Faithful Embeddings of Irregular and Asynchronous Data for Online Log-NCDEs

Benjamin Walker
University of Oxford
Continuous Intelligence Limited

Alexandre Bloch
University of Oxford

Lingyi Yang
University of Nottingham
Continuous Intelligence Limited

Sam Morley
University of Oxford

Terry Lyons
University of Oxford
Imperial College London

Abstract

Continuous-time models are a natural choice for irregular and asynchronous data. A central design choice is how to embed discrete observations into continuous time. Interpolation- and imputation-based embeddings reconstruct a continuous observation path, making the model sensitive to the choice of reconstruction. We show that this reconstruction step is unnecessary; under mild conditions, compact-set universality on the model input space transfers to the data space whenever the embedding from data to input is continuous and injective. Guided by this result, and building on the rectilinear control path for Neural Controlled Differential Equations (NCDEs), we introduce a continuous and injective embedding for Log-NCDEs, a universal class of continuous-time models. Our approach records observations as increments and composes them over arbitrary query intervals to directly form log-signatures. This provides interval-level summaries without first interpolating the observed variables, while supporting online computation. Experiments on synthetic controlled dynamics and real-world time-series datasets show that the representation is accurate, efficient, and robust to irregular, asynchronous, and sparse observations.

1 Introduction

Sequence architectures, such as Transformers [Vaswani et al., 2017], Structured State-Space Models (SSMs) [Gu et al., 2022, Gu and Dao, 2024], and Recurrent Neural Networks (RNNs), are built around discrete updates on a prescribed grid [Elman, 1990, Hochreiter and Schmidhuber, 1997]. This discretisation is natural for text and regularly sampled time series, but is limiting in interactive and physical settings, where observations arrive irregularly, actions are taken asynchronously, and the underlying environment continues to evolve between measurements [Marvasti, 2001, Heemels et al., 2012]. Continuous-time machine learning provides a natural framework for such data [Pearlmutter, 1989, Rico-Martínez et al., 1992]. Given a continuous-time embedding of the observations, the hidden state can evolve continuously and be queried at arbitrary times [Kidger et al., 2020].

A central design choice is how to construct this embedding. One common strategy is to complete the data by imputation, interpolation, or learned interpolation before applying a model [Che et al., 2018, Shukla and Marlin, 2019, 2021, Challu et al., 2023, Wang et al., 2025]. Neural Controlled Differential Equations (NCDEs) and ContiFormer are prominent examples of this approach [Kidger et al., 2020, Morrill et al., 2022, Jhin et al., 2022, Chen et al., 2023]. Other models avoid explicit interpolation by combining continuous latent dynamics with discrete updates, as in ODE-RNNs and

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

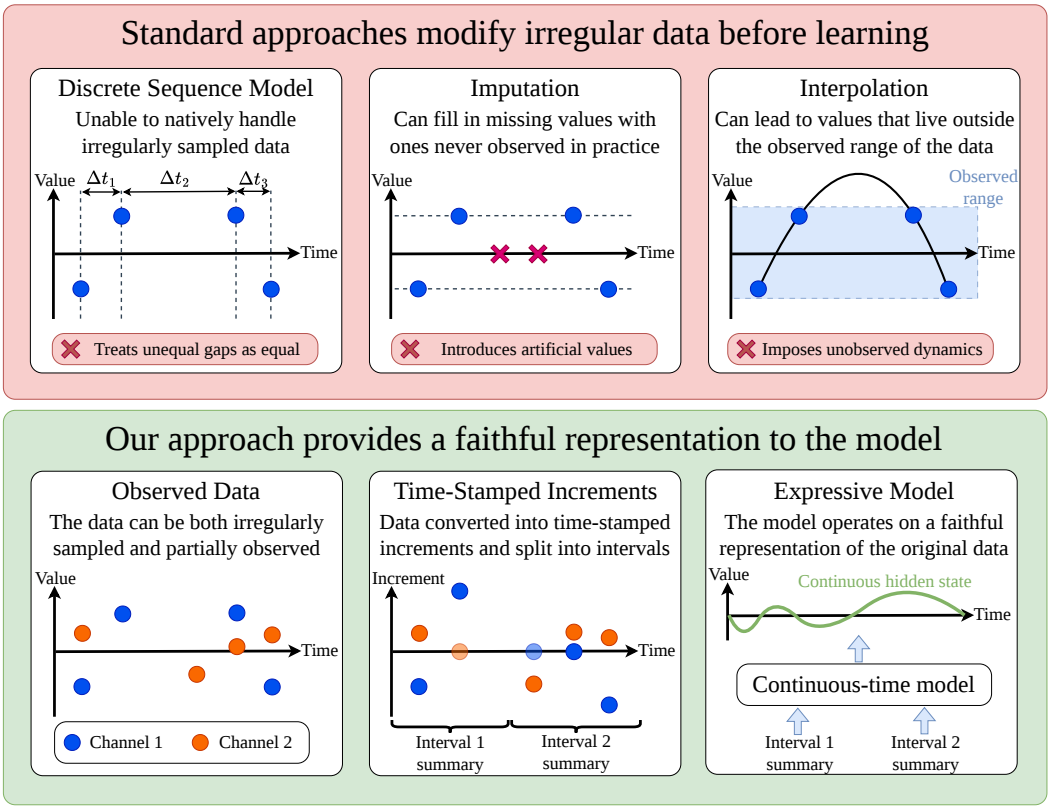


Figure 1: **Faithful representations of irregular data.** Imputation and interpolation complete the data before modelling, introducing values or dynamics that may be unrealistic. Our approach records the observed stream as time-stamped increments and computes interval summaries of a faithful (continuous and injective) representation of the discrete and irregular data.

GRU-ODE-Bayes [Rubanova et al., 2019, De Brouwer et al., 2019]. These approaches differ in how they represent discrete, irregular, and asynchronous data in continuous time.

The central principle of this paper is that a continuous-time representation of discrete observations should preserve only the information that was observed, rather than guess the unobserved behaviour between measurements. The latter should form part of the modelling as opposed to part of the representation. When guesses are built into the representation, the model may consume unrealistic inputs, as illustrated in Figure 1. Morrill et al. [2022] studied control-path embeddings for NCDEs, exploring properties such as online, smooth, bounded, and unique, or equivalently injective. For Log-NCDEs, smoothness is no longer intrinsic, since the model uses log-signatures on intervals rather than pointwise values of a control path [Walker et al., 2024]. We prove that, when the data and model input spaces are Hausdorff, universality of the model on compact subsets of its input space transfers to the data space as long as the data embedding is continuous and injective. We call such embeddings *faithful*. Since NCDEs and Log-NCDEs are universal on path space [Kidger et al., 2020, Walker et al., 2024], this result applies to both model classes for faithful observation embeddings.

This leads to three desiderata for a Log-NCDE control-path embedding. It should be faithful, computable online, and support efficient calculation of the embedded path's log-signature over arbitrary query intervals. We construct such an embedding by mapping each observation to a local log-signature contribution that records which channels were observed and their values. These contributions compose over arbitrary query intervals, so the representation can be built directly from the observations in each interval. Continuously observed variables, such as time, contribute between observations. Figure 1 illustrates our approach.

This yields a faithful representation of the observed stream without interpolation, decouples the input sampling times from the output grid, and supports parallel-in-time computation. The resulting

algebraic formulation can also accommodate observations beyond first-order channel values, such as the signed area. In the first-order case with time as the only continuously observed variable, the continuous path corresponding to our embedding is the rectilinear control path introduced for NCDEs by Morrill et al. [2022]. Our construction can therefore be viewed as a Log-NCDE-compatible generalisation of the rectilinear scheme.

Our contributions are as follows:

- We formalise the notion of a faithful embedding and prove that faithful embeddings transfer universality from compact subsets of one Hausdorff space to another in Section 2.1.
- We introduce a faithful control-path embedding for Log-NCDEs whose interval log-signatures can be computed efficiently over arbitrary intervals in Section 2.4.
- We demonstrate the computational benefits and robustness to irregularity of the resulting models on synthetic controlled systems and real-world time-series classification in Section 3.

2 Faithful Embeddings of Irregular Time Series for Log-NCDEs

2.1 Faithful Embeddings

Before modelling, data must be embedded into the model input space. We call such an embedding *faithful* if it is continuous and injective, thus ensuring that nearby data remain nearby after embedding and that distinct data remain distinguishable. Together, these conditions are sufficient for universality to transfer from the downstream model to the original data space, as formalised by the following lemma.

Lemma 1 (Universality Transfer)**.** *Let $\mathcal{X}$ and $\mathcal{P}$ be Hausdorff topological spaces, and let $\varphi : \mathcal{X} \to \mathcal{P}$ be a continuous injection. Suppose $\mathcal{F} \subseteq C(\mathcal{P}; \mathbb{R})$ is a family of continuous real-valued functions such that for every compact $K \subseteq \mathcal{P}$, the restrictions $\mathcal{F}|_K$ are dense in $C(K; \mathbb{R})$ in the uniform topology. Then for every compact $K_0 \subseteq \mathcal{X}$, the restrictions $(\mathcal{F} \circ \varphi)|_{K_0}$ are dense in $C(K_0; \mathbb{R})$ in the uniform topology, where $\mathcal{F} \circ \varphi := \{f \circ \varphi : f \in \mathcal{F}\}$.*

The proof is in Appendix A. With $\mathcal{X}$ the data space, $\mathcal{P}$ the model input space, and φ the embedding, Lemma 1 shows that faithful embeddings transfer universality back to the original data. This recovers the usual signature-universality argument: Stone–Weierstrass gives universality on signature space, and injectivity transfers it back to paths [Stone, 1948, Levin et al., 2016]. Appendix A also notes the obvious converse obstruction: if φ identifies two data objects, no downstream model can separate them. Appendix B discusses the scope of this universality statement and its distinction from sample efficiency or optimisation guarantees.

We now construct a faithful embedding of an irregular observation stream as a path. The construction is tailored to Log-NCDEs, which require interval log-signatures rather than pointwise values of an interpolated path. It therefore computes the required log-signatures directly from the observations, without interpolating the data. The procedure follows the signature-computation viewpoint of Morley and Lyons [2024]. We begin with an introduction to the tensor algebra, signatures, and log-signatures.

2.2 Tensor algebra, signatures, and log-signatures

Let $X : [0, T] \to \mathbb{R}^{d_X}$ be a path of bounded variation, with coordinates $X = (X^1, \dots, X^{d_X})$. The signature of X over $[s, t]$ records all iterated integrals of its coordinate increments,

$$S_{s,t}(X)^I = \int_{s<u_1<\cdots<u_k<t} \mathrm{d}X^{i_1}_{u_1} \cdots \mathrm{d}X^{i_k}_{u_k}, \tag{1}$$

where $I = i_1 \cdots i_k$ is a word of length $|I| = k$ with letters in $\{1, \dots, d_X\}$ [Lyons et al., 2007, Friz and Victoir, 2010]. The empty word is denoted by $\emptyset$, with $S_{s,t}(X)^\emptyset = 1$. The kth level consists of all signature coordinates $S_{s,t}(X)^I$ indexed by words I of length k. The collection of all levels forms an element of the tensor algebra $T((\mathbb{R}^{d_X}))$, equipped with the product

$$(\mathbf{a} \otimes \mathbf{b})^I = \sum_{I=JK} \mathbf{a}^J \mathbf{b}^K, \tag{2}$$

where the sum is over all decompositions of I into two words J and K. The unit is $\mathbf{1}$, with $\mathbf{1}^{\emptyset} = 1$ and $\mathbf{1}^{I} = 0$ for $|I| \geq 1$. Chen's identity [Chen, 1954] states that, for $s \leq u \leq t$,

$$S_{s,t}(X) = S_{s,u}(X) \otimes S_{u,t}(X), \tag{3}$$

so this product is the algebraic counterpart of concatenating intervals [Chevyrev and Kormilitzin, 2016, McLeod and Lyons, 2025].

The tensor product is non-commutative, giving the Lie bracket $[\mathbf{a}, \mathbf{b}] = \mathbf{a} \otimes \mathbf{b} - \mathbf{b} \otimes \mathbf{a}$. The free Lie algebra $\mathfrak{L}((\mathbb{R}^{d_X}))$ is generated by degree-one tensors under this bracket [Reutenauer, 1993]. The log-signature is defined

$$\log S_{s,t}(X) = \sum_{m=1}^{\infty} \frac{(-1)^{m+1}}{m} \left(S_{s,t}(X) - \mathbf{1}\right)^{\otimes m} \tag{4}$$

and lives in $\mathfrak{L}((\mathbb{R}^{d_X}))$ as signatures of bounded variation paths are group-like elements of $T((\mathbb{R}^{d_x}))$ [Chen, 1957]. Although it contains the same information as the signature, each level is represented in the lower-dimensional free Lie algebra rather than in the full tensor algebra. The depth-N truncated tensor algebra $T^N(\mathbb{R}^{d_X})$ keeps only levels with degree at most N, and $S^N_{s,t}(X)$ and $\log S^N_{s,t}(X)$ are the projections of $S_{s,t}(X)$ and $\log S_{s,t}(X)$ onto these levels. We now introduce Log-NCDEs, which interact with the driving path on an interval through the truncated log-signature.

2.3 Neural controlled differential equations

An NCDE models a continuous-time hidden state $h_t \in \mathbb{R}^{d_h}$ driven by a path X through a parametrised vector field $f_\theta : \mathbb{R}^{d_h} \to \mathbb{R}^{d_h \times d_X}$ [Kidger et al., 2020],

$$\mathrm{d}h_t = f_\theta(h_t)\, \mathrm{d}X_t. \tag{5}$$

For $a \in \mathbb{R}^{d_X}$, $f_\theta(\cdot)a : \mathbb{R}^{d_h} \to \mathbb{R}^{d_h}$ denotes the vector field obtained by multiplying the matrix output of f_θ by a. The vector field may be nonlinear, or may act linearly on the hidden state as in a Linear NCDE, $f_\theta(h_t)\mathrm{d}X_t = \sum_{i=1}^{d_X} A^i_\theta h_t \mathrm{d}X^i_t$ [Cirone et al., 2024]. Predictions are produced from the hidden state by a readout map l_ψ. With a linear readout of the terminal hidden state, both nonlinear and Linear NCDEs are universal for continuous path-to-point maps [Kidger et al., 2020]. With a nonlinear readout of the hidden path, they are universal for continuous online path-to-path maps [Cirone et al., 2024]. Structured Linear CDEs (SLiCEs) restrict the matrices A^i_θ to structured matrix families for efficiency. Several structures, including block-diagonal, diagonal-plus-low-rank, sparse, and the exponentially weighted signature retain the same path-to-point and path-to-path universality guarantees as dense Linear NCDEs [Walker et al., 2025, Movahedi et al., 2025, Bloch et al., 2026].

The Log-ODE method approximates CDE dynamics on an interval by constructing an autonomous ODE from the truncated log-signature of the driving path and the iterated Lie brackets of the vector fields [Castell and Gaines, 1995]. A Log-NCDE applies this approximation to NCDEs, so that the hidden-state evolution depends on the driving path only through its log-signature over intervals. Let $\bar{f}_\theta$ be the Lie algebra homomorphism extending f_θ from degree-one tensors to $\mathfrak{L}^N(\mathbb{R}^{d_X})$, so that $\bar{f}_\theta(\cdot)a = f_\theta(\cdot)a$ for $a \in \mathbb{R}^{d_X}$ and recursively

$$\bar{f}_\theta(\cdot)[a, b] = [\bar{f}_\theta(\cdot)a, \bar{f}_\theta(\cdot)b] = J_{\bar{f}_\theta(\cdot)b}\bar{f}_\theta(\cdot)a - J_{\bar{f}_\theta(\cdot)a}\bar{f}_\theta(\cdot)b, \tag{6}$$

where $J_{\bar{f}_\theta(\cdot)b}$ denotes the Jacobian of the vector field $\bar{f}_\theta(\cdot)b : \mathbb{R}^{d_h} \to \mathbb{R}^{d_h}$ and f_θ needs to be $\mathrm{Lip}(\gamma)$ for $\gamma \geq N$ [Walker et al., 2024]. On an interval $[r_i, r_{i+1}]$, the Log-ODE update solves

$$\frac{\mathrm{d}\bar{h}_s}{\mathrm{d}s} = \bar{f}_\theta(\bar{h}_s)\frac{\log S^N_{r_i,r_{i+1}}(X)}{r_{i+1} - r_i}, \qquad s \in [r_i, r_{i+1}], \qquad \bar{h}_{r_i} = h_{r_i}, \tag{7}$$

and sets $h_{r_{i+1}} \approx \bar{h}_{r_{i+1}}$.

For Linear NCDEs and SLiCEs, the interval flow has the closed-form $\exp\left(\bar{A}_\theta\left(\log S^N_{s,t}(X)\right)\right)$, where $\bar{A}_\theta$ is obtained from the matrices A^i_θ by the same Lie-bracket construction, with vector-field brackets replaced by sign-flipped matrix commutators. Each interval acts on the hidden state by a linear map and the hidden states can be computed in parallel using an associative scan [Walker et al., 2025]. For SLiCEs, the efficiency of the chosen matrix structure is retained only when the Log-ODE construction

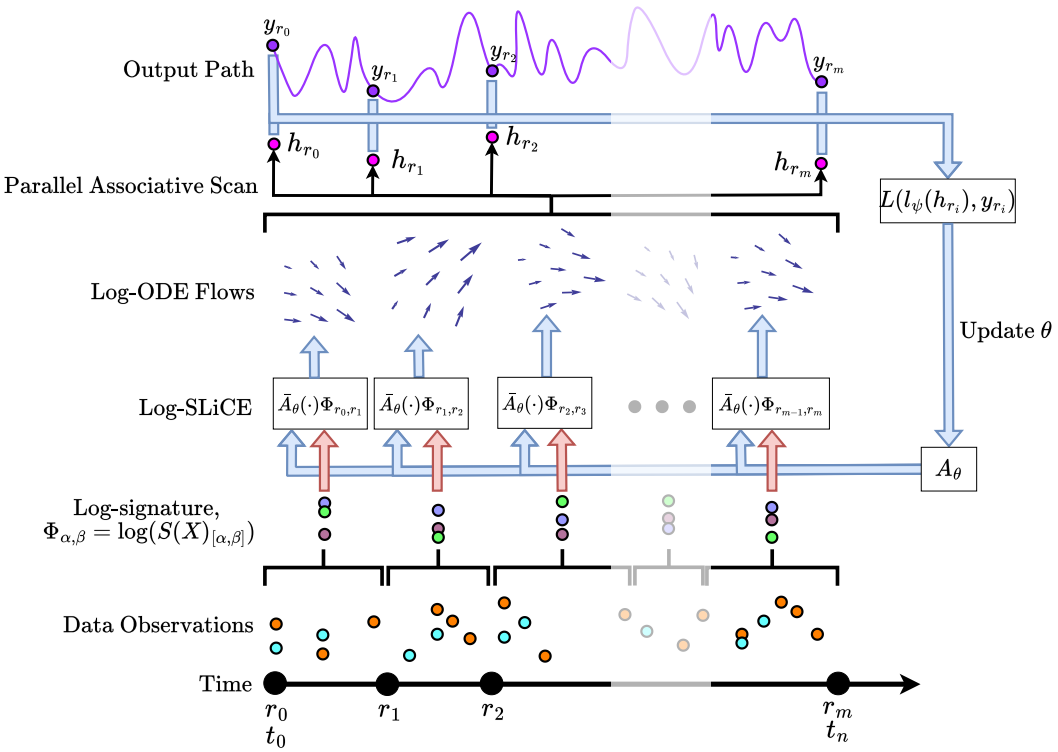


Figure 2: **Schematic of a Log-SLiCE.** Irregular observations are first summarised by interval log-signatures $\Phi_{\alpha,\beta} = \log(S(X)_{[\alpha,\beta]})$. The iterated Lie brackets of the matrices A_θ^i, represented by $\bar{A}_\theta$, map each log-signature to a Log-ODE flow. These interval-wise hidden-state updates are composed using a parallel associative scan. The resulting hidden states h_{r_i} are decoded to produce estimates of the output values y_{r_i}, and the loss $L(l_\psi(h_{r_i}), y_{r_i})$ is used to update the parameters.

remains inside that structure. This is true for block-diagonal SLiCEs, since block-diagonal matrices are closed under products, commutators, and matrix exponentials [Walker et al., 2025].

We refer to the resulting structure-preserving Log-ODE variant of SLiCE as Log-SLiCE. As shown in Section 3.1, Log-SLiCE reduces the time per training step by up to $3{,}000\times$ on time series with roughly $18{,}000$ observations when compared to standard NCDEs. Figure 2 illustrates the Log-SLiCE pipeline. The next section constructs the map from data observations to log-signatures over intervals.

2.4 From observations to log-signatures

Suppose we have d_{disc} channels for which we observe values at discrete times in $[0, T]$. We write the observation stream as

$$x = ((t_0, s_0, x_0), \ldots, (t_n, s_n, x_n)), \qquad 0 \le t_0 < \cdots < t_n \le T, \tag{8}$$

where $s_i \subseteq \{1, \ldots, d_{\text{disc}}\}$ is the non-empty set of channels observed at time t_i, and $x_i^k \in \mathbb{R}$ is the observed value of channel $k \in s_i$. The stream can be irregular and asynchronous, because the gaps $t_{i+1} - t_i$ need not be equal and different subsets s_i can be observed at each time. Suppose we also have d_{cont} continuously observed channels for which we can access their value at any time, with time itself being the canonical example. The goal is to define a faithful path embedding of these continuous variables and observation streams, together with a procedure for efficiently computing the embedded path's log-signature over any query interval, so that it can be used with Log-NCDEs. Rather than first constructing an interpolated path and then computing its log-signature, we define the interval log-signatures directly.

We first choose the coordinates of the path space. For an embedding to be injective, the construction must retain both the last recorded value of each discretely observed channel and whether a new

observation has occurred. We therefore use

$$d_X = 2d_{\text{disc}} + d_{\text{cont}}, \tag{9}$$

where the first d_{disc} coordinates give the current recorded values of the discretely observed channels, the next d_{disc} coordinates record observation counts in each channel, and the final d_{cont} coordinates store continuously observed variables. The count coordinates distinguish an observation that repeats the previous recorded value from the absence of a new observation in that channel. Where this distinction is irrelevant, the count coordinates can be omitted. The resulting embedding then intentionally identifies streams that differ only by the insertion or removal of value-preserving observations. This is a modelling choice, with the invariance-computational tradeoff discussed in Appendices A.5 and B.

We convert each observation into an increment in $\mathbb{R}^{d_X}$. For $k \in s_i$, let $\text{prev}(i,k)$ be the most recent index $j < i$ such that $k \in s_j$, i.e. the previous index where channel k is observed. We base-point augment the stream by setting $x^k_{\text{prev}(i,k)} = 0$ if no such index exists. The event increment at time t_i is

$$\delta_i = \sum_{k \in s_i} \left[\left(x^k_i - x^k_{\text{prev}(i,k)} \right) e_k + e_{d_{\text{disc}}+k} \right] \in \mathbb{R}^{d_X}, \tag{10}$$

where $\{e_i\}_{i=1}^{d_X}$ is the standard basis of $\mathbb{R}^{d_X}$. Thus each observed channel contributes its change since its previous observation, together with a unit increment in its count coordinate.

Each δ_i is then included into the truncated free Lie algebra. By default, we place δ_i in degree one and set all higher-order components to zero, $\Delta_i = (0, \delta_i, 0, \ldots, 0) \in \mathfrak{L}^N(\mathbb{R}^{d_X})$. This is where additional local information can be incorporated. For example, if an observation event includes precomputed signed-area terms, or if a local segment has known iterated integrals, these higher-order terms can be stored directly in Δ_i. We provide an example of this for Brownian motion paths in Section 3.3. The corresponding signature factor associated with the observation event is $E_i = \exp(\Delta_i) \in T^N(\mathbb{R}^{d_X})$.

Let $C : [0,T] \to \mathbb{R}^{d_{\text{cont}}}$ be a continuous path of bounded-variation representing the continuously observed variables, embedded in $\mathbb{R}^{d_X}$ as $\widetilde{C}_t = (0, \ldots, 0, C_t)$, and write $\Gamma_{u,v} = S^N_{u,v}(\widetilde{C})$ and $B_C = \exp(\widetilde{C}_0)$. For a query interval $[\alpha, \beta)$ containing events $\alpha \le t_{i_1} < \cdots < t_{i_m} < \beta$, the signature over the interval is defined by

$$G_{\alpha,\beta}(x,C) = \begin{cases} B_C \otimes \Gamma_{\alpha,t_{i_1}} \otimes E_{i_1} \otimes \Gamma_{t_{i_1},t_{i_2}} \otimes E_{i_2} \otimes \cdots \otimes E_{i_m} \otimes \Gamma_{t_{i_m},\beta}, & \alpha = 0, \\ \Gamma_{\alpha,t_{i_1}} \otimes E_{i_1} \otimes \Gamma_{t_{i_1},t_{i_2}} \otimes E_{i_2} \otimes \cdots \otimes E_{i_m} \otimes \Gamma_{t_{i_m},\beta}, & \alpha > 0, \end{cases} \tag{11}$$

with B_C base-point augmenting C. Events at a right endpoint are assigned to the next interval, except that events at T are included in the final interval. The interval log-signature is

$$\Phi_{\alpha,\beta}(x,C) = \log G_{\alpha,\beta}(x,C) \in \mathfrak{L}^N(\mathbb{R}^{d_X}). \tag{12}$$

The Log-ODE method requires interval log-signatures on a partition $\pi = \{0 = r_0 < r_1 < \cdots < r_M = T\}$, which we denote by $\varphi_\pi(x,C) = \big(\Phi_{r_0,r_1}(x,C), \ldots, \Phi_{r_{M-1},r_M}(x,C)\big)$. The construction is depicted schematically in Figure 3.

Once the event increments have been formed, each interval can be processed independently and its factors composed in parallel using an associative scan, allowing for efficient computation. Furthermore, the construction is online, since for any π, the information needed to compute each $\Phi_{r_i,r_{i+1}}(x,C)$ depends only on information up to time r_{i+1}. Thus the only remaining desired property is that these interval summaries arise from a faithful path embedding, since NCDEs and Log-NCDEs are universal models on path space. We therefore proceed by constructing a continuous path $\Psi(x,C)$ whose interval log-signatures equal $\Phi_{\alpha,\beta}(x,C)$ and which satisfies the following proposition.

Proposition 2 (Faithfulness (informal))**.** *Assume the observation-count coordinates are included and that C contains time as a channel. Then the map $(x,C) \to \Psi(x,C)$ is faithful.*

The construction of $\Psi(x,C)$ and formal statement and proof of Proposition 2 are given in Appendix A.4. As with the rectilinear construction [Morrill et al., 2022], each observation event is assigned an interval of auxiliary time along which value and count coordinates move continuously while physical time is held fixed. Continuously observed variables contribute their signature factors over the physical-time gaps between events. Since signatures are invariant under reparametrisations of the auxiliary parameter, the durations assigned to these event intervals do not affect the resulting physical-time interval signatures.

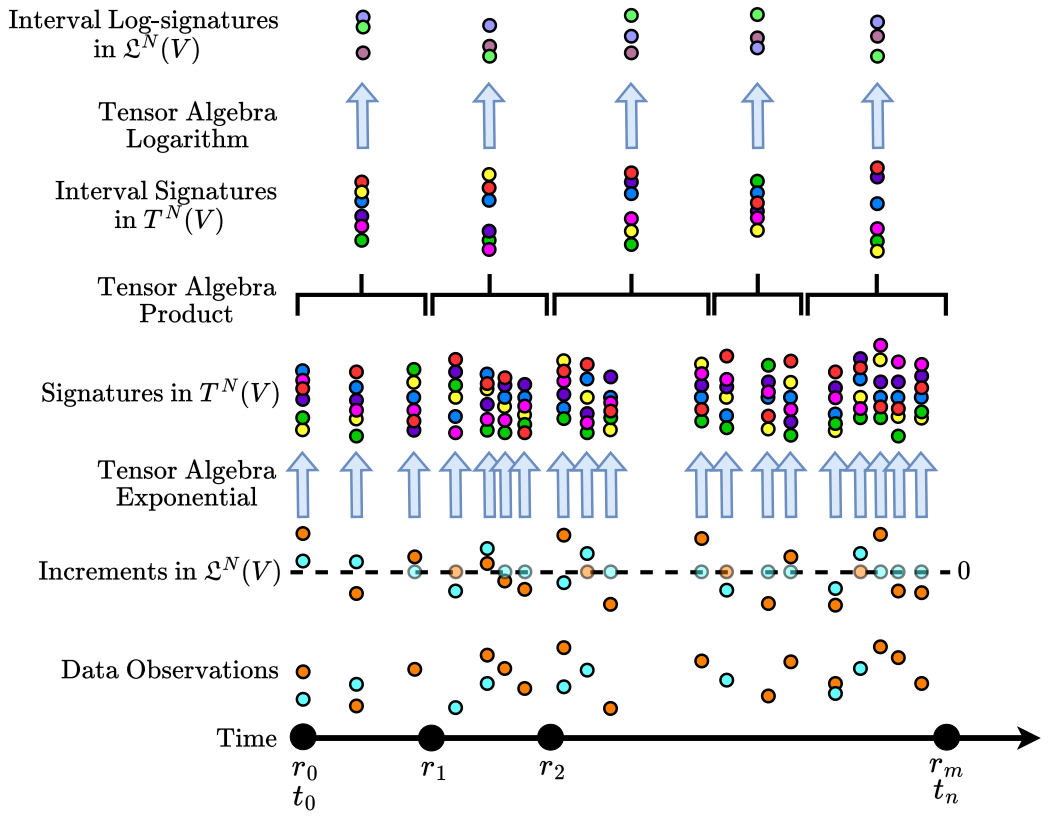


Figure 3: **Constructing interval log-signatures from irregular observations.** Each observation is converted into a Lie-algebra increment containing value increments and observation-count increments. Continuously observed variables, such as time, contribute signature factors over gaps between observations. Event and gap factors are composed over each query interval using a parallel associative scan with the tensor product, then mapped back to the free Lie algebra by the tensor logarithm. This gives interval log-signatures without interpolating the discretely observed channels.

3 Experiments

We evaluate our proposed embedding in four settings. First, we use EigenWorms, the longest dataset in the UEA multivariate time-series classification archive, to test the effect of the Log-ODE method on accuracy, speed, and GPU memory. Second, we test prediction on a synthetic sinusoidal system with output grids independent of input observation times. Third, we test prediction on a Brownian controlled system whose solution depends on second-order information. Fourth, we evaluate robustness on six UEA classification datasets under input dropping. The empirical scope is discussed further in Appendix B.

3.1 Effect of the Log-ODE Method

Table 1 compares three NCDE models with and without the Log-ODE method on EigenWorms, holding all hyperparameters fixed to those selected by Walker et al. [2024] for NCDE and Log-NCDE. EigenWorms has 17,984 observations per series, making it a direct test of whether interval-level computation reduces the cost of long input sequences without sacrificing accuracy. Timing and memory measurements were run on an NVIDIA H100 GPU with a batch size of 1. Further experimental details are given in Appendix C.3.1. On this benchmark, every Log-ODE variant is faster and more accurate. Overall, combining the Log-ODE method with a SLiCE increases test accuracy by 11.1 percentage points over a standard NCDE, while reducing the time per training step by a factor of almost 3,000. This cuts the time for 10,000 training steps from 3 days to 1.5 minutes.

Table 1: **NCDE models with and without the Log-ODE method on EigenWorms.** Results marked † are from Walker et al. [2024]; results marked ‡ are from Walker et al. [2025].

Model family	Log-ODE	Test Acc.	Time per step (s)	GPU Mem. (MB)
NCDE	×	$75.0 \pm 3.9^{\dagger}$	26.02	3484
	✓	$85.6 \pm 5.1^{\dagger}$	2.263	3494
LNCDE	×	87.2 ± 5.2	0.100	13730
	✓	$88.3 \pm 3.7^{\ddagger}$	0.018	3492
BD-SLiCE	×	81.1 ± 6.7	0.024	3494
	✓	$86.1 \pm 3.5^{\ddagger}$	0.009	3494

Within each family, the Log-ODE method speeds up NCDEs by approximately $11.5\times$, LNCDEs by approximately $5.6\times$, and BD-SLiCE by approximately $2.7\times$.

3.2 Synthetic coupled data

For a synthetic benchmark, we consider the multivariate system

$$\begin{aligned} z(t) &= \omega t + \phi, \\ x_i(t) &= A_i \sin\big(z(t) + \delta_i\big), \qquad i = 1, \dots, d, \end{aligned} \tag{13}$$

where z is a shared latent phase which gives cross-channel structure. The parameters ω and ϕ are independently drawn per sample. The parameters A_i and δ_i are sampled independently per channel. We fix $d = 2$. For further information on the set-up see Appendix C.1.

The observations are generated under four sampling regimes. The channels may be observed synchronously and regularly, synchronously and irregularly, asynchronously and irregularly, or at significantly different frequencies, where one channel is much sparser than the other.

For each sample, we draw a random query partition $0 = q_0 < q_1 < \cdots < q_m = T$, where m varies between samples. We test whether models can learn the value at the end of each query interval $[q_k, q_{k+1})$, i.e. $x(q_{k+1})$, constructed from (13). This is not next-observation prediction since query endpoints need not coincide with observation times. The task therefore tests whether the model can use interval summaries to predict on an output grid chosen independently of the input sampling grid.

We first train one model for each data observation regime. Illustrative observation patterns and sample model outputs are shown in Figures 6–9 of Appendix C.1. We then evaluate these pre-trained models across the other regimes to assess cross-regime generalisation. Figure 4 reports the mean and standard deviation of this analysis. A model trained on synchronously observed, regularly sampled data struggles to generalise to asynchronously observed data in which one channel is sampled much more sparsely. This suggests that the model learns shortcuts when trained only on regularly sampled data. Conversely, because the asynchronous sparse regime is the hardest setting, a model trained on it remains robust across all other regimes. For irregularly sampled data, there is little difference between synchronous and asynchronous observations. This agrees with our view of faithful embeddings, since both regimes contain similar quantities of information and the model can learn to use it when it is faithfully represented. Figure 10 in Appendix C.1 shows that their training curves are also very similar.

3.3 Linear system driven by Brownian motion

Next we test whether the model can exploit higher order information if it is available. We use precomputed Brownian rough paths, with the Lévy area computed as described by Jelinčič et al. [2025]. Each sample consists of a 4-dimensional Brownian stream on a fine regular grid (2048 observations between $[0, 1]$), represented by log-signatures up to level 2, giving an input dimension of 10. These Brownian paths are used to drive a Stratonovich linear controlled system

$$\mathrm{d}X_t = V_1 X_t \circ \mathrm{d}W_t^1 + V_2 X_t \circ \mathrm{d}W_t^2, \tag{14}$$

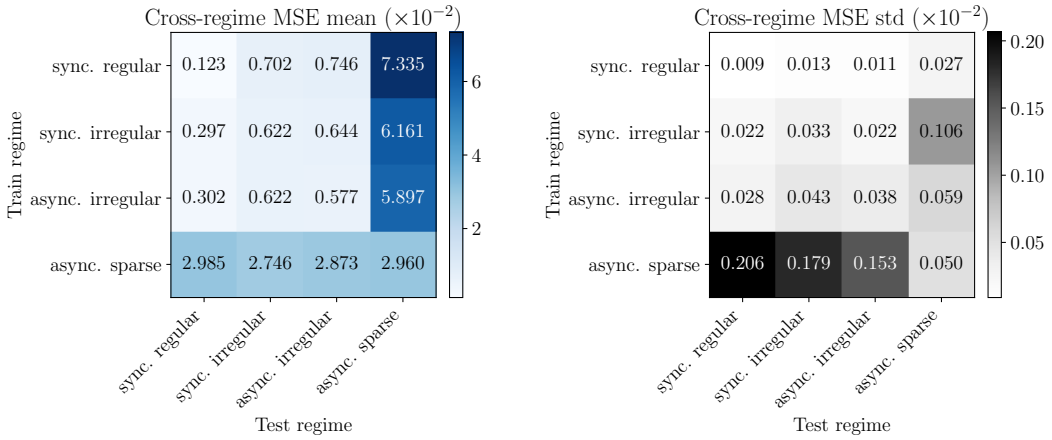


Figure 4: **Heatmap showing how well trained sinusoid models generalise across data sampling regimes.** The mean and standard deviation are computed over 5 random seeds.

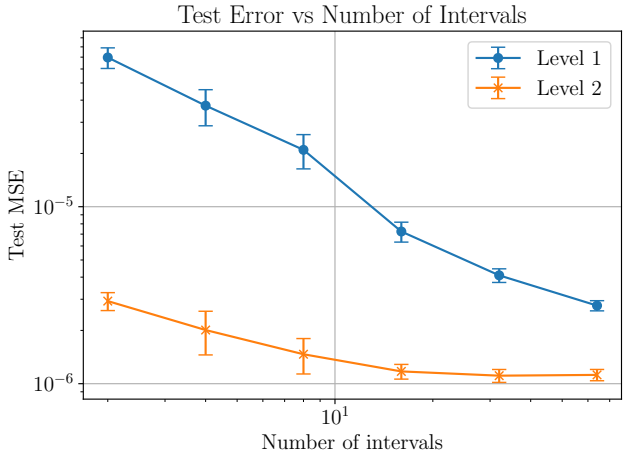


Figure 5: **Average test MSE against number of query intervals used for linear system driven by Brownian motion over 5 seeds.** As the system is driven by area information, we see a much lower MSE for when using level 2 log-signatures as we would expect, especially for fewer query intervals.

where $V_1, V_2 \in \mathbb{R}^{r\times r}$ are fixed non-commuting matrices. Note that the CDE is only being controlled by two dimensions of the Brownian stream, therefore part of the task is to also learn which streams are redundant. We fix the dimension of the output path $r = 2$. The non-commutativity ensures that the solution depends not only on first-level Brownian increments, but also on second-level Lie bracket terms. The model predicts the state at the next query endpoint, $y_k = X_{k+1},\ k = 0, \ldots, m-1$.

In Figure 5, we compare how our model performs when Lévy area information is provided compared with using only first level information, against the number of query intervals used. The solution depends intrinsically on second-level information, and we see that lower MSE is achieved with the higher order information. As the number of query intervals increase, the truncation error of using only first level decreases in magnitude, hence the test MSE decreases for the model trained on level 1. This highlights the flexibility of our approach in encompassing this additional information where it is available. Precise MSE averaged over 5 seeds can be seen in Table 4 in Appendix C.2.

3.4 UEA input dropping

We evaluate robustness under input dropping on the six UEA time-series classification datasets chosen by Walker et al. [2024]. The comparison includes four structured recurrent baselines, S5 [Smith et al., 2023], LRU [Orvieto et al., 2023], S6, and Mamba [Gu and Dao, 2024], and an NCDE using Hermite

Table 2: **Results on the six UEA datasets.** Average test accuracy (in %) and average dataset rank with no dropping and 30%, 70%, and 95% input dropping. Lower rank is better. No-drop values marked † are from Walker et al. [2024]; no-drop values marked ‡ are from Walker et al. [2025].

	No drop		30% drop		70% drop		95% drop	
Model	Av. Acc.	Av. Rank	Av. Acc.	Av. Rank	Av. Acc.	Av. Rank	Av. Acc.	Av. Rank
S5	61.8†	4.67†	63.4	3.25	60.1	4.83	56.7	5.33
LRU	61.7†	4.50†	60.9	5.58	61.0	5.00	60.8	4.17
S6	62.0†	5.00†	58.4	6.50	61.4	4.00	58.9	5.17
Mamba	58.6†	6.67†	59.5	4.83	60.6	4.00	57.6	6.17
NCDE	60.2†	5.33†	59.6	5.92	58.9	6.42	58.7	5.33
Log-NCDE	**64.3**†	3.00†	**64.2**	3.08	**62.7**	4.67	60.9	4.17
D-SLiCE	61.7‡	4.50‡	61.7	3.83	62.2	3.83	62.2	**2.83**
BD-SLiCE	64.0‡	**2.33**‡	63.7	**3.00**	62.5	**3.25**	**62.5**	**2.83**

Table 3: **Matched input-representation ablation.** Each row replaces BD-SLiCE in Table 2 by BD-SLiCE with the given input representation and re-ranks all eight models on each dataset. Entries give its average dataset rank and, in parentheses, its position among the eight models (= denotes a tie). Avg. averages the dataset ranks over the three dropping levels.

Input representation	30% drop	70% drop	95% drop	Avg.
Event-increment embedding (ours)	**3.00** (1)	3.25 (1)	**2.83** (= 1)	**3.03**
Cubic Hermite interpolation	3.58 (3)	**3.00** (1)	3.58 (2)	3.39
Gaussian-process imputation	4.33 (4)	4.50 (5)	3.58 (2)	4.14
Global-mean imputation	5.17 (5)	3.75 (2)	3.33 (2)	4.08

cubic interpolation with backward differences [Kidger et al., 2020]. We test our proposed embedding using a Log-NCDE [Walker et al., 2024], diagonal Log-SLiCE (D-SLiCE), and block-diagonal Log-SLiCE (BD-SLiCE) [Walker et al., 2025]. For path-based models, we keep the original query windows fixed and recompute interval log-signatures from the remaining observations. Full details and per-dataset results are given in Appendix C.3.

Table 2 shows that the continuous-time models using our embedding are competitive across all dropping levels. Log-NCDE gives the best macro-average accuracy at no drop, 30% drop, and 70% drop. BD-SLiCE gives the best macro-average accuracy at 95% drop and the best or tied-best average rank at every dropping level. Its accuracy changes from 64.0% with no dropping to 62.5% with 95% dropping, indicating robustness under severe sparsification. To isolate the input representation, Table 3 repeats the evaluation with BD-SLiCE while changing only its input representation, holding the architecture, hyperparameters, data splits, and seeds fixed. Our embedding is the only representation that places BD-SLiCE first or joint first at every dropping level, and it attains the best mean rank. Cubic Hermite interpolation attains a slightly better rank at 70% drop but not at 30% or 95%, and the imputation-based alternatives are less competitive overall.

4 Conclusion

This paper argues that continuous-time representations should preserve observed data rather than attempt to reconstruct unobserved behaviour. This principle is formalised through a general universality transfer result. For Hausdorff data and model input spaces, any continuous injective embedding transfers universality on compact sets from the model input space back to the data space. Thus, for a universal downstream model, an embedding need only be continuous and injective. Guided by this, we introduced a faithful embedding for irregular and partially observed streams tailored to Log-NCDEs. The construction records observations as tensor-algebra factors, composes them over arbitrary query intervals, and maps the result to interval log-signatures for use with the Log-ODE method. This approach avoids interpolation of observed variables, supports online prediction, decouples input observation times from output query times, and is computable in parallel over intervals and parallel-in-time within intervals. Experiments on synthetic controlled systems and UEA classification

tasks show that the resulting models are accurate, efficient, and robust under irregularity, asynchrony, and severe input dropping.

Acknowledgements

Benjamin Walker is supported by UK Research and Innovation (UKRI) through the Engineering and Physical Sciences Research Council (EPSRC) via Programme Grant [Grant No. UKRI1010: High order mathematical and computational infrastructure for streamed data that enhance contemporary generative and large language models] and CIMDA@Oxford, part of the AIR@InnoHK initiative funded by the Innovation and Technology Commission, HKSAR Government. Lingyi Yang is supported by the Maths Degrees for the Future Programme Grant MDFF 2024-08 awarded by the London Mathematical Society. Terry Lyons is supported by UK Research and Innovation (UKRI) through the Engineering and Physical Sciences Research Council (EPSRC) via Programme Grants [Grant No. UKRI1010: High order mathematical and computational infrastructure for streamed data that enhance contemporary generative and large language models], [Grant No. EP/S026347/1: Unparameterised multi-model data, high order signatures and the mathematics of data science], [Grant No. EP/Y028872/1: Mathematical Foundations of Intelligence: An Erlangen Programme for AI], and the UKRI AI for Science award [Grant No. UKRI2385: Creating Foundational Benchmarks for AI in Physical and Biological Complexity]. Terry Lyons is also supported by The Alan Turing Institute under the Defence and Security Programme (funded by the UK Government) and through the provision of research facilities; by the UK Government; and through CIMDA@Oxford, part of the AIR@InnoHK initiative funded by the Innovation and Technology Commission, HKSAR Government. The authors acknowledge support from His Majesty's Government in the development of this research. The authors would like to acknowledge the use of the University of Oxford Advanced Research Computing (ARC) facility in carrying out this work. https://doi.org/10.5281/zenodo.22558. Continuous Intelligence Limited is supported by SPRIND through their Next Frontier AI Challenge.

References

Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The UEA multivariate time series classification archive, 2018. *arXiv preprint, arXiv:1811.00075*, 2018.

Torben Berndt, Benjamin Walker, Tiexin Qin, Jan Stühmer, and Andrey Kormilitzin. Permutation equivariant neural controlled differential equations for dynamic graph representation learning. In *Proceedings of the 39th Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Alexandre Bloch, Benjamin Walker, Joël Mouterde, Sam Morley, Samuel N. Cohen, and Terry Lyons. Extending SSMs with the exponentially weighted signature. *arXiv preprint arXiv:2603.19198*, 2026.

Horatio Boedihardjo, Xi Geng, Terry Lyons, and Danyu Yang. The signature of a rough path: Uniqueness. *Advances in Mathematics*, 293:720–737, 2016.

Jared Boyer, T. Konstantin Rusch, and Daniela Rus. Learning to dissipate energy in oscillatory state-space models. *arXiv preprint arXiv:2505.12171*, 2025.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

Fabienne Castell and Jessica Gaines. An efficient approximation method for stochastic differential equations by means of the exponential Lie series. *Mathematics and Computers in Simulation*, 38(1):13–19, 1995.

Mihriban Ceylan, Anna P. Kwossek, and David J. Prömel. Universal approximation with signatures of non-geometric rough paths, 2026. URL https://arxiv.org/abs/2602.05898.

Cristian Challu, Kin G Olivares, Boris N Oreshkin, Federico Garza Ramirez, Max Mergenthaler Canseco, and Artur Dubrawski. N-HiTS: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 6989–6997, 2023.

Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific Reports*, 8(1):6085, 2018.

Kuo Tsai Chen. Iterated integrals and exponential homomorphisms. *Proceedings of the London Mathematical Society*, s3-4(1):502–512, 1954.

Kuo Tsai Chen. Integration of paths, geometric invariants and a generalized Baker–Hausdorff formula. *Annals of Mathematics*, 65(1), 1957.

Yuqi Chen, Kan Ren, Yansen Wang, Yuchen Fang, Weiwei Sun, and Dongsheng Li. ContiFormer: Continuous-time transformer for irregular time series modeling. In *Advances in Neural Information Processing Systems*, 2023.

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

Ilya Chevyrev, Emilio Ferrucci, Darrick Lee, Terry Lyons, Harald Oberhauser, and Nikolas Tapia. Orthogonal polynomials on path-space, 2026. URL https://arxiv.org/abs/2602.18808.

Jeongwhan Choi and Noseong Park. Graph neural rough differential equations for traffic forecasting. *ACM Transactions on Intelligent Systems and Technology*, 14(4), 2023.

Jeongwhan Choi, Hwangyong Choi, Jeehyun Hwang, and Noseong Park. Graph neural controlled differential equations for traffic forecasting. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI-22)*, 2022.

Nicola Muca Cirone and Cristopher Salvi. Rough kernel hedging. *arXiv preprint arXiv:2501.09683*, 2025.

Nicola Muca Cirone, Antonio Orvieto, Benjamin Walker, Cristopher Salvi, and Terry Lyons. Theoretical foundations of deep selective state-space models. In *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

Thomas Cochrane, Peter Foster, Varun Chhabra, Maud Lemercier, Terry Lyons, and Cristopher Salvi. SK-Tree: a systematic malware detection algorithm on streaming trees via the signature kernel. In *2021 IEEE International Conference on Cyber Security and Resilience (CSR)*, pages 35–40. IEEE, 2021.

Samuel N. Cohen, Silvia Lui, Will Malpass, Giulia Mantoan, Lars Nesheim, Áureo de Paula, Andrew Reeves, Craig Scott, Emma Small, and Lingyi Yang. Nowcasting with signature methods. *arXiv preprint arXiv:2305.10256*, 2023.

Samuel N Cohen, James Foster, Peter Foster, Hang Lou, Terry Lyons, Sam Morley, James Morrill, Hao Ni, Edward Palmer, Bo Wang, Yue Wu, Lingyi Yang, and Weixin Yang. Subtle variation in sepsis-III definitions markedly influences predictive performance within and across methods. *Scientific Reports*, 14(1920), 2024.

Christa Cuchiero, Francesca Primavera, and Sara Svaluto-Ferro. Universal approximation theorems for continuous functions of càdlàg paths and Lévy-type signature models, 2023. URL https://arxiv.org/abs/2208.02293.

Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. GRU-ODE-Bayes: Continuous modeling of sporadically-observed time series. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

Mónika Farsang, Radu Grosu, Ramin Hasani, and Daniela Rus. Parallelization of non-linear state-space models: Scaling up liquid-resistance liquid-capacitance networks for efficient sequence modeling. In *Proceedings of the 39th Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Peter K Friz and Nicolas B Victoir. *Multidimensional stochastic processes as rough paths: theory and applications*, volume 120. Cambridge University Press, 2010.

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

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *Proceedings of The 10th International Conference on Learning Representations (ICLR)*, 2022.

B. Hambly and T. Lyons. Uniqueness for the signature of a path of bounded variation and the reduced path group. *Annals of Mathematics*, 171:109–167, 2010.

W. P. M. H. Heemels, K. H. Johansson, and P. Tabuada. An introduction to event-triggered and self-triggered control. In *Proceedings of the 51st IEEE Conference on Decision and Control*, 2012.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Christian Holberg and Cristopher Salvi. Exact gradients for stochastic spiking neural networks driven by rough signals. In *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

Blanka Horvath, Maud Lemercier, Chong Liu, Terry Lyons, and Cristopher Salvi. Optimal stopping via distribution regression: A higher rank signature approach. *arXiv preprint arXiv:2304.01479*, 2023.

Andraž Jelinčič, Jiajie Tao, William F. Turner, Thomas Cass, James Foster, and Hao Ni. Generative modeling of Lévy area for high order SDE simulation. *SIAM Journal on Mathematics of Data Science*, 7(4):1541–1567, 2025. doi: 10.1137/23M161077X. URL https://doi.org/10.1137/23M161077X.

Sheo Yon Jhin, Jaehoon Lee, Minju Jo, Seungji Kook, Jinsung Jeon, Jihyeon Hyeong, Jayoung Kim, and Noseong Park. EXIT: Extrapolation and interpolation-based neural controlled differential equations for time-series classification and forecasting. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 3102–3112, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450390965. doi: 10.1145/3485447.3512030. URL https://doi.org/10.1145/3485447.3512030.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural Controlled Differential Equations for Irregular Time Series. In *Proceedings of the 34th Conference on Neural Information Processing System (NeurIPS)*, 2020.

Franz J. Király and Harald Oberhauser. Kernels for sequentially ordered data. *Journal of Machine Learning Research*, 20(31):1–45, 2019.

Kangjun Lee, Minha Kim, Youngho Jun, and Simon S. Woo. GDFlow: Anomaly detection with NCDE-based normalizing flow for advanced driver assistance system. *arXiv preprint arXiv:2409.05346*, 2024.

Maud Lemercier, Cristopher Salvi, Thomas Cass, Edwin V Bonilla, Theodoros Damoulas, and Terry Lyons. SigGPDE: Scaling sparse gaussian processes on sequential data. In *International Conference on Machine Learning (ICML)*. PMLR, 2021a.

Maud Lemercier, Cristopher Salvi, Theodoros Damoulas, Edwin Bonilla, and Terry Lyons. Distribution regression for sequential data. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR, 2021b.

Daniel Levin, Terry Lyons, and Hao Ni. Learning from the past, predicting the statistics for the future, learning an evolving system. *arXiv preprint arXiv:1309.0260*, 2016.

Seonkyu Lim, Jeongwhan Choi, Noseong Park, Sang-Ha Yoon, ShinHyuck Kang, Young-Min Kim, and Hyunjoong Kang. Bridging dynamic factor models and neural controlled differential equations for nowcasting GDP. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 2024.

Terry Lyons. Rough paths, signatures and the modelling of functions on streams. In *Proceedings of the International Congress of Mathematicians (ICM)*, volume 4, 2014.

Terry J Lyons, Michael Caruana, and Thierry Lévy. *Differential Equations Driven by Rough Paths*. Springer, 2007.

Georg Manten, Cecilia Casolo, Emilio Ferrucci, Søren Wengel Mogensen, Cristopher Salvi, and Niki Kilbertus. Signature kernel conditional independence tests in causal discovery for stochastic processes. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*, 2025.

Farokh Marvasti, editor. *Nonuniform Sampling: Theory and Practice*. Kluwer Academic/Plenum Publishers, New York, 2001.

Andrew McLeod and Terry Lyons. Signature methods in machine learning. *EMS Surveys in Mathematical Sciences*, 2025.

P J Moore, T J Lyons, J Gallacher, and Alzheimer's Disease Neuroimaging Initiative. Using path signatures to predict a diagnosis of Alzheimer's disease. *PLoS One*, 14(9), 2019.

Fernando Moreno-Pino, Álvaro Arroyo, Harrison Waldon, Xiaowen Dong, and Álvaro Cartea. Rough transformers: Lightweight and continuous time series modelling through signature patching. In *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

Sam Morley and Terry Lyons. Roughpy: Streaming data is rarely smooth. In *Proceedings of the 23rd Python in Science Conference*, 2024.

James Morrill, Andrey Kormilitzin, Alejo Nevado-Holgado, Sumanth Swaminathan, Sam Howison, and Terry Lyons. The signature-based model for early detection of sepsis from electronic health records in the intensive care unit. *Computing in Cardiology (CinC)*, 2019.

James Morrill, Patrick Kidger, Lingyi Yang, and Terry Lyons. On the choice of interpolation scheme for neural CDEs. *Transactions on Machine Learning Research*, 2022(9), 2022.

Sajad Movahedi, Felix Sarnthein, Nicola Muca Cirone, and Antonio Orvieto. Fixed-point RNNs: Interpolating from diagonal to dense. In *Proceedings of the 39th Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Antonio Orvieto, Samuel L. Smith, Albert Gu, Anushan Fernando, Çaglar Gülçehre, Rázvan Pascanu, and Soham De. Resurrecting recurrent neural networks for long sequences. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

Barak A. Pearlmutter. Learning state space trajectories in recurrent neural networks. *Neural Computation*, 1(2): 263–269, 1989.

Imanol Perez Arribas. Derivatives pricing using signature payoffs. *arXiv preprint arXiv:1809.09466*, 2018.

Imanol Perez Arribas, Guy M Goodwin, John R Geddes, Terry Lyons, and Kate E A Saunders. A signature-based machine learning model for distinguishing bipolar disorder and borderline personality disorder. *Transl Psychiatry*, 8(1), 2018.

Imanol Perez Arribas, Cristopher Salvi, and Lukasz Szpruch. Sig-SDEs model for quantitative finance. In *ACM International Conference on AI in Finance*, 2020.

Tiexin Qin, Benjamin Walker, Terry Lyons, Hong Yan, and Haoliang Li. Learning dynamic graph embeddings with neural controlled differential equations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Jeremy F. Reizenstein and Benjamin Graham. Algorithm 1004: The iisignature library: Efficient calculation of iterated-integral signatures and log signatures. *ACM Transactions on Mathematical Software*, 46(1):1–21, March 2020. doi: 10.1145/3371237.

C. Reutenauer. *Free Lie Algebras*. London Mathematical Society Monographs. Clarendon Press, 1993.

R. Rico-Martínez, K. Krischer, I. G. Kevrekidis, M. C. Kube, and J. L. Hudson. Discrete-vs. continuous-time nonlinear signal processing of Cu electrodissolution data. *Chemical Engineering Communications*, 118(1): 25–48, 1992.

Yulia Rubanova, Tian Qi Chen, and David Kristjanson Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

T. Konstantin Rusch and Daniela Rus. Oscillatory state-space models. In *Proceedings of The 13th International Conference on Learning Representations (ICLR)*, 2025.

Cristopher Salvi. *Rough paths, kernels, differential equations and an algebra of functions on streams*. PhD thesis, University of Oxford, 2021.

Cristopher Salvi, Thomas Cass, James Foster, Terry Lyons, and Weixin Yang. The signature kernel is the solution of a Goursat PDE. *SIAM Journal on Mathematics of Data Science*, 3(3):873–899, 2021.

Cristopher Salvi, Joscha Diehl, Terry Lyons, Rosa Preiss, and Jeremy Reizenstein. A structure theorem for streamed information. *Journal of Algebra*, 634:911–938, 2023.

Nabeel Seedat, Fergus Imrie, Alexis Bellot, Zhaozhi Qian, and Mihaela van der Schaar. Continuous-time modeling of counterfactual outcomes using neural controlled differential equations. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, 2022.

Daniil Shmelev and Cristopher Salvi. Sparse signature coefficient recovery via kernels. *arXiv preprint arXiv:2412.08579*, 2024.

Satya Narayan Shukla and Benjamin M. Marlin. Interpolation-prediction networks for irregularly sampled time series. In *International Conference on Learning Representations*, 2019.

Satya Narayan Shukla and Benjamin M. Marlin. Multi-time attention networks for irregularly sampled time series. In *International Conference on Learning Representations*, 2021.

Jimmy T. H. Smith, Andrew Warrington, and Scott W. Linderman. Simplified state space layers for sequence modeling. In *Proceedings of The 11th International Conference on Learning Representations (ICLR)*, 2023.

M. H. Stone. The generalized Weierstrass approximation theorem. *Mathematics Magazine*, 21(4):167–184, 1948.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

Andre Vauvelle, Paidi Creed, and Spiros Denaxas. Neural-signature methods for structured EHR prediction. *BMC Medical Informatics and Decision Making*, 22(1), 2022.

Benjamin Walker, Andrew D. McLeod, Tiexin Qin, Yichuan Cheng, Haoliang Li, and Terry Lyons. Log neural controlled differential equations: The Lie brackets make a difference. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

Benjamin Walker, Lingyi Yang, Nicola Muca Cirone, Cristopher Salvi, and Terry Lyons. Structured linear CDEs: Maximally expressive and parallel-in-time sequence models. In *Proceedings of the 39th Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Bo Wang, Maria Liakata, Hao Ni, Terry Lyons, Alejo J. Nevado-Holgado, and Kate Saunders. A Path Signature Approach for Speech Emotion Recognition. In *Proc. Interspeech 2019*, pages 1661–1665, 2019.

Jun Wang, Wenjie Du, Yiyuan Yang, Linglong Qian, Wei Cao, Keli Zhang, Wenjia Wang, Yuxuan Liang, and Qingsong Wen. Deep learning for multivariate time series imputation: A survey. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, 2025.

Ernst Witt. Treue darstellung liescher ringe. *Journal für die reine und angewandte Mathematik*, 177:152–160, 1937.

Sihang Zeng, Lucas Jing Liu, Jun Wen, Meliha Yetisgen, Ruth Etzioni, and Gang Luo. Trajsurv: Learning continuous latent trajectories from electronic health records for trustworthy survival prediction. In *Proceedings of the Machine Learning for Healthcare (MLHC) Conference*, 2025.

A Additional Details on Faithful Embeddings

This appendix gives the technical details behind the faithful-embedding criterion used in Section 2.1. The key point is that a continuous injective embedding preserves compact sets up to homeomorphism. Consequently, a universal function class on the embedded space can be pulled back to a universal function class on the original data space. We first prove the transfer result, then state the corresponding non-injectivity obstruction, and finally record the standard signature-universality consequence.

A.1 Universality transfer

Lemma 3 (Universality Transfer)**.** *Let $\mathcal{X}$ and $\mathcal{P}$ be Hausdorff topological spaces, and let $\varphi : \mathcal{X} \to \mathcal{P}$ be a continuous injection. Suppose $\mathcal{F} \subseteq C(\mathcal{P}; \mathbb{R})$ is a family of continuous real-valued functions such that for every compact $K \subseteq \mathcal{P}$, the restrictions $\mathcal{F}|_K$ are dense in $C(K; \mathbb{R})$ in the uniform topology. Then for every compact $K_0 \subseteq \mathcal{X}$, the restrictions $(\mathcal{F} \circ \varphi)|_{K_0}$ are dense in $C(K_0; \mathbb{R})$ in the uniform topology, where*

$$\mathcal{F} \circ \varphi := \{f \circ \varphi : f \in \mathcal{F}\}.$$

Proof. Let $K_0 \subseteq \mathcal{X}$ be compact, let $u \in C(K_0; \mathbb{R})$, and let $\varepsilon > 0$. Set $K := \varphi(K_0) \subseteq \mathcal{P}$. The continuous image of a compact set is compact, so K is compact. The restriction $\varphi|_{K_0} : K_0 \to K$ is a continuous bijection from a compact space to a Hausdorff space, hence a homeomorphism. In particular, $(\varphi|_{K_0})^{-1}$ is continuous.

Let

$$g := u \circ (\varphi|_{K_0})^{-1} : K \to \mathbb{R}.$$

Then $g \in C(K; \mathbb{R})$. By assumption, $\mathcal{F}|_K$ is dense in $C(K; \mathbb{R})$ in the uniform topology, so there exists $h \in \mathcal{F}$ with

$$\sup_{p \in K} |g(p) - h(p)| < \varepsilon.$$

For every $x \in K_0$,

$$g(\varphi(x)) = u((\varphi|_{K_0})^{-1}(\varphi(x))) = u(x).$$

Therefore

$$\sup_{x \in K_0} |u(x) - h(\varphi(x))| = \sup_{x \in K_0} |g(\varphi(x)) - h(\varphi(x))| = \sup_{p \in K} |g(p) - h(p)| < \varepsilon.$$

Since $h \circ \varphi \in \mathcal{F} \circ \varphi$, this proves that $(\mathcal{F} \circ \varphi)|_{K_0}$ is dense in $C(K_0; \mathbb{R})$. □

The proof uses only the standard fact that a continuous bijection from a compact space to a Hausdorff space is a homeomorphism. In the main text, $\mathcal{X}$ is the original data space, $\mathcal{P}$ is the model input space, and φ is the embedding.

A.2 Non-injectivity obstruction

We now record the complementary failure mode. If an embedding identifies two distinct data objects, then every downstream function of the embedding identifies them as well. Thus no downstream class can approximate continuous functions that separate those data objects.

Proposition 4 (Non-injective implies non-universal)**.** *Let $\mathcal{X}$ and $\mathcal{P}$ be Hausdorff topological spaces, let $\varphi : \mathcal{X} \to \mathcal{P}$ be continuous, and let $\mathcal{F} \subseteq C(\mathcal{P}, \mathbb{R})$. If φ is not injective on a compact $K_0 \subseteq \mathcal{X}$, then $(\mathcal{F} \circ \varphi)|_{K_0}$ is not dense in $C(K_0; \mathbb{R})$ in the uniform topology, where*

$$\mathcal{F} \circ \varphi := \{f \circ \varphi : f \in \mathcal{F}\}.$$

Proof. By assumption, there exist distinct $x_1, x_2 \in K_0$ with $\varphi(x_1) = \varphi(x_2)$. Since K_0 is compact Hausdorff, it is normal. Hence there exists $f \in C(K_0; \mathbb{R})$ with $f(x_1) \neq f(x_2)$. Set

$$\varepsilon = \frac{|f(x_1) - f(x_2)|}{3}.$$

Let $h \in \mathcal{F}$ be arbitrary. Since $\varphi(x_1) = \varphi(x_2)$, we have $h(\varphi(x_1)) = h(\varphi(x_2))$, and therefore

$$|f(x_1) - h(\varphi(x_1))| + |f(x_2) - h(\varphi(x_2))| \geq |f(x_1) - f(x_2)| = 3\varepsilon.$$

Hence one of the two summands is at least $3\varepsilon/2 > \varepsilon$, so

$$\sup_{x \in K_0} |f(x) - h(\varphi(x))| > \varepsilon.$$

Since $h \in \mathcal{F}$ was arbitrary, no element of $\mathcal{F} \circ \varphi$ approximates f to within ε on K_0. Thus $(\mathcal{F} \circ \varphi)|_{K_0}$ is not dense in $C(K_0; \mathbb{R})$. □

Lemma 3 and Proposition 4 give the criterion used in the main text. Continuous and injective embeddings preserve universality on compact sets. Non-injective embeddings cannot preserve universality, because they discard information about the original data.

A.3 Signature representations of data

Classical signature universality for paths has the same form as Lemma 3, after restricting to a class of paths on which the signature map is injective, or equivalently after quotienting by tree-like equivalence [Hambly and Lyons, 2010, Boedihardjo et al., 2016]. Recent work has also extended signature universality results to stochastic settings [Cuchiero et al., 2023, Ceylan et al., 2026, Chevyrev et al., 2026]. The following corollary records the same schema with paths replaced by data streams. Any continuous injection from data into the tensor algebra inherits universality through linear functionals.

Corollary 5 (Universality of signature representations of data)**.** *Let $d, m, N \in \mathbb{N}$, let*

$$T((\mathbb{R}^m)) = \prod_{k=0}^{\infty} (\mathbb{R}^m)^{\otimes k}$$

be equipped with the product topology, and let $G \subseteq T((\mathbb{R}^m))$ be the group-like elements with the subspace topology. Let $\iota : (\mathbb{R}^d)^N \to G$ be a continuous injection. Then for every compact $K_0 \subseteq (\mathbb{R}^d)^N$, every continuous $f : K_0 \to \mathbb{R}$, and every $\varepsilon > 0$, there exists a continuous linear functional ℓ on $T((\mathbb{R}^m))$ depending on only finitely many tensor-algebra coordinates with

$$\sup_{x \in K_0} \left| f(x) - \ell(\iota(x)) \right| < \varepsilon.$$

Proof. Apply Lemma 3 with $\mathcal{X} = (\mathbb{R}^d)^N$, $\mathcal{P} = G$, $\varphi = \iota$, and $\mathcal{F}$ the family of restrictions to G of finitely supported continuous linear functionals on $T((\mathbb{R}^m))$. The density hypothesis on $\mathcal{F}$ follows from the shuffle identity and the real Stone–Weierstrass theorem, since these functionals contain the constants, separate points of G, and form an algebra on G [Reutenauer, 1993, Stone, 1948, Lyons, 2014]. □

This is the same mechanism used in signature-based universality results. The only additional requirement is that the representation must separate the original data objects. For paths, this is the injectivity of the signature after removing tree-like equivalence. For data streams, it is the injectivity of the chosen embedding ι.

Since linear functionals of signatures are universal on compact path classes once injectivity is established, truncated signature features provide expressive representations for statistical learning tasks [Levin et al., 2016]. Signature-based kernels extend this viewpoint to kernel methods on sequential data [Király and Oberhauser, 2019], while subsequent work develops learning directly with the full infinite-dimensional signature kernel [Salvi et al., 2021, Salvi, 2021, Lemercier et al., 2021b,a, Manten et al., 2025].

Signatures have a number of other useful properties, including that their iterated-integral coordinates are unchanged by translations of the path and by increasing reparameterisations of time [Chen, 1954]. Together with universality, these properties have led to their use across a wide range of domains, including healthcare [Perez Arribas et al., 2018, Moore et al., 2019, Wang et al., 2019, Morrill et al., 2019, Cohen et al., 2024, Vauvelle et al., 2022], quantitative finance [Perez Arribas, 2018, Perez Arribas et al., 2020, Horvath et al., 2023, Cirone and Salvi, 2025, Cohen et al., 2023], information theory [Salvi et al., 2023, Shmelev and Salvi, 2024], cybersecurity [Cochrane et al., 2021], and computational neuroscience [Holberg and Salvi, 2024].

Neural controlled differential equations extend this framework to learnable continuous-time models driven by paths, with applications including counterfactual prediction in healthcare [Seedat et al., 2022], economic nowcasting [Lim et al., 2024], survival prediction [Zeng et al., 2025], anomaly detection for driving assistance [Lee et al., 2024], dynamic graph learning [Qin et al., 2025, Berndt et al., 2025], and traffic forecasting [Choi et al., 2022, Choi and Park, 2023].

A.4 Proof of faithfulness

We give a formal version of Proposition 2. The input consists of a continuously observed path C and an observation stream x. We assume throughout that C contains physical time as one of its channels.

Let $\tau \in \{1, \dots, d_{\text{cont}}\}$ denote the continuous coordinate corresponding to physical time. Thus

$$C_t^\tau = t, \qquad t \in [0, T].$$

Let $\mathcal{C}_{\text{BV}}^\tau$ be the space of bounded-variation paths

$$C : [0, T] \to \mathbb{R}^{d_{\text{cont}}}$$

with $C_t^\tau = t$, $\forall t \in [0, T]$, equipped with the uniform topology.

For a fixed event pattern

$$\sigma = (s_0, \dots, s_{m-1}),$$

let $\mathcal{X}_\sigma$ be the corresponding stream component. On this component, the number of events and the observed sets are fixed, while the event times and local event records vary. We equip each $\mathcal{X}_\sigma$ with the standard product topology. We equip

$$\mathcal{X} = \coprod_\sigma \mathcal{X}_\sigma$$

with the disjoint-union topology. The joint input space is

$$\mathcal{D} = \mathcal{C}^\tau_{\mathrm{BV}} \times \mathcal{X}.$$

For a stream with m events, set

$$I_m = [0, T + m + 1].$$

The target path space is

$$\mathcal{P} = \coprod_{m \geq 0} C(I_m, \mathbb{R}^{d_X}),$$

with the disjoint-union topology and the uniform topology on each component. The realised paths below lie in the bounded-variation subspace of $\mathcal{P}$.

We write o_i for the local record inserted at event i. For each observed set s, assume that local records are included through a continuous injective map

$$\iota_s : o \mapsto \Delta \in \mathfrak{L}^N(\mathbb{R}^{d_X}).$$

We assume that Δ uses only the first $2d_{\mathrm{disc}}$ coordinates. We also assume that its degree-one count coordinates satisfy

$$\pi_1(\Delta)^{d_{\mathrm{disc}}+k} = \mathbf{1}_{\{k \in s\}}.$$

Thus the count coordinates identify the observed set. For higher-order local information, the inserted object must be expressed in log-signature coordinates. If a local segment has a known truncated signature H, then the element stored as the event factor is

$$\Delta = \log H.$$

Assume that each local factor has a continuous bounded-variation realisation. For each observed set s, assume there is a continuous map

$$o \mapsto \eta_s(o)$$

from local records into bounded-variation paths

$$\eta_s(o) : [0, 1] \to \mathbb{R}^{d_X},$$

with the uniform topology, such that

$$\eta_s(o)(0) = 0, \qquad S^N_{0,1}(\eta_s(o)) = \exp(\iota_s(o)).$$

We assume that $\eta_s(o)$ uses only the first $2d_{\mathrm{disc}}$ coordinates. For event i, write

$$\Delta_i = \iota_{s_i}(o_i), \qquad \eta_i = \eta_{s_i}(o_i).$$

The default construction in the main text is the special case

$$\Delta_i = \sum_{k \in s_i} \left[\left(x^k_i - x^k_{\mathrm{prev}(i,k)} \right) e_k + e_{d_{\mathrm{disc}}+k} \right],$$

with all higher-order components equal to zero. In this degree-one case, one may take

$$\eta_i(u) = u\, \Delta_i.$$

On each fixed event-pattern component, replacing raw observed values by these value increments is a homeomorphism. The inverse is given by cumulative summation along each channel, starting from the fixed base point.

We now construct the path realisation. Fix

$$(C, x) \in \mathcal{D}, \qquad x = ((t_0, s_0, o_0), \ldots, (t_{m-1}, s_{m-1}, o_{m-1})).$$

Let $\widetilde{C}$ denote C embedded in the final d_{cont} coordinates of $\mathbb{R}^{d_X}$.

First define the initial auxiliary segment. For $u \in I_m$, set

$$\rho_*(u) = \min\{1, \max\{0, u\}\}.$$

This segment encodes the initial continuous value $\widetilde{C}_0$. Its truncated signature is

$$S^N_{0,1}(u \mapsto u\widetilde{C}_0) = \exp(\widetilde{C}_0) = B_C.$$

For each event $i = 0, \dots, m-1$, define

$$\rho_i^x(u) = \min\{1, \max\{0, u - 1 - t_i - i\}\}.$$

The function ρ_i^x is zero before the event segment for event i. It increases linearly from zero to one during that event segment and remains at one after that event segment.

Define

$$q_x(u) = u - \rho_*(u) - \sum_{i=0}^{m-1} \rho_i^x(u).$$

The function q_x is used only to write the path explicitly. Since C contains physical time as a channel, q_x becomes the physical-time coordinate of the realised path.

Define

$$\Psi(x, C)(u) = \rho_*(u)\widetilde{C}_0 + \Big(\widetilde{C}(q_x(u)) - \widetilde{C}_0\Big) + \sum_{i=0}^{m-1} \eta_i\big(\rho_i^x(u)\big).$$

This is a path in $\mathbb{R}^{d_X}$. At $u = 0$, the path starts at zero. During the initial auxiliary segment, the path moves from 0 to $\widetilde{C}_0$ while physical time is held fixed at 0. After the initial auxiliary segment, the continuously observed coordinates follow $\widetilde{C}(q_x(u))$. During event segments, the continuously observed coordinates are held fixed while the event path η_i is traversed.

Since $C_t^\tau = t$, and since the event paths η_i use only the first $2d_{\mathrm{disc}}$ coordinates, the coordinate $2d_{\mathrm{disc}} + \tau$ of $\Psi(x, C)$ is

$$\Psi(x, C)^{2d_{\mathrm{disc}}+\tau}(u) = q_x(u).$$

Thus physical time is already one of the channels of $\Psi(x, C)$. During the initial auxiliary segment and during event segments, this time channel is constant. During ordinary parts of the path, this time channel increases at unit speed.

For a realised path $\Psi = \Psi(x, C)$, define

$$a_\Psi(t) = \inf\Big\{u \in I_m : \Psi^{2d_{\mathrm{disc}}+\tau}(u) = t\Big\}.$$

For $\beta < T$, define

$$b_\Psi(\beta) = a_\Psi(\beta).$$

For $\beta = T$, define

$$b_\Psi(T) = T + m + 1.$$

This final-endpoint convention includes events at T in the final interval.

For a physical-time interval $[\alpha, \beta)$, define

$$S^N_{\alpha,\beta}(\Psi) = S^N_{a_\Psi(\alpha), b_\Psi(\beta)}(\Psi).$$

This is the signature of the ordered auxiliary-time portion of Ψ whose time channel lies in $[\alpha, \beta)$, with events at T included when $\beta = T$.

Proposition 6 (Formal faithfulness)**.** *Assume the observation-count coordinates are included, C contains time as a channel, and $N \geq 1$. Under the local-factor assumptions above, the map*

$$(C, x) \mapsto \Psi(x, C)$$

from $\mathcal{D}$ to $\mathcal{P}$ is continuous and injective. Moreover, for every $0 \leq \alpha < \beta \leq T$,

$$\log S^N_{\alpha,\beta}(\Psi(x, C)) = \Phi_{\alpha,\beta}(x, C).$$

Proof. We first prove the interval-signature identity. Let $[\alpha, \beta)$ be a physical-time interval, with the same final-endpoint convention as in the main text. Suppose the events assigned to this interval are

$$\alpha \leq t_{i_1} < \cdots < t_{i_r} < \beta,$$

with events at T included when $\beta = T$.

If $\alpha = 0$, then the selected auxiliary-time portion begins with the initial segment $u \mapsto u\widetilde{C}_0$. The signature of this initial segment is

$$B_C = \exp(\widetilde{C}_0).$$

The selected portion is then the ordered concatenation of this initial segment, the continuous pieces, and the event pieces. By Chen's identity,

$$S^N_{0,\beta}(\Psi(x, C)) = B_C \otimes \Gamma_{0,t_{i_1}} \otimes \exp(\Delta_{i_1}) \otimes \Gamma_{t_{i_1},t_{i_2}} \otimes \exp(\Delta_{i_2}) \otimes \cdots \otimes \exp(\Delta_{i_r}) \otimes \Gamma_{t_{i_r},\beta}.$$

This is exactly $G_{0,\beta}(x, C)$.

If $\alpha > 0$, then the selected auxiliary-time portion does not include the initial segment. It is the ordered concatenation of the continuous pieces and event pieces

$$\widetilde{C}|_{[\alpha,t_{i_1}]}, \eta_{i_1}, \widetilde{C}|_{[t_{i_1},t_{i_2}]}, \eta_{i_2}, \ldots, \eta_{i_r}, \widetilde{C}|_{[t_{i_r},\beta]}.$$

Chen's identity gives

$$S^N_{\alpha,\beta}(\Psi(x, C)) = \Gamma_{\alpha,t_{i_1}} \otimes \exp(\Delta_{i_1}) \otimes \Gamma_{t_{i_1},t_{i_2}} \otimes \exp(\Delta_{i_2}) \otimes \cdots \otimes \exp(\Delta_{i_r}) \otimes \Gamma_{t_{i_r},\beta}.$$

This is exactly $G_{\alpha,\beta}(x, C)$.

Therefore, for every $0 \le \alpha < \beta \le T$,

$$\log S^N_{\alpha,\beta}(\Psi(x, C)) = \log G_{\alpha,\beta}(x, C) = \Phi_{\alpha,\beta}(x, C).$$

We now prove continuity. It is enough to work on a fixed event-pattern component. On such a component, the number of events and observed sets are fixed. Let

$$(C^{(j)}, x^{(j)}) \to (C, x)$$

in this component. Then

$$C^{(j)} \to C$$

uniformly, and for every event i,

$$t_i^{(j)} \to t_i, \qquad o_i^{(j)} \to o_i.$$

The map

$$a \mapsto \min\{1, \max\{0, u - a\}\}$$

is 1-Lipschitz uniformly in u. Hence

$$\|\rho_i^{x^{(j)}} - \rho_i^x\|_\infty \le |t_i^{(j)} - t_i| \to 0.$$

It follows that

$$\|q_{x^{(j)}} - q_x\|_\infty \le \sum_{i=0}^{m-1} \|\rho_i^{x^{(j)}} - \rho_i^x\|_\infty \to 0.$$

We control the initial continuous segment first. Since

$$C_0^{(j)} \to C_0,$$

we have

$$\sup_{u \in I_m} \left\| \rho_*(u)\widetilde{C}_0^{(j)} - \rho_*(u)\widetilde{C}_0 \right\| \to 0.$$

We next control the continuously observed part after subtracting its initial value. Define

$$F_C(t) = \widetilde{C}(t) - \widetilde{C}_0.$$

Then

$$F_{C^{(j)}} \to F_C$$

uniformly. Since F_C is continuous on the compact interval $[0, T]$, it is uniformly continuous. Therefore

$$\begin{aligned} &\sup_{u \in I_m} \|F_{C^{(j)}}(q_{x^{(j)}}(u)) - F_C(q_x(u))\| \\ &\quad \le \|F_{C^{(j)}} - F_C\|_\infty + \sup_{u \in I_m} \|F_C(q_{x^{(j)}}(u)) - F_C(q_x(u))\|. \end{aligned}$$

The first term tends to zero by uniform convergence. The second term tends to zero by uniform continuity of F_C and the uniform convergence of $q_{x^{(j)}}$ to q_x.

The local realisations depend continuously on the local records. Thus

$$\eta_i^{(j)} \to \eta_i$$

uniformly on $[0, 1]$. Since each η_i is uniformly continuous on $[0, 1]$, we have

$$\eta_i^{(j)}(\rho_i^{x^{(j)}}(\cdot)) \to \eta_i(\rho_i^x(\cdot))$$

uniformly. Combining the initial segment, the continuous part, and the event part gives

$$\|\Psi(x^{(j)}, C^{(j)}) - \Psi(x, C)\|_\infty \to 0.$$

Since the argument holds on each fixed event-pattern component, and since both the stream space and the target path space use disjoint-union topologies, the map

$$(C, x) \mapsto \Psi(x, C)$$

is continuous.

We now prove injectivity. The target component of $\mathcal{P}$ determines the number of events m. The time coordinate

$$u \mapsto \Psi(x, C)^{2d_{\mathrm{disc}}+\tau}(u)$$

is the function $q_x(u)$. Let

$$d_x(u) = u - q_x(u).$$

The initial auxiliary segment is the part where $0 < d_x(u) < 1$. For event i, the event segment is the closure of the set

$$\{u \in I_m : 1 + i < d_x(u) < 2 + i\}.$$

On this event segment, the time coordinate is constant and equal to t_i. Thus the realised path recovers all event times.

The final d_{cont} coordinates of $\Psi(x, C)$ recover C. Indeed, after the initial auxiliary segment, the final d_{cont} coordinates are exactly

$$C(q_x(u)).$$

The event paths η_i use only the first $2d_{\mathrm{disc}}$ coordinates. The time coordinate q_x is surjective onto $[0, T]$. Therefore, for every $t \in [0, T]$, choosing any $u \geq 1$ with $q_x(u) = t$ recovers

$$C(t)$$

from the final d_{cont} coordinates of $\Psi(x, C)(u)$.

Now consider the event segment corresponding to event i. On this segment, the first $2d_{\mathrm{disc}}$ coordinates of $\Psi(x, C)$ traverse η_i, up to translation by the previously completed event pieces. Signatures are invariant under translation. Therefore the signature of the event segment is

$$S^N_{0,1}(\eta_i) = \exp(\Delta_i).$$

Taking the truncated logarithm recovers

$$\Delta_i.$$

The degree-one count coordinates of Δ_i recover the observed set s_i. Since

$$\Delta_i = \iota_{s_i}(o_i)$$

and ι_{s_i} is injective, the local event record o_i is recovered by applying the inverse of ι_{s_i} on its image.

In the default construction, the degree-one value coordinates of Δ_i are

$$x^k_i - x^k_{\mathrm{prev}(i,k)}, \qquad k \in s_i.$$

Together with the recovered event order and observed sets, these increments recover the raw observed values recursively from the fixed base point. If

$$i_1 < i_2 < \cdots < i_r$$

are the events at which channel k is observed, then

$$x^k_{i_\ell} = \sum_{j=1}^{\ell} \left(x^k_{i_j} - x^k_{\mathrm{prev}(i_j,k)} \right).$$

Thus the full input (C, x) is uniquely determined by $\Psi(x, C)$. The path realisation is injective. □

A.5 Invariance of the Embedding

The path realisation in Proposition 6 is faithful when count coordinates are included and C contains time as a channel. We describe two natural relaxations, dropping the count coordinates, and dropping the continuous variables, in terms of the invariances they induce in the corresponding log-signatures over intervals.

First suppose the count coordinates $e_{d_{\mathrm{disc}}+k}$ are dropped from the driving space, so that $d_X = d_{\mathrm{disc}} + d_{\mathrm{cont}}$ and the discrete increment becomes

$$\delta_i \;=\; \sum_{k \in s_i} \big(x^k_i - x^k_{\mathrm{prev}(i,k)}\big) e_k \;\in\; \mathbb{R}^{d_X}.$$

An observation event at time t_i with $x^k_i = x^k_{\mathrm{prev}(i,k)}$ for every $k \in s_i$ contributes $\delta_i = 0$, hence $E_i = \exp(0) = \mathbf{1}$, and the chronological tensor product $G_{\alpha,\beta}(x)$ is unaffected by the event's presence or absence.

Define the equivalence relation $\sim_{\mathrm{val}}$ on $\mathcal{X}$ by $x \sim_{\mathrm{val}} x'$ if and only if x' is obtained from x by a finite sequence of insertions or removals of value-preserving observation events. Then $G_{\alpha,\beta}(x) = G_{\alpha,\beta}(x')$ for every sub-interval $[\alpha, \beta)$ whenever $x \sim_{\mathrm{val}} x'$, so φ_π without count coordinates is an invariant of $\mathcal{X}/\sim_{\mathrm{val}}$ rather than of $\mathcal{X}$. The same holds true for the realised path embedding.

This is benign for streams in which value repeats carry no information. For continuous-valued sensor data, the set of streams with exact value repeats is measure-zero under any non-degenerate observation distribution, and a stream and its value-preserving insertions are typically indistinguishable for prediction purposes. For event-based discrete data, where value repeats are common and informative, for example a counter incremented at irregular times, or a binary state oscillating between two values, the equivalence class loses information that downstream tasks need, and count coordinates should be retained.

Now suppose the continuous variables are absent, so $\mathbb{R}^{d_X} = \mathbb{R}^{2d_{\mathrm{disc}}}$ (or $\mathbb{R}^{d_{\mathrm{disc}}}$ if count coordinates are also dropped). Then there are no gap factors $\Gamma_{u,v}$, and the interval signature reduces to a product of event factors,

$$G_{[\alpha,\beta)}(x) \;=\; \prod_{i\,:\,t_i \in [\alpha,\beta)} E_i.$$

The product depends only on which events fall in the interval and their order, not on the precise event times t_i themselves. An empty interval contributes $\mathbf{1}$ regardless of length.

Define the equivalence relation $\sim_{\mathrm{time}}$ on $\mathcal{X}$ by $x \sim_{\mathrm{time}} x'$ if and only if x' is obtained from x by changing event times $\{t_i\}$ to any other strictly increasing sequence in $[0, T]$, with the same observed channel sets and values. Then $G_{\alpha,\beta}(x) = G_{\alpha,\beta}(x')$ for every sub-interval $[\alpha, \beta)$ that contains the same event indices in x and x'. For the partitioned encoding φ_π, this means $\varphi_\pi(x) = \varphi_\pi(x')$ whenever $x' \sim_{\mathrm{time}} x$ and the event-to-interval assignments are preserved. For the realised path embedding, this means that the path consists only of auxiliary event segments. The gaps between events carry no path increment, so changing the event times only changes the allocation of these auxiliary segments to physical-time intervals; if that allocation is preserved, the interval signatures of the realised path are unchanged.

The timestamps remain in the data stream as metadata, used to specify query intervals and to drive downstream continuous-time models, but they no longer enter the encoding itself. This is appropriate when the modelling task is concerned with the sequence of events and their values, not with the precise inter-event timings; for example, in classification problems where temporal regularity is irrelevant, or in event-driven control problems where only the event ordering matters. When precise timing information is needed, physical time should be included as a continuous variable.

B Limitations

This section clarifies the main assumptions and scope of the proposed embedding and experiments.

The first limitation concerns the faithful embedding itself. Proposition 6 assumes that observation-count coordinates are included and that physical time is present as a continuous channel. The count coordinates increase the dimension of the driving path, from $d_{\mathrm{disc}} + d_{\mathrm{cont}}$ to $2d_{\mathrm{disc}} + d_{\mathrm{cont}}$. This increases the size of the truncated tensor and log-signature representations. For truncation depth N, the number of tensor coordinates up to depth N scales as $\sum_{k=1}^{N} d_X^k = O(d_X^N)$. Although the log-signature dimension is smaller, scaling like $O(d_X^N/N)$ by the Witt formula [Witt, 1937, Reizenstein and Graham, 2020], this still grows exponentially in N. Thus enforcing exact injectivity through auxiliary channels can increase computational cost, especially for high-dimensional streams or high truncation depths. As discussed in Appendix A.5, omitting count coordinates or physical time is a modelling choice that intentionally introduces invariances. This can be appropriate when repeated observations or exact timestamps are not informative, but it can discard information in event-based or timing-sensitive tasks.

The second limitation concerns the interpretation of universality. The universality result is a compact-set approximation result and should not be interpreted as a guarantee of sample efficiency, optimisation success, or improved performance on every dataset. It shows that a continuous injective embedding does not obstruct approximation by a universal downstream model. Furthermore, faithfulness concerns the realised path embedding; interval summaries at a fixed truncation depth and partition need not be injective. The empirical behaviour still depends on the model class, truncation depth, optimisation procedure, data distribution, and availability of informative observations.

The final limitation concerns empirical scope. Our experiments cover synthetic controlled systems and six UEA time-series classification datasets, including robustness to input dropping. They do not exhaust all irregular, asynchronous, sparse, or event-based data regimes. The Brownian experiment assumes that second-order information, such as Lévy area, is available or precomputed.

C Further details on experiments

Single runs for all experiments can be completed on a 24GB NVIDIA RTX 4090 GPU in less than 24 hours. We use the following publicly available datasets, libraries, and baseline models:

- **UEA Multivariate Time Series Classification Archive** Bagnall et al. [2018]. License: GPL-3.0. URL: https://www.timeseriesclassification.com/, https://github.com/time-series-machine-learning/tsml-repo
- **S5** Smith et al. [2023]. License: Apache 2.0. URL: https://github.com/lindermanlab/S5
- **LRU** Orvieto et al. [2023]. License: MIT. URL: https://github.com/NicolasZucchet/minimal-LRU
- **S6/Mamba** Gu and Dao [2024]. License: Apache 2.0. URL: https://github.com/state-spaces/mamba
- **Log-NCDE** Walker et al. [2024]. License: CC-BY-4.0 URL: https://github.com/Benjamin-Walker/log-neural-cdes
- **JAX** Bradbury et al. [2018]. License: Apache 2.0. URL: https://github.com/google/jax
- **PyTorch** Paszke et al. [2019]. License: BSD Style, see here https://github.com/pytorch/pytorch?tab=License-1-ov-file URL: https://github.com/pytorch/pytorch

C.1 Synthetic coupled data

This section gives further experimental details for the synthetic coupled experiment described in Section 3.2. Each sample is generated on the time interval $[0, T]$ with terminal time $T = 10$ and observed channels $d = 2$. We restate the underlying continuous-time signal for ease of reference

$$z(t) = \omega t + \phi, \qquad x_i(t) = A_i \sin(z(t) + \delta_i), \qquad i = 1, \dots, d.$$

For each sample, the shared angular frequency and global phase are drawn as

$$\omega \sim \mathcal{U}(0.8, 1.6), \qquad \phi \sim \mathcal{U}(0, 2\pi),$$

and the channel-specific parameters are sampled independently as

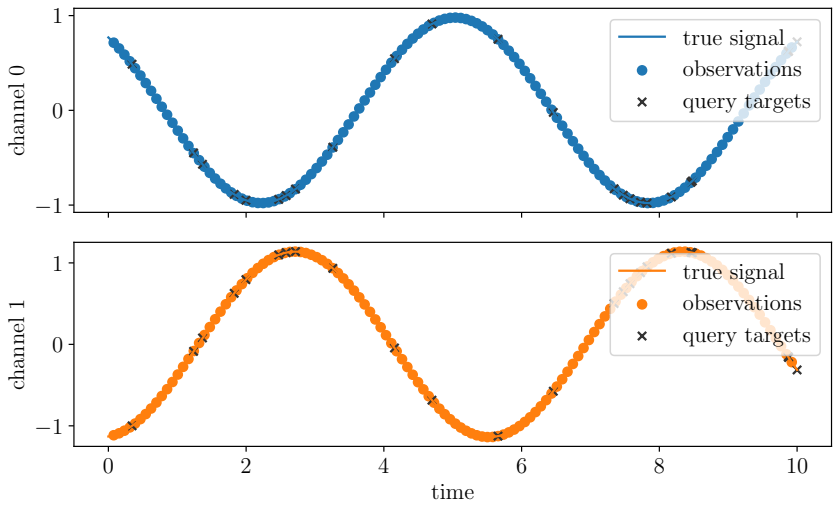
$$A_i \sim \mathcal{U}(0.7, 1.3), \qquad \delta_i \sim \mathcal{U}(0, 2\pi).$$

The shared latent phase $z(t)$ induces dependence between the two channels, while the channel-specific amplitudes and phase offsets prevent the task from reducing to identical copies of a one-dimensional signal. No observation noise is added in this experiment.

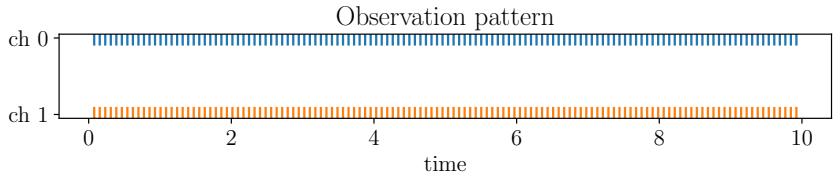
The observations are taken in $(0, T)$ and the target values are computed directly from the underlying continuous dynamics rather than from the observed samples. We evaluate four observation regimes. In the *synchronous regular* regime, both channels are observed at the same fixed regular grid of 128 interior time points. In the *synchronous irregular* regime, 128 observation times are sampled independently and uniformly from $(0, T)$, then sorted, and shared by both channels. In the *asynchronous irregular* regime, each channel has its own independently sampled irregular grid of 128 observation times, generated independently and uniformly from $(0, T)$. Therefore, the synchronous regular, synchronous irregular, and asynchronous irregular regimes have the same number of observations per channel. They differ only in the regularity and synchronisation of their observation grids. Finally, in the *asynchronous sparse* regime, channels are observed asynchronously but at substantially different frequencies. The first channel is dense and second channel is sparse. Dense channels use Poisson sampling with rates $\lambda_i \sim \mathcal{U}(12, 20)$, while sparse channels use rates $\lambda_i \sim \mathcal{U}(2, 4)$. For the asynchronous sparse regime, if a channel has fewer than two observations, then two observation times are sampled uniformly from $(0, T)$ as a fallback. This guarantees that every channel is observed at least twice.

Figures 6–9 show samples from the four regimes. The synchronous regular case (Figure 6) has a fixed common input grid, the synchronous irregular case (Figure 7) has an irregular but shared input grid, the asynchronous irregular case (Figure 8) has independent irregular grids, and the asynchronous sparse case (Figure 9) combines asynchronous observation with a substantial difference in observation frequency across the channels.

We convert the observation stream to the increment representation. In this experiment we do not include observation-count coordinates in the model input, since the sampled sinusoid observations almost surely do not contain exact repeated event values. Physical time is appended as a continuously observed channel. Thus the model input has $d + 1 = 3$ channels: two discretely observed value-increment channels and one continuously observed time channel.

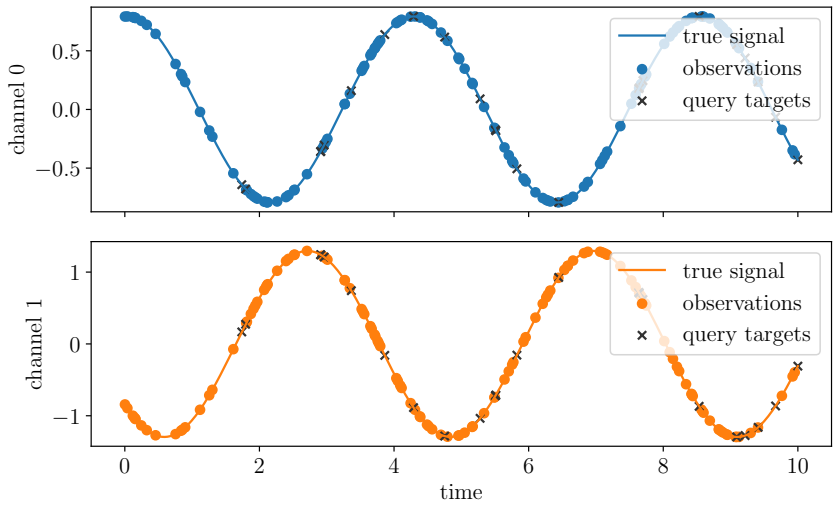


(a) True underlying signal, observation, and query targets.

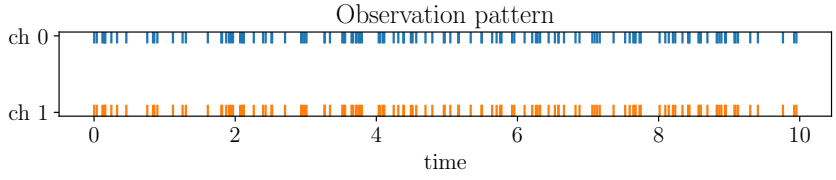


(b) Observation pattern of the two channels.

Figure 6: **Observation patterns and signal example for synchronous regular regime.**

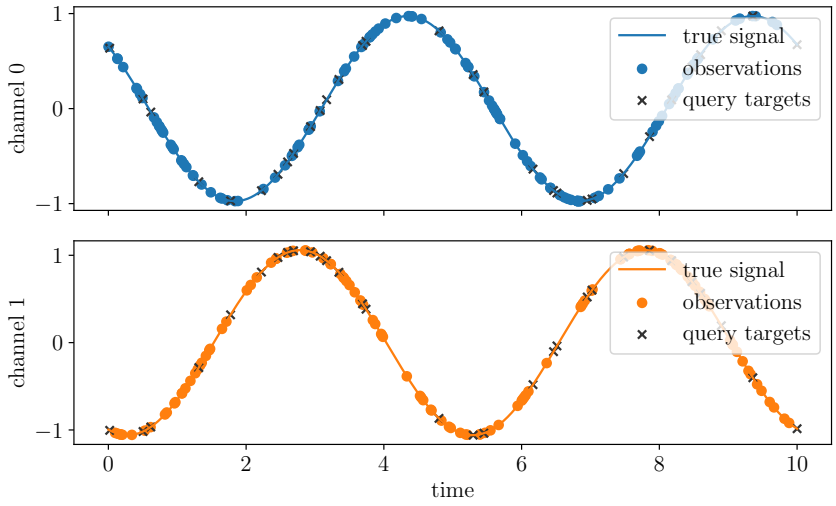


(a) True underlying signal, observation, and query targets.

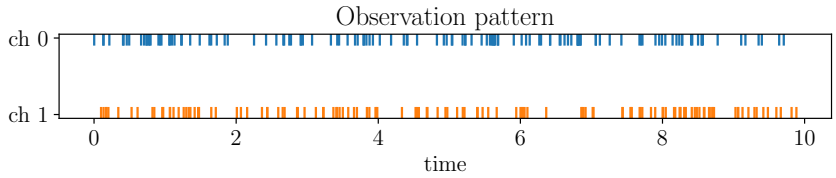


(b) Observation pattern of the two channels.

Figure 7: **Observation patterns and signal example for synchronous irregular regime.**

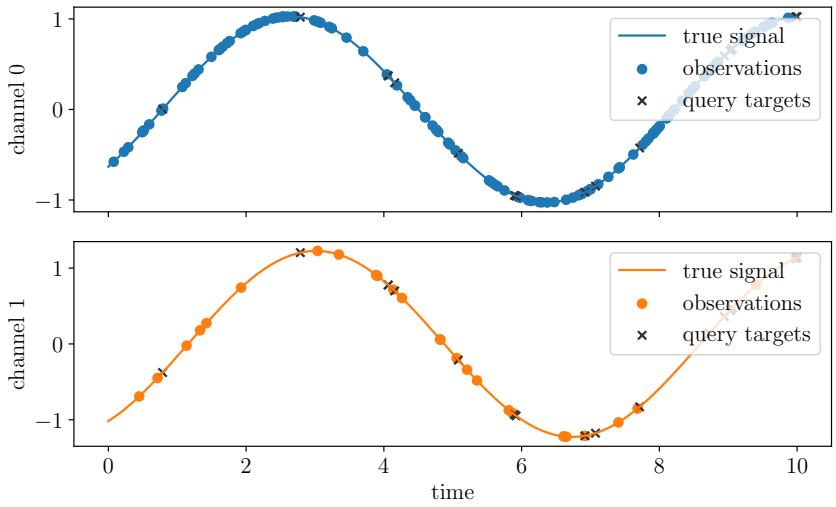


(a) True underlying signal, observation, and query targets.

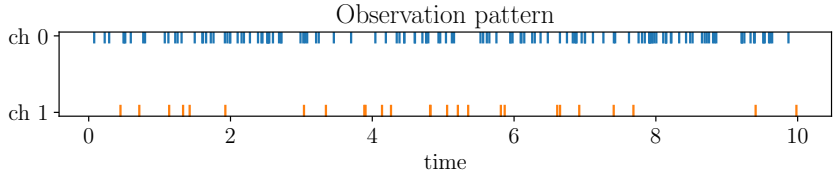


(b) Observation pattern of the two channels.

Figure 8: **Observation patterns and signal example for asynchronous irregular regime.**



(a) True underlying signal, observation, and query targets.



(b) Observation pattern of the two channels.

Figure 9: **Observation patterns and signal example for asynchronous sparse regime.**

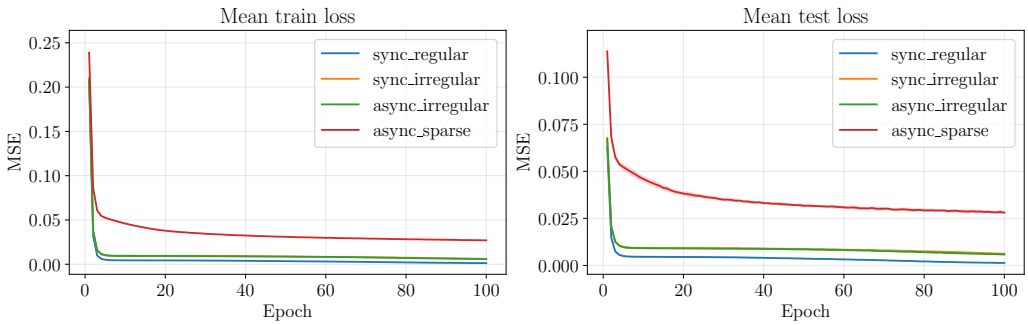


Figure 10: **Training/test loss over epochs for the synthetic coupled task.**

For each sequence, we also store the first observed value in each channel. This vector is passed to the input-dependent initialisation of the controlled differential equation after a $\tanh$ transformation. This gives the model access to the starting observed level of each channel while the subsequent path records increments from one observation to the next.

For every sample, we draw an output query partition independently of the observation times with the number of query intervals m sampled uniformly from $\{16, 17, \dots, 32\}$. We then sample $m - 1$ interior query points uniformly from $(0, T)$, sort them, and append the endpoints 0 and T, giving

$$0 = q_0 < q_1 < \cdots < q_m = T.$$

The model receives the log-signature of the observed stream over each interval $[q_k, q_{k+1})$ and is trained to predict the true endpoint value

$$y_k = x(q_{k+1}) \in \mathbb{R}^2.$$

Since q_{k+1} is not required to be an observation time, the task is therefore not next-observation prediction. It tests whether the interval summaries allow the model to predict the underlying state on an output grid chosen independently of the input sampling grid.

We train one model on data generated from each observation regime. Each training set contains 2048 samples and each test set contains 512 samples. We use batch size 32 and train for 100 epochs. The model is a block diagonal log-SLiCE with hidden dimension 64, log-signature depth 2, block size 8. A linear decoder is applied to the hidden state at each query interval to produce the two-dimensional prediction. We optimise the mean squared error over all non-padded query intervals using Adam with learning rate 10^{-3} and global gradient clipping at norm 1.

Because the number of query intervals varies between samples, interval targets are padded within each batch. The loss is computed only on valid intervals. If $\hat{y}_{b,k}$ and $y_{b,k}$ denote the prediction and target for sample b and interval k, and $m_{b,k} \in \{0, 1\}$ is the padding mask, the training objective is

$$\mathcal{L} = \frac{\sum_{b,k} m_{b,k} \, \|\hat{y}_{b,k} - y_{b,k}\|_2^2}{d_{\text{out}} \sum_{b,k} m_{b,k}},$$

where $d_{\text{out}} = 2$ is the number of predicted channels.

After training one model on each regime, we evaluate each trained model on test sets from all four regimes. This gives the cross-regime matrix seen in Figure 4, which reports the mean and standard deviation of the test MSE across five repeats, scaled by 10^{-2}.

This cross-regime protocol is designed to test sampling-regime robustness. We fixed the observation budget to be 128 observations per channel for the synchronous regular, synchronous irregular, and asynchronous irregular regimes, which allows us to isolate the effects of sampling irregularity and asynchrony from differences in observation frequency. A model trained only on synchronous regular data can exploit the fixed sampling structure seen during training, and therefore achieves very low error on the matching test regime. However, this does not transfer well to irregular, asynchronous, or sparse observations. In contrast, the asynchronous sparse regime is the hardest training distribution because it exposes the model to asynchronous observations with unequal inter-observation gaps, and severe imbalance in observation frequency. Models trained in this regime therefore generalise more robustly to the other regimes, as shown by the more consistent off-diagonal errors in the final row of Figure 4. The similarity between the synchronous irregular and asynchronous irregular rows is consistent with the training curves in Figure 10. Since these two regimes have the same observation budget per channel, the results suggest that making the two channels asynchronous does not substantially change the difficulty of the task for our model if the data are already irregular.

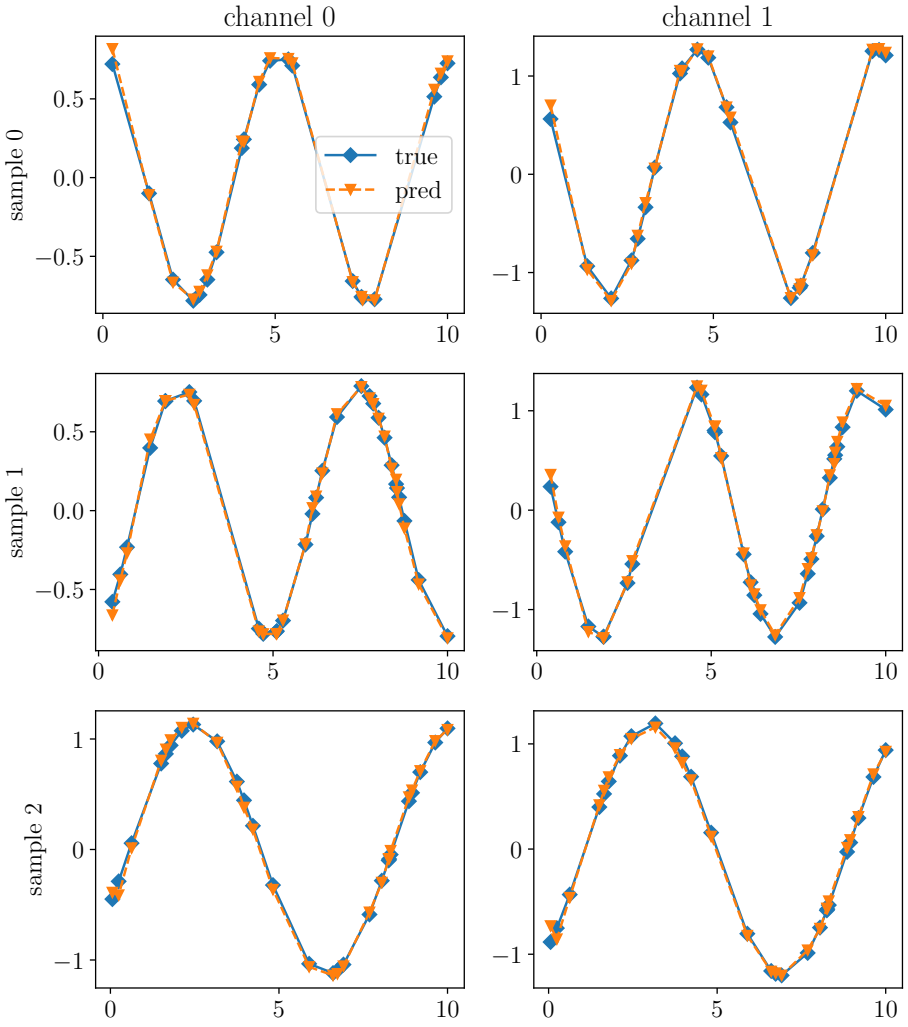


Figure 11: **Prediction examples for model trained on synchronous regular data.**

In Figures 11–14, we see example outputs from trained models under different data sampling regimes. The predictions closely track the sinusoidal targets in most cases. In the asynchronous sparse case, note that the second channel is the sparsely observed channel, hence the larger error for Figure 14.

C.2 Linear system driven by Brownian motion

This section gives the full experimental details for the Brownian controlled system in Section 3.3. This experiment tests whether our Log-SLiCEs can exploit higher-order information when it is present in the data. The solution of the system considered here depends on second-order Lie bracket information.

We use a precomputed ensemble of 4-dimensional Brownian rough paths on the interval $[0, 1]$. Each path is stored on a fine regular grid with 2048 time steps. The stored input at each step is a truncated log-signature increment up to depth 2, and so this is of dimension 10. That is, each Brownian sample is represented by an array of shape 2048×10.

For each Brownian path, we construct targets by simulating a two-dimensional linear controlled system driven by the first two Brownian coordinates, with initial state $X_0 = (1, 0)^\top \in \mathbb{R}^2$.

$$\mathrm{d}X_t = V_1 X_t \circ \mathrm{d}W_t^1 + V_2 X_t \circ \mathrm{d}W_t^2,$$

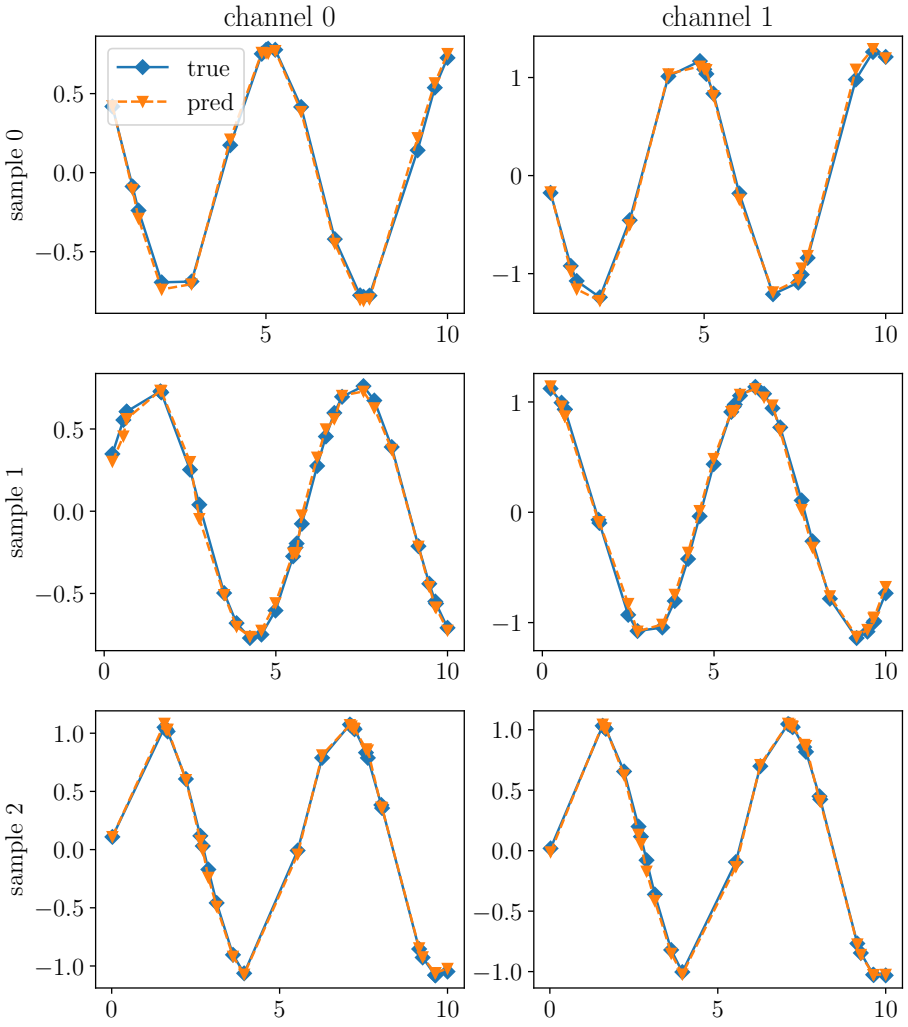


Figure 12: **Prediction examples for model trained on synchronous irregular data.**

The two degree-one vector fields are fixed linear maps

$$V_1 = 0.15 \begin{pmatrix} -0.5 & -1.0 \\ 1.0 & -0.5 \end{pmatrix}, \qquad V_2 = 0.15 \begin{pmatrix} -0.2 & 0.8 \\ 0.3 & -0.7 \end{pmatrix}.$$

The use of non-commuting matrices here ensures that second-order information cannot be removed without changing the solution.

Although these Brownian streams are four-dimensional, our target system here is driven only by the first two Brownian coordinates. The remaining coordinates are therefore irrelevant for the target dynamics and must be ignored by the model.

We loop over a number of query intervals

$$m \in \{2, 4, 8, 16, 32, 64\}.$$

The interval endpoints are aligned with the fine Brownian grid. For a given random seed, we sample $m - 1$ cut indices uniformly without replacement from the interior grid indices, sort them, and append the endpoints 0 and 2048. After rescaling by 2048, this gives a random partition

$$0 = q_0 < q_1 < \cdots < q_m = 1.$$

The same partition is used for all samples within a run. The target associated with interval $[q_k, q_{k+1})$ is the simulated state at the right endpoint,

$$y_k = X(q_{k+1}) \in \mathbb{R}^2.$$

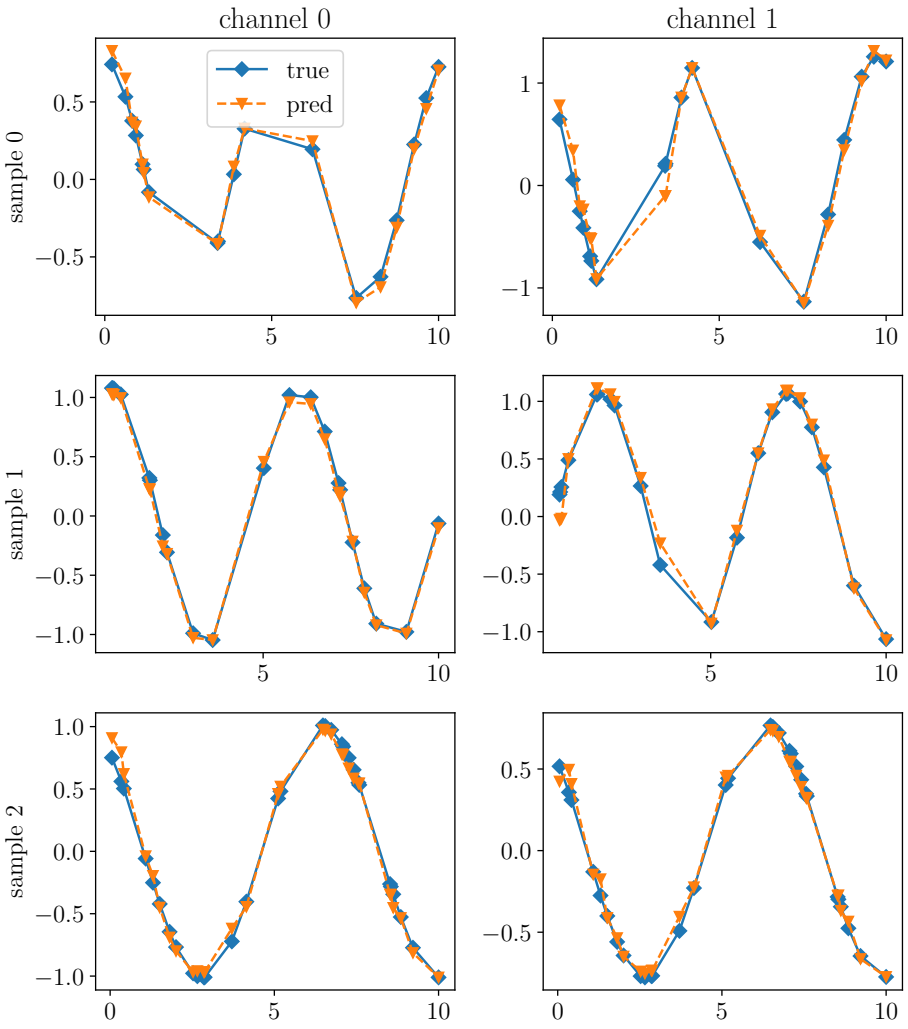


Figure 13: **Prediction examples for model trained on asynchronous irregular data.**

Although the model predicts endpoint values on a query grid that is separate from the fine input grid, the query endpoints are chosen to coincide with fine-grid points to make use of the available second level information.

When only depth-1 information is used, we keep only the first four coordinates, i.e. the Brownian increments themselves. In both cases, physical time is appended as a continuously observed channel. Therefore the model input has $10 + 1 = 11$ channels in the depth-2 setting and $4 + 1 = 5$ channels in the depth-1 setting.

We use block diagonal Log-SLiCE with hidden dimension 64 and block size 8. A linear decoder maps the hidden state at each query interval to the two-dimensional predicted state. Unlike the sinusoid experiment, we do not use input-dependent initialisation in this experiment.

Each run uses 2048 training paths and 512 test paths. We train with batch size 32 for 40 epochs using Adam with learning rate 10^{-3} and global gradient clipping at norm 1. For each value of $m \in \{2, 4, 8, 16, 32, 64\}$, we repeat the experiment over five random seeds. Like the sinusoid task, the training objective is the masked mean squared error over non-padded intervals.

Figure 5 and Table 4 report the test MSE as a function of the number of query intervals. Example output paths can be seen in Figures 15 and 16. As expected, the level-2 model achieves substantially lower error, especially when the number of query intervals is small. As the number of intervals increases, the gap between the two models decreases. When the partition is refined, each interval becomes shorter, and the effect of unobserved second-level terms on each interval becomes smaller. The level-1 model can therefore partially compensate for the lack of second level information by using more frequent updates. However, the level-2 model remains

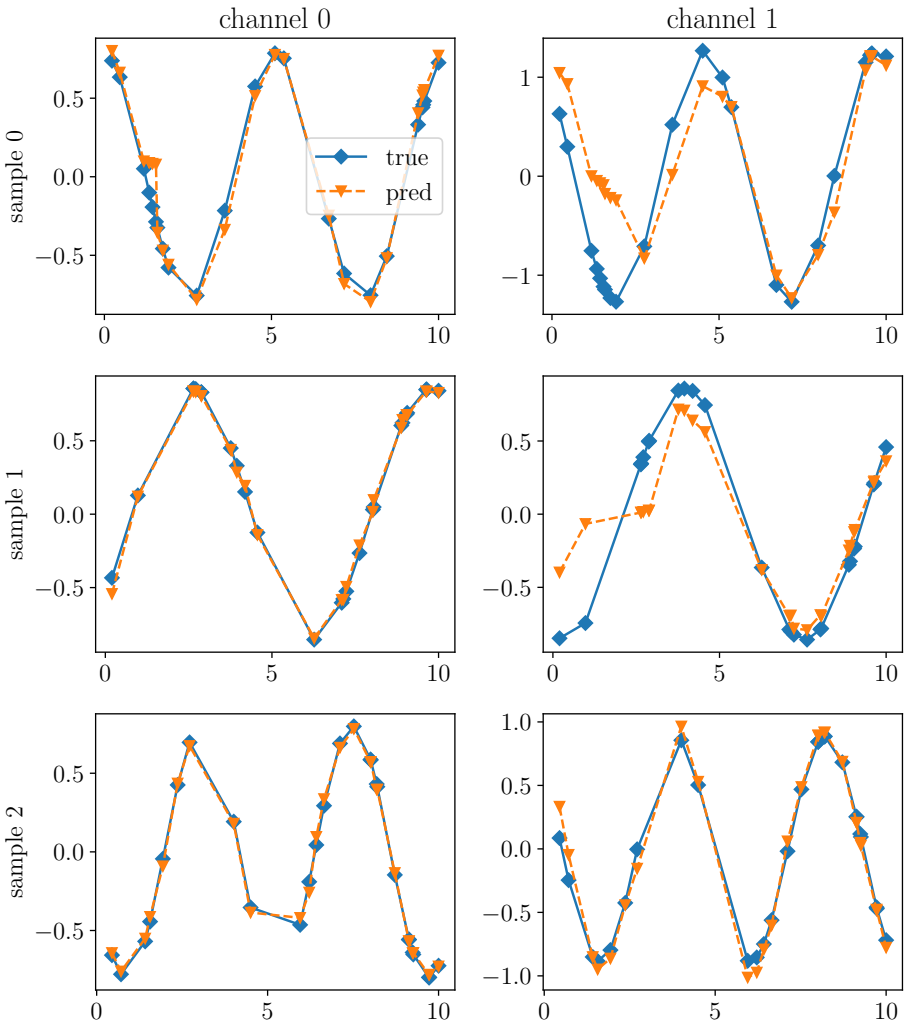


Figure 14: **Prediction examples for model trained on asynchronous sparse data.**

more accurate across the interval counts considered, confirming that our approach can incorporate and exploit higher-order information directly when such information is available.

C.3 UEA experiments

We follow the UEA multivariate time series classification archive (UEA-MTSCA) [Bagnall et al., 2018] protocol of Walker et al. [2024] and the SLiCE protocol of Walker et al. [2025]. The benchmark consists of EigenWorms, EthanolConcentration, Heartbeat, MotorImagery, SelfRegulationSCP1, and SelfRegulationSCP2. As in Walker et al. [2024], duplicated EigenWorms series are removed before splitting. The original train and test cases are pooled and re-split into train, validation, and test sets with relative sizes $70/15/15$. Hyperparameters are selected using validation accuracy on a fixed random split and are then kept fixed for evaluation over five random seeds.

The recurrent baselines are S5, LRU, S6, and Mamba [Smith et al., 2023, Orvieto et al., 2023, Gu and Dao, 2024]. The continuous baselines are an NCDE with Hermite cubic interpolation and backward differences [Kidger et al., 2020], and a Log-NCDE, diagonal SLiCE (D-SLiCE), and block-diagonal SLiCE (BD-SLiCE) using our embedding [Walker et al., 2024, 2025]. The aim of these experiments is not to establish a new state of the art on the UEA-MTSCA benchmark. The selected baselines are chosen to match the protocols of Walker et al. [2024] and Walker et al. [2025], and we keep the selected hyperparameters fixed rather than re-tuning each model for the dropped-input setting. Recent models, including Rough Transformers [Moreno-Pino et al., 2024], oscillatory state-space models such as LinOSS [Rusch and Rus, 2025] and D-LinOSS [Boyer et al., 2025], and

Table 4: **Test MSE versus number of intervals.** All values are scaled by 10^{-5} and reported as mean $\pm$ standard deviation across 5 seeds.

Intervals	Level 1	Level 2
2	6.97 ± 0.93	0.29 ± 0.03
4	3.72 ± 0.86	0.20 ± 0.06
8	2.10 ± 0.46	0.15 ± 0.03
16	0.72 ± 0.09	0.12 ± 0.01
32	0.41 ± 0.04	0.11 ± 0.01
64	0.28 ± 0.02	0.11 ± 0.01

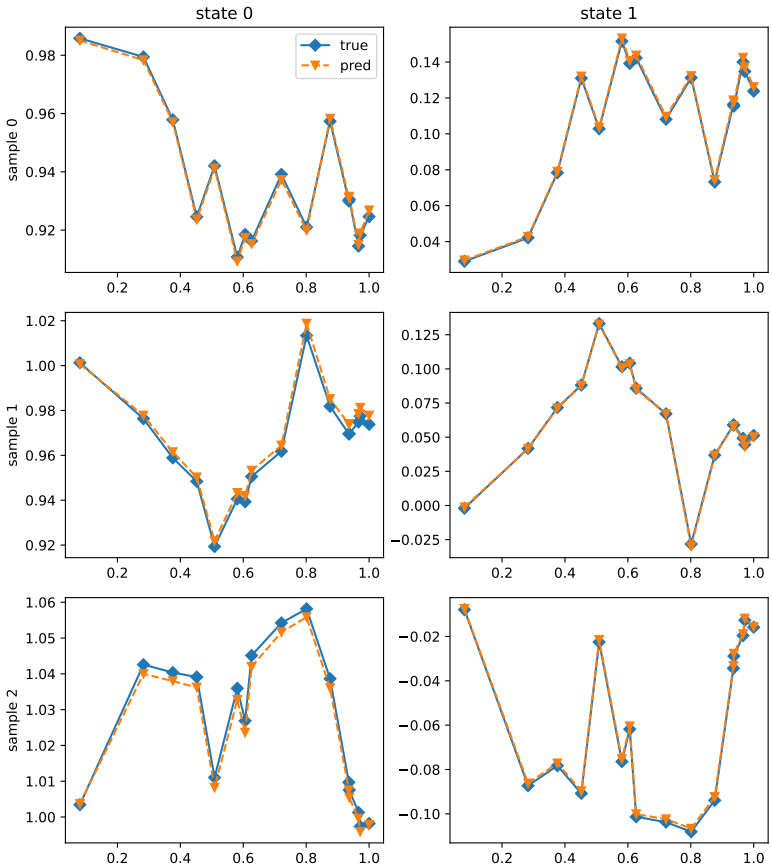


Figure 15: **Prediction of a linear system driven by Brownian motion using a model trained with only increments (level 1) data.** Three examples are taken from the test set with 16 intervals.

nonlinear parallel state-space models such as LrcSSM [Farsang et al., 2025], report higher no-drop accuracies on this six-dataset benchmark. Our comparison instead focuses on the effect of the proposed embedding and interval-level Log-ODE computation under controlled hyperparameters and increasing input sparsification.

Since the task is classification, time is treated as a discretely observed variable when included in the driving path, and there are no continuously observed variables. For D-SLiCE and BD-SLiCE, we use the Log-NCDE hyperparameters selected by Walker et al. [2024], replacing only the non-linear vector field by the corresponding structured linear vector field, following Walker et al. [2025]. For BD-SLiCE, the block size is $b_i = 4$. For the SLiCE models with the Log-ODE method, flows are composed using an associative parallel scan with chunks of size 128. Table 5 gives the Log-NCDE hyperparameters used by Log-NCDE, D-SLiCE, and BD-SLiCE. Table 6 gives the selected hyperparameters for the recurrent baselines. All of these hyperparameters are kept fixed across the no-drop and input-dropping experiments.

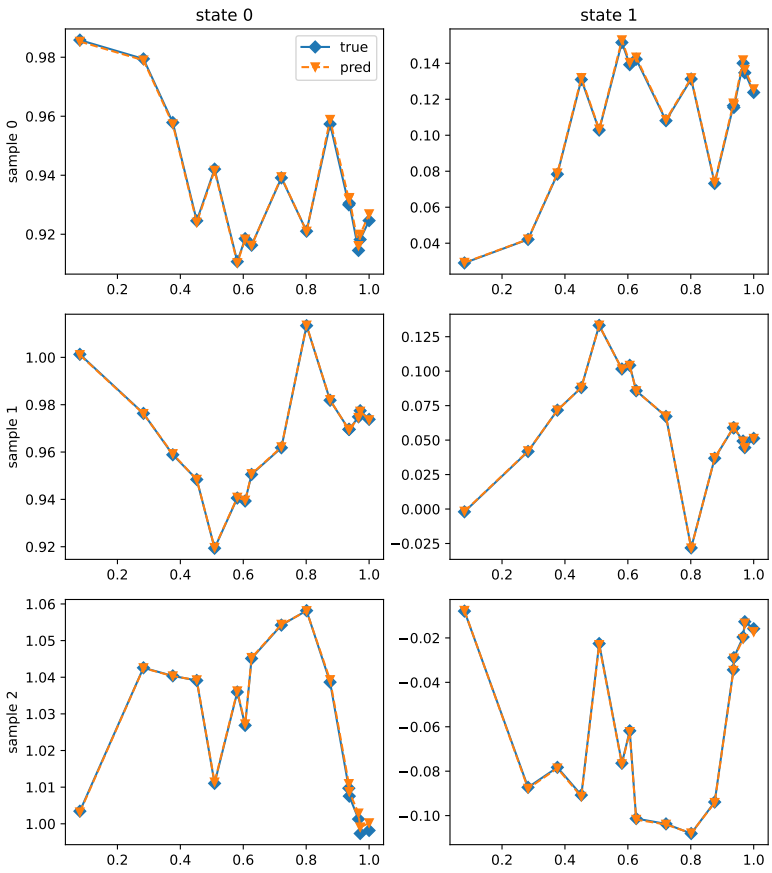


Figure 16: **Prediction of a linear system driven by Brownian motion using a model trained with both increments and area (level 2) data.** Three examples are taken from the test set with 16 intervals.

Table 5: **Log-NCDE hyperparameters for the UEA path models.** D-SLiCE and BD-SLiCE use the same settings as Log-NCDE, replacing only the vector field structure.

Dataset	LR	Include time	Hidden Dimension	Log-ODE		λ
				Depth	Step	
EW	10^{-3}	✓	128	2	12	10^{-3}
EC	10^{-4}	✓	64	1	1	10^{-6}
HB	10^{-3}	✓	16	2	2	10^{-6}
MI	10^{-3}	×	16	2	16	10^{-3}
SCP1	10^{-4}	×	64	2	16	0
SCP2	10^{-4}	×	128	2	4	10^{-3}

All reported accuracies are test accuracies in percent, reported as mean $\pm$ standard deviation over five seeds. Average accuracy is the macro-average over the six datasets. Average rank is computed by ranking the models separately on each dataset at a fixed dropping level and averaging the ranks over datasets. No-drop values marked with † are taken from Walker et al. [2024], and no-drop values marked with ‡ are taken from Walker et al. [2025]. All dropped-input results are produced by our runs using the same selected hyperparameters, without retuning for the drop level.

Table 6: **Selected recurrent-model hyperparameters for the UEA experiments.** The values are those selected by the hyperparameter optimisation of Walker et al. [2024] and are kept fixed in all dropping experiments.

Model	Hyperparameter	EW	EC	HB	MI	SCP1	SCP2
LRU	LR	10^{-3}	10^{-5}	10^{-4}	10^{-3}	10^{-3}	10^{-3}
	Include time	×	✓	✓	×	×	✓
	Hidden dimension	64	64	128	16	64	64
	Layers	4	6	2	6	2	2
	State dimension	64	16	256	256	16	16
S5	LR	10^{-4}	10^{-5}	10^{-3}	10^{-3}	10^{-3}	10^{-4}
	Include time	✓	✓	×	×	×	✓
	Hidden dimension	64	128	128	16	128	16
	Layers	2	2	4	6	6	2
	State dimension	16	16	16	64	16	64
	Init. blocks	8	8	4	4	8	2
S6	LR	10^{-3}	10^{-5}	10^{-3}	10^{-3}	10^{-4}	10^{-3}
	Include time	×	✓	✓	✓	×	×
	Hidden dimension	16	16	16	16	64	16
	Layers	4	4	4	4	2	2
	State dimension	64	16	16	256	16	256
Mamba	LR	10^{-3}	10^{-3}	10^{-4}	10^{-5}	10^{-5}	10^{-4}
	Include time	✓	✓	×	×	×	✓
	Hidden dimension	16	64	64	128	128	64
	Layers	6	4	4	2	2	6
	State dimension	64	256	256	64	16	64
	Conv. dimension	2	4	2	3	4	2
	Expansion factor	1	4	4	1	1	2

C.3.1 EigenWorms

EigenWorms is additionally used for the timing and memory comparison in Table 1 because it is the longest UEA dataset in the benchmark, with $17{,}984$ observations per series. This makes it a direct stress test of whether the Log-ODE method can reduce the cost of long input streams. For this experiment, we keep the hyperparameters fixed to the values selected in the UEA protocol above and compare each model family with and without interval-level Log-ODE updates.

For the nonlinear NCDE family, the non-Log-ODE row corresponds to the standard NCDE baseline and the Log-ODE row corresponds to the Log-NCDE. For the dense linear and block-diagonal linear families, all rows are computed in parallel-in-time using an associative scan. The non-Log-ODE rows use depth-1 Log-ODE updates on intervals aligned with consecutive observations. This is equivalent to solving the CDE exactly with a linearly interpolated observation path. The Log-ODE rows instead use the selected coarser interval partition and higher-order interval log-signatures, then compose the resulting linear flows with the same associative-scan mechanism. Thus the comparison isolates the effect of replacing observation-level updates by interval-level Log-ODE updates, while keeping the architecture family and selected hyperparameters fixed.

Timing and memory are measured on an NVIDIA H100 GPU with batch size 1. The reported time is the wall-clock time per training step averaged over 1000 training steps and the reported memory is peak GPU memory. The EigenWorms results show that the Log-ODE method is beneficial for all three model families. In particular, BD-SLiCE with Log-ODE achieves higher accuracy than the standard NCDE baseline while reducing the time per training step from 26.02 seconds to 0.009 seconds.

C.3.2 Input dropping

The input-dropping experiments test robustness to sparsification of the observed stream. For each drop level $p \in \{0.3, 0.7, 0.95\}$, a random mask is sampled independently for each series and applied to the input before model-specific preprocessing. All models use the same hyperparameters as in the no-drop setting, with no retuning for the drop level. The same drop level is used throughout each run, and results are averaged over five seeds.

For path-based models, dropping observations changes the observed stream but should not change the physical time horizon. We therefore keep the original Log-ODE query partition from the undropped series fixed, and recompute each interval log-signature using the surviving observations whose timestamps fall in that interval. If no discrete observation remains in an interval, the interval still contributes time, if it is included as a channel

Table 7: **Per-dataset test accuracy (mean $\pm$ std over 5 seeds, in %) for the UEA benchmark at no drop, with macro-average accuracy and average dataset rank.** Lower rank is better.

Model	EigenWorms	Ethanol	Heartbeat	MotorImagery	SCP1	SCP2	Avg. Acc.	Avg. Rank
S5	81.1 ± 3.7	24.1 ± 4.3	77.7 ± 5.5	47.7 ± 5.5	$\mathbf{89.9 \pm 4.6}$	50.5 ± 2.6	61.8	4.67
LRU	$\mathbf{87.8 \pm 2.8}$	21.5 ± 2.1	$\mathbf{78.4 \pm 6.7}$	48.4 ± 5.0	82.6 ± 3.4	51.2 ± 3.6	61.7	4.50
S6	85.0 ± 16.1	26.4 ± 6.4	76.5 ± 8.3	51.3 ± 4.7	82.8 ± 2.7	49.8 ± 9.5	62.0	5.00
Mamba	70.9 ± 15.8	27.9 ± 4.5	76.2 ± 3.8	47.7 ± 4.5	80.7 ± 1.4	48.2 ± 3.9	58.6	6.67
NCDE	75.0 ± 3.9	29.9 ± 6.5	73.9 ± 2.6	49.5 ± 2.8	79.8 ± 5.6	53.0 ± 2.8	60.2	5.33
Log-NCDE	85.6 ± 5.1	$\mathbf{34.4 \pm 6.4}$	75.2 ± 4.6	53.7 ± 5.3	83.1 ± 2.8	53.7 ± 4.1	**64.3**	3.00
D-SLiCE	79.4 ± 5.7	27.1 ± 4.6	72.9 ± 5.0	$\mathbf{54.4 \pm 6.3}$	83.5 ± 2.1	53.0 ± 5.8	61.7	4.50
BD-SLiCE	86.1 ± 3.5	28.6 ± 6.4	77.4 ± 5.6	53.0 ± 2.0	84.9 ± 1.9	$\mathbf{54.0 \pm 7.4}$	64.0	**2.33**

Table 8: **Per-dataset test accuracy (mean $\pm$ std over 5 seeds, in %) for the UEA benchmark at 30% drop, with macro-average accuracy and average dataset rank.** Lower rank is better.

Model	EigenWorms	Ethanol	Heartbeat	MotorImagery	SCP1	SCP2	Avg. Acc.	Avg. Rank
S5	79.4 ± 4.2	26.6 ± 1.1	74.8 ± 2.4	$\mathbf{58.9 \pm 2.9}$	$\mathbf{88.7 \pm 1.6}$	51.9 ± 4.4	63.4	3.25
LRU	78.9 ± 6.2	25.3 ± 4.7	70.3 ± 4.6	56.8 ± 5.6	82.6 ± 3.1	51.2 ± 1.7	60.9	5.58
S6	80.0 ± 10.5	25.9 ± 6.7	74.0 ± 7.5	42.4 ± 4.0	78.0 ± 5.2	50.1 ± 8.4	58.4	6.50
Mamba	65.3 ± 15.8	27.2 ± 1.8	74.8 ± 6.2	51.3 ± 2.5	80.4 ± 1.9	$\mathbf{57.7 \pm 4.6}$	59.5	4.83
NCDE	77.8 ± 4.6	28.9 ± 7.0	67.1 ± 4.3	51.2 ± 5.6	80.5 ± 6.5	51.9 ± 3.8	59.6	5.92
Log-NCDE	82.8 ± 7.3	$\mathbf{37.2 \pm 3.6}$	74.2 ± 4.6	54.7 ± 6.2	82.6 ± 2.2	53.7 ± 6.7	**64.2**	3.08
D-SLiCE	76.7 ± 5.4	27.3 ± 4.8	$\mathbf{77.4 \pm 6.5}$	51.2 ± 3.9	84.0 ± 1.8	53.7 ± 5.3	61.7	3.83
BD-SLiCE	$\mathbf{84.4 \pm 4.2}$	32.7 ± 4.6	69.0 ± 10.0	57.9 ± 3.1	85.4 ± 2.8	53.0 ± 3.4	63.7	**3.00**

in the path. This preserves the original temporal alignment and avoids shortening or rescaling the path after observations are removed.

For the discrete recurrent models S5, LRU, S6, and Mamba, the randomly retained observations are passed directly to the model in chronological order. When the selected hyperparameters include time, the timestamp is retained as an input channel after dropping. When time is not included, the model receives only the retained observed values.

Table 2 summarises the results across the six datasets. Tables 7–10 give the corresponding per-dataset results. The continuous-time models using our embedding remain competitive across all drop levels. BD-SLiCE is the most stable model under severe sparsification, with its macro-average accuracy changing from 64.0% with no dropping to 62.5% with 95% dropping.

Table 9: **Per-dataset test accuracy (mean $\pm$ std over 5 seeds, in %) for the UEA benchmark at 70% drop, with macro-average accuracy and average dataset rank.** Lower rank is better.

Model	EigenWorms	Ethanol	Heartbeat	MotorImagery	SCP1	SCP2	Avg. Acc.	Avg. Rank
S5	72.8 ± 3.2	27.3 ± 1.3	70.3 ± 4.3	$\mathbf{56.5 \pm 6.4}$	$\underline{85.4 \pm 2.2}$	48.4 ± 6.2	60.1	4.83
LRU	81.1 ± 6.4	25.3 ± 4.7	75.5 ± 6.6	$\underline{54.4 \pm 4.6}$	78.4 ± 3.8	51.2 ± 5.9	61.0	5.00
S6	$\underline{84.7 \pm 6.3}$	25.3 ± 7.4	76.1 ± 4.1	51.3 ± 2.2	79.2 ± 1.9	51.7 ± 3.0	61.4	4.00
Mamba	69.1 ± 13.3	27.2 ± 1.8	$\mathbf{76.7 \pm 5.8}$	53.0 ± 2.2	79.2 ± 4.8	$\mathbf{58.4 \pm 3.2}$	60.6	4.00
NCDE	78.9 ± 2.8	24.1 ± 6.2	69.7 ± 5.9	50.5 ± 2.3	80.2 ± 5.8	49.8 ± 5.9	58.9	6.42
Log-NCDE	$\mathbf{87.2 \pm 5.4}$	$\mathbf{36.2 \pm 5.9}$	73.5 ± 4.4	50.2 ± 5.5	78.8 ± 7.1	50.5 ± 7.6	**62.7**	4.67
D-SLiCE	80.6 ± 7.2	29.4 ± 8.2	$\underline{76.5 \pm 5.3}$	50.2 ± 4.1	84.5 ± 3.7	51.9 ± 2.4	62.2	$\underline{3.83}$
BD-SLiCE	81.7 ± 6.2	$\underline{31.9 \pm 5.1}$	72.6 ± 5.8	50.5 ± 3.2	$\mathbf{85.9 \pm 3.0}$	$\underline{52.3 \pm 5.0}$	$\underline{62.5}$	**3.25**

Table 10: **Per-dataset test accuracy (mean $\pm$ std over 5 seeds, in %) for the UEA benchmark at 95% drop, with macro-average accuracy and average dataset rank.** Lower rank is better.

Model	EigenWorms	Ethanol	Heartbeat	MotorImagery	SCP1	SCP2	Avg. Acc.	Avg. Rank
S5	61.1 ± 3.5	27.3 ± 1.3	70.3 ± 6.3	$\mathbf{53.7 \pm 5.7}$	81.9 ± 4.4	45.6 ± 6.1	56.7	5.33
LRU	$\mathbf{83.9 \pm 2.7}$	25.1 ± 4.8	71.6 ± 4.6	46.7 ± 6.5	81.6 ± 4.5	$\mathbf{56.1 \pm 8.4}$	60.8	$\underline{4.17}$
S6	78.8 ± 9.7	25.7 ± 6.9	71.5 ± 4.1	50.9 ± 4.2	74.9 ± 3.7	51.7 ± 3.2	58.9	5.17
Mamba	72.2 ± 12.2	27.2 ± 1.8	71.0 ± 8.2	46.6 ± 5.0	78.0 ± 1.8	50.5 ± 5.0	57.6	6.17
NCDE	77.2 ± 8.3	28.1 ± 2.5	68.7 ± 4.3	50.2 ± 5.2	76.9 ± 6.6	50.9 ± 3.5	58.7	5.33
Log-NCDE	81.7 ± 3.8	$\mathbf{34.9 \pm 4.7}$	73.5 ± 6.8	49.8 ± 4.5	76.0 ± 4.7	49.5 ± 5.7	60.9	$\underline{4.17}$
D-SLiCE	80.6 ± 5.3	$\underline{30.6 \pm 7.1}$	$\mathbf{75.2 \pm 4.8}$	49.1 ± 4.6	$\underline{84.5 \pm 2.6}$	$\underline{53.0 \pm 3.6}$	$\underline{62.2}$	**2.83**
BD-SLiCE	$\underline{83.3 \pm 3.5}$	30.1 ± 9.3	$\underline{74.2 \pm 7.8}$	$\underline{53.0 \pm 1.7}$	$\mathbf{85.2 \pm 2.6}$	49.5 ± 5.1	**62.5**	**2.83**

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main theoretical contribution, the faithful embedding construction, and the empirical evaluation. The claims are supported by Lemma 1, Proposition 2, and the experiments in Section 3 and Appendix C.3.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations in Appendix B, including the assumptions needed for faithfulness, the computational cost of retaining count coordinates, the scope of the universality result, and the empirical scope of the experiments.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The theoretical results state their assumptions in Lemma 1, Lemma 3, Proposition 4, Corollary 5, and Proposition 6. Complete proofs are provided in Appendix A and Appendix A.4.

   Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper specifies datasets, data generation procedures, train/test sizes, splitting protocols, hyperparameters, optimisers, batch sizes, model dimensions, and evaluation metrics in Section 3 and Appendix C.1 through Appendix C.3. Public data sources and libraries are listed in the experimental appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The paper uses public datasets and public libraries, with URLs and licenses listed in the experimental appendix.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.

- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Training and test details are specified in Appendix C.1, Appendix C.2, and Appendix C.3. These include splits, sample counts, hyperparameters, optimizer, learning rates, batch sizes, number of epochs, model dimensions, and hyperparameter selection protocols.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The main experimental results report mean and standard deviation over five seeds, as stated in the captions and experimental descriptions for Tables 1, 4, and 7–10. The standard deviation grid is also displayed in Figure 4. Timing results are reported as averages over 1000 training steps because they measure resource use rather than statistical performance.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The paper reports the GPUs used, wall-clock timing, peak GPU memory for the EigenWorms timing experiment, and that single runs can be completed on a 24GB NVIDIA RTX 4090 GPU in less than 24 hours. Section 3.1 and the experimental appendix provide the relevant compute details.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The work uses public benchmark datasets, synthetic data, and standard machine-learning experiments. No human-subject data collection or high-risk deployment is involved.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [No]

Justification: The paper is primarily methodological and does not include a dedicated broader impacts discussion. Potential positive impacts include more efficient and faithful modelling of irregular data, while possible negative impacts would depend on downstream applications that are not studied in this work.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release high-risk models, scraped datasets, pre-trained language models, image generators, or other assets that present a high risk of misuse. The experiments use synthetic data and established public time-series benchmarks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites the datasets, libraries, and baseline implementations used in the experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce a new dataset, benchmark, pretrained model, or released asset.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing, user studies, or new research with human subjects. All experiments use synthetic data or established public benchmark datasets.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing, user studies, or new research with human subjects, so IRB or equivalent approval is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not used as an important, original, or non-standard component of the core methodology, experiments, or analysis in this research.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.