Do We Still Need Trend Decomposition? Patch-State and Delta-Dynamics Modeling for Time Series Forecasting

Anonymous authors
Paper under double-blind review

Abstract

Time-series forecasting uses trends in historical observations to inform the extrapolation of future trajectories. Decomposition-based forecasting organizes this process around trend and seasonal components, separating sustained changes from recurrent fluctuations and learning their temporal dependencies separately. This component-wise modeling paradigm also extends to patch-based forecasting, where consecutive observations are encoded as local trajectory tokens. However, we find that raw patches already preserve the local tendencies and recurring patterns targeted by decomposition, revealing substantial overlap in their representational roles. Moreover, similar local tendencies and shapes do not uniquely determine the magnitude and evolution of successive changes, motivating explicit modeling of transition dynamics alongside trajectory structure. We therefore introduce a state–dynamics forecasting paradigm and instantiate it as the Patch-State and Delta-Dynamics Network (PaDNet). Delta-Dynamics explicitly represents the direction and magnitude of adjacent changes and compares increments across steps to characterize their evolution. In parallel, Patch-State regularizes trajectory encoding to capture the overall temporal tendency and structure of local trajectories. Together, these representations condition forecasting on both local trajectory states and their transition dynamics. Extensive experiments on time-series datasets demonstrate the effectiveness of PaDNet, achieving average reductions of up to 7.3% in MSE and 5.1% in MAE over the best-performing state-of-the-art baseline.

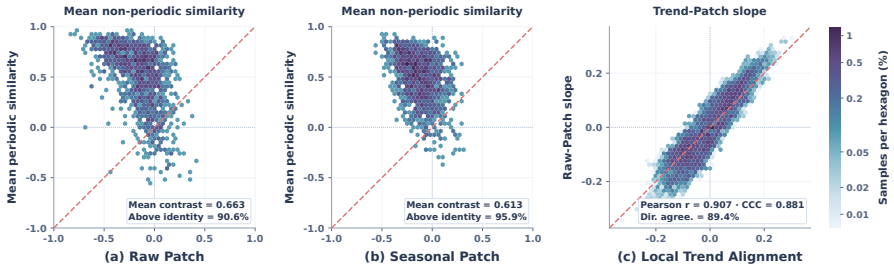


Figure 1: Patch-level periodic contrast and trend alignment. (a,b) Raw and seasonal patches exhibit broadly similar periodic versus non-periodic similarity patterns. (c) Raw and EMA trend patches show strong local slope agreement. Dashed lines indicate equality; colors denote sample percentages per hexagonal bin. Calculation details are provided in Appendix.

1 Introduction

Time-series forecasting (Montgomery et al., 2015) predicts future observations from historical context. This requires characterizing both the observed states and their evolution over time, with trends capturing their sustained tendency. To exploit this persistent temporal structure, explicit trend modeling has become a widely adopted strategy in time-series forecasting.

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.

A common paradigm for such modeling is trend–seasonal decomposition, which partitions a sequence into trend and seasonal components for separate modeling (Wu et al., 2021; Zhou et al., 2022). This paradigm is motivated by the observation that sustained changes and recurrent fluctuations can obscure one another, making their separation beneficial for identifying component-specific temporal dependencies. To enrich local context while reducing attention costs, prior work has introduced patching, which jointly encodes consecutive observations as segment-level tokens, shifting the representation unit from individual observations to local trajectories (Nie et al., 2023; Lee et al., 2024; Luo & Wang, 2024b; Zhang et al., 2024). Recent approaches (Liu, 2026; Wu et al., 2024; Zhong et al., 2026) integrate the decomposition paradigm into patch-based forecasting to combine the benefits of component separation and local context aggregation (Stitsyuk & Choi, 2025). This integration raises a question about their representational complementarity: does explicit decomposition offer additional benefits for capturing local tendencies and recurrent patterns, or do these benefits partly overlap with those already afforded by patching?

As visualized in Figure 1, seasonal extraction does not substantially enhance the periodic contrast of raw patches, while raw and trend patches exhibit strong local slope agreement, indicating considerable representational overlap between explicit decomposition and patching. From a representational perspective, decomposition-based (Wu et al., 2021) and patch-based (Nie et al., 2023) representations share a limitation in modeling temporal changes: both organize historical observations around trajectory-level structures, leaving transitions between successive observations implicitly encoded. Crucially, similarity in local tendencies and overall shapes does not imply equivalence in transition dynamics: it specifies neither the magnitude of individual changes nor how those changes strengthen or weaken over time. These distinctions are essential to characterizing how a trajectory evolves, yet remain implicit in trajectory-centered representations, exposing a gap between modeling trajectory structure and explicitly characterizing its transition dynamics.

To address this representational gap, we introduce a state–dynamics forecasting paradigm that treats the local trajectory state and its transition dynamics as two explicit conditioning variables, instantiated by the Patch-State and Delta-Dynamics Network (PaDNet). Existing trajectory-centered representations leave successive changes implicit; Delta-Dynamics therefore explicitly preserves the direction and magnitude of adjacent changes and compares increments across steps to characterize how the transition process evolves. Meanwhile, although raw patches retain local tendencies and recurring patterns, patchification alone imposes no structural constraint on how these patterns are encoded as trajectory states. Patch-State therefore regularizes trajectory encoding to favor a compact description of overall local tendencies and recurring structures. By jointly modeling these two aspects, PaDNet conditions forecasting on both what state a local trajectory exhibits and how that state is evolving.

Our contributions are summarized as follows:

- We revisit trend–seasonal decomposition in patch-based forecasting and reveal substantial overlap between the temporal structures exposed by explicit decomposition and those preserved by raw patches. We identify that trajectory-centered representations leave increment patterns and their evolution only implicitly encoded.
- We introduce a state–dynamics forecasting paradigm and instantiate it as the Patch-State and Delta-Dynamics Network, which jointly models compact Patch-States and explicit Delta-Dynamics.
- Extensive experiments on multiple real-world time-series datasets demonstrate the strong forecasting performance of PaDNet.

2 Preliminaries

2.1 Forecasting Problem

Let t denote the current time index. Given a batch of historical multivariate sequences $\mathbf{X} = \mathbf{X}_{t-L+1:t} \in \mathbb{R}^{B\times L\times C}$, time-series forecasting aims to estimate the subsequent H obser-

vations:

$$\widehat{\mathbf{Y}} = f_\theta(\mathbf{X}) \in \mathbb{R}^{B\times H\times C}, \qquad \mathbf{Y} = \mathbf{X}_{t+1:t+H}. \tag{1}$$

Here, B is the batch size, L is the look-back length, H is the forecasting horizon, and C is the number of variables. $\mathbf{Y} \in \mathbb{R}^{B\times H\times C}$ denotes the ground-truth future sequence, $\widehat{\mathbf{Y}}$ is its prediction, and f_θ is a forecasting model with learnable parameters θ.

2.2 Temporal Patchification

Under channel-independent modeling, the variable dimension is folded into the batch dimension:

$$\mathbf{X}^{\mathrm{CI}} = \mathrm{CI}(\mathbf{X}) \in \mathbb{R}^{M\times L}, \qquad M = BC, \tag{2}$$

where $\mathrm{CI}(\cdot)$ denotes the reshaping operation and M is the number of resulting univariate sequences.

For the m-th sequence, the n-th Patch with length P and stride S is defined as $\mathbf{p}_{m,n} = [x_{m,nS+1}, x_{m,nS+2}, \ldots, x_{m,nS+P}]^\top \in \mathbb{R}^P$, where $m \in \{1,\ldots,M\}$ indexes the univariate sequence and $n \in \{0,\ldots,N-1\}$ indexes the Patch position. The number of patches is $N = \left\lfloor \frac{L'-P}{S} \right\rfloor + 1$, where L' denotes the effective sequence length after optional end padding. Collecting the patches yields

$$\mathbf{X}^{\mathrm{patch}} = \mathrm{Patch}_{P,S}\left(\mathbf{X}^{\mathrm{CI}}\right) \in \mathbb{R}^{M\times N\times P}, \tag{3}$$

where $\mathrm{Patch}_{P,S}(\cdot)$ denotes Patch extraction with length P and stride S.

2.3 Component Decomposition and State–Dynamics Representation

Definition 2.1 (Trend–Seasonal Decomposition). For the m-th univariate sequence $\mathbf{x}_m \in \mathbb{R}^L$, a decomposition-based representation is defined as $\Phi_{\mathrm{dec}}(\mathbf{x}_m) = (\mathbf{x}_m^{\mathrm{tr}}, \mathbf{x}_m^{\mathrm{se}})$, where $\mathbf{x}_m^{\mathrm{tr}} = \mathcal{M}_\kappa(\mathbf{x}_m)$ and $\mathbf{x}_m^{\mathrm{se}} = \mathbf{x}_m - \mathbf{x}_m^{\mathrm{tr}}$. Here, $\mathcal{M}_\kappa(\cdot)$ is a smoothing operator with temporal scale κ. $\mathbf{x}_m^{\mathrm{tr}} \in \mathbb{R}^L$ denotes the trend component, while $\mathbf{x}_m^{\mathrm{se}} \in \mathbb{R}^L$ denotes the remaining seasonal or residual component. The two components satisfy $\mathbf{x}_m = \mathbf{x}_m^{\mathrm{tr}} + \mathbf{x}_m^{\mathrm{se}}$.

Definition 2.2 (Patch-State and Increment-Dynamics Representation). Given the Patch sequence defined in Section 2.2, we represent $\mathbf{x}_m$ using $\Phi_{\mathrm{SD}}(\mathbf{x}_m) = (\mathbf{P}_m, \boldsymbol{\delta}_m)$, where $\mathbf{P}_m = [\mathbf{p}_{m,0}, \mathbf{p}_{m,1}, \ldots, \mathbf{p}_{m,N-1}]^\top \in \mathbb{R}^{N\times P}$ is the Patch-State sequence, and $\boldsymbol{\delta}_m = [\delta_{m,2}, \delta_{m,3}, \ldots, \delta_{m,L}]^\top \in \mathbb{R}^{L-1}$ is the increment-dynamics sequence, with $\delta_{m,j} = x_{m,j} - x_{m,j-1}$. Here, $\delta_{m,j} \in \mathbb{R}$ describes the direction and magnitude of the transition from position $j-1$ to j.

The pair $(\mathbf{P}_m, \boldsymbol{\delta}_m)$ represents local states and successive transitions rather than additive components.

3 Theoretical Motivation

We characterize the predictive expressiveness of raw patches and the contribution of explicit dynamics conditioning. Let $\mathbf{x} \in \mathbb{R}^L$ denote the historical sequence and $\mathbf{y} \in \mathbb{R}^H$ the forecasting target. Complete proofs appear in Appendix D.

Assumptions. All random vectors have finite second moments. The decomposition analysis assumes complete patch coverage, fixed linear decomposition, and unrestricted affine predictors. For the dynamics analysis, let $\mathbf{u}, \mathbf{v} \in \mathbb{R}^H$ denote the Patch-State and Delta-Dynamics responses. Branch mappings are fixed, and both predictors share the same state response. For affine fusion, define the following quantities using centered versions of $\mathbf{y}, \mathbf{u}, \mathbf{v}$. Let $\mathbf{A}_{\mathrm{s}}\mathbf{u}$ and $\mathbf{K}\mathbf{u}$ be the least-squares projections of $\mathbf{y}$ and $\mathbf{v}$ onto the state response:

$$\mathbf{e} = \mathbf{y} - \mathbf{A}_{\mathrm{s}}\mathbf{u}, \qquad \mathbf{v}_\perp = \mathbf{v} - \mathbf{K}\mathbf{u}, \qquad \mathbf{C} = \mathbb{E}[\mathbf{e}\mathbf{v}_\perp^\top]. \tag{4}$$

For strict improvement under affine fusion, we assume residual predictive relevance: the dynamics response retains a component outside the linear span of the state response that is correlated with the state-only prediction residual, i.e., $\mathbf{C} \neq \mathbf{0}$.

Predictive expressiveness of raw patches. Let $\mathbf{p} = \mathbf{Q}\mathbf{x}$ stack the raw patches. Complete coverage gives a reconstruction matrix $\mathbf{J}$ with $\mathbf{J}\mathbf{Q} = \mathbf{I}_L$. For the decomposition operator $\mathbf{T}$, an affine component-wise predictor satisfies

$$f_{\text{dec}}(\mathbf{x}) = \mathbf{A}\mathbf{T}\mathbf{x} + \mathbf{B}(\mathbf{I}_L - \mathbf{T})\mathbf{x} + \mathbf{b} = \big[\mathbf{A}\mathbf{T} + \mathbf{B}(\mathbf{I}_L - \mathbf{T})\big]\mathbf{J}\mathbf{p} + \mathbf{b}. \tag{5}$$

Conversely, any affine predictor $\mathbf{W}\mathbf{x} + \mathbf{b}$ is recovered by setting $\mathbf{A} = \mathbf{B} = \mathbf{W}$. Thus, the predictor classes induced by raw patches and decomposed components coincide:

$$\mathcal{F}_{\text{patch}} = \mathcal{F}_{\text{dec}}, \qquad \inf_{f \in \mathcal{F}_{\text{patch}}} \mathcal{R}(f) = \inf_{f \in \mathcal{F}_{\text{dec}}} \mathcal{R}(f), \tag{6}$$

where $\mathcal{R}(f) = \mathbb{E}\|\mathbf{y} - f(\mathbf{x})\|_2^2$. Explicit decomposition introduces no additional predictive expressiveness within this class. Its practical effects depend on architectural constraints, regularization, and optimization.

Conditional predictive value of dynamics. The preceding equivalence concerns raw observations; encoding can restrict the temporal variations accessible to prediction. The conditional future means satisfy

$$\boldsymbol{\mu}_{\text{s}}(\mathbf{u}) = \int \boldsymbol{\mu}_{\text{sd}}(\mathbf{u}, \mathbf{b})\, \Pi_{\mathbf{v}|\mathbf{u}}(d\mathbf{b} \mid \mathbf{u}), \tag{7}$$

where $\boldsymbol{\mu}_{\text{s}}(\mathbf{u}) = \mathbb{E}[\mathbf{y} \mid \mathbf{u}]$, $\boldsymbol{\mu}_{\text{sd}}(\mathbf{u}, \mathbf{v}) = \mathbb{E}[\mathbf{y} \mid \mathbf{u}, \mathbf{v}]$, and $\Pi_{\mathbf{v}|\mathbf{u}}$ is the conditional distribution of the dynamics response.

Under squared loss, conditional-expectation orthogonality yields

$$\mathcal{R}^*_{\mathbf{u}} - \mathcal{R}^*_{\mathbf{u},\mathbf{v}} = \mathbb{E}\|\boldsymbol{\mu}_{\text{sd}}(\mathbf{u}, \mathbf{v}) - \boldsymbol{\mu}_{\text{s}}(\mathbf{u})\|_2^2 \geq 0, \tag{8}$$

where $\mathcal{R}^*_{\mathbf{z}}$ is the minimum population risk over measurable predictors of $\mathbf{z}$. The gain measures conditional-mean variation resolved by dynamics conditioning but averaged out when prediction depends only on the state response.

Realizable gain under response-level fusion. PaDNet fuses the responses through $\widehat{\mathbf{y}} = \mathbf{A}\mathbf{u} + \mathbf{B}\mathbf{v} + \mathbf{b}$. Using the centered residuals defined above, let $\boldsymbol{\Sigma}_{\perp} = \mathbb{E}[\mathbf{v}_{\perp}\mathbf{v}_{\perp}^{\top}]$. Residualizing the joint least-squares objective gives

$$\mathcal{R}^{\text{lin}}_{\text{s}} - \mathcal{R}^{\text{lin}}_{\text{sd}} = \operatorname{tr}(\mathbf{C}\boldsymbol{\Sigma}_{\perp}^{\dagger}\mathbf{C}^{\top}) > 0, \tag{9}$$

where the risks are minimized over affine predictors using $\mathbf{u}$ and $(\mathbf{u}, \mathbf{v})$, respectively, and † denotes the Moore–Penrose pseudoinverse. Without residual predictive relevance, the difference remains nonnegative and equals zero when $\mathbf{C} = \mathbf{0}$.

Thus, explicit dynamics conditioning strictly reduces population risk when it exposes residual predictive structure accessible to the fusion map. This result concerns fixed branch responses and optimal affine fusion; experiments assess its realization through learned representations.

4 Methodology

4.1 PaDNet Overview

We propose the Patch-State and Delta-Dynamics Network (PaDNet), which jointly models local states and transition dynamics. The Patch-State stream regularizes trajectory encoding to capture overall local tendencies and temporal structures, while the Delta-Dynamics stream captures successive increments and their cross-step evolution. The two streams produce horizon-wise outputs that are fused into the final forecast.

Following the channel-independent representation defined in Eq. 2, PaDNet operates on $\mathbf{X}^{\text{CI}} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M]^{\top} \in \mathbb{R}^{M \times L}$, $M = BC$, where M is the number of channel-independent sequences and $\mathbf{x}_m \in \mathbb{R}^L$ denotes the m-th univariate historical sequence. All sequences are processed independently using shared model parameters.

As illustrated in Figure 2, the complete PaDNet mapping is formulated as

$$\widehat{\mathbf{Y}} = \text{CI}^{-1}\big(\mathcal{F}_{\text{sd}}\big[\mathcal{H}_{\text{s}}\big(\mathcal{E}_{\text{s}}\big(\mathcal{P}_{\text{LR}}(\mathbf{X}^{\text{CI}}; P, S, r)\big)\big) \,\|\, \mathcal{H}_{\text{d}}\big(\mathcal{E}_{\text{d}}\big(\mathcal{R}_k(\Delta\mathbf{X}^{\text{CI}}; \tau, \epsilon)\big)\big)\big]\big), \tag{10}$$

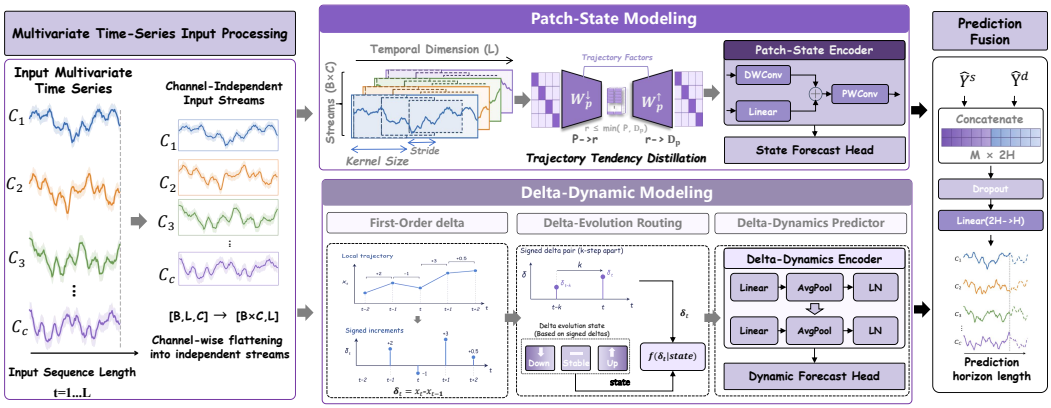


Figure 2: Overview of PaDNet. PaDNet models local trajectory states through shared patch projection and stepwise transition dynamics through state-conditioned delta routing. The two branch predictions are then linearly fused.

where $\mathcal{P}_{\mathrm{LR}}(\cdot)$ denotes the Patch-State construction module, implemented with patch length P, stride S, and rank r. The matrix $\Delta\mathbf{X}^{\mathrm{CI}}$ contains the first-order increments, while $\mathcal{R}_k(\cdot)$ performs delta-evolution routing using interval k, threshold τ, and stability constant ϵ.

Moreover, $\mathcal{E}_{\mathrm{s}}$ and $\mathcal{E}_{\mathrm{d}}$ denote the patch-state and delta-dynamics encoders, respectively; $\mathcal{H}_{\mathrm{s}}$ and $\mathcal{H}_{\mathrm{d}}$ project their representations into the forecasting horizon; $\|$ denotes feature concatenation; and $\mathcal{F}_{\mathrm{sd}}$ integrates the two responses. Finally, $\mathrm{CI}^{-1}(\cdot)$ restores the variable dimension and produces $\widehat{\mathbf{Y}} \in \mathbb{R}^{B\times H\times C}$.

The two streams serve complementary roles: Patch-State captures what local temporal structures are present, while Delta-Dynamics characterizes how they evolve. PaDNet thereby shifts from predefined component separation to joint state–dynamics modeling.

4.2 Patch-State Modeling

Following Eq. 3, we organize the input into the patch tensor $\mathbf{X}^{\mathrm{patch}} = \mathrm{Patch}_{P,S}\big(\mathbf{X}^{\mathrm{CI}}\big) \in \mathbb{R}^{M\times N\times P}$, where each $\mathbf{p}_{m,n} \in \mathbb{R}^P$ contains the observations of a local trajectory. Replicated end padding is applied when required. These patches serve as inputs to the shared trajectory-to-state mapping.

The shared embedding map $\mathbf{W}_p \in \mathbb{R}^{D_p\times P}$ is parameterized as

$$\mathbf{W}_p = \mathbf{W}_p^{\uparrow}\mathbf{W}_p^{\downarrow}, \qquad \mathbf{W}_p^{\downarrow} \in \mathbb{R}^{r\times P}, \qquad \mathbf{W}_p^{\uparrow} \in \mathbb{R}^{D_p\times r}, \tag{11}$$

where D_p is the patch-state embedding dimension and r controls the rank of the shared embedding map:

$$1 \le r < \min(P, D_p), \qquad \mathrm{rank}(\mathbf{W}_p) = \mathrm{rank}(\mathbf{W}_p^{\uparrow}\mathbf{W}_p^{\downarrow}) \le r. \tag{12}$$

The resulting Patch-State construction is expressed by

$$\mathbf{E}^{\mathrm{s}} = \mathcal{P}_{\mathrm{LR}}\big(\mathbf{X}^{\mathrm{CI}}; P, S, r\big) = \phi_{\mathrm{s}}\Big(\mathbf{X}^{\mathrm{patch}}\,\big(\mathbf{W}_p^{\downarrow}\big)^{\top}\,\big(\mathbf{W}_p^{\uparrow}\big)^{\top} + \mathbf{b}_p\Big) \in \mathbb{R}^{M\times N\times D_p}, \tag{13}$$

where $\mathbf{b}_p \in \mathbb{R}^{D_p}$ is a learnable bias broadcast across sequences and patches, and ϕ_{s} denotes the nonlinear transformation used by the patch-state stream. For each patch, $\mathbf{z}^{\mathrm{s}}_{m,n} = \mathbf{W}_p^{\downarrow}\mathbf{p}_{m,n} \in \mathbb{R}^r$ denotes its latent coordinates.

The rank constraint encourages recurrent local structures to share latent factors while preserving patch-specific coordinates. The resulting embeddings are aggregated as

$$\mathbf{F}^{\mathrm{s}} = \mathcal{E}_{\mathrm{s}}(\mathbf{E}^{\mathrm{s}}) \in \mathbb{R}^{M\times D_{\mathrm{s}}}, \tag{14}$$

where $\mathcal{E}_{\mathrm{s}}$ is a lightweight encoder and $\mathbf{F}^{\mathrm{s}}$ denotes the final Patch-State representation.

4.3 Delta-Dynamics Modeling

Although Patch-State captures the overall structure of local trajectories, it leaves stepwise transitions implicitly encoded, such that similar states may correspond to distinct evolution patterns. To resolve this ambiguity, PaDNet introduces Delta-Dynamics to explicitly model successive changes and their cross-step evolution, as illustrated in Figure 4.

4.3.1 First-order delta construction.

For each channel-independent sequence $\mathbf{x}_m$, we construct an aligned delta sequence as

$$\bar{\delta}_{m,t} = \begin{cases} 0, & t = 1, \\ x_{m,t} - x_{m,t-1}, & t = 2, \dots, L, \end{cases} \tag{15}$$

where each noninitial delta retains the direction and magnitude of the corresponding transition. In implementation, the operation is performed using the first-order difference matrix

$$\mathbf{G}_1 = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ -1 & 1 & 0 & \cdots & 0 \\ 0 & -1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 & 1 \end{bmatrix} \in \mathbb{R}^{L \times L}. \tag{16}$$

Accordingly, the delta sequences of all channels are computed as

$$\mathbf{D} = \Delta \mathbf{X}^{\mathrm{CI}} = \mathbf{X}^{\mathrm{CI}} \mathbf{G}_1^{\top} = \big[\bar{\delta}_{m,t}\big]_{\substack{m=1,\dots,M \\ t=1,\dots,L}} \in \mathbb{R}^{M \times L}. \tag{17}$$

4.3.2 Delta-Evolution Routing

A first-order delta characterizes an individual transition but does not capture its evolution across successive steps. To characterize this evolution and condition the delta encoding accordingly, PaDNet compares deltas separated by an interval k. For $t > k$, the normalized cross-step evolution is defined as

$$\rho_{m,t}^{(k)} = \frac{\bar{\delta}_{m,t} - \bar{\delta}_{m,t-k}}{|\bar{\delta}_{m,t}| + |\bar{\delta}_{m,t-k}| + \epsilon}, \tag{18}$$

where $k \in \{1, \dots, L-1\}$ is the comparison interval and $\epsilon > 0$ is a numerical stability constant.

The numerator measures the change between the current and lagged deltas, while the denominator normalizes this change by their combined magnitudes. Since $|\bar{\delta}_{m,t} - \bar{\delta}_{m,t-k}| \le |\bar{\delta}_{m,t}| + |\bar{\delta}_{m,t-k}|$, the resulting measure satisfies $\rho_{m,t}^{(k)} \in (-1, 1)$ for $\epsilon > 0$, providing a bounded, scale-normalized characterization of cross-step delta evolution.

Using this measure, PaDNet assigns each valid position to one of three mutually exclusive evolution states under a threshold $\tau \ge 0$:

$$q_{m,t}^{(k)} = \begin{cases} \text{down}, & \rho_{m,t}^{(k)} < -\tau, \\ \text{stable}, & |\rho_{m,t}^{(k)}| \le \tau, \\ \text{up}, & \rho_{m,t}^{(k)} > \tau. \end{cases} \tag{19}$$

The states down, stable, and up indicate that the signed delta is decreasing, approximately unchanged, or increasing relative to its k-step lagged value, respectively. For $t \le k$, where a valid lagged delta is unavailable, we assign $q_{m,t}^{(k)} = \text{stable}$.

To enable state-specific modulation of the continuous delta, we encode the evolution state as a one-hot vector:

$$\mathbf{h}_{m,t}^{(k)} = \begin{bmatrix} \mathbb{I}\big(q_{m,t}^{(k)} = \text{down}\big) \\ \mathbb{I}\big(q_{m,t}^{(k)} = \text{stable}\big) \\ \mathbb{I}\big(q_{m,t}^{(k)} = \text{up}\big) \end{bmatrix} = \begin{bmatrix} h_{m,t}^{\text{down}} \\ h_{m,t}^{\text{stable}} \\ h_{m,t}^{\text{up}} \end{bmatrix} \in \{0,1\}^3, \tag{20}$$

where $\mathbb{I}(\cdot)$ denotes the indicator function, with dependence on k omitted from individual entries for brevity. Exactly one entry equals 1, so $h_{m,t}^{\text{down}} + h_{m,t}^{\text{stable}} + h_{m,t}^{\text{up}} = 1$.

To incorporate cross-step evolution into the transition encoding, we combine the continuous delta and its categorical state as

$$\mathbf{q}_{m,t} = \begin{bmatrix} \bar{\delta}_{m,t} \\ h_{m,t}^{\text{down}} \\ h_{m,t}^{\text{stable}} \\ h_{m,t}^{\text{up}} \end{bmatrix} \in \mathbb{R}^4. \tag{21}$$

A shared linear projection then produces the routed delta:

$$\widetilde{\delta}_{m,t} = [w_d \quad w_{\text{down}} \quad w_{\text{stable}} \quad w_{\text{up}}]\, \mathbf{q}_{m,t} + b_d, \tag{22}$$

where w_d controls the contribution of the continuous delta; w_{down}, w_{stable}, and w_{up} are learnable state-specific offsets; and b_d is a shared bias.

The routing mechanism becomes explicit in the following equivalent form for $t > k$:

$$\widetilde{\delta}_{m,t} = w_d \bar{\delta}_{m,t} + \begin{cases} w_{\text{down}} + b_d, & \rho_{m,t}^{(k)} < -\tau, \\ w_{\text{stable}} + b_d, & |\rho_{m,t}^{(k)}| \le \tau, \\ w_{\text{up}} + b_d, & \rho_{m,t}^{(k)} > \tau. \end{cases} \tag{23}$$

For the first k positions, the stable branch is selected. The continuous term encodes the signed delta, while the state-specific offset conditions this encoding on its cross-step evolution. Thus, routing incorporates both the individual transition and its relative change across steps.

The continuous delta and its evolution state are integrated before temporal encoding. Collecting the one-hot states as $\mathbf{H}^{(k)} = [\mathbf{h}_{m,t}^{(k)}] \in \mathbb{R}^{M \times L \times 3}$, the routed sequence is

$$\widetilde{\mathbf{D}} = w_d \mathbf{D} + \mathbf{H}^{(k)} \mathbf{w}_{\text{e}} + b_d \mathbf{1}_{M \times L} \in \mathbb{R}^{M \times L}, \tag{24}$$

where $\mathbf{w}_{\text{e}} = [w_{\text{down}}, w_{\text{stable}}, w_{\text{up}}]^\top \in \mathbb{R}^3$ contains the learnable evolution-state offsets, and the tensor–vector product contracts the last dimension of $\mathbf{H}^{(k)}$.

The integrated sequence is then encoded as

$$\mathbf{D}_{\text{e}} = \mathcal{E}_{\text{d}}\left(\widetilde{\mathbf{D}}\right) \in \mathbb{R}^{M \times D_{\text{d}}}, \tag{25}$$

where $\mathcal{E}_{\text{d}}$ encodes the continuous deltas and evolution-dependent offsets into the Delta-Dynamics representation $\mathbf{D}_{\text{e}}$.

4.4 State–Dynamics Response Integration

PaDNet maps the Patch-State and Delta-Dynamics representations into a common forecasting space, where their responses are integrated through a learnable fusion map:

$$\widehat{\mathbf{Y}}^{\text{s}} = \mathcal{H}_{\text{s}}(\mathbf{F}^{\text{s}}) \in \mathbb{R}^{M \times H}, \qquad \widehat{\mathbf{Y}}^{\text{d}} = \mathcal{H}_{\text{d}}(\mathbf{D}_{\text{e}}) \in \mathbb{R}^{M \times H}, \tag{26}$$

where $\mathcal{H}_{\text{s}}$ and $\mathcal{H}_{\text{d}}$ are the respective prediction heads. The two horizon-wise responses are concatenated and fused as

$$\widehat{\mathbf{Y}}^{\text{CI}} = \left[\widehat{\mathbf{Y}}^{\text{s}} \parallel \widehat{\mathbf{Y}}^{\text{d}}\right] \mathbf{W}_f^\top + \mathbf{1}\mathbf{b}_f^\top \in \mathbb{R}^{M \times H}, \tag{27}$$

where $\parallel$ denotes concatenation, $\mathbf{W}_f \in \mathbb{R}^{H \times 2H}$ and $\mathbf{b}_f \in \mathbb{R}^H$ are learnable parameters, and $\mathbf{1} \in \mathbb{R}^M$ broadcasts the bias across sequences.

The fusion map allows each future position to aggregate responses from all horizon positions of both streams, conditioning the forecast on local trajectory structure and transition dynamics. Finally, $\widehat{\mathbf{Y}}^{\text{CI}}$ is reshaped via CI^{-1} to obtain $\widehat{\mathbf{Y}} \in \mathbb{R}^{B \times H \times C}$.

Table 1: Multivariate forecasting results averaged across four prediction horizons: $\{12, 24, 48, 96\}$ for PEMS and $\{96, 192, 336, 720\}$ for the other datasets. Both MSE and MAE are reported as averages. The best results are in red and the second-best in blue.

Datasets	PaDNet		ParallelTime		XPatch		DeformableTST		SST		PatchMLP		RLinear		PatchTST		DLinear	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	0.4314	0.4318	0.4570	0.4492	0.4557	0.4463	0.4402	0.4347	0.4531	0.4435	0.4675	0.4531	0.4429	0.4311	0.4428	0.4361	0.4557	0.4504
ETTm1	0.3753	0.3937	0.3900	0.4038	0.3875	0.4031	0.3747	0.3945	0.3914	0.4012	0.3978	0.4041	0.4102	0.4010	0.3849	0.3975	0.4028	0.4073
ETTm2	0.2768	0.3229	0.2859	0.3327	0.2791	0.3259	0.2834	0.3288	0.2795	0.3255	0.2823	0.3277	0.2862	0.3283	0.2868	0.3297	0.3325	0.3846
Solar	0.1945	0.2502	0.2142	0.2659	0.1982	0.2539	0.2017	0.2634	0.2221	0.2673	0.2147	0.2634	0.2641	0.2814	0.2692	0.3211	0.2532	0.3143
Electricity	0.1611	0.2555	0.1647	0.2588	0.1617	0.2559	0.1686	0.2667	0.1731	0.2671	0.1753	0.2734	0.1691	0.2615	0.1635	0.2568	0.1668	0.2646
Exchange	0.3525	0.3964	0.3660	0.4068	0.4034	0.4194	0.3663	0.4079	0.3844	0.4104	0.3718	0.4110	0.3776	0.4085	0.3552	0.3981	0.3446	0.4023
Pems03	0.0866	0.1876	0.1045	0.2095	0.0909	0.1932	0.0939	0.1986	0.1194	0.2241	0.0948	0.2003	0.1342	0.2286	0.1282	0.2331	0.1303	0.2333
Pems04	0.0939	0.1957	0.1204	0.2302	0.1017	0.2054	0.1131	0.2219	0.1229	0.2338	0.0937	0.2025	0.1482	0.2491	0.1445	0.2522	0.1432	0.2489
Pems07	0.0709	0.1690	0.0892	0.1971	0.0765	0.1781	0.0836	0.1898	0.0908	0.2001	0.0814	0.1882	0.1281	0.2314	0.1216	0.2338	0.1288	0.2337
Pems08	0.1103	0.1935	0.1410	0.2208	0.1136	0.1984	0.1300	0.2157	0.1595	0.2364	0.1516	0.2184	0.2255	0.2680	0.2077	0.2649	0.1993	0.2630
Count 1st	7	9	0	0	0	0	1	0	0	0	1	0	0	1	0	0	1	0

5 Experiments

5.1 Experimental Settings

5.1.1 Training Details

All models are trained with the MSE objective under a unified experimental protocol and evaluated using MSE, MAE, and RMSE. Model-specific configurations follow their official implementations, with complete training details provided in Appendix.

5.1.2 Datasets

We evaluate PaDNet on multiple real-world multivariate forecasting benchmarks: ETTh1, ETTm1, and ETTm2 (Zhou et al., 2021); Electricity, Solar, and Exchange (Wu et al., 2021); and PEMS03, PEMS04, PEMS07, and PEMS08 (Chen et al., 2001; Liu et al., 2022a). On the PEMS datasets, we use prediction horizons of $\{12, 24, 48, 96\}$; for the other datasets, the prediction horizons are set to $\{96, 192, 336, 720\}$. Detailed dataset statistics are provided in Appendix.

5.1.3 Baselines

We compare PaDNet with representative recent long-term forecasting methods, including XPatch (Stitsyuk & Choi, 2025), SST (Xu et al., 2025), DLinear (Zeng et al., 2023), PatchTST (Nie et al., 2023), RLinear (Li et al., 2023), PatchMLP (Tang & Zhang, 2025), DeformableTST (DeformTST) (Luo & Wang, 2024a), and ParallelTime (Katav & Kontorovich, 2025). These baselines cover linear forecasters, patch-based architectures, and recent temporal representation learning approaches.

5.2 Main Results

Table 1 reports forecasting results averaged across prediction horizons. PaDNet achieves strong performance in both MSE and MAE on most datasets, demonstrating its effective-

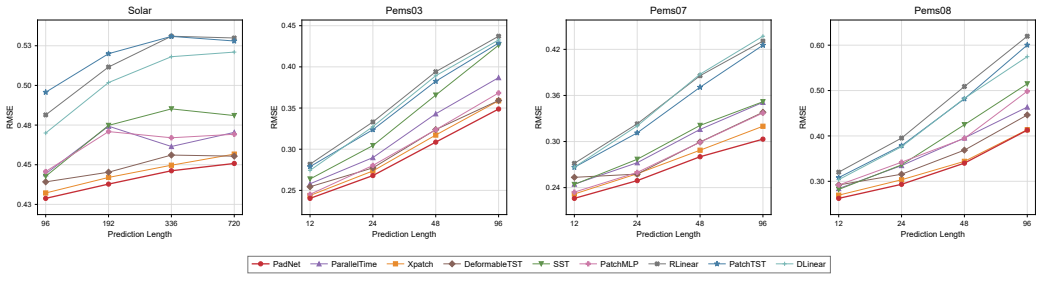


Figure 3: RMSE comparison of PaDNet and representative baselines across forecasting horizons on Solar, PEMS03, PEMS07, and PEMS08.

ness across heterogeneous temporal characteristics. Figure 3 further visualizes RMSE across prediction horizons, complementing the averaged results with comparisons at individual forecasting lengths. The gains are particularly evident on PEMS07, where PaDNet reduces MSE and MAE by an average of 7.3% and 5.1%, respectively, across prediction horizons. On strongly periodic or high-dimensional datasets, PaDNet remains competitive, although improvements are smaller in some settings. These results suggest that explicit delta representations complement patch-based trajectory modeling, with their benefits varying across datasets. Complete horizon-wise results are reported in the appendix.

5.3 Analysis of Temporal Representations

5.3.1 Component ablation.

We examine the contributions of delta dynamics and the patch-state constraint. w/o Delta-Dynamics (w/o DD) removes the delta-dynamics branch, while w/o Patch-State Constraint (w/o PSC) replaces the constrained patch-state projection with a standard linear projection. The results in Table 2 show that the complete PaDNet generally achieves better forecasting performance, supporting the contributions of both components.

Table 2: Component ablation of PaDNet, evaluating the contributions of Delta-Dynamics and the Patch-State encoding constraint. Results are averaged over four prediction horizons.

	ETTh1			ETTm1			Exchange			Pems04		
	MSE	MAE	RMSE	MSE	MAE	RMSE	MSE	MAE	RMSE	MSE	MAE	RMSE
w/o DD	0.4394	0.4376	0.6621	0.3802	0.3969	0.6154	0.3687	0.4050	0.5567	0.0954	0.1981	0.3073
w/o PSC	0.4362	0.4346	0.6596	0.3782	0.3970	0.6138	0.3679	0.4056	0.5562	0.0944	0.1968	0.3057
PaDNet	0.4314	0.4318	0.6561	0.3753	0.3937	0.6114	0.3525	0.3964	0.5444	0.0939	0.1957	0.3047

5.3.2 Comparison with explicit decomposition

We further compare the proposed paradigm with alternatives that use raw inputs or explicit decomposition. Reg feeds the same raw sequence into the patch-based and full-sequence branches without decomposition. MA adopts the moving-average decomposition used in Autoformer (Wu et al., 2021), while EMA and DEMA employ the exponential smoothing schemes considered in XPatch (Stitsyuk & Choi, 2025). These decomposition variants feed the residual and smoothed component into the patch-based and full-sequence branches, respectively. PaDNet instead combines patch states with delta dynamics. Table 3 summarizes the results, showing favorable overall forecasting performance for the proposed paradigm.

Table 3: Forecasting performance of PaDNet against a baseline without decomposition and explicit trend decomposition variants. Results are averaged over four prediction horizons.

Datasets	Reg			MA			EMA			DEMA			Delta		
	MSE	MAE	RMSE	MSE	MAE	RMSE	MSE	MAE	RMSE	MSE	MAE	RMSE	MSE	MAE	RMSE
ETTh1	0.4344	0.4353	0.6585	0.4429	0.4387	0.6647	0.4424	0.4382	0.6643	0.4453	0.4386	0.6665	0.4314	0.4318	0.6561
ETTm1	0.3800	0.3967	0.6152	0.3847	0.4002	0.6191	0.3810	0.3969	0.6160	0.3865	0.3992	0.6205	0.3753	0.3937	0.6114
Electricity	0.1617	0.2553	0.4008	0.1650	0.2596	0.4050	0.1628	0.2569	0.4022	0.1651	0.2596	0.4052	0.1611	0.2555	0.4002
Pems03	0.0869	0.1876	0.2918	0.0960	0.1981	0.3062	0.0908	0.1928	0.2980	0.0982	0.2015	0.3095	0.0866	0.1876	0.2914
Pems04	0.0952	0.1976	0.3068	0.1060	0.2105	0.3232	0.1025	0.2066	0.3181	0.1126	0.2180	0.3325	0.0939	0.1957	0.3047
Pems08	0.1162	0.1965	0.3344	0.1214	0.2039	0.3443	0.1155	0.1986	0.3347	0.1322	0.2082	0.3577	0.1103	0.1935	0.3272

6 Conclusion

We revisit explicit trend decomposition in patch-based forecasting and identify substantial overlap between patch representations and trend information. Motivated by this observation, we propose a Patch-State and Delta-Dynamics paradigm and instantiate it as PaDNet, combining local trajectory states with explicit stepwise dynamics. Experiments demonstrate favorable forecasting performance, while ablation studies support the contributions of its components. These findings highlight the value of jointly modeling temporal states and their evolution for time series forecasting.

Reproducibility Statement

To support reproducibility, we provide the complete model formulation, training objective, detailed architectural descriptions, and theoretical proofs in the appendix. The experimental setup specifies the datasets, data splits, preprocessing procedures, evaluation metrics, and training protocol. Complete hyperparameter settings, implementation details, horizon-wise results, and ablation studies are also provided in the appendix. Source code and experiment configurations will be included in the supplementary material. Anonymized source code and experiment configurations are available at https://anonymous.4open.science/r/PaDnet-0D8E/.

AI Assistance

Generative AI tools were used to assist with literature retrieval and discovery, drafting portions of the manuscript, and polishing the writing. The authors reviewed and revised all AI-assisted text and independently verified the relevant literature.

References

Md Atik Ahamed and Qiang Cheng. Timemachine: A time series is worth 4 mambas for long-term forecasting. In ECAI 2024: 27th European Conference on Artificial Intelligence, 19–24 October 2024, Santiago de Compostela, Spain–Including 13th Conference on Prestigious Applications of Intelligent Systems (PAIS 2024), pp. 1688–1695. SAGE Publications Pvt. Ltd 1 Oliver's Yard, 55 City Road, London, EC1Y 1SP, 2024.

Nesreen K. Ahmed, Amir F. Atiya, Neamat El Gayar, and Hisham El-Shishiny. An empirical comparison of machine learning models for time series forecasting. Econometric Reviews, 29(5-6):594–621, 2010. doi: 10.1080/07474938.2010.481556.

George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. Time series analysis: forecasting and control. John Wiley & Sons, 2015.

Lingxiao Cao, Bin Wang, Guiyuan Jiang, Yanwei Yu, and Junyu Dong. Spatiotemporal-aware trend-seasonality decomposition network for traffic flow forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 39, pp. 11463–11471, 2025a.

Yizhi Cao, Zijian Tian, Wenjie Guo, and Xinggao Liu. Mspatch: A multi-scale patch mixing framework for multivariate time series forecasting. Expert Systems with Applications, 273: 126849, 2025b.

Chao Chen, Karl Petty, Alexander Skabardonis, Pravin Varaiya, and Zhanfeng Jia. Freeway performance measurement system: mining loop detector data. Transportation research record, 1748(1):96–102, 2001.

Alexandre Drouin, Étienne Marcotte, and Nicolas Chapados. Tactis: Transformer-attentional copulas for time series. In International Conference on Machine Learning, pp. 5447–5493. PMLR, 2022.

Carson Eisenach, Yagna Patel, and Dhruv Madeka. Mqtransformer: Multi-horizon forecasts with context dependent and feedback-aware attention. arXiv preprint arXiv:2009.14799, 2020.

Jake Grigsby, Zhe Wang, Nam Nguyen, and Yanjun Qi. Long-range transformers for dynamic spatiotemporal forecasting. arXiv preprint arXiv:2109.12218, 2021.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In First conference on language modeling, 2024.

Timothy Hall and Khaled Rasheed. A survey of machine learning methods for time series prediction. Applied Sciences, 15(11), 2025. ISSN 2076-3417. doi: 10.3390/app15115957. URL https://www.mdpi.com/2076-3417/15/11/5957.

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

Ao Hu, Liangjian Wen, Jiang Duan, Yong Dai, He Yan, Dongkai Wang, Jun Wang, Yukun Zhang, Ruoxi Jiang, and Zenglin Xu. Pmdformer: Patch-mean decoupling information transformer for long-term forecasting, 2026. URL https://arxiv.org/abs/2606.26549.

Jiaxi Hu, Disen Lan, Ziyu Zhou, Qingsong Wen, and Yuxuan Liang. Time-ssm: Simplifying and unifying state space models for time series forecasting. arXiv preprint arXiv:2405.16312, 2024.

Qihe Huang, Lei Shen, Ruixin Zhang, Jiahuan Cheng, Shouhong Ding, Zhengyang Zhou, and Yang Wang. Hdmixer: Hierarchical dependency with extendable patch for multivariate time series forecasting. In Proceedings of the AAAI conference on artificial intelligence, volume 38, pp. 12608–12616, 2024.

Itay Katav and Aryeh Kontorovich. Paralleltime: Dynamically weighting the balance of short-and long-term temporal dependencies. arXiv preprint arXiv:2507.13998, 2025.

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

Bernd Klockl. Multivariate time series models applied to the assessment of energy storage in power systems. In Proceedings of the 10th International Conference on Probablistic Methods Applied to Power Systems, pp. 1–8. IEEE, 2008.

Seunghan Lee, Taeyoung Park, and Kibok Lee. Learning to embed time series patches independently, 2024. URL https://arxiv.org/abs/2312.16427.

Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. Advances in neural information processing systems, 32, 2019.

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

Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. International journal of forecasting, 37(4):1748–1764, 2021.

Yang Lin, Irena Koprinska, and Mashud Rana. Ssdnet: State space decomposition neural network for time series forecasting. In 2021 IEEE International conference on data mining (ICDM), pp. 370–378. IEEE, 2021.

Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. Scinet: Time series modeling and forecasting with sample convolution and interaction. Advances in neural information processing systems, 35:5816–5828, 2022a.

Shenglin Liu. Decompose-disentangle-decode time series forecasting (d3tsf): dual-scale time series forecasting via seasonality of patch-wise state space and sparse trend learning. Journal of King Saud University Computer and Information Sciences, 2026.

Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In International conference on learning representations, 2021a.

Yong Liu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Non-stationary transformers: Exploring the stationarity in time series forecasting. Advances in neural information processing systems, 35:9881–9893, 2022b.

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, volume 2024, pp. 11116–11140, 2024.

Zhenyu Liu, Zhengtong Zhu, Jing Gao, and Cheng Xu. Forecast methods for time series data: A survey. Ieee Access, 9:91896–91912, 2021b.

Donghao Luo and Xue Wang. Deformabletst: Transformer for time series forecasting without over-reliance on patching. Advances in Neural Information Processing Systems, 37:88003–88044, 2024a.

Donghao Luo and Xue Wang. Moderntcn: A modern pure convolution structure for general time series analysis. In ICLR, 2024b.

Kasun Mendis, Manjusri Wickramasinghe, and Pasindu Marasinghe. Multivariate time series forecasting: A review. In Proceedings of the 2024 2nd Asia Conference on Computer Vision, Image Processing and Pattern Recognition, pp. 1–9, 2024.

Douglas C Montgomery, Cheryl L Jennings, and Murat Kulahci. Introduction to time series analysis and forecasting. John Wiley & Sons, 2015.

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, volume 39, pp. 19581–19588, 2025.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In The Eleventh International Conference on Learning Representations, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Xiangfei Qiu, Hanyin Cheng, Xingjian Wu, Junkai Lu, Jilin Hu, Chenjuan Guo, Christian S Jensen, and Bin Yang. A comprehensive survey of deep learning for multivariate time series forecasting: A channel strategy perspective. arXiv preprint arXiv:2502.10721, 2025.

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

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

José F Torres, Dalil Hadjout, Abderrazak Sebaa, Francisco Martínez-Álvarez, and Alicia Troncoso. Deep learning for time series forecasting: a survey. Big data, 9(1):3–21, 2021.

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

Meng Wan, Qi Su, Huan Hao, Jue Wang, Yuexiu Cui, Yuxuan Bi, Rongqiang Cao, Peng Shi, Yangang Wang, Zonghua Qiu, et al. Ppdformer: Channel-specific periodic patch division for time series forecasting. In ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5. IEEE, 2025.

Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. Micn: Multi-scale local and global context modeling for long-term series forecasting. In The eleventh international conference on learning representations, 2023.

Zixuan Weng, Jindong Han, Wenzhao Jiang, and Hao Liu. Sde: A simplified and disentangled dependency encoding framework for state space models in time series forecasting. In Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2, KDD '25, pp. 3168–3179, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400714542. doi: 10.1145/3711896.3737119.

Harya Widiputra, Adele Mailangkay, and Elliana Gautama. Multivariate cnn-lstm model for multiple parallel financial time-series prediction. Complexity, 2021(1):9903518, 2021.

Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Cost: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. arXiv preprint arXiv:2202.01575, 2022a.

Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Etsformer: Exponential smoothing transformers for time-series forecasting. arXiv preprint arXiv:2202.01381, 2022b.

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

Qiang Wu, Gechang Yao, Zhixi Feng, and Shuyuan Yang. Peri-midformer: Periodic pyramid transformer for time series analysis. In Advances in Neural Information Processing Systems, volume 37, pp. 13035–13073. Curran Associates, Inc., 2024. doi: 10.52202/079017-0415.

Xingjian Wu, Xiangfei Qiu, Hanyin Cheng, Zhengyu Li, Jilin Hu, Chenjuan Guo, and Bin Yang. Enhancing time series forecasting through selective representation spaces: A patch perspective. Advances in Neural Information Processing Systems, 38:23328–23354, 2026.

Xiongxiao Xu, Canyu Chen, Yueqing Liang, Baixiang Huang, Guangji Bai, Liang Zhao, and Kai Shu. Sst: Multi-scale hybrid mamba-transformer experts for time series forecasting. In Proceedings of the 34th ACM International Conference on Information and Knowledge Management, CIKM '25, pp. 3655–3665, New York, NY, USA, 2025. Association for Computing Machinery.

Zhonghua Yang, Zhaohui Zhang, Wen Liu, Cun Du, Xifan Gao, and Shihao Zhou. Dpatchformer: A deformable patch transformer with intra-patch and inter-patch interactions for multivariate time series forecasting. Signal, Image and Video Processing, 20(6):312, 2026.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In Proceedings of the AAAI conference on artificial intelligence, volume 37, pp. 11121–11128, 2023.

Yitian Zhang, Liheng Ma, Soumyasundar Pal, Yingxue Zhang, and Mark Coates. Multi-resolution time-series transformer for long-term forecasting. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li (eds.), Proceedings of The 27th International Conference on Artificial Intelligence and Statistics, volume 238 of Proceedings of Machine Learning Research, pp. 4222–4230. PMLR, 02–04 May 2024.

Xiaoting Zhong, Yunsen Zhou, Yinxin Bao, and Quan Shi. Decomppatchtst: A hybrid framework fusing learnable polynomials and patch transformers for time-series forecasting. Symmetry, 18(3), 2026. ISSN 2073-8994. doi: 10.3390/sym18030516. URL https://www.mdpi.com/2073-8994/18/3/516.

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, volume 35, pp. 11106–11115, 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 International conference on machine learning, pp. 27268–27286. PMLR, 2022.

A Related Work

A.1 Evolution of Forecasting Architectures

Time-series forecasting (Mendis et al., 2024; Liu et al., 2021b) is a fundamental sequence modeling task with applications in energy (Klockl, 2008) and finance (Widiputra et al., 2021). Early forecasting studies drew on statistical methods, including ARIMA (Box et al., 2015), alongside conventional machine learning approaches (Ahmed et al., 2010; Hall & Rasheed, 2025). With the development of deep learning (Qiu et al., 2025; Torres et al., 2021), long short-term memory networks (LSTMs) (Hochreiter & Schmidhuber, 1997) became widely used to model nonlinear temporal dependencies. Transformer-based methods subsequently introduced attention mechanisms for modeling long-range dependencies (Vaswani et al., 2017). Later studies explored attention computation, temporal dependencies, and relationships among variables in time-series forecasting (Li et al., 2019; Eisenach et al., 2020; Lim et al., 2021; Grigsby et al., 2021; Liu et al., 2021a; 2022b; Drouin et al., 2022; Liu et al., 2024). More recently, state-space models (Gu & Dao, 2024) have attracted interest for efficient long-sequence modeling, inspiring further forecasting architectures based on state-space and Mamba-style designs (Xu et al., 2025; Weng et al., 2025; Ahamed & Cheng, 2024; Hu et al., 2024).

A.2 Decomposition-Based Forecasting

Long-term time series often contain intertwined temporal patterns that obscure their underlying dependencies and increase forecasting difficulty. Autoformer (Wu et al., 2021) addressed this challenge by incorporating progressive trend–seasonal decomposition into the forecasting architecture. Subsequent studies extended component-wise modeling through state-space formulations (Lin et al., 2021), frequency-domain modeling (Zhou et al., 2022), disentangled representation learning (Woo et al., 2022a), and exponential smoothing (Woo et al., 2022b). This paradigm has also been explored in linear predictors (Zeng et al., 2023), local–global convolutional architectures (Wang et al., 2023), and spatiotemporal forecasting (Cao et al., 2025a). More recently, decomposition has been combined with patch-based representations to separately model trend and seasonal or residual components (Stitsyuk & Choi, 2025; Hu et al., 2026). However, whether explicit decomposition remains complementary when local trajectories are already represented by patches has received limited examination.

A.3 Patch-Based Forecasting

PatchTST (Nie et al., 2023) established patch-based modeling as an effective approach to time-series forecasting by grouping consecutive observations into local tokens, preserving local temporal information while reducing the computational cost of modeling long histories. Subsequent studies (Huang et al., 2024; Cao et al., 2025b; Wan et al., 2025; Wu et al., 2026; Yang et al., 2026; Murad et al., 2025) further explored flexible construction of temporal tokens to better accommodate diverse temporal structures and sampling characteristics, extending the representational capacity of patch-based forecasting. Alongside these developments, recent research (Stitsyuk & Choi, 2025) has integrated patch-based modeling with temporal decomposition, combining local temporal representations with explicit modeling of distinct temporal components. These developments have broadened the use of patch-based representations for capturing temporal structure in forecasting.

B Architecture and Implementation Details

B.1 Delta-Dynamics Construction

Figure 4 illustrates the construction of the Delta-Dynamics input. First-order differences capture the magnitude and direction of successive changes. Delta-Evolution Routing compares deltas across a temporal lag and applies a threshold to identify upward, stable, and downward evolution states. The resulting states are encoded through a codebook and com-

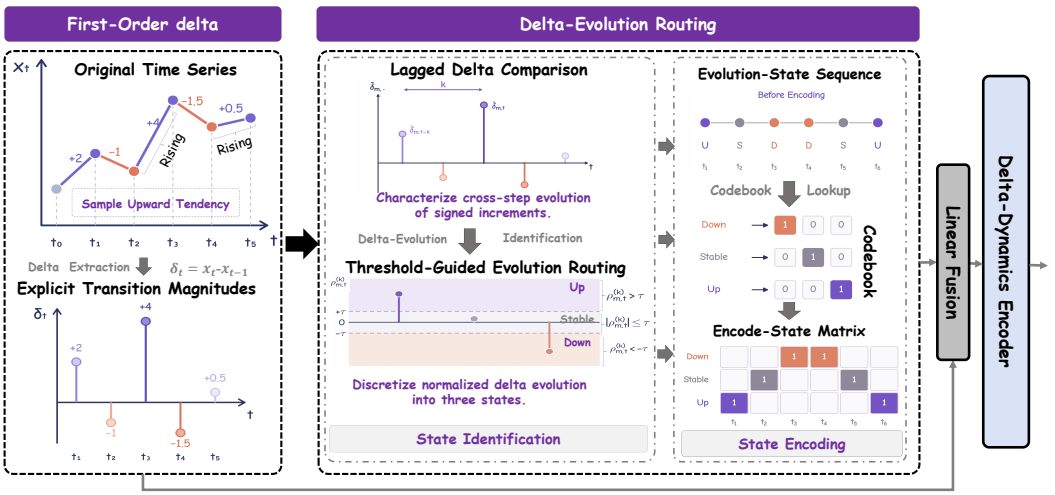
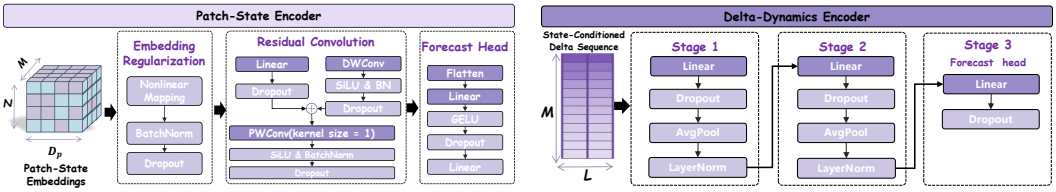


Figure 4: Overview of the Delta-Dynamics module.

bined with the first-order deltas to form the state-conditioned sequence $\widetilde{\mathbf{D}}$, which is then passed to the Delta-Dynamics encoder.

B.2 Detailed Branch Implementations

Figure 5 presents the detailed implementations of the two forecasting branches in PaDNet. The Patch-State branch takes the patch-state embeddings $\mathbf{E}^{\mathrm{s}} \in \mathbb{R}^{M\times N\times D_p}$ as input. Its encoder produces the Patch-State representation $\mathbf{F}^{\mathrm{s}} \in \mathbb{R}^{M\times D_{\mathrm{s}}}$, which is subsequently mapped by the state forecast head to $\widehat{\mathbf{Y}}^{\mathrm{s}} \in \mathbb{R}^{M\times H}$. The Delta-Dynamics branch operates on the state-conditioned delta sequence $\widetilde{\mathbf{D}} \in \mathbb{R}^{M\times L}$. Its encoder produces the Delta-Dynamics representation $\mathbf{D}_{\mathrm{e}} \in \mathbb{R}^{M\times D_{\mathrm{d}}}$, which is mapped by the dynamic forecast head to $\widehat{\mathbf{Y}}^{\mathrm{d}} \in \mathbb{R}^{M\times H}$. Here, $M = BC$ denotes the number of channel-independent sequences. The internal operations are shown directly in the corresponding subfigures. RevIN is applied to the input before both branches, and first-order differences are computed from the normalized input. After the two branch responses are fused, the resulting forecast is restored to the input scale through inverse RevIN.



(a) Patch-State branch. (b) Delta-Dynamics branch.

Figure 5: Detailed implementations of the two forecasting branches in PaDNet. Each branch consists of a representation encoder and its corresponding forecast head.

B.3 Hyperparameter Settings

Table 4 summarizes the default training and architectural hyperparameters of PaDNet. All experiments are conducted on a single NVIDIA RTX 4090 GPU.

Table 4: Default training and architectural configurations of PaDNet.

Training and Normalization		PaDNet Configuration	
Setting	Value	Setting	Value
Optimizer	Adam	Patch length P	16
Learning rate	1×10^{-4}	Patch stride S	8
Batch size	16	Patch padding	end
Training epochs	100	Patch-State rank r	6
Early-stopping patience	5	Patch head width	192
Objective	MSE	Delta hidden width	96
Random seed	42	Cross-step interval k	4
RevIN	1	Stability constant ϵ	1×10^{-6}
RevIN affine	1	State threshold τ	0.1
Subtract last	0	Patch / Delta dropout	0.1/0.1

C Experimental Settings

C.1 Dataset Details

We evaluate PaDNet on a wide range of real-world multivariate time series forecasting benchmarks. The ETT benchmark includes ETTh1, ETTm1, and ETTm2 (Zhou et al., 2021). We additionally evaluate on Electricity, Solar, and Exchange (Wu et al., 2021), together with four traffic flow datasets: PEMS03, PEMS04, PEMS07, and PEMS08 (Chen et al., 2001; Liu et al., 2022a). These datasets cover transformer temperature, electricity load, solar energy, traffic flow, and exchange rates, with heterogeneous sampling frequencies, dimensionalities, and temporal patterns. The dataset statistics are summarized in Table 5.

Table 5: Statistics of the multivariate time series datasets used in our experiments.

Dataset	ETTh1	ETTm1	ETTm2	Electricity	Solar	PEMS03	PEMS04	PEMS07	PEMS08	Exchange
Variables	7	7	7	321	137	358	307	883	170	8
Time steps	14400	57600	57600	26304	52560	26208	16992	28224	17856	7588
Frequency	1 hour	15 mins	15 mins	1 hour	10 mins	5 mins	5 mins	5 mins	5 mins	1 day
Domain	Electricity	Electricity	Electricity	Electricity	Energy	Transportation	Transportation	Transportation	Transportation	Finance

C.2 Baselines

We compare PaDNet with representative long-term forecasting methods from different modeling paradigms, including XPatch (Stitsyuk & Choi, 2025), SST (Xu et al., 2025), DLinear (Zeng et al., 2023), PatchTST (Nie et al., 2023), RLinear (Li et al., 2023), PatchMLP (Tang & Zhang, 2025), DeformableTST (Luo & Wang, 2024a), and ParallelTime (Katav & Kontorovich, 2025).

- XPatch (Stitsyuk & Choi, 2025) is a dual-stream forecasting framework that decomposes the input into seasonal and trend components for separate temporal modeling.
- ParallelTime (Katav & Kontorovich, 2025) models short-range patterns and long-range dependencies through parallel pathways with adaptive fusion.
- SST (Xu et al., 2025) combines Mamba and Transformer experts at multiple temporal scales to capture diverse sequence dependencies.
- PatchMLP (Tang & Zhang, 2025) is an MLP-based forecasting model that uses multi-scale patch representations for efficient long-sequence modeling.
- DeformableTST (DeformTST) (Luo & Wang, 2024a) introduces adaptive temporal modeling to capture flexible dependencies in time series.
- RLinear (Li et al., 2023) is a lightweight linear forecasting model equipped with reversible normalization for robust prediction.

- DLinear (Zeng et al., 2023) decomposes time series into trend and seasonal components and forecasts each component with an independent linear layer.
- PatchTST (Nie et al., 2023) applies channel-independent patch tokenization and Transformer encoding for long-term forecasting.

C.3 Training Protocol

We follow the standard chronological data splits for long-term forecasting. The ETT and PEMS datasets are partitioned into training, validation, and test sets with a ratio of $6:2:2$, while the remaining datasets use a ratio of $7:1:2$. For each dataset, normalization statistics are computed only from the training split and then applied to the validation and test splits.

The input sequence length is set to 96 for ETTh1, ETTm1, ETTm2, and Exchange, and 336 for PEMS03, PEMS04, PEMS07, PEMS08, Electricity, and Solar.

To ensure a fair comparison, all models are optimized using Adam (Kingma & Ba, 2014) with the MSE objective. Shared training configurations, including the initial learning rate, maximum number of training epochs, learning-rate schedule, early-stopping patience, and other optimization settings, are kept consistent across all methods. Early stopping and model selection are both based on the validation MSE, and the checkpoint achieving the lowest validation MSE is used for final evaluation. Model-specific architectural hyperparameters are configured following their original implementations.

C.4 Optimization and Performance Assessment

C.4.1 Training Objective

All models are optimized using Mean Squared Error (MSE). Let $\widehat{\mathbf{Y}}, \mathbf{Y} \in \mathbb{R}^{B\times H\times C}$ denote the predicted and ground-truth future sequences, where B is the batch size, H is the prediction length, and C denotes the number of variables. The training objective is defined as

$$\mathcal{L}_{\text{train}} = \frac{1}{BHC}\sum_{b=1}^{B}\sum_{h=1}^{H}\sum_{c=1}^{C}\left(\widehat{Y}_{b,h,c} - Y_{b,h,c}\right)^2. \tag{28}$$

C.4.2 Evaluation Metrics

We report MSE, Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE):

$$\text{MSE} = \frac{1}{BHC}\sum_{b=1}^{B}\sum_{h=1}^{H}\sum_{c=1}^{C}\left(\widehat{Y}_{b,h,c} - Y_{b,h,c}\right)^2, \tag{29}$$

$$\text{MAE} = \frac{1}{BHC}\sum_{b=1}^{B}\sum_{h=1}^{H}\sum_{c=1}^{C}\left|\widehat{Y}_{b,h,c} - Y_{b,h,c}\right|, \tag{30}$$

and

$$\text{RMSE} = \sqrt{\frac{1}{BHC}\sum_{b=1}^{B}\sum_{h=1}^{H}\sum_{c=1}^{C}\left(\widehat{Y}_{b,h,c} - Y_{b,h,c}\right)^2}. \tag{31}$$

Lower values indicate better forecasting performance. MSE is used as the primary metric, while MAE and RMSE provide complementary error assessments.

D Theoretical Analysis

We investigate two questions underlying PaDNet: whether explicit trend–seasonal decomposition is necessary for predictive expressiveness, and when incorporating Delta-Dynamics improves forecasting over Patch-State alone. We first analyze the information retained

by raw patches and establish an equivalence between affine predictors on patches and decomposed components. We then derive the reduction in conditional forecasting risk and characterize the improvement attainable by PaDNet's response-level linear fusion.

Let $\mathbf{x} \in \mathbb{R}^L$ denote a historical sequence and $\mathbf{y} \in \mathbb{R}^H$ its forecasting target. The channel index is omitted for clarity. Expectations are taken over the population distribution, with $\mathbb{E}\|\mathbf{x}\|_2^2 < \infty$ and $\mathbb{E}\|\mathbf{y}\|_2^2 < \infty$.

For a representation $\mathbf{z}$, define its optimal squared-error risk as

$$\mathcal{R}_{\mathbf{z}}^* = \inf_f \mathbb{E}\|\mathbf{y} - f(\mathbf{z})\|_2^2, \tag{32}$$

where the infimum is over measurable predictors with finite second moments. Writing $\boldsymbol{\mu}_{\mathbf{z}} = \mathbb{E}[\mathbf{y} \mid \mathbf{z}]$, conditional expectation gives

$$\mathbb{E}\big[(\mathbf{y} - \boldsymbol{\mu}_{\mathbf{z}})^\top (\boldsymbol{\mu}_{\mathbf{z}} - f(\mathbf{z})) \mid \mathbf{z}\big] = 0. \tag{33}$$

Expanding the squared error and taking expectations therefore yields

$$\mathbb{E}\|\mathbf{y} - f(\mathbf{z})\|_2^2 = \mathbb{E}\|\mathbf{y} - \boldsymbol{\mu}_{\mathbf{z}}\|_2^2 + \mathbb{E}\|\boldsymbol{\mu}_{\mathbf{z}} - f(\mathbf{z})\|_2^2. \tag{34}$$

Thus, the conditional mean attains the optimal risk.

D.1 Raw Patches and Explicit Decomposition

Let $\mathbf{p} \in \mathbb{R}^{NP}$ stack the N raw patches in a fixed order. Patch extraction can be expressed as

$$\mathbf{p} = \mathbf{Q}\mathbf{x}, \qquad \mathbf{Q} \in \mathbb{R}^{NP \times L}, \tag{35}$$

where $\mathbf{Q}$ also accounts for replicated end padding when applicable.

Assumption 1 (Complete patch coverage). Every historical observation appears in at least one patch. A reconstruction matrix $\mathbf{J} \in \mathbb{R}^{L \times NP}$ can therefore select one occurrence of each observation, satisfying

$$\mathbf{J}\mathbf{Q} = \mathbf{I}_L, \qquad \mathbf{x} = \mathbf{J}\mathbf{p}. \tag{36}$$

Consider a deterministic measurable decomposition $\mathbf{x}^{\mathrm{tr}} = \mathcal{M}_\kappa(\mathbf{x})$ and $\mathbf{x}^{\mathrm{se}} = \mathbf{x} - \mathbf{x}^{\mathrm{tr}}$. Its components reconstruct $\mathbf{x}$ exactly.

Proposition D.1 (Information equivalence). Under Assumption 1, the raw patches and the pair of decomposed components generate the same σ-algebra:

$$\sigma(\mathbf{p}) = \sigma(\mathbf{x}^{\mathrm{tr}}, \mathbf{x}^{\mathrm{se}}). \tag{37}$$

Consequently, their conditional target distributions agree almost surely, and

$$\mathcal{R}_{\mathbf{p}}^* = \mathcal{R}_{\mathrm{tr,se}}^*. \tag{38}$$

Proof. Under Assumption 1, $\mathbf{x} = \mathbf{J}\mathbf{p}$. Hence, the decomposed components can be recovered from the raw patches as $\mathbf{x}^{\mathrm{tr}} = \mathcal{M}_\kappa(\mathbf{J}\mathbf{p})$ and $\mathbf{x}^{\mathrm{se}} = \mathbf{J}\mathbf{p} - \mathcal{M}_\kappa(\mathbf{J}\mathbf{p})$. Both are measurable functions of $\mathbf{p}$, so $\sigma(\mathbf{x}^{\mathrm{tr}}, \mathbf{x}^{\mathrm{se}}) \subseteq \sigma(\mathbf{p})$.

Conversely, since $\mathbf{x} = \mathbf{x}^{\mathrm{tr}} + \mathbf{x}^{\mathrm{se}}$, the raw patches satisfy $\mathbf{p} = \mathbf{Q}(\mathbf{x}^{\mathrm{tr}} + \mathbf{x}^{\mathrm{se}})$. Thus, $\mathbf{p}$ is a measurable function of the decomposed components, giving $\sigma(\mathbf{p}) \subseteq \sigma(\mathbf{x}^{\mathrm{tr}}, \mathbf{x}^{\mathrm{se}})$. The two inclusions establish equation 37.

For any measurable target set $\mathcal{B} \subseteq \mathbb{R}^H$, the conditional probability is the conditional expectation of $\mathbb{I}(\mathbf{y} \in \mathcal{B})$. Conditioning on the equal σ-algebras therefore gives the same conditional target law, up to null sets. The corresponding conditional means also agree almost surely:

$$\mathbb{E}[\mathbf{y} \mid \mathbf{p}] = \mathbb{E}[\mathbf{y} \mid \mathbf{x}^{\mathrm{tr}}, \mathbf{x}^{\mathrm{se}}]. \tag{39}$$

Their optimal squared-error risks are therefore equal by equation 34.

This proposition establishes that complete raw patches and the pair of decomposed components retain the same information for predicting the target. It concerns information before neural encoding and does not assert identical behavior for compressed representations. We next specify a predictor class in which explicit decomposition is also unnecessary for predictive expressiveness.

Assumption 2 (Fixed linear decomposition). The trend is extracted using a fixed linear operator $\mathbf{T} \in \mathbb{R}^{L\times L}$:

$$\mathbf{x}^{\mathrm{tr}} = \mathbf{T}\mathbf{x}, \qquad \mathbf{x}^{\mathrm{se}} = (\mathbf{I}_L - \mathbf{T})\mathbf{x}. \tag{40}$$

This includes moving averages with fixed linear boundary rules. The operator need not be an orthogonal projection.

Consider the affine predictors

$$f_{\mathrm{patch}}(\mathbf{x}) = \mathbf{U}\mathbf{Q}\mathbf{x} + \mathbf{b}, \qquad f_{\mathrm{dec}}(\mathbf{x}) = \mathbf{A}\mathbf{T}\mathbf{x} + \mathbf{B}(\mathbf{I}_L - \mathbf{T})\mathbf{x} + \mathbf{b}, \tag{41}$$

where $\mathbf{U} \in \mathbb{R}^{H\times NP}$, $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{H\times L}$, and $\mathbf{b} \in \mathbb{R}^H$ are unrestricted. Denote their function classes by $\mathcal{F}_{\mathrm{patch}}$ and $\mathcal{F}_{\mathrm{dec}}$, respectively.

Proposition D.2 (Predictive expressiveness equivalence). Under Assumptions 1 and 2, $\mathcal{F}_{\mathrm{patch}} = \mathcal{F}_{\mathrm{dec}}$. In particular,

$$\inf_{f\in\mathcal{F}_{\mathrm{patch}}} \mathbb{E}\|\mathbf{y} - f(\mathbf{x})\|_2^2 = \inf_{f\in\mathcal{F}_{\mathrm{dec}}} \mathbb{E}\|\mathbf{y} - f(\mathbf{x})\|_2^2. \tag{42}$$

Proof. Take a decomposition-based predictor and define $\mathbf{C}_{\mathrm{dec}} = \mathbf{A}\mathbf{T} + \mathbf{B}(\mathbf{I}_L - \mathbf{T})$. Using Assumption 1,

$$f_{\mathrm{dec}}(\mathbf{x}) = \mathbf{C}_{\mathrm{dec}}\mathbf{x} + \mathbf{b} = (\mathbf{C}_{\mathrm{dec}}\mathbf{J})\mathbf{Q}\mathbf{x} + \mathbf{b}. \tag{43}$$

The choice $\mathbf{U} = \mathbf{C}_{\mathrm{dec}}\mathbf{J}$ therefore realizes the same predictor from raw patches.

Conversely, take any patch predictor and set $\mathbf{A} = \mathbf{B} = \mathbf{U}\mathbf{Q}$. Then

$$f_{\mathrm{dec}}(\mathbf{x}) = \mathbf{U}\mathbf{Q}[\mathbf{T} + (\mathbf{I}_L - \mathbf{T})]\mathbf{x} + \mathbf{b} = \mathbf{U}\mathbf{Q}\mathbf{x} + \mathbf{b}. \tag{44}$$

Each function class contains the other, proving the claim.

Within these affine classes, explicit component separation does not enlarge the set of predictive mappings available from raw patches. This conclusion does not exclude differences in regularization, optimization, or nonlinear architectures; it identifies conditions under which decomposition is not required for predictive expressiveness.

D.2 Conditional Forecasting with State and Dynamics

Let $\mathbf{u}, \mathbf{v} \in \mathbb{R}^H$ denote the Patch-State and Delta-Dynamics responses entering PaDNet's fusion layer. These are encoded responses rather than the complete raw patch collection.

Assumption 3 (Fixed branch responses). The branch mappings are fixed during the risk comparison. Their responses have finite second moments, and the state-only and joint predictors use the same state response $\mathbf{u}$.

Let $\Pi_{\mathbf{v}|\mathbf{u}}$ denote a regular conditional distribution of $\mathbf{v}$ given $\mathbf{u}$, and use analogous notation for the target distributions. This notation does not require probability densities. All conditional identities below hold for almost every conditioning value.

For any measurable target set $\mathcal{B} \subseteq \mathbb{R}^H$, marginalizing the dynamics response gives

$$\Pi_{\mathbf{y}|\mathbf{u}}(\mathcal{B} \mid \mathbf{u}) = \int \Pi_{\mathbf{y}|\mathbf{u},\mathbf{v}}(\mathcal{B} \mid \mathbf{u}, \mathbf{b})\, \Pi_{\mathbf{v}|\mathbf{u}}(d\mathbf{b} \mid \mathbf{u}). \tag{45}$$

Thus, conditioning only on the state response averages over the dynamics responses compatible with that state.

Define the conditional future means as

$$\boldsymbol{\mu}_{\mathrm{s}}(\mathbf{u}) = \mathbb{E}[\mathbf{y} \mid \mathbf{u}], \qquad \boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{v}) = \mathbb{E}[\mathbf{y} \mid \mathbf{u}, \mathbf{v}]. \tag{46}$$

Integrating the target with respect to the conditional distributions yields

$$\begin{aligned} \boldsymbol{\mu}_{\mathrm{s}}(\mathbf{u}) &= \int \left[\int \mathbf{a}\, \Pi_{\mathbf{y}|\mathbf{u},\mathbf{v}}(d\mathbf{a} \mid \mathbf{u}, \mathbf{b}) \right] \Pi_{\mathbf{v}|\mathbf{u}}(d\mathbf{b} \mid \mathbf{u}) \\ &= \int \boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{b})\, \Pi_{\mathbf{v}|\mathbf{u}}(d\mathbf{b} \mid \mathbf{u}). \end{aligned} \tag{47}$$

Integrability follows from the finite second moment of $\mathbf{y}$. Equivalently, the tower property gives

$$\boldsymbol{\mu}_{\mathrm{s}}(\mathbf{u}) = \mathbb{E}[\boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{v}) \mid \mathbf{u}]. \tag{48}$$

Proposition D.3 (Conditional-mean risk reduction). Under Assumption 3,

$$\mathcal{R}^*_{\mathbf{u}} - \mathcal{R}^*_{\mathbf{u},\mathbf{v}} = \mathbb{E}\|\boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{v}) - \boldsymbol{\mu}_{\mathrm{s}}(\mathbf{u})\|_2^2 \geq 0. \tag{49}$$

The inequality is strict precisely when the two conditional means differ on a set of positive probability.

Proof. For a fixed state response, write its optimal conditional risk as

$$r_{\mathrm{s}}(\mathbf{u}) = \iint \|\mathbf{a} - \boldsymbol{\mu}_{\mathrm{s}}(\mathbf{u})\|_2^2 \, \Pi_{\mathbf{y}|\mathbf{u},\mathbf{v}}(d\mathbf{a} \mid \mathbf{u}, \mathbf{b}) \, \Pi_{\mathbf{v}|\mathbf{u}}(d\mathbf{b} \mid \mathbf{u}). \tag{50}$$

The residual inside the integral decomposes as

$$\mathbf{a} - \boldsymbol{\mu}_{\mathrm{s}}(\mathbf{u}) = [\mathbf{a} - \boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{b})] + [\boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{b}) - \boldsymbol{\mu}_{\mathrm{s}}(\mathbf{u})]. \tag{51}$$

The first bracket has zero conditional mean:

$$\int [\mathbf{a} - \boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{b})] \, \Pi_{\mathbf{y}|\mathbf{u},\mathbf{v}}(d\mathbf{a} \mid \mathbf{u}, \mathbf{b}) = \mathbf{0}. \tag{52}$$

The second bracket is constant with respect to the inner integration variable $\mathbf{a}$. Consequently, the inner integral of the cross term is zero.

Define

$$r_{\mathrm{sd}}(\mathbf{u}, \mathbf{b}) = \int \|\mathbf{a} - \boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{b})\|_2^2 \, \Pi_{\mathbf{y}|\mathbf{u},\mathbf{v}}(d\mathbf{a} \mid \mathbf{u}, \mathbf{b}). \tag{53}$$

Expanding equation 50 therefore gives

$$r_{\mathrm{s}}(\mathbf{u}) = \int r_{\mathrm{sd}}(\mathbf{u}, \mathbf{b}) \, \Pi_{\mathbf{v}|\mathbf{u}}(d\mathbf{b} \mid \mathbf{u}) + \int \|\boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{b}) - \boldsymbol{\mu}_{\mathrm{s}}(\mathbf{u})\|_2^2 \, \Pi_{\mathbf{v}|\mathbf{u}}(d\mathbf{b} \mid \mathbf{u}). \tag{54}$$

Taking expectations over $\mathbf{u}$ yields

$$\mathcal{R}^*_{\mathbf{u}} = \mathcal{R}^*_{\mathbf{u},\mathbf{v}} + \mathbb{E}\|\boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{v}) - \boldsymbol{\mu}_{\mathrm{s}}(\mathbf{u})\|_2^2. \tag{55}$$

The final term is the expectation of a nonnegative random variable. It equals zero exactly when the conditional means agree almost surely, establishing both claims.

This identity measures the predictive information exposed by the dynamics response beyond the state response. Its right-hand side is the average variation of conditional future means across compatible dynamics. The next result identifies the part of this predictive structure accessible to PaDNet's linear fusion, which need not represent an arbitrary conditional mean.

D.3 Risk Reduction under Response-Level Linear Fusion

PaDNet concatenates the two horizon-wise responses and applies an affine fusion map. Partitioning its weight matrix gives

$$\widehat{\mathbf{y}} = \mathbf{A}\mathbf{u} + \mathbf{B}\mathbf{v} + \mathbf{b}, \qquad \mathbf{A}, \mathbf{B} \in \mathbb{R}^{H \times H}. \tag{56}$$

For the fixed responses in Assumption 3, define

$$\mathcal{R}^{\mathrm{lin}}_{\mathrm{s}} = \inf_{\mathbf{A},\mathbf{b}} \mathbb{E}\|\mathbf{y} - \mathbf{A}\mathbf{u} - \mathbf{b}\|_2^2, \qquad \mathcal{R}^{\mathrm{lin}}_{\mathrm{sd}} = \inf_{\mathbf{A},\mathbf{B},\mathbf{b}} \mathbb{E}\|\mathbf{y} - \mathbf{A}\mathbf{u} - \mathbf{B}\mathbf{v} - \mathbf{b}\|_2^2. \tag{57}$$

The state-only class permits an optimal affine recalibration of the same state response. Thus, the comparison isolates the contribution of the dynamics response at the fusion stage.

For fixed $\mathbf{A}$ and $\mathbf{B}$, the optimal intercept is

$$\mathbf{b}_* = \mathbb{E}[\mathbf{y}] - \mathbf{A}\mathbb{E}[\mathbf{u}] - \mathbf{B}\mathbb{E}[\mathbf{v}]. \tag{58}$$

Accordingly, centering all variables leaves the optimal risks unchanged. In the remainder of the analysis, $\mathbf{y}$, $\mathbf{u}$, and $\mathbf{v}$ denote their centered versions.

Write $\mathbf{\Sigma}_{ab} = \mathbb{E}[\mathbf{a}\mathbf{b}^\top]$, and let † denote the Moore–Penrose pseudoinverse. Let $\mathbf{A}_{\mathrm{s}}\mathbf{u}$ and $\mathbf{K}\mathbf{u}$ be the least-squares linear projections of $\mathbf{y}$ and $\mathbf{v}$ onto the coordinates of $\mathbf{u}$. Such projections exist because these coordinates span a finite-dimensional, hence closed, subspace of L^2. Their coefficients satisfy the normal equations

$$\mathbf{A}_{\mathrm{s}}\mathbf{\Sigma}_{uu} = \mathbf{\Sigma}_{yu}, \qquad \mathbf{K}\mathbf{\Sigma}_{uu} = \mathbf{\Sigma}_{vu}. \tag{59}$$

Define the residuals

$$\mathbf{e} = \mathbf{y} - \mathbf{A}_{\mathrm{s}}\mathbf{u}, \qquad \mathbf{v}_\perp = \mathbf{v} - \mathbf{K}\mathbf{u}. \tag{60}$$

The normal equations imply

$$\mathbb{E}[\mathbf{e}\mathbf{u}^\top] = \mathbf{0}, \qquad \mathbb{E}[\mathbf{v}_\perp\mathbf{u}^\top] = \mathbf{0}. \tag{61}$$

Here, $\mathbf{e}$ is the optimal state-only forecasting residual, and $\mathbf{v}_\perp$ is the dynamics response not linearly explained by the state response. Although projection coefficients need not be unique when $\mathbf{\Sigma}_{uu}$ is singular, the projected random vectors and their residuals are unique almost surely.

Their relevant second moments are

$$\mathbf{\Sigma}_\perp = \mathbb{E}[\mathbf{v}_\perp\mathbf{v}_\perp^\top], \qquad \mathbf{C} = \mathbb{E}[\mathbf{e}\mathbf{v}_\perp^\top]. \tag{62}$$

Theorem D.4 (Exact risk reduction under linear fusion). Under Assumption 3 and finite target second moment,

$$\mathcal{R}_{\mathrm{s}}^{\mathrm{lin}} - \mathcal{R}_{\mathrm{sd}}^{\mathrm{lin}} = \mathrm{tr}(\mathbf{C}\mathbf{\Sigma}_\perp^\dagger\mathbf{C}^\top) \geq 0. \tag{63}$$

The improvement is strict if and only if $\mathbf{C} \neq \mathbf{0}$.

Proof. Setting $\mathbf{B} = \mathbf{0}$ recovers the state-only predictor, establishing inclusion of the two predictor classes. To derive the exact gain, substitute $\mathbf{v} = \mathbf{K}\mathbf{u} + \mathbf{v}_\perp$:

$$\mathbf{A}\mathbf{u} + \mathbf{B}\mathbf{v} = (\mathbf{A} + \mathbf{B}\mathbf{K})\mathbf{u} + \mathbf{B}\mathbf{v}_\perp. \tag{64}$$

Combining this expression with the definition of $\mathbf{e}$ gives

$$\mathbf{y} - \mathbf{A}\mathbf{u} - \mathbf{B}\mathbf{v} = \mathbf{e} - \mathbf{B}\mathbf{v}_\perp + (\mathbf{A}_{\mathrm{s}} - \mathbf{A} - \mathbf{B}\mathbf{K})\mathbf{u}. \tag{65}$$

equation 61 implies $\mathbb{E}[(\mathbf{e} - \mathbf{B}\mathbf{v}_\perp)\mathbf{u}^\top] = \mathbf{0}$. The cross term therefore vanishes after taking expectations:

$$\mathbb{E}\|\mathbf{y} - \mathbf{A}\mathbf{u} - \mathbf{B}\mathbf{v}\|_2^2 = \mathbb{E}\|\mathbf{e} - \mathbf{B}\mathbf{v}_\perp\|_2^2 + \mathbb{E}\|(\mathbf{A}_{\mathrm{s}} - \mathbf{A} - \mathbf{B}\mathbf{K})\mathbf{u}\|_2^2. \tag{66}$$

For every $\mathbf{B}$, choosing $\mathbf{A} = \mathbf{A}_{\mathrm{s}} - \mathbf{B}\mathbf{K}$ sets the second term to zero. Hence,

$$\mathcal{R}_{\mathrm{sd}}^{\mathrm{lin}} = \inf_{\mathbf{B}} \mathbb{E}\|\mathbf{e} - \mathbf{B}\mathbf{v}_\perp\|_2^2, \qquad \mathcal{R}_{\mathrm{s}}^{\mathrm{lin}} = \mathbb{E}\|\mathbf{e}\|_2^2. \tag{67}$$

Expanding the remaining objective and using the trace identity for scalar quadratic forms gives

$$\begin{aligned}\mathbb{E}\|\mathbf{e} - \mathbf{B}\mathbf{v}_\perp\|_2^2 &= \mathbb{E}[\mathbf{e}^\top\mathbf{e}] - 2\mathbb{E}[\mathbf{e}^\top\mathbf{B}\mathbf{v}_\perp] + \mathbb{E}[\mathbf{v}_\perp^\top\mathbf{B}^\top\mathbf{B}\mathbf{v}_\perp] \\ &= \mathcal{R}_{\mathrm{s}}^{\mathrm{lin}} - 2\,\mathrm{tr}(\mathbf{B}\mathbf{C}^\top) + \mathrm{tr}(\mathbf{B}\mathbf{\Sigma}_\perp\mathbf{B}^\top).\end{aligned} \tag{68}$$

Differentiation with respect to $\mathbf{B}$ yields

$$\nabla_{\mathbf{B}}\mathbb{E}\|\mathbf{e} - \mathbf{B}\mathbf{v}_\perp\|_2^2 = 2\mathbf{B}\mathbf{\Sigma}_\perp - 2\mathbf{C}. \tag{69}$$

Thus, an optimal coefficient matrix satisfies

$$\mathbf{B}\mathbf{\Sigma}_\perp = \mathbf{C}. \tag{70}$$

To establish solvability without assuming an invertible covariance, take any $\mathbf{a} \in \ker(\mathbf{\Sigma}_\perp)$. By the definition of $\mathbf{\Sigma}_\perp$,

$$\mathbb{E}[(\mathbf{a}^\top\mathbf{v}_\perp)^2] = \mathbf{a}^\top\mathbb{E}[\mathbf{v}_\perp\mathbf{v}_\perp^\top]\mathbf{a} = \mathbf{a}^\top\mathbf{\Sigma}_\perp\mathbf{a} = 0. \tag{71}$$

The Cauchy–Schwarz inequality then gives

$$\|\mathbf{C}\mathbf{a}\|_2^2 = \left\|\mathbb{E}[\mathbf{e}(\mathbf{a}^\top \mathbf{v}_\perp)]\right\|_2^2 \le \mathbb{E}\|\mathbf{e}\|_2^2\, \mathbb{E}[(\mathbf{a}^\top \mathbf{v}_\perp)^2] = 0. \tag{72}$$

Therefore, $\ker(\mathbf{\Sigma}_\perp) \subseteq \ker(\mathbf{C})$. Because $\mathbf{\Sigma}_\perp$ is symmetric, $\mathbf{\Sigma}_\perp^\dagger \mathbf{\Sigma}_\perp$ is the orthogonal projector onto its range. The null-space inclusion implies

$$\mathbf{C} = \mathbf{C}\mathbf{\Sigma}_\perp^\dagger \mathbf{\Sigma}_\perp. \tag{73}$$

Consequently, the choice $\mathbf{B}_* = \mathbf{C}\mathbf{\Sigma}_\perp^\dagger$ satisfies the normal equation:

$$\mathbf{B}_* \mathbf{\Sigma}_\perp = \mathbf{C}\mathbf{\Sigma}_\perp^\dagger \mathbf{\Sigma}_\perp = \mathbf{C}. \tag{74}$$

Let $\mathbf{G} = \mathbf{B} - \mathbf{B}_*$. Substituting $\mathbf{B} = \mathbf{B}_* + \mathbf{G}$ into equation 68, the terms linear in $\mathbf{G}$ cancel because $\mathbf{B}_* \mathbf{\Sigma}_\perp = \mathbf{C}$. Moreover,

$$\mathrm{tr}(\mathbf{B}_* \mathbf{\Sigma}_\perp \mathbf{B}_*^\top) = \mathrm{tr}(\mathbf{B}_* \mathbf{C}^\top) = \mathrm{tr}(\mathbf{C}\mathbf{\Sigma}_\perp^\dagger \mathbf{C}^\top). \tag{75}$$

It follows that

$$\mathbb{E}\|\mathbf{e} - \mathbf{B}\mathbf{v}_\perp\|_2^2 = \mathcal{R}_{\mathrm{s}}^{\mathrm{lin}} - \mathrm{tr}(\mathbf{C}\mathbf{\Sigma}_\perp^\dagger \mathbf{C}^\top) + \mathrm{tr}(\mathbf{G}\mathbf{\Sigma}_\perp \mathbf{G}^\top). \tag{76}$$

The final term equals $\mathbb{E}\|\mathbf{G}\mathbf{v}_\perp\|_2^2$ and is nonnegative. It vanishes at $\mathbf{G} = \mathbf{0}$, proving the exact risk identity.

For strictness, first suppose that $\mathbf{\Sigma}_\perp$ has positive rank. Write its eigendecomposition on its range as $\mathbf{\Sigma}_\perp = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$, where $\mathbf{V}$ has orthonormal columns and $\mathbf{\Lambda}$ has strictly positive diagonal entries. equation 73 gives $\mathbf{C} = \mathbf{C}\mathbf{V}\mathbf{V}^\top$. Therefore,

$$\mathrm{tr}(\mathbf{C}\mathbf{\Sigma}_\perp^\dagger \mathbf{C}^\top) = \|\mathbf{C}\mathbf{V}\mathbf{\Lambda}^{-1/2}\|_{\mathrm{F}}^2. \tag{77}$$

This expression vanishes exactly when $\mathbf{C}\mathbf{V} = \mathbf{0}$, which is equivalent to $\mathbf{C} = \mathbf{0}$ by the range identity. If $\mathbf{\Sigma}_\perp = \mathbf{0}$, its null space is the entire response space, so equation 72 also gives $\mathbf{C} = \mathbf{0}$. This completes the proof.

The theorem identifies the contribution usable by the fusion layer: the dynamics response must contain a component outside the linear span of the state response that correlates with the remaining forecasting error. Distinct branch outputs alone do not establish a gain.

The relationship with conditional forecasting follows from the tower property. Since $\mathbf{v}_\perp$ is a function of $(\mathbf{u}, \mathbf{v})$,

$$\begin{aligned} \mathbf{C} &= \mathbb{E}\big[\mathbb{E}[(\mathbf{y} - \mathbf{A}_{\mathrm{s}}\mathbf{u})\mathbf{v}_\perp^\top \mid \mathbf{u}, \mathbf{v}]\big] \\ &= \mathbb{E}\big[(\boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{v}) - \mathbf{A}_{\mathrm{s}}\mathbf{u})\mathbf{v}_\perp^\top\big]. \end{aligned} \tag{78}$$

Hence, the linear-fusion gain depends on how the unexplained dynamics align with the conditional future mean remaining after the optimal linear state prediction.

D.4 A Structural Condition for Strict Improvement

We now specify a predictive structure under which the strict improvement condition holds.

Assumption 4 (Additive predictive structure). For the centered variables, the target satisfies

$$\mathbf{y} = \mathbf{A}_0\mathbf{u} + \mathbf{B}_0\mathbf{v} + \boldsymbol{\varepsilon}, \qquad \mathbb{E}[\boldsymbol{\varepsilon} \mid \mathbf{u}, \mathbf{v}] = \mathbf{0}, \tag{79}$$

where $\mathbf{A}_0, \mathbf{B}_0 \in \mathbb{R}^{H \times H}$. The unexplained dynamics have a nonzero contribution:

$$\mathbf{B}_0 \mathbf{\Sigma}_\perp \mathbf{B}_0^\top \neq \mathbf{0}. \tag{80}$$

This condition requires both residual variation in the dynamics response and a nonzero predictive effect of that variation.

Corollary D.5 (Strict improvement from predictive dynamics). Under Assumptions 3 and 4,

$$\mathcal{R}_{\mathrm{sd}}^{\mathrm{lin}} = \mathbb{E}\|\boldsymbol{\varepsilon}\|_2^2, \qquad \mathcal{R}_{\mathrm{s}}^{\mathrm{lin}} - \mathcal{R}_{\mathrm{sd}}^{\mathrm{lin}} = \mathrm{tr}(\mathbf{B}_0 \mathbf{\Sigma}_\perp \mathbf{B}_0^\top) > 0. \tag{81}$$

Proof. Substituting $\mathbf{v} = \mathbf{K}\mathbf{u} + \mathbf{v}_\perp$ into equation 79 gives

$$\mathbf{y} = (\mathbf{A}_0 + \mathbf{B}_0\mathbf{K})\mathbf{u} + \mathbf{B}_0\mathbf{v}_\perp + \boldsymbol{\varepsilon}. \tag{82}$$

By conditional zero mean and the tower property,

$$\mathbb{E}[\boldsymbol{\varepsilon}\mathbf{u}^\top] = \mathbf{0}, \qquad \mathbb{E}[\boldsymbol{\varepsilon}\mathbf{v}_\perp^\top] = \mathbf{0}. \tag{83}$$

Together with $\mathbb{E}[\mathbf{v}_\perp\mathbf{u}^\top] = \mathbf{0}$, these identities show that $(\mathbf{A}_0 + \mathbf{B}_0\mathbf{K})\mathbf{u}$ is the optimal linear projection of $\mathbf{y}$ onto the state response. Thus,

$$\mathbf{e} = \mathbf{B}_0\mathbf{v}_\perp + \boldsymbol{\varepsilon}, \qquad \mathbf{C} = \mathbb{E}[\mathbf{e}\mathbf{v}_\perp^\top] = \mathbf{B}_0\boldsymbol{\Sigma}_\perp. \tag{84}$$

Substituting into the exact linear-fusion risk identity,

$$\begin{aligned}\mathcal{R}_{\mathrm{s}}^{\mathrm{lin}} - \mathcal{R}_{\mathrm{sd}}^{\mathrm{lin}} &= \mathrm{tr}\Big(\mathbf{B}_0\boldsymbol{\Sigma}_\perp\boldsymbol{\Sigma}_\perp^\dagger\boldsymbol{\Sigma}_\perp\mathbf{B}_0^\top\Big) \\ &= \mathrm{tr}(\mathbf{B}_0\boldsymbol{\Sigma}_\perp\mathbf{B}_0^\top),\end{aligned} \tag{85}$$

where the second equality uses the pseudoinverse identity $\boldsymbol{\Sigma}_\perp\boldsymbol{\Sigma}_\perp^\dagger\boldsymbol{\Sigma}_\perp = \boldsymbol{\Sigma}_\perp$.

The joint conditional mean is $\mathbf{A}_0\mathbf{u} + \mathbf{B}_0\mathbf{v}$, which belongs to the joint affine predictor class. Its residual is $\boldsymbol{\varepsilon}$, so the joint optimal risk equals $\mathbb{E}\|\boldsymbol{\varepsilon}\|_2^2$. Finally, $\mathbf{B}_0\boldsymbol{\Sigma}_\perp\mathbf{B}_0^\top$ is positive semidefinite and nonzero under Assumption 4. At least one of its eigenvalues is positive, making its trace strictly positive.

The conditional-mean analysis and linear-fusion analysis coincide under an additional linear conditional-mean condition on the dynamics response. Specifically, suppose

$$\mathbb{E}[\mathbf{v} \mid \mathbf{u}] = \mathbf{K}\mathbf{u}. \tag{86}$$

Combining this condition with Assumption 4 gives

$$\begin{aligned}\boldsymbol{\mu}_{\mathrm{s}}(\mathbf{u}) &= \int (\mathbf{A}_0\mathbf{u} + \mathbf{B}_0\mathbf{b})\,\Pi_{\mathbf{v}|\mathbf{u}}(d\mathbf{b} \mid \mathbf{u}) \\ &= \mathbf{A}_0\mathbf{u} + \mathbf{B}_0\mathbb{E}[\mathbf{v} \mid \mathbf{u}] = (\mathbf{A}_0 + \mathbf{B}_0\mathbf{K})\mathbf{u}.\end{aligned} \tag{87}$$

Therefore,

$$\boldsymbol{\mu}_{\mathrm{sd}}(\mathbf{u}, \mathbf{v}) - \boldsymbol{\mu}_{\mathrm{s}}(\mathbf{u}) = \mathbf{B}_0(\mathbf{v} - \mathbf{K}\mathbf{u}) = \mathbf{B}_0\mathbf{v}_\perp. \tag{88}$$

Both conditional means now belong to their respective affine predictor classes. Their risk reductions consequently agree:

$$\mathcal{R}_{\mathbf{u}}^* - \mathcal{R}_{\mathbf{u},\mathbf{v}}^* = \mathcal{R}_{\mathrm{s}}^{\mathrm{lin}} - \mathcal{R}_{\mathrm{sd}}^{\mathrm{lin}} = \mathbb{E}\|\mathbf{B}_0\mathbf{v}_\perp\|_2^2. \tag{89}$$

This establishes a setting in which the conditional predictive value of dynamics is fully realizable by the fusion form used in PaDNet.

D.5 Implications and Scope

The analysis establishes two complementary conclusions. First, complete raw patches retain the information used by deterministic trend–seasonal decomposition, and fixed linear decomposition does not enlarge the unrestricted affine predictor class available from those patches. Second, for fixed branch responses, adding Delta-Dynamics through linear fusion cannot increase the optimal population risk and strictly reduces it when the unexplained dynamics response correlates with the state-only forecasting residual.

These conclusions apply at different representation stages. The first concerns raw observations before compression; the second concerns responses produced by learned branch mappings. Delta-Dynamics is derived from the same historical input and introduces no new information relative to the complete observations. Its predictive contribution arises when its response exposes variation not linearly accessible from the Patch-State response.

The risk results concern optimal, unregularized population predictors sharing the same fixed state response. They do not equate separately trained encoders or guarantee finite-sample test improvements. The empirical comparisons and ablations evaluate whether the learned PaDNet responses realize the predictive conditions identified here.

Table 6: Complete multivariate forecasting results in MSE and MAE. The look-back length is 96 for the ETT and Exchange datasets, and 336 for Solar, Electricity, and the PEMS datasets. Prediction horizons are $\{12, 24, 48, 96\}$ for PEMS and $\{96, 192, 336, 720\}$ for the other datasets. Avg denotes the average across prediction horizons. The best and second-best results are highlighted in red and blue, respectively.

Models		PaDNet		ParallelTime		XPatch		DeformableTST		SST		PatchMLP		RLinear		PatchTST		DLinear	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.3707	0.3948	0.3870	0.4042	0.3805	0.4033	0.3808	0.3973	0.3810	0.3975	0.3946	0.4049	0.3856	0.3941	0.3810	0.3963	0.3851	0.3992
	192	0.4226	0.4236	0.4407	0.4337	0.4412	0.4371	0.4246	0.4278	0.4465	0.4365	0.4471	0.4370	0.4343	0.4215	0.4291	0.4253	0.4367	0.4308
	336	0.4621	0.4441	0.4886	0.4601	0.4907	0.4587	0.4793	0.4474	0.4914	0.4590	0.4995	0.4671	0.4751	0.4422	0.4769	0.4493	0.4864	0.4620
	720	0.4703	0.4646	0.5117	0.4986	0.5102	0.4861	0.4760	0.4664	0.4936	0.4809	0.5289	0.5033	0.4765	0.4666	0.4840	0.4735	0.5147	0.5096
	Avg	0.4314	0.4318	0.4570	0.4492	0.4557	0.4463	0.4402	0.4347	0.4531	0.4435	0.4675	0.4531	0.4429	0.4311	0.4428	0.4361	0.4557	0.4504
ETTm1	96	0.3163	0.3561	0.3380	0.3718	0.3335	0.3705	0.3173	0.3597	0.3305	0.3667	0.3250	0.3636	0.3530	0.3700	0.3222	0.3598	0.3431	0.3709
	192	0.3537	0.3791	0.3700	0.3903	0.3611	0.3841	0.3560	0.3811	0.3708	0.3868	0.3719	0.3883	0.3875	0.3863	0.3613	0.3828	0.3820	0.3933
	336	0.3833	0.4006	0.3991	0.4097	0.3950	0.4089	0.3815	0.3999	0.4031	0.4096	0.4037	0.4099	0.4197	0.4069	0.3936	0.4055	0.4122	0.4127
	720	0.4480	0.4389	0.4528	0.4432	0.4603	0.4487	0.4438	0.4374	0.4612	0.4418	0.4905	0.4545	0.4805	0.4409	0.4626	0.4419	0.4739	0.4524
	Avg	0.3753	0.3937	0.3900	0.4038	0.3875	0.4031	0.3747	0.3945	0.3914	0.4012	0.3978	0.4041	0.4102	0.4010	0.3849	0.3975	0.4028	0.4073
ETTm2	96	0.1737	0.2571	0.1780	0.2627	0.1762	0.2605	0.1771	0.2610	0.1749	0.2598	0.1738	0.2598	0.1818	0.2646	0.1793	0.2615	0.1885	0.2835
	192	0.2394	0.3005	0.2454	0.3081	0.2421	0.3035	0.2438	0.3053	0.2442	0.3044	0.2436	0.3051	0.2469	0.3053	0.2452	0.3050	0.2708	0.3467
	336	0.2988	0.3386	0.3074	0.3489	0.3004	0.3419	0.3057	0.3456	0.3019	0.3407	0.3034	0.3426	0.3085	0.3440	0.3104	0.3469	0.3618	0.4118
	720	0.3953	0.3952	0.4127	0.4110	0.3976	0.3978	0.4071	0.4032	0.3971	0.3971	0.4084	0.4032	0.4075	0.3992	0.4124	0.4053	0.5089	0.4964
	Avg	0.2768	0.3229	0.2859	0.3327	0.2791	0.3259	0.2834	0.3288	0.2795	0.3255	0.2823	0.3277	0.2862	0.3283	0.2868	0.3297	0.3325	0.3846
Solar	96	0.1838	0.2348	0.1974	0.2463	0.1868	0.2429	0.1929	0.2553	0.1959	0.2570	0.1986	0.2444	0.2317	0.2646	0.2458	0.3024	0.2209	0.2932
	192	0.1917	0.2502	0.2250	0.2759	0.1953	0.2513	0.1983	0.2653	0.2255	0.2657	0.2218	0.2648	0.2618	0.2789	0.2704	0.3190	0.2518	0.3128
	336	0.1991	0.2570	0.2130	0.2647	0.2022	0.2592	0.2080	0.2629	0.2355	0.2764	0.2181	0.2746	0.2820	0.2915	0.2818	0.3287	0.2684	0.3236
	720	0.2032	0.2586	0.2213	0.2767	0.2086	0.2623	0.2074	0.2700	0.2313	0.2702	0.2202	0.2697	0.2808	0.2905	0.2788	0.3341	0.2715	0.3274
	Avg	0.1945	0.2502	0.2142	0.2659	0.1982	0.2539	0.2017	0.2634	0.2221	0.2673	0.2147	0.2634	0.2641	0.2814	0.2692	0.3211	0.2532	0.3143
Electricity	96	0.1302	0.2263	0.1317	0.2277	0.1309	0.2266	0.1352	0.2370	0.1410	0.2385	0.1404	0.2416	0.1406	0.2360	0.1314	0.2271	0.1403	0.2379
	192	0.1485	0.2434	0.1505	0.2452	0.1478	0.2422	0.1535	0.2510	0.1611	0.2550	0.1656	0.2629	0.1541	0.2478	0.1490	0.2422	0.1539	0.2522
	336	0.1656	0.2605	0.1699	0.2643	0.1658	0.2613	0.1749	0.2748	0.1768	0.2720	0.1814	0.2799	0.1713	0.2650	0.1678	0.2625	0.1692	0.2673
	720	0.2002	0.2916	0.2066	0.2979	0.2021	0.2934	0.2106	0.3038	0.2133	0.3029	0.2136	0.3092	0.2103	0.2972	0.2059	0.2952	0.2037	0.3008
	Avg	0.1611	0.2555	0.1647	0.2588	0.1617	0.2559	0.1686	0.2667	0.1731	0.2671	0.1753	0.2734	0.1691	0.2615	0.1635	0.2568	0.1668	0.2646
Exchange	96	0.0787	0.1952	0.0837	0.2031	0.0878	0.2040	0.0882	0.2063	0.0843	0.2022	0.0912	0.2124	0.0822	0.1994	0.0816	0.1977	0.0776	0.1994
	192	0.1729	0.2949	0.1822	0.3040	0.1842	0.3043	0.1953	0.3150	0.1773	0.2986	0.1856	0.3086	0.1784	0.2987	0.1730	0.2944	0.1599	0.2969
	336	0.3191	0.4074	0.3356	0.4207	0.3419	0.4223	0.3229	0.4109	0.3233	0.4098	0.3389	0.4217	0.3428	0.4200	0.3221	0.4094	0.2965	0.4099
	720	0.8394	0.6880	0.8623	0.6992	0.9996	0.7470	0.8586	0.6995	0.9527	0.7309	0.8715	0.7011	0.9068	0.7160	0.8440	0.6910	0.8443	0.7028
	Avg	0.3525	0.3964	0.3660	0.4068	0.4034	0.4194	0.3663	0.4079	0.3844	0.4104	0.3718	0.4110	0.3776	0.4085	0.3552	0.3981	0.3446	0.4023
Pems03	12	0.0578	0.1576	0.0666	0.1731	0.0594	0.1596	0.0647	0.1715	0.0696	0.1767	0.0601	0.1627	0.0794	0.1835	0.0778	0.1865	0.0752	0.1805
	24	0.0718	0.1736	0.0840	0.1913	0.0749	0.1778	0.0769	0.1818	0.0927	0.2024	0.0786	0.1827	0.1109	0.2123	0.1048	0.2161	0.1072	0.2147
	48	0.0952	0.1975	0.1176	0.2236	0.1006	0.2044	0.1048	0.2089	0.1337	0.2398	0.1047	0.2122	0.1553	0.2466	0.1463	0.2516	0.1515	0.2532
	96	0.1217	0.2218	0.1497	0.2499	0.1287	0.2311	0.1290	0.2320	0.1817	0.2776	0.1357	0.2437	0.1912	0.2719	0.1840	0.2782	0.1873	0.2848
	Avg	0.0866	0.1876	0.1045	0.2095	0.0909	0.1932	0.0939	0.1986	0.1194	0.2241	0.0948	0.2003	0.1342	0.2286	0.1282	0.2331	0.1303	0.2333
Pems04	12	0.0700	0.1708	0.0803	0.1881	0.0730	0.1753	0.0799	0.1877	0.0800	0.1878	0.0704	0.1749	0.0926	0.2006	0.0917	0.2031	0.0896	0.1978
	24	0.0826	0.1845	0.1021	0.2161	0.0869	0.1914	0.1040	0.2151	0.1014	0.2136	0.0831	0.1919	0.1242	0.2322	0.1218	0.2346	0.1205	0.2294
	48	0.1009	0.2049	0.1332	0.2445	0.1091	0.2140	0.1250	0.2362	0.1324	0.2475	0.0996	0.2093	0.1695	0.2687	0.1659	0.2719	0.1639	0.2695
	96	0.1220	0.2227	0.1661	0.2719	0.1379	0.2410	0.1434	0.2485	0.1776	0.2861	0.1218	0.2338	0.2065	0.2949	0.1985	0.2992	0.1986	0.2989
	Avg	0.0939	0.1957	0.1204	0.2302	0.1017	0.2054	0.1131	0.2219	0.1229	0.2338	0.0937	0.2025	0.1482	0.2491	0.1445	0.2522	0.1432	0.2489
Pems07	12	0.0511	0.1449	0.0598	0.1635	0.0537	0.1493	0.0642	0.1697	0.0596	0.1616	0.0548	0.1532	0.0737	0.1803	0.0710	0.1790	0.0711	0.1762
	24	0.0620	0.1592	0.0742	0.1810	0.0666	0.1675	0.0663	0.1685	0.0767	0.1870	0.0675	0.1718	0.1044	0.2145	0.0969	0.2135	0.1027	0.2115
	48	0.0785	0.1792	0.0997	0.2107	0.0833	0.1872	0.0897	0.1975	0.1030	0.2160	0.0894	0.2003	0.1488	0.2508	0.1374	0.2491	0.1503	0.2570
	96	0.0918	0.1928	0.1232	0.2331	0.1024	0.2082	0.1143	0.2234	0.1237	0.2356	0.1137	0.2273	0.1856	0.2801	0.1810	0.2935	0.1910	0.2902
	Avg	0.0709	0.1690	0.0892	0.1971	0.0765	0.1781	0.0836	0.1898	0.0908	0.2001	0.0814	0.1882	0.1281	0.2314	0.1216	0.2338	0.1288	0.2337
Pems08	12	0.0689	0.1685	0.0806	0.1839	0.0727	0.1686	0.0853	0.1908	0.0796	0.1823	0.0851	0.1837	0.1025	0.2023	0.0950	0.1981	0.0922	0.1945
	24	0.0861	0.1795	0.1120	0.2086	0.0921	0.1869	0.0996	0.2019	0.1130	0.2106	0.1171	0.1994	0.1563	0.2408	0.1433	0.2337	0.1416	0.2351
	48	0.1156	0.2019	0.1564	0.2338	0.1184	0.2058	0.1360	0.2202	0.1804	0.2486	0.1557	0.2317	0.2590	0.2923	0.2321	0.2905	0.2331	0.2894
	96	0.1704	0.2241	0.2149	0.2570	0.1712	0.2322	0.1991	0.2498	0.2649	0.3041	0.2485	0.2587	0.3841	0.3367	0.3604	0.3371	0.3303	0.3331
	Avg	0.1103	0.1935	0.1410	0.2208	0.1136	0.1984	0.1300	0.2157	0.1595	0.2364	0.1516	0.2184	0.2255	0.2680	0.2077	0.2649	0.1993	0.2630
Count 1st		39	42	0	0	1	1	3	2	0	0	3	0	0	4	0	2	4	0

E Additional Experimental Results

E.1 Complete Forecasting Results

Table 6 presents the complete MSE and MAE comparison between PaDNet and all evaluated baselines across different datasets and prediction horizons. Compared with the averaged results reported in the main text, these detailed results provide a more comprehensive view of model performance over shorter and longer forecasting horizons, covering both low- and high-dimensional multivariate time series.

PaDNet maintains competitive performance across these settings, with particularly notable improvements on PEMS07. The results on lower-dimensional datasets support its ability to capture temporal patterns within individual series, while those on higher-dimensional datasets demonstrate its applicability to forecasting tasks involving many variables. Although the relative gains vary across datasets and prediction horizons, these results support the effectiveness of combining patch-state representations with explicit delta-dynamics modeling across diverse forecasting conditions.

E.2 Patch-Level Analysis of Explicit Decomposition

This section provides the computation details for Figure 1. Panels (a,b) compare the periodic contrast patterns of raw and seasonal patches, while panel (c) examines the agreement between the local slopes of raw and trend patches.

E.2.1 Analysis Setup

We use the second numerical signal channel of ETTh1 and the first 60% of the series, corresponding to the training split. Input windows contain 96 observations and are sampled with a stride of 8. Each window is standardized using its own mean and standard deviation. Windows containing non-finite values or negligible variation are excluded.

Following the EMA decomposition used in XPatch (Stitsyuk & Choi, 2025), we extract the trend component with a smoothing coefficient of $\alpha = 0.3$ and take the residual as the seasonal component. EMA is applied independently within each standardized window, with its initial value set to the first observation. Raw, trend, and seasonal patches are extracted at identical positions using a patch length of 16 and a stride of 8, without padding.

E.2.2 Panel (a): Periodic Contrast of Raw Patches

For each window, the last raw patch serves as the query. Candidate patches are earlier patches that do not overlap with the query. Let ℓ_j denote the start-position lag between candidate j and the query. Using a reference period of $T_0 = 24$ observations, we compute the distance to the nearest positive period multiple:

$$d_j = |\ell_j - T_0 \max(1, \lfloor \ell_j/T_0 + 1/2 \rfloor)| \,. \tag{90}$$

Candidates satisfying $d_j \le 1$ form the periodic set $\mathcal{P}$, whereas those satisfying $d_j > 2$ form the non-periodic set $\mathcal{N}$. Candidates between these thresholds are excluded.

For a query patch $\mathbf{q}$ and candidate patch $\mathbf{p}_j$, their similarity is measured by Pearson correlation, denoted by $\rho(\mathbf{q}, \mathbf{p}_j)$. Each valid window contributes two mean similarities:

$$s_{\text{non}} = \frac{1}{|\mathcal{N}|} \sum_{j \in \mathcal{N}} \rho(\mathbf{q}, \mathbf{p}_j), \qquad s_{\text{per}} = \frac{1}{|\mathcal{P}|} \sum_{j \in \mathcal{P}} \rho(\mathbf{q}, \mathbf{p}_j). \tag{91}$$

Panel (a) plots s_{non} on the horizontal axis and s_{per} on the vertical axis. A sample above the identity line therefore indicates greater similarity to candidates at approximately periodic lags.

The periodic contrast of window w is defined as $\mathrm{c}_w = s_{\text{per},w} - s_{\text{non},w}$. For W valid windows, the mean contrast and the fraction above the identity line are

$$\bar{c} = \frac{1}{W} \sum_{w=1}^{W} c_w, \qquad f_+ = \frac{1}{W} \sum_{w=1}^{W} \mathbb{I}[c_w > 0]. \tag{92}$$

In the figure, $\bar{c}$ and f_+ are labeled Mean contrast and Above identity, respectively, with f_+ displayed as a percentage.

E.2.3 Panel (b): Periodic Contrast of Seasonal Patches

Panel (b) applies the same computation to the seasonal patches obtained after EMA trend removal. The query position, candidate positions, and lag-based grouping remain identical to those in panel (a).

To ensure a matched comparison, a candidate is retained only when its query–candidate correlation is defined for both raw and seasonal patches. Each retained window must contain at least two valid periodic candidates and two valid non-periodic candidates. Panels (a) and (b) consequently use the same windows and candidate positions.

The distinction between these panels is the signal being compared: panel (a) uses raw observations, whereas panel (b) uses the seasonal residual. In the illustrated example, the

mean contrast is 0.663 for raw patches and 0.613 for seasonal patches. The corresponding percentages above the identity line are 90.6% and 95.9%. Mean contrast describes the average difference between periodic and non-periodic similarities, whereas the percentage above the identity line describes how often this difference is positive. Despite these numerical differences, both panels exhibit a similar concentration above the identity line.

E.2.4 Panel (c): Local Trend Alignment

Panel (c) compares raw and EMA trend patches at matching positions. Unlike panels (a,b), which compare a query with historical candidates, panel (c) compares the raw and trend versions of the same local segment. All patch positions in valid standardized windows are included, without an EMA warm-up exclusion.

For a patch $\mathbf{p} = (p_0, \ldots, p_{P-1})$, we estimate its local slope by least-squares linear fitting:

$$b(\mathbf{p}) = \frac{\sum_{j=0}^{P-1}(j-\bar{j})(p_j-\bar{p})}{\sum_{j=0}^{P-1}(j-\bar{j})^2}, \qquad \bar{j} = \frac{P-1}{2}, \tag{93}$$

where $\bar{p}$ is the mean patch value. Raw and trend patches retain the same window-level normalization and are not independently rescaled. The horizontal axis shows the trend-patch slope b_i^{trend}, and the vertical axis shows the raw-patch slope b_i^{raw}.

We report three complementary statistics. First, Pearson correlation measures the linear association between the paired slopes:

$$r = \frac{\sum_{i=1}^{N}(b_i^{\text{raw}} - \bar{b}^{\text{raw}})(b_i^{\text{trend}} - \bar{b}^{\text{trend}})}{\sqrt{\sum_{i=1}^{N}(b_i^{\text{raw}} - \bar{b}^{\text{raw}})^2}\sqrt{\sum_{i=1}^{N}(b_i^{\text{trend}} - \bar{b}^{\text{trend}})^2}}, \tag{94}$$

where N is the number of paired patches.

Second, the concordance correlation coefficient (CCC) measures agreement while accounting for differences in the means and variances:

$$\text{CCC} = \frac{2\ \text{Cov}(b^{\text{raw}}, b^{\text{trend}})}{\text{Var}(b^{\text{raw}}) + \text{Var}(b^{\text{trend}}) + (\bar{b}^{\text{raw}} - \bar{b}^{\text{trend}})^2}. \tag{95}$$

The empirical variances and covariance are computed with normalization factor $1/N$.

Third, direction agreement measures whether the paired patches share the same rising, falling, or flat tendency. We assign a direction according to the fitted end-to-end change:

$$D(b) = \begin{cases} +1, & (P-1)b > \tau, \\ -1, & (P-1)b < -\tau, \\ 0, & \text{otherwise}, \end{cases} \qquad \tau = 0.02. \tag{96}$$

The direction agreement fraction is

$$A_{\text{dir}} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\left[D(b_i^{\text{raw}}) = D(b_i^{\text{trend}})\right], \tag{97}$$

which is displayed as a percentage.

In the illustrated example, Pearson correlation is 0.907, CCC is 0.881, and direction agreement is 89.4%. These results show strong agreement between the local slopes of raw and EMA trend patches.

E.2.5 Visualization and Interpretation

All panels use hexagonal binning. For panel k, the color value of bin h is

$$q_{k,h} = 100\,\frac{n_{k,h}}{N_k}, \tag{98}$$

Table 7: Effect of patch granularity: MSE, MAE, and RMSE across all prediction horizons for patch lengths of 12, 16, 24, and 32. PaDNet adopts a patch length of 16 by default.

Datasets		Patch Length 12			Patch Length 16			Patch Length 24			Patch Length 32		
		MSE	MAE	RMSE	MSE	MAE	RMSE	MSE	MAE	RMSE	MSE	MAE	RMSE
ETTh1	96	0.3705	0.3939	0.6087	0.3707	0.3948	0.6088	0.3717	0.3953	0.6096	0.3730	0.3972	0.6108
	192	0.4220	0.4239	0.6496	0.4226	0.4236	0.6501	0.4236	0.4246	0.6508	0.4237	0.4274	0.6510
	336	0.4594	0.4432	0.6778	0.4621	0.4441	0.6798	0.4700	0.4486	0.6856	0.4749	0.4501	0.6891
	720	0.4744	0.4675	0.6888	0.4703	0.4646	0.6858	0.4904	0.4769	0.7003	0.4748	0.4702	0.6890
ETTm1	96	0.3197	0.3578	0.5654	0.3163	0.3561	0.5624	0.3211	0.3597	0.5667	0.3146	0.3557	0.5609
	192	0.3594	0.3793	0.5995	0.3537	0.3791	0.5947	0.3555	0.3793	0.5963	0.3571	0.3803	0.5975
	336	0.3851	0.4013	0.6205	0.3833	0.4006	0.6191	0.3853	0.4007	0.6207	0.3891	0.4022	0.6238
	720	0.4561	0.4395	0.6753	0.4480	0.4389	0.6693	0.4502	0.4421	0.6709	0.4561	0.4442	0.6753
ETTm2	96	0.1737	0.2570	0.4168	0.1737	0.2571	0.4167	0.1749	0.2576	0.4182	0.1757	0.2583	0.4191
	192	0.2371	0.2987	0.4869	0.2394	0.3005	0.4893	0.2399	0.3004	0.4898	0.2413	0.3045	0.4913
	336	0.2997	0.3410	0.5475	0.2988	0.3386	0.5467	0.3030	0.3430	0.5504	0.3033	0.3429	0.5507
	720	0.3948	0.3953	0.6283	0.3953	0.3952	0.6287	0.4004	0.3987	0.6328	0.3994	0.3976	0.6320
Solar	96	0.1859	0.2402	0.4311	0.1838	0.2348	0.4288	0.1795	0.2347	0.4237	0.1753	0.2342	0.4186
	192	0.1950	0.2524	0.4416	0.1917	0.2502	0.4378	0.1940	0.2501	0.4404	0.1938	0.2496	0.4402
	336	0.1965	0.2539	0.4433	0.1991	0.2570	0.4462	0.1996	0.2577	0.4467	0.2013	0.2586	0.4486
	720	0.2044	0.2620	0.4521	0.2032	0.2586	0.4508	0.2025	0.2595	0.4500	0.2044	0.2611	0.4521
Electricity	96	0.1309	0.2268	0.3618	0.1302	0.2263	0.3609	0.1307	0.2274	0.3615	0.1302	0.2262	0.3608
	192	0.1479	0.2420	0.3846	0.1485	0.2434	0.3854	0.1482	0.2429	0.3849	0.1475	0.2419	0.3841
	336	0.1660	0.2610	0.4075	0.1656	0.2605	0.4070	0.1672	0.2617	0.4090	0.1664	0.2608	0.4079
	720	0.2003	0.2917	0.4475	0.2002	0.2916	0.4474	0.1995	0.2918	0.4466	0.2003	0.2912	0.4475
Exchange	96	0.0796	0.1979	0.2822	0.0787	0.1952	0.2806	0.0796	0.1960	0.2822	0.0796	0.1968	0.2821
	192	0.1720	0.2947	0.4148	0.1729	0.2949	0.4158	0.1743	0.2968	0.4175	0.1795	0.3010	0.4237
	336	0.3462	0.4245	0.5884	0.3191	0.4074	0.5649	0.3412	0.4225	0.5841	0.3502	0.4280	0.5918
	720	0.8907	0.7083	0.9438	0.8394	0.6880	0.9162	0.8416	0.6906	0.9174	0.9297	0.7266	0.9642
Pems03	12	0.0589	0.1591	0.2427	0.0578	0.1576	0.2403	0.0574	0.1563	0.2396	0.0576	0.1568	0.2401
	24	0.0726	0.1747	0.2695	0.0718	0.1736	0.2680	0.0720	0.1740	0.2684	0.0714	0.1728	0.2671
	48	0.0960	0.1989	0.3099	0.0952	0.1975	0.3086	0.0946	0.1975	0.3076	0.0954	0.1975	0.3089
	96	0.1203	0.2204	0.3468	0.1217	0.2218	0.3488	0.1179	0.2194	0.3434	0.1176	0.2184	0.3429
Pems04	12	0.0714	0.1726	0.2671	0.0700	0.1708	0.2645	0.0708	0.1724	0.2662	0.0716	0.1739	0.2675
	24	0.0844	0.1893	0.2905	0.0826	0.1845	0.2874	0.0823	0.1844	0.2869	0.0826	0.1852	0.2873
	48	0.1047	0.2103	0.3236	0.1009	0.2049	0.3177	0.1028	0.2061	0.3206	0.1024	0.2058	0.3200
	96	0.1333	0.2291	0.3650	0.1220	0.2227	0.3493	0.1208	0.2198	0.3475	0.1263	0.2275	0.3554
Pems07	12	0.0511	0.1448	0.2261	0.0511	0.1449	0.2260	0.0512	0.1447	0.2262	0.0505	0.1429	0.2247
	24	0.0617	0.1592	0.2485	0.0620	0.1592	0.2491	0.0609	0.1579	0.2468	0.0617	0.1596	0.2484
	48	0.0772	0.1780	0.2778	0.0785	0.1792	0.2802	0.0760	0.1766	0.2757	0.0729	0.1721	0.2701
	96	0.0955	0.1983	0.3090	0.0918	0.1928	0.3031	0.0882	0.1881	0.2970	0.0876	0.1875	0.2961
Pems08	12	0.0685	0.1684	0.2618	0.0689	0.1685	0.2624	0.0677	0.1674	0.2601	0.0658	0.1632	0.2564
	24	0.0897	0.1873	0.2996	0.0861	0.1795	0.2934	0.0878	0.1849	0.2963	0.0850	0.1793	0.2915
	48	0.1235	0.2095	0.3514	0.1156	0.2019	0.3400	0.1193	0.1999	0.3455	0.1226	0.2027	0.3501
	96	0.1757	0.2269	0.4192	0.1704	0.2241	0.4128	0.1828	0.2317	0.4276	0.1657	0.2198	0.4070

where $n_{k,h}$ is the number of displayed samples in the bin and N_k is the total number of displayed samples in that panel. The panels share a logarithmic color scale. The sampling unit is a window in panels (a,b) and a matched window–patch pair in panel (c).

The identity line serves different purposes in the two comparisons. In panels (a,b), samples above the line have higher periodic than non-periodic similarity. In panel (c), samples near the line have similar raw-patch and trend-patch slopes.

Taken together, the illustrated results show high similarity between raw and seasonal patches in their periodic contrast patterns, together with strong agreement between raw and EMA trend patches in their local slopes.

E.3 Effect of Patch Granularity

We examine the sensitivity of PaDNet to patch length using $P \in \{12, 16, 24, 32\}$. Table 7 reports MSE, MAE, and RMSE at each prediction horizon. The preferred patch length varies across datasets and horizons. Shorter patches generally yield lower errors on ETTh1 and ETTm2, whereas longer patches improve performance in several PEMS03 and PEMS07 settings. Electricity exhibits relatively small variations across the evaluated lengths, while Exchange shows greater sensitivity at longer prediction horizons. The default length of

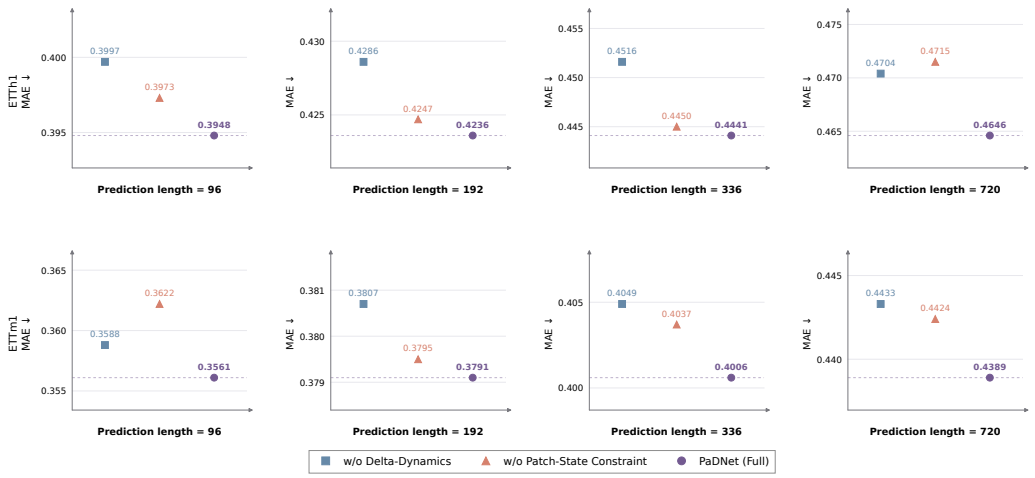


Figure 6: Component ablation on ETTh1 and ETTm1 across prediction horizons. Dashed lines indicate PaDNet's MAE. Lower is better.

16 provides competitive performance across diverse settings, including the lowest errors on ETTm1 at horizons 192, 336, and 720. These results indicate that patch granularity affects forecasting performance, with no single length being uniformly optimal.

E.4 Additional Component Ablation

To further assess the contributions of Delta-Dynamics and the Patch-State constraint, we conduct component ablations on ETTh1 and ETTm1 across individual prediction horizons. While the averaged results in Table 2 summarize the overall effects of these components, horizon-wise comparisons provide a more detailed view of how their contributions vary with the forecasting length. In particular, we examine whether the benefits of explicit transition modeling and constrained trajectory encoding are consistent across horizons or more pronounced in particular forecasting settings. We compare the complete PaDNet with w/o Delta-Dynamics (w/o DD), which removes the delta-dynamics branch, and w/o Patch-State Constraint (w/o PSC), which replaces the constrained patch-state projection with a standard linear projection. These comparisons allow us to examine the performance changes associated with removing each component.

Figure 6 presents the MAE results for prediction horizons $\{96, 192, 336, 720\}$. The upper and lower rows correspond to ETTh1 and ETTm1, respectively. Each panel compares the three variants at a fixed horizon, with the dashed line indicating PaDNet's MAE.

PaDNet achieves the lowest MAE in all evaluated settings. Removing Delta-Dynamics increases MAE across both datasets, supporting the benefit of explicitly modeling successive changes alongside local trajectory states. Removing the Patch-State constraint also degrades performance, indicating the value of regularizing trajectory encoding. The relative contributions vary with the dataset and prediction horizon, but retaining both components consistently provides better results in these comparisons.

E.5 Discussion of Decomposition Paradigms

To examine the role of decomposition in patch-based forecasting, we investigate whether explicitly separating temporal components, particularly trend and residual information, provides predictive benefits beyond those afforded by raw patch representations. This analysis addresses a central question underlying our formulation: when local trajectories already preserve trend-related structure, does explicit trend decomposition remain beneficial, and how does its effectiveness compare with alternative ways of organizing temporal information?

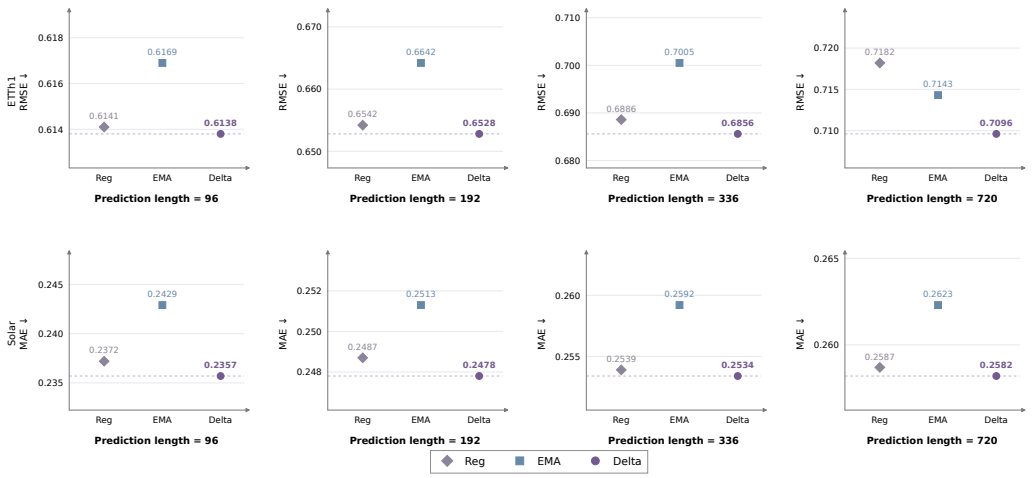


Figure 7: Comparison of Reg, EMA, and Delta in XPatch across prediction horizons: RMSE on ETTh1 and MAE on Solar.

To address these questions, we compare three formulations: direct modeling of raw sequences, explicit trend decomposition, and the proposed state–dynamics representation. The comparison between raw-input and decomposition-based variants evaluates the contribution of component separation, while the comparison with state–dynamics modeling examines the benefits of explicitly representing successive changes alongside trajectory states. We consider multiple trend decomposition schemes and conduct the comparisons within both PaDNet and XPatch, enabling us to assess how forecasting performance varies with the decomposition strategy and whether the observed patterns are consistent across different architectures.

Effect of Decomposition. We compare the proposed formulation with a baseline without decomposition (Reg) and explicit trend decomposition variants. Reg supplies the raw sequence to both the patch-based and full-sequence branches. Moving average decomposition (MA) (Wu et al., 2021), exponential moving average decomposition (EMA), and double exponential moving average decomposition (DEMA) (Stitsyuk & Choi, 2025) instead supply the residual and smoothed components to the respective branches. Our proposed formulation, denoted as Delta in the figures and tables, retains the original sequence for Patch-State modeling and explicitly represents successive changes through Delta-Dynamics, corresponding to the complete PaDNet model in this comparison.

Table 3 summarizes the results averaged over four prediction horizons. PaDNet achieves lower average MSE, MAE, and RMSE than the explicit decomposition variants across the evaluated datasets. Reg also generally outperforms these variants, indicating that separating a smoothed trend component does not consistently improve forecasting when patch representations are already available. Figure 8 further presents the results on ETTh1, ETTm1 and Pems04 across prediction horizons. Compared with Reg, PaDNet exhibits favorable performance at longer horizons, particularly in MAE, while differences at shorter horizons are generally modest. These improvements vary across horizons and metrics, suggesting that the benefit of explicit transition modeling depends on the forecasting setting.

Applicability Across Architectures. We further examine these formulations in XPatch (Stitsyuk & Choi, 2025), a representative architecture combining patch modeling with explicit trend decomposition. We compare its original exponential moving average decomposition (EMA), a baseline without decomposition (Reg), and the proposed Patch-State and Delta-Dynamics formulation (Delta). EMA processes residual and smoothed components in separate branches, Reg supplies the raw sequence to both branches, and Delta combines the original sequence with explicit increments for state–dynamics modeling.

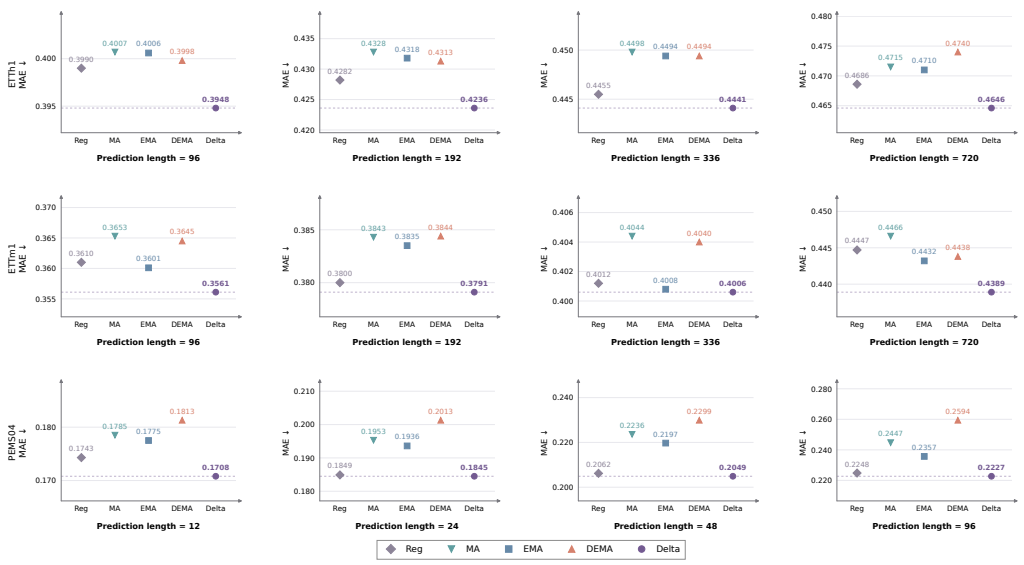


Figure 8: MAE comparison of Reg, MA, EMA, DEMA, and Delta within PaDNet across prediction horizons on ETTh1, ETTm1, and PEMS04.

Table 8 reports forecasting errors averaged over four prediction horizons on ETTh1, Solar, Electricity, Exchange, and PEMS03. Reg obtains lower average MSE and MAE than EMA across all evaluated datasets. Delta further reduces both metrics relative to Reg, although the magnitude of improvement varies by dataset. Figure 7 provides comparisons at individual prediction horizons, reporting RMSE on ETTh1 and MAE on Solar. Delta achieves the lowest error at each evaluated horizon in both comparisons.

Across the two architectures, the results indicate limited benefits from explicit trend separation relative to direct sequence modeling in the evaluated settings. This observation is consistent with patch representations retaining trend-related trajectory information. The additional gains from Delta suggest that explicitly representing successive changes complements trajectory-state modeling, providing empirical support for the state–dynamics formulation beyond a single architecture.

E.6 Optimization Dynamics

We compare the validation loss curves of PaDNet and w/o Delta-Dynamics to examine the contribution of explicit dynamics modeling during training, particularly as the prediction horizon increases. Experiments are conducted on ETTm1 with prediction horizons $\{192, 336\}$ and PEMS04 with $\{12, 24\}$. Both variants are trained for the same fixed number of epochs under otherwise identical training settings, with early stopping disabled to observe the full validation trajectories.

As shown in Figure 9, both variants exhibit decreasing validation loss as training progresses. For longer prediction horizons, PaDNet shows a more pronounced reduction in validation loss and a clearer advantage over the variant without Delta-Dynamics. This observation suggests

Table 8: Forecasting results of XPatch with Reg, EMA, and Delta, averaged over four prediction horizons. Lower values are better.

	ETTh1			Solar			Electricity			Exchange			PEMS03		
	MSE	MAE	RMSE	MSE	MAE	RMSE	MSE	MAE	RMSE	MSE	MAE	RMSE	MSE	MAE	RMSE
Reg	0.4488	0.4435	0.6688	0.1949	0.2496	0.4413	0.1616	0.2558	0.4008	0.3731	0.4069	0.5592	0.0875	0.1889	0.2931
EMA	0.4557	0.4463	0.6740	0.1982	0.2539	0.4452	0.1617	0.2559	0.4008	0.4034	0.4194	0.5775	0.0909	0.1932	0.2984
Delta	0.4441	0.4397	0.6655	0.1948	0.2488	0.4413	0.1615	0.2556	0.4006	0.3729	0.4061	0.5586	0.0870	0.1887	0.2924

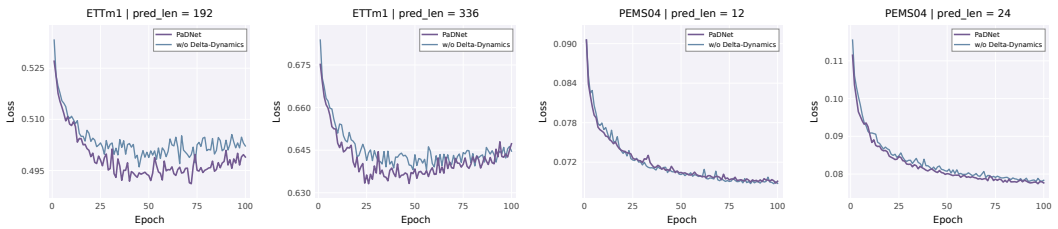
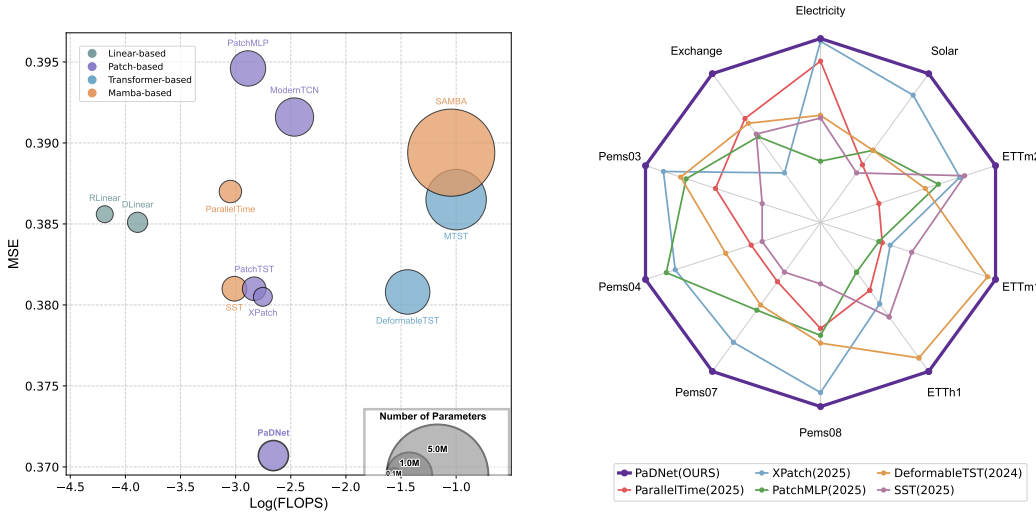


Figure 9: Validation-loss curves of PaDNet and w/o Delta-Dynamics on ETTm1 and PEMS04 without early stopping.

that explicitly modeling successive changes can help the model learn temporal dependencies relevant to forecasting over longer horizons. The lower validation loss, together with the forecasting results, supports the contribution of Delta-Dynamics to predictive performance in these settings.

E.7 Accuracy–Efficiency Trade-off

We further evaluate PaDNet in terms of computational cost, model size, and forecasting accuracy to assess whether its predictive gains are accompanied by substantial resource requirements. Specifically, we compare forecasting MSE, FLOPs, and parameter counts under a common forecasting setting, and examine average MAE across multiple datasets to assess the consistency of its performance under different temporal characteristics. In addition to the main baselines, the computational comparison includes representative architectures from different forecasting paradigms: the multi-scale patch model MTST (Zhang et al., 2024), ModernTCN (Luo & Wang, 2024b), which combines patching with convolutions, and the state-space model SAMBA (Weng et al., 2025). Together, these comparisons provide complementary views of PaDNet's accuracy–efficiency trade-off and its relative forecasting performance across datasets.



(a) Accuracy–efficiency comparison.

(b) Performance across datasets.

Figure 10: Accuracy, efficiency, and cross-dataset performance comparison. (a) Results on ETTh1 with look-back and prediction lengths of 96. The x-axis shows $\log_{10}(\text{FLOPs}/10^9)$, the y-axis reports MSE, and bubble size represents the parameter count. (b) Radar comparison based on average MAE across prediction horizons. Each spoke represents a dataset, with scores normalized to $[0, 1]$ across the compared models; larger values indicate lower MAE.

Figure 10a reports the accuracy–efficiency comparison on ETTh1 with both the look-back and prediction lengths set to 96. The horizontal axis shows $\log_{10}(\text{FLOPs}/10^9)$, corresponding to the base-10 logarithm of the computational cost measured in GFLOPs. The vertical axis reports forecasting MSE, while bubble size represents the number of parameters. Models closer to the lower-left region therefore achieve lower prediction error with less computation, and smaller bubbles indicate fewer parameters. PaDNet obtains the lowest MSE among the compared methods while maintaining a relatively small parameter count and moderate FLOPs. Some alternatives require less computation but produce higher forecasting errors, whereas greater computational cost does not consistently result in improved accuracy.

Figure 10b complements this single-setting analysis by comparing forecasting performance across datasets. Each spoke corresponds to a dataset. For each dataset d, we convert the average MAE across prediction horizons into a relative score:

$$s_{m,d} = \frac{e_d^{\max} - e_{m,d}}{e_d^{\max} - e_d^{\min}}, \tag{99}$$

where $e_{m,d}$ denotes the average MAE of model m, and $e_d^{\max}$ and $e_d^{\min}$ are the maximum and minimum values among the compared models on dataset d. The resulting score ranges from 0 to 1: a score of 1 corresponds to the lowest average MAE, and a score of 0 corresponds to the highest. Positions farther from the center therefore indicate better relative forecasting performance. Since normalization is performed separately for each dataset, the radial values summarize relative performance within each dataset rather than absolute error differences across datasets. PaDNet exhibits a consistently competitive profile across datasets with heterogeneous temporal characteristics, supporting its overall forecasting performance across diverse settings.