JOINTLY CORRECTED PARALLEL RESPONSES IN FORECASTING MODELS FOR NONLINEAR DYNAMICAL SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

How should added nonlinear computation be organized for forecasting nonlinear dynamical systems? We introduce the Jointly Corrected Parallel Forecaster (JCPF), which combines explicit aggregation of learned parallel increments with response-conditioned vector correction. The Compact Shared Forecaster (CSF) provides a common encoder/head reference and remains competitive on several generic forecasting tasks. With known future driving inputs, JCPF substantially improves on CSF across three measured-system benchmarks. On Silverbox and Wiener–Hammerstein, it reduces mean MSE relative to near-budget additive branches by 18.99% and 32.54%, with every seed agreeing, and improves on a wider residual MLP. Retrained ablations support retaining both direct aggregation and response-conditioned correction. Branch-conditioned and aggregate-conditioned corrections attain comparable final accuracy but different add-only-to-full error reductions: a larger correction gain can reflect greater reliance on that path rather than a better predictor. Complete increments can also be composed as parallel macro modules, with K denoting their number. From $K = 1$ to $K = 2$, WH improves under both fixed total budgets and fixed module widths, while F16 retains comparable accuracy. These findings support joint correction of parallel responses for nonlinear-system forecasting and distinguish final predictive value from the division of work learned by internal paths.

1 INTRODUCTION

Strong linear baselines, channel-independent patch models, and all-MLP forecasters show that useful forecasts need not require elaborate architectures (Zeng et al., 2023; Nie et al., 2023; Chen et al., 2023). Multiscale mixing and cross-variable models offer further ways to organize prediction (Wang et al., 2024a; Liu et al., 2024). For forecasting nonlinear dynamical systems, these results motivate a question beyond model size: *once a compact predictor is effective, how should additional nonlinear computation be organized?* A wider map, independent parallel branches, and jointly processed responses can use comparable resources while imposing different computational structures.

We investigate parallel nonlinear increments that contribute through two routes: an explicit normalized sum and an additive vector correction conditioned on their responses. The intuition comes from nonlinear dynamics: concurrent effects may enter a state equation additively, while their finite-interval response depends on joint evolution. This motivates preserving direct contributions alongside learned joint processing of complex nonlinear responses. Residual and multi-branch architectures provide established ingredients (He et al., 2016; Xie et al., 2017; Szegedy et al., 2017). Our target is their specific organization around learned latent increments, together with evidence about its predictive value, learned division of work, and reuse.

The Jointly Corrected Parallel Forecaster (JCPF) instantiates this organization within a shared forecasting interface. The Compact Shared Forecaster (CSF) supplies the compact encoder/head reference; JCPF adds four separately parameterized branches whose responses enter both aggregation and correction. All encoder and head weights are learned independently for each model. Nine generic datasets establish where the compact reference is competitive and where richer maps improve prediction. The main structural evidence comes from measured nonlinear systems with known future

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.

driving inputs: JCPF improves on CSF across the three primary benchmarks, while richer alternatives and ablations isolate the value of its organization.

The study connects three questions. First, does this organization improve prediction over alternative uses of nonlinear capacity? Second, must an effective correction read each branch separately, and what does its measured contribution reveal about the fitted predictor? Third, can a complete aggregation–correction unit be reused as a module? Our main findings are:

1. **Joint correction improves the use of parallel increments.** On Silverbox and Wiener–Hammerstein (WH), JCPF improves on near-budget additive branches and a wider residual MLP. Independently retrained ablations support retaining both direct aggregation and response-conditioned correction.
2. **Similar accuracy can reflect different path contributions.** Branch-conditioned and aggregate-conditioned corrections attain comparable final accuracy but different add-only-to-full error reductions. A larger correction gain can reflect greater reliance on that path rather than a better final predictor.
3. **The complete unit supports modular reuse.** With K denoting the number of complete JCPF-increment modules, composition from $K = 1$ to $K = 2$ is evaluated under both fixed total budgets and fixed module widths. WH improves on both axes, while F16 retains comparable prediction accuracy.

Together, these findings establish JCPF as a concrete organization of added nonlinear computation for nonlinear-system forecasting and distinguish three aspects of its value: prediction after retraining, the division of work within a fitted predictor, and reuse at a higher level. The experiments keep input information and training protocols matched within each structural comparison and report measured computation costs alongside accuracy.

2 Related work

Compact and structured forecasting. DLinear (Zeng et al., 2023), PatchTST (Nie et al., 2023), and TSMixer (Chen et al., 2023) establish complementary roles for simple temporal projection, channel-independent sharing, and MLP mixing. Other compact designs add cycle or sparse-interaction structure (Lin et al., 2024b;a; Zhang et al., 2025), while TimeDistill transfers multi-scale and multi-period patterns from Transformer or CNN teachers to an MLP forecaster (Ni et al., 2026). CSF is a directly trained dense shared reference without those added mechanisms.

Multiscale methods organize representations through wavelet and patch mixing (WPMixer), adaptive time–frequency mixing (TimeMixer++), and semantic enhancement before progressive mixing (SEMixer) (Murad et al., 2025; Wang et al., 2025; Zhang et al., 2026). Our comparisons fix the outer encoder/head form and input interface within each setting, train each model independently, and vary how its middle map combines nonlinear responses.

Information sharing and the input contract. Strong generic-benchmark forecasts need not rely on cross-channel attention. DLinear uses channel-wise linear temporal maps; PatchTST applies temporal attention independently to each channel (Zeng et al., 2023; Nie et al., 2023). TSMixer's time-mixing-only ablation also performs close to its full time-and-feature mixer on standard long-term benchmarks (Chen et al., 2023). This motivates a channel-independent generic interface for our structural study. Other settings benefit from richer communication: iTransformer attends across variate tokens, SOFTS redistributes an aggregated core, and TimeXer separates endogenous and exogenous inputs (Liu et al., 2024; Han et al., 2024; Wang et al., 2024b). RevIN addresses instance-level normalization and restoration (Kim et al., 2022). Within each of our settings, the supplied information is held fixed.

Residual aggregation and joint correction. ResNet, ResNeXt, and Inception-ResNet establish residual addition, parallel transformations, and post-branch projection (He et al., 2016; Xie et al., 2017; Szegedy et al., 2017). Selective Kernel Networks use branch responses to condition branch weights (Li et al., 2019); residual path analysis studies unequal contributions within jointly trained networks (Veit et al., 2016). Time-MoE separates total capacity from active computation through sparse experts (Shi et al., 2025).

The two TSMixer papers address different constructions. Chen et al. (2023) alternate time and feature mixing to study the value of multivariate information. Ekambaram et al. (2023) add forecast reconciliation: a cross-channel head uses neighboring predictions for residual adjustment, while hierarchical patch reconciliation encourages consistency across temporal scales. These are relevant precedents for learned combination and correction. JCPF operates on learned latent increments within the middle map: every increment enters an explicit normalized sum and conditions an additional vector correction. It studies this arrangement through alternative allocations, retrained correction representations, fitted-path diagnostics, and complete-unit reuse; it does not require prediction-space channel or hierarchy constraints.

Nonlinear response prediction. Silverbox, WH, and Cascaded Tanks offer measured systems with different nonlinear responses (Wigren & Schoukens, 2013; Schoukens et al., 2009; Schoukens & Noël, 2017). Recent state-space Kolmogorov–Arnold networks study accuracy and interpretability on Silverbox and WH (Cruz et al., 2025); online neural state-space learning addresses state estimation and recursive identification (Györök et al., 2026). These approaches learn state evolution and its estimation. Our complementary target is a direct finite-window response map with known driving inputs, evaluated through local matched comparisons. The next section develops the dynamical motivation and defines the two forecasting structures.

3 METHOD

3.1 A SHARED FORECASTING INTERFACE AND COMPACT REFERENCE

Let L denote the observation length and H the prediction horizon. Each shared-interface model uses a history encoder, a middle transformation $T : \mathbb{R}^d \to \mathbb{R}^d$, and a linear prediction head. This shared outer interface makes the added middle computation explicit. Encoder and head weights are learned independently for every model and seed; in multivariate forecasting, sharing means that channels within one model use the same weights.

For channel c, let μ_c and s_c be the history mean and scale. With encoder $W \in \mathbb{R}^{d\times L}$ and head $V \in \mathbb{R}^{H\times d}$,

$$q_c = W\frac{x_c - \mu_c}{s_c} + b, \qquad \hat{y}_c = x_{L,c}\mathbf{1}_H + s_c VT(q_c). \tag{1}$$

Channels are processed independently. For driven physical systems, the encoder instead receives

$$v = [y_{\text{hist}}; u_{\text{hist}}; u_{\text{future}}] \in \mathbb{R}^{2L+H}, \qquad q = Wv + b, \qquad \hat{y} = y_L\mathbf{1}_H + VT(q). \tag{2}$$

Here u_{future} is the known driving input over the response horizon. The physical adapter uses training-only standardization, and every physical comparator receives this same input contract.

The Compact Shared Forecaster (CSF) uses SiLU (Elfwing et al., 2018) as its whole middle map:

$$T_{\text{CSF}}(q) = \text{SiLU}(q) = q\sigma(q). \tag{3}$$

It is a compact nonlinear control inside the common encoder/head framework. Added computation must improve on a learned history representation, nonlinear activation, and direct residual output.

3.2 FROM ADDITIVE EFFECTS TO JOINT RESPONSE CORRECTION

A useful design intuition comes from systems in which several effects act on the same state. Parallel block models are established within nonlinear-identification design spaces (Schoukens et al., 2015; Schoukens & Ljung, 2019). For example, a driven nonlinear oscillator can be written as

$$\ddot{s} = -\delta\dot{s} - \alpha s - \beta s^3 + u. \tag{4}$$

Damping, restoration, and forcing enter additively, although their finite-interval response depends on their joint evolution. The standard local flow expansion makes the same distinction. For a smooth autonomous system $\dot{x} = f(x) = \sum_b f_b(x)$,

$$x(t+\Delta) = x(t) + \Delta\sum_b f_b(x) + \frac{\Delta^2}{2}\sum_{i,j} Df_i(x)f_j(x) + O(\Delta^3), \tag{5}$$

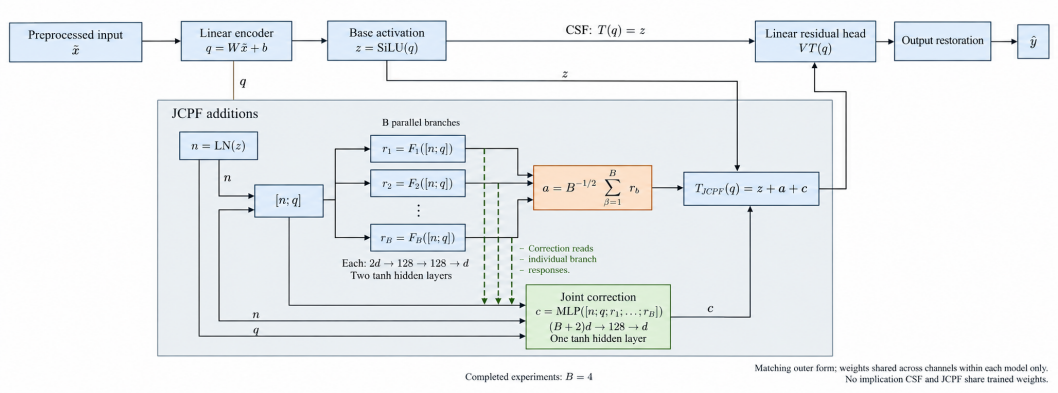


Figure 1: The shared forecasting interface and JCPF's direct aggregation and response-conditioned correction.

where all terms are evaluated at $x = x(t)$. The first-order increment is additive and higher-order response terms can couple contributions (Hairer et al., 2006). This motivates retaining an aggregate route alongside joint processing. It is not the implemented update rule: learned branches need not represent f_b, and the correction need not be a Taylor coefficient or a physical interaction.

3.3 Jointly Corrected Parallel Forecaster

JCPF begins with $z = \mathrm{SiLU}(q)$ and $n = \mathrm{LN}(z)$ (Ba et al., 2016). It computes B separately parameterized branch responses and combines a direct aggregate with a correction:

$$r_b = \mathrm{MLP}_b([n; q]), \qquad b = 1, \ldots, B, \tag{6}$$

$$a = B^{-1/2} \sum_{b=1}^{B} r_b, \tag{7}$$

$$c = \mathrm{MLP}_C([n; q; r_1; \ldots; r_B]), \tag{8}$$

$$T_{\mathrm{JCPF}}(q) = z + a + c. \tag{9}$$

Each branch maps $\mathbb{R}^{2d}$ to $\mathbb{R}^d$ through two tanh hidden layers and a linear output. The correction maps $\mathbb{R}^{(B+2)d}$ to $\mathbb{R}^d$ through one tanh hidden layer and a linear output. The completed experiments use hidden width 128 and $B = 4$; B remains a design parameter.

The aggregate makes branch contributions directly available to the head. The correction reads individual responses before aggregation, so it can implement a general conditional residual map over their joint configuration. It also reads q and n. The two routes are learned together; route outputs are therefore not separately trained component predictors. The factor $B^{-1/2}$ provides a reference normalization for comparably scaled, uncorrelated responses, while trained branches may be correlated. Small initial output maps start the richer transformation near the compact base.

3.4 Alternative allocations, correction representation, and route ablations

The near-parameter parallel model (PNEAR) uses

$$T_{\mathrm{PNEAR}}(q) = z + B^{-1/2} \sum_{b} \widetilde{\mathrm{MLP}}_b([n; q]). \tag{10}$$

It retains the bypass and direct aggregate, omits response-conditioned correction, and widens its branches to match the overall parameter scale. The ordinary wide residual MLP (R2048, hidden width 2,048) uses

$$T_{\mathrm{R2048}}(q) = z + \mathrm{MLP}_R([z; q]), \tag{11}$$

with one tanh hidden layer and a linear output. It has no parallel branch group or joint correction. PNEAR and R2048 are alternative budget allocations, not single-module ablations.

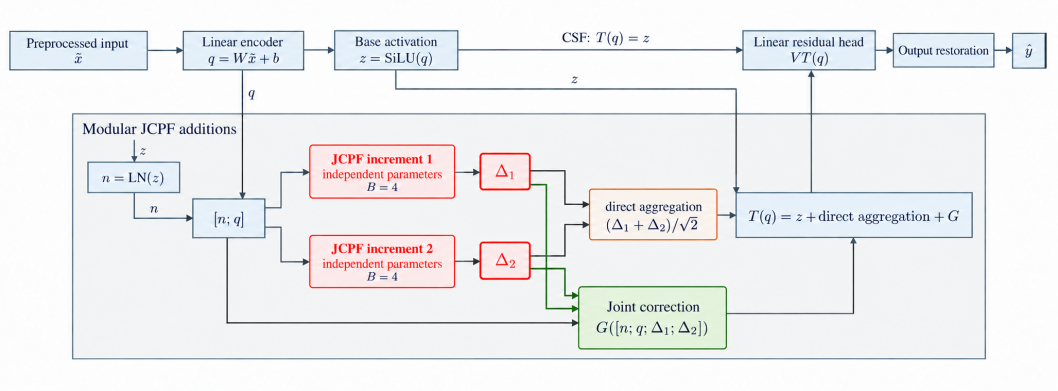


Figure 2: Reuse of complete JCPF increments: z is added once; Δ_1 and Δ_2 feed both the sum and outer correction G.

The matched aggregate-conditioned variant Cagg retains the direct sum and changes only the correction input to

$$c_{\mathrm{agg}} = \mathrm{MLP}_{\mathrm{agg}}([n; q; a]). \tag{12}$$

Its width is adjusted to match the JCPF parameter budget. Two independently retrained route variants instead remove branch-response access while retaining the direct sum, or remove the direct sum while retaining branch access through the correction. These variants test the complete two-route organization. All alternatives are trained independently; exact sizes are in Appendix A.

3.5 Macro composition of JCPF transitions

We also reuse a complete JCPF increment as a parallel macro module. Here K denotes the number of parallel complete JCPF-increment macro modules; it is distinct from the internal branch count B, hidden widths, and training seeds. For independently parameterized modules,

$$r_{k,b} = F_{k,b}([n; q]), \qquad \Delta_k = B^{-1/2} \sum_{b=1}^{B} r_{k,b} + C_k([n; q; r_{k,1}; \ldots; r_{k,B}]), \tag{13}$$

$$T_{\mathrm{K2Joint}}(q) = z + \frac{\Delta_1 + \Delta_2}{\sqrt{2}} + G([n; q; \Delta_1; \Delta_2]). \tag{14}$$

Independent replaces G with $D([n; q])$, and Sum omits the outer map. The K=2 construction reuses the aggregation–correction organization at the level of complete increments.

4 Experiment

4.1 Datasets and controlled protocol

The generic matrix contains ETTm1, ETTm2, ETTh1, ETTh2 (Zhou et al., 2021), Exchange, Solar, Electricity, Traffic (Lai et al., 2018), and Weather. Silverbox, Wiener–Hammerstein (WH), and Cascaded Tanks provide measured nonlinear systems (Wigren & Schoukens, 2013; Schoukens et al., 2009; Schoukens & Noël, 2017). Module reuse is evaluated on WH and the F-16 Ground Vibration Test (F16), measured on a full-scale aircraft with nonlinear payload-mounting interfaces (Noël & Schoukens, 2020). Physical comparisons share past outputs, past inputs, and known future inputs. F16 extends equation 2 to three jointly predicted acceleration channels and one force input.

All comparisons use three seeds. Structural models within each comparison share an AdamW recipe (Loshchilov & Hutter, 2019), update budget, sampler, and validation-MSE checkpoint rule (Cawley & Talbot, 2010); ablations are independently retrained for the full budget. TEST is evaluated after validation selects the checkpoint. PatchTST retains its recorded task-specific optimizer schedule. A separate historical nine-dataset screen identified it as a strong overall small-scale baseline among eight evaluated profiles (Appendix E.4).

4.2 COMPARISONS FOR THE THREE FINDINGS

Nonlinear organization. The generic matrix establishes the compact CSF reference. The measured systems then compare JCPF with near-parameter additive branches (PNEAR) and an ordinary residual MLP with hidden width 2,048 (R2048). These test alternative allocations of computation. Two retrained ablations make the route question more specific: *independent correction* retains the sum but prevents the correction from reading branch responses; *no direct sum* retains response-conditioned correction but removes explicit aggregation.

Correction representation and path contribution. On Silverbox and WH, aggregate-conditioned correction (Cagg) retains the four branches and direct sum, but supplies $[n; q; a]$ to the correction. Its hidden width is adjusted to nearly match JCPF's total parameters; both receive the same training and selection protocol. This retrained comparison tests correction representations. Separately, each selected checkpoint is evaluated with and without its fitted correction to measure how its two paths share predictive work.

Modular reuse. A fresh $K = 1$ reference is compared with $K = 2$ Joint, Independent, and Sum compositions. The fixed-budget axis narrows the modules to maintain total parameter scale; the fixed-width axis retains each module's original width and increases capacity. On each axis, Joint and Independent use the same internal module widths and differ in response access and the outer width needed to match parameters. Sum reallocates the outer budget on the fixed-budget axis. Seven configurations, two datasets, and three seeds give 42 fresh runs; the shared K1 reference belongs only to this extension.

4.3 EVALUATION AND FITTED-PATH DIAGNOSTICS

MSE over the complete horizon is primary, with MAE complementary. Tables report three-seed means and, where shown, sample standard deviations. Relative reductions use seed-mean MSE; generic aggregation uses an equal-dataset geometric mean of MSE ratios. Parameter counts and measured latency, throughput, and memory describe resource use. Exact protocols and full matrices appear in the appendix.

For the fitted-path diagnostic, let y_{add} be the decoded prediction with the selected correction removed, δ its decoded correction, and $e_{\mathrm{add}} = y - y_{\mathrm{add}}$. Then

$$\mathrm{MSE}_{\mathrm{add}} - \mathrm{MSE}_{\mathrm{full}} = 2\,\mathrm{mean}(e_{\mathrm{add}}\delta) - \mathrm{mean}(\delta^2). \tag{15}$$

Means cover evaluation windows, forecast steps, and output channels. Original models remove only C; macro models remove only outer G. Both evaluations use the same validation-selected weights, so this measures a fitted path's contribution rather than a retrained no-correction architecture.

5 RESULTS

5.1 THE COMPACT REFERENCE AND PERFORMANCE CONTEXT

Table 1 places the structural study in context. On the lower-dimensional ETT and Exchange tasks, the shared-interface models and PatchTST have relatively close errors. Larger separations appear on Solar, Electricity, and Traffic. Across all nine datasets, JCPF's geometric-mean MSE reductions over PNEAR and R2048 are 0.73% and 0.12%. The near-ties establish a strong compact reference and motivate examining the settings where added computation has a clearer effect.

The compact CSF remains competitive on several datasets, while richer middle maps improve prediction more visibly on Solar, Electricity, and Traffic. All shared-interface variants use the same validation-selected latent width on Traffic. These improvements arise within a channel-independent adapter and thus concern temporal response modeling. The measured systems below provide a complementary setting in which richer nonlinear response maps can be compared under known driving inputs.

5.2 JOINT CORRECTION AS A NONLINEAR ORGANIZATION

Across the three measured systems, JCPF substantially improves on CSF, showing the value of a richer response map under known future driving inputs. Comparisons among richer maps then

Table 1: Generic $H = 96$ normalized MSE (three-seed mean $\pm$ sample SD). Bold marks row minima.

Dataset	CSF	JCPF	PNEAR	R2048	PatchTST
ETTm1	0.3983 ± 0.0034	0.3998 ± 0.0044	0.4000 ± 0.0046	$\mathbf{0.3969 \pm 0.0056}$	0.4035 ± 0.0014
ETTm2	0.1555 ± 0.0002	0.1576 ± 0.0007	0.1576 ± 0.0008	0.1582 ± 0.0012	$\mathbf{0.1545 \pm 0.0005}$
ETTh1	0.4663 ± 0.0041	0.4669 ± 0.0007	0.4858 ± 0.0192	$\mathbf{0.4614 \pm 0.0019}$	0.4868 ± 0.0040
ETTh2	0.2461 ± 0.0026	0.2515 ± 0.0025	0.2498 ± 0.0026	0.2524 ± 0.0022	$\mathbf{0.2394 \pm 0.0017}$
Exchange	0.0871 ± 0.0017	0.0851 ± 0.0008	0.0857 ± 0.0012	$\mathbf{0.0849 \pm 0.0011}$	0.0923 ± 0.0008
Weather	0.1868 ± 0.0020	$\mathbf{0.1774 \pm 0.0016}$	0.1781 ± 0.0026	0.1807 ± 0.0005	0.1877 ± 0.0008
Solar	0.2458 ± 0.0011	0.2213 ± 0.0060	0.2187 ± 0.0061	$\mathbf{0.2147 \pm 0.0034}$	0.2930 ± 0.0024
Electricity	0.1931 ± 0.0012	$\mathbf{0.1735 \pm 0.0005}$	0.1757 ± 0.0008	0.1759 ± 0.0008	0.2025 ± 0.0010
Traffic	0.5182 ± 0.0005	$\mathbf{0.4542 \pm 0.0033}$	0.4640 ± 0.0013	0.4646 ± 0.0029	0.5541 ± 0.0019

Table 2: Measured-system $H = 96$ normalized MSE (three-seed mean $\pm$ sample SD).

System	CSF	JCPF	PNEAR	R2048	PatchTST
Silverbox	0.0542 ± 0.0019	$\mathbf{0.0142 \pm 0.0001}$	0.0175 ± 0.0001	0.0162 ± 0.0003	0.0364 ± 0.0010
Tanks	0.4935 ± 0.0404	0.3829 ± 0.0070	0.4012 ± 0.0110	$\mathbf{0.3418 \pm 0.0232}$	0.5629 ± 0.0144
WH	0.0074 ± 0.0002	$\mathbf{0.0022 \pm 0.00004}$	0.0032 ± 0.00003	0.0040 ± 0.0004	0.0198 ± 0.00007

examine its organization. On Silverbox and WH, JCPF lowers mean MSE relative to PNEAR by 18.99% and 32.54%, with every seed agreeing, and also improves on the wider R2048 (Table 2). These gains support assigning part of the nonlinear computation to joint response correction: the comparisons include both a near-matched additive budget and a larger single residual map. Cascaded Tanks instead favors the ordinary R2048 allocation.

The retrained variants in Figure 3 make this comparison more specific. *Independent correction* keeps an extra nonlinear path but removes its access to branch responses; *no direct sum* keeps response access but removes explicit aggregation. Both have higher MSE and MAE than full JCPF at every tested seed on Silverbox and WH. The consistent direction after reoptimization supports the complete two-route organization. It connects the dynamical motivation to a useful learned construction: explicitly aggregated contributions and response-conditioned joint processing improve prediction together.

The controls clarify how composition helps. Independent correction retains a nonlinear function of the encoded context, but full JCPF also lets this function use the concurrent branch responses. No direct sum retains those responses and passes them to the head through correction, whereas full JCPF additionally provides their direct aggregate. The comparisons therefore support both response access and explicit aggregation. In combination with the ordinary R2048 capacity control, they provide evidence for the particular organization of computation beyond the benefit of adding an MLP or a branch group.

5.3 SIMILAR ACCURACY, DIFFERENT PATH CONTRIBUTIONS

Joint correction remains effective with two response representations. Cagg reads the aggregate $[n; q; a]$ and attains final accuracy comparable to branch-conditioned JCPF (Figure 3). Silverbox slightly favors JCPF in all three seeds; WH's lower Cagg mean is driven by one seed, with the other two slightly worse. These outcomes support comparable predictive performance for the two parameterizations. Both preserve the response-conditioned correction that the preceding ablations support.

Table 3 reveals a distinction concealed by final MSE. Cagg has substantially larger add-only error, yet its learned correction brings full prediction close to JCPF. Its larger correction gain therefore accompanies a less accurate additive prediction within the jointly trained model. Equation 15 explains why: the gain depends on both the additive residual and the correction's alignment and energy. Across fitted models, a larger gain can reflect more predictive work assigned to correction, not a better final predictor. Thus the retrained accuracy comparison and the fitted-path diagnostic answer different questions.

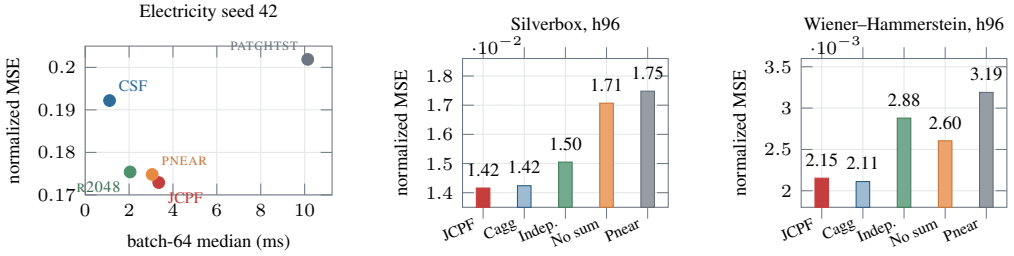


Figure 3: Inference cost on Electricity (left) and retrained $H = 96$ comparisons on Silverbox and WH (center/right).

Table 3: Full and add-only normalized MSE from the same selected checkpoint (three-seed means).

System	Model	Full MSE	Add-only MSE
Silverbox	JCPF	0.01416	0.06130
Silverbox	Cagg	0.01424	0.10631
WH	JCPF	0.00215	0.12801
WH	Cagg	0.00211	0.21708

Table 4: Macro-composition normalized TEST MSE (three-seed means). Joint–Ind. is $100(\text{Joint}/\text{Independent} - 1)$; negative favors Joint.

Setting	Axis	K1	Joint	Independent	Joint–Ind.
F16	fixed budget	0.413518	0.415392	0.414507	$+0.21\%$
F16	fixed width	0.413518	0.414956	0.417414	-0.59%
WH	fixed budget	0.002152	0.001964	0.002129	-7.78%
WH	fixed width	0.002152	0.002056	0.002174	-5.43%

This distinction also informs the choice of correction input. Cagg presents a lower-dimensional summary of branch responses; JCPF supplies their individual vectors. In either case, the branches and outer maps are trained together and can adapt to the available representation. Their comparable full MSE coexists with different additive predictions and correction gains. The experiment thus identifies flexibility within the proposed organization: two ways of exposing responses support similar final predictions while forming different internal divisions of work. The combination of retrained accuracy and fitted-path measurements makes this difference observable.

Figure 4 resolves this division by forecast step. The distinct gain profiles show how reliance on correction varies over the horizon instead of reducing it to a single score. Together with the retrained ablations, this supplies an empirical connection between the proposed two-route design and its learned use: the complete organization improves prediction in the tested systems, while its internal division of work adapts to the correction representation. Seed-level diagnostics and the outer-correction decomposition are in Appendix G.

5.4 Reuse of the complete aggregation–correction unit

Macro composition tests whether the construction remains useful when each parallel input is itself a complete JCPF increment. Table 4 compares Joint with the fresh K1 reference and response-independent outer correction. WH improves over both on both resource axes; Joint also has lower mean error than Sum in the full matrix (Appendix D). The benefit can therefore occur when the total budget is nearly unchanged, as well as when original-width modules add capacity.

Joint and Independent retain identical internal module widths on each axis; only response access and the outer hidden width used for parameter matching differ. Their WH comparison supports reading complete module responses at the second level. Fixed-budget Joint also outperforms fixed-width Joint, so increased capacity alone does not account for the improvement. F16 retains prediction accuracy close to K1 on both axes.

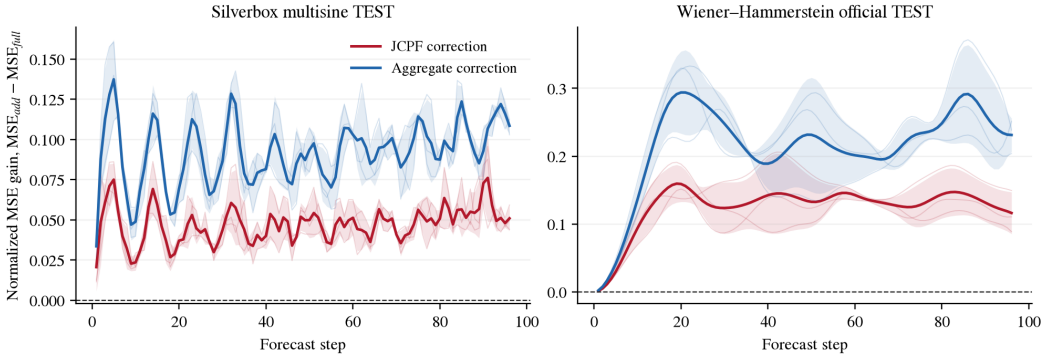


Figure 4: Forecast-step MSE reduction from fitted correction: JCPF (red) and Cagg (blue), three-seed means and variation.

The two axes distinguish reorganizing resources from increasing capacity. Under a fixed budget, internal module width is traded for multiple complete increments and outer correction. Under fixed width, each increment retains its capacity and composition adds computation. This separates modular organization from the effect of added capacity.

F16 also has positive whole-split outer-correction removal gains in the fitted models. This extends the preceding distinction: a correction can be useful within a trained composition without making that composition more accurate than K1. Batch-64 inference for K2 Joint costs roughly twice that of K1 in the recorded measurements. The result establishes a usable form of modular reuse while keeping its accuracy benefit separate from the cost of additional computation.

5.5 Findings and scope

The three findings connect organization, learned use, and reuse. Controlled alternatives support direct aggregation with response-conditioned correction; the representation comparison shows that similar predictive quality can conceal different path contributions; and macro composition reuses the complete unit on two resource axes. The shared encoder/head form makes these structural questions comparable without changing the information supplied to a predictor.

The evidence covers the reported datasets, three training seeds, and fixed protocols. The strongest architectural evidence comes from the matched nonlinear-system comparisons; small generic gaps and near-ties between correction forms remain descriptive. Fitted-path diagnostics characterize a predictor's dependence on its learned routes. Module reuse is demonstrated from $K = 1$ to $K = 2$, leaving broader transfer and scaling behavior open.

6 Conclusion

JCPF organizes added nonlinear computation for forecasting nonlinear dynamical systems through explicit aggregation of parallel increments and response-conditioned joint correction. The compact CSF remains competitive on several generic forecasting tasks; measured nonlinear systems show clearer gains from richer response maps. A common outer forecasting form supports controlled comparisons of prediction, path contributions, and reuse.

The matched nonlinear-system comparisons identify benefits of the complete two-route organization over alternative capacity allocations. Comparable accuracy with different correction representations reveals flexibility in the learned division of work, which fitted-path measurements make observable. Reusing complete increments demonstrates modular potential under both fixed-budget and fixed-width controls, with measured accuracy and computation costs. Future work should examine how dynamical characteristics and training shape the benefits and division of predictive work, and extend modular reuse to broader systems and module counts.

REFERENCES

Valentino Assandri, Sam Heshmati, Burhaneddin Yaman, Anton Iakovlev, and Ariel Emiliano Repetur. Deep double descent for time series forecasting: Avoiding undertrained models. *arXiv preprint arXiv:2311.01442*, 2023. URL https://arxiv.org/abs/2311.01442.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. URL https://arxiv.org/abs/1607.06450.

Gavin C. Cawley and Nicola L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(70):2079–2107, 2010. URL https://www.jmlr.org/papers/v11/cawley10a.html.

Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O. Arik, and Tomas Pfister. TSMixer: An all-mlp architecture for time series forecasting. *Transactions on Machine Learning Research*, 2023.

Gonçalo G. Cruz, Balázs Renczes, Mark C. Runacres, and Jan Decuyper. State-space Kolmogorov Arnold networks for interpretable nonlinear system identification. *IEEE Control Systems Letters*, 9:847–852, 2025. doi: 10.1109/LCSYS.2025.3578019. URL https://ieeexplore.ieee.org/document/11029059/.

Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. TSMixer: Lightweight MLP-Mixer model for multivariate time series forecasting. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 459–469, 2023. doi: 10.1145/3580305.3599533. URL https://doi.org/10.1145/3580305.3599533.

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

Bendegúz Györök, Tamás Péni, Maarten Schoukens, and Roland Tóth. Online learning of neural state-space models. *arXiv preprint arXiv:2607.17614*, 2026. URL https://arxiv.org/abs/2607.17614.

Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*, volume 31 of *Springer Series in Computational Mathematics*. Springer, 2 edition, 2006. doi: 10.1007/3-540-30666-8. URL https://doi.org/10.1007/3-540-30666-8.

Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. SOFTS: Efficient multivariate time series forecasting with series-core fusion. In *Advances in Neural Information Processing Systems*, 2024.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=cGDAkQo1C0p.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2018.

Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 510–519, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Li_Selective_Kernel_Networks_CVPR_2019_paper.html.

Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. CycleNet: Enhancing time series forecasting through modeling periodic patterns. In *Advances in Neural Information Processing Systems*, 2024a.

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

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations*, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Md Mahmuddun Nabi Murad, Mehmet Aktukmak, and Yasin Yilmaz. WPMixer: Efficient multi-resolution mixing for long-term time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

Juntong Ni, Zewen Liu, Shiyu Wang, Ming Jin, and Wei Jin. TimeDistill: Efficient long-term time series forecasting with MLP via cross-architecture distillation. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2026.

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

Jean-Philippe Noël and Maarten Schoukens. F-16 aircraft benchmark based on ground vibration test data. 4TU.ResearchData, 2020. URL `https://doi.org/10.4121/12954911`. Dataset.

Johan Schoukens and Lennart Ljung. Nonlinear system identification: A user-oriented road map. *IEEE Control Systems*, 39(6):28–99, 2019. doi: 10.1109/MCS.2019.2938121.

Johan Schoukens, Johan Suykens, and Lennart Ljung. Wiener-hammerstein benchmark. In *15th IFAC Symposium on System Identification*, 2009. URL `https://www.nonlinearbenchmark.org/benchmarks/wiener-hammerstein`.

Maarten Schoukens and Jean-Philippe Noël. Three benchmarks addressing open challenges in nonlinear system identification. *IFAC-PapersOnLine*, pp. 448–453, 2017. doi: 10.1016/j.ifacol.2017.08.071. URL `https://www.nonlinearbenchmark.org/benchmarks/cascaded-tanks`.

Maarten Schoukens, Anna Marconato, Rik Pintelon, Gerd Vandersteen, and Yves Rolain. Parametric identification of parallel wiener–hammerstein systems. *Automatica*, 51:111–122, 2015. doi: 10.1016/j.automatica.2014.10.105.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-MoE: Billion-scale time series foundation models with mixture of experts. In *International Conference on Learning Representations*, 2025.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017.

Andreas Veit, Michael Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. In *Advances in Neural Information Processing Systems*, 2016.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y. Zhang, and Jun Zhou. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *International Conference on Learning Representations*, 2024a.

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

Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Guo Qin, Haoran Zhang, Yong Liu, Yunzhong Qiu, Jianmin Wang, and Mingsheng Long. TimeXer: Empowering transformers for time series forecasting with exogenous variables. In *Advances in Neural Information Processing Systems*, volume 37, 2024b. doi: 10.52202/079017-0015.

Torbjörn Wigren and Johan Schoukens. Three free data sets for development and benchmarking in nonlinear system identification. In *European Control Conference*, pp. 2933–2938, 2013. doi: 10.23919/ECC.2013.6669201. URL https://www.nonlinearbenchmark.org/benchmarks/silverbox.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

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

Xu Zhang, Qitong Wang, Peng Wang, and Wei Wang. A lightweight sparse interaction network for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 13304–13312, 2025.

Xu Zhang, Qitong Wang, Peng Wang, and Wei Wang. SEMixer: Semantics enhanced MLP-Mixer for multiscale mixing and long-term time series forecasting. In *Proceedings of the ACM Web Conference*, 2026.

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

REPRODUCIBILITY STATEMENT

Source code is provided in the supplementary code package. The package contains the CSF/JCPF models, paper-specific controls, data adapters, and training and evaluation scripts for the main comparisons, retrained ablations, aggregate-conditioned correction (Cagg), and macro composition under both fixed total budgets and fixed module widths. It also includes the Traffic width-selection study, training-budget comparison, synthetic negative control, fixed-checkpoint correction diagnostics, inference benchmarks, and scripts for the selected learning curves and prediction plots. Frozen configurations, tests, data-access instructions, and compact metric and figure records accompany the code; model identifiers follow the paper's terminology.

The README and experiment-coverage inventory map the main-text and appendix studies to their entry points and reproduction guides. External baselines are documented through official source-code and paper links. The historical eight-baseline comparison additionally includes its MSE/MAE records, protocol, ranks, and implementation provenance (Appendix E.4). Dataset preparation, training objectives, and evaluation metrics are detailed in Appendices A–C. Reproduction runs select checkpoints using validation data before evaluating TEST. Datasets, checkpoints, logs, caches, and local historical artifacts are excluded from the supplementary package; `.gitignore` also excludes these generated files from version control.

A DATASETS AND PREDICTION PROTOCOL

This appendix first specifies the data, model configurations, training objective, and metric populations. It then presents the horizon-96 accuracy and efficiency summaries, validation experiments, and selected learning and prediction curves. Numerical summaries are derived from the retained run artifacts; new visualizations use retained checkpoints without additional model training.

A.1 GENERIC FORECASTING DATASETS

Details for Section 4.

The nine generic tasks cover several types of multivariate temporal observations. ETTm1, ETTm2, ETTh1, and ETTh2 contain electricity-transformer measurements, including oil temperature, at the two recorded temporal resolutions (Zhou et al., 2021). Exchange comprises exchange-rate series; Electricity comprises electricity-consumption series; Traffic comprises traffic measurements; Solar comprises solar-power generation series (Lai et al., 2018). Weather contains meteorological variables. Dataset identifiers refer to the retained local versions, whose numeric feature order and chronological split boundaries are recorded with the corresponding runs.

The number of numeric channels is 7 for each ETT task, 8 for Exchange, 21 for Weather, 137 for Solar, 321 for Electricity, and 862 for Traffic. Timestamp columns are excluded from the numeric prediction targets. Channels share model weights in the four shared-interface architectures, but each channel is forecast from its own history; channel correlation is therefore a property of the observed dataset rather than an explicitly modelled cross-channel path.

Each channel is standardized using its training-segment mean and population standard deviation. The fitted statistics are then applied to validation and test values without refitting. The generic shared-interface models additionally normalize the history window internally and restore their prediction around its last observation. These two transformations have distinct roles: dataset standardization defines the common output scale, while history normalization defines the model's local input coordinates.

The training windows have $L = 96$ observations and $H = 96$ targets. Validation and test targets begin within their designated chronological segments; their first histories may use the preceding 96 observations. Evaluation origins are sampled every 24 time steps, with the last legal origin included. The actual origin lists are authoritative: for example, ETTm1 has 477 validation origins and 981 test origins, rather than the much larger population obtained by evaluating every possible test start. All full-table values use the recorded populations.

The preserved ETT loader ends training at $12 \times 30 \times 24 \times r$ and validation at $16 \times 30 \times 24 \times r$, where $r = 4$ for ETTm and $r = 1$ for ETTh. Its final test segment uses the remaining recorded file tail. The other generic datasets end training at $\lfloor 0.7T \rfloor$ and validation at $T - \lfloor 0.2T \rfloor$, where T is the registered row count. These are the partitions used by the frozen artifacts, rather than an assumption that every split matches a different paper's benchmark convention.

A.2 MEASURED NONLINEAR SYSTEMS

Details for Section 4.

Silverbox is an electronic nonlinear benchmark with Duffing-like behavior; WH combines linear dynamic blocks and a static nonlinearity; Cascaded Tanks contains interacting flow and overflow dynamics (Wigren & Schoukens, 2013; Schoukens et al., 2009; Schoukens & Noël, 2017). Each experiment uses one driving-input series and one measured-output series. The predictor receives 96 past outputs, 96 past inputs, and 96 known future inputs, and predicts the next 96 outputs. Future output values are never supplied as model inputs.

Input and output scaling statistics are fitted only on the physical training segment. Unlike the generic adapter, this setting does not apply per-window normalization. Each physical evaluation window remains completely inside its assigned segment, and every legal origin is evaluated with stride one. The known future input supplies the driving sequence for response prediction; the task is not an autonomous rollout from a learned physical state.

Indices below are zero-based, with half-open intervals. Silverbox uses multisine observations $[40650, 105712)$ for the train/validation record and $[105712, 127400)$ for its main test segment. WH uses $[5200, 105200)$ for train/validation and $[105200, 184000)$ for final evaluation. Each train/validation record is divided chronologically into 60% training, 20% selection validation, and 20% assessment. Cascaded Tanks uses 614/205/205 observations from its 1,024-point estimation record, followed by the official 1,024-point validation record as the final test record. Its internal training, selection, and assessment segments contain 423, 14, and 14 legal windows, respectively.

The primary physical table includes one main segment per system. Silverbox's arrow-full $[100, 40575)$ and arrow-core $[100, 32100)$ views are retained separately as overlapping views of the same measured system. They are not counted as additional independent systems or pooled into the main multisine result.

A.3 Data access

Details for Section 4.

The formal optimization and validation procedures do not load test labels. The numerical tables report fixed-protocol observations of validation-selected checkpoints. Illustrative predictions use retained checkpoint identities checked against the archived records. The physical test records had historical development exposure; this is documented separately from the formal training flag `test_loaded=false`.

Missing measurements, constant features, and unavailable historical records are reported explicitly in the relevant tables and manifests. Data frequencies, segment lengths, evaluation-origin counts, and retained source identifiers accompany the complete dataset tables so that readers can distinguish file size from the number of forecast windows used in each metric.

B Architectures, optimization, and loss design

B.1 Exact architecture and initialization

Details for Section 3.

All main experiments use observation length $L = 96$ and horizon $H = 96$. Generic latent width is $d = 32$, except for the validation-selected Traffic width $d = 128$. The history scale in Equation equation 1 is the population scale

$$s_c = \sqrt{L^{-1} \sum_t (x_{tc} - \mu_c)^2 + 10^{-5}}.$$

The mean and scale are computed from history only; the output offset is the last observation rather than the history mean. Physical models instead apply training-only standardization before concatenating past output and past/future input.

Each JCPF branch has widths $2d \to 128 \to 128 \to d$, with tanh after the two hidden maps. The correction has widths $(B + 2)d \to 128 \to d$, with tanh after its hidden map. All output maps are linear and $B = 4$. Layer normalization has learned scale and bias and epsilon 10^{-5}. Input and intermediate linear maps use independent zero-mean Gaussian initialization with standard deviation equal to the inverse square root of their input width. Branch and correction output maps use standard deviation $0.01/\sqrt{\text{hidden width}}$, the forecast head uses $0.01/\sqrt{d}$, and biases start at zero. Source random-generator order is retained for reproducibility.

For PNEAR, hidden branch widths are 147, 158, and 170 at latent widths 32, 64, and 128, respectively. Its four branches have the same two-hidden-layer tanh form as JCPF but omit the correction. R2048 has one width-2,048 tanh hidden layer. Independent correction uses $2d \to 297 \to d$, giving 157,065 physical parameters, nine more than complete JCPF. No direct sum preserves JCPF widths, correction inputs, and parameter count. All its branches remain connected to the loss.

B.2 PatchTST configuration and known-input adapter

Details for Section 4.

The generic PatchTST implementation uses channel-independent shared patch projection, learned positional embeddings, a Transformer encoder, and a flattened linear prediction head. Patches have length 16 and stride 8, giving eleven patches from each length-96 channel history. The embedding width is 32; the encoder has two layers, four attention heads, feedforward width 64, GELU activation, and dropout 0.1. It uses PyTorch pre-LayerNorm encoder layers and a final LayerNorm. History mean/scale normalization and restoration are non-affine, and attention scores are not carried between layers. These are the recorded implementation choices; the model uses PatchTST's core patch-based channel-independent architecture without asserting identity to every default in the official implementation.

For physical tasks, the three-group known-input adapter separately patchifies past output, past input, and future input. Their eleven-patch sequences form 33 tokens for the encoder, followed by a flattened horizon head and last-output residual restoration. This gives each comparator the same known-input information. The external baseline source and paper links are provided in the reproducibility package.

B.3 A COMMON PREDICTION OBJECTIVE

Details for Section 3.

All models are trained to predict the observed future output, rather than a latent branch target. Let $z_{i,h,c}$ denote the target in training-standardized units for sampled window i, forecast position h, and output channel c, and let $\hat{z}_{i,h,c}(\theta)$ be the final model prediction after its output restoration. The minibatch objective is

$$\mathcal{L}(\theta;\mathcal{B}) = \frac{1}{|\mathcal{B}|HC} \sum_{i\in\mathcal{B}} \sum_{h=1}^{H} \sum_{c=1}^{C} \big(\hat{z}_{i,h,c}(\theta) - z_{i,h,c}\big)^2. \tag{16}$$

Here $H = 96$, while C is the number of predicted channels for generic forecasting and $C = 1$ for each physical system. Every horizon position and output channel has equal weight in this standardized coordinate system. Training normalization gives different raw-unit channels a common scale; it is fitted using only the training segment.

For the generic shared-interface models, history-window normalization occurs inside the network. The predicted residual is multiplied by that window's scale and added to the last observed value before Equation equation 16 is evaluated. The loss is therefore on the restored prediction in training-standardized data units, not on the internal normalized residual. The physical models likewise apply the loss to their restored output prediction, using the output scale fitted from the physical training segment. PatchTST uses its recorded prediction restoration but the same squared-output-error objective.

There is no extra penalty on branch independence, correction magnitude, physical energy, derivatives, or rollout stability. The direct sum and correction are learned jointly from the forecast error. SiLU, branch responses, and correction values are not separately supervised. AdamW weight decay is an optimizer operation (Loshchilov & Hutter, 2019), not an additional physical or interaction term in Equation equation 16. This choice makes the ablation experiments comparisons of the prediction map under a common output objective.

B.4 MINIBATCHES, NUMERICAL SETTINGS, AND VALIDATION SELECTION

Details for Section 4.

The structural models use AdamW with learning rate 0.003, $\beta = (0.9, 0.99)$, epsilon 10^{-8}, weight decay 10^{-4}, and gradient norm clipping at 5. The formal budget is 4,096 updates, with effective batch size 256 and microbatches of 64. Generic high-channel tasks also divide the channel dimension into chunks of at most 64. A chunk loss is multiplied by its fraction of the effective batch and its fraction of output channels before accumulation. This preserves the full objective's averaging convention rather than giving a small final chunk the weight of a full chunk.

Training uses FP32 with TF32 disabled. The formal seeds are 42, 43, and 44. A seed determines model initialization and training randomness, including the window sampler initialized with $10{,}000 +$ seed. Repeated runs consequently describe seed-dependent training variation, rather than

isolating initialization alone. Validation is performed every 64 updates, with the terminal checkpoint evaluated when required by the recorded stopping budget. The configured rule retains the checkpoint with the lowest selection-validation MSE and keeps the earlier incumbent in a tie. Test observations do not choose a replacement checkpoint.

The plotted training loss is the minibatch objective recorded during optimization. Validation loss is computed in evaluation mode on a fixed validation population. Differences between these curves can reflect their different populations and sampling, as well as training progress; a training minibatch value is not a full-training-set evaluation. The curve section distinguishes recorded values from any plotting aggregation and leaves unlogged stretches visible.

B.5 PatchTST optimizer schedules

Details for Section 4.

PatchTST retains its task-specific training recipe. Generic runs use Adam without weight decay or gradient clipping. ETTm1 and ETTm2 use a OneCycle schedule with warm-up fraction 0.4; Electricity and Traffic use warm-up fraction 0.2. The peak learning rate is 10^{-4}, with initial and final divisors 25 and 10^4. The remaining generic tasks use the recorded epoch-based decay schedule: the first effective epoch uses 4×10^{-6}, followed by the 10^{-4} base rate and subsequent factors of 0.9 according to the stored epoch rule. The effective epoch length is the ceiling of the number of legal training windows divided by 256.

The physical three-group PatchTST uses Adam and OneCycle with peak learning rate 10^{-4} and warm-up fraction 0.2, with no weight decay or clipping. Its recorded beta and learning-rate evolution are retained in the run configuration. These schedules are documented to reproduce the evaluated configurations; equal update budgets do not imply that all architectures use identical optimizers or that a separate exhaustive tuning exercise was performed for each model.

B.6 Resumption and computation budgets

Details for Section 4.

The configured resumption procedure retains optimizer and scheduler state, global random-number-generator state, sampler state, update count, and the incumbent validation checkpoint. Logs and budgets distinguish the completed update count from the number of updates executed after resumption. Some archived curves begin after an earlier training interval; absent earlier log points are not reconstructed.

A 1,024-update observation and a 4,096-update result answer different questions depending on their provenance. An endpoint score evaluates the model at that update; a prefix-best score evaluates the best validation checkpoint available by that update. A paired continuation can show how a fixed recorded training trajectory changes with additional updates. Comparisons that alter data splits, batch size, evaluation origins, optimization, or seed set are reported as separate protocol records and cannot isolate the effect of the update budget. The budget comparison below identifies which of these cases each row represents.

Training duration is a substantive part of a forecasting comparison. Assandri et al. (2023) study FEDformer, Autoformer, and Informer with extended training and modified learning-rate schedules. They report new best results relative to the Transformer results compared in their study in 50 of 72 benchmark settings, counted as nine datasets, four prediction lengths, and two metrics. This is approximately 69% of those settings, not 70% of forecasting architectures. Their modifications include moving from a short training/early-stopping recipe to much longer training and adjusting the learning-rate evolution. That result motivates documenting budgets carefully, but does not imply that extending our runs must improve every model or that a 1,024-versus-4,096 comparison demonstrates double descent.

C ACCURACY AND EFFICIENCY

C.1 METRICS AND ACCURACY TABLES

Details for Section 5.

For evaluation origins Ω, horizon $H = 96$, and C output channels, let $e_{i,h,c}$ be the prediction error. We report

$$\mathrm{MSE} = \frac{\sum_{i\in\Omega}\sum_{h=1}^{H}\sum_{c=1}^{C} e_{i,h,c}^2}{|\Omega|HC}, \qquad \mathrm{MAE} = \frac{\sum_{i\in\Omega}\sum_{h=1}^{H}\sum_{c=1}^{C} |e_{i,h,c}|}{|\Omega|HC}. \tag{17}$$

Generic errors use TRAIN-standardized channel values; physical raw-unit errors use the restored measured output. If its training scale is σ_y, then $\mathrm{MSE}^{raw} = \sigma_y^2\mathrm{MSE}^{norm}$ and $\mathrm{MAE}^{raw} = \sigma_y\mathrm{MAE}^{norm}$. Raw-unit values from different systems are not comparable measures of difficulty.

Entries are means of the three formal seeds. The displayed tables use only the common full horizon of 96. Stored shorter prefixes are cumulative parts of that same forecast, not independently trained prediction-length tasks, and are not averaged together here. Bold marks the lowest displayed error within a dataset/segment and metric; underlining marks the second distinct displayed value. Ties at display precision share the same rank. These descriptive rankings do not establish statistical significance.

Relative reductions compare seed-mean errors: $\Delta_M = 100(1 - \overline{\mathrm{MSE}}_{\mathrm{JCPF}}/\overline{\mathrm{MSE}}_M)$. G9 is the geometric mean of nine per-dataset ratios; G6 first combines the four ETT ratios, then equally weights six domains. The frozen values are retained with the short provenance note in Appendix C.4.

C.2 RECORDED GENERIC EFFICIENCY

Details for Section 5.

Parameter count measures trainable scalars. Batch latency is elapsed time per forward batch, not per example. The generic table reports archived measurements averaged over seeds, with batch-one and batch-64 results separated. Recorded training-event time can include loading, validation, checkpointing, and resumed execution; it is not necessarily the total from-scratch training time. Missing measurements remain marked as unavailable.

Table 11: Archived generic inference measurements averaged over the three recorded seeds. Latencies are per forward batch and peak allocated memory is reported in MiB. No training-time field or standard deviation is repeated here.

Dataset	Model	Params	B1 (ms)	B64 (ms)	Peak allocated (MiB)
ETTm1	CSF	6,176	0.239	0.247	68.3
ETTm1	JCPF	150,912	0.719	0.771	72.9
ETTm1	Pnear	150,428	0.648	0.714	72.6
ETTm1	R2048	204,864	0.349	0.368	81.4
ETTm1	PatchTST	51,936	0.646	0.585	112.7
ETTm2	CSF	6,176	0.249	0.262	68.3
ETTm2	JCPF	150,912	0.709	0.772	72.9
ETTm2	Pnear	150,428	0.633	0.716	72.6
ETTm2	R2048	204,864	0.349	0.366	81.4
ETTm2	PatchTST	51,936	0.637	0.577	112.7
ETTh1	CSF	6,176	0.215	0.241	67.4
ETTh1	JCPF	150,912	0.707	0.765	72.0
ETTh1	Pnear	150,428	0.650	0.675	71.7
ETTh1	R2048	204,864	0.343	0.369	80.5
ETTh1	PatchTST	51,936	0.635	0.553	111.8
ETTh2	CSF	6,176	0.246	0.254	67.4
ETTh2	JCPF	150,912	0.688	0.721	72.0
ETTh2	Pnear	150,428	0.647	0.712	71.7

Dataset	Model	Params	B1 (ms)	B64 (ms)	Peak allocated (MiB)
ETTh2	R2048	204,864	0.348	0.366	80.5
ETTh2	PatchTST	51,936	0.600	0.584	111.8
Exchange	CSF	6,176	0.217	0.220	67.7
Exchange	JCPF	150,912	0.598	0.688	72.6
Exchange	Pnear	150,428	0.530	0.818	72.3
Exchange	R2048	204,864	0.310	0.298	81.4
Exchange	PatchTST	51,936	0.483	0.525	118.3
Weather	CSF	6,176	0.211	0.208	75.7
Weather	JCPF	150,912	0.702	0.599	85.9
Weather	Pnear	150,428	0.627	0.551	84.9
Weather	R2048	204,864	0.356	0.349	109.1
Weather	PatchTST	51,936	0.540	0.874	207.0
Solar	CSF	6,176	0.585	0.627	138.5
Solar	JCPF	150,912	2.477	2.869	152.9
Solar	Pnear	150,428	1.709	1.563	150.5
Solar	R2048	204,864	0.889	1.006	217.3
Solar	PatchTST	51,936	1.621	4.449	524.8
Electricity	CSF	6,176	1.158	1.101	210.8
Electricity	JCPF	150,912	3.673	3.347	213.1
Electricity	Pnear	150,428	4.095	3.063	213.1
Electricity	R2048	204,864	1.694	2.047	260.2
Electricity	PatchTST	51,936	3.189	10.186	565.5
Traffic	CSF	24,704	3.526	2.548	434.9
Traffic	JCPF	403,584	9.086	8.106	440.7
Traffic	Pnear	403,552	8.677	7.616	440.7
Traffic	R2048	813,312	3.878	7.449	446.9
Traffic	PatchTST	51,936	7.612	27.651	686.4

C.3 PHYSICAL INFERENCE MEASUREMENTS

Details for Section 5.

The retained supplementary physical inference measurements use an NVIDIA GeForce RTX 5080, PyTorch 2.13.0/CUDA 13.0, and float32. Each main comparison uses its retained seed-42 checkpoint whose update matches the archived TEST identity. This check does not newly establish validation-best weight provenance. No training, checkpoint reselection, or replacement of frozen accuracy results is performed.

Batch one uses the first recorded TEST origin; batch 64 uses the first 64 consecutive legal origins. Each benchmark has 25 warm-ups and 100 timed inference-mode calls, with host wall-clock timing and CUDA synchronization after every call. The table reports median latency. The machine-readable ledger also retains p90, the 90th percentile. Throughput is batch size divided by median latency and is expressed in thousands of prediction windows per second.

Batch-64 allocated memory counts live tensor allocations; reserved memory counts the allocator's pool. Peak counters are reset for the benchmark, with loaded model and inputs included; one MiB is 2^{20} bytes. These quantities describe inference, not training memory. The full-TEST wall time separately includes input preparation, transfer, and forward execution over all retained segments in batches of 64, including Silverbox's separate overlapping arrow views. No targets are passed to the model and no TEST error is computed. These previously recorded timings remain separate from the archived generic timing protocol.

Table 12: Previously recorded physical inference measurements on the retained seed-42 checkpoints whose updates match the archived TEST identity. B1/B64 are medians after 25 warm-ups and 100 synchronized calls. Full TEST includes input preparation, transfer, and forward execution over every recorded TEST segment in B64, without target scoring.

System	Model	Params	B1 ms	B64 ms	B64 k/s	Peak A/R MiB	Full TEST s
Cascaded Tanks	CSF	12,320	0.053	0.079	811.7	32.3/34.0	0.003
Cascaded Tanks	JCPF	157,056	0.366	0.453	141.2	32.9/34.0	0.010
Cascaded Tanks	Pnear	156,572	0.324	0.409	156.3	32.9/34.0	0.009
Cascaded Tanks	R2048	211,008	0.140	0.151	424.7	34.5/36.0	0.005
Cascaded Tanks	PatchTST	120,224	0.584	0.571	112.0	34.9/36.0	0.011
Silverbox	CSF	12,320	0.054	0.071	897.0	32.3/34.0	0.320
Silverbox	JCPF	157,056	0.452	0.547	117.0	32.9/34.0	0.962
Silverbox	Pnear	156,572	0.325	0.408	156.8	32.9/34.0	0.864
Silverbox	R2048	211,008	0.135	0.153	418.4	34.5/36.0	0.441
Silverbox	PatchTST	120,224	0.590	0.570	112.3	34.9/36.0	1.012
Wiener–Hammerstein	CSF	12,320	0.055	0.067	958.1	32.3/34.0	0.269
Wiener–Hammerstein	JCPF	157,056	0.474	0.446	143.4	32.9/34.0	0.815
Wiener–Hammerstein	Pnear	156,572	0.327	0.403	158.8	32.9/34.0	0.736
Wiener–Hammerstein	R2048	211,008	0.129	0.152	422.4	34.5/36.0	0.375
Wiener–Hammerstein	PatchTST	120,224	0.587	0.582	109.9	34.9/36.0	0.864

C.4 CHECKPOINT-SELECTION PROVENANCE

Details for Section 5.

The reported generic results use validation-selected weights with traceable per-run checkpoint identities. Selection uses the complete recorded validation trajectory, with earlier incumbents retained on exact ties. TEST is observed only after the checkpoint is fixed.

D K=2 MACRO-COMPOSITION EXTENSION

D.1 FROZEN PROTOCOL AND FRESH REFERENCE

Details for Section 5.4.

This extension evaluates F16 and Wiener–Hammerstein (WH) under the frozen input, split, normalization, update, and validation-selection strategy. F16 uses a 1,024-step history and 256-step prediction, with its primary aggregate equally weighting three operating levels and three output channels; each row contains 372 TEST windows. WH uses its official TEST record with a 96-step history and 96-step prediction, giving 78,609 TEST windows per row. The shared K1 reference is trained once per dataset and is shared only by the two axes in this extension study. It is a fresh matched reference, not a replacement for or a numerical pool with the earlier WH main-matrix row.

The matrix has seven configurations, seeds 42/43/44, and 42 fresh 4,096-update FP32 runs with TF32 disabled. Every checkpoint is selected from its internal validation data before TEST is observed. K2 Joint and Independent are near-parameter alternatives that differ in whether the outer map reads $[\Delta_1; \Delta_2]$; Sum omits the outer correction. At the fixed-macro-width axis, each K=2 macro has width 128, so K1-to-Joint changes both capacity and compute. The Joint–Independent comparison is the response-access comparison on each axis.

Table 13: Macro-composition configurations. Outer width identifies the outer map family and hidden width; "none" means no outer correction. Parameter counts are taken from the recorded F16 and WH inference ledgers.

Axis	configuration	macro width	outer width	F16 params.	WH params.
fresh K1	K1	128	none	309,376	157,056
fixed budget	K2 Joint	73	G, 73	309,523	157,203
fixed budget	Independent	73	D, 121	309,507	157,187
fixed budget	Sum	78	none	309,948	157,628
fixed width	K2 Joint	128	G, 128	474,688	322,368
fixed width	Independent	128	D, 212	474,644	322,324
fixed width	Sum	128	none	454,048	301,728

Table 14: Fresh macro-composition normalized TEST MSE. Each row retains the three validation-selected seed observations and their arithmetic mean; no TEST result selects a checkpoint. K1 is listed once per dataset because it is shared by the two extension axes.

Dataset	axis	configuration	seed 42	seed 43	seed 44	mean MSE
F16	fresh K1	K1	0.413630	0.411502	0.415421	0.413518
F16	fixed budget	K2 Joint	0.414010	0.416662	0.415504	0.415392
F16	fixed budget	Independent	0.412316	0.416222	0.414984	0.414507
F16	fixed budget	Sum	0.413679	0.416055	0.422368	0.417367
F16	fixed width	K2 Joint	0.413410	0.414346	0.417113	0.414956
F16	fixed width	Independent	0.416543	0.417250	0.418449	0.417414
F16	fixed width	Sum	0.416281	0.413048	0.418850	0.416060
WH	fresh K1	K1	0.002117	0.002142	0.002197	0.002152
WH	fixed budget	K2 Joint	0.001817	0.001981	0.002093	0.001964
WH	fixed budget	Independent	0.001993	0.002195	0.002200	0.002129
WH	fixed budget	Sum	0.002225	0.002107	0.002020	0.002117
WH	fixed width	K2 Joint	0.001975	0.002070	0.002124	0.002056
WH	fixed width	Independent	0.002184	0.002051	0.002288	0.002174
WH	fixed width	Sum	0.002266	0.002194	0.002365	0.002275

D.2 INFERENCE EFFICIENCY

Details for Section 5.4.

Inference measurements use FP32 with TF32 disabled, 20 warm-up calls, and 100 CUDA-synchronized calls. Latency is the median forward-pass time; batch-64 throughput is prediction windows per second; peak memory is `torch.cuda.max_memory_allocated` after model and input allocation. Host-to-device transfer is excluded. The measurements describe inference only, not training cost.

Table 15: Fresh macro-composition inference measurements. B1/B64 are median latencies in milliseconds, B64 throughput is thousands of prediction windows per second, and memory is peak allocated MiB.

Dataset	configuration	parameters	B1 (ms)	B64 (ms)	B64 (k/s)	memory (MiB)
F16	fresh K1	309,376	0.336	0.430	148.9	35.7
F16	fixed budget K2 Joint	309,523	0.657	0.830	77.1	35.7
F16	fixed budget Independent	309,507	0.644	0.821	78.0	35.7
F16	fixed budget Sum	309,948	0.615	0.790	81.0	35.7
F16	fixed width K2 Joint	474,688	0.651	0.822	77.9	36.3
F16	fixed width Independent	474,644	0.651	0.829	77.2	36.3
F16	fixed width Sum	454,048	0.607	0.780	82.0	36.3

Dataset	configuration	parameters	B1 (ms)	B64 (ms)	B64 (k/s)	memory (MiB)
WH	fresh K1	157,056	0.336	0.420	152.3	32.9
WH	fixed budget K2 Joint	157,203	0.640	0.818	78.2	32.9
WH	fixed budget Independent	157,187	0.652	0.823	77.7	32.9
WH	fixed budget Sum	157,628	0.609	0.762	84.0	32.9
WH	fixed width K2 Joint	322,368	0.652	0.813	78.7	33.6
WH	fixed width Independent	322,324	0.658	0.820	78.1	33.6
WH	fixed width Sum	301,728	0.616	0.766	83.6	33.5

E VALIDATION EXPERIMENTS

E.1 TRAFFIC WIDTH SELECTION

Details for Section 4.

A validation-only pilot used seeds 48/49 to select one common latent width for the four shared-interface models. Their normalized geometric validation scores were 1.000000 at $d = 32$, 0.96906448 at $d = 64$, and 0.95547904 at $d = 128$. The $d = 64/d = 128$ ratio, 1.01422, exceeded the 1% simplicity tolerance, selecting $d = 128$ before TEST. The pilot changes representation size and parameter count together and does not establish an individually optimal width for each model.

Table 16: Traffic width-lock validation MSE. The common width was selected on validation only; no final TEST metric is present in this pilot.

Model	Width	Validation MSE
JCPF	32	0.411957
JCPF	64	0.401879
JCPF	128	0.400409
Pnear	32	0.419777
Pnear	64	0.413236
Pnear	128	0.40837
R2048	32	0.420396
R2048	64	0.409072
R2048	128	0.407248
CSF	32	0.497939
CSF	64	0.469917
CSF	128	0.453078

E.2 THE 1,024- AND 4,096-UPDATE BUDGETS

Details for Section 4.

The budget table reports archived scalar best-prefix validation MSE at 1,024 and 4,096 updates under the recorded training recipe. Each value is the best selection score available by that budget, averaged over the formal seeds; it is not a TEST score or a fixed-endpoint score. The corresponding checkpoint identities are supported by the complete available validation trajectories and recorded weight-validation checks.

For a complete common trajectory, best-prefix risk is nonincreasing by definition. A lower later prefix score indicates a better observed validation selection, not guaranteed TEST improvement. The comparison does not establish double descent.

Table 17: Validation-only prefix-best MSE at the recorded 1,024- and 4,096-update budgets. Values are three-seed means from the archived `metrics_validation.prefix_best` field, not TEST scores. Each row has all three archived seed records.

Dataset	Model	Prefix-best MSE at 1,024	Prefix-best MSE at 4,096
ETTm1	CSF	0.40826	0.40672
ETTm1	JCPF	0.40211	0.40211
ETTm1	Pnear	0.40583	0.40583
ETTm1	R2048	0.40346	0.40346
ETTm1	PatchTST	0.42685	0.40518
ETTm2	CSF	0.12704	0.12642
ETTm2	JCPF	0.12955	0.12955
ETTm2	Pnear	0.12982	0.12982
ETTm2	R2048	0.13000	0.13000
ETTm2	PatchTST	0.13416	0.12530
ETTh1	CSF	0.69460	0.68683
ETTh1	JCPF	0.69037	0.69037
ETTh1	Pnear	0.69375	0.69375
ETTh1	R2048	0.68888	0.68888
ETTh1	PatchTST	0.72824	0.72613
ETTh2	CSF	0.21394	0.21394
ETTh2	JCPF	0.22920	0.22920
ETTh2	Pnear	0.22765	0.22765
ETTh2	R2048	0.22575	0.22575
ETTh2	PatchTST	0.22586	0.22571
Exchange	CSF	0.12741	0.12741
Exchange	JCPF	0.13640	0.13640
Exchange	Pnear	0.13701	0.13701
Exchange	R2048	0.13361	0.13361
Exchange	PatchTST	0.14470	0.14467
Weather	CSF	0.50063	0.45161
Weather	JCPF	0.43854	0.43405
Weather	Pnear	0.44306	0.43367
Weather	R2048	0.44441	0.43362
Weather	PatchTST	0.45833	0.44437
Solar	CSF	0.19413	0.17885
Solar	JCPF	0.17474	0.16385
Solar	Pnear	0.17404	0.16526
Solar	R2048	0.17374	0.16625
Solar	PatchTST	0.22757	0.20028
Electricity	CSF	0.18347	0.17854
Electricity	JCPF	0.16933	0.15716
Electricity	Pnear	0.17182	0.16092
Electricity	R2048	0.17047	0.16035
Electricity	PatchTST	0.20856	0.18522
Traffic	CSF	0.46821	0.45150
Traffic	JCPF	0.41712	0.39914
Traffic	Pnear	0.42575	0.40595
Traffic	R2048	0.42284	0.40600
Traffic	PatchTST	0.57210	0.46614

E.3 SUPPLEMENTARY SYNTHETIC NEGATIVE CONTROL

This is an independent supplementary exploration, rather than detailed data for a main-text result.

For $x_{t+1} = 2x_t - x_{t-1} - 0.05^2(x_t + \kappa x_t^3)$ with $\kappa \in \{0, 1, 4\}$ and no drive or noise, three data replicates and two initializations per replicate give primary CSF–R2048 risk differences -0.000945, 0.003647, and 0.019691 at 4,096 updates. All replicate-level intervals cross zero. At $\kappa = 4$, PNEAR

is slightly better than JCPF under endpoint and prefix-best comparison. This is a local negative control rather than a monotonicity or coupling claim.

E.4 HISTORICAL SMALL-BASELINE SCREENING

This separate historical screen informed the PatchTST comparator in Section 4; it is not a main-matrix result. It covers one completed seed (20260904) for each of eight profiles on the nine $L = H = 96$ datasets. PatchTST has the lowest mean rank for both MSE (2.00; first on six datasets) and MAE (2.11; first on five datasets), making it a strong overall reference within this screening. D-LinOSS is a local equation-level PyTorch port and A-USSM an independent PyTorch reference implementation; neither is claimed equivalent to an upstream implementation. Three custom Flow candidates are intentionally excluded from this baseline table.

The historical screen uses chronological ETT 12/4/4-month and other 70/10/20% splits, shuffled training minibatches, every legal TEST sliding window (2,785 for ETTh1 and 11,425 for ETTm1), and a one-seed AdamW recipe with at most 1,000 updates. The paper's PatchTST table instead uses seeds 42–44, recorded task-specific Adam schedules, and stride-24 TEST origins (117 for ETTh1 and 477 for ETTm1). These separate runs, evaluation populations, model configurations, and optimization protocols explain why the two tables are not directly pooled; the historical metadata does not support attributing differences to random split assignment.

Table 18: Historical $H = 96$ screen of eight completed profiles on nine datasets. MSE and MAE are globally train-standardized recorded values; lower is better. Each cell comes from one historical seed (20260904) and its own recorded protocol, not a matched re-evaluation. D-LinOSS and A-USSM are port/reference implementations; custom Flow candidates are intentionally excluded. Bold marks the lowest and underlining the second distinct value within a dataset–metric row.

Dataset	metric	DLinear	PatchTST	iTransformer	GRU	D-LinOSS	A-USSM	Persist.	Season.
ETTh1	MSE	0.3828	**0.3802**	0.3857	0.4567	0.4364	0.4592	1.2944	0.5122
ETTh1	MAE	**0.3955**	0.4029	0.4029	0.4542	0.4470	0.4644	0.7132	0.4333
ETTh2	MSE	0.3231	**0.2961**	0.3086	0.3151	0.3253	0.3358	0.4317	0.3905
ETTh2	MAE	0.3799	**0.3453**	0.3556	0.3617	0.3661	0.3753	0.4216	0.3802
ETTm1	MSE	**0.3418**	0.3521	0.3507	0.4799	0.4192	0.3701	1.2139	0.4227
ETTm1	MAE	**0.3717**	0.3765	0.3784	0.4490	0.4236	0.4009	0.6648	0.3865
ETTm2	MSE	0.1889	**0.1788**	0.1881	0.2019	0.1895	0.1943	0.2665	0.2631
ETTm2	MAE	0.2848	**0.2628**	0.2736	0.2872	0.2720	0.2761	0.3278	0.3005
Electricity	MSE	0.1980	**0.1923**	0.2053	0.3083	0.3145	0.3028	1.5878	0.3211
Electricity	MAE	0.2838	**0.2729**	0.2926	0.3828	0.3905	0.3816	0.9455	0.3258
Exchange	MSE	**0.0783**	0.0853	0.0889	0.0808	0.0844	0.0940	0.0811	0.0811
Exchange	MAE	0.1986	0.2039	0.2106	**0.1960**	0.2011	0.2121	0.1964	0.1964
Solar	MSE	0.2885	**0.2600**	0.2748	0.3620	0.2914	0.2689	1.5391	1.9764
Solar	MAE	0.3759	**0.2881**	0.3098	0.4073	0.3417	0.3242	0.8155	1.0007
Traffic	MSE	0.6515	**0.5250**	0.5663	0.8839	1.2226	1.1474	2.7145	1.2171
Traffic	MAE	0.3999	**0.3420**	0.3866	0.5303	0.5373	0.5065	1.0772	0.4975
Weather	MSE	0.1938	0.1838	0.1892	0.2058	0.1697	**0.1678**	0.2591	0.3490
Weather	MAE	0.2534	0.2241	0.2308	0.2421	0.2155	**0.2129**	0.2542	0.3329

F SELECTED LEARNING CURVES AND PREDICTIONS

F.1 TRAINING AND VALIDATION TRAJECTORIES

Details for Section 5.

Figures below retain Traffic, Cascaded Tanks, and WH. Training points are sampled minibatch MSE; validation points use the fixed selection population. Lines show the three-seed median and bands the minimum–maximum range, not confidence intervals. Both losses use training-standardized units, but their different populations make their vertical separation unsuitable as a direct generalization-gap estimate. Missing log events are not reconstructed.

Traffic and WH retain later validation improvement, whereas Cascaded Tanks shows rising selection loss for JCPF and PNEAR despite declining sampled training loss. These observed trajectories explain why checkpoint selection matters; they do not establish double descent or guarantee that longer training improves TEST accuracy.

Table 5: Frozen generic dataset overview. Train, validation, and TEST are contiguous chronological partitions; their row counts are reported in that order. Generic validation and formal TEST use stride 24 with the terminal legal origin retained.

Dataset	Cadence	C	Rows	Train/val/TEST rows	TEST origins
ETTm1	15 min	7	69,680	34,560/11,520/23,600	981
ETTm2	15 min	7	69,680	34,560/11,520/23,600	981
ETTh1	1 h	7	17,420	8,640/2,880/5,900	243
ETTh2	1 h	7	17,420	8,640/2,880/5,900	243
Exchange	1 day	8	7,588	5,311/760/1,517	61
Weather	10 min	21	52,696	36,887/5,270/10,539	437
Solar	not retained	137	52,560	36,792/5,256/10,512	435
Electricity	1 h	321	26,304	18,412/2,632/5,260	217
Traffic	1 h	862	17,544	12,280/1,756/3,508	144

Table 6: Frozen physical-system train, selection, assessment, and formal TEST records. Segment-origin counts are copied from retained final records; this table does not load physical TEST values.

System	Δt (s)	Train/sel/assess rows	Formal TEST segments: origins
Cascaded Tanks	4	614/205/205	official_test: 833
Silverbox	0.0016384	39,037/13,012/13,013	multisine_test: 21497; arrow_full: 40284; arrow_no_extrapolation: 31809
Wiener–Hammerstein	1.9531e-05	60,000/20,000/20,000	official_test: 78609

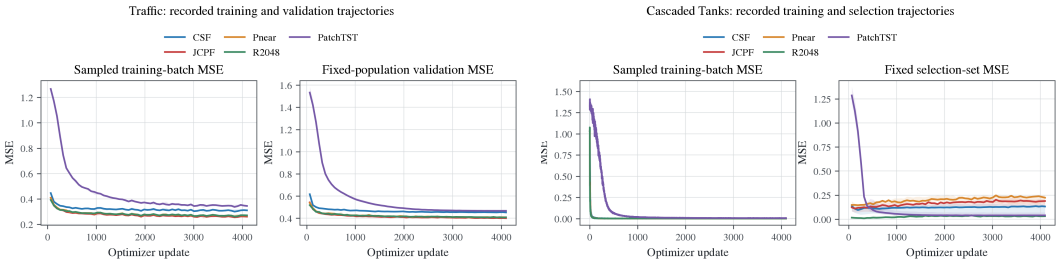


Figure 5: Traffic (left) and Cascaded Tanks (right): sampled training MSE and fixed-population selection MSE. Lines are three-seed medians; bands are the seed minimum–maximum range. JCPF is red.

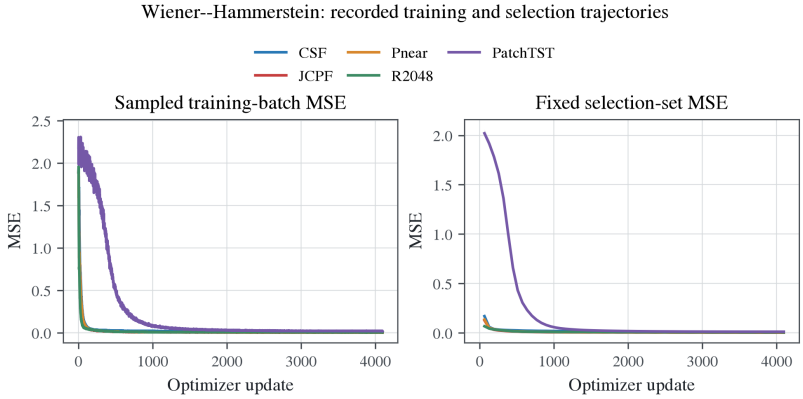


Figure 6: WH: sampled training MSE and fixed-population selection MSE. Lines are three-seed medians; bands are the seed minimum–maximum range. JCPF is red.

Table 7: Trainable parameters in the evaluated configurations.

Setting	CSF	JCPF	PNEAR	R2048	PatchTST
Generic $d = 32$	6,176	150,912	150,428	204,864	51,936
Traffic $d = 128$	24,704	403,584	403,552	813,312	51,936
Physical $d = 32$	12,320	157,056	156,572	211,008	120,224

Table 8: Generic formal TEST MSE and MAE at $H = 96$ in training-standardized units. Values are three-seed means; lower is better. Bold marks the lowest displayed value within a row and metric, and underlining marks the second distinct displayed value. Display-precision ties share a rank.

	CSF		JCPF		Pnear		R2048		PatchTST	
Dataset	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	0.398	0.408	0.400	0.406	0.400	0.407	**0.397**	**0.404**	0.404	0.408
ETTm2	**0.155**	**0.255**	0.158	**0.255**	0.158	0.256	0.158	0.256	**0.155**	0.257
ETTh1	0.466	**0.450**	0.467	0.458	0.486	0.467	**0.461**	0.455	0.487	0.462
ETTh2	0.246	0.328	0.251	0.331	0.250	0.331	0.252	0.332	**0.239**	**0.322**
Exchange	0.087	0.203	**0.085**	**0.201**	0.086	0.202	**0.085**	**0.201**	0.092	0.214
Weather	0.187	0.226	**0.177**	**0.218**	0.178	0.219	0.181	0.220	0.188	0.227
Solar	0.246	0.284	0.221	0.267	0.219	**0.266**	**0.215**	0.267	0.293	0.311
Electricity	0.193	0.277	**0.173**	**0.262**	0.176	0.266	0.176	0.267	0.202	0.285
Traffic	0.518	0.340	**0.454**	**0.301**	0.464	0.313	0.465	0.315	0.554	0.364

Table 9: Physical-system formal TEST MSE and MAE at $H = 96$ in each system's raw output unit. Values are three-seed means. Rankings compare only models within the same system and TEST segment; raw values are not comparable across systems. Bold and underlining follow the convention of Table 8.

		CSF		JCPF		Pnear		R2048		PatchTST	
System	TEST segment	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Cascaded Tanks	official TEST	2.170	1.103	1.684	0.960	1.764	0.988	**1.503**	**0.910**	2.475	1.214
Silverbox	multisine	1.62e-4	0.010	**4.24e-5**	**0.005**	5.24e-5	0.006	4.85e-5	**0.005**	1.09e-4	0.008
Silverbox	arrow full	2.90e-4	0.011	**7.29e-5**	**0.005**	9.19e-5	0.006	9.15e-5	0.006	3.47e-4	0.012
Silverbox	arrow core	1.19e-4	0.008	**2.97e-5**	**0.004**	3.90e-5	0.005	3.46e-5	**0.004**	1.18e-4	0.008
Wiener–Hammerstein	official TEST	4.38e-4	0.015	**1.27e-4**	**0.009**	1.89e-4	0.011	2.39e-4	0.012	0.001	0.026

Table 10: Physical ablation formal TEST MSE and MAE at $H = 96$ in training-standardized units. Values are three-seed means. The JCPF row uses the matched main-matrix checkpoint; rankings compare the three variants only within a system and TEST segment.

		JCPF		Independent correction		No direct sum	
System	TEST segment	MSE	MAE	MSE	MAE	MSE	MAE
Silverbox	multisine	**0.014**	**0.094**	0.015	0.097	0.017	0.103
Silverbox	arrow full	**0.024**	**0.098**	0.028	0.107	0.029	0.106
Silverbox	arrow core	**0.010**	**0.072**	0.012	0.080	0.012	0.078
Wiener–Hammerstein	official TEST	**0.002**	**0.036**	0.003	0.041	0.003	0.039

F.2 FIXED-WINDOW PREDICTIONS

Details for Section 5.

The Traffic, Cascaded Tanks, and WH illustrations compare the target with CSF, JCPF, and PatchTST on the same 96-step window using seed 42. The origin is the middle index of the recorded legal-origin list, $\lfloor (|\Omega| - 1)/2 \rfloor$ in zero-based indexing. Traffic uses the final numeric source column; each physical system uses its single output. The window and channel are fixed by index, not chosen

by prediction error. The original information contract is unchanged, including known future driving inputs for the physical tasks.

Values are shown in source output units. These local examples reveal amplitude and timing behavior; the aggregate tables remain the quantitative model comparison. Red denotes JCPF throughout the selected figures.

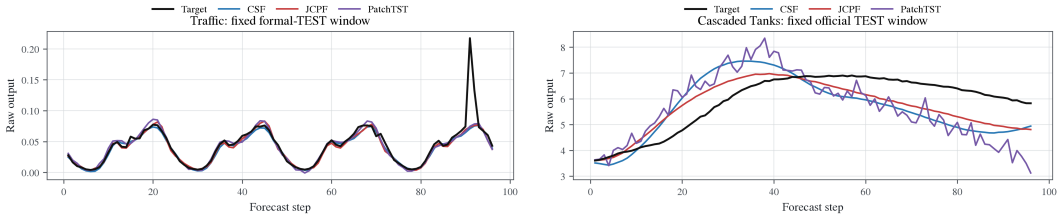


Figure 7: Traffic (left) and Cascaded Tanks (right): observed output and CSF, JCPF, and PatchTST predictions on fixed seed-42 TEST windows in source output units. JCPF is red.

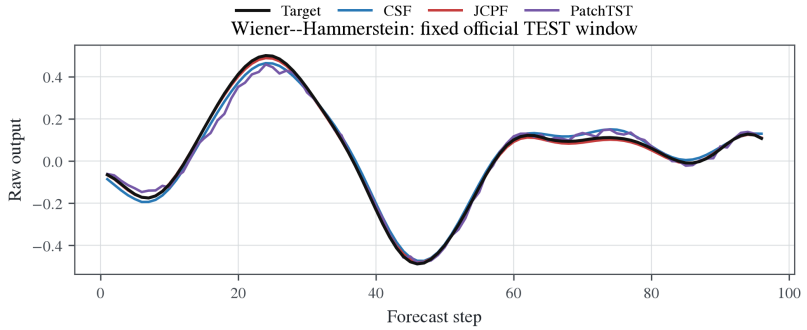


Figure 8: WH: observed output and CSF, JCPF, and PatchTST predictions on the fixed seed-42 TEST window in source output units. JCPF is red.

G RESPONSE-ACCESS SUPPLEMENT AND CORRECTION DIAGNOSTICS

This protocol and diagnostic record gives details for the correction-representation comparison in Section 5.3 and the fitted-path evaluation in Section 4.

The response-access supplement retrains exactly six Cagg models: Silverbox and WH with seeds 42, 43, and 44. It preserves the shared adapter, LayerNorm, direct sum, four width-128 branches, optimizer, sampler, update budget, and validation selection rule. Only the correction input changes from $[n; q; r_1; \ldots; r_4]$ to $[n; q; a]$, namely

$$c_{\text{agg}} = \text{MLP}_{\text{agg}}([n; q; a]). \tag{18}$$

To keep the budget matched, its tanh hidden width is 223 rather than 128, yielding 157,023 versus 157,056 parameters. Shared initialization tensors are identical at a common seed; the differently shaped correction matrices are not. Candidate TEST values were read only after the completed six-row validation lock.

Silverbox favors the individual-response implementation in all three seeds; WH has a lower aggregate-conditioned mean because of seed 42, whereas its other two seeds are slightly higher. Under the evaluated datasets and training protocol, branch-conditioned and aggregate-conditioned correction attain comparable performance, without a stable advantage for either form. These data do not select Cagg as a replacement main model, establish statistical equivalence, or make individual-response access a universal requirement.

For a fitted checkpoint, let y_{add} be its decoded direct path, δ the decoded selected correction, and $e_{\text{add}} = y - y_{\text{add}}$. The diagnostic reports

$$\text{gain} = \text{MSE}_{\text{add}} - \text{MSE}_{\text{full}} = 2\,\text{mean}(e_{\text{add}}\delta) - \text{mean}(\delta^2). \tag{19}$$

Table 19: Cagg versus JCPF on validation-selected checkpoints. Values are primary-TEST normalized MSE and MAE. Cagg uses 157,023 parameters and JCPF uses 157,056; each Cagg row used 4,096 updates.

System	Seed	Cagg MSE	JCPF MSE	Cagg MAE	JCPF MAE
Silverbox multisine	42	0.014178	0.014035	0.093923	0.093423
Silverbox multisine	43	0.014267	0.014207	0.094137	0.093900
Silverbox multisine	44	0.014275	0.014237	0.094206	0.094130
WH official TEST	42	0.001947	0.002117	0.033928	0.035660
WH official TEST	43	0.002149	0.002142	0.035608	0.035619
WH official TEST	44	0.002237	0.002197	0.036389	0.036255

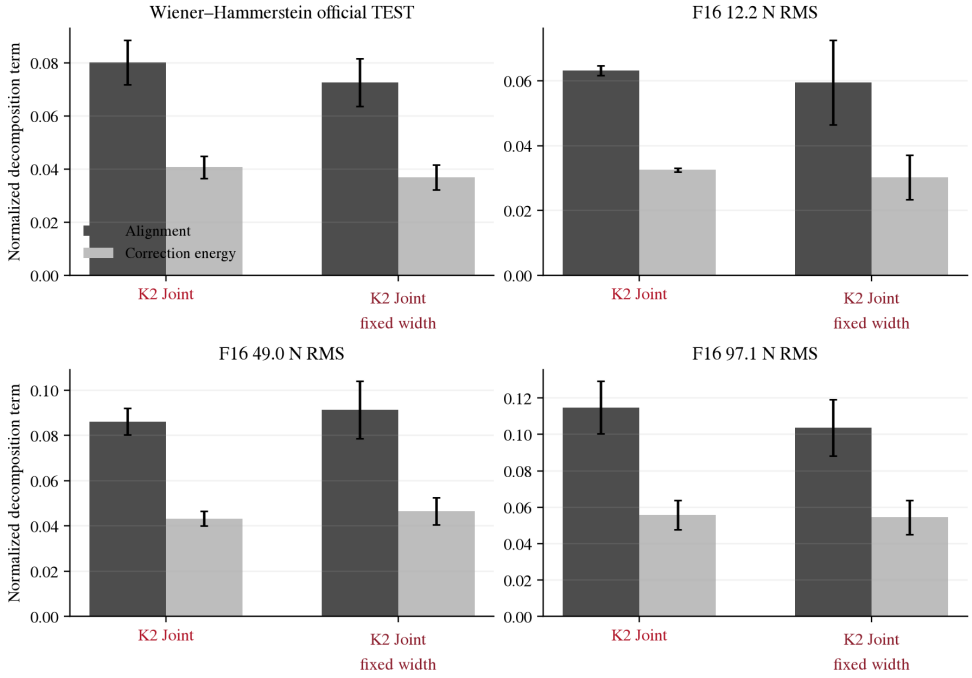


Figure 9: Prediction-space alignment and correction energy after removing only outer G from the fitted K2 macro models. Each panel is a TEST system or F16 excitation level; error bars are sample SD across three seeds. These are descriptive co-adapted removal terms, not retrained architecture comparisons.

For original physical JCPF, only C is removed. For macro models, only outer G is removed while both complete JCPF increments remain. Thus this is a within-checkpoint removal diagnostic, not the performance of a separately trained no-correction architecture; gains across independently trained maps are not an accuracy ranking.

All 18 prescribed original and macro checkpoints have positive whole-split selection and primary-TEST gains. The F16 macro gain is positive after aggregation for every output channel, although not every individual forecast step is positive. Figure 9 shows the alignment and correction-energy terms for both macro resource axes. Error bars are sample standard deviations across the three seeds, not confidence intervals. The full per-step, channel, split, and F16-level records remain in the reproducibility package.