CHANNEL-DEPENDENT STATE SPACE MODEL FOR MULTIVARIATE TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multivariate time series forecasting (MTSF) is critical across many real-world domains. Existing deep learning approaches fall into two paradigms with distinct limitations: channel-independent (CI) methods unconditionally ignore cross-variable dependencies and model only temporal dynamics, while channel-dependent (CD) methods consider both but typically rely on architectural compromises to mitigate overfitting and computational overhead. We therefore propose Chameleon, a specialized CD state space model (SSM) that enables data-dependent, fine-grained interactions across variables while scaling linearly with their number. By connecting selective SSMs with the Kalman filter, we leverage the missing measurement update in the former for cross-variable modeling while preserving the SSM backbone for robust temporal modeling. We further identify favorable inductive biases of GatedDeltaNet for time series, adapt it as our backbone, and improve generalization through additional techniques, including a previously unexplored stochastic perturbation of reversible instance normalization. On strongly dependent ODE and PEMS datasets, Chameleon achieves the best MSE and MAE across all settings, while its CI ablation and prior CD methods incur 61–178% higher MSE on average. Across 28 standard benchmark settings, Chameleon also achieves better MSE and MAE than each baseline in at least 27 and 22 cases, respectively. Training-time and peak-memory analyses on Traffic and ETT further demonstrate competitive efficiency and favorable memory scalability across different variable counts.

1 INTRODUCTION

Multivariate time series forecasting (MTSF) refers to predicting the future values of multiple continuous and potentially correlated variables based on their histories. Accurate forecasts are valuable across applications including energy demand prediction, traffic flow estimation, weather forecasting, financial asset prediction, and product quality estimation in manufacturing (Suganthi & Samuel, 2012; Hernandez et al., 2014; Jiang & Luo, 2022; Lana et al., 2018; Bi et al., 2023; Price et al., 2025; Elliott & Timmermann, 2008; Sezer et al., 2020; Ren et al., 2023). With the growth of data volume and complexity, deep neural networks (DNNs) have become widely adopted for MTSF. However, designing a generalizable architecture across diverse multivariate systems remains a long-standing challenge because cross-variable dependencies vary widely in strength and structure. Moreover, time series data are generally noisier, less regular, and less abundant than text or images, while both the temporal and variable dimensions can span hundreds of elements, making unconstrained joint modeling prone to overfitting, computationally expensive, and difficult to scale.

One simplified paradigm, channel-independent (CI) modeling, has therefore recently emerged and proven unexpectedly effective on benchmark datasets. PatchTST (Nie et al., 2023) pioneered this line by forecasting each variable, also referred to as a channel or series, independently through a shared Transformer (Vaswani et al., 2017) that treats temporal patches as tokens while avoiding interactions across variables, achieving state-of-the-art (SOTA) performance at its release. Subsequent approaches, such as FITS (Xu et al., 2024) and SFNN (Sun et al., 2025), reinforced this trend by matching or even surpassing prior, more complex channel-dependent (CD) architectures through one or a few feedforward layers. Together, these results highlight the importance of effective intra-series modeling and suggest that many current benchmarks exhibit weak cross-variable dependencies.

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.

Nevertheless, information from other variables can be crucial in some settings, motivating renewed interest in CD models. Inspired by CI, a prominent family of CD methods embeds each variable independently over time before modeling dependencies across variables. In addition, to mitigate the overfitting risk, architectural compromises are commonly adopted, often at the cost of modeling granularity or capacity for temporal dynamics. iTransformer (Liu et al., 2024) popularized these strategies by representing each variable's entire input window as a single token using only a linear layer, while reserving the more expressive Transformer for cross-variable interactions. This formulation is retained by approaches such as SOFTS (Han et al., 2024) and S-Mamba (Wang et al., 2025), which replace the Transformer with more scalable designs. Other variants, including Crossformer (Zhang & Yan, 2023), Leddam (Yu et al., 2024), and TimeMachine (Ahamed & Cheng, 2024), follow the same factorization but employ either self-attention or Mamba along both temporal and variable dimensions. However, this added capacity incurs higher computational cost without consistently translating into better performance, especially on benchmarks, where simple CI SFNN (Sun et al., 2025) and later, more constrained CD methods such as DUET (Qiu et al., 2025) and TimeFilter (Hu et al., 2025) remain empirically superior. To improve generalization, DUET and TimeFilter instead restrict interactions to selected variable or patch pairs, yet still inherit similar compromises or introduce greater scalability concerns by constructing pairwise dependencies across patches from all variables. These developments therefore leave an unresolved trade-off among modeling granularity, temporal capacity, generalization, efficiency, and scalability.

These observations raise two natural questions:

1. Can we incorporate fine-grained cross-variable interactions into a strong temporal modeling backbone while remaining efficient and scalable?
2. If so, can we avoid overfitting and achieve SOTA performance in multivariate systems with both weak and strong dependencies across variables?

These questions become even more compelling in light of recent advances in selective state space models (SSMs), pioneered by Mamba (Gu & Dao, 2024). Mamba retains the recency-emphasizing inductive bias of recurrent neural networks while overcoming their long-range memory limitations. Among its successors, GatedDeltaNet (Yang et al., 2025) further introduces delta-based memory updates (Schmidhuber, 1992), enabling both rapid forgetting and targeted memory rewrites, which are particularly desirable properties for noisy time series. Despite these advantages, GatedDeltaNet has not yet been explored for MTSF, while existing Mamba-based attempts largely inherit established architectures by replacing their original, often attention-based, backbones with Mamba (Ahamed & Cheng, 2024; Wang et al., 2025), leaving specialized cross-variable interaction mechanisms underexplored.

This work therefore presents **Chameleon**, an MTSF model built on our proposed CD SSM blocks to address these gaps. Our contributions are:

1. We draw a parallel between selective SSMs and the Kalman filter (Kalman, 1960) for MTSF, exploiting this connection to design an explicit cross-variable measurement update while maintaining linear complexity in the number of variables.
2. We identify favorable inductive biases of GatedDeltaNet for time series, tailor it into an MTSF-specialized backbone, and further introduce a previously unexplored stochastic perturbation of reversible instance normalization (Kim et al., 2022) during training to improve generalization.
3. Chameleon achieves SOTA performance across standard long-horizon benchmarks and strongly dependent systems.

2 PRELIMINARIES

Problem Formulation. Given a historical window $\mathbf{X} \in \mathbb{R}^{N \times T}$ of N variables observed at T regularly spaced time steps, MTSF aims to forecast the next F steps $\mathbf{Y} \in \mathbb{R}^{N \times F}$. Exact timestamp information is not explicitly used, and stationarity is not assumed, as trends and seasonality are prevalent in real-world time series.

Selective SSMs and GatedDeltaNet. Selective SSMs (Gu & Dao, 2024; Dao & Gu, 2024) have emerged as strong sequence learners through efficient long-context modeling and data-dependent state updates. GatedDeltaNet (Yang et al., 2025) further combines the global decay of Mamba2 (Dao & Gu, 2024) for rapid forgetting with the delta rule of DeltaNet (Yang et al., 2024) for targeted memory rewrites:

$$\mathbf{S}_t = \alpha_t \mathbf{S}_{t-1}(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top) + \beta_t \mathbf{v}_t \mathbf{k}_t^\top \in \mathbb{R}^{D_v \times D_k}, \tag{1}$$

where $\alpha_t, \beta_t \in (0, 1)$ control global decay and targeted updating, respectively, and $\mathbf{v}_t \in \mathbb{R}^{D_v}$ and $\mathbf{k}_t \in \mathbb{R}^{D_k}$ are input-dependent projections. This adaptive memory management provides a favorable inductive bias for filtering irrelevant information in noisy time series.

The Kalman Filter. The Kalman filter (Kalman, 1960) estimates dynamic system states through a recursive two-step cycle. The "time update" produces the prior state estimate $\mathbf{S}_t^- \in \mathbb{R}^{D_s}$ and error covariance $\mathbf{P}_t^- \in \mathbb{R}^{D_s \times D_s}$:

$$\mathbf{S}_t^- = \mathbf{A}_t \mathbf{S}_{t-1} + \mathbf{B}_t \mathbf{u}_t, \quad \mathbf{P}_t^- = \mathbf{A}_t \mathbf{P}_{t-1} \mathbf{A}_t^\top + \mathbf{Q}_t, \tag{2}$$

where $\mathbf{A}_t \in \mathbb{R}^{D_s \times D_s}$ is the state transition matrix, $\mathbf{u}_t \in \mathbb{R}^{D_u}$ is the control input, $\mathbf{B}_t \in \mathbb{R}^{D_s \times D_u}$ is the control matrix, and $\mathbf{Q}_t \in \mathbb{R}^{D_s \times D_s}$ is the process noise covariance. The "measurement update" corrects this predicted prior state using new observations $\mathbf{z}_t \in \mathbb{R}^{D_z}$:

$$\mathbf{K}_t = \mathbf{P}_t^- \mathbf{H}_t^\top (\mathbf{H}_t \mathbf{P}_t^- \mathbf{H}_t^\top + \mathbf{R}_t)^{-1}, \quad \mathbf{S}_t = \mathbf{S}_t^- + \mathbf{K}_t(\mathbf{z}_t - \mathbf{H}_t \mathbf{S}_t^-), \quad \mathbf{P}_t = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t)\mathbf{P}_t^-, \tag{3}$$

where $\mathbf{H}_t \in \mathbb{R}^{D_z \times D_s}$ maps state to measurement space, $\mathbf{R}_t \in \mathbb{R}^{D_z \times D_z}$ is the measurement noise covariance, and $\mathbf{K}_t \in \mathbb{R}^{D_s \times D_z}$ is the Kalman gain that weights the innovation by the relative confidence between prediction and measurement. The recurrent state update of the aforementioned SSMs naturally parallels the time update, with $\mathbf{v}_t$ serving as the control input. However, explicit measurement updates for SSMs remain underexplored as domain-specific modeling mechanisms. This work leverages this connection to introduce a learnable measurement update for cross-variable modeling in MTSF.

3 METHOD

The proposed framework processes the multivariate input through the following stages: optional reversible instance normalization (RevIN) (Kim et al., 2022) and decomposition, patching and embedding, an encoder built from our proposed Chameleon blocks, and an MLP-based decoder. Before entering the encoder, each variable is independently transformed into patch-level representations, which are then processed by the encoder and mapped to forecasts by the decoder.

3.1 INPUT PROCESSING

The input $\mathbf{X} \in \mathbb{R}^{N \times T}$ optionally undergoes RevIN and decomposition to mitigate distribution shifts and preserve prominent trend structure, respectively. The criteria for their use and supporting ablations are provided in Appendix B.

Reversible Instance Normalization. When enabled, the mean μ_i and standard deviation σ_i for each variable i are computed over a predefined normalization window and used to normalize the input as $\tilde{x}_{i,t} = (x_{i,t} - \mu_i)/\sigma_i$. When disabled, we instead set $\mu_i = 0$ and $\sigma_i = 1$ for all instances and variables. These statistics are stored for denormalizing the final predictions. To further improve generalization under distribution shifts, we introduce a simple yet previously unexplored data augmentation that perturbs μ_i and σ_i during training:

$$\tilde{\mu}_i = \mu_i + \epsilon_{\mu,i}, \quad \tilde{\sigma}_i = \sigma_i \cdot \exp(\epsilon_{\sigma,i}), \quad \epsilon_{\mu,i} \sim \mathcal{N}(0, \delta_\mu^2), \quad \epsilon_{\sigma,i} \sim \mathcal{N}(0, \delta_\sigma^2), \tag{4}$$

where δ_μ and δ_σ control the perturbation magnitude. The multiplicative parameterization of $\tilde{\sigma}_i$ ensures positivity without additional constraints. During training, μ_i and σ_i in both normalization and denormalization are replaced by $\tilde{\mu}_i$ and $\tilde{\sigma}_i$, respectively.

Decomposition, Patching, and Embedding. Following Autoformer (Wu et al., 2021), when decomposition is applied, each series is separated into a trend component $\tilde{x}_{i,t}^{\text{trend}}$ via a length-preserving moving average kernel and a seasonal component $\tilde{x}_{i,t}^{\text{seasonal}} = \tilde{x}_{i,t} - \tilde{x}_{i,t}^{\text{trend}}$. Each component is then

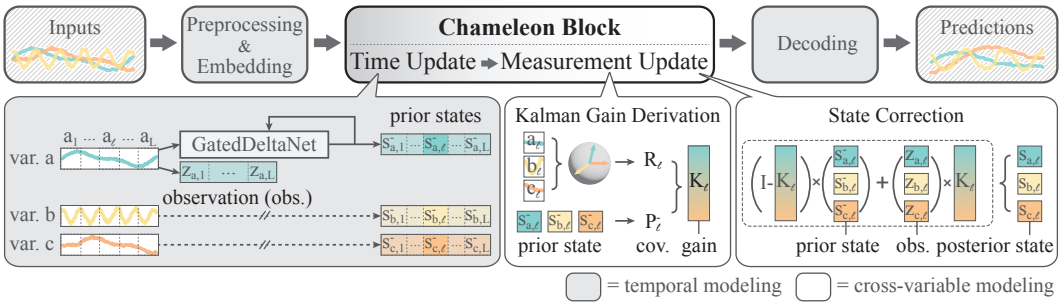


Figure 1: Overview of Chameleon, which preserves the recurrent GatedDeltaNet backbone for temporal modeling and introduces a Kalman-filter-inspired measurement update for cross-variable modeling through data-dependent observation signals and covariances (cov.).

divided into $L = \lceil T/T_p \rceil$ non-overlapping patches of length T_p following PatchTST (Nie et al., 2023), producing $\mathbf{X}_i^{\text{trend}}, \mathbf{X}_i^{\text{seasonal}} \in \mathbb{R}^{L \times T_p}$. Replication padding is used to preserve length during decomposition and complete the final patch when needed. Rather than processing the two components through separate forecasting networks and combining their outputs only at the final stage, we embed them through separate linear projections and immediately sum the resulting representations, keeping the additional cost minimal:

$$\mathbf{x}_{i,l}^{\text{embed}} = \mathbf{W}^{(\text{s})}\mathbf{x}_{i,l}^{\text{seasonal}} + \mathbf{W}^{(\text{t})}\mathbf{x}_{i,l}^{\text{trend}} + \mathbf{e}_l \in \mathbb{R}^{D_v}, \tag{5}$$

where $\mathbf{W}^{(\text{s})}, \mathbf{W}^{(\text{t})} \in \mathbb{R}^{D_v \times T_p}$ are shared across patches and variables, and $\mathbf{e}_l$ is the fixed sinusoidal positional encoding (Vaswani et al., 2017). When decomposition is disabled, $\tilde{x}_{i,t}^{\text{seasonal}} = \tilde{x}_{i,t}$, and the embedding reduces accordingly to $\mathbf{x}_{i,l}^{\text{embed}} = \mathbf{W}^{(\text{s})}\mathbf{x}_{i,l}^{\text{seasonal}} + \mathbf{e}_l$.

3.2 Chameleon Blocks and Encoder

The encoder is composed of a stack of repeated units. Each unit consists of root mean square layer normalization (RMSNorm) (Zhang & Sennrich, 2019) followed by our proposed Chameleon block and an optional feedforward network block. As illustrated in Figure 1, the Chameleon block unifies the sequential modeling of GatedDeltaNet (Yang et al., 2025) with a Kalman-filter-inspired measurement update for cross-variable state calibration. In the classical multivariate Kalman filter, both the time and measurement updates operate on the joint state of all variables at each time step. However, such joint modeling has been found suboptimal for Mamba in MTSF (Weng et al., 2025). Inspired by CI and recent factorized CD architectures, we accordingly assign temporal and variable modeling to the time and measurement updates, respectively. The "time update" first processes each patch token as a control input to drive intra-series evolution recurrently, after which the "measurement update" derives an observation signal from the same token to correct hidden states across variables.

Time Update. The GatedDeltaNet block is leveraged as the temporal modeling backbone due to its advantageous inductive biases. For variable i, given the RMSNorm-processed l-th patch token $\bar{\mathbf{x}}_{i,l} \in \mathbb{R}^{D_v}$, the SSM computes a prior state via Equation 1:

$$\mathbf{S}_{i,l}^{-} = \alpha_{i,l}\mathbf{S}_{i,l-1}(\mathbf{I} - \beta_{i,l}\mathbf{k}_{i,l}\mathbf{k}_{i,l}^{\top}) + \beta_{i,l}\mathbf{v}_{i,l}\mathbf{k}_{i,l}^{\top} \in \mathbb{R}^{D_v \times D_k}, \tag{6}$$

where $\alpha_{i,l}$, $\beta_{i,l}$, $\mathbf{k}_{i,l}$, and $\mathbf{v}_{i,l}$ are all projected from $\bar{\mathbf{x}}_{i,l}$. Unlike the original GatedDeltaNet, the convolutional layers for deriving $\mathbf{k}_{i,l}$ and $\mathbf{v}_{i,l}$ are omitted for efficiency, as the patch-based embedding already captures local details. Structurally, $\alpha_{i,l}(\mathbf{I} - \beta_{i,l}\mathbf{k}_{i,l}\mathbf{k}_{i,l}^{\top}) \in \mathbb{R}^{D_k \times D_k}$, $\beta_{i,l}\mathbf{k}_{i,l}^{\top}$, and $\mathbf{v}_{i,l}$ are respectively analogous to the state transition matrix, control matrix, and control input in the Kalman-filter time update of Equation 2, but the update here is performed independently within each variable's state without computation across variables.

Measurement Update. To capture dependencies across variables, we instantiate the measurement update as a learnable state calibration mechanism operating only along the variable dimension. The same patch tokens $\bar{\mathbf{x}}_{i,l}$ are reused as sources of observation signals for this step. Under our factorized

design, the two updates extract distinct representations for their respective modeling dimensions, avoiding suboptimal joint modeling and potential redundancies. The time update derives control inputs from the tokens to drive intra-series state evolution, whereas the measurement update performs cross-variable state correction based on the resulting observations. Unlike the original Kalman filter, where the two stages alternate recursively, we instead perform this correction once across all patch steps after completing the time update recurrence. This design is rationalized by feeding the outputs generated from all calibrated stepwise states into subsequent blocks, rather than relying only on the final one, as in PatchTST (Nie et al., 2023) and other CI methods with deep sequential backbones. These outputs can therefore still interact temporally in later blocks, while the separation preserves GatedDeltaNet's chunk-parallel acceleration along the time dimension and maintains efficient cross-variable modeling.

The remaining challenge is to construct a learnable and compatible observation signal $\mathbf{Z}_l$ and Kalman gain $\mathbf{K}_l \in \mathbb{R}^{N\times N}$ analogous to Equation 3, which involves specifying $\mathbf{H}_l, \mathbf{P}_l^-, \mathbf{R}_l \in \mathbb{R}^{N\times N}$. Because the observations here are not actual physical measurements but conceptual signals derived from the input patch tokens, we produce them directly in the hidden-state space, naturally leading to $\mathbf{H}_l = \mathbf{I}$. Specifically, input projections $\mathbf{o}_{i,l} \in \mathbb{R}^{D_v}$ and $\mathbf{h}_{i,l} \in \mathbb{R}^{D_k}$ are introduced following the same design as $\mathbf{v}_{i,l}$ and $\mathbf{k}_{i,l}$, and their outer products $\mathbf{o}_{i,l}\mathbf{h}_{i,l}^\top \in \mathbb{R}^{D_v\times D_k}$ are flattened and stacked across variables to form $\mathbf{Z}_l \in \mathbb{R}^{N\times(D_vD_k)}$. Since the time update operates independently for each variable and the corrected states are not fed back into the recurrence, we accordingly represent the prior error covariance $\mathbf{P}_l^-$ as a diagonal matrix $\mathrm{diag}(\mathbf{p}_l)$, where $\mathbf{p}_l \in \mathbb{R}_+^N$. Meanwhile, the measurement noise covariance admits the covariance-correlation decomposition $\mathbf{R}_l = \mathbf{\Lambda}_l^{1/2}\mathbf{C}_l\mathbf{\Lambda}_l^{1/2}$, where $\mathbf{\Lambda}_l = \mathrm{diag}(\mathbf{d}_l)$ with $\mathbf{d}_l \in \mathbb{R}_+^N$, and $\mathbf{C}_l \in \mathbb{R}^{N\times N}$ is a correlation matrix. Both $\mathbf{p}_l$ and $\mathbf{d}_l$ are designed to be data-dependent and are elaborated later.

To build $\mathbf{C}_l$, an additional input projection $\boldsymbol{\eta}_{i,l} \in \mathbb{R}^{D_\eta}$ is normalized onto the unit sphere for each variable i. Stacking these projections yields $\boldsymbol{\eta}_l \in \mathbb{R}^{N\times D_\eta}$, whose Gram matrix $\boldsymbol{\eta}_l\boldsymbol{\eta}_l^\top$ contains pairwise cosine similarities that reflect dependencies among variables in the projected space. We further introduce a global learnable scale $\gamma \in (0,1)$ and define:

$$\mathbf{C}_l = (1-\gamma)\mathbf{I} + \gamma\boldsymbol{\eta}_l\boldsymbol{\eta}_l^\top \in \mathbb{R}^{N\times N}, \tag{7}$$

which is symmetric positive semidefinite with unit diagonals, forming a valid correlation matrix. The parameter γ controls the strength of cross-variable interactions and can attenuate them when dependencies are weak. A small initial value of γ also limits unreliable interactions early in training before the learned dependencies become informative. Defining $\mathbf{U}_l = \sqrt{\gamma}\mathbf{\Lambda}_l^{1/2}\boldsymbol{\eta}_l$ gives $\mathbf{R}_l = (1-\gamma)\mathbf{\Lambda}_l + \mathbf{U}_l\mathbf{U}_l^\top$, and thus $\mathbf{P}_l^- + \mathbf{R}_l = \mathrm{diag}(\mathbf{p}_l + (1-\gamma)\mathbf{d}_l) + \mathbf{U}_l\mathbf{U}_l^\top$. The Kalman gain in Equation 3 therefore admits the Woodbury form:

$$\mathbf{K}_l = \mathbf{P}_l^-(\mathbf{P}_l^- + \mathbf{R}_l)^{-1} = \mathrm{diag}(\mathbf{p}_l)\left[\mathbf{\Omega}_l - \mathbf{\Omega}_l\mathbf{U}_l\left(\mathbf{I} + \mathbf{U}_l^\top\mathbf{\Omega}_l\mathbf{U}_l\right)^{-1}\mathbf{U}_l^\top\mathbf{\Omega}_l\right] \in \mathbb{R}^{N\times N}, \tag{8}$$

where $\mathbf{\Omega}_l = \mathrm{diag}(\mathbf{p}_l + (1-\gamma)\mathbf{d}_l)^{-1}$. Although $\mathbf{C}_l$ and $\mathbf{K}_l$ are expressed as $N\times N$ matrices, neither is explicitly materialized for large N. Instead, the computation operates directly on the diagonal terms and the low-rank factor $\mathbf{U}_l$ when applying the gain to the subsequent state correction. This avoids $\mathcal{O}(N^2)$ storage and reduces the inversion complexity from $\mathcal{O}(N^3)$ to $\mathcal{O}(ND_\eta^2 + D_\eta^3)$, which is linear in N for fixed D_η and substantially improves scalability.

Since $\mathbf{p}_l$ and $\mathbf{d}_l$ reflect the uncertainty of the prior state $\mathbf{S}_{i,l}^-$ and observation signal, respectively, we make both dependent on the input tokens, with $\mathbf{p}_l$ additionally conditioned on the prior state. We therefore introduce another input projection to produce the log observation variance $\log d_{i,l}$ for each variable i. Because multiplying $\mathbf{p}_l$ and $\mathbf{d}_l$ by the same positive scalar leaves the Kalman gain unchanged, their common scale is unidentifiable. We remove this ambiguity by centering $\log d_{i,l}$ across variables, fixing the geometric mean of $\mathbf{d}_l$ to one. With this common scale fixed, rather than estimating $p_{i,l}$ directly, we model its ratio to $d_{i,l}$ through $\tau_{i,l} = \log(p_{i,l}/d_{i,l})$. Following the state readout design of GatedDeltaNet, an additional normalized query vector extracts a compact representation from $\mathbf{S}_{i,l}^-$, which is combined with the input token to produce $\tau_{i,l}$, and the prior variance is recovered as $p_{i,l} = d_{i,l}\exp(\tau_{i,l})$. Both the centered $\log d_{i,l}$ and $\tau_{i,l}$ are bounded by predefined values for numerical stability, while exponentiation guarantees $d_{i,l}, p_{i,l} > 0$.

The prior state $\mathbf{S}_l^- \in \mathbb{R}^{N\times D_v\times D_k}$ is then flattened into $\tilde{\mathbf{S}}_l^- \in \mathbb{R}^{N\times(D_vD_k)}$ and calibrated via the data-dependent Kalman gain $\mathbf{K}_l$ and observation signal $\mathbf{Z}_l$:

$$\tilde{\mathbf{S}}_l = (\mathbf{I} - \mathbf{K}_l)\tilde{\mathbf{S}}_l^- + \mathbf{K}_l\mathbf{Z}_l \in \mathbb{R}^{N\times(D_vD_k)}. \tag{9}$$

After restoring the $D_v \times D_k$ state shape, the posterior $\mathbf{S}_{i,l}$ and prior $\mathbf{S}_{i,l}^-$ are fused and read out independently for each variable through another global learnable scalar gate $\omega \in (0, 1)$:

$$\mathbf{y}_{i,l}^{\text{enc}} = [\omega\mathbf{S}_{i,l} + (1-\omega)\mathbf{S}_{i,l}^-]\,\mathbf{q}_{i,l} \in \mathbb{R}^{D_v}, \tag{10}$$

where $\mathbf{q}_{i,l} \in \mathbb{R}^{D_k}$ is the GatedDeltaNet output query vector projected from $\bar{\mathbf{x}}_{i,l}$. The prior branch provides a gated skip connection around the measurement update step to improve stability and flexibility. Similar to γ, ω can reduce cross-variable influence when dependencies are weak, while a small initialization keeps the Chameleon block close to the CI backbone early in training.

3.3 Decoder

The decoder maps the encoded representations to the forecast horizon F in a CI manner. For each variable i, the patch representations $\mathbf{Y}_i^{\text{enc}} \in \mathbb{R}^{L\times D_v}$ are flattened into a vector of dimension LD_v, followed by a SiLU activation and a linear projection to produce $\hat{\mathbf{Y}}' \in \mathbb{R}^{N\times F}$. Optional decoder blocks consisting of RMSNorm, SiLU, and a width-preserving linear layer can be appended but are typically not used. When RevIN is enabled, the final predictions $\hat{\mathbf{Y}} \in \mathbb{R}^{N\times F}$ are obtained by reversing the normalization with the stored statistics; otherwise, $\hat{\mathbf{Y}} = \hat{\mathbf{Y}}'$.

4 Experiments

Chameleon is trained with smooth L1 (Huber) loss using Adam (Kingma & Ba, 2014) under a StepLR schedule, with parameters additionally smoothed via exponential moving average. All experiments are conducted on a single NVIDIA L40S (46GB) or H100 (80GB) GPU. Full hyperparameters and code are available at `https://github.com/anonymous/Chameleon-TSF`.

4.1 Datasets

Based on Granger causality analysis and empirical comparisons of prior CI and CD methods, Abdelmalak et al. (2025) argue that current long-horizon MTSF benchmarks exhibit weak dependencies, and accordingly suggest several simulation datasets with strong coupling governed by physical ordinary differential equations (ODEs) (Gilpin, 2021). To examine the proposed measurement update, we therefore select Double Pendulum and Lorenz Coupled, two of the strongest datasets identified in these analyses. As a complementary real-world setting with a different dependency structure and higher dimensionality, PEMS (Guo et al., 2019) contains traffic flow measurements from hundreds of spatially distributed sensors on California highways. Traffic propagation can induce approximately linear relationships between sensors at temporal offsets, making maximum pairwise lagged correlation particularly informative. Although Granger causality does not indicate stronger dependencies than in the benchmarks, our analysis shows that all PEMS datasets exhibit an average maximum pairwise lagged correlation above 0.8, compared with around 0.45 for the Traffic benchmark. PEMS03 and PEMS08 are selected because they show the strongest dependencies under both lagged-correlation and Granger-causality analyses. While PEMS provides additional spatial information, it is not used by Chameleon or any of the compared baselines.

Still, we test Chameleon on seven long-horizon benchmarks to evaluate whether it can achieve SOTA performance in these common settings, where more competitive baselines are available. These experiments further evaluate the effectiveness of the GatedDeltaNet backbone, the proposed RevIN perturbation, and other temporal modeling designs, and examine whether the measurement update can exploit relatively stronger dependencies without degrading performance when they are weak. The evaluated benchmarks include ETTm1, ETTm2, ETTh1, ETTh2 (Zhou et al., 2021), Electricity, Solar Energy, and Traffic (Lai et al., 2018). Following SFNN (Sun et al., 2025), we exclude Weather, Illness, and Exchange Rate due to known issues, such as low predictability or insufficient samples.

We follow the standard practice of splitting all datasets chronologically into training, validation, and test partitions. Appendix A provides the split ratios and brief descriptions of all datasets.

Table 1: Results on strongly dependent datasets. Chameleon (CI) removes the proposed cross-variable measurement update. Best and second-best results are highlighted. Underlining denotes statistical significance over all other baselines ($p < 0.05$). The Avg./Max. increase rows report each baseline's average/maximum relative error increase over Chameleon across all settings, with larger values indicating worse performance.

Model		**Chameleon**		Chameleon (CI)		S-Mamba+ACN		TimeFilter		DUET		Crossformer	
Dataset	(F)	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Double Pendulum	96	0.0678	0.1176	0.1258	0.1880	0.0927	0.1642	0.0870	0.1616	0.2510	0.3271	0.1208	0.1975
	192	0.3053	0.3216	0.3845	0.3836	0.3325	0.3701	0.3522	0.3787	0.5044	0.5052	0.4774	0.4717
Lorenz Coupled	48	0.0179	0.0389	0.0493	0.0761	0.0708	0.1247	0.0424	0.0934	0.1570	0.2137	0.0581	0.1152
	96	0.1569	0.1842	0.3368	0.3213	0.3666	0.3837	0.2935	0.3230	0.5026	0.4758	0.3019	0.3299
PEMS03	48	0.0905	0.1920	0.1045	0.1991	0.0927	0.1968	0.1048	0.2099	0.1028	0.2050	0.0968	0.1989
	96	0.1156	0.2157	0.1359	0.2229	0.2309	0.3330	0.1407	0.2389	0.1447	0.2407	0.1211	0.2208
PEMS08	48	0.0975	0.1783	0.1187	0.1924	0.1266	0.1985	0.1617	0.2047	0.1244	0.1929	0.1525	0.2111
	96	0.1307	0.1910	0.1703	0.2115	0.1869	0.2064	0.2960	0.3193	0.1663	0.2087	0.2201	0.2335
Avg. Increase		–	–	61%	34%	81%	57%	62%	47%	178%	110%	74%	55%
Max. Increase		–	–	175%	96%	296%	221%	137%	140%	777%	449%	225%	196%

4.2 Setup

Protocol. We use the common forecast horizons $F \in \{96, 192, 336, 720\}$ for all benchmarks, while the ODE and PEMS datasets are evaluated at two horizons each. We exclude longer horizons for the chaotic ODE systems, where forecast errors grow rapidly, and use $F \in \{48, 96\}$ for PEMS due to better baseline availability. Because the conventional look-back size $T = 96$ can omit important periodic context and disadvantage methods with greater long-range temporal modeling capacity, as analyzed in Appendix C, we follow recent works (Abdelmalak et al., 2025; Sun et al., 2025; Liu et al., 2024; Qiu et al., 2025; Hu et al., 2025; Xu et al., 2024) in evaluating multiple look-back sizes and adopt SFNN's grids (Sun et al., 2025) for the benchmarks. Full dataset-specific T and F settings are provided in Appendix A. For each dataset-horizon-model combination, we evaluate every T over five independent runs, average the mean squared error (MSE) and mean absolute error (MAE) across runs, and report the results from the best T.

Baselines. We compare against six competitive methods spanning the major modeling paradigms. S-Mamba (Wang et al., 2025) with adaptive channel normalization (ACN) (Lee et al., 2025) is, to our knowledge, currently the strongest CD SSM-based method, while TimeFilter (Hu et al., 2025) and DUET (Qiu et al., 2025) are SOTA CD approaches leading under both conventional and longer look-back sizes. Crossformer (Zhang & Yan, 2023), the strongest model evaluated by Abdelmalak et al. (2025) on Double Pendulum and Lorenz Coupled at shorter horizons, is included for the strongly dependent datasets. For the benchmarks, the primarily CI SFNN (Sun et al., 2025) and CI FITS (Xu et al., 2024) provide additional strong references. When no public configuration is available, mostly for the ODE datasets, we align shared hyperparameters with Chameleon and otherwise retain each baseline's defaults.

4.3 Results

Across all ODE and PEMS datasets and horizons, Chameleon achieves the best MSE and MAE over all baselines, with statistical significance over all baselines in seven of the eight settings, as shown in Table 1. Specifically, removing the measurement update raises MSE and MAE by 61% and 34% on average, reaching maximum increases of 175% and 96%, respectively, on Lorenz Coupled ($F = 48$), supporting the effectiveness of the proposed update in capturing cross-variable dependencies. The gaps to prior CD methods are even larger, with MSE and MAE increases of 62–178% and 47–110%, respectively, relative to Chameleon. Table 2 presents the benchmark results averaged across horizons for each dataset, with individual results, cross-run standard deviations, and statistical significance reported in Appendix D. Under weaker dependencies, Chameleon still ranks first in MSE on all seven benchmarks and in MAE on six, while the baselines incur 3.2–11.3% higher MSE and 1.9–9.8% higher MAE. Notably, prior CD methods frequently trail Chameleon's CI variant on the strongly

Table 2: Results on long-horizon benchmarks, averaged across four forecasting horizons. Highlighting and Avg./Max. increase conventions follow Table 1.

Model	**Chameleon**		S-Mamba+ACN		TimeFilter		DUET		SFNN		FITS	
Dataset	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Traffic	0.3636	0.2409	0.4012	0.2744	0.3738	0.2580	0.3868	0.2548	0.3836	0.2656	0.4074	0.2783
Solar	0.1783	0.2204	0.1936	0.2531	0.2020	0.2438	0.1884	0.2150	0.1831	0.2434	0.2086	0.2495
Electricity	0.1522	0.2418	0.1590	0.2576	0.1573	0.2541	0.1575	0.2469	0.1554	0.2452	0.1609	0.2552
ETTh1	0.3870	0.4168	0.4484	0.4538	0.4477	0.4431	0.4078	0.4214	0.3964	0.4208	0.4087	0.4207
ETTh2	0.3310	0.3773	0.3850	0.4074	0.3928	0.4155	0.3440	0.3864	0.3481	0.3832	0.3331	0.3820
ETTm1	0.3298	0.3673	0.3666	0.3952	0.3509	0.3812	0.3365	0.3694	0.3329	0.3687	0.3501	0.3759
ETTm2	0.2356	0.2995	0.2654	0.3252	0.2673	0.3213	0.2512	0.3108	0.2438	0.3034	0.2416	0.3098
Avg. Increase	–	–	11.3%	9.8%	10.5%	7.2%	4.8%	1.9%	3.2%	3.8%	7.1%	6.0%
Max. Increase	–	–	16.3%	14.8%	18.7%	10.6%	6.6%	5.8%	5.5%	10.4%	17.0%	15.5%

Table 3: Ablation studies across five datasets. **Chameleon** reports the average MSE across horizons, while the other columns report the relative MSE increase over Chameleon. Chameleon (CI) deactivates the cross-variable measurement update, w/o RevIN Pert disables the stochastic RevIN perturbation, and w/o Decomp removes decomposition. GatedDeltaNet denotes the vanilla backbone without these three designs. w/ Mamba2 replaces the GatedDeltaNet backbone with Mamba2.

Dataset	**Chameleon**	GatedDeltaNet	Chameleon (CI)	w/o RevIN Pert	w/o Decomp	w/ Mamba2
PEMS08	0.1141	+26.8%	+28.5%	+2.16%	+1.09%	+0.38%
Traffic	0.3636	+2.47%	+2.31%	+2.67%	+2.53%	+1.82%
Solar	0.1783	+7.20%	+3.89%	+1.07%	+3.05%	+3.76%
ETTh2	0.3310	+0.77%	+0.55%	+0.77%	+0.80%	+0.85%
ETTm2	0.2356	+1.00%	+0.04%	+0.77%	+0.27%	+0.83%

dependent datasets or CI methods such as FITS (Xu et al., 2024) and the primarily CI SFNN (Sun et al., 2025) on the benchmarks, whereas Chameleon remains strongest across datasets with varying dependency strengths, highlighting the importance of preserving strong temporal modeling capacity.

We further conduct ablation studies across multiple datasets and evaluate changes in test MSE averaged across horizons, as shown in Table 3. We ablate the measurement update, RevIN perturbation, and decomposition, compare against vanilla GatedDeltaNet (Yang et al., 2025) without these designs but with its original convolutional layers, and replace the GatedDeltaNet backbone with Mamba2 (Dao & Gu, 2024). Together with Table 1, these results suggest that the benefit of the measurement update broadly aligns with dependency strength. Removing it increases MSE by 28.5% on PEMS08 and by even larger margins on the ODE datasets. Among the benchmarks, Traffic and Solar have higher average Granger F-scores at lag 24 (6.3 and 5.5) than ETTh2 and ETTm2 (2.2 and 1.4), respectively, and correspondingly show larger MSE increases of around 2–4% under the ablation. Even on the weakly dependent ETT datasets, the measurement update provides incremental improvements without degrading performance, with only a 4.3% increase in parameter count on average. Figure 2 provides complementary evidence through the learned effective Kalman gains $\omega \mathbf{K}_l$. For ETTh2 and Traffic, we visualize the gains with the largest mean off-diagonal magnitudes during testing, whereas the Double Pendulum example is randomly sampled. Even under this selection, Double Pendulum exhibits substantially stronger cross-variable interactions and greater reliance on $\mathbf{K}_l$, with $\omega = 0.901$, compared with 0.580 for Traffic and 0.108 for ETTh2. Moreover, the same Double Pendulum patch yields distinct gains in temporally offset input windows because the corresponding prior states differ. Such conditioning cannot be achieved by simply applying attention or an SSM along the variable dimension to patchwise representations, as in many prior works.

Meanwhile, removing the RevIN perturbation or decomposition, or replacing GatedDeltaNet with Mamba2, increases MSE by around 1–4% on Traffic and Solar. Although the effects of the perturbation and decomposition on mean MSE are sometimes incremental, especially on ETTh2 and ETTm2, they reduce cross-run standard deviation by 64% and 72% on average, respectively, providing substan-

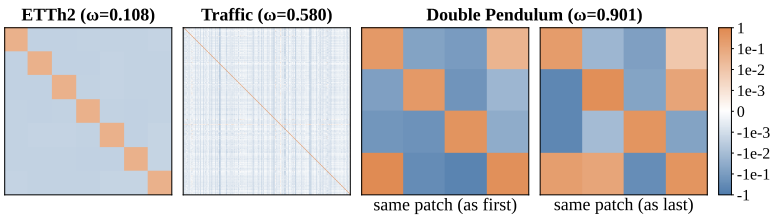
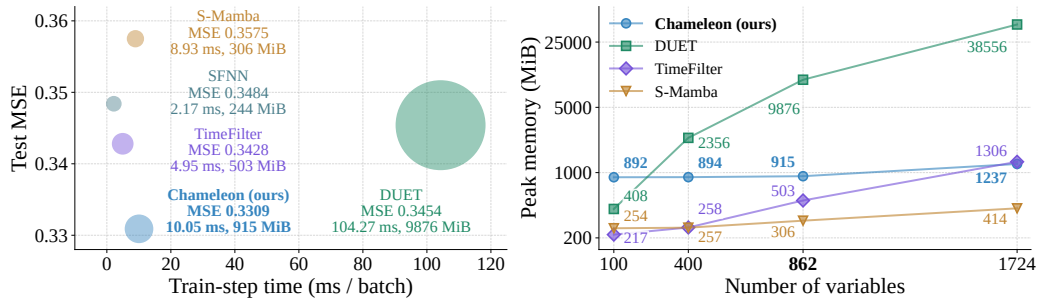


Figure 2: Learned effective Kalman gains $\omega \mathbf{K}_l$ on ETTh2, Traffic, and Double Pendulum. Larger off-diagonal magnitudes indicate stronger interactions across variables, while $\omega \in (0, 1)$ is a learned gate controlling the reliance on the raw Kalman gain $\mathbf{K}_l$. The same Double Pendulum patch yields different gains when serving as the first and last patch in two temporally offset input windows.



(a) Test MSE versus training time, with bubble areas proportional to training peak memory.

(b) Training peak memory versus number of variables.

Figure 3: Efficiency and scalability analyses on Traffic ($F = 96$) with a batch size of one.

tial stability gains with little additional overhead. Specifically, the perturbation introduces negligible computational cost, while decomposition and using GatedDeltaNet instead of Mamba2 add only 3.6% and 0.03% more parameters on average, respectively. The gap between vanilla GatedDeltaNet and full Chameleon also largely matches or exceeds the largest individual ablation gap on each dataset, suggesting little interference among the designs and confirming that the overall gains extend beyond backbone selection alone.

Finally, Figure 3 evaluates training efficiency and scalability on Traffic, the dataset with the most variables (862). Chameleon achieves the best test MSE in Figure 3a while keeping training time and peak memory within the same order of magnitude as TimeFilter (Hu et al., 2025) and S-Mamba (Wang et al., 2025), and substantially below DUET (Qiu et al., 2025). More importantly, Figure 3b varies the number of variables by retaining subsets or duplicating all variables and shows that Chameleon's peak memory scales comparably to S-Mamba, consistent with their theoretical $\mathcal{O}(N)$ complexity, whereas TimeFilter and DUET grow considerably faster with their quadratic scaling in N. Parameter counts and corresponding efficiency and scalability analyses on the ETT datasets with fewer variables are provided in Appendix D.

5 Conclusion

We presented Chameleon, a CD SSM that integrates a Kalman-filter-inspired cross-variable update mechanism into a strong GatedDeltaNet-based temporal backbone, enabling data-dependent, fine-grained interactions with linear complexity in the number of variables. On strongly dependent ODE and PEMS datasets, Chameleon achieves the best MSE and MAE in every case, while removing the update raises MSE by 61% on average. Across 28 standard benchmark settings, it also ranks first in 25 for MSE and 20 for MAE. Traffic and ETT analyses further show competitive training efficiency and favorable memory scalability across different variable counts. Together, these results suggest that Chameleon exploits strong cross-variable dependencies without sacrificing temporal modeling capacity or scalability across diverse MTSF settings.

AI USE STATEMENT

In this work, we used generative AI tools for partial code revision. We have not used generative AI tools for developing theoretical models or conceptual frameworks, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing the full method, supporting qualitative or thematic data analysis, or interpreting the results, and assistance in formulating or proving mathematical claims and cleaning or reformatting datasets are not applicable to this work. Additionally, we used generative AI tools for visualizing the analyses and polishing the writing. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Ibram Abdelmalak, Kiran Madhusudhanan, Jungmin Choi, Christian Kloetergens, Vijaya Krishna Yalavarit, Maximilian Stubbemann, and Lars Schmidt-Thieme. Channel dependence, limited lookback windows, and the simplicity of datasets: How biased is time series forecasting? *arXiv preprint arXiv:2502.09683*, 2025.

Md Atik Ahamed and Qiang Cheng. TimeMachine: A time series is worth 4 Mambas for long-term forecasting. In *The 27th European Conference on Artificial Intelligence*, pp. 1688–1695, 2024.

Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3D neural networks. *Nature*, 619(7970):533–538, 2023.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Graham Elliott and Allan Timmermann. Economic forecasting. *Journal of Economic Literature*, 46 (1):3–56, 2008.

William Gilpin. Chaos as an interpretable benchmark for forecasting and data-driven modelling. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024.

Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention-based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on Artificial Intelligence*, volume 33, pp. 922–929, 2019.

Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. SOFTS: Efficient multivariate time series forecasting with series-core fusion. *Advances in Neural Information Processing Systems*, 37: 64145–64175, 2024.

Luis Hernandez, Carlos Baladron, Javier M Aguiar, Belén Carro, Antonio J Sanchez-Esguevillas, Jaime Lloret, and Joaquim Massana. A survey on electric power demand forecasting: future trends in smart grids, microgrids and smart buildings. *IEEE Communications Surveys & Tutorials*, 16(3): 1460–1495, 2014.

Yifan Hu, Guibin Zhang, Peiyuan Liu, Disen Lan, Naiqi Li, Dawei Cheng, Tao Dai, Shu-Tao Xia, and Shirui Pan. TimeFilter: Patch-specific spatial-temporal graph filtration for time series forecasting. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Weiwei Jiang and Jiayun Luo. Graph neural network for traffic forecasting: A survey. *Expert Systems with Applications*, 207:117921, 2022.

Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

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

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

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 95–104, 2018.

Ibai Lana, Javier Del Ser, Manuel Velez, and Eleni I Vlahogianni. Road traffic forecasting: Recent advances and new challenges. *IEEE Intelligent Transportation Systems Magazine*, 10(2):93–109, 2018.

Seunghan Lee, Taeyoung Park, and Kibok Lee. Channel normalization for time series channel identification. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

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 *The Twelfth International Conference on Learning Representations*, 2024.

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.

Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, et al. Probabilistic weather forecasting with machine learning. *Nature*, 637(8044):84–90, 2025.

Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. DUET: Dual clustering enhanced multivariate time series forecasting. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pp. 1185–1196, 2025.

Lei Ren, Zidi Jia, Yuanjun Laili, and Di Huang. Deep learning for time-series prediction in iiot: progress, challenges, and prospects. *IEEE Transactions on Neural Networks and Learning Systems*, 35(11):15072–15091, 2023.

Jürgen Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139, 1992.

Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied Soft Computing*, 90:106181, 2020.

L Suganthi and Anand A Samuel. Energy models for demand forecasting—a review. *Renewable and Sustainable Energy Reviews*, 16(2):1223–1240, 2012.

Fan-Keng Sun, Yu-Cheng Wu, and Duane S Boning. Simple feedforward neural networks are almost all you need for time series forecasting. *arXiv preprint arXiv:2503.23621*, 2025.

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.

Zihan Wang, Fanheng Kong, Shi Feng, Ming Wang, Xiaocui Yang, Han Zhao, Daling Wang, and Yifei Zhang. Is Mamba effective for time series forecasting? *Neurocomputing*, 619:129178, 2025.

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*, pp. 3168–3179, 2025.

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.

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

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear Transformers with the delta rule over sequence length. *Advances in Neural Information Processing Systems*, 37: 115491–115522, 2024.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving Mamba2 with delta rule. In *The Thirteenth International Conference on Learning Representations*, 2025.

Guoqi Yu, Jing Zou, Xiaowei Hu, Angelica I Aviles-Rivero, Jing Qin, and Shujun Wang. Revitalizing multivariate time series forecasting: Learnable decomposition with inter-series dependencies and intra-series variations modeling. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2023.

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.

Table 4: Dataset descriptions. The first three have stronger cross-variable dependencies, while the others are standard long-horizon benchmarks.

Dataset	Description	
Double Pendulum	Simulated double pendulum dynamics with angles and angular velocities.	(Gilpin, 2021)
Lorenz Coupled	Two coupled simulated Lorenz systems.	(Gilpin, 2021)
PEMS	Traffic flow collected in the Caltrans PEMS.	(Guo et al., 2019)
Traffic	Road occupancy rates from 862 sensors on Bay Area freeways.	(Lai et al., 2018)
Solar	Solar power production from 137 photovoltaic plants in Alabama.	(Lai et al., 2018)
Electricity	Electricity consumption of 321 clients recorded in kWh from 2012 to 2014.	(Lai et al., 2018)
ETT	Oil temperature and six load indicators of electricity transformers.	(Zhou et al., 2021)

Table 5: Experimental setup. Split: train/validation/test ratio by time; Var.: number of variables; Freq.: sampling frequency.

Dataset	Total Length	Split	Freq.	Var.	Look-back Window Size (T)	Horizon (F)
Double Pendulum	60,000	7:1:2	N/A	4	{96, 192, 336}	{96, 192}
Lorenz Coupled	60,000	7:1:2	N/A	4	{96, 192, 336}	{48, 96}
PEMS03	26,208	6:2:2	5 min	358	{96, 288}	{48, 96}
PEMS08	17,856	6:2:2	5 min	170	{96, 288}	{48, 96}
Traffic	17,544	7:1:2	1 hour	862	{168, 336, 672, 1344}	{96, 192, 336, 720}
Solar	52,560	7:1:2	10 min	137	{144, 288, 576, 1008}	{96, 192, 336, 720}
Electricity	26,304	7:1:2	1 hour	321	{168, 336, 672, 1344}	{96, 192, 336, 720}
ETTh	14,400	6:2:2	1 hour	7	{168, 336, 672, 1344}	{96, 192, 336, 720}
ETTm	57,600	6:2:2	15 min	7	{96, 192, 384, 672, 1344}	{96, 192, 336, 720}

A Dataset Details

Descriptions and detailed settings for all datasets are provided in Table 4 and Table 5, respectively. The T grids for the benchmarks and ODE datasets follow Sun et al. (2025) and Abdelmalak et al. (2025), respectively. However, we omit $\{512, 720\}$ for the latter because neither yields the best performance across all settings tested by Abdelmalak et al. (2025).

B Preprocessing Guidance for Reversible Instance Normalization and Decomposition

Table 6: Trend and distribution-shift diagnostics computed from the training and validation splits. All values report the 95th percentile across variables. Values exceeding the selection thresholds (Trend/Mean Shift/KS > 0.2; Std. Shift > 1.2) are highlighted.

Dataset	Trend	Mean Shift	Std. Shift	KS
Double Pendulum	0.058	0.093	1.105	0.074
Lorenz Coupled	0.089	0.076	1.010	0.043
PEMS03	0.471	0.213	1.173	0.174
PEMS08	0.815	0.467	1.283	0.270
Traffic	0.782	0.507	1.604	0.303
Solar	0.222	0.056	1.058	0.072
Electricity	1.271	0.633	1.823	0.381
ETTh1	1.427	0.498	1.754	0.182
ETTh2	2.317	1.465	5.486	0.685
ETTm1	1.427	0.478	1.746	0.183
ETTm2	2.317	1.465	5.512	0.689

Table 7: RevIN ablation results at $F = 96$. Better results in each column are highlighted.

Dataset	ETTh2		ETTm2		Solar		Double Pendulum	
RevIN	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Yes	0.2628	0.3271	0.1528	0.2409	0.1534	0.1981	0.1158	0.1753
No	0.2968	0.3623	0.1695	0.2671	0.1725	0.2110	0.0678	0.1176

This section provides practical guidance for deciding whether to apply RevIN and decomposition to a given dataset. RevIN improves robustness to temporal distribution shifts in level and scale, while decomposition can preserve prominent trend structure. We therefore examine four diagnostics using only the training and validation splits. For each variable i, the Trend diagnostic is computed by fitting an ordinary least squares line over the combined training and validation segment:

$$\mathrm{Trend}_i = \frac{\left|\hat{b}_i\right| (T_{\mathrm{tv}} - 1)}{\sigma_i}, \tag{11}$$

where $\hat{b}_i$ is the fitted slope, T_{tv} is the number of time steps in the combined segment, and σ_i is its standard deviation. This measures the magnitude of the fitted net change in units of the variable's own scale. To capture scale changes between splits symmetrically, we define:

$$\text{Std. Shift}_i = \max\left(\frac{\sigma_{i,\mathrm{val}}}{\sigma_{i,\mathrm{train}}}, \frac{\sigma_{i,\mathrm{train}}}{\sigma_{i,\mathrm{val}}}\right), \tag{12}$$

where $\sigma_{i,\mathrm{train}}$ and $\sigma_{i,\mathrm{val}}$ are the corresponding standard deviations, and a value of one indicates no scale change. Mean Shift measures the change in level as

$$\text{Mean Shift}_i = \frac{|\mu_{i,\mathrm{val}} - \mu_{i,\mathrm{train}}|}{\sigma_{i,\mathrm{train}}}, \tag{13}$$

where $\mu_{i,\mathrm{train}}$ and $\mu_{i,\mathrm{val}}$ are the corresponding means. Finally, KS_i denotes the Kolmogorov–Smirnov statistic between the training and validation marginal distributions as a broader measure of distribution shift. Each diagnostic is computed per variable, and the 95th percentile across variables is reported for each dataset.

As practical selection criteria, we treat Trend, Mean Shift, and KS values above 0.2, and Std. Shift values above 1.2, as noticeable. RevIN is used when any diagnostic exceeds its threshold, while decomposition is applied when Trend exceeds 0.2. These diagnostics provide fast and interpretable guidance rather than definitive rules, while directly comparing validation performance with and without each preprocessing step can be more reliable, especially for borderline cases.

As shown in Table 6, Double Pendulum and Lorenz Coupled are the only datasets that fall below all four thresholds. We therefore remove both RevIN and decomposition for these two ODE datasets and retain them for all remaining datasets. Table 7 further supports the RevIN choice at $F = 96$, where RevIN improves both MSE and MAE on ETTh2, ETTm2, and Solar, while disabling it substantially improves both metrics on Double Pendulum. The corresponding decomposition ablation is reported in Table 3.

C Periodicity Analyses

We conduct periodicity analyses for the Traffic, PEMS08, Double Pendulum, and Lorenz Coupled datasets (Lai et al., 2018; Guo et al., 2019; Gilpin, 2021). Each analysis includes a Fourier transform, autocorrelation, and lag-copy test. Traffic and PEMS08 exhibit dominant periods of 168 and 288 time steps, corresponding to one week and one day, respectively. Both exceed the conventional look-back size of 96, supporting evaluation with larger T that preserve important periodic context. In contrast, while all seven benchmarks are periodic, the two chaotic ODE datasets have no clear dominant period. Without such regular temporal structure, their forecast errors grow much faster with horizon, and we therefore retain shorter horizons for evaluation. Together with their strong and distinct dependencies across variables, this contrasting temporal behavior makes the ODE datasets especially informative complements to the benchmarks.

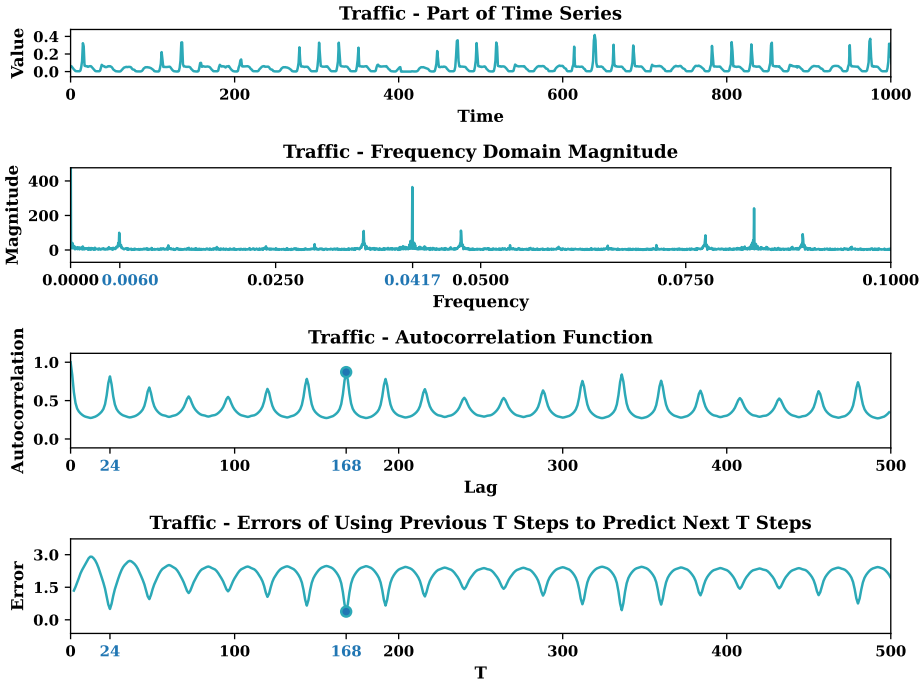


Figure 4: Traffic periodicity analysis, identifying 168 time steps (one week) as the strongest period.

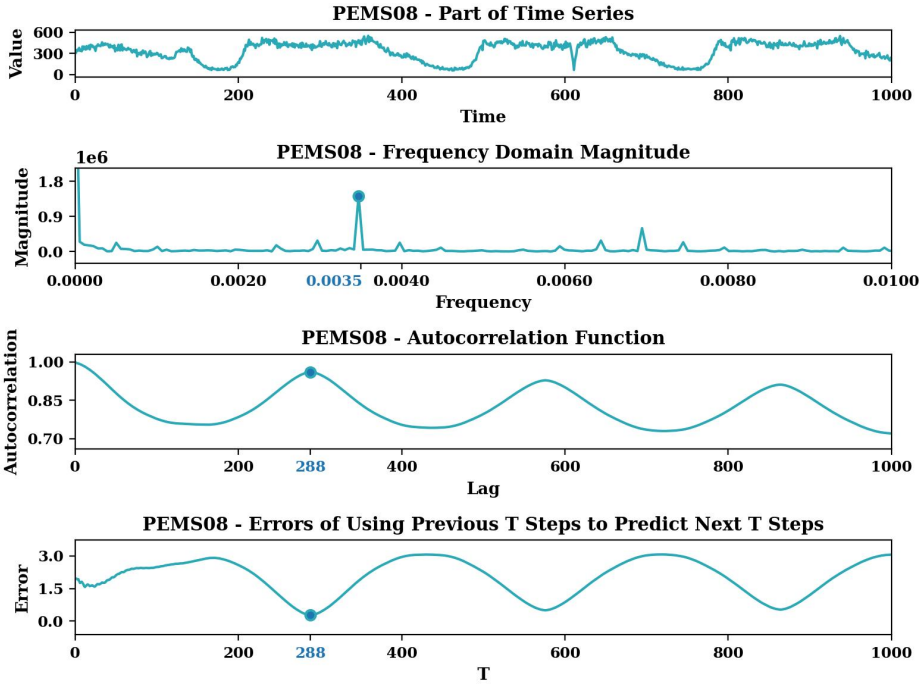


Figure 5: PEMS08 periodicity analysis, identifying 288 time steps (one day) as the strongest period.

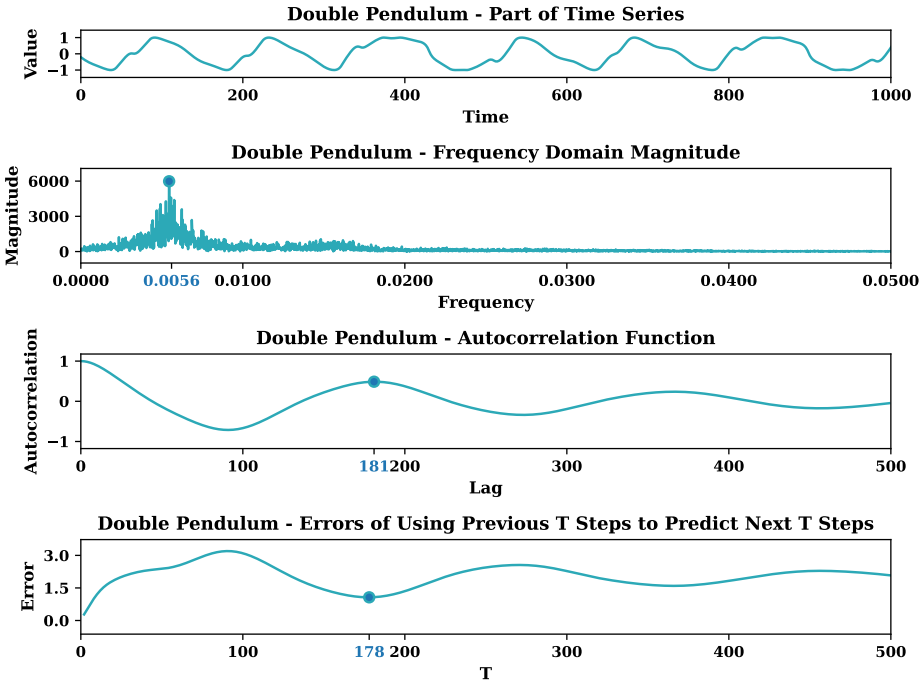


Figure 6: Double Pendulum periodicity analysis. No clear dominant period is identified.

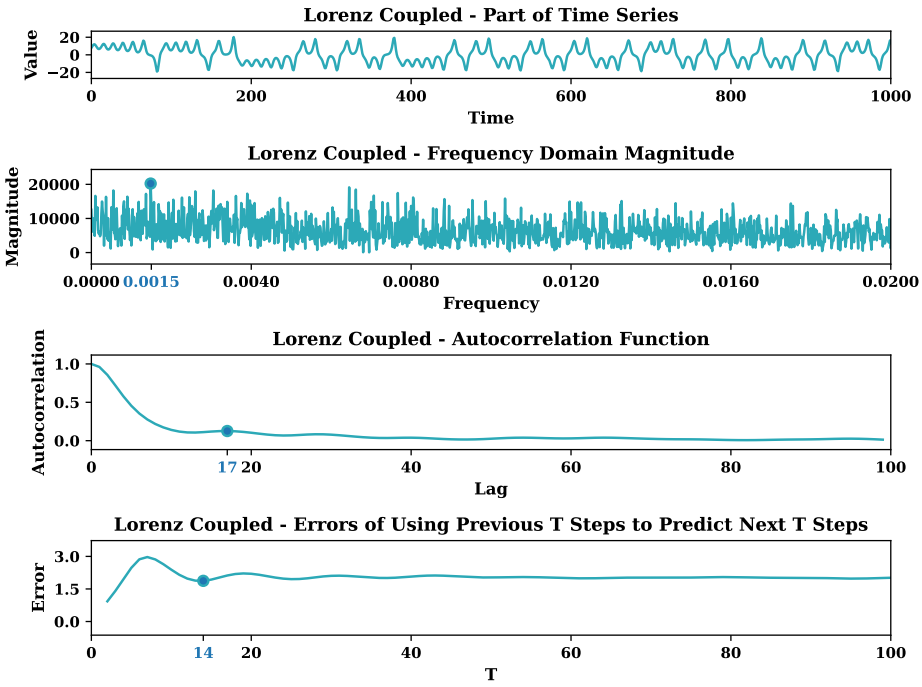


Figure 7: Lorenz Coupled periodicity analysis. No clear dominant period is identified, with even weaker periodicity than Double Pendulum.

D Full Experimental Results and Additional Analyses

Table 8: Detailed experimental results (mean ± std) on long-horizon benchmarks. Highlighting and underlining follow Table 1.

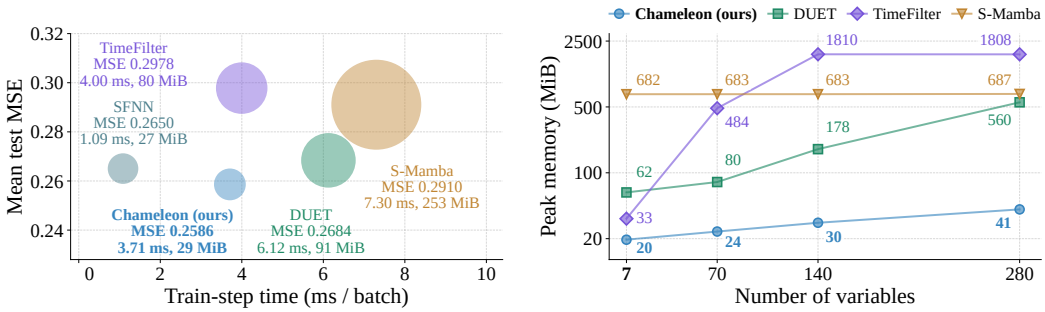
Model		**Chameleon**		S-Mamba+ACN		TimeFilter		DUET		SFNN		FITS	
Dataset	(F)	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Traffic	96	0.3309 ±0.0018	0.2247 ±0.0021	0.3653 ±0.0009	0.2563 ±0.0007	0.3428 ±0.0003	0.2412 ±0.0003	0.3493 ±0.0008	0.2367 ±0.0003	0.3484 ±0.0002	0.2478 ±0.0003	0.3820 ±0.0008	0.2674 ±0.0011
	192	0.3485 ±0.0026	0.2328 ±0.0014	0.3755 ±0.0029	0.2664 ±0.0019	0.3609 ±0.0003	0.2505 ±0.0002	0.3691 ±0.0010	0.2456 ±0.0002	0.3697 ±0.0031	0.2582 ±0.0013	0.3944 ±0.0003	0.2713 ±0.0003
	336	0.3651 ±0.0039	0.2418 ±0.0009	0.4055 ±0.0014	0.2746 ±0.0003	0.3775 ±0.0003	0.2600 ±0.0002	0.3922 ±0.0003	0.2592 ±0.0002	0.3853 ±0.0014	0.2661 ±0.0006	0.4077 ±0.0003	0.2777 ±0.0004
	720	0.4105 ±0.0047	0.2647 ±0.0026	0.4587 ±0.0045	0.3002 ±0.0016	0.4141 ±0.0004	0.2804 ±0.0003	0.4366 ±0.0038	0.2778 ±0.0002	0.4310 ±0.0010	0.2903 ±0.0007	0.4454 ±0.0004	0.2969 ±0.0006
Solar	96	0.1533 ±0.0017	0.1979 ±0.0020	0.1743 ±0.0027	0.2218 ±0.0019	0.1827 ±0.0100	0.2282 ±0.0058	0.1647 ±0.0013	0.1875 ±0.0007	0.1553 ±0.0027	0.2001 ±0.0010	0.1880 ±0.0004	0.2400 ±0.0008
	192	0.1750 ±0.0029	0.2198 ±0.0019	0.1867 ±0.0035	0.2519 ±0.0046	0.1974 ±0.0140	0.2396 ±0.0097	0.1864 ±0.0044	0.2190 ±0.0002	0.1855 ±0.0019	0.2521 ±0.0045	0.2059 ±0.0002	0.2483 ±0.0006
	336	0.1899 ±0.0024	0.2291 ±0.0022	0.1993 ±0.0026	0.2617 ±0.0019	0.2056 ±0.0059	0.2466 ±0.0056	0.1950 ±0.0017	0.2345 ±0.0023	0.1935 ±0.0024	0.2638 ±0.0031	0.2194 ±0.0003	0.2546 ±0.0003
	720	0.1951 ±0.0019	0.2346 ±0.0018	0.2141 ±0.0021	0.2771 ±0.0032	0.2225 ±0.0151	0.2606 ±0.0096	0.2077 ±0.0015	0.2189 ±0.0006	0.1980 ±0.0011	0.2574 ±0.0015	0.2211 ±0.0001	0.2552 ±0.0003
Electricity	96	0.1246 ±0.0001	0.2146 ±0.0001	0.1339 ±0.0026	0.2311 ±0.0017	0.1282 ±0.0002	0.2220 ±0.0001	0.1275 ±0.0002	0.2179 ±0.0002	0.1261 ±0.0003	0.2166 ±0.0003	0.1323 ±0.0001	0.2291 ±0.0001
	192	0.1432 ±0.0001	0.2324 ±0.0001	0.1534 ±0.0012	0.2486 ±0.0010	0.1484 ±0.0006	0.2413 ±0.0001	0.1467 ±0.0002	0.2355 ±0.0002	0.1448 ±0.0002	0.2337 ±0.0002	0.1472 ±0.0000	0.2424 ±0.0001
	336	0.1566 ±0.0001	0.2475 ±0.0000	0.1637 ±0.0023	0.2645 ±0.0027	0.1554 ±0.0006	0.2598 ±0.0001	0.1623 ±0.0005	0.2518 ±0.0006	0.1601 ±0.0003	0.2503 ±0.0001	0.1631 ±0.0000	0.2586 ±0.0001
	720	0.1842 ±0.0029	0.2727 ±0.0026	0.1849 ±0.0026	0.2861 ±0.0024	0.1972 ±0.0005	0.2932 ±0.0006	0.1936 ±0.0010	0.2825 ±0.0009	0.1908 ±0.0002	0.2802 ±0.0004	0.2012 ±0.0000	0.2906 ±0.0001
ETTh1	96	0.3428 ±0.0003	0.3844 ±0.0003	0.3889 ±0.0015	0.4082 ±0.0005	0.3801 ±0.0040	0.3992 ±0.0022	0.3579 ±0.0010	0.3830 ±0.0009	0.3503 ±0.0004	0.3846 ±0.0004	0.3739 ±0.0001	0.3952 ±0.0001
	192	0.3808 ±0.0006	0.4095 ±0.0004	0.4242 ±0.0011	0.4470 ±0.0007	0.4365 ±0.0078	0.4333 ±0.0039	0.3986 ±0.0010	0.4098 ±0.0010	0.3880 ±0.0005	0.4087 ±0.0003	0.4065 ±0.0001	0.4138 ±0.0001
	336	0.4083 ±0.0006	0.4267 ±0.0005	0.4754 ±0.0028	0.4586 ±0.0014	0.4859 ±0.0112	0.4612 ±0.0052	0.4245 ±0.0016	0.4274 ±0.0018	0.4119 ±0.0025	0.4241 ±0.0012	0.4294 ±0.0001	0.4276 ±0.0002
	720	0.4162 ±0.0010	0.4468 ±0.0004	0.5051 ±0.0147	0.5016 ±0.0083	0.4884 ±0.0274	0.4788 ±0.0135	0.4503 ±0.0159	0.4656 ±0.0104	0.4355 ±0.0010	0.4660 ±0.0016	0.4249 ±0.0002	0.4462 ±0.0003
ETTh2	96	0.2628 ±0.0004	0.3271 ±0.0005	0.3016 ±0.0082	0.3510 ±0.0045	0.3110 ±0.0173	0.3551 ±0.0148	0.2846 ±0.0022	0.3356 ±0.0016	0.2734 ±0.0032	0.3315 ±0.0019	0.2707 ±0.0002	0.3359 ±0.0001
	192	0.3250 ±0.0007	0.3688 ±0.0003	0.3887 ±0.0061	0.4028 ±0.0028	0.3824 ±0.0096	0.4018 ±0.0062	0.3333 ±0.0012	0.3758 ±0.0007	0.3392 ±0.0052	0.3743 ±0.0032	0.3305 ±0.0001	0.3742 ±0.0001
	336	0.3523 ±0.0005	0.3915 ±0.0004	0.4089 ±0.0080	0.4254 ±0.0037	0.4584 ±0.0149	0.4639 ±0.0107	0.3662 ±0.0014	0.4038 ±0.0011	0.3817 ±0.0014	0.4036 ±0.0010	0.3541 ±0.0001	0.3955 ±0.0000
	720	0.3837 ±0.0023	0.4219 ±0.0010	0.4410 ±0.0077	0.4505 ±0.0043	0.4195 ±0.0047	0.4411 ±0.0030	0.3917 ±0.0049	0.4306 ±0.0029	0.3981 ±0.0023	0.4235 ±0.0015	0.3772 ±0.0002	0.4222 ±0.0001
ETTm1	96	0.2755 ±0.0007	0.3309 ±0.0004	0.3067 ±0.0046	0.3578 ±0.0027	0.2942 ±0.0009	0.3432 ±0.0006	0.2905 ±0.0003	0.3332 ±0.0003	0.2790 ±0.0003	0.3306 ±0.0002	0.3039 ±0.0002	0.3455 ±0.0002
	192	0.3133 ±0.0014	0.3570 ±0.0005	0.3451 ±0.0010	0.3807 ±0.0009	0.3289 ±0.0017	0.3680 ±0.0015	0.3188 ±0.0001	0.3592 ±0.0006	0.3197 ±0.0007	0.3585 ±0.0003	0.3372 ±0.0001	0.3649 ±0.0001
	336	0.3413 ±0.0008	0.3768 ±0.0005	0.3769 ±0.0006	0.4026 ±0.0002	0.3597 ±0.0012	0.3905 ±0.0007	0.3463 ±0.0013	0.3776 ±0.0002	0.3514 ±0.0002	0.3807 ±0.0003	0.3611 ±0.0001	0.3860 ±0.0001
	720	0.3891 ±0.0003	0.4044 ±0.0004	0.4378 ±0.0024	0.4398 ±0.0013	0.4208 ±0.0014	0.4231 ±0.0009	0.3902 ±0.0001	0.4076 ±0.0003	0.3814 ±0.0003	0.4050 ±0.0008	0.3981 ±0.0000	0.4071 ±0.0000
ETTm2	96	0.1528 ±0.0004	0.2409 ±0.0003	0.1759 ±0.0013	0.2661 ±0.0024	0.1706 ±0.0013	0.2545 ±0.0019	0.1677 ±0.0004	0.2513 ±0.0006	0.1568 ±0.0007	0.2423 ±0.0005	0.1617 ±0.0003	0.2553 ±0.0002
	192	0.2075 ±0.0002	0.2805 ±0.0004	0.2360 ±0.0041	0.3066 ±0.0029	0.2306 ±0.0021	0.2968 ±0.0022	0.2264 ±0.0000	0.2891 ±0.0001	0.2118 ±0.0018	0.2819 ±0.0020	0.2170 ±0.0001	0.2905 ±0.0001
	336	0.2554 ±0.0006	0.3141 ±0.0004	0.2800 ±0.0007	0.3365 ±0.0005	0.2860 ±0.0028	0.3349 ±0.0028	0.2704 ±0.0001	0.3258 ±0.0002	0.2642 ±0.0012	0.3185 ±0.0011	0.2593 ±0.0002	0.3242 ±0.0001
	720	0.3268 ±0.0004	0.3623 ±0.0002	0.3698 ±0.0023	0.3915 ±0.0010	0.3820 ±0.0075	0.3991 ±0.0089	0.3401 ±0.0004	0.3771 ±0.0007	0.3422 ±0.0050	0.3711 ±0.0023	0.3283 ±0.0004	0.3691 ±0.0001
1^{st} Count		25	20	0	0	1	0	0	4	1	3	1	1
w/ $p < 0.05$		23	18	0	0	1	0	0	3	1	2	1	1

Tables 8 and 9 report the complete means and standard deviations across runs, together with statistical significance for each dataset-horizon combination, corresponding to Tables 2 and 1, respectively. Besides achieving the best mean MSE and MAE in most cases, Chameleon generally exhibits lower or comparable cross-run standard deviations than the other baselines, except for the lightweight and robust FITS (Xu et al., 2024).

We further conduct the same training efficiency and scalability analyses on the ETT datasets, which contain only seven variables, complementing those on Traffic in Figure 3. Figure 8a reports results

Table 9: Detailed experimental results (mean ± std) on strongly dependent datasets. Highlighting and underlining follow Table 1.

Model		**Chameleon**		Chameleon (CI)		S-Mamba+ACN		TimeFilter		DUET		Crossformer	
Dataset	(F)	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Double Pendulum	96	0.0678 ±0.0029	0.1176 ±0.0026	0.1258 ±0.0043	0.1880 ±0.0040	0.0927 ±0.0019	0.1642 ±0.0013	0.0870 ±0.0097	0.1616 ±0.0101	0.2510 ±0.0053	0.3271 ±0.0046	0.1208 ±0.0074	0.1975 ±0.0039
	192	0.3053 ±0.0051	0.3216 ±0.0033	0.3845 ±0.0035	0.3836 ±0.0027	0.3325 ±0.0032	0.3701 ±0.0024	0.3522 ±0.0144	0.3787 ±0.0123	0.5044 ±0.0050	0.5052 ±0.0036	0.4774 ±0.0209	0.4717 ±0.0217
Lorenz Coupled	48	0.0179 ±0.0016	0.0389 ±0.0014	0.0493 ±0.0015	0.0761 ±0.0017	0.0708 ±0.0004	0.1247 ±0.0005	0.0424 ±0.0025	0.0934 ±0.0059	0.1570 ±0.0036	0.2137 ±0.0037	0.0581 ±0.0027	0.1152 ±0.0014
	96	0.1569 ±0.0075	0.1842 ±0.0040	0.3368 ±0.0052	0.3213 ±0.0042	0.3666 ±0.0013	0.3837 ±0.0014	0.2935 ±0.0098	0.3230 ±0.0075	0.5026 ±0.0043	0.4758 ±0.0025	0.3019 ±0.0085	0.3299 ±0.0045
PEMS03	48	0.0905 ±0.0007	0.1920 ±0.0007	0.1045 ±0.0007	0.1991 ±0.0003	0.0927 ±0.0012	0.1968 ±0.0013	0.1048 ±0.0004	0.2099 ±0.0004	0.1028 ±0.0054	0.2050 ±0.0062	0.0968 ±0.0028	0.1989 ±0.0020
	96	0.1156 ±0.0015	0.2157 ±0.0014	0.1359 ±0.0005	0.2229 ±0.0005	0.2309 ±0.0385	0.3330 ±0.0337	0.1407 ±0.0011	0.2389 ±0.0008	0.1447 ±0.0045	0.2407 ±0.0044	0.1211 ±0.0074	0.2208 ±0.0050
PEMS08	48	0.0975 ±0.0019	0.1783 ±0.0008	0.1187 ±0.0039	0.1924 ±0.0008	0.1266 ±0.0067	0.1985 ±0.0097	0.1617 ±0.0094	0.2047 ±0.0063	0.1244 ±0.0076	0.1929 ±0.0017	0.1525 ±0.0049	0.2111 ±0.0043
	96	0.1307 ±0.0024	0.1910 ±0.0003	0.1703 ±0.0034	0.2115 ±0.0006	0.1869 ±0.0041	0.2064 ±0.0026	0.2960 ±0.0258	0.3193 ±0.0167	0.1663 ±0.0068	0.2087 ±0.0011	0.2201 ±0.0104	0.2335 ±0.0078
1^{st} Count		8	8	0	0	0	0	0	0	0	0	0	0
w/ $p < 0.05$		7	7	0	0	0	0	0	0	0	0	0	0

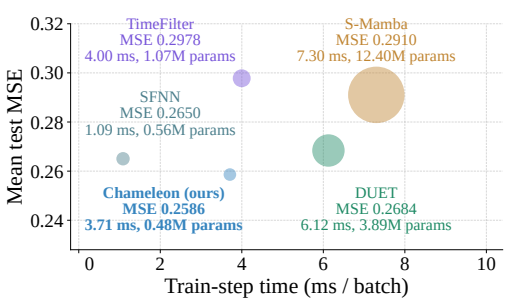


(a) Test MSE versus training time, with bubble areas proportional to training peak memory. Results are averaged across the four ETT datasets.

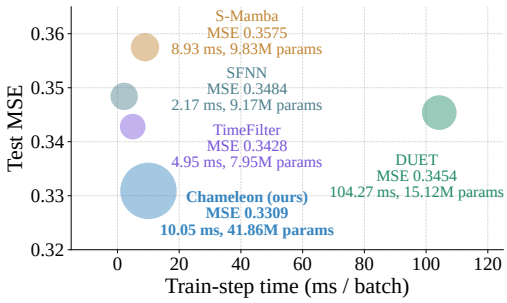
(b) Training peak memory versus number of variables on ETTh2.

Figure 8: Efficiency and scalability analyses on ETT at $F = 96$ with a batch size of one.

averaged across the four ETT datasets at $F = 96$ and shows that Chameleon achieves the best MSE while requiring the least training time and peak memory among all evaluated CD methods, with only the CI reference SFNN (Sun et al., 2025) requiring less of both. Figure 8b follows the same procedure as the Traffic analysis by duplicating all seven variables of ETTh2 to progressively increase their number. Chameleon's peak memory again scales comparably to S-Mamba (Wang et al., 2025), whereas TimeFilter (Hu et al., 2025) and DUET (Qiu et al., 2025) grow considerably faster. Figure 9 further replaces peak memory with parameter counts for both ETT and Traffic, providing a complementary view of model efficiency. Together, these additional analyses suggest that Chameleon maintains competitive efficiency and strong scalability across datasets with different numbers of variables and model configurations, rather than reflecting a single favorable setting.



(a) Test MSE versus training time, with bubble areas proportional to parameter counts. Results are averaged across the four ETT datasets.



(b) Test MSE versus training time on Traffic, with bubble areas proportional to parameter counts.

Figure 9: Additional parameter-count analyses at $F = 96$ with a batch size of one.