *ICML*, 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. In *ICLR*, 2024. URL https://openreview.net/forum?id=JePfAI8fah.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Donghao Luo and Xue Wang. ModernTCN: A modern pure convolution structure for general time series analysis. In *ICLR*, 2024. URL https://openreview.net/forum?id=vpJMJerXHU.

Aitian Ma, Dongsheng Luo, and Mo Sha. Mixlinear: Extreme low resource multivariate time series forecasting with $0.1K$ parameters. In *ICLR*, 2026.

Yiming Niu, Jinliang Deng, and Yongxin Tong. Phaseformer: From patches to phases for efficient and effective time series forecasting. In *ICLR*, 2026.

Oren Rippel, Jasper Snoek, and Ryan P. Adams. Spectral representations for convolutional neural networks. In *NeurIPS*, 2015. URL https://arxiv.org/abs/1506.03767.

Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *NeurIPS*, 2016. URL https://arxiv.org/abs/1602.07868.

Kexuan Shi, Xingyu Zhou, and Shuhang Gu. Improved implicit neural representation with Fourier reparameterized training. In *CVPR*, pp. 25985–25994, 2024. URL https://arxiv.org/abs/2401.07402.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *NeurIPS*, 2021. URL https://arxiv.org/abs/2106.13008.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *ICLR*, 2023.

Zhijian Xu, Ailing Zeng, and Qiang Xu. Fits: Modeling time series with 10k parameters. In *ICLR*, 2024.

Qingyuan Yang, Shizhuo Deng, Dongyue Chen, Da Teng, and Zehua Gan. Frwkv: Frequency-domain linear attention for long-term time series forecasting. *arXiv:2512.07539*, 2025.

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

Kun Yi, Jingru Fei, Qi Zhang, Hui He, Shufeng Hao, Defu Lian, and Wei Fan. Filternet: Harnessing frequency filters for time series forecasting. In *NeurIPS*, 2024. arXiv:2411.01623; PaiFilter and TexFilter are the plain and contextual shaping filters of this framework.

Chulhee Yun, Shankar Krishnan, and Hossein Mobahi. A unifying view on implicit bias in training linear neural networks. In *ICLR*, 2021.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *ICML*, pp. 27268–27286, 2022. URL https://proceedings.mlr.press/v162/zhou22g.html.

The supporting analyses follow the distinction between a forecast and the optimization that learns it. Appendix A establishes when QR can be folded without changing the predictor; Appendix B carries that equivalence through optimizer states and training. Appendices C and D test which parts of the induced rule explain terminal and feature learning. Appendix E then connects these mechanisms to the complete forecasting results and backbone adaptations. Deployment cost and the relation to prior interventions appear in Appendices F and G.

A Readout, host, and the folded forecast

A.1 Phase-Structured Predictor

The host determines whether a change to the terminal coordinates also changes the representation that reaches them. We specify its phase, frequency, and input-conditioned routes to identify every consumer of the readout kernel. The attention signal $m_{k,b}$ below measures mean kernel magnitude; the Jacobian scale $a_b = n/c_b$ instead measures how a terminal coordinate moves the forecast.

Let $x \in \mathbb{R}^{B\times N\times L}$ contain B length-L, N-variate windows, normalized as in Section 4.1. Each selected period p divides L and H, giving $n = L/p$ and $S_H = H/p$ future steps. Phase $0 \le j < p$ collects the subseries $x^{(j)} = (x_j, x_{j+p}, x_{j+2p}, \ldots)$, and step $1 \le s \le S_H$ predicts the horizon index $t = (s-1)p + j + 1$. The symbol S_H denotes this lead count throughout the appendix; the permutation S of Equation (8) is a different object.

The factorized base disables all three routes: phase subseries are independent, spectral bins do not interact, and every input uses the same readout kernel. Let h_j be the time-domain state of phase j, $\widetilde{X}^{\mathrm{E}}_{j,b}$ its aligned spectrum before cross-bin mixing, $X^{\mathrm{E}}_{j,b}$ the spectrum after mixing, and W_s the kernel of lead s. The three exclusions are zero cross-Jacobians:

$$\frac{\partial h_j}{\partial x^{(k)}} = 0 \ (j \neq k), \qquad \frac{\partial X^{\mathrm{E}}_{j,b}}{\partial \widetilde{X}^{\mathrm{E}}_{j,b'}} = 0 \ (b \neq b'), \qquad \frac{\partial W_s}{\partial x} = 0. \tag{12}$$

These derivatives concern normalized inputs and intermediate states.

Figure 2 draws the three routes and the single tensor that separates Host from QR. Phase attention lets a phase position use observations from other positions of the same period. Let $R_{j,b} = \mathrm{rfft}(x^{(j)}, n)_b$ and $\omega_b = 2\pi b/n$. Kernels are shared within three contiguous phase groups $k(j)$, including W_q in QR, and the base kernel $W^{k(j),b}_{\mathrm{base},s}$ is initialized to $e^{i\omega_b s}$, which advances bin b by s samples. Attention mixes phases through the real and the imaginary part of $m_{j,b}R_{j,b}$, where $m_{k,b} = S_H^{-1}\sum_s |W^{k,b}_{\mathrm{base},s}|$ is the mean readout magnitude over the S_H leads; write $\mathcal{A}_j$ for the mixed input that attention returns at phase j. With that feature and a zero-initialized length-n projection P_{att},

$$h_j = x^{(j)} + P_{\mathrm{att}}(\mathcal{A}_j), \tag{13}$$

so the block is initially the identity and later opens the first route of Equation (12). Each phase spectrum is then aligned with its last observed sample,

$$\widetilde{X}^{\mathrm{E}}_{j,b} = \mathrm{rfft}(h_j, n)_b \, e^{i\omega_b(n-1)}. \tag{14}$$

After alignment the same lead index s denotes the same future offset in every phase and sharing group, which is why the phase and group indices are omitted in the main text.

A learned map $\mathcal{M} : \mathbb{C}^{n_b} \to \mathbb{C}^{n_b}$ opens the second route by mapping concatenated real and imaginary coefficients to a mixed complex spectrum, with a zero-initialized scalar α controlling the residual:

$$X^{\mathrm{E}} = \widetilde{X}^{\mathrm{E}} + \alpha\Big[\mathcal{M}(\widetilde{X}^{\mathrm{E}}) - \widetilde{X}^{\mathrm{E}}\Big]. \tag{15}$$

At initialization this map returns the aligned spectrum; once α changes, each output bin can depend on every input bin. A magnitude-conditioned gate opens the third route by making the sample-shared prediction kernel respond to the current window:

$$G(X^{\mathrm{E}}) = \sigma\big(M|X^{\mathrm{E}}| + M_{\mathrm{agg}}\big|\mathrm{sg}(\mu_N(X^{\mathrm{E}}))\big|\big), \qquad W_s = W_{\mathrm{base},s} + G(X^{\mathrm{E}}) \odot \Delta W_s, \tag{16}$$

where μ_N averages spectra across variables, sg blocks gradients through that average, M and M_{agg} map variable-specific and aggregate bin magnitudes, σ is the logistic sigmoid, and ΔW_s is a residual kernel. Setting $\alpha = \Delta W_s = 0$ preserves the initial predictor.

We use Q for the quadrature tensor, also denoted W_q in the diagrams and diagnostics, and $D = \Delta W$ for the content-conditioned residual.

The two attention configurations differ in this dependency, not in their trainable parameter count. Weighted attention reads $|W_{\text{base}}|$ through $m_{k,b}$, so changing the base kernel can change the spectrum before prediction. Plain attention removes this multiplier and uses the base kernel only at the terminal map. This distinction determines whether folding preserves a trained predictor through caching or preserves the entire end-to-end function family directly.

A.2 Forecast equivalence and deployment folding

A terminal fold must first preserve the forecast produced from a given spectrum. The one-sided multiplier has real and imaginary derivatives

$$\frac{\partial z_s}{\partial \mathrm{Re}(W_s^b)} = \frac{c_b}{n}\,\mathrm{Re}(X_b^{\mathrm{E}}), \qquad \frac{\partial z_s}{\partial \mathrm{Im}(W_s^b)} = -\frac{c_b}{n}\,\mathrm{Im}(X_b^{\mathrm{E}}).$$

Together with $\mathrm{Im}(XW) = \mathrm{Re}(X(-iW))$, these derivatives give the absorption in Equation (4): QR changes the coordinate scale without adding a fixed-feature forecast direction. Raw rFFT endpoints are real at DC and, for even n, Nyquist. The cross-bin mixer can make these entries of X^{E} complex, activating both derivatives and bringing endpoint bins into the scale analysis. Figure 3 shows the two terminal paths and their common folded forecast.

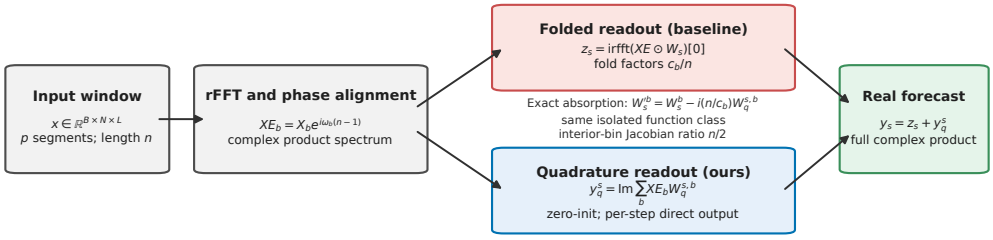


Figure 3: The readout and the quadrature coordinate. Both branches consume the same spectrum. The folded readout applies the step kernel inside the inverse transform; the quadrature readout adds a zero-initialized coordinate outside it and contributes its forecast directly. The middle annotation states the fold that collapses the two terminal weights into one effective kernel, so the branches differ in how they are trained rather than in the forecast they produce from the same features. The diagram is schematic and plots no measured quantity.

Folding with gates and shared attention weights. PSP's readout kernel is the gated kernel $W_s(x) = W_{\text{base},s} + G(X^{\mathrm{E}}(x)) \odot \Delta W_s$ that contracts with the spectrum. The base tensor also supplies phase-attention magnitudes, so deployment must preserve that use when folding the terminal weight. For each phase group k, cache

$$m_{k,b} = S_H^{-1} \sum_s |W_{\text{base},s}^{k,b}|, \qquad U_s^{k,b} = W_{\text{base},s}^{k,b} - i(n/c_b)W_q^{s,k,b}.$$

Keep the trained attention, mixer, gate, and normalization parameters unchanged. Compute attention with the cached m, then use $U_s + G(X^{\mathrm{E}}(x)) \odot \Delta W_s$ in the folded readout. The spectrum and gates remain unchanged for every input, and Equation (4) gives exactly the trained QR forecast. Phase-group sharing is preserved because W_q and W_{base} have the same allocation.

Cached folding preserves trained features and terminal forecasts. It removes the QR tensor and contraction using KF real magnitudes $m_{k,b}$, which Host can also precompute. Recomputing these

magnitudes from U would instead change attention and the resulting spectrum. The cached predictor is therefore identical to the trained QR model, while retaining its learned attention signal. With all three modules active, nonzero parameters, $p \in \{2, 12, 24\}$, and three seeds, the maximum float32 output discrepancy after this fold is 6.68×10^{-6}, about 56 float32 steps at unit magnitude.

End-to-end equivalence with plain attention. Removing shared kernel conditioning gives a direct end-to-end function-family comparison. In the parameter-matched plain-attention control (Appendix A.3), attention omits $m_{k,b}$ and the base kernel appears only in the terminal map. Write all other parameters as θ, so $X^{\mathrm{E}}_\theta(x)$ and $G_\theta(x)$ are independent of W_{base} and W_q. For every $(\theta, W_{\mathrm{base}}, W_q)$, replacing W_{base} by U and removing QR preserves the forecast for every x. Conversely, $W_q = 0$ embeds every plain-attention Host. The two full function families are therefore equal. The 12-pair comparison in Appendix B.3 tests training under this stronger equivalence.

A.3 Plain-attention host

Plain attention supplies the stronger equivalence needed to study a branch-free optimizer. It removes the mean-kernel-magnitude multiplier before the shared width-eight attention embedding and keeps all trainable attention parameters. Features and gates then depend on shared parameters rather than on the base and quadrature weights separately. The terminal absorption above consequently preserves the full function family. Appendix D.2 gives the settings used to test training in this configuration.

B The induced optimization rule

B.1 The induced effective optimization rule

Forecast equivalence alone leaves the training rule undetermined: two decompositions of the same folded kernel can carry different optimizer states. We derive the conditions that make those states recoverable from a single moment pair. The resulting rule turns QR's redundant coordinate into a direct update of the effective kernel.

Effective kernel and architecture criterion. Write $a_b = n/c_b$ for the Jacobian scale of bin b, and let W_b and Q_b denote the base and quadrature weights of bin b, with W_b abbreviating the base kernel and Q also written as W_q in the architecture diagrams and diagnostics. The base kernel and the quadrature weight enter the readout only through

$$U_b = W_b - i a_b Q_b. \tag{17}$$

The gate residual D is not rescaled: Q folds into the ungated base kernel, not into the gated residual term. Let A be the diagonal scaling that multiplies the real and imaginary coordinates of bin b by a_b. A whole forward map is representable by (θ, U, D) exactly when it is constant over decompositions sharing U,

$$(W, Q) \sim (W', Q') \iff W - iAQ = W' - iAQ' = U. \tag{18}$$

A function of U cannot distinguish equal-U decompositions, which proves necessity; constancy over those decompositions proves sufficiency. This closure holds when a is fixed, positive, and input-independent, the terminal map has Equation (4)'s form with ungated Q, and features, gates, and auxiliary losses do not read W or Q separately. Gates may be learned and nonlinear, and compatible sharing of W/Q parameters is allowed. Learned or input-dependent multipliers additionally affect derivatives and optimizer states, so they require a different closure analysis.

Matched gradient descent. Closure converts the coordinate change into a gradient relation. Let g_U be the collapsed loss gradient, and let R represent multiplication by $-i$ in real coordinates, mapping $(\mathrm{Re}\, z, \mathrm{Im}\, z)$ to $(\mathrm{Im}\, z, -\mathrm{Re}\, z)$. Then $g_W = g_U$ and $g_Q = R^\top A g_U$, giving the redundant model's terminal gradient norm

$$\|g_W\|^2 + \|g_Q\|^2 = \sum_b (1 + a_b^2) \|g_{U,b}\|^2. \tag{19}$$

Zero-decay gradient descent with branch rates η_W, η_Q and clipping ρ yields

$$U_b^+ = U_b - \rho\,(\eta_W + a_b^2 \eta_Q)\, g_{U,b}, \tag{20}$$

The matched rate is therefore $\eta_W + a_b^2\eta_Q$ per bin, or $1 + a_b^2$ times the common branch rate. Active bins with unequal a_b require different rates whenever the quadrature rate is nonzero.

Matched Adam. Adam additionally needs a moment map. With equal betas, step counters and bias correction, the matched states satisfy

$$m_Q = R^\top A m_W, \qquad v_Q = S^\top A^2 v_W, \qquad S = |R|. \tag{21}$$

Zero initialization satisfies Equation (21), and nonzero warm starts can satisfy it too. Writing $s_b = \sqrt{\hat{v}_{W,b}}$ for the bias-corrected second moment, the folded update is

$$\Delta U_b = -\hat{m}_{W,b}\left[\frac{\eta_W}{s_b + \epsilon_{W,b}} + \frac{a_b\eta_Q}{s_b + \epsilon_{Q,b}/a_b}\right]. \tag{22}$$

At zero epsilon and $s_b > 0$ this is the fixed per-bin rate $\eta_W + a_b\eta_Q$, a factor $1 + a_b$ at equal branch rates. The choice $\epsilon_{Q,b} = a_b\epsilon_{W,b}$ reproduces the same fixed rate exactly, with the single denominator $s_b + \epsilon_{W,b}$. Equal nonzero epsilons still admit an exact collapse, but the rate becomes state-dependent,

$$\eta_W + a_b\eta_Q\,\frac{s_b + \epsilon_{W,b}}{s_b + \epsilon_{Q,b}/a_b}.$$

For $a_b \geq 1$ this rate lies between $\eta_W + a_b\eta_Q$ and $\eta_W + a_b^2\eta_Q$. For a zero first moment with $s_b = 0$, an explicit zero-update convention is needed. This does not define native zero-epsilon Adam.

Why the moment map holds. The moment relation persists because the gradient relation persists. Common clipping preserves that relation, the first-moment recurrence is linear, and elementwise squaring removes the signs while retaining the permutation and a_b^2 scaling. One W-coordinate moment pair therefore determines both branch updates through Equation (22). Decoupled decay must also preserve the decomposition: $d_W W + ARd_Q Q = d_W U + (d_Q - d_W)ARQ$ depends only on U when the branch decay factors agree. Matching the forward fold further requires matching differentiation. A stop-gradient on the quadrature feature path removes $\partial y_q^s/\partial\theta = \mathrm{Im}\sum_b(\partial X_b^{\mathrm{E}}/\partial\theta)Q_b$. It vanishes at $Q = 0$ but can transmit the intervention upstream once Q moves, provided the contraction and loss gradient are nonzero.

Augmented clipping. Removing the explicit branch must preserve its contribution to the global gradient norm. The folded optimizer reconstructs this contribution from g_U:

$$N^2 = \|g_\theta\|^2 + \|g_D\|^2 + \sum_b (1 + a_b^2)\|g_{U,b}\|^2, \tag{23}$$

Every group uses this coefficient. Shared parameters and D follow their ordinary optimizers using the matched clipped gradients; their updates receive no additional bin-scale multiplier. Native clipping of the folded model would omit the virtual quadrature contribution and change these shared updates as well. Implementation-level matching also preserves any additive denominator stabilization, such as PyTorch's $\min(1, C/(N + 10^{-6}))$. This is an additive term, not a floor on N.

Trajectory induction. Suppose both runs satisfy Equation (18) and begin with equal (θ, U, D), mapped optimizer states, and equal step counters. Feed them the same batches and stochastic masks, with matching backward rules and no unmatched regularization, projection, or state reset. Closure gives equal predictions and shared gradients; the coordinate relation gives the virtual branch gradients. Equation (23) then gives the same clipping coefficient, while the moment map and Equation (22) give the same next U. Shared parameters and D also remain equal, establishing the hypotheses for the next step. This induction covers trainable features and nonlinear gates. Weighted attention falls outside closure because it reads $|W_{\mathrm{base}}|$ separately; cached deployment folding preserves its trained predictor but does not make its training state a function of U alone.

B.2 Numerical verification of the identities

Numerical checks connect the symbolic rule to the operations used during training. Table 3 evaluates the identities in float64 NumPy, including a nonlinear tanh feature extractor, sigmoid gate, finite epsilon, and active clipping. An independent central-difference gradient check agrees to 1.16×10^{-10}.

Table 3: Analytic identities of the induced rule in float64 NumPy, as maximum absolute error. Columns are the checks of the derivation: the fold on a single evaluation, matched gradient descent, matched Adam at the three epsilon choices of Equation (22), and a gated run whose clipping is active at every one of its 100 updates. The clipped run reproduces θ, the upstream gradient and the clipping coefficient to $1.1\mathrm{e}{-16}$, $2.2\mathrm{e}{-16}$ and $1.4\mathrm{e}{-16}$ as well.

Quantity	Fold absorption	Matched GD	Adam $\epsilon = 0$	Adam $\epsilon_Q{=}a_b\epsilon_W$	Adam $\epsilon_Q{=}\epsilon_W$	Gated clipped Adam
Steps	1	9	11	11	11	100
Effective kernel	–	$2.5\mathrm{e}{-16}$	$2.2\mathrm{e}{-16}$	$4.4\mathrm{e}{-16}$	$2.2\mathrm{e}{-16}$	$1.4\mathrm{e}{-15}$
Prediction	$8.9\mathrm{e}{-16}$	$6.7\mathrm{e}{-16}$	$6.7\mathrm{e}{-16}$	$1.0\mathrm{e}{-15}$	$8.9\mathrm{e}{-16}$	$1.2\mathrm{e}{-15}$
Moment map	–	–	$7.1\mathrm{e}{-15}$	$3.6\mathrm{e}{-15}$	$7.1\mathrm{e}{-15}$	$2.8\mathrm{e}{-17}$
Per-bin identity	$1.8\mathrm{e}{-15}$	–	–	–	–	–

Table 4: Full-model replay of the explicit-coordinate model against the collapsed rule: maximum absolute differences over 12 updates. Real inputs come from the training splits of the four diagnostic cells. The separate float64/complex128 replay uses batches of two with clipping active at every update. Shared denotes shared parameters; moment error is the maximum over the recorded first- and second-moment comparisons, including mapped branch moments and shared states where recorded. Two-line entries span cell maxima; the separate-path row reports their maximum.

Run	Precision	Cells	Steps	Eff. kernel	Prediction	Shared	Moment error
Synthetic, $Q_0 = 0$	float64	1	12	$2.2\mathrm{e}{-16}$	$1.3\mathrm{e}{-15}$	$8.9\mathrm{e}{-16}$	$5.6\mathrm{e}{-17}$
Synthetic, $Q_0 \neq 0$	float64	1	12	$1.5\mathrm{e}{-13}$	$3.7\mathrm{e}{-12}$	$1.3\mathrm{e}{-11}$	$8.9\mathrm{e}{-13}$
				$2.5\mathrm{e}{-7}$	$1.7\mathrm{e}{-6}$	$5.4\mathrm{e}{-7}$	$1.5\mathrm{e}{-8}$
Real batches	FP32	4	12	$5.3\mathrm{e}{-5}$	$7.6\mathrm{e}{-6}$	$3.0\mathrm{e}{-6}$	$9.4\mathrm{e}{-8}$
Real batches, separate path	float64	4	12	$1.7\mathrm{e}{-14}$	$7.1\mathrm{e}{-15}$	$4.8\mathrm{e}{-15}$	$8.3\mathrm{e}{-17}$

Table 4 then runs explicit QR and the collapsed optimizer through the full PSP forward and backward paths, using both synthetic inputs and real training batches.

The checks test different parts of the construction. Equal epsilon requires the state-dependent rate in Equation (22); the nonlinear clipped run tests upstream updates; and the nonzero-Q_0 replay tests a mapped warm start, with maximum effective-kernel difference 1.5×10^{-13}. Real-batch replays carry the rule through the native model at float64 and FP32 precision. The next comparison tests whether these matched updates recover QR's validation-selected forecasts after complete training.

B.3 Constructive equivalence on the plain-attention host

The plain-attention host makes the optimizer explanation testable end to end because its entire function family satisfies closure. We compare Host, explicit QR, and two branch-free arms on ETTh2 h720, ETTm2 h720, ECL h96, and Solar h96 at seeds 42–44. Appendix D.2 supplies the common recipe: zero-decay AdamW, constant rate, unit global clipping, at most 30 epochs, and validation-best checkpoints. All four arms share CPU training, initialization, mini-batches, and stochastic states.

Exact stores U with one moment pair, reconstructs the augmented norm in Equation (23), and applies the finite-epsilon update in Equation (22). *Fixed-bin scaling* instead multiplies the ordinary per-bin update by $1 + a_b$, retaining the ideal Adam scale but omitting finite-epsilon matching and augmented clipping. Both leave gate-residual and upstream groups unscaled. The exact arm tests reproduction of QR; fixed-bin scaling tests how much forecasting value remains in the fixed geometric multiplier.

The branch-free exact optimizer reproduces QR at the validation-selected test metric. Both reduce mean paired MSE by 4.4826% against Host over the 12 pairs in Table 5. Their largest absolute test-MSE difference is 1.389×10^{-7}, their largest individual relative difference is $4.02 \times 10^{-5}\%$, and their mean individual relative change is $4.6 \times 10^{-6}\%$. Even the smallest seed-to-seed spread among the four cells exceeds that absolute discrepancy by more than three orders of magnitude. Per-pair differences expose the numerical agreement that rounded cell means would conceal.

Table 5: Paired panel for the induced optimizer on the plain-attention host. Each row is one cell–seed pair; MSE is the test value at the validation-best checkpoint. QR is the explicit-quadrature arm. The last two columns describe the collapsed arms: $|\Delta|_{\mathrm{ex}}$ is the absolute test-MSE difference between the exact collapsed optimizer and QR, and Δ_{fix} is the fixed-bin arm's relative test-MSE change from QR in percent, positive meaning larger MSE. QR and the exact collapsed arm agree at the displayed MSE precision.

		Reference		Collapsed arms			
Cell	Seed	Host	QR	Exact	Fixed-bin	$\lvert\Delta\rvert_{\mathrm{ex}}$	Δ_{fix}
ETTh2 h720	42	0.453740	0.406729	0.406729	0.406218	6.5×10^{-9}	-0.126
ETTh2 h720	43	0.452311	0.403717	0.403717	0.403389	3.0×10^{-9}	-0.081
ETTh2 h720	44	0.454414	0.408147	0.408147	0.407799	7.1×10^{-10}	-0.085
ETTm2 h720	42	0.370018	0.347861	0.347861	0.348816	3.1×10^{-9}	$+0.275$
ETTm2 h720	43	0.376841	0.345725	0.345725	0.348123	1.4×10^{-7}	$+0.693$
ETTm2 h720	44	0.372102	0.348733	0.348733	0.349905	1.6×10^{-8}	$+0.336$
ECL h96	42	0.128269	0.128198	0.128198	0.128105	7.4×10^{-9}	-0.072
ECL h96	43	0.128478	0.128799	0.128799	0.128810	1.5×10^{-9}	$+0.009$
ECL h96	44	0.128022	0.128843	0.128843	0.128844	2.6×10^{-9}	$+0.001$
Solar h96	42	0.173072	0.172805	0.172805	0.172380	2.8×10^{-9}	-0.246
Solar h96	43	0.170667	0.167027	0.167027	0.167992	1.0×10^{-8}	$+0.578$
Solar h96	44	0.170395	0.169496	0.169496	0.170695	1.9×10^{-9}	$+0.707$

The fixed multiplier retains most of the forecasting benefit without reconstructing QR's full trajectory. It lowers mean paired MSE by 4.3235% against Host, improving three of four cell means; ECL h96 is 0.26% above Host. Relative to QR, it is lower in 5 of 12 pairs and higher in 7, with mean change $+0.1657\%$ and maximum $+0.707\%$. These comparisons use QR as denominator. On the common Host-relative scale, the reductions 4.4826% and 4.3235% differ by 0.159 percentage points, giving 96.5% recovery of QR's gain. This links bin geometry to branch-free training.

The evidence connects the update identity to its forecasting consequence. Under whole-model closure and matched optimizer assumptions, induction establishes trajectory equality in exact arithmetic, and the numerical replays test it through the model's forward and backward operations. The 30-epoch panel then compares the resulting validation-selected forecasts. Both collapsed arms remove the quadrature tensor from the forward model; the exact arm reproduces its induced rule, while fixed-bin scaling uses the geometric multiplier directly.

The rate controls in Appendix C.4 complement this construction by intervening on terminal scale in the weighted host. Fixed-bin scaling changes finite-epsilon matching and clipping together, so these results estimate their combined influence on training.

C Coordinate scale, curvature, and terminal rates

C.1 Frozen representations and update-scale controls

With the features held fixed, the terminal coordinates change how training moves without changing the forecasts available from those features. These controls train terminal corrections on frozen representations, separating the added coordinate's update scale from its orientation.

Frozen PSP representations. The validation-best Host fixes both the representation and the forecast from which the two corrections start. We freeze its parameters and train zero-initialized folded and quadrature corrections for 30 epochs with unclipped, zero-decay AdamW. The corrections receive identical ordered mini-batches and use their own validation-best checkpoints for testing. Initial predictions agree in all 12 runs. Absorbing the trained QR correction into a folded correction gives a maximum observed prediction discrepancy of 1.565×10^{-6} at the same fixed features and gate outputs. Table 6 therefore compares optimization within an unchanged terminal forecast family.

Forecast equivalence removes added terminal capacity as an explanation for these gains, leaving coordinate orientation and scale to distinguish. Table 7 first compares a rotated branch with a

Table 6: QR reaches a lower validation minimum in all 12 frozen-representation pairs and lowers their mean paired test MSE. Each correction uses its own validation-best checkpoint; entries average seeds 42–44, and negative values favor QR.

Cell	n	Epoch-1 train	Epoch-3 train	Best-valid	Test
Weather h96	30	$+0.125$	$+0.114$	-0.207	-0.058
ETTh2 h720	60	-4.014	-3.425	-1.276	$+0.053$
ETTh1 h720	120	-2.866	-7.340	-0.602	-3.210
ETTm2 h720	360	-1.033	-2.188	-1.678	-0.677
Mean over 12 pairs	–	–	–	-0.941	-0.973

duplicate folded branch at equal scale; their identical histories show that the rotation does not change training in this control. Halving both redundant-branch rates then recovers the single-fold history exactly. Matching the aggregate prediction-update RMS instead gives near parity. The successive comparisons locate the operative difference in update scale.

Table 7: Rotation- and scale-matched coordinates recover identical finite-budget histories on frozen ETTh2 h720. For seed 42 and $n = 60$, Step ratio is prediction-update RMS divided by that of the single folded correction; "Exact" marks the fold-scaled/duplicate-fold and half-rate rows, whose training, validation, and test histories coincide; the RMS-matched row reaches near parity.

Intervention	First-step ratio	Consequence
Folded + raw quadrature	42.372	different trajectory
Fold-scaled quadrature / duplicate fold	2.000	identical pair
Both redundant branches at half rate	1.000	exact vs. baseline
Raw quadrature, RMS matched	1.000	near parity[†]

[†] AdamW validation/test change $-0.013\%/+0.066\%$. SGD is numerical zero.

C.2 Synthetic length and displacement grid

The scale law predicts that changing transform length changes optimization even when the target construction is held fixed. We test this by varying n and the target displacement ratio $\kappa = \mathrm{mean}_{s,b} |\Delta_{s,b}| / \mathrm{mean}_{s,b} |W^{s,b}_{\mathrm{init}}|$. Figures 4 and 5 test the Jacobian and curvature predictions; Table 8 measures their consequences for held-out MSE after 800 updates across 16 (n, κ) settings and five seeds. Thus κ controls the target displacement; the optimizer's ρ instead controls gradient clipping.

The synthetic task is constructed as follows. Inputs are length-n windows of three sinusoids at 1.0, 2.3, and 4.7 cycles per window with amplitudes 1.0, 0.5 and 0.3, plus Gaussian input noise of standard deviation 0.05, over a window set of $60n$ windows. We apply the phase alignment of Appendix A.1, and the target follows from $W^* = W_{\mathrm{init}} + \Delta$ and Equation (1), where Δ is a complex Gaussian draw whose DC and Nyquist imaginary parts are set to zero and which is then renormalized so that $\mathrm{mean}\,|\Delta| = \kappa\,\mathrm{mean}\,|W_{\mathrm{init}}|$ exactly. One draw per (n, κ) cell defines the target for all seven arms. Independent output noise of standard deviation 0.01 precedes a random 80/20 split. This fixed-feature task uses AdamW with no gradient clipping, the default decoupled weight decay 0.01, learning rate 10^{-3}, and batch size 64.

The seven arms distinguish coordinate orientation, forward scale, and the location of rate scaling. *Base* trains one folded weight; *quad* adds the raw quadrature coordinate. *Q.-scaled* multiplies the quadrature contribution by c_b/n, while *dupfold* adds a second folded weight. These last two constructions match branch scale and differ only by rotation. Both retain two trainable paths, so at equal rates their combined update need not match Base. *Q.-pergrp* instead applies the interior-bin inverse scale $2/n$ to the quadrature group's rate alone. *Q.-global-lr* and *base-global-lr* apply $2/n$ to every parameter's rate in their respective models. This distinguishes a targeted branch adjustment from slowing the entire model. All seven arms start from equal predictions.

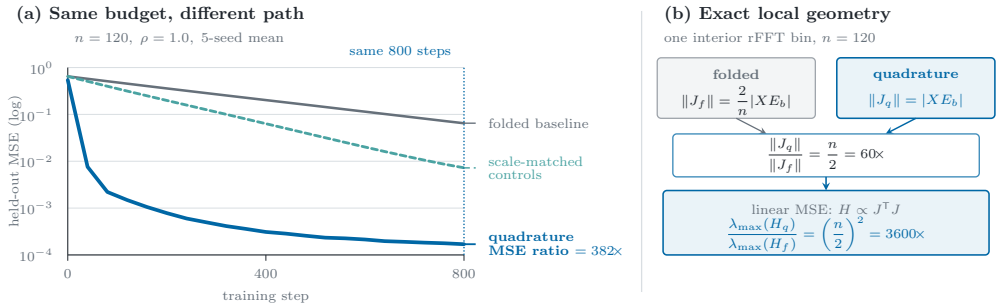


Figure 4: Jacobian and curvature measurements follow the predicted $n/2$ and $(n/2)^2$ laws. Ratios compare quadrature and folded blocks. At $n = 120$, $\kappa = 1.0$, the Jacobian ratio is 60; predicted interior-bin and measured full-spectrum curvature ratios are 3600 and 3601.2.

Table 8: Scale-matched controls agree to within 2.1% in the 14 stable synthetic settings, while the quadrature arm reaches lower fixed-budget MSE. Entries are five-seed held-out MSE after 800 updates for all 16 cells; the two bold cells mark the high-n stability boundary.

n	κ	base	quad	q.-scaled	dupfold	q.-pergrp	q.-global-lr	base-global-lr
30	0.3	3.4e-4	1.2e-4	2.5e-4	2.5e-4	2.5e-4	3.4e-4	2.2e-2
30	0.6	3.2e-3	1.7e-4	1.1e-3	1.1e-3	1.1e-3	2.7e-3	1.2e-1
30	1.0	3.1e-2	6.3e-4	5.6e-3	5.6e-3	5.6e-3	2.5e-2	3.7e-1
30	1.5	1.6e-1	2.4e-3	3.4e-2	3.4e-2	3.4e-2	1.3e-1	8.9e-1
60	0.3	2.9e-4	1.3e-4	2.0e-4	2.0e-4	2.0e-4	2.8e-4	4.9e-2
60	0.6	7.0e-3	1.3e-4	8.5e-4	8.5e-4	8.5e-4	6.2e-3	2.2e-1
60	1.0	8.4e-2	2.0e-4	9.6e-3	9.6e-3	9.5e-3	7.7e-2	6.4e-1
60	1.5	3.8e-1	5.5e-4	9.1e-2	9.1e-2	9.0e-2	3.6e-1	1.5e0
120	0.3	3.3e-4	1.7e-4	2.4e-4	2.4e-4	2.4e-4	3.3e-4	5.1e-2
120	0.6	5.2e-3	1.4e-4	1.0e-3	1.0e-3	1.0e-3	4.6e-3	2.2e-1
120	1.0	6.5e-2	1.7e-4	7.2e-3	7.2e-3	7.2e-3	6.0e-2	6.3e-1
120	1.5	3.1e-1	3.4e-4	6.7e-2	6.7e-2	6.6e-2	3.0e-1	1.4e0
360	0.3	2.9e-4	**1.2e-3**	2.0e-4	2.0e-4	2.0e-4	2.9e-4	3.1e-2
360	0.6	1.8e-3	**1.0e-3**	8.6e-4	8.6e-4	8.6e-4	1.8e-3	1.3e-1
360	1.0	2.0e-2	4.5e-4	3.4e-3	3.4e-3	3.3e-3	2.0e-2	3.6e-1
360	1.5	1.1e-1	2.8e-4	2.1e-2	2.1e-2	2.0e-2	1.1e-1	8.2e-1

Length changes the useful step scale as well as the optimization speed. In the 14 stable settings, QR reaches lower held-out MSE than Base, while the fold-scaled, duplicate-fold, and branch-rate controls agree within 2.1% on unrounded values. At $n = 360$, the two bold settings show different consequences of the larger scale: QR ends above Base at $\kappa = 0.3$, and below Base but above the scale-matched controls at $\kappa = 0.6$. QR improves on Base in 15 of 16 settings overall. These AdamW trajectories motivate a curvature analysis of the same coordinates; the following experiment tests the exact stability threshold for plain gradient descent.

C.3 Curvature and the stability boundary

The Jacobian scale links the observed step changes to a calculable stability range. With fixed features and squared loss, the parameter Hessian is proportional to $J^\top J$, whose nonzero eigenvalues equal those of $JJ^\top$. The per-bin identity $K_{q,b} = a_b^2 K_{f,b}$ therefore predicts the curvature ratio as well as the plain-gradient update ratio. Table 9 checks the Hessian with automatic differentiation, includes endpoint bins, and tests full-batch GD around its predicted threshold. Figure 5 connects the eigenvalue ratio to the loss slice and stability sweep. The table's $\mathrm{BASE_LR}/\mathrm{lr}_{\mathrm{crit}}$ row compares the common reference rate 10^{-3} with this threshold; the actual stability tests use 0.9 and 1.05 times the threshold.

At every n, gradient descent on the quadrature block is stable at $0.9 \times \mathrm{lr}_{\mathrm{crit}}$ and diverges at $1.05 \times \mathrm{lr}_{\mathrm{crit}}$, matching $\mathrm{lr} < 2/\lambda_{\max}(\mathcal{H})$ for the quadrature-block Hessian $\mathcal{H}$. The common 10^{-3} rate exceeds this

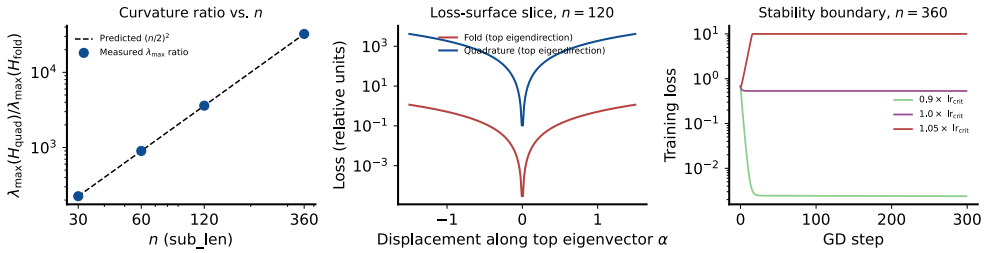


Figure 5: Quadrature curvature follows $(n/2)^2$ and predicts the plain-GD stability boundary. Left: measured Hessian top-eigenvalue ratio of the quadrature block to the folded block. Middle: the $n = 120$ loss slice is $\sim 3600\times$ steeper than the folded one. Right: $n = 360$ full-batch GD at $0.9\times$, $1.0\times$, and $1.05\times$ the predicted stability limit.

Table 9: Curvature and the GD stability transition match the analytic scale law at all four lengths.

Quantity	$n = 30$	$n = 60$	$n = 120$	$n = 360$
Interior eig. ratio, max rel. err. vs. $(n/2)^2$	1.6e−13	5.4e−13	1.5e−12	5.9e−12
Full-spectrum $\lambda_{\max}$ ratio (measured)	225.08	900.30	3601.2	32,410.9
Predicted $(n/2)^2$	225	900	3600	32,400
Deviation from predicted	0.035%	0.034%	0.033%	0.034%
Endpoint-bin power fraction	0.61%	0.55%	0.52%	0.51%
Autograd cross-check, fold block (max abs. err.)	1.8e−15	6.7e−16	1.1e−15	4.7e−15
Autograd cross-check, quad block (max abs. err.)	6.0e−13	3.1e−12	1.9e−11	2.6e−10
$\mathrm{lr}_{\mathrm{crit}}$(quad. block), plain GD	8.65e−3	2.16e−3	5.41e−4	6.01e−5
$\mathrm{BASE_LR}/\mathrm{lr}_{\mathrm{crit}}$(quad. block)	0.116	0.464	1.85	16.63
GD divergence check ($1.05 \times \mathrm{lr}_{\mathrm{crit}}$)	diverges	diverges	diverges	diverges
GD stability check ($0.9 \times \mathrm{lr}_{\mathrm{crit}}$)	stable	stable	stable	stable

GD threshold from $n = 120$ onward, although the AdamW synthetic task converges at $n = 120$. The Hessian calculation thus characterizes unpreconditioned curvature, whereas Equation (5) gives the corresponding Adam first-step scale under its stated assumptions. Appendix C.2 measures its later stability as the moment states evolve.

The endpoint-bin power fraction in Table 9 is 0.5–0.6% because the controlled input is a low-frequency three-sinusoid mixture. This yields a 0.03–0.04% deviation from the interior-bin $(n/2)^2$ prediction. Concentrating power at DC and Nyquist changes the full-spectrum ratio: at $n = 120$, the measured 14,400.0 matches $n^2 = 14{,}400$ to relative error 4.4×10^{-7} and equals $4.000\times$ the interior-bin prediction. The interior-bin identity is input-independent, whereas the full-spectrum ratio depends on power distributed across interior and endpoint bins.

C.4 Sustained kernel-rate and branch-rate controls

The scale explanation has an end-to-end consequence: changing terminal rates should recover or suppress part of QR's forecasting effect. We test both directions on the weighted host while leaving upstream learning rates unchanged. Because phase attention consumes $|W_{\mathrm{base}}|$, the terminal intervention also changes its conditioning signal as training proceeds. Complete PSP runs use zero-decay AdamW, unit global clipping, 30 epochs, and seeds 42–44. Paired runs share common-parameter initialization, mini-batches, and forward random-number-generator states; validation MSE selects each test checkpoint.

For the Host kernel-group control, the optimizer group contains the base-kernel and content-conditioned tensors that form W_s in Equation (16); every other tensor retains the adopted rate. Seed-42 validation selects a multiplier λ_W on the adopted rate of that group, equal to 2, 16, 60 and 180 for Weather h96, ETTh2 h720, ETTh1 h720, and ETTm2 h720 respectively, from $\{0.25, 0.5, 1, 2, 4, 8, 16, n/2\}$. Each value is fixed before the three-seed test evaluation, which includes an identically initialized default QR reference.

For the branch-rate-calibrated control, all upstream tensors retain the adopted rate. The existing folded-multiplier group uses λ times that rate and the quadrature tensor uses $2\lambda/n$ times that rate throughout training; the factor $2/n$ is the interior-bin inverse Jacobian scale c_b/n of Section 3.1. Because the folded multiplier supplies the phase-attention weight, this sustained calibration also changes an upstream conditioning signal after the first update. A one-dimensional search chooses λ using only the first shuffled training mini-batch, minimizing the difference between Host and QR prediction-update RMS. Mean multipliers in increasing-n order are 0.592, 0.593, 0.594, and 0.600.

Table 10: Sustained branch-rate calibration reduces QR's 8.02 percentage-point advantage over the Host to 0.84 while matching the first prediction step. Results average seeds 42–44; ΔMSE is relative to the same paired default-rate Host, validation selects λ_W, and Step reports the raw/calibrated RMS ratio to Host. Bold marks the largest reduction in each row.

Cell	n	λ_W	Kernel-rate Host	QR	Calibrated QR	Step
Weather h96	30	2	$+0.087$	$\mathbf{-0.997}$	-0.798	$2.000/1.000$
ETTh2 h720	60	16	$\mathbf{-10.698}$	-10.415	$+0.052$	$3.178/1.000$
ETTh1 h720	120	60	-7.951	$\mathbf{-13.339}$	-1.106	$4.222/1.000$
ETTm2 h720	360	180	-7.095	$\mathbf{-7.328}$	-1.515	$9.719/1.000$
Mean over 12 pairs	–	–	-6.414	$\mathbf{-8.020}$	-0.842	–

Table 11: All 12 calibrated runs match Host's first prediction step within 1.91×10^{-5}. MSE columns are paired test changes (%) relative to the same paired default-rate Host, so negative values favor the row's method; Step gives the RMS ratio to Host before and after calibration.

Cell	Seed	Kernel-rate Host	QR	Calibrated QR	Raw step	Calibrated step
Weather h96	42	$+0.215$	-1.439	-0.734	1.812	1.000003
Weather h96	43	$+1.101$	-1.076	-1.372	2.031	0.999982
Weather h96	44	-1.055	-0.477	-0.289	2.158	1.000012
ETTh2 h720	42	-10.462	-10.273	-0.139	3.072	1.000009
ETTh2 h720	43	-10.361	-10.620	$+0.143$	3.638	0.999981
ETTh2 h720	44	-11.270	-10.352	$+0.152$	2.823	1.000004
ETTh1 h720	42	-9.065	-15.122	-1.465	4.284	1.000007
ETTh1 h720	43	-10.842	-13.090	-1.474	4.504	0.999998
ETTh1 h720	44	-3.946	-11.807	-0.380	3.877	1.000001
ETTm2 h720	42	-5.440	-6.272	-1.383	9.666	0.999999
ETTm2 h720	43	-6.074	-6.434	-1.280	9.904	1.000010
ETTm2 h720	44	-9.771	-9.279	-1.881	9.589	1.000001

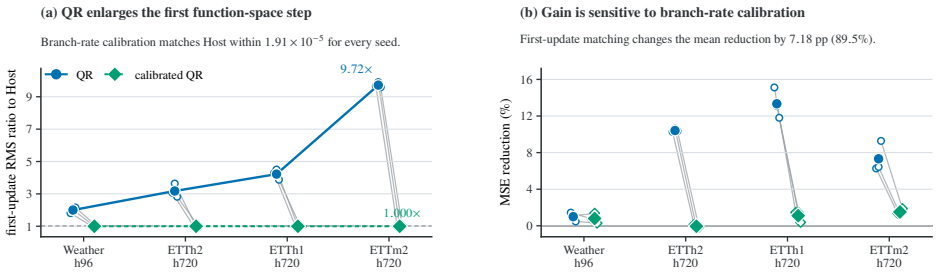


Figure 6: Terminal branch rates under a sustained intervention, the curve summarized by Table 10. (a) Calibration matches the first prediction-update root-mean-square change of the two models. (b) The calibrated rates then stay fixed for the remaining training, so later steps are not matched. Filled markers are cell means and open markers individual runs.

The two interventions move forecasting accuracy in the directions predicted by the scale explanation. Against the same paired default-rate Host, QR reduces mean MSE by 8.020%, validation-tuned Host kernel rates by 6.414%, and calibrated QR by 0.842% (Tables 10 and 11). Calibration matches the first prediction step within 1.906×10^{-5}, yet the subsequent rates stay fixed rather than matching every later update. The 7.178-percentage-point reduction in QR's advantage thus measures a sustained rate intervention, while optimizer states and attention conditioning evolve.

D END-TO-END LEARNING

D.1 SYNCHRONIZED TRAINING TRACE

The terminal explanation becomes a feature-learning explanation only if its changed updates reach the shared representation. We test how the quadrature branch propagates into upstream features by resolving the order in which coordinates, gradients, and features separate. For matched Host and QR tensors a_{H} and a_{O}, respectively, define the Host-relative distance $d(a_{\mathrm{H}}, a_{\mathrm{O}}) = \|a_{\mathrm{H}} - a_{\mathrm{O}}\|_2 / \|a_{\mathrm{H}}\|_2$. For shared-block gradients, define the symmetric distance $s_\ell = 2\|g_{\mathrm{O},\ell} - g_{\mathrm{H},\ell}\|_2 / (\|g_{\mathrm{O},\ell}\|_2 + \|g_{\mathrm{H},\ell}\|_2) \in [0, 2]$. Zero denominators return zero. The 17 blocks cover attention, mixer, gate, base-kernel, affine, and normalization tensors. A separate probe records the phase state h_j before its FFT and spectral mixing.

Host and QR start from identical shared parameters and receive the same mini-batches and paired random-number-generator state. This trace uses a fixed 30-epoch budget, unclipped AdamW, and zero weight decay. At initialization, predictions and shared gradients agree, while the quadrature-gradient norm is nonzero. After the first update, both sides have moved their shared parameters by the same amount, so those parameters still agree, while the quadrature coordinate has left zero. At the next mini-batch, predictions, shared-parameter gradients, and phase-state gradients differ in all three seeds. Subsequent updates separate the learned features. Omitting global clipping isolates this coordinate-first sequence; the benchmark and full-model rate controls retain unit clipping. Each trace selects its own validation-best checkpoint for Table 13.

The observed order is terminal movement, gradient divergence, then feature divergence. Across seeds 42–44, the first update gives mean $\|W_q\| = 0.105$ while shared parameters still agree. At the next mini-batch, mean Host-relative distances reach 10.4% for shared-parameter gradients and 12.3% for phase-state gradients. Pooling 17 shared blocks across three seeds gives median symmetric gradient distance 0.095, rising to 1.219 by epoch 1. At that epoch, mean shared-parameter and phase-state distances are 0.136 and 0.708. Table 12 shows positive next-batch gradient distances in every seed; Figure 7 shows their distribution across blocks. The initial equality of shared updates follows from equal gradients and matched optimizer states.

Table 12: Per-seed separation order for the synchronized ETTh2 h720 trace. Predictions and shared gradients agree at initialization. The first update moves shared parameters equally in both models and creates a nonzero quadrature coordinate; at the next mini-batch their predictions and gradients differ. Gradient columns are Host-relative distances on that next mini-batch.

Seed	Quadrature grad., init	$\|W_q\|$ after first update	Shared-param grad. d	Phase-state grad. d	Phase-state d, epoch 1	Paired ΔMSE
42	1.235	0.1046	9.8%	12.3%	0.708	−10.48%
43	0.875	0.1046	11.8%	12.9%	0.729	−10.74%
44	0.722	0.1046	9.7%	11.6%	0.687	−10.43%
Mean	0.944	0.1046	10.4%	12.3%	0.708	−10.55%

Table 13: All three synchronized ETTh2 h720 runs retain a 10.55% mean paired MSE reduction after QR initiates the observed gradient divergence. Test MSE comes from validation-best checkpoints.

Seed	Host best epoch	QR best epoch	Host MSE	QR MSE	ΔMSE
42	3	1	0.453615	0.406062	−10.483%
43	2	1	0.451956	0.403435	−10.736%
44	2	1	0.455139	0.407684	−10.427%
Mean	–	–	0.453570	0.405727	−10.548%

D.2 ALTERNATIVE RATE AND CONFIGURATION CHOICES

Rate controls refine the design principle extracted by fixed-bin scaling: which part of the update must change to recover forecasting value? Base-kernel-only controls separate the bin profile from scale

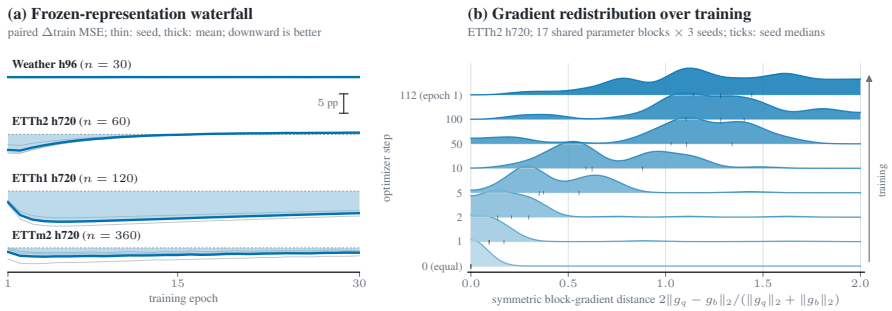


Figure 7: From the terminal update to feature learning. (a) Training MSE of a zero-initialized folded correction and a zero-initialized quadrature correction on the same frozen representation; thin lines are seeds and thick lines their mean. (b) Pooled symmetric gradient distance over the 17 shared parameter blocks: the blocks agree at initialization and separate once W_q has moved under unclipped AdamW. Panel (b) pools blocks; the main-text distances of 10.4% and 12.3% refer to shared-parameter gradients and phase-state gradients, respectively.

magnitude; joint terminal-rate controls change the affected parameter group. Symmetric configuration selection tests how the paired estimate depends on which model selects the training bundle. The terminal-rate experiment uses ETTh2 h720, ETTm2 h720, ECL h96, and Solar h96 at seeds 42–44, with Table 14's settings. Training uses zero-decay AdamW, a constant rate, unit global clipping, at most 30 epochs, and three-epoch patience; validation MSE selects checkpoints. Joint terminal-rate and configuration controls use CUDA; the base-kernel-only controls below share Appendix B.3's CPU setting. Each experiment retains its own paired Host reference.

Table 14: Fixed plain-host settings for the four diagnostic cells, held constant across seeds. Lookback is 720 and seeds are 42–44.

Cell	p	n	Base rate	Batch
ETTh2 h720	12	60	10^{-3}	64
ETTm2 h720	2	360	10^{-4}	32
ECL h96	24	30	10^{-3}	32
Solar h96	24	30	10^{-3}	32

From readout geometry to base-kernel scaling. Exact expresses QR's trajectory in one kernel; fixed-bin extracts its geometric multiplier as a training intervention. The scalar controls ask which part of that intervention carries forecasting value. All modify the base-kernel Adam update alone, keeping the content-conditioned residual, gates, and upstream rates unchanged. They share Appendix B.3's plain-attention architecture, settings, CPU runtime, initialization, and data order. A uniform multiplier $s = 1 + n/2$ preserves fixed-bin's interior scale but replaces its $1 + n$ multiplier at DC and Nyquist.

Validation selection then adjusts the magnitude of this uniform intervention. We search base-kernel multipliers $\{1, 2, 4, 8, 16, n/2, 1 + n/2\}$ using seed-42 validation and fix the selected rate for seeds 43–44. The selected multipliers are 30, 180, 4, and 15 for ETTh2 h720, ETTm2 h720, ECL h96, and Solar h96. The prespecified uniform rule is evaluated across all three seeds independently of selection; held-out test scores do not select rates.

Table 15 follows this progression. QR and Exact yield the same 4.4826% mean paired MSE reduction, establishing the trajectory correspondence. Fixed-bin retains 4.3235% through the extracted multiplier. Replacing its endpoint profile with a uniform scale yields 4.3776%, and validation-selected magnitude yields 4.5700%, improving all four cell means. The uniform result shows that this panel's forecasting gain also survives without the endpoint distinction. Fixed-bin therefore connects the complete optimizer to a useful intervention on the existing kernel, while scalar selection adapts that intervention's strength. Full state mapping establishes trajectory identity; the scale controls identify a reusable source of forecasting value.

Table 15: Base-kernel scale controls: QR and Exact test trajectory correspondence, Fixed-bin applies the geometric multiplier, and Uniform and Tuned isolate the roles of its bin profile and overall magnitude. Test MSE entries are three-seed means on the plain-attention CPU panel. Uniform uses $1 + n/2$; Tuned selects its scalar on seed-42 validation. Both modify the base kernel alone. The last row averages 12 paired relative MSE changes; negative values indicate improvement over Host.

Cell	Host	QR	Exact	Fixed-bin	Uniform	Tuned
ETTh2 h720	0.453488	0.406197	0.406197	0.405802	0.404733	0.404584
ETTm2 h720	0.372987	0.347440	0.347440	0.348948	0.348835	0.348822
ECL h96	0.128256	0.128613	0.128613	0.128586	0.128530	0.127944
Solar h96	0.171378	0.169776	0.169776	0.170356	0.170524	0.170047
Δ (%)	–	-4.4826	-4.4826	-4.3235	-4.3776	-4.5700

Extending the scaled parameter group. Joint terminal-rate controls extend scaling from the base kernel to the content-conditioned residual. We search multipliers $\{1, 2, 4, 8, 16, n/2\}$ for this group, leaving upstream and gate rates unchanged. Seed-42 validation MSE selects the multiplier, with ties favoring the smaller value, and fixes it for seeds 43–44. Default-rate QR uses the base rate for both shared and quadrature parameters. The selected Host multipliers are 16, 1, $n/2$, and 2 in the four-cell order above. Tuning lowers Host MSE by 2.42% over 12 pairs, with four improving pairs and the gain concentrated on ETTh2 h720 (10.79%). Using this tuned Host as the denominator gives a 1.89% mean paired MSE reduction for default-rate QR, with nine improving pairs. The differences occur on ETTm2 h720, ECL h96, and Solar h96, with near parity on ETTh2 h720 (Table 16). Joint terminal scaling also recovers forecasting value, extending the scale intervention to a broader parameter group.

Two further controls reuse a training-calibrated scalar $\beta \leq 1$ in different locations. The fixed-β control multiplies the quadrature contribution by β before the fold. On seed 42's first shuffled training mini-batch, bracketing and bisection select a feasible forward gain under a cap of twice the Host prediction-update RMS, giving $\beta = 0.52, 0.12, 0.96, 1.00$ in cell order. The QR-rate control then uses this same scalar as the quadrature parameter group's learning-rate multiplier, with forward gain one. Both reuse the selected scalar across seeds; calibration uses training data alone. Relative to the tuned Host, the two placements lower MSE by 1.13% and 1.31%, each over eight improving pairs. Both placements of the calibrated scale retain forecasting value.

Table 16: Joint terminal scaling and two placements of a calibrated QR scale on the plain-attention model. The Host control scales both the base kernel and content-conditioned residual, extending the base-only intervention. Seed-42 validation selects among six Host multipliers; raw QR uses default rates. Fixed-β QR uses a training-calibrated forward gain, and the QR-rate control reuses that scalar as its quadrature learning-rate multiplier. Cell entries average paired relative test-MSE changes (%) over three seeds; negative values favor the row's method. Mean and Wins cover all 12 pairs. Denominators are the default-rate Host above and the tuned Host below.

	ETTh2 h720	ETTm2 h720	ECL h96	Solar h96	Mean	Wins
Change from the default-rate Host						
Tuned Host rate	-10.79	$+0.00$	$+0.91$	$+0.18$	-2.42	4/12
QR	-10.43	-6.27	$+0.28$	-0.90	-4.33	10/12
Fixed-β QR	-10.20	-3.48	$+0.27$	-0.90	-3.58	10/12
QR-rate control	-10.25	-4.13	$+0.27$	-0.90	-3.75	10/12
Change from the tuned Host						
QR	$+0.40$	-6.27	-0.63	-1.08	-1.89	9/12
Fixed-β QR	$+0.67$	-3.48	-0.63	-1.08	-1.13	8/12
QR-rate control	$+0.61$	-4.13	-0.63	-1.08	-1.31	8/12

Symmetric configuration selection. A matched comparison fixes the training bundle, but the model used to select that bundle can affect the measured gap. Both selectors receive six identical (period, rate, batch) candidates: $(24, 10^{-3}, 32)$, $(24, 3{\times}10^{-4}, 64)$, $(12, 10^{-3}, 64)$, $(6, 10^{-4}, 64)$, $(2, 10^{-4}, 32)$, and $(12, 3{\times}10^{-4}, 32)$. Each candidate is evaluated at four horizons on ETTh1, ETTh2, ETTm2, and Weather. Mean validation MSE across horizons selects one bundle per dataset and

selector; both bundles are then applied to both models over three seeds. QR lowers mean MSE by 2.91% under Host-selected bundles and 8.27% under QR-selected bundles, winning 38 and 41 of 48 pairs respectively (Table 17). Moving from Host-selected to QR-selected bundles raises Host MSE by 4.38% on average over 48 pairs, with 8 changes negative; QR changes by -1.58%, with 32 negative. The intervention remains useful under either selector, while the bundle changes the magnitude of the paired estimate. These bundles vary period, rate, and batch jointly. The full benchmark uses configurations selected with QR.

Table 17: Symmetric configuration selection (CUDA). Both selectors choose from identical bundles by mean validation MSE, and both selected bundles are applied to both models. Entries are paired QR-versus-Host relative test-MSE changes (%) averaged over four horizons and three seeds per dataset; negative values favor QR. Wins count negative changes among 48 dataset–horizon–seed pairs.

Configuration chosen by	ETTh1	ETTh2	ETTm2	Weather	Mean	Wins
Host validation	-4.34	-6.91	-0.39	$+0.02$	-2.91	38/48
QR validation	-11.47	-14.44	-7.57	$+0.41$	-8.27	41/48

E Forecasting results and transfer

In the PSP forecasting tables below, Host denotes weighted PSP trained with ordinary AdamW, and QR denotes the same backbone with quadrature coordinates, as in Table 1. The branch-free Host/Fixed-bin benchmark has its own runs and selection budgets in Appendix H.

E.1 Adopted configurations and validation-best reporting

Table 18 gives the base period, learning rate η_0 and batch size. Period and rate–batch choices for ETTh1, ETTh2, ETTm2 and Weather were selected separately using QR, seed 42 and validation MSE; the other four datasets use the common default. With period and batch size fixed, each cell evaluates shared-rate multipliers $s \in \{0.5, 1, 2\}$ and QR-only multipliers $q \in \{0.25, 0.5, 1, 2, 4\}$. Seed-42 validation MSE selects one (s, q) from these 15 candidates. Host and the shared tensors of QR use $s\eta_0$; QR's quadrature tensor uses $sq\eta_0$. Table 19 lists the choices, fixed for seeds 42–46. Validation selects checkpoints, and the complete configuration set is fixed before test evaluation.

Table 18: Base configurations before cell-specific rate scaling. Four rows use the default $(p, \mathrm{rate}, \mathrm{batch}) = (24, 10^{-3}, 32)$; QR's validation MSE at seed 42 selects the other rows.

Dataset	p	LR	Batch	Dataset	p	LR	Batch
ETTh1	6	10^{-4}	64	ETTm1	24	10^{-3}	32
ETTh2	12	10^{-3}	64	ECL	24	10^{-3}	32
ETTm2	2	10^{-4}	32	Traffic	24	10^{-3}	32
Weather	24	3×10^{-4}	64	Solar	24	10^{-3}	32

Table 19: Validation-selected shared and QR-only rate multipliers. Each entry is (s, q); the shared rate is $s\eta_0$ and the quadrature rate $sq\eta_0$. Host uses the same shared rate.

Dataset	h96	h192	h336	h720
ETTh1	$(0.5, 1)$	$(0.5, 2)$	$(0.5, 4)$	$(1, 1)$
ETTh2	$(0.5, 1)$	$(0.5, 2)$	$(0.5, 4)$	$(1, 4)$
ETTm1	$(0.5, 0.5)$	$(2, 4)$	$(2, 1)$	$(0.5, 1)$
ETTm2	$(0.5, 1)$	$(0.5, 1)$	$(0.5, 1)$	$(0.5, 1)$
ECL	$(2, 0.25)$	$(0.5, 0.5)$	$(0.5, 0.25)$	$(0.5, 0.25)$
Traffic	$(2, 2)$	$(1, 1)$	$(2, 2)$	$(1, 1)$
Weather	$(1, 1)$	$(2, 0.25)$	$(2, 1)$	$(1, 4)$
Solar	$(1, 1)$	$(1, 0.25)$	$(0.5, 0.25)$	$(1, 1)$

Host and QR cover all four horizons and five seeds. Mechanism controls and the forecast-window diagnostic use their separately specified parameter-group rates; the forecasting grid uses Table 19.

E.2 Complete paired forecasting grid

The horizon-level grid tests whether the dataset averages conceal different effects across forecast lengths. Table 20 reports five-seed means of Δ. ETTh1, ETTh2, ETTm1, ETTm2, ECL, and Traffic improve at all four horizons; Weather and Solar contain the four non-improving cells. The grid therefore locates the broad reduction in six consistent horizon profiles, not in any single forecast length.

Table 20: QR lowers five-seed mean MSE in 28 of 32 dataset–horizon cells. Entries average the five paired relative changes at $L = 720$; negative entries favor QR.

Dataset	h96	h192	h336	h720	mean ↓
ETTh1	-10.84	-13.60	-16.86	-13.56	-13.71
ETTh2	-5.45	-7.98	-8.20	-7.94	-7.39
ETTm1	-0.95	-1.05	-1.42	-1.76	-1.29
ETTm2	-10.66	-11.99	-12.53	-5.59	-10.19
ECL	-0.04	-0.22	-0.17	-0.68	-0.28
Traffic	-1.00	-0.92	-1.86	-0.33	-1.03
Weather	-0.81	$+0.33$	$+0.04$	-0.22	-0.16
Solar	-0.32	-0.20	$+0.26$	$+0.47$	$+0.05$
Mean -4.25%; 124/160 paired changes are negative					

To quantify stability across repeated training, let $\bar{\Delta}_r$ be the mean of Δ over all 32 cells at seed r. Each seed is one complete benchmark repetition, retaining dependencies among cells within that repetition. For seeds 43–46, which did not supply configuration-selection scores, $\bar{\Delta}_r$ is -4.263%, -4.375%, -4.244% and -4.098%. A two-sided one-sample t-test of these four seed means gives $t(3) = -74.54$, $p = 5.32 \times 10^{-6}$ and a 95% interval of $[-4.426, -4.063]\%$. This estimate measures training variation at fixed benchmark data and selected configurations.

Variation across datasets asks a different question from variation across training seeds. An exact Wilcoxon signed-rank test on the eight dataset-mean paired changes gives $W = 1$ and $p = 0.015625$ under a symmetric-difference null. The seed analysis holds benchmark data fixed; the signed-rank analysis assesses the balance of the dataset-level effects.

Separating selection from repeated training. Seed 42 supplies the validation scores for rate selection. Removing its 32 pairs leaves a 4.24% mean reduction and 97 improvements in 128 pairs, compared with 4.25% and 124 of 160 in the complete grid. The dataset-level signed-rank result remains $W = 1$, $p = 0.015625$. The four non-selection seeds used in the fixed-benchmark analysis above thus retain the aggregate effect.

Per-seed view of the grid. Every seed covers all 32 cells. Table 21 reports each dataset's four-horizon mean for every seed. The benchmark-wide mean is negative at all five seeds, ranging from -4.37% to -4.10%, and ETTh1, ETTh2, ETTm1, ETTm2 and Traffic improve at every seed. Including seed 42, the benchmark-level test gives $t(4) = -95.22$ and $p = 7.29 \times 10^{-8}$.

E.3 Absolute accuracy across the benchmark

Table 22 reports absolute MSE and mean absolute error (MAE) for all 32 dataset–horizon cells and their per-dataset averages. Host and QR are five-seed means under the adopted protocol; each external entry averages seeds 42–44 for FITS, PaiFilter, MixLinear, and TexFilter, while SparseTSF reports seed 42. Every score comes from a checkpoint selected by validation MSE. Cross-protocol values are shown without rank formatting, and the 23/32 count in Section 4.2 uses unrounded MSE.

Table 21: Per-seed view of the paired grid. Entries are four-horizon dataset means of Δ, in percent, with negative values favoring QR; the last column repeats the five-seed means of Table 20. The final row counts negative pairs out of each seed's 32 cells.

Dataset	seed 42	seed 43	seed 44	seed 45	seed 46	Mean
ETTh1	-14.10	-13.74	-13.76	-14.41	-12.56	-13.71
ETTh2	-6.74	-7.26	-7.96	-7.12	-7.87	-7.39
ETTm1	-1.39	-0.67	-1.36	-0.95	-2.10	-1.29
ETTm2	-10.88	-10.18	-10.06	-10.32	-9.53	-10.19
ECL	$+0.31$	-0.06	-0.60	-0.51	-0.52	-0.28
Traffic	-0.86	-0.92	-1.22	-1.18	-0.95	-1.03
Weather	-0.33	-0.52	-0.30	$+0.22$	$+0.10$	-0.16
Solar	-0.24	-0.76	$+0.27$	$+0.33$	$+0.65$	$+0.05$
Mean over datasets	-4.28	-4.26	-4.37	-4.24	-4.10	-4.25
Negative pairs	$27/32$	$23/32$	$27/32$	$23/32$	$24/32$	$124/160$

Table 22: QR matches or beats the best external-model MSE in 23 of 32 horizon cells on unrounded values. Host and QR average five seeds; FITS, PaiFilter, MixLinear, and TexFilter average seeds 42–44.

Dataset	H	Host		QR		FITS		PaiFilter		MixLinear		TexFilter		SparseTSF	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.425	0.428	0.379	0.400	0.490	0.480	0.388	0.418	0.361	0.392	0.394	0.424	0.363	0.389
	192	0.481	0.461	0.416	0.424	0.514	0.492	0.428	0.438	0.395	0.407	0.431	0.447	0.401	0.411
	336	0.520	0.485	0.432	0.436	0.528	0.500	0.441	0.448	0.421	0.427	0.454	0.465	0.430	0.426
	720	0.509	0.497	0.440	0.460	0.519	0.513	0.475	0.484	0.425	0.450	0.484	0.495	0.426	0.446
	Avg	0.4838	0.4677	0.4167	0.4300	0.5128	0.4962	0.4330	0.4470	0.4005	0.4190	0.4407	0.4577	0.4050	0.4180
ETTh2	96	0.303	0.364	0.286	0.347	0.286	0.348	0.301	0.361	0.288	0.344	0.327	0.378	0.286	0.342
	192	0.370	0.408	0.340	0.382	0.343	0.382	0.368	0.404	0.340	0.377	0.400	0.420	0.340	0.377
	336	0.404	0.436	0.371	0.413	0.363	0.401	0.409	0.440	0.362	0.399	0.423	0.441	0.360	0.397
	720	0.455	0.470	0.419	0.450	0.387	0.428	0.464	0.478	0.382	0.423	0.442	0.463	0.381	0.423
	Avg	0.3830	0.4195	0.3540	0.3980	0.3448	0.3897	0.3855	0.4208	0.3430	0.3858	0.3980	0.4255	0.3417	0.3848
ETTm1	96	0.302	0.352	0.299	0.350	0.327	0.367	0.304	0.361	0.342	0.375	0.320	0.369	0.339	0.372
	192	0.342	0.376	0.338	0.373	0.351	0.379	0.342	0.381	0.436	0.435	0.363	0.394	0.355	0.379
	336	0.374	0.394	0.368	0.392	0.378	0.395	0.379	0.400	0.376	0.393	0.391	0.412	0.378	0.391
	720	0.431	0.422	0.423	0.419	0.424	0.420	0.450	0.438	0.435	0.426	0.460	0.449	0.422	0.416
	Avg	0.3623	0.3860	0.3570	0.3835	0.3700	0.3902	0.3687	0.3950	0.3972	0.4073	0.3835	0.4060	0.3735	0.3895
ETTm2	96	0.182	0.273	0.162	0.254	0.168	0.259	0.179	0.268	0.186	0.274	0.188	0.278	0.178	0.265
	192	0.246	0.319	0.216	0.291	0.222	0.296	0.244	0.313	0.249	0.322	0.245	0.318	0.228	0.300
	336	0.307	0.359	0.269	0.327	0.272	0.329	0.292	0.345	0.280	0.334	0.297	0.350	0.277	0.331
	720	0.371	0.400	0.351	0.381	0.352	0.380	0.368	0.396	0.363	0.388	0.384	0.405	0.356	0.381
	Avg	0.2765	0.3377	0.2495	0.3132	0.2535	0.3160	0.2707	0.3305	0.2695	0.3295	0.2785	0.3377	0.2598	0.3192
ECL	96	0.128	0.221	0.128	0.221	0.274	0.387	0.131	0.229	0.157	0.252	0.142	0.240	0.138	0.233
	192	0.147	0.240	0.147	0.239	0.286	0.395	0.149	0.245	0.170	0.263	0.168	0.264	0.151	0.245
	336	0.163	0.257	0.163	0.257	0.299	0.403	0.166	0.264	0.186	0.279	0.177	0.276	0.166	0.261
	720	0.201	0.290	0.200	0.288	0.332	0.423	0.204	0.298	0.225	0.312	0.212	0.309	0.221	0.307
	Avg	0.1598	0.2520	0.1595	0.2512	0.2978	0.4020	0.1625	0.2590	0.1845	0.2765	0.1747	0.2722	0.1690	0.2615
Traffic	96	0.366	0.252	0.362	0.249	0.722	0.488	0.376	0.273	0.423	0.292	0.372	0.276	0.390	0.267
	192	0.386	0.264	0.383	0.259	0.734	0.492	0.393	0.282	0.436	0.298	0.384	0.281	0.399	0.271
	336	0.403	0.272	0.396	0.267	0.749	0.497	0.406	0.287	0.455	0.309	0.405	0.291	0.412	0.277
	720	0.437	0.289	0.435	0.288	0.777	0.507	0.442	0.305	0.485	0.324	0.446	0.310	0.448	0.297
	Avg	0.3980	0.2692	0.3940	0.2657	0.7455	0.4960	0.4042	0.2867	0.4497	0.3058	0.4017	0.2895	0.4123	0.2780
Weather	96	0.147	0.197	0.146	0.196	0.173	0.228	0.148	0.202	0.187	0.248	0.154	0.209	0.182	0.238
	192	0.189	0.238	0.190	0.239	0.215	0.263	0.192	0.243	0.227	0.280	0.205	0.256	0.219	0.267
	336	0.240	0.278	0.240	0.279	0.260	0.297	0.243	0.284	0.277	0.316	0.263	0.298	0.264	0.300
	720	0.313	0.330	0.312	0.330	0.321	0.340	0.323	0.339	0.327	0.345	0.337	0.347	0.324	0.343
	Avg	0.2223	0.2608	0.2220	0.2610	0.2423	0.2820	0.2265	0.2670	0.2545	0.2973	0.2398	0.2775	0.2473	0.2870
Solar	96	0.171	0.227	0.170	0.228	0.201	0.253	0.182	0.244	0.332	0.459	0.185	0.249	0.212	0.256
	192	0.189	0.237	0.188	0.239	0.219	0.261	0.201	0.260	0.341	0.442	0.205	0.269	0.225	0.261
	336	0.194	0.242	0.195	0.244	0.235	0.268	0.206	0.268	0.431	0.473	0.216	0.273	0.238	0.265
	720	0.199	0.247	0.200	0.249	0.240	0.271	0.209	0.273	0.242	0.266	0.216	0.278	0.241	0.266
	Avg	0.1883	0.2382	0.1882	0.2400	0.2238	0.2632	0.1995	0.2612	0.3365	0.4100	0.2055	0.2672	0.2290	0.2620

E.4 Quadrature corrections on external forecasters

FITS and PaiFilter. Transfer tests whether the same correction remains useful when another architecture supplies the spectrum. We retain the original prediction paths of FITS and PaiFilter and add a zero-initialized correction to each backbone's own spectral readout, factorized with rank four and trained under that backbone's fixed settings. Table 23 reports the 96 dataset–horizon–

seed pairs per backbone. FITS improves in 78 pairs and PaiFilter in 60, with mean relative MSE changes -0.191% and -0.184%. Seven of eight dataset means and all three seed-aggregated means improve for each backbone. The 95% t intervals over eight dataset means are $[-0.360, -0.023]\%$ and $[-0.360, -0.007]\%$. ECL, Traffic, Weather, and Solar contribute 44/48 FITS wins and 32/48 PaiFilter wins, with all eight backbone–dataset averages improving. The operators below specify what is transferred in each architecture.

Table 23: Zero-initialized corrections added to FITS and PaiFilter, seeds 42–44. Entries are mean paired relative test-MSE changes (%) over each dataset's 12 cell–seed pairs, negative favoring the correction; wins count negative pairs. Each host contributes 96 dataset–horizon–seed pairs, with 12 paired runs for each dataset.

	FITS		PaiFilter	
Dataset	Δ (%)	Wins	Δ (%)	Wins
ETTh1	-0.603	12/12	-0.120	6/12
ETTh2	-0.061	7/12	-0.035	7/12
ETTm1	-0.107	8/12	-0.148	8/12
ETTm2	$+0.030$	7/12	$+0.040$	7/12
ECL	-0.131	12/12	-0.158	8/12
Traffic	-0.267	12/12	-0.121	9/12
Weather	-0.069	9/12	-0.275	6/12
Solar	-0.323	11/12	-0.653	9/12
All 96 pairs	-0.191	78/96	-0.184	60/96

Two-sided 95% t interval over the eight dataset means: FITS $[-0.360, -0.023]\%$, PaiFilter $[-0.360, -0.007]\%$.

Both corrections are stored as rank-four complex factors $U \in \mathbb{C}^{H\times 4}$ and $V \in \mathbb{C}^{F\times 4}$, where F is the number of host-spectrum bins. Small-random U and zero V make the added forecast initially zero. For FITS, let $\widehat{Y}$ be the one-sided, zero-padded upsampled spectrum, $T = L+H$ the inverse-transform length, and $\gamma = T/L$. For rank component r and zero-based forecast step t,

$$a_r^t = \left[\mathrm{ifft}\Big(\widehat{Y} \odot V_{\cdot,r}, n = T\Big)\right]_{T-H+t}, \qquad y_q^t = \gamma\,\mathrm{Im}\left(\sum_r U^{t,r} a_r^t\right). \tag{24}$$

This complex inverse FFT retains its $1/T$ normalization.

PaiFilter instead uses orthonormal transforms. Let $\mathcal{F}_o$ denote that rFFT and u the normalized input. Its main path sends $v = \mathcal{F}_o^{-1}(\mathcal{F}_o(u) \odot \mathcal{F}_o(w))$ to the forecasting MLP. For the auxiliary correction, we apply the same filter to v again, forming $\widehat{Y} = \mathcal{F}_o(v) \odot \mathcal{F}_o(w)$. The correction therefore acts on the twice-filtered spectrum and contracts frequency before time:

$$a_r = \sum_b \widehat{Y}_b V^{b,r}, \qquad y_q^t = \mathrm{Im}\left(\sum_r U^{t,r} a_r\right). \tag{25}$$

Both branches add y_q^t before reversing normalization, preserving initial predictions through zero initialization. Their normalization and parameter sharing follow their respective backbones.

FITS and PaiFilter use rank-four factorized corrections initialized to produce zero output, preserving the original forecasts at initialization. These paired experiments test the usefulness of spectral readout corrections under each backbone's normalization and parameter sharing.

MixLinear. MixLinear (Ma et al., 2026) supplies a different capacity-preserving readout. Let h be its existing rank-two complex latent and W_F its final complex frequency-map weight. We add the same correction, $\mathrm{Im}(Qh)$, after the real inverse FFT, inside the unchanged frequency-path mixture weight $1-\alpha$. With $Q = 0$, the starting prediction is unchanged. Folding uses

$$\widetilde{W}_F = W_F - i\,\mathrm{FFT}_{\mathrm{out}}(Q)$$

to absorb the coordinate into the original weight. The FFT acts along the output-frequency dimension, so its inverse contributes $-iQh$ and its real part equals $\mathrm{Im}(Qh)$. The time branch, biases, latent rank, and folded parameter count remain unchanged.

We use period 24, four retained frequency bins, rank two, $\alpha = 0.9$, lookback 720, and batch size 32. The original normalization and both prediction paths are retained. Training uses AdamW at base rate 10^{-3}, zero decay, unit global clipping, a constant schedule, at most 30 epochs, and three-epoch validation patience. ETTh2 h720, ETTm2 h720, ECL h96, and Solar h96 form a four-cell diagnostic panel with seeds 42–44. For each cell, seed-42 validation selects the QR-only rate multiplier from $\{0.25, 1, 4\}$ and the original output-weight multiplier from $\{1, 4, 16\}$ separately. Their winners are $(1, 4, 4, 1)$ and $(16, 16, 16, 4)$ in cell order. All other rates remain at the base value; selected configurations are fixed before the remaining seeds and test evaluation. Paired arms and seeds share A100-40GB for ETTh2 and ETTm2, or A100-80GB for ECL and Solar.

Across 12 cell–seed pairs, QR lowers MSE by 0.184% relative to default-rate MixLinear and by 0.014% relative to the terminal-rate-selected Host (Table 24). The complete panel includes improvements on ETTh2 and ECL and near-parity cell means on ETTm2 and Solar. This comparison extends the intervention to an existing time–frequency mixture: its forecasting benefit is realized during training, while the added coordinate is removed by folding.

Table 24: MixLinear transfer on four diagnostic cells, seeds 42–44. Host denotes default-rate MixLinear. Absolute test MSE entries are seed means. QR and Tuned Host use separately validation-selected coordinate and output-weight rates. Relative changes average paired seed-level MSE changes; negative values favor QR. All four cells enter the panel mean.

Cell	Host	Tuned Host	QR	Δ_{Host} (%)	Δ_{Tuned} (%)
ETTh2 h720	0.382476	0.381157	0.380690	-0.467	-0.123
ETTm2 h720	0.362233	0.361911	0.362573	$+0.094$	$+0.183$
ECL h96	0.156752	0.156273	0.156085	-0.386	-0.120
Solar h96	0.331862	0.331933	0.331940	$+0.023$	$+0.002$
All 12 pairs	–	–	–	-0.184	-0.014

E.5 Illustrative window and complete per-lead profile

A time-domain view connects aggregate MSE to the forecast changes that produce it. Figure 8 pairs a selected ETTh2 example with the complete-test per-lead profile at $H = 720$, seed 42. Host and QR use validation-best checkpoints and Table 18's base rates with $s = q = 1$. Among 2,161 test windows, zero-based window 440 has the largest Host-minus-QR MSE gap on the oil-temperature (OT) channel, changing from 0.513157 to 0.309049. Its largest-gap contiguous 96-step interval spans steps 580–675 and changes from 1.503532 to 0.836520. The complete-test profile averages every window and channel at each forecast lead; QR has lower MSE at 719 of 720 leads.

The selected window makes the forecast difference visible, while the complete-test profile shows that lower per-lead error extends across the horizon. Together they connect the benchmark MSE comparison to the prediction-space comparison in Figure 1a.

F Training and deployment cost

F.1 Parameter count and computational complexity

The optimization intervention adds training coordinates without requiring an additional deployment branch. The cached fold in Appendix A.2 removes the quadrature tensor while preserving the trained forecast. Counting allocated parameters and contractions makes this distinction explicit: training stores and updates QR, whereas inference uses its contribution through the folded kernel.

Closed-form parameter count. Let $n = \lceil L/p \rceil$ be the phase-subsequence length after zero-padding the temporal axis to a multiple of p, and $F = \lfloor n/2 \rfloor + 1$ the number of rFFT bins. Each phase position holds $S_H = \lceil H/p \rceil$ future steps, the allocated phase-position parameter groups number $K = 3$, and the phase-attention width is $d = 8$. Phase position j uses group

$$k(j) = \min\left(\left\lfloor \frac{j}{\max(1, \lfloor p/K \rfloor)} \right\rfloor, K - 1 \right), \tag{26}$$

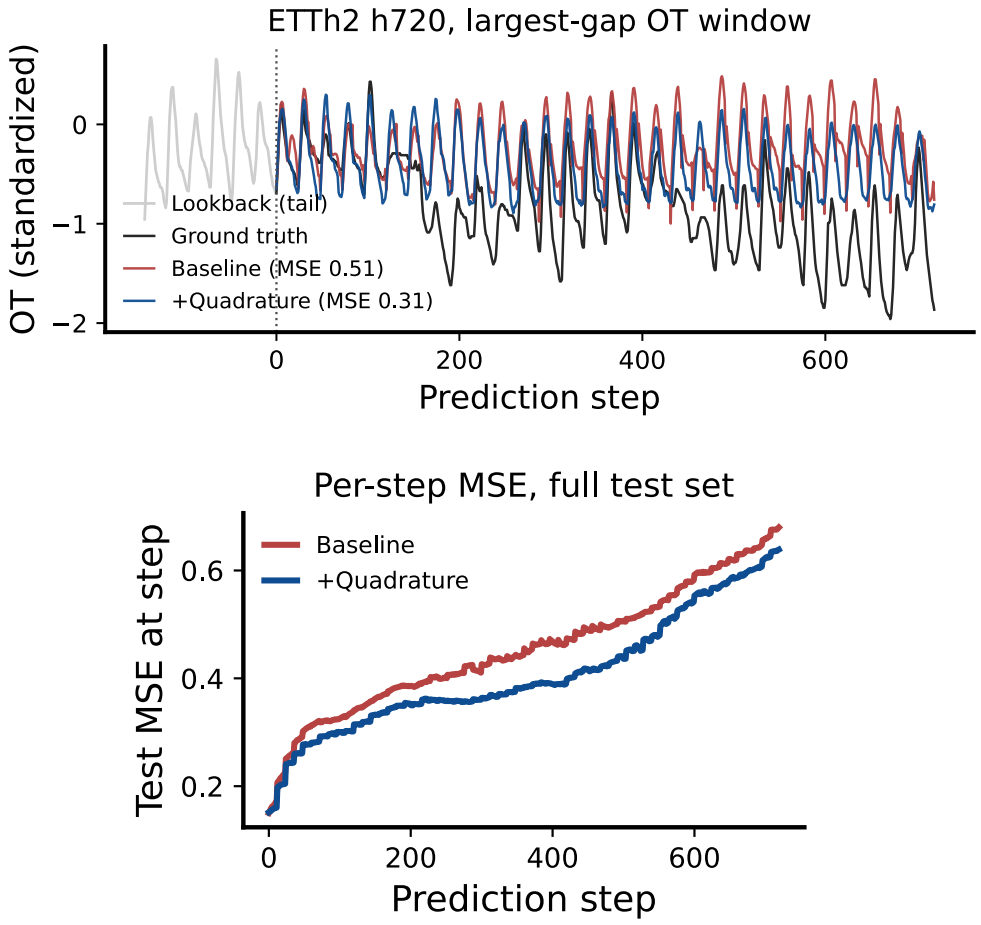


Figure 8: At default parameter-group rates, QR lowers per-lead test MSE at 719 of 720 ETTh2 leads for $H = 720$, seed 42. Top: the selected largest-gap OT-channel window changes from Host/QR MSE $0.513/0.309$. Bottom: the complete-test profile, which averages every test window and channel at each of the 720 leads, with the largest separation in the horizon's second half. The count is over lead positions, not over individual predicted points.

so the p phase positions map contiguously into the first $\min(p, K)$ allocated groups. The ceiling convention covers padded diagnostics; adopted benchmark periods divide L and H exactly. The encoder's real-scalar count includes every allocated trainable tensor, including unused groups when $p < K$:

$$P_{\text{host}}^{\text{enc}} = (8 + 2K)F^2 + 4S_H KF + 2Fd + pd + 3d^2 + dn + 1 + 2N, \tag{27}$$

The terms are, in order, the cross-bin mixer $(2 \times (2F)^2$ real weights), two real gate tensors of shape $K \times F \times F$ $(2KF^2)$, and two complex kernels of shape $S_H \times K \times F$ (W_{base} and ΔW, $4S_H KF$ real scalars in total). Phase attention contributes $2Fd$ embedding, pd positional, $3d^2$ query/key/value and dn output-projection scalars, plus one mixer-blend scalar and internal per-channel scale/bias $(2N)$. Outer instance normalization adds a separate $2N$ common to every model, giving $P_{\text{host}} = P_{\text{host}}^{\text{enc}} + 2N$. QR adds one complex tensor $W_q \in \mathbb{C}^{S_H \times K \times F}$ with the same allocation as W_{base}:

$$\Delta P_q = S_H KF \text{ complex entries} = 2S_H KF \text{ real scalars} = O\left(\frac{KHL}{p^2}\right). \tag{28}$$

We count each complex64 entry as two real scalars; PyTorch's `numel()` instead counts one complex entry. At Weather h96, $n = 30$, $F = 16$, and $S_H = 4$, so Equation (28) gives 192 complex entries, or 384 real scalars. This is 7.2% of the 5,317-scalar full host. At ETTm2 h720, $n = 360$, $F = 181$, and $S_H = 360$, giving 195,480 complex entries, or 390,960 real scalars, which is 31.4% of the

1,246,587-scalar full host. Across all 32 cells, the real-scalar overhead ranges from 4.4% to 31.4%; under the complex-entry `numel()` convention, it ranges from 2.3% to 22.8%.

Time complexity. The current implementation evaluates the cross-bin map and magnitude-gate contraction once per forward pass over the whole batch, at $O(BNpF^2)$ time, and applies the step-dependent multiplier with one batched contraction, at $O(BNpS_HF) = O(BNHF)$ time. The QR branch adds another contraction of this order and therefore introduces no new asymptotic term. Computing the single-index readout through Equation (1) costs $O(F)$ per lead and variable, while the phase-subsequence rFFT costs $O(BNp\,n\log n)$ per forward pass over the batch. Phase attention with width d adds $O(BNpFd + BNpd^2 + BNp^2d + BNpdn)$ time and $O(BNp^2)$ attention memory. Under the adopted protocol, $d = 8$ and $p \leq 24$. Cached folding removes the QR contraction while retaining the trained forecast, as shown in Appendix A.2.

G Relation to prior interventions

Prior interventions show that training coordinates can encode useful structure even when their branches are later merged. Table 25 distinguishes changing a basis, merging temporal kernels, and expressing structural priors as optimizer multipliers. Our spectral fold makes the coordinate scale explicit; mapped moments, finite-ϵ dependence, and the augmented clipping norm extend it to an exact Adam update. This finite-step correspondence addresses a different object from learning-rate and epsilon scaling for width transfer (Everett et al., 2024), as discussed in Section 2.

Table 25: From training-coordinate interventions to explicit optimizer rules.

Approach	Training intervention	Studied effect
Spectral filters (Rippel et al., 2015)	Fourier coordinates for convolutional filters	Adaptive updates in a spectral basis
Fourier implicit networks (Shi et al., 2024)	Fixed Fourier bases for multilayer-perceptron weights	Reduced low-frequency training bias
RepOptimizers (Ding et al., 2023)	Gradient multipliers matching linear-branch updates	Structural priors in optimizer updates
ModernTCN (Luo & Wang, 2024)	Large and small temporal kernels, merged at inference	Efficient temporal feature learning
QR and folded rules	Quadrature coordinates and their single-kernel optimizer	Exact updates, closure and branch-free scaling

H A BRANCH-FREE TRAINING RULE ACROSS THE FULL BENCHMARK

Fixed-bin scaling turns the geometry of the spectral readout into an intervention on the existing base kernel. The plain-attention controls in Appendix D.2 establish its connection to QR and separate the bin profile from its overall scale. We evaluate Host and Fixed-bin on all 32 cells of the weighted-PSP benchmark, extending the training rule to kernel-conditioned attention. The comparisons distinguish three questions: whether the prescribed rule improves the adopted training recipe, whether its effect persists after independent global-rate selection, and how it compares with a broader search over base-kernel rates. Together they test the extracted rule beyond the exact-reproduction architecture.

H.1 RULE, CONFIGURATIONS, AND SELECTION BUDGETS

Every arm retains the full weighted PSP Host of Appendix A, including its magnitude-conditioned attention, content-conditioned residual, gates, and upstream parameters. Host uses ordinary AdamW; Fixed-bin changes the base-kernel update without adding quadrature coordinates. Let $\delta W_{t,b}^{\mathrm{Adam}}$ denote the ordinary zero-decay Adam update after unit global clipping. Fixed-bin applies

$$W_{t+1,b} = W_{t,b} + (1 + a_b)\delta W_{t,b}^{\mathrm{Adam}}, \qquad a_b = \frac{n}{c_b}, \tag{29}$$

where $c_b = 2$ for conjugate-paired bins and $c_b = 1$ at DC and, when n is even, Nyquist. Thus the multipliers are $1 + n/2$ and $1 + n$, respectively. The rule scales the normalized parameter update, with no extra gradient scaling or optimizer moments. The residual and upstream parameters use their ordinary updates at the adopted global rate. Section 3.4 derives this practical intervention. Whole-model closure is required for the exact QR trajectory construction.

The benchmark uses the eight datasets and chronological splits of Section 4.1, lookback $L = 720$, horizons $H \in \{96, 192, 336, 720\}$, and seeds 42–46. Table 26 lists the dataset-specific constants, shared by all horizons and comparison arms. Training uses AdamW with zero weight decay, constant rates, unit global clipping, dropout 0.1, at most 30 epochs, and three-epoch patience. Each run reports the test score of its validation-best checkpoint.

Table 26: Shared configurations for the weighted-host Fixed-bin benchmark. The base rate η_0, period p, and batch size are fixed across horizons and methods. With $n = 720/p$, $s = 1 + n/2$ gives the prescribed interior-bin multiplier and centers the scalar search for Strong Host.

Dataset	p	n	η_0	Batch size	s
ETTh1	6	120	10^{-4}	64	61
ETTh2	12	60	10^{-3}	64	31
ETTm1	24	30	10^{-3}	32	16
ETTm2	2	360	10^{-4}	32	181
ECL	24	30	10^{-3}	32	16
Traffic	24	30	10^{-3}	32	16
Weather	24	30	3×10^{-4}	64	16
Solar	24	30	10^{-3}	32	16

The default comparison fixes the global multiplier at $g = 1$ for both Host and Fixed-bin. The equal-budget comparison independently selects each method's $g \in \{0.5, 1, 2\}$ using seed-42 validation MSE, so both methods receive three candidates per cell. Strong Host jointly selects the global and base-kernel multipliers

$$b \in \{1, s/4, s/2, s, 2s\}, \qquad s = 1 + n/2.$$

It therefore receives 15 candidates per cell, versus three for Fixed-bin. Its base kernel uses rate $gb\eta_0$ and all other parameters use $g\eta_0$. This control tests whether selecting a uniform base-kernel update magnitude recovers the benefit of the prescribed bin scales. Settings remain fixed for all five seeds.

Hyperparameters are selected using validation scores alone, and test scores are computed at the validation-best checkpoint of each run. Seeds 43–46 exclude the hyperparameter-selection seed.

H.2 FORECASTING EFFECT AND THE SCALE INTERPRETATION

To give every dataset–horizon cell equal weight, we first average each method's test error over seeds and then compute its relative change. For comparison methods A and B, define

$$\overline{m}_c^A = \frac{1}{|S|}\sum_{r\in S} m_{c,r}^A, \qquad \Delta_{\text{cell}}(A,B) = \frac{100}{32}\sum_{c=1}^{32}\frac{\overline{m}_c^A - \overline{m}_c^B}{\overline{m}_c^B}. \tag{30}$$

Here $S = \{42,\dots,46\}$ unless stated otherwise, and m is test MSE or MAE. Negative changes favor Fixed-bin. This cell-mean reduction is distinct from averaging the individual seed-pair changes in Equation (4.1). Both are reported below, with denominators from the stated comparison.

Table 27 shows that the prescribed rule lowers mean cell-relative MSE by 3.573% at default rates and by 3.158% under independent, equal-budget global-rate selection. Both comparisons improve 27 of 32 cell means. MAE decreases by 2.264% and 2.079%, respectively. Thus a fixed base-kernel rule improves the adopted recipe, and its effect persists when the competing Host receives its own global-rate selection.

Table 27: Fixed-bin scaling improves the full weighted benchmark under default and equal-budget rates, and attains near-zero aggregate MSE difference from the larger-search Host. Each change is relative to the comparison's own Host denominator and averages 32 cell-level relative changes. Negative values favor Fixed-bin. Wins count cells with lower unrounded mean MSE. Seeds 43–46 exclude the validation-selection seed.

Comparison	Δ MSE (%)	Wins	Δ MAE (%)	Δ MSE, 43–46	Wins
Default	−3.573	27/32	−2.264	−3.587	26/32
Equal-budget	−3.158	27/32	−2.079	−3.222	26/32
Strong Host	−0.011	17/32	+0.072	−0.091	15/32

The larger base-kernel search brings Strong Host to nearly the same aggregate accuracy: the mean MSE change is -0.011%, with Fixed-bin improving 17 of 32 cell means, while the mean MAE change is $+0.072\%$. This comparison identifies the value of the geometric prescription as obtaining comparable aggregate accuracy with three global-rate candidates instead of a 15-candidate joint search. The result is consistent with the scalar controls in Appendix D.2: changing base-kernel update scale carries forecasting value, while its best magnitude can also be found by direct validation search.

Table 28 resolves this aggregate by dataset. Fixed-bin improves all four horizons on ETTh1, ETTh2, ETTm1, ETTm2, and Traffic under both default and equal-budget comparisons. The largest average reductions occur on the ETT datasets. ECL, Weather, and Solar have smaller, mixed changes.

Table 28: Dataset-level view of the three MSE comparisons. Each dataset entry averages its four horizon-level relative changes after averaging five seeds within each method and horizon. Wins count improving horizon means. All eight datasets and four horizons contribute to the final row.

	Default		Equal-budget		Strong Host	
Dataset	Δ (%)	Wins	Δ (%)	Wins	Δ (%)	Wins
ETTh1	−11.441	4/4	−9.027	4/4	+0.101	2/4
ETTh2	−7.685	4/4	−7.675	4/4	+0.351	0/4
ETTm1	−0.710	4/4	−1.086	4/4	−0.548	4/4
ETTm2	−7.892	4/4	−6.701	4/4	+0.098	1/4
ECL	−0.019	2/4	+0.109	2/4	+0.132	2/4
Traffic	−1.035	4/4	−1.144	4/4	−0.562	3/4
Weather	−0.229	3/4	−0.025	3/4	−0.017	3/4
Solar	+0.426	2/4	+0.282	2/4	+0.353	2/4
Mean / total	−3.573	27/32	−3.158	27/32	−0.011	17/32

Removing seed 42 leaves mean cell-relative MSE reductions of 3.587% at default rates and 3.222% under equal-budget selection, with 26 of 32 improving cells in each comparison. Against Strong

Host, the corresponding aggregate change is -0.091%, with 15 of 32 improving cells. The default and equal-budget effects therefore persist in the four seeds that did not select the rates. Under the individual seed-pair convention of Equation (4.1), the five-seed mean changes are -3.564%, -3.146%, and -0.008%, with 121, 119, and 79 improvements out of 160 pairs, respectively. The Strong Host comparison therefore indicates aggregate parity with cell-specific variation.

H.3 Complete test accuracy

Tables 29 and 30 report all 32 cells for the five arms. Standard deviations describe training-seed variation within a cell. The separate default and independently selected columns distinguish a fixed-recipe intervention from a comparison of validation-selected models. Strong Host searches 15 global and base-kernel combinations.

Table 29: Complete weighted-host test MSE for the branch-free benchmark. Entries are mean (sample standard deviation) over seeds 42–46. Lower is better. Default columns use $g = 1$. Selected Host and Fixed-bin each search three global rates. Strong Host searches 15 global/base-kernel combinations. Bold compares Host and Fixed-bin within each of the two matched-budget column pairs, using unrounded means. The larger-budget Strong Host is shown separately.

		Default rates		Three-rate selection		15 settings
Dataset	H	Host	Fixed-bin	Host	Fixed-bin	Strong Host
ETTh1	96	0.4023 (0.0039)	**0.3785** (0.0009)	0.3939 (0.0044)	**0.3785** (0.0009)	0.3797 (0.0014)
	192	0.4666 (0.0128)	**0.4122** (0.0017)	0.4522 (0.0136)	**0.4128** (0.0040)	0.4112 (0.0014)
	336	0.5121 (0.0166)	**0.4389** (0.0053)	0.4889 (0.0180)	**0.4420** (0.0027)	0.4396 (0.0024)
	720	0.5089 (0.0058)	**0.4381** (0.0054)	0.5089 (0.0058)	**0.4381** (0.0054)	0.4392 (0.0057)
ETTh2	96	0.2934 (0.0031)	**0.2840** (0.0014)	0.2944 (0.0102)	**0.2847** (0.0015)	0.2847 (0.0015)
	192	0.3682 (0.0040)	**0.3392** (0.0017)	0.3642 (0.0069)	**0.3395** (0.0024)	0.3386 (0.0025)
	336	0.3995 (0.0038)	**0.3639** (0.0023)	0.3994 (0.0014)	**0.3639** (0.0023)	0.3639 (0.0024)
	720	0.4551 (0.0027)	**0.4061** (0.0054)	0.4577 (0.0064)	**0.4039** (0.0055)	0.3996 (0.0104)
ETTm1	96	0.2999 (0.0021)	**0.2997** (0.0019)	0.3020 (0.0021)	**0.3012** (0.0034)	0.3015 (0.0025)
	192	0.3424 (0.0023)	**0.3401** (0.0020)	0.3420 (0.0012)	**0.3374** (0.0008)	0.3420 (0.0012)
	336	0.3724 (0.0013)	**0.3717** (0.0055)	0.3724 (0.0013)	**0.3707** (0.0012)	0.3724 (0.0013)
	720	0.4305 (0.0053)	**0.4225** (0.0031)	0.4318 (0.0040)	**0.4219** (0.0028)	0.4232 (0.0025)
ETTm2	96	0.1791 (0.0018)	**0.1660** (0.0012)	0.1746 (0.0013)	**0.1642** (0.0015)	0.1642 (0.0015)
	192	0.2422 (0.0022)	**0.2203** (0.0009)	0.2359 (0.0014)	**0.2185** (0.0008)	0.2182 (0.0008)
	336	0.3029 (0.0048)	**0.2722** (0.0007)	0.2972 (0.0058)	**0.2713** (0.0010)	0.2707 (0.0013)
	720	0.3713 (0.0040)	**0.3522** (0.0006)	0.3713 (0.0040)	**0.3537** (0.0009)	0.3534 (0.0007)
ECL	96	**0.1283** (0.0004)	0.1288 (0.0003)	**0.1281** (0.0005)	0.1289 (0.0004)	0.1278 (0.0008)
	192	0.1473 (0.0008)	**0.1468** (0.0007)	0.1473 (0.0008)	**0.1467** (0.0005)	0.1466 (0.0005)
	336	0.1640 (0.0016)	**0.1636** (0.0017)	0.1640 (0.0016)	**0.1636** (0.0017)	0.1640 (0.0016)
	720	**0.2006** (0.0008)	0.2009 (0.0019)	**0.2000** (0.0012)	0.2009 (0.0019)	0.2013 (0.0023)
Traffic	96	0.3682 (0.0047)	**0.3657** (0.0031)	0.3657 (0.0045)	**0.3616** (0.0021)	0.3640 (0.0054)
	192	0.3863 (0.0025)	**0.3812** (0.0024)	0.3859 (0.0031)	**0.3799** (0.0039)	0.3783 (0.0048)
	336	0.4045 (0.0016)	**0.3978** (0.0019)	0.4030 (0.0045)	**0.3956** (0.0044)	0.4030 (0.0045)
	720	0.4369 (0.0019)	**0.4348** (0.0013)	0.4355 (0.0008)	**0.4352** (0.0022)	0.4359 (0.0014)
Weather	96	**0.1467** (0.0010)	0.1473 (0.0014)	0.1467 (0.0010)	**0.1465** (0.0009)	0.1450 (0.0009)
	192	0.1889 (0.0013)	**0.1889** (0.0008)	0.1889 (0.0006)	**0.1889** (0.0008)	0.1893 (0.0009)
	336	0.2397 (0.0015)	**0.2379** (0.0015)	0.2397 (0.0027)	**0.2392** (0.0019)	0.2397 (0.0010)
	720	0.3129 (0.0029)	**0.3110** (0.0010)	**0.3103** (0.0012)	0.3110 (0.0010)	0.3132 (0.0028)
Solar	96	0.1708 (0.0024)	**0.1706** (0.0022)	0.1717 (0.0011)	**0.1704** (0.0016)	0.1717 (0.0011)
	192	0.1886 (0.0012)	**0.1876** (0.0010)	0.1880 (0.0014)	**0.1873** (0.0011)	0.1874 (0.0013)
	336	**0.1941** (0.0011)	0.1979 (0.0014)	**0.1944** (0.0017)	0.1979 (0.0014)	0.1944 (0.0006)
	720	**0.1989** (0.0008)	0.1997 (0.0008)	**0.1989** (0.0008)	0.1997 (0.0008)	0.1989 (0.0008)

Table 30: Complete weighted-host test MAE for the branch-free benchmark. Entries are mean (sample standard deviation) over seeds 42–46. Lower is better. Default columns use $g = 1$. Selected Host and Fixed-bin each search three global rates. Strong Host searches 15 global/base-kernel combinations. Bold compares Host and Fixed-bin within each of the two matched-budget column pairs, using unrounded means. The larger-budget Strong Host is shown separately.

		Default rates		Three-rate selection		15 settings
Dataset	H	Host	Fixed-bin	Host	Fixed-bin	Strong Host
ETTh1	96	0.4170 (0.0022)	**0.4002** (0.0004)	0.4127 (0.0018)	**0.4002** (0.0004)	0.4004 (0.0008)
	192	0.4529 (0.0068)	**0.4230** (0.0015)	0.4456 (0.0067)	**0.4234** (0.0030)	0.4216 (0.0014)
	336	0.4811 (0.0080)	**0.4413** (0.0036)	0.4685 (0.0090)	**0.4434** (0.0021)	0.4411 (0.0016)
	720	0.4969 (0.0032)	**0.4588** (0.0028)	0.4969 (0.0032)	**0.4588** (0.0028)	0.4595 (0.0034)
ETTh2	96	0.3584 (0.0019)	**0.3458** (0.0008)	0.3581 (0.0056)	**0.3462** (0.0017)	0.3461 (0.0018)
	192	0.4074 (0.0020)	**0.3826** (0.0006)	0.4049 (0.0026)	**0.3839** (0.0006)	0.3801 (0.0021)
	336	0.4333 (0.0014)	**0.4062** (0.0015)	0.4330 (0.0009)	**0.4062** (0.0015)	0.4061 (0.0016)
	720	0.4705 (0.0017)	**0.4410** (0.0038)	0.4720 (0.0023)	**0.4392** (0.0039)	0.4360 (0.0063)
ETTm1	96	0.3509 (0.0008)	**0.3508** (0.0009)	0.3518 (0.0013)	**0.3506** (0.0020)	0.3507 (0.0020)
	192	0.3748 (0.0011)	**0.3731** (0.0014)	0.3749 (0.0015)	**0.3709** (0.0013)	0.3749 (0.0015)
	336	0.3934 (0.0017)	**0.3911** (0.0025)	0.3934 (0.0017)	**0.3904** (0.0013)	0.3934 (0.0017)
	720	0.4221 (0.0027)	**0.4177** (0.0014)	0.4240 (0.0016)	**0.4176** (0.0019)	0.4171 (0.0011)
ETTm2	96	0.2704 (0.0014)	**0.2594** (0.0011)	0.2680 (0.0011)	**0.2558** (0.0020)	0.2559 (0.0020)
	192	0.3165 (0.0016)	**0.2959** (0.0016)	0.3117 (0.0024)	**0.2932** (0.0011)	0.2927 (0.0013)
	336	0.3559 (0.0040)	**0.3322** (0.0011)	0.3527 (0.0022)	**0.3295** (0.0011)	0.3287 (0.0012)
	720	0.3976 (0.0039)	**0.3861** (0.0023)	0.3976 (0.0039)	**0.3841** (0.0009)	0.3837 (0.0009)
ECL	96	**0.2216** (0.0008)	0.2220 (0.0006)	**0.2212** (0.0007)	0.2220 (0.0005)	0.2206 (0.0008)
	192	0.2399 (0.0007)	**0.2385** (0.0006)	0.2399 (0.0007)	**0.2385** (0.0007)	0.2381 (0.0006)
	336	0.2580 (0.0013)	**0.2570** (0.0023)	0.2580 (0.0013)	**0.2570** (0.0023)	0.2580 (0.0013)
	720	**0.2890** (0.0004)	0.2895 (0.0015)	**0.2890** (0.0014)	0.2895 (0.0015)	0.2889 (0.0017)
Traffic	96	0.2536 (0.0034)	**0.2515** (0.0020)	0.2524 (0.0031)	**0.2488** (0.0019)	0.2507 (0.0037)
	192	0.2637 (0.0025)	**0.2591** (0.0013)	0.2625 (0.0025)	**0.2585** (0.0022)	0.2570 (0.0029)
	336	0.2734 (0.0021)	**0.2673** (0.0014)	0.2723 (0.0034)	**0.2665** (0.0020)	0.2723 (0.0034)
	720	0.2891 (0.0016)	**0.2877** (0.0010)	**0.2872** (0.0012)	0.2886 (0.0015)	0.2882 (0.0013)
Weather	96	**0.1971** (0.0013)	0.1986 (0.0016)	**0.1971** (0.0013)	0.1977 (0.0012)	0.1962 (0.0014)
	192	0.2381 (0.0011)	**0.2379** (0.0005)	0.2382 (0.0011)	**0.2379** (0.0005)	0.2388 (0.0010)
	336	0.2781 (0.0017)	**0.2774** (0.0014)	**0.2779** (0.0022)	0.2785 (0.0019)	0.2794 (0.0006)
	720	0.3300 (0.0017)	**0.3296** (0.0010)	**0.3282** (0.0010)	0.3296 (0.0010)	0.3310 (0.0011)
Solar	96	**0.2269** (0.0023)	0.2277 (0.0027)	**0.2268** (0.0021)	0.2281 (0.0020)	0.2268 (0.0021)
	192	**0.2368** (0.0028)	0.2401 (0.0013)	**0.2363** (0.0019)	0.2393 (0.0010)	0.2389 (0.0016)
	336	**0.2430** (0.0013)	0.2463 (0.0004)	**0.2423** (0.0011)	0.2463 (0.0004)	0.2432 (0.0007)
	720	**0.2470** (0.0009)	0.2494 (0.0018)	**0.2470** (0.0009)	0.2494 (0.0018)	0.2470 (0.0009)

H.4 SELECTED RATES FOR REPRODUCTION

Table 31 gives every selected global and base-kernel multiplier. The selection criterion is minimum seed-42 validation MSE, with exact ties resolved by a fixed candidate order. Default rates remain $g = 1$. Strong Host includes $b = 1$, so it contains the three-rate Host candidates as a subset.

Table 31: Validation-selected multipliers for every cell. The global multiplier g scales the adopted dataset rate η_0. Strong Host additionally scales the base-kernel rate by b, giving $gb\eta_0$ for that kernel and $g\eta_0$ elsewhere. Fixed-bin applies its prescribed bin multiplier after the globally scaled Adam step. All choices use seed-42 validation and remain fixed for seeds 43–46.

Dataset	H	Host g	Fixed-bin g	Strong Host g	Strong Host b
ETTh1	96	2	1	0.5	122
	192	2	0.5	0.5	122
	336	2	2	1	122
	720	1	1	1	61
ETTh2	96	2	0.5	0.5	31
	192	2	0.5	0.5	62
	336	2	1	1	31
	720	0.5	0.5	0.5	62
ETTm1	96	0.5	0.5	0.5	8
	192	0.5	0.5	0.5	1
	336	1	0.5	1	1
	720	2	0.5	0.5	16
ETTm2	96	2	0.5	0.5	181
	192	2	0.5	0.5	362
	336	2	0.5	0.5	362
	720	1	0.5	0.5	181
ECL	96	2	0.5	2	4
	192	1	0.5	1	8
	336	1	1	1	1
	720	2	1	0.5	8
Traffic	96	2	2	2	8
	192	2	2	2	16
	336	2	2	2	1
	720	2	2	0.5	8
Weather	96	1	2	2	8
	192	2	1	2	4
	336	2	2	2	32
	720	0.5	1	2	32
Solar	96	0.5	0.5	0.5	1
	192	2	0.5	0.5	4
	336	0.5	1	0.5	4
	720	1	1	1	1

H.5 MEASURED TRAINING COST AND PRACTICAL CONCLUSION

Fixed-bin adds no trainable tensor or quadrature contraction and retains Host's forward graph. Its implementation scales the existing kernel update after the Adam step. Table 32 reports paired training cost for the 96 default-configuration cell–seed pairs with comparable measurements. Every included pair shares the cell, seed, GPU model, software version, and CPU-thread setting.

Table 32: Measured default-configuration training cost on A100-SXM4-40GB GPUs (PyTorch 2.11.0, four CPU threads). Rows include matched measurements for both methods under the same cell, seed, and runtime. Time and memory columns give medians. F/H is the median of paired Fixed-bin/Host wall-time ratios. Wall time includes setup, training, and validation through each method's early stopping, excluding GPU queue time. Memory is peak allocated CUDA memory in MiB. Cost coverage is 96 of 160 pairs.

		Full training time			Peak memory (MiB)	
Dataset	Pairs	Host (s)	Fixed-bin (s)	F/H	Host	Fixed-bin
ETTh1	12	36.3	12.9	0.347	51.9	51.9
ETTh2	12	13.0	10.0	0.724	51.8	51.8
ETTm1	16	89.3	76.8	0.742	35.1	35.1
ETTm2	16	128.6	63.1	0.479	42.7	42.7
ECL	16	344.9	244.3	0.857	867.9	867.9
Traffic	12	607.9	655.8	1.012	2359.2	2359.2
Weather	8	168.5	144.2	0.910	132.1	132.1
Solar	4	188.1	175.9	0.962	394.3	394.3
All measured pairs	96	130.4	76.8	0.723	54.4	53.7

Across the 96 measured pairs, the median Fixed-bin/Host wall-time ratio is 0.723, quantifying realized training cost under the adopted stopping rule. The unchanged forward graph establishes the absence of an added inference branch; the timing results concern the full training pipeline.

The weighted-PSP benchmark shows that the geometry-derived update rule remains useful with kernel-conditioned attention. Fixed-bin improves accuracy without an auxiliary training branch, and its gain survives independent global-rate selection and evaluation on non-selection seeds. The strong scalar search reaches comparable aggregate accuracy with five times as many candidate configurations, supporting base-kernel update scale as the reusable design principle. Together with the profile controls, this comparison identifies base-kernel update scale as the main source of the rule's forecasting benefit under ordinary clipping.

H.6 READOUT SCALING AT A SHARED TRAINING STATE

Fixed-bin changes the base-kernel update while retaining Host's predictor and forecasting objective. We examine the first update of weighted PSP on ETTh2 with $L = H = 720$, period 12, and seed 42, using the default rate 10^{-3} and zero weight decay. The two recorded runs share identical initial parameters and identical updates outside the base kernel. Their base-kernel updates match the prescribed bin multipliers to a maximum absolute discrepancy of 5.78×10^{-8}. All losses below are evaluated on the same 256 evenly spaced validation windows, over every channel and forecast lead.

Figure 9 connects the update geometry to its forecasting effect. Starting from MSE 0.9926, Host and Fixed-bin reach 0.9839 and 0.9608 after one update: decreases of 0.0087 and 0.0318, respectively. Fixed-bin therefore achieves 3.66 times Host's loss decrease in this initial-state diagnostic. Their full parameter updates form an angle of $44.6°$, with a norm ratio of 22.7. To distinguish direction from magnitude, we also evaluate the Fixed-bin vector rescaled to Host's norm. This equal-norm step decreases MSE by only 0.0015, below Host's decrease. The comparison identifies update magnitude as a necessary contributor to the observed one-step advantage; the direction change alone is insufficient at Host's step length.

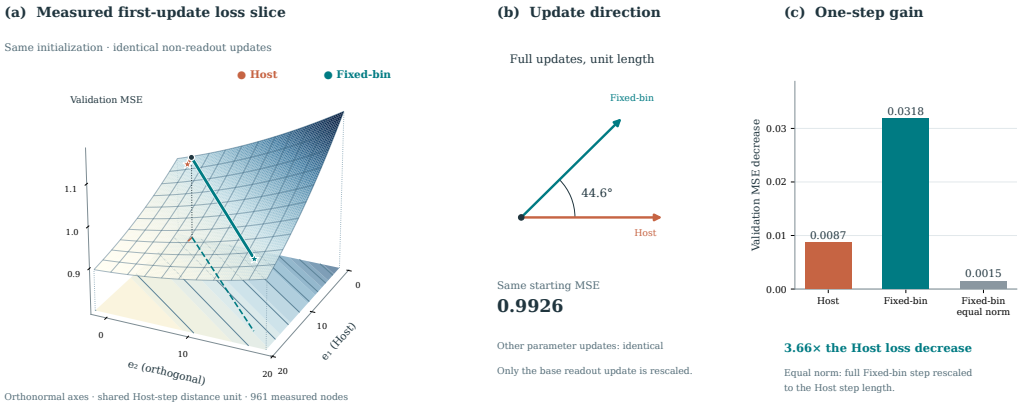


Figure 9: **Measured update geometry and one-step validation gain.** (a) A 31×31 validation-loss grid through the common initialization and both first-update endpoints. The orthonormal axes follow Host's update and the orthogonal component of Fixed-bin's update, both measured in Host-step norm units. Curves evaluate 21 points along each update segment; they are not multi-step training trajectories. Surface shading uses bilinear interpolation of the measured grid. (b) Full parameter updates normalized to unit length. (c) Actual loss decreases and the counterfactual obtained by rescaling the full Fixed-bin update to Host's norm. This comparison uses weighted PSP on ETTh2, seed 42, and the same 256 validation windows.

The origin and both update vectors lie in the measured plane, so this diagnostic requires no trajectory projection. The equal-norm control rescales the entire Fixed-bin vector, whereas the actual Fixed-bin rule modifies only the base-kernel update. Accordingly, the full-update angle reflects the changed weighting of the readout relative to the other parameters as well as differences across frequency bins. This local analysis complements the multi-seed forecasting comparisons above.