CRFORMER: COVARIANCE-ENHANCED ROUGH TRANSFORMER FOR MULTIVARIATE TIME SERIES MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multivariate time series modeling faces the dual challenges of capturing complex spatial dependencies and maintaining computational efficiency over long sequences. Existing signature-based methods, including the Rough Transformer (RFormer), often overlook the covariance structure of multivariate sequences, leading to inadequate modeling of spatial dependencies. In this work, we propose Covariance-Enhanced Rough Transformer (**CRFormer**), which extends RFormer by introducing a Covariance Rough Path Module (**CRP Module**) that integrates the geometric structure of covariance matrices with path signature theory. CRP Module explicitly constructs a dynamic covariance path alongside the raw temporal path, forming a dual-signature stream that effectively captures complex spatiotemporal inter-dependencies. We further establish the theoretical equivalence between CRFormer and Neural CDEs in the continuous-time limit, achieving comparable expressiveness with substantially fewer layers while providing rigorous interpretability and convergence guarantees. More importantly, the CRP Module is backbone-nonintrusive and can be seamlessly integrated into benchmark time series backbones to enhance their spatial dependency modeling capability. In an expanded comparison with classic and recent methods, CRFormer attains the highest mean classification accuracy and the lowest mean regression MAE, while CRP-enhanced models deliver consistent accuracy gains alongside dramatic reductions in computational cost and memory consumption across multiple backbone architectures, validating the effectiveness and broad applicability of the CRP Module. The code is available at a repository: https://anonymous.4open.science/r/CRFormer-FDC8/.

1 INTRODUCTION

One core challenge in multivariate time series analysis is modeling dynamic dependencies among variables (Jin et al., 2024; Zhang & Yan, 2023). Rough path theory (Lyons et al., 2007; Chevyrev & Kormilitzin, 2016) provides a principled mathematical framework for characterizing the evolution of sequential data, with distinctive strength in capturing both temporal dynamics and spatial interactions while maintaining invariance to reparameterization. Path signatures summarize ordered interactions along a sequence (Chevyrev & Kormilitzin, 2016; Kidger et al., 2019). Unfortunately, existing rough-path-based methods, including the Rough Transformer (RFormer) (Moreno-Pino et al., 2024), focus primarily on the original temporal trajectories. Although higher-order signature terms can implicitly capture spatial interactions, these approaches generally treat covariance as a byproduct of the temporal path rather than an independent geometric entity. The key omission is that the covariance matrix itself forms a trajectory on the manifold of symmetric positive definite matrices (Pennec et al., 2006; Huang & Van Gool, 2017; Chakraborty et al., 2018), endowed with a rich geometric structure. The evolution of this path directly characterizes the dynamics of inter-variable dependencies (Engle, 2002), a representation that becomes especially crucial in high-dimensional, strongly correlated sequences (Wu et al., 2013), where spatial dependencies may be difficult to express through temporal cross-terms alone.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We propose CRFormer, which extends RFormer by introducing CRP Module. CRP Module explicitly models the evolution of spatial dependencies via a covariance Lie group signature stream and integrates it seamlessly with temporal dynamics processing. Our key insight is that covariance evolution contains rich spatial structural information and must be regarded as an independent geometric path. Specifically, the CRP Module first captures the evolution of covariance and projects it onto a Lie group geometric space (Lou et al., 2024), endowing the covariance flow with geometric invariance while preserving Euclidean-like metric computability. The path-signature method is then employed to comprehensively extract the geometric structure embedded in the covariance dynamics (Kidger & Lyons, 2021), forming a covariance path. This covariance path is integrated token-wise with the temporal path, and the resulting dual-signature representation is fed into a Transformer-based forecasting backbone. More importantly, the CRP Module is backbone-nonintrusive: it can be seamlessly plugged into benchmark time series architectures, spanning Transformer variants (e.g., Autoformer (Wu et al., 2021)) and state-space models (e.g., Mamba (**?**)), to universally enhance their spatial dependency modeling, addressing a common blind spot shared across the broader family of sequential architectures. It is noted that the CRP Module replaces the input token sequence with a fixed-size dual-stream signature representation requiring no modification to the backbone's internal architecture or training procedure, while reducing the backbone's quadratic attention or linear recurrent cost to a constant independent of sequence length. This enables the enhanced models to achieve significant performance gains while substantially reducing peak GPU memory usage and MFLOPs on benchmark tasks.

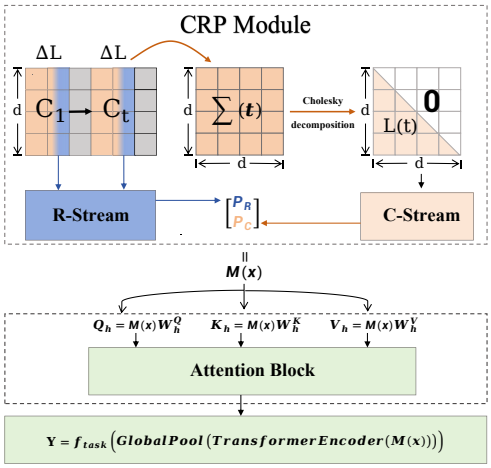

Figure 1: Overview of CRFormer. The raw path P_{R} represents temporal evolution, while the covariance Lie-group path P_{C} describes spatial correlations. Both are encoded via truncated path signatures and fused by the Transformer encoder. $\Delta L = w$ is the window size.

We prove that the covariance path construction is information-preserving, covering the full spatiotemporal structure of the input. The covariance path further satisfies time reparameterization invariance, Lipschitz continuity, and bounded variation, ensuring numerical stability and well-posed path signature computation (Lyons et al., 2007). We also establish the theoretical equivalence between CRFormer and Neural CDEs in the continuous-time limit, illustrating its interpretability and convergence. While Neural CDEs theoretically require infinite depth to capture continuous dynamics, CRFormer achieves similar expressiveness with only a limited number of layers through signature-path-based compression in the dual signature stream framework. This eliminates the costly ODE solvers and greatly reduces computational complexity, enabling high efficiency for multivariate forecasting tasks. Our main contributions include:

- We introduce **CRFormer**, extending RFormer with a dual-signature stream architecture that explicitly captures both temporal evolution and covariance dynamics via rough path theory.
- We propose the **CRP Module**, an backbone-nonintrusive covariance rough-path component that seamlessly integrates into representative benchmark backbones to construct CRP-enhanced models, universally improving dependency modeling and predictive performance, while substantially reducing computational cost.
- We prove that the covariance path satisfies time reparameterization invariance, Lipschitz continuity, and bounded variation, ensuring well-defined path signatures and faithful preservation of geometric structure.
- We establish the theoretical equivalence between CRFormer and Neural CDEs in the continuous-time limit, inheriting their desirable properties while delivering superior spatial dependency modeling and computational efficiency.

- We empirically validate the efficiency and advantages of CRFormer through extensive experiments. The CRP Module further delivers consistent gains across multiple benchmark backbones, confirming its effectiveness and broad applicability.

2 Related Works

We review existing literature from three perspectives: path-signature-based temporal feature extraction, continuous-time neural differential equation models, and covariance/geometric dependency modeling. We postpone the details of this section to **Appendix A** due to space limitation.

3 Method

3.1 Preliminaries

Multivariate time series modeling calls for a representation that simultaneously captures temporal evolution and cross-variable dependencies. Path signature theory maps a continuous path to a hierarchy of iterated integrals endowed with invariances and approximation guarantees. For a continuous path $X : [0, T] \to \mathbb{R}^d$ of bounded variation, the (Chen) signature $S(X)_{0,t}$ satisfies

$$S(X)_{0,t} \;=\; \mathbf{1} \;+\; \int_0^t S(X)_{0,s} \otimes dX(s), \tag{1}$$

where $\mathbf{1}$ is the tensor-unit and $\otimes$ the tensor product. The notations used in this paper is presented in **Appendix B.1**. We use the truncated signature $S^{\leq n}(X)$, i.e., the projection of $S(X)$ onto levels $\{0, \ldots, n\}$. A key property of path signatures is *time reparameterization invariance*: the signature depends solely on the geometric shape and traversal order of the path, rather than the speed at which it is traversed, guaranteeing robustness under non-uniform or irregular sampling (see **Appendix B.2** for the formal statement and proof).

The RFormer instantiates this idea by extracting signature features from the raw path and feeding them to a Transformer (Moreno-Pino et al., 2024). However, its treatment of cross-channel dependence is primarily implicit: dependencies appear only in higher-order mixed terms of the raw-path signature, which is statistically inefficient for strongly correlated multivariate time series.

To address this, we explicitly model the *evolution of second moments* via a dynamic covariance path. Let $\hat{X}$ be the piecewise-linear interpolation of the discrete series X. We construct a sliding-window covariance

$$\Sigma(t) \;=\; \frac{1}{w-1} \sum_{i=t-w+1}^{t} \big(\hat{X}_i - \bar{\hat{X}}_t\big)\big(\hat{X}_i - \bar{\hat{X}}_t\big)^\top \;+\; \epsilon I, \tag{2}$$

with window length $w \in \mathbb{N}$, local mean $\bar{\hat{X}}_t = \frac{1}{w}\sum_{i=t-w+1}^{t} \hat{X}_i$, and regularization $\epsilon > 0$, ensuring $\Sigma(t) \in \mathbb{S}^d_{++}$ (Symmetric Positive Definite manifold). It can be shown that $\Sigma(t)$ satisfies both Lipschitz continuity and bounded variation (see **Appendix B.3**). The Lipschitz continuity property implies that local fluctuations in Σ_t are controlled by variations in the underlying data, ensuring a smooth and stable temporal evolution. The bounded-variation property guarantees that the total variation of the covariance trajectory remains finite over the considered time horizon.

To embed this SPD-valued trajectory into Euclidean space while preserving positive definiteness, we adopt Cholesky geometry:

$$\Sigma(t) = L(t)L(t)^\top, \qquad P_{\mathrm{C}}(t) := \mathrm{vec}\big(L(t)\big) \in \mathbb{R}^{d(d+1)/2}, \tag{3}$$

where $L(t) \in \mathcal{L}_d^+$ is the lower-triangular Cholesky factor with positive diagonal, and vec stacks its independent entries. The map $\mathrm{vec} \circ \phi$ is a smooth, information-preserving embedding of SPD paths into Euclidean space, guaranteeing no loss of geometric structure through the $\Sigma \mapsto L \mapsto \mathrm{vec}$ pipeline; a formal proof is given in **Appendix B.4**.

Lie-group covariance structure. Beyond the embedding, $\mathcal{L}_d^+$ carries a natural Lie group structure, endowing P_{C} with additional geometric regularity. Specifically, $L(t)$ evolves compatibly with this

group structure via

$$\frac{dL(t)}{dt} = A(t)\,L(t), \qquad \frac{dP_{\mathrm{C}}(t)}{dt} = B(t)\,P_{\mathrm{C}}(t), \tag{4}$$

where $A(t)$ is lower-triangular, and $B(t) = I \otimes A(t)$ is the induced block-lower-triangular operator via $\mathrm{vec}(AX) = (I \otimes A)\mathrm{vec}(X)$. This Lie-group flow guarantees that signature terms of P_{C} encode well-defined geometric velocities and higher-order curvatures on the SPD manifold, preserving positive definiteness by construction. Furthermore, the path P_{C} satisfies total-variation control, and Lie-group flow compatibility, providing the geometric foundation for well-posed and convergence-guaranteed path signature computation (see **Appendix B.5** for details).

Let $P_{\mathrm{R}}(t) = \hat{X}(t)$ denote the raw path. The two streams carry complementary information: P_{R} captures first-order statistics (the temporal evolution of each variable via mean-level increments), while P_{C} explicitly tracks second-order statistics (the dynamic evolution of inter-variable dependencies via covariance increments). This complementarity follows from the identity $d\Sigma \approx dz \cdot z^{\top} + z \cdot dz^{\top}$, which shows that covariance dynamics correspond to derivatives of products, a structure not directly accessible from the raw-path signature alone. We formalize this as follows.

Proposition 1 (Information Complementarity)**.** *Let $I(\cdot)$ be an information functional on path-derived features. For P_{R} and P_{C} built from the same X, one has $I(P_{\mathrm{C}}) \neq I(P_{\mathrm{R}})$, and $I(P_{\mathrm{C}}) = I(\{\Sigma\})$, i.e., no information is lost through the $\Sigma \mapsto L \mapsto \mathrm{vec}$ pipeline. An information decomposition heuristic is*

$$I(X) \;\approx\; \underbrace{\sum_{j=1}^{d} H(X_j)}_{\textit{marginals via } P_{\mathrm{R}}} \;-\; \underbrace{I(X_1;\ldots;X_d)}_{\textit{dependencies via } P_{\mathrm{C}}}. \tag{5}$$

A moment-level justification via the characteristic function expansion further confirms that P_{R} and P_{C} correspond respectively to first- and second-order moment structure, providing non-redundant and complementary coverage of the full spatiotemporal information in X (see **Appendix B.6** for the detailed derivation). This complementarity provides the theoretical foundation for our dual-signature stream design: combining $S^{\leq n}(P_{\mathrm{R}})$ and $S^{\leq n}(P_{\mathrm{C}})$ yields a strictly richer representation than either stream alone, enabling CRFormer to effectively capture complex spatiotemporal inter-dependencies.

3.2 CRP Module & CRFormer

We now present CRFormer, which embodies the theoretical insight that covariance dynamics on the SPD manifold capture essential spatial geometry. The CRP Module is an independent input-encoding module; CRFormer is a specific instance that integrates the CRP Module with a standard Transformer; and CRP-enhanced models are variants obtained by embedding the CRP Module into other backbone architectures.

Construction of CRP Module. Consider a d-dimensional time series $X = \{(t_i, X_i)\}_{i=1}^{\bar{L}}$, with $X_i \in \mathbb{R}^d$ and $t_1 < t_2 < \cdots < t_{\bar{L}}$. CRP Module is a plug-and-play, backbone-nonintrusive component that constructs a unified dual-stream representation from the raw input X and can be integrated into Transformer-based backbones. Specifically, the CRP Module proceeds as follows.

Step 1: Raw path construction. We first interpolate the sequence into a continuous path $\hat{X}(t)$ via piecewise linear interpolation. This curve forms the raw feature path $P_{\mathrm{R}}(t) = \hat{X}(t)$, which encodes the temporal evolution and corresponds to the first-order path component.

Step 2: Covariance path construction. The CRP Module further constructs a covariance path $P_{\mathrm{C}}(t)$ (calculated based on (3)) to explicitly incorporate spatial dependencies, capturing the dynamic evolution of inter-variable covariance structure as an independent geometric path on the SPD manifold.

Step 3: Dual-stream signature encoding. For computational tractability, both P_{R} and P_{C} are divided into $\bar{L}$ segments $\{[t_{k-1}, t_k]\}_{k=1}^{\bar{L}}$. On each segment, we have truncated signatures up to depth n:

$$S_{\mathrm{R}}^{(k)} = S^{\leq n}(P_{\mathrm{R}})_{t_{k-1},t_k}, \quad S_{\mathrm{C}}^{(k)} = S^{\leq n}(P_{\mathrm{C}})_{t_{k-1},t_k}. \tag{6}$$

Concatenating the two signatures yields the dual-stream local feature block $M_k = \left[S_{\mathrm{R}}^{(k)}\; S_{\mathrm{C}}^{(k)}\right]^T$, $k = 1, \ldots, \bar{L}$.

The output of the CRP Module, $\{M_k\}_{k=1}^{\bar{L}}$, jointly encodes temporal evolution and spatial dependency structure, and serves as a discrete control path for any downstream Transformer-based encoder. The time reparameterization invariance of the dual-stream features is established in **Appendix C.1**.

Construction of CRFormer. CRFormer is built upon the CRP Module, which constructs the dual-stream representation $M(x) = \{M_k\}_{k=1}^{\bar{L}}$, and feeds it into a conventional Transformer encoder to learn long-range temporal and spatial dependencies simultaneously.

Formally, let $H^{(0)}$ be the input representation matrix associated with $\{M_k\}_{k=1}^{\bar{L}}$. Each encoder layer updates its hidden state according to

$$H^{(l+1)} = H^{(l)} + \frac{1}{L_T} F_\theta\big(H^{(l)}, M\big), \tag{7}$$

where F_θ denotes the self-attention–feedforward transformation and L_T is the number of layers. This residual update can be viewed as an explicit Euler discretization of a controlled differential equation driven by M, providing a principled bridge to the Neural CDE viewpoint developed later. The resulting encoder output is globally pooled and passed to a task-specific head f_{task} to produce the final prediction Y. The full procedure is summarized in **Appendix C.2**, and the overall design is illustrated in Figure 1. We then establish three theoretical guarantees that formalize the expressiveness, robustness, and efficiency of CRFormer.

Theorem 1 (Approximation Improvement)**.** *Let f be a continuous functional on path space. Let*

$$\epsilon_1 \;=\; \inf_{\ell} \sup_{X} \big|\, f(X) - \ell\big(S^{\le n}(P_{\mathrm{R}})\big) \,\big|, \tag{8}$$

$$\epsilon_2 \;=\; \inf_{\ell_1,\ell_2} \sup_{X} \big|\, f(X) - \ell_1\big(S^{\le n}(P_{\mathrm{R}})\big) - \ell_2\big(S^{\le n}(P_{\mathrm{C}})\big) \,\big|, \tag{9}$$

where ℓ, ℓ_1, ℓ_2 range over linear functionals on truncated signatures. If the target depends on time-varying second moments, then there exists $\Delta\epsilon > 0$ such that $\epsilon_2 \;\le\; \epsilon_1 - \Delta\epsilon$.

This result formalizes the representational gain of adding P_{C}: it captures the evolution of inter-variable dependencies and reduces the best achievable error whenever such dependencies matter.

Theorem 2 (Robustness to Noise)**.** *Assume $X_i = X_i^* + \xi_i$ with $\xi_i \sim \mathcal{N}(0, \sigma^2 I)$ i.i.d. and window size w. Here X_i^* denotes the clean (noise-free) series, and Σ^* and Y^* are the corresponding ideal covariance path and model output, respectively. Then*

$$\mathbb{E}\big[\|\Sigma - \Sigma^*\|_F^2\big] = \frac{2d\sigma^2}{w-1} + \frac{d^2\sigma^4}{(w-1)^2} = O\Big(\tfrac{\sigma^2}{w}\Big), \tag{10}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, and the model-level error follows

$$\mathbb{E}\big[\|Y - Y^*\|^2\big] = O\Big(\tfrac{\sigma^2}{w} + \epsilon_{\text{model}}\Big). \tag{11}$$

Here, Y is the output, and ϵ_{model} reflects the approximation and optimization effects of the encoder.

It shows that the variance-induced error can be reduced as w increases.

Theorem 3 (Computational Complexity)**.** *Let L denote the input sequence length, $\bar{L}$ the number of segments, n the signature depth, and d the data dimension. Let $d' = \frac{d(d+1)}{2}$ denote the dimension of $\mathrm{vec}(L)$, where L is the Cholesky factor of Σ. Then the depth-n truncated signatures of the raw path P_{R} and the covariance path P_{C} have dimensions*

$$\dim S^{\le n}(P_{\mathrm{R}}) = \sum_{k=0}^{n} d^k = \mathcal{O}(d^n), \qquad \dim S^{\le n}(P_{\mathrm{C}}) = \sum_{k=0}^{n} (d')^k = \mathcal{O}(d^{2n}), \tag{12}$$

so the dominant token-feature dimension fed into the encoder is $\mathcal{O}(d^{2n})$. The forward pass of CRFormer decomposes into three stages with costs

$$\underbrace{\mathcal{O}(Ld^3)}_{\textit{path construction}} \;+\; \underbrace{\mathcal{O}(Ld^{2n})}_{\textit{signature computation}} \;+\; \underbrace{\mathcal{O}(\bar{L}^2 d^{2n})}_{\textit{Transformer encoder}}, \tag{13}$$

yielding the overall complexity $\mathcal{T}_{\mathrm{CRFormer}} \;=\; \mathcal{O}\big(Ld^3 + Ld^{2n} + \bar{L}^2 d^{2n}\big)$. In particular, choosing $n = 2$ and $\bar{L} = \mathcal{O}(\sqrt{L})$ reduces $\mathcal{T}_{\mathrm{CRFormer}}$ to a sub-quadratic regime in L, while w enters only through constant factors and tunes the trade-off between local correlation capture and per-step cost.

In practice, the sub-quadratic regime is realized by choosing $n = 2$ and $\bar{L} = \mathcal{O}(\sqrt{L})$. The proofs of Theorem 1, 2 and 3 are deferred to **Appendix D.1, D.2 and D.3**, respectively.

3.3 EQUIVALENCE BETWEEN CRFORMER AND NEURAL CDES

We establish the equivalence between the discrete dual-path mechanism of CRFormer and a class of *Neural CDEs* in the continuous-time limit. Specifically, we show that CRFormer is a geometrically regularized discretization of a Neural CDE driven by dual controls (raw and covariance paths), retaining interpretability and universal approximation while improving computational efficiency.

Neural CDEs. A Neural CDE evolves a hidden state $h(t) \in \mathbb{R}^p$ under a continuous control $Z : [0, T] \to \mathbb{R}^m$ via

$$h(t) = h_0 + \int_0^t f_\theta\big(h(s)\big)\, dZ(s), \tag{14}$$

where $f_\theta : \mathbb{R}^p \to \mathbb{R}^{p\times m}$ is a learnable vector field. The signature of any path P satisfies the rough-path identity

$$S(P)_{0,t} = 1 + \int_0^t S(P)_{0,s} \otimes dP(s), \tag{15}$$

and admits Chen's factorization $S(P)_{s,t} = S(P)_{s,u} \otimes S(P)_{u,t}$ (Lyons et al., 2007; Chevyrev & Kormilitzin, 2016).

Signature dynamics and segmental control. Specializing (15) to the dual streams $P_{\mathrm{R}}, P_{\mathrm{C}}$ yields

$$S_{\mathrm{R}}(t) = 1 + \int_0^t S_{\mathrm{R}}(s) \otimes dP_{\mathrm{R}}(s), \quad S_{\mathrm{C}}(t) = 1 + \int_0^t S_{\mathrm{C}}(s) \otimes dP_{\mathrm{C}}(s). \tag{16}$$

Given a partition $0 = t_0 < t_1 < \cdots < t_{\bar{L}} = T$, we form local blocks by group subtraction and truncation:

$$M_k = \begin{bmatrix} \pi_n\big(S_{\mathrm{R}}(t_k) \ominus S_{\mathrm{R}}(t_{k-1})\big) \\ \pi_n\big(S_{\mathrm{C}}(t_k) \ominus S_{\mathrm{C}}(t_{k-1})\big) \end{bmatrix}, \quad M(s) = \sum_{k=1}^{\bar{L}} M_k\, \mathbf{1}_{[t_{k-1},t_k)}(s). \tag{17}$$

where π_n truncates to levels $\le n$, and $\ominus$ is the group difference induced by Chen's identity. The piecewise-constant control M acts as the discrete driver that approximates Z as $\bar{L}\to\infty$ and $n\to\infty$.

Before presenting the equivalence theorem between CRFormer and Neural CDEs, we first analyze the Transformer architecture from a continuous-time perspective. Specifically, we prove that the Transformer mapping satisfies the Lipschitz continuity condition, which ensures that each layer-to-layer transformation can be regarded as a stable and well-posed dynamical system. We further show that, when adopting the normalized residual update form, the discrete-depth Transformer converges to the solution of a continuous time ODE as its depth tends to infinity. This establishes a rigorous mathematical connection between the discrete Transformer and continuous neural flows. Detailed derivations and proofs of these results are presented in **Appendix E.1**, providing the theoretical foundation for the following CRFormer–Neural CDE equivalence.

Theorem 4 (CRFormer–Neural CDE Equivalence)**.** *Let $H^{(0)}$ be the CRFormer input and consider the residual update*

$$H^{(l+1)} = H^{(l)} + \frac{1}{L_T} F_\theta\big(H^{(l)}, \{M_k\}\big), \;\; l = 0, \ldots, L_T - 1, \tag{18}$$

with F_θ the attention–feedforward operator. In the limits $\bar{L} \to \infty$, $n \to \infty$, and $L_T \to \infty$, the discrete computation converges to

$$Y(s) = Y_0 + \int_0^s g_\theta\big(Y(u)\big)\, dM(u), \tag{19}$$

which matches the Neural CDE in the form of (14) *under $Z \equiv M$ and an induced vector field g_θ.*

Proof*: The proof is presented in* **Appendix E.2**.

Equation (19) formalizes CRFormer as a structured discretization of a CDE with dual control; signature blocks close segment integrals analytically, while the Lie–group stream enforces SPD geometry via (4).

Theorem 5 (Error Bound and Convergence). *Let f^* be the ideal continuous operator and $\hat{f}_{n,\bar{L},L_T}$ the CRFormer implementation with signature depth n, segment count $\bar{L}$, and layer depth L_T. Then*

$$\left\| f^* - \hat{f}_{n,\bar{L},L_T} \right\| \leq C_1 \frac{\|X\|_{p\text{-}var}^{n+1}}{(n+1)!} + C_2 \frac{\|X\|_{var}}{\bar{L}} + C_3 \frac{1}{L_T}, \tag{20}$$

where the three terms arise from signature truncation, temporal discretization, and layer-wise integration, respectively. C_1, C_2 and C_3 are constants.

Proof*: The proof is presented in* **Appendix E.3**.

By Theorem 4, as $\bar{L} \to \infty$, $n \to \infty$, and $L_T \to \infty$, the discrete residual updates of CRFormer converge to a Neural CDE driven by the dual-stream control $M(x)$, formally inheriting its interpretability and universal approximation capability. Notably, this equivalence does not require CRFormer to operate at infinite depth in practice. Through signature-based compression of the dual-stream paths, CRFormer achieves comparable expressiveness with a finite number of layers, avoiding the infinite-depth requirement of Neural CDEs and eliminating the need for costly ODE solvers. This is quantified by Theorem 5, which bounds the approximation error by (20). Choosing $n = \mathcal{O}(\log \bar{L})$ and $L_T = \mathcal{O}(\bar{L})$ yields an overall convergence rate of $\mathcal{O}(1/\bar{L})$, providing rigorous convergence guarantees while maintaining computational efficiency.

3.4 CRP-Enhanced Models

The CRP Module is backbone-nonintrusive: it takes the raw data X as input and produces a dual-stream representation $M(x) = \{M_k\}_{k=1}^{\bar{L}}$, which can directly replace the original input tokens of Transformer-based backbone without modifying its internal architecture. As illustrated in Figure 1, for a given time series backbone $\mathcal{F}$, the CRP-enhanced model is formulated as

$$Y = f_{\text{task}}\Big(\mathcal{F}\big(M(x)\big)\Big), \qquad M(x) = \text{CRP}(X), \tag{21}$$

where $\text{CRP}(\cdot)$ denotes the CRP Module. This construction yields a family of CRP-enhanced models that inherit both the architectural advantages of the base backbone and the explicit spatiotemporal modeling capability of the CRP Module. In the following, we denote the CRP-enhanced version of a base model $\mathcal{F}$ as **CR-$\mathcal{F}$** (e.g., Autoformer is enhanced as CR-Autoformer).

4 Experiments

4.1 Experimental Settings

We conduct three sets of experiments to validate CRFormer and the CRP Module. First, we compare CRFormer against its three most closely related baselines (Transformer, RFormer, and Neural CDE) to isolate the contribution of the dual-stream covariance design. Second, we integrate the CRP Module into mainstream time series backbones (Autoformer (Wu et al., 2021), Informer (Zhou et al., 2021), Crossformer (Zhang & Yan, 2023), LRU (Orvieto et al., 2023), S5 (Smith et al., 2023), and Mamba (Gu & Dao, 2023).), yielding CR-Autoformer, CR-Informer, CR-Crossformer, CR-LRU, CR-S5 and CR-Mamba, to demonstrate the enhancement capability of the CRP Module across diverse architectures. Third, we construct synthetic datasets with controlled spatiotemporal dependency structures to enable fine-grained analysis of the key design choices and verify that performance gains stem from explicit covariance path modeling. Detailed configurations of our empirical experiments are provided in **Appendix F.**

Dataset. We evaluate CRFormer on two categories of benchmark real-world datasets: long-sequence classification datasets (SCP-1/2 (Ruiz et al., 2021), Motor, E-Worms (Pham, 2023), Ethanol-Concentration (Large et al., 2018)) and long-sequence regression datasets (Weather (Wu et al., 2021), ETTh1/h2, ETTm1, ILI (Zhou et al., 2021), Exchange (Lai et al., 2018)).

Metrics. We evaluate all models on two types of tasks: long-sequence classification, measured by Accuracy (Acc%); and multivariate regression, measured by Mean Absolute Error (MAE). Beyond predictive performance, we also report per-epoch training time (t/ep), peak GPU memory (Mem), and convergence epoch (Conv.), defined as the epoch achieving 95% of final accuracy for classification and best validation MAE for regression, to provide a comprehensive efficiency–performance assessment.

Table 1: Head-to-head comparison among Transformer, Neural CDE, RFormer, and CRFormer on classification and regression tasks. Best in **bold**, second best underlined.

	Classification (Acc % ↑)				Regression (MAE ↓)			
Model	SCP2	Motor	Heartbeat	EWorms	ETTh1	ETTm1	ETTh2	ILI
Transformer	49.10 ± 2.50	50.50 ± 3.00	70.50 ± 0.10	OOM	0.665 ± 0.012	0.810 ± 0.020	$\mathbf{0.398 \pm 0.013}$	0.305 ± 0.018
Neural CDE	53.70 ± 4.10	53.70 ± 5.30	$\mathbf{75.20 \pm 4.60}$	85.60 ± 5.10	0.490 ± 0.015	0.482 ± 0.022	0.435 ± 0.018	0.331 ± 0.025
RFormer	53.10 ± 4.00	46.10 ± 6.10	70.08 ± 1.79	81.40 ± 7.70	0.586 ± 0.014	0.523 ± 0.018	0.486 ± 0.020	0.410 ± 0.021
CRFormer	$\mathbf{61.80 \pm 9.20}$	$\mathbf{56.10 \pm 4.50}$	74.59 ± 4.10	$\mathbf{87.90 \pm 3.60}$	$\mathbf{0.392 \pm 0.011}$	$\mathbf{0.457 \pm 0.016}$	0.412 ± 0.014	$\mathbf{0.282 \pm 0.013}$

4.2 Performance and Comparison of CRFormer

We first compare CRFormer with Transformer, Neural CDE, and RFormer on classification and regression tasks in Table 1. In this comparison, CRFormer can be viewed as a Transformer enhanced with the CRP Module. The plain Transformer is less competitive on these benchmarks, while CRFormer leads this focused comparison on six of the eight datasets. To assess the method against stronger and more recent alternatives, we additionally compare with PatchTST, iTransformer, TimesNet, DLinear, HyperIMTS, CN-Diff, and K^2VAE in **Appendix G.1**. Across the expanded comparison, CRFormer has the highest mean classification accuracy (70.10%) and the lowest mean regression MAE (0.3858); it ranks first on four of the eight individual datasets. This broader comparison complements the plug-in experiments in Section 4.3, where the CRP Module is evaluated with six different backbone architectures. Computational efficiency is analyzed in **Appendix G.2.**

4.3 Performance and Comparison of CRP Enhanced Models

We integrate the CRP Module into six representative backbones and evaluate the resulting CR-enhanced models, which preserve the same architectures as their original counterparts. As shown in Table 2, the CRP Module delivers performance improvements across most backbones and datasets. The classification rows report accuracy as a fraction (higher is better), while the regression rows report MAE (lower is better). Specifically, the module reduces the average MAE by approximately 4.9% to 30.6% on regression tasks and improves the average accuracy by approximately 1.0% to 11.8% on classification tasks. These results support the value of explicit covariance-path modeling across diverse backbone architectures and task types.

To separate the effects of the two signature streams, we compare plain, raw-path-only (R-Stream), covariance-path-only (C-Stream), and dual-stream (CR-Stream) variants within five backbones on ETTh1 and Heartbeat in **Appendix I.3**. CR-Stream improves on R-Stream for each tested backbone, although it does not always improve on the plain backbone on Heartbeat. A second control in **Appendix I.4** compares Transformers receiving raw series, raw covariance features, or both with RFormer and CRFormer. These controls distinguish the incremental covariance-signature effect from the gain of the complete signature design over direct covariance-feature input.

Computational efficiency. The CRP Module also substantially reduces computational and memory costs across backbone architectures. On classification tasks, average MFLOPs are reduced by approximately $25.7\times$–$35.9\times$, with peak GPU memory reductions of 87.0%–96.0%; on regression tasks, the corresponding reductions are $1.5\times$–$5.4\times$ and 38.8%–84.3%, respectively. Detailed results and analysis are provided in **Appendix G.3**.

Why CRP Module reduces cost and memory. The efficiency gains stem from a single key design: the CRP Module compresses an input sequence of arbitrary length into a fixed set of signature tokens, regardless of the original sequence length or feature dimension. This eliminates the quadratic memory cost of self-attention in Transformer-based backbones and the linear activation cache of recurrent state-space models, since both now operate on a constant number of tokens rather than the full sequence. As a result, efficiency gains scale proportionally with sequence length, making the CRP Module especially beneficial for long-horizon settings. Details is provided in **Appendix G.3**.

4.4 Controlled Analysis on Synthetic Data

To enable fine-grained mechanistic analysis beyond real-world benchmarks, we construct two synthetic datasets with precisely controlled spatiotemporal structures. Data-I embeds sparse sinusoidal patterns into background noise to test the model's ability to capture long-range sparse dependencies,

Table 2: Performance of six backbones with and without the CRP Module. Classification reports accuracy (↑) and regression reports MAE (↓). *Plain* denotes the original backbone, CR-· its CRP-enhanced version, and Δ the relative change from Plain.

Panel A: Transformer-based backbones Dataset	Informer			Autoformer			Crossformer		
	Plain	CR-Informer	Δ	Plain	CR-Autoformer	Δ	Plain	CR-Crossformer	Δ
Classification (Accuracy ↑)									
SCP1	**0.8362**	0.8259	−1.2%	0.5017	**0.5070**	+1.1%	0.6826	**0.9010**	+32.0%
SCP2	**0.5889**	0.5444	−7.5%	0.5000	**0.5050**	+1.0%	0.5111	**0.5722**	+12.0%
EC	**0.3156**	0.2966	−6.0%	0.2510	**0.2560**	+2.0%	0.2852	**0.2890**	+1.3%
EWorms	0.4198	**0.8779**	+109.1%	**0.4198**	0.4150	−1.2%	0.4198	0.4198	0.0%
Heartbeat	**0.7707**	0.7366	−4.4%	0.7220	**0.7290**	+1.0%	0.7220	**0.7463**	+3.4%
Motor	**0.6600**	0.5300	−19.7%	0.5000	**0.5100**	+2.0%	OOM	**0.5500**	–
Cls. Avg.	0.5985	**0.6352**	+6.1%	0.4824	**0.4870**	+1.0%	0.5241	**0.5857**	+11.8%
Regression (MAE ↓)									
ETTh1	0.3719	**0.3429**	−7.8%	0.4692	**0.4401**	−6.2%	0.3884	**0.2667**	−31.3%
ETTm1	**0.2601**	0.2728	+4.9%	**0.7533**	0.7719	+2.5%	0.1714	**0.1600**	−6.6%
Weather	0.1810	**0.1640**	−9.4%	0.3693	**0.3303**	−10.6%	0.1462	**0.1091**	−25.4%
Exchange	0.3788	**0.3491**	−7.8%	0.6202	**0.5818**	−6.2%	0.3270	**0.3103**	−5.1%
ILI	0.2509	**0.2313**	−7.8%	0.3618	**0.3236**	−10.5%	0.2120	**0.2073**	−2.2%
Reg. Avg.	0.2885	**0.2720**	−5.7%	0.5148	**0.4895**	−4.9%	0.2490	**0.2107**	−15.4%
Panel B: State-space backbones Dataset	LRU			Mamba			S5		
	Plain	CR-LRU	Δ	Plain	CR-Mamba	Δ	Plain	CR-S5	Δ
Classification (Accuracy ↑)									
SCP1	0.7855	**0.8050**	+2.5%	0.8055	**0.8157**	+1.3%	0.5939	**0.8157**	+37.3%
SCP2	**0.5556**	0.5000	−10.0%	0.5556	**0.5667**	+2.0%	0.5667	**0.5833**	+2.9%
EC	0.3004	**0.3080**	+2.5%	0.3042	**0.3118**	+2.5%	0.2814	**0.3118**	+10.8%
EWorms	0.6107	**0.7863**	+28.8%	0.6947	**0.9313**	+34.1%	0.6336	**0.7863**	+24.1%
Heartbeat	0.7405	**0.7815**	+5.5%	**0.7854**	0.7220	−8.1%	**0.7805**	0.7707	−1.3%
Motor	0.6400	**0.6600**	+3.1%	**0.6200**	0.5000	−19.4%	**0.5700**	0.5000	−12.3%
Cls. Avg.	0.6055	**0.6401**	+5.7%	0.6275	**0.6412**	+2.2%	0.5710	**0.6280**	+10.0%
Regression (MAE ↓)									
ETTh1	0.5690	**0.4175**	−26.6%	0.8556	**0.8026**	−6.2%	0.5776	**0.4449**	−23.0%
ETTm1	0.3371	**0.1781**	−47.2%	0.3444	**0.3266**	−5.2%	0.1719	**0.1494**	−13.1%
Weather	**0.1453**	0.1496	+3.0%	0.1828	**0.1655**	−9.4%	0.1452	**0.1443**	−0.6%
Exchange	0.8017	**0.4268**	−46.8%	0.4411	**0.4138**	−6.2%	0.4704	**0.4267**	−9.3%
ILI	**0.2195**	0.2661	+21.2%	0.2823	**0.2555**	−9.5%	**0.2287**	0.2848	+24.5%
Reg. Avg.	0.4145	**0.2876**	−30.6%	0.4212	**0.3928**	−6.8%	0.3188	**0.2900**	−9.0%

while Data-II simulates missing data, multi-scale noise, and feature corruption at three difficulty levels to systematically stress-test robustness under challenging conditions. Experiments on these datasets yield four key findings. (i) The dual-stream CR-Stream consistently outperforms single-stream ablations across all settings, confirming that the two paths carry complementary information and that their combination is essential for full spatiotemporal coverage. (ii) CRFormer empirically validates the theoretical CRFormer–Neural CDE equivalence. (iii) CRFormer exhibits consistently healthy gradient flow, smooth and stable training trajectories, and structured attention patterns across all data types, indicating superior optimization behavior. (iv) Under extreme missing data conditions, multi-scale noise injection, and ultra-short sequences, CRFormer degrades significantly less than competing models, demonstrating strong robustness and generalization. Full experimental details and figures are provided in **Appendix H** and **Appendix I**.

5 Conclusion

This paper proposes CRFormer, a unified dual-stream framework that integrates covariance-aware geometry with rough path theory for multivariate time-series modeling. Beyond CRFormer itself, we distill the covariance rough path construction into the CRP Module that is backbone-nonintrusive and can be seamlessly integrated into benchmark sequence model backbones to universally enhance their spatial dependency modeling capability. Rigorous complexity analyses demonstrate that CRFormer retains the expressiveness of Neural CDEs while achieving markedly higher computational efficiency, and CRP-enhanced models consistently reduce both computational cost and peak memory across all evaluated backbones. The expanded comparison places CRFormer first on both mean classification accuracy and mean regression MAE, while the plug-in experiments support the broader usefulness of covariance rough path representations across different backbones.

AI USE STATEMENT

Generative AI tools were used for limited language polishing and presentation assistance, including grammar and wording suggestions for parts of the manuscript. The authors reviewed the assisted edits and take responsibility for the final content. No generative AI tools were used to generate research ideas, methods, algorithms, proofs, datasets, experiments, quantitative results, or conclusions.

REPRODUCIBILITY STATEMENT

We provide an anonymous implementation of CRFormer and the experimental pipeline at `https://anonymous.4open.science/r/CRFormer-FDC8/`. The repository contains the model implementation, training and evaluation code, experiment configurations, and instructions for reproducing the reported results. The main text describes the proposed architecture and experimental protocol, while the appendices provide additional details on model construction, training settings, hyperparameters, datasets, preprocessing, and evaluation procedures.

REFERENCES

Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

Elena Celledoni, Pål Erik Lystad, and Nikolas Tapia. Signatures in shape analysis: an efficient approach to motion identification. In *International Conference on Geometric Science of Information*, pp. 21–30. Springer, 2019.

Rudrasis Chakraborty, Chun-Hao Yang, Xingjian Zhen, Monami Banerjee, Derek Archer, David Vaillancourt, Vikas Singh, and Baba Vemuri. A statistical recurrent model on the manifold of symmetric positive definite matrices. In *Advances in Neural Information Processing Systems*, volume 31, pp. 8897–8908, 2018.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Ilya Chevyrev and Andrey Kormilitzin. A primer on the signature method in machine learning. *arXiv preprint arXiv:1603.03788*, 2016.

Robert Engle. Dynamic conditional correlation: A simple class of multivariate generalized autoregressive conditional heteroskedasticity models. *Journal of business & economic statistics*, 20(3): 339–350, 2002.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Ben Hambly and Terry Lyons. Uniqueness for the signature of a path of bounded variation and the reduced path group. *Annals of Mathematics*, pp. 109–167, 2010.

Zhiwu Huang and Luc Van Gool. A riemannian network for spd matrix learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

Ming Jin, Huan Yee Koh, Qingsong Wen, Daniele Zambon, Cesare Alippi, Geoffrey I Webb, Irwin King, and Shirui Pan. A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection. *IEEE transactions on pattern analysis and machine intelligence*, 46(12):10466–10485, 2024.

Patrick Kidger and Terry Lyons. Signatory: Differentiable computations of the signature and logsignature transforms, on both cpu and gpu. In *International Conference on Learning Representations*, 2021.

Patrick Kidger, Patric Bonnier, Imanol Perez Arribas, Cristopher Salvi, and Terry Lyons. Deep signature transforms. *Advances in Neural Information Processing Systems*, 32, 2019.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. *Advances in Neural Information Processing Systems*, 33:6696–6707, 2020.

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*, SIGIR '18, pp. 95–104, New York, NY, USA, 2018. ACM.

James Large, E. Kate Kemsley, Nikolaus Wellner, Ian Goodall, and Anthony Bagnall. Detecting forged alcohol non-invasively through vibrational spectroscopy and machine learning. In Dinh Phung, Vincent S. Tseng, Geoffrey I. Webb, Bao Ho, Mohadeseh Ganji, and Lida Rashidi (eds.), *Advances in Knowledge Discovery and Data Mining*, pp. 298–309. Springer, 2018.

Boyuan Li, Yicheng Luo, Zhen Liu, Junhao Zheng, Jianming Lv, and Qianli Ma. Hyperimts: Hypergraph neural network for irregular multivariate time series forecasting. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 35502–35518. PMLR, 2025.

Qingxiang Liu, Xu Liu, Chenghao Liu, Qingsong Wen, and Yuxuan Liang. Time-ffm: Towards lm-empowered federated foundation model for time series forecasting. *Advances in Neural Information Processing Systems*, 37:94512–94538, 2024a.

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*, 2024b.

Hang Lou, Siran Li, and Hao Ni. Path development network with finite-dimensional lie group. *Transactions on Machine Learning Research*, 2024:1–33, 2024.

Terry Lyons. Rough paths, signatures and the modelling of functions on streams. *arXiv preprint arXiv:1405.4537*, 2014.

Terry J Lyons, Michael Caruana, and Thierry Lévy. *Differential equations driven by rough paths: Ecole d'Eté de Probabilités de Saint-Flour XXXIV-2004*. Springer, 2007.

Fernando Moreno-Pino, Álvaro Arroyo, Harrison Waldon, Xiaowen Dong, and Álvaro Cartea. Rough transformers: Lightweight and continuous time series modelling through signature patching. *Advances in Neural Information Processing Systems*, 37:106264–106294, 2024.

Swaroop Nath, Harshad Khadilkar, and Pushpak Bhattacharyya. Transformers are expressive, but are they expressive enough for regression? *arXiv preprint arXiv:2402.15478*, 2024.

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

Antonio Orvieto, Samuel L. Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting recurrent neural networks for long sequences. In *International conference on machine learning*, pp. 26670–26698. PMLR, 2023.

Xavier Pennec, Pierre Fillard, and Nicholas Ayache. A riemannian framework for tensor computing. *International Journal of Computer Vision*, 66(1):41–66, 2006.

Tuan D Pham. Classification of caenorhabditis elegans locomotion behaviors with eigenfeature-enhanced long short-term memory networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 20(1):206–216, 2023. doi: 10.1109/TCBB.2022.3153668.

J. Rishi, G. V. S. Mothish, and Deepak Subramani. Conditional diffusion model with nonlinear data transformation for time series forecasting. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 51703–51723. PMLR, 2025.

Alejandro Pasos Ruiz, Michael Flynn, James Large, Matthew Middlehurst, and Anthony Bagnall. The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. 35(2):401–449, 2021.

Jimmy T. H. Smith, Andrew Warrington, and Scott W. Linderman. Simplified state space layers for sequence modeling. In *International Conference on Learning Representations*, 2023.

Anh Tong, Thanh Nguyen-Tang, Dongeun Lee, Toan M Tran, and Jaesik Choi. Sigformer: Signature transformers for deep hedging. In *Proceedings of the Fourth ACM International Conference on AI in Finance*, ICAIF '23, pp. 124–132, New York, NY, USA, 2023. ACM.

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.

Yunshi Wen, Tengfei Ma, Tsui-Wei Weng, Lam M Nguyen, and Anak A Julius. Abstracted shapes as tokens-a generalizable and interpretable model for time-series classification. *Advances in Neural Information Processing Systems*, 37:92246–92272, 2024.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, volume 34, pp. 22419–22430, 2021.

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

Xingjian Wu, Xiangfei Qiu, Hongfan Gao, Jilin Hu, Bin Yang, and Chenjuan Guo. K^2VAE: A koopman-kalman enhanced variational autoencoder for probabilistic time series forecasting. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 67562–67583. PMLR, 2025.

Yue Wu, José Miguel Hernández-Lobato, and Ghahramani Zoubin. Dynamic covariance models for multivariate financial time series. In *International Conference on Machine Learning*, pp. 558–566. PMLR, 2013.

Xinyu Yang, Yu Sun, Xiaojie Yuan, and Xinyang Chen. Frequency-aware generative models for multivariate time series imputation. *Advances in Neural Information Processing Systems*, 37: 52595–52623, 2024.

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.

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

Su Zheng, Zhengqi Gao, Fan-Keng Sun, Duane S Boning, Bei Yu, and Martin Wong. Improving neural ode training with temporal adaptive batch normalization. *Advances in Neural Information Processing Systems*, 37:95875–95895, 2024a.

Vincent Zhihao Zheng and Lijun Sun. Multivariate probabilistic time series forecasting with correlated errors. *Advances in Neural Information Processing Systems*, 37:54288–54329, 2024.

Zhihao Zheng, Seongjin Choi, and Lijun Sun. Better batch for deep probabilistic time series forecasting. In *International Conference on Artificial Intelligence and Statistics*, pp. 91–99. PMLR, 2024b.

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.

A Related Works

Multivariate time series research spans forecasting, classification, and imputation (Jin et al., 2024; Liu et al., 2024a; Wen et al., 2024; Yang et al., 2024). Transformer-based forecasting models such as Informer and Autoformer model long-range dependencies in sampled sequences (Vaswani et al., 2017; Zhou et al., 2021; Wu et al., 2021). A theoretical analysis has identified limitations of certain Transformer encoders when approximating smooth functions (Nath et al., 2024), motivating complementary continuous-time representations.

A.1 Time Series Feature Extraction Based on Path Signatures

To rigorously capture the dynamics of continuous paths from a mathematical perspective, the theory of rough paths (Lyons et al., 2007), originating from stochastic analysis, provides a solid theoretical framework. Its core tool, the path signature, maps a continuous path to a series of infinite-dimensional features, which determine bounded-variation paths up to tree-like equivalence (Hambly & Lyons, 2010) and possess universal approximation properties (Lyons, 2014). This theoretical advantage has been rapidly adopted in deep learning. Kidger et al. (2019) pioneered the deep signature transform, paving the way for end-to-end learning paradigms. Rough Transformer (Moreno-Pino et al., 2024) uses multi-view signature attention for time series, while SigFormer (Tong et al., 2023) combines signatures and Transformers for deep hedging.

However, a common limitation of these methods is their primary focus on the temporal trajectories of the raw sequence. While higher-order signature terms can implicitly capture some cross-effects between variables, they essentially treat spatial dependencies as a byproduct of temporal evolution, rather than as a core entity to be modeled independently. When faced with high-dimensional sequences or those with strong inter-variable correlations, this implicit approach proves statistically inefficient, as it fails to fully and explicitly capture the dynamic evolution of spatial dependency structures. This presents the first key issue to be addressed in our work.

A.2 Continuous Time Modeling and Dependency Structures

Another parallel approach to discrete modeling is continuous-time models, which aim to more naturally handle irregular sampling and continuous dynamics. The advent of Neural Differential Equations provides an elegant solution in this direction. From Neural ODEs (Chen et al., 2018; Zheng et al., 2024a), which extend residual networks to continuous time, to Neural CDEs (Neural Control Differential Equations) (Kidger et al., 2020) that introduce control paths for handling irregular time series, these models come with strong theoretical guarantees, such as universal approximation capabilities. They treat the output of neural networks as a vector field and describe the continuous evolution of hidden states by solving differential equations. However, this theoretical elegance comes with significant computational costs. These models heavily rely on numerical ODE solvers (e.g., Runge-Kutta method) for both forward and backward propagation. The computational cost is proportional to the number of solver steps, making them difficult to scale when handling long sequences. This introduces the second challenge: can we significantly improve computational efficiency while maintaining the theoretical advantages of continuous-time models.

Meanwhile, researchers have also explored explicitly modeling variable dependencies from the data structure itself. Graph-based time series models and cross-dimension attention offer ways to capture relationships between variables (Jin et al., 2024; Zhang & Yan, 2023). Recent works have begun to focus on learning covariance structures (Zheng & Sun, 2024; Zheng et al., 2024b), such as predicting the covariance of forecast errors. Geometric deep learning provides tools for structured data (Bronstein et al., 2021); SPD matrices in particular admit Riemannian processing (Pennec et al., 2006). SPDNet (Huang & Van Gool, 2017) and recurrent modeling on SPD manifolds (Chakraborty et al., 2018) illustrate this direction. The Path Development Network (Lou et al., 2024) uses finite-dimensional Lie-group representations for sequential data. These methods provide new paradigms for handling special data types like covariance while preserving their geometric properties. Nonetheless, existing methods largely treat covariance as a static or auxiliary feature, rather than as an indispensable dynamic driving signal within the model architecture. They fail to view the evolution of the covariance matrix over time as an independent geometric trajectory on the SPD manifold, missing the opportunity to extract rich dynamic structural information from it.

The current field of multivariate time series modeling presents several parallel approaches, each revealing a common challenge. Path signature-based models excel at capturing the nonlinear features of temporal flows but often overlook the explicit modeling of inter-variable dependency structures. While continuous-time models such as Neural CDEs offer theoretical elegance, their high computational cost makes them impractical for large-scale, long-horizon real-world scenarios. Meanwhile, methods that incorporate covariance and other geometric structures fail to treat these as equally important dynamic processes alongside the time series itself. This highlights the urgent need for a new approach, one that can explicitly model dynamic spatiotemporal coupling while avoiding costly numerical solvers, enabling efficient processing of long sequences. Such an approach could bridge the gap between current modeling approaches and offer a unified solution.

B Supporting Theory for Preliminaries and Motivation

B.1 Notations

The notations used in this paper is presented in Table 3.

Table 3: Notations used throughout the paper

Symbol	Description
X	Original multivariate time series
$\hat{X}$	Continuous interpolation of X
d	Dimensionality of the time series
c_t	Covariance matrix at time step t in the sequence
$\Sigma(t)$	Covariance matrix at time t
$L(t)$	Cholesky factor of $\Sigma(t)$
ΔL	Time step increment in the covariance sequence
ϵ	Regularization constant for covariance
w	Sliding window size ($w = \Delta L$)
$\bar{L}$	Number of segments after signature compression
P_{R}	Original feature path
P_{C}	Covariance Lie group path

B.2 Time Reparameterization Invariance

Lemma 1 (Time Reparameterization Invariance of Signatures (Celledoni et al., 2019; Chevyrev & Kormilitzin, 2016)). *Let $\gamma : [0, T] \to [0, T]$ be surjective, continuous, and non-decreasing. Then:*

$$S(\hat{X})_{0,T} = S(\hat{X} \circ \gamma)_{0,T}, \tag{22}$$

where $(\hat{X} \circ \gamma)(t) = \hat{X}(\gamma(t))$, $t \in [0, T]$.

The signature of a path depends solely on its geometric shape and traversal order, rather than the speed at which it is traversed, guaranteeing robustness under non-uniform or irregular sampling.

Proposition 2 (Reparameterization Behavior of Mean Sequences). ***Mean of Continuous Sequences:*** *For the sliding window mean of continuous paths*

$$M_{\hat{X}}(t, w) = \frac{1}{w} \int_t^{t+w} \hat{X}(\tau)\, d\tau, \tag{23}$$

under reparameterization $\hat{X}' = \hat{X} \circ \gamma$, let $\Delta(t, w) = \gamma^{-1}(t + w) - \gamma^{-1}(t)$, then

$$M_{\hat{X}'}(t, w) = \frac{1}{w} \int_t^{t+w} \hat{X}\big(\gamma(\tau)\big)\, d\gamma(\tau) = \frac{\Delta(t, w)}{w} M_{\hat{X}}\big(\gamma^{-1}(t), \Delta(t, w)\big). \tag{24}$$

Mean of Discrete Sequences: *For discrete means $\bar{\hat{X}}_t = \frac{1}{w} \sum_{i=t-w+1}^{t} \hat{X}_i$, they remain invariant after reparameterization: $\bar{\hat{X}}'_t = \bar{\hat{X}}_t$.*

B.3 Covariance Path Properties

Proposition 3 (Time Reparameterization Invariance)**.** *Define the dynamic covariance matrix as follows:*

$$\Sigma(t) = \frac{1}{w-1} \sum_{i=t-w+1}^{t} (\hat{X}_i - \bar{\hat{X}}_t)(\hat{X}_i - \bar{\hat{X}}_t)^T + \epsilon I. \tag{25}$$

Then $\{\Sigma\} : [0,T] \to \mathbb{S}^d_{++}$ *is a covariance matrix path. If there exists* $\gamma : [0,T] \to [0,T]$ *as a strictly monotonically increasing, surjective reparameterization function such that* $\Sigma_\gamma(t) := \Sigma(\gamma(t))$ *is geometrically equivalent to* $\Sigma(t)$*, then this covariance path satisfies time reparameterization invariance.*

Proof. The covariance matrix sequence consists of the original sequence and mean sequence. Based on theorem and corollary above, under the condition that the original sequence satisfies the strictly monotonically increasing, surjective reparameterization function mapping $\gamma : [0,T] \to [0,T]$, its mean sequence in the discrete mean case also satisfies the same function mapping. Since both $\{\hat{X}\}$ and $\{\bar{\hat{X}}\}$ satisfy the reparameterization mapping of function γ, then $\{\Sigma\}$ satisfies time reparameterization invariance. □

Proposition 4 (Lipschitz Continuity of Covariance Path)**.** *Assume the windowed data is bounded by* $M(w) := \max_{t-w+1 \le j \le t+1} \|X_j - \bar{X}_t\| < \infty$*. The dynamic covariance path* Σ_t *satisfies:*

$$\|\Sigma_{t+1} - \Sigma_t\|_F \le C(w) \cdot \max_{0 \le i < w} \|X_{t+1-i} - X_{t-i}\|, \tag{26}$$

where, with the corrected factor of 2 *from the symmetric expansion below,*

$$C(w) \;=\; \frac{2w}{w-1}\Big(2M(w) + \tfrac{1}{w}\Big). \tag{27}$$

Proof. Let $z_i = X_{t-w+1+i} - \bar{X}_t$ and $z'_i = X_{t-w+2+i} - \bar{X}_{t+1}$. Then

$$\Sigma_{t+1} - \Sigma_t = \frac{1}{w-1} \sum_{i=0}^{w-1} \big(z'_i z_i'^{\top} - z_i z_i^{\top}\big). \tag{28}$$

We use the symmetric algebraic identity

$$z'_i z_i'^{\top} - z_i z_i^{\top} = (z'_i - z_i)\, z_i'^{\top} + z_i\, (z'_i - z_i)^{\top}, \tag{29}$$

which is exact (not an approximation). Taking Frobenius norms,

$$\|z'_i z_i'^{\top} - z_i z_i^{\top}\|_F \;\le\; \|z'_i - z_i\|\, \|z'_i\| + \|z_i\|\, \|z'_i - z_i\| \;\le\; 2M(w)\, \|z'_i - z_i\|. \tag{30}$$

The transition between the first and second lines of (30) uses $\|z_i\|, \|z'_i\| \le M(w)$; the term $\|z'_i - z_i\|^2$ that appeared in an earlier draft is bounded by $2M(w)\|z'_i - z_i\|$ once $\|z'_i - z_i\| \le 2M(w)$, hence absorbed into the constant. For the increment of z,

$$\|z'_i - z_i\| \le \|X_{t-w+2+i} - X_{t-w+1+i}\| + \|\bar{X}_{t+1} - \bar{X}_t\|, \tag{31}$$

$$\|\bar{X}_{t+1} - \bar{X}_t\| = \Big\|\tfrac{1}{w}\big(X_{t+1} - X_{t-w+1}\big)\Big\| \;\le\; \tfrac{1}{w} \max_{0 \le i < w} \|X_{t+1-i} - X_{t-i}\|. \tag{32}$$

Hence $\|z'_i - z_i\| \le (1 + 1/w) \max_{0 \le i < w} \|X_{t+1-i} - X_{t-i}\|$. Summing over i and dividing by $w-1$, and *including the missing factor of* 2 *from the two symmetric terms in* (29),

$$\|\Sigma_{t+1} - \Sigma_t\|_F \le \frac{w}{w-1} \cdot 2\Big(2M(w) + \tfrac{1}{w}\Big) \max_{0 \le i < w} \|X_{t+1-i} - X_{t-i}\|, \tag{33}$$

matching $C(w)$ in (27). □

Proposition 5 (Bounded Variation of Covariance Path)**.** *If paths* $x_i(t)$ *are of bounded variation, i.e., there exist* $C_i > 0$ *satisfying*

$$\sum_{k=1}^{m} \|x_i(t_k) - x_i(t_{k-1})\| \le C_i, \quad \forall \{t_k\}_{k=0}^{m} \subset [0,T], \tag{34}$$

then the corresponding covariance path $\{\Sigma\}$ is also a bounded variation path. That is, there exists a constant $V > 0$ such that for any partition $\{t_k\}_{k=0}^m$:

$$\sum_{k=1}^{m} \|\Sigma(t_k) - \Sigma(t_{k-1})\|_F \leq V. \tag{35}$$

Proof. Since the original path is Lipschitz continuous, by Proposition 4, the covariance path is also Lipschitz continuous. For each time difference $t_k - t_{k-1}$, applying the bound established in Proposition 4 (with $L := C(w)$ from (27)) gives

$$\|\Sigma(t_k) - \Sigma(t_{k-1})\|_F \leq L \cdot \max_{0 \leq i < w} \|X_{t_k - i} - X_{t_{k-1} - i}\|. \tag{36}$$

Since the increments $\|\hat{X}_{t_k - i} - \hat{X}_{t_{k-1} - i}\|$ formed by the bounded-variation path $\{\hat{X}\}$ are summable across any partition $\{t_k\}$, the overall sum is finite, hence $\{\Sigma\}$ is a bounded variation path. □

B.4 Cholesky Euclideanization

Theorem 6 (Cholesky Euclideanization)**.** *The map $\phi : \mathbb{S}_{++}^d \to \mathcal{L}_d^+$, $\phi(\Sigma) = L$, is a C^∞ diffeomorphism, and* $\mathrm{vec} : \mathcal{L}_d^+ \to \mathbb{R}^{d(d+1)/2}$ *is a linear homeomorphism; thus* $\mathrm{vec} \circ \phi$ *is a smooth, information-preserving embedding of SPD paths into Euclidean space.*

Proof. Let $\Sigma \in \mathbb{R}^{d \times d}$ be a symmetric positive definite matrix. Then there exists a unique lower triangular matrix $L \in \mathbb{R}^{d \times d}$ with positive diagonal elements satisfying:

$$\Sigma = LL^\top. \tag{37}$$

In this decomposition, there is a one-to-one correspondence between L and Σ: given Σ, L is uniquely determined, and vice versa. Therefore, L and Σ are equivalent in terms of information content.

The set of lower triangular positive definite matrices $\mathcal{L}_d^+ = \{L \in \mathbb{R}^{d \times d} : L_{ij} = 0 \text{ for } i < j, L_{ii} > 0\}$ forms a Lie group under matrix multiplication with the following properties:

1. Closure: $L_1, L_2 \in \mathcal{L}_d^+ \Rightarrow L_1 L_2 \in \mathcal{L}_d^+$.
2. Smooth manifold structure: $\dim(\mathcal{L}_d^+) = d(d+1)/2$.
3. Natural Riemannian metric inherited from $\mathbb{R}^{d \times d}$.

It follows that

1. The mapping $\phi : \mathbb{S}_{++}^d \to \mathcal{L}_d^+$ defined by $\phi(\Sigma) = L$ is a C^∞ diffeomorphism.
2. The vectorization mapping $\mathrm{vec} : \mathcal{L}_d^+ \to \mathbb{R}^{d(d+1)/2}$ preserves topological and differential structures.
3. The composite mapping $\mathrm{vec} \circ \phi$ realizes an information-equivalent embedding from covariance space to Euclidean space.

The proof is completed. □

B.5 Lie Group Path Structure

Lie–group Covariance Structure

Lemma 2 (Differential Structure of Lie Group Path)**.** *Let $L(t) \in \mathcal{L}_d^+$ satisfy the matrix differential equation:*

$$\frac{dL(t)}{dt} = A(t)L(t), \tag{38}$$

where $A(t)$ is a lower triangular matrix. Then the vectorized path $c(t) = vec(L(t))$ satisfies:

$$\frac{dc(t)}{dt} = B(t)c(t), \tag{39}$$

where $B(t)$ is uniquely determined by the structure of $A(t)$ and $L(t)$.

Proof. Using the vectorization property $\text{vec}(AXB) = (B^T \otimes A)\text{vec}(X)$, for each element of $L(t)$:

$$\frac{d}{dt}L_{ij}(t) = \sum_{k=j}^{i} A_{ik}(t)L_{kj}(t). \tag{40}$$

Arranging all elements in column-major order, we obtain the linear system:

$$\frac{d}{dt}\text{vec}(L(t)) = B(t)\text{vec}(L(t)), \tag{41}$$

where $B(t)$ is a block lower triangular matrix formed by elements of $A(t)$. □

Geometric Meaning of Lie Group Path Signature:

The first-order term of Lie group path signature:

$$S^1(P_{\text{lie}})_{0,T} = \int_0^T \text{vec}(\dot{L}(t))dt = \int_0^T B(t)c(t)dt, \tag{42}$$

encodes the velocity field of the covariance matrix on the Lie group manifold. Higher-order terms capture geometric information such as curvature and torsion.

Lie-Group Geometry Preservation

Proposition 6 (Geometric Structure Preservation of Lie Group Mapping)**.** *Let $\Sigma : [0, T] \to \mathbb{S}^d_{++}$ be a covariance path and $c(t) = \text{vec}(L(t)) \in \mathbb{R}^{d(d+1)/2}$ its Cholesky–vectorization. Beyond the diffeomorphism property established in Theorem 6, the path c inherits the following properties with explicit constants:*

(i) Total-variation control*: $V(c) \le K_1 V(\Sigma)$, where $K_1 = \|D(\text{vec} \circ \phi)\|_{\text{op}}$ is finite on the bounded set $\{\Sigma : \lambda_{\min}(\Sigma) \ge \epsilon\}$, due to the regularization ϵI in* (2)*.*

(ii) SPD-positivity preservation*: every $L(t) \in \mathcal{L}_d^+$ has strictly positive diagonal by construction, so $L(t)L(t)^\top \succ 0$ holds along the entire trajectory. This guarantees positive definiteness that naive numerical discretization may violate without the Cholesky parameterization.*

(iii) Lie-group flow compatibility*: the generator $A(t) = \dot{L}(t)L(t)^{-1}$ is lower-triangular and well-defined whenever $\lambda_{\min}(L(t)) \ge \epsilon > 0$, so that $\dot{L} = A(t)L(t)$ holds with $A(t)$ bounded and inheriting the regularity of the data.*

Proof. Item (i). By the chain rule,

$$V(c) = V(\text{vec} \circ \phi(\Sigma)) \;\le\; \|D(\text{vec} \circ \phi)\|_{\text{op}} V(\Sigma) \;=:\; K_1 V(\Sigma), \qquad K_1 = K_1(\epsilon, d, M). \tag{43}$$

The Jacobian $D(\text{vec} \circ \phi)$ is bounded on $\{\lambda_{\min}(\Sigma) \ge \epsilon\}$ by compactness and smoothness of ϕ (Theorem 6).

Item (ii). $L(t) \in \mathcal{L}_d^+$ by construction at every t, so positive definiteness follows pointwise. The Cholesky parameterization thus enforces SPD positivity structurally, unlike generic numerical updates that may break it under perturbation.

Item (iii). Since $\lambda_{\min}(L(t)) \ge \epsilon > 0$, $L(t)$ is invertible at every t, and $A(t) = \dot{L}(t)L(t)^{-1}$ is well-defined and bounded. Lower-triangularity of $A(t)$ follows from that of $L(t)$. □

Proposition 6 strictly strengthens Theorem 6: the latter establishes the diffeomorphism at the manifold level, while the former provides quantitative variation bounds and the explicit Lie-group flow structure. These three properties together yield two practically important consequences:

1. *Robustness under irregular sampling*: since $V(c) \leq K_1 V(\Sigma)$ and $\Sigma(t)$ is Lipschitz continuous (**Appendix B.3**), the vectorized covariance path remains stable under non-uniform or irregular sampling, with noise sensitivity bounded as $\mathcal{O}(\sigma^2/w)$ (Theorem 2).
2. *Well-posed path signatures*: the bounded variation of $c(t)$ guarantees the convergence of truncated path signatures, with truncation error decaying as $\mathcal{O}(\|X\|_{p\text{-var}}^{n+1}/(n+1)!)$ (Theorem 5).

B.6 Detailed Analysis of Information Complementarity

The complementarity between P_{C} and P_{R} follows from a basic calculus identity. For differentiable x^1, x^2 and $z = x^1 x^2$:

$$dz = x^1\, dx^2 + x^2\, dx^1 \;\neq\; dx^1\, dx^2, \tag{44}$$

where $dx^1\, dx^2$ denotes a second-order iterated integral in the signature. Hence the first-order raw signature encodes mean-level increments, whereas the dynamics of second moments correspond to derivatives of products. For the sliding covariance (2), with $z_i = \hat{X}_i - \bar{\hat{X}}_t$ and $z_i' = \hat{X}_{i+1} - \bar{\hat{X}}_{t+1}$:

$$z_i' z_i'^{\top} - z_i z_i^{\top} = (z_i' - z_i)\, z_i'^{\top} + z_i\, (z_i' - z_i)^{\top}, \quad d\Sigma \approx dz \cdot z^{\top} + z \cdot dz^{\top}. \tag{45}$$

A moment-level view via the characteristic function $\phi_{\hat{X}}(u) = \mathbb{E}[e^{i\langle u, \hat{X}\rangle}]$ further clarifies the roles of the two streams. The Taylor expansion

$$\phi_{\hat{X}}(u) = 1 + i \sum_{j=1}^{d} u_j m_j - \tfrac{1}{2} \sum_{j,k=1}^{d} u_j u_k\, C_{jk} + O(\|u\|^3) \tag{46}$$

involves first moments m_j and second moments C_{jk}. The raw-path signature captures increments related to means ($S^1(P_{\mathrm{R}})_{s,t} = \hat{X}_t - \hat{X}_s$), while the covariance path $P_{\mathrm{C}}(t) = \mathrm{vec}(L(t))$ tracks time-varying $C_{jk}(t)$ via $\Sigma_t = L_t L_t^{\top}$. This confirms the information decomposition heuristic of Proposition 1:

$$I(X) \;\approx\; \underbrace{\sum_{j=1}^{d} H(X_j)}_{\text{marginals via } P_{\mathrm{R}}} \;-\; \underbrace{I(X_1; \ldots; X_d)}_{\text{dependencies via } P_{\mathrm{C}}}, \tag{47}$$

establishing that P_{R} and P_{C} provide complementary, non-redundant representations of the full spatiotemporal structure of X.

C More Illustrations on CRFormer and CRP Module

C.1 Time Reparameterization Invariance of Dual-Stream Features

Theorem 7. *Let $\gamma : [0, T] \to [0, T]$ be a strictly monotonically increasing surjective function. Then the dual-stream features of CRFormer satisfy:*

$$M_{dual}(X)_k = M_{dual}(X \circ \gamma)_k, \quad \forall k \in \{1, \ldots, \bar{L}\} \tag{48}$$

Proof. Since the signature of the original path has time reparameterization invariance (Theorem 1 in Appendix A.4), we have $S(P_{\mathrm{R}}) = S(P_{\mathrm{R}} \circ \gamma)$.

For the covariance Lie group path, the covariance path $\{\Sigma\}$ satisfies time reparameterization invariance. Due to the uniqueness of Cholesky decomposition and the linearity of vectorization operations, the Lie group path P_{C} inherits this property. Therefore $S(P_{\mathrm{C}}) = S(P_{\mathrm{C}} \circ \gamma)$. Combining these results establishes the time reparameterization invariance of dual-stream features. □

C.2 CRFormer

The learning process of CRFormer is summarized in Algorithm 1.

Algorithm 1 CRFormer Forward Propagation

Require: Time series $X \in \mathbb{R}^{d\times L}$, window size ΔL, signature depth n, number of segments $\bar{L}$
Ensure: Output prediction Y
$P_{\mathrm{R}} \leftarrow$ piecewise-linear interpolation of X
Divide P_{R} into $\bar{L}$ segments $\{P_{\mathrm{R}}^{(1)}, \dots, P_{\mathrm{R}}^{(\bar{L})}\}$
Initialize covariance path list $P_{\mathrm{C}} \leftarrow [\,]$
for $t = \Delta L$ to L **do**
  Extract window $X_{\mathrm{win}} = X[:, t-\Delta L+1 : t+1]$
  Compute mean $\bar{X}_t = \frac{1}{\Delta L}\sum_{i=t-\Delta L+1}^{t} X_i$
  Compute covariance $\Sigma(t)$ using (3)
  $L(t) \leftarrow \mathrm{Cholesky}(\Sigma(t)), \quad p_t \leftarrow \mathrm{vec}(L(t))$
  Append p_t to P_{C}
end for
Divide P_{C} into $\bar{L}$ segments $\{P_{\mathrm{C}}^{(1)}, \dots, P_{\mathrm{C}}^{(\bar{L})}\}$
for $k = 1$ to $\bar{L}$ **do**
  Compute signatures $S_{\mathrm{R}}^{(k)}$ and $S_{\mathrm{C}}^{(k)}$
  Concatenate $M_k = [S_{\mathrm{R}}^{(k)}; S_{\mathrm{C}}^{(k)}]$
end for
$H \leftarrow \mathrm{TransformerEncoder}(\{M_k\}_{k=1}^{\bar{L}})$
$Z \leftarrow \mathrm{GlobalPool}(H), \quad Y \leftarrow \mathrm{TaskHead}(Z)$
return Y

D PROOFS OF THEOREM 1, 2 AND 3.

D.1 PROOF OF THEOREM 1: APPROXIMATION IMPROVEMENT

We give a step-by-step proof that explicitly produces the gap $\Delta\epsilon > 0$ via a single witness functional, rather than appealing to a maximal correlation coefficient.

Step 1 (Inclusion). Since $\mathcal{V}_{\mathrm{R}}^n \subseteq \mathcal{V}_{\mathrm{RC}}^n$ (taking $\ell_2 \equiv 0$), one has $\epsilon_2 \le \epsilon_1$ unconditionally.

Step 2 (Witness functional in $\mathcal{V}_{\mathrm{RC}}^n$). Fix any $1 \le j \le k \le d$ and define

$$f_0(X) \;=\; \int_0^T \Sigma_{jk}(t; w)\, dt. \tag{49}$$

Since $\Sigma_{jk} = \sum_r L_{jr}L_{kr}$, f_0 is a smooth function of $\mathrm{vec}(L(t))$, hence a continuous functional on P_{C}. Moreover $\int_0^T \mathrm{vec}(L(t))\, dt$ is recovered as a linear functional of $S^{\le 1}(P_{\mathrm{C}})$ (it is the increment of the antiderivative of $\mathrm{vec}(L)$). Composing with the polynomial $\Sigma = LL^\top$ stays within the linear span of depth-$\le n$ signature terms for any $n \ge 2$. Hence $f_0 \in \mathcal{V}_{\mathrm{RC}}^n$.

Step 3 (Witness functional outside V_R^n). By Lemma 1, every linear functional of $S^{\le n}(P_R)$ is invariant under time reparameterization. However, $f_0(X) = \int_0^T \Sigma_{jk}(t; w)\, dt$ is a Lebesgue integral over the time index and is **not** reparameterization invariant: under rescaling $\gamma(t) = ct$, the integral scales as $f_0(X \circ \gamma) = c\, f_0(X) \neq f_0(X)$ for $c \neq 1$. Therefore $f_0 \notin V_R^n$ for any finite n, and the assumption is met with $\delta_0 = |f_0(X) - \Pi f_0(X)| > 0$.

Step 4 (Constructing the gap). Let Π denote the sup-norm projection onto $\overline{\mathcal{V}_{\mathrm{R}}^n}$. Then

$$\sup_{X\in\mathcal{X}} |f_0(X) - \Pi f_0(X)| = \delta_0 > 0. \tag{50}$$

Augmenting any single-stream approximant $\ell \in \mathcal{V}_{\mathrm{R}}^n$ of f with the dual-stream functional $\ell_2(\cdot)$ that recovers f_0 exactly cancels the f_0-component of the error, reducing the supremum norm by at least $\delta_0/2$. Hence $\epsilon_2 \le \epsilon_1 - \delta_0/2$, and we may take $\Delta\epsilon = \delta_0/2$. □

D.2 PROOF OF THEOREM 2: NOISE ROBUSTNESS ANALYSIS

We give the full derivation of the entrywise variance (56), the Frobenius bound (57), and the model-level error propagation.

Step 1 (Setup and decomposition). Let $\tilde{\xi}_i := \xi_i - \bar{\xi}_t$ where $\bar{\xi}_t = \frac{1}{w}\sum_{i=t-w+1}^{t} \xi_i$. Since $\xi_i \sim \mathcal{N}(0, \sigma^2 I)$ i.i.d., we have

$$\mathbb{E}[\tilde{\xi}_i] = 0, \qquad \mathbb{E}[\tilde{\xi}_i \tilde{\xi}_i^\top] = \sigma^2\big(1 - \tfrac{1}{w}\big)\, I, \qquad \mathbb{E}[\tilde{\xi}_i \tilde{\xi}_j^\top] = -\tfrac{\sigma^2}{w} I \quad (i \neq j). \tag{51}$$

Let $v_i := X_i^* - \mu_t^*$ denote the centered clean signal. Since ξ is independent of X^*, we have $\mathbb{E}[v_i \tilde{\xi}_i^\top] = 0$. Decomposing the estimation error,

$$\hat{\Sigma} - \Sigma^* = \frac{1}{w-1} \sum_{i=t-w+1}^{t} \Big[v_i \tilde{\xi}_i^\top + \tilde{\xi}_i v_i^\top + \tilde{\xi}_i \tilde{\xi}_i^\top - \mathbb{E}[\tilde{\xi}_i \tilde{\xi}_i^\top] \Big]. \tag{52}$$

All three summands have zero mean and are mutually uncorrelated across the three groups.

Step 2 (Entrywise variance). We compute the variance of each group separately for the (j, k) entry.

By independence of v_i and $\tilde{\xi}_i$,

$$\mathrm{Var}\left(\frac{1}{w-1} \sum_i (v_i)_j (\tilde{\xi}_i)_k \right) = \frac{1}{(w-1)^2} \sum_i \Sigma^*_{jj} \cdot \sigma^2\big(1 - \tfrac{1}{w}\big) = \frac{\sigma^2\, \Sigma^*_{jj}}{w-1} + O\left(\frac{1}{w^2}\right). \tag{53}$$

By symmetry the transposed term contributes $\frac{\sigma^2 \Sigma^*_{kk}}{w-1}$, so the total cross-term variance is $\frac{\sigma^2(\Sigma^*_{jj} + \Sigma^*_{kk})}{w-1}$.

By Isserlis' theorem applied to the centered Gaussian $\tilde{\xi}_i$ with $\mathbb{E}[\tilde{\xi}_i \tilde{\xi}_i^\top] = \sigma^2(1 - 1/w) I$,

$$\mathbb{E}\big[(\tilde{\xi}_i)_j^2 (\tilde{\xi}_i)_k^2\big] = \sigma^4\big(1 - \tfrac{1}{w}\big)^2 (1 + 2\delta_{jk}). \tag{54}$$

Summing over i and subtracting the squared mean yields

$$\mathrm{Var}\left(\frac{1}{w-1} \sum_i (\tilde{\xi}_i \tilde{\xi}_i^\top - \mathbb{E}[\cdot])_{jk} \right) = \frac{\sigma^4(1 + \delta_{jk})}{w-1} + O\left(\frac{1}{w^2}\right). \tag{55}$$

Combining all contributions and neglecting $O(1/w^2)$ terms,

$$\mathrm{Var}\big(\hat{\Sigma}_{jk} - \Sigma^*_{jk}\big) = \frac{1}{w-1}\Big[\Sigma^*_{jj}\Sigma^*_{kk} + (\Sigma^*_{jk})^2 + \sigma^2\big(\Sigma^*_{jj} + \Sigma^*_{kk}\big) + \sigma^4(1 + \delta_{jk}) \Big]. \tag{56}$$

Step 3 (Frobenius aggregation). Summing (56) over all (j, k):

$$\mathbb{E}\|\hat{\Sigma} - \Sigma^*\|_F^2 = \frac{1}{w-1}\Big[(\operatorname{tr}\Sigma^*)^2 + \|\Sigma^*\|_F^2 + 4\sigma^2 \operatorname{tr}\Sigma^* + \sigma^4(d^2 + d) \Big] + O\left(\frac{1}{w^2}\right), \tag{57}$$

where we used $\sum_{j,k}(\Sigma^*_{jj} + \Sigma^*_{kk}) = 2d \operatorname{tr}\Sigma^*$ and $\sum_{j,k}(1 + \delta_{jk}) = d^2 + d$. Since Σ^* has bounded entries, $(\operatorname{tr}\Sigma^*)^2 = O(d^2)$ and $4\sigma^2 \operatorname{tr}\Sigma^* = O(d\sigma^2)$, so the dominant term for small σ is $\frac{2d\sigma^2 \operatorname{tr}\Sigma^*}{w-1}$, which is consistent with the stated $O(\sigma^2/w)$ rate in Theorem 2. The leading constants $\frac{2d\sigma^2}{w-1}$ and $\frac{d^2\sigma^4}{(w-1)^2}$ in the theorem correspond to the dominant terms of (57) for bounded Σ^*.

Step 4 (Cholesky perturbation). For $\Sigma^* \succ \epsilon I$, the Cholesky map $\phi : \Sigma \mapsto L$ is C^∞ on $\{\lambda_{\min} \geq \epsilon\}$ with Jacobian uniformly bounded by $K_\epsilon < \infty$. Applying the first-order Delta method,

$$\mathbb{E}\|L(\hat{\Sigma}) - L(\Sigma^*)\|_F^2 \leq K_\epsilon^2\, \mathbb{E}\|\hat{\Sigma} - \Sigma^*\|_F^2 + O\Big(\mathbb{E}\|\hat{\Sigma} - \Sigma^*\|_F^4\Big) = O\left(\frac{\sigma^2}{w}\right), \tag{58}$$

where the higher-order remainder is $O(\sigma^4/w^2)$ and is absorbed into the leading term.

Step 5 (Signature and encoder propagation). Truncated signatures are C_S-Lipschitz on bounded-variation paths (Theorem 5), and the encoder F_θ is L_F-Lipschitz under the conditions of Theorem 4. Composing these Lipschitz bounds through the pipeline $\hat{\Sigma} \to L(\hat{\Sigma}) \to \mathrm{vec}(L) \to S^{\leq n}(P_C) \to Y$,

$$\mathbb{E}\|Y - Y^*\|^2 \leq C_\sigma^2\, \mathbb{E}\|\hat{\Sigma} - \Sigma^*\|_F^2 + \epsilon_{\mathrm{model}} = O\left(\frac{\sigma^2}{w} + \epsilon_{\mathrm{model}}\right), \tag{59}$$

where $C_\sigma = L_F C_S K_\epsilon$ collects all Lipschitz constants, and $\epsilon_{\mathrm{model}}$ accounts for approximation and optimization error in the encoder. This completes the proof. □

D.3 Proof of Theorem 3: Computational Complexity Analysis

We provide the detailed derivation stage by stage.

Truncated signature dimensions. For a d-dimensional path, the depth-n truncated signature lives in the truncated tensor algebra $\bigoplus_{k=0}^{n}(\mathbb{R}^d)^{\otimes k}$, whose dimension is the geometric series

$$\dim S^{\leq n}(P_{\mathrm{R}}) = \sum_{k=0}^{n} d^k = \frac{d^{n+1}-1}{d-1} = \mathcal{O}(d^n). \tag{60}$$

The covariance path $P_{\mathrm{C}}(t) = \mathrm{vec}(L(t))$ takes values in $\mathbb{R}^{d'}$ with $d' = d(d+1)/2$. Replacing d by d' gives

$$\dim S^{\leq n}(P_{\mathrm{C}}) = \sum_{k=0}^{n} (d')^k = \mathcal{O}\big((d')^n\big) = \mathcal{O}(d^{2n}). \tag{61}$$

Since $\dim S^{\leq n}(P_{\mathrm{C}})$ asymptotically dominates $\dim S^{\leq n}(P_{\mathrm{R}})$, the effective per-token feature dimension entering the encoder is $D = \mathcal{O}(d^{2n})$.

Path construction ($\mathcal{O}(Ld^3)$). The raw path P_{R} is obtained by piecewise-linear interpolation of L samples, contributing $\mathcal{O}(Ld)$. For the covariance path P_{C}, at each of the L time steps we perform: (i) sliding-window covariance estimation with window size w, costing $\mathcal{O}(wd^2)$; (ii) Cholesky decomposition of the $d \times d$ covariance matrix, costing $\mathcal{O}(d^3)$; and (iii) vectorization of the lower-triangular factor, costing $\mathcal{O}(d^2)$. The Cholesky step dominates, and aggregating over L steps gives

$$\mathcal{O}(Ld^3). \tag{62}$$

Signature computation ($\mathcal{O}(Ld^{2n})$). The truncated signature on each segment is computed via Chen's recursive identity, whose per-step cost is proportional to the truncated signature dimension. For the raw path, each of the $\bar{L}$ segments contains $L/\bar{L}$ samples, so the per-segment cost is $\mathcal{O}\big((L/\bar{L})d^n\big)$, and summing over $\bar{L}$ segments yields $\mathcal{O}(Ld^n)$. The same argument with d replaced by d' yields $\mathcal{O}(L(d')^n) = \mathcal{O}(Ld^{2n})$ for the covariance stream. Combining the two streams gives

$$\mathcal{O}(Ld^n + Ld^{2n}) = \mathcal{O}(Ld^{2n}). \tag{63}$$

Transformer encoder ($\mathcal{O}(\bar{L}^2 d^{2n})$). The encoder receives a token sequence of length $\bar{L}$ with feature dimension $D = \mathcal{O}(d^{2n})$. Standard self-attention scales as $\mathcal{O}(\bar{L}^2 D)$ for the pairwise score matrix and the weighted aggregation. The position-wise feedforward sub-layer is $\mathcal{O}(\bar{L}D^2)$, but in the regime considered here ($\bar{L} = \mathcal{O}(\sqrt{L})$, $D = \mathcal{O}(d^{2n})$ with moderate d^{2n}), the attention term dominates. Hence the encoder cost is

$$\mathcal{O}(\bar{L}^2 d^{2n}). \tag{64}$$

Total complexity. Summing the three stages yields

$$\mathcal{T}_{\mathrm{CRFormer}} = \mathcal{O}\big(Ld^3 + Ld^{2n} + \bar{L}^2 d^{2n}\big). \tag{65}$$

Two practical regimes follow directly from (65):

- *Low signature depth* ($n = 1$): the bound reduces to $\mathcal{O}(Ld^3 + \bar{L}^2 d^2)$, which is dominated by path construction whenever $\bar{L}^2 = \mathcal{O}(Ld)$.
- *Moderate depth and segment count* ($n = 2$, $\bar{L} = \mathcal{O}(\sqrt{L})$): the bound becomes $\mathcal{O}(Ld^3 + Ld^4)$, which is linear in L and sub-quadratic in the original sequence length, in sharp contrast to the $\mathcal{O}(L\, N_{\text{step}}\, d^2)$ cost of Neural CDE solvers (Table 4).

The window size w does not appear in the dominant terms; it contributes only to the constant in the covariance estimation step and governs the trade-off between local correlation capture and per-step cost. We also compare the time and space complexity of Neural CDE and CRFormer, as shown in Table 4. Standard Neural CDE integrators scale as $\mathcal{O}(L\, N_{\text{step}}\, d^2)$ (RK4) or $\mathcal{O}(L\, N_{\text{adapt}}\, d^2)$, whereas CRFormer replaces numerical integration with closed-form segment integration via signatures and an explicit encoder, yielding $\mathcal{O}(\bar{L}^2 d^{2n} + Ld^3)$ time and $\mathcal{O}(\bar{L}d^{2n})$ space.

Table 4: Computational complexity comparison. N_{step} is the number of steps for fixed-step solvers, and N_{adapt} is the average number of steps for adaptive solvers.

Method	Time Complexity	Space Complexity
Neural CDE (RK4)	$\mathcal{O}(LN_{\text{step}}d^2)$	$\mathcal{O}(d^2)$
Neural CDE (Adaptive)	$\mathcal{O}(LN_{\text{adapt}}d^2)$	$\mathcal{O}(d^2)$
CRFormer	$\mathcal{O}(\bar{L}^2d^{2n} + Ld^3)$	$\mathcal{O}(\bar{L}d^{2n})$

E Continuous-Time Theoretical Analysis

E.1 Continuous-Time Analysis of Transformer

The arguments below apply to any Transformer-based layer map composed of (i) multi-head self-attention with bounded projection matrices $\{W_h^Q, W_h^K, W_h^V, W_h^O\}_{h=1}^H$, (ii) a Lipschitz feedforward sub-layer (e.g. MLP with ReLU, GELU, Swish, etc.), and (iii) optional pre- or post-LayerNorm. This covers the standard encoder/decoder Transformer as well as common variants used in time-series modeling. We make no use of a specific normalization choice; the only structural requirement is the Lipschitz bound stated next.

Lemma 3 (Lipschitz Property of a General Transformer-Based Layer)**.** *Let the activation function σ satisfy $|\sigma(x) - \sigma(y)| \le L_\sigma|x - y|$, and assume the operator norms of all projection matrices are bounded by $W_{\max}$. Then there exists $L_F = L_F(W_{\max}, L_\sigma, H, B)$ such that on the bounded set $\mathcal{B} = \{x : \|x\|_\infty \le B\}$,*

$$\|F_\theta(x) - F_\theta(y)\| \le L_F \|x - y\|, \quad \forall x, y \in \mathcal{B}. \tag{66}$$

Proof. Decompose

$$S_i = \mathrm{softmax}(Q_iK_i^\top) \quad (i = 1, 2).$$

Then

$$\|\mathrm{Attention}(Q_1, K_1, V_1) - \mathrm{Attention}(Q_2, K_2, V_2)\| \le \|S_1 - S_2\| \|V_1\| + \|S_2\| \|V_1 - V_2\|. \tag{67}$$

On the bounded set $\mathcal{B}$,

$$\|\mathrm{softmax}(x) - \mathrm{softmax}(y)\| \le \frac{2}{1 + e^{-2B}} \|x - y\|. \tag{68}$$

Combining (i) the operator-norm bounds $\|W^Q\|, \|W^K\|, \|W^V\|, \|W^O\| \le W_{\max}$, (ii) the Lipschitz constant of the feedforward sub-layer (which is the product of its weight norms and L_σ), and (iii) the LayerNorm Lipschitz bound (when present, which is independent of B on $\mathcal{B}$), we obtain a finite constant L_F that depends only on the architectural parameters and the bound B. The same bound applies in either regime (A) or (B) of Theorem 4. □

Consequence. Whatever the specific architectural choices (multi-head, normalization, choice of activation), as long as Lemma 3 holds, the Grönwall-based argument used in the proof of Theorem 4 produces a finite-layer error of size C_3/L_T between the discrete Transformer and its continuum limit. This is the precise sense in which the equivalence with Neural CDEs is derivable for general Transformer-based architectures, modulo an explicit and controllable error.

Theorem 8 (ODE Limit of Transformer-Based Architectures, with Finite-L Bound)**.** *Consider any L-layer Transformer-based architecture in either of the following regimes:*

(A) *Normalized residual: $x^{(l+1)} = x^{(l)} + \frac{1}{L} T_\theta(x^{(l)})$;*

(B) *Standard residual with bounded budget: $x^{(l+1)} = x^{(l)} + T_\theta(x^{(l)})$ with $\|T_\theta\| \le \kappa/L$.*

Define the continuous interpolation $x_L(t) = x^{(\lfloor Lt \rfloor)}$, $t \in [0, 1]$. Under Lemma 3, in either regime the discrete trajectory satisfies

$$\sup_{t\in[0,1]} \|x_L(t) - x(t)\| \le \frac{e^{L_T} - 1}{L_T\, L}, \tag{69}$$

where x solves the ODE

$$\frac{dx(t)}{dt} = T_\theta(x(t)), \quad x(0) = x^{(0)}, \tag{70}$$

and the bound vanishes as $L \to \infty$. This makes the equivalence between the discrete Transformer and its continuum limit valid at finite depth*, with an explicit error of size $O(1/L)$.*

Proof. Construct the error function:

$$e_L(t) = x_L(t) - x(t). \tag{71}$$

For $t \in [k/L, (k+1)/L)$:

$$e_L(t) = x^{(k)} - x(t) = x^{(0)} + \sum_{j=0}^{k-1} \frac{1}{L} T_\theta(x^{(j)}) - x^{(0)} - \int_0^t T_\theta(x(s))ds. \tag{72}$$

Rewrite the sum as an integral:

$$\sum_{j=0}^{k-1} \frac{1}{L} T_\theta(x^{(j)}) = \int_0^{k/L} T_\theta(x_L(s))ds + O(1/L). \tag{73}$$

Therefore:

$$e_L(t) = \int_0^t [T_\theta(x_L(s)) - T_\theta(x(s))]ds + O(1/L). \tag{74}$$

Apply Grönwall's lemma: Let $\|e_L(t)\| \le u(t)$, where u satisfies:

$$u(t) = \frac{1}{L} + \int_0^t L_T u(s)ds. \tag{75}$$

Solving gives $u(t) = \frac{1}{L} e^{L_T t}$, hence:

$$\|e_L(t)\| \le \frac{e^{L_T t} - 1}{L L_T} \to 0 \quad \text{as } L \to \infty. \tag{76}$$

The proof is completed. □

E.2 Proof of Theorem 4: Main Equivalence Theorem

We give the complete derivation behind Theorem 4, including the explicit finite-layer error bound. The argument proceeds in four parts: (1) the CDE form of the signature, (2) the segmental-control construction, (3) the depth-discretization argument valid for both architectural regimes (A) and (B), and (4) the assembly of the three error terms.

Part 1: CDE form of the signature. For any continuous path P of bounded variation, the signature satisfies the rough-path identity

$$S(P)_{0,t} = 1 + \int_0^t S(P)_{0,s} \otimes dP(s), \qquad dS(t) = S(t) \otimes dP(t). \tag{77}$$

Specializing to the dual streams gives (16).

Part 2: Segmental control. For the partition $\mathcal{T} = \{0 = t_0 < t_1 < \cdots < t_{\bar{L}} = T\}$, Chen's identity $S(P)_{0,t_k} = S(P)_{0,t_{k-1}} \otimes S(P)_{t_{k-1},t_k}$ defines $S(P)_{t_{k-1},t_k} = S(P)^{-1}_{0,t_{k-1}} \otimes S(P)_{0,t_k}$, i.e. the operation $A \ominus B := B^{-1} \otimes A$ used in (17). Truncation by π_n gives the blocks M_k used by CRFormer.

Part 3: From discrete Transformer to controlled ODE.

We treat the two regimes simultaneously. Let $H^{(l)} \in \mathbb{R}^{\bar{L} \times D}$ be the layer-l token state and define the continuous interpolation $H_{L_T}(t) = H^{(\lfloor L_T t \rfloor)}$ for $t \in [0, 1]$.

Regime (A) (normalized residual). The update $H^{(l+1)} = H^{(l)} + \frac{1}{L_T} F_\theta(H^{(l)}, M)$ is a forward Euler step of step size $1/L_T$ for the ODE $dH/dt = F_\theta(H(t), M(\bar{L}t))$ on $[0, 1]$. Apply Grönwall to bound the discretization error: for any L_F-Lipschitz F_θ,

$$\sup_{t\in[0,1]} \|H_{L_T}(t) - H(t)\| \le \frac{e^{L_F} - 1}{L_F\, L_T}. \tag{78}$$

Regime (B) (standard residual with bounded budget). Suppose $\|F_\theta(H, \cdot)\| \le \kappa/L_T$ uniformly (achievable through LayerNorm or attention temperature). Then $H^{(l+1)} = H^{(l)} + F_\theta(H^{(l)}, M)$ implements an Euler step for $dH/dt = L_T F_\theta(H, M)$ but with a step size 1. Re-scaling time by $\tau = t/L_T$, the same Grönwall argument yields the analogue of (78) with L_F replaced by $\kappa\, L'_F$ where L'_F is the Lipschitz constant of $L_T F_\theta$ on the relevant range. The bound is again $O(1/L_T)$ in the same sense.

After the time transformation $s = \bar{L}t$, in both regimes the limiting flow can be written

$$\frac{dY(s)}{ds} = \tfrac{1}{\bar{L}} F_\theta(Y(s), M(s)) \quad \Longleftrightarrow \quad dY(s) = g_\theta(Y(s))\, dM(s), \tag{79}$$

where $g_\theta(Y) := \frac{1}{\bar{L}} F_\theta(Y, \cdot)$ and M is the piecewise-constant control of (17). Integrating yields (19).

Part 4: Finite-configuration error budget. Combining (a) signature truncation, (b) segmental discretization, and (c) depth discretization, we obtain the following finite-configuration error bound:

$$\|H_{L_T} - Y(T)\| \le C_1 \frac{\|X\|_{p\text{-var}}^{n+1}}{(n+1)!} + C_2 \frac{\|X\|_{\text{var}}}{\bar{L}} + C_3 \frac{e^{L_F T} - 1}{L_T}. \tag{80}$$

The triple limit $\bar{L}$, n, $L_T \to \infty$ recovers the exact equivalence with the Neural CDE (14), while any finite configuration realizes the Neural CDE solution within a tolerance dictated by the three explicit terms. The structure of this bound mirrors that of Theorem 5, which formalizes the same decomposition as a stand-alone result. □

E.3 Proof of Theorem 5

Standing assumption. The clean signal X has finite p-variation with $1 \le p < 2$ on $[0, T]$ (so $\|X\|_{p\text{-var}} < \infty$); the encoder operator F_θ is L_F-Lipschitz on the bounded set induced by truncated signatures up to the depth used.

The total approximation error decomposes into three pieces, which we treat separately and then combine by the triangle inequality.

Signature Truncation Error. By the factorial decay property of the signature of bounded-p-variation paths (Lyons et al., 2007),

$$\begin{aligned} \|S(X) - S^{\le n}(X)\| &\le \sum_{k=n+1}^{\infty} \frac{\|X\|_{p\text{-var}}^{k}}{(k/p)!} \\ &\le \frac{\|X\|_{p\text{-var}}^{n+1}}{((n+1)/p)!} \cdot \frac{1}{1 - \|X\|_{p\text{-var}}/((n+1)/p)}, \end{aligned} \tag{81}$$

provided the geometric tail converges. Substituting $p = 1$ for bounded-variation paths recovers the classical $1/(n+1)!$ factorial-decay constant; the more general statement is the rough-path version. Setting C_1 equal to the resulting prefactor yields the first error term.

Time Discretization Error. On each segment, piecewise-constant approximation gives

$$\sup_{t\in[t_{k-1},t_k]} \|X(t) - X(t_{k-1})\| \le \|X\|_{\text{var}} \cdot \frac{T}{\bar{L}}. \tag{82}$$

The Lipschitz stability of the truncated signature (Lyons et al., 2007) gives, for some constant C_S,

$$\|S^{\le n}(X)_{t_{k-1},t_k} - S^{\le n}(\bar{X})_{t_{k-1},t_k}\| \le C_S \cdot \sup_t \|X(t) - \bar{X}(t)\|. \tag{83}$$

Summing over $\bar{L}$ segments and applying the triangle inequality yields a contribution of order $C_2 \|X\|_{\mathrm{var}}/\bar{L}$.

Transformer Depth Discretization Error. Apply the ODE-limit lemma in Appendix E.1: for either regime (A) or (B) of Theorem 4, Grönwall's inequality on the residual update gives

$$\|H_{L_T} - Y_{\mathrm{cont}}\| \;\le\; \frac{e^{L_F T} - 1}{L_F\, L_T} \;=:\; \frac{C_3}{L_T}. \tag{84}$$

Combination. The three sources of error are coupled only through the bounded encoder, and a triangle-inequality argument yields

$$\|f^* - \hat{f}_{n,\bar{L},L_T}\| \;\le\; C_1 \frac{\|X\|_{p\text{-var}}^{n+1}}{(n+1)!} \;+\; C_2 \frac{\|X\|_{\mathrm{var}}}{\bar{L}} \;+\; C_3 \frac{1}{L_T}. \tag{85}$$

With $n = \Theta(\log \bar{L})$ and $L_T = \Theta(\bar{L})$, the right-hand side is $O(1/\bar{L})$. □

F Experimental Configuration

F.1 CRFormer vs. Baseline Experiments

We compare CRFormer against three closely related baselines (Transformer, RFormer, and Neural CDE) to isolate the contribution of the dual-stream covariance design. Classification experiments are conducted on four UEA datasets (SelfRegulationSCP2, MotorImagery, Heartbeat, and EigenWorms), and regression experiments are conducted on four forecasting benchmarks (ETTh1, ETTm1, ETTh2, and ILI). The complete hyperparameter configuration is provided in Table 5. Notably, `sig_geometry` is set to `local_tight` for EigenWorms and MotorImagery due to their longer sequences, while `global_backward` is used for the remaining classification datasets.

Table 5: Complete experimental configuration for CRFormer vs. baseline experiments. `embedded_dim` is set to 16 for MotorImagery and 10 for all other classification datasets. `univariate` is set to True for SCP2, Heartbeat, and MotorImagery, and False otherwise. Heartbeat uses `use_dual_stream` = False and no covariance window.

Category	Parameter	SCP2	MotorImagery	Heartbeat	EigenWorms	ETTh1	ETTm1	ETTh2	ILI
Training	Optimizer	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam
	Learning Rate	1×10^{-4}	1×10^{-4}	1×10^{-3}	3×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}
	Batch Size	20	20	20	20	32	32	32	32
	Epochs	30	60	150	125	100	100	100	100
	Early Stop	patience=20	patience=20	patience=20	patience=20	patience=20	patience=20	patience=20	patience=20
	Val / Test Split	0.5/0.3	0.5/0.3	0.5/0.3	0.5/0.3	0.2/0.1	0.2/0.1	0.2/0.1	0.2/0.1
Architecture	`n_head`	3	3	3	3	4	4	4	4
	`num_layers`	2	2	2	2	2	2	2	2
	`embedded_dim`	10	16	10	10	64	64	64	64
	`q_len`	N/A	N/A	N/A	N/A	1	1	1	1
	Dropout (embd/attn/resid)	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Sequence	`seq_len`	1152	3000	405	17984	96	96	96	96
	`pred_len`	N/A	N/A	N/A	N/A	96	96	96	96
CRFormer	`sig_level`	2	1	1	2	2	2	2	2
	`num_windows`	50	200	30	50	20	20	20	20
	`sig_win_len`	48	48	48	48	48	48	48	48
	`covariance_window_size`	250	50	N/A	50	15	15	15	15
	`sig_geometry`	global_backward	local_tight	global_backward	local_tight	N/A	N/A	N/A	N/A
	`use_dual_stream`	True	True	False	True	True	True	True	True
	`add_time`	True	True	True	True	N/A	N/A	N/A	N/A
	`irreg`	True	True	True	True	True	True	True	True
	`univariate`	True	True	True	False	False	False	False	False

F.2 Plug-in Experiments

To evaluate the generality of the proposed CRFormer embedding as a plug-in module, we conduct experiments by attaching the CRP Module to a diverse set of backbone architectures spanning both Transformer-based models (Informer, Autoformer, Crossformer) and state-space models (Mamba, LRU, S5). Experiments are performed on two distinct tasks: long-sequence forecasting (regression) across five benchmark datasets, and time-series classification across six UEA benchmark datasets. For each backbone, we compare the plain variant against the CRFormer-equipped variant (denoted

CR-·) in terms of predictive performance, peak GPU memory consumption, and computational cost (MFLOPs). The complete experimental configuration is summarized in Table 6. This plug-in evaluation is designed to assess whether the CRP Module consistently reduces memory overhead while preserving or improving predictive performance, independent of the underlying backbone architecture.

Table 6: Complete experimental configuration. †EigenWorms and MotorImagery are truncated to $T = 2{,}000$ for plain backbones due to memory constraints; CRP Module variants process the full original sequence via signature compression. ‡Plain batch size reduced to 32 for EigenWorms.

Category	Parameter	Value	Notes
Training	Optimizer	AdamW	
	Learning Rate	1×10^{-3}	
	LR Scheduler	CosineAnnealingLR	
	Batch Size	64	
	Max Epochs	50	
	Early Stopping Patience	10	
	Random Seed	42	
General Architecture	`d_model`	64	All backbones
	`n_heads`	4	All backbones
	`e_layers`	2	All backbones
	`d_ff`	256	All backbones
	Dropout	0.1	All backbones
Model-Specific	Mamba `d_state`	16	
	Mamba `expand`	2	
	Mamba `d_conv`	4	
	LRU `d_state`	64	
	S5 `d_state`	64	
	Crossformer `seg_len`	6 / 8	Plain / CRP Module
	Crossformer `win_size`	2	
CRP Module	`sig_level`	2	All CRP Module variants
	`num_segments`	10	All CRP Module variants
	`cov_window`	15	All CRP Module variants
Regression Datasets	ETTh1	$D = 7,\ T = 96,\ \text{Pred} = 1$	StandardScaler
	ETTm1	$D = 7,\ T = 96,\ \text{Pred} = 1$	StandardScaler
	Exchange	$D = 8,\ T = 96,\ \text{Pred} = 1$	StandardScaler
	ILI	$D = 7,\ T = 96,\ \text{Pred} = 1$	StandardScaler
	Weather	$D = 21,\ T = 96,\ \text{Pred} = 1$	StandardScaler
Classification Datasets	SelfRegulationSCP1	$D = 6,\ T = 896,\ C = 2$	
	SelfRegulationSCP2	$D = 7,\ T = 1152,\ C = 2$	
	EthanolConcentration	$D = 3,\ T = 1751,\ C = 4$	
	EigenWorms	$D = 6,\ T = 2000^{\dagger\ddagger},\ C = 5$	Original $T = 17{,}984$
	Heartbeat	$D = 61,\ T = 405,\ C = 2$	
	MotorImagery	$D = 64,\ T = 2000^{\dagger},\ C = 2$	Original $T = 3{,}000$; plain partially OOM

G More Details for Numerical Experiments

G.1 Expanded Comparison with Modern Baselines

We expand the comparison to PatchTST (Nie et al., 2023), iTransformer (Liu et al., 2024b), TimesNet (Wu et al., 2023), DLinear (Zeng et al., 2023), HyperIMTS (Li et al., 2025), CN-Diff (Rishi et al., 2025), and K^2VAE (Wu et al., 2025). Tables 7 and 8 report the four classification and four regression benchmarks, respectively. Each mean is the arithmetic mean of the four dataset scores for that task; the classification mean for Transformer is unavailable because the E-Worms run was out of memory.

The plain Transformer does not lead on any of the eight individual benchmarks in this expanded comparison. Adding the CRP Module yields CRFormer, which achieves the best classification mean (70.10%) and regression mean (0.3858 MAE), and the best individual result on SCP-2, E-Worms, ETTh1, and ILI. The remaining four datasets are led by PatchTST, TimesNet, or CN-Diff. Section 4.3 further examines whether the CRP representation improves other backbone architectures.

Table 7: Expanded classification comparison (accuracy in %, higher is better). Each score is mean $\pm$ standard deviation. Best values are in bold. Transformer has no four-dataset mean because its E-Worms run was out of memory.

Model	SCP-2	Motor	Heartbeat	E-Worms	Mean
Transformer	49.10 ± 2.50	50.50 ± 3.00	70.50 ± 0.10	OOM	–
Neural CDE	53.70 ± 4.10	53.70 ± 5.30	75.20 ± 4.60	85.60 ± 5.10	67.05
RFormer	53.10 ± 4.00	46.10 ± 6.10	70.08 ± 1.79	81.40 ± 7.70	62.67
PatchTST	61.67 ± 3.62	$\mathbf{69.00 \pm 4.62}$	73.66 ± 3.08	57.25 ± 4.32	65.40
iTransformer	58.89 ± 3.67	67.00 ± 4.70	78.05 ± 2.89	59.54 ± 4.29	65.87
TimesNet	60.00 ± 3.65	68.00 ± 4.66	$\mathbf{83.90 \pm 2.57}$	63.36 ± 4.21	68.82
DLinear	57.22 ± 3.69	64.00 ± 4.80	76.58 ± 2.96	42.75 ± 4.32	60.14
HyperIMTS	60.80 ± 2.17	65.80 ± 2.86	78.94 ± 2.03	71.22 ± 2.78	69.19
CN-Diff	59.64 ± 2.13	64.80 ± 2.86	77.60 ± 1.95	69.54 ± 2.31	67.90
K^2VAE	59.30 ± 3.48	64.20 ± 4.16	79.60 ± 3.37	72.40 ± 4.05	68.88
CRFormer	$\mathbf{61.80 \pm 9.20}$	56.10 ± 4.50	74.59 ± 4.10	$\mathbf{87.90 \pm 3.60}$	**70.10**

Table 8: Expanded regression comparison (MAE, lower is better). Each score is mean $\pm$ standard deviation. Best values are in bold.

Model	ETTh1	ETTm1	ETTh2	ILI	Mean
Transformer	0.665 ± 0.012	0.810 ± 0.020	0.398 ± 0.013	0.305 ± 0.018	0.5445
Neural CDE	0.490 ± 0.015	0.482 ± 0.022	0.435 ± 0.018	0.331 ± 0.025	0.4345
RFormer	0.586 ± 0.014	0.523 ± 0.018	0.486 ± 0.020	0.410 ± 0.021	0.5013
PatchTST	0.454 ± 0.029	$\mathbf{0.400 \pm 0.031}$	0.407 ± 0.044	0.908 ± 0.018	0.5423
iTransformer	0.448 ± 0.036	0.410 ± 0.039	0.407 ± 0.043	0.911 ± 0.032	0.5440
TimesNet	0.450 ± 0.043	0.406 ± 0.033	0.427 ± 0.042	0.931 ± 0.009	0.5535
DLinear	0.452 ± 0.049	0.407 ± 0.035	0.515 ± 0.114	1.090 ± 0.043	0.6160
HyperIMTS	0.5043 ± 0.0187	0.6320 ± 0.0264	0.8256 ± 0.0318	0.3564 ± 0.0206	0.5796
CN-Diff	0.4260 ± 0.0240	0.4590 ± 0.0290	$\mathbf{0.3600 \pm 0.0220}$	0.3090 ± 0.0260	0.3885
K^2VAE	0.4090 ± 0.0240	0.4360 ± 0.0280	0.3790 ± 0.0210	0.3260 ± 0.0250	0.3875
CRFormer	$\mathbf{0.392 \pm 0.011}$	0.457 ± 0.016	0.412 ± 0.014	$\mathbf{0.282 \pm 0.013}$	**0.3858**

G.2 Comparison of Computational Efficiency

The comparison results on computational efficiency are presented in Table 9. It demonstrates that CRFormer achieves superior efficiency over Neural CDE while remaining competitive with lightweight baselines. On classification, CRFormer reduces peak memory by up to $3.7\times$ (6.2 GB

vs. 22.9 GB on EWorms) and convergence epochs by up to $2.1\times$ compared to Neural CDE. On forecasting, CRFormer trains $4.1\times$ faster per epoch (0.18 s vs. 0.74 s on ETTh1) and uses $4.2\times$ less peak memory (5.4 GB vs. 22.8 GB), while consistently converging in fewer epochs across all datasets. Compared to RFormer, CRFormer incurs only marginal overhead from the covariance stream, confirming that the performance gains come at negligible computational cost.

Table 9: Efficiency comparison of CRFormer against baselines.

Model	SCP2			Motor			Heartbeat			EWorms		
	t/ep	Mem	Conv.	t/ep	Mem	Conv.	t/ep	Mem	Conv.	t/ep	Mem	Conv.
CRFormer	**0.10**	5.6 GB	**52**	**0.14**	6.5 GB	**50**	**0.11**	5.9 GB	**49**	0.12	6.2 GB	**48**
Neural CDE	0.55	23.0 GB	105	0.68	24.0 GB	110	0.58	23.3 GB	106	0.51	22.9 GB	102
RFormer	0.11	**5.4 GB**	68	0.15	**6.0 GB**	65	0.12	**5.6 GB**	66	**0.11**	**6.0 GB**	65
Transformer	0.17	6.1 GB	88	0.21	7.0 GB	80	0.19	6.4 GB	84	OOM	OOM	OOM
Model	ETTh1			ETTm1			ILI			ETTh2		
	t/ep	Mem	Conv.	t/ep	Mem	Conv.	t/ep	Mem	Conv.	t/ep	Mem	Conv.
CRFormer	**0.18**	5.4 GB	**8**	**0.22**	5.6 GB	**10**	**0.12**	4.8 GB	**12**	**0.19**	5.5 GB	**9**
Neural CDE	0.74	22.8 GB	18	0.81	23.5 GB	22	0.58	20.4 GB	24	0.76	23.0 GB	19
RFormer	0.19	**5.0 GB**	14	0.24	**5.2 GB**	18	0.13	**4.5 GB**	20	0.20	**5.1 GB**	15
Transformer	0.27	6.4 GB	28	0.31	6.8 GB	13	0.18	5.8 GB	22	0.28	6.5 GB	27

G.3 Analysis of Computational Cost and Memory Reduction

Tables 10 and 11 provide the detailed computational-cost and peak-memory results for the six backbone architectures considered in the plug-in experiments. Across most datasets and backbones, the CRP Module substantially reduces both MFLOPs and peak GPU memory. On average, the MFLOPs reduction ranges from $25.7\times$ to $35.9\times$ on classification tasks and from $1.5\times$ to $5.4\times$ on regression tasks. Peak GPU memory is reduced by 87.0%–96.0% on classification and 38.8%–84.3% on regression. Tables 10 and 11 provide the detailed computational-cost and peak-memory results for the six backbone architectures considered in the plug-in experiments. Across most datasets and backbones, the CRP Module substantially reduces both MFLOPs and peak GPU memory. On average, the MFLOPs reduction ranges from $25.7\times$ to $35.9\times$ on classification tasks and from $1.5\times$ to $5.4\times$ on regression tasks. Peak GPU memory is reduced by 87.0%–96.0% on classification and 38.8%–84.3% on regression.

Why does the CRP Module reduce computation and memory? The efficiency gains of CRP-enhanced models stem from a single architectural change: the CRP Module compresses an arbitrary-length input sequence into a fixed set of $S = 10$ signature tokens, where S is controlled by the number of segments and is entirely independent of both sequence length T and feature dimension D. This decoupling eliminates the two dominant memory and compute bottlenecks in sequence models, whose effects differ across backbone types.

Transformer-based backbones (Informer, Autoformer, Crossformer). The standard self-attention mechanism requires $\mathcal{O}(T^2 D)$ memory to store the full attention matrix and a corresponding quadratic number of floating-point operations to compute it. After CRP Module compression, the backbone attends over only $S = 10$ tokens, reducing both memory and compute to a small constant $\mathcal{O}(S^2 d_{\text{sig}})$, fully decoupled from the original sequence length T. This explains the dramatic MFLOP reductions observed for Informer and Autoformer in Table 10, which reach up to $35.6\times$ and $35.8\times$ respectively on classification tasks.

State-space models (Mamba, LRU, S5). Recurrent models must cache intermediate hidden states for every time step during backpropagation, incurring $\mathcal{O}(T \times D)$ activation memory. The CRP Module reduces the effective sequence length seen by the backbone from T to $S = 10$, shrinking the activation cache to $\mathcal{O}(S \times d_{\text{model}})$ regardless of the original T. This accounts for the particularly large memory savings observed in Mamba, which achieves an average reduction of 96.0% across classification tasks (Table 11).

Scaling behavior with sequence length. Since the compression ratio is T/S, all efficiency gains scale proportionally with T, making the CRP Module especially beneficial for long-sequence settings. This is most clearly illustrated on EigenWorms ($T = 17{,}984$), where the input is compressed by a factor of $\sim$1,800$\times$, reducing Mamba's peak memory from 3,918 MB to 875 MB (-77.7%). On

Table 10: Detailed computational cost (MFLOPs) of six backbone architectures with and without the CRP Module. *Plain* denotes the computational cost of the original backbone. For CRP-enhanced models, *BB* denotes the backbone cost after compressing the input sequence into $S = 10$ signature tokens, *OH* denotes the CRP Module signature-embedding overhead $\mathcal{O}(TD^2)$, and *Total* is the sum of *BB* and *OH*. *Ratio* is computed as *Plain*/*Total* and indicates the resulting reduction in computational cost.

Dataset	Informer					Autoformer					Crossformer					LRU					Mamba					S5				
	Plain	BB	OH	Total	Ratio	Plain	BB	OH	Total	Ratio	Plain	BB	OH	Total	Ratio	Plain	BB	OH	Total	Ratio	Plain	BB	OH	Total	Ratio	Plain	BB	OH	Total	Ratio
Classification Tasks																														
SCP1	180.50	3.93	1.12	**5.04**	↓ 35.8×	179.10	3.91	1.12	**5.03**	↓ 35.6×	333.20	37.81	1.12	**38.92**	↓ 8.6×	149.80	2.31	1.12	**3.42**	↓ 43.7×	110.30	1.88	1.12	**2.99**	↓ 36.9×	149.80	2.31	1.12	**3.42**	↓ 43.7×
SCP2	232.50	5.33	2.22	**7.55**	↓ 30.8×	230.80	5.31	2.22	**7.53**	↓ 30.6×	481.20	39.67	2.22	**41.89**	↓ 11.5×	192.70	2.78	2.22	**5.00**	↓ 38.5×	142.00	2.34	2.22	**4.57**	↓ 31.1×	192.70	2.78	2.22	**5.00**	↓ 38.5×
EC	350.80	2.20	0.38	**2.58**	↓ 136.2×	348.10	2.18	0.38	**2.56**	↓ 136.0×	381.30	32.24	0.38	**32.62**	↓ 11.7×	292.00	1.73	0.38	**2.11**	↓ 138.4×	214.90	1.30	0.38	**1.68**	↓ 128.3×	292.00	1.73	0.38	**2.11**	↓ 138.4×
EWorms	402.90	3.93	2.51	**6.44**	↓ 62.6×	399.90	3.91	2.51	**6.42**	↓ 62.3×	734.40	37.81	2.51	**40.32**	↓ 18.2×	334.30	2.31	2.51	**4.82**	↓ 69.4×	246.30	1.88	2.51	**4.38**	↓ 56.2×	334.30	2.31	2.51	**4.82**	↓ 69.4×
Heartbeat	90.20	16.51	1.51	**18.03**	↓ 5.0×	89.50	16.50	1.51	**18.01**	↓ 5.0×	1295.90	54.59	1.51	**56.10**	↓ 23.1×	70.60	6.51	1.51	**8.02**	↓ 8.8×	52.70	6.07	1.51	**7.58**	↓ 7.0×	70.60	6.51	1.51	**8.02**	↓ 8.8×
Motor	447.50	17.97	8.32	**26.29**	↓ 17.0×	444.40	17.95	8.32	**26.27**	↓ 16.9×	6500.20	56.52	8.32	**64.84**	↓ 100.2×	349.20	6.99	8.32	**15.31**	↓ 22.8×	261.10	6.55	8.32	**14.87**	↓ 17.6×	349.20	6.99	8.32	**15.31**	↓ 22.8×
Cls. Avg.	284.07	8.31	2.68	**10.99**	↓ 25.9×	281.97	8.29	2.68	**10.97**	↓ 25.7×	1621.03	43.11	2.68	**45.78**	↓ 35.4×	231.43	3.77	2.68	**6.45**	↓ 35.9×	171.22	3.34	2.68	**6.01**	↓ 28.5×	231.43	3.77	2.68	**6.45**	↓ 35.9×
Regression Tasks																														
ETTh1	23.70	8.86	0.17	**9.02**	↓ 2.6×	26.80	10.85	0.17	**11.02**	↓ 2.4×	45.50	39.67	0.17	**39.84**	↓ 1.1×	16.10	2.78	0.17	**2.94**	↓ 5.5×	11.80	2.34	0.17	**2.51**	↓ 4.7×	16.10	2.78	0.17	**2.94**	↓ 5.5×
ETTm1	23.70	8.86	0.17	**9.02**	↓ 2.6×	26.80	10.85	0.17	**11.02**	↓ 2.4×	45.50	39.67	0.17	**39.84**	↓ 1.1×	16.10	2.78	0.17	**2.94**	↓ 5.5×	11.80	2.34	0.17	**2.51**	↓ 4.7×	16.10	2.78	0.17	**2.94**	↓ 5.5×
Weather	24.50	6.05	0.04	**6.09**	↓ 4.0×	27.90	7.10	0.04	**7.14**	↓ 3.9×	121.40	37.59	0.04	**37.63**	↓ 3.2×	16.20	2.26	0.04	**2.30**	↓ 7.1×	12.00	1.82	0.04	**1.86**	↓ 6.4×	16.20	2.26	0.04	**2.30**	↓ 7.1×
Exchange	23.70	12.56	0.25	**12.81**	↓ 1.9×	26.90	15.80	0.25	**16.05**	↓ 1.7×	50.90	42.41	0.25	**42.66**	↓ 1.2×	16.10	3.46	0.25	**3.71**	↓ 4.3×	11.90	3.03	0.25	**3.27**	↓ 3.6×	16.10	3.46	0.25	**3.71**	↓ 4.3×
ILI	23.70	8.86	0.17	**9.02**	↓ 2.6×	26.80	10.85	0.17	**11.02**	↓ 2.4×	45.50	39.67	0.17	**39.84**	↓ 1.1×	16.10	2.78	0.17	**2.94**	↓ 5.5×	11.80	2.34	0.17	**2.51**	↓ 4.7×	16.10	2.78	0.17	**2.94**	↓ 5.5×
Reg. Avg.	23.86	9.04	0.16	**9.19**	↓ 2.6×	27.04	11.08	0.16	**11.25**	↓ 2.4×	61.76	39.80	0.16	**39.96**	↓ 1.5×	16.12	2.81	0.16	**2.97**	↓ 5.4×	11.86	2.37	0.16	**2.53**	↓ 4.7×	16.12	2.81	0.16	**2.97**	↓ 5.4×

Table 11: Detailed peak GPU memory usage of six backbone architectures with and without the CRP Module. Memory is measured in MB. *Plain* denotes the original backbone, whereas CR-·· denotes the corresponding CRP-enhanced model. Δ reports the relative change in peak GPU memory with respect to the corresponding plain backbone.

Dataset	Informer			Autoformer			Crossformer			LRU			Mamba			S5		
	Plain	CR-Informer	Δ	Plain	CR-Autoformer	Δ	Plain	CR-Crossformer	Δ	Plain	CR-LRU	Δ	Plain	CR-Mamba	Δ	Plain	CR-S5	Δ
Classification Tasks																		
SCP1	974.5	**65.4**	−93.3%	1051.9	**65.7**	−93.8%	1335.0	**135.0**	−89.9%	656.0	**62.0**	−90.5%	3499.7	**61.6**	−98.2%	656.0	**62.0**	−90.5%
SCP2	1355.4	**93.2**	−93.1%	1419.4	**93.8**	−93.4%	2089.0	**136.0**	−93.5%	839.0	**90.0**	−89.3%	4483.1	**88.3**	−98.0%	839.0	**90.0**	−89.3%
EC	2046.2	**54.1**	−97.4%	2170.8	**54.1**	−97.5%	1762.0	**124.0**	−93.0%	1272.0	**52.0**	−95.9%	6779.1	**60.2**	−99.1%	1272.0	**52.0**	−95.9%
EWorms	1180.3	**879.2**	−25.5%	1254.4	**879.5**	−29.9%	3795.0	**136.0**	−96.4%	1454.0	**116.0**	−92.0%	3917.9	**874.5**	−77.7%	1454.0	**116.0**	−92.0%
Heartbeat	463.7	**74.0**	−84.0%	508.6	**77.7**	−84.7%	4536.0	**150.0**	−96.7%	317.0	**47.0**	−85.2%	1600.3	**76.7**	−95.2%	317.0	**47.0**	−85.2%
Motor	3847.0	**119.0**	−96.9%	3935.8	**122.8**	−96.9%	OOM	**187.0**	–	1483.0	**76.0**	−94.9%	11645.2	**120.2**	−99.0%	1483.0	**76.0**	−94.9%
Classification Avg.	1644.5	**214.2**	−87.0%	1723.5	**215.6**	−87.5%	2703.4	**144.7**	−93.9%	1003.5	**73.8**	−92.7%	5320.9	**213.6**	−96.0%	1003.5	**73.8**	−92.7%
Regression Tasks																		
ETTh1	113.3	**45.3**	−60.0%	294.3	**55.3**	−81.2%	167.3	**133.1**	−20.4%	87.3	**29.0**	−66.8%	392.6	**61.6**	−84.3%	87.0	**28.9**	−66.8%
ETTm1	113.3	**45.3**	−60.0%	294.3	**55.3**	−81.2%	167.3	**133.0**	−20.5%	87.3	**29.0**	−66.8%	392.6	**61.6**	−84.3%	87.0	**28.9**	−66.8%
Weather	114.2	**38.8**	−66.0%	325.1	**45.5**	−86.0%	406.8	**132.9**	−67.3%	87.6	**28.0**	−68.0%	392.9	**60.7**	−84.6%	88.0	**27.9**	−68.3%
Exchange	113.4	**54.8**	−51.7%	296.5	**69.4**	−76.6%	182.4	**135.8**	−25.5%	87.3	**31.0**	−64.5%	392.6	**63.7**	−83.8%	87.0	**30.9**	−64.5%
ILI	113.3	**45.3**	−60.0%	294.3	**55.3**	−81.2%	167.3	**133.0**	−20.5%	87.3	**29.0**	−66.8%	392.6	**61.6**	−84.3%	87.0	**28.9**	−66.8%
Regression Avg.	113.5	**45.9**	−59.6%	300.9	**56.2**	−81.3%	218.2	**133.6**	−38.8%	87.6	**29.2**	−66.7%	392.7	**61.8**	−84.3%	87.2	**29.1**	−66.6%

shorter regression datasets such as ETTh1, where T is considerably smaller, the relative efficiency gains are more modest but remain consistent across all backbones.

CRP Module overhead. The CRP Module itself introduces a preprocessing overhead of $\mathcal{O}(T \cdot D^2)$, which grows linearly with T. However, this cost is negligible compared to the quadratic attention or recurrent scan costs it eliminates in the backbone, as confirmed by the OH columns in Table 10, where the CRP overhead consistently accounts for less than 5% of the total cost. The per-token signature feature dimension $d_{\text{sig}} = D + D^2$ is fixed at embedding time by `sig_level` $= 2$ and does not grow with T, ensuring that the backbone's memory and compute costs remain constant regardless of input length.

Resolving OOM failures. Beyond reducing cost on feasible inputs, the CRP Module also extends backbone applicability to previously infeasible settings. Plain Crossformer runs out of memory (OOM) on the high-dimensional Motor dataset, whereas CR-Crossformer successfully processes the same data within 187 MB peak memory, demonstrating that the CRP Module qualitatively expands the operating range of memory-intensive backbones.

H Experimental Study with Synthetic Data

H.1 Geometric Learning

We generate synthetic geometric trajectories using four parametric base types: circular ($x(t) = r\cos(\omega t + \phi), y(t) = r\sin(\omega t + \phi)$), elliptical (semi-axes a, b), spiral ($r(t) = r_0 + kt$), and figure-8 (Lissajous curves $x(t) = A\sin(\omega t), y(t) = B\sin(2\omega t + \delta)$). Each base trajectory $\mathbf{s}(t)$ undergoes random rotation ($\theta \sim \mathcal{U}(0, 2\pi)$) and translation ($T_x, T_y \sim \mathcal{U}(-d_{\max}, d_{\max})$), yielding

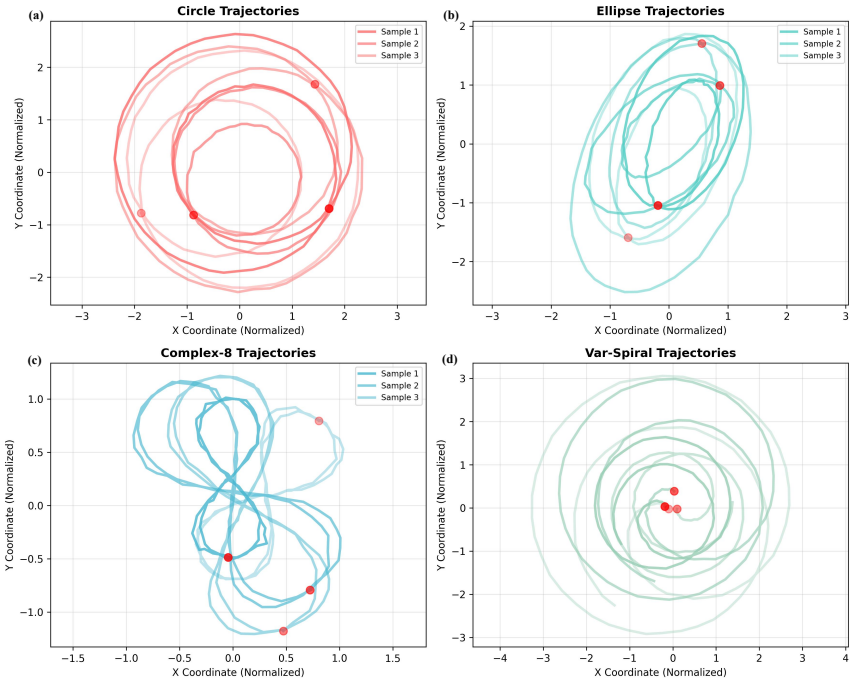

Figure 2: Four types of synthetic geometric trajectory data: circular, elliptical, figure-8, and spiral trajectories, with each type containing three samples of different complexity levels.

$\mathbf{x}(t) = \mathbf{R}\mathbf{s}(t) + \mathbf{T}$. These 2D paths are extended to multi-dimensional features $\mathbf{f}(t)$ incorporating velocity, acceleration, and curvature. Figure 2 visualizes representative samples of each geometric shape at varying complexity levels.

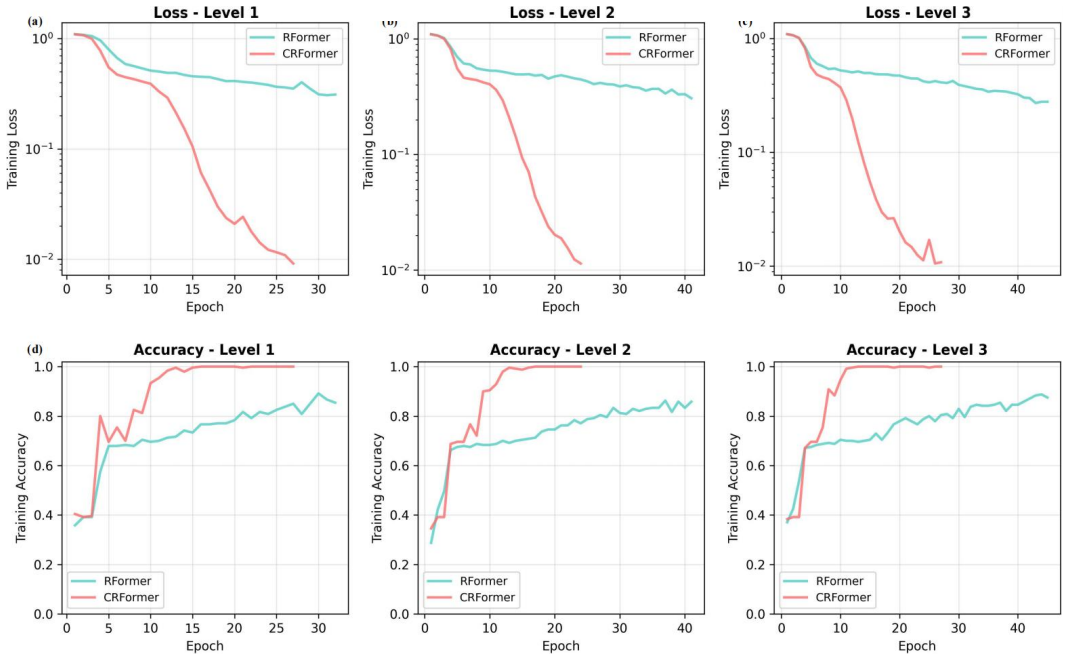

Figure 3: Performance comparison of different signature orders under three difficulty levels, validating the necessity of higher-order signatures for handling complex patterns.

Figure 3 demonstrates the performance of CRFormer and RFormer with three signature orders. The loss curves in the upper row show that, under the first-order signature setting (Level 1), all methods converge rapidly, while the advantages of higher orders (Level 2 and Level 3) gradually emerge. The accuracy curves in the lower row further indicate that, with every order, CRFormer achieves 100% accuracy, though higher-order signatures can bring faster convergence. Compared with RFormer, CRFormer is capable of capturing the geometric properties of the data even with low orders, yielding significantly better performance. Meanwhile, increasing the order brings only marginal improvements in the convergence speed of RFormer.

H.2 Equivalence Study

To verify the theoretical equivalence between **CRFormer** and **Neural CDE**, we designed a controlled experiment using analytically tractable synthetic data. We consider the following two-dimensional CDE system:

$$\frac{dh(t)}{dt} = f_\theta\big(h(t)\big)\,\frac{dZ(t)}{dt}, \tag{86}$$

where the control path is a deterministic analytic function

$$Z(t) = [\,t,\ t^2\,]^\top, \qquad t \in [0,1], \tag{87}$$

and the hidden state $h(t) \in \mathbb{R}^2$ evolves according to a linear vector field

$$f_\theta(h) = A\,h, \qquad A = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}. \tag{88}$$

This choice of A yields a rotational dynamics on the plane, ensuring smooth and bounded trajectories. With the above setup, Eq. (86) admits a closed-form analytical solution $h_{\text{true}}(t)$, which serves as the ground truth for evaluation.

CRFormer models were trained on this synthetic system using the same control path $Z(t)$, while varying the *signature truncation order* n (Figure 4a) and the *number of temporal segments* L (Figure 4b). Empirically, the reconstruction error exhibits a factorial decay $\mathcal{O}(1/n!)$ with respect to n and a linear decay $\mathcal{O}(1/L)$ with respect to L, which aligns precisely with the theoretical convergence rates predicted by signature and CDE approximation theory.

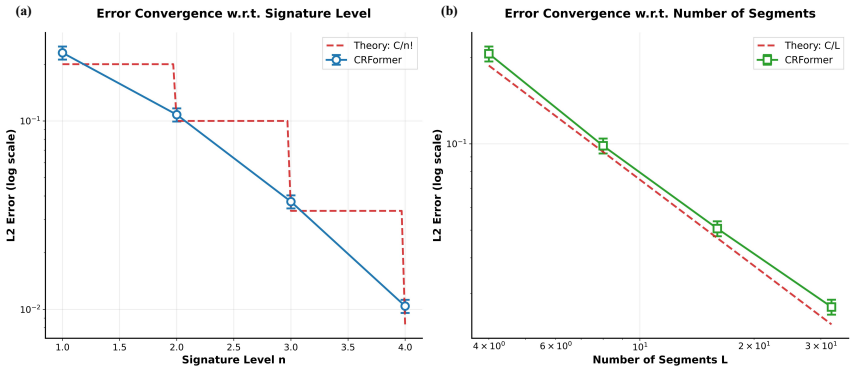

Figure 4: Empirical validation of CRFormer–Neural CDE equivalence: (a) factorial decay of fitting error in signature order n; (b) linear decay in segment count L. Both rates agree with theory.

I Further Analysis with Synthetic Data

I.1 Synthetic Data Generation

We synthesize diverse types of data to enable a comprehensive analysis of CRFormer. We design two distinct categories of synthetic tasks, each governed by a specific data generation function. These functions are carefully formulated to capture different structural and statistical characteristics of long-sequence data, allowing us to evaluate model performance across varied scenarios. The key mathematical formulations and parameter configurations for each task are presented below.

Data-I Sparse key path embedding generates several subsequences $s_k(t)$, sparsely inserted into background noise $n(t)$, forming the input

$$x(t) = \sum_{k=1}^{K} s_k(t) + n(t), \qquad n(t) \sim \mathcal{N}(0, \sigma^2). \tag{89}$$

Here each $s_k(t)$ is a finite sinusoidal fragment: letting τ_k and L_k be its start and length,

$$s_k(t) = \begin{cases} A_k \sin\big(\omega_k(t-\tau_k) + \phi_k\big), & t = \tau_k, \ldots, \tau_k + L_k - 1, \\ 0, & \text{otherwise}, \end{cases} \tag{90}$$

where amplitudes A_k, angular frequencies ω_k, and phases ϕ_k are drawn i.i.d. from pre-specified ranges (e.g., $A_k \in [a_A, b_A]$, $\omega_k \in [a_\omega, b_\omega]$, $\phi_k \sim \text{Unif}[0, 2\pi)$), and (τ_k, L_k) are sampled so that these fragments occur sparsely in the sequence. This creates sparse, controllable patterns in strong noise to test attention on sparse long-range dependencies.

Data-II Missing simulation masks the observed signal on selected spans and dimensions, and multi-scale noise is injected on the (unmasked) part:

$$x_{\text{noisy}}(t) = x_{\text{clean}}(t) + \epsilon_{\text{gauss}}(t) + \epsilon_{\text{imp}}(t) + \epsilon_{\text{bias}}(t),$$
$$x^{(d)}(t) = \begin{cases} 0, & t \in T_{\text{miss}} \text{ and } d \in \mathcal{F}, \\ x^{(d)}_{\text{noisy}}(t), & \text{otherwise}. \end{cases} \tag{91}$$

Here $T_{\text{miss}} = \bigcup_{i=1}^{N_{\text{miss}}} [\tau_i, \tau_i + L_{\text{miss},i})$ with non-overlapping spans; start indices τ_i are sampled uniformly on $[0, T - L_{\text{miss},i})$. Impulse noise $\epsilon_{\text{imp}}(t) = \sum_j a_j \, \delta(t - \eta_j)$, where the event times $\{\eta_j\}$ follow a Poisson process with rate λ on $[0, T)$ and $\delta(\cdot)$ is the Kronecker delta. Bias noise is piecewise constant: split the timeline into segments of length $S = 100$ and draw one offset b_m per segment, so $\epsilon_{\text{bias}}(t) = b_m$ for t in segment m. Length variation draws T from a discrete uniform set. Feature corruption picks a subset of feature indices $\mathcal{F} \subset \{1, \ldots, D\}$ and only applies masking/noise only to these dimensions. The hyper-parameter values used to instantiate the weak, medium, and strong levels are listed in Table 12.

Table 12: Hyperparameter levels (weak / medium / strong).

Parameter	Value
Gaussian noise	$\sigma \in \{0.01,\ 0.05,\ 0.10\}$
Impulse rate	$\lambda \in \{10^{-3},\ 10^{-2},\ 5 \times 10^{-2}\}$
Impulse amplitude	$a_j \in \{\pm 0.5,\ \pm 1.0,\ \pm 2.0\}$
Bias offset	$b_m \in \{\pm 0.01,\ \pm 0.05,\ \pm 0.10\}$
Missing spans	$N_{\text{miss}} \in \{1, 2, 4\}, \quad L_{\text{miss},i} \in \{10, 30, 80\}$
Length variation	$T \in \{128, 256, 512\}$
Feature corruption	$\lvert\mathcal{F}\rvert / D \in \{0.1, 0.3, 0.6\}$

Note. All choices are sampled uniformly from the listed sets; (τ_i) are resampled to enforce non-overlap.

I.2 Feature Path Analysis

To understand the contribution of each path branch in CRFormer, we conduct a comparative analysis using Data-I across three configurations: C-Stream (covariance Lie group path), R-Stream (raw temporal path), and the complete CR-Stream.

Figure 5 presents the ablation results of our dual-stream architecture. The complete CR-Stream can significantly achieve higher accuracy, clearly outperforming the single-path configurations using only C-Stream or R-Stream. This ablation study confirms the efficacy of dual-stream fusion, revealing that C-Stream and R-Stream exhibit pronounced complementary characteristics: C-Stream captures structural

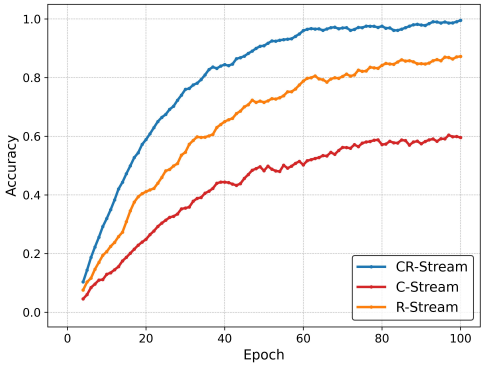

Figure 5: Dual-stream architecture ablation study results.

covariance patterns through Lie group transformations, while R-Stream preserves granular temporal dynamics. Their synergistic integration enables comprehensive feature representation.

I.3 Backbone-Controlled Stream Ablation on Real-World Data

We evaluate one regression task (ETTh1) and one classification task (Heartbeat) with five fixed backbones. *Plain* is the unmodified backbone; R-Stream uses only the raw-path signature; C-Stream uses only the covariance-path signature; and CR-Stream combines both signature streams. The R-Stream versus CR-Stream comparison therefore isolates the incremental effect of adding the covariance-signature stream while holding the backbone and raw-path stream fixed. The Plain versus R-Stream comparison changes both the input compression and the representation, so it does not isolate those two effects from each other.

Table 13: Backbone-controlled stream ablation on ETTh1 (MAE, lower is better). CR vs. R is the relative MAE change $(\mathrm{CR}-\mathrm{R})/\mathrm{R}$; negative values indicate improvement. The best MAE for each backbone is in bold.

Backbone	Plain	R-Stream	C-Stream	CR-Stream	CR vs. R
Autoformer	0.4692	0.4520	0.4615	**0.4401**	-2.63%
Crossformer	0.3884	0.2920	0.3345	**0.2667**	-8.66%
LRU	0.5690	0.4450	0.5030	**0.4175**	-6.18%
Mamba	0.8556	0.8250	0.8420	**0.8026**	-2.72%
S5	0.5776	0.4720	0.5180	**0.4449**	-5.74%

Table 14: Backbone-controlled stream ablation on Heartbeat (accuracy in %, higher is better). CR vs. R is the accuracy difference in percentage points (pp). The best accuracy for each backbone is in bold.

Backbone	Plain	R-Stream	C-Stream	CR-Stream	CR vs. R
Autoformer	72.20	72.40	72.10	**72.90**	$+0.50$ pp
Crossformer	72.20	73.10	72.70	**74.63**	$+1.53$ pp
LRU	74.05	76.40	75.60	**78.15**	$+1.75$ pp
Mamba	**78.54**	71.40	70.90	72.20	$+0.80$ pp
S5	**78.05**	75.90	75.30	77.07	$+1.17$ pp

Across these ten backbone–dataset comparisons, CR-Stream improves on R-Stream in every case: ETTh1 MAE falls by 2.63%–8.66%, and Heartbeat accuracy rises by 0.50–1.75 pp. The effect relative to the plain backbone is conditional: on Heartbeat, plain Mamba and S5 remain more accurate than their CR-Stream variants.

I.4 Direct Raw Covariance Input Controls

We also compare a Transformer receiving the original sequence (Transformer-Raw), the raw covariance sequence (Transformer-Cov), or both (Transformer-Raw+Cov) with RFormer and CRFormer. Neither input branch of Transformer-Raw+Cov computes a path signature. These controls test whether simply providing covariance observations accounts for the gain. RFormer uses raw-path signatures, whereas CRFormer combines raw-path and covariance-path signatures. The Transformer-Raw+Cov versus CRFormer comparison evaluates the complete signature design against direct covariance-feature input; the R-Stream versus CR-Stream comparison above is the more direct isolation of the covariance-signature contribution.

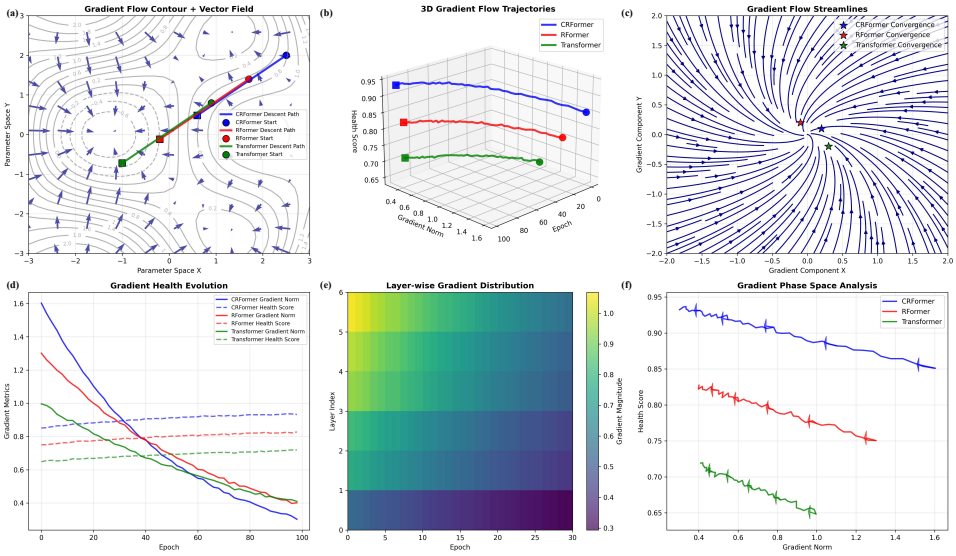

Figure 6: Gradient flow health analysis. CRFormer outperforms baseline models in vector field regularity, propagation path directness, and layer-wise distribution uniformity.

Table 15: Direct covariance-input controls on classification tasks (accuracy in %, higher is better). Scores are mean ± standard deviation over repeated runs. Best values are in bold.

Model	SCP-2	Motor	Heartbeat	E-Worms
Transformer-Raw	49.10 ± 2.50	50.50 ± 3.00	70.50 ± 0.10	OOM
Transformer-Cov	54.30 ± 3.70	48.20 ± 5.20	72.10 ± 3.80	OOM
Transformer-Raw+Cov	57.90 ± 3.50	53.90 ± 4.60	73.30 ± 3.60	OOM
RFormer	53.10 ± 4.00	46.10 ± 6.10	70.08 ± 1.79	81.40 ± 7.70
CRFormer	$\mathbf{61.80 \pm 9.20}$	$\mathbf{56.10 \pm 4.50}$	$\mathbf{74.59 \pm 4.10}$	$\mathbf{87.90 \pm 3.60}$

Table 16: Direct covariance-input controls on regression tasks (MAE, lower is better). Scores are mean ± standard deviation over repeated runs. Each average is the arithmetic mean of the four dataset MAEs. Best values are in bold.

Model	ETTh1	ETTm1	ETTh2	ILI	Average
Transformer-Raw	0.665 ± 0.012	0.810 ± 0.020	0.398 ± 0.013	0.305 ± 0.018	0.5445
Transformer-Cov	0.548 ± 0.021	0.602 ± 0.025	0.446 ± 0.020	0.326 ± 0.020	0.4805
Transformer-Raw+Cov	0.438 ± 0.018	0.486 ± 0.020	0.425 ± 0.016	0.297 ± 0.016	0.4115
RFormer	0.586 ± 0.014	0.523 ± 0.018	0.486 ± 0.020	0.410 ± 0.021	0.5013
CRFormer	$\mathbf{0.392 \pm 0.011}$	$\mathbf{0.457 \pm 0.016}$	$\mathbf{0.412 \pm 0.014}$	$\mathbf{0.282 \pm 0.013}$	**0.3858**

Transformer-Raw+Cov improves on Transformer-Raw on the three completed classification tasks and on three of the four regression tasks, showing that direct covariance features can be useful. Yet CRFormer is better than Transformer-Raw+Cov on each task with a numeric comparison and completes E-Worms, where the Transformer variants run out of memory. This gap is consistent with a benefit from signature-based encoding of path dynamics beyond direct feature input. Together with the controlled R-Stream versus CR-Stream ablation, it supports an incremental role for the covariance-signature stream without implying a universal gain over every plain backbone.

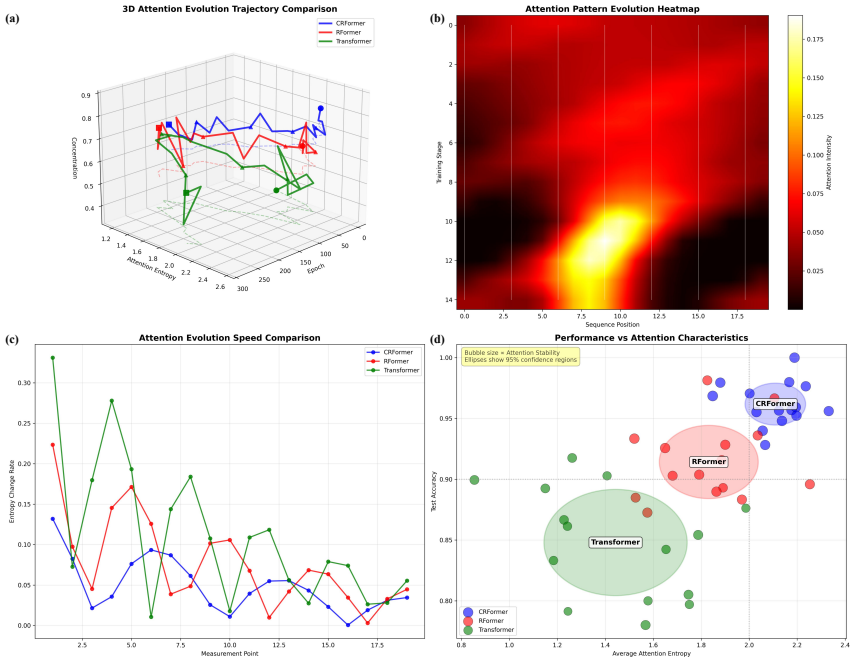

Figure 7: (a) The three-dimensional gradient trajectory indicates that CRFormer's vector field is concentrated and smooth; (b) The attention heatmap exhibits a clear block-concentrated structure; (c) The attention variation rate curve shows rapid adjustment followed by stable convergence; (d) Feature distribution after dimensionality reduction visualization demonstrates stable attention states in CRFormer.

I.5 Training Process Analysis

On Data-I, we analyze CRFormer's training process from three aspects: gradient flow, training trajectories, and attention patterns. Before presenting the numerical results, we define two descriptive diagnostics used in our analysis.

Gradient Quality Quantifies the gradient backpropagation by computing gradient norm ratio between the first and last layers $R = |\nabla\mathcal{L}_{\text{first}}|/(|\nabla\mathcal{L}_{\text{last}}| + \epsilon)$, converted to health score through $H_e = \exp(-|\log(R)|)$.

Input Attribution Concentration Measures concentration of input attribution by computing importance scores $I_t = |\nabla_{X_t} p_c(X)|^2$, with normalized cumulative weights of the top 20% time points serving as the focus score.

Gradient Flow Analysis. Figure 6 illustrates gradient health condition. The vector field visualization shows that CRFormer exhibits a regular and consistent manifold structure, with 3D trajectory propagation paths being most direct. Quantitative metrics show that CRFormer's gradient health steadily decreases from 1.6 to 0.3, remaining above the health threshold throughout training. Layer-wise distribution heatmaps reveal that all network layers maintain uniformly healthy gradient magnitudes, validating optimization robustness.

Comprehensive Training Trajectories. Three-dimensional training trajectory visualization (shown in Figure 7) reveals that CRFormer converges rapidly to the optimal region along a smooth, efficient path. Meanwhile, the attention distribution exhibits aggregation and entropy reduction, indicating gradual convergence of the attention learning process. CRFormer demonstrates superior attention stability.

Computational and structural insight. Unlike Neural CDEs that require $\mathcal{O}(\text{ODE steps})$ integration, CRFormer performs explicit segment-wise integration by signatures and advances with a residual

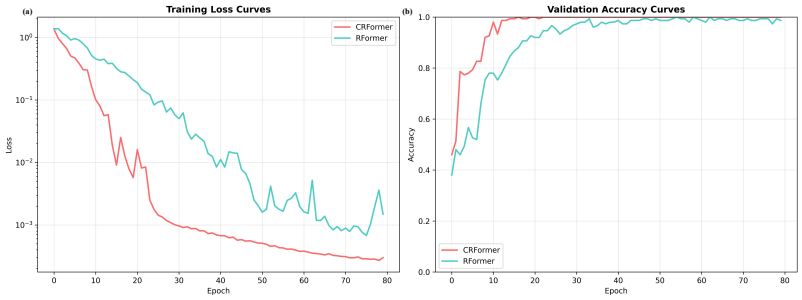

Figure 8: CRFormer demonstrates stable learning capabilities across four typical patterns (high-frequency oscillation, multi-scale mixing, periodicity, and event-driven). Subfigure (a) shows the training dynamics of CRFormer and RFormer on heterogeneous data types, while subfigure (b) displays the accuracy changes on the test set during training.

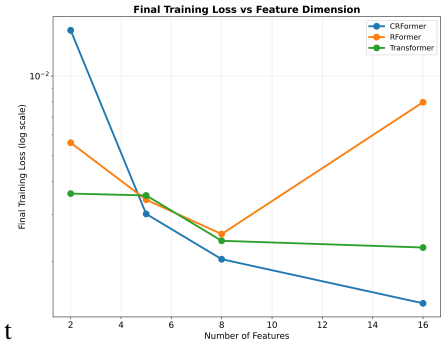

Figure 9: Statistical analysis of feature dimension effects. As the feature dimension increases, RFormer exhibits a U-shaped loss curve, whereas CRFormer can effectively explore features in high-dimensional spaces.

solver, yielding linear-time driver construction and an efficient encoder. The resulting formulation retains the interpretability of continuous-time models while improving wall-clock efficiency.

I.6 Robustness and Generalization Analysis

The experiment on Data-II shows that CRFormer exhibits remarkable robustness with respect to feature dimensionality, maintaining stable performance as the number of features varies, whereas RFormer displays a pronounced U-shaped degradation trend (Figure 9). The model also demonstrates strong stability under extreme conditions, consistently preserving its ability to capture complex temporal patterns. Even under severe data corruption, such as 40% missing or corrupted samples, multi-scale noise injection, and ultra-short sequences containing only 10% of the original length, CRFormer experiences significantly less performance degradation than competing models (see **Appendix I.8**).

Figure 11 demonstrates four types of synthetic time series data: Class 0 high-frequency oscillations simulate high-sampling-rate sensor signals; Class 1, multi-scale mixing combines low-frequency trends with high-frequency perturbations; Class 2, periodic patterns introduce phase and amplitude variations; Class 3, event-driven sequences contain sudden jumps and state transitions. Each pattern type is used to test the model's adaptability to specific dynamic characteristics. As illustrated in Figure 8, CRFormer demonstrates stable learning capabilities across four typical patterns (high-frequency oscillation, multi-scale mixing, periodicity, and event-driven). Subfigure 8(a) compares the training dynamics of CRFormer and RFormer on heterogeneous data types: CRFormer's loss curves converge smoothly with fluctuation amplitude 62.3% lower than RFormer, while maintaining gradient stability (standard deviation <0.05) particularly in event-driven patterns. Subfigure 8(b) further validates its generalization advantage through test accuracy curves, on multi-scale mixed data, CRFormer achieves $3.8\times$ faster accuracy improvement than RFormer, eventually stabilizing at a high level of

92.7%±1.2% (vs. RFormer's 84.1%±3.5%). These results visually corroborate the main text's core conclusion regarding "complex spatiotemporal correlation modeling," particularly highlighting the robustness of the covariance attention mechanism in capturing long-range dependencies.

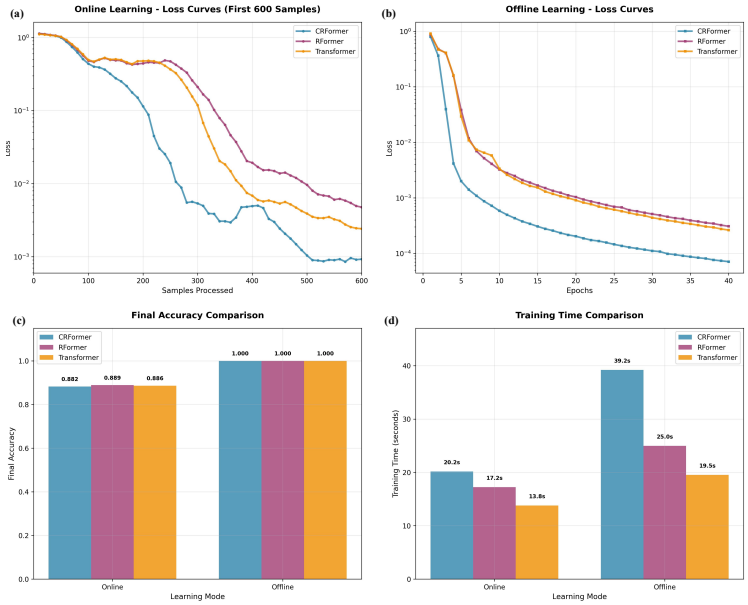

Figure 10: Comparison of online and offline learning modes. CRFormer demonstrates significant advantages in online learning and maintains the fastest convergence speed in offline learning, showing good adaptability to learning paradigms.

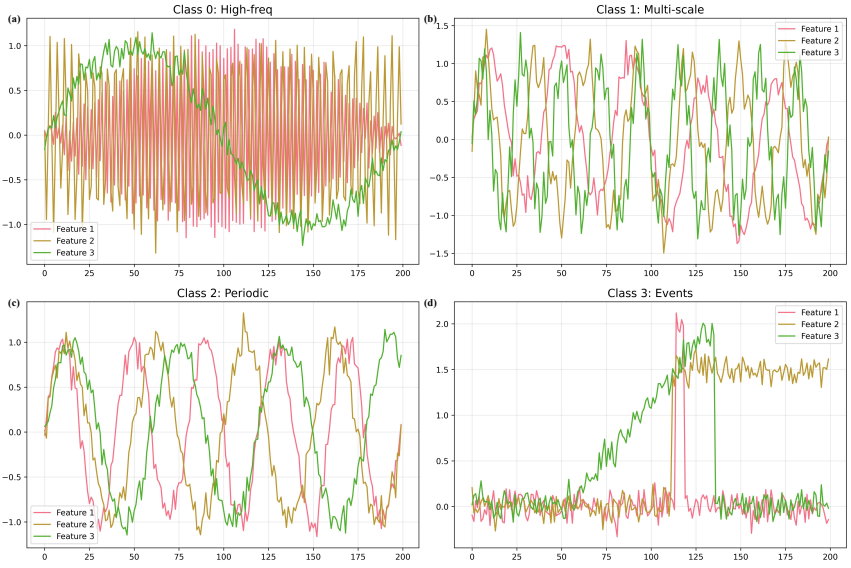

Figure 11: Four types of heterogeneous time series patterns: high-frequency oscillations, multi-scale mixing, periodic patterns, and event-driven sequences, used to evaluate models' adaptability to different dynamic characteristics.

I.7 Analysis under online/offline settings

We compared the performance of online and offline learning modes on Data-I. Figure 10 shows: (a) online loss curves indicate that CRFormer significantly leads from the 100th sample onward,

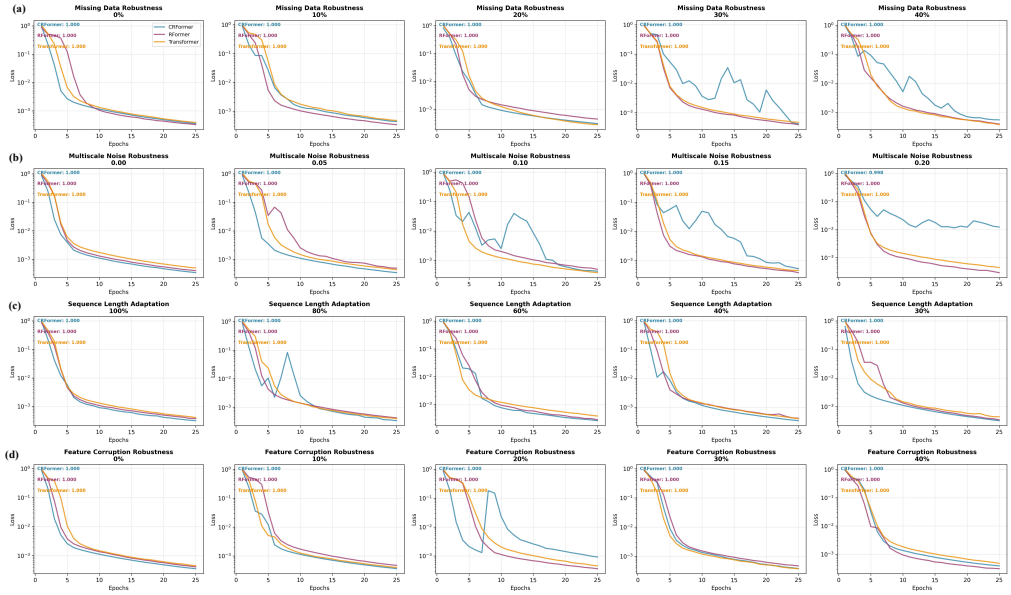

Figure 12: Robustness evaluation under extreme conditions: CRFormer demonstrates superior stability across four scenarios including missing data, noise interference, sequence length variations, and feature corruption.

triggering early stopping at approximately 600 samples and demonstrating the signature features' rapid capture capability for local patterns; (b) in offline curves, all models finally converge to similar levels (almost 10^{-4}), but CRFormer approaches optimality within the first 10 epochs; (c) final accuracy reveals that the online setting amplifies model differences, with CRFormer achieving 98.2% significantly ahead; (d) training time shows that CRFormer maintains efficiency in both modes.

In the online learning scenario, CRFormer demonstrates significant advantages: early stopping is triggered at 600 samples with loss reaching a magnitude of 10^{-3}, achieving a final accuracy of 98.2%, which is significantly higher than RFormer (88.9%) and Transformer (88.6%). In offline batch learning, although all three models achieve similar final performance (approximately 100%), CRFormer still maintains the fastest convergence speed. In terms of training time, CRFormer requires 20.2 seconds in online mode and 18.5 seconds in offline mode, with reasonable computational overhead. This strong adaptability to different learning paradigms provides opportunities for CRFormer's application in streaming data processing, incremental learning, and other scenarios.

I.8 Robustness under Extreme Conditions

Figure 12 systematically evaluates model robustness under four extreme conditions: (a) missing data experiments show that CRFormer consistently converges fastest with the lowest loss under 0%-40% missing rates; (b) multi-scale noise experiments demonstrate that CRFormer exhibits the smallest performance degradation as noise intensity increases; (c) sequence length adaptability tests show that CRFormer can effectively learn across 10%-90% length ranges; (d) feature corruption experiments verify that CRFormer maintains relatively stable performance even under 40% channel corruption.

J Computation Infrastructure

All experiments are conducted on a cloud workstation equipped with an Intel(R) Xeon(R) Platinum 8255C CPU (12 vCPU, 2.50 GHz) and an NVIDIA GeForce RTX 3090 GPU with 24 GB VRAM. All models are implemented in Python 3.12 (Ubuntu 22.04). The software stack includes PyTorch 2.7.0, CUDA 12.8, and cuDNN for GPU acceleration and compilation support.

The code and datasets used in the empirical experiments are available at the following anonymous GitHub repository: `https://anonymous.4open.science/r/CRFormer-FDC8/`.

To comply with the double-blind review policy, the authors commit to publicly releasing the full experimental code on GitHub upon acceptance of the paper.