Log Neural Controlled Differential Equations: The Lie Brackets Make a Difference

Benjamin Walker [1] **Andrew D. McLeod** [1] **Tiexin Qin** [2] **Yichuan Cheng** [2] **Haoliang Li** [2] **Terry Lyons** [1]

Abstract

The vector field of a controlled differential equation (CDE) describes the relationship between a *control* path and the evolution of a *solution* path. Neural CDEs (NCDEs) treat time series data as observations from a control path, parameterise a CDE's vector field using a neural network, and use the solution path as a continuously evolving hidden state. As their formulation makes them robust to irregular sampling rates, NCDEs are a powerful approach for modelling real-world data. Building on neural rough differential equations (NRDEs), we introduce Log-NCDEs, a novel, effective, and efficient method for training NCDEs. The core component of Log-NCDEs is the Log-ODE method, a tool from the study of rough paths for approximating a CDE's solution. Log-NCDEs are shown to outperform NCDEs, NRDEs, the linear recurrent unit, S5, and MAMBA on a range of multivariate time series datasets with up to $50{,}000$ observations.

1. Introduction

1.1. Multivariate Time Series Modelling

Neural controlled differential equations (NCDEs) offer a number of advantages for modelling real-world multivariate time series. These include being robust to irregular sampling rates and decoupling the number of forward passes through their neural network from the number of observations in the time series. However, there exists a gap in performance between NCDEs and current state-of-the-art approaches for time series modelling, such as S5, the linear recurrent unit (LRU), and MAMBA (Smith et al., 2023; Orvieto et al., 2023; Gu & Dao, 2023).

[1]Mathematical Institute, University of Oxford, UK [2]Department of Electrical Engineering, City University of Hong Kong. Correspondence to: Benjamin Walker <mlbenjamin walker@gmail.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

This paper demonstrates that, on a range of multivariate time series benchmarks, NCDEs can outperform current state-of-the-art approaches by utilising a tool from the study of rough paths, the Log-ODE method. We refer to this new approach as Log-NCDEs[1].

1.2. Neural Controlled Differential Equations

Let $\{(t_i, x_i)\}_{i=0}^{n}$ denote a set of observations from a multivariate time series, where $x_i \in \mathbb{R}^{v-1}$. Let $X : [t_0, t_n] \to \mathbb{R}^v$ be a continuous interpolation, such that $X_{t_i} = (t_i, x_i)$, where subscript on time-dependent variables denotes evaluation. Let $h_t \in \mathbb{R}^u$ and $z_t \in \mathbb{R}^w$ be the NCDE's hidden state and output at time t respectively. Let $\xi_\phi : \mathbb{R}^v \to \mathbb{R}^u$ and $f_\theta : \mathbb{R}^u \to \mathbb{R}^{u \times v}$ be neural networks, and $l_\psi : \mathbb{R}^u \to \mathbb{R}^w$ be a linear map, where ϕ, θ, and ψ represent the learnable parameters. A NCDE is defined by

$$\begin{aligned} h_{t_0} &= \xi_\phi(t_0, x_0), \\ h_t &= h_{t_0} + \int_{t_0}^{t} f_\theta(h_s)\mathrm{d}X_s, \\ z_t &= l_\psi(h_t), \end{aligned} \tag{1}$$

for $t \in [t_0, t_n]$, where $f_\theta(h_s)\mathrm{d}X_s$ is matrix-vector multiplication (Kidger et al., 2020). Details on the regularity required for existence and uniqueness of the solution to (1) can be found in Appendix A.1. By an extension of the Picard-Lindelöf Theorem, a sufficient condition is X being of bounded variation and f_θ being Lipschitz continuous (Lyons et al., 2007, Theorem 1.3).

NCDEs are an attractive option for modelling multivariate time series. They are universal approximators of continuous real-valued functions on time series data (Kidger, 2022, Theorem 3.9). Additionally, since they interact with time series data through X, NCDEs are unaware of when the time series was observed. This makes them robust to irregular sampling rates. Furthermore, the number of forward passes through f_θ when evaluating (1) is controlled by the differential equation solver. This is opposed to recurrent models, where it is controlled by the number of observations n. By decoupling the number of forward passes through

[1] https://github.com/Benjamin-Walker/Log-Neural-CDEs

their neural network from the number of observations in the time series, NCDEs can mitigate exploding or vanishing gradients on highly-sampled time series.

To make use of the techniques developed for training neural ordinary differential equations (ODEs) (Chen et al., 2018), NCDEs are typically rewritten as an ODE,

$$\tilde{h}_t = \tilde{h}_{t_0} + \int_{t_0}^{t} g_{\theta,X}(\tilde{h}_s)\mathrm{d}s, \tag{2}$$

where $\tilde{h}_{t_0} = h_{t_0}$. Kidger et al. (2020) proposed taking X to be a differentiable interpolation and

$$g_{\theta,X}(\tilde{h}_s) = f_\theta(\tilde{h}_s)\frac{\mathrm{d}X}{\mathrm{d}s}. \tag{3}$$

1.3. Neural Rough Differential Equations

Neural rough differential equations (NRDEs) are based on the Log-ODE method, which is discussed briefly here, and in full detail in Section 2.

Given a set of intervals $\{[r_i, r_{i+1}]\}_{i=0}^{m-1}$ satisfying $t_0 = r_0 < \ldots < r_m = t_n$, the Log-ODE method replaces a CDE with an ODE on each interval. For a depth$-N$ method, the ODE on $[r_i, r_{i+1}]$ is defined by

$$g_{\theta,X}(\tilde{h}_s) = \bar{f}_\theta\left(\tilde{h}_s\right)\frac{\log(S^N(X)_{[r_i,r_{i+1}]})}{r_{i+1} - r_i}, \tag{4}$$

where $\bar{f}_\theta$ is constructed using the iterated Lie brackets of f_θ and $\log(S^N(X)_{[r_i,r_{i+1}]})$ is the depth-N truncated log-signature of X over $[r_i, r_{i+1}]$ (Boutaib et al., 2013). Informally, $\bar{f}_\theta$ is a high order description of the vector field f_θ and $\log(S^N(X)_{[r_i,r_{i+1}]})$ is a high order description of the control path X over $[r_i, r_{i+1}]$. Note that when using (3), $\tilde{h}_t$ is exactly h_t for all $t \in [t_0, t_n]$, whereas when using (4), $\tilde{h}_t$ is an approximation of h_t when $t \in \{r_i\}_{i=1}^{m}$.

NRDEs replace (3) with (4), but instead of calculating $\bar{f}_\theta$ using the iterated Lie brackets of f_θ, it is treated as a neural network $\bar{f}_\theta : \mathbb{R}^u \to \mathbb{R}^{u\times\beta(v,N)}$, where $\beta(v, N)$ is the dimension of a depth$-N$ truncated log-signature of a v dimensional path. By neglecting the structure of $\bar{f}_\theta$, NRDEs are able to reduce the computational burden of evaluating the vector field, at the cost of increasing the output dimension of the neural network.

Compared to NCDEs, NRDEs can reduce the number of forward passes through the network while evaluating the model, as the vector field is autonomous on each interval $[r_i, r_{i+1}]$. This has been shown to lead to improved classification accuracy, alongside reduced time and memory-usage, on time series with up to 17,000 observations (Morrill et al., 2021). Furthermore, as it is no longer necessary to apply a differentiable interpolation to the time series data, NRDEs are applicable to a wider range of input signals.

1.4. Contributions

This paper introduces Log-NCDEs, which build on NRDEs by constructing $\bar{f}_\theta$ using the iterated Lie brackets of a NCDE's vector field, f_θ. For depth's $N \geq 2$, this change drastically reduces the output dimension of the model's neural network, without impacting the model's expressivity. Furthermore, it improves model performance on every dataset considered in this paper. Calculating the Lie brackets requires that f_θ satisfies a regularity constraint, specifically being $\mathrm{Lip}(\gamma)$ for $\gamma \in (N-1, N]$. Section 3.2 presents a novel theoretical result bounding the $\mathrm{Lip}(\gamma)-$norm for a class of fully connected neural networks when $1 < \gamma \leq 2$. Following this, Section 3.3 details how to efficiently calculate the Lie brackets of a $\mathrm{Lip}(\gamma)$ neural network using standard machine learning tools. The paper concludes by showing that, over a range of multivariate time series benchmarks, Log-NCDEs outperform NCDEs, NRDEs, S5, LRU, and MAMBA.

2. Mathematical Background

The following section provides a thorough mathematical description of the Log-ODE method. It will introduce $\mathrm{Lip}(\gamma)$ regularity, the Lie bracket of two vector fields, and the signature and log-signature of a path, alongside their respective spaces, the tensor algebra and the free Lie algebra.

2.1. The Tensor Algebra

Definition 2.1. Let U, V, and W be vector spaces. The tensor product space $U \otimes V$ is the unique (up to isomorphism) space such that for all bilinear functions $\kappa : U \times V \to W$ there exists a unique linear map $\tau : U \otimes V \to W$, such that $\kappa = \tau \circ \otimes$ (Roman, 2007, Chapter 14).

As an example, let $V = \mathbb{R}^2$ and $W = \mathbb{R}^3$. In this case, the tensor product is the outer product of the two vectors, and the resulting tensor product space is the space of 2×3 matrices, $\mathbb{R}^2 \otimes \mathbb{R}^3 = \mathbb{R}^{2\times 3}$. The tensor product of $v \in \mathbb{R}^2$ and $w \in \mathbb{R}^3$ is defined by

$$v \otimes w = \begin{bmatrix} a \\ b \end{bmatrix} \otimes \begin{bmatrix} c \\ d \\ e \end{bmatrix} = \begin{bmatrix} ac & ad & ae \\ bc & bd & be \end{bmatrix}, \tag{5}$$

where any bilinear function $\kappa(v, w)$ can be written as a linear function $\tau(v \otimes w)$.

Definition 2.2. The tensor algebra space is the space

$$T((V)) = \{x = (x^0, x^1, \ldots) | x^i \in V^{\otimes i}\}, \tag{6}$$

where $V^{\otimes 0} = \mathbb{R}$, $V^{\otimes 1} = V$, and $V^{\otimes j} = V \otimes V^{\otimes j-1}$ (Roman, 2007, Chapter 14).

Details on the choice of norm for $V^{\otimes i}$ when V is a complete

normed vector space, i.e. a Banach space, can be found in Appendix A.2.

2.2. $\mathrm{Lip}(\gamma)$ Functions

Let V and W be Banach spaces and $\mathbf{L}(V, W)$ denote the space of linear mappings from V to W equipped with the operator norm.

Definition 2.3. A linear map $l \in \mathbf{L}(V^{\otimes j}, W)$ is j symmetric if for all $v_1 \otimes \cdots \otimes v_j \in V^{\otimes j}$ and all bijective functions $p : \{1, \ldots, j\} \to \{1, \ldots, j\}$ (Roman, 2007, Chapter 14),

$$l(v_1 \otimes \ldots \otimes v_j) = l(v_{p(1)} \otimes \cdots \otimes v_{p(j)}). \tag{7}$$

Let $\mathbf{L}_s(V^{\otimes j}, W)$ denote the set of all j symmetric linear maps.

Definition 2.4. Let $\gamma > 0$, $k \in \mathbb{Z}$ such that $\gamma \in (k, k+1]$, F be a closed subset of V, and $f^0 : F \to W$. For $j \in \{1, \ldots, k\}$, let $f^j : F \to \mathbf{L}_s(V^{\otimes j}, W)$. The collection $(f^0, f^1, \ldots, f^k)$ is an element of $\mathrm{Lip}(\gamma)$ if there exists $M \geq 0$ such that for $j \in \{0, \ldots, k\}$,

$$\sup_{x \in F} ||f^j(x)||_{\mathbf{L}(V^{\otimes j}, W)} \leq M, \tag{8}$$

and for $j \in \{0, \ldots, k\}$, all $x, y \in F$, and each $v \in V^{\otimes j}$ (Stein, 1970),

$$\frac{\left|\left| f^j(y)(v) - \sum_{l=0}^{k-j} \frac{f^{j+l}(x)(v \otimes (y-x)^{\otimes l})}{l!} \right|\right|_W}{|x-y|_V^{\gamma - j}} \leq M. \tag{9}$$

If $f = (f^0, f^1, \ldots, f^k) \in \mathrm{Lip}(\gamma)$, then the $\mathrm{Lip}(\gamma)-$norm, denoted $||f||_{\mathrm{Lip}(\gamma)}$, is the smallest M for which (8) and (9) hold. When $0 < \gamma \leq 1$, then $k = 0$ and $f^0 \in \mathrm{Lip}(\gamma)$ implies f^0 is bounded and $\gamma-$Hölder continuous. When $\gamma = 1$, then f^0 is bounded and Lipschitz. In this paper, we are concerned with the regularity of vector fields on $\mathbb{R}^u$. In this case, $f \in \mathrm{Lip}(\gamma)$ for $\gamma \in (k, k+1]$ implies that f is bounded, has k bounded derivatives, and the kth derivative satisfies

$$||D^k f(y) - D^k f(x)|| \leq M|x-y|^{\gamma - k}, \tag{10}$$

for all $x, y \in \mathbb{R}^u$.

2.3. The Free Lie Algebra

Definition 2.5. A Lie algebra is a vector space V with a bilinear map $[\cdot, \cdot] : V \times V \to V$ satisfying $[w, w] = 0$ and the Jacobi identity,

$$[[x, y], z] + [[y, z], x] + [[z, x], y] = 0, \tag{11}$$

for all $w, x, y, z \in V$. The map $[\cdot, \cdot]$ is called the Lie bracket (Reutenauer, 1993).

Any associative algebra, $(V, \times)$, has a Lie bracket structure with Lie bracket defined by

$$[x, y] = x \times y - y \times x, \tag{12}$$

for all $x, y \in V$. For example, $V = \mathbb{R}^{n \times n}$ with the matrix product. For another example, consider the vector space of infinitely differentiable functions from $\mathbb{R}^u$ to $\mathbb{R}^u$ with pointwise addition, denoted $C^\infty(\mathbb{R}^u, \mathbb{R}^u)$. This space is a Lie algebra when equipped with the Lie bracket

$$[a, b](x) = J_a(x)b(x) - J_b(x)a(x), \tag{13}$$

for $a, b \in C^\infty(\mathbb{R}^u, \mathbb{R}^u)$ and $x \in \mathbb{R}^u$, where $J_a(x) \in \mathbb{R}^{u \times u}$ is the Jacobian of a with entries given by $J_a^{ij}(x) = \partial_j a^i(x)$ for $i, j \in \{1, \ldots, u\}$ (Kirillov, 2008).

Definition 2.6. Let A be a non-empty set, L_0 be a Lie algebra, and $\phi : A \to L_0$ be a map. The Lie algebra L_0 is said to be the free Lie algebra generated by A if for any Lie algebra L and any map $f : A \to L$, there exists a unique Lie algebra homomorphism $g : L_0 \to L$ such that $g \circ \phi = f$ (Reutenauer, 1993).

The free Lie algebra generated by V is the space

$$\mathfrak{L}((V)) = \{(l_0, l_1, \ldots) : l_i \in L_i\}, \tag{14}$$

where $L_0 = 0$, $L_1 = V$, and L_{i+1} is the span of $[v, l]$ for $v \in V$ and $l \in L_i$.

2.4. The Log-Signature

Let $X : [t_0, t_n] \to V$ have bounded variation and define

$$X^n_{[t_0,t_n]} = \underbrace{\int \cdots \int}_{\substack{u_1 < \cdots < u_n \\ u_i \in [t_0, t_n]}} \mathrm{d}X_{u_1} \otimes \cdots \otimes \mathrm{d}X_{u_n} \in V^{\otimes n}. \tag{15}$$

The signature of the path X is

$$S(X)_{[t_0,t_n]} = (1, X^1_{[t_0,t_n]}, \ldots, X^n_{[t_0,t_n]}, \ldots) \in \tilde{T}((V)), \tag{16}$$

where $\tilde{T}((V)) = \{x \in T((V)) | x = (1, \ldots)\}$ (Lyons et al., 2007). The depth-N truncated signature is defined as $S^N(X)_{[t_0,t_n]} = (1, X^1_{[t_0,t_n]}, \ldots, X^N_{[t_0,t_n]}) \in \tilde{T}^N(V)$. The signature is an infinite dimensional vector which describes the path X over the interval $[t_0, t_n]$. In fact, assuming X contains time as a channel, linear maps on $S(X)_{[t_0,t_n]}$ are universal approximators for continuous, real-valued functions of X (Lyons, 2014; Arribas, 2018). This property of the signature relies on the shuffle-product identity, which states that polynomials of the elements in a truncated signature can be rewritten as linear maps on the signature truncated at greater depth (Ree, 1958; Chevyrev & Kormilitzin, 2016). A consequence of the shuffle-product identity is that not every term in the signature provides new information about the path X. The transformation which removes the information redundancy is the logarithm.

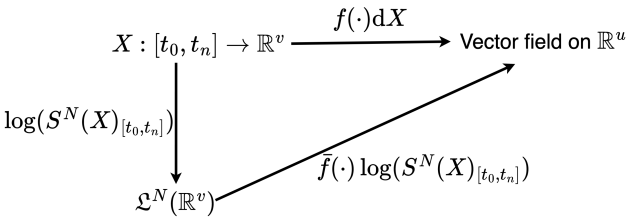


Figure 1. A schematic diagram of the Log-ODE method.

Definition 2.7. For $\mathbf{x} \in \tilde{T}((V))$, the logarithm is defined by

$$\log(\mathbf{x}) = \log(1 + \mathbf{t}) = \sum_{n=1}^{\infty} \frac{(-1)^n}{n} \mathbf{t}^{\otimes n}, \tag{17}$$

where $\mathbf{t} = (0, x^1, x^2, \ldots)$ (Reutenauer, 1993).

It was shown by Chen (1957) that

$$\log(S(X)_{[t_0,t_n]}) \in \mathfrak{L}((V)). \tag{18}$$

This result plays a crucial role in the Log-ODE method.

2.5. The Log-ODE Method

The vector field of a CDE is typically thought of as a matrix-valued function $f : \mathbb{R}^u \to \mathbb{R}^{u \times v}$. An equivalent formulation is f being a linear map acting on $\mathrm{d}X \in \mathbb{R}^v$ and returning a vector field on $\mathbb{R}^u$. In other words, $f(\cdot)\mathrm{d}X : \mathbb{R}^u \to \mathbb{R}^u$. This formulation will prove more useful in the following section.

Let $X : [t_0, t_n] \to \mathbb{R}^v$ be a continuous path. The (truncated) log-signature is a map which takes X to the (truncated) free Lie algebra $\mathfrak{L}^N(\mathbb{R}^v)$. If $f(\cdot)\mathrm{d}X$ is restricted to smooth vector fields on $\mathbb{R}^u$, then $f(\cdot)\mathrm{d}X$ is a linear map from $\mathbb{R}^v$ to the Lie algebra $C^\infty(\mathbb{R}^u, \mathbb{R}^u)$. By definition 2.6, there exists a unique linear map $\bar{f}$ from $\mathfrak{L}^N(\mathbb{R}^v)$ to the smooth vector fields on $\mathbb{R}^u$. Figure 1 is a schematic diagram of this relationship. Since $\bar{f}$ is a Lie algebra homomorphism, it can be defined recursively by

$$\bar{f}(\cdot)a = f(\cdot)a, \quad a \in \mathbb{R}^v \tag{19}$$

and

$$\bar{f}(\cdot)[a, b] = [\bar{f}(\cdot)a, \bar{f}(\cdot)b] \tag{20}$$

for $[a, b] \in \mathfrak{L}^N(\mathbb{R}^v)$.

Over an interval, the Log-ODE method approximates a CDE using an autonomous ODE constructed by applying the linear map $\bar{f}$ to the truncated log-signature of the control, as seen in (21) (Lyons, 2014). There exist theoretical results bounding the error in the Log-ODE method's approximation, including when the control and solution paths live in infinite dimensional Banach spaces (Boutaib et al., 2013). However, for a given set of intervals, the series of vector fields $\{g_X(\cdot)\}_{N=1}^{\infty}$ is not guaranteed to converge. In practice, N is typically chosen as the smallest N such that a reasonably sized set of intervals $\{r_i\}_{i=0}^m$ gives an approximation error of the desired level. A recent development has been the introduction of an algorithm which adaptively updates N and $\{r_i\}_{i=0}^m$ (Bayer et al., 2023).

3. Method

3.1. Log Neural Controlled Differential Equations

Log-NCDEs use the same underlying model as NRDEs

$$\begin{aligned} h_t &= h_{t_0} + \int_{t_0}^{t} g_{\theta,X}(h_s)\mathrm{d}s, \\ g_{\theta,X}(h_s) &= \bar{f}_\theta(h_s) \frac{\log(S^N(X)_{[r_i,r_{i+1}]})}{r_{i+1} - r_i}, \end{aligned} \tag{21}$$

but with two major changes. First, instead of parameterising $\bar{f}_\theta$ using a neural network, it is constructed using the iterated Lie brackets of a NCDE's neural network, f_θ, via (19) and (20). Second, f_θ is ensured to be a $\mathrm{Lip}(\gamma)$ function for $\gamma \in (N-1, N]$. Figure 2 is a schematic diagram of a Log-NCDE.

These changes have a major benefit. For $N > 1$, Log-NCDEs are exploring a drastically smaller output space during training than NRDEs, while maintaining the same expressivity, as NCDEs are universal approximators. This is because the output dimension of f_θ is $u \times v$, whereas the output dimension of $\bar{f}_\theta$ is $u \times \beta(v, N)$. Figure 3 compares these values for paths of dimension v from 1 to 15 and truncation depths of $N = 1$ and $N = 2$. This benefit comes at the cost of needing to calculate the iterated Lie brackets when evaluating Log-NCDEs, which will be quantified in Section 3.4 and explored empirically in Section 5.2.

When $N = 1$, (21) simplifies to

$$g_{\theta,X}(h_s) = f_\theta(h_s) \frac{X_{r_{i+1}} - X_{r_i}}{r_{i+1} - r_i}. \tag{22}$$

Hence, in this case the only difference between Log-NCDEs and NRDEs is the regularisation of f_θ. Furthermore, (22) and (3) are equivalent when X is a linear interpolation. Therefore, the approach of NCDEs, NRDEs, and Log-NCDEs coincide when using a depth-1 Log-ODE approximation (Morrill et al., 2021).

3.2. $\mathrm{Lip}(\gamma)$ Neural Networks

The composition of two $\mathrm{Lip}(\gamma)$ functions is $\mathrm{Lip}(\gamma)$ (Cass et al., 2012, Lemma 2.2). Hence, a simple approach to ensuring a fully connected neural network (FCNN) is $\mathrm{Lip}(\gamma)$ is to make each layer $\mathrm{Lip}(\gamma)$. This can be achieved by choosing an infinitely differentiable activation function, such as SiLU (Elfwing et al., 2018). However, in practice this may not

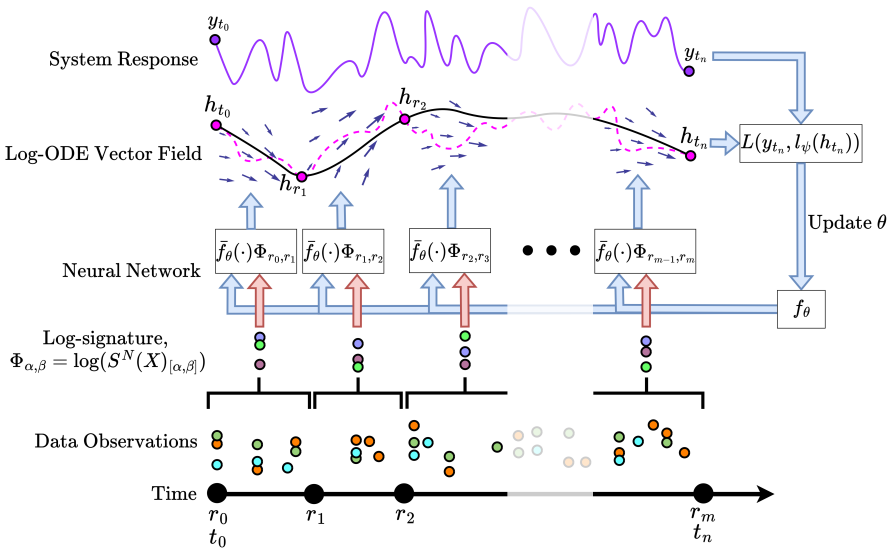


Figure 2. A schematic diagram of the training loop for a Log-NCDE. The coloured circles labelled data observations represent irregular samples from a time series. The purple line labelled system response is a potentially time varying label one wants to predict. The log-signatures of the data observations over each interval $[r_i, r_{i+1}]$ are combined with the iterated Lie brackets of f_θ to produce the vector field $g_{\theta,X}$ from (21). The pink dashed line represents the solution of (1) and the solid black line represents the approximation obtained by solving (21). A linear map l_ψ gives the Log-NCDE's prediction and a loss function $L(\cdot,\cdot)$ is used to update f_θ's parameters.

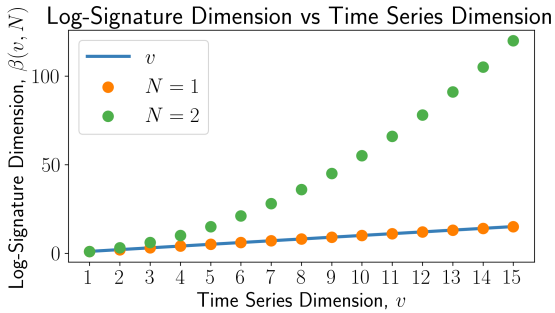


Figure 3. A plot of $\beta(v, N)$ against v for $N = 1, 2$. The output dimension of a NRDE's neural network is $\mathbb{R}^{u\times\beta(v,N)}$, whereas for a Log-NCDE it is $\mathbb{R}^{u\times v}$.

ensure sufficient regularity, as demonstrated by Theorem 3.1.

Theorem 3.1. *Let f_θ be a FCNN with input dimension n_{in}, hidden dimension n_h, depth m, and activation function SiLU. Assuming the input $\mathbf{x} = [x_1, \ldots, x_{n_{in}}]^T$ satisfies $|x_j| \leq 1$ for $j = 1, \ldots, n_{in}$, then $f_\theta \in \mathrm{Lip}(2)$ and*

$$||f_\theta||_{Lip(2)} \leq C P_{m!}\left(\{||W^i||_2, ||\mathbf{b}^i||_2\}_{i=1}^m\right) \tag{23}$$

where C is a constant depending on n_{in}, n_h, and m, $\{W^i\}_{i=1}^m$ and $\{\mathbf{b}^i\}_{i=1}^m$ are the weights and biases of ith layer of f_θ, and $P_{m!}$ is a polynomial of order $m!$.

Proof. A proof is given in Appendix B. □

Assuming that each layer $\{L^i\}_{i=1}^m$ of f_θ satisfies $||L^i||_{\mathrm{Lip}(2)} = 1$, an explicit evaluation of (23) gives

$$||f_\theta||_{\mathrm{Lip}(2)} \leq 5^{2^{m-1}-1}. \tag{24}$$

For a depth 7 FCNN, this bound is greater than the maximum value of a single precision floating point number. Hence, it may be necessary to control $||f_\theta||_{\mathrm{Lip}(2)}$ explicitly during training. This is achieved by modifying the neural network's loss function L to

$$L \mapsto L + \lambda\left(\sum_{i=1}^m ||W^i||_2 + ||\mathbf{b}^i||_2\right), \tag{25}$$

where λ is a hyperparameter controlling the weight of the penalty. This is an example of weight regularisation, which has long been understood to improve generalisation in NNs (Hinton, 1987; Krogh & Hertz, 1991). Equation 25 is specifically a variation of spectral norm regularisation (Yoshida & Miyato, 2017).

3.3. Constructing the Log-ODE Vector Field

The linear map $\bar{f}_\theta$ in (21) is defined recursively by (19) and (20). Assuming $f_\theta(\cdot)a$ is infinitely differentiable, then $f_\theta(\cdot)a$ is an element of the Lie algebra $C^\infty(\mathbb{R}^u, \mathbb{R}^u)$ and

$$[f_\theta(\cdot)a, f_\theta(\cdot)b] = J_{f_\theta(\cdot)a} f_\theta(\cdot)b - J_{f_\theta(\cdot)b} f_\theta(\cdot)a, \tag{26}$$

as discussed in Section 2.3. Let $\{e_j\}_{j=1}^v$ be the usual basis of $\mathbb{R}^v$. A choice of basis for $\mathfrak{L}^N(\mathbb{R}^v)$ is a Hall basis, denoted

$\{\hat{e}_k\}_{k=1}^{\beta(v,N)}$, which is a specific subset of up to the $(N-1)^{\text{th}}$ iterated Lie brackets of $\{e_j\}_{j=1}^{v}$ (Hall, 1950). Rewriting (21) using a Hall basis,

$$\bar{f}_\theta(h_s)\frac{\log(S^N(X)_{[r_i,r_{i+1}]})}{r_{i+1}-r_i} = \sum_{k=1}^{\beta(v,N)} \lambda_k \bar{f}_\theta(h_s)\hat{e}_k, \quad (27)$$

where λ_k is the term in the scaled log-signature corresponding to the basis element $\hat{e}_k$. Since each $\hat{e}_k$ can be written as iterated Lie brackets of $\{e_j\}_{j=1}^{v}$, it is possible to replace $\bar{f}_\theta(\cdot)\hat{e}_k$ with the iterated Lie brackets of $f_\theta(\cdot)e_i$ using (19) and (20). Each $f_\theta(\cdot)e_i : \mathbb{R}^u \to \mathbb{R}^u$ is a vector field defined by the ith column of the neural network's output. Hence, $g_{\theta,X}$ can be evaluated at a point using iterated Jacobian-vector products (JVPs) of f_θ.

3.4. Computational Cost

When the signature truncation depth N is greater than 1, NRDEs and Log-NCDEs incur an additional computational cost for each evaluation of the vector field, which we quantify here for $N = 2$. Assume that a NCDE, NRDE, and Log-NCDE are all using an identical FCNN as their vector field, except for the dimension of the final layer in the NRDE. Let m and n_h be the depth and dimension of the FCNN's hidden layers respectively, and u and v be the dimension of h_t and X from (1) respectively. Letting F_{x} be the number of FLOPs to evaluate model x's vector field,

$$\begin{aligned} F_{\text{NCDE}} &= 2un_h + 2(m-1)n_h^2 + 2uvn_h, \\ F_{\text{NRDE}} &= 2un_h + 2(m-1)n_h^2 + u(v^2-v)n_h, \quad (28) \\ F_{\text{Log-NCDE}} &= 3vF_{\text{NCDE}}, \end{aligned}$$

where the number of FLOPs to calculate a JVP is 3 times that of evaluating the FCNN and v JVPs of f_θ are needed to evaluate (27) when $N = 2$ (Griewank & Walther, 2008, Chapter 4). Log-NCDEs and NRDEs have the same asymptotic computational complexity in each variable. However, each JVP is evaluated at the same point h_s. This allows the computational burden of Log-NCDEs on high-dimensional time series to be reduced by constructing a batched function using Jax's vmap (Bradbury et al., 2018). The computational cost is evaluated empirically in Section 5.2.

3.5. Limitations

In this paper, we only consider Log-NCDEs which use a depth-1 or depth-2 Log-ODE approximation. This is due to the following two limitations. First, there are no theoretical results explicitly bounding the $\text{Lip}(\gamma)$ norm of a neural network for $\gamma > 2$. Second, the computational cost required to evaluate $g_{\theta,X}$ grows rapidly with the depth N. This can make $N > 2$ computationally infeasible, especially for high-dimensional time series. Another general limitation of NCDEs is the need to solve the differential equation recursively, preventing parallelisation. This is in contrast to non-selective structured state-space models, whose underlying model is a differential equation that can be solved parallel in time (Gu et al., 2022).

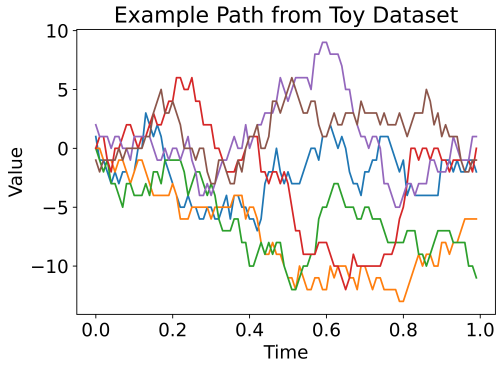


Figure 4. An example path from the toy dataset, where each colour represents a channel in the path.

4. Experiments

4.1. Baseline Methods

Log-NCDEs are compared against six models, which represent the state-of-the-art for a range of deep learning approaches to time series modelling. Four of these models are stacked recurrent models, which use the same general architecture, but with different recurrent layers. The architecture used in this paper is based on the one introduced by Smith et al. (2023) and the four different recurrent layers considered are LRU, S5, MAMBA, and S6, the selective state-space recurrence introduced as a component of MAMBA (Orvieto et al., 2023; Smith et al., 2023; Gu & Dao, 2023). The other two baseline models are continuous models; a NCDE using a Hermite cubic spline with backward differences as the interpolation and a NRDE (Kidger et al., 2020; Morrill et al., 2021). Further details on all model architectures can be found in Appendices C.1 and C.2.

4.2. Toy Dataset

We construct a toy dataset of 100,000 time series with 6 channels and 100 regularly spaced samples each. For every time step, the change in each channel is sampled independently from the discrete probability distribution with density

$$p(n) = \int_{n-0.5}^{n+0.5} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} \mathrm{d}x, \quad (29)$$

where $n \in \mathbb{Z}$. In other words, the change in a channel at each time step is a sample from a standard normal distribution rounded to the nearest integer. Figure 4 is a plot of a sample path from the toy dataset.

We consider four different binary classifications on the toy dataset. Each classification is a specific term in the signature of the path which depends on a different number of channels.

1. Was the change in the third channel, $\int_0^1 \mathrm{d}X_s^3$, greater than zero?
2. Was the area integral of the third and sixth channels, $\int_0^1 \int_0^u \mathrm{d}X_s^3 \mathrm{d}X_u^6$, greater than zero?
3. Was the volume integral of the third, sixth, and first channels, $\int_0^1 \int_0^v \int_0^u \mathrm{d}X_s^3 \mathrm{d}X_u^6 \mathrm{d}X_v^1$, greater than zero?
4. Was the 4D volume integral of the third, sixth, first, and fourth channels, $\int_0^1 \int_0^w \int_0^v \int_0^u \mathrm{d}X_s^3 \mathrm{d}X_u^6 \mathrm{d}X_v^1 \mathrm{d}X_w^4$, greater than zero?

On this dataset, all models use a hidden state of dimension 64 and Adam with a learning rate of 0.0001 (Kingma & Ba, 2017). LRU, S5, S6, and MAMBA use six blocks. NRDE and Log-NCDE take $r_{i+1} - r_i$ to be 4 observations and the signature truncation depth N to be 2. Full hyperparameter choices are given in Appendix C.3.

4.3. UEA Multivariate Time Series Classification Archive

The models considered in this paper are evaluated on six datasets from the UEA multivariate time series classification archive (UEA-MTSCA)[2]. These six datasets were chosen via the following two criteria. First, only datasets with more than 200 total cases were considered. Second, the six datasets with the most observations were chosen, as datasets with many observations have previously proved challenging for deep learning approaches to time series modelling. Further details on the chosen datasets can be found in Appendix C.4. Following Morrill et al. (2021), the original train and test cases are combined and resplit into new random train, validation, and test cases using a $70 : 15 : 15$ split.

Hyperparameters for all models are found using a grid search over the validation accuracy on a fixed random split of the data. Full details on the hyperparameter grid search are in Appendix C.4. Having fixed their hyperparameters, models are compared on their average test set accuracy over five different random splits of the data. In order to compare models on their average GPU memory and run time, 1000 steps of training are run on an NVIDIA RTX 4090. The average run time is estimated by combining the time for 1000 training steps with the average total number of training steps from the five runs over the random data splits.

[2]As of June 1st 2024, the EigenWorms dataset at https://timeseriesclassification.com has 23 duplicated time series, which were removed for the experiments in this paper.

4.4. PPG-DaLiA

PPG-DaLiA is a multivariate time series regression dataset, where the aim is to predict a person's heart rate using data collected from a wrist-worn device (Reiss et al., 2019). The dataset consists of fifteen individuals with around 150 minutes of recording each at a maximum sampling rate of $128\,\mathrm{Hz}$. There are six channels; blood volume pulse, electrodermal activity, body temperature, and three-axis acceleration. The data is split into a train, validation, and test set following a $70 : 15 : 15$ split for each individual. After splitting the data, a sliding window of length 49920 and step size 4992 is applied.

Hyperparameters are found using the same method as for the UEA-MTSCA, but with validation mean squared error and slightly different hyperparameter choices given the high number of observations. Full details can be found in Appendix C.4. Having fixed their hyperparameters, models are compared on their average mean squared error over five different runs on the same fixed data split.

5. Results

5.1. Toy Dataset

Figure 5 compares the performance of the models considered in this paper on the four different toy dataset classifications. As expected, given that the classifications considered are solutions to CDEs, NCDE's are the best performing model. Since NRDEs and Log-NCDEs are fixed to $r_{i+1} - r_i$ being 4 observations and $N = 2$, they are both approximations of a CDE. Notably, Log-NCDEs consistently outperform NRDEs, providing empirical evidence that NRDEs do not always accurately learn the Lie bracket structure of $\bar{f}_\theta$. From the stacked recurrent models, MAMBA achieves the highest validation accuracy on all four datasets. However, the performance of all the stacked recurrent models decreases drastically as the number of channels the label depends on increases.

5.2. UEA-MTSCA

Table 1 reports the average and standard deviation of each model's test set accuracy over five different splits of the data. On the datasets considered, NCDEs achieve the lowest average accuracy. NRDEs improve upon NCDEs in both average accuracy and rank, but are still outperformed by three of the stacked recurrent models, LRU, S5, and S6. Log-NCDEs are the best performing model on average test set accuracy and average rank. Compared to NRDEs, they achieve a higher average test set accuracy on all six datasets and a lower standard deviation on five datasets. These results highlight the improvement in performance which can be achieved by calculating the Lie brackets.

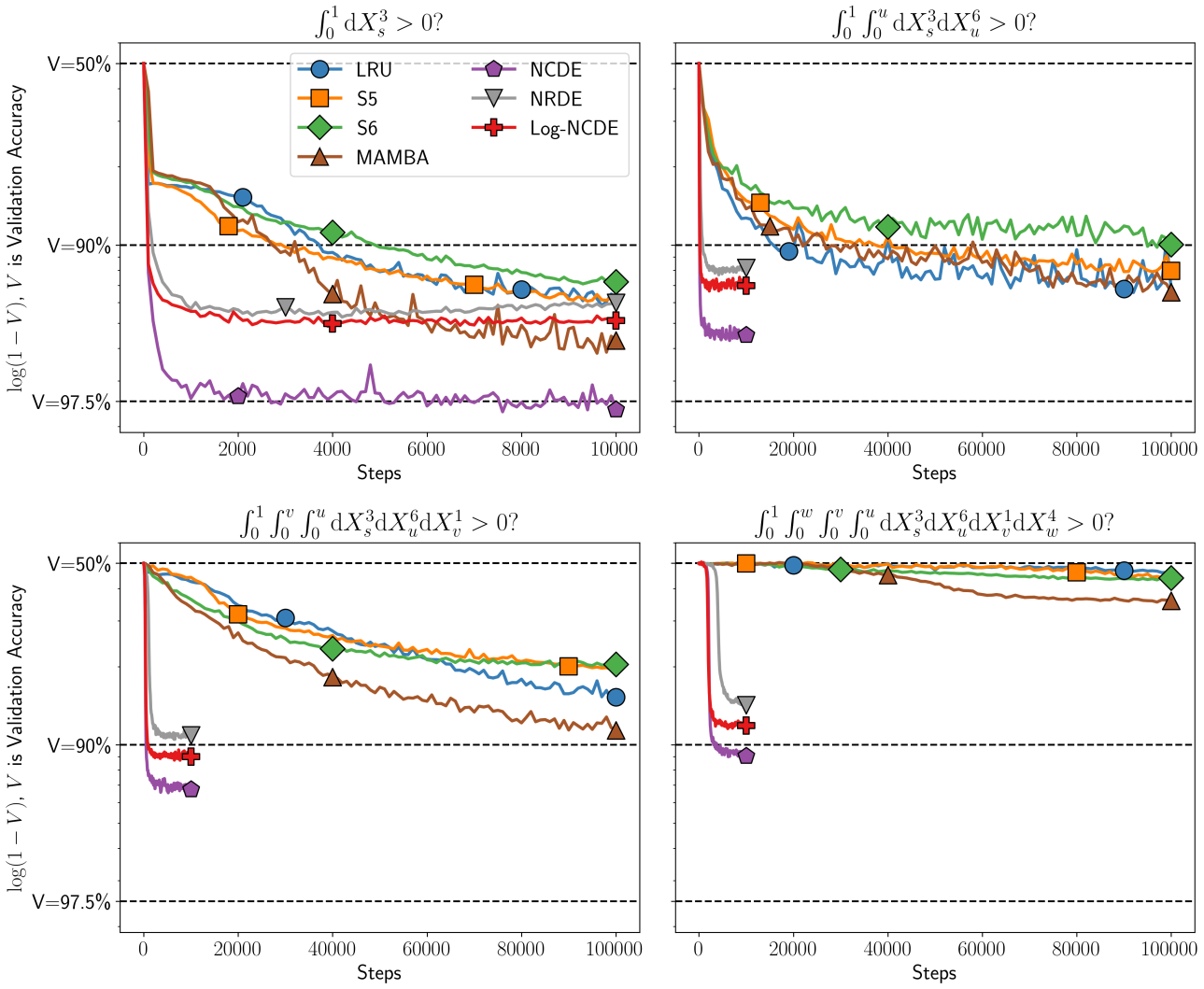


Figure 5. Validation accuracy against number of steps for LRU, S5, S6, MAMBA, NCDE, NRDE, and Log-NCDE on the four different classifications considered for the toy dataset.

Table 1. Test set accuracy on a subset of the UEA-MTSCA. The best performing model is highlighted in bold and the second best is underlined. The average accuracy and average rank are also reported.

Dataset	Method						
	LRU	S5	S6	MAMBA	NCDE	NRDE	Log-NCDE
EigenWorms	**85.0 ± 6.2**	83.9 ± 4.1	**85.0 ± 16.1**	70.9 ± 15.8	62.2 ± 2.2	77.2 ± 7.1	82.8 ± 2.7
EthanolConcentration	23.8 ± 2.8	25.6 ± 3.5	26.4 ± 6.4	27.9 ± 4.5	22.0 ± 1.0	31.4 ± 4.5	**35.9 ± 6.1**
Heartbeat	**78.1 ± 7.6**	73.9 ± 3.1	76.5 ± 8.3	76.2 ± 3.8	68.1 ± 5.8	73.9 ± 2.6	74.2 ± 2.0
MotorImagery	51.9 ± 8.6	53.0 ± 3.9	51.3 ± 4.7	47.7 ± 4.5	51.6 ± 6.2	54.0 ± 7.8	**57.2 ± 5.6**
SelfRegulationSCP1	84.5 ± 4.6	**87.1 ± 2.1**	82.8 ± 2.7	80.7 ± 1.4	80.0 ± 2.0	76.7 ± 5.6	82.1 ± 1.4
SelfRegulationSCP2	47.4 ± 4.0	**55.1 ± 3.3**	49.9 ± 9.4	48.2 ± 3.9	49.1 ± 6.2	48.1 ± 11.4	54.0 ± 2.6
Av.	61.8	63.1	62.0	58.6	55.5	60.2	**64.4**
Av. Rank	3.6	3.1	3.3	4.8	6.0	4.6	**2.7**

Table 2. Average GPU memory and run time for each model over the six datasets from the UEA-MTSCA experiments.

Model	Av. GPU Mem. (MB)	Av. run time (s)
LRU	5484.3	379.5
S5	5327.7	441.4
S6	2608.0	578.1
MAMBA	4450.3	1609.1
NCDE	1674.3	7720.7
NRDE	3338.3	8338.3
Log-NCDE	2661.3	2200.1

Table 3. Average test set mean squared error on the PPG-DaLiA dataset.

Model	MSE ($\times 10^{-2}$)
LRU	15.64 ± 1.67
S5	12.79 ± 0.72
S6	12.88 ± 2.05
MAMBA	$\underline{10.65 \pm 2.20}$
NCDE	13.77 ± 0.41
NRDE	18.79 ± 2.69
Log-NCDE	$\mathbf{9.20 \pm 0.55}$

Table 2 details the average GPU memory and run time for each model, with the results for individual datasets given in Appendix C.5. The major contributors to NCDEs high average run time are time series with many observations. Using a depth-2 Log-ODE method decreases the computational burden on datasets with many observations for both NRDEs and Log-NCDEs. Although NRDEs and Log-NCDEs have the same asymptotic computational complexity, using a batched function to calculate the Lie brackets leads to Log-NCDEs having a lower computational burden on high-dimensional time series than NRDEs, as discussed in Section 3.4. Hence, Log-NCDEs have a lower average run time than both NCDEs and NRDEs. Despite these improvements, all four stacked recurrent models have lower average run times than Log-NCDEs.

5.3. PPG-DaLiA

Table 3 contains the average and standard deviation of each model's test set mean squared error on the PPG-DaLiA dataset. In contrast to the UEA-MTSCA experiments, MAMBA is the best performing stacked recurrent model and NRDEs do not improve over NCDEs. However, Log-NCDEs still achieve the best performance, obtaining the lowest average test set mean squared error and the second lowest standard deviation.

6. Discussion

Recent theoretical work on the expressive power of structured state space models may provide an explanation for their performance on the toy dataset. It has been shown that the recurrent layer of non-selective state space models, such as S5, are unable to capture terms in the signature that depend on more than one channel. Instead, the computational burden is placed on the non-linear mixing in-between recurrent blocks. Furthermore, it has been shown that stacked selective state-space models, such as MAMBA, can capture higher order terms in the signature with only linear mixing layers (Cirone et al., 2024). Although the toy dataset highlights a potential limitation of the stacked recurrent models, this did not translate into poor performance on the real-world datasets considered in this paper.

Log-NCDEs achieve the highest average accuracy on the UEA-MTSCA datasets and the lowest mean squared error on the PPG-DaLiA dataset. In particular, they improve upon NRDEs on every dataset. These results highlight the effectiveness of the Log-ODE method for improving the performance of NCDEs and the importance of calculating the Lie brackets when applying the Log-ODE method. Furthermore, despite increasing the computational cost of each vector field evaluation, Log-NCDEs have a lower average run time than both NCDEs and NRDEs on the UEA-MTSCA datasets. Empirical evidence suggests this is due to the Log-ODE method improving efficiency on time series with many observations, and calculating the Lie brackets lowering the computational burden of the Log-ODE method on high-dimensional time series. In addition to achieving state-of-the-art performance on the regularly sampled datasets considered in this paper, Log-NCDEs maintain the ability of NCDEs to naturally handle irregular sampling, making them an attractive option for real-world applications.

7. Conclusion

Building on NRDEs, this paper introduced Log-NCDEs, which utilise the Log-ODE method to train NCDEs in an effective and efficient manner. This required proving a novel theoretical result bounding the $\mathrm{Lip}(\gamma)$ norm of fully connected neural networks for $1 < \gamma \leq 2$, as well as developing an efficient method for calculating the iterated Lie brackets of a neural network. A thorough empirical evaluation demonstrated the benefits of calculating the Lie brackets when applying the Log-ODE method. Furthermore, it showed that Log-NCDEs can achieve state-of-the-art performance on a range of multivariate time series datasets.

A reasonable direction of future work is extending Log-NCDE's to depth$-N$ Log-ODE methods for $N > 2$. This would require proving an equivalent result to Theorem 3.1 for $\gamma > 2$. Furthermore, it would be necessary to address the computational cost of the iterated Lie brackets. This could be achieved by using a structured neural network with cheap Jacobian-vector products as the CDE vector field. Another avenue of future work could be incorporating the recently developed adaptive version of the Log-ODE method (Bayer et al., 2023).

Impact Statement

This paper presents Log Neural Controlled Differential Equations, a novel approach aimed at advancing the field of time series modelling. Potential applications of the method include healthcare, finance, and biology, where accurate time series modeling plays a crucial role. Despite the clear potential for positive impact, care must be taken to further understand the capabilities and limitations of the model before real-world deployment. Additionally, structured state-space models, an alternative approach to time series modelling, have recently been integrated into large language models (LLMs). The advancement of LLMs has many potential societal consequences, both positive and negative.

Acknowledgements

Benjamin Walker was funded by the Hong Kong Innovation and Technology Commission (InnoHK Project CIMDA). Andrew McLeod was funded in part by the EPSRC [grant number EP/S026347/1] and in part by The Alan Turing Institute under the EPSRC grant EP/N510129/1. Terry Lyons was funded in part by the EPSRC [grant number EP/S026347/1], in part by The Alan Turing Institute under the EPSRC grant EP/N510129/1, the Data Centric Engineering Programme (under the Lloyd's Register Foundation grant G0095), the Defence and Security Programme (funded by the UK Government) and the Office for National Statistics & The Alan Turing Institute (strategic partnership) and in part by the Hong Kong Innovation and Technology Commission (InnoHK Project CIMDA). The authors would like to acknowledge the use of the University of Oxford Advanced Research Computing (ARC) facility in carrying out this work. http://dx.doi.org/10.5281/zenodo.22558

References

Arribas, I. P. Derivatives pricing using signature payoffs. arXiv:1809.09466, 2018.

Bayer, C., Breneis, S., and Lyons, T. An adaptive algorithm for rough differential equations. arXiv:2307.12590, 2023.

Boedihardjo, H., Geng, X., Lyons, T., and Yang, D. The signature of a rough path: Uniqueness. *Advances in Mathematics*, 293:720–737, 2016. ISSN 0001-8708.

Boutaib, Y., Gyurkó, L., Lyons, T., and Yang, D. Dimension-free euler estimates of rough differential equations. *Revue Roumaine des Mathematiques Pures et Appliquees*, 59: 25–53, 2013.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Cass, T., Litterer, C., and Lyons, T. *New Trends in Stochastic Analysis and Related Topics: A Volume in Honour of Professor K. D. Elworthy*. Interdisciplinary mathematical sciences. World Scientific, 2012. ISBN 9789814360913.

Chen, K. T. Integration of paths, geometric invariants and a generalized baker- hausdorff formula. *Annals of Mathematics*, 65:163, 1957.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. Neural ordinary differential equations. In *NeurIPS*, pp. 6572–6583, 2018.

Chevyrev, I. and Kormilitzin, A. A primer on the signature method in machine learning. arXiv:1603.03788, 2016.

Cirone, N. M., Orvieto, A., Walker, B., Salvi, C., and Lyons, T. Theoretical foundations of deep selective state-space models. arXiv:2402.19047, 2024.

Dauphin, Y. N., Fan, A., Auli, M., and Grangier, D. Language modeling with gated convolutional networks. arXiv:1612.08083, 2017.

Elfwing, S., Uchibe, E., and Doya, K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018. Special issue on deep reinforcement learning.

Griewank, A. and Walther, A. *Evaluating Derivatives*. Society for Industrial and Applied Mathematics, 3600 Market Street, Floor 6, Philadelphia, PA, 2nd edition, 2008.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. arXiv:2312.00752, 2023.

Gu, A., Goel, K., and Ré, C. Efficiently modeling long sequences with structured state spaces. In *The International Conference on Learning Representations (ICLR)*, 2022.

Hall, M. A basis for free lie rings and higher commutators in free groups. In *Proceedings of the American Mathematical Society*, volume 1, pp. 575–581, 1950.

Hinton, G. E. Learning translation invariant recognition in massively parallel networks. In *Proceedings of the Parallel Architectures and Languages Europe, Volume I: Parallel Architectures PARLE*, pp. 1–13, Berlin, Heidelberg, 1987. Springer-Verlag.

Kidger, P. On neural differential equations. arXiv: 2202.02435, 2022.

Kidger, P., Morrill, J., Foster, J., and Lyons, T. Neural Controlled Differential Equations for Irregular Time Series. In *Advances in Neural Information Processing Systems*, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. arXiv:1412.6980, 2017.

Kirillov, Jr, A. *An Introduction to Lie Groups and Lie Algebras*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2008.

Krogh, A. and Hertz, J. A. A simple weight decay can improve generalization. In *Proceedings of the 4th International Conference on Neural Information Processing Systems*, NIPS'91, pp. 950–957, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc.

Lyons, T. Differential equations driven by rough signals (i): An extension of an inequality of l. c. young. *Mathematical Research Letters*, 1:451–464, 1994.

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

Lyons, T. and Qian, Z. *Path Integration Along Rough Paths*. Oxford University Press, 12 2002.

Lyons, T., Caruana, M., and Lévy, T. *Differential Equations Driven by Rough Paths: École D'été de Probabilités de Saint-Flour XXXIV-2004*. Springer, 2007.

Lyons, T. J. Differential equations driven by rough signals. *Revista Matemática Iberoamericana*, 14(2):215–310, 1998.

Morrill, J., Salvi, C., Kidger, P., Foster, J., and Lyons, T. Neural rough differential equations for long time series, 2021.

Orvieto, A., Smith, S. L., Gu, A., Fernando, A., Gulcehre, C., Pascanu, R., and De, S. Resurrecting recurrent neural networks for long sequences. arXiv:2303.06349, 2023.

Ree, R. Lie elements and an algebra associated with shuffles. *Annals of Mathematics*, 68:210, 1958.

Reiss, A., Indlekofer, I., Schmidt, P., and Van Laerhoven, K. Deep ppg: Large-scale heart rate estimation with convolutional neural networks. *Sensors*, 19(14), 2019.

Reutenauer, C. *Free Lie Algebras*. LMS monographs. Clarendon Press, 1993.

Roman, S. *Advanced Linear Algebra*. Graduate Texts in Mathematics. Springer New York, 2007.

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

Stein, E. M. *Singular Integrals and Differentiability Properties of Functions (PMS-30)*. Princeton University Press, 1970.

Yoshida, Y. and Miyato, T. Spectral norm regularization for improving the generalizability of deep learning. ArXiv: 1705.10941, 2017.

Young, L. C. An inequality of the hölder type, connected with stieltjes integration. *Acta Mathematica*, 67:251–282, 1936.

A. Additional Mathematical Details

A.1. Existence and Uniqueness

Let V and W be Banach spaces, $X : [0, T] \to V$ and $y : [0, T] \to W$ be continuous paths, and $f(\cdot)\nu$ be a linear map from $\nu \in V$ to vector fields on W. Assume that X, Y, and f are regular enough for the integral

$$\int_0^t f(Y_s)\mathrm{d}X_s \tag{30}$$

to be defined for all $t \in [0, T]$ in the Young sense (Young, 1936). The path Y is said to obey a controlled differential equation (CDE) if

$$Y_t = Y_0 + \int_0^t f(Y_s)\mathrm{d}X_s, \tag{31}$$

for $t \in [0, T]$, where $Y_0 \in W$ is the initial condition and X is the control (Lyons et al., 2007). The existence and uniqueness of the solution to a CDE depends on the smoothness of the control path X and the vector field f. We will measure the smoothness of a path by the smallest $p \geq 1$ for which the $p-$variation is finite and the smoothness of a vector field by the largest $\gamma > 0$ such that the function is Lip(γ) (defined in Section 2.2).

Definition A.1. *(Partition)* A partition of a real interval $[0, T]$ is a set of real numbers $\{r_i\}_{i=0}^m$ satisfying $0 = r_0 < \ldots < r_m = T$.

Definition A.2. *($p-$variation (Young, 1936))* Let V be a Banach space, $\mathcal{D} = (r_0, \cdots, r_m) \subset [0, T]$ be a partition of $[0, T]$, and $p \geq 1$ be a real number. The $p-$variation of a path $X : [0, T] \to V$ is defined as

$$||X||_p = \left[\sup_{\mathcal{D}} \sum_{r_i \in \mathcal{D}} |X_{r_i} - X_{r_{i+1}}|^p\right]^{\frac{1}{p}}. \tag{32}$$

Theorem A.3. *Let $1 \leq p < 2$ and $p - 1 < \gamma \leq 1$. If W is finite-dimensional, X has finite $p-$variation, and f is Lip(γ), then (31) admits a solution for every $y_0 \in W$ (Lyons, 1994).*

Theorem A.4. *Let $1 \leq p < 2$ and $p < \gamma$. If X has finite $p-$variation and f is Lip(γ), then (31) admits a unique solution for every $y_0 \in W$ (Lyons, 1994).*

These theorems extend the classic differential equation existence and uniqueness results to controls with unbounded variation but finite $p-$variation for $p < 2$. A proof of these theorems is can be found in (Lyons et al., 2007). These theorems are sufficient for the differential equations considered in this paper. However, there are many settings where the control has infinite $p-$variation for all $p < 2$, such as Brownian motion. The theory of rough paths was developed in order to give meaning to (31) when the control's $p-$variation is finite only for $p \geq 2$ (Lyons, 1998). An introduction to rough path theory can be found in (Lyons et al., 2007).

A.2. The Tensor Algebra

Let V be a Banach space and $V^{\otimes n}$ denote the tensor powers of V,

$$V^{\otimes n} = V \underbrace{\otimes \cdots \otimes}_{\text{n-1 times}} V. \tag{33}$$

There is choice in the norm of $V^{\otimes n}$. In this paper, we follow the setting of (Boedihardjo et al., 2016) and (Lyons & Qian, 2002). It is assumed that each $V^{\otimes n}$ is endowed with a norm such that the following conditions hold for all $v \in V^{\otimes n}$ and $w \in V^{\otimes m}$:

1. $||v|| = ||v_1 \otimes \cdots \otimes v_n|| = ||v_{p(1)} \otimes \cdots \otimes v_{p(n)}||$ for all all bijective functions $p : \{1, \ldots, n\} \to \{1, \ldots, n\}$,
2. $||v \otimes w|| \leq ||v||\ ||w||$,
3. for any bounded linear functional f on $V^{\otimes n}$ and g on $V^{\otimes m}$, there exists a unique bounded linear functional $f \otimes g$ on $V^{\otimes(m+n)}$ such that $(f \otimes g)(v \otimes w) = f(v)g(w)$.

Definition A.5. *(The Tensor Algebra (Lyons et al., 2007))* For $n \geq 1$, let $V^{\otimes n}$ be equipped with a norm satisfying the above conditions, and define $V^{\otimes 0} = \mathbb{R}$. The tensor algebra space is the set

$$T((V)) = \{\mathbf{x} = (x^0, x^1, \ldots) | x^k \in V^{\otimes k}\} \tag{34}$$

with product $\mathbf{z} = \mathbf{x} \otimes \mathbf{y}$ defined by

$$z^k = (\mathbf{x} \otimes \mathbf{y})^k = \sum_{j=0}^{k} x^j \otimes y^{k-j}. \tag{35}$$

The tensor algebra's product is associative and has unit $\mathbf{1} = (1, 0, 0, \ldots)$. As $T((V))$ is an associative algebra, it has a Lie algebra structure, with Lie bracket

$$[\mathbf{x}, \mathbf{y}] = \mathbf{x} \otimes \mathbf{y} - \mathbf{y} \otimes \mathbf{x} \tag{36}$$

for $\mathbf{x}, \mathbf{y} \in T((V))$ (Reutenauer, 1993).

B. Proof of Theorem 3.1

The proof of Theorem 3.1 relies on two lemmas. The first is a bound on the $\mathrm{Lip}(\gamma)-$norm of the composition of two $\mathrm{Lip}(\gamma)$ functions. The second is a bound on the $\mathrm{Lip}(2)-$norm of each layer of a fully connected neural network (FCNN).

B.1. Composition of Lip(γ) Functions

Lemma B.1. (Composed $\mathrm{Lip}(\gamma)-$norm (Cass et al., 2012)) *Let U, V, and W be Banach spaces and $\Sigma \subset U$ and $\Omega \subset V$ be closed. For $\gamma \geq 1$, let $f \in Lip(\gamma, \Sigma, \Omega)$ and $g \in Lip(\gamma, \Omega, W)$. Then the composition $g \circ f : \Omega \to W$ is Lip(γ) with*

$$||g \circ f||_{Lip(\gamma)} \leq C_\gamma ||g||_{Lip(\gamma)} \max\left\{||f||^{k+1}_{Lip(\gamma)}, 1\right\}, \tag{37}$$

where k is the unique integer such that $\gamma \in (k, k+1]$ and C_γ is a constant independent of f and g.

The original statement of lemma B.1 in (Cass et al., 2012) gives (37) as

$$||g \circ f||_{\mathrm{Lip}(\gamma)} \leq C_\gamma ||g||_{\mathrm{Lip}(\gamma)} \max\left\{||f||^{k}_{\mathrm{Lip}(\gamma)}, 1\right\}. \tag{38}$$

We believe this is a small erratum, as for $g : [0, 1] \to [0, 1]$ defined as $g(x) = x$, (38) implies there exists $C_1 > 0$ such that

$$||g \circ f||_{\mathrm{Lip}(1)} = ||f||_{\mathrm{Lip}(1)} \leq C_1 ||g||_{\mathrm{Lip}(1)} = C_1 \tag{39}$$

for all bounded and Lipschitz $f : [0, 1] \to [0, 1]$. As a counterexample, for any $C_1 > 0$, take $f(x) = x^n$ with $n > \max\{C_1, 1\}$. The following proof of lemma B.1 is given in (Cass et al., 2012).

Proof. Let $(g \circ f)^0, \ldots, (g \circ f)^k$ be defined by the generalisation of the chain rule to higher derivatives. Explicit calculation can be used to verify that if f and g are $\mathrm{Lip}(\gamma)$, definition 2.4 implies $g \circ f$ is $\mathrm{Lip}(\gamma)$ with $||g \circ f||_{\mathrm{Lip}(\gamma)}$ obeying (37). □

Bounding the $\mathrm{Lip}(\gamma)-$norm of a neural network (NN) requires an explicit form for C_γ in (37). This can be obtained via the explicit calculations mentioned in the proof of lemma B.1. Here, we present the case $\gamma \in (1, 2]$.

Lemma B.2. *Let U, V, and W be Banach spaces and $\Sigma \subset U$ and $\Omega \subset V$ be closed. For $\gamma \in (1, 2]$, let $f = (f^{(0)}, f^{(1)}) \in Lip(\gamma, \Sigma, \Omega)$ and $g = (g^{(0)}, g^{(1)}) \in Lip(\gamma, \Omega, W)$. Consider $h^{(0)} : \Sigma \to W$ and $h^{(1)} : \Sigma \to \mathbf{L}(V, W)$ defined for $p \in \Sigma$ and $v \in V$ by*

$$h^{(0)}(p) := g^{(0)}\left(f^{(0)}(p)\right) \qquad \textit{and} \qquad h^{(1)}(p)[v] := g^{(1)}\left(f^{(0)}(p)\right)\left[f^{(1)}(p)[v]\right]. \tag{40}$$

Then $h := \left(h^{(0)}, h^{(1)}\right) \in Lip(\gamma, \Sigma, W)$ and

$$||h||_{Lip(\gamma,\Sigma,W)} \leq (1 + 2^\gamma)\, ||g||_{Lip(\gamma,\Omega,W)} \max\left\{1, ||f||^{\gamma}_{Lip(\gamma,\Sigma,\Omega)}\right\}. \tag{41}$$

Proof. From definition 2.4, $f^{(0)} : \Sigma \to \Omega$, $f^{(1)} : \Sigma \to \mathbf{L}(U, V)$, $g^{(0)} : \Omega \to W$ and $g^{(1)} : \Omega \to \mathbf{L}(V, W)$. Furthermore, for all $p \in \Sigma$

$$(\mathbf{I}) \quad \left|\left|f^{(0)}(p)\right|\right|_V \leq ||f||_{\mathrm{Lip}(\gamma,\Sigma,\Omega)} \qquad \text{and} \quad (\mathbf{II}) \quad \left|\left|f^{(1)}(p)\right|\right|_{\mathbf{L}(U,V)} \leq ||f||_{\mathrm{Lip}(\gamma,\Sigma,\Omega)}. \tag{42}$$

Similarly, for all $x \in \Omega$ we have that

$$(\mathbf{I}) \quad \left|\left|g^{(0)}(x)\right|\right|_W \leq ||g||_{\mathrm{Lip}(\gamma,\Omega,W)} \qquad \text{and} \quad (\mathbf{II}) \quad \left|\left|g^{(1)}(x)\right|\right|_{\mathbf{L}(V,W)} \leq ||g||_{\mathrm{Lip}(\gamma,\Omega,W)}. \tag{43}$$

Define $R_0^f : \Sigma \times \Sigma \to V$ and $R_1^f : \Sigma \times \Sigma \to \mathbf{L}(U, V)$ by

$$\begin{aligned} R_0^f(p, q) &:= f^{(0)}(q) - f^{(0)}(p) - f^{(1)}(p)[q - p], \\ R_1^f(p, q)[u] &:= f^{(1)}(q)[u] - f^{(1)}(p)[u], \end{aligned} \tag{44}$$

for any $p, q \in \Sigma$ and $u \in U$. Then

$$\begin{aligned} &(\mathbf{I}) \quad \left|\left|R_0^f(p, q)\right|\right|_V \leq ||f||_{\mathrm{Lip}(\gamma,\Sigma,\Omega)} ||q - p||_U^{\gamma}, \\ &(\mathbf{II}) \quad \left|\left|R_1^f(p, q)\right|\right|_{\mathbf{L}(U,V)} \leq ||f||_{\mathrm{Lip}(\gamma,\Sigma,\Omega)} ||q - p||_U^{\gamma-1}. \end{aligned} \tag{45}$$

Similarly, define $R_0^g : \Omega \times \Omega \to W$ and $R_1^g : \Omega \times \Omega \to \mathbf{L}(V, W)$ by

$$\begin{aligned} R_0^g(x, y) &:= g^{(0)}(y) - g^{(0)}(x) - g^{(1)}(x)[y - x], \\ R_1^g(x, y)[v] &:= g^{(1)}(y)[v] - g^{(1)}(x)[v], \end{aligned} \tag{46}$$

for $x, y \in \Omega$ and $v \in V$. Then,

$$\begin{aligned} &(\mathbf{I}) \quad ||R_0^g(x, y)||_W \leq ||g||_{\mathrm{Lip}(\gamma,\Omega,W)} ||y - x||_V^{\gamma} \\ &(\mathbf{II}) \quad ||R_1^g(x, y)||_{\mathbf{L}(V,W)} \leq ||g||_{\mathrm{Lip}(\gamma,\Omega,W)} ||y - x||_V^{\gamma-1}. \end{aligned} \tag{47}$$

Define $h^{(0)} : \Sigma \to W$ and $h^{(1)} : \Sigma \to \mathbf{L}(V, W)$ as in (40),

$$h^{(0)}(p) := g^{(0)}\left(f^{(0)}(p)\right) \qquad \text{and} \qquad h^{(1)}(p)[u] := g^{(1)}\left(f^{(0)}(p)\right)\left[f^{(1)}(p)[u]\right], \tag{48}$$

for $p \in \Sigma$ and $u \in U$. Finally, define remainder terms $R_0^h : \Sigma \times \Sigma \to W$ and $R_1^h : \Sigma \times \Sigma \to \mathbf{L}(U, W)$ by

$$\begin{aligned} R_0^h(p, q) &:= h^{(0)}(q) - h^{(0)}(p) - h^{(1)}(p)[q - p], \\ R_1^h(p, q)[u] &:= h^{(1)}(q)[u] - h^{(1)}(p)[u], \end{aligned} \tag{49}$$

for $p, q \in \Sigma$ and $u \in U$. We now establish that $h = (h^{(0)}, h^{(1)}) \in \mathrm{Lip}(\gamma, \Sigma, W)$ and that the norm estimate claimed in (41) is satisfied.

First we consider the bounds on $h^{(0)}$ and $h^{(1)}$. For any $p \in \Sigma$, (**I**) in (43) implies that

$$\left|\left|h^{(0)}(p)\right|\right|_W = \left|\left|g^{(0)}\left(f^{(0)}(p)\right)\right|\right|_W \leq ||g||_{\mathrm{Lip}(\gamma,\Omega,W)} \tag{50}$$

since $f^{(0)}(p) \in \Omega$. Further, for any $p \in \Sigma$ and any $u \in U$, (43) and (**II**) in (42) imply that

$$\begin{aligned} \left|\left|h^{(1)}(p)[u]\right|\right|_W &= \left|\left|g^{(1)}\left(f^{(0)}(p)\right)\left[f^{(1)}(p)[u]\right]\right|\right|_W \\ &\leq \left|\left|g^{(1)}\left(f^{(0)}(p)\right)\right|\right|_{\mathbf{L}(V,W)} \left|\left|f^{(1)}(p)\right|\right|_{\mathbf{L}(U,V)} ||u||_U \\ &\leq ||g||_{\mathrm{Lip}(\gamma,\Omega,W)} ||f||_{\mathrm{Lip}(\gamma,\Sigma,\Omega)} ||u||_U \end{aligned}$$

since $f^{(0)}(p) \in \Omega$. Taking the supremum over $u \in U$ with unit U-norm, it follows that

$$\left|\left|h^{(1)}(p)\right|\right|_{\mathbf{L}(U,W)} \leq ||g||_{\text{Lip}(\gamma,\Omega,W)}||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}. \tag{51}$$

Now we consider the bounds on R_0^h and R_1^h. For this purpose we fix $p, q \in \Sigma$ and $u \in U$. We first assume that $||q-p||_U > 1$. In this case we may use (50) and (51) to compute that

$$\begin{aligned}\left|\left|R_0^h(p,q)\right|\right|_W &= \left|\left|h^{(0)}(q) - h^{(0)}(p) - h^{(1)}(p)[q-p]\right|\right|_W \\ &\leq 2||g||_{\text{Lip}(\gamma,\Omega,W)} + ||g||_{\text{Lip}(\gamma,\Omega,W)}||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}||q-p||_U.\end{aligned}$$

Since $\gamma > 1$ means that $1 < ||q-p||_U < ||q-p||_U^\gamma$, we deduce that

$$\left|\left|R_0^h(p,q)\right|\right|_W \leq ||g||_{\text{Lip}(\gamma,\Omega,W)}\left(2 + ||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}\right)||q-p||_U^\gamma. \tag{52}$$

Similarly, we may use (51) and that $||q-p||_U^{\gamma-1} > 1$ to compute that

$$\left|\left|R_1^h(p,q)[u]\right|\right|_W = \left|\left|h^{(1)}(q)[u] - h^{(1)}(p)[u]\right|\right|_W \leq 2||g||_{\text{Lip}(\gamma,\Omega,W)}||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}||q-p||_U^{\gamma-1}||u||_U. \tag{53}$$

Taking the supremum over $u \in U$ with unit U-norm in (53) yields the estimate that

$$\left|\left|R_1^h(p,q)\right|\right|_{\mathbf{L}(V,W)} \leq 2||g||_{\text{Lip}(\gamma,\Omega,W)}||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}||q-p||_U^{\gamma-1}. \tag{54}$$

Together, (52) and (54) establish the remainder term estimates required to conclude that $h = (h^{(0)}, h^{(1)}) \in \text{Lip}(\gamma, \Sigma, W)$ in the case that $||q-p||_U > 1$.

We next establish similar remainder term estimates when $||q-p||_U < 1$. Thus we fix $p, q \in \Sigma$ and assume that $||q-p||_U < 1$. Note that $\gamma > 1$ means that $||q-p||_U^\gamma < ||q-p||_U < 1$. Additionally,

$$\begin{aligned}\left|\left|f^{(0)}(q) - f^{(0)}(p)\right|\right|_V &\overset{(44)}{=} \left|\left|f^{(1)}(p)[q-p] + R_0^f(p,q)\right|\right|_V, \\ &\leq ||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}\left(||q-p||_U + ||q-p||_U^\gamma\right), \\ &\leq 2||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}||q-p||_U,\end{aligned} \tag{55}$$

where (**II**) in (42) and (**I**) in (45) have been used. We now consider the term $R_0^h(p,q)$. We start by observing that

$$\begin{aligned}R_0^h(p,q) &\overset{(49)}{=} h^{(0)}(q) - h^{(0)}(p) - h^{(1)}(p)[q-p] \\ &\overset{(48)}{=} g^{(0)}\left(f^{(0)}(q)\right) - g^{(0)}\left(f^{(0)}(p)\right) - g^{(1)}\left(f^{(0)}(p)\right)\left[f^{(1)}(p)[q-p]\right] \\ &\overset{(46)}{=} g^{(1)}\left(f^{(0)}(p)\right)\left[f^{(0)}(q) - f^{(0)}(p) - f^{(1)}(p)[q-p]\right] + R_0^g\left(f^{(0)}(p), f^{(0)}(q)\right) \\ &\overset{(44)}{=} g^{(1)}\left(f^{(0)}(p)\right)\left[R_0^f(p,q)\right] + R_0^g\left(f^{(0)}(p), f^{(0)}(q)\right).\end{aligned}$$

Consequently, by using (**II**) in (43) to estimate the term $g^{(1)}\left(f^{(0)}(p)\right)$, (**I**) in (45) to estimate the term $R_0^f(p,q)$, and (**I**) in (47) to estimate the term $R_0^g\left(f^{(0)}(p), f^{(0)}(q)\right)$, we may deduce that

$$\left|\left|R_0^h(p,q)\right|\right|_W \leq ||g||_{\text{Lip}(\gamma,\Omega,W)}\left(||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}||q-p||_U^\gamma + \left|\left|f^{(0)}(q) - f^{(0)}(p)\right|\right|_V^\gamma\right). \tag{56}$$

The combination of (55) and (56) yields the estimate

$$\left|\left|R_0^h(p,q)\right|\right|_W \leq ||g||_{\text{Lip}(\gamma,\Omega,W)}\left(||f||_{\text{Lip}(\gamma,\Sigma,\Omega)} + 2^\gamma||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}^\gamma\right)||q-p||_U^\gamma. \tag{57}$$

Turning our attention to R_1^h, we fix $u \in U$ and compute that

$$\begin{aligned} R_1^h(p,q)[u] &\overset{(49)}{=} h^{(1)}(q)[u] - h^{(1)}(p)[u] \\ &\overset{(48)}{=} g^{(1)}\left(f^{(0)}(q)\right)\left[f^{(1)}(q)[u]\right] - g^{(1)}\left(f^{(0)}(p)\right)\left[f^{(1)}(p)[u]\right] \\ &\overset{(46)}{=} g^{(1)}\left(f^{(0)}(p)\right)\left[f^{(1)}(q)[u] - f^{(1)}(p)[u]\right] + R_1^g\left(f^{(0)}(p), f^{(0)}(q)\right)\left[f^{(1)}(q)[u]\right] \\ &\overset{(44)}{=} g^{(1)}\left(f^{(0)}(p)\right)\left[R_1^f(p,q)[u]\right] + R_1^g\left(f^{(0)}(p), f^{(0)}(q)\right)\left[f^{(1)}(q)[u]\right]. \end{aligned}$$

Consequently, by using (**II**) in (43) to estimate the term $g^{(1)}\left(f^{(0)}(p)\right)$, (**II**) in (42) to estimate the term $f^{(1)}(q)$, (**II**) in (45) to estimate the term $R_1^f(p,q)$, and (**II**) in (47) to estimate the term $R_1^g\left(f^{(0)}(p), f^{(0)}(q)\right)$, we may deduce that

$$\left|\left|R_1^h(p,q)[u]\right|\right|_W \leq ||g||_{\text{Lip}(\gamma,\Omega,W)}||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}\left(||q-p||_U^{\gamma-1} + \left|\left|f^{(0)}(q) - f^{(0)}(p)\right|\right|_V^{\gamma-1}\right)||u||_U. \tag{58}$$

The combination of (55) and (58) yields the estimate that

$$\left|\left|R_1^h(p,q)[u]\right|\right|_W \leq ||g||_{\text{Lip}(\gamma,\Omega,W)}\left(||f||_{\text{Lip}(\gamma,\Sigma,\Omega)} + 2^{\gamma-1}||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}^{\gamma}\right)||q-p||_U^{\gamma-1}||u||_U. \tag{59}$$

Taking the supremum over $u \in U$ with unit U-norm in (59) yields the estimate that

$$\left|\left|R_1^h(p,q)\right|\right|_{\mathbf{L}(V,W)} \leq ||g||_{\text{Lip}(\gamma,\Omega,W)}\left(||f||_{\text{Lip}(\gamma,\Sigma,\Omega)} + 2^{\gamma-1}||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}^{\gamma}\right)||q-p||_U^{\gamma-1}. \tag{60}$$

Finally, we complete the proof by combining the various estimates we have established for h to obtain the $\text{Lip}(\gamma,\Sigma,W)$-norm bound claimed in (41).

We start this task by combining (52) and (57) to deduce that for every $p, q \in \Sigma$ we have

$$\left|\left|R_0^h(p,q)\right|\right|_W \leq \begin{cases} ||g||_{\text{Lip}(\gamma,\Omega,W)}\left(2 + ||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}\right)||q-p||_U^{\gamma} & \text{if } ||q-p||_U > 1 \\ ||g||_{\text{Lip}(\gamma,\Omega,W)}\left(||f||_{\text{Lip}(\gamma,\Sigma,\Omega)} + 2^{\gamma}||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}^{\gamma}\right)||q-p||_U^{\gamma} & \text{if } ||q-p||_U \leq 1. \end{cases} \tag{61}$$

Moreover, the combination of (54) and (60) yields the estimate that

$$\left|\left|R_1^h(p,q)\right|\right|_{\mathbf{L}(V,W)} \leq \begin{cases} 2||g||_{\text{Lip}(\gamma,\Omega,W)}||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}||q-p||_U^{\gamma-1} & \text{if } ||q-p||_U > 1 \\ ||g||_{\text{Lip}(\gamma,\Omega,W)}\left(||f||_{\text{Lip}(\gamma,\Sigma,\Omega)} + 2^{\gamma-1}||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}^{\gamma}\right)||q-p||_U^{\gamma-1} & \text{if } ||q-p||_U \leq 1. \end{cases} \tag{62}$$

A consequence of (61) is that

$$\left|\left|R_0^h(p,q)\right|\right|_W \leq (1+2^{\gamma})\,||g||_{\text{Lip}(\gamma,\Omega,W)}\max\left\{||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}^{\gamma}, 1\right\}||q-p||_U^{\gamma}, \tag{63}$$

whilst a consequence of (62) is that

$$\left|\left|R_1^h(p,q)\right|\right|_{\mathbf{L}(V,W)} \leq \left(1+2^{\gamma-1}\right)||g||_{\text{Lip}(\gamma,\Omega,W)}\max\left\{||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}^{\gamma}, 1\right\}||q-p||_U^{\gamma-1}. \tag{64}$$

Therefore, by combining (50), (51), (63), and (64), we conclude both that $h = (h^{(0)}, h^{(1)}) \in \text{Lip}(\gamma,\Sigma,W)$ and that

$$||h||_{\text{Lip}(\gamma,\Sigma,W)} \leq (1+2^{\gamma})\,||g||_{\text{Lip}(\gamma,\Omega,W)}\max\left\{||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}^{\gamma}, 1\right\}. \tag{65}$$

□

Note that (41) is a stricter bound than (37), as for $\gamma \in (k, k+1]$,

$$\max\left\{||f||_{\text{Lip}(\gamma,\Sigma,\Omega)}^{\gamma}, 1\right\} \leq \max\left\{||f||_{\text{Lip}(\gamma)}^{k+1}, 1\right\}. \tag{66}$$

There is equality when $||f||_{\text{Lip}(\gamma)} \leq 1$ or $\gamma = 2$.

B.2. Lip$(2)-$norm of a Neural Network Layer

Lemma B.2 allows us to bound the Lip$(2)-$norm of a neural network (NN) given a bound on the Lip$(2)-$norm of each layer of a NN. We demonstrate this here for a simple NN.

Definition B.3. *(Fully Connected NN)* Let $m, n_{in}, n_{out}, n_h \in \mathbb{N}$ and f_θ be a fully connected NN with m layers, input dimension n_{in}, output dimension n_{out}, hidden dimension n_h, and activation function σ. Given an input $\mathbf{x} \in \mathbb{R}^{n_{in}}$, the output of the NN is defined as

$$\mathbf{y} = L^m(\cdots(L^1(\mathbf{x}))\cdots), \tag{67}$$

where $L^1 : \mathbb{R}^{n_{in}} \to \mathbb{R}^{n_h}$, $L^i : \mathbb{R}^{n_h} \to \mathbb{R}^{n_h}$ for $i = 2, \ldots, m-1$, and $L^m : \mathbb{R}^{n_h} \to \mathbb{R}^{n_{out}}$. Each layer is defined by

$$L^i(\mathbf{y}) = \begin{bmatrix} L^i_1(\mathbf{y}) \\ \vdots \\ L^i_\alpha(\mathbf{y}) \end{bmatrix} = \begin{bmatrix} \sigma(l^i_1(\mathbf{y})) \\ \vdots \\ \sigma(l^i_\alpha(\mathbf{y})) \end{bmatrix} = \begin{bmatrix} \sigma(W^i_1 \cdot \mathbf{y} + b^i_1) \\ \vdots \\ \sigma(W^i_\alpha \cdot \mathbf{y} + b^i_\alpha) \end{bmatrix}, \tag{68}$$

where $\mathbf{y} \in \mathbb{R}^\beta$, $W^i = [W^i_1, \ldots, W^i_\alpha]^T \in \mathbb{R}^{\alpha\times\beta}$ and $\mathbf{b}^i = [b^i_1, \ldots, b^i_\alpha]^T \in \mathbb{R}^\alpha$ are the learnable parameters and

$$(\alpha, \beta) = \begin{cases} (n_h, n_{in}), & i = 1, \\ (n_h, n_h), & i = 2, \ldots, m-1, \\ (n_{out}, n_h), & i = m. \end{cases} \tag{69}$$

Definition B.4. *(SiLU (Elfwing et al., 2018))* The activation function *SiLU* $: \mathbb{R} \to \mathbb{R}$ is defined as

$$\textit{SiLU}(y) = \frac{y}{1 + e^{-y}}. \tag{70}$$

Lemma B.5. *Let f_θ be a fully connected NN with activation function SiLU. Assume the input is normalised such that $\mathbf{x} = [x_1, \ldots, x_{n_{in}}]^T$ satisfies $|x_j| \leq 1$ for $j = 1, \ldots, n_{in}$. Then*

$$||L^i||_{\textit{Lip}(2)} \leq \left(\sum_{j=1}^{n} \max\left\{0.5||W^i_j||_2^2, 1.1||W^i_j||_2, \Gamma^i||W^i_j||_2 + |b^i_j|\right\}^2 \right)^{\frac{1}{2}}, \tag{71}$$

where

$$\Gamma^i = \sqrt{n_h}||W^{i-1}||_2 \left(\cdots \left(\sqrt{n_h}||W^2||_2 \left(\sqrt{n_{in}}||W^1||_2 + ||\mathbf{b}^1||_2\right) + ||\mathbf{b}^2||_2 \right) + \cdots \right) + ||\mathbf{b}^{i-1}||_2. \tag{72}$$

Proof. Consider $L^1 : A \to \mathbb{R}^n$, where $A \subset \mathbb{R}^{n_{in}}$ is the set of $\mathbf{x} = [x_1, \ldots, x_{n_{in}}]^T$ satisfying $|x_j| \leq 1$ for $j = 1, \ldots, n_{in}$. Each $L^1_j : \mathbb{R}^{n_{in}} \to \mathbb{R}$ is composed of a linear layer l^1_j and a SiLU. First, we show that

$$(L^1_j, \nabla L^1_j) = \left(\frac{l^1_j}{1 + e^{-l^1_j}}, \frac{e^{l^1_j}(e^{l^1_j} + l^1_j + 1)}{(e^{l^1_j} + 1)^2} W^1_j \right) \in \text{Lip}(2), \tag{73}$$

where $l^1_j(\mathbf{x}) = W^1_j \cdot \mathbf{x} + b^1_j$. So,

$$\max_{\mathbf{x}\in A} |L^1_j(\mathbf{x})| \leq \max_{\mathbf{x}\in A} |l^1_j(\mathbf{x})| \leq \sqrt{n_{in}}||W^1_j||_2 + |b^1_j| \tag{74}$$

and

$$||\nabla L^1_j||_2 \leq 1.1||W^1_j||_2. \tag{75}$$

Since L^1_j is at least twice differentiable,

$$\frac{|L^1_j(\mathbf{y}) - L^1_j(\mathbf{x}) - \nabla L^1_j(\mathbf{x})(\mathbf{y} - \mathbf{x})|}{||\mathbf{y} - \mathbf{x}||_2^2} = \frac{|(\mathbf{y} - \mathbf{x})^T \nabla^2 L^1_j(\mathbf{b})(\mathbf{y} - \mathbf{x}))|}{2|\mathbf{x} - \mathbf{y}|_2^2} \leq \frac{1}{2}\beta(\nabla^2 L^1_j(\mathbf{b})), \tag{76}$$

where $\mathbf{b} = \mathbf{x} + t(\mathbf{y} - \mathbf{x})$ for some $t \in (0, 1)$ and $\beta(A) = \max\{|\lambda_{\max}(A)|, |\lambda_{\min}(A)|\}$. Similarly,

$$\frac{||\nabla L_j^1(\mathbf{y}) - \nabla L_j^1(\mathbf{x})||_2}{||\mathbf{y} - \mathbf{x}||_2} = \frac{|\nabla^2 L_j^1(\mathbf{z})(\mathbf{y} - \mathbf{x})|_2}{|\mathbf{x} - \mathbf{y}|_2} \leq ||\nabla^2 L_j^1(\mathbf{z})||_2 = \beta(\nabla^2 L_j^1(\mathbf{z})), \tag{77}$$

where $\mathbf{z} = \mathbf{x} + \alpha(\mathbf{y} - \mathbf{x})$ for some $\alpha \in (0, 1)$. Now,

$$\nabla^2 L_j^1 = \frac{e^{l_j^1}(2 + 2e^{l_j^1} + l_j^1(1 - e^{l_j^1})}{(e^{l_j^1} + 1)^3} W_j^1 \otimes W_j^1, \tag{78}$$

which has one non-zero eigenvalue

$$\beta(\nabla^2 L_j^1) = \frac{e^{l_j^1}(2 + 2e^{l_j^1} + l_j^1(1 - e^{l_j^1})}{(e^{l_j^1} + 1)^3} ||W_j^1||_2^2 \tag{79}$$

satisfying

$$\max_{\mathbf{x} \in \mathbb{R}^{n_{in}}} \beta(\nabla^2 L_j^1(\mathbf{x})) = 0.5||W_j^1||_2^2. \tag{80}$$

Therefore,

$$||L_j^1||_{\text{Lip}(2)} = \max\left\{0.5||W_j^1||_2^2, 1.1||W_j^1||_2, \sqrt{n_{in}}||W_j^1||_2 + |b_j^1|\right\}, \tag{81}$$

and

$$||L^1||_{\text{Lip}(2)} = \left(\sum_{j=1}^{n} \max\left\{0.5||W_j^1||_2^2, 1.1||W_j^1||_2, \sqrt{n_{in}}||W_j^1||_2 + |b_j^1|\right\}^2\right)^{\frac{1}{2}}. \tag{82}$$

The calculations for subsequent layers are very similar, except that the input to each layer is no longer restricted to A. For example,

$$\begin{aligned} \max_{\mathbf{x} \in A} |L_j^2(L^1(\mathbf{x}))| &= \max_{\mathbf{x} \in A} |W_j^2 \cdot \text{SiLU}(W^1\mathbf{x} + \mathbf{b}^1) + b_j^2|, \\ &\leq \left(\sqrt{n_{in}}||W^1||_2 + ||\mathbf{b}^1||_2\right) ||W_j^2||_2 + |b_j^2|. \end{aligned} \tag{83}$$

In general

$$||L^i||_{\text{Lip}(2)} \leq \left(\sum_{j=1}^{n} \max\left\{0.5||W_j^i||_2^2, 1.1||W_j^i||_2, \Gamma^i||W_j^i||_2 + |b_j^i|\right\}^2\right)^{\frac{1}{2}}, \tag{84}$$

where

$$\Gamma^i = \sqrt{n_h}||W^{i-1}||_2\left(\cdots\left(\sqrt{n_h}||W^2||_2\left(\sqrt{n_{in}}||W^1||_2 + ||\mathbf{b}^1||_2\right) + ||\mathbf{b}^2||_2\right) + \cdots\right) + ||\mathbf{b}^{i-1}||_2. \tag{85}$$

□

B.3. Proof of Theorem 3.1

Theorem B.6. *Let f_θ be a FCNN with input dimension n_{in}, hidden dimension n_h, depth m, and activation function SiLU (Elfwing et al., 2018). Assuming the input $\mathbf{x} = [x_1, \ldots, x_{n_{in}}]^T$ satisfies $|x_j| \leq 1$ for $j = 1, \ldots, n_{in}$, then $f_\theta \in \text{Lip}(2)$ and*

$$||f_\theta||_{Lip(2)} \leq CP_{m!}(||W^1||_2, \ldots, ||W^m||_2, ||\mathbf{b}^1||_2, \ldots, ||\mathbf{b}^m||_2) \tag{86}$$

where C is a constant depending on n_{in}, n_h, and m, $\{W^i\}_{i=1}^m$ and $\{\mathbf{b}^i\}_{i=1}^m$ are the weights and biases of ith layer of f_θ, and $P_{m!}$ is a polynomial of order $m!$.

Proof. Lemma B.5 means that each layer L^j of f_θ is $\text{Lip}(2)$ with norm satisfying

$$||L^j||_{\text{Lip}(2)} \leq CP_j(||W_1||_2, \ldots, ||W_j||_2, ||\mathbf{b}_1||_2, \ldots, ||\mathbf{b}_j||_2), \tag{87}$$

where C is a constant depending on n and n_{in} and P_j is a jth order polynomial. Applying lemma B.2 gives the bound in (86). □

Table 4. A summary of the subset of the UEA-MTSCA datasets used in this paper.

Dataset	Dimension	Number of Observations	Classes
EigenWorms	6	17984	5
EthanolConcentration	3	1751	4
Heartbeat	61	405	2
MotorImagery	64	3000	2
SelfRegulationSCP1	6	896	2
SelfRegulationSCP2	7	1152	2

C. Experimental Details

C.1. Stacked Recurrent Models

The stacked recurrent models considered in this paper are based on the official implementation of S5 located at `https://github.com/lindermanlab/S5` (Smith et al., 2023). A recurrent block consists of a batch or layer normalisation, a recurrent layer, a GLU layer (Dauphin et al., 2017), dropout with rate 0.1, and a skip connection. A full model consists of a linear encoder, a number of stacked recurrent blocks, and a final linear layer. The four different recurrent layers considered are the linear recurrent unit, S5, S6, and MAMBA, where S6 refers to the selective state-space recurrence introduced by Gu & Dao (2023) and MAMBA refers to the combination of a gated MLP, convolution, and S6 recurrence that was also introduced by Gu & Dao (2023). S5 and LRU use batch normalisation, whereas S6 and MAMBA use layer normalisation.

C.2. CDE Models

NCDEs, NRDEs, and Log-NCDEs use a single linear layer as ξ_ϕ. NCDEs and NRDEs use FCNNs as their vector fields configured in the same way as their original papers (Kidger et al., 2020; Morrill et al., 2021). NCDEs use ReLU activation functions for the hidden layers and a final activation function of $\tanh$. NRDEs use the same, but move the $\tanh$ activation function to be before the final linear layer in the FCNN. Log-NCDEs use a FCNN with SiLU activation functions for the hidden layers and a final activation function of $\tanh$. NRDEs and Log-NCDEs take their intervals $r_{i+1} - r_i$ to be a fixed number of observations, referred to as the Log-ODE step.

C.3. Toy Dataset Details

On the toy dataset, all models use a hidden state of dimension 64 and Adam with a learning rate of 0.0001 (Kingma & Ba, 2017). LRU, S5, S6, and MAMBA use 6 blocks and S5, S6, and MAMBA use a state dimension of 64. S5 uses 2 initialisation blocks and MAMBA uses a convolution dimension of 4 and an expansion factor of 2. NCDEs, NRDEs, and Log-NCDEs use a FCNN with width 128 and depth 3 as their vector field. Furthermore, they all use Heun as their differential equation solver with a fixed stepsize of 0.01. NRDEs and Log-NCDEs use a Log-ODE step of 4 and a signature truncation depth of 2. Log-NCDEs do not use any $\mathrm{Lip}(\gamma)$ regularisation, i.e. $\lambda = 0$.

C.4. UEA-MTSCA and PPG-DaLiA Details

Table 4 provides details on the dimension, number of observations, and number of classes for the datasets chosen from the UEA-MTSCA for the experiments conducted in this paper. Care is necessary when using the EigenWorms dataset. As of June 1st 2024, the train and test splits obtained from `https://timeseriesclassification.com/description.php?Dataset=EigenWorms` contain repeated time series. The repeated data was removed for the experiments in this paper. Tables 5 and 6 give an overview of the hyperparameters optimised over during the UEA-MTSCA and PPG-DaLiA experiments for the stacked recurrent models and the NCDE models respectively. The optimisation was performed using a grid search of the validation accuracy for the UEA-MTSCA datasets and the mean squared error for the PPG-DaLiA dataset. All models and experiments used Adam as their optimiser and a batch size of 32, except the stacked recurrent models on PPG-DaLiA, which used a batch size of 4 due to memory constraints. NCDEs, NRDEs, and Log-NCDEs use Heun as their differential equation solver with a fixed stepsize of $1/\max\{500, 1 + (\text{Time series length}/\text{Log-ODE step})\}$, with Log-ODE step $= 1$ for NCDEs. Additionally, Log-NCDEs scale down their initial FCNN parameters by a factor of 1000 to reduce the starting $\mathrm{Lip}(2)$ norm of the vector field.

Table 5. Hyperparameters selected by the optimisation for LRU, S5, S6, and MAMBA on the UEA-MTSCA datasets and PPG-DaLiA dataset. The following abbreviations are used: EigenWorms (EW), EthanolConcentration (EC), Heartbeat (HB), MotorImagery (MI), SelfRegulationSCP1 (SCP1), SelfRegulationSCP2 (SCP2), and PPG-DaLiA (PPG). A ✗ denotes that the hyperparameter is not applicable to that model.

Hyperparameters	Options	Method			
		LRU	S5	S6	MAMBA
Learning Rate	10^{-3}	EW, MI, SCP2, PPG	EW, EC, PPG	EW, HB, MI, SCP2	EW, EC, PPG
	10^{-4}	EC, HB, SCP1	HB, MI, SCP1, SCP2	SCP1, PPG	HB, SCP2
	10^{-5}			EC	MI, SCP1
Include Time	True	EW, SCP1, PPG	EW, EC, MI, SCP2, PPG	EC, HB, MI, PPG	EW, EC, SCP2
	False	EC, HB, MI, SCP2	HB, SCP1	EW, SCP1, SCP2	HB, MI, SCP1, PPG
Hidden Dimension	16	PPG	EC, SCP1, PPG	EW, EC, HB, MI, SCP2	EW
	64	MI, SCP1	HB, SCP2	SCP1, PPG	EC, HB, SCP2
	128	EW, EC, HB, SCP2	EW, MI		MI, SCP1, PPG
Number of Layers	2	EC	EW, HB, SCP2	SCP1, SCP2, PPG	MI, SCP1
	4		MI, SCP1	EW, EC, HB, MI	EC, HB, PPG
	6	EW, HB, MI, SCP1, SCP2, PPG	EC, PPG		EW, SCP2
State Dimension	16	EW, EC	EC, HB	EC, HB, SCP1	SCP1
	64	SCP2	SCP1	EW, PPG	EW, MI, SCP2, PPG
	256	HB, MI, SCP1, PPG	EW, MI, SCP2, PPG	MI, SCP2	EC, HB
S5 Initialisation Blocks	2	✗	EC	✗	✗
	4	✗	HB, MI	✗	✗
	8	✗	EW, SCP1, SCP2, PPG	✗	✗
Convolution Dimension	2	✗	✗	✗	EW, HB, SCP2
	3	✗	✗	✗	MI, PPG
	4	✗	✗	✗	EC, SCP1
Expansion Factor	1	✗	✗	✗	EW, MI, SCP1
	2	✗	✗	✗	SCP2, PPG
	4	✗	✗	✗	EC, HB

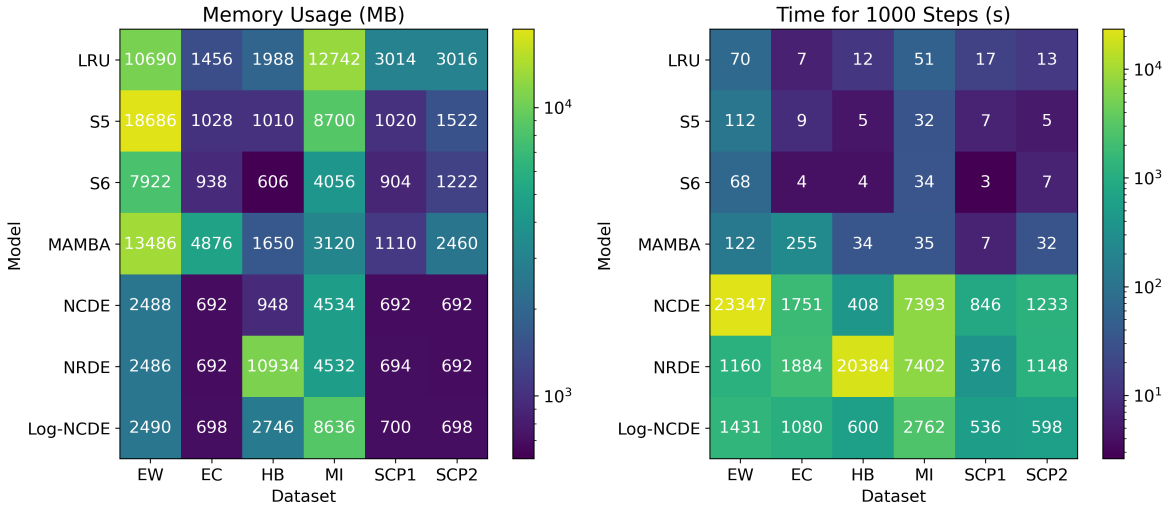
Table 6. Hyperparameters selected by the optimisation for NCDE, NRDE, and Log-NCDE on the UEA-MTSCA datasets and PPG-DaLiA dataset. Given the length of each timeseries in the PPG-DaLiA dataset, different choices were considered for the Log-ODE depth and step, which are shown here in red. The following abbreviations are used: EigenWorms (EW), EthanolConcentration (EC), Heartbeat (HB), MotorImagery (MI), SelfRegulationSCP1 (SCP1), SelfRegulationSCP2 (SCP2), and PPG-DaLiA (PPG). A ✗ denotes that the hyperparameter is not applicable to that model.

Hyperparameters	Options	Method		
		NCDE	NRDE	Log-NCDE
Learning Rate	10^{-3}	HB, PPG	EW, SCP1, SCP2, PPG	HB, SCP1, SCP2, PPG
	10^{-4}	EW, MI, SCP1	EC, HB, MI	EW, EC, MI
	10^{-5}	EC, SCP2		
Include Time	True	EW, EC, SCP2, PPG	EW, PPG	EC, HB, MI, SCP2, PPG
	False	HB, MI, SCP1	EC, HB, MI, SCP1, SCP2	EW, SCP1
Hidden Dimension	16	EC, HB, SCP1, PPG	EW, SCP1, SCP2, PPG	HB
	64	SCP2		EW, EC, MI, SCP2, PPG
	128	EW, MI	EC, HB, MI	SCP1
Vector Field (Depth, Width)	(2, 32)	HB	SCP1	HB
	(3, 64)	EC		SCP1
	(3, 128)	EW, SCP1, PPG	EC, SCP2	EC, MI, SCP2
	(4, 128)	MI, SCP2	EW, HB, MI, PPG	EW, PPG
Log-ODE (Depth, Step)	(1, 1)	✗	EC, MI, SCP2	
	(2, 2)	✗	HB	EC, HB
	(2, 4)	✗	SCP1	MI
	(2, 8)	✗		SCP2
	(2, 12)	✗		EW, SCP1
	(2, 16)	✗	EW	
	(1, 10)	✗		PPG
	(2, 10)	✗		
	(2, 100)	✗		
	(2, 1000)	✗	PPG	
Regularisation λ	10^{-3}	✗	✗	SCP1, SCP2
	10^{-6}	✗	✗	EW, MI, PPG
	0	✗	✗	EC, HB

C.5. Additional Memory and run time Results

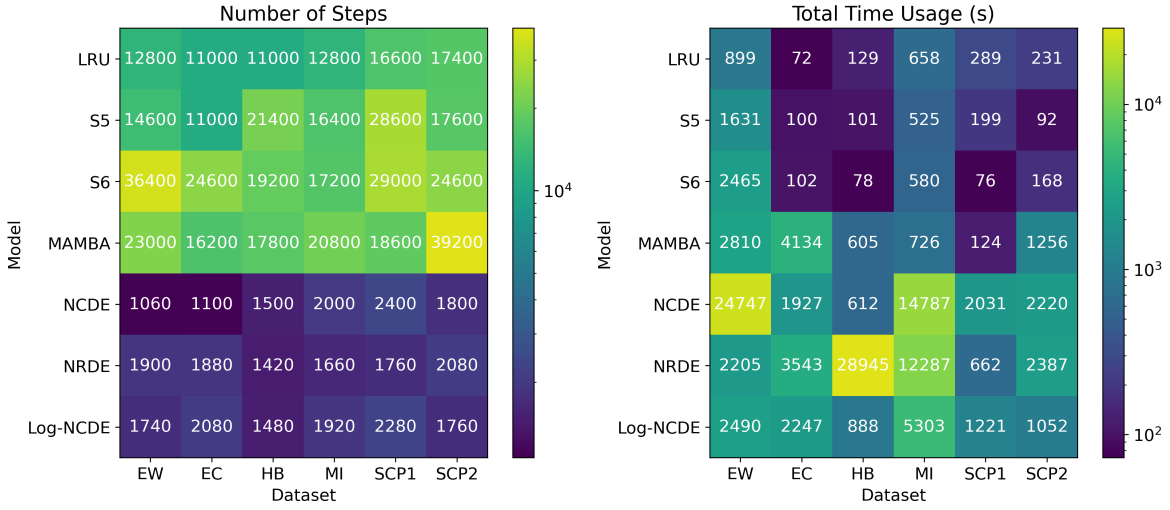
Models are compared on their average GPU memory usage and run time for the UEA-MTSCA datasets. In order to compare the models, 1000 steps of training were run on an NVIDIA RTX 4090 with each model using the hyperparameters obtained from the hyperparameter optimisation. The specific choices of the hyperparameters are listed in Tables 5 and 6. In addition to the time for 1000 steps and GPU memory usage, shown in Figures 6a and 6b respectively, the average number of total training steps taken to produce the results in Table 1 is recorded in Figure 6c. Combining the results for time for 1000 training steps and total number of training steps gives an approximation of the total run time on the same hardware, and these results are shown in Figure 6d.

Although the time per training step is lower for stacked recurrent models than NCDEs, NRDEs, or Log-NCDEs, they also require more training steps to converge. Additionally, NCDEs, NRDEs, and Log-NCDEs require less GPU memory. The largest contributors to the average run time of NCDEs are the datasets with the most observations, EigenWorms and MotorImagery. The positive impact of the Log-ODE method on computational burden is demonstrated empirically by the decrease in run time achieved by NRDEs and Log-NCDEs on EigenWorms when using a depth-2 Log-ODE method. When a depth-1 Log-ODE method is used, such as NRDEs on MotorImagery, the same decrease is not observed. Section 3.4 demonstrated that Log-NCDEs and NRDEs have the same asymptotic computational complexity. However, when using a depth-2 Log-ODE approximation and the same stepsize, NRDEs and Log-NCDEs exhibit drastically different run times on Heartbeat, a high-dimensional dataset. This difference is partly explained by the model's having different optimal hyperparameter choices, but even when using identical hyperparameters to the NRDE, Log-NCDE's run time for 1000 steps of training is 3017 seconds, whereas NRDE's run time is 20384 seconds. The remaining difference in run time is due to calculating the JVPs of f_θ using a batched function, as discussed in Section 3.4. If instead the JVPs are calculated using a for-loop, then Log-NCDEs run time increases to 15551 seconds.



(a) Memory

(b) Time



(c) Number of steps

(d) Total time

Figure 6. Memory, time for 1000 steps, number of steps, and approximate total time for each model and dataest from the UEA-MTSCA considered in this paper on an NVIDIA RTX 4090. The following abbreviations are used: EigenWorms (EW), EthanolConcentration (EC), Heartbeat (HB), MotorImagery (MI), SelfRegulationSCP1 (SCP1), and SelfRegulationSCP2 (SCP2).