RNNs, Neural CDEs, and Signature: an Exploration of Path-Dependent Dynamics

Bachelor Thesis

written by
John Skelton

supervised by
Prof. Dr. Josef Teichmann

ETH Zürich
Autumn 2022 / Spring 2023

Contents

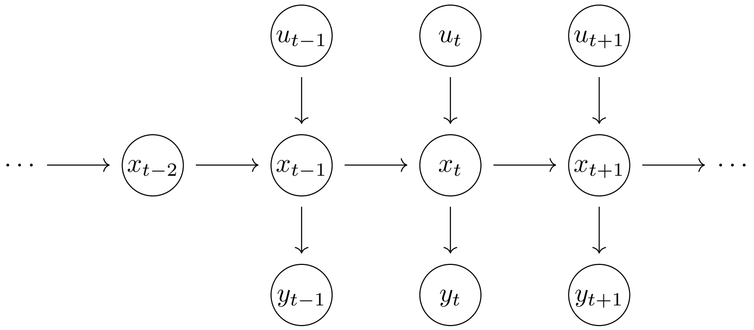
1 Introduction

Needless to say in 2023, machine learning has become the dominant modelling paradigm in big data applications. With the release of giant transformer networks such as ChatGPT and DALL·E, it is clearer than ever that machine learning technology will continue to exhibit unrivaled performance. However, the black-box nature of machine learning, and the appearance of strange phenomena like adversarial vulnerability - the ability to ruin a model by a tiny input perturbation - reveal that the spectacular performance of machine learning is no longer explained by theoretical understanding.

In this thesis, we seek to explore some of the theoretical results underpinning these successes in the setting of *sequential* data processing, where the object of interest is a *stochastic process* $(X_t)_{t\in\tau}$ - a time-ordered sequence of random variables. Unlike in natural language processing, where models like ChatGPT should be anticipative of the future, many domains of science and engineering require the arrow of time to be respected; the model should only incorporate past information. In the case of mathematical finance, this concept of causality, known as *adaptedness*, is of particular importance not least because incorporating future information constitutes insider trading. Consequently, given a time-series $(u_t)_{t\in\mathbb{Z}}$ - a stochastic process with a discrete time set - it is natural to consider a model of the form

$$\begin{cases} x_t = F(x_{t-1}, u_t) \\ y_t = h(x_t) \end{cases}$$

for some function F and a hidden state x_t. Such a model is clearly causal: its prediction y_t is based only on the input u_t and on the hidden state x_t which attempts to memorise the past. A graphical representation of this process is given below.



When the function F is a neural network, this is termed a *recurrent neural network* (RNN) due to the recycling of the hidden state, and before transformers emerged, these used to be the state-of-the-art models for sequential data. In particular, a neural network choice known as LSTM achieved superhuman gaming performance in OpenAI Five (2019) and used to be the go-to model for speech recognition by Google (2015). LSTMs still form a core component of modern deep learning architectures since they excel at short term memory [Lim+21].

Chapter 2 explores the universality of recurrent neural networks - can they approximate any sensible path-dependent functional arbitrarily well? Under a technical condition known as *fading memory* which encodes the fact that the hidden state cannot remember the entire past, this universality statement is proved, forming the dynamic analogue of the classical *Cybenko et al.* approximation theorems [Cyb89]. The chapter concludes with functional analytic insights into the fading memory property as weak-* continuity, as well as practical aspects stemming from neuroscience which reveal performance increases when the fading memory property is about to break.

Chapter 3 considers the continuous-time limit of recurrent neural networks when the time-step is taken to zero. The resulting differential equations provide an elegant and unified framework not just for RNNs, but for other neural architectures too, such as standard feedforward networks and diffusion models. Moreover, this theoretical framework allows one to leverage sophisticated techniques from control theory, differential geometry, rough path theory etc. to provide satisfying explanations for the surprising simplicity of training neural networks. We explore one avenue here via *signature* methods which yields universal approximation theorems and suggests that efficient signature variants may be promising features for general machine learning. Finally, the thesis culminates in the derivation of a signature surrogate for RNNs [Fer+21] with potentially attractive implementation strategies.

I am grateful for supervision by Prof. Josef Teichmann, whose deep insight and zeal have been the main driving forces behind this thesis. Furthermore, I would like to thank Prof. Chong Liu and Valentin Tissot-Daguette for invaluable resources.

2 RNNs in Discrete Time

2.1 Definitions and Basic Properties

State-Space Systems. The fundamental objects of interest in this chapter are *state-space systems*. These are open discrete-time dynamical systems in the form

$$\begin{cases} x_t = F(x_{t-1}, u_t) \\ y_t = h(x_t) \end{cases} \tag{2.1}$$

for some state transition function $F : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^n$, and a readout map $h : \mathbb{R}^n \to \mathbb{R}^k$. In this way, a state-space system is able to capture information from the input stream u_t and propagate it forwards in time through the hidden state x_t. Specifically, any path dependence is factorised as iterated function composition: the influence of u_{t-n} on y_t is filtered through n applications of the state transition F. We postpone specifying a time-horizon as this is delicate. An immediate question to ask is what generality this definition affords if we admit a sufficiently rich class of state transition functions.

As a first step, let us convince ourselves that state-space systems can model some important processes with varying degrees of path-dependence. The archetypal example of a system with no path-dependence is a time-homogeneous discrete-time Markov chain $(X_t)_{t \in \mathbb{Z}_+}$, whose evolution is governed by its matrix of transition probabilities $P_{ij} = \mathbb{P}[X_{t+1} = j | X_t = i]$. Specifically, the forward Kolmogorov equation asserts that the conditional distribution of X_n given an initial distribution vector $\alpha_0^j = P(X_0 = j)$ evolves in time via

$$\alpha_{t+1} = P^\top \alpha_t.$$

By the Markov property, the conditional distribution of X_{t+1} depends only on the most recent observation of X, so if we choose the current state X_t of the Markov chain as the control u_t, then the state-space system given by

$$\begin{cases} x_t = F(x_{t-1}, u_t) = P^T u_t \\ y_t = x_t \end{cases}$$

exactly reproduces the conditional probability $y_t = \mathbb{P}[X_{t+1} | X_t]$ of the Markov chain. Notice that the state transition does not depend on the past state x_{t-1}. This reflects the inherent path-independence of Markov chains.

A classical example of systems with path-dependence is linear finite impulse response (FIR) filters which are ubiquitous in engineering and filtering. These can also be exactly implemented by a state-space system if we admit linear state transitions and readout functions. For simplicity, consider a one-dimensional filter U with kernel $(K_t)_{t \in \{0,\dots,N\}}$, so that its output is given by the discrete convolution

$$U(u)_t = \sum_{j=0}^{N} K_j u_{t-j}.$$

To write this filter in state-space form, we choose an $(N+1)$-dimensional hidden state, and represent the state transition (in x_t) by the upper shift map given by a zero matrix with a superdiagonal of ones.

We define the readout map as the convolution of the hidden state and the kernel. The corresponding state-space model is

$$\begin{cases} x_t = F(x_{t-1}, u_t) = Ax_{t-1} + Bu_t \\ y_t = K * x_t = \sum_{j=0}^{N} K_j x_t^{j+1}, \end{cases}$$

where

$$A = \begin{pmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ & & & & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}.$$

It is clear that this state-space system coincides with the desired filter: the hidden state captures precisely the past $N + 1$ inputs, and the readout map convolves these with the kernel. Crucially, the nilpotency of the shift matrix encodes the finite memory of the hidden state, as any input is forgotten after $N + 1$ steps. This is analogous to the classical idea of turning an $AR(k)$ process into a Markov process by stacking the past $k - 1$ random variables as a vector. Processes whose path-dependence cannot be absorbed by such finite state augmentation are termed fully non-Markovian and appear for instance in rough volatility [BFG16, p.11].

These examples demonstrate that state-space systems span three distinct paradigms of path-dependence: none (as seen in Markov chains), finite (as in FIR filters), and full path dependence, which is the main focus of this paper.

Recurrent Neural Networks. A *recurrent neural network* (RNN) is a state-space system (2.1) whose state-transition function is a neural network - functions given by an alternating composition of affine and non-linear maps. Specifically, we consider a neural network with input dimension $d \in \mathbb{N}$, output dimension $k \in \mathbb{N}$, and depth $l \in \mathbb{N}$ to be a function of the type

$$\mathbb{R}^m \to \mathbb{R}^d \qquad x \mapsto A_l \circ \sigma \circ A_{l-1} \circ \ldots \circ \sigma \circ A_0(x),$$

where A_j are affine maps with compatible dimensions for composition, and $\sigma : \mathbb{R} \to \mathbb{R}$ is a non-linearity understood to be applied componentwise. The width of the neural network is the maximal dimension of the affine maps. The set of all neural networks with input dimension $d \in \mathbb{N}$, output dimension $k \in \mathbb{N}$, width h, and depth $l \in \mathbb{N}$ is denoted by $\mathcal{NN}_{d,k}^{h,l}$. The classical universal approximation theorem [Cyb89] now states the following.

Theorem 2.1 (Classical UAT)**.** *Let $K_d \subset \mathbb{R}^d$ be compact, $f \in C^0(K, \mathbb{R}^k)$, and let $\sigma : \mathbb{R} \to \mathbb{R}$ be a non-polynomial activation function. Then for all $\epsilon > 0$ there exist affine functions A_0, A_1 with compatible dimensions such that*

$$\|f - A_1 \circ \sigma \circ A_0\|_\infty < \epsilon.$$

In other words, the set of shallow neural networks $\mathcal{NN}_{d,k}^{\cdot,1}$ is dense in $C^0(K, \mathbb{R}^k)$.

It is important to point out that the classical UAT permits and even *requires* arbitrarily high hidden dimension between the affine maps. An alternative is to enforce bounded width but arbitrary depth, in which case the same density statement holds under mild conditions [KL20]. Geometrically, wide networks inject the domain into a high-dimensional space before compressing it componentwise and projecting back down, while deep networks alternate between affine maps and compression in a comparatively low dimension. Beyond the increased 'curvature' (higher Lipschitz constants, see [HKT20a]) produced efficiently by repeated non-linearities, depth has the additional benefit of improved local approximation - particularly the ReLU activation easily produces wavelet frames, which are the canonical choice for multi-scale approximation [SCC18]. On the other hand, width can be necessary to satisfy topological constraints on the data. This phenonenon is also manifests itself in the deep UAT [KL20]: the width is required to be the sum of the input and output dimensions *plus an extra two*. Bearing in mind the practical superiority of depth, we will be interested in establishing a universal approximation result for both arbitrary width, and arbitrary depth networks.

2.2 Universal Approximation on an Infinite Time-Horizon

While neural networks act as universal *function* approximators, RNNs act as universal *dynamical system* approximators. We begin with the simple case, namely universal approximation on a finite time-horizon $\{1, \dots, T\}$. In this case, the existence and uniqueness of a solution sequence $y \in (\mathbb{R}^k)^T$ to equation (2.1) for any input sequence $u \in (\mathbb{R}^d)^T$ is clear by initialising $y_1 = h(F(0, u_1))$ and iterating (2.1). The operator $U : (\mathbb{R}^d)^T \to (\mathbb{R}^k)^T$ which associates this solution to an input is called the filter associated to the RNN. If we consider state transition functions which are continuous, and a compact domain $(K_d)^T \subset (\mathbb{R}^d)^T$, then the supremum norm $\|\|U\|\|_\infty = \sup_{u \in K_d} \|U(u)\|_\infty$ offers a natural metric for uniformly approximating filters. The UAT in this dynamic setting now takes the following form.

Theorem 2.2 (Dynamic UAT on a finite time-horizon)**.** *Consider the state-space system (2.1) driven by continuous functions $F : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^n$ and $h : \mathbb{R}^n \to \mathbb{R}^k$. Let $K_d \subset \mathbb{R}^d$ be compact, and let $U : (K_d)^T \to (\mathbb{R}^k)^T$ be the associated filter. Then for all $\epsilon > 0$ there exists an RNN of the form*

$$\begin{cases} x_t = \sigma(Ax_{t-1} + Cz_t) \\ y_t = Wx_t \end{cases}$$

for compatible affine maps A, C, W, such that its associated filter U_{RNN} satisfies

$$\|\|U - U_{RNN}\|\|_\infty < \epsilon.$$

A simple proof of this theorem which exploits the finite time-horizon and the classical UAT can be found in [SZ06]. However, the approximation quality depends fundamentally on the *fixed* nature of the time horizon used to invoke the UAT, which leads to a degradation of quality if the time-horizon is extended. Any approximation on an infinite time-horizon would have to be uniform in time. In fact, this reveals a practical motivation for an extension of this theorem to an infinite time-horizon, since it provides confidence that this statement is not merely a corollary of the classical UAT as the proof of theorem 2.2 would suggest. Furthermore, considering an infinite time-horizon forces stationarity, which is of course convenient for modelling stationary time-series.

Three problems arise in an infinite time-horizon, which will guide us in the proof:

- First, existence and uniqueness of solutions to (2.1) is no longer self-evident on an infinite time-horizon as we cannot initialise and iterate. If this existence and uniqueness does hold, we say that the system has the *echo state property* (ESP). Under this condition, the RNN determines a well-defined filter $U : (K_d)^{\mathbb{Z}} \to (K_k)^{\mathbb{Z}}$ with its output given by the unique solution to (2.1).

- Second, recalling that any RNN factors path-dependence via iterated function composition, we must strengthen the meaning of continuity for a filter because information will be lost over time; RNNs cannot store the entire path in their hidden state, and can generally only model filters which also have decaying memory, which will be called the *fading memory property* (FMP).

- Finally, the set of uniformly bounded sequences ${K_d}^{\mathbb{Z}}$ for a compact set $K_d \subset \mathbb{R}^d$ is *not* compact in $\ell^\infty(\mathbb{R}^d)$. For instance, the delta sequence elements δ_i, which are zero except for a 1 in the i-th time-coordinate, have no uniformly convergent subsequence as $i \to \infty$, so ${K_d}^{\mathbb{Z}}$ cannot be sequentially compact. The fading memory property will allow us to re-establish the compactness of the uniformly bounded sequences by changing the topology.

To solve the latter two problems, we will introduce a weighted sequence space. Let $w : \mathbb{N} \to (0, 1]$ be a sequence with zero limit. The associated weighted norm on $(\mathbb{R}^d)^{\mathbb{Z}_-}$ is given by

$$\|z\|_w = \sup_{t \in \mathbb{Z}_-} \|z_t w_{-t}\|.$$

The space

$$\ell^w_-(\mathbb{R}^d) := \{z \in (\mathbb{R}^d)^{\mathbb{Z}_-} : \|z\|_w < \infty\}$$

is a Banach space endowed with the weighted norm $\|\cdot\|_w$ [GO18a, Prop. 5.2]. It contains sequences which grow at most as quickly as the weighting sequence decays. Correspondingly, $(\ell_-^\infty(\mathbb{R}^d), \|\cdot\|_\infty) \hookrightarrow (\ell_-^w(\mathbb{R}^d), \|\cdot\|_w)$ embeds continuously. We say that a filter $U : (K_d)^{\mathbb{Z}_-} \to (K_k)^{\mathbb{Z}_-}$ has the fading memory property with respect to the weighting sequence w if it is a continuous map between $((K_d)^{\mathbb{Z}_-}, \|\cdot\|_w)$ and $((K_k)^{\mathbb{Z}_-}, \|\cdot\|_w)$. It is immediate to check that the fading memory property implies continuity in the sense of the $\|\cdot\|_\infty$-norm, regardless of the weighting sequence. We can apply these topological considerations in the following way. Obviously, ${K_d}^{\mathbb{Z}_-} \subset \ell_-^\infty(\mathbb{R}^d) \subset \ell_-^w(\mathbb{R}^d) \subset (\mathbb{R}^d)^{\mathbb{Z}_-}$. It turns out that on ${K_d}^{\mathbb{Z}_-}$, the subspace topology inherited from $\ell_-^w(\mathbb{R}^d)$ coincides with the product topology inherited from $(\mathbb{R}^d)^{\mathbb{Z}_-}$ [GO18a, Corollary 2.7]. As ${K_d}^{\mathbb{Z}_-}$ is a product of compact sets, $({K_d}^{\mathbb{Z}_-}, \|\cdot\|_w)$ is a compact space by Tychonoff's theorem. Remarkably, this is independent of the weighting sequence; all weighted norms induce the product topology on ${K_d}^{\mathbb{Z}_-}$, so the fading memory property is actually a purely topological notion and does not convey any information about the rate at which memory decays. Fading memory with respect to any weighting sequence implies fading memory with respect to any other.

Finally, before stating the theorem, we introduce two ideas which allow us to approximate filters that do not even originate from a state-space system (2.1). This is an important generalisation because long memory processes do not admit such finite-dimensional state-space representations. In fact, any finite-dimensional state-space system has an exponentially decaying autocovariance function due to the iterated application of the transition function, so any processes with slower than exponential autocovariance decay *cannot* have such a representation [CP98, Theorem 2.1].[1]

- **Causality**: a filter is considered to be causal if for any $z, w \in (K_d)^{\mathbb{Z}}$ such that $z_t = w_t \;\; \forall t \leq \tau$, then $U(z)_\tau = U(w)_\tau$. In other words, the output at time $\tau \in \mathbb{Z}$ depends on an input $(z_t)_{t\in\mathbb{Z}}$ only through $(z_t)_{t\leq\tau}$.
- **Time-invariance**: a filter is time-invariant if it commutes with the time shift operators given by $(T_\tau(u))_t = u_{t+\tau}$ for all $\tau \in \mathbb{Z}$.

Naturally, a causal time-invariant filter defined on $(K_d)^{\mathbb{Z}_-}$ already determines its extension to $(K_d)^{\mathbb{Z}}$ by time-shifting, hence we interchange between these two domains freely. There is a bijection between causal time-invariant filters and $\mathbb{R}^d$-valued functionals on $(\mathbb{R}^d)^{\mathbb{Z}_-}$: to each causal time-invariant filter U, we may associate a functional $H_U : (\mathbb{R}^d)^{\mathbb{Z}_-} \to \mathbb{R}^k$ defined by $H_U(z) = U(z)_0$, and conversely, we may associate a causal time invariant filter $U(z) = H((T_t \circ P)(z))$ to any functional, where $P : (\mathbb{R}^d)^{\mathbb{Z}} \to (\mathbb{R}^d)^{\mathbb{Z}_-}$ is the natural projection. This bijection preserves the FMP if a filter or functional has it [GO18a, Prop. 2.11]. Therefore, to each causal and time-invariant filter $U : {K_d}^{\mathbb{Z}} \to (\mathbb{R}^k)^{\mathbb{Z}}$ possessing the FMP, we associate a functional $H_U : {K_d}^{\mathbb{Z}_-} \to \mathbb{R}^k$ which then inherits the FMP. The purpose of this is that we are left with continuous functionals on a compact space, which will allow us to invoke the Stone-Weierstrass theorem. Without causality and time-invariance, this reduction would not be possible, and the codomain would remain infinite-dimensional.

With these notions fixed, we are able to give a universal approximation theorem for RNNs in the category of causal, time-invariant filters with the FMP when the state transitions are shallow neural networks. Up to an elementary extension from one dimension to k dimensions, the theorem and its proof are both adapted from [GO18a, Theorem 4.1] and [GO18b]. In the following, fix $I_d = [-1, 1]^d$ and a sigmoidal function $\sigma : \mathbb{R} \to \mathbb{R}$.

Theorem 2.3 (Dynamic universal approximation on an infinite time-horizon)**.** *Let $U : (I_d)^{\mathbb{Z}_-} \to (\mathbb{R}^k)^{\mathbb{Z}_-}$ be a causal and time-invariant filter with the FMP. Then for any $\epsilon > 0$, there is an RNN of the form*

$$\begin{cases} x_t = \sigma(Ax_{t-1} + Cz_t) \\ y_t = Wx_t \end{cases}$$

[1] As finite-dimensional state-space systems, RNNs are notorious for struggling to learn long-term dependence. This has led to the development of the LSTM (long short-term memory) network which learns a gating procedure to control the rate of memory decay as it is trained.

with compatible affine maps A, C, and W such that its associated filter $U_{RNN} : (I_d)^{\mathbb{Z}_-} \to (\mathbb{R}^k)^{\mathbb{Z}_-}$ satisfies

$$\left|\left|\left| U - U_{RNN} \right|\right|\right|_\infty < \epsilon.$$

If the RNN has the ESP, then the filter associated to it is unique.

Proof. The proof proceeds in three steps. First, as it is not given that U has a state-space representation, we approximate the functional associated to U by a functional with a polynomial state transition. This polynomial is then approximated by a neural network. Finally, we transfer the approximation of the state transition function back to the entire filter.

Step 1. Recall that we associate a functional $H_U : {K_d}^{\mathbb{Z}_-} \to \mathbb{R}^k$ to the filter U, and that this inherits the FMP. By the previous discussion about the FMP, it follows that $H_U \in C^0((K_d)^{\mathbb{Z}_-}, \mathbb{R}^k)$ where $(K_d)^{\mathbb{Z}_-}$ is equipped with the product topology. A natural way to obtain a dense family in this space is via the Stone-Weierstrass theorem 4.1. For this purpose, we define a non-homogeneous state-affine (SAS) system as a state space transformation

$$\begin{cases} x_t = p(z_t)x_{t-1} + q(z_t) \\ y_t = Wx_t \end{cases} \tag{2.2}$$

for polynomials p and q with matrix coefficients, and a linear map W. Now let $K, L > 0$ such that

$$K < \frac{L}{L+1} < 1, \tag{2.3}$$

and consider SAS-filters which satisfy $\sup_{z\in I_d} \sigma_{max}(p(z)) < K$ and $\sup_{z\in I_d} \sigma_{max}(q(z)) < K$. The singular value bound on p guarantees convergence of the (closed-form) solution

$$\begin{cases} x_t = \displaystyle\sum_{j=0}^{\infty} \left(\prod_{k=0}^{j-1} p(z_{t-k}) \right) q(z_{t-j}) \\ y_t = Wx_t \end{cases} \tag{2.4}$$

but the proof of this [GO18b, Prop. 14] is technical and omitted. Meanwhile, the singular value on q yields a bound on the output. Furthermore, such filters have the ESP and FMP, and the corresponding functionals do indeed form a subalgebra of $C^0((K_d)^{\mathbb{Z}_-}, \mathbb{R}^k)$ [GO18b, Prop. 17]. To prove that the subalgebra is point-separating in every coordinate of $\mathbb{R}^k$, the FIR linear filters corresponding to nilpotent state transitions (see section 2.1) already suffice. More precisely, choose a coordinate $i_0 \in \{1, \ldots, d\}$, let $z_1, z_2 \in K^{\mathbb{Z}_-}$ such that $z_1 \neq z_2$, and let $t_0 \in \mathbb{N}$ be the first time where $(z_1^{i_0})_{-t_0} \neq (z_2^{i_0})_{-t_0}$. Let $A \in \mathbb{R}^{(n+1)\times(n+1)}$ be the upper shift matrix consisting of a superdiagonal of $K/2$, let $B \in \mathbb{R}^{(n+1)\times d}$ be zero except for a $K/2$ in entry $(n+1, i_0)$, let $W \in \mathbb{R}^{(n+1)\times k}$ be the projection to coordinate i_0, and consider the system

$$\begin{cases} x_t = Ax_{t-1} + Bz_t \\ y_t = Wx_t = x_t^{i_0}. \end{cases} \tag{2.5}$$

Both A and B satisfy the spectral bound as their maximum singular value is $K/2$. By definition of t_0, the filter $U^{A,B}$ associated to (2.5) satisfies

$$U^{A,B}(z_1 - z_2)_0 = \sum_{j=0}^{t_0} A^j B(z_1 - z_2)_{-j} = A^{t_0} B(z_1 - z_2)_{-t_0} = \frac{K^{t_0+1}}{2^{t_0+1}} \left((z_1^{i_0})_{-t_0} - (z_2^{i_0})_{-t_0} \right) \neq 0,$$

or in other words, $H_{U^{A,B}}(z_1) \neq H_{U^{A,B}}(z_2)$ as required. All constant functionals can be easily obtained by setting p zero, q constant, and reading out the desired vector component with a linear map. By the $\mathbb{R}^k$-valued Stone-Weierstrass theorem (4.1), the subalgebra of SAS-filters satisfying the spectral bound is dense in $C^0((K_d)^{\mathbb{Z}_-}, \mathbb{R}^k)$. Therefore, for any $\epsilon_1 > 0$, there exists a SAS-filter U_{SAS} such that

$$\left|\left|\left| H_U - H_{SAS} \right|\right|\right|_\infty < \epsilon_1.$$

It is straightforward to show [GO18a, Prop. 2.12] that this estimate transfers to the filter itself, that is,

$$|||U - U_{SAS}|||_\infty < \epsilon_1.$$

This completes the first part of the proof, as we can approximate *any* causal time-invariant filter possessing the FMP with a filter that has a polynomial state system representation.

Step 2. Next, we show that SAS-filters can be approximated by RNNs. Define the state transition map corresponding to U_{SAS} as

$$\begin{aligned} F_{SAS} : \overline{B(0,L)} \times I_d &\longrightarrow \mathbb{R}^{N_1}. \\ (x,z) &\longmapsto p(z)x + q(z), \end{aligned}$$

where $N_1 \in \mathbb{N}$ is the dimension of the state space of the SAS-filter. The condition $K < 1$ renders F_{SAS} a Lipschitz contraction in the first variable. Indeed, for any $(x,z),(y,z) \in \overline{B(0,L)} \times I_d$,

$$\|F_{SAS}(x,z) - F_{SAS}(y,z)\| \leq \|p(z)x - p(z)y\| \leq \|p(z)\|_2 \|x-y\| < K\|x-y\|.$$

Similarly,

$$\|F_{SAS}\|_\infty = \sup_{(x,z)\in\overline{B(0,L)}\times I_d} \|p(z)x+q(z)\| \leq \sup_{(x,z)\in\overline{B(0,L)}\times I_d} \|p(z)\|_2\|x\| + \|q(z)\| \leq KL + L < L,$$

where the last inequality follows from condition (2.3) on L. Hence F_{SAS} actually maps into $\overline{B(0,L)}$, which allows us to iterate the function. At this point, the classical UAT implies that for any $\epsilon_2 > 0$, there exists a neural network

$$\begin{aligned} F_{NN} : \overline{B(0,L)} \times I_d &\longrightarrow \mathbb{R}^{N_1}. \\ (x,z) &\longmapsto E\sigma(Gx + Cz), \end{aligned}$$

which satisfies

$$\|F_{SAS} - F_{NN}\|_\infty < \epsilon_2.$$

For ϵ_2 small enough, this bound implies that F_{NN} also maps into $\overline{B(0,L)}$.

Step 3. To begin the final step - extending this internal approximation to the entire filter - we remark that the state transition F_{NN} has an associated filter U_{NN} [GO18a, Theorem 3.1 (i)]. This is essentially a consequence of Schauder's fixed point theorem and depends critically on the compactness of $(K_d)^{\mathbb{Z}_-}$ equipped with the product topology, but beware that we do not have the ESP, and therefore no uniqueness of the filter. Let us now leverage the Lipschitz contractivity to show that for any $\epsilon_3 > 0$,

$$\|F_{NN} - F_{SAS}\|_\infty < (1-K)\epsilon_3 \implies |||U_{NN} - U_{SAS}|||_\infty \leq \epsilon_3.$$

Let $z \in (K_d)^{\mathbb{Z}_-}$. Then for any $t \in \mathbb{Z}_-$,

$$\begin{aligned} \|U_{NN}(z)_t - U_{SAS}(z)_t\| &= \|F_{NN}(U_{NN}(z)_{t-1}, z_t) \ - \ F_{SAS}(U_{SAS}(z)_{t-1}, z_t)\| \\ &\leq \|F_{NN}(U_{NN}(z)_{t-1}, z_t) - F_{SAS}(U_{NN}(z)_{t-1}, z_t)\| \\ &\qquad + \|F_{SAS}(U_{NN}(z)_{t-1}, z_t) - F_{SAS}(U_{SAS}(z)_{t-1}, z_t)\| \\ &\leq \|F_{NN}(U_{NN}(z)_{t-1}, z_t) - F_{SAS}(U_{NN}(z)_{t-1}, z_t)\| + K\|U_{NN}(z)_{t-1} - U_{SAS}(z)_{t-1}\|. \end{aligned}$$

Here, the first inequality follows from the triangle inequality, and the second from the contractivity of F_{SAS}. Applying this reasoning to the second term n times yields

$$\begin{aligned} \|U_{NN}(z)_t - U_{SAS}(z)_t\| &\leq \|F_{NN}(U_{NN}(z)_{t-1}, z_t) - F_{SAS}(U_{NN}(z)_{t-1}, z_t)\| \\ &+ K\|F_{NN}(U_{NN}(z)_{t-2}, z_{t-1}) - F_{SAS}(U_{NN}(z)_{t-2}, z_{t-1})\| + \dots \\ &+ K^{n-1}\|F_{NN}(U_{NN}(z)_{t-n}, z_{t-(n+1)}) - F_{SAS}(U_{NN}(z)_{t-n}, z_{t-(n+1)})\| \\ &+ K^n\|U_{NN}(z)_{t-n} - U_{SAS}(z)_{t-n}\|. \end{aligned}$$

Now if $\|F_{NN} - F_{SAS}\|_\infty < (1-K)\epsilon_3$, then

$$\begin{aligned} \|U_{NN}(z) - U_{SAS}(z)\|_\infty &= \sup_{t\in\mathbb{Z}_-} \|U_{NN}(z)_t - U_{SAS}(z)_t\| \\ &\leq 2LK^n + \left(1 + \ldots + K^{n-1}\right)(1-K)\,\epsilon_3 \xrightarrow{n\to\infty} \frac{1}{1-K}(1-K)\epsilon_3 = \epsilon_3. \end{aligned}$$

As $z \in (K_d)^{\mathbb{Z}_-}$ was arbitrary, we conclude that

$$|||U_{NN} - U_{SAS}|||_\infty < \epsilon_3.$$

Composing these two filters with the readout map W, and choosing ϵ_1, ϵ_2, and ϵ_3 conveniently yields the desired approximation

$$|||U - U^W_{NN}|||_\infty \leq |||U - U^W_{SAS}|||_\infty + |||U^W_{SAS} - U^W_{NN}|||_\infty \leq \frac{\epsilon}{2} + \frac{\epsilon}{2} \leq \epsilon.$$

Note that the neural network is *not* yet in the form stated in the theorem since it is followed an affine map E. To write it in the desired form, we let $A = GE$ and verify that there exists a filter associated to F_{NN} (recalling the lack of uniqueness!) which coincides with the desired filter. This last part is arduous and hence omitted. ■

An immediate extension of this theorem to deep networks follows from replacing the classical UAT in step 2 of the previous proof by the deep UAT [KL20]. We state this formally in the following corollary.

Corollary 2.3.1. *Let $U : (I_d)^{\mathbb{Z}_-} \to (\mathbb{R}^k)^{\mathbb{Z}_-}$ be a causal and time-invariant filter with the FMP, and let $\sigma : \mathbb{R} \to \mathbb{R}$ be non-affine and continuously differentiable in at least one point. Then for any $\epsilon > 0$, there exists a neural network $\phi \in \mathcal{NN}^{2N+d+2}_{N+d,N}$ for some $N \in \mathbb{N}$, and there exists an RNN of the form*

$$\begin{cases} x_t = \phi(x_t, z_t) \\ y_t = Wx_t \end{cases} \tag{2.6}$$

such that its associated filter $U_{RNN} : (I_d)^{\mathbb{Z}_-} \to (\mathbb{R}^k)^{\mathbb{Z}_-}$ satisfies

$$|||U - U_{RNN}|||_\infty < \epsilon. \tag{2.7}$$

If the RNN has the ESP, then the filter associated to it is unique.

Proof. The only part of the proof that requires modification is step 2. Instead of approximating

$$\begin{aligned} F_{SAS} : \overline{B(0,L)} \times I_d &\longrightarrow \mathbb{R}^N. \\ (x, z) &\longmapsto p(z)x + q(z), \end{aligned}$$

with a wide neural network, we invoke the deep UAT [KL20] to obtain a neural network $NN \in \mathcal{NN}^{2N+d+2}_{N+d,N}$ which approximates F_{SAS} uniformly. Note that this network is of the type $\phi(x) = A_l \circ \sigma \circ A_{l-1} \circ \ldots \circ \sigma \circ A_0(x)$ for compatible affine maps. As in the previous proof, we still need to absorb the outermost affine map A_l into the network. This part was omitted for its technicalities, but inspecting the original proof of [GO18a, Theorem 4.1] reveals that there is no difference between wide and deep networks here; the same reservoir morphism applies, finishing the proof. ■

Comments.

- Instead of a polynomial state transition and a linear readout map, one could also generate a relevant point-separating subalgebra with linear state transitions and polynomial readout maps [GO18b, Corollary 3.4]. However, the subject of reservoir computing reveals that it is favourable to absorb the complexity of the system into the state transition. The training process in this setting amounts to regressing the readout map, hence it is desirable for this map to be as simple as possible. A polynomial readout would inherit all the curses of high-order polynomial regression, such as poor generalisation properties and devastating condition numbers.

- In general, the approximating RNN may lack the ESP and FMP. Sufficient conditions are given in [GO18a, Corollary 3.2]. For example, if the activation σ is differentiable, then $\|A\|_2 \cdot \sup_{x\in\mathbb{R}} \sigma'(x) < 1$ implies both the ESP and FMP. Alternatively, the follow-up paper [GO21] by the same authors showed that if the activation function is Lipschitz continuous and bounded, then the subset of RNNs satisfying the ESP and FMP is already dense. RNNs which break the FMP or ESP do not contribute to the universality result. However, an emerging strand of literature on criticality in neuroscience suggests that it is desirable to tune reservoirs close to chaos; remarkable improvements in performance are observed when the FMP is about to break but still satisfied [BNL04].

- The FMP is the fundamental property which allows us to extend to an infinite horizon. In classical control theory, the FMP is also the missing link in the spurious folklore theorem that any continuous causal linear time-invariant filter $U : \ell^\infty_-(\mathbb{R}) \to \ell^\infty_-(\mathbb{R})$ has a convolution representation
$$U(z)_t = \sum_{j\in\mathbb{Z}_-} h_j z_{t+j}$$
for some $h \in \ell^1_-(\mathbb{R})$. This statement becomes true if U has the FMP. Indeed, the FMP can be reformulated as continuity in the compact open topology. In $\ell^\infty_-(\mathbb{R})$, the compact-open topology coincides with the weak-* topology, so that the FMP amounts to weak-* continuity of the associated functional [BC85, Section IX]! As any weak-* continuous linear functional on a dual space is represented by an element of the predual (which is $\ell^1_-(\mathbb{R})$ here), we get exactly the desired convolution representation. Furthermore, this is an equivalence: LTI systems admit a convolution representation if and only if they have the FMP [BC85, Theorem 5]. An example of a causal continuous LTI system which does not have the FMP and hence no convolution representation is a Banach limit, which is a Hahn-Banach extension of the limit operator defined on the convergent sequences $c \subseteq \ell^\infty_-(\mathbb{R})$. Its impulse response is clearly zero, so its convolution representation suggests that the operator should be identically zero - a contradiction. Due to the relationship between the axiom of choice and explicit elements in $(\ell^\infty_-(\mathbb{R}))^*$ not represented by sequences in $\ell^1_-(\mathbb{R})$, examples of continuous causal LTIs which fail the FMP are necessarily non-constructive. An example of a *non-linear* system which fails to have the FMP is the peak-hold operator $U(z)_t = \sup_{\tau\le t} \|z_\tau\|$ as it can maintain arbitrarily long memory in the supremum.

3 Towards Continuous-Time RNNs: Neural Controlled Differential Equations

In light of the discussion on width vs. depth following Theorem 2.1, it is natural to ask what the limiting case of *infinite* or *continuous* depth neural networks corresponds to[2]. It will turn out that this yields a differential equation, and as is prevalent in mathematics, passing from discrete to continuous objects allows us to leverage powerful analytical tools and obtain more parsimonious models. The field of neural differential equations (popularised by [Che+18]) considers neural networks as differential equations, and yields a unified framework for many architectures - feedforward networks are ODEs, RNNs are CDEs, diffusion models are SDEs, etc. [Kid21]. In this section we shall study RNNs as CDEs, link this approach to signature methods, and conclude with a way to model generic path-dependent dynamics by a signature-based RNN surrogate.

[2] For the sake of completeness, infinite width networks are in many cases considered to correspond to Gaussian processes if network parameters are distributed i.i.d. Gaussian, as a consequence of the central limit theorem.

3.1 Universal Approximation for Neural ODEs

Neural ODEs. We may rewrite a neural network

$$NN(x) = A_m \circ \sigma \circ A_{m-1} \circ \ldots \circ \sigma \circ A_0(x)$$

as a dynamical system by splitting the layer evaluations

$$x \mapsto \sigma \circ A_0(x) \mapsto \sigma \circ A_1 \circ \sigma \circ A_0(x) \mapsto \ \ldots \ \mapsto NN(x).$$

Adding an identity to each layer transition yields a *residual* network characterised by the system

$$x_{n+1} = x_n + \sigma \circ A_n(x_n). \tag{3.1}$$

Residual networks hence model the *difference* between layers. Notice that (3.1) is an explicit Euler discretisation (with unit time-step) of the ODE

$$\frac{dx_t}{dt} = \sigma \circ A_t(x_t).$$

Each layer of the network corresponds to one Euler step. Therefore, one may consider residual networks as discretised ODEs, or conversely, ODEs as infinite-depth residual networks. This observation has spawned the topic of neural ODEs, which considers a differential equation driven by a neural network,

$$\frac{dx_t}{dt} = NN(x_t), \quad x_0 = x$$

and attempts to approximate a map $x \mapsto F(x)$ by the flow of the differential equation $x \mapsto x_T$ after some time T. The objective is to choose the neural network so as to best transport the initial conditions x to the desired target $F(x)$ (compare figure 1). In practice, neural ODEs are implemented by calling an ODE solver to approximate this, effectively yielding a network with *adaptive* depth as the ODE solver chooses its step size. It is important to note that neural ODEs are susceptible to constraints on the width just as neural networks are. Since the trajectories of an ODE cannot cross - this would violate uniqueness of the solution - one must augment the ambient dimension by zero-padding to provide sufficient space for the flow. This is exemplified in figure 1 where the network is constrained to two dimensions and struggles to separate two concentric circles.

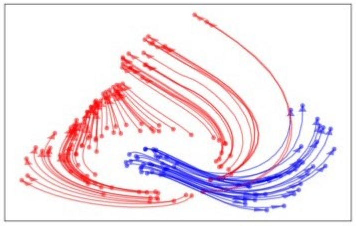


Figure 1: Neural ODE as a flow. Figure from [DDT19]

Neural ODEs which are constrained to the ambient dimension are in fact *not* universal approximators. Whenever a flow is well-defined by the Picard-Lindelöf theorem, it is a homeomorphism and must preserve topological properties of the input. This means that the separation of two circles in figure 1 is impossible [DDT19] in two dimensions. However, *augmented* neural ODEs, where the input is embedded into a higher dimension, are universal approximators. We summarise this in the following theorem, stated as in [Kid21, Theorem 2.12] and proved as in [Zha+20, Theorem 7]. Here, $L(U, V)$ stands for the set of affine maps between two spaces U and V.

Theorem 3.1 (Augmented Neural ODEs are Universal Approximators)**.** *Fix $d, d_l, d_o \in \mathbb{N}$ with $d_l = d + d_o$. For $f \in \mathcal{NN}_{1+d_l,d_l}$, $\ell_1 \in L(\mathbb{R}^d, \mathbb{R}^{d_l})$, $\ell_2 \in L(\mathbb{R}^{d_l}, \mathbb{R}^{d_o})$, let $\phi_{f,\ell_1,\ell_2} : \mathbb{R}^d \to \mathbb{R}^{d_o}$ denote the map $x \mapsto z$ where*

$$y(0) = \ell_1(x), \qquad \frac{dy}{dt}(t) = f(t, y(t)), \qquad \textit{for } t \in [0, T], \qquad z = \ell_2(y(T)).$$

Then the set

$$\{\phi_{f,\ell_1,\ell_2} : f \in \mathcal{NN}_{1+d_l,d_l},\ \ell_1 \in L(\mathbb{R}^d, \mathbb{R}^{d_l}),\ \ell_2 \in L(\mathbb{R}^{d_l}, \mathbb{R}^{d_o})\}$$

is dense in $C^0(\mathbb{R}^d, \mathbb{R}^{d_o})$ for the compact-open topology (see appendix 4 for topological explanations).

The result is equally true if we replace neural networks with all Lipschitz continuous functions since both sets are dense in $C^0(\mathbb{R}^d, \mathbb{R}^{d_o})$. One may view this theorem as the continuous-time analogue of the deep UAT [KL20]. Indeed, this is a universality result for bounded width (equal to input plus output dimensions) and infinite depth. The proof and an explanation of the notation are relegated to the appendix (4) since this is technical. Nevertheless, let us note that the construction is ultimately just integrating a constant neural network f with conveniently chosen dimensions; there are no dynamics, which is disappointing since it defeats the point of adding a time-dimension. A different approach based on differential geometry and control theory [CLT20] yields much deeper insight by capitalising on the time-dimension. A continuous-time analogue of the arbitrary width UAT is the following theorem from [Kid21, Theorem 2.13].

Theorem 3.2. *Fix $d, d_0 \in \mathbb{N}$. With the same notation as in the previous theorem, for each $d_l \in \mathbb{N}$, there exists an $f_{d_l} \in C^0(\mathbb{R}^{d_l}, \mathbb{R}^{d_l})$ for which $\phi_{f_{d_l}, \ell_1, \ell_2}$ is uniquely defined, and for which*

$$\{\phi_{f_{d_l}, \ell_1, \ell_2} : d_l \in \mathbb{N},\ \ell_1 \in L(\mathbb{R}^d, \mathbb{R}^{d_l}),\ \ell_2 \in L(\mathbb{R}^{d_l}, \mathbb{R}^{d_o})\}$$

is dense in $C^0(\mathbb{R}^d, \mathbb{R}^{d_o})$ for the compact-open topology. In other words, the vector field does not need to be a universal approximator if an arbitrarily high hidden dimension is permitted.

Proof. We shall construct f_{d_l} so that the corresponding ODE is solved by a vector of monomials. The linear map ℓ_2 then combines these into a polynomial, at which point we may conclude by the Stone-Weierstrass theorem. More precisely, let $x \in \mathbb{R}^d$, and consider the system of ODEs

$$\begin{aligned}
y_0(0) = x \in \mathbb{R} \quad & \frac{dy_0}{dt}(t) = 0, \\
y_1(0) = 0 \in \mathbb{R}^d \quad & \frac{dy_1}{dt}(t) = y_0(t) \otimes y_0(t), \\
y_2(0) = 0 \in \mathbb{R}^{d \otimes d} \quad & \frac{dy_2}{dt}(t) = y_1(t) \otimes y_0(t), \\
\dots \quad & \\
y_M(0) = 0 \in \mathbb{R}^{d^M} \quad & \frac{dy_M}{dt}(t) = y_{M-1}(t) \otimes y_0(t).
\end{aligned} \tag{3.2}$$

One may verify that the solution is given at time $T = 1$ by

$$(x, x^{\otimes 2}, x^{\otimes 3}, \dots, x^{\otimes (M+1)})^\top. \tag{3.3}$$

Compactness in $C^0(\mathbb{R}^d, \mathbb{R}^{d_o})$ for the compact-open topology is equivalent to compactness in $C^0(K, \mathbb{R}^{d_o})$ for the supremum norm for all compact $K \subset \mathbb{R}^d$ [Mun00, Theorem 46.8]. Therefore, let $K \subset \mathbb{R}^d$ be compact. By the Stone-Weierstrass theorem, the polynomials are dense in $C^0(K, \mathbb{R}^{d_o})$, so for any $F \in C^0(K, \mathbb{R}^{d_o})$ and any $\epsilon > 0$, there exists a polynomial P such that $\|F - P\|_\infty < \epsilon$. Finally, let $d_l = \sum_{k=1}^{M+1} d^k$, let f_{d_l} parametrise the ODE (3.2), embed x into the initial condition with an affine map ℓ_1, and combine the monomials in (3.3) into a polynomial P via the affine map ℓ_2. Then the solution of the ODE realises the polynomial P and hence approximates F. ■

Although neural ODEs are naturally suited to modelling systems which evolve continuously in time, they come with a severe drawback: the solution trajectory of an ODE is fully determined by the initial condition, so unless one allows the vector fields to depend on the data itself, there is no way to react to incoming data as the system evolves. This is an uncomfortable property if we wish to model time-series, and motivates the introduction of a *control* which replaces the ”dt” term and allows us to locally control the evolution of the ODE.

3.2 Controlled Differential Equations

Controlled Differential Equations. Fix a dimension $l \in \mathbb{N}$ and $d \in \mathbb{N}$ vector fields

$$V_i : \Omega \times \mathbb{R}_{\geq 0} \times \mathbb{R}^l \to \mathbb{R}^l, (\theta, t, x) \mapsto V_i^\theta(t, x) \quad 1 \leq i \leq d$$

which are Lipschitz continuous in x and càglàd[3] in t. The variable $\theta \in \Omega$ is to be seen as a trainable parameter. The controls are given by càdlàg functions

$$u_i : \mathbb{R}_{\geq 0} \to \mathbb{R}, \;\; t \mapsto u_i(t)$$

which have bounded variation and start at zero. We now consider the *controlled ordinary differential equation* (CDE)

$$dX_t^\theta = \sum_{i=1}^d V_i^\theta(t, X_{t-}^\theta) du_i(t), \;\; X_0^\theta = x \tag{3.4}$$

where $x \in \mathbb{R}^l$ is the initial state and the subscript minus denotes the left limit of a function. As with any differential equation, what is meant by this is actually a solution to the integral equation

$$X_t^{\theta,x} = x + \sum_{i=1}^d \int_0^t V_i^\theta(s, X_{s-}^{\theta,x}) du_i(t),$$

which exists and is unique when the vector fields are Lipschitz in x [Pro92, Theorem 7, Chapter V]. This equation can now be seen in two ways. If we fix a bounded variation control u and a time $T > 0$, then the CDE induces a solution map

$$\mathbb{R}^l \to \mathbb{R}^l, \;\; x \mapsto X_T^{\theta,x}.$$

However, if we fix $x \in \mathbb{R}^l$, then the CDE forms an *operator* on the space of bounded variation curves starting at zero,

$$BV_c^0([0,T]) \to BV_c^x([0,T]), \;\; u \mapsto X^{\theta,x}.$$

Neural Networks as CDEs. Let us show that the first interpretation allows us to write *any* neural network exactly as a CDE solution map. Fix a neural network

$$A_m \circ \sigma \circ A_{m-1} \circ \ldots \circ \sigma \circ A_0 : \mathbb{R}^l \to \mathbb{R}^l,$$

where we use zero-padding wherever necessary to embed into a sufficiently high dimension $l \in \mathbb{N}$. This is necessary because the CDE evolves in a fixed dimension, so there cannot be a mismatch between the domain and target dimensions. The control is defined to be a stair function with m steps

$$u(t) = \sum_{i=1}^{m+1} \mathbb{1}_{[i,\infty)}(t),$$

and we use a single vector field defined by

$$V^\theta(t,x) = \sum_{i=1}^m \mathbb{1}_{(i-1,i]}(t)(\sigma \circ A_{i-1}(x) - x) \; + \; \mathbb{1}_{(m,\infty)}(t)(A_m(x) - x),$$

where θ parametrises the affine maps. It is a worthwhile exercise to verify (detailed in [HKT20a, Example D.1]) that

$$\begin{aligned} X_0^{\theta,x} &= x \\ X_i^{\theta,x} &= \sigma \circ A_{i-1} \circ \ldots \circ A_0(x), \;\; 1 \leq i \leq m \\ X_{m+1}^{\theta,x} &= A_m \circ \sigma \circ A_{m-1} \circ \ldots \circ \sigma \circ A_0(x), \end{aligned}$$

[3] càglàd: French acronym for left continuous with right limits. càdlàg: right continuous with left limits.

which means that the neural network is the output of the CDE after time $m+1$. This representation of a neural network as a CDE also elucidates the role of the control: it determines the architecture of the network. In this example, the differential of a step function is a dirac measure, so that each jump-discontinuity of the control corresponds to evaluation of a network layer.

Backpropagation in CDEs. Neural networks are generally trained via stochastic gradient descent, which requires the gradient with respect to the parameters $\partial_\theta NN^\theta(x)$. Naively backpropagating through the ODE solver used to calculate the output can incur huge memory costs, as the gradient depends on the entire path. In the CDE setting, there is a particularly elegant alternative. Following [Pro92, Chapter V] and [Tei21], if we assume sufficient smoothness to interchange differential operators, then differentiating the CDE (3.4) with respect to time yields

$$d\partial_\theta X_t^\theta = \sum_{i=1}^d \left(\partial_\theta V_i^\theta(t, X_{t-}^\theta) + dV_i^\theta(t, X_{t-}^\theta)\partial_\theta X_{t-}^\theta\right) du_i(t),$$

which is a linear inhomogeneous first order ODE in $\partial_\theta X_t^\theta$. If we denote by $J_{s,t}$ the evolution operator of the associated *homogeneous* equation

$$d\partial_\theta X_t^\theta = \sum_{i=1}^d dV_i^\theta(t, X_{t-}^\theta)\partial_\theta X_{t-}^\theta du_i(t),$$

then by variation of constants (ultimately simply Fubini's theorem, detailed in [Tei21, lecture 2]),

$$\partial_\theta X_T^\theta = \sum_{i=1}^d \int_0^T J_{s+,T}\partial_\theta V_i^\theta(s, X_{s-}^\theta)du_i(s). \tag{3.5}$$

This formula asserts that the derivative of X_T^θ with respect to θ is obtained by integrating the derivative of the vector fields - propagated forwards to the time T - over the solution trajectory of X^θ. Alternatively, one may interpret this as the average sensitivity of the vector fields over the solution trajectory. In the neural ODE literature, this is known as adjoint backpropagation (with slightly different notation) and is a memory-efficient way to calculate gradients since it does not scale with the time-horizon T. Indeed, equation (3.5) can be integrated with a single call to an ODE solver which simultaneously evolves the CDE (3.4), $J_{s+,T}$, and (3.5) backwards from T to zero. The memory cost of such an ODE solver is independent of T, which makes this a viable algorithm for very long time-series [Kid+20]. Even despite this speedup, training neural ODEs is prohibitively slow. One recent insight which has led to a significantly faster *second-order* method is to view the problem of finding an optimal θ as an optimal control problem. This allows one to derive a set of coupled ODEs which describe both $\partial_\theta X_T^\theta$ and $\partial_\theta^2 X_T^\theta$ and can be solved with a single ODE solver call completely analogous to adjoint backpropagation [LCT21].

Comments.

- This section presents the CDE framework using differential notation from stochastic calculus despite the deterministic setup. The reason for this is that stochastic integration coincides with Lebesgue-Stieltjes integration when the integrator is of bounded variation. Furthermore, stochastic integration extends beyond bounded variation controls (which are the most general integrators for the Lebesgue-Stieltjes integral) to semimartingales which are of fundamental interest to mathematical finance. This stochastic CDE setting is treated for example in [Pro92, Chapter V, sections 7-10] and has connections to signature methods.

- In the context of CDEs, the control u enters linearly. One may suspect that a non-linear function of the control (and integrating with respect to some measure dominating du) would yield greater modelling power. However, for continuous controls $u \in BV_c([0,T])$, this is actually not the case, and actually deteriorates the quality! With simplified notation, every CDE of the form

$$dX_t = V(X_{t-}, u_{t-})dt$$

can be realised by a CDE of the form

$$d\tilde{X}_t = \tilde{V}(\tilde{X}_{t-})du(t)$$

if the control is a continuous curve. The reason for this is that one can augment the vector field to absorb the control into the state $\tilde{X}_t$ at which point in can be processed non-linearly. For instance, u can be absorbed by writing

$$d\begin{pmatrix} X_t \\ u_t \end{pmatrix} = \begin{pmatrix} V(X_{t-}, u_{t-}) \\ id \end{pmatrix} du(t).$$

The reverse is not possible; a non-linear function of the control cannot always mimic a linear control. This is proved in detail in [Kid21, Theorem C.27] but it is overlooked that this depends critically on the continuity of the control: in the previous equations, the continuity of u makes the left limits redundant. If the control is discontinuous, then the left limit prevents non-linear processing at each jump. Intuitively, this discrepancy boils down to the fact that if du has no atoms, then it has no instantaneous effect on the CDE, while an atom does. Consequently, [Kid21, Theorem C.27] does *not* extend past continuous controls, and if jump controls are desired, one must rethink how to obtain non-linear processing. One possibility is to allow the vector fields to depend on the time-series. This is essentially what is done in [RCD19] and [HKT20b].

- Neither smooth nor pure jump controls capture the reality of sampling a continuously evolving system at discrete points: a pure jump control yields a piecewise constant time-series, while the classical neural ODE is fully determined by the initial state. A hybrid approach which combines both of these approaches is the residual ODE-RNN, an adaptation of the ODE-RNN [RCD19]. The idea is to evolve continuously in time according to a neural ODE but to add a jump discontinuity whenever new data is observed. This amounts to the CDE

  $$dY(t) = f^\theta(Y_{t-}, t-)dt + rRNNCell(Y_{t-}, x_t)du(t)$$

  where $rRNNCell$ is a residual RNN function as described in section 3.1, and u is a pure jump process which jumps at every observation time. This CDE can be written equivalently as

  $$\begin{cases} Y_{t_{i+1}-} = ODEsolve(f^\theta, Y_{t_i}, (t_i, t_{i+1})) \\ Y_{t_{i+1}} = Y_{t_{i+1}-} + rRNNCell(Y_{t_{i+1}-}, x_i), \end{cases} \tag{3.6}$$

  which is proved in [HKT20b, Prop. C.1]. Note that the non-linear dependence on the time-series x_t is exactly the previous point. The benefit of this formulation is that it can handle irregular data with ease; the system continues evolving according to the neural ODE "dt" term when there in no data. Furthermore, predictions can be read out at any time between observations. See [HKT20b] for an adaptation of this approach to predicting conditional expectations from a discretely sampled stochastic process.

3.3 Signature and Universal Approximation for Neural CDEs

The *signature*, a collection of all iterated integrals of a path, plays a central role in the study of path-dependent functionals. In fact, we have already encountered the signature without comment: equation (3.2) is nothing more than a truncated signature - an observation which we will illustrate later. Neglecting a plethora of spectacular algebraic and analytic properties, this repeated appearance is ultimately due to its universality - any continuous path-dependent functional can be approximated by a linear combination of signature components. In this sense, the signature acts both as a *universal non-linearity* which linearises path-dependence, and as a *feature extractor* which yields a graded summary of the essence of a path. These aspects have attracted considerable attention from the machine-learning community, especially in finance where path-dependence continues to pose serious challenges.

The purpose of this section is to introduce the signature as a tool for obtaining universality results. Along the way, we will highlight its shortcomings, whose resolution has led to promising developments, such as truncated, randomised [Cuc+21b], or discrete-time signature [Cuc+21a], and recent kernel methods [GGO22]. We will restrict our attention to bounded variation controls as before, but it would be remiss not to mention that the signature is the starting point of rough path theory - in particular, it is involved in the construction of a coherent integration theory for low regularity paths described by $\frac{1}{p}$-Hölder-contintuity for $p \geq 2$. Importantly, this regime encompasses almost all of stochastic analysis. A more comprehensive introduction to the signature from the perspective of rough path theory is given in [LCL07], and a shorter, less technical introduction focused on machine learning is found in [CK16].

Since the signature is an at first sight arcane object, let us first introduce the tensor algebras where signature takes values. Let V be a finite-dimensional vector space with basis $e_1, \ldots, e_d$. The generalised tensor algebra over V is defined by

$$T((V)) = \bigoplus_{n=0}^{\infty} V^{\otimes n} = \{v = (v_0, v_1, v_2, \ldots) = \sum_{n=0}^{\infty} v_n : v_n \in V^{\otimes n}\},$$

admits element-wise addition, and multiplication as $(vw)_n = \sum_{j=0}^{n} v_j w_{n-j}$. Tensor multiplication $(\otimes)$ is suppressed and is instead left implicit. The truncated tensor algebra is given (suppressing an isomorphism) by all tensors of bounded rank

$$T^N(V) = \bigoplus_{n=0}^{N} V^{\otimes n} = \{v = (v_0, \ldots, v_N) \in T((V))\},$$

and any tensor powers higher than rank N are zeroed whenever encountered, i.e. $v_N v_0 = 0$ in $T^N(V)$.

Signature. Let $I = [s, t]$ be a compact interval and let $X : I \to \mathbb{R}^d$ be a bounded variation curve. For a multi-index $\mathbf{i} = (i_1, \ldots, i_n)$ with components in $\{1, \ldots, d\}$, the signature coordinate associated to $\mathbf{i}$ is defined as

$$S^{\mathbf{i}}(X) e_{i_1} \ldots e_{i_n} = \int_{s \leq t_1 \leq \cdots \leq t_n \leq t} dX_{t_1}^{i_1} \ldots dX_{t_n}^{i_n} e_{i_1} \ldots e_{i_n}.$$

The *signature* of X is the collection of all iterated integrals, written compactly as

$$S(X) = (1, S(X)^{(1)}, S(X)^{(2)}, \ldots) \in T((V)), \tag{3.7}$$

where $S(X)^{(k)} = \sum_{\mathbf{i}=(i_1,\ldots,i_k)} S^{\mathbf{i}}(X)$. Analogously, the truncated signature of degree N is given by

$$S_N(X) = (1, S(X)^{(1)}, S(X)^{(2)}, \ldots, S(X)^{(N)}).$$

Signatures as Monomials. To see how the signature of a path has anything to do with solutions of CDEs as in section 3.2, let us immediately work towards a Taylor-like approximation formula. Consider the simplest possible CDE

$$dX_t = dt, \quad X_0 = x \in \mathbb{R},$$

with its evolution operator given by $Evol_{s,t}(x) = x + (t - s)$. For a smooth function $f : \mathbb{R} \to \mathbb{R}$, we may invoke the fundamental theorem of calculus twice to write

$$\begin{aligned} f(Evol_{s,t}(x)) = f(x + (t - s)) &= f(x) + \int_s^t f'(x + t_2 - s) dt_2 \\ &= f(x) + \int_s^t \left(f'(x) + \int_s^{t_2} f''(x + t_1 - s) dt_1 \right) dt_2 \\ &= f(x) + (t - s) f'(x) + \int_{s \leq t_1 \leq t_2 \leq t} f''(x + t_1 - s) dt_1 dt_2. \end{aligned}$$

Iterating this substitution and integrating out the derivatives $f^{(k)}(x)$, we obtain Taylor's formula

$$f(x + (t - s)) = \sum_{k=0}^{M} f^{(k)}(x) \cdot \frac{(t - s)^k}{k!} + R_M(t, s, f).$$

Incidentally, besides providing undoubtedly the most elegant proof of Taylor's theorem, this approach reveals the often overlooked fact that the monomials $\frac{(t-s)^k}{k!}$ appearing in the Taylor expansion are actually iterated integrals of the (trivial) control $u(t) = t$! The signature components $S^{(k)}$ will play an analogous role to that of the monomials. Indeed, we may apply the exact same approach to the CDE

$$dX_t = \sum_{i=1}^{d} V_i(X_t)du_i(t), \quad X_0 = x$$

to obtain the following theorem from [Cuc+21b, Theorem 2.1]. For notational brevity, we define the transport operator $V : C^\infty(\mathbb{R}^d) \to C^\infty(\mathbb{R}^d)$ associated to a (synonymous) smooth vector field $V : \mathbb{R}^d \to \mathbb{R}^d$ by $Vf(x) = df(x)V(x)$.

Theorem 3.3 (Taylor Expansion for CDEs)**.** *Let $V_i : \mathbb{R}^d \to \mathbb{R}^d$ be smooth vector fields and let $Evol$ be a smooth evolution operator satisfying*

$$d_t Evol_{s,t}(x) = \sum_{i=1}^{d} V_i(Evol_{s,t}(x))du^i(t).$$

Then for any smooth function $f : \mathbb{R}^d \to \mathbb{R}$ and any $s \leq t$ and $M \in \mathbb{N}$,

$$f(Evol_{s,t}(x)) = \sum_{k=0}^{M} \sum_{i_1,\dots,i_k=1}^{d} V_{i_1} \cdots V_{i_k} f(x) \int_{s \leq t_1 \leq \cdots \leq t_k \leq t} du^{i_1}(t_1) \cdots du^{i_k}(t_k) + R_M(t,s,f),$$

with remainder term

$$R_M(t,s,f) = \sum_{i_0,\dots,i_M=1}^{d} \int_{s \leq t_0 \leq \cdots \leq t_{M+1} \leq t} V_{i_0} \cdots V_{i_M} f(Evol_{s,t_0}(x))du^{i_0}(t_0) \cdots du^{i_M}(t_M).$$

In the case of Taylor's theorem, the remainder term decays factorially given a bound on higher order derivatives. For the previous theorem, it is not a priori clear whether this behaviour also applies to iterated integrals. The following theorem from [LCL07, Prop. 2.2] confirms that this is indeed the case, reinforcing the intuition that iterated integrals can be thought of as a generalisation of monomials.

Theorem 3.4 (Signature terms decay factorially)**.** *Let $X : [0,T] \to \mathbb{R}^d$ be a path of bounded variation. Then for each $k \geq 1$, one has*

$$\left\| \int_{0<t_1<\cdots<t_k<T} dX_{t_1} \cdots dX_{t_k} \right\| \leq \frac{\|X\|_1^k}{k!},$$

where $\|X\|_1$ denotes the total variation norm of X.

Proof. Let us denote the total variation of X up to time t by $\phi(t) = \|X\|_{1,[0,t]}$. Since ϕ is non-decreasing, it is invertible (taking the left-limit whenever ϕ is not injective), and by reparametrising X with ϕ, we may normalise its total variation to

$$\|X \circ \phi^{-1}\|_{1,[0,t]} = t.$$

Since path-integrals are invariant under reparametrisations, we may suppose without loss of generality that $\|X\|_{1,[0,t]} = t$ for all $t \leq T$. By the smoothness (in time) of the total variation of X, it follows that X is Lipschitz-continuous, and by Rademacher's theorem that it is almost everywhere differentiable with $|\dot{X}_t| = 1$. Hence

$$\begin{aligned} \left\| \int_{0<t_1<\cdots<t_k<T} dX_{t_1} \cdots dX_{t_k} \right\| &= \left\| \int_{0<t_1<\cdots<t_k<T} \dot{X}_{t_1} \cdots \dot{X}_{t_k} dt_1 \dots dt_k \right\| \\ &\leq \int_{0<t_1<\cdots<t_k<T} dt_1 \dots dt_k \\ &= \frac{T^k}{k!}. \end{aligned}$$

Since $T = \|X\|_{1,[0,T]}$, this proves the result. ∎

This theorem guarantees that at least in principle, the signature acts as an efficient feature extractor $S : BV([0,T], \mathbb{R}^d) \to T((\mathbb{R}^d))$ with higher order terms containing less and less information. However, there are d^N iterated integrals of order N for an $\mathbb{R}^d$-valued path, which limits the truncation level to at most around $N = 20$ in practice. An approach to circumvent this detrimental scaling is described in [Cuc+21b] on randomised signature.

Finally, let us link the truncated signature to the proof of theorem 3.2 on the universality of neural ODEs. It is straightforward to verify that the truncated signature of order N is in fact the solution of the CDE

$$dY_t = Y_t dX_t, \quad Y_0 = (1, 0, \ldots, 0) \in T^N(\mathbb{R}^d).$$

Indeed, writing out the equation as a vector with each component corresponding to the tensor rank, we obtain the system

$$d \begin{pmatrix} Y_t^1 \\ Y_t^2 \\ Y_t^3 \\ \vdots \\ Y_t^N \end{pmatrix} = \begin{pmatrix} 0 \\ Y_t^1 dX_t \\ Y_t^2 dX_t \\ \vdots \\ Y_t^{N-1} dX_t \end{pmatrix}$$

and these equations are satisfied by the truncated signature components. Appropriately interpreted, this CDE holds for full signature too, which is why signature if often considered a non-commutative exponential of a path. Returning to the proof of theorem 3.2, it is clear that the ODE

$$\begin{aligned} y_0(0) = x \in \mathbb{R} \quad & \frac{dy_0}{dt}(t) = 0, \\ y_1(0) = 0 \in \mathbb{R}^d \quad & \frac{dy_1}{dt}(t) = y_0(t) \otimes y_0(t), \\ y_2(0) = 0 \in \mathbb{R}^{d \otimes d} \quad & \frac{dy_2}{dt}(t) = y_1(t) \otimes y_0(t), \\ & \cdots \\ y_M(0) = 0 \in \mathbb{R}^{d^M} \quad & \frac{dy_M}{dt}(t) = y_{M-1}(t) \otimes y_0(t), \end{aligned} \tag{3.8}$$

is nothing more than the truncated signature of the linear control $X_t = xt$ for some $x \in \mathbb{R}^d$. This system is solved by the monomials, which should not be surprising in light of the previous discussion on monomials as signature terms of a linear control. Therefore we see how the section on neural ODEs is subsumed by the study of CDEs and signature.

Signature as a Universal Non-linearity. We have already mentioned that path-integrals are invariant under reparametrisations, meaning that the signature cannot discern the speed at which a path is traversed. Furthermore, backtracking - concatenating a path with itself run backwards - undoes signature since it is a path-integral [LCL07, Prop. 2.14]. Therefore any excursions which are immediately traversed backwards cannot be detected by signature either. However, it turns out that these are the only cases where the signature fails to be injective. Formally, the signature of a path describes it up to *tree-like* equivalence, that is, up to repeated (potentially infinitesimal) excursions which are backtracked [LCL07, Theorem 2.29]. If one augments a path $t \mapsto X_t$ with a time-coordinate to form $t \mapsto \bar{X}_t := (t, X_t)$, this equivalence reduces to a singleton; signature becomes injective.

Theorem 3.5 (Injectivity of Time-augmented Signature [Tei21])**.** *Time-augmented signature* $\tilde{S} : BV_0([0,T], \mathbb{R}^d) \to T((\mathbb{R}^d)), \quad X \mapsto S(\bar{X})_T$ *is injective on the set of bounded variation curves starting at zero.*

Proof. The signature component corresponding to k-fold integration of the time-coordinate and one integral of the i-th coordinate of the path is given by

$$\int_{0<t_1<\cdots<t_k<u<T} dX_{t_0}^0 \ldots dX_{t_k}^0 dX_u^i = \int_{0<u<T} \frac{1}{k!} u^k dX_u^i.$$

Since polynomials are dense in $C^0([0,T],\mathbb{R})$, these integrals characterise the Fourier transform of X^i (by having the polynomials approximate *sin* and *cos*) and therefore X^i as well.

■

Finally, let us state the core theorem of this section - universality of signature [KO19, Theorem 1].

Theorem 3.6 (Universality of Signature)**.** *Let $K \subset BV_0([0,T],\mathbb{R}^d)$ be a compact set of bounded variation curves starting at zero. Then the linear functionals of augmented signature,*

$$\{\langle l, \tilde{S}\rangle : l \in T(V)\}$$

form a dense subset of $C^0(K,\mathbb{R})$. In other words, for every $f \in C^0(K,\mathbb{R})$ and for every $\epsilon > 0$, there exists $l \in T(V)$, the dual space of $T((V))$, such that

$$\sup_{h\in K} |f(h) - \langle l, \tilde{S}(h)\rangle| < \epsilon.$$

Proof. As usual, we wish to apply Stone-Weierstrass to the set of linear functionals of signature. This set is indeed an algebra - products of signature components are linear combinations of signature components. This is essentially just the Leibniz rule for integration, and is elegantly expressed using the *shuffle-product*. We refer to [LCL07] for a treatment of this, since it is not essential to this discussion. Furthermore, this algebra is point-separating by injectivity of augmented signature. Since any point-separating subalgebra is dense by Stone-Weierstrass, the result follows. ■

For specific applications, one must establish the compactness of some relevant subset of $BV_0([0,T],\mathbb{R}^d)$, which is non-trivial since balls are never compact in infinite dimensions. See the discussion surrounding [KNT22, Prop. 3.10] for compactness criteria and references. The practical relevance of this theorem is that any continuous path-dependent functional can be arbitrarily well approximated by a linear combination of signature components, yielding a tremendous compression of complexity. In essence, one may capture a complicated path-dependent functional by a single string of coefficients representing the weights - an incredibly parsimonious representation which may be used to accelerate classically numerically demanding tasks like path-dependent option pricing.

3.4 RNNs, RKHS, and Signature

Kernel Learning. The preceding discussion on the signature as a universal non-linearity is strongly reminiscent of *kernel learning*. The general paradigm of kernel learning is to consider a set of real-valued functionals $\mathcal{F} \subseteq C^0(\mathcal{X},\mathbb{R})$ on a topological space $\mathcal{X}$, and the objective is to find a universal feature map $\phi : \mathcal{X} \to H$ into some Hilbert space H such that the set $\{l \circ \phi : l \in H^*\}$ is dense in $\mathcal{F}$. In other words, the feature map ϕ absorbs the non-linearity of $\mathcal{F}$, thereby reducing the task of approximating a functional to a linear regression. A graphical representation of this is given below.

$$\begin{tikzcd}
\mathcal{X} \arrow[r, "\phi"] \arrow[dr, "f\in\mathcal{F}"'] & H \arrow[d, "l\in H^*"] \\
& \mathbb{R}
\end{tikzcd}$$

As an example, consider perhaps the simplest choice $\mathcal{X} = [0,1]^d$ and $F = C^0(\mathcal{X},\mathbb{R})$. By the Stone-Weierstrass theorem, the polynomials are a dense subset of $\mathcal{F}$, and we may split the polynomial map in two: a monomial lift followed by a linear map combining these into polynomials. This splitting corresponds to the choices[4] $H = T((\mathbb{R}^d))$ and $\phi(x) = (1, x, x^{\otimes 2}, ...)$. The linear map $l \in H^*$ then encodes the coefficients of the polynomial

$$l \circ \phi(x) = \sum_{n=0}^{M} \alpha_n x^{\otimes n},$$

and the task of approximating a function $f \in C^0([0,1]^d, \mathbb{R})$ reduces to classical polynomial regression (linear regression if one views the monomials as features).

In the case of signature, the domain $\mathcal{X}$ is the path-space $BV_0([0,T], \mathbb{R}^d)$, and unsurprisingly, the monomial lift is replaced by the augmented signature $X \mapsto \phi(X) = \tilde{S}(X)$. However, keeping in mind the factorial - and therefore square-summable - decay of signature for bounded variation curves (theorem 3.4), we see that the image of signature is actually a subset of the Hilbert space $H := \{a \in T((\mathbb{R}^d)) : \|a\|_H := \sum_{n\in\mathbb{N}} \|a^n\|^2_{(\mathbb{R}^d)^{\otimes n}} < \infty\}$ with its natural inner product given by $\langle a, b\rangle_H = \sum_{n\in\mathbb{N}} \langle a, b\rangle_{(\mathbb{R}^d)^{\otimes n}}$. Therefore we may view theorem 3.6 on the universality of signature as a manifestation of a polynomial kernel learning problem which we illustrate below.

$$\begin{tikzcd}
BV_0([0,T], \mathbb{R}^d) \arrow[r, "\tilde{S}"] \arrow[dr, "f"'] & H \subset T((\mathbb{R}^d)) \arrow[d, "{\langle l, \cdot \rangle}"] \\
 & \mathbb{R}
\end{tikzcd}$$

In general, one may define a kernel corresponding to the feature map ϕ by

$$\begin{aligned} K : \mathcal{X} \times \mathcal{X} &\to \mathbb{R} \\ (x, y) &\mapsto \langle \phi(x), \phi(y)\rangle, \end{aligned}$$

and also an associated reproducing kernel Hilbert space (RKHS) $H_\phi = \overline{span}\{K_x = \langle \phi(\cdot), \phi(x)\rangle : x \in \mathcal{X}\}$) which is dense in $\mathcal{F}$. In the case of signature, this means that that the RKHS $H_{\tilde{S}}$ induced by the signature kernel

$$\begin{aligned} K : BV_0([0,T], \mathbb{R}^d) \times BV_0([0,T], \mathbb{R}^d) &\to \mathbb{R} \\ (X, Y) &\mapsto \langle \tilde{S}(X), \tilde{S}(Y)\rangle \end{aligned}$$

is dense in $C^0(K, \mathbb{R})$ for any compact $K \subset BV_0([0,T], \mathbb{R}^d)$.

RNNs as Signature Functionals. As the grand finale of this chapter, let us leverage CDEs, signature, and kernel methods to write a recurrent neural network as a linear functional of signature. This part closely follows [Fer+21]. The construction consists of three steps:

1. First, a residual RNN, written suggestively in the form

   $$h_{n+1} = h_n + \frac{1}{T} f(h_n, x_{n+1}),$$

   is considered as an Euler discretisation of the neural CDE

   $$dH_t = f(H_t, X_t)dt, \ H_0 = h_0$$

   where X is a continuous interpolation of the time-series $x_1, \ldots, x_T \in \mathbb{R}^m$. Under mild conditions, this induces an error of the form $\mathcal{O}(\frac{1}{T})$, that is,

   $$\|H_{\frac{j}{T}} - h_j\| \leq \frac{c}{T}$$

   for some constant $c > 0$ [Fer+21, Prop.1]. We are interested in approximating $\phi \circ h_T$ for a fixed linear function $\phi : \mathbb{R}^m \to \mathbb{R}$. This setting encompasses classification, for example.

[4] Technically, here we consider a locally convex space and its dual, not a Hilbert space.

2. Next, we leverage Kidger's point that any "dt" control non-linear in X_t can be written as a linear "dX_t" control (see the second comment in section 3.2) to replace the CDE above by

$$d\bar{H}_t = F(\bar{H}_t)d\bar{X}_t = \sum_{i=1}^{d} F^i(H_t)dX_t^i$$

where the bar superscipt denotes time-augmentation. This procedure incurs a dimension increase from m to $m+d$ and is admittedly awkward because any *adapted* continuous interpolation of a time-series 'lags behind' it, and as discussed in the aforementioned comment, this construction fails if we do not enforce continuity.

3. Finally, we use the Taylor expansion for CDEs (3.3) to write the CDE operator as a linear functional of signature. Because of the factorial decay of signature, this linear functional can be written as an element of the reproducing kernel Hilbert space (RKHS) spanned by the signature kernel, and can be identified with the element $\xi_\alpha \in H_{\tilde{S}}$ with coefficients given by

$$\alpha_k^{(i_1,\dots,i_k)} = \phi \circ Proj(F^{i_i} \cdots F^{i_k}(\bar{H}_0))$$

in accordance with the formula in theorem 3.3. The projection $Proj$ selects the first m components to remove the state-augmentation. The RKHS element acts on data paths via $\xi_\alpha(X) = \langle \alpha, S(\bar{X})\rangle$.

This derivation is subject to highly technical constraints which guarantee that the signature coefficients in the Taylor CDE expansion have square-summable decay and therefore lie in the RKHS. The interested reader is referred to [Fer+21] for the full proofs of these statements. Apart from unifying RNNs, CDEs, signature methods, and kernel learning, this approach also has the added benefit of revealing training and regularisation techniques for RNNs. Instead of training via stochastic gradient descent on a given data set of n points $(x^{(i)}, y^{(i)})$ with some loss function l, one could instead consider the RKHS element minimising the empirical risk

$$\hat{\mathcal{R}} = \frac{1}{n}\sum_i l\left(y^{(i)}, \langle \alpha, S(\bar{X}^{(i)})\rangle\right).$$

By the representer theorem, the optimiser lies in the span of the kernels centred at the data, which in this case correspond to signatures of the interpolated time-series. This yields an *iteration-free* training method, has convergence guarantees, and can easily incorporate penalisation of slow signature coefficient decay to improve the Lipchitz continuity and adversarial robustness of the RNN. However, Fermanian et al. point out that calculating such a penalty is non-trivial [Fer+21, Section 3.2]. An interesting direction for further research would be to consider the possibility of using randomised signature to enable the implementation of these RKHS ideas in practice.

4 Appendix

Theorem 4.1 ($\mathbb{R}^d$-valued Stone-Weierstrass)**.** *Let K be a Hausdorff topological space, and let $\mathcal{A} \subseteq C^0(K, \mathbb{R}^d)$ be a subalgebra, where $\mathbb{R}^d$ is equipped with the standard topology. If for $i \in \{1, \ldots, d\}$, the componentwise subalgebras $\mathcal{A}_i = \{\pi_i \circ f : f \in \mathcal{A}\} \subseteq C^0(K, \mathbb{R})$ are each point-separating, and $\mathcal{A}$ contains the componentwise constants $c_i : K \to \mathbb{R}^d \;\; x \mapsto e_i$, then $\mathcal{A}$ is dense in $C^0(K, \mathbb{R}^d)$.*

Proof. Notice that the topological equivalence of norms in finite dimensions implies that $C^0(K, (\mathbb{R}^d, \|\cdot\|)) = C^0(K, (\mathbb{R}^d, \|\cdot\|_\infty))$ for any norm $\|\cdot\|$ on $\mathbb{R}^d$. Hence it suffices to show the density of $\mathcal{A}$ for the supremum norm. Since each subalgebra is point-separating, it follows from the Stone-Weierstrass theorem that each $\mathcal{A}_i$ is dense in $C^0(K, \mathbb{R})$. More precisely, for every $f \in C^0(K, \mathbb{R}^d)$, there exist $g_i \in \mathcal{A}$ such that $\sup_{x \in K} |f^i(x) - g_i^i(x)| < \epsilon$ for $i \in \{1, \ldots, d\}$. As the componentwise constants c_i lie in the algebra, we have that $g := c_1 \cdot g_1 + \ldots + c_d \cdot g_d \in \mathcal{A}$. By construction, $g^i = g_i^i$, which immediately implies that

$$\sup_{x \in K} \|f(x) - g(x)\|_\infty = \sup_{x \in K} \max_{1 \le i \le d} |f^i(x) - g_i^i(x)| < \epsilon.$$

■

Theorem 3.1. *Fix $d, d_l, d_o \in \mathbb{N}$ with $d_l = d + d_o$. For $f \in \mathcal{NN}_{1+d_l, d_l}$, $\ell_1 \in L(\mathbb{R}^d, \mathbb{R}^{d_l})$, $\ell_2 \in L(\mathbb{R}^{d_l}, \mathbb{R}^{d_o})$, let $\phi_{f,\ell_1,\ell_2} : \mathbb{R}^d \to \mathbb{R}^{d_o}$ denote the map $x \mapsto z$ where*

$$y(0) = \ell_1(x), \qquad \frac{dy}{dt}(t) = f(t, y(t)), \qquad \text{for } t \in [0, T], \qquad z = \ell_2(y(T)).$$

Then the set

$$\{\phi_{f,\ell_1,\ell_2} : f \in \mathcal{NN}_{1+d_l, d_l},\ \ell_1 \in L(\mathbb{R}^d, \mathbb{R}^{d_l}),\ \ell_2 \in L(\mathbb{R}^{d_l}, \mathbb{R}^{d_o})\}$$

is dense in $C^0(\mathbb{R}^d, \mathbb{R}^{d_o})$ for the compact-open topology.

Proof. Let us begin by noting that neural networks are a posteriori Lipschitz maps, so that the ODE has a unique solution by the Picard-Lindelöf theorem. Since we do not assume familiarity with the compact-open topology, let us simply note that in this context, density in the compact-open topology corresponds to uniform density in $C^0(K, \mathbb{R}^{d_o})$ for all compact $K \subset \mathbb{R}^d$ [Mun00, Theorem 46.8]. Hence this theorem follows from proving the statement for compact sets and is merely a convenient way of stating it without specifying a compact domain.

Correspondingly, let $K \subset \mathbb{R}^d$ be compact and fix any function $H : K \to \mathbb{R}^{d_o}$ and $\epsilon > 0$. By the universal approximation theorem 2.1, there exists a shallow neural network $G \in \mathcal{NN}_{d,d_o}$ with $\|H - G\|_\infty < \epsilon$. We now augment this neural network to one in $\mathcal{NN}_{1+d_l,d_l}$, where $d_l = d + d_o$. First, partition the input vector $x = [x^{(0)}, x^{(d)}, x^{(d_o)}] \in \mathbb{R}^{1+d_l}$ and let $f(x) = [0^{(d)}, G(x^{(d)})]$. This can be realised by carefully zero-padding the two affine maps of G. Now, by letting $\ell_1(x) = [x, 0^{(d_o)}]$ and $\ell_2([y^{(d)}, y^{(d_o)}]) = y^{(d_o)}$, the ODE

$$y(0) = \ell_1(x), \qquad \frac{dy}{dt}(t) = f(t, y(t)), \qquad \text{for } t \in [0, T], \qquad z = \ell_2(y(T))$$

is solved at time $T = 1$ by

$$y(T) = y(0) + \int_0^1 f(t, y(t))dt = [x^{(d), 0^{(d_o)}}] + [0^{(d)}, G(x^{(d)})] \quad z = \ell_2(y(T)) = G(x^{(d)}).$$

Therefore the ODE flow realises the network G and approximates H up to accuracy ϵ. Inspecting the construction of f, we see that this is nothing more than gluing the input and output dimensions and integrating a constant for one unit of time.

■

References

[BC85] Stephen Boyd and Leon Chua. "Fading memory and the problem of approximating nonlinear operators with Volterra series". In: *IEEE Transactions on circuits and systems* 32.11 (1985), pp. 1150–1161.

[Cyb89] George Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[Pro92] Philip Protter. *Stochastic Integration and Differential Equation.* Second. Berlin, Heidelberg: Springer-Verlag, 1992.

[CP98] Ngai Hang Chan and Wilfredo Palma. "State space modeling of long-memory processes". In: *The Annals of Statistics* 26.2 (1998), pp. 719–740.

[Mun00] James Munkres. *Topology.* Prentice Hall, Inc., 2000.

[BNL04] Nils Bertschinger, Thomas Natschläger, and Robert Legenstein. "At the edge of chaos: Real-time computations and self-organized criticality in recurrent neural networks". In: *Advances in neural information processing systems* 17 (2004).

[SZ06] Anton Maximilian Schäfer and Hans Georg Zimmermann. "Recurrent neural networks are universal approximators". In: *Artificial Neural Networks–ICANN 2006: 16th International Conference, Athens, Greece, September 10-14, 2006. Proceedings, Part I 16.* Springer. 2006, pp. 632–640.

[LCL07] Terry J Lyons, Michael Caruana, and Thierry Lévy. *Differential equations driven by rough paths.* Springer, 2007.

[BFG16] Christian Bayer, Peter Friz, and Jim Gatheral. "Pricing under rough volatility". In: *Quantitative Finance* 16.6 (2016), pp. 887–904.

[CK16] Ilya Chevyrev and Andrey Kormilitzin. "A primer on the signature method in machine learning". In: *arXiv preprint arXiv:1603.03788* (2016).

[Che+18] Ricky TQ Chen et al. "Neural ordinary differential equations". In: *Advances in neural information processing systems* 31 (2018).

[GO18a] Lyudmila Grigoryeva and Juan-Pablo Ortega. "Echo state networks are universal". In: *Neural Networks* 108 (2018), pp. 495–508.

[GO18b] Lyudmila Grigoryeva and Juan-Pablo Ortega. "Universal discrete-time reservoir computers with stochastic inputs and linear readouts using non-homogeneous state-affine systems". In: *Journal of Machine Learning Research* 19 (2018), pp. 1–40.

[SCC18] Uri Shaham, Alexander Cloninger, and Ronald R Coifman. "Provable approximation properties for deep neural networks". In: *Applied and Computational Harmonic Analysis* 44.3 (2018), pp. 537–557.

[YFL18] Boxuan Yue, Junwei Fu, and Jun Liang. "Residual recurrent neural networks for learning sequential representations". In: *Information* 9.3 (2018), p. 56.

[DDT19] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. "Augmented Neural ODEs". In: *Advances in Neural Information Processing Systems.* Vol. 32. 2019.

[KO19] Franz J Király and Harald Oberhauser. "Kernels for sequentially ordered data". In: *Journal of Machine Learning Research* 20 (2019).

[RCD19] Yulia Rubanova, Ricky TQ Chen, and David Duvenaud. "Latent odes for irregularly-sampled time series (2019)". In: *arXiv preprint arXiv:1907.03907* (2019).

[CLT20] Christa Cuchiero, Martin Larsson, and Josef Teichmann. "Deep neural networks, generic universal interpolation, and controlled ODEs". In: *SIAM Journal on Mathematics of Data Science* 2.3 (2020), pp. 901–919.

[HKT20a] Calypso Herrera, Florian Krach, and Josef Teichmann. "Estimating full lipschitz constants of deep neural networks". In: *arXiv preprint arXiv:2004.13135* (2020).

[HKT20b] Calypso Herrera, Florian Krach, and Josef Teichmann. "Neural jump ordinary differential equations: Consistent continuous-time prediction and filtering". In: *arXiv preprint arXiv:2006.04727* (2020).

[KL20] Patrick Kidger and Terry Lyons. "Universal approximation with deep narrow networks". In: *Conference on learning theory*. PMLR. 2020, pp. 2306–2327.

[Kid+20] Patrick Kidger et al. "Neural Controlled Differential Equations for Irregular Time Series". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. 2020, pp. 6696–6707.

[Zha+20] Han Zhang et al. "Approximation capabilities of neural ODEs and invertible residual networks". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 11086–11095.

[Cuc+21a] Christa Cuchiero et al. "Discrete-time signatures and randomness in reservoir computing". In: *IEEE Transactions on Neural Networks and Learning Systems* 33.11 (2021), pp. 6321–6330.

[Cuc+21b] Christa Cuchiero et al. "Expressive power of randomized signature". In: *The Symbiosis of Deep Learning and Differential Equations*. 2021.

[Fer+21] Adeline Fermanian et al. "Framing RNN as a kernel method: A neural ODE approach". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 3121–3134.

[GO21] Lukas Gonon and Juan-Pablo Ortega. "Fading memory echo state networks are universal". In: *Neural Networks* 138 (2021), pp. 10–13.

[Kid21] Patrick Kidger. "On neural differential equations". PhD thesis. University of Oxford, 2021.

[Lim+21] Bryan Lim et al. "Temporal fusion transformers for interpretable multi-horizon time series forecasting". In: *International Journal of Forecasting* 37.4 (2021), pp. 1748–1764.

[LCT21] Guan-Horng Liu, Tianrong Chen, and Evangelos Theodorou. "Second-Order Neural ODE Optimizer". In: *Advances in Neural Information Processing Systems*. Vol. 34. 2021, pp. 25267–25279.

[Tei21] Josef Teichmann. *Universal approximation - a dynamic point of view (machine learning in finance, lecture 2)*. 2021. URL: https://gist.github.com/jteichma/ea5e54b772a0a3ec5633cbd90e1a3f5f.

[GGO22] Lukas Gonon, Lyudmila Grigoryeva, and Juan-Pablo Ortega. "Reservoir kernels and Volterra series". In: *arXiv preprint arXiv:2212.14641* (2022).

[KNT22] Florian Krach, Marc Nübel, and Josef Teichmann. "Optimal Estimation of Generic Dynamics by Path-Dependent Neural Jump ODEs". In: *arXiv preprint arXiv:2206.14284* (2022).