Learning time-dependent data with the signature transform

Adeline Fermanian

▶ **To cite this version:**

Adeline Fermanian. Learning time-dependent data with the signature transform. Statistics [math.ST]. Sorbonne Université, 2021. English. NNT : 2021SORUS224 . tel-03507274

HAL Id: tel-03507274
https://theses.hal.science/tel-03507274

Submitted on 3 Jan 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Sorbonne Université
LPSM

École doctorale **École Doctorale Sciences Mathématiques de Paris Centre**

Unité de recherche **Laboratoire de Probabilités, Statistique et Modélisation**

Thèse présentée par **Adeline Fermanian**

Soutenue le **15 octobre 2021**

En vue de l'obtention du grade de docteur de Sorbonne Université

Discipline **Mathématiques appliquées**

Spécialité **Statistique**

Learning time-dependent data with the signature transform

Thèse dirigée par Gérard Biau directeur
Benoît Cadre co-directeur

Composition du jury

Rapporteurs	Stéphane Chrétien	professeur à l'Université Lyons 2	
	Josef Teichmann	professeur à l'ETH Zürich	
Examinateurs	Claire Boyer	MCF à Sorbonne Université	
	Marianne Clausel	professeure à l'Université de Lorraine	présidente du jury
	Terry Lyons	professeur à l'University of Oxford	
	Lorenzo Zambotti	professeur à Sorbonne Université	
Directeurs de thèse	Gérard Biau	professeur à Sorbonne Université	
	Benoît Cadre	professeur à l'Université Rennes 2	

This work was supported by the Paris Ile-de-France Region via the DIM Math Innov program.

À Papy Jacques,

We have to build the Republic of Heaven where we are.

Philip Pullman

Un lieu qui n'est pas empreint de féminité n'est pas fiable.

Ibn 'Arabi

Learning time-dependent data with the signature transform

Résumé

Les applications modernes de l'intelligence artificielle amènent à travailler avec des données temporelles multivariées de grande dimension qui posent de nombreux défis. Par une approche géométrique des flux de données, la notion de signature, représentation d'un processus en un vecteur infini de ses intégrales itérées, est un outil prometteur. Ses propriétés développées dans le cadre de la théorie des chemins rugueux en font en effet un bon candidat pour jouer le rôle de features, ensuite injectées dans des algorithmes d'apprentissage. Si la définition de la signature remonte aux travaux de Chen (1960), son utilisation en apprentissage est récente et de nombreuses questions théoriques et méthodologiques restent à explorer. Nous nous intéressons donc à l'utilisation de la signature pour développer des algorithmes génériques et performants pour les données temporelles de grande dimension, ainsi que de leur fournir des garanties théoriques. Ce but se déploie principalement dans deux directions : d'une part, développer de nouveaux algorithmes prenant en entrée la signature des données, d'autre part utiliser la signature comme un outil théorique pour étudier les algorithmes existants d'apprentissage profond, via la notion récente de *neural ordinary differential equation* qui fait le lien entre apprentissage profond et équations différentielles.

Mots clés : signatures, données temporelles, apprentissage séquentiel, réseaux de neurones récurrents

Abstract

Modern applications of artificial intelligence lead to high-dimensional multivariate temporal data that pose many challenges. Through a geometric approach to data flows, the notion of signature, a representation of a process as an infinite vector of its iterated integrals, is a promising tool. Its properties, developed in the context of rough path theory, make it a good candidate to play the role of features, then injected in learning algorithms. If the definition of the signature goes back to the work of Chen (1960), its use in machine learning is recent. Many theoretical and methodological questions remain to be explored. We are therefore interested in using the signature to develop generic and efficient algorithms for high-dimensional temporal data, with theoretical guarantees. This goal is mainly deployed in two directions: on the one hand, to develop new algorithms taking the signature of the data as input, and, on the other hand, to use the signature as a theoretical tool to study existing deep learning algorithms, via the recent notion of neural ordinary differential equation which makes the link between deep learning and differential equations.

Keywords: signatures, temporal data, sequential learning, recurrent neural networks

Laboratoire de Probabilités, Statistique et Modélisation
Sorbonne Université – Campus Pierre et Marie Curie – 4 place Jussieu – 75005 Paris – France

Table des matières

Chapter 1

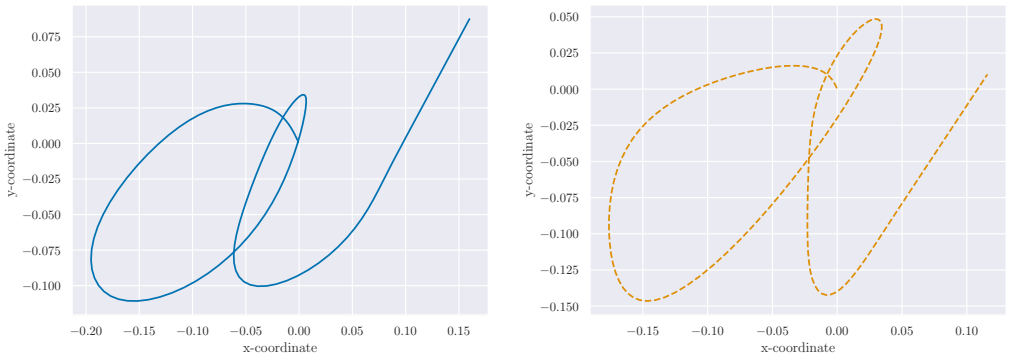
Introduction

Contents

The goal of this thesis is to study the application of signatures to statistics and machine learning. We are interested in data which have an intrinsic sequential or temporal nature, that is, a natural ordering. In some cases, such data may be considered as continuous (random) functions X from an interval $[a, b] \subset \mathbb{R}$ into $\mathbb{R}^d$, where $d \geq 1$. For example, X can be a time series representing the price of a stock observed over some time. If several stocks are observed, then X is a multivariate time series and takes its values in $\mathbb{R}^d$ with $d > 1$. Functional data also fall within this setting, with typical datasets being spectrometric curves or temperature profiles. In machine learning, the fields of speech recognition, character recognition, and natural language processing also fit in this setting, where the interval $[a, b]$ may again correspond to time but also to position in a sentence. From now on, we refer to such data as paths $X : [a, b] \to \mathbb{R}^d$ and following the notation from stochastic analysis, we write X_t for $X(t)$.

Given these paths X, we want to learn a certain output Y, function of X. For example, the output Y can be a forecast of the future value of the series, another related scalar quantity, or a label in the classification case. Learning Y from X brings up the question of the representation

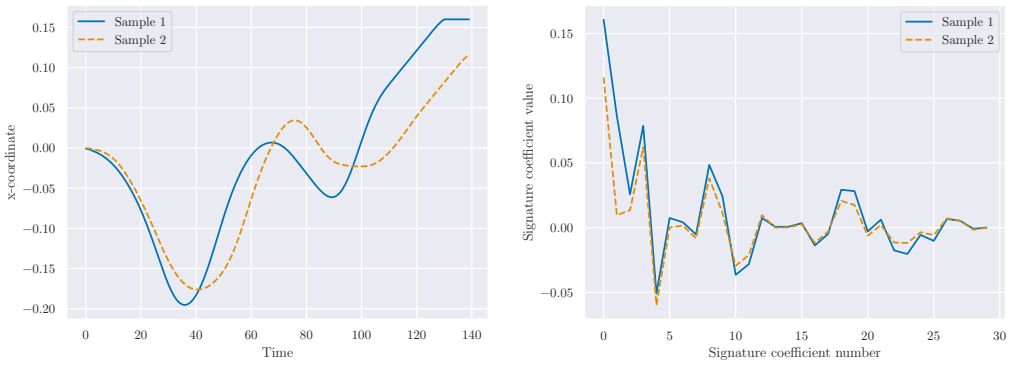
of X. Indeed, most learning algorithms such as least-squares regression, decision trees, or neural networks, take as input a vector. A natural choice for this vector is to sample the values of X but there are other options such as using a Fourier representation of X, or, more generally, expanding X on basis functions. We will see that signatures are yet another option.



(a) Sample 1 (b) Sample 2

Figure 1.1 – Two samples from the Character Trajectories dataset corresponding to the same label "a"

Let us consider as an illustrative example the two samples from the Character Trajectories dataset (Dua and Graff, 2017) shown in Figure 1.1. We are given the x and y coordinates of a pen trajectory drawing the letter "a". In Figure 1.2 we compare two possible representations of these samples: in Figure 1.2a the x-coordinates of the two samples and in Figure 1.2b the truncated signature (after a proper normalization). In a learning context, it would be desirable that representations of two samples of the same class be close, which seems to be the case for the signature, whereas they are further apart when looking at the x-coordinates. In terms of Euclidean distances, the distance between the signatures (truncated at order 4) is 0.096 whereas it is 1.0 for the sampled paths. In a nutshell, the signature captures the shape of the paths and seems therefore to be a relevant data representation.



(a) x-coordinates as a function of time (b) Normalized truncated signatures

Figure 1.2 – Two different representations of the samples from Figure 1.1

The signature was first defined for smooth paths by Chen in the 60s (Chen, 1957; 1958; 1977)

and was rediscovered in the 90s in the context of rough path theory (Lyons, 1998; Lyons et al., 2007; Friz and Victoir, 2010; Friz and Hairer, 2020), which is an important area of stochastic analysis. From a differential equation point of view, the signature summarises the effect of a path X on an output path Y when Y is "controlled" by X. More precisely, let $F : \mathbb{R}^e \to \mathbb{R}^{e\times d}$ be a vector field, and Y be the solution of the controlled differential equation (CDE)

$$dY_t = F(Y_t)dX_t, \quad Y_0 = y_0,$$

where $F(Y_t)dX_t$ should be understood as a matrix multiplication. Then, Y is entirely defined by the function F and the signature of X. In other words, Y depends on X only through its signature. From a learning perspective, Y can be seen as a quantity to predict, for example the translation of a speech, and F is now an unknown function that links the input X and the target Y. The theory of (controlled) differential equation then tells us that Y will depend only on the signature of X, making it a relevant representation or feature set for any algorithm that tries to learn F.

During the last few years, the signature has been combined to machine learning algorithms, achieving state-of-the-art performance in several domains such as character recognition (Yang et al., 2016a), finance (Perez Arribas, 2018), medicine (Morrill et al., 2019), or human action recognition (Yang et al., 2017). Several methodological works have accompanied these applied achievements, bridging the gap between stochastic analysis and machine learning (Levin et al., 2013; Király and Oberhauser, 2019; Liao et al., 2019; Kidger et al., 2019). However, many questions remain open; we will contribute to answer some of them in this manuscript.

We first present the definition of signatures in Section 1.1, then move on to its main properties in Section 1.2, and finally give an overview of our contributions in Section 1.3.

1.1 Preliminaries

To define signatures, we need two mathematical ingredients: integration along paths and tensor spaces, which we both present now. In all the following, E is a Banach space of dimension d equipped with a norm $\|\cdot\|$. Typically, it should be thought of as $\mathbb{R}^d$.

1.1.1 Paths of bounded variation

Let $X : [0,1] \to E$ be a continuous path. When $E = \mathbb{R}^d$, we denote by $(X_t^1, \dots, X_t^d)^\top \in \mathbb{R}^d$ the different coordinates of X.

Definition 1.1 (Path of bounded variation)**.** *Let $X : [0,1] \to E$ be a continuous path. For any $p \geq 1$, $[s,t] \subset [0,1]$, the p-variation of X is defined by*

$$\|X\|_{p\text{-var};[s,t]} = \Big(\sup_{(t_0,\dots,t_k)\in D_{s,t}} \sum_{i=1}^{k} \|X_{t_i} - X_{t_{i-1}}\|^p \Big)^{1/p},$$

where $D_{s,t}$ denotes the set of all finite partitions of $[s,t]$, that is,

$$D_{s,t} = \big\{(t_0,\dots,t_k) \,|\, k \geq 0,\ s = t_0 < t_1 < \cdots < t_{k-1} < t_k = t\big\}.$$

The path X is said to be of finite p-variation on $[s,t]$ if its p-variation is finite.

In this manuscript, we restrict to the case $p = 1$ but it is worth keeping in mind that the extension to less regular paths (i.e., $p > 1$) is an active area of research. When $[s,t] = [0,1]$,

we often write $\|X\|_{1\text{-var}}$ instead of $\|X\|_{1\text{-var};[0,1]}$. If $\|X\|_{1\text{-var}} < \infty$ we say that X is of bounded variation and denote by $BV(E)$ the set of paths of bounded variation. If X is continuously differentiable and $X' : [0,1] \to \mathbb{R}^d$ denotes its derivative, then

$$\|X\|_{1\text{-var}} = \int_0^1 \|X'_t\| dt.$$

From a geometric point of view, $\|X\|_{1\text{-var}}$ is the length of the path X. Note also that $\|\cdot\|_{1\text{-var}}$ is a semi-norm (the 1-variation of a constant path being null) and that we can define a norm on $BV(E)$ by letting

$$\|X\|_{BV(E)} = \|X\|_{1\text{-var}} + \sup_{t\in[0,1]} \|X_t\|.$$

This norm equips $BV(E)$ with a Banach structure. Given a path of bounded variation, it is possible to integrate along this path, which defines the Riemann-Stieljes integral.

Definition 1.2 (Riemann-Stieljes integral)**.** *Let X and Y be two paths from $[0,1]$ to $\mathbb{R}$ and $[s,t] \subset [0,1]$. For any $n \geq 0$, let $(t_0^{(n)}, \dots, t_n^{(n)}) \in D_{s,t}$ be a partition of $[s,t]$ of length n, such that its mesh size converges to zero when n grows:*

$$\sup_{1\leq i\leq n} |t_i^{(n)} - t_{i-1}^{(n)}| \underset{n\to\infty}{\to} 0.$$

Let $(s_1^{(n)}, \dots, s_n^{(n)})$ be a sequence such that, for any $i \in \{1, \dots, n\}$, $s_i^{(n)} \in [t_{i-1}^{(n)}, t_i^{(n)}]$. Then, if the sum

$$\sum_{i=1}^{n} Y_{s_i^{(n)}} (X_{t_i^{(n)}} - X_{t_{i-1}^{(n)}})$$

converges to a limit I independent of the choice of partitions, we say that the Riemann-Stieljes integral of Y against X exists, is equal to I, and denoted by $I := \int_s^t Y_u dX_u$.

This definition generalizes to the case when X and Y are vector-valued by letting, if $X, Y : [0,1] \to \mathbb{R}^d$,

$$\int_s^t Y_u dX_u = \begin{pmatrix} \int_s^t Y_u^1 dX_u^1 \\ \vdots \\ \int_s^t Y_u^d dX_u^d \end{pmatrix}.$$

It can be shown that if Y is continuous and X of bounded variation, then the Riemann-Stieljes integral $\int_s^t Y_u dX_u$ exists (see, e.g., Friz and Victoir, 2010, Proposition 2.2). This result has been extended to the case when X has finite p-variation and Y finite q-variation with $1/p + 1/q > 1$; I is then called the Young integral (Young, 1936). From now on, all integrals will be taken as Riemann-Stieljes integrals. Note that if X is continuously differentiable, then

$$\int_s^t Y_u dX_u := \int_s^t Y_u X'_u du,$$

where the last integral is the usual Riemann integral (extended to $\mathbb{R}^d$).

1.1.2 Tensor spaces

We now introduce some definition and notation on tensor spaces. We refer the reader to Purbhoo (2012) for supplementary details on tensor products and to Appendix A for proofs.

Definition 1.3 (Tensor product of vector spaces)**.** *Let E and F be two vector spaces. A tensor product of E and F, denoted by $E \otimes F$, is a vector space with a bilinear map $\phi : E \times F \to E \otimes F$ such that for any basis $e = (e_i)_{i\in I}$ of E and $f = (f_j)_{j\in J}$, then*

$$\phi(e \times f) = \big\{\phi(e_i, f_j) \,|\, e_i \in e, f_j \in f\big\}$$

is a basis of $E \otimes F$. For any $x \in E$ and $y \in F$, $\phi(x, y)$ is denoted $x \otimes y$ and called the tensor product of x and y.

It is known that such a product exists and is unique up to isomorphism. The nth tensor power of a vector space E is defined as the order n tensor product of E with itself:

$$E^{\otimes n} := \overbrace{E \otimes \cdots \otimes E}^{n}.$$

By convention, $E^{\otimes 0} = \mathbb{R}$. It may be useful to identify $E^{\otimes n}$ with the space of homogeneous non-commuting polynomials of degree n. Indeed, let $(e_1, \dots, e_d)$ be a basis of E (assumed to be finite-dimensional). Then, any element of $E^{\otimes n}$ can be written as a sum

$$\sum_{I=(i_1,\dots,i_n)\subset\{1,\dots,d\}^n} a^I e_{i_1} \otimes \cdots \otimes e_{i_n},$$

which can be thought of as $\sum a^I X_{i_1} \dots X_{i_n}$ where $X_1, \dots, X_n$ are non commuting indeterminates. Note that, by construction, $\dim(E \otimes F) = \dim(E) \times \dim(F)$. Thus, if $E = \mathbb{R}^d$, $E^{\otimes n}$ is of dimension d^n. Then, we can also identify $E^{\otimes n}$ with $\mathbb{R}^{d^n}$ (for example, $E^{\otimes 2}$ can be identified with the space of $d \times d$ matrices).

Definition 1.4. *We denote by $T(E)$ the space of formal series of tensors of E, i.e.,*

$$T(E) = \big\{(a_0, \dots, a_n, \dots) \,|\, \forall n \geq 0,\, a_n \in E^{\otimes n}\big\},$$

and

$$T^N(E) = \big\{(a_0, \dots, a_N) \,|\, \forall n \in \{0, \dots, N\},\, a_n \in E^{\otimes n}\big\},$$

the truncated tensor space up to order N.

We sometimes write elements of $T(E)$ as formal sums: any $a \in T(E)$ may be written as $a = \sum_{n\geq 0} a_n$. We endow $T(E)$ with the following operations: for any $a, b \in T(E)$, $\lambda \in \mathbb{R}$,

$$\begin{aligned} a + b &= (a_0 + b_0, a_1 + b_1, \dots, a_n + b_n, \dots) \\ \lambda \cdot a &= (\lambda \cdot a_0, \lambda \cdot a_1, \dots, \lambda \cdot a_n, \dots) \\ a \otimes b &= (c_0, c_1, \dots, c_n, \dots), \quad \text{where} \quad c_n = \sum_{k=0}^{n} a_k \otimes b_{n-k}. \end{aligned}$$

Then, the following proposition is clear from the definition.

Proposition 1.1. *$(T(E), +, \cdot, \otimes)$ is a real non-commutative algebra with neutral element $\mathbf{1} := (1, 0, \dots, 0, \dots)$.*

Definition 1.5. *The canonical projection π_N of an element of $T(E)$ on the truncated tensor*

space $T^N(E)$ is defined by

$$T(E) \to T^N(E)$$
$$\pi_N : (a_0, \dots, a_N, \dots) \mapsto (a_0, \dots, a_N).$$

An element of $T(E)$ is invertible if and only if $a_0 \neq 0$. Thus, the space

$$\widetilde{T}(E) = \{a \in T(E) \,|\, \pi^0(a) = 1\}$$

is a group.

Proposition 1.2. *$(\widetilde{T}(E), \otimes)$ is a Lie group, and for any $a \in \widetilde{T}(E)$,*

$$a^{-1} = \sum_{k \geq 0} (\mathbf{1} - a)^{\otimes k}.$$

The tensor powers of E may be endowed with an Euclidean scalar product and its associated norm, defined as follows.

Definition 1.6. *Let $a, b \in E^{\otimes n}$ and $(e_1, \dots, e_d)$ a basis of E. Then, if*

$$a = \sum_{I=(i_1,\dots,i_n) \subset \{1,\dots,d\}^n} a^I e_{i_1} \otimes \cdots \otimes e_{i_n}, \quad b = \sum_{I=(i_1,\dots,i_n) \subset \{1,\dots,d\}^n} b^I e_{i_1} \otimes \cdots \otimes e_{i_k},$$

the scalar product and norm on $E^{\otimes n}$ are defined by

$$\langle a, b \rangle_{E^{\otimes n}} = \sum_{I \subset \{1,\dots,d\}^n} a^I b^I, \quad \|a\|^2_{E^{\otimes n}} = \sum_{I \subset \{1,\dots,d\}^n} (a^I)^2.$$

We can then endow $T(E)$ with the following scalar product: for any $a, b \in T(E)$,

$$\langle a, b \rangle_{T(E)} = \sum_{k \geq 0} \langle a_k, b_k \rangle_{E^{\otimes k}}. \tag{1.1}$$

The square-summable elements in $T(E)$ form a Hilbert space for this scalar product.

1.1.3 The signature of a path

Definition 1.7. *Let $X \in BV(\mathbb{R}^d)$. For any $[s,t] \subset [0,1]$, the signature of X on $[s,t]$ is defined by*

$$S_{[s,t]}(X) = (1, \mathbf{X}^1_{[s,t]}, \mathbf{X}^2_{[s,t]}, \dots, \mathbf{X}^n_{[s,t]}, \dots) \in T(\mathbb{R}^d),$$

where, for each integer n,

$$\mathbf{X}^n_{[s,t]} = \int\limits_{s<u_1<\cdots<u_n<t} \cdots \int dX_{u_1} \otimes \cdots \otimes dX_{u_n} \in (\mathbb{R}^d)^{\otimes n}.$$

Before going any further, we introduce a series of notation.

- For any $N \geq 1$, the signature truncated at order N is

$$S^N(X) := (1, \mathbf{X}^1, \mathbf{X}^2, \dots, \mathbf{X}^N) = \pi_N\big(S(X)\big).$$

- The simplex in $[s,t]^n$ is denoted by

$$\Delta_{n;[s,t]} = \{(u_1, \dots, u_n) \in [s,t]^n \mid s < u_1 < \cdots < u_n < t\}.$$

- For any multi-index $(i_1, \dots, i_n) \subset \{1, \dots, d\}^n$, the signature coefficient along $(i_1, \dots, i_n)$ is denoted by

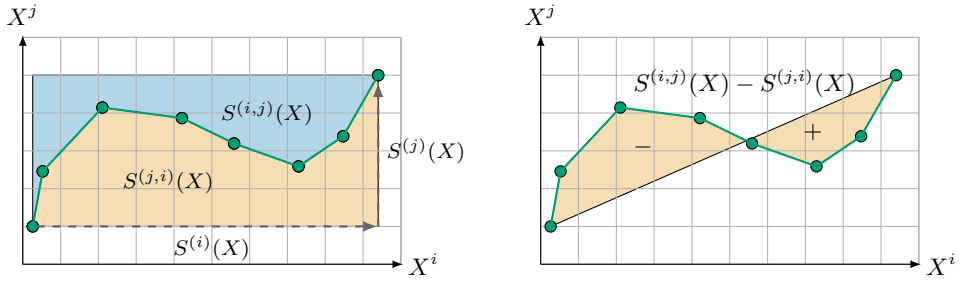
$$S^{(i_1,\dots,i_n)}_{[s,t]}(X) = \int\cdots\int_{(u_1,\dots,u_n)\in\Delta_{n;[s,t]}} dX^{i_1}_{u_1} \dots dX^{i_n}_{u_n}.$$

  If $(e_1, \dots, e_d)$ is the canonical basis of $\mathbb{R}^d$, then

$$\mathbf{X}^n_{[s,t]} = \sum_{(i_1,\dots,i_n)\subset\{1,\dots,d\}^n} S^{(i_1,\dots,i_n)}_{[s,t]}(X) e_{i_1} \otimes \cdots \otimes e_{i_n}.$$

- When $[s,t] = [0,1]$ we omit the domain of integration and write for example $S(X)$ instead of $S_{[s,t]}(X)$.

Remark 1.1. *If X is of bounded p-variation with $1 < p < 2$, then the signature is still well-defined using the Young integral. If $p \geq 2$, the iterated integrals are no longer uniquely defined. This is where rough path theory comes into play to show their existence for various stochastic processes (see, e.g., Le Jan and Qian, 2013, for the case of Brownian motion).*



(a) Order 1 and 2 coefficients correspond to increments and areas.

(b) The Levy area corresponds to the signed orange area.

Figure 1.3 – Geometric interpretation of signature coefficients

In short, the signature is an infinite vector in $T(\mathbb{R}^d)$ corresponding to the iterated integrals of X against itself, on a simplex. It has a natural interpretation in terms of areas under the path, as shown in Figure 1.3. The first order terms are just the increments of the path: for any $1 \leq i \leq d$,

$$S^{(i)}(X) = X^i_1 - X^i_0.$$

The second order terms correspond to areas under the curve delimited by each pair of coordinates (Figure 1.3a). Moreover, the Levy area, which is the signed area between a curve and the chord connecting its two endpoints (Figure 1.3b), can be recovered from these coefficients of order 2. If $\mathscr{A}_{i,j}$ is the Levy area of the curve (X^i_t, X^j_t), then

$$\mathscr{A}_{i,j} = S^{(i,j)}(X) - S^{(j,i)}(X).$$

This quantity corresponds exactly to the coefficient of order 2 of the logsignature, which will be

defined later.

Example 1.1 (Path in 2d)**.** *If $d = 2$, $X_t = (X_t^1, X_t^2)$, then $(\mathbb{R}^2)^{\otimes 1}$ can be identified with $\mathbb{R}^2$ and $(\mathbb{R}^2)^{\otimes 2}$ with the space of 2×2 matrices:*

$$\mathbf{X}^1 = \int_0^1 dX_t = \begin{pmatrix} \int_0^1 dX_u^1 \\ \int_0^1 dX_u^2 \end{pmatrix}, \quad \mathbf{X}^2 = \int_0^1 \int_0^u dX_v \otimes dX_u = \begin{pmatrix} \int_0^1 \int_0^u dX_v^1 dX_u^1 & \int_0^1 \int_0^u dX_v^1 dX_u^2 \\ \int_0^1 \int_0^u dX_v^2 dX_u^1 & \int_0^1 \int_0^u dX_v^2 dX_u^2 \end{pmatrix}.$$

The coefficient of order 3, $\mathbf{X}^3$, can be seen as a $2 \times 2 \times 2$ tensor, and so on.

Example 1.2 (Parametrized curve)**.** *Let $X \in BV(\mathbb{R}^2)$ be a parametrized curve: for any $t \in [0,1]$, $X_t = (t, f(t))$, where $f : \mathbb{R} \to \mathbb{R}$ is a smooth function. Then,*

$$S^{(1)}(X) = \int_0^1 dX_u^1 = \int_0^1 du = 1$$
$$S^{(2)}(X) = \int_0^1 dX_u^2 = \int_0^1 f'(u)du = f(1) - f(0),$$

where f' denotes the derivative of f. Similarly, the signature coefficient along $(1,2)$ is

$$S^{(1,2)}(X) = \int_0^1 \int_0^u dX_v^1 dX_u^2 = \int_0^1 \Big(\int_0^u dv \Big) f'(u)du = \int_0^1 uf'(u)du = f(1) - \int_0^1 f(u)du,$$

and so on.

Example 1.3 (Linear path)**.** *If $X : [0,1] \to \mathbb{R}^d$ is a linear path, i.e., $X_t = X_0 + (X_1 - X_0)t$, then for any $I = (i_1, \dots, i_n) \in \{1, \dots, d\}^n$,*

$$\begin{aligned} S^I(X) &= \int_{\Delta_n} dX_{u_1}^{i_1} \dots dX_{u_n}^{i_n} = \int_{\Delta_n} (X_1 - X_0)^{i_1} \dots (X_1 - X_0)^{i_n} du_1 \dots du_n \\ &= \prod_{j=1}^n (X_1 - X_0)^{i_j} \int_{\Delta_n} du_1 \dots du_n = \frac{1}{n!} \prod_{j=1}^n (X_1 - X_0)^{i_j}. \end{aligned}$$

With tensor notation, we have $\mathbf{X}^n = (X_1 - X_0)^{\otimes n}$.

Example 1.4 (Path in one dimension)**.** *In one dimension, the signature is directly related to the moments of X. Indeed, let $X : [0,1] \to \mathbb{R}$ be a one-dimensional path. Then, for any $k \geq 0$, $\mathbf{X}^k \in \mathbb{R}$, and, for any $t \in [0,1]$,*

$$\mathbf{X}^n{}_{[0,t]} = \int_{\Delta_{n;[0,t]}} dX_{u_1} \dots dX_{u_n} = \frac{1}{n!}(X_t - X_0)^n.$$

Therefore, if X is a time-continuous stochastic process of bounded variation, then

$$\mathbb{E}\big[\mathbf{X}_{[0,t]}^n\big] = \frac{1}{n!}\mathbb{E}\big[(X_t - X_0)^n\big].$$

The link between signatures and moments is also relevant in the general case. Indeed, signatures can be thought of as moment-generating functions for stochastic processes. Recall that if

Z is a real random variable, then its moment-generating function, defined by

$$\lambda \mapsto \mathbb{E}[e^{\lambda Z}] = \sum_{k=0}^{\infty} \frac{\lambda^k}{k!} \mathbb{E}[Z^k],$$

characterizes the law of Z. Now take a stochastic process X of bounded variation, then Chevyrev and Lyons (2016) construct a characteristic function for X as $M \mapsto \mathbb{E}[M(S(X))]$, where, without going into the details of its definition, M plays the role of λ. A corollary is that, provided $\mathbb{E}[S(X)]$ is well-defined, then the law of X (and of $S(X)$) is entirely determined by $\mathbb{E}[S(X)]$. In other words, the signature is the equivalent of an exponential for vector-valued processes, and signature coefficients of order k are the equivalent of moments.

1.2 Properties of the signature

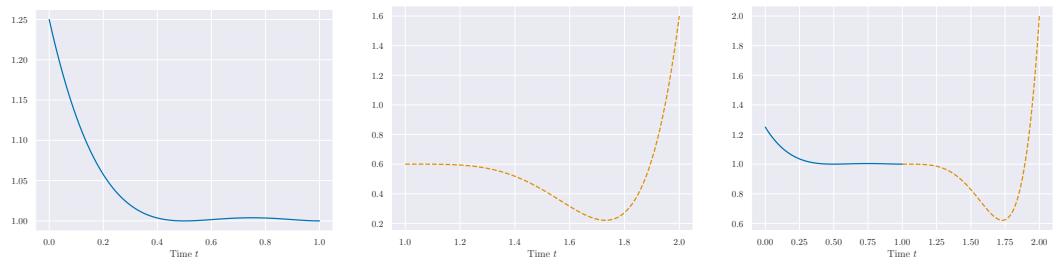
Signatures have a number of good analytic and algebraic properties, which guarantee their relevance in machine learning. We briefly present them here and refer to Appendix A for the proofs.

1.2.1 Invariances and uniqueness

First, we present several algebraic properties that translate properties on paths into properties on signatures. We proceed with a description of the loss of information resulting from taking the signature and with a uniqueness result.

The first property, known as Chen's identity, provides a formula to compute the signature of a concatenation of paths, which is crucial for practical implementation. Let $X : [s,t] \to \mathbb{R}^d$ and $Y : [t,u] \to \mathbb{R}^d$ be two paths. The concatenation of X and Y (see Figure 1.4) is the path $X * Y : [s,u] \to \mathbb{R}^d$ defined, for any $v \in [s,u]$, by

$$(X * Y)_v = \begin{cases} X_v & \text{if } v \in [s,t] \\ X_t + Y_v - Y_t & \text{if } v \in [t,u]. \end{cases}$$



X Y $X * Y$

Figure 1.4 – Concatenation of paths

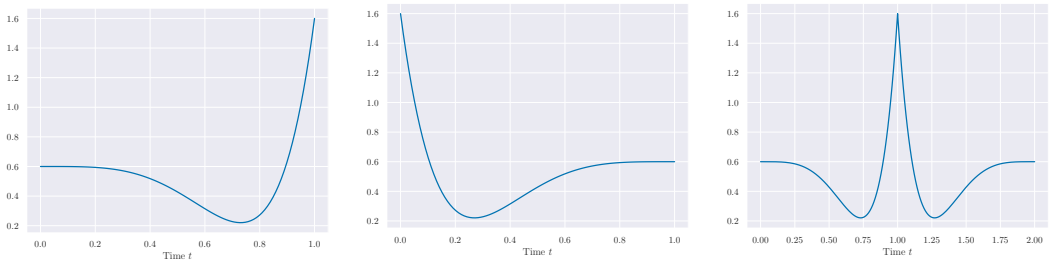
Theorem 1.3 (Chen's identity)**.** *Let $X : [s,t] \to \mathbb{R}^d$ and $Y : [t,u] \to \mathbb{R}^d$ be two continuous paths of bounded variation. Then*

$$S_{[s,u]}(X * Y) = S_{[s,t]}(X) \otimes S_{[t,u]}(Y).$$

A crucial consequence of Chen's identity is that the signature of a piecewise linear path can be easily computed.

Example 1.5. *Let $X : [0,1] \to \mathbb{R}^d$ be a piecewise linear path and let $0 = t_0 < t_1 < \cdots < t_k = 1$ be a partition such that X is linear on each $[t_{j-1}, t_j]$. On each $[t_{j-1}, t_j]$, X is a linear path so $S_{[t_{j-1},t_j]}(X)$ can be obtained from Example 1.3. Then, by Chen's identity,*

$$S(X) = S_{[t_0,t_1]}(X) \otimes \cdots \otimes S_{[t_{k-1},t_k]}(X).$$



(a) X (b) $\overleftarrow{X}$ (c) $X * \overleftarrow{X}$ is a tree-like path

Figure 1.5 – Time-reversal of a path

The next property shows that the inverse of the signature of a path is the signature of its time-reversal (see Figures 1.5a and 1.5b).

Proposition 1.4 (Time-reversal)**.** *Let $X \in BV(\mathbb{R}^d)$ and let $\overleftarrow{X}$ be its time-reversal, defined, for any $t \in [0,1]$, by $\overleftarrow{X}_t = X_{1-t}$. Then,*

$$S(X) \otimes S(\overleftarrow{X}) = \mathbf{1}.$$

Given that the first element of the signature is 1, signatures are invertible elements in $T(\mathbb{R}^d)$. Proposition 1.4 then tells us that $S(X)^{-1} = S(\overleftarrow{X})$. We can rephrase Chen's identity by saying that the signature is a homomorphism from $(BV(\mathbb{R}^d), *)$ to the group $(\widetilde{T}(\mathbb{R}^d), \otimes)$. In other words, the signature maps properties of paths in $BV(\mathbb{R}^d)$ to algebraic properties in the group $\widetilde{T}(\mathbb{R}^d)$.

By definition it is clear that signatures are invariant by translation (take $x \in \mathbb{R}$ and $\overline{X}_t = x + X_t$, then $d\overline{X}_t = dX_t$ and $\overline{\mathbf{X}}^n = \mathbf{X}^n$). The next lemma gives their second invariance, namely to reparametrizations.

Lemma 1.5 (Invariance under time reparametrisation)**.** *Let $X \in BV(\mathbb{R}^d)$, $\psi : [0,1] \to [0,1]$ be a reparametrisation (continuously differentiable non-decreasing surjection). Let $\widetilde{X} \in BV(\mathbb{R}^d)$ be the reparametrization of X, that is, $\widetilde{X}_t = X_{\psi(t)}$ for any $t \in [0,1]$. Then, for any $[s,t] \subset [0,1]$,*

$$S_{[s,t]}(\widetilde{X}) = S_{[\psi(s),\psi(t)]}(X).$$

This lemma is a direct consequence of the change of variable formula for Riemann-Stieljes integral. From a learning perspective, this property is important since it amounts to saying that the signature does not contain any information on the time parametrization. Depending on the context, invariance under translation and reparametrization can be either an advantage or a disadvantage. If these invariances are not needed, the notion of *embedding* or *augmentation*

provides a useful way to reincorporate information into the signature features. The idea is to modify the path X before computing the signature, for example by adding new coordinates and augmenting the dimension d. In particular, the signature of the time-augmented path $t \mapsto \overline{X}_t = (X_t^\top, t)^\top \in \mathbb{R}^{d+1}$ is not invariant under reparametrization of X.

We now investigate what it means for two paths to have the same signature. It is clear that for two paths X and Y, $S(X) = S(Y)$ does not imply that $X = Y$. Indeed, we have seen that the signature is invariant by translation and reparametrization. Moreover, according to Theorem 1.3 and Proposition 1.4, $S(X * \overleftarrow{X}) = \mathbf{1}$ so the path $X * \overleftarrow{X}$ has the same signature as a constant path. This path is actually an example of a *tree-like* path (Hambly and Lyons, 2010, Definition 1.2)—see Figure 1.5c. Let X, Y, and Z be three paths, then $X * Y * \overleftarrow{Y} * \overleftarrow{X}$ and $X * Y * \overleftarrow{Y} * Z * \overleftarrow{Z} * \overleftarrow{X}$ are other examples of tree-like paths: informally, they can be reduced to a null path by cancelling the parts that retrace themselves. With translation and parametrization, the notion of tree-like path encapsulates exactly the information lost when taking the signature. In other words, signatures do not see the initial position of the path, its parametrization, and its subpaths that are tree-like. Apart from these, the path can be entirely recovered from its signature.

Definition 1.8 (Tree-like path)**.** *A path $X : [0,1] \to \mathbb{R}^d$ is tree-like if there exists a continuous function $h : [0,1] \to [0, +\infty)$ such that $h(0) = h(1) = 0$ and such that for any $s, t \in [0,1]$, $s \leq t$,*

$$\|X_s - X_t\| \leq h(s) + h(t) - 2 \inf_{u \in [s,t]} h(u).$$

The function h is called a height function of X, and we say that X is a Lipschitz tree-like path if h can be chosen of bounded variation.

Definition 1.9. *Let $X, Y \in BV(\mathbb{R}^d)$, we say that X and Y are tree-like equivalent if $X * \overleftarrow{Y}$ is a tree-like path, and denote this relation by $X \sim Y$.*

Example 1.6. *A sufficient condition for a path not to be tree-like is to have one monotone coordinate. Indeed, let $X = (X^1, \dots, X^d)^\top \in BV(\mathbb{R}^d)$ be such that X^1 is a non-constant increasing function. By contradiction, assume that X is tree-like and let h be a height function. Then, for any $t \in [0,1]$,*

$$|X_t^1 - X_0^1| \leq \|X_t - X_0\| \leq h(t).$$

Thus, h is an increasing function. But $h(1) = h(0) = 0$, and therefore, for any $t \in [0,1]$, $h(t) = 0$ and $X_t = X_0$. X is therefore a constant path, which is a contradiction.

We can now state the uniqueness theorem of Hambly and Lyons (2010, Theorem 4).

Theorem 1.6. *For any $X, Y \in BV(\mathbb{R}^d)$, then $S(X) = \mathbf{1}$ if and only if X is tree-like. Moreover, $S(X) = S(Y)$ if and only if $X \sim Y$.*

Note that, by Chen's identity (Theorem 1.3), the second part of the theorem is a direct consequence of the first. Extending this uniqueness result to less regular paths is an active area of research (see, e.g., Le Jan and Qian, 2013; Boedihardjo and Geng, 2015; Boedihardjo et al., 2016). We conclude by a lemma that gives a sufficient condition to have a unique signature.

Lemma 1.7. *Let $X \in BV(\mathbb{R}^d)$ be a path with at least one strictly monotone coordinate. Then $S(X)$ determines X uniquely, up to reparametrizations and translations.*

In a learning context, we can use an embedding with a monotone coordinate to ensure uniqueness of signature features. For example, the time-augmented path $\overline{X}$ mentioned above has a unique signature.

1.2.2 Analytic properties

We now turn to the analytic properties of the signature. First, the next lemma shows that the signature is a solution of an "exponential" differential equation.

Lemma 1.8. *Let $X \in BV(\mathbb{R}^d)$ and $Y : [0,1] \to T^N(\mathbb{R}^d)$. The controlled differential equation*

$$dY_t = \pi_N(Y_t \otimes dX_t), \quad Y_0 = (1, 0, \dots, 0),$$

has a unique solution. This solution is the signature of X truncated at order N, that is, the path from $[0,1]$ to $T^N(\mathbb{R}^d)$ defined by $t \mapsto S^N_{[0,t]}(X)$.

An immediate consequence is that the truncated signatures seen as maps from $BV(\mathbb{R}^d)$ to $T^N(\mathbb{R}^d)$ are continuous.

Corollary 1.9. *The truncated signature map $\pi_N \circ S : BV(\mathbb{R}^d) \to T^N(\mathbb{R}^d)$ is continuous for any $N \geq 0$.*

This corollary is crucial if one wants to quantify the distance between two paths by the distance between their signatures: if two paths are close (in the bounded variation norm), then their signatures are close (for the tensor norm). Moreover, we have an upper bound on the norm of signature coefficients, which ensures that high-order coefficients are small.

Proposition 1.10. *Let $X \in BV(\mathbb{R}^d)$. Then, for any $n \geq 0$,*

$$\|\mathbf{X}^n\|_{(\mathbb{R}^d)^{\otimes n}} \leq \frac{1}{n!}\|X\|^n_{1\text{-var}} < \infty$$

and

$$\|S(X)\|_{T(\mathbb{R}^d)} \leq \exp(\|X\|_{1\text{-var}}) < \infty.$$

This proposition is the main argument for truncating signatures when learning: from an approximation point of view, if N is large, then $S^N(X)$ is close to $S(X)$. Moreover, it guarantees that the scalar product in $T(\mathbb{R}^d)$, defined by (1.1), is well-defined for signatures. This allows to define the signature kernel:

$$\begin{aligned} K : BV(\mathbb{R}^d) \times BV(\mathbb{R}^d) &\to \mathbb{R} \\ (X, Y) &\mapsto \langle S(X), S(Y) \rangle_{T(\mathbb{R}^d)}. \end{aligned} \tag{1.2}$$

This opens a wide range of applications: any kernel-based algorithm can be extended to sequential data using the signature kernel (Király and Oberhauser, 2019).

Furthermore, the signature naturally appears in the context of controlled differential equations, that is, differential equations of the form

$$dY_t = F(Y_t)dX_t, \quad Y_0 = y_0, \tag{1.3}$$

where $F : \mathbb{R}^e \to \mathbb{R}^{e \times d}$ is a vector field, $X \in BV(\mathbb{R}^d)$, $Y \in BV(\mathbb{R}^e)$, and $y_0 \in \mathbb{R}^e$. For illustrative purposes, assume that F is a linear vector field. For any $y \in \mathbb{R}^e$, $x \in \mathbb{R}^d$, $F(y)x$ is linear both in y and x and can therefore be rewritten as $F(y)x = \overline{F}(x)y$, where $\overline{F} : \mathbb{R}^d \to \mathbb{R}^{e \times e}$ is linear.

Then, Picard iterations yield

$$\begin{aligned} Y_t = y_0 + \int_0^t F(Y_u)dX_u &= y_0 + \int_0^t \overline{F}(dX_u)Y_u \\ &= y_0 + \int_0^t \overline{F}(dX_u)\Big(y_0 + \int_0^u \overline{F}(dX_v)Y_v\Big) \\ &= y_0 + \overline{F}(\int_0^t dX_u)y_0 + \int_0^t\int_0^u \overline{F}(dX_u)\overline{F}(dX_v)Y_v \\ &= y_0 + \overline{F}(\mathbf{X}^1_{[0,t]})y_0 + \int_0^t\int_0^u \overline{F}(dX_u)\overline{F}(dX_v)Y_v. \end{aligned}$$

We see that the signature of order 1 naturally appears. Further iteration brings up terms of higher order. Let $\overline{F}^{\otimes n} : (\mathbb{R}^d)^{\otimes n} \to \mathbb{R}^e$ be defined by $\overline{F}^{\otimes n}(e_1 \otimes \cdots \otimes e_n) = \overline{F}(e_1)\cdots\overline{F}(e_n)$, then

$$\begin{aligned} Y_t &= y_0 + \overline{F}(\mathbf{X}^1_{[0,t]})y_0 + \int_0^t\int_0^u \overline{F}^{\otimes 2}(dX_u \otimes dX_v)Y_v \\ &= y_0 + \overline{F}(\mathbf{X}^1_{[0,t]})y_0 + \overline{F}^{\otimes 2}(\mathbf{X}^2_{[0,t]})y_0 + \cdots = \sum_{n=0}^{\infty} \overline{F}^{\otimes n}(\mathbf{X}^n_{[0,t]})y_0, \end{aligned} \tag{1.4}$$

(convergence of the infinite sum is ensured by Proposition 1.10). We summarize this result in the following proposition.

Proposition 1.11. *Let $F : \mathbb{R}^e \to \mathbb{R}^{e\times d}$ be a linear vector field, $X \in BV(\mathbb{R}^d)$, $y_0 \in \mathbb{R}^e$ and Y be the solution of the CDE (1.3). Then, Y_t is a linear function of $S_{[0,t]}(X)$.*

The proposition can be extended to smooth non-linear vector fields, but in this case conditions on the norms of differentials of F must be added to ensure convergence of the approximation (1.4). In addition to motivating the definition of the signature, we will see that this proposition is very useful in a learning situation. Indeed, assume that Y is a target and X a predictor, if the learning model can be approximated by (1.3), then Proposition 1.11 reduces the problem to a linear model on the signature.

We proceed with the definition of the shuffle product, which endows the space of linear forms on the signature with a structure of algebra. Then, by applying the Stone-Weierstrass theorem, we obtain that this linear space is dense in the space of continuous functions of paths, which provides a more general approximation theorem.

Definition 1.10 (Shuffle product)**.** *A permutation σ of $\{1,\dots,n+m\}$ is called a (n,m)-shuffle if $\sigma^{-1}(1) < \cdots < \sigma^{-1}(n)$ and $\sigma^{-1}(n+1) < \cdots < \sigma^{-1}(n+m)$.*

Let $I = (i_1,\dots,i_n)$ and $J = (j_1,\dots,j_m)$ be two multi-indices with $i_1,\dots,i_n,j_1,\dots,j_m \in \{1,\dots,d\}$. Then, the shuffle product of I and J, denoted by $I \sqcup J$, is a finite set of multi-indices of length $n+m$, defined by

$$I \sqcup J = \big\{(r_{\sigma(1)},\dots,r_{\sigma(n+m)}) \,|\, \sigma \in \mathit{Shuffles}(n,m)\big\},$$

with $(r_1,\dots,r_n,r_{n+1},\dots,r_{n+m}) = (i_1,\dots,i_n,j_1,\dots,j_m)$.

This amounts to shuffling the words I and J without changing the order of their letters. Note that there are $(m+n)!/n!m!$ elements in $I \sqcup J$ and that the shuffle product is commutative and associative.

Theorem 1.12 (Shuffle product identity)**.** *Let $X \in BV(\mathbb{R}^d)$ and two multi-indices $I = (i_1, \dots, i_n) \subset \{1, \dots, d\}^n$ and $J = (j_1, \dots, j_m) \subset \{1, \dots, d\}^m$. Then,*

$$S^I(X)S^J(X) = \sum_{K \in I \sqcup J} S^K(X).$$

Example 1.7. *Let $I = (1)$ and $J = (2)$, then Theorem 1.12 yields that*

$$S^{(1)}(X)S^{(2)}(X) = S^{(1,2)}(X) + S^{(2,1)}(X).$$

Similarly, if $I = (1)$ and $J = (2, 3)$, then we obtain

$$S^{(1)}(X)S^{(2,3)}(X) = S^{(1,2,3)}(X) + S^{(2,3,1)}(X) + S^{(2,1,3)}(X).$$

Theorem 1.13 (Linear approximations)**.** *Let D be a compact subset of $BV(\mathbb{R}^d)$ of paths that are not tree-like equivalent. Let $f : D \to \mathbb{R}$ be a continuous function. Then, for any $\varepsilon > 0$, there exists $a \in T(\mathbb{R}^d)$ such that, for any $X \in D$,*

$$\left| f(X) - \langle a, S(X) \rangle \right| \leq \varepsilon.$$

This theorem is the core approximation result on signatures: any continuous regression function that maps an input path X to an output $Y \in \mathbb{R}$ can be approximated by a linear function on signatures. While in Proposition 1.11, the coefficients of the linear approximation of Y are explicit, Theorem 1.13 is only an existence result.

1.2.3 The logsignature

We conclude this section by presenting a related object, the *logsignature*, which also gives a finer understanding of the subspace of $\widetilde{T}(\mathbb{R}^d)$ containing signatures. This subspace is actually characterized by the shuffle product property, in the sense that any element of $T(\mathbb{R}^d)$ that satisfies this property is the signature of a path with bounded variation. Moreover, it can be shown that it is satisfied if and only if the logarithm of the element is a Lie formal series, which brings up the notion of logsignature.

Definition 1.11. *Let $a \in \widetilde{T}(\mathbb{R}^d)$. The element a is said to be group-like if for any multi-indices $I \subset \{1, \dots, d\}^n$, $J \subset \{1, \dots, d\}^m$, then*

$$a^I a^J = \sum_{K \in I \sqcup J} a^K,$$

where $a = \sum_{n=0}^{\infty} \sum_{I=(i_1,\dots,i_n) \subset \{1,\dots,d\}^n} a^I e_{i_1} \otimes \cdots \otimes e_{i_n}$. We denote by $G^{(*)}(\mathbb{R}^d)$ the space of group-like elements of $\widetilde{T}(\mathbb{R}^d)$.

Theorem 1.12 states that signature are group-like elements of $\widetilde{T}(\mathbb{R}^d)$. In fact, all group-like elements of $\widetilde{T}(\mathbb{R}^d)$ are signatures of some paths of bounded variation (Hambly and Lyons, 2010).

Proposition 1.14. *Let $N \geq 0$ and $G^{(N)}(\mathbb{R}^d) = \pi_N(G^{(*)}(\mathbb{R}^d))$. Every element of $G^{(N)}(\mathbb{R}^d)$ is the truncated signature of a path of bounded variation. In other words, $G^{(N)}(\mathbb{R}^d)$ is the range of the function*

$$\pi_N \circ S : BV(\mathbb{R}^d) \to T^N(\mathbb{R}^d).$$

Definition 1.12. *The exponential in the tensor algebra is the function* $\exp : T(\mathbb{R}^d) \to \widetilde{T}(\mathbb{R}^d)$ *defined, for any* $a \in T(\mathbb{R}^d)$*, by*

$$\exp(a) = \sum_{n \geq 0} \frac{a^{\otimes n}}{n!}.$$

The logarithm is the function $\log : \widetilde{T}(\mathbb{R}^d) \to T(\mathbb{R}^d)$ *defined, for any* $a \in \widetilde{T}(\mathbb{R}^d)$*, by*

$$\log(a) = \sum_{n \geq 1} \frac{(-1)^n}{n} (\mathbf{1} - a)^{\otimes n}.$$

Let $T_0(\mathbb{R}^d) = \{a \in T(\mathbb{R}^d) \,|\, \pi_0(a) = 0\}$ be the subset of elements in $T(\mathbb{R}^d)$ whose first term is 0. Then $\exp : T_0(\mathbb{R}^d) \to \widetilde{T}(\mathbb{R}^d)$ and $\log : \widetilde{T}(\mathbb{R}^d) \to T_0(\mathbb{R}^d)$ are inverses of each other. The logsignature is then defined as the logarithm of the signature in the tensor algebra and denoted by $\log(S(X))$. For example, its terms of order 2 are equal to

$$\log(S(X))^{(i,j)} = S^{(i,j)}(X) - S^{(j,i)}(X),$$

which corresponds exactly to the Levy area of Figure 1.3b.

Example 1.8. *We can rewrite Example 1.3 in a more compact form. If* X *is a linear path (i.e.,* $X_t = X_0 + (X_1 - X_0)t$*) then*

$$S(X) = \exp(X_1 - X_0), \quad \text{and} \quad \log(S(X)) = X_1 - X_0.$$

Definition 1.13 (Lie formal series)**.** *We endow the tensor algebra with a Lie bracket : for any* $a, b \in T(\mathbb{R}^d)$*, we let*

$$[a, b] = a \otimes b - b \otimes a.$$

Then, if F_1 *and* F_2 *are linear subspaces of* $T(\mathbb{R}^d)$ *we denote by* $[F_1, F_2]$ *the linear span of all elements* $[a, b]$ *such that* $a \in F_1$ *and* $b \in F_2$*. Then, we define recursively* $L_0 = 0$*,* $L_1 = \mathbb{R}^d$*,* $L_2 = [\mathbb{R}^d, L_1] = [\mathbb{R}^d, \mathbb{R}^d]$*,* $L_3 = [\mathbb{R}^d, L_2] = [\mathbb{R}^d, [\mathbb{R}^d, \mathbb{R}^d]]$ *and for any* $n > 0$ $L_{n+1} = [\mathbb{R}^d, L_n]$*.* L_n *is a linear subspace of* $(\mathbb{R}^d)^{\otimes n}$ *and is called the space of homogeneous Lie polynomials of degree* n*. The space of Lie formal series over* $\mathbb{R}^d$ *is defined by*

$$\mathscr{L}(\mathbb{R}^d) = \{\ell = (\ell_0, \ell_1, \dots, \ell_n, \dots) \,|\, \forall n \geq 0,\, \ell_n \in L_n\},$$

and its truncated counterpart by

$$\mathscr{L}^N(\mathbb{R}^d) = \{\ell = (\ell_0, \ell_1, \dots, \ell_N) \,|\, \forall n \in \{0, \dots, N\},\, \ell_n \in L_n\}.$$

The following theorem states that an element is group-like if and only if it is the exponential of an element of $\mathscr{L}(\mathbb{R}^d)$.

Theorem 1.15. *For any* $a \in \widetilde{T}(\mathbb{R}^d)$*,* $a \in G^{(*)}(\mathbb{R}^d) \Leftrightarrow \log(a) \in \mathscr{L}(\mathbb{R}^d)$*.*

Therefore, for any path $X \in BV(\mathbb{R}^d)$, $\log(S(X)) \in \mathscr{L}(\mathbb{R}^d)$. Note that several basis of $\mathscr{L}(\mathbb{R}^d)$ exist, the usual one being the Lyndon basis (Reizenstein, 2017). The dimension of $\mathscr{L}^N(\mathbb{R}^d)$ is equal to

$$p(N) = \frac{1}{N} \sum_{q|N} \mu\Big(\frac{N}{q}\Big) d^q,$$

where the sum is over all possible divisors q of N and μ is the Möbius function—see Reutenauer

(2003). Some values of $p(N)$ are given in Table 1.1, compared to the dimension of the tensor space $T^N(\mathbb{R}^d)$.

Table 1.1 – Dimension of $T^N(\mathbb{R}^d)$ and $\mathscr{L}^N(\mathbb{R}^d)$

	$d=2$		$d=3$		$d=6$	
	$T^N(\mathbb{R}^d)$	$\mathscr{L}^N(\mathbb{R}^d)$	$T^N(\mathbb{R}^d)$	$\mathscr{L}^N(\mathbb{R}^d)$	$T^N(\mathbb{R}^d)$	$\mathscr{L}^N(\mathbb{R}^d)$
$N=1$	2	2	3	3	6	6
$N=2$	6	3	12	6	42	21
$N=5$	62	14	363	80	9330	1960
$N=7$	254	41	3279	508	335922	49685

Table 1.1 illustrates that the logsignature is particularly useful in a learning context to reduce the dimension. Indeed, we have seen that signatures do not live in the whole space $T^N(\mathbb{R}^d)$ but in the much smaller space of group-like elements, $G^{(N)}(\mathbb{R}^d)$. However, $G^{(N)}(\mathbb{R}^d)$ is not a vector space, so we typically represent signatures on a basis of $T^{(N)}(\mathbb{R}^d)$, of size $\mathscr{O}(d^N)$. On the other hand, the logsignature lives in a Lie algebra, therefore in a vector space, of much smaller dimension than $T^{(N)}(\mathbb{R}^d)$. Given that there is an isomorphism between signatures and logsignatures, using the logsignature as a feature set is a reasonable choice. Must properties of the signature transfer to the logsignature. However, the linear approximation theorem (Theorem 1.13) is not true for logsignatures: there is a trade-off between having a compact representation and good approximation properties.

1.3 Contributions

From a learning perspective, we have seen in the last section several properties of signatures that make it a relevant feature set. First, we know exactly which information is lost when summarizing a path by its signature (Theorem 1.6). Second, signatures linearize functions, in the sense that continuous functions of paths can be approximated arbitrarily well by a linear function on the signature (Theorem 1.13). Finally, signatures can be computed efficiently in practice, with several libraries available in Python (Reizenstein and Graham, 2020; Kidger and Lyons, 2020).

The general procedure can be summarized by the following pipeline:

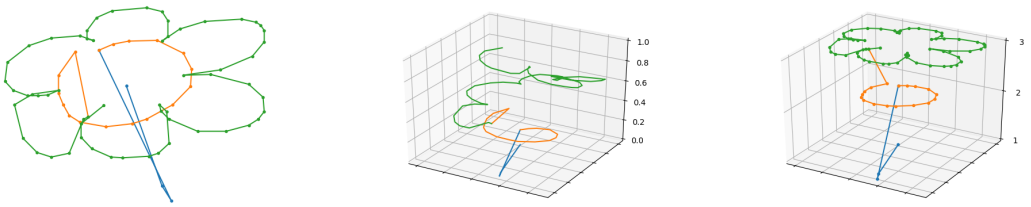
$$\text{Raw data} \to \text{Path} \to \text{Signatures} \to \text{Learning algorithm}. \tag{1.5}$$

This general pipeline can be adapted to the problem at hand, and raises many applied and theoretical questions. We detail below our contributions to leveraging the theory of signatures to statistics and machine learning.

Signatures as a feature set

The general procedure (1.5) is very flexible. First, from a raw input, there are many choices of representation as a continuous path (of bounded variation). Next, from a path in $BV(\mathbb{R}^d)$, signatures can be computed on any subinterval $[s,t] \subset [0,1]$. Finally, the choice of algorithm is open although slightly dependent on the type of signature features constructed. There is no consensus in the literature on which option among these many choices performs best for practical applications.

Our first contribution is to investigate the choice of representation of the raw data as a continuous path. Such a choice is called an *embedding* or an *augmentation*. We have seen for example that it is possible to add the time as a new coordinate to the path, which removes the parametrization invariance and ensures uniqueness of signatures (Lemmas 1.5 and 1.7). However this is only one choice among others; some examples are shown in Figure 1.6.



(a) Linear path (b) Time path (c) Stroke path, version 1

Figure 1.6 – Different possible embeddings of a sample from the Quick, Draw! dataset (2017). Each stroke is plotted with a different color.

We show in Chapter 2 that this choice drastically impacts the performance of the whole methodology, independently of the dataset or the algorithm. In particular, the *lead-lag* and *time* embeddings have a very good performance.

We continue in Chapter 3 with a systematic study of the different variations of signature features used in the literature. In addition to the choice of augmentation, we compare the choice between signatures and logsignatures, and the choice of the interval on which to compute signatures. We perform an extensive empirical study on 26 datasets of time series classification and show in particular that dyadic partitions perform very well, together with invariance-removing augmentations. This leads us to the definition of a canonical signature pipeline, summarized in Figure 1.7, which is shown to be a good domain-agnostic starting point for practitioners. We show that this pipeline, combined with a random forest classifier, is competitive with state-of-the-art algorithms for time series classification, including deep neural networks and ensemble methods.

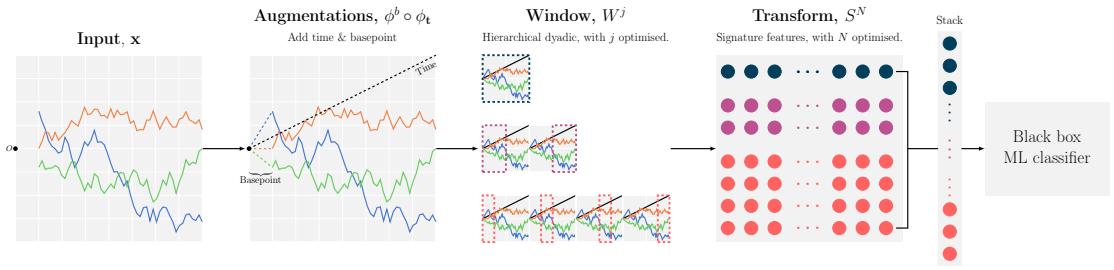


Figure 1.7 – Pictorial representation of the canonical signature pipeline. First, we apply the time and basepoint augmentations to the input paths, and then we compute the signature features over dyadic windows. These features can now be compiled together and fed into any standard machine learning classifier.

Linear regression with signatures

We move on in Chapter 4 to the problem of estimation in a linear model on signatures. Indeed, Theorem 1.13 motivates the definition of the following model. Let $(X, Y) \in BV(\mathbb{R}^d) \times \mathbb{R}$ be a random sample, then we assume that there exists $m^* \in \mathbb{N}$, $\beta^*_{m^*} \in \mathbb{R}^{(d^{m^*}-1)/(d-1)}$, such that

$$\mathbb{E}[Y|X] = \langle \beta^*_{m^*}, S^{m^*}(X) \rangle \quad \text{and} \quad \mathrm{Var}(Y|X) \leq \sigma^2 < \infty.$$

There are two unknown quantities in this model: the truncation parameter m^* and the coefficients vector $\beta^*_{m^*}$. We define estimators of these two quantities, $\widehat{m}$ and $\widehat{\beta}$, and upper bound their rate of convergence. Our main result is stated informally below, where n denotes the sample size.

Theorem 1.16. *Let $0 < \rho < \frac{1}{2}$. For any $n \geq n_0$,*

$$\mathbb{P}(\widehat{m} \neq m^*) \leq C_1 \exp\left(-C_2 n^{1-2\rho}\right),$$

where n_0, C_1 and C_2 are explicit constants. We then have

$$\mathbb{E}\Big(\langle \hat{\beta}_{\widehat{m}}, S^{\widehat{m}}(X) \rangle - \langle \beta^*_{m^*}, S^{m^*}(X) \rangle\Big)^2 = \mathcal{O}\Big(n^{-1/2}\Big).$$

We conclude by comparing this regression model to traditional functional linear models and show that it performs better in high dimensions (that is, when d is large).

Signature kernel and RNN

Taking a slightly different perspective, in Chapter 5, we use signatures as a theoretical tool for analyzing recurrent neural networks (RNN). The neural ODE paradigm, first formulated for residual neural networks by Chen et al. (2018), shows that some networks can be seen as discretizations of ordinary differential equations. More precisely, consider a recurrent neural network of the form

$$h_{j+1} = h_j + \frac{1}{T} f(h_j, x_{j+1}), \quad h_0 = 0,$$

where $\mathbf{x} = (x_1, \ldots, x_T) \in (\mathbb{R}^d)^T$ is a sequential input, $(h_1, \ldots, h_T) \in (\mathbb{R}^e)^T$ are the hidden states, and $f : \mathbb{R}^e \to \mathbb{R}^e$ is an arbitratry smooth function. At each time step $1 \leq j \leq T$, the output of the network is $z_j = \psi(h_j)$, where ψ is a linear function. Then, this network can be approximated by the ordinary differential equation (ODE)

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

By increasing the dimension of the problem, this ODE can be written as a CDE of the form (1.3) and, with arguments similar to Proposition 1.11, we are able to show that the output of the RNN h_T can be rewritten as a linear function on the signature. This frames the RNN into a reproducing kernel Hilbert space (RKHS) $\mathscr{H}$ associated to the signature kernel K defined by (1.2). We have the following informal theorem.

Theorem 1.17. *There exists a function $\xi_\alpha \in \mathscr{H}$ such that*

$$|z_T - \xi_\alpha(\mathbf{x})| \leq \frac{c}{T}.$$

We have reduced the problem of learning the weights of a RNN to a linear problem on the signature. This allows to derive generalization bounds, stability guarantees, and regularization

strategies for RNN. Indeed, the RKHS $\mathscr{H}$ defines naturally a norm on networks, which can be computed in practice and used for regularization. A small norm in the RKHS ensures stability of the network. Indeed, if z_T and z'_T are the outputs associated to two inputs $\mathbf{x}$ and $\mathbf{x}'$, we have

$$\|z_T - z'_T\| \leq \frac{2c}{T} + \|\xi_\alpha(\mathbf{x}) - \xi_\alpha(\mathbf{x}')\| \leq \frac{2c}{T} + \|\xi_\alpha\|_{\mathscr{H}} \|S(\overline{X}) - S(\overline{X}')\|_{\mathscr{T}},$$

where $\overline{X}$ and $\overline{X}'$ are two embeddings of the inputs $\mathbf{x}$ and $\mathbf{x}'$, and $\mathscr{T}$ is a subspace of $T(\mathbb{R}^d)$.

Signature inversion

We conclude in Chapter 6 with contributions to the problem of inverting the signature. Given Theorem 1.6, we can hope to reconstruct a path X from its signature $S(X)$, up to translations, reparametrizations, and tree-like pieces. We are interested in the insertion algorithm, first proposed by Chang and Lyons (2019), which focuses on the inversion of signatures of piecewise linear paths. We propose a slightly different version of this algorithm and provide a rate of convergence for our version. We implement the algorithm in the open-source package Signatory (Kidger and Lyons, 2020). The algorithm is based on the definition of a linear operator, the *insertion operator* denoted by $\mathscr{L}^n_{p,X}$, $p \in \{1, \dots, n+1\}$. Given a signature truncated at order $n+1$, we recover $n+1$ derivatives of X by solving the optimization problem

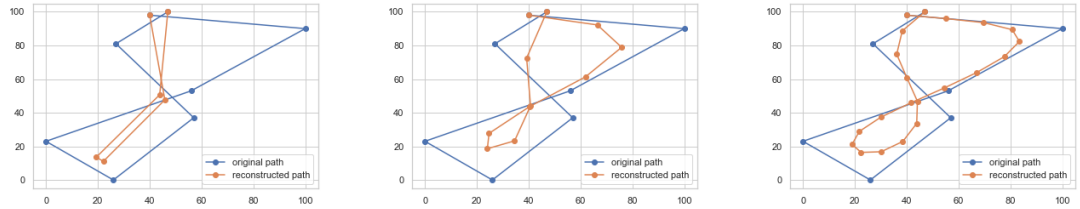
$$\min_{y \in \mathbb{R}^d} \|\mathscr{L}^n_{p,X}(y) - (n+1)\mathbf{X}^{\mathbf{n+1}}\|. \tag{1.6}$$

More precisely, let $X : [0,1] \to \mathbb{R}^d$ be a piecewise linear path and $0 = t_0 < \cdots < t_M = 1$ be the minimal partition such that X is linear on each $[t_{i-1}, t_i]$: there exists $\alpha_1, \dots, \alpha_M, \beta_1, \dots, \beta_M \in \mathbb{R}^d$ such that

$$X_t = \alpha_i + \beta_i t, \quad \text{for } t \in [t_{i-1}, t_i], \quad i \in \{1, \dots, M\}.$$

We have the following informal result, illustrated in Figure 1.8.

Theorem 1.18. *For any $1 \leq i \leq M$, $n \geq 0$, and $p = \lfloor (3t_i + t_{i-1})(n+1)/4 \rfloor$, let $y^*_{p,n}$ be the solution of (1.6). Then, there exists a strictly increasing sequence $(k_n)_{n \geq 0}$ such that y^*_{p,k_n} converges to β_i as n increases.*



(a) $n = 5$ (b) $n = 10$ (c) $n = 20$

Figure 1.8 – Signature inversion of a sample of the class "8" from the Pendigits dataset

Outline of the manuscript

Each chapter is independent and self-contained. Therefore, the notation may vary from chapter to chapter. The proofs are systematically given in the associated appendices. We summarize

the contents of the chapters below.

- Chapter 2 has been published in *Computational Statistics and Data Analysis*, volume 157.
- Chapter 3 is a joint work with James Morrill (University of Oxford), Patrick Kidger (University of Oxford), and Terry Lyons (University of Oxford). It has been submitted for publication.
- Chapter 4 has been submitted for publication.
- Chapter 5 is a joint work with Pierre Marion (Sorbonne Université), Jean-Philippe Vert (Google Research), and Gérard Biau (Sorbonne Université). It has been submitted for publication.
- Chapter 6 is an ongoing joint work with Terry Lyons (University of Oxford).

Résumé détaillé

Le but de cette thèse est d'étudier l'application des signatures aux statistiques et à l'apprentissage automatique. Nous nous intéressons aux données séquentielles ou temporelles, c'est-à-dire qui possèdent un ordre. De telles données peuvent être considérées comme des fonctions (continues) X d'un intervalle $[a, b] \subset \mathbb{R}$ dans $\mathbb{R}^d$, $d \geq 1$. Par exemple, X peut être une série temporelle représentant le prix d'une action observé pendant un certain intervalle de temps. Si plusieurs actions sont observées, X est alors une série temporelle multivariée et est à valeurs dans $\mathbb{R}^d$ avec $d > 1$. Les données fonctionnelles entrent également dans ce cadre, avec des jeux de données classiques tels que des courbes spectrométriques ou des profils de température. En apprentissage automatique, les domaines de la reconnaissance vocale, de la reconnaissance de caractères ou du traitement du langage naturel s'inscrivent également dans ce cadre. Dans ce cas, l'intervalle $[a, b]$ peut correspondre au temps mais aussi à une position dans une phrase. Prenant un point de vue géométrique, nous appelerons ce type de données des chemins $X : [a, b] \to \mathbb{R}^d$.

Ayant des chemins X disponibles, nous voulons apprendre une certaine sortie Y, fonction de X. Par exemple, la sortie Y peut être une prévision de la valeur future de la série, une autre quantité scalaire liée, ou en classification un label. Apprendre Y à partir de X soulève la question de la représentation de X. En effet, la plupart des algorithmes d'apprentissage, tels que la régression des moindres carrés, les arbres de décision ou les réseaux de neurones, prennent en entrée un vecteur. Un choix naturel pour ce vecteur est d'échantillonner les valeurs de X mais il existe d'autres options comme l'utilisation d'une représentation de Fourier de X ou, plus généralement, la projection de X sur une base de fonctions. Nous allons voir que les signatures sont encore une autre option.

La signature a été définie pour la première fois par Chen dans les années 60 (Chen, 1957 ; 1958 ; 1977) et a été redécouverte dans les années 90 dans le contexte de la théorie des chemins rugueux (Lyons, 1998 ; Lyons et al., 2007 ; Friz et Victoir, 2010 ; Friz et Hairer, 2020), domaine important de l'analyse stochastique. Du point de vue des équations différentielles, la signature résume l'effet d'un chemin X sur un chemin de sortie Y lorsque Y est "contrôlé" par X. Plus précisément, si $F : \mathbb{R}^e \to \mathbb{R}^{e \times d}$ est un champ de vecteurs, et Y est la solution de l'équation différentielle contrôlée

$$dY_t = F(Y_t)dX_t, \quad Y_0 = y_0,$$

alors Y est entièrement défini par F et par la signature de X. En d'autres termes, Y ne dépend de X qu'à travers sa signature. Dans une perspective d'apprentissage, Y peut être considéré comme une quantité à prédire, par exemple la traduction d'un phrase, et F est maintenant une fonction inconnue qui relie l'entrée X et la sortie Y. La théorie des équations différentielles nous dit alors que Y ne dépendra que de la signature de X, ce qui en fait une représentation pertinente pour tout algorithme qui tente d'apprendre F.

Au cours des dernières années, la signature a été combinée avec succès à des algorithmes d'apprentissage automatique dans plusieurs domaines tels que la reconnaissance de caractères (Yang et al., 2016a), la finance (Perez Arribas, 2018), la médecine (Morrill et al., 2019) ou la vision assistée par ordinateur (Yang et al., 2017). Plusieurs travaux méthodologiques ont accompagné ces applications, créant un pont entre l'analyse stochastique et l'apprentissage automatique (Levin et al., 2013 ; Király et Oberhauser, 2019 ; Liao et al., 2019 ; Kidger et al., 2019). De manière générale, on peut résumer l'utilisation de la signature en apprentissage par le schéma suivant :

$$\text{Données brutes} \to \text{Chemin} \to \text{Signatures} \to \text{Algorithme d'apprentissage.}$$

Cette méthodologie générale peut être adaptée au problème à résoudre, et soulève de nombreuses questions statistiques et méthodologiques.

Les signatures comme ensemble de features. La procédure générale ci-dessus est très flexible. Tout d'abord, il existe de nombreux choix de représentation des données brutes en un chemin continu. De plus, à partir d'un chemin, les signatures peuvent être calculées sur tout sous-intervalle $[s, t] \subset [0, 1]$ et enfin le choix de l'algorithme est libre. Il n'y a pas de consensus dans la littérature sur quelles options sont les plus performantes en pratique. Nos premières contributions ont tenté de répondre à cette question.

Tout d'abord, nous nous intéressons aux différentes représentations des données brutes en chemin continu. Un tel choix est appelé *embedding* ou *augmentation*. Nous montrons dans le chapitre 2 que ce choix a un impact considérable sur la performance de l'ensemble de la méthodologie, indépendamment du type de données ou de l'algorithme. Nous poursuivons dans le chapitre 3 par une étude systématique des différentes variations utilisées dans la littérature. Nous réalisons une étude complète de la performance de ces variations sur 26 jeux de données de classification de séries temporelles. Cette étude conduit à la définition d'un algorithme canonique qui s'avère être un bon algorithme de référence pour utiliser en pratique les signatures. En effet, cet algorithme est compétitif par rapport à l'état de l'art en classification de séries temporelles, en particulier par rapport aux réseaux de neurones profonds et aux méthodes d'aggrégation.

Régression linéaire sur les signatures. Nous poursuivons dans le chapitre 4 par une étude d'un modèle linéaire sur signatures. Les signatures étant un objet infini, il est nécessaire de les tronquer à un certain ordre. Cet ordre de troncature contrôle la taille du modèle et est donc une quantité cruciale à estimer. Nous proposons un estimateur de cet ordre de troncature, ainsi que du vecteur de coefficients, avec des garanties théoriques. Nous concluons en comparant ce modèle de régression aux modèles linéaires fonctionnels traditionnels et observons qu'il est particulièrement performant en grande dimension.

Méthodes à noyaux, signatures et RNN. En adoptant une perspective légèrement différente, nous utilisons dans le chapitre 5 les signatures comme outil théorique pour analyser les réseaux de neurones récurrents (RNN). Le paradigme des *neural ODE*, formulé pour les réseaux résiduels par Chen et al. (2018), montre que certains réseaux peuvent être considérés comme des discrétisations d'équations différentielles ordinaires. En tirant parti des bonnes propriétés analytiques des signatures, nous montrons que la sortie d'un RNN peut alors être réécrite comme une fonction dans un espace de Hilbert à noyau reproduisant (RKHS). Nous avons ainsi réécrit le problème d'apprentissage d'un RNN comme une méthode à noyau, ce qui permet de dériver des bornes de généralisation, des garanties de stabilité et des stratégies de régularisation.

Inversion de la signature. Nous concluons dans le chapitre 6 par une contribution au problème d'inversion de la signature. Nous nous intéressons à l'algorithme d'insertion, proposé pour la première fois par Chang et Lyons (2019), qui se concentre sur l'inversion des signatures de chemins linéaires par morceaux. Nous proposons une version légèrement différente de cet algorithme et fournissons des garanties théoriques. Nous implémentons cet algorithme dans la librairie open-source Signatory (Kidger et Lyons, 2020), basée sur Pytorch (Paszke et al., 2019).

Bibliography

Boedihardjo, H., and Geng, X. (2015). The uniqueness of signature problem in the non-markov setting. *Stochastic Processes and their Applications*, *125*, 4674–4701.

Boedihardjo, H., Geng, X., Lyons, T., and Yang, D. (2016). The signature of a rough path: uniqueness. *Advances in Mathematics*, *293*, 720–737.

Chang, J., and Lyons, T. (2019). Insertion algorithm for inverting the signature of a path. *arXiv:1907.08423*.

Chen, K.-T. (1957). Integration of paths, geometric invariants and a generalized baker-hausdorff formula. *Annals of Mathematics*, 163–178.

Chen, K.-T. (1958). Integration of paths—a faithful representation of paths by non-commutative formal power series. *Transactions of the American Mathematical Society*, *89*, 395–407.

Chen, K.-T. (1977). Iterated path integrals. *Bulletin of the American Mathematical Society*, *83*, 831–879.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. (2018). Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 6572–6583). Curran Associates, Inc.

Chevyrev, I., and Lyons, T. (2016). Characteristic functions of measures on geometric rough paths. *The Annals of Probability*, *44*, 4049–4082.

Dua, D., and Graff, C. (2017). UCI machine learning repository. http://archive.ics.uci.edu/ml

Friz, P. K., and Hairer, M. (2020). *A course on rough paths*. Springer.

Friz, P. K., and Victoir, N. B. (2010). *Multidimensional stochastic processes as rough paths: theory and applications* (Vol. 120). Cambridge University Press.

Google. (2017). The quick, draw! dataset. https://github.com/googlecreativelab/quickdraw-dataset

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

Kidger, P., Bonnier, P., Perez Arribas, I., Salvi, C., and Lyons, T. (2019). Deep signature transforms. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 3099–3109). Curran Associates, Inc.

Kidger, P., and Lyons, T. (2020). Signatory: differentiable computations of the signature and logsignature transforms, on both CPU and GPU. *arXiv:2001.00706*. https://github.com/patrick-kidger/signatory

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

Le Jan, Y., and Qian, Z. (2013). Stratonovich's signatures of brownian motion determine brownian sample paths. *Probability Theory and Related Fields*, *157*, 209–223.

Levin, D., Lyons, T., and Ni, H. (2013). Learning from the past, predicting the statistics for the future, learning an evolving system. *arXiv:1309.0260*.

Liao, S., Lyons, T., Yang, W., and Ni, H. (2019). Learning stochastic differential equations using RNN with log signature features. *arXiv:1908.08286*.

Lyons, T., Caruana, M., and Lévy, T. (2007). *Differential equations driven by rough paths* (Vol. 1908). Springer.

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

Morrill, J., Kormilitzin, A., Nevado-Holgado, A., Swaminathan, S., Howison, S., and Lyons, T. (2019). The signature-based model for early detection of sepsis from electronic health records in the intensive care unit. *International Conference in Computing in Cardiology*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 8024–8035). Curran Associates, Inc.

Perez Arribas, I. (2018). Derivatives pricing using signature payoffs. *arXiv:1809.09466*.

Purbhoo, K. (2012). Notes on tensor products and the exterior algebra.

Reizenstein, J. (2017). Calculation of iterated-integral signatures and log signatures. *arXiv:1712.02757*.

Reizenstein, J., and Graham, B. (2020). Algorithm 1004: the iisignature library: efficient calculation of iterated-integral signatures and log signatures. *ACM Transactions on Mathematical Software*.

Reutenauer, C. (2003). Free lie algebras. *Handbook of algebra* (pp. 887–903). Elsevier.

Yang, W., Jin, L., and Liu, M. (2016a). DeepWriterID: An end-to-end online text-independent writer identification system. *IEEE Intelligent Systems*, *31*, 45–53.

Yang, W., Lyons, T., Ni, H., Schmid, C., Jin, L., and Chang, J. (2017). Developing the path signature methodology and its application to landmark-based human action recognition. *arXiv:1707.03993*.

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

Chapter 2

Embedding and learning with signatures

Sequential and temporal data arise in many fields of research, such as quantitative finance, medicine, or computer vision. A novel approach for sequential learning, called the signature method and rooted in rough path theory, is considered. Its basic principle is to represent multi-dimensional paths by a graded feature set of their iterated integrals, called the signature. This approach relies critically on an embedding principle, which consists in representing discretely sampled data as paths, i.e., functions from $[0, 1]$ to $\mathbb{R}^d$. After a survey of machine learning methodologies for signatures, the influence of embeddings on prediction accuracy is investigated with an in-depth study of three recent and challenging datasets. It is shown that a specific embedding, called lead-lag, is systematically the strongest performer across all datasets and algorithms considered. Moreover, an empirical study reveals that computing signatures over the whole path domain does not lead to a loss of local information. It is concluded that, with a good embedding, combining signatures with other simple algorithms achieves results competitive with state-of-the-art, domain-specific approaches.

Contents

2.1 Introduction

Sequential or temporal data are arising in many fields of research, due to an increase in storage capacity and to the rise of machine learning techniques. An illustration of this vitality is the recent relaunch of the Time Series Classification repository (Bagnall et al., 2018), with more than a hundred new datasets. Sequential data are characterized by the fact that each sample consists of an ordered array of values. The order need not correspond to time, for example, text documents or DNA sequences have an intrinsic ordering, and are, therefore, considered as sequential. Besides, when time is involved, several values can be recorded simultaneously, giving rise to an ordered array of vectors, which is, in the field of time series, often referred to as multidimensional time series. To name only a few domains, market evolution is described by financial time series, and physiological variables (e.g., electrocardiograms, electroencephalograms) are recorded simultaneously in medicine, yielding multidimensional time series. Finally, smartphone and GPS sensors data, or character recognition problems, present both spatial and temporal aspects. These high-dimensional datasets open up new theoretical and practical challenges, as both algorithms and statistical methods need to be adapted to their sequential nature.

Different communities have addressed this problem. First, time series forecasting has been an active area of research in statistics since the 1950s, resulting in several monographs, such as Hamilton (1994), Box et al. (2015) and Shumway and Stoffer (2017), to which the reader is referred for overviews of the domain. Time series are considered as realizations of various stochastic processes, such as the famous ARIMA models. Much work in this field has been done on parameter estimation and model selection. These models have been developed for univariate time series but have been extended to the multivariate case (Lütkepohl, 2005), with the limitation that they become more complicated and harder to fit.

More recently, the field of functional data analysis has extended traditional statistical methods, in particular regression and Principal Component Analysis, to functional inputs. Ramsay and Silverman (2005) and Ferraty and Vieu (2006) provide introductions to the area. Kokoszka et al. (2017) give an account of recent advances. In particular, longitudinal functional data analysis is concerned with the analysis of repeated observations, where each observation is a function (Greven et al., 2011; Park and Staicu, 2015). The data arising from this setting may be considered as a set of vector-valued functions with correlated coordinates, each function corresponding to one subject and each coordinate corresponding to one specific observation.

Although these various disciplines work with sequential data, their goals usually differ. Typically, time series analysis is concerned with predicting future values of one observed function, whereas (longitudinal) functional data analysis usually collects several functions and is then concerned with the prediction of another response variable. However, all these methods rely on strong assumptions on the regularity of the data and need to be adapted to each specific application. Therefore, modern datasets have highlighted their limitations: a lot of choices, in basis functions or model parameters, need to be handcrafted and are valid only on a small-time range. Moreover, these techniques struggle to model multidimensional series, in particular, to incorporate information about interactions between various dimensions.

On the other side, time series classification has attracted the interest of the data mining community. A broad range of algorithms have been developed, reviewed by Bagnall et al. (2017) in the univariate case. Much attention has been paid to the development of similarity measures adapted to temporal data, a popular baseline being the Dynamic Time Warping metric (Berndt and Clifford, 1996), combined with a 1-nearest neighbor algorithm. Bagnall et al. (2017) state that this baseline is beaten only by ensemble strategies, which combine different feature mappings. However, a great limitation of these methods is their complexity, as they have difficulty handling large time series. Recently, deep learning seems to be a promising approach and solves

some problems mentioned above. For example, Fawaz et al. (2019) claim that some architectures perform systematically better than previous data mining algorithms. However, deep learning methods are costly in memory and computing power, and often require a lot of training data.

The present article is concerned with a novel approach for sequential learning, called the signature method, and coming from rough path theory. Its main idea is to summarize temporal or sequential inputs by the graded feature set of their iterated integrals, the signature. In rough path theory, functions are referred to as paths, to emphasize their geometrical aspects. Indeed, the importance of iterated integrals had been noticed by geometers in the 60s, as presented in the work of Chen (1958). It has been rediscovered by Lyons (1998) in the context of stochastic analysis and controlled differential equations, and is at the heart of rough path theory. This theory, of which Lyons et al. (2007) and Friz and Victoir (2010) give a recent account, focuses on developing a new notion of paths to make sense of evolving irregular systems. Notably, Hairer (2013) was awarded a Fields medal in 2014 for its solution to the Kardar-Parisi-Zhang equation built with rough path theory. In this context, it has been shown that the signature provides an accurate summary of a (smooth) path and allows to obtain arbitrarily good linear approximations of continuous functions of paths. Therefore, to learn an output $Y \in \mathbb{R}$, which is an unknown function of a random path $X : [0, 1] \to \mathbb{R}^d$, rough path theory suggests that the signature is a relevant tool to describe X.

The signature has recently received the attention of the machine learning community and has achieved a series of successful applications. To cite some of them, Yang et al. (2016a) have achieved state-of-the-art results for handwriting recognition with a recurrent neural network combined with signature features. Graham (2013) used the same approach for character recognition, and Gyurkó et al. (2014) coupled Lasso with signature features for financial data streams classification. Kormilitzin et al. (2016) investigated its use for the detection of bipolar disorders, and Yang et al. (2017) for human action recognition. For introductions to the signature method in machine learning, the reader is referred to the work of Levin et al. (2013) and to Chevyrev and Kormilitzin (2016b).

However, despite many promising empirical successes, a lot of questions remain open, both practical and theoretical. In particular, to compute signatures, it is necessary to embed discretely sampled data points into paths. While authors use different approaches, this embedding is only mentioned in some articles, and rarely discussed. Thus, the purpose of this paper is to take a step forward in understanding how signature features should be constructed for machine learning tasks, with a special focus on the embedding step. The article is organized as follows.

(*i*) In Section 2.2, a brief exposition of the signature definition and properties is given, along with a survey of different approaches undertaken in the literature to combine signatures with machine learning algorithms. Datasets used throughout the paper are also presented in Section 2.3.

(*ii*) In Section 2.4, potential embeddings are reviewed and their predictive performance is compared with an empirical study on 3 real-world datasets. This study indicates that the embedding is a step as crucial as the algorithm choice since it can drastically impact accuracy results. In particular, we find that the lead-lag embedding systematically outperforms other embeddings, consistently over different datasets and algorithms. This finding is reinforced by a simulation study with autoregressive processes in Section 2.5.

(*iii*) In Section 2.6 the choice of signature domain is investigated. Signatures can be computed on any sub-interval of the path definition domain, and it is natural to wonder whether some local information is lost when signatures of the whole path are computed. The section ends by showing that, with a good embedding, the signature combined with a simple algorithm, such as a random forest classifier, obtains results comparable to state-of-the-art approaches

in different application areas, while remaining a generic approach and computationally simple.

(*iv*) In Section 2.7 some open questions for future work are discussed.

These empirical results are based on three recent datasets, in different fields of application. One is a univariate sound recording dataset, called Urban Sound (Salamon et al., 2014), whereas the others are multivariate. One has been made available by Google (2017), and consists of drawing trajectories, while the other is made up of 12 channels recorded from smartphone sensors (Malekzadeh et al., 2018). They are each of a different nature and present a variety of lengths, noise levels, and dimensions. In this way, generic and domain-agnostic results are obtained. The code is available at https://github.com/afermanian/embedding_with_signatures.

2.2 A first glimpse of the signature method

2.2.1 Definition and main properties

In this subsection, the notion of signature is introduced and some of its important properties are reviewed. The reader is referred to Lyons et al. (2007) or Friz and Victoir (2010) for a more involved mathematical treatment with proofs. Throughout the article, the basic objects are paths, that is, functions from $[0,1] \to \mathbb{R}^d$, where $d \in \mathbb{N}^*$. The main assumption is that these paths are of bounded variation, i.e., they have finite length.

Definition 2.1. *Let*

$$\begin{aligned} X : [0,1] &\longrightarrow \mathbb{R}^d \\ t &\longmapsto (X_t^1, \dots, X_t^d). \end{aligned}$$

The total variation of X is defined by

$$\|X\|_{1-var} = \sup_D \sum_{t_i \in D} \|X_{t_i} - X_{t_{i-1}}\|,$$

where the supremum is taken over all finite partitions

$$D = \left\{(t_0, \dots, t_k) \,|\, k \geq 1,\, 0 = t_0 < t_1 < \cdots < t_{k-1} < t_k = 1\right\}$$

of $[0,1]$, and $\|\cdot\|$ denotes the Euclidean norm on $\mathbb{R}^d$. The path X is said to be of bounded variation if its total variation is finite.

The set of bounded variation paths is exactly the set of functions whose first derivatives exist almost everywhere. Being of bounded variation is therefore not a particularly restrictive assumption. It contains, for example, all Lipschitz functions. In particular, if X is continuously differentiable, and $\dot{X}$ denotes its first derivative with respect to t, then

$$\|X\|_{1-var} = \int_0^1 \|\dot{X}_t\| dt.$$

The assumption of bounded variation allows us to define Riemann-Stieljes integrals along paths. An exposition of this integration theory is not given here, but the interested reader is referred to Lyons et al. (2007). From now on, it is assumed that the integral of a continuous

path $Y : [0,1] \to \mathbb{R}^d$ against a path of bounded variation $X : [0,1] \to \mathbb{R}^d$ is well-defined on any $[s,t] \subset [0,1]$, and denoted by

$$\int_s^t Y_u dX_u = \begin{pmatrix} \int_s^t Y_u^1 dX_u^1 \\ \vdots \\ \int_s^t Y_u^d dX_u^d \end{pmatrix} \in \mathbb{R}^d,$$

where $X = (X^1, \dots, X^d)$, and $Y = (Y^1, \dots, Y^d)$. When X is continuously differentiable, this integral is equal to the standard Riemann integral, that is,

$$\int_s^t Y_u dX_u = \int_s^t Y_u \dot{X}_u du.$$

As an example, assume that X is linear, i.e.,

$$X_t = (X_t^1, \dots, X_t^d) = (a_1 + b_1 t, \dots, a_d + b_d t), \quad 0 \leq t \leq 1, \tag{2.1}$$

where $a_1, \dots, a_d, b_1, \dots b_d \in \mathbb{R}$. Then

$$\int_s^t dX_u = \int_s^t \dot{X}_u du = \begin{pmatrix} \int_s^t b_1 du \\ \vdots \\ \int_s^t b_d du \end{pmatrix} = \begin{pmatrix} b_1(t-s) \\ \vdots \\ b_d(t-s) \end{pmatrix}.$$

The formula above is useful since in practice only integrals of linear paths are computed, as discussed later in this subsection. It is now possible to define the signature.

Definition 2.2. *Let $X : [0,1] \to R^d$ be a path of bounded variation, $I = (i_1, \dots, i_k) \subset \{1, \dots d\}^k$, $k \in \mathbb{N}^*$, be a multi-index of length k, and $[s,t] \subset [0,1]$ be an interval. The signature coefficient of X corresponding to the index I on $[s,t]$ is defined by*

$$S^I(X)_{[s,t]} = \int\limits_{s \leq u_1 < \dots < u_k \leq t} \cdots \int dX_{u_1}^{i_1} \dots dX_{u_k}^{i_k} = \int_s^t \Big(\int_{u_1}^t \Big(\int_{u_2}^t \dots \int_{u_{k-1}}^t dX_{u_k}^{i_k} \Big) dX_{u_2}^{i_2} \Big) dX_{u_1}^{i_1}. \tag{2.2}$$

$S^I(X)_{[s,t]}$ is then said to be a signature coefficient of order k.

The signature of X is the sequence containing all signature coefficients, i.e.,

$$S(X)_{[s,t]} = \big(1, S^{(1)}(X)_{[s,t]}, \dots, S^{(d)}(X)_{[s,t]}, S^{(1,1)}(X)_{[s,t]}, \dots, S^{(i_1,\dots,i_k)}(X)_{[s,t]}, \dots\big).$$

The signature of X truncated at order K, denoted by $S_K(X)$, is the sequence containing all signature coefficients of order lower than or equal to K, that is

$$S_K(X)_{[s,t]} = \big(1, S^{(1)}(X)_{[s,t]}, S^{(2)}(X)_{[s,t]}, \dots, S^{\overbrace{(d, \dots, d)}^{K}}(X)_{[s,t]}\big).$$

For simplicity, when $[s,t] = [0,1]$, the interval is omitted in the notations, and, e.g., $S_K(X)$ is written instead of $S_K(X)_{[0,1]}$.

From these definitions, it follows that the linear interpolation of a (multivariate) time series observed on a finite time horizon will be of bounded variation, and therefore that its signature is well defined. Note that this procedure of mapping a discrete time series into a continuous path

is called an embedding, and linear interpolation is only one embedding among others, which will be studied in Section 2.4. For example, Brownian motion is not of bounded variation but is instead of finite p-variation for any $p > 2$. However, its signature can still be defined with Itô or Stratonovitch integrals.

Before giving an explicit calculation of signatures, some comments are in order. First, it is worth noting that, for a path in $\mathbb{R}^d$, there are d^k coefficients of order k. The signature truncated at order K is therefore a vector of dimension

$$\sum_{k=0}^{K} d^k = \frac{d^{K+1} - 1}{d - 1} \quad \text{if } d \neq 1, \tag{2.3}$$

and $K + 1$ if $d = 1$. Unless otherwise stated, it is assumed that $d \neq 1$, as this is in practice usually the case. Thus, the size of $S_K(X)$ increases exponentially with K, and polynomially with d—some typical values are presented in Table 2.1.

	$d = 2$	$d = 3$	$d = 6$
$K = 1$	2	3	6
$K = 2$	6	12	42
$K = 5$	62	363	9330
$K = 7$	254	3279	335922

Table 2.1 – Typical sizes of $S_K(X)$ for different values of K and d, where $X : [0, 1] \to \mathbb{R}^d$.

Moreover, the set of coefficients of order k can be seen as an element of the kth tensor product of $\mathbb{R}^d$ with itself, denoted by $(\mathbb{R}^d)^{\otimes k}$. For example, the d coefficients of order 1 can be written as a vector, and the d^2 coefficients of order 2 as a matrix:

$$\begin{pmatrix} S^{(1)}(X) \\ \vdots \\ S^{(d)}(X) \end{pmatrix} \in \mathbb{R}^d, \quad \begin{pmatrix} S^{(1,1)}(X) & \dots & S^{(1,d)}(X) \\ \vdots & & \vdots \\ S^{(d,1)}(X) & \dots & S^{(d,d)}(X) \end{pmatrix} \in \mathbb{R}^{d \times d} \approx (\mathbb{R}^d)^{\otimes 2}.$$

Similarly, coefficients of order 3 can be written as a tensor of order 3, and so on. Then, $S(X)$ can be seen as an element of the tensor algebra

$$\mathbb{R} \oplus \mathbb{R}^d \oplus (\mathbb{R}^d)^{\otimes 2} \oplus \cdots \oplus (\mathbb{R}^d)^{\otimes k} \oplus \cdots .$$

This structure of the tensor algebra will not be used in the present article but is used to derive properties of the signature (Lyons, 1998; Friz and Victoir, 2010; Hambly and Lyons, 2010).

It should be noted that due to the ordering in the integration domain in (2.2), the signature coefficients are not symmetric. For example, $S^{(1,2)}(X)$ is not the same as $S^{(2,1)}(X)$. Finally, Chevyrev and Kormilitzin (2016b) show how, under certain assumptions, empirical statistical moments can be explicitly recovered from signature coefficients. Typically, the empirical mean can be recovered from signature coefficients of order 1, the variance from coefficients of order 2, and so on. Therefore, the larger the truncation order, the more detailed the information encoded in the signature.

As a toy example, consider the linear path (2.1) again, and assume for simplicity that $d = 2$:

$$X_t = \begin{pmatrix} X_t^1 \\ X_t^2 \end{pmatrix} = \begin{pmatrix} a_1 + b_1 t \\ a_2 + b_2 t \end{pmatrix}.$$

Then, for any $[s,t] \subset [0,1]$ the signature coefficients of order 1 are

$$S^{(1)}(X)_{[s,t]} = \int_s^t dX_u^1 = b_1(t-s) \quad \text{and} \quad S^{(2)}(X)_{[s,t]} = \int_s^t dX_u^2 = b_2(t-s).$$

The first coefficient of order 2 is

$$S^{(1,1)}(X)_{[s,t]} = \int_s^t \int_{u_1}^t dX_{u_2}^1 dX_{u_1}^1 = \int_s^t \int_{u_1}^t b_1^2 du_2 du_1 = b_1^2 \int_s^t (t-u_1) du_1 = \frac{b_1^2(t-s)^2}{2}.$$

Similarly,

$$S^{(1,2)}(X)_{[s,t]} = S^{(2,1)}(X) = \frac{b_1 b_2 (t-s)^2}{2} \quad \text{and} \quad S^{(2,2)}(X)_{[s,t]} = \frac{b_2^2(t-s)^2}{2}.$$

For any index $I = (i_1, \dots, i_k) \subset \{1,2\}^k$, it is easily obtained that

$$S^{(i_1,\dots,i_k)}(X)_{[s,t]} = \int_{s \le u_1 < \cdots < u_k \le t} \cdots \int dX_{u_1}^{i_1} \dots dX_{u_k}^{i_k} = \frac{b_{i_1} \dots b_{i_k} (t-s)^k}{k!}. \tag{2.4}$$

A crucial feature of the signature is that it encodes geometric properties of the path. Indeed, coefficients of order 2 correspond to some areas outlined by the path, as shown in Figure 2.1. For higher orders of truncation, the signature contains information about the joint evolution of tuples of coordinates (Yang et al., 2017). Furthermore, the signature possesses several properties that make it a good statistical summary of paths, as shown in the next four propositions.

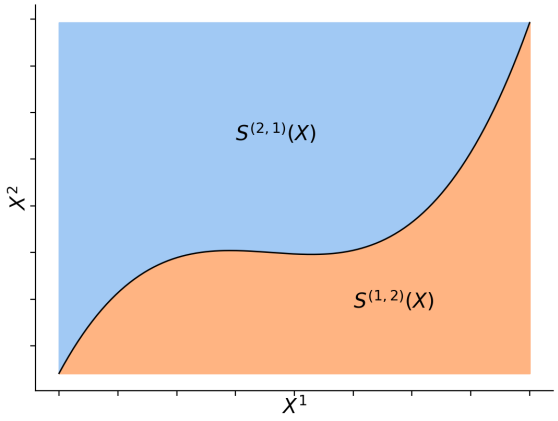


Figure 2.1 – Geometric interpretation of signature coefficients.

Proposition 2.1. *Let $X : [0,1] \to \mathbb{R}^d$ be a path of bounded variation, and $\psi : [0,1] \to [0,1]$ be a non-decreasing surjection. Then, if $\widetilde{X}_t = X_{\psi(t)}$ is the reparametrization of X under ψ,*

$$S(\widetilde{X}) = S(X).$$

This proposition is a consequence of the properties of integrals and bounded variation paths (Friz and Victoir, 2010, Proposition 7.10). In other words, the signature of a path is the same up to any reasonable time change. There is, therefore, no information about the path parametrization in signature coefficients. However, when relevant for the application, it is possible to include this information by adding the time parametrization as a coordinate of the path. This procedure plays a decisive role in the construction of time embeddings, which will be thoroughly discussed in Section 2.4.

A second important property is a condition ensuring the uniqueness of signatures.

Proposition 2.2. *If X has at least one monotone coordinate, then $S(X)$ determines X uniquely up to translations.*

It should be noticed that having a monotone coordinate is a sufficient condition, but a necessary one can be found in Hambly and Lyons (2010), together with a proof of this proposition. The principal significance of this result is that it provides a practical procedure to guarantee signature uniqueness: it is sufficient to add a monotone coordinate to the path X. For example, the time embedding mentioned above will satisfy this condition.

This result does not provide a practical procedure to reconstruct a path from its signature. However, this is an active area of research (Chang et al., 2017; Lyons and Xu, 2017; Lyons and Xu, 2018). In particular, (Lyons and Xu, 2017) derive an explicit expression of rectilinear paths, defined in Section 2.4.1, in terms of their signatures; and (Lyons and Xu, 2018) construct, from the signature of a $\mathcal{C}^1$ path, a sequence of piecewise linear approximations converging to the initial path.

The next proposition reveals that the signature linearizes functions of X. We refer the reader to Király and Oberhauser (2019, Theorem 1) for a proof.

Proposition 2.3. *Let D be a compact subset of the space of bounded variation paths from $[0,1]$ to $\mathbb{R}^d$ and such that for any $X \in D$, $X_0 = 0$ and X has at least one monotone coordinate. Let $f : D \to \mathbb{R}$ be continuous. Then, for every $\varepsilon > 0$, there exists $N \in \mathbb{N}$, $w \in \mathbb{R}^N$, such that, for any $X \in D$,*

$$\big|f(X) - \langle w, S(X)\rangle\big| \leq \varepsilon,$$

where $\langle \cdot, \cdot \rangle$ denotes the Euclidean scalar product on $\mathbb{R}^N$.

This proposition is a consequence of the Stone-Weierstrass theorem. The classical Weierstrass approximation theorem states that every real-valued continuous function on a closed interval can be uniformly approximated by a polynomial function. Similarly, this theorem states that any real-valued continuous function on a compact subset D of bounded variation paths can be uniformly approximated by a linear form on the signature. Linear forms on the signature can, therefore, be thought of as the equivalent of polynomial functions for paths.

Chen's theorem (Chen, 1958) now provides a formula to compute recursively the signature of a concatenation of paths. Let $X : [s,t] \to \mathbb{R}^d$ and $Y : [t,u] \to \mathbb{R}^d$ be two paths, $0 \leq s < t < u \leq 1$. Then, the concatenation of X and Y, denoted by $X * Y$, is defined as the path from $[s,u]$ to $\mathbb{R}^d$ such that, for any $v \in [s,u]$,

$$(X * Y)_v = \begin{cases} X_v & \text{if } v \in [s,t], \\ X_t + Y_v - Y_t & \text{if } v \in [t,u]. \end{cases}$$

Proposition 2.4 (Chen)**.** *Let $X : [s,t] \to \mathbb{R}^d$ and $Y : [t,u] \to \mathbb{R}^d$ be two paths with bounded*

variation. Then, for any multi-index $(i_1, \dots, i_k) \subset \{1, \dots, d\}^k$,

$$S^{(i_1,\dots,i_k)}(X * Y) = \sum_{\ell=0}^{k} S^{(i_1,\dots,i_\ell)}(X) \cdot S^{(i_{\ell+1},\dots,i_k)}(Y). \tag{2.5}$$

This proposition is an immediate consequence of the linearity property of integrals (Lyons et al., 2007, Theorem 2.9). However, it is essential for the explicit calculation of signatures. Indeed, in practice, X is observed at a finite number of times and becomes by interpolation a continuous piecewise linear path. To compute its signature, it is then sufficient to iterate the following two steps:

1. Compute with equation (2.4) the signature of a linear section of the path.
2. Concatenate it to the other pieces with Chen's formula (2.5).

This procedure is implemented in the Python library `iisignature` (Reizenstein and Graham, 2020). Thus, for a sample consisting of ℓ points in $\mathbb{R}^d$, if the path formed by their linear interpolation is considered, the computation of the path signature truncated at level K takes $O(\ell d^K)$ operations. The complexity is therefore linear in the number of sampled points but exponential in the truncation order K. Notice that the size of the signature vector is also exponential in the truncation order K, as shown in Table 2.1. Therefore, in applications, K has to remain small, typically of the order less than 10.

We conclude this section by giving some insights into the behavior of the expected value of the signature. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space and X a $\mathbb{R}^d$-valued stochastic process defined on $[0, 1]$. Chevyrev and Lyons (2016) have shown that, under some assumptions, $\mathbb{E}[S(X)]$ characterizes the law of X. In other words, if X and Y are two stochastic processes such that $\mathbb{E}[S(X)] = \mathbb{E}[S(Y)]$, then X and Y have the same distribution. The assumptions of this result have been relaxed by Chevyrev and Oberhauser (2018), who only need to assume that the signature is well-defined to prove a similar result for a particular renormalization of the signature. It is instructive to compare this property to the case of random variables. Indeed, recall that if X is a real-valued random variable, then its moment-generating function, defined by $t \mapsto \mathbb{E}[e^{tX}]$ characterizes the law of X. Now if X is a stochastic process, its expected signature has the same property and should, therefore, be thought of as a generalization of the moment-generating function of a process. This interpretation of the signature is particularly clear in the case where $d = 1$: let $X : [0, 1] \mapsto \mathbb{R}$, then

$$\mathbb{E}[S(X)] = \Big(1, \mathbb{E}[X_1 - X_0], \frac{\mathbb{E}[(X_1 - X_0)^2]}{2!}, \dots, \frac{\mathbb{E}[(X_1 - X_0)^k]}{k!}, \dots\Big),$$

and the signature corresponds exactly to an infinite sequence of the moments of the path.

2.2.2 Signature and machine learning

Now that the signature and its properties have been presented, we focus on its use in machine learning. In a statistical context, our goal is to understand the relationship between a random input path $X : [0, 1] \to \mathbb{R}^d$ and a random output $Y \in \mathbb{R}$. In a classical setting, we would be given a set of independent and identically distributed (i.i.d.) observations $\big\{(X_1, Y_1), \dots, (X_n, Y_n)\big\}$, drawn from (X, Y). However, in applications, a realization X_i is observed only at a discrete set of times $0 \leq t_1 < \cdots < t_{\ell_i} \leq 1$, $\ell_i \in \mathbb{N}^*$. Therefore, we are given an i.i.d. sample

$\{(\mathbf{x_1}, Y_1), \dots, (\mathbf{x_n}, Y_n)\}$, where $\mathbf{x_i}$ takes the form of a matrix:

$$\mathbf{x_i} = \begin{pmatrix} x_{i,1}^1 & \dots & x_{i,\ell_i}^1 \\ \vdots & & \vdots \\ x_{i,1}^d & \dots & x_{i,\ell_i}^d \end{pmatrix} \in \mathbb{R}^{d \times \ell_i}. \tag{2.6}$$

In this notation, $x_{i,j}^k$ denotes the kth coordinate of the ith sample observed at time t_j.

It is worth clarifying the terminology used and how it relates to other disciplines. In time series analysis, $\mathbf{x_i}$ would be a time series evolving in $\mathbb{R}^d$ and observed *at* ℓ_i time points. The output Y would be a future step of the time series, for example, if $d = 1$ then $Y_i = x_{i,\ell_i+1}^1$. If $d = 1$, then $\mathbf{x_i}$ is a univariate time series, whereas if $d > 1$, then it is called a multidimensional, multivariate, vector-valued or multiple time series. The parameter ℓ_i would be the series length or the time horizon. If all time series have the same length ℓ, then notation simplifies as $\mathbf{x_i} \in \mathbb{R}^{d \times \ell}$. On the other hand, in functional data analysis, X_i would be called functional data, a functional observation, or a curve, Y_i would be a scalar response, and ℓ_i would be the number of measurements. If this functional data was longitudinal, each $x_{i,\cdot}^k$ would be a functional observation or a profile of the subject i, and d would be the number of repeated measurements. Finally, in time series classification, $\mathbf{x_i}$ is a time series and Y_i the label of its class. Borrowing from the machine learning vocabulary, we will also refer to $\mathbf{x_i}$ as the raw data or the input data, and to Y_i as the output or the response.

The assumption that Y_i is a real number excludes several situations from our study. For example, the goal of functional longitudinal data analysis is usually the prediction of the next functional profile, which does not fall within our setting. Similarly, prediction of functional responses, which are a topic of interest in functional data analysis, or of multiple time points in time series analysis, are not considered.

Finally, it is worth noting the dependence of the length ℓ_i on i. In other words, each observation may have a different length. The signature dimension being independent of the number of sampled points, representing time series by their signature naturally handles inputs of various lengths, whereas traditional methods often require them to be normalized to a fixed length. Moreover, no assumption is made on the sampling intervals $t_1, \dots, t_{\ell_i}$, which can therefore be irregularly spaced and vary from one sample to another. To sum up, the signature method is appropriate for learning with discretely sampled multidimensional time series, possibly of different lengths and irregularly sampled.

As an example, consider the Google dataset Quick, Draw! (Google, 2017). It consists of the pen trajectories of millions of drawings, divided into 340 classes. Some examples are shown in Figure 2.2. In this case, the y_i are discrete labels of the drawing's class, and the $\mathbf{x_i}$ are matrices of pen coordinates. In this example, $d = 2$ and p_i varies for each drawing but is typically in the order of a few dozen points.

As discussed in the introduction, to use signature features, one needs to embed the observations $\mathbf{x_i}$ into paths of bounded variation $X_i : [0, 1] \to \mathbb{R}^d$. This step, which also consists of adding other coordinates, such as time, will be thoroughly discussed in Section 2.4. Assume for the moment that a set of embeddings X_i, $1 \leq i \leq n$, are given. When an embedding has been chosen, one can compute signature features truncated at a certain order K and use them in combination with a learning algorithm. The choice of K corresponds to a classical bias-variance tradeoff: the larger K, the larger the feature set. Therefore, it should be selected in a data-driven way, for example with cross-validation. The procedure can be summarized as follows:

$$\text{Raw data} \to \text{Embedding} \to \text{Signature features} \to \text{Algorithm}.$$

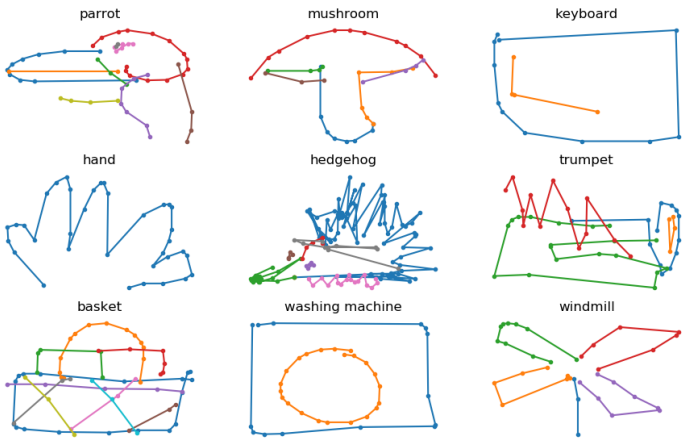


Figure 2.2 – 9 drawings from the Quick, Draw! dataset

The literature on the combination of signature features with learning algorithms can be divided into three groups. These groups correspond to the nature of the algorithm's input: it is either a vector, a sequence, or an image. In the context of deep learning, this division matches the different classes of neural network architectures: feedforward, recurrent and convolutional networks. The latter only deals with input paths in $\mathbb{R}^2$, with applications such as characters or handwriting recognition.

The first approach is to compute the signature of X on its whole domain, that is, on $[0, 1]$. In this way, the time-dependent input X is mapped into a time-independent finite set of coefficients, that is then fed into a predictive algorithm, typically a feedforward neural network. Any time-independent additional covariates may be added to this feature set. This strategy is implemented by Yang et al. (2017) for skeleton-based human action recognition. From a sequence of human joints' positions, the authors construct a high dimensional vector of signature coefficients, which is then the input of a small dense network. Gyurkó et al. (2014) and Lyons et al. (2014) also apply this method to financial time series, combining it with Lasso and ordinary least squares regression.

A second family of methods consists in describing the input path by a sequence of signature coefficients. There are several variants of this approach, but the one of Wilson-Nunn et al. (2018) is presented here in detail for its simplicity and representativeness. To create a signature sequence, the time interval $[0, 1]$ is divided into a dyadic partition

$$0 \leq 2^{-q} < \cdots < j2^{-q} < \cdots < (2^q - 1)2^{-q} \leq 1, \tag{2.7}$$

where $q \in \mathbb{N}$. By computing the signature truncated at order K on every dyadic interval $[j2^{-q}, (j + 1)2^{-q}]$, $0 \leq j < 2^q$, a sequence of 2^q signature vectors is obtained, each of dimension $(d^{K+1} - 1)/(d - 1)$. This sequence is typically fed into a recurrent network, as illustrated in Figure 2.3. In this case, the whole approach boils down to transforming the original sequential data into another sequence of signature coefficients. Such a procedure may be surprising, as the original data could have been itself the input of a recurrent network, instead of being mapped into a new signature sequence. However, Lai et al. (2017), Liu et al. (2017) and Wilson-Nunn et al. (2018) show the superiority of this type of approach for several tasks such as writer and

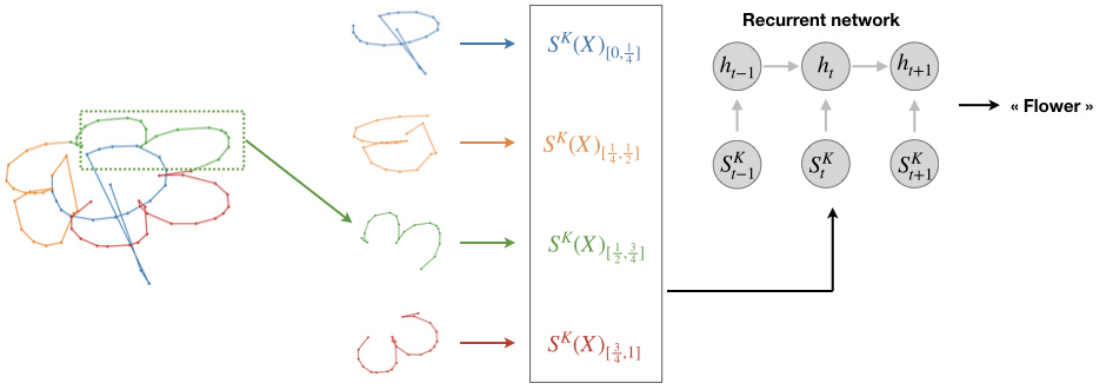


Figure 2.3 – Signature and recurrent neural network.

forgeries recognition.

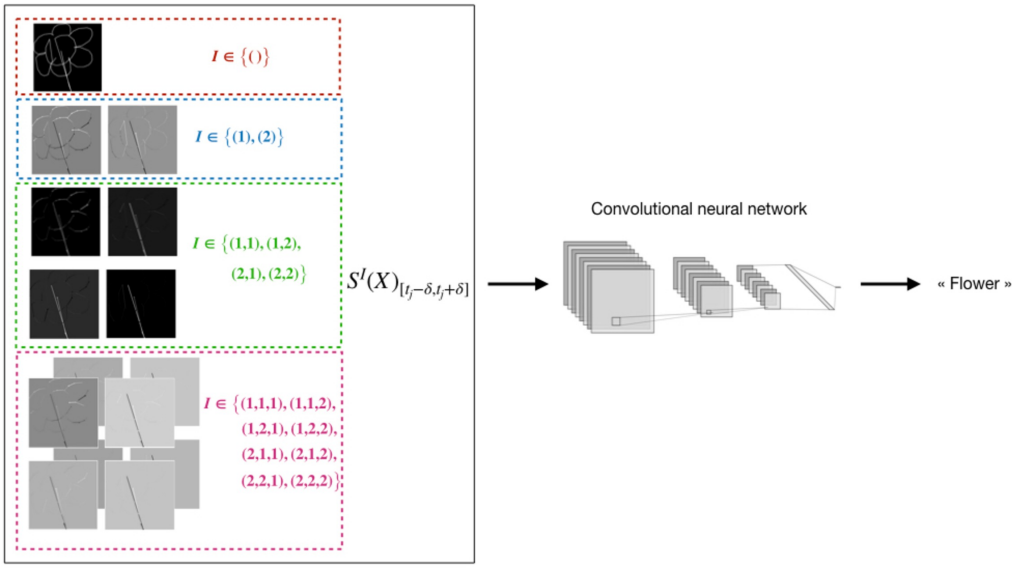


Figure 2.4 – Signature and convolutional neural network.

Finally, a third group of authors has taken these ideas further and created images of signature coefficients. Their rationale is to mix the temporal and pictorial aspects of the data. Indeed, assuming that the input path is a trajectory in $\mathbb{R}^2$, it can be turned into an image by forgetting its temporal aspect and setting the pixel values to 1 along the trajectory, and 0 elsewhere. Then, starting from this representation, a bunch of images is created, such that each image corresponds to a signature coefficient, as shown in Figure 2.4. This can be done in various ways. However, the general idea is to consider a sliding window following the path and to set the pixel value at the center of the window to be equal to the signature coefficient computed over the window. If the

signature is truncated at order K, this yields $2^{K+1}-1$ sparse grey pictures, which can then be the input of a convolutional neural network. Graham (2013) and Yang et al. (2015), 2016a have obtained significant accuracy improvements for character recognition and writer identification with this approach.

To sum up, the signature may be used in various ways, and for different applications. Several points of view coexist, and none of them has shown to be systematically better. In particular, on the one hand, signatures may be used to remove temporal aspects and to reduce the dimension of the problem, whereas, on the other hand, they may do the opposite and increase the dimension of the algorithm's input. Moreover, they are combined with various learning algorithms and it may be hard to distinguish the properties of the signature from those of the algorithms. Nevertheless, all these methods assume that discrete data points have been embedded into actual continuous paths. As will be seen in Section 2.4, the choice of path is crucial. Therefore, we describe in the next section the datasets used throughout the article to understand their underlying structure and find suitable embeddings.

2.3 Datasets

The datasets used in this article have been chosen to cover a broad range of applications while being recent and challenging in various ways. Moreover, they present a variety of sampling frequencies and dimensions. They illustrate therefore different potential embeddings.

First, the Quick, Draw! dataset (Google, 2017), which was already discussed in Section 2.2.2, and illustrated in Figure 2.2, is a public Google dataset. It consists of 50 million drawings, each drawing being a sequence of time-stamped pen stroke trajectories, divided into 340 categories. It takes approximately 7 gigabytes of hard disk space and is, therefore, a particularly large dataset. To compute the signature of every sample, it would thus be necessary to design a specific architecture, which cannot be implemented on a standard laptop computer. However, the goal of this study is not to achieve the best possible performance, but to understand embedding properties. Moreover, the experiments should be easily reproducible without requiring much computational ressources. Therefore, only a subset of the data is used: 68 000 training samples in Sections 2.4.2 and 2.6.1, 12 million in Section 2.6.2.

Let us describe more precisely the data format. When an object is drawn, two pieces of information are recorded: pen positions, sampled at different times, and pen jumps. Therefore, one drawing consists of a set of strokes, one stroke being a segment of the drawing between two pen jumps, represented with different colors in Figure 2.2. As each stroke can be of different length, if $\ell_{i,s}$ is the number of points in the sth stroke of drawing number i, and if this drawing has S_i strokes, then one drawing consists of S_i tables of sizes $2 \times \ell_{i,1}, \dots, 2 \times \ell_{i,S_i}$, where the factor 2 corresponds to the plane $\mathbb{R}^2$. For example, in Figure 2.2, the windmill drawing has $S_i = 5$ strokes: the first stroke is the blue one with 3 points ($\ell_{i,1} = 3$), the second one the orange with $\ell_{i,2} = 5$ points, and so on. The data has been preprocessed by Google, resulting in the so-called "simplified drawing files". The reader is referred to Google (2017) for a complete description of the preprocessing steps. Finally, each sample i can be encoded under the following compact form:

$$\mathbf{x_i} = \begin{pmatrix} x^1_{i,1} & \dots & x^1_{i,\ell_{i,1}} & \dots & x^1_{i,\ell_{i,1}+\dots+\ell_{i,S_i-1}+1} & \dots & x^1_{i,\ell_{i,1}+\dots+\ell_{i,S_i}} \\ x^2_{i,1} & \dots & x^2_{i,\ell_{i,1}} & \dots & x^2_{i,\ell_{i,1}+\dots+\ell_{i,S_i-1}+1} & \dots & x^2_{i,\ell_{i,1}+\dots+\ell_{i,S_i}} \\ 1 & \dots & 1 & \dots & S_i & \dots & S_i \end{pmatrix} \in \mathbb{R}^{3\times \ell_i}, \tag{2.8}$$

where $(x^1_{i,j}, x^2_{i,j})$ are the coordinates of the jth point of the drawing number i, and there is a pen

jump when the last row of $\mathbf{x_i}$ increases by 1. As in (2.6), $\ell_i = \ell_{i,1} + \cdots + \ell_{i,S_i}$ denotes the total number of points of drawing i.

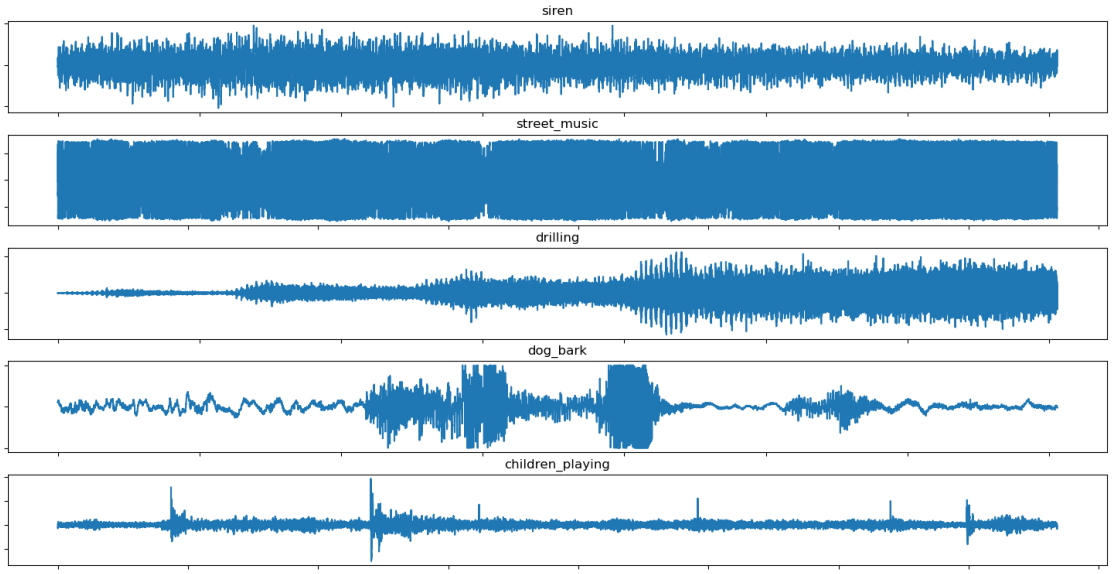


Figure 2.5 – 5 samples from the Urban sound dataset

The second dataset is the Urban Sound dataset (Salamon et al., 2014). It consists of sound recordings, divided into 10 classes: car horn, dog barking, air conditioner, children playing, drilling, engine idling, gunshot, jackhammer, siren, and street music. It contains both mono and stereo recordings, so some samples take values in $\mathbb{R}$, and some in $\mathbb{R}^2$. By averaging the two channels of stereo sounds, the data has been normalized to mono recordings, so that each sample is a one-dimensional time series: $\mathbf{x_i} \in \mathbb{R}^{1\times\ell_i}$. This yields a collection of 5 435 time series of various lengths. On average, they are sampled at approximately 170 000 points, which makes them long time series, typically hard to model along the whole time range. Figure 2.5 depicts some examples of these noisy time series.

Finally, the MotionSense dataset is composed of smartphone sensory data generated by accelerometer and gyroscope sensors (Malekzadeh et al., 2018). This data has been recorded while some participants performed an activity among walking upstairs and downstairs, walking, jogging, sitting, and standing. In total, there are 10 classes and 360 recordings, which correspond to 24 participants performing 15 different trials. During each trial, 12 variables are measured: 3 directions for attitude, gravity, user acceleration, and rotation rate, respectively. Information about the participants is provided but we focus on the task of recognizing the activity performed from the multidimensional time series formed by sensors data. In Figure 2.6, three samples are shown, and, for each of them, curves of different colors correspond to the various quantities measured by sensors. Therefore, every sample can be written as $\mathbf{x_i} \in \mathbb{R}^{12\times\ell_i}$. It is clear from Figure 2.6 that these series are noisy and highly dimensional. They are shorter than the Urban Sound's ones, with an average of approximately 4 000 time steps.

Finally, for all datasets, the response Y is a class label, that is $Y \in \{1, \ldots, C\}$, where C is the number of classes. Table 2.2 summarizes some characteristics of these three datasets. As each sample may have a different length, the data average length is recorded, defined by $\overline{\ell} = \frac{1}{n}\sum_{i=1}^{n} \ell_i$. These datasets illustrate the diversity of problems in sequential learning, where time appears in

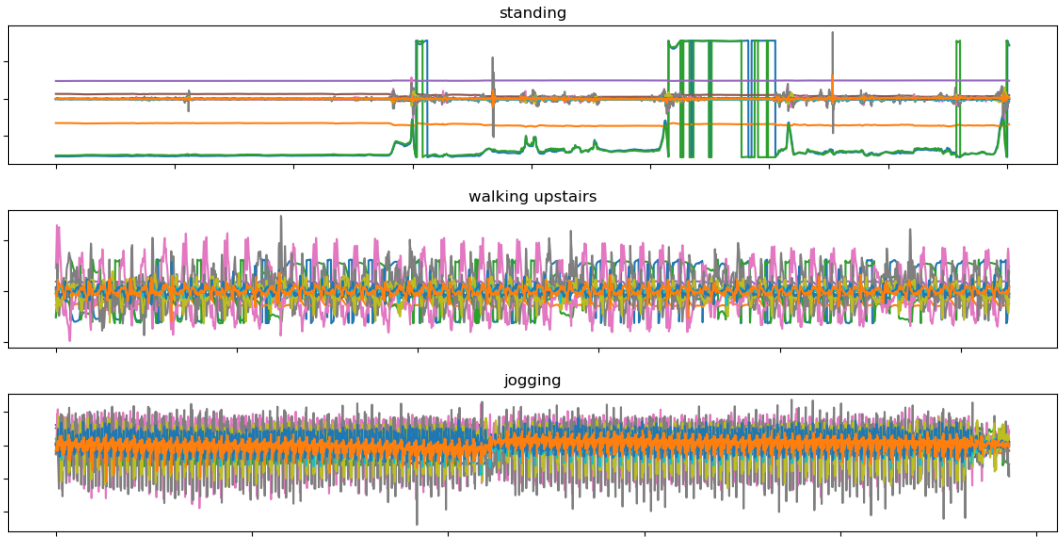


Figure 2.6 – 3 samples from the Motion sense dataset

different ways.

	Quick, Draw!	Urban Sound	Motion Sense
Number of classes C	340	10	6
Dimension d	2	1	12
Average length $\bar{\ell}$	44	171 135	3 924
Training set size n	68 000	4 435	300
Validation set size n_{val}	6 800	500	30
Test set size n_{test}	6 800	500	30

Table 2.2 – Datasets summary

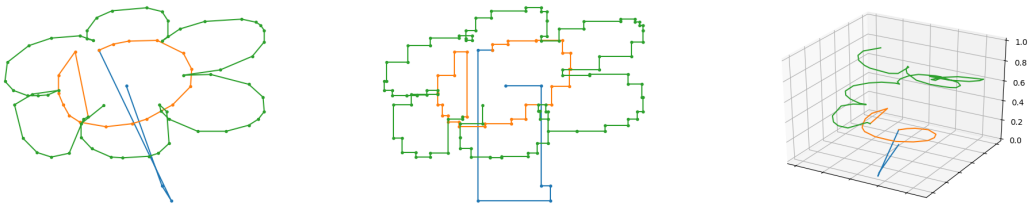
2.4 The embedding

In practice, a matrix of observations $\mathbf{x_i} \in \mathbb{R}^{d \times \ell_i}$ is given, written in (2.6), where columns correspond to points in $\mathbb{R}^d$ sampled at times $0 \leq t_1 < \cdots < t_{\ell_i} \leq 1$. As explained in Section 2.2.2, the goal is to construct a continuous path $X_i : [0,1] \to \mathbb{R}^d$ from the matrix $\mathbf{x_i}$. Therefore, an interpolation method needs to be chosen, but, to ensure some properties such as signature uniqueness (Proposition 2.2), new coordinates may be added to the path, which increases the dimension d of the embedding space. When not hidden, the embedding is generally only mentioned in the literature, without in-depth discussions. Therefore, the purpose is not only to compare embeddings' performance, but also to give a first systematic survey of their use in the context of learning with signatures.

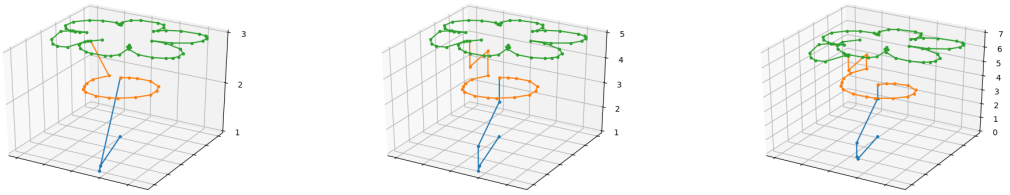
2.4.1 Definition and review of potential embeddings

First, different embeddings are reviewed while adapting them to the Quick, Draw! dataset for illustrative purposes. The extension to other datasets follows immediately. All embeddings considered here are continuous piecewise linear, but their difference lies in the way this interpolation is performed. From a computational point of view, signatures of continuous piecewise linear paths can be computed with the library `iisignature`, as mentioned in Subsection 2.2.1. From now on, consider a sample $\mathbf{x} \in \mathbb{R}^{3\times p}$, which can be written as the matrix (2.8) where the index i has been removed to simplify notations.

Linear path A first natural choice is to interpolate data points linearly, that is to connect each consecutive points by a straight line. Note that for the Quick Draw! data, information about pen jumps is then lost. Thus, for a particular sample, if ℓ positions of the pen $[(x_1^1, x_1^2), \dots, (x_\ell^1, x_\ell^2)]$ are given, the piecewise linear path $X : [0,1] \to \mathbb{R}^2$ is defined as the path equal to (x_j^1, x_j^2) at t_j, where $0 = t_1 < t_2 < \cdots < t_\ell = 1$ is a partition of $[0,1]$ into ℓ points. This yields a two-dimensional continuous path with coordinates (X_t^1, X_t^2). This path, represented in Figure 2.7a, is the most often used in the literature, for example by Graham (2013), Lai et al. (2017), or Yang et al. (2016a).



(a) Linear path. (b) Rectilinear path. (c) Time path.



(d) Stroke path, version 1. (e) Stroke path, version 2. (f) Stroke path, version 3.

Figure 2.7 – Different embeddings of a Quick, Draw! sample. Each stroke is plotted with a different color only for the sake of illustration.

Rectilinear path Another interpolation method is often used in the literature (Chevyrev and Kormilitzin, 2016b; Kormilitzin et al., 2016) and referred to as an "axis path" or "rectilinear path". It is also piecewise linear but each linear section is parallel to an axis. In other words, to move from one point (x_j^1, x_j^2) to another point (x_{j+1}^1, x_{j+1}^2), a first linear segment goes from (x_j^1, x_j^2) to (x_{j+1}^1, x_j^2), parallel to the x-axis, and a second segment from (x_{j+1}^1, x_j^2) to (x_{j+1}^1, x_{j+1}^2),

parallel to the y-axis. This path is depicted in Figure 2.7b. A crucial aspect of this path is that there exists a simple way to reconstruct it from its signature features (Lyons and Xu, 2017). Note that for unidimensional data, such as the Urban Sound dataset, the linear and rectilinear interpolations are identical.

Time path The third approach builds upon the linear path and enriches it by adding a monotone coordinate. This ensures the uniqueness of the signature, as stated in Proposition 2.2. It usually corresponds to adding the time parametrization as a coordinate of the path, as is done by Yang et al. (2017). Therefore, if $t \mapsto (X_t^1, X_t^2)$ is the linear path described above, which is piecewise linear, the time embedding is the 3-dimensional path $t \mapsto (X_t^1, X_t^2, t)$, shown in Figure 2.7c.

Lead-lag path Introduced by Chevyrev and Kormilitzin (2016b) and Flint et al. (2016), the lead-lag transformation has been applied by, e.g., Gyurkó et al. (2014), Lyons et al. (2014), Kormilitzin et al. (2016), , and Yang et al. (2017). Building on the time path, the idea is to add lagged versions of the coordinates X^1 and X^2 as new dimensions. Let ℓ be the length of the input path, and $0 = t_1 < t_2 < \cdots < t_\ell < t_{\ell+1} = 1$ be a partition of $[0, 1]$ into $\ell + 1$ points. Then the lead-lag path with lag 1 is defined by

$$\begin{aligned} X : [0,1] &\to \mathbb{R}^5 \\ t &\mapsto (X_t^1, X_t^2, t, X_t^3, X_t^4). \end{aligned}$$

In this definition, X^1 and X^2 are a linear interpolation of the sequence

$$\big[(x_1^1, x_1^2), \ldots, (x_\ell^1, x_\ell^2), (x_\ell^1, x_\ell^2)\big],$$

in which the last point is repeated twice, and

$$X_t^3 = \begin{cases} 0 & \text{if } t < t_1 \\ X_{t-t_1}^1 & \text{otherwise} \end{cases}, \qquad X_t^4 = \begin{cases} 0 & \text{if } t < t_1 \\ X_{t-t_1}^2 & \text{otherwise} \end{cases}. \tag{2.9}$$

This yields a 5-dimensional path such that the last two coordinates are delayed copies of the first two, with a delay of t_1. The process can be iterated, creating a path in $\mathbb{R}^7$ with two lags $t \mapsto (X_t^1, X_t^2, t, X_t^3, X_t^4, X_t^5, X_t^6)$, where X^1 and X^2 are linear interpolations of the data with the last point repeated three times, X^3 and X^4 are defined by (2.9), and

$$X_t^5 = \begin{cases} 0 & \text{if } t < t_2 \\ X_{t-t_2}^1 & \text{otherwise} \end{cases}, \qquad X_t^6 = \begin{cases} 0 & \text{if } t < t_2 \\ X_{t-t_2}^2 & \text{otherwise} \end{cases}.$$

In this way, the lead-lag path can naturally be defined for any lag in $\mathbb{N}^*$. This path is highly dimensional (in $\mathbb{R}^7$ for a lag of 2) and cannot be represented easily. Therefore, Figure 2.8 shows some coordinates against time, namely X^1, X^3, and X^5.

Stroke path For the Quick, Draw! data, extra information about pen jumps is provided. In the context of Arabic handwriting recognition, Wilson-Nunn et al. (2018) have introduced the idea of encoding information about jumps into a new coordinate. In essence, the approach is to use a 3-dimensional path in which the last dimension corresponds to strokes, in a similar way to the encoding of matrix (2.8). This procedure can be deployed in various ways and we restrict our attention to three of them.

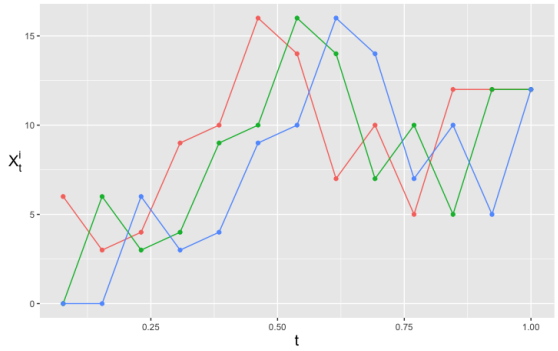


Figure 2.8 – X_t^1 (red), X_t^3 (green), and X_t^5 (blue), coordinates of the lead-lag embedding with lag 2 against t, for $t \in [0, 1]$.

The first approach uses the description of a drawing as given in (2.8). Recall that, in this matrix, the stroke categorical variable is initialized to 1, and increased by 1 each time a different stroke begins. The idea is then to linearly interpolate the columns of the matrix, considered as points in $\mathbb{R}^3$. As can be seen in Figure 2.7d, each stroke is then represented in a different horizontal plane. This procedure looks like the most natural way to encode jumps information, and from now on is called "version 1" of the stroke path.

A related approach is considered by Wilson-Nunn et al. (2018). It is represented in Figure 2.7e and subsequently called "version 2". Here, each stroke is indexed by odd integers, that is the first stroke is indexed by 1, the second by 3,..., and the kth by $2k - 1$. Two intermediary points are added between each stroke, indexed by even integers. For example, if $(x^1_{\ell_1}, x^2_{\ell_1}, 1)$ is the last point of the first stroke, and $(x^1_{\ell_1+1}, x^2_{\ell_1+1}, 3)$ is the first point of the second stroke, the points $(x^1_{\ell_1}, x^2_{\ell_1}, 2)$ and $(x^1_{\ell_1+1}, x^2_{\ell_1+1}, 2)$ are added to the path and linearly interpolated. This is represented in Figure 2.7e. The only difference with the previous path is how the path moves from one plane to another one. Instead of doing a straight line, it moves in two steps: one in a horizontal plane and another one parallel to the vertical axis. With this embedding and a recurrent network, Wilson-Nunn et al. (2018) have achieved a significant decrease in the error rate of Arabic characters recognition.

Finally, for comparison purposes, a strictly monotone coordinate is also considered. It has jumps of 1 when a new stroke begins, and otherwise grows linearly inside one stroke, such that it has increased by 1 between the beginning and the end of the stroke. In this definition, the goal is to check whether having a strictly monotone coordinate increases accuracy, while in the two previous versions the stroke coordinate is piecewise constant. This embedding can be seen as a mix between time and stroke paths and could inherit the good properties of both. The resulting path is called "version 3" and shown in Figure 2.7f.

To conclude, there exists a broad range of embeddings, living in spaces of various dimensions. They lead to different signature features, which therefore do not have the same statistical properties. The embedding choice will prove to have a significant influence on accuracy.

2.4.2 Results

In this subsection, the results of our study on embedding performance are presented. To this end, the first approach described in Section 2.2.2 is implemented. Starting from the raw data, it is first embedded into a continuous path, then its truncated signature is computed and used as input for a learning algorithm. The embeddings described in the previous section are used. Note

that the lead-lag path is taken with lag 1, but other lags will be discussed in Section 2.6.2. Each feature is normalized by the absolute value of its maximum so that all input values lie in $[-1, 1]$. The findings should be independent of the data and the underlying statistical model so a range of different algorithms is used. Their hyperparameters have been set to their default values, without trying to optimize them for each dataset. Indeed, the goal is not to select the best algorithm or to achieve a particularly good accuracy, but rather to compare the performance of different embeddings. The classification metric to assess prediction quality is the accuracy score. Denoting by $(y_1, \dots, y_{n_{\text{test}}})$ the test set's labels, and $(\hat{y}_1, \dots, \hat{y}_{n_{\text{test}}})$ the predicted labels, this score is defined by

$$\text{Acc}_{\text{test}} = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \mathbb{1}_{\hat{y}_i = y_i}. \tag{2.10}$$

The four following algorithms have been used throughout the study.

- Following Yang et al. (2017), a dense network with one hidden layer composed of 64 units with linear activation functions is first considered. A softmax output layer and the categorical cross-entropy loss are used, which yields a linear model equivalent to logistic regression. This architecture is a sensible choice, since Proposition 2.3 states that linear functions of the signature approximate arbitrarily well any continuous function of the input path. The Python library `keras` (Chollet et al., 2015), with TensorFlow backend, is used. The network is regularized by adding a dropout layer after the input layer, with a rate of 0.5. Optimization is done with stochastic gradient descent with an initial learning rate of 1. It is reduced by 2 when no improvement is seen on a validation set during 10 consecutive epochs. The maximal number of epochs is set to 200 and the mini-batch size to 128.
- Furthermore, the performance of a random forest classifier with 50 trees, implemented in scikit-learn (Pedregosa et al., 2011), is tested. It is a nonlinear very popular method initially proposed by Breiman (2001).
- The XGBoost algorithm, introduced by Chen and Guestrin (2016), and implemented in the Python package `xgboost`, is also used. It is a state-of-the-art gradient boosting technique, building upon the work of (Friedman, 2001). The maximum number of iterations is set to 100 and early stopping with a patience of 5 is used to prevent overfitting and speed up training. The maximum depth of a tree is set to 3 and the minimum loss reduction to make a split to 0.5.
- Finally, a nearest neighbor classifier is run with a default value of 5 neighbors. This method is known to suffer from the curse of dimensionality, so it is of interest to see how the signature truncation order affects its performance.

For each of the algorithms described above and each dataset of Section 2.3 (Quick, Draw!, Urban Sound, and Motion Sense), the following steps are repeated:

1. Split the data into training, validation, and test sets, as described in Table 2.2.
2. Choose an embedding and transform samples $\mathbf{x_i}$ into continuous paths $X_i : [0, 1] \to \mathbb{R}^d$.
3. For $k = 1, \dots, K$:
   (a) Compute $S_k(X_i)$, the signature truncated at order k, for every sample i. This results in training, validation and test sets of the form

$$\big\{S_k(X_1), \dots, S_k(X_n)\big\},$$

   where $S_k(X_i) \in \mathbb{R}^{\frac{d^{k+1}-1}{d-1}}$ if $d > 1$, and $\mathbb{R}^k$ if $d = 1$.

(b) Fit the algorithm on the training data. Validation data is used when the algorithm chosen is the linear neural network or XGboost, to adapt the learning rate and to implement early stopping, respectively.

(c) Compute the accuracy, defined by (2.10), on the test set.

The maximal value considered for the truncation order, denoted by K, is fixed so that the number of features are computationally reasonable. The meaning of "reasonable" depends on each dataset, as they have a different number of samples and classes. For Quick, Draw!, we will consider up to 10^5 samples, for Urban Sound 10^4 and for Motion Sense 5×10^5.

For a path X in $\mathbb{R}^d$, the number of features is equal to $(d^{k+1} - 1)/(d - 1)$ if $d > 1$, and to k if $d = 1$ (see Table 2.1 for some values). Therefore, the number of features depends on the dimension d of the embedding and the truncation order k. But d is different depending on the dataset and the embedding. Thus, to compare the quality of different embeddings, the accuracy score is plotted against the log number of features, which yields one curve per embedding, where each point corresponds to a different truncation order k. One embedding curve being above the others means that, at equal input size, this embedding performs better.

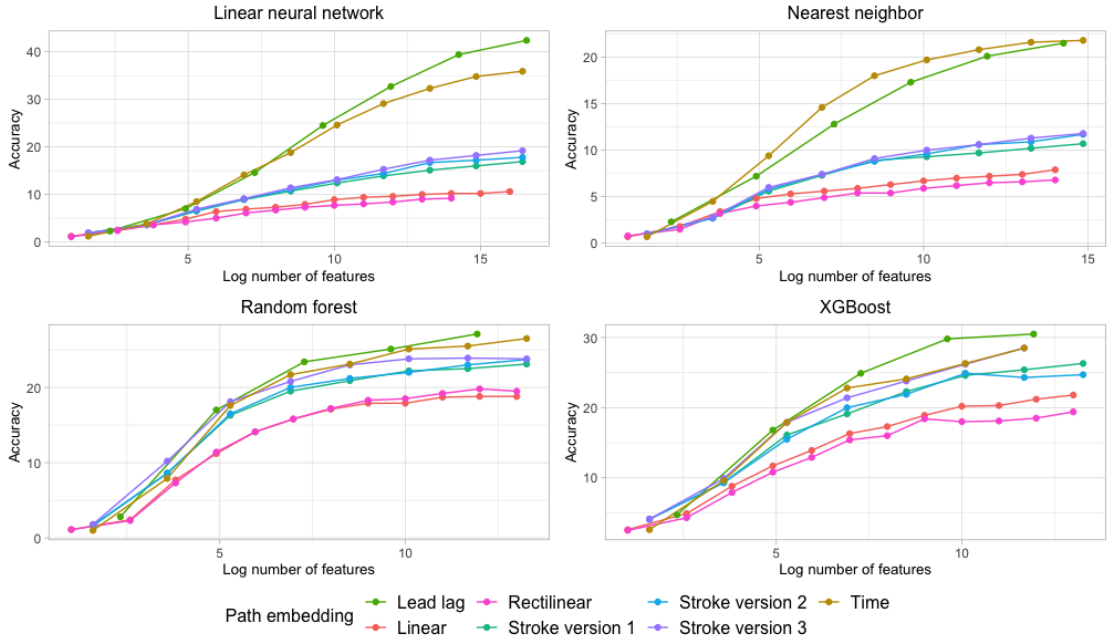


Figure 2.9 – Quick, Draw! dataset: prediction accuracy on the test set, for different algorithms and embeddings.

The results of this procedure are plotted in Figures 2.9, 2.10 and 2.11, which correspond respectively to the Quick, Draw!, Urban Sound and Motion Sense datasets. A first observation is that some embeddings, namely the time and lead-lag, seem consistently better, whatever the algorithm and the data used. It suggests that this performance is due to the intrinsic theoretical properties of signatures and embeddings, not to domain-specific characteristics. It is particularly remarkable as the dimension of input streams is different from one dataset to another.

The linear and rectilinear embeddings (red and pink curves), which are often used in the literature, appear to give the worst results. These two interpolation methods do not differ much in their results, although the linear path seems to be slightly better. Moreover, it seems that

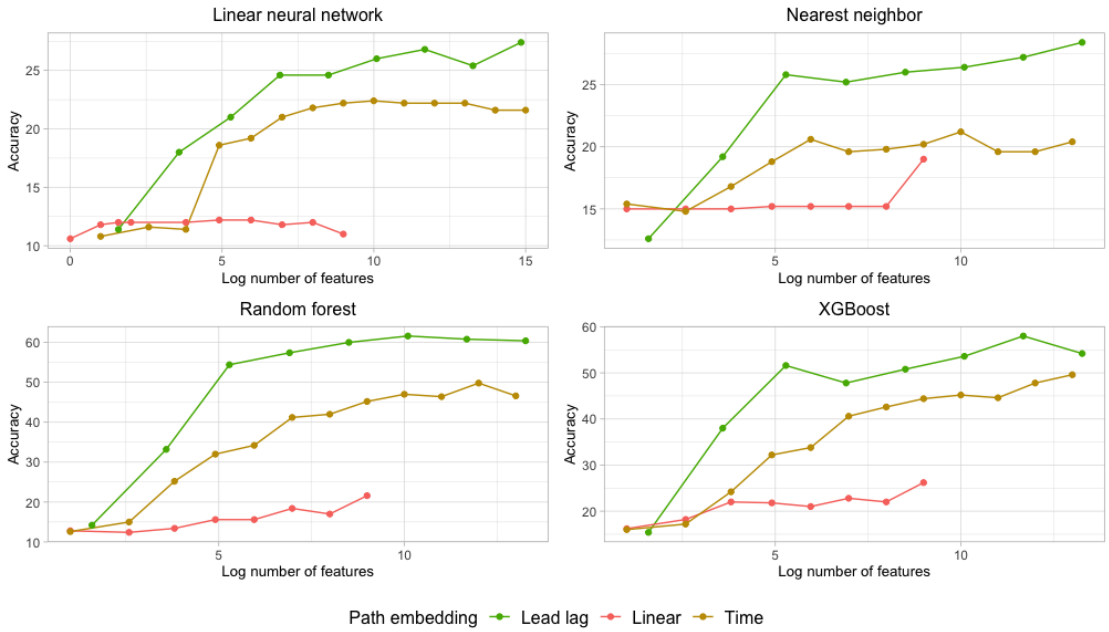


Figure 2.10 – Urban Sound dataset: prediction accuracy on the test set, for different algorithms and embeddings.

the smaller the dimension d, the worse their performance. Indeed, the linear embedding is especially bad for the Urban Sound dataset, which is unidimensional, whilst the difference is less pronounced for the Motion Sense dataset, which has values in $\mathbb{R}^{12}$. This bad performance can be explained by the fact that there is no guarantee that the signature transformation is unique when using the linear or rectilinear embeddings. Therefore, two different paths can have the same signature, without necessarily corresponding to the same class.

On the other hand, the best embedding is the lead-lag path (green curve), followed closely by the time path (brown curve). The difference between these two embeddings is again most important for the Urban Sound dataset. For the Quick, Draw! data, stroke paths have intermediate results, better than the linear path but still worse than the time and lead-lag paths. Yet stroke paths are the only embeddings in which new information, about pen jumps, is included. It is surprising how little impact this information seems to have on prediction accuracy. Note that in all of these cases, the uniqueness of the signature is ensured so it cannot explain the performance differences.

Good performance of the lead-lag path has already been noticed in the literature. However, up to our knowledge, there are few theoretical results. Still, Flint et al. (2016) have considered a discretely sampled input path X, assumed to be a continuous semimartingale, and have studied convergence results of its associated lead-lag path, called Hoff process, when sampling frequency increases. Thus, a lot of questions remain open concerning the statistical performance of the time and lead-lag embeddings, with, to our knowledge, no theoretical result in classification or regression frameworks.

To conclude this section, the take-home message is that using the lead-lag embedding seems to be the best choice, regardless of the data and algorithm used. It does not cost much computationally and can drastically improve prediction accuracy. Moreover, the linear and stroke paths yield surprisingly poor results, despite their frequent use in the literature.

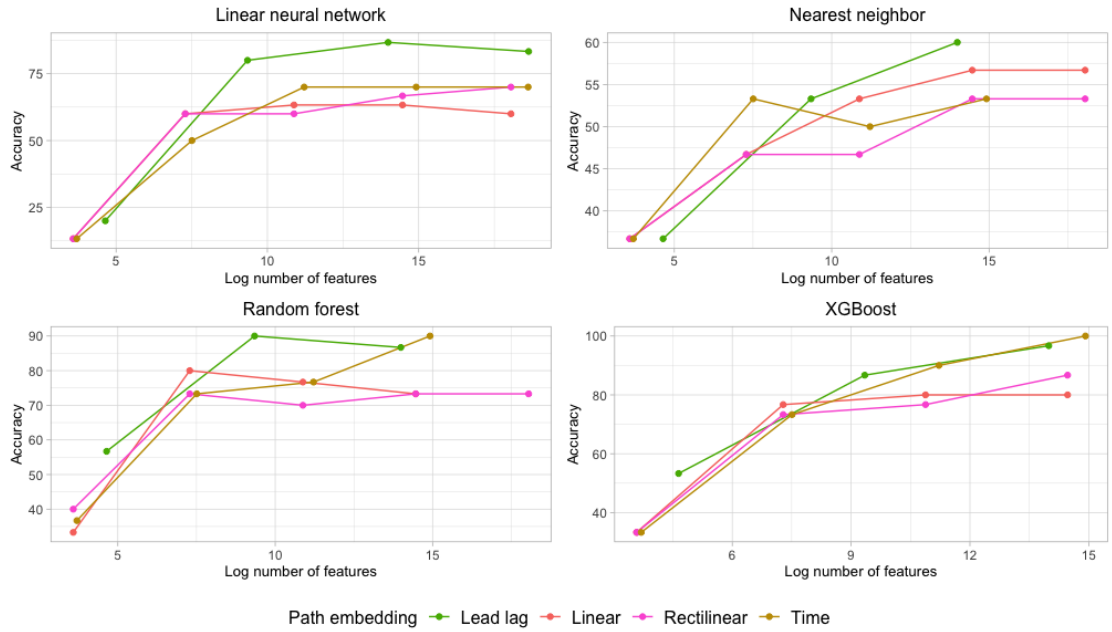


Figure 2.11 – Motion Sense dataset: prediction accuracy on the test set, for different algorithms and embeddings.

2.4.3 Running times

To conclude this study on embeddings, this section presents some results on the computational complexity of the different embeddings and truncation order. In Figure 2.12, the running times for computing signature features and fitting a random forest are shown as functions of the truncation order for various embeddings. The experiments were run on 32 Intel Xeon E5-4660 cores and parallelized with the `multiprocessing` Python package. The most expensive embedding is the lead-lag, which is not surprising as it doubles the path dimension. Moreover, increasing the truncation order increases exponentially the running time. For Quick, Draw! the running time is of the order of 10-100 seconds, for Motion Sense of the order of 100 seconds and for Urban Sound around 1000 seconds. This is directly linked to the length of the series: the longer the series, the more expensive it is to compute signatures.

In Figure 2.13 is presented the number of input features for each combination of embedding and truncation order. This is proportional to the memory needed to run each experiment. As given by equation (2.3), it is clear that the storage cost increases exponentially with the truncation order, which is the main limitation of the signature method.

2.5 Simulation study of autoregressive processes

In order to confirm the previous findings on the performance of the lead-lag embedding, we undertake a simulation study with autoregressive processes. This study will also provide insights on the sensitivity of the method to some hyperparameters such as the lag and the signature truncation order. We place ourselves in a regression setting, that is we consider n realizations of

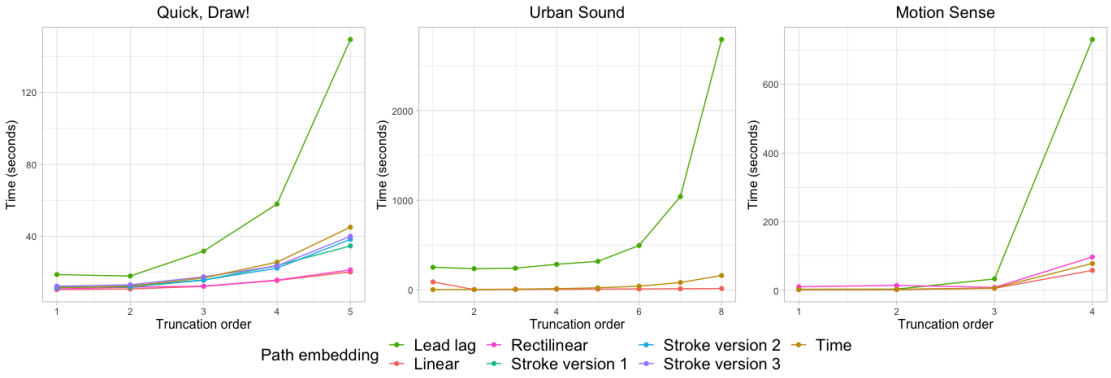


Figure 2.12 – Running time (in seconds) to compute signature features and fit a random forest classifier for various embeddings, as a function of the truncation order.

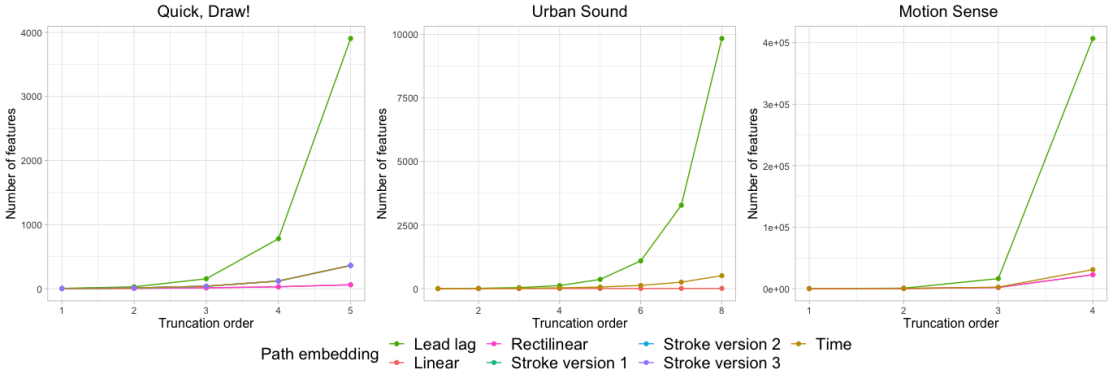


Figure 2.13 – Number of features of various embeddings as a function of the truncation order.

an AR(p) process Z_t, observed on $\ell = 100$ time points, and defined by

$$Z_t = \phi_1 Z_{t-1} + \cdots + \phi_p Z_{t-p} + \varepsilon_t, \quad 1 \leq t \leq \ell, \tag{2.11}$$

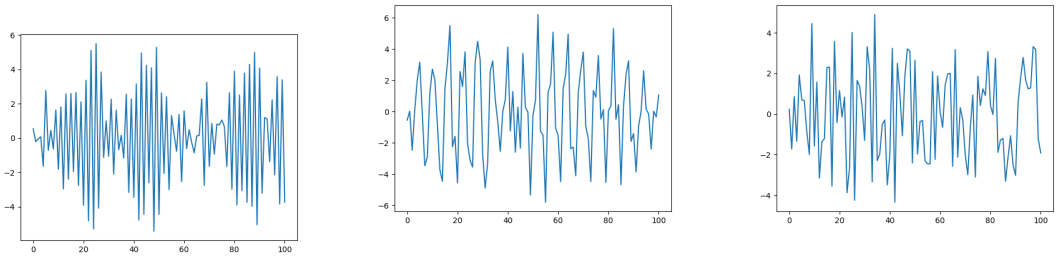
where ε_t is a gaussian random variable with mean 0 and variance 1. Some examples are shown in Figure 2.14. Our goal is to predict the next time step, that is $y = Z_{\ell+1}$, with a linear regression and signature features. The input data is therefore a set $\{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_n}, y_n)\}$, where each $\mathbf{x_i}$ is one realization of an AR(p) process:

$$\mathbf{x_i} = \left(Z_{i,1}, \ldots, Z_{i,\ell}\right) \in \mathbb{R}^{\ell \times 1},$$

and $y_i = Z_{i,\ell+1}$, where $Z_{i,\cdot}$ follows (2.11).

First, Figure 2.15 shows the results of the same study on the embeddings performance as in Section 2.4 for different AR(1) processes. The parameter ϕ_1 is equal to -0.9, -1 and 0.5, in order to obtain both stationary and nonstationary models. The performance of the different embeddings is plotted against a range of truncation orders. The metric is the L_2 error on a test set, which is defined by

$$S = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} (y_i - \hat{y}_i)^2,$$



(a) AR(1): $\phi_1 = -0.9$

(b) AR(3): $\phi_1 = \phi_2 = 0, \phi_3 = -0.9$

(c) AR(8): $\phi_1 = \cdots = \phi_7 = 0, \phi_8 = -0.9$

Figure 2.14 – Sample paths of an AR(p) process.

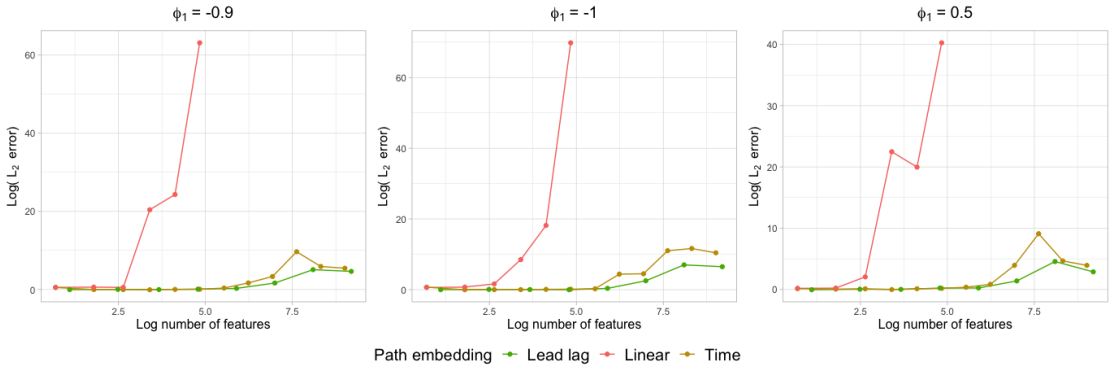


Figure 2.15 – Logarithm of the test error for different embeddings and truncation orders for prediction of an AR(1) process. The left panel corresponds to $\phi_1 = -0.9$, the middle one to $\phi_1 = -1$ and the right one to $\phi_1 = 0.5$.

with the same notations as (2.10). Contrary to Figures 2.9, 2.10 and 2.11 which use as metric the accuracy, the smaller S the better the prediction. It is clear in Figure 2.15 that the time and lead-lag embeddings have the smallest errors, confirming the findings of the previous section on real-world datasets. Moreover, the figures are very similar for stationary or nonstationary series, and different strength of time dependence. This shows the generality of the signature method, which does not require strong assumptions on the law of the underlying process.

A natural question is whether the lag parameter is linked to the dependence in the time series. To tackle this issue, Figure 2.16 shows a boxplot of the L_2 error as a function of the lag for 3 different values of p. The parameters of model (2.11) are set to the following values: for $p = 1$, $\phi_1 = -0.9$; for $p = 3$, $\phi_1 = \phi_2 = 0$ and $\phi_3 = -0.9$; for $p = 8$, $\phi_1 = \cdots = \phi_7 = 0$ and $\phi_8 = -0.9$. For each lag, the best truncation order is selected with a validation set, that is the truncation order is chosen to be the one achieving the lowest error on a validation set. The procedure is then evaluated on another test set. This procedure is iterated 20 times to obtain estimates of the variability of the error. Figure 2.16 shows that when p increases, the best lag increases. For $p = 1$, all lags seem to achieve similar errors, for $p = 3$, there is an error jump between a lag of 1 and a lag of 2, and for $p = 8$ the error decreases towards the best lag of 6. This is strong evidence of the link between the time dependence and the lag parameter.

Finally, Figures 2.17 and 2.18 investigate the link between the prediction error, the sample size and the truncation order of the signature for an AR(3) process with the same parameters as

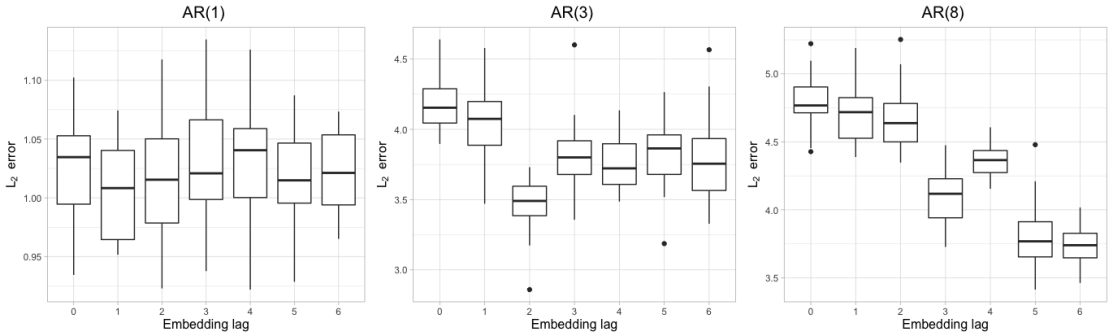


Figure 2.16 – Error boxplots of the L_2 error as a function of the lag of the embedding, for different AR processes.

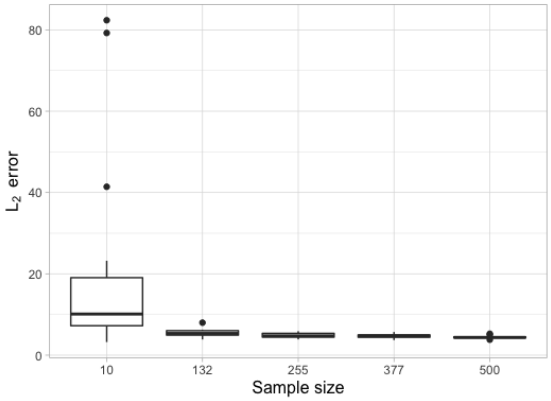


Figure 2.17 – Error boxplot for different sample sizes.

before. We choose a lead-lag embedding with a lag of 2, as suggested by Figure 2.16. For each sample size and truncation order, the model is fitted and evaluated 20 times. In Figure 2.17, the truncation order is chosen as the minimizer of the error on a validation set. Then, a boxplot of the errors is plotted as a function of the sample size. Both the error and its variance decrease fast when the sample size increases.

On the other hand, Figure 2.18 shows error boxplots as a function of the truncation order K, for various sample sizes. When the sample size is large enough, typically larger than 200, a bias-variance tradeoff can be observed: the error first decreases with the truncation order until a minimum is reached and then the error and its variance increase fast because the number of covariates is too large compared to the number of observations. It is interesting to see that the best truncation order increases when the sample size increases but stabilizes at $K = 4$. Moreover, the error variance is very large for a sample size of 10 but stays reasonably small for larger sample sizes.

2.6 Signature domain and performance

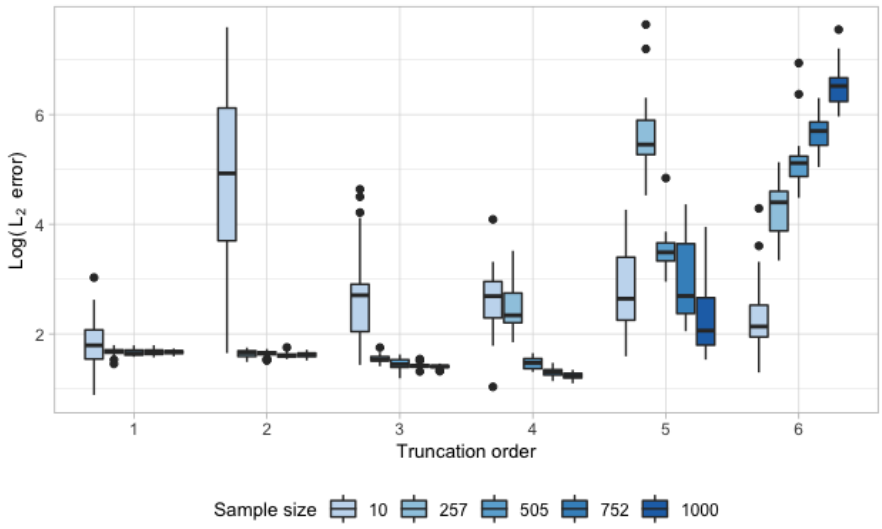


Figure 2.18 – Average error for different truncation orders and sample sizes.

2.6.1 Comparison of local and global signature features

As discussed in Section 2.2.2, several authors do not compute signatures on the whole time interval but use instead a partition of $[0,1]$. The rationale for this division is to describe the path by a sequence of truncated signatures, rather than one signature computed over the whole domain. Therefore, it is not surprising that this approach is typically used in combination with recurrent neural networks. In this section, it is investigated whether signatures computed on a sub-interval contain some local information not present in signatures of the whole path. To this end, following Wilson-Nunn et al. (2018), a dyadic partition of $[0,1]$ is considered, and defined by (2.7):

$$0 \leq 2^{-q} < \cdots < j2^{-q} < \cdots < (2^q - 1)2^{-q} \leq 1, \quad 0 < j \leq 2^q$$

where q is the dyadic order. For different values of q, signature coefficients are computed on each interval $\left[(j-1)2^{-q}, j2^{-q}\right]$ of the dyadic partition. Therefore, for each input path X_i, a collection of signature vectors is obtained, which is then stacked into one large vector. This vector is then the input of a learning algorithm, and the prediction accuracy curves of different dyadic orders are compared. The time embedding with the linear neural network described in Section 2.4.2 is used. This process is summarized below.

1. Split the data into training, validation and test sets.
2. For $q = 0, \ldots, Q$, and $k = 1, \ldots, K$:
   (*i*) For $j = 1, \ldots, 2^q$, compute the signature truncated at order k on $\left[(j-1)2^{-q}, j2^{-q}\right]$, denoted by
   $$S_k(X_i)_{[(j-1)2^{-q}, j2^{-q}]},$$
   where X_i is the time embedding of sample $\mathbf{x_i}$. Repeat this over all training samples.
   (*ii*) For each training sample X_i, concatenate all signature vectors and obtain one vector $\tilde{S}_k(X_i)$ containing all $S_k(X_i)_{[(j-1)2^{-q}, j2^{-q}]}$, for $j = 1, \ldots, 2^q$. This yields a dataset
   $$\left\{\tilde{S}_k(X_1), \ldots, \tilde{S}_k(X_n)\right\}.$$

(*iii*) Fit a linear neural network with this data as features.

(*iv*) Compute accuracy on the test set.

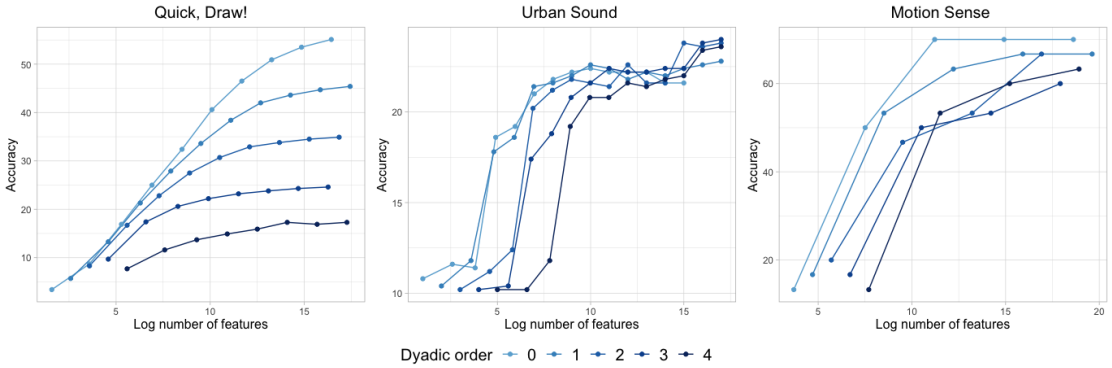


Figure 2.19 – Test accuracy for different dyadic partitions of the path.

The results of this procedure are shown in Figure 2.19. First, it is clear that a dyadic order of 0, which corresponds to computing the signature on the whole interval $[0, 1]$, yields the best results. Indeed, the curve is always above the others for the Quick, Draw! and Motion Sense datasets. This is less obvious for the Urban Sound dataset, as the curve $q = 0$ is really close to the one corresponding to a dyadic order of 1. However, it still achieves a better result for most truncation orders k. This difference between datasets can be linked to their length: it seems that the longer the series, the better the accuracy of thin dyadic partitions. Indeed, high dyadic orders perform best for the Urban Sound dataset, which has an average of 170 000 sampled time points (see Table 2.2), whereas it is clear that each new dyadic split decreases accuracy for the Quick, Draw! data, which has an average of 44 sampled points.

In a nutshell, little local information seems to be lost when the signature of the whole path is computed. However, it may be worth considering partitions of the path for long streams.

2.6.2 Performance of the signature

The message of previous sections is that the lead-lag embedding is the most appropriate in a learning context and that signatures should be computed over the whole path domain. As a natural continuation, the effect of the algorithm is now examined more closely and the prediction scores are compared to the literature. It turns out that the signature combined with a lead-lag embedding has an excellent representation power, to the extent that it achieves prediction scores close to state-of-the-art methods, without using any domain-specific knowledge.

Before starting the comparison, it is worth pointing out that the lead-lag embedding has a hyperparameter that has not yet be tuned, which is the number of lags. It is now selected with the same approach as in previous sections: for each lag, the test accuracy is plotted against the number of features for various truncation orders. The lag which gives a curve above the others is selected. Figure 2.20 highlights that curves overlap for the Motion Sense and Urban sound cases, therefore, when there is a doubt on which curve is above, the smallest lag is picked. For the Quick, Draw! and Motion Sense datasets, the best lag is then 1, whereas it is 5 for the Urban Sound dataset.

Finally, the truncation order is selected with a validation set: the truncation order achieving the highest accuracy on a validation set is picked. In general, the truncation order and the number of lags could also be selected with cross-validation.

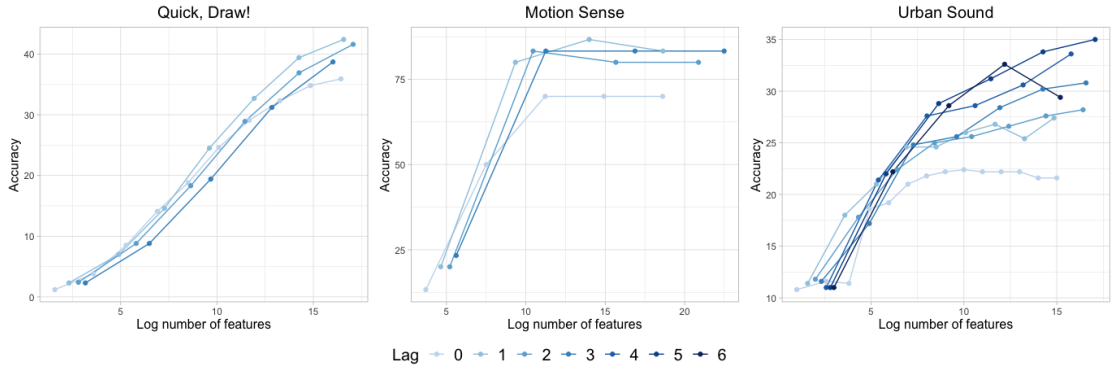


Figure 2.20 – Test accuracy for different lags.

The Motion Sense and Urban Sound datasets do not require a lot of computational resources, as they have a reasonable size (a few hundred samples for Motion Sense and thousand for Urban Sound—see Table 2.2) and a small number of classes. On the other hand, the Quick, Draw! recognition task, which is comprised of 340 classes, is more involved and requires an elaborate algorithm as well as a significant number of training samples. Therefore, more data will be used than in the previous sections: 12 185 600 training samples and 87 040 validation samples.

For the Quick, Draw! dataset, our results are compared to a Kaggle competition (Kaggle.com, 2018). In this competition, 1 316 teams competed for a prize of 25 000$. State-of-the-art deep convolutional networks, such as MobileNet or ResNet, trained with several millions of samples, were among the best competitors. Teams on the podium used ensembles of such networks. These winning methods require a lot of computing resources and are specific to images. The metric used in the competition was the mean average precision, defined as follows. Denoting by $\{y_1, \dots, y_{n_{\text{test}}}\}$ the test set labels, three ranked predictions are made for each sample, denoted by

$$\{(\hat{y}_1^1, \hat{y}_1^2, \hat{y}_1^3), \dots, (\hat{y}_{n_{\text{test}}}^1, \hat{y}_{n_{\text{test}}}^2, \hat{y}_{n_{\text{test}}}^3)\},$$

where $\hat{y}_i^1$ is the class with the largest probability, $\hat{y}_i^2$ the second largest, and so on. Then, the mean average precision at rank 3 is defined by

$$MAP_3 = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \sum_{j=1}^{3} \frac{\mathbb{1}\{\hat{y}_i^j = y_i\}}{j}.$$

Mean average precision is computed by the competition platform on 91% of a test set of 112 200 samples. The small neural network used in Section 2.4.2 is enhanced by using ReLU activation functions and adding three hidden layers with 256 nodes. The network is trained during 300 epochs with an Adam optimizer. At each epoch, it is trained on 609 280 samples randomly selected among the 12 185 600 training samples. The best team obtains a MAP_3 of 95% whereas this small network combined with signature features truncated at order 6 already achieves 54%. The winners use an ensemble of several dozens of deep neural networks, trained on 49 million samples. This kind of architectures requires considerably more computational capacities than ours.

For the Urban Sound dataset, state-of-the-art results are obtained by Ye et al. (2017). The authors combine feature extraction with a mixture of expert models and achieve 77.36 % accuracy, defined by (2.10). The feature extraction step is specific to sound data and is based on several ingredients, such as whitened spectrogram, dictionary learning, soft-thresholding, recurrence

quantification analysis, and so on. These crafting operations make use of a lot of domain-specific knowledge and cannot be extended easily to other applications. On the other hand, it is clear from Figure 2.10 that a random forest classifier performs well with signature features. Therefore, its hyperparameters are tuned with a lead-lag embedding, a lag of 5, and a signature truncated at order 5. An accuracy of 70 % is obtained with 460 trees with a maximum depth of 30 and in which 500 random features are considered at each split.

Finally, Malekzadeh et al. (2019) tackle the problem of mobile sensor data anonymization. They build a deep neural network architecture that preserves user privacy but still detects the activity performed. The architecture is built on autoencoders combined with a multi-objective loss function. There is a trade-off between activity recognition and privacy but good activity recognition results are achieved. Performance of the classifier of Malekzadeh et al. (2019) is measured with the average F_1 score, defined as follows. Assume there are C different classes, and denote by $(y_1, \dots, y_{n_{\text{test}}})$ the test labels, and by $(\hat{y}_1, \dots, \hat{y}_{n_{\text{test}}})$ the predicted ones. Then, the F_1 score is defined by

$$F_1 = \frac{1}{C} \sum_{c=1}^{C} \frac{2 \cdot \text{Precision}_c \cdot \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c},$$

where

$$\text{Precision}_c = \frac{\sum_{i=1}^{n} \mathbb{1}\{\hat{y}_i = y_i = c\}}{\sum_{i=1}^{n} \mathbb{1}\{\hat{y}_i = c\}} \text{ and } \text{Recall}_c = \frac{\sum_{i=1}^{n} \mathbb{1}\{\hat{y}_i = y_i = c\}}{\sum_{i=1}^{n} \mathbb{1}\{y_i = c\}}.$$

Malekzadeh et al. (2019) report an average F_1 score above 92%, while the signature truncated at order 3 and combined with a XGBoost classifier achieves a F_1 score of 93.5%. These two scores are close, but the signature approach is computationally much less demanding.

Despite not being tuned to a specific application, the combination signature + generic algorithm achieves results close to the state-of-the-art in several domains, while requiring few computing resources and no domain-specific knowledge. Indeed, it takes approximately 52 seconds to compute the signature at order 3 of 68 000 Quick, Draw! samples on one core of a laptop, which results in 0.0008 second per sample. Besides, signature computations can be parallelized, making the approach scalable to big datasets. Lastly, the signature method achieves its best results for the high dimensional Motion Sense dataset, which suggests that it is especially relevant for multidimensional streams.

2.7 Conclusion

The signature method is a generic way of creating a feature set for sequential data and has recently caught the machine learning community's attention. Indeed, it yields results competitive with state-of-the-art methods, while being generic, computationally efficient, and able to handle multidimensional series. One of its appealing properties is that it captures geometric properties of the process underlying the data and does not depend on a specific basis. In this paper, its use in a learning context, and several of its successful applications have been reviewed. The use of signatures relies on representing discretely sampled data as continuous paths, a mechanism called embedding. In the literature, authors use various embeddings, without any systematic comparison. We have compared different common embeddings and concluded that the lead-lag seems to be systematically better, whatever the algorithm or dataset used. Moreover, we have pointed out that the signature of the whole path appears to contain as much information as the signature of subpaths, therefore encoding both global and local properties of the input stream.

Our study is a first step towards understanding how signature features can be used in statistics, and a lot of issues remain open, both practical and theoretical. First, it would be of great

interest to understand the theoretical statistical properties of embeddings, in particular, to explain the good performance of the lead-lag path. Moreover, in Section 2.2.2, we have seen that signature features may be combined with feedforward, recurrent, or convolutional neural networks. For each of these architectures, the point of view on signature features is different: they are considered respectively as a vector, a temporal process, or an image. A more detailed understanding of these representations would be valuable. Finally, it could be worth investigating the robustness of the signature method when the truncation order becomes large. Indeed, Figures 2.9, 2.11, and 2.10 suggest that the signature may be robust to dimension: the accuracy curves do not decrease when the number of features becomes large, even when a nearest neighbor algorithm is used with more than a hundred thousand features. This phenomenon may deserve a more in-depth study.

Bibliography

Bagnall, A., Dau, H. A., Lines, J., Flynn, M., Large, J., Bostrom, A., Southam, P., and Keogh, E. (2018). The uea multivariate time series classification archive, 2018. *arXiv:1811.00075*.

Bagnall, A., Lines, J., Bostrom, A., Large, J., and Keogh, E. (2017). The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, *31*, 606–660.

Berndt, D. J., and Clifford, J. (1996). Finding patterns in time series: a dynamic programming approach. *Advances in knowledge discovery and data mining* (pp. 229–248). American Association for Artificial Intelligence.

Box, G. E., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2015). *Time series analysis: forecasting and control. 5th edition.* Wiley.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*, 5–32.

Chang, J., Duffield, N., Ni, H., and Xu, W. (2017). Signature inversion for monotone paths. *Electronic Communications in Probability*, *22*, 1–11.

Chen, K.-T. (1958). Integration of paths—a faithful representation of paths by non-commutative formal power series. *Transactions of the American Mathematical Society*, *89*, 395–407.

Chen, T., and Guestrin, C. (2016). XGBoost: a scalable tree boosting system. *Proceedings of the 22nd International Conference on Knowledge Discovery and Data Mining*, 785–794.

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

Chevyrev, I., and Lyons, T. (2016). Characteristic functions of measures on geometric rough paths. *The Annals of Probability*, *44*, 4049–4082.

Chevyrev, I., and Oberhauser, H. (2018). Signature moments to characterize laws of stochastic processes. *arXiv:1810.10971*.

Chollet, F. et al. (2015). Keras. https://keras.io

Fawaz, H. I., Forestier, G., Weber, J., Idoumghar, L., and Muller, P.-A. (2019). Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, *33*, 917–963.

Ferraty, F., and Vieu, P. (2006). *Nonparametric functional data analysis: theory and practice.* Springer.

Flint, G., Hambly, B., and Lyons, T. (2016). Discretely sampled signals and the rough Hoff process. *Stochastic Processes and their Applications*, *126*, 2593–2614.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *The Annals of Statistics*, *29*, 1189–1232.

Friz, P. K., and Victoir, N. B. (2010). *Multidimensional stochastic processes as rough paths: theory and applications* (Vol. 120). Cambridge University Press.

Google. (2017). The quick, draw! dataset. https://github.com/googlecreativelab/quickdraw-dataset

Graham, B. (2013). Sparse arrays of signatures for online character recognition. *arXiv:1308.0371*.

Greven, S., Crainiceanu, C., Caffo, B., and Reich, D. (2011). Longitudinal functional principal component analysis. *Recent advances in functional data analysis and related topics* (pp. 149–154). Springer.

Gyurkó, L. G., Lyons, T., Kontkowski, M., and Field, J. (2014). Extracting information from the signature of a financial data stream. *arXiv:1307.7244*.

Hairer, M. (2013). Solving the KPZ equation. *The Annals of Mathematics*, *178*, 559–664.

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

Hamilton, J. D. (1994). *Time series analysis.* Princeton University Press.

Kaggle.com. (2018). Quick, draw! doodle recognition challenge. https://www.kaggle.com/c/quickdraw-doodle-recognition

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

Kokoszka, P., Oja, H., Park, B., and Sangalli, L. (2017). Special issue on functional data analysis. *Econometrics and statistics*, *1*, 99–100.

Kormilitzin, A., Saunders, K., Harrison, P., Geddes, J., and Lyons, T. (2016). Application of the signature method to pattern recognition in the cequel clinical trial. *arXiv:1606.02074*.

Lai, S., Jin, L., and Yang, W. (2017). Online signature verification using recurrent neural network and length-normalized path signature descriptor. *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, *1*, 400–405.

Levin, D., Lyons, T., and Ni, H. (2013). Learning from the past, predicting the statistics for the future, learning an evolving system. *arXiv:1309.0260*.

Liu, M., Jin, L., and Xie, Z. (2017). Ps-lstm: capturing essential sequential online information with path signature and lstm for writer identification. *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, *1*, 664–669.

Lütkepohl, H. (2005). *New introduction to multiple time series analysis*. Springer.

Lyons, T., Caruana, M., and Lévy, T. (2007). *Differential equations driven by rough paths* (Vol. 1908). Springer.

Lyons, T., Ni, H., and Oberhauser, H. (2014). A feature set for streams and an application to high-frequency financial tick data. *Proceedings of the 2014 International Conference on Big Data Science and Computing*, 5.

Lyons, T., and Xu, W. (2018). Inverting the signature of a path. *Journal of the European Mathematical Society*, *20*, 1655–1687.

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

Lyons, T. J., and Xu, W. (2017). Hyperbolic development and inversion of signature. *Journal of Functional Analysis*, *272*, 2933–2955.

Malekzadeh, M., Clegg, R. G., Cavallaro, A., and Haddadi, H. (2018). Protecting sensory data against sensitive inferences. *Proceedings of the 2018 Workshop on Privacy by Design in Distributed Systems*, 1–6.

Malekzadeh, M., Clegg, R. G., Cavallaro, A., and Haddadi, H. (2019). Mobile sensor data anonymization. *Proceedings of the 2019 International Conference on Internet-of-Things Design and Implementation*, 49–58.

Park, S. Y., and Staicu, A.-M. (2015). Longitudinal functional data analysis. *Stat*, *4*, 212–226.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Ramsay, J. O., and Silverman, B. W. (2005). *Functional data analysis. 2nd edition.* Springer.

Reizenstein, J., and Graham, B. (2020). Algorithm 1004: the iisignature library: efficient calculation of iterated-integral signatures and log signatures. *ACM Transactions on Mathematical Software*.

Salamon, J., Jacoby, C., and Bello, J. P. (2014). A dataset and taxonomy for urban sound research. *Proceedings of the 2014 International Conference on Multimedia*, 1041–1044.

Shumway, R. H., and Stoffer, D. S. (2017). *Time series analysis and its applications: with r examples*. Springer.

Wilson-Nunn, D., Lyons, T., Papavasiliou, A., and Ni, H. (2018). A path signature approach to online arabic handwriting recognition. *Proceedings of the 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)*, 135–139.

Yang, W., Jin, L., and Liu, M. (2015). Chinese character-level writer identification using path signature feature, dropstroke and deep cnn. *Proceedings of the 13th International Conference on Document Analysis and Recognition (ICDAR)*, 546–550.

Yang, W., Jin, L., and Liu, M. (2016a). DeepWriterID: An end-to-end online text-independent writer identification system. *IEEE Intelligent Systems*, *31*, 45–53.

Yang, W., Lyons, T., Ni, H., Schmid, C., Jin, L., and Chang, J. (2017). Developing the path signature methodology and its application to landmark-based human action recognition. *arXiv:1707.03993*.

Ye, J., Kobayashi, T., and Murakawa, M. (2017). Urban sound event classification based on local and global features aggregation. *Applied Acoustics*, *117*, 246–256.

Chapter 3

A Generalised Signature Method for Multivariate Time Series Feature Extraction

The 'signature method' refers to a collection of feature extraction techniques for multivariate time series, derived from the theory of controlled differential equations. There is a great deal of flexibility as to how this method can be applied. On the one hand this flexibility allows the method to be tailored to specific problems, but on the other hand can make precise application challenging. This paper makes two contributions. First, the variations on the signature method are unified into a general approach, the *generalised signature method*, of which previous variations are special cases. A primary aim of this unifying framework is to make the signature method more accessible to any machine learning practitioner, whereas it is now mostly used by specialists. Second, and within this framework, we derive a canonical collection of choices that provide a domain-agnostic starting point. We derive these choices as a result of an extensive empirical study on 26 datasets, and go on to show competitive performance against current benchmarks for multivariate time series classification. Finally, to ease practical application, we make our techniques available as part of the open source `sktime` project.

Contents

3.1 Introduction

A multivariate time series is obtained by observing d quantities evolving with time, which can be written as an array $\mathbf{x} = (x_1, \ldots, x_n)$, where n is the length of the series, and $x_i \in \mathbb{R}^d$ for each $i \in \{1, \ldots, n\}$. These data are common in various fields (finance, health, energy...) and offer several specific challenges: they are often highly dimensional, as both the number of channels d and the length of the series n may be large, the values x_i are correlated, and the different channels may interact. Finally, the inputs may be of different length and the data may be irregularly sampled.

One approach is to construct models that directly accept some of these issues; for example recurrent neural networks handle correlated inputs with varying lengths. A second option is to use feature extraction techniques, which normalise the data so that other techniques may then be applied. Methods such as the shapelet transform Ye and Keogh (2009), Grabocka et al. (2014), and Kidger et al. (2020b), Gaussian process adapters Li and Marlin (2016), Futoma et al. (2017), and Moor et al. (2020), and in particular the signature method Levin et al. (2013), all fit into this category.

The approach taken by the signature method, coming from rough path theory (Lyons et al., 2007; Friz and Victoir, 2010), is to interpret a multivariate time series as a discretisation of an underlying continuous path. The *signature transform*, also known as the *path signature* or *signature*, can then be applied, which produces a vector of real-valued features that are known to characterise the path.

Benefits of the signature method include: a high degree of flexibility, making it possible to customise the method to specific datasets; strong theoretical guarantees; an interpretable feature set; ease of handling irregularly sampled and/or partially observed data; and it being well-defined for some highly irregular processes such as ARMA, Gaussian processes or even Brownian motion. Also, signature features do not need to learned, which can make them particularly effective on (but not limited to) low sample datasets.

The flexibility of the signature method has made it possible to be tailored to specific applications and achieve state-of-the-art performance in wide range of problem domains, such as handwriting recognition (Wilson-Nunn et al., 2018; Yang et al., 2016b), action recognition Yang et al. (2016a), Yang et al. (2017), and medical time series prediction tasks (Morrill et al., 2019; Morrill et al., 2020c). However, this flexibility comes at the cost of additional complexity in the model search space.

To the best of our knowledge, no comprehensive studies exist that collate and combine the most common method variations found in the literature and assemble them under a common mathematical framework. Additionally, no baseline signature model has ever been tested against other time series classification baselines. Our goal will be to address both of these issues, alongside the development of an open source implementation, so as to make the methods more accessible to a wider audience.

Contributions We introduce a *generalised signature method* that contains the many existing variations as special cases. In doing so we are able to understand their conceptual groupings into

what we term *augmentations*, *windows*, *transforms* and *rescalings*. This involves a comprehensive review of the existing variations across the literature. By understanding their commonality, we are then able to combine different variations, and propose new options that fit into this framework.

We go on to examine which choices within this framework are most important to success by performing an extensive empirical study across 26 datasets. To the best of our knowledge this is the first study of this type.

In doing so, we are then able to produce a canonical signature pipeline. This represents a domain agnostic starting point that may then be adapted for the task at hand. We show that the performance of this canonical pipeline is comparable to current state-of-the-art classifiers for multivariate time series classification, including deep recurrent and convolutional neural networks. This has led to the implementation of this generalised approach in the open source [redacted] package.

3.2 Context

3.2.1 Background theory

We begin with a few mathematical definitions necessary throughout the article.

Definition 3.1. *Let $d \in \mathbb{N}$, we denote the space of time series over $\mathbb{R}^d$ as*

$$\mathcal{S}(\mathbb{R}^d) = \{(x_1, \ldots, x_n) \,|\, x_i \in \mathbb{R}^d, n \in \mathbb{N}, n \geq 1\}.$$

If $d = 1$, then $\mathbf{x}$ is a univariate time series, whereas if $d > 1$, $\mathbf{x}$ is a multivariate time series. Given $\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{S}(\mathbb{R}^d)$, n is called the length of $\mathbf{x}$ and d its dimension or number of channels. We assume that in addition to the array of values $\mathbf{x} \in \mathcal{S}(\mathbb{R}^d)$, we have access to a vector of increasing time stamps $\mathbf{t} = (t_1, \ldots, t_n)$. If the data is regularly sampled, then $\mathbf{t}$ can be set to $\mathbf{t} = (1, \ldots, n)$, which will often be the case.

We consider a *dataset* to be a collection of such samples. Note that the time stamps $\mathbf{t}$ for each sample may be different, and the sample lengths n can vary. That is, we accept varying length and irregular sampling without modification. We are now in a position to define the signature of a time series.

Definition 3.2. *Let $\mathbf{x} \in \mathcal{S}(\mathbb{R}^d)$ and $\mathbf{t} = (t_1, \ldots, t_n)$ its associated timestamps. Let $X = (X_t^1, \ldots, X_t^d)_{t \in [t_1, t_n]}$ be a piecewise linear interpolation of $\mathbf{x}$ such that for any $i \in \{1, \ldots, n\}$, $X_{t_i} = x_i$. Then the depth-N signature transform of $\mathbf{x}$ is the vector defined by*

$$\mathrm{Sig}^N(\mathbf{x}) = \left(\{S(\mathbf{x})^{(i)}\}_{i=1}^d, \{S(\mathbf{x})^{(i,j)}\}_{i,j=1}^d, \ldots, \{S(\mathbf{x})^{(i_1,\ldots,i_N)}\}_{i_1,\ldots,i_N=1}^d\right) \in \mathbb{R}^{\frac{d^{N+1}-1}{d-1}}$$

where for any $(i_1, \ldots, i_k) \in \{1, \ldots, d\}^k$,

$$S(\mathbf{x})^{(i_1,\ldots,i_k)} = \int_{t_1 \leq u_1 < \cdots < u_k \leq t_n} \cdots \int \mathrm{d}X_{u_1}^{i_1} \ldots \mathrm{d}X_{u_k}^{i_k} \in \mathbb{R}.$$

While this definition may seem somewhat technical, there are several intuitions that can be made with regard to the signature features. We present a geometric interpretation of the first two levels of the signature and log-signature in Figure 3.1. The depth-1 terms, $S(\mathbf{x})^{(i)}$, equate to the displacement of the path over the interval in the ith coordinate, denoted by ΔX^i in Figure 3.1. The depth-2 terms, $S(\mathbf{x})^{(i,j)}$, have interpretations in areas generated over the interval.

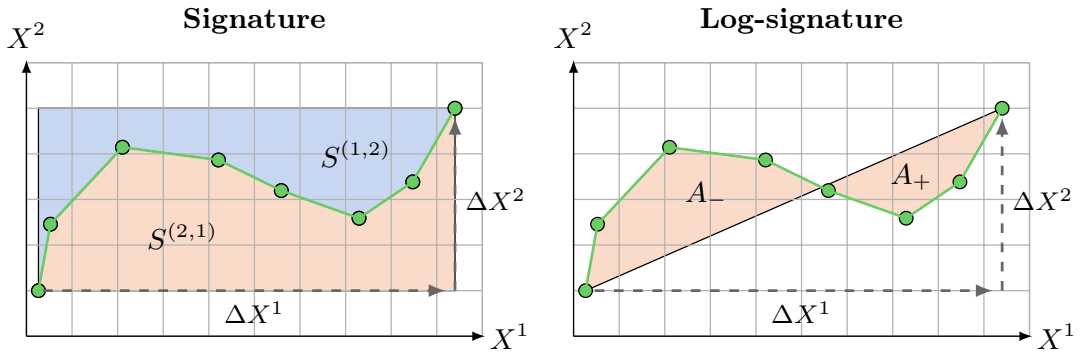


Figure 3.1 – Geometric depiction of the depth-2 signature and log-signature. The depth-1 term of both transforms equate to the displacements of the path over the interval in each coordinate, these being $\Delta X^{1,2}$. **Left:** The signature. Depth-2 terms $S^{(1,2)}, S^{(2,1)}$ correspond to the areas of the blue and orange regions respectively. **Right:** The log-signature. Only one depth-2 term which is given by the signed area $A_+ - A_-$. This is known as the *Lévy area* of the path.

From a statistical point of view, the signature can be thought of as the equivalent of a moment-generating function for time series. Let Z be a random variable, then the moment-generating function of Z is the function

$$t \mapsto \mathbb{E}[e^{tz}] = \sum_{k=0}^{\infty} \frac{t^k}{k!}\mathbb{E}[Z^k]$$

and, if well-defined, it characterizes the distribution of Z. Assume that X is a random time series (that is a stochastic process), its signature now has the same properties as a moment-generating function: the powers of Z are replaced by integrals of products of coordinates and Chevyrev and Lyons (2016) show that the expected signature characterizes the law of X.

Moreover, we have the following two properties that make the signature a good feature set in a machine-learning context—precise statements may be found in Kidger et al. (2019, Appendix A).

Uniqueness Hambly and Lyons (2010) show that under mild assumptions, the full collection of features $\mathrm{Sig}(\mathbf{x}) = \lim_{N\to\infty} \mathrm{Sig}^N(\mathbf{x})$ uniquely determines $\mathbf{x}$ up to translations and reparametrizations.

Universal nonlinearity Linear functionals on the signature are dense in the set of functions on $\mathbf{x}$. Suppose we wish to learn the function f that maps data $\mathbf{x}$ to labels y, the universal nonlinearity property states that, under some assumptions, for any $\varepsilon > 0$, there exists a linear function L such that

$$\|f(\mathbf{x}) - L\big(\mathrm{Sig}(\mathbf{x})\big)\| \leq \varepsilon. \tag{3.1}$$

Note that contrary to Fourier or wavelet basis, signatures provide a natural basis for functions of the time series rather than for the time series itself—(3.1) concerns $f(\mathbf{x})$ and not $\mathbf{x}$. In the context of time series classification this shift of perspective is particularly well-suited since the object of interest is not the time series itself but its link to a label.

From a computational point of view, computing the depth-N signature of a time series $\mathbf{x} \in \mathcal{S}(\mathbb{R}^d)$ of length n has a complexity of $\mathcal{O}(nd^N)$, which can be done with high performance software (Reizenstein and Graham, 2020; Kidger and Lyons, 2020). The size of the depth-N signature is $(d^{N+1} - 1)/(d - 1)$ so the memory cost is independent of the series length n, which is a huge

advantage when dealing with high frequency time series. Note that small values of N already show a good performance—for example $N = 3$ in the baseline algorithm—so the exponential dependence on N is not a huge computational bottleneck.

Logsignature transform The signature contains some redundant information: for example we can see in the left panel of Figure 3.1 that the sum of the blue and orange areas is equal to the product of displacements $\Delta X^1 \Delta X^2$:

$$S(\mathbf{x})^{(1,2)} + S(\mathbf{x})^{(2,1)} = S(\mathbf{x})^{(1)} S(\mathbf{x})^{(2)}.$$

The *logsignature transform* is essentially the signature with these redundancies removed. For example, the logsignature encodes the blue and orange areas from the left panel with the orange signed area in the right panel. However, the logsignature does not have a universal nonlinearity property such as (3.1). We refer the reader to Morrill et al. (2020b) or Liao et al. (2019, Section 2) for a precise definition of the logsignature.

A pedagogical introduction to the background theory of signatures is Lyons et al. (2007), whilst a comprehensive textbook is Friz and Victoir (2010). For introductions to the signature method, we recommend Kidger et al. (2019, Appendix A) and Chevyrev and Kormilitzin (2016a).

3.2.2 Related work

The signature transform has been used in a wide range of applications in machine learning predictive tasks. For example, as mentioned in the introduction, the signature has been used as a feature extraction layer in classifiers for both Arabic (Wilson-Nunn et al., 2018) and Chinese (Yang et al., 2016b) handwriting recognition. Similarly, it was successfully used in human action recognition by Li et al. (2017), Yang et al. (2017), and Liao et al. (2019) and in the medical domain as part of the top performing model at the Physionet 2019 challenge for prediction of sepsis (Reyna et al., 2020; Morrill et al., 2019; Morrill et al., 2020c). Other applications involve finance (Lyons et al., 2014; Perez Arribas, 2018), mental health (Kormilitzin et al., 2017; Arribas et al., 2018), and emotion recognition (Wang et al., 2019; Wang et al., 2020).

In almost all these applications, the method has been utilised in different ways. Many authors consider transformations of the input time series before application of the signature (Levin et al., 2013; Flint et al., 2016; Lyons and Oberhauser, 2017; Yang et al., 2017; Liao et al., 2019; Kidger and Lyons, 2020; Wu et al., 2021). People have also explored different windows over which the signature transform should be taken, so as to extract information over different scales (Yang et al., 2017; Kidger et al., 2019). Additionally a choice must be made between the signature and logsignature transforms, as must choices for the scaling of the terms in the signature (Chevyrev and Kormilitzin, 2016a; Lai et al., 2017).

The differences between some of these choice have been shown by Fermanian (2021) to significantly impact the performance of the methodology. However this study used a small collection of datasets and considered only some of the most common variations that exist in the literature. There is therefore a need for a comprehensive study and unification of all these different choices.

3.3 The generalized signature method

In this section we collate the modifications to the signature transform that have been proposed in signature literature to date. We will show that each can be categorised into one of the following groups:

- **Augmentations** These describe the transformation of a time series into one or more new series, in order to return different information in the signature features and deal with dimensionality issues.
- **Windows** Splitting the time series over different subsequences (or windows), so that signatures may be applied locally.
- **Transform** The choice between the signature or the logsignature transform.
- **Rescaling** Ways of normalising the terms in the signature.

We then go on to show that these groupings can themselves be synergised into a single mathematical framework that we term *the generalised signature method*. For clarity, we will begin by discussing each of these individually, and then afterwards show how they may be combined.

As before, assume that we observe some collection of sequences $\mathbf{x} \in \mathcal{S}(\mathbb{R}^d)$ with timestamps $\mathbf{t} \in \mathcal{S}(\mathbb{R})$.

3.3.1 Augmentations

We define an *augmentation* to be a transform of an initial sequence $\mathbf{x} \in \mathcal{S}(\mathbb{R}^d)$ into one or several new sequences. Augmentations have several different uses:

1. Remove the signature invariance to translation and/or reparametrization.
2. Lower the dimension d of the time series, so that higher orders of the signature are reachable—recall that the depth-N signature is of size $\mathcal{O}(d^N)$.
3. Preprocess the time series prior to the signature map so that information is more easily extracted.

For some $e, p \in \mathbb{N}$, we define an augmentation as a map

$$\phi \colon \mathcal{S}(\mathbb{R}^d) \to \mathcal{S}(\mathbb{R}^e)^p.$$

There are many pre-signature operations which have been proposed in the literature, and which we categorise as augmentations. We refer the reader to Appendix B.1 for full details of the many such operations proposed in the literature, but will focus on several important examples here.

Let us give some examples in the first group of sensitivity-inducing augmentations. For any vector of increasing timestamps $\mathbf{t}$, we call *time augmentation* (Levin et al., 2013) the operation $\phi_{\mathbf{t}} : \mathcal{S}(\mathbb{R}^d) \mapsto \mathcal{S}(\mathbb{R}^{d+1})$ defined by

$$\phi_{\mathbf{t}}(\mathbf{x}) = \big((t_1, x_1), \dots, (t_n, x_n)\big). \tag{3.2}$$

This transformation, which basically consists in adding the timestamps as an extra coordinate, has two key properties: it guarantees the uniqueness of the signature (Hambly and Lyons, 2010) and it adds information about the parametrization of the time series.

Another example is the basepoint augmentation (Kidger and Lyons, 2020), which is the map $\phi^b : \mathcal{S}(\mathbb{R}^d) \mapsto \mathcal{S}(\mathbb{R}^d)$ defined by

$$\phi^b(\mathbf{x}) = (0, x_1, \dots, x_n), \tag{3.3}$$

which simply adds a zero at the beginning of the time series—note that this zero could also be put at the end. This transformation makes the signature sensitive to translations of the time series. The invisibility-reset transformation (Yang et al., 2017; Wu et al., 2021) also adds translation sensitivity, but does so by increasing the dimension.

In the second group of augmentations for dimensionality reduction, we consider random projections (Lyons and Oberhauser, 2017), which consist in applying multiple random linear

maps to the time series, or coordinate projections, which project along (multiple subsets of) the coordinate axes.

In the third group, the lead-lag augmentation (Chevyrev and Kormilitzin, 2016a; Flint et al., 2016; Yang et al., 2017) captures the quadratic variation by transforming the time series to

$$\phi(\mathbf{x}) = \big((x_1, x_1), (x_2, x_1), (x_2, x_2), (x_3, x_2), (x_3, x_3), \dots, (x_n, x_n)\big) \in \mathcal{S}(\mathbb{R}^{2d}).$$

Another important example of this kind of augmentation are the stream preserving neural networks of Kidger et al. (2019), who learn a map ϕ from the data. They map a time series in $\mathbb{R}^d$ to another series in $\mathbb{R}^e$ by setting ϕ to some neural network, typically either convolutional or recurrent. We extend this idea by defining the multi-headed stream preserving augmentation, which simply consists in stacking p such transformations. To our knowledge, this is the first time that such learned augmentations are compared to 'handcrafted' ones such as time, basepoint and lead-lag augmentations.

Importantly, these various augmentations may be combined together. For example, in order to add sensitivity to both parametrization and translation, the time and basepoint augmentations may be combined: first apply the time augmentation, which gives a sequence $\phi_{\mathbf{t}}(x) \in \mathcal{S}(\mathbb{R}^{d+1})$, and then the basepoint augmentation, which yields

$$\phi^b \circ \phi_{\mathbf{t}}(\mathbf{x}) = \big((0,0), (t_1, x_1), \dots, (t_n, x_n)\big). \tag{3.4}$$

3.3.2 Windows

The second step is to choose a windowing operation. Much like the window functions used with a short time Fourier transform, this localises the signature computation to extract information over particular time intervals.

We define a window to be a map

$$W \colon \mathcal{S}(\mathbb{R}^e) \to \mathcal{S}(\mathbb{R}^e)^w,$$

for some $w \in \mathbb{N}$. In short, W maps a time series in $\mathbb{R}^e$ into w new time series in the same space. The simplest possible window is the global window, defined by

$$W(\mathbf{x}) = (\mathbf{x}), \tag{3.5}$$

which outputs the time series itself. To get finer-scale information, we consider three other types of windows: sliding, expanding and hierarchical dyadic windows. For $\mathbf{x} = (x_1, \dots, x_n) \in \mathcal{S}(\mathbb{R}^e)$ and $1 \le i \le j \le n$, let $\mathbf{x}_{i:j} = (x_i, \dots, x_j) \in \mathcal{S}(\mathbb{R}^e)$ be a subsequence of $\mathbf{x}$. Then, a sliding window of length ℓ and step l is defined by

$$W(\mathbf{x}) = (\mathbf{x}_{1:\ell}, \mathbf{x}_{l+1:l+\ell}, \mathbf{x}_{2l+1:2l+\ell}, \dots),$$

and an expanding window of initial length ℓ and step l by

$$W(\mathbf{x}) = (\mathbf{x}_{1:\ell}, \mathbf{x}_{1:l+\ell}, \mathbf{x}_{1:2l+\ell}, \dots).$$

The expanding window produces time series of increasing length, and is analogous to the history processes of stochastic analysis whereas the sliding window produces time series of fixed length but shifted in time.

Finally we consider a hierarchical dyadic window, which captures information at different scales. Let $q \in \mathbb{N}$ be fixed and assume for simplicity that 2^{q-1} divides n. Then, the hierarchical

dyadic window of depth q consists of q sliding windows $W^1, \dots, W^q$, where W^i has length and step both equal to $n2^{-(i-1)}$. This yields $w = 2^q - 1$ time series of length n, $n/2$, $n/4$, $\dots$, $n/2^{q-1}$. The larger the value of q, the finer the scale on which the information is extracted. If the other window functions are analogous to the short time Fourier transform, then hierarchical dyadic windows are analogous to the multi-scale nature of wavelets.

3.3.3 The signature and logsignature transforms

Central to the signature methodology is of course the signature transform itself. Two choices must be made; whether to use the signature or logsignature transform, and what depth to calculate the transform to—that is, what depth N in Definition 3.2 to use. Choosing a logsignature lowers the feature vector dimension at the cost of loosing linear approximation properties. There is no consensus on which one should be favored for a machine learning task.

3.3.4 Rescaling

The depth-k term in the signature is of size $\mathcal{O}(1/k!)$. Typically, rescaling these terms to $\mathcal{O}(1)$ will aid in subsequent learning procedures. To this end, we can apply *pre-signature* scaling whereby we scale the path before signature computation, or *post-signature* where we scale the signature terms themselves. Specifics on how this is done in practice are given in Appendix B.2.

3.3.5 Putting the pieces together

Let $\phi\colon \mathcal{S}(\mathbb{R}^d) \to \mathcal{S}(\mathbb{R}^e)^p$ be the final augmentation function, $\phi : \mathbf{x} \mapsto \big(\phi^1(\mathbf{x}), \dots, \phi^p(\mathbf{x})\big)$, which can be a composition of augmentations such as (3.4). Let $W : \mathcal{S}(\mathbb{R}^e) \to \mathcal{S}(\mathbb{R}^e)^w$, $W : \mathbf{x} \mapsto \big(W^1(\mathbf{x}), \dots, W^w(\mathbf{x})\big)$, be the window map, such that $W^j(\mathbf{x}) \in \mathcal{S}(\mathbb{R}^e)$ for any $1 \le j \le w$. Let S^N represent either the signature or logsignature transform of depth N. Let ρ_{pre} and ρ_{post} represent the different types of features rescaling. Then given an input $\mathbf{x} \in \mathcal{S}(\mathbb{R}^d)$, the general framework for extracting signature features is given by the collection of

$$\mathbf{z}_{i,j} = (\rho_{\text{post}} \circ S^N \circ \rho_{\text{pre}} \circ W^j \circ \phi^i)(\mathbf{x}) \tag{3.6}$$

over all $i \in \{1, \dots, p\}$, $j \in \{1, \dots, w\}$. We refer to the procedure of computing $\mathbf{x} \mapsto (\mathbf{z}_{i,j})$ as the *generalised signature method*.

This final procedure is a little involved, but is simply a combination of different elementary operations used to impact the final feature set. The overall procedure now offers a degree of flexibility and generality which has, to our knowledge, never been achieved for signature methods.

The collection of features $(\mathbf{z}_{i,j})$ may then be fed into any later machine learning algorithm, which will depend on the application. In general, the $\mathbf{z}_{i,j}$ will be stacked together and considered as a vector. However, if one wants to use a sequential algorithm such as a recurrent network, it is possible to turn the features $\mathbf{z}_{i,j}$ into a sequence by choosing a sliding or expanding window. Indeed, these windows induce an ordering in the features: the terms $\mathbf{z}_{i,1}$ will correspond to the first values of $\mathbf{x}$, the terms $\mathbf{z}_{i,2}$ to the following values, and so on.

3.4 Empirical study

We perform a first-of-its-kind empirical study across 26 datasets to determine the most important aspects of this framework.

3.4.1 Methodology

Datasets The datasets used are the Human Activities and Postural Transitions dataset provided by Reyes-Ortiz et al. (2016), the Speech Commands dataset provided by Warden (2018), and 24 datasets from the UEA time series classification archive, provided by Bagnall et al. (2018). A few datasets from the UEA archive were excluded due to their high number of channels resulting in too large a computational burden.

Baseline We begin by defining a single baseline procedure, representing a simple and straightforward collection of choices for the generalised signature method. This baseline is to take the augmentation ϕ as appending time as defined by (3.2), W as the global window defined by (3.5), have the transform be a signature transform of depth 3, and to use pre-signature scaling of the path. This means that the input features are the collection

$$\mathbf{z} = \mathrm{Sig}^3 \circ \rho_{\mathrm{pre}} \circ \phi_{\mathbf{t}}(\mathbf{x}).$$

Individual variations With respect to this baseline procedure, we then consider, in turn, the groups described in Section 3.3. These were *augmentations*, *windows*, *transform*, and *rescaling*. For each group we modify the baseline by implementing each option in the group one-by-one. Each such variation defines a particular form of the generalised signature method as in (3.6). Example variations are to switch to using a logsignature transform of depth 5, or to use a sliding window instead of a global window. We discuss the precise variations below.

Models On top of every variation, we then consider four different models: logistic regression, random forest, Gated Recurrent Unit (GRU) (Cho et al., 2014), and a residual Convolutional Neural Network (CNN) (He et al., 2016). We test nearly every combination of dataset, variation of the generalised signature method, and model. Different datasets and variations produce different numbers of features $\mathbf{z}_{i,j}$, so to reduce the computational burden we omit those cases for which the number of features is greater than 10^5. Of the 9984 total combinations of dataset, variation, and model, this leaves out 1415 combinations. See Appendix B.3.2 for a break down of the omitted combinations by different cases.

Analysis We define the performance of a variation on a dataset as the best performance across the four models considered, to reflect the fact that different models are better suited for different problems. We then follow the methodology of Demšar (2006), Benavoli et al. (2016), and Ruiz et al. (2020) to compare the variations across the multiple datasets. We first perform a Friedman test to reject the null hypothesis that all methods are equivalent. If it is rejected, we perform pairwise Wilcoxon signed-rank tests to form cliques of not-significant methods, and use critical difference plots to visualize the performance of each signature method.

A critical difference plot shows the different variations ordered by their average rank: for example, in Figure 3.2, the best variation is "Time + Basepoint" with an average rank of 2.5. Then, a thick line indicates that the Wilcoxon test between variations inside the clique is not rejected at significance threshold of 5%, subject to Bonferroni's multiple testing correction. In Figure 3.2 there are two groups of significantly different variations: one with "Basepoint" and "None" and one with all other variations.

We refer the reader to Appendix B.3 for further details on the methodology, such as precise architectural choices, learning rates, and so on.

3.4.2 Results

Due to the large number of variations and datasets considered, we present only the critical difference plots in the main paper. See Appendix B.4 for all the tables of the underlying numerical values.

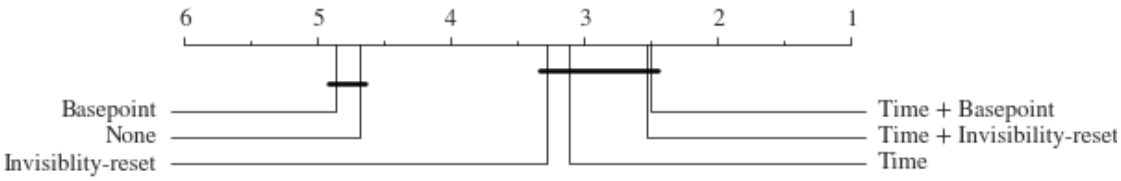


Figure 3.2 – Performance of invariance-removing augmentations.

Augmentations We split the augmentations into two categories. The first category consists of those augmentations which remove the signature's invariance to translation (basepoint augmentation, invisibility-reset augmentation) or reparameterisation (time augmentation). We see in Figure 3.2 that augmenting with time, and either basepoint or invisibility-reset, are both typically important. This is expected; in general a problem need not be invariant to either translation or reparametersiation.

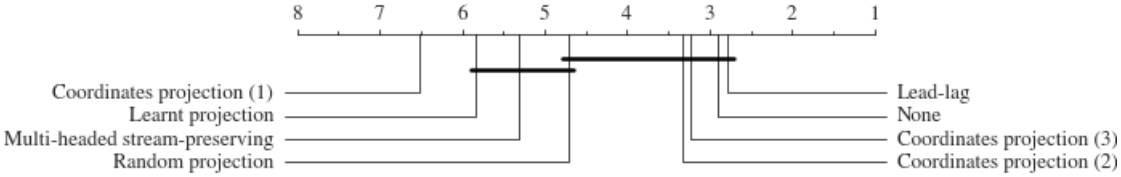


Figure 3.3 – Performance of other augmentations.

The second category consists of those augmentations which either seek to reduce dimensionality or introduce additional information. We see in Figure 3.3 that most augmentations actually do not help matters, except for lead-lag which usually represents a good choice. We posit that the best augmentation is likely to be dataset dependent, so we break this down by dataset characteristics in Table B.5.

Table 3.1 – Average ranks for different augmentations by data type. Lower is better. CP (2) stands for coordinate projections with pairs and LP for Learnt projections.

	Augmentation				
Data type	None	Lead-lag	CP (2)	LP	MHSP
EEG	4.88	4.83	3.13	**2.75**	**2.75**
HAR	2.25	**1.78**	3.50	6.50	6.50
MOTION	2.63	**1.75**	4.50	7.33	5.00
OTHER	2.88	3.92	**2.63**	6.00	5.21

Here we indeed see that there is generally a better choice than doing nothing at all, but that this better choice is dependent on some characteristic of the dataset. For example, learnt projections and multi-headed stream preserving transformations do substantially better on EEG datasets, while lead-lag is better for human action and motion recognition.

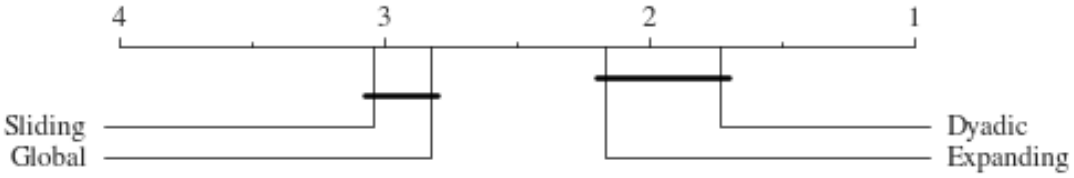


Figure 3.4 – Performance of different windows.

Windows We consider the possibility of global, sliding, expanding, and dyadic windows. The results are shown in Figure 3.4. We see that the dyadic and expanding windows are significantly better than sliding and global windows. The poor performance of sliding windows is a little surprising, but tallies with the observations of Fermanian (2021). This is an important finding, as global and sliding windows tend to be commonly used with signature methods.

Signature versus logsignature transforms We consider the signature and logsignature transforms with depths ranging from 1 to 6. As higher depths always produce more information, we define the performance of the (log)signature transform as the best performance across all depths. With this metric, the signature transform is significantly better than the logsignature transform, with a p-value of 0.01 for the Wilcoxon signed-rank test.

The key results To conclude, these results show that invariance-removing transformations such as time and basepoint augmentations should a priori be used, that the lead-lag performs well but not significantly better than no additional augmentation, and that the hierarchical dyadic window performs significantly better than the sliding and global ones. The poor performance of deep learning approaches for augmentation is also notable and an additional motivation for this work: although slightly technical, the augmentations tailored to the signature transform are a significant addition in a machine learning pipeline and cannot be easily replaced by neural networks.

3.4.3 Further results

See Appendix B.4 for further results, in particular on the running times, the different types of rescaling, augmentations broken down by dataset characteristics, an additional study on signature depth, and the precise numerical results for each individual test considered here.

3.5 The canonical signature pipeline

In this section we define *the canonical signature pipeline*. Using the results from Section 3.4 we evaluate the top performing options over all the datasets so as to provide a domain-agnostic starting point for any dataset, from which other variations can be easily explored. We show that

this pipeline shows competitive performance against traditional benchmarks and even against deep neural networks.

3.5.1 Definition

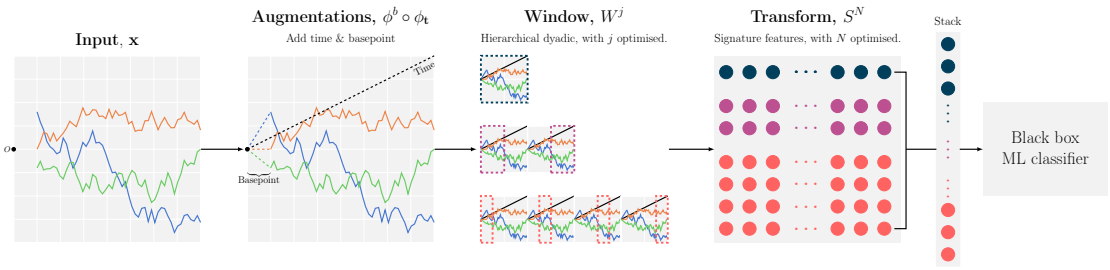


Figure 3.5 – Pictorial representation of the canonical signature pipeline. First, we apply the time and basepoint augmentations to the input paths, then we compute the signature features over dyadic windows, and finally compute the signature features over each dyadic window. These features can now be compiled together and fed into any standard machine learning classifier.

In a nutshell, the pipeline consists in applying the basepoint and time augmentations, a hierarchical dyadic window and a signature transform, which can be written as a particular case of (3.6) as follows. Let W be a hierarchical dyadic window of depth q, $\phi_{\mathbf{t}}$ and ϕ^b be the time and basepoint augmentations, then the canonical signature pipeline may be written as

$$\mathbf{z}_j = S^N \circ W^j \circ \phi^b \circ \phi_{\mathbf{t}}(\mathbf{x}), \quad j \in \{1, \dots, 2^q - 1\}. \tag{3.7}$$

We give a graphical depiction of this in Figure 3.5. Signature and window depths (N, q) must be optimised for the problem (typically via cross-validation). We note that this canonical method may be adapted to the problem at hand in two ways: if the problem is known to be parametrization invariant, as is the case for example for characters recognition, then the time augmentation should not be applied. Moreover, if the problem is translation-invariant, then the basepoint augmentation is not applied. We emphasise that this pipeline does not represent a best option for every application, but is meant to represent a compromise between broad applicability, ease of implementation, computational cost, and good performance.

3.5.2 Performance

We validate the performance of the pipeline against the 26 datasets in the multivariate UEA archive [1]. To our knowledge, the most recent benchmarks for the UEA archive are the results from Ruiz et al. (2020). We compare their results to the canonical signature pipeline with a random forest classifier—see Appendix B.3.3 for more details.

The benchmarks include variants on classical Dynamic Time Wrapping (DTWI, DTWD and DTWA); an ensemble of univariate classifiers, HIVE COTE (Bagnall et al., 2020), known to be highly perfromant in the univariate case; a random shapelet forest (Karlsson et al., 2016), denoted gRSF, and a bag of words based algorithm, MUSE (Schäfer and Leser, 2017); two deep learning methods, TapNet (Wang et al., 2017) and MLCN (Karim et al., 2019). The MLCN

1. This is not to be confused with the UCR archive which is a collection of 128 univariate datasets.

architecture combines long short term memory layers (LSTM) and convolutional layers while TapNet combines 3 blocks: random projections on the different dimensions, convolutional layers and a final attention block to compare candidate time series representations.

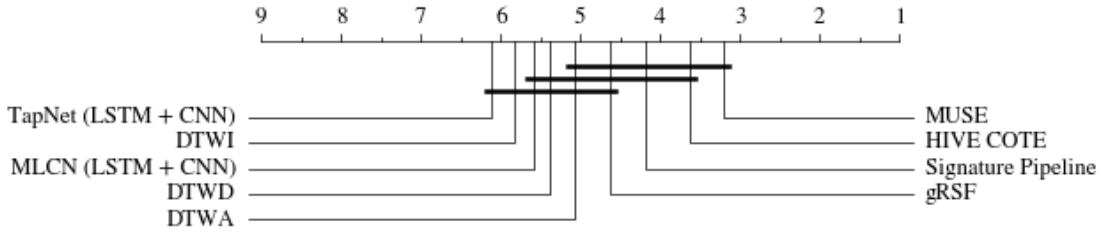


Figure 3.6 – Performance on UEA datasets.

Figure 3.6 shows the critical difference plot of this comparison. The signature pipeline is in the first clique, that is the group of classifiers that achieve the best accuracy while not being significantly different from one another. The two algorithms with a better rank than the signature pipeline are MUSE and HIVE-COTE. It is worth noting that MUSE is very memory intensive—Ruiz et al. (2020) report that it could not finish on 5 of the 26 UEA datasets on a computer with 500GB of memory—whilst HIVE-COTE is an ensemble of several sub-classifiers, and thus has very high training and inference costs. On the other hand, all experiments for the canonical signature pipeline were completed with no memory errors on a computer with less memory, and are significantly faster to run than HIVE-COTE—see Appendix B.4.

The canonical signature pipeline is meant to be a sensible starting point from which the user can propose additional variations following the structure defined in (3.6), but as a standalone classifier this pipeline performs comparably to state-of-the-art classifiers, on the UEA data, whilst being less computationally demanding.

3.6 Conclusion

We introduce a generalised signature method as a framework to capture recently proposed variations on the signature method. We go on to perform a first-of-its-kind extensive empirical investigation as to which elements of this framework are most important for performance in a domain-independent setting. In particular, we highlight the performance of hierarchical dyadic windows and signature-tailored augmentations such as lead-lag, time and basepoint. As a result, we are able to present a canonical signature pipeline that represents a best-practices domain-agnostic starting point, which shows competitive performance against state-of-the-art classifiers for multivariate time series classification.

Bibliography

Arribas, I. P., Goodwin, G. M., Geddes, J. R., Lyons, T., and Saunders, K. E. (2018). A signature-based machine learning model for distinguishing bipolar disorder and borderline personality disorder. *Translational psychiatry*, *8*, 1–7.

Bagnall, A., Dau, H. A., Lines, J., Flynn, M., Large, J., Bostrom, A., Southam, P., and Keogh, E. (2018). The uea multivariate time series classification archive, 2018. *arXiv:1811.00075*.

Bagnall, A. J., Flynn, M., Large, J., Lines, J., and Middlehurst, M. (2020). A tale of two toolkits, report the third: on the usage and performance of hive-cote v1.0. *arXiv:2004.06069*.

Benavoli, A., Corani, G., and Mangili, F. (2016). Should we really use post-hoc tests based on mean-ranks? *Journal of Machine Learning Research*, *17*, 152–161.

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

Chevyrev, I., and Lyons, T. (2016). Characteristic functions of measures on geometric rough paths. *The Annals of Probability*, *44*, 4049–4082.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 1724–1734.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*, 1–30.

Fermanian, A. (2021). Embedding and learning with signatures. *Computational Statistics & Data Analysis*, *157*, 107148.

Flint, G., Hambly, B., and Lyons, T. (2016). Discretely sampled signals and the rough Hoff process. *Stochastic Processes and their Applications*, *126*, 2593–2614.

Friz, P. K., and Victoir, N. B. (2010). *Multidimensional stochastic processes as rough paths: theory and applications* (Vol. 120). Cambridge University Press.

Futoma, J., Hariharan, S., and Heller, K. (2017). Learning to detect sepsis with a multitask Gaussian process RNN classifier. *Proceedings of the 34th International Conference on Machine Learning*, *70*, 1174–1182.

Grabocka, J., Schilling, N., Wistuba, M., and Schmidt-Thieme, L. (2014). Learning time-series shapelets. *Proceedings of the 20th International Conference on Knowledge Discovery and Data Mining*, 392–401.

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

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Karim, F., Majumdar, S., Darabi, H., and Harford, S. (2019). Multivariate lstm-fcns for time series classification. *Neural Networks*, *116*, 237–245.

Karlsson, I., Papapetrou, P., and Boström, H. (2016). Generalized random shapelet forests. *Data Mining and Knowledge Discovery*, *30*, 1053–1085.

Kidger, P., Bonnier, P., Perez Arribas, I., Salvi, C., and Lyons, T. (2019). Deep signature transforms. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 3099–3109). Curran Associates, Inc.

Kidger, P., and Lyons, T. (2020). Signatory: differentiable computations of the signature and logsignature transforms, on both CPU and GPU. *arXiv:2001.00706*. https://github.com/patrick-kidger/signatory

Kidger, P., Morrill, J., and Lyons, T. (2020b). Generalised Interpretable Shapelets for Irregular Time Series. *arXiv:2005.13948*.

Kormilitzin, A., Saunders, K. E., Harrison, P. J., Geddes, J. R., and Lyons, T. (2017). Detecting early signs of depressive and manic episodes in patients with bipolar disorder using the signature-based model. *arXiv:1708.01206*.

Lai, S., Jin, L., and Yang, W. (2017). Online signature verification using recurrent neural network and length-normalized path signature descriptor. *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, *1*, 400–405.

Levin, D., Lyons, T., and Ni, H. (2013). Learning from the past, predicting the statistics for the future, learning an evolving system. *arXiv:1309.0260*.

Li, C., Zhang, X., and Jin, L. (2017). LPSNet: a novel log path signature feature based hand gesture recognition framework. *2017 IEEE International Conference on Computer Vision Workshop*, 631–639.

Li, S. C.-X., and Marlin, B. M. (2016). A scalable end-to-end Gaussian process adapter for irregularly sampled time series classification. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 1804–1812). Curran Associates, Inc.

Liao, S., Lyons, T., Yang, W., and Ni, H. (2019). Learning stochastic differential equations using RNN with log signature features. *arXiv:1908.08286*.

Lyons, T., Caruana, M., and Lévy, T. (2007). *Differential equations driven by rough paths* (Vol. 1908). Springer.

Lyons, T., Ni, H., and Oberhauser, H. (2014). A feature set for streams and an application to high-frequency financial tick data. *Proceedings of the 2014 International Conference on Big Data Science and Computing*, 5.

Lyons, T., and Oberhauser, H. (2017). Sketching the order of events. *arXiv:1708.09708*.

Moor, M., Horn, M., Bock, C., Borgwardt, K., and Rieck, B. (2020). Path Imputation Strategies for Signature Models. *arXiv:2005.12359*.

Morrill, J., Kormilitzin, A., Nevado-Holgado, A., Swaminathan, S., Howison, S., and Lyons, T. (2019). The signature-based model for early detection of sepsis from electronic health records in the intensive care unit. *International Conference in Computing in Cardiology*.

Morrill, J., Salvi, C., Kidger, P., Foster, J., and Lyons, T. (2020b). Neural rough differential equations for long time series. *arXiv:2009.08295*.

Morrill, J. H., Kormilitzin, A., Nevado-Holgado, A. J., Swaminathan, S., Howison, S. D., and Lyons, T. J. (2020c). Utilization of the signature method to identify the early onset of sepsis from multivariate physiological time series in critical care monitoring. *Critical Care Medicine*, *48*, e976–e981.

Perez Arribas, I. (2018). Derivatives pricing using signature payoffs. *arXiv:1809.09466*.

Reizenstein, J., and Graham, B. (2020). Algorithm 1004: the iisignature library: efficient calculation of iterated-integral signatures and log signatures. *ACM Transactions on Mathematical Software*.

Reyes-Ortiz, J.-L., Oneto, L., Samá, A., Parra, X., and Anguita, D. (2016). Transition-aware human activity recognition using smartphones. *Neurocomputing*, *171*, 754–767.

Reyna, M. A., Josef, C. S., Jeter, R., Shashikumar, S. P., Westover, M. B., Nemati, S., Clifford, G. D., and Sharma, A. (2020). Early prediction of sepsis from clinical data: the physionet/computing in cardiology challenge 2019. *Critical Care Medicine*, *48*, 210–217.

Ruiz, A. P., Flynn, M., and Bagnall, A. (2020). Benchmarking multivariate time series classification algorithms. *arXiv:2007.13156*.

Schäfer, P., and Leser, U. (2017). Fast and accurate time series classification with weasel. *Proceedings of the 2017 Conference on Information and Knowledge Management*, 637–646.

Wang, B., Liakata, M., Ni, H., Lyons, T., Nevado-Holgado, A. J., and Saunders, K. (2019). A path signature approach for speech emotion recognition. *Interspeech 2019*, 1661–1665.

Wang, B., Wu, Y., Taylor, N., Lyons, T., Liakata, M., Nevado-Holgado, A. J., and Saunders, K. E. (2020). Learning to detect bipolar disorder and borderline personality disorder with language and speech in non-clinical interviews, 437–441.

Wang, Z., Yan, W., and Oates, T. (2017). Time series classification from scratch with deep neural networks: a strong baseline. *Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN)*, 1578–1585.

Warden, P. (2018). Speech commands: a dataset for limited-vocabulary speech recognition. *arXiv:1804.03209*.

Wilson-Nunn, D., Lyons, T., Papavasiliou, A., and Ni, H. (2018). A path signature approach to online arabic handwriting recognition. *Proceedings of the 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)*, 135–139.

Wu, Y., Ni, H., Lyons, T. J., and Hudson, R. L. (2021). Signature features with the visibility transformation. *2020 25th International Conference on Pattern Recognition (ICPR)*, 4665–4672.

Yang, W., Jin, L., and Liu, M. (2016a). DeepWriterID: An end-to-end online text-independent writer identification system. *IEEE Intelligent Systems*, *31*, 45–53.

Yang, W., Jin, L., Tao, D., Xie, Z., and Feng, Z. (2016b). Dropsample: a new training method to enhance deep convolutional neural networks for large-scale unconstrained handwritten chinese character recognition. *Pattern Recognition*, *58*, 190–203.

Yang, W., Lyons, T., Ni, H., Schmid, C., Jin, L., and Chang, J. (2017). Developing the path signature methodology and its application to landmark-based human action recognition. *arXiv:1707.03993*.

Ye, L., and Keogh, E. (2009). Time series shapelets: a new primitive for data mining. *Proceedings of the 15th International Conference on Knowledge Discovery and Data Mining*, 947–956.

Chapter 4

Linear functional regression with truncated signatures

We place ourselves in a functional regression setting and propose a novel methodology for regressing a real output on vector-valued functional covariates. This methodology is based on the notion of signature, which is a representation of a function as an infinite series of its iterated integrals. The signature depends crucially on a truncation parameter for which an estimator is provided, together with theoretical guarantees. An empirical study on both simulated and real-world datasets shows that the resulting methodology is competitive with traditional functional linear models, in particular when the functional covariates take their values in a high dimensional space.

Contents

4.1 Introduction

In a classical regression setting, a real output Y is described by a finite number of predictors. A typical example would be to model the price of a house as a linear function of several characteristics such as surface area, number of rooms, location, and so on. These predictors are typically encoded as a vector in $\mathbb{R}^p$, $p \in \mathbb{N}^*$. However, some applications do not fall within this setting. For example, in medicine, a classical task consists in predicting the state of a patient (for example, ill or not) from the recording of several physiological variables over some time. The input data is then a function of time and not a vector. Similarly, sound recognition or stock market prediction tasks both consist of learning from time series, possibly multidimensional. Then, a natural idea is to extend the linear model to this more general setting, where one wants to predict from a functional input, of the form $X : [0,1] \to \mathbb{R}^d$, $d \geq 1$.

This casts our problem into the field of functional data analysis and more specifically within the framework of functional linear regression (Ramsay and Dalzell, 1991; Marx and Eilers, 1999). This rich domain has undergone considerable developments in recent decades, as illustrated by the monographs of Ramsay and Silverman (2005) and Ferraty and Vieu (2006), and the review by Morris (2015). One of the core principles of functional data analysis is to represent input functions on a set of basis functions, for example, splines, wavelets, or the Fourier basis. Another approach also consists in extracting relevant handcrafted features, depending on the field of application. For example, Benzeghiba et al. (2007) and Turaga et al. (2008) provide overviews of learning methods specific to speech and human action recognition, respectively.

In this article, we build on the work of Levin et al. (2013) and explore a novel approach to linear functional regression, called the signature linear model. Its main strength is that it is naturally adapted to vector-valued functions, which is not the case of most of the methods previously mentioned. Its principle is to represent a function by its signature, defined as an infinite series of its iterated integrals. Signatures date back from the 60s when Chen (1958) showed that a smooth path can be faithfully represented by its iterated integrals and it has been at the center of rough path theory in the 90s (Lyons et al., 2007; Friz and Victoir, 2010). Rough path theory has seen extraordinary developments in recent times, and, in particular, has gained attention from the machine learning community. Indeed, signatures combined with (deep) learning algorithms have been successfully applied in various fields, such as characters recognition (Yang et al., 2015; 2016a; Lai et al., 2017; Liu et al., 2017), human action recognition (Li et al., 2017; Yang et al., 2017), speech emotion recognition (Wang et al., 2019), medicine (Arribas et al., 2018; Moore et al., 2019; Morrill et al., 2019; Morrill et al., 2020c), or finance (Arribas et al., 2020). We refer the reader to Chevyrev and Kormilitzin (2016b) for an introduction to signatures in machine learning, and to Fermanian (2021) for a more recent overview.

We stress again that the main advantage of the signature approach is that it can handle multidimensional input functions, that is, functions $X : [0,1] \to \mathbb{R}^d$ where $d \geq 2$, whereas traditional methods were designed for real-valued functions. Many modern datasets come in this form with a large dimension d. Moreover, the signature method requires little assumptions on the regularity of X and encodes nonlinear geometric information, that is, gives rise to interpretable regression coefficients. Finally, it is theoretically grounded by good approximation properties: any continuous function can be approximated arbitrarily well by a linear function of the truncated signature (Király and Oberhauser, 2019).

Since any continuous function of X can be approximated by a linear function on its truncated signature, the estimation of a regression function boils down to the estimation of the coefficients in this scalar product. The truncation order of the signature is therefore a crucial parameter as it controls the complexity of the model. Thus, in our quest for a linear model on the signature, one of the main purposes of our article will be to estimate this parameter. With an estimator

of the truncation order at hand, the methodology is complete and the signature linear model can be applied to both simulated and real-world data, demonstrating its good performance for practical applications. To summarize, our document is organized as follows.

(*i*) First, in Section 4.2, we set the mathematical framework of functional regression and recall the definition of the signature and its main properties.

(*ii*) Then, in Section 4.3, we introduce our model, called 'signature linear model', and define estimators of its parameters. Their rates of convergence are given in Section 4.4.

(*iii*) Finally, Section 4.5 is devoted to the practical implementation of the signature linear model. We conclude by demonstrating its performance on simulated data in Section 4.6 and on real-world data in Section 4.7.

For the sake of clarity, the proofs of the mathematical results are postponed to Appendix C.1 and C.2. The code is completely reproducible and available at https://github.com/afermanian/signature-regression.

4.2 Mathematical framework

4.2.1 Functional linear regression

We place ourselves in a functional linear regression setting with scalar responses: we are given a dataset $D_n = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$, where the pairs (X_i, Y_i) are independent and identically distributed copies of a random couple (X, Y), where X is a (random) function, $X : [0, 1] \to \mathbb{R}^d$, $d \in \mathbb{N}$, and Y a real random variable. For example, for the Canadian Weather dataset (Ramsay and Silverman, 2005), each sample corresponds to a location in Canada, the predictor X_i is the curve of the daily temperature at this location averaged from 1960 to 1994, and the response Y_i is the average total annual precipitation over the same period—see Figure 4.1. Our goal is to approximate the regression function $f(X) = \mathbb{E}[Y|X]$ by a parametrized linear function f_θ and to build an estimator of θ.

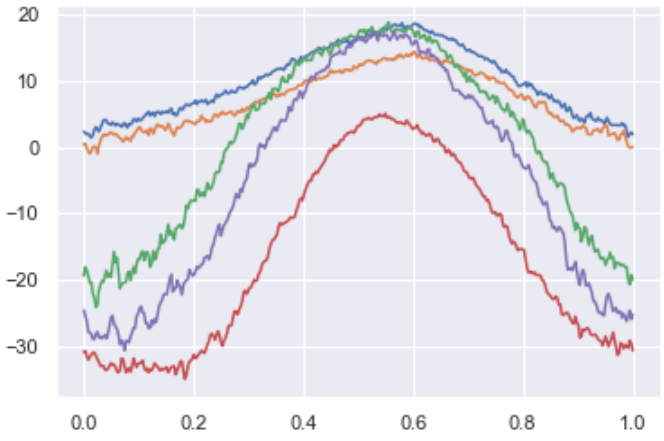


Figure 4.1 – 5 samples from the Canadian Weather dataset

In the univariate case, that is when $d = 1$, the classical functional linear model (Frank and Friedman, 1993; Hastie and Mallows, 1993) writes

$$Y = \alpha + \int_0^1 X(t)\beta(t)dt + \varepsilon, \tag{4.1}$$

where $\alpha \in \mathbb{R}$, $\beta : [0,1] \to \mathbb{R}$ and ε is a random noise. The functional coefficients β and the functional covariates X_i are then expanded on basis functions:

$$\beta(t) = \sum_{k=1}^{K} b_k \phi_k(t), \qquad X_i(t) = \sum_{k=1}^{K} c_{ik} \phi_k(t), \tag{4.2}$$

where $\phi_1, \dots, \phi_K$ are a set of real-valued basis functions (for example the monomials $1, t, t^2, \dots, t^K$ or the Fourier basis). Equation (4.1) can then be rewritten in terms of the c_{ik}s and b_ks, which brings the problem back to the well-known multivariate linear regression setting. Different approaches can then be used in terms of choice of basis functions and regularization (see Ramsay and Silverman, 2005, Chapter 15). Note that another common approach is functional principal components regression (Cardot et al., 1999; Brunel et al., 2016). The idea is to perform a functional principal components analysis (fPCA) on X, which gives a representation of X as a sum of K orthonormal principal components, and to use these as basis functions ϕ_ks.

We can see that in both cases, the functional nature of the problem is dealt with by projecting the functions X on a smaller linear space, spanned by basis functions. This basis expansion is not straightforward to extend to the vector-valued case, that is when $d > 1$, the common approach being to expand each coordinate of X independently. This amounts to assuming that there are no interactions between coordinates, which is a strong assumption and not an efficient representation when the coordinates are highly correlated. Moreover, to our knowledge, the only theoretical results in the vector-valued case are found in the domain of longitudinal data analysis (Greven et al., 2011; Park and Staicu, 2015). In this case, the different coordinates are assumed to be repeated measurements of a quantity of interest on a patient and each coordinate is given a parametric model, in the same spirit as ANOVA models. These parametric models do not apply in the general case when the coordinates may correspond to different quantities such as the evolution of different stocks or the x-y-z coordinates of a pen trajectory.

The signature approach removes the need to make such assumptions: the focus moves from finding a functional model for X to finding a basis for functions of X. In other words, instead of using a basis of functions, we use a basis of functions of functions. In a regression setting, this shift of perspective is particularly adequate since the object of interest is the regression function $f(X)$ and not X itself. The whole approach is based on the signature transformation, which takes as input a function X and outputs an infinite vector of coefficients known to characterize X under some smoothness assumptions. In particular, there are no assumptions on the structure of dependance in the different coordinates of X. In other words, the signature is naturally adapted to the vector-valued case.

Before we delve into the signature linear model, we gently introduce the notion of signature and review some of its important properties.

4.2.2 The signature of a path

We give here a brief presentation of signatures but the reader is referred to Lyons et al. (2007) or Friz and Victoir (2010) for a more involved mathematical treatment with proofs. To follow the vocabulary from rough path theory, we will often call the functional covariate $X : [0,1] \to \mathbb{R}^d$

a path. Our basic assumption is that X is of bounded variation, i.e., it has finite length.

Definition 4.1. *Let*

$$\begin{aligned} X : [0,1] &\longrightarrow \mathbb{R}^d \\ t &\longmapsto (X_t^1, \dots, X_t^d). \end{aligned}$$

The total variation of X is defined by

$$\|X\|_{TV} = \sup_{\mathcal{I}} \sum_{(t_0,\dots,t_k)\in\mathcal{I}} \|X_{t_i} - X_{t_{i-1}}\|,$$

where the supremum is taken over all finite subdivisions of $[0,1]$, and $\|\cdot\|$ denotes the Euclidean norm on $\mathbb{R}^d$. The set of paths of bounded variation is then defined by

$$BV(\mathbb{R}^d) = \big\{X : [0,1] \to \mathbb{R}^d \,|\, \|X\|_{TV} < \infty\big\}.$$

We recall that $BV(\mathbb{R}^d)$ endowed with the norm

$$\|X\|_{BV(\mathbb{R}^d)} = \|X\|_{TV} + \sup_{t\in[0,1]} \|X_t\|$$

is a Banach space. We stress that the basis functions traditionnaly used in functional data analysis are of bounded variation so the assumption that $X \in BV(\mathbb{R}^d)$ is much less restrictive than assuming an expansion such as (4.2). This assumption allows to define Riemann-Stieljes integrals along paths, which puts us in a position to define the signature.

Definition 4.2. *Let $X \in BV(\mathbb{R}^d)$ and $I = (i_1, \dots, i_k) \subset \{1, \dots d\}^k$, $k \geq 1$, be a multi-index of length k. The signature coefficient of X along the index I on $[0,1]$ is defined by*

$$S^I(X) = \int\cdots\int_{0\leq u_1<\cdots<u_k\leq 1} dX_{u_1}^{i_1} \dots dX_{u_k}^{i_k}. \tag{4.3}$$

$S^I(X)$ is then said to be a signature coefficient of order k.

The signature of X is the sequence containing all signature coefficients, i.e.,

$$S(X) = \big(1, S^{(1)}(X), \dots, S^{(d)}(X), S^{(1,1)}(X), S^{(1,2)}(X), \dots, S^{(i_1,\dots,i_k)}(X), \dots\big).$$

The signature of X truncated at order m, denoted by $S^m(X)$, is the sequence containing all signature coefficients of order lower than or equal to m, that is

$$S^m(X) = \big(1, S^{(1)}(X), S^{(2)}(X), \dots, S^{\overbrace{(d,\dots,d)}^{\text{length } m}}(X)\big).$$

Note that the assumption that $X \in BV(\mathbb{R}^d)$ may be relaxed: the signature may still be defined when the Riemann-Stieljes integrals are not well-defined. For example, the signature of the Brownian motion may be defined with Itô or Stratonovitch integrals. Integrating paths that are not of bounded variation is actually one of the motivations behind the definition of the signature in rough path theory.

A crucial feature of the signature is that it encodes the geometric properties of the path, as shown in Figure 4.2. Indeed, coefficients of order 1 correspond to the increments of the path

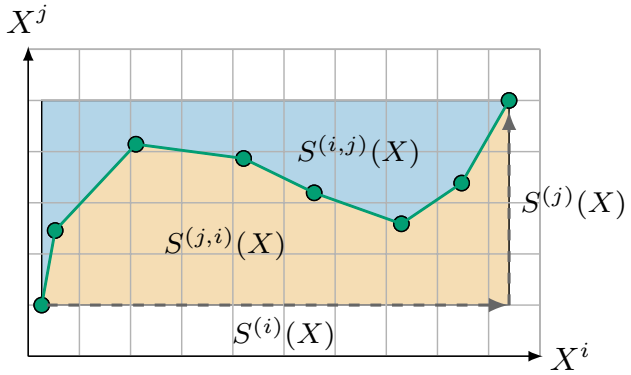


Figure 4.2 – Geometric interpretation of the signature coefficients. The terms $S^{(i)}(X)$ and $S^{(j)}(X)$ are the increments of the coordinates i and j respectively. The terms $S^{(i,j)} and S^{(j,i)}$ correspond to the areas of the blue and orange regions respectively.

in each coordinate and the coefficients of order 2 correspond to areas outlined by the path. For higher orders of truncation, the signature contains information about the joint evolution of tuples of coordinates. Moreover, it is clear from its definition as an integral that the signature is independent of the time parametrization (2010, Proposition 7.10) and that it is invariant by translation. Therefore, the signature looks at functions as purely geometric objects, without any information about sampling frequency, speed, or travel time, hence the terminology of 'paths'.

Note that the definition can be extended to paths defined on any interval $[s,t] \subset \mathbb{R}$ by changing the integration bounds in (4.3). We can see that there are d^k signature coefficients of order k. The signature truncated at order m is therefore a vector of dimension $s_d(m)$, where

$$s_d(m) = \sum_{k=0}^{m} d^k = \frac{d^{m+1}-1}{d-1} \quad \text{if } d \geq 2,$$

and $s_d(m) = m+1$ if $d = 1$. Thus, provided $d \geq 2$, the size of $S^m(X)$ increases exponentially with m and polynomially with d—some typical values are presented in Table 4.1.

Table 4.1 – Typical values of $s_d(m)$.

	$d=2$	$d=3$	$d=6$
$m=1$	2	3	6
$m=2$	6	12	42
$m=5$	62	363	9330
$m=7$	254	3279	335922

The set of coefficients of order k can be seen as an element of the kth tensor product of $\mathbb{R}^d$ with itself, denoted by $(\mathbb{R}^d)^{\otimes k}$. For example, the d coefficients of order 1 can be written as a vector, and the d^2 coefficients of order 2 as a matrix, i.e.,

$$\begin{pmatrix} S^{(1)}(X) \\ \vdots \\ S^{(d)}(X) \end{pmatrix} \in \mathbb{R}^d, \quad \begin{pmatrix} S^{(1,1)}(X) & \dots & S^{(1,d)}(X) \\ \vdots & & \vdots \\ S^{(d,1)}(X) & \dots & S^{(d,d)}(X) \end{pmatrix} \in \mathbb{R}^{d\times d} \approx (\mathbb{R}^d)^{\otimes 2}.$$

Similarly, coefficients of order 3 can be written as a tensor of order 3, and so on. Then, $S(X)$ can be seen as an element of the tensor algebra

$$\mathbb{R} \oplus \mathbb{R}^d \oplus (\mathbb{R}^d)^{\otimes 2} \oplus \cdots \oplus (\mathbb{R}^d)^{\otimes k} \oplus \cdots .$$

Although not fundamental in the present paper, this structure of tensor algebra turns out to be useful to derive properties of the signature (Lyons et al., 2007; Friz and Victoir, 2010).

Let us give two examples of paths and of their signatures.

Example 4.1. *Let X be a parametrized curve: for any $t \in [0,1]$, $X_t = (t, f(t))$, where $f : \mathbb{R} \to \mathbb{R}$ is a smooth function. Then,*

$$S^{(1)}(X) = \int_0^1 dX_t^1 = \int_0^1 dt = 1$$
$$S^{(2)}(X) = \int_0^1 dX_t^2 = \int_0^1 f'(t)dt = f(1) - f(0),$$

where f' denotes the derivative of f. Similarly, the signature coefficient along $(1,2)$ is

$$S^{(1,2)}(X) = \int_0^1 \int_0^t dX_u^1 dX_t^2 = \int_0^1 \Big(\int_0^t du \Big) f'(t)dt = \int_0^1 t f'(t)dt$$
$$= f(1) - \int_0^1 f(t)dt,$$

and so on.

Example 4.2. *Let X be a d-dimensional linear path:*

$$X_t = \begin{pmatrix} X_t^1 \\ \vdots \\ X_t^d \end{pmatrix} = \begin{pmatrix} a_1 + b_1 t \\ \vdots \\ a_d + b_d t \end{pmatrix}.$$

Then, for any index $I = (i_1, \ldots, i_k) \subset \{1, \ldots, d\}^k$, the signature coefficient along I is

$$S^{(i_1,\ldots,i_k)}(X) = \int_{0 \le u_1 < \cdots < u_k \le 1} \cdots \int dX_{u_1}^{i_1} \ldots dX_{u_k}^{i_k} = \frac{b_{i_1} \ldots b_{i_k}}{k!}. \tag{4.4}$$

It is clear here that the signature is invariant by translation: $S(X)$ depends only on the slope of X and not on the initial position $(a_1, \ldots, a_d)$.

We now recall a series of properties of the signature that motivate the definition of the signature linear model. A first important property provides a criterion for the uniqueness of signatures.

Proposition 4.1. *Assume that $X \in BV(\mathbb{R}^d)$ contains at least one monotone coordinate, then $S(X)$ characterizes X up to translations and reparametrizations.*

This is a sufficient condition, a necessary one has been derived by Hambly and Lyons (2010) and is based on the construction of an equivalence relation between paths, called tree-like equivalence. For any path $X \in BV(\mathbb{R}^d)$, the time-augmented path $\tilde{X}_t = (X_t, t) \in BV(\mathbb{R}^{d+1})$ satisfies the assumption of Proposition 4.1, which ensures signature uniqueness. Enriching the path with

new dimensions is actually a classic part of the learning process when signatures are used, and is discussed by Fermanian (2021) and Morrill et al. (2020a). We will always use this time-augmentation transformation before computing signatures.

The next proposition states that the signature linearizes functions of X and is the core motivation of the signature linear model. We refer the reader to Király and Oberhauser (2019), Theorem 1, for a proof in a similar setting.

Proposition 4.2. *Let $D \subset BV(\mathbb{R}^d)$ be a compact set of paths that have at least one monotone coordinate and such that, for any $X \in D$, $X_0 = 0$. Let $f : D \to \mathbb{R}$ be continuous. Then, for every $\varepsilon > 0$, there exists $m^* \in \mathbb{N}$, $\beta^* \in \mathbb{R}^{s_d(m^*)}$, such that, for any $X \in D$,*

$$\left| f(X) - \langle \beta^*, S^{m^*}(X) \rangle \right| \leq \varepsilon,$$

where $\langle \cdot, \cdot \rangle$ denotes the Euclidean scalar product on $\mathbb{R}^{s_d(m^*)}$.

This proposition is a consequence of the Stone-Weierstrass theorem. The classical Weierstrass approximation theorem states that every real-valued continuous function on a closed interval can be uniformly approximated by a polynomial function. Linear forms on the signature can, therefore, be thought of as the equivalent of polynomial functions for paths.

Finally, the following bound on the norm of the truncated signature allows us to control the rate of decay of signature coefficients of high order—see Lyons (2014, Lemma 5.1) for a proof.

Proposition 4.3. *Let $X : [0, 1] \to \mathbb{R}^d$ be a path in $BV(\mathbb{R}^d)$. Then, for any $m \geq 0$,*

$$\|S^m(X)\| \leq \sum_{k=0}^{m} \frac{\|X\|_{TV}^k}{k!} \leq e^{\|X\|_{TV}}.$$

4.3 The signature linear model

4.3.1 Presentation of the model

We are now in a position to present the signature linear model. Recall that our goal is to model the relationship between a real random variable $Y \in \mathbb{R}$ and a random input path $X \in BV(\mathbb{R}^d)$. Without loss of generality, we now assume that $d \geq 2$ (if $d = 1$, considering the time-augmented path brings us back to the case $d = 2$). Proposition 4.2 states that linear functions of the signature are dense in the set of continuous functions on a compact subset of $BV(\mathbb{R}^d)$, which says in essence that it is reasonable to model a continuous function of X as a linear function of its signature truncated at some order. This justifies the following model that was first introduced in a slightly different form by Levin et al. (2013). We assume that there exists $m \in \mathbb{N}$, $\beta_m^* \in \mathbb{R}^{s_d(m)}$, such that

$$\mathbb{E}[Y|X] = \left\langle \beta_m^*, S^m(X) \right\rangle \quad \text{and} \quad \mathrm{Var}(Y|X) \leq \sigma^2 < \infty. \tag{4.5}$$

We consider throughout the article the smallest $m^* \in \mathbb{N}$ such that there exists $\beta_{m^*}^* \in \mathbb{R}^{s_d(m^*)}$ satisfying

$$\mathbb{E}[Y|X] = \left\langle \beta_{m^*}^*, S^{m^*}(X) \right\rangle.$$

In other words, we assume a regression model, where the regression function is a linear form on the signature. Moreover, it can be noticed that, since the first term of signatures is always equal to 1, this regression model contains an intercept. Therefore, when $m^* = 0$, (4.5) is a constant model. Finally, it should be pointed out that there are two unknown quantities in model (4.5):

m^* and $\beta^*_{m^*}$. The parameter m^* is the truncation order of the signature of X and controls the model size, whereas $\beta^*_{m^*}$ is the vector of regression coefficients, whose size $s_d(m^*)$ depends on m^*.

It is instructive to compare this model to the functional model (4.1). We can see that much less assumptions on X are needed: it is only assumed to be of finite variation, whereas in (4.1) it has to be expanded on basis functions. Moreover, our model is directly adapted to the vector-valued case. Finally, it depends directly on a finite vector $\beta^*_{m^*}$, whereas (4.1) is written in terms of a function β, which must itself be written on basis functions. Note that the choice of basis need to be adapted to each particular application, whereas the signature linear model only depends on two parameters. In a nutshell, it is a more general model with less hyperparameters.

The signature truncation order m^* is a key quantity in this model and influences the rest of the study. Indeed, it controls the number of coefficients and therefore the computational feasibility of the whole method. However, it is in general little discussed in the literature and small values are picked arbitrarily. For example, Liu et al. (2017) consider values of m up to 2, Yang et al. (2015) up to 3, Arribas et al. (2018) and Lai et al. (2017) up to 4, Yang et al. (2016a) up to 5 , and Yang et al. (2017) up to 8. Thus, one of our main objectives is to establish a rigorous procedure to estimate m^*, and, to this end, we define a consistent estimator of m^*. As we will see later, a simple estimator of $\beta^*_{m^*}$, and therefore of the regression function, is then also obtained.

4.3.2 Estimating the truncation order

Let

$$D_n = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$$

be i.i.d. observations drawn according to the law of (X, Y). We use the approach of penalized empirical risk minimization. For the moment, let us fix a certain truncation order $m \in \mathbb{N}$, and let $\alpha > 0$ denote a fixed positive number. Then, the ball in $\mathbb{R}^{s_d(m)}$ of radius α centered at 0 is denoted by

$$B_{m,\alpha} = \big\{\beta \in \mathbb{R}^{s_d(m)} \mid \|\beta\| \leq \alpha\big\},$$

where $\|\cdot\|$ stands for the Euclidean norm, whatever the dimension. By a slight abuse of notation, the sequence $(B_{m,\alpha})_{m\in\mathbb{N}}$ can be seen as a nested sequence of balls, i.e.,

$$B_{0,\alpha} \subset B_{1,\alpha} \subset \cdots \subset B_{m,\alpha} \subset B_{m+1,\alpha} \subset \cdots .$$

From now on, we will only consider coefficients within these balls. Therefore, we assume that the true coefficient $\beta^*_{m^*}$ lies within such a ball, i.e., we make the assumption:

(H_α) $\beta^*_{m^*} \in B_{m^*,\alpha}$.

On the one hand, for a fixed truncation order m, the theoretical risk is defined by

$$\mathcal{R}_m(\beta) = \mathbb{E}\big(Y - \langle \beta, S^m(X)\rangle\big)^2.$$

The minimal theoretical risk for a certain truncation order m, denoted by $L(m)$ is then

$$L(m) = \inf_{\beta\in B_{m,\alpha}} \mathcal{R}_m(\beta) = \mathcal{R}_m(\beta^*_m),$$

where $\beta^*_m \in \operatorname{argmin}_{\beta\in B_{m,\alpha}} \mathcal{R}_m(\beta)$ (note that the existence of β^*_m is ensured by convexity of the problem). Since the sets $(B_{m,\alpha})_{m\in\mathbb{N}}$ are nested, L is a decreasing function of m. Its minimum is

attained at $m = m^*$, and, provided $m \geq m^*$, $L(m)$ is then constant and equal to

$$\mathcal{R}(\beta^*_{m^*}) = \mathbb{E}\big(Y - \langle \beta^*_{m^*}, S^{m^*}(X)\rangle\big)^2 = \mathbb{E}\big(\mathrm{Var}(Y|X)\big) \leq \sigma^2.$$

On the other hand, the empirical risk with signature truncated at order m is defined by

$$\widehat{\mathcal{R}}_{m,n}(\beta) = \frac{1}{n}\sum_{i=1}^{n}\big(Y_i - \langle \beta, S^m(X_i)\rangle\big)^2,$$

where $\beta \in B_{m,\alpha}$. The minimum of $\widehat{\mathcal{R}}_{m,n}$ over $B_{m,\alpha}$ is denoted by $\widehat{L}_n(m)$ and defined as

$$\widehat{L}_n(m) = \min_{\beta \in B_{m,\alpha}} \widehat{\mathcal{R}}_{m,n}(\beta) = \widehat{\mathcal{R}}_{m,n}(\widehat{\beta}_m),$$

where $\widehat{\beta}_m$ denotes a point in $B_{m,\alpha}$ where the minimum is attained. Note that $\beta \mapsto \widehat{\mathcal{R}}_{m,n}(\beta)$ is a convex function so $\widehat{\beta}_m$ exists. We point out that minimizing $\widehat{\mathcal{R}}_{m,n}$ over $B_{m,\alpha}$ is equivalent to performing a Ridge regression with a certain regularization parameter which depends on α.

In short, for a fixed truncation order m, a Ridge regression gives the best parameter $\widehat{\beta}_m$ to model Y as a linear form on the signature of X truncated at order m. Recall that our goal is to find a truncation order $\widehat{m}$ close to the true one m^*. Since the $(B_{m,\alpha})_{m\in\mathbb{N}}$ are nested, the sequence $(\widehat{L}_n(m))_{m\in\mathbb{N}}$ decreases with m. Indeed, increasing m makes the set of parameters larger and therefore decreases the empirical risk. An estimator of m^* can then be defined by a trade-off between this decreasing empirical risk and an increasing function that penalizes the number of coefficients:

$$\widehat{m} = \min\Big(\operatorname*{argmin}_{m\in\mathbb{N}}\big(\widehat{L}_n(m) + \mathrm{pen}_n(m)\big)\Big),$$

where $\mathrm{pen}_n(m)$ is an increasing function of m that will be defined in Theorem 4.4. If the minimum of $\widehat{L}_n + \mathrm{pen}_n$ is reached by several values, we choose for $\widehat{m}$ the smallest one. The procedure is illustrated in Figure 4.3 with the Canadian Weather dataset.

Now that we have an estimate of m^*, which is a key ingredient in establishing the whole process of the expected signature method, and before presenting the whole procedure, we justify the estimator by some theoretical results in the next section.

4.4 Performance bounds

In this section, we show that it is possible to calibrate a penalization that ensures exponential convergence of $\widehat{m}$ to m^*. The proof is given in Appendix C.1. In addition to (H_α), we need the following assumption:

(H_K) there exists $K_Y > 0$ and $K_X > 0$ such that almost surely $|Y| \leq K_Y$ and $\|X\|_{TV} \leq K_X$.

In a nutshell, (H_K) says that the trajectories have a length uniformly bounded by K_X, which is in practice a reasonable assumption. We shall also use the constant K, defined by

$$K = 2(K_Y + \alpha e^{K_X})e^{K_X}. \tag{4.6}$$

The main result of the section is the following.

Theorem 4.4. *Let $K_{\mathrm{pen}} > 0$, $0 < \rho < \frac{1}{2}$, and*

$$\mathrm{pen}_n(m) = K_{\mathrm{pen}} n^{-\rho}\sqrt{s_d(m)}. \tag{4.7}$$

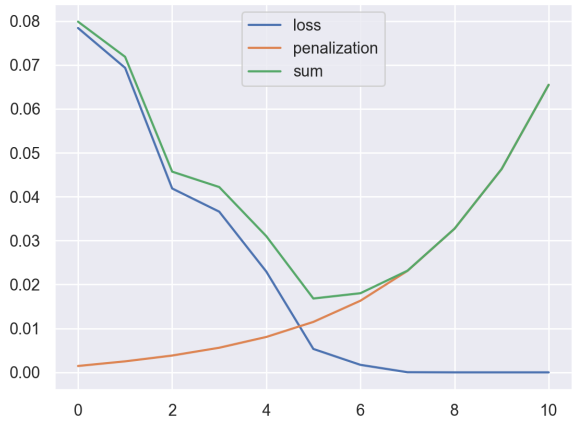


Figure 4.3 – The functions $m \mapsto \widehat{L}_n(m)$ (blue curve), $m \mapsto \text{pen}_n(m)$ (orange curve) and $m \mapsto \widehat{L}_n(m) + \text{pen}_n(m)$ (green curve) in the case of the Canadian Weather dataset. The estimator $\widehat{m}$ is chosen to be the minimize of the green curve: $\widehat{m} = 5$.

Let n_0 be the smallest integer satisfying

$$(n_0)^{\tilde{\rho}} \geq (432K\alpha\sqrt{\pi} + K_{\text{pen}})\Big(\frac{2\sqrt{s_d(m^*+1)}}{L(m^*-1)-\sigma^2} + \frac{\sqrt{2s_d(m^*+1)}}{K_{\text{pen}}\sqrt{d^{m^*+1}}}\Big), \tag{4.8}$$

where $\tilde{\rho} = \min(\rho, \frac{1}{2} - \rho)$. Then, under the assumptions (H_α) and (H_K), for any $n \geq n_0$,

$$\mathbb{P}\left(\widehat{m} \neq m^*\right) \leq C_1 \exp\left(-C_2 n^{1-2\rho}\right),$$

where the constants C_1 and C_2 are defined by

$$C_1 = 74 \sum_{m>0} e^{-C_3 s_d(m)} + 148 m^*, \tag{4.9}$$

where

$$C_3 = \frac{K_{\text{pen}}^2 d^{m^*+1}}{128 s_d(m^*+1)(72K^2\alpha^2 + K_Y^2)},$$

and

$$C_2 = \frac{1}{16(1152K^2\alpha^2 + K_Y^2)} \min\Big(\frac{K_{\text{pen}}^2 d^{m^*+1}}{8 s_d(m^*+1)}, L(m^*-1) - \sigma^2\Big). \tag{4.10}$$

This theorem provides a non-asymptotic bound on the convergence of $\widehat{m}$. It implies the almost sure convergence of $\widehat{m}$ to m^*. We can note that the penalty decreases slowly with n (more slowly than a square-root) and, if $d \geq 2$, increases with m exponentially, i.e., as $d^{m/2}$. The penalty includes an arbitrary constant K_{pen}. Its value that minimizes n_0 is

$$K_{\text{pen}}^* = \sqrt{\frac{(L(m^*-1)-\sigma^2)432\sqrt{\pi}\alpha K}{d^{m^*+1}}},$$

and, in practice, it is calibrated with the slope heuristics method of Birgé and Massart (2007), described in Section 4.5. The proof of Theorem 4.4 is based on chaining tail inequalities that bound uniformly the tails of the risk. We refer the reader to Appendix C.1 for a detailed proof.

To give some insights into this estimator it is interesting to look at the behavior of the constants when different quantities vary.

- If the dimension of the path d gets large, $d^{m^*+1} \sim s_d(m^*+1)$ so the constants C_1 and C_2 stay of the same order (provided that the risk $L(m^*-1)$ stays constant). Therefore, the quality of the bound does not change in high dimensions. However, the constant n_0 increases at the rate of $\mathcal{O}(d^{m^*/2\tilde{\rho}})$: we neeed exponentially more data when d grows.
- If the true truncation parameter m^* increases, the same phenomenon is observed except that C_1 increases linearly: C_2 and C_3 stay of the same order, $C_1 \sim 148m^*$, and n_0 increases at the rate of $\mathcal{O}(d^{m^*/2\tilde{\rho}})$. It is not surprising: when m^* increases, the size of the coefficient $\beta^*_{m^*}$ increases and therefore more data are needed to estimate it.
- If α increases, n_0 and C_1 increase while C_2 decreases. In other words, more data is needed and the quality of the estimator deteriorates. Indeed, when α gets larger, the parameter spaces $B_{m,\alpha}$ gets larger for any m so estimation is harder.
- The last quantity of interest is $L(m^*-1) - \sigma^2 \leq L(m^*-1) - L(m^*)$, which measures the difference of risk between a smaller model and the model truncated at m^*. By definition, it is a strictly positive quantity. When it gets close to zero, it means that a model truncated at m^*-1 is almost as good as a model truncated at m^*. We can see that when this difference decreases, n_0 increases and C_2 decreases: it is harder to find that a truncation order of m^* is better than m^*-1, therefore the estimator $\widehat{m}$ deteriorates.

With an estimator of $\widehat{m}$ at hand, one can simply choose to estimate $\beta^*_{m^*}$ by $\widehat{\beta}_{\widehat{m}}$, which gives an estimator of the regression function in model (4.5). As a by-product of Theorem 4.4, we then get the following bound.

Corollary 4.5.

$$\mathbb{E}\Big(\big\langle \hat{\beta}_{\widehat{m}}, S^{\widehat{m}}(X)\big\rangle - \big\langle \beta^*_{m^*}, S^{m^*}(X)\big\rangle\Big)^2 = \mathcal{O}\Big(n^{-1/2}\Big).$$

This rate of convergence in $\mathcal{O}(n^{-1/2})$ is similar to the ones usually obtained for functional linear models when $d=1$, except that much less assumptions are needed on the path X. Indeed, the rates obtained on the regression function usually depend on regularity assumptions on X and β in (4.1). For example, it can depend on the Fourier coefficients of X (Hall, Horowitz, et al., 2007), on the number of Lipschitz-continuous derivatives of β (Cardot et al., 2003), or on the periodicity of X (Li and Hsing, 2007).

The proof is given in Appendix C.2. We have now all the ingredients necessary to implement this signature linear model. Before looking at its performance on real-world datasets, we present in the next section the complete methodology.

4.5 Computational aspects

4.5.1 The signature linear model algorithm

Computing the signature A first step towards practical application is to be able to compute signatures efficiently. Typically, the input data consists of arrays of sampled values of X. We choose to interpolate the sampled points linearly, and therefore our problem reduces to computing signatures of piecewise linear paths. To this end, equation (4.4) gives the signature of a linear

path and Chen's theorem (Chen, 1958), stated below, provides a formula to compute recursively the signature of a concatenation of paths.

Let $X : [s,t] \to \mathbb{R}^d$ and $Y : [t,u] \to \mathbb{R}^d$ be two paths, $0 \leq s < t < u \leq 1$, the concatenation of X and Y, denoted by $X * Y$, is defined as the path from $[s,u]$ to $\mathbb{R}^d$ such that, for any $v \in [s,u]$,

$$(X * Y)_v = \begin{cases} X_v & \text{if } v \in [s,t], \\ X_t + Y_v - Y_t & \text{if } v \in [t,u]. \end{cases}$$

Proposition 4.6 (Chen). *Let $X : [s,t] \to \mathbb{R}^d$ and $Y : [t,u] \to \mathbb{R}^d$ be two paths with bounded variation. Then, for any multi-index $(i_1, \dots, i_k) \subset \{1, \dots, d\}^k$,*

$$S^{(i_1,\dots,i_k)}(X * Y) = \sum_{\ell=0}^{k} S^{(i_1,\dots,i_\ell)}(X) \cdot S^{(i_{\ell+1},\dots,i_k)}(Y). \tag{4.11}$$

This proposition is an immediate consequence of the linearity property of integrals (Lyons et al., 2007, Theorem 2.9). Therefore, to compute the signature of a piecewise linear path, it is sufficient to iterate the following two steps:

1. Compute with equation (4.4) the signature of a linear section of the path.
2. Concatenate it to the other pieces with Chen's formula (4.11).

This procedure is implemented in the Python library `iisignature` (Reizenstein and Graham, 2020). Thus, for a sample consisting of p points in $\mathbb{R}^d$, if we consider the path formed by their linear interpolation, the computation of the path signature truncated at level m takes $\mathcal{O}(pd^m)$ operations. The complexity is therefore linear in the number of sampled points but exponential in the truncation order m.

Algorithme 1 : Pseudo-code for the signature linear model.

Data : $\{(\mathbf{x}_1, Y_1), \dots, (\mathbf{x}_n, Y_n)\}$
Result : Estimators $\hat{m}$ and $\hat{\beta}_{\hat{m}}$

Interpolate linearly the columns of $\mathbf{x}_i$ so as to have a set of continuous piecewise linear paths $X_i : [0,1] \to \mathbb{R}^d$, $1 \leq i \leq n$. Add a time dimension, i.e., consider the path $\widetilde{X}_i : [0,1] \to \mathbb{R}^{d+1}$, where $\widetilde{X}_i^j = X_i^j$ for $1 \leq j \leq d$, and $X_{i,t}^{d+1} = t$, $t \in [0,1]$.

Select the Ridge regularization parameter λ by cross validation on the regression model with $\{S^1(\widetilde{X}_1), \dots, S^1(\widetilde{X}_n)\}$ as predictors.

for $m = 1, \dots, M$ **do**

  Compute signatures truncated at level m: $\{S^m(\widetilde{X}_1), \dots, S^m(\widetilde{X}_n)\}$.

  Fit a Ridge regression on the pairs $\{(S^m(\widetilde{X}_1), Y_1), \dots, (S^m(\widetilde{X}_n, Y_n)\}$. Compute its squared loss $\widehat{L}_n(m)$.

  Compute the penalization $\mathrm{pen}_n(m) = K_{\mathrm{pen}} \frac{\sqrt{s_d(m)}}{n^\rho}$.

Choose $\widehat{m} = \underset{0 \leq m \leq M}{\operatorname{argmin}} \big(\widehat{L}_n(m) + \mathrm{pen}_n(m)\big)$.

Compute $\widehat{\beta}_{\widehat{m}}$ by fitting a Ridge regression on $\{(S^{\widehat{m}}(\widetilde{X}_1), Y_1), \dots, (S^{\widehat{m}}(\widetilde{X}_n, Y_n)\}$.

Procedure In practice, we are given a dataset $\{(\mathbf{x}_1, Y_1), \dots, (\mathbf{x}_n, Y_n)\}$, where, for any $1 \leq i \leq n$, $Y_i \in \mathbb{R}$ and $\mathbf{x}_i \in \mathbb{R}^{d \times p_i}$. The columns of the matrix $\mathbf{x}_i$ correspond to values of a process X_i

in $\mathbb{R}^d$ sampled at p_i different times. We fix $M \in \mathbb{N}$ such that, for any $m \geq M$, the function $m \mapsto \widehat{L}_n(m) + \mathrm{pen}_n(m)$ is strictly increasing and apply the procedure described in Algorithm 1.

Note that in the first step of Algorithm 1 there exist other choices for the embedding of the matrix $\mathbf{x_i}$ into a continuous path $\widetilde{X}_i$ (Fermanian, 2021). The parameter ρ is set to 0.4. The constant K_{pen} is calibrated with the so-called slope heuristics method, first proposed by Birgé and Massart (2007).

4.5.2 A toy example

This section is devoted to illustrating the different steps of Algorithm 1 and the convergence of the estimator $\widehat{m}$ with simulated data. It will be implemented on real data in Section 4.7. We first simulate a dataset $\{(\mathbf{x_1}, Y_1), \ldots, (\mathbf{x_n}, Y_n)\}$ following the signature model (4.5).

For any $1 \leq i \leq n$, let $X_i : [0,1] \to \mathbb{R}^d$, $X_{i,t} = (X^1_{i,t}, \ldots, X^d_{i,t})$ be defined by

$$X^k_{i,t} = \alpha^k_{i,1} + 10\alpha^k_{i,2} \sin\left(\frac{2\pi t}{\alpha^k_{i,3}}\right) + 10(t - \alpha^k_{i,4})^3, \quad 1 \leq k \leq d, \tag{4.12}$$

where the parameters $\alpha^k_{i,\ell}$, $1 \leq \ell \leq 4$ are sampled uniformly on $[0,1]$. Let $(t_0, t_1, \ldots, t_{p-1})$ be a regular partition of $[0,1]$ of length p, the matrix of the path values

$$\mathbf{x_i} = (x^k_{i,j})_{\substack{1 \leq k \leq d \\ 1 \leq j \leq p}} \in \mathbb{R}^{d \times p}$$

is then a discretization of X_i on $[0,1]$: $x^k_{i,j} = X^k_{i,t_j}$. It will cause no confusion to use the same notation $\mathbf{x_i}$ to denote the matrix of values of X_i on the partition $(t_0, \ldots, t_{p-1})$ and their piecewise linear interpolation. Figure 4.4 shows one sample $\mathbf{x_i}$ with $p = 100$ and $d = 5$.

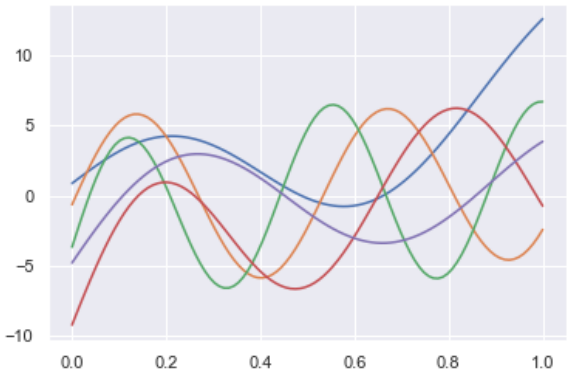


Figure 4.4 – One sample X_i from model (4.12) with $d = 5$.

For any $m^* \in \mathbb{N}$, the output Y_i is now defined as $Y_i = \langle \beta, S^{m^*}(\mathbf{x_i}) \rangle + \varepsilon_i$, where ε_i is a uniform random variable on $[-100, 100]$ and β is given by

$$\beta_j = \frac{1}{1000} u_j, \quad 1 \leq j \leq s_d(m^*),$$

where u_j is sampled uniformly on $[0,1]$. Then, m^* is estimated with the procedure described

in Algorithm 1 for different sample sizes n. To select the constant K_{pen}, we use the dimension jump method, that is we plot $\widehat{m}$ as a function of K_{pen}, find the value of K_{pen} that corresponds to the first big jump of $\widehat{m}$ and fix K_{pen} to be equal to twice this value. For a recent account of the theory of slope heuristics, we refer the reader to the review by Arlot (2019). For example, for $m^* = 5$ and $d = 2$, plotting $\widehat{m}$ against K_{pen} yields Figure 4.5. In this case, K_{pen} is selected at 20.

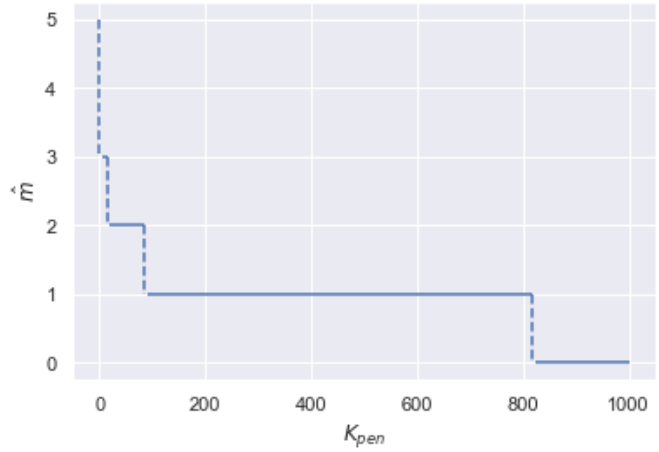


Figure 4.5 – Selection of K_{pen} with the slope heuristics method.

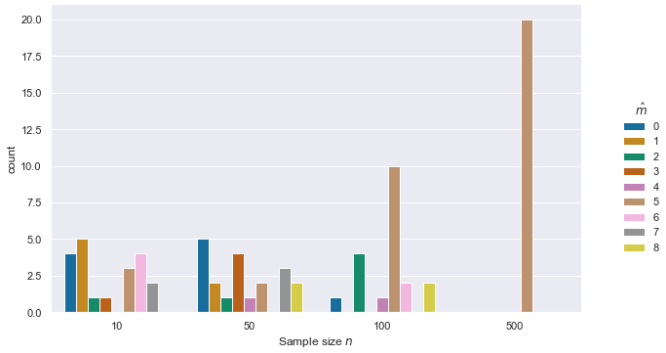


Figure 4.6 – Histogram of $\widehat{m}$ as a function of n over 20 iterations. The functional predictors X are simulated following (4.12) and the response Y follows the linear model on signatures with $m^* = 5$. The hyperparameters are $\rho = 0.4$ and $K_{\text{pen}} = 20$.

We fix $d = 2$ and $m^* = 5$ and, for different sample sizes n, we iterate the whole process 20 times. In Figure 4.6, a histogram of the values taken by $\widehat{m}$ is plotted against n. We can see that when n increases, the estimator converges to the true value $m^* = 5$. For $n = 500$ we always pick $\widehat{m} = 5$ over the 20 iterations.

4.6 Experiments

Now that we have a complete procedure at hand, we demonstrate in this section its performance compared to canonical approaches in functional data analysis. We show in particular that it performs better in high dimension, that is when d is large.

Throughout the section, since the focus is now on the performance of the signature linear model and to simplify the computations, we select $\widehat{m}$ via cross-validation. We compare our model to the functional linear model with basis functions presented in Section 4.2.1 and to the functional principal component regression (fPCR). We take for $\phi_1, \dots, \phi_K$ the B-sline and Fourier basis such as defined in Ramsay and Silverman (2005). Then, the approach consists in projecting the function $X : [0,1] \to \mathbb{R}^d$ onto the ϕ_is, coordinate by coordinate, and the output Y is assumed to be linear on the coefficients of X in this basis. The number K of basis functions is selected via cross-validation (with a minimum of 4 and maximum of 14 for Fourier and B-splines, and a minimum of 1 and maximum of 6 for the fPCR). For the fPCR, we first smooth the functional covariates with 7 B-splines. This procedure is implemented with the Python package `scikit-fda` (Ramos-Carreño et al., 2019).

4.6.1 Smooth paths

Our goal is to see the influence of the dimension d on the quality of the different models: the signature linear model and the 3 linear functional models. To this end, we simulate some paths following model (4.12) and try to predict the average or maximal value of the path at the next time step. More precisely, let $(t_0, t_1, \dots, t_p)$ be a partition of $[0,1]$ of length $p+1$, then we sample X_i following (4.12) and let

$$\begin{aligned} \mathbf{x_i} &= (X_{i,t_0}, \dots, X_{i,t_{p-1}}) \in \mathbb{R}^{d \times p}, \\ Y_i^{(\text{mean})} &= \frac{1}{d} \sum_{k=1}^{d} X_{i,t_p}^k, \\ Y_i^{(\text{max})} &= \max_{1 \le k \le d} X_{i,t_p}^k. \end{aligned}$$

For both models (mean and max), we let d vary on a grid from 1 to 11, simulate some train and test data, and assess the performance of the model with the mean squared error (MSE) on the test set. We iterate the procedure 20 times, which gives, for each model (signature, Fourier, B-spline, and fPCR), a boxplot of errors, shown in Figure 4.7.

It is first clear that for the mean response (left panel), when d increases, the signature gets better relatively to the 3 other models. The behavior of the performance in the right panel, where we try to predict the maximum value of the path accross the dimensions, is also interesting. When d increases this task gets harder since there are more dimensions along which to take the maximum, however the performance of the signature model stays approximately the same. On the contrary, the B-spline and Fourier basis errors increase steadily with d, while the performance of fPCR is bell-shaped: the error increases for d up to 3 and then decreases.

In this model, the different dimensions of the path were sampled independently from each other, which favors the traditional models with basis functions. We therefore perform the same study with paths X which have dependent dimensions, to see if the signature model is better in this case. The paths are very close to the ones in the previous study, the only difference lies in the generation of the parameters $\alpha_{i,\ell}^k$ in (4.12). For each sample $1 \le i \le n$, we sample four parameters $(\alpha_{i,1}, \alpha_{i,2}, \alpha_{i,3}, \alpha_{i,4})$ uniformly on $[0,1]$ and for each coordinate $k \in \{1, \dots, d\}$, we

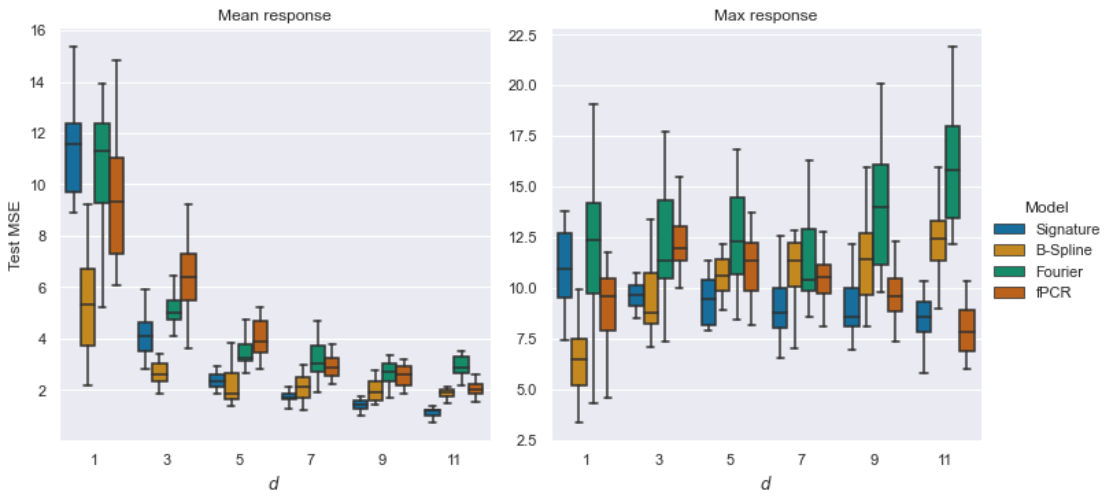


Figure 4.7 – Test MSE for the signature linear model, functional regression with B-Spline and Fourier basis functions, and functional Principal Component Regression (fPCR). The data follows (4.12) and Y is the mean (left panel) or maximum (right panel) response at the next time step.

sample a new parameter a_k uniformly in $[0, 1]$. Then, we let $\alpha^k_{i,\ell} = a_k \times \alpha_{i,\ell}$, $1 \leq \ell \leq 4$. Each coordinate X^k_i is then equal to (4.12) with these new parameters. In this way, the different coordinates of each sample X_i share the parameters $\alpha_{i,\ell}$, which are randomly multiplied by a_k. One such sample is plotted in Figure 4.8, which can be compared to Figure 4.4, and a boxplot of the test MSE is shown in Figure 4.9.

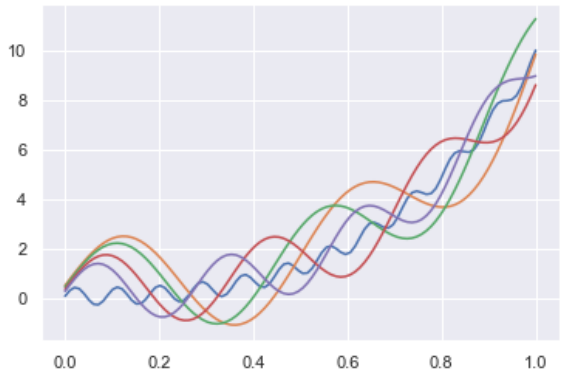


Figure 4.8 – One sample X from the dependent model with $d = 5$

If we compare Figures 4.7 and 4.9, we can see that the signature is slightly better in the dependent case. For example, in the mean response case (left panel in Figure 4.9), the signature is better than B-splines from $d = 5$ whereas it is better in the independent case (left panel in Figure 4.7) from $d = 7$. For the max response model, the variance of the error decreases in the dependent case for signatures, as would be expected (the maximum coordinate is more stable when d increases when the dimensions are correlated), whereas it does not for the Fourier and

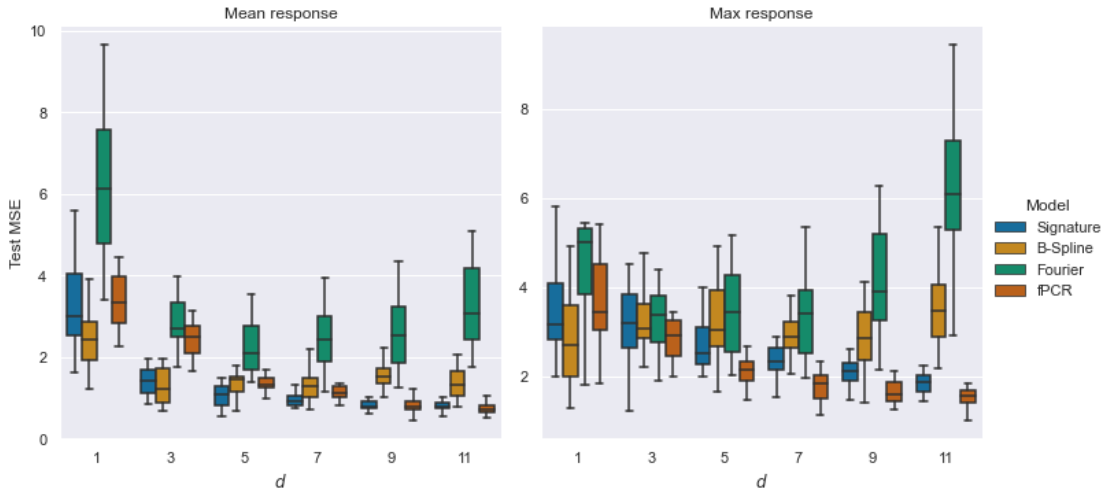


Figure 4.9 – Test MSE for the signature linear model, functional regression with B-Spline and Fourier basis functions, and functional Principal Component Regression (fPCR). The data follows (4.12) with dependent coordinate parameters and Y is the mean (left panel) or maximum (right panel) response at the next time step.

B-spline models. In other words, the signature model is more stable with regards to the structure in the data. Note also that the performance of the fPCR is similar (and slightly better for the max response) to the signature model, emphasizing the relevance of the signature model as a benchmark model since it performs as well and sometimes better than the most commonly used model.

4.6.2 Gaussian processes

We conclude this simulation study with more complex paths: Gaussian processes. Let $d \geq 1$, $1 \leq i \leq n$, we define the path $X_i = (X_t^1, \dots, X_t^d)_{t\in[0,1]}$ by

$$X_{i,t}^k = \alpha_i^k t + \xi_{i,t}^k, \quad 1 \leq k \leq d, \quad t \in [0,1], \tag{4.13}$$

where α_i^k is sampled uniformly in $[-3,3]$ and ξ_i^k is a Gaussian process with exponential covariance matrix (with length-scale 1). The response is the norm of the trend slope: $Y_i = \|\alpha_i\|$. Figure 4.10 shows a realization of X_i with $d = 5$.

We vary the dimension d on the same grid as before and iterate the whole procedure 20 times, which gives the results in Figure 4.11. We can see that for these more complicated paths, the signature is better than the 3 other models even for $d = 1$ and that the difference in performance with B-spline and Fourier basis increases with d.

4.7 Real-world applications

4.7.1 The Canadian Weather dataset

We close this study by implementing the signature linear model on real-world datasets. First, we consider the Canadian Weather dataset, presented in Section 4.2.1. We split the data into a

Figure 4.10 – One sample X from the Gaussian process model (4.13) with $d = 5$

training set and a test set (of size 23 and 12 respectively).

We implement the signature linear model as presented in Algorithm 1, which we compare to the same algorithms as before. We compute the MSE on 20 random train/test splits. Note that the constant K_{pen} is kept the same on all train/test splits (since it has to be manually selected via the slope heuristics method). The results of this procedure are shown in Figure 4.12. We see that for this particular application, the signature has a similar but slightly worse performance than the 3 other functional linear models. This is not surprising since this is the perfect setting for basis functions: the curves are smooth and unidimensional. However, it is worth noting that signatures do not perform badly in this setting. Moreover, this simple example allows us to discuss further the interpretation of the regression coefficients $\widehat{\beta}_{\widehat{m}}$, plotted as a heatmap in Figure 4.13.

The first row corresponds to the intercept, the second row to the coefficients against $S^{(1)}(X)$ and $S^{(2)}(X)$, the third row to the order 2 signature coefficients, and so on. A first thing to notice is that the coefficients get more sparse when they correspond to higher order signatures: almost all coefficients in the last row are equal to zero, whereas in the row corresponding to the order 2 half of them are significantly not null. Moreover, recall that the second coordinate of the path is equal to the time, so the coefficients corresponding to the indices $(2), (2,2), \ldots, (2,2,2,2)$ should not be significant, which is indeed the case—these are the last coefficient of each row. Moreover, we see that $S^{(1,2)}(X)$ and $S^{(2,1)}(X)$ have coefficients almost equal but opposite to each other: this means that the quantity of interest is the difference $S^{(1,2)}(X) - S^{(2,1)}(X)$, which is exactly the quantity known in stochastic analysis as the Levy area, depicted in Figure 4.14. We can conclude from this analysis that the total annual precipitations depend strongly on the area of the temperature curves, that is, the total temperature over the year. However, since some coefficients of order higher than 2 are not null, the relationship is slightly more complicated: the shape of the temperature profile also influences the total precipitations.

4.7.2 Electricity consumption prediction

We conclude these experiments with a study of the UCI dataset 'ElectricityLoadDiagrams20112014' (Dua and Graff, 2017), later called Electricity Loads. It consists of the electricity consumption of 370 clients, recorded every 15min from 2011 to 2014. We average the data to obtain hourly data and focus on the following task: given the electricity consumption of a subset of clients over

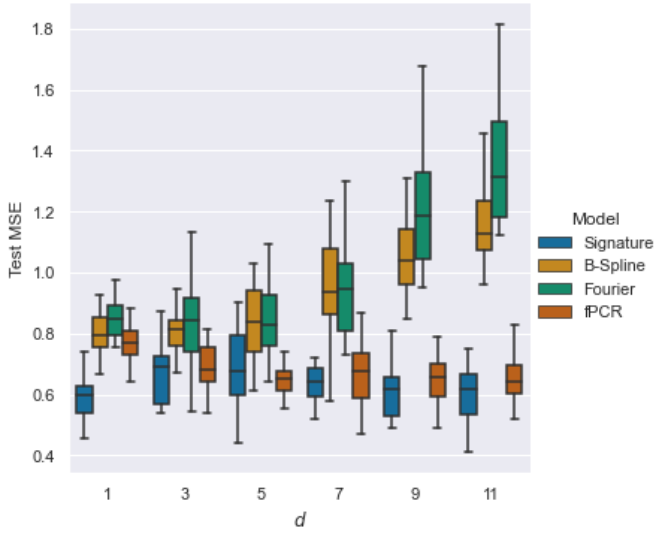


Figure 4.11 – Test MSE for the signature linear model, functional regression with B-Spline and Fourier basis functions, and functional Principal Component Regression (fPCR). The input time series are gaussian processes with a random linear trend, as defined by (4.13), and the reponse is the norm of the trend slope.

a week, we want to predict the consumption peak of the following week, that is, the maximal hourly consumption summed over all clients. We vary the number of clients observed, which allows us to do a similar analysis as in Section 4.6: each data point is a path in $\mathbb{R}^d$, where d is the number of clients observed. Such a sample with $d = 5$ is shown in Figure 4.15.

Note that when the number of clients d increases, we should expect the error to decrease because we just add new information in the data and the response remains unchanged. Figure 4.16 shows a boxplot of the test MSE over 20 random train/test splits of the data, for the four models considered. We can see that for d up to 5 all models perform similarly, then for $d = 10$ the variance of the Fourier and B-spline models increases and for d larger than 10 their errors increase a lot, whereas the error of the signature method and fPCR continue to decrease with a rather fixed variance. This confirms what has been observed on simulated data: the signature linear model is robust to dimension and performs similarly or better than traditional functional linear models.

4.8 Conclusion and perspectives

In this paper, we have provided a complete and ready-to-use methodology to implement the signature linear model. This led us to define a consistent estimator of the signature truncation order. We show on both simulated and real-world datasets that this model performs at least as well as traditional functional linear models, and is particularly relevant for vector-valued functions in high dimensions.

The signature is a flexible representation tool for multidimensional time series and can be used in various contexts. This study is just a first step towards understanding how it should be used in a statistical setting and there are a lot of potential extensions. For example, Figure 4.13 suggests that the vector of coefficients on the signature is sparse. Studying different sparsifying

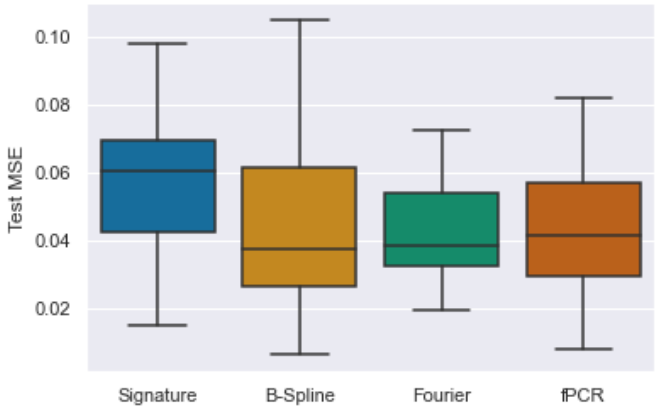


Figure 4.12 – Test MSE over 20 train/test splits for the Canadian Weather datasets for four different linear models: the signature linear model, functional regression with B-Spline and Fourier basis functions, and functional Principal Component Regression (fPCR).

procedures for signatures would be a valuable extension of our results. Another interesting topic would be to investigate statistical models with the logsignature transform, which is a more compact representation of the signature. The main difference is that the logsignature does not possess linear approximation properties such as Proposition 4.2 and therefore requires to depart from a linear model.

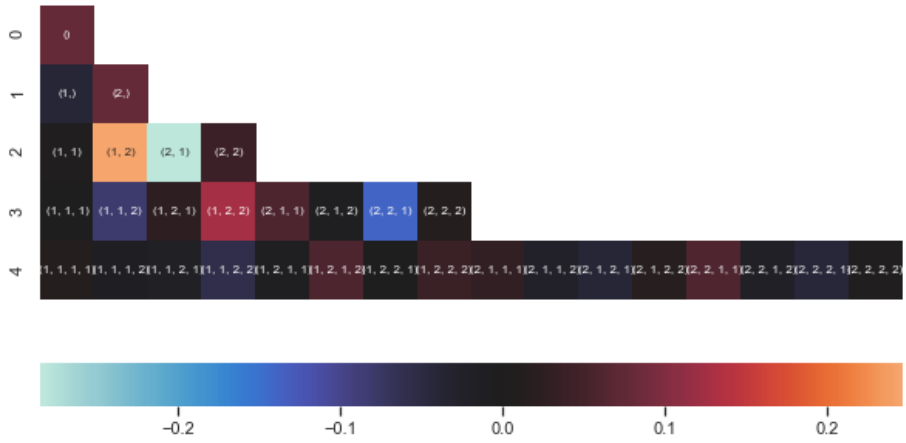


Figure 4.13 – Heatmap of coefficients up to order 4 obtained for the Canadian weather dataset signature regression. The vertical axis represents the order of the coefficients: on top the coefficient of order 0, then the two coefficients of order 1, then the four coefficients of order 2, and so on.

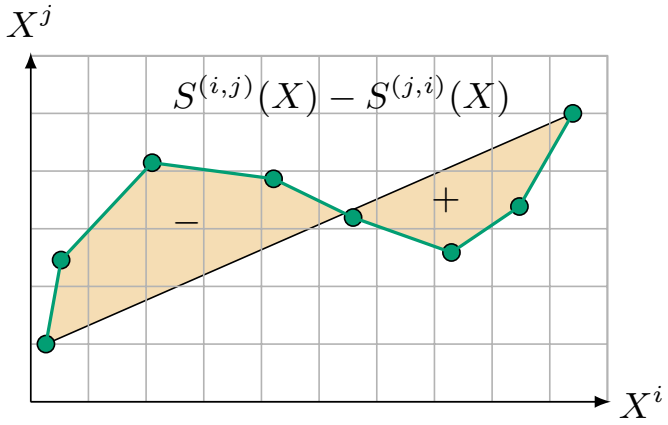


Figure 4.14 – The quantity $S^{(i,j)}(X) - S^{(j,i)}(X)$ corresponds to the sum of the signed orange areas, which is also known as the Levy area.

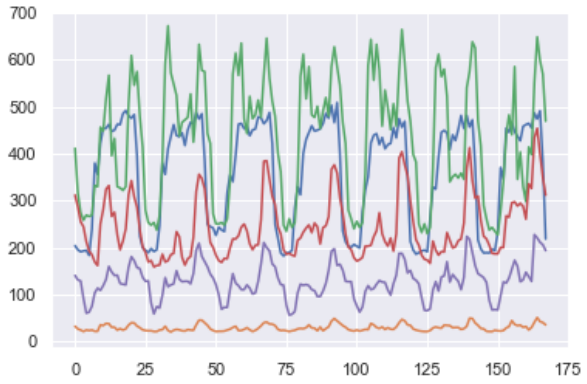


Figure 4.15 – One sample from the Electricity Loads dataset, where we observe the hourly energy consumption of 5 clients over a week.

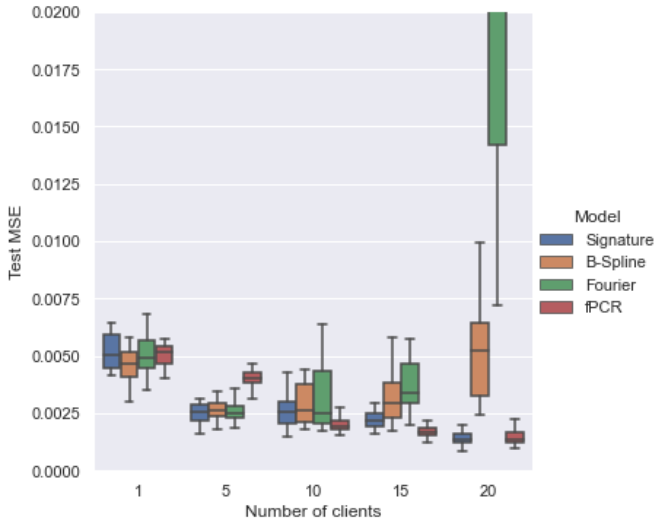


Figure 4.16 – Test MSE over 20 random train/test splits for four different linear models: the signature linear model, functional regression with B-Spline and Fourier basis functions, and functional Principal Component Regression (fPCR).

Bibliography

Arlot, S. (2019). Minimal penalties and the slope heuristics: a survey. *Journal de la Société Française de Statistique*, *160*, 1–106.

Arribas, I. P., Goodwin, G. M., Geddes, J. R., Lyons, T., and Saunders, K. E. (2018). A signature-based machine learning model for distinguishing bipolar disorder and borderline personality disorder. *Translational psychiatry*, *8*, 1–7.

Arribas, I. P., Salvi, C., and Szpruch, L. (2020). Sig-SDEs model for quantitative finance. *arXiv:2006.00218*.

Benzeghiba, M., De Mori, R., Deroo, O., Dupont, S., Erbes, T., Jouvet, D., Fissore, L., Laface, P., Mertins, A., Ris, C., et al. (2007). Automatic speech recognition and speech variability: a review. *Speech communication*, *49*, 763–786.

Birgé, L., and Massart, P. (2007). Minimal penalties for gaussian model selection. *Probability Theory and Related Fields*, *138*, 33–73.

Brunel, É., Mas, A., and Roche, A. (2016). Non-asymptotic adaptive prediction in functional linear models. *Journal of Multivariate Analysis*, *143*, 208–232.

Cardot, H., Ferraty, F., and Sarda, P. (1999). Functional linear model. *Statistics & Probability Letters*, *45*(1), 11–22.

Cardot, H., Ferraty, F., and Sarda, P. (2003). Spline estimators for the functional linear model. *Statistica Sinica*, 571–591.

Chen, K.-T. (1958). Integration of paths—a faithful representation of paths by non-commutative formal power series. *Transactions of the American Mathematical Society*, *89*, 395–407.

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

Dua, D., and Graff, C. (2017). UCI machine learning repository. http://archive.ics.uci.edu/ml

Fermanian, A. (2021). Embedding and learning with signatures. *Computational Statistics & Data Analysis*, *157*, 107148.

Ferraty, F., and Vieu, P. (2006). *Nonparametric functional data analysis: theory and practice*. Springer.

Frank, L. E., and Friedman, J. H. (1993). A statistical view of some chemometrics regression tools. *Technometrics*, *35*, 109–135.

Friz, P. K., and Victoir, N. B. (2010). *Multidimensional stochastic processes as rough paths: theory and applications* (Vol. 120). Cambridge University Press.

Greven, S., Crainiceanu, C., Caffo, B., and Reich, D. (2011). Longitudinal functional principal component analysis. *Recent advances in functional data analysis and related topics* (pp. 149–154). Springer.

Hall, P., Horowitz, J. L. et al. (2007). Methodology and convergence rates for functional linear regression. *The Annals of Statistics*, *35*, 70–91.

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

Hastie, T., and Mallows, C. (1993). [a statistical view of some chemometrics regression tools]: discussion. *Technometrics*, *35*, 140–143.

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

Lai, S., Jin, L., and Yang, W. (2017). Online signature verification using recurrent neural network and length-normalized path signature descriptor. *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, *1*, 400–405.

Levin, D., Lyons, T., and Ni, H. (2013). Learning from the past, predicting the statistics for the future, learning an evolving system. *arXiv:1309.0260*.

Li, C., Zhang, X., and Jin, L. (2017). LPSNet: a novel log path signature feature based hand gesture recognition framework. *2017 IEEE International Conference on Computer Vision Workshop*, 631–639.

Li, Y., and Hsing, T. (2007). On rates of convergence in functional linear regression. *Journal of Multivariate Analysis*, *98*, 1782–1804.

Liu, M., Jin, L., and Xie, Z. (2017). Ps-lstm: capturing essential sequential online information with path signature and lstm for writer identification. *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, *1*, 664–669.

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

Lyons, T., Caruana, M., and Lévy, T. (2007). *Differential equations driven by rough paths* (Vol. 1908). Springer.

Marx, B. D., and Eilers, P. H. (1999). Generalized linear regression on sampled signals and curves: a P-spline approach. *Technometrics*, *41*, 1–13.

Moore, P., Lyons, T., and Gallacher, J. (2019). Using path signatures to predict a diagnosis of Alzheimer's disease. *PloS ONE*, *14*.

Morrill, J., Fermanian, A., Kidger, P., and Lyons, T. (2020a). A generalised signature method for multivariate time series feature extraction. *arXiv:2006.00873*.

Morrill, J., Kormilitzin, A., Nevado-Holgado, A., Swaminathan, S., Howison, S., and Lyons, T. (2019). The signature-based model for early detection of sepsis from electronic health records in the intensive care unit. *International Conference in Computing in Cardiology*.

Morrill, J. H., Kormilitzin, A., Nevado-Holgado, A. J., Swaminathan, S., Howison, S. D., and Lyons, T. J. (2020c). Utilization of the signature method to identify the early onset of sepsis from multivariate physiological time series in critical care monitoring. *Critical Care Medicine*, *48*, e976–e981.

Morris, J. S. (2015). Functional regression. *Annual Review of Statistics and Its Application*, *2*, 321–359.

Park, S. Y., and Staicu, A.-M. (2015). Longitudinal functional data analysis. *Stat*, *4*, 212–226.

Ramos-Carreño, C., Torrecilla, J. L., and Suárez, A. (2019). Scikit-fda: a python package for functional data analysis. *3rd International Workshop on Advances in Functional Data Analysis*, *5*.

Ramsay, J. O., and Dalzell, C. (1991). Some tools for functional data analysis. *Journal of the Royal Statistical Society. Series B (Methodological)*, *53*, 539–561.

Ramsay, J. O., and Silverman, B. W. (2005). *Functional data analysis. 2nd edition.* Springer.

Reizenstein, J., and Graham, B. (2020). Algorithm 1004: the iisignature library: efficient calculation of iterated-integral signatures and log signatures. *ACM Transactions on Mathematical Software*.

Turaga, P., Chellappa, R., Subrahmanian, V. S., and Udrea, O. (2008). Machine recognition of human activities: a survey. *IEEE Transactions on Circuits and Systems for Video technology*, *18*, 1473–1488.

Wang, B., Liakata, M., Ni, H., Lyons, T., Nevado-Holgado, A. J., and Saunders, K. (2019). A path signature approach for speech emotion recognition. *Interspeech 2019*, 1661–1665.

Yang, W., Jin, L., and Liu, M. (2015). Chinese character-level writer identification using path signature feature, dropstroke and deep cnn. *Proceedings of the 13th International Conference on Document Analysis and Recognition (ICDAR)*, 546–550.

Yang, W., Jin, L., and Liu, M. (2016a). DeepWriterID: An end-to-end online text-independent writer identification system. *IEEE Intelligent Systems*, *31*, 45–53.

Yang, W., Lyons, T., Ni, H., Schmid, C., Jin, L., and Chang, J. (2017). Developing the path signature methodology and its application to landmark-based human action recognition. *arXiv:1707.03993*.

Chapter 5

Framing RNN as a kernel method: A neural ODE approach

Building on the interpretation of a recurrent neural network (RNN) as a continuous-time neural differential equation, we show, under appropriate conditions, that the solution of a RNN can be viewed as a linear function of a specific feature set of the input sequence, known as the signature. This connection allows us to frame a RNN as a kernel method in a suitable reproducing kernel Hilbert space. As a consequence, we obtain theoretical guarantees on generalization and stability for a large class of recurrent networks. Our results are illustrated on simulated datasets.

Contents

5.1 Introduction

Recurrent neural networks (RNN) are among the most successful methods for modeling sequential data. They have achieved state-of-the-art results in difficult problems such as natural language processing (e.g., Mikolov et al., 2010; Collobert et al., 2011) or speech recognition (e.g., Hinton et al., 2012; Graves et al., 2013). This class of neural networks has a natural interpretation in terms of (discretization of) ordinary differential equations (ODE), which casts them in the field of neural ODE (Chen et al., 2018). This observation has led to the development of

continuous-depth models for handling irregularly-sampled time-series data, including the ODE-RNN model (Rubanova et al., 2019), GRU-ODE-Bayes (De Brouwer et al., 2019), or neural CDE models (Kidger et al., 2020a; Morrill et al., 2020b). In addition, the time-continuous interpretation of RNN allows to leverage the rich theory of differential equations to develop new recurrent architectures (Chang et al., 2019; Herrera et al., 2021; Erichson et al., 2021), which are better at learning long-term dependencies.

On the other hand, the development of kernel methods for deep learning offers theoretical insights on the functions learned by the networks (Cho and Saul, 2009; Belkin et al., 2018; Jacot et al., 2018). Here, the general principle consists in defining a reproducing kernel Hilbert space (RKHS)—that is, a function class $\mathscr{H}$—, which is rich enough to describe the architectures of networks. A good example is the construction of Bietti and Mairal (2017), 2019, who exhibit a RKHS for convolutional neural networks. This kernel perspective has several advantages. First, by separating the representation of the data from the learning process, it allows to study invariances of the representations learned by the network. Next, by reducing the learning problem to a linear one in $\mathscr{H}$, generalization bounds can be more easily obtained. Finally, the Hilbert structure of $\mathscr{H}$ provides a natural metric on neural networks, which can be used for example for regularization (Bietti et al., 2019).

Contributions. By taking advantage of the neural ODE paradigm for RNN, we show that RNN are, in the continuous-time limit, linear predictors over a specific space associated with the signature of the input sequence (Levin et al., 2013). The signature transform, first defined by Chen (1958) and central in rough path theory (Lyons et al., 2007; Friz and Victoir, 2010), summarizes sequential inputs by a graded feature set of their iterated integrals. Its natural environment is a tensor space that can be endowed with a RKHS structure (Király and Oberhauser, 2019). We exhibit general conditions under which classical recurrent architectures such as feedforward RNN, Gated Recurrent Units (GRU, Cho et al., 2014), or Long Short-Term Memory networks (LSTM, Hochreiter and Schmidhuber, 1997), can be framed as a kernel method in this RKHS. This enables us to provide generalization bounds for RNN as well as stability guarantees via regularization. The theory is illustrated with some experimental results.

Related works. The neural ODE paradigm was first formulated by Chen et al. (2018) for residual neural networks. It was then extended to RNN in several articles, with a focus on handling irregularly sampled data (Rubanova et al., 2019; Kidger et al., 2020a) and learning long-term dependencies (Chang et al., 2019). The signature transform has recently received the attention of the machine learning community (Levin et al., 2013; Kidger et al., 2019; Liao et al., 2019; Toth and Oberhauser, 2020; Fermanian, 2021) and, combined with deep neural networks, has achieved state-of-the-art performance for several applications (Yang et al., 2016a; Yang et al., 2017; Perez Arribas, 2018; Wang et al., 2019; Morrill et al., 2020c). Király and Oberhauser (2019) use the signature transform to define kernels for sequential data and develop fast computational methods. The connection between continuous-time RNN and signatures has been pointed out by Lim (2021) for a specific model of stochastic RNN. Deriving generalization bounds for RNN is an active research area (Zhang et al., 2018; Akpinar et al., 2019; Tu et al., 2019). By leveraging the theory of differential equations, our approach encompasses a large class of RNN models, ranging from feedforward RNN to LSTM. This is in contrast with most existing generalization bounds, which are architecture-dependent. Close to our point of view is the work of Bietti and Mairal (2017) for convolutional neural networks.

Mathematical context. We place ourselves in a supervised learning setting. The input data is a sample of n i.i.d. vector-valued sequences $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)}\}$, where $\mathbf{x}^{(i)} = (x_1^{(i)}, \dots, x_T^{(i)}) \in (\mathbb{R}^d)^T$,

$T \geq 1$. The outputs of the learning problem can be either labels (classification setting) or sequences (sequence-to-sequence setting). Even if we only observe discrete sequences, each $\mathbf{x}^{(i)}$ is mathematically considered as a regular discretization of a continuous-time process $X^{(i)} \in BV([0,1], \mathbb{R}^d)$, where $BV([0,1], \mathbb{R}^d)$ is the space of continuous functions from $[0,1]$ to $\mathbb{R}^d$ of finite total variation. Informally, the total variation of a process corresponds to its length. Formally, for any $[s,t] \subset [0,1]$, the total variation of a process $X \in BV([0,1], \mathbb{R}^d)$ on $[s,t]$ is defined by

$$\|X\|_{TV;[s,t]} = \sup_{(t_0,\dots,t_k)\in D_{s,t}} \sum_{j=1}^{k} \|X_{t_j} - X_{t_{j-1}}\|,$$

where $D_{s,t}$ denotes the set of all finite partitions of $[s,t]$ and $\|\cdot\|$ the Euclidean norm. We therefore have that $x_j^{(i)} = X_{j/T}^{(i)}$, $1 \leq j \leq T$, where $X_t^{(i)} := X^{(i)}(t)$. We make two assumptions on the processes $X^{(i)}$. First, they all begin at zero, and second, their lengths are bounded by $L \in (0,1)$. These assumptions are not too restrictive, since they amount to data translation and normalization, common in practice. Accordingly, we denote by $\mathscr{X}$ the subspace of $BV([0,1], \mathbb{R}^d)$ defined by

$$\mathscr{X} = \left\{X \in BV([0,1], \mathbb{R}^d) \,|\, X_0 = 0 \quad \text{and} \quad \|X\|_{TV;[0,1]} \leq L\right\}$$

and assume therefore that $X^{(1)}, \dots, X^{(n)}$ are i.i.d. according to some $X \in \mathscr{X}$. The norm on all spaces $\mathbb{R}^m$, $m \geq 1$, is always the Euclidean one. Observe that assuming that $X \in \mathscr{X}$ implies that, for any $t \in [0,1]$, $\|X_t\| = \|X_t - X_0\| \leq \|X\|_{TV;[0,1]} \leq L$.

Recurrent neural networks. Classical RNN are defined by a sequence of hidden states $h_1, \dots, h_T \in \mathbb{R}^e$, where, for $\mathbf{x} = (x_1, \dots, x_T)$ a generic data sample,

$$h_{j+1} = f(h_j, x_{j+1}), \quad h_0 = 0.$$

At each time step $1 \leq j \leq T$, the output of the network is $z_j = \psi(h_j)$, where ψ is a linear function. In the present article, we rather consider the following residual version, which is a natural adaptation of classical RNN in the neural ODE framework (see, e.g., Yue et al., 2018):

$$h_{j+1} = h_j + \frac{1}{T} f(h_j, x_{j+1}), \quad h_0 = 0. \tag{5.1}$$

The simplest choice for the function f is the feedforward model, say f_{RNN}, defined by

$$f_{\text{RNN}}(h, x) = \sigma(Uh + Vx + b), \tag{5.2}$$

where σ is an activation function, $U \in \mathbb{R}^{e\times e}$ and $V \in \mathbb{R}^{e\times d}$ are weight matrices, and $b \in \mathbb{R}^e$ is the bias. The function f_{RNN}, equipped with a smooth activation σ (such as the logistic or hyperbolic tangent functions), will be our leading example throughout the paper. However, the GRU and LSTM models can also be rewritten under the form (5.1), as shown in Appendix D.1.1. Thus, model (5.1) is flexible enough to encompass most recurrent networks used in practice.

Overview. Section 5.2 is devoted to framing RNN as linear functions in a suitable RKHS. We start by embedding iteration (5.1) into a continuous-time model, which takes the form of a controlled differential equation (CDE). This allows, after introducing the signature transform, to define the appropriate RKHS, and, in turn, to show that model (5.1) boils down, in the continuous-time limit, to a linear problem on the signature. This framework is used in Section 5.3 to derive generalization bounds and stability guarantees. We conclude with some experiments

in Section 5.4. All proofs are postponed to the supplementary material.

5.2 Framing RNN as a kernel method

Roadmap. First, we quantify the difference between the discrete recurrent network (5.1) and its continuous-time counterpart (Proposition 5.1). Then, we rewrite the corresponding ODE as a CDE (Proposition 5.2). Under appropriate conditions, Proposition 5.4 shows that the solution of this equation is a linear function of the signature of the driving process. Importantly, these assumptions are valid for a feedforward RNN, as stated by Proposition 5.5. We conclude in Theorem 5.6.

5.2.1 From discrete to continuous time

Recall that $h_0, \dots, h_T$ denote the hidden states of the RNN (5.1), and let $H : [0,1] \to \mathbb{R}^e$ be the solution of the ODE

$$dH_t = f(H_t, X_t)dt, \quad H_0 = h_0. \tag{5.3}$$

By bounding the difference between $H_{j/T}$ and h_j, the following proposition shows how to pass from discrete to continuous time, provided f satisfies the following assumption:

(A_1) The function f is Lipschitz continuous in h and x, with Lipschitz constants K_h and K_x. We let $K_f = \max(K_h, K_x)$.

Proposition 5.1. *Assume that* (A_1) *is verified. Then there exists a unique solution* H *to* (5.3) *and, for any* $0 \le j \le T$,

$$\|H_{j/T} - h_j\| \le \frac{c_1}{T},$$

where $c_1 = K_f e^{K_f}\big(L + \sup_{\|h\|\le M, \|x\|\le L} \|f(h,x)\| e^{K_f}\big)$ *and* $M = \sup_{\|x\|\le L} \|f(h_0,x)\| e^{K_f}$. *Moreover, for any* $t \in [0,1]$, $\|H_t\| \le M$.

Then, following Kidger et al. (2020a), we show that the ODE (5.3) can be rewritten under the form of a CDE. At the cost of increasing the dimension of the hidden state from e to $e+d$, this allows us to reframe model (5.3) as a linear model in dX, in the sense that X has been moved 'outside' of f.

Proposition 5.2. *Assume that* (A_1) *is verified. Let* $H : [0,1] \to \mathbb{R}^e$ *be the solution of* (5.3), *and let* $\bar{X} : [0,1] \to \mathbb{R}^{d+1}$ *be the time-augmented process* $\bar{X}_t = (X_t^\top, \frac{1-L}{2}t)^\top$. *Then there exists a tensor field* $\mathbf{F} : \mathbb{R}^{\bar{e}} \to \mathbb{R}^{\bar{e}\times\bar{d}}$, $\bar{e} = e + d$, $\bar{d} = d + 1$, *such that if* $\bar{H} : [0,1] \to \mathbb{R}^{\bar{e}}$ *is the solution of the CDE*

$$d\bar{H}_t = \mathbf{F}(\bar{H}_t)d\bar{X}_t, \quad \bar{H}_0 = (H_0^\top, X_0^\top)^\top, \tag{5.4}$$

then its first e *coordinates are equal to* H.

Equation (5.4) can be better understood by the following equivalent integral equation:

$$\bar{H}_t = \bar{H}_0 + \int_0^t \mathbf{F}(\bar{H}_u)d\bar{X}_u,$$

where the integral should be understood as Riemann-Stieljes integral (Friz and Victoir, 2010, Section I.2). Thus, the output of the RNN can be approximated by the solution of the CDE (5.4), and, according to Proposition 5.1, the approximation error is $\mathcal{O}(1/T)$.

Example 5.1. *Consider f_{RNN} as in (5.2). If σ is Lipschitz continuous with constant K_σ, then, for any $h_1, h_2 \in \mathbb{R}^e$, $x_1, x_2 \in \mathbb{R}^d$,*

$$\begin{aligned}\|f_{\mathrm{RNN}}(h_1, x_1) - f_{\mathrm{RNN}}(h_2, x_1)\| &= \|\sigma(Uh_1 + Vx_1 + b) - \sigma(Uh_2 + Vx_1 + b)\| \\ &\leq K_\sigma \|U\|_{\mathrm{op}} \|h_1 - h_2\|,\end{aligned}$$

where $\|\cdot\|_{\mathrm{op}}$ denotes the operator norm—see Appendix D.1.3. Similarly, $\|f(h_1, x_1) - f(h_1, x_2)\| \leq K_\sigma \|V\|_{\mathrm{op}} \|x_1 - x_2\|$. Thus, assumption (A_1) is satisfied. The tensor field $\mathbf{F}_{\mathrm{RNN}}$ of Proposition 5.2 corresponding to this network is defined for any $\bar{h} \in \mathbb{R}^{\bar{e}}$ by

$$\mathbf{F}_{\mathrm{RNN}}(\bar{h}) = \begin{pmatrix} 0_{e\times d} & \frac{2}{1-L}\sigma(W\bar{h} + b) \\ I_{d\times d} & 0_{d\times 1} \end{pmatrix}, \quad \text{where} \quad W = \begin{pmatrix} U & V \end{pmatrix} \in \mathbb{R}^{e\times \bar{e}}. \tag{5.5}$$

5.2.2 The signature

An essential ingredient towards our construction is the signature of a continuous-time process, which we briefly present here. We refer to Chevyrev and Kormilitzin (2016a) for a gentle introduction and to Lyons et al. (2007) and Levin et al. (2013) for details.

Tensor Hilbert spaces. We denote by $(\mathbb{R}^d)^{\otimes k}$ the kth tensor power of $\mathbb{R}^d$ with itself, which is a Hilbert space of dimension d^k. The key space to define the signature and, in turn, our RKHS, consists in infinite square-summable sequences of tensors of increasing order:

$$\mathscr{T} = \left\{ a = (a_0, \dots, a_k, \dots) \,\middle|\, a_k \in (\mathbb{R}^d)^{\otimes k}, \sum_{k=0}^{\infty} \|a_k\|^2_{(\mathbb{R}^d)^{\otimes k}} < \infty \right\}. \tag{5.6}$$

Endowed with the scalar product $\langle a, b\rangle_{\mathscr{T}} := \sum_{k=0}^{\infty} \langle a_k, b_k\rangle_{(\mathbb{R}^d)^{\otimes k}}$, $\mathscr{T}$ is a Hilbert space, as shown in Appendix D.1.4.

Definition 5.1. *Let $X \in BV([0,1], \mathbb{R}^d)$. For any $t \in [0,1]$, the signature of X on $[0,t]$ is defined by $S_{[0,t]}(X) = (1, \mathbb{X}^1_{[0,t]}, \dots, \mathbb{X}^k_{[0,t]}, \dots)$, where, for each $k \geq 1$,*

$$\mathbb{X}^k_{[0,t]} = k! \int_{0\leq u_1 < \cdots < u_k \leq t} \cdots \int dX_{u_1} \otimes \cdots \otimes dX_{u_k} \in (\mathbb{R}^d)^{\otimes k}.$$

Although this definition is technical, the signature should simply be thought of as a feature map that embeds a bounded variation process into an infinite-dimensional tensor space. The signature has several good properties that make it a relevant tool for machine learning (e.g., 2013; Chevyrev and Kormilitzin, 2016a; Fermanian, 2021). In particular, under certain assumptions, $S(X)$ characterizes X up to translations and reparameterizations, and has good approximation properties. We also highlight that fast libraries exist for computing the signature (Reizenstein and Graham, 2020; Kidger and Lyons, 2020).

The expert reader is warned that this definition differs from the usual one by the normalization of $\mathbb{X}^k_{[0,t]}$ by $k!$, which is more adapted to our context. When the signature is taken on the whole interval $[0,1]$, we simply write $S(X)$ and $\mathbb{X}^k$. In the sequel, for any index $(i_1, \dots, i_k) \subset \{1, \dots, d\}^k$, $S^{(i_1,\dots,i_k)}_{[0,t]}(X)$ denotes the term associated with the coordinates $(i_1, \dots, i_k)$ of $\mathbb{X}^k_{[0,t]}$.

Example 5.2. *Let X be the d-dimensional linear path defined by $X_t = (a_1 + b_1 t, \dots, a_d + b_d t)^\top$, $a_i, b_i \in \mathbb{R}$. Then $S^{(i_1,\dots,i_k)}(X) = b_{i_1} \cdots b_{i_k}$ and $\mathbb{X}^k = b^{\otimes k}$.*

The next proposition, which ensures that $S_{[0,t]}(\bar{X}) \in \mathscr{T}$, is an important step.

Proposition 5.3. *Let $X \in \mathscr{X}$ and $\bar{X}_t = (X_t^\top, \frac{1-L}{2}t)^\top$ as in Proposition 5.2. Then, for any $t \in [0,1]$, $\|S_{[0,t]}(\bar{X})\|_{\mathscr{T}} \leq 2(1-L)^{-1}$.*

The signature kernel. By taking advantage of the structure of Hilbert space of $\mathscr{T}$, it is natural to introduce the following kernel:

$$\begin{aligned} K : \mathscr{X} \times \mathscr{X} &\to \mathbb{R} \\ (X, Y) &\mapsto \langle S(\bar{X}), S(\bar{Y}) \rangle_{\mathscr{T}}, \end{aligned}$$

which is well defined according to Proposition 5.3. We refer to Király and Oberhauser (2019) for a general presentation of kernel methods with signatures and to Salvi et al. (2020) for a kernel trick. The RKHS associated with K is the space of functions

$$\mathscr{H} = \left\{ \xi_\alpha : \mathscr{X} \to \mathbb{R} \,|\, \xi_\alpha(X) = \langle \alpha, S(\bar{X}) \rangle_{\mathscr{T}}, \alpha \in \mathscr{T} \right\}, \tag{5.7}$$

with scalar product $\langle \xi_\alpha, \xi_\beta \rangle_{\mathscr{H}} = \langle \alpha, \beta \rangle_{\mathscr{T}}$ (see, e.g., Schölkopf and Smola, 2002).

5.2.3 From the CDE to the signature kernel

An important property of signatures is that the solution of the CDE (5.4) can be written, under certain assumptions, as a linear function of the signature of the driving process X. This operation can be thought of as a Taylor expansion for CDE. More precisely, let us rewrite (5.4) as

$$dH_t = \mathbf{F}(H_t) dX_t = \sum_{i=1}^{d} F^i(H_t) dX_t^i, \tag{5.8}$$

where $X_t = (X_t^1, \dots, X_t^d)^\top$, $\mathbf{F} : \mathbb{R}^e \to \mathbb{R}^{e \times d}$, and $F^i : \mathbb{R}^e \to \mathbb{R}^e$ are the columns of $\mathbf{F}$—to avoid heavy notation, we momentarily write e, d, H, and X instead of $\bar{e}$, $\bar{d}$, $\bar{H}$, and $\bar{X}$. Throughout, the bold notation is used to distinguish tensor fields and vector fields. We recall that a vector field $F : \mathbb{R}^e \to \mathbb{R}^e$ or a tensor field $\mathbf{F} : \mathbb{R}^e \to \mathbb{R}^{e \times d}$ are said to be smooth if each of their coordinates is $\mathscr{C}^\infty$.

Definition 5.2. *Let $F, G : \mathbb{R}^e \to \mathbb{R}^e$ be smooth vector fields and denote by $J(\cdot)$ the Jacobian matrix. Their differential product is the smooth vector field $F \star G : \mathbb{R}^e \to \mathbb{R}^e$ defined, for any $h \in \mathbb{R}^e$, by*

$$(F \star G)(h) = \sum_{j=1}^{e} \frac{\partial G}{\partial h_j}(h) F_j(h) = J(G)(h) F(h).$$

In differential geometry, $F \star G$ is simply denoted by FG. Since the $\star$ operation is not associative, we take the convention that it is evaluated from right to left, i.e., $F^1 \star F^2 \star F^3 := F^1 \star (F^2 \star F^3)$.

Taylor expansion. Let H be the solution of (5.8), where $\mathbf{F}$ is assumed to be smooth. We now show that H can be written as a linear function of the signature of X, which is the crucial

step to embed the RNN in the RKHS $\mathscr{H}$. The step-N Taylor expansion of H (Friz and Victoir, 2008) is defined by

$$H_t^N = H_0 + \sum_{k=1}^{N} \frac{1}{k!} \sum_{1 \le i_1, \dots, i_k \le d} S_{[0,t]}^{(i_1,\dots,i_k)}(X) F^{i_1} \star \cdots \star F^{i_k}(H_0).$$

Throughout, we let

$$\Lambda_k(\mathbf{F}) = \sup_{\|h\| \le M, 1 \le i_1, \dots, i_k \le d} \|F^{i_1} \star \cdots \star F^{i_k}(h)\|.$$

Example 5.3. *Let $\mathbf{F} = \mathbf{F}_{\mathrm{RNN}}$ defined by (5.5) with an identity activation. Then, for any $\bar{h} \in \mathbb{R}^{\bar{e}}$, $1 \le i \le d+1$, $F_{\mathrm{RNN}}^i(\bar{h}) = W_i \bar{h} + b_i$, where b_i is the $(i+d)$th vector of the canonical basis of $\mathbb{R}^{\bar{e}}$, and*

$$W_i = 0_{\bar{e} \times \bar{e}}, \quad W_{d+1} = \begin{pmatrix} \frac{2}{1-L} W \\ 0_{d \times \bar{e}} \end{pmatrix}, \quad \text{and} \quad b_{d+1} = \begin{pmatrix} \frac{2}{1-L} b \\ 0_d \end{pmatrix}.$$

The vector fields F_{RNN}^i are then affine, $J(F_{\mathrm{RNN}}^i) = W_i$, and the iterated star products have a simple expression: for any $1 \le i_1, \dots, i_k \le d$, $F_{\mathrm{RNN}}^{i_1} \star \cdots \star F_{\mathrm{RNN}}^{i_k}(\bar{h}) = W_{i_k} \cdots W_{i_2}(W_{i_1} \bar{h} + b_{i_1})$.

The next proposition shows that the step-N Taylor expansion H^N is a good approximation of H.

Proposition 5.4. *Assume that the tensor field $\mathbf{F}$ is smooth. Then, for any $t \in [0,1]$,*

$$\|H_t - H_t^N\| \le \frac{d^{N+1}}{(N+1)!} \Lambda_{N+1}(\mathbf{F}). \tag{5.9}$$

Thus, provided that $\Lambda_N(\mathbf{F})$ is not too large, the right-hand side of (5.9) converges to zero, hence

$$H_t = H_0 + \sum_{k=1}^{\infty} \frac{1}{k!} \sum_{1 \le i_1, \dots, i_k \le d} S_{[0,t]}^{(i_1,\dots,i_k)}(X) F^{i_1} \star \cdots \star F^{i_k}(H_0).$$

We conclude from the above representation that the solution H of (5.8) is in fact a linear function of the signature of X. A natural concern is to know whether the upper bound of Proposition 5.4 vanishes with N for standard architectures. This property is encapsulated in the following more general assumption:

$$(A_2) \quad \text{The tensor field } \mathbf{F} \text{ is smooth and } \sum_{k=0}^{\infty} \left(\frac{d^k}{k!} \Lambda_k(\mathbf{F}) \right)^2 < \infty.$$

Clearly, if (A_2) is verified, then the right-hand side of (5.9) converges to 0. The next proposition states formally the conditions under which (A_2) is verified for $\mathbf{F}_{\mathrm{RNN}}$. It is further illustrated in Figure 5.1, which shows that the convergence is fast with two common activation functions. We let $\|\sigma\|_\infty = \sup_{\|h\| \le M, \|x\| \le L} \|\sigma(Uh + Vx + b)\|$ and $\|\sigma^{(k)}\|_\infty = \sup_{\|h\| \le M, \|x\| \le L} \|\sigma^{(k)}(Uh + Vx + b)\|$.

Proposition 5.5. *Let $\mathbf{F}_{RNN}$ be defined by (5.5). If σ is the identity function, then (A_2) is satisfied. In the general case, (A_2) holds if σ is smooth and there exists $a > 0$ such that, for any $k \ge 0$,*

$$\|\sigma^{(k)}\|_\infty \le a^{k+1} k! \quad \text{and} \quad \|W\|_F < \frac{1-L}{8a^2 d}, \tag{5.10}$$

where $\|\cdot\|_F$ *is the Frobenius norm. Moreover,* $\Lambda_N(\mathbf{F}_{RNN}) \leq \sqrt{2}a\Big(\frac{8a^2\|W\|_F}{1-L}\Big)^{N-1}N!$.

The proof of Proposition 5.5, based on the manipulation of higher-order derivatives of tensor fields, is highly non-trivial. We highlight that the conditions on σ are mild and verified for common smooth activations. For example, they are verified for the logistic function (with $a = 2$) and for the hyperbolic tangent function (with $a = 4$)—see Appendix D.1.5. The second inequality of (5.10) puts a constraint on the norm of the weights, and can be regarded as a radius of convergence for the Taylor expansion.

Putting everything together. We now have all the elements at hand to embed the RNN into the RKHS $\mathscr{H}$. To fix the idea, we assume in this paragraph that we are in a ± 1 classification setting. In other words, given an input sequence $\mathbf{x}$, we are interested in the final output $z_T = \psi(h_T) \in \mathbb{R}$, where h_T is the solution of (5.1). The predicted class is $\mathbf{1}(z_T > 0)$.

By Propositions 5.1 and 5.2, z_T is approximated by the first e coordinates of the solution of the CDE (5.4), which outputs a $\mathbb{R}^{e+d}$-valued process $\bar{H}$. According to Proposition 5.4, $\bar{H}$ is a linear function of the signature of the time-augmented process $\bar{X}$. Thus, on top of $\bar{H}$, it remains to successively apply the projection Proj on the e first coordinates followed by the linear function ψ to obtain an element of the RKHS $\mathscr{H}$. This mechanism is summarized in the following theorem.

Theorem 5.6. *Assume that* (A_1) *and* (A_2) *are verified. Then there exists a function* $\xi_\alpha \in \mathscr{H}$ *such that*

$$|z_T - \xi_\alpha(X)| \leq \|\psi\|_{\text{op}} \frac{c_1}{T}, \tag{5.11}$$

where $\xi_\alpha(X) = \langle \alpha, S(\bar{X})\rangle_{\mathscr{T}}$ *and* $\bar{X}_t = (X_t^\top, \frac{1-L}{2}t)^\top$. *We have* $\alpha = (\alpha_k)_{k=0}^\infty$, *where each* $\alpha_k \in (\mathbb{R}^d)^{\otimes k}$ *is defined by*

$$\alpha_k^{(i_1,\dots,i_k)} = \frac{1}{k!}\psi \circ \text{Proj}\big(F^{i_1} \star \cdots \star F^{i_k}(\bar{H}_0)\big).$$

Moreover, $\|\alpha\|_{\mathscr{T}}^2 \leq \|\psi\|_{\text{op}}^2 \sum_{k=0}^\infty \Big(\frac{d^k}{k!}\Lambda_k(\mathbf{F})\Big)^2$.

We conclude that in the continuous-time limit, the output of the network can be interpreted as a scalar product between the signature of the (time-augmented) process $\bar{X}$ and an element of $\mathscr{T}$. This interpretation is important for at least two reasons: (i) it facilitates the analysis of generalization of RNN by leveraging the theory of kernel methods, and (ii) it provides new insights on regularization strategies to make RNN more robust. These points will be explored in the next section. Finally, we stress that the approach works for a large class of RNN, such as GRU and LSTM. The derivation of conditions (A_1) and (A_2) beyond the feedforward RNN is left for future work.

5.3 Generalization and regularization

5.3.1 Generalization bounds

Learning procedure. A first consequence of framing a RNN as a kernel method is that it gives natural generalization bounds under mild assumptions. In the learning setup, we are given an i.i.d. sample of n random pairs of observations $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) \in (\mathbb{R}^d)^T \times \mathscr{Y}$, where $\mathbf{x}^{(i)} = (x_1^{(i)}, \dots, x_T^{(i)})$. We distinguish the binary classification problem, where $\mathscr{Y} = \{-1, 1\}$, from the sequential prediction problem, where $\mathscr{Y} = (\mathbb{R}^p)^T$ and $\mathbf{y}^{(i)} = (y_1^{(i)}, \dots, y_T^{(i)})$. The RNN is assumed

to be parameterized by $\theta \in \Theta \subset \mathbb{R}^q$, where Θ is a compact set. To clarify the notation, we use a θ subscript whenever a quantity depends on θ (e.g., f_θ for f, etc.). In line with Section 5.2, it is assumed that the tensor field $\mathbf{F}_\theta$ associated with f_θ satisfies (A_1) and (A_2), keeping in mind that Proposition 5.5 guarantees that these requirements are fulfilled by a feedforward recurrent network with a smooth activation function.

Let $g_\theta : (\mathbb{R}^d)^T \to \mathscr{Y}$ denote the output of the recurrent network. The parameter θ is fitted by empirical risk minimization using a loss function $\ell : \mathscr{Y} \times \mathscr{Y} \to \mathbb{R}^+$. The theoretical and empirical risks are respectively defined, for any $\theta \in \Theta$, by

$$\mathscr{R}(\theta) = \mathbb{E}[\ell(\mathbf{y}, g_\theta(\mathbf{x}))] \quad \text{and} \quad \widehat{\mathscr{R}}_n(\theta) = \frac{1}{n}\sum_{i=1}^n \ell\big(\mathbf{y}^{(i)}, g_\theta(\mathbf{x}^{(i)})\big),$$

where the expectation $\mathbb{E}$ is evaluated with respect to the distribution of the generic random pair $(\mathbf{x}, \mathbf{y})$. We let $\widehat{\theta}_n \in \operatorname{argmin}_{\theta\in\Theta} \widehat{\mathscr{R}}_n(\theta)$ and aim at upper bounding $\mathbb{P}(\mathbf{y} \neq g_{\widehat{\theta}_n}(\mathbf{x}))$ in the classification regime (Theorem 5.7) and $\mathscr{R}(\widehat{\theta}_n)$ in the sequential regime (Theorem 5.8). To reach this goal, our strategy is to approximate the RNN by its continuous version and then use the RKHS machinery of Section 5.2.

Binary classification. In this context, the network outputs a real number $g_\theta(\mathbf{x}) = \psi(h_T) \in \mathbb{R}$ and the predicted class is $\mathbf{1}(z_T > 0)$. The loss $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+$ is assumed to satisfy the assumptions of Bartlett and Mendelson (2002, Theorem 7), that is, for any $y \in \{-1, 1\}$, $\ell(\mathbf{y}, g_\theta(\mathbf{x})) = \phi(\mathbf{y} g_\theta(\mathbf{x}))$, where $\phi(u) \geq \mathbf{1}(u \leq 0)$, $\phi(0) = 0$, and ϕ is Lipschitz-continuous with constant K_ℓ. For example, the cross-entropy loss satisfies such assumptions. We let $\xi_{\alpha_\theta} \in \mathscr{H}$ be the function of Theorem 5.6 that approximates the RNN with parameter θ. Thus, $z_T \approx \xi_{\alpha_\theta}(\bar{X}) = \langle \alpha_\theta, S(\bar{X})\rangle_{\mathscr{T}}$, up to a $\mathscr{O}(1/T)$ term.

Theorem 5.7. *Assume that for all $\theta \in \Theta$, (A_1) and (A_2) are verified. Assume, in addition, that there exists a constant $B > 0$ such that for any $\theta \in \Theta$, $\|\xi_{\alpha_\theta}\|_{\mathscr{H}} \leq B$. Then with probability at least $1 - \delta$,*

$$\mathbb{P}\big(\mathbf{y} \neq g_{\widehat{\theta}_n}(\mathbf{x})\big) \leq \widehat{\mathscr{R}}_n(\widehat{\theta}_n) + \frac{c_2}{T} + \frac{16 K_\ell B}{(1-L)\sqrt{n}} + \frac{2BK_\ell}{1-L}\sqrt{\frac{\log(1/\delta)}{2n}}, \tag{5.12}$$

where $c_2 = K_\ell \sup_\theta \Big(\|\psi\|_{\mathrm{op}} K_{f_\theta} e^{K_{f_\theta}} \big(L + \|f_\theta\|_\infty e^{K_{f_\theta}}\big)\Big)$.

Close to our result are the bounds obtained by Zhang et al. (2018), Tu et al. (2019), and Chen et al. (2020). The main difference is that the term in $1/T$ does not usually appear, since it comes from the continuous viewpoint on RNN, whereas the speed in $1/\sqrt{n}$ is more classical. The take-home message is that the detour by continuous-time neural ODE provides a theoretical framework adapted to RNN, at the modest price of an additional $\mathscr{O}(1/T)$ term. Moreover, we note that the bound (5.12) is 'simple' and holds under mild conditions for a large class of RNN. More precisely, for any recurrent network of the form (5.1), provided (A_1) and (A_2) are satisfied, then (5.12) is valid with constants c_2 and B depending on the architecture. Such constants are given below in the example of a feedforward RNN.

Example 5.4. *Take a feedforward RNN with logistic activation, and $\Theta = \{(W, b, \psi) \,|\, \|W\|_F \leq K_W < (1-L)/32d, \|b\| \leq K_b, \|\psi\|_{\mathrm{op}} \leq K_\psi\}$. Then, Proposition 5.5 states that (A_2) is satisfied*

and, with Theorem 5.6, ensures that

$$\sup_{\theta\in\Theta}\|\xi_{\alpha_\theta}\|_{\mathscr{H}} \leq \frac{\sqrt{2}K_\psi(1-L)}{1-L-32dK_W} := B, \quad K_{f_\theta} = \max(\|U\|_{\text{op}}, \|V\|_{\text{op}}), \quad \textit{and} \quad \|f_\theta\|_\infty = 1.$$

Sequence-to-sequence learning. We conclude by showing how to extend both the RKHS embedding of Theorem 5.6 and the generalization bound of Theorem 5.7 to the setting of sequence-to-sequence learning. In this case, the output of the network is a sequence

$$g_\theta(\mathbf{x}) = (z_1, \dots, z_T) \in (\mathbb{R}^p)^T.$$

An immediate extension of Theorem 5.6 ensures that there exist p elements $\alpha_{1,\theta}, \dots, \alpha_{p,\theta} \in \mathscr{T}$ such that, for any $1 \leq j \leq T$,

$$\left\|z_j - \left(\langle\alpha_{1,\theta}, S_{[0,j/T]}(\bar{X})\rangle_{\mathscr{T}}, \dots, \langle\alpha_{p,\theta}, S_{[0,j/T]}(\bar{X})\rangle_{\mathscr{T}}\right)^\top\right\| \leq \|\psi\|_{\text{op}}\frac{c_1}{T}. \tag{5.13}$$

The properties of the signature guarantee that $S_{[0,j/T]}(X) = S(\tilde{X}_{[j]})$ where $\tilde{X}_{[j]}$ is the process equal to $\bar{X}$ on $[0, j/T]$ and then constant on $[j/T, 1]$—see Appendix D.1.6. With this trick, we have, for any $1 \leq \ell \leq p$, $\langle\alpha_{\ell,\theta}, S_{[0,j/T]}(\bar{X})\rangle_{\mathscr{T}} = \langle\alpha_{\ell,\theta}, S(\tilde{X}_{[j]})\rangle_{\mathscr{T}}$, so that we are back in $\mathscr{H}$. Observe that the only difference with (5.11) is that we consider vector-valued sequential outputs, which requires to introduce the process $\tilde{X}_{[j]}$, but that the rationale is exactly the same.

We let $\ell : (\mathbb{R}^p)^T \times (\mathbb{R}^p)^T \to \mathbb{R}^+$ be the L_2 distance, that is, for any $\mathbf{y} = (y_1, \dots, y_T)$, $\mathbf{y}' = (y'_1, \dots, y'_T)$, $\ell(\mathbf{y}, \mathbf{y}') = \frac{1}{T}\sum_{j=1}^T \|y_j - y'_j\|^2$. It is assumed that $\mathbf{y}$ takes its values in a compact subset of $\mathbb{R}^q$, i.e., there exists $K_y > 0$ such that $\|y_j\| \leq K_y$.

Theorem 5.8. *Assume that for all $\theta \in \Theta$, (A_1) and (A_2) are verified. Assume, in addition, that there exists a constant $B > 0$ such that for any $1 \leq \ell \leq p$, $\theta \in \Theta$, $\|\xi_{\alpha_{\ell,\theta}}\|_{\mathscr{H}} \leq B$. Then with probability at least $1-\delta$,*

$$\mathscr{R}(\widehat{\theta}_n) \leq \widehat{\mathscr{R}}_n(\widehat{\theta}_n) + \frac{c_3}{T} + \frac{8pc_4B(1-L)^{-1}}{\sqrt{n}} + \sqrt{\frac{2c_5\log(1/\delta)}{n}}, \tag{5.14}$$

where $c_3 = \sup_\theta\big(c_{1,\theta} + \|\psi\|_{\text{op}}\|f_\theta\|_\infty\big) + 2\sqrt{p}B(1-L)^{-1} + 2K_y$, $c_4 = B(1-L)^{-1} + K_y$, and $c_5 = 4pB(1-L)^{-1}c_4 + K_y^2$.

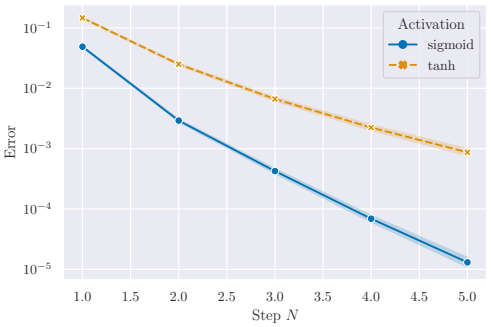
5.3.2 Regularization and stability

In addition to providing a sound theoretical framework, framing deep learning in a RKHS provides a natural norm, which can be used for regularization, as shown for example in the context of convolutional neural networks by Bietti et al. (2019). This regularization ensures stability of predictions, which is crucial in particular in a small sample regime or in the presence of adversarial examples (Gao et al., 2018; Ko et al., 2019). In our binary classification setting, for any inputs $\mathbf{x}, \mathbf{x}' \in (\mathbb{R}^d)^T$, by the Cauchy-Schwartz inequality, we have

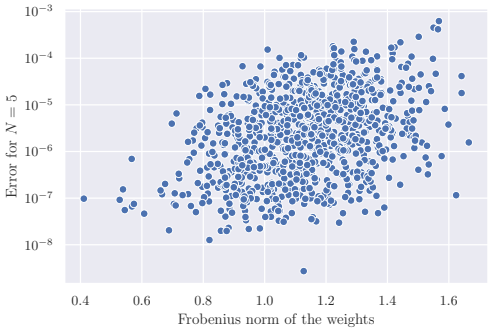
$$\|z_T - z'_T\| \leq 2\|\psi\|_{\text{op}}\|\frac{c_1}{T} + \|\xi_{\alpha_\theta}(\bar{X}) - \xi_{\alpha_\theta}(\bar{X}')\| \leq 2\|\psi\|_{\text{op}}\|\frac{c_1}{T} + \|\xi_{\alpha_\theta}\|_{\mathscr{H}}\|S(\bar{X}) - S(\bar{X}')\|_{\mathscr{T}}.$$

If $\mathbf{x}$ and $\mathbf{x}'$ are close, so are their associated continuous processes X and X' (which can be approximated for example by taking a piecewise linear interpolation), and so are their signatures. The term $\|S(\bar{X}) - S(\bar{X}')\|_{\mathscr{T}}$ is therefore small (Friz and Victoir, 2010, Proposition 7.66). Therefore,

when T is large, we see that the magnitude of $\|\xi_{\alpha_\theta}\|_{\mathscr{H}}$ determines how close the predictions are. A natural training strategy to ensure stable predictions, for the types of networks covered in the present article, is then to penalize the problem by minimizing the loss $\widehat{\mathscr{R}}_n(\theta) + \lambda\|\xi_{\alpha_\theta}\|^2_{\mathscr{H}}$. From a computational point of view, it is possible to compute the norm in $\mathscr{H}$, up to a truncation at N of the Taylor expansion, which we know by Proposition 5.4 to be reasonable. It remains that computing this norm is a non-trivial task, and implementing smart surrogates is an interesting problem for the future.



(a) Error on a logarithmic scale as a function of N



(b) Error as a function of the norm of the weights

Figure 5.1 – Approximation of the RNN ODE by the step-N Taylor expansion

5.4 Numerical illustrations

This section is here for illustration purposes. Our objective is not to achieve competitive performance, but rather to illustrate the theoretical results. We refer to Appendix D.4 for implementation details.

Convergence of the Taylor expansion towards the solution of the ODE. We illustrate Proposition 5.4 on a toy example. The process X is a 2-dimensional spiral, and we take feedforward RNN with 2 hidden units. Repeating this procedure with 10^3 uniform random weight initializations, we observe in Figure 5.1a that the signature approximation converges exponentially fast in N. As seen in Figure 5.1b, the rate of convergence depends in particular on the norm of the weight matrices, as predicted by Proposition 5.5. However, condition (5.10) seems to be over-restrictive, since convergence happens even for weights with norm larger than the bound (we have $1/(8a^2d) \simeq 0.01$ here).

Adversarial robustness. We illustrate the penalization proposed in Section 5.3.2 on a toy task that consists in classifying the rotation direction of 2-dimensional spirals. We take a feedforward RNN with 32 hidden units and hyperbolic tangent activation. It is trained on 50 examples, with and without penalization, for 200 epochs. Once trained, the RNN is tested on adversarial examples, generated with the projected gradient descent algorithm with Frobenius norm (Madry et al., 2018), which modifies test examples to maximize the error while staying in a ball of radius ε. We observe in Figure 5.2 that adding the penalization seems to make the network more stable.

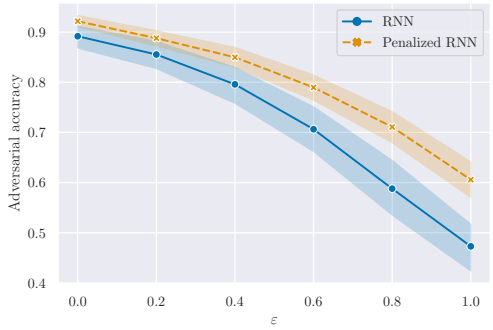


Figure 5.2 – Adversarial accuracy as a function of the adversarial perturbation ε

Comparison of the trained networks. The evolution of the Frobenius norm of the weights $\|W\|_F$ and the RKHS norm $\|\xi_{\alpha_\theta}\|_{\mathscr{H}}$ during training is shown in Figure 5.3. This points out that the penalization, which forces the RNN to keep a small norm in $\mathscr{H}$, leads indeed to learning different weights than the non-penalized RNN. The results also suggest that the Frobenius and RKHS norms are decoupled, since both networks have Frobenius norms of similar magnitude but very different RKHS norms. The figures show one random run, but we observe similar qualitative behavior on others.

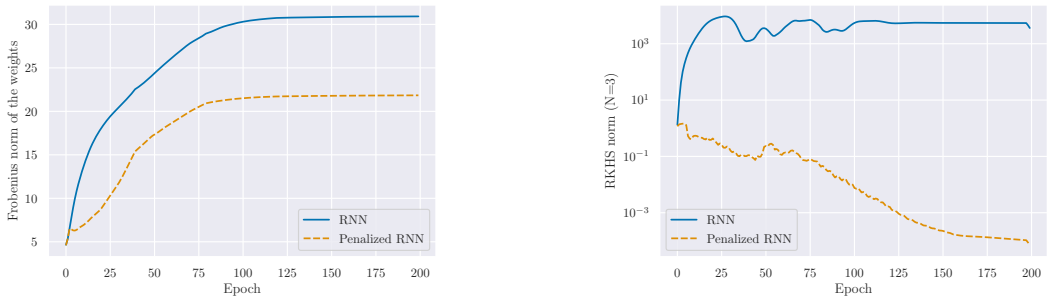


Figure 5.3 – Evolution of the Frobenius norm of the weights and of the RKHS norm during training

5.5 Conclusion

By bringing together the theory of neural ODE, the signature transform, and kernel methods, we have shown that a recurrent network can be framed in the continuous-time limit as a linear function in a well-chosen RKHS. In addition to giving theoretical insights on the function learned by the network and providing generalization guarantees, this framing suggests regularization strategies to obtain more robust RNN. We have only scratched the surface of the potentialities of leveraging this theory to practical applications, which is a subject of its own and will be tackled in future work.

Bibliography

Akpinar, N.-J., Kratzwald, B., and Feuerriegel, S. (2019). Sample complexity bounds for recurrent neural networks with application to combinatorial graph problems. *arXiv:1901.10289*.

Bartlett, P. L., and Mendelson, S. (2002). Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, *3*, 463–482.

Belkin, M., Ma, S., and Mandal, S. (2018). To understand deep learning we need to understand kernel learning. In J. Dy and A. Krause (Eds.), *Proceedings of the 35th international conference on machine learning* (pp. 541–549). PMLR.

Bietti, A., and Mairal, J. (2017). Invariance and stability of deep convolutional representations. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 6210–6220). Curran Associates, Inc.

Bietti, A., and Mairal, J. (2019). Group invariance, stability to deformations, and complexity of deep convolutional representations. *Journal of Machine Learning Research*, *20*, 1–49.

Bietti, A., Mialon, G., Chen, D., and Mairal, J. (2019). A kernel perspective for regularizing deep neural networks. In K. Chaudhuri and R. Salakhutdinov (Eds.), *Proceedings of the 36th international conference on machine learning* (pp. 664–674).

Chang, B., Chen, M., Haber, E., and Chi, E. H. (2019). AntisymmetricRNN: A dynamical system view on recurrent neural networks. *International Conference on Learning Representations*.

Chen, K.-T. (1958). Integration of paths—a faithful representation of paths by non-commutative formal power series. *Transactions of the American Mathematical Society*, *89*, 395–407.

Chen, M., Li, X., and Zhao, T. (2020). On generalization bounds of a family of recurrent neural networks. In S. Chiappa and R. Calandra (Eds.), *Proceedings of the twenty third international conference on artificial intelligence and statistics* (pp. 1233–1243).

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. (2018). Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 6572–6583). Curran Associates, Inc.

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

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 1724–1734.

Cho, Y., and Saul, L. (2009). Kernel methods for deep learning. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta (Eds.), *Advances in neural information processing systems* (pp. 342–350). Curran Associates, Inc.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, *12*, 2493–2537.

De Brouwer, E., Simm, J., Arany, A., and Moreau, Y. (2019). GRU-ODE-Bayes: Continuous modeling of sporadically-observed time series. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 7379–7390). Curran Associates, Inc.

Erichson, N. B., Azencot, O., Queiruga, A., Hodgkinson, L., and Mahoney, M. W. (2021). Lipschitz recurrent neural networks. *International Conference on Learning Representations*.

Fermanian, A. (2021). Embedding and learning with signatures. *Computational Statistics & Data Analysis*, *157*, 107148.

Friz, P., and Victoir, N. (2008). Euler estimates for rough differential equations. *Journal of Differential Equations*, *244*, 388–412.

Friz, P. K., and Victoir, N. B. (2010). *Multidimensional stochastic processes as rough paths: theory and applications* (Vol. 120). Cambridge University Press.

Gao, J., Lanchantin, J., Soffa, M. L., and Qi, Y. (2018). Black-box generation of adversarial text sequences to evade deep learning classifiers. *2018 IEEE Security and Privacy Workshops*, 50–56.

Graves, A., Mohamed, A.-r., and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 6645–6649.

Herrera, C., Krach, F., and Teichmann, J. (2021). Neural jump ordinary differential equations: consistent continuous-time prediction and filtering. *International Conference on Learning Representations*.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. *IEEE Signal Processing Magazine*, *29*, 82–97.

Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*, 1735–1780.

Jacot, A., Gabriel, F., and Hongler, C. (2018). Neural tangent kernel: convergence and generalization in neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 8580–8589). Curran Associates, Inc.

Kidger, P., Bonnier, P., Perez Arribas, I., Salvi, C., and Lyons, T. (2019). Deep signature transforms. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 3099–3109). Curran Associates, Inc.

Kidger, P., and Lyons, T. (2020). Signatory: differentiable computations of the signature and logsignature transforms, on both CPU and GPU. *arXiv:2001.00706*. https://github.com/patrick-kidger/signatory

Kidger, P., Morrill, J., Foster, J., and Lyons, T. (2020a). Neural controlled differential equations for irregular time series. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), *Advances in neural information processing systems* (pp. 6696–6707). Curran Associates, Inc.

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

Ko, C.-Y., Lyu, Z., Weng, L., Daniel, L., Wong, N., and Lin, D. (2019). POPQORN: Quantifying robustness of recurrent neural networks. In K. Chaudhuri and R. Salakhutdinov (Eds.), *Proceedings of the 36th international conference on machine learning* (pp. 3468–3477).

Levin, D., Lyons, T., and Ni, H. (2013). Learning from the past, predicting the statistics for the future, learning an evolving system. *arXiv:1309.0260*.

Liao, S., Lyons, T., Yang, W., and Ni, H. (2019). Learning stochastic differential equations using RNN with log signature features. *arXiv:1908.08286*.

Lim, S. H. (2021). Understanding recurrent neural networks using nonequilibrium response theory. *Journal of Machine Learning Research*, *22*, 1–48.

Lyons, T., Caruana, M., and Lévy, T. (2007). *Differential equations driven by rough paths* (Vol. 1908). Springer.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. *International Conference on Learning Representations*.

Mikolov, T., Karafiát, M., Burget, L., Černocký, J., and Khudanpur, S. (2010). Recurrent neural network based language model. *Proceedings of the 11th Annual Conference of the International Speech Communication Association*, *2*, 1045–1048.

Morrill, J., Salvi, C., Kidger, P., Foster, J., and Lyons, T. (2020b). Neural rough differential equations for long time series. *arXiv:2009.08295*.

Morrill, J. H., Kormilitzin, A., Nevado-Holgado, A. J., Swaminathan, S., Howison, S. D., and Lyons, T. J. (2020c). Utilization of the signature method to identify the early onset of sepsis from multivariate physiological time series in critical care monitoring. *Critical Care Medicine*, *48*, e976–e981.

Perez Arribas, I. (2018). Derivatives pricing using signature payoffs. *arXiv:1809.09466*.

Reizenstein, J., and Graham, B. (2020). Algorithm 1004: the iisignature library: efficient calculation of iterated-integral signatures and log signatures. *ACM Transactions on Mathematical Software*.

Rubanova, Y., Chen, R. T. Q., and Duvenaud, D. K. (2019). Latent ordinary differential equations for irregularly-sampled time series. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 5320–5330). Curran Associates, Inc.

Salvi, C., Cass, T., Foster, J., Lyons, T., and Yang, W. (2020). The signature kernel is the solution of a goursat pde. *arXiv:2006.14794*.

Schölkopf, B., and Smola, A. J. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Toth, C., and Oberhauser, H. (2020). Bayesian learning from sequential data using Gaussian processes with signature covariances. In H. Daumé III and A. Singh (Eds.), *Proceedings of the 37th international conference on machine learning* (pp. 9548–9560). PMLR.

Tu, Z., He, F., and Tao, D. (2019). Understanding generalization in recurrent neural networks. *International Conference on Learning Representations*.

Wang, B., Liakata, M., Ni, H., Lyons, T., Nevado-Holgado, A. J., and Saunders, K. (2019). A path signature approach for speech emotion recognition. *Interspeech 2019*, 1661–1665.

Yang, W., Jin, L., and Liu, M. (2016a). DeepWriterID: An end-to-end online text-independent writer identification system. *IEEE Intelligent Systems*, *31*, 45–53.

Yang, W., Lyons, T., Ni, H., Schmid, C., Jin, L., and Chang, J. (2017). Developing the path signature methodology and its application to landmark-based human action recognition. *arXiv:1707.03993*.

Yue, B., Fu, J., and Liang, J. (2018). Residual recurrent neural networks for learning sequential representations. *Information*, *9*, 56.

Zhang, J., Lei, Q., and Dhillon, I. (2018). Stabilizing gradients for deep neural networks via efficient SVD parameterization. In J. Dy and A. Krause (Eds.), *Proceedings of the 35th international conference on machine learning* (pp. 5806–5814). PMLR.

Chapter 6

The insertion algorithm for signature inversion

The signature is a representation of a path as an infinite sequence of its iterated integrals. Under some assumptions, the signature characterizes the path, up to translation and reparametrization. Therefore, a crucial question of interest is the development of efficient algorithms to invert the signature, that is, reconstruct the path from the information of its (truncated) signature. In this article, we study the insertion algorithm, originally introduced by Chang and Lyons (2019), from both a theoretical and practical point of view. After a description of our version of this algorithm, we give its rate of convergence for piecewise linear paths. This is accompanied by an implementation of this algorithm in Pytorch. This algorithm is parallelized, meaning that it is very efficient at inverting a batch of signatures simultaneously. Its performance is illustrated with both real-world and simulated examples.

Contents

6.1 Introduction

To any multivariate path, that is, a smooth function $X : [0, 1] \to \mathbb{R}^d$, $d \geq 1$, one can associate its signature, denoted by $S(X)$, which is a representation of X as an infinite sequence of tensors of iterated integrals. The signature was first introduced by Chen (1957), then was at the heart of rough path theory (Lyons et al., 2007; Friz and Victoir, 2010), and is now extensively used in machine learning (Levin et al., 2013; Chevyrev and Kormilitzin, 2016a; Király and Oberhauser,

2019; Kidger et al., 2019; Fermanian, 2021; Liao et al., 2019; Toth and Oberhauser, 2020). The combination of the signature transform and machine learning algorithms has been sucessfully applied in various domaines such as character recognition (Yang et al., 2016a; Wilson-Nunn et al., 2018), finance (Lyons et al., 2014; Perez Arribas, 2018), human action recognition (Yang et al., 2017; Li et al., 2017), medicine (Morrill et al., 2019; Morrill et al., 2020c), and emotion recognition (Wang et al., 2019).

The signature is a faithful representation of a path as a geometric object: up to translations, reparametrizations, and "tree-like" pieces—sections of a path where it retraces itself backward, see Hambly and Lyons (2010)—the signature characterizes the path. A natural question is therefore to design fast algorithms that can invert the signature, that is, reconstruct X from the information in $S(X)$. This question has been an active area of research, with several results in the last years. To cite only a few, Lyons and Xu (2017) provide a procedure for piecewise linear paths, Lyons and Xu (2018) for $\mathscr{C}_1$ paths and Chang et al. (2017) for monotone paths. To our knowledge, the most recent and fastest algorithm is the insertion algorithm, first proposed by Chang and Lyons (2019), which is our object of study.

The insertion algorithm focuses on piecewise linear paths: given the signature of a piecewise linear path truncated at order n, this algorithm reconstructs a piecewise linear path of n pieces. In this article, we present a new version of this insertion algorithm, give its rate of convergence, and provide its implementation in the `PyTorch` (Paszke et al., 2019) framework. This implementation is now part of the package `Signatory` (Kidger and Lyons, 2020). We also perform a study of the computational cost of the algorithm. Our main goal is to make this algorithm available to a large audience, in particular to the machine learning community. Having a fast inversion algorithm with theoretical guarantees opens several new applications to signatures: sequential data generation with Generative Adversarial Networks (GANs), time series smoothing, or compression, to name only a few.

The article is organized as follows.

(*i*) Section 6.2 gives some preliminaries on signatures and tensors.

(*ii*) Section 6.3 describes the insertion algorithm and gives the main theorems.

(*iii*) Section 6.4 gives some examples of signature inversion and a study of running times.

The proofs are given in Appendix E.

6.2 Preliminaries

We introduce in this section the mathematical setting together with some notations used throughout the paper. In all the following, we assume that $\mathbb{R}^d$ is equipped with the Euclidean norm, denoted by $\|\cdot\|$. We refer the reader to Friz and Victoir (2010), Chapter 1, and Lyons et al. (2007) for further details on signatures and tensors.

6.2.1 Path of bounded variation

Definition 6.1 (Path of bounded variation)**.** *Let $X : [0,1] \to \mathbb{R}^d$ be a continuous path. For any $[u,v] \subset [0,1]$, the total variation of X on $[u,v]$ is defined by*

$$\|X\|_{TV;[u,v]} = \sup_{(u_1,\dots,u_n)\in D_{u,v}} \sum_{i=1}^{n} \|X_{u_i} - X_{u_{i-1}}\|,$$

where $D_{u,v}$ denotes the set of all finite partitions of $[u,v]$, that is,

$$D_{u,v} = \big\{(u_0,\dots,u_n)|\, n \geq 0,\, u = u_0 \leq u_1 \leq \cdots \leq u_{n-1} \leq u_n = v\big\}.$$

The path X is said to be of bounded variation on $[u,v]$ if its total variation is finite. We denote by $BV(\mathbb{R}^d)$ the set of continuous paths of bounded variation on $[0,1]$ with values in $\mathbb{R}^d$.

When $[u,v] = [0,1]$, we often write $\|X\|_{TV}$ instead of $\|X\|_{TV;[0,1]}$.

Example 6.1. *Let $X : [0,1] \to \mathbb{R}^d$ be a continuous piecewise linear path, and $0 = t_0 < t_1 < \cdots < t_{M-1} < t_M = 1$ denote the minimal partition such that X is linear on each $[t_{i-1}, t_i]$: there exists $\alpha_1,\dots,\alpha_M, \beta_1,\dots,\beta_M \in \mathbb{R}^d$ such that*

$$X_t = \alpha_i + \beta_i t, \qquad \text{for } t \in [t_{i-1}, t_i], \quad i \in \{1,\dots,M\}.$$

Then,

$$\|X\|_{TV} = \sum_{i=1}^{M} \|\beta_i\|(t_{i-1} - t_i).$$

6.2.2 Tensor space

Let E and F be two real vector spaces, we denote by $E \otimes F$ their tensor product. It is also a vector space, and, if $(e_i)_{i\in I}$, $(f_j)_{j\in J}$, are basis of E and F respectively, then $(e_i \otimes f_j)_{i\in I, j\in J}$ is a basis of $E \otimes F$. In the finite-dimensional case, if $\dim(E) = d$, $\dim(F) = e$, then $\dim(E \otimes F) = de$. By convention, $E^{\otimes 0} = \mathbb{R}$. We refer the reader to Purbhoo (2012) for more details on tensor products.

The nth tensor power of a vector space E is defined as the order n tensor product of E with itself:

$$E^{\otimes n} = \overbrace{E \otimes \cdots \otimes E}^{n}.$$

When E is of finite dimension, if $(e_1,\dots,e_d)$ is a basis of E, any element a of $E^{\otimes n}$ can be written as

$$a = \sum_{I=(i_1,\dots,i_n)\in\{1,\dots,d\}^n} a_I e_{i_1} \otimes \cdots \otimes e_{i_n}, \qquad a_I \in \mathbb{R}.$$

The space $E^{\otimes n}$ is then of dimension d^n, which means that we can identify $E^{\otimes n}$ with $\mathbb{R}^{d^n}$. In particular, $E^{\otimes 2}$ can be identified with the space of $d \times d$ matrices.

We now restrict to the Euclidean case $E = \mathbb{R}^d$, where $\mathbb{R}^d$ is endowed with its canonical basis, denoted throughout the article by $(e_1,\dots,e_d)$. We wish to endow the tensor spaces $(\mathbb{R}^d)^{\otimes n}$, $n \geq 0$, with norms inherited from the Euclidean norm on $\mathbb{R}^d$. These norms should behave "well" with the tensor product, which is summarized by the notion of admissible norms. Before giving their definition, we introduce the notion of permutations on $(\mathbb{R}^d)^{\otimes n}$.

Definition 6.2. *Let σ be a permutation of $\{1,\dots,n\}$, that is, a bijective function $\sigma : \{1,\dots,n\} \to \{1,\dots,n\}$. Then, for any $a \in (\mathbb{R}^d)^{\otimes n}$,*

$$a = \sum_{(i_1,\dots,i_n)\in\{1,\dots,d\}^n} a_{(i_1,\dots,i_n)} e_{i_1} \otimes \cdots \otimes e_{i_n},$$

$\sigma(a)$ is defined by

$$\sigma(a) = \sum_{(i_1,\dots,i_n)\in\{1,\dots,d\}^n} a_{(i_1,\dots,i_n)} e_{i_{\sigma(1)}} \otimes \cdots \otimes e_{i_{\sigma(n)}}.$$

Example 6.2. *Let $n = 2$ and $\sigma : \{1,2\} \to \{1,2\}$ be the function that switches 1 and 2: $\sigma(1) = 2$ and $\sigma(2) = 1$. Then, any $a \in (\mathbb{R}^2)^{\otimes 2}$ may be written as*

$$a = a_{(1,1)}e_1 \otimes e_1 + a_{(1,2)}e_1 \otimes e_2 + a_{(2,1)}e_2 \otimes e_1 + a_{(2,2)}e_2 \otimes e_2,$$

and

$$\sigma(a) = a_{(1,1)}e_2 \otimes e_2 + a_{(1,2)}e_2 \otimes e_1 + a_{(2,1)}e_1 \otimes e_2 + a_{(2,2)}e_1 \otimes e_1.$$

Definition 6.3 (Admissible norms)**.** *Assume that for each n, $(\mathbb{R}^d)^{\otimes n}$ is endowed with a norm. Then, one says that these norms are admissible if*

(i) For any $n \geq 1$, for any permutation σ of elements of $a \in (\mathbb{R}^d)^{\otimes n}$, then

$$\|\sigma(a)\| = \|a\|.$$

(ii) For any $n, m \geq 1$, $a \in (\mathbb{R}^d)^{\otimes n}$, $b \in (\mathbb{R}^d)^{\otimes m}$,

$$\|a \otimes b\| = \|a\|\|b\|.$$

Remark 6.1. *For simplicity, we write "$(\mathbb{R}^d)^{\otimes n}$ is equipped with an admissible tensor norm" instead of "For each n, $(\mathbb{R}^d)^{\otimes n}$ is equipped with a tensor norm, and these norms are admissible". We also refer to "an admissible norm" instead of "a set of admissible tensor norms".*

There exists several admissible norms. The most natural is the Euclidean tensor norm, defined as follows. Let $a, b \in (\mathbb{R}^d)^{\otimes n}$, then

$$\langle a, b\rangle = \sum_{I\in\{1,\dots,d\}^n} a_I b_I, \quad \text{and} \quad \|a\| = \Big(\sum_{I\in\{1,\dots,d\}^n} a_I^2\Big)^{1/2}.$$

It is straightforward to show that this norm is admissible. Another important admissible norm is the projective norm, which is the largest possible admissible norm. Indeed, let $\|\cdot\|$ be any admissible norm and write an element $a \in (\mathbb{R}^d)^{\otimes n}$ as

$$a = \sum_{i=1}^{k} a_{1,i} \otimes \cdots \otimes a_{n,i}, \qquad a_{1,i}, \dots, a_{n,i} \in \mathbb{R}^d. \tag{6.1}$$

Note that such a representation always exist since we can take the $a_{j,i}$ as basis elements. Then, if we want *(ii)* in Definition 6.3 to be satisfied, by the triangle inequality, we must have

$$\|a\| \leq \sum_{i=1}^{k} \|a_{1,i}\| \cdots \|a_{n,i}\|.$$

Since this is true for any representation of the form (6.1), we must therefore have

$$\|a\| \leq \inf\Big\{ \sum_{i=1}^{k} \|a_{1,i}\| \cdots \|a_{n,i}\| \,|\, a = \sum_{i=1}^{k} a_{1,i} \otimes \cdots \otimes a_{n,i},\ k \geq 1 \Big\}.$$

The right hand side is exactly the projective norm, denoted by $\|\cdot\|_\pi$:

$$\|a\|_\pi = \inf\Big\{ \sum_{i=1}^{k} \|a_{1,i}\| \cdots \|a_{n,i}\| \,|\, a = \sum_{i=1}^{k} a_{1,i} \otimes \cdots \otimes a_{n,i},\ k \geq 1 \Big\}. \tag{6.2}$$

It is easily shown that $\|\cdot\|_\pi$ is a norm and is admissible.

From now on, we assume that the tensor powers of $\mathbb{R}^d$ have been equipped with admissible norms. Finally, we define the tensor algebra as the formal sum of tensor powers.

Definition 6.4. *We denote by $T(\mathbb{R}^d)$ the space of formal series of tensors of $\mathbb{R}^d$, i.e.,*

$$T(\mathbb{R}^d) = \big\{(a_0, \dots, a_k, \dots) \,|\, \forall k \geq 0, \quad a_k \in (\mathbb{R}^d)^{\otimes k}\big\},$$

and

$$T^n(\mathbb{R}^d) = \big\{(a_0, \dots, a_n) \,|\, \forall k \in \{0, \dots, n\}, \quad a_k \in (\mathbb{R}^d)^{\otimes k}\big\}$$

the truncated tensor space up to order n.

6.2.3 The signature of a path

We are now in a position to define the signature and provide some elementary examples.

Definition 6.5. *Let $X \in BV(\mathbb{R}^d)$. The signature of X is defined by*

$$S(X) = (1, \mathbf{X}^1, \mathbf{X}^2, \dots, \mathbf{X}^n, \dots) \in T(\mathbb{R}^d),$$

where, for each integer n,

$$\mathbf{X}^n = \int\cdots\int_{0 \leq u_1 \leq \cdots \leq u_n \leq 1} dX_{u_1} \otimes \cdots \otimes dX_{u_n} \in (\mathbb{R}^d)^{\otimes n}.$$

Note that the integrals should be understood as Riemann-Stieljes integrals.

Example 6.3. *If $d = 2$, $X_t = (X_t^1, X_t^2)$, $(\mathbb{R}^2)^{\otimes n}$ can be identified with $\mathbb{R}^{2^n}$ and the signatures of order 1 and 2 are equal to*

$$\mathbf{X}^1 = \int_0^1 dX_t = \begin{pmatrix} \int_0^1 dX_t^1 \\ \int_0^1 dX_t^2 \end{pmatrix}$$

$$\mathbf{X}^2 = \int_0^1 \int_0^t dX_s \otimes dX_t = \begin{pmatrix} \int_0^1 \int_0^t dX_s^1 dX_t^1 & \int_0^1 \int_0^t dX_s^1 dX_t^2 \\ \int_0^1 \int_0^t dX_s^2 dX_t^1 & \int_0^1 \int_0^t dX_s^2 dX_t^2 \end{pmatrix}.$$

We introduce a series of notations.

- For any $n \geq 0$, the truncated signature of order n is denoted by

$$S^n(X) = (1, \mathbf{X}^1, \dots, \mathbf{X}^n).$$

- For any $u < v$, the simplex in $[u, v]^n$ is denoted by

$$\Delta_{n;[u,v]} = \{(u_1, \dots, u_n) \in [u, v]^n | u \leq u_1 \leq \cdots \leq u_n \leq v\}. \tag{6.3}$$

  When $[u, v] = [0, 1]$, we simply write Δ_n.
- For a multi-index $I = (i_1, \dots, i_n) \in \{1, \dots, d\}^n$, the coefficient of $\mathbf{X}^n$ corresponding to this multi-index is

$$S^I(X) = \int_{(u_1,\dots,u_n)\in\Delta_n} dX^{i_1}_{u_1} \dots dX^{i_n}_{u_n},$$

  where $X_t = (X^1_t, \dots, X^d_t)^\top$, $t \in [0, 1]$. We can then rewrite the signature of order n as

$$\mathbf{X}^n = \sum_{I\in\{1,\dots,d\}^n} S^I(X) e_{i_1} \otimes \cdots \otimes e_{i_n}.$$

- We will sometimes consider the signature of a path restricted to a specific interval $[u, v] \subset [0, 1]$; its signature is then denoted by $S_{[u,v]}(X)$ (respectively $\mathbf{X}^n_{[u,v]}$, and so on).

The signature is an element of $T(\mathbb{R}^d)$ and the truncated signature $S^n(X)$ is in $T^n(\mathbb{R}^d)$. We now give two examples of paths for which we can directly compute the signature: linear and univariate paths.

Example 6.4 (Linear path)**.** *Let $X : [0, 1] \to \mathbb{R}^d$ be a linear path, i.e., $X_t = \alpha + \beta t$, $\alpha, \beta \in \mathbb{R}^d$, for any $t \in [0, 1]$. Then, for any $n \geq 0$, $u, v \in [0, 1]$ such that $u < v$,*

$$\begin{aligned}\mathbf{X}^n_{[u,v]} &= \int_{\Delta_{n;[u,v]}} dX_{u_1} \otimes \cdots \otimes dX_{u_n} \\ &= \int_{\Delta_{n;[u,v]}} (\beta du_1) \otimes \cdots \otimes (\beta du_n) = \beta^{\otimes n} \int_{\Delta_{n;[u,v]}} du_1 \dots du_n = \beta^{\otimes n} \frac{(v-u)^n}{n!}.\end{aligned} \tag{6.4}$$

We now recall several properties of signatures that are useful to understand the insertion algorithm. We refer the reader to Lyons et al. (2007), Chapter 2, for their proofs.

Proposition 6.1.

(i) Let $X \in BV(\mathbb{R}^d)$, $u, v, w \in [0, 1]$ such that $u < v < w$. Then

$$\mathbf{X}^n_{[u,w]} = \sum_{k=0}^{n} \mathbf{X}^k_{[u,v]} \otimes \mathbf{X}^{n-k}_{[v,w]}.$$

(ii) Let $X \in BV(\mathbb{R}^d)$, $\psi : [0, 1] \to [0, 1]$ be a reparametrisation (non-decreasing surjection), and let $\widetilde{X}_t = X_{\psi(t)}$ for any $t \in [0, 1]$. Then, $S(\widetilde{X}) = S(X)$.

The first part of Proposition 6.1 is known as Chen's identity. It gives a procedure to compute signatures of piecewise linear paths: given the explicit formula for the signature of a linear path in Example 6.4, Chen's formula gives the signature of a concatenation of two linear sections. Iterating this procedure enables to compute the signature of any piecewise linear path. This is the basis of the available software in Python such as `iisignature` and `Signatory`.

Since we are interested in reconstructing a path from its signature, we need to discuss what it means for two paths to have the same signature. It is clear from their definition that signatures are invariant by translation, and Proposition 6.1, *(ii)*, states that they are invariant by reparametrizations. The right notion to encapsulate how the signatures characterize paths is the notion of tree-like equivalence, introduced by Hambly and Lyons (2010) and defined as follows.

Definition 6.6 (Tree-like equivalence)**.**

- *A path $X \in BV(\mathbb{R}^d)$ is tree-like if there exists a continuous function $h : [0,1] \to [0,+\infty)$ such that $h(0) = h(1) = 0$ and such that for any $s,t \in [0,1]$, $s \leq t$,*

$$\|X_s - X_t\| \leq h(s) + h(t) - 2 \inf_{u \in [s,t]} h(u).$$

- *Let $X, Y \in BV(\mathbb{R}^d)$, we say that X and Y are tree-like equivalent if $X * \overleftarrow{Y}$ is a tree-like path, and denote this relation by $X \sim Y$.*

Informally, a tree-like path is a path that retraces itself backward. We can now state the uniqueness theorem of Hambly and Lyons (2010, Theorem 4).

Theorem 6.2. *For any $X, Y \in BV(\mathbb{R}^d)$, then $S(X) = S(Y)$ if and only if $X \sim Y$. Moreover, for any $X \in BV(\mathbb{R}^d)$, there exists a unique path of minimal length in its equivalence class, denoted by $\overline{X}$ and called the reduced path.*

This theorem can be summarized as follows: the signature loses the information of the initial position of the path, of the time parametrization, and of any tree-like piece. For example, the two paths of Figure 6.1 have the same signature: the orange part is a tree-like piece and is therefore "not seen" by the signature.

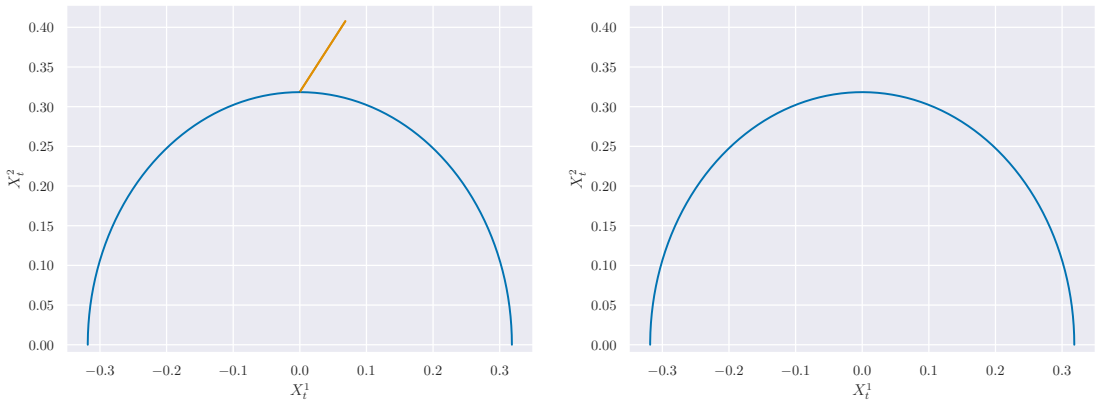


Figure 6.1 – Path with a tree-like piece, plotted in orange (left) and the same path where the tree-like piece has been removed (right). These two paths have the same signature.

Finally, we give below an upper bound on the size of signature coefficients.

Proposition 6.3. *Let $X \in BV(\mathbb{R}^d)$, $u, v \in [0,1]$ such that $u < v$. For any admissible tensor norm and $n \geq 0$,*

$$\|\mathbf{X}^n_{[u,v]}\| \leq \frac{\|X\|^n_{TV;[u,v]}}{n!}.$$

6.3 The insertion algorithm

6.3.1 Theoretical guarantees

We are now in a position to describe the insertion algorithm and our theoretical setting. We assume that we are given a piecewise linear path $X : [0,1] \to \mathbb{R}^d$ as defined in Example 6.1. We

denote by

$$0 = t_0 < t_1 < \cdots < t_{M-1} < t_M = 1$$

the minimal partition such that X is linear on each $[t_{i-1}, t_i]$: there exists $\alpha_1, \dots, \alpha_M, \beta_1, \dots, \beta_M \in \mathbb{R}^d$ such that

$$X_t = \alpha_i + \beta_i t, \qquad \text{for } t \in [t_{i-1}, t_i], \quad i \in \{1, \dots, M\}. \tag{6.5}$$

We assume that X is the reduced path in its equivalence class, for the tree-like equivalence (see Theorem 6.2). This amounts to assuming that the angle between any two consecutive linear segments is not equal to zero. More precisely, for any $i \in \{1, \dots, M-1\}$, we denote by ω_i the angle between the linear segments $[t_{i-1}, t_i]$ and $[t_i, t_{i+1}]$, defined by

$$\omega_i = \operatorname{Arccos}\Big(\frac{\langle \beta_i, \beta_{i+1} \rangle}{\|\beta_i\| \|\beta_{i+1}\|}\Big) \in [0, \pi[,$$

where $\langle \cdot, \cdot, \rangle$ denotes the Euclidean scalar product, and Arccos the arccosine function. Note that $\omega_i \in [0, \pi[$ because we have taken a minimal partition. Then, we make the assumption

$$(A_1) \quad \omega = \min_{i \in \{1, \dots, M\}} \omega_i > 0. \tag{6.6}$$

Assumption (A_1) implies that the signature of X characterizes X up to translation and reparametrization. The invariance by translation amounts to saying that we cannot recover α_0 from $S(X)$. By continuity of X, the α_is for $i \in \{1, \dots, M\}$ are entirely determined by $\alpha_0, \beta_1, \dots, \beta_M$. Therefore, the only information we hope to recover is the information on the slopes β_is, up to reparametrizations of the path. Given this invariance, we can choose a specific parametrization of X. Let ℓ denote the length of X, $\ell = \|X\|_{TV}$, then we define the reparametrization

$$\begin{aligned} \phi : [0,1] &\to [0,1] \\ t &\mapsto \frac{\|X\|_{TV;[0,t]}}{\ell}. \end{aligned}$$

The function ϕ is strictly increasing and continuous piecewise linear. Indeed, if $t \in [t_{i-1}, t_i[$, then

$$\phi(t) = \ell^{-1} \sum_{k=1}^{i-1} \|\beta_k\|(t_k - t_{k-1}) + \ell^{-1} \|\beta_i\|(t - t_{i-1}).$$

The path $\overline{X}_t = X_{\phi^{-1}(t)}$ is therefore a reparametrization of X, and $S(X) = S(\overline{X})$. More precisely, $\overline{X}$ is linear by part on the partition $0 = u_0 < u_1 < \cdots < u_M = 1$, where

$$u_i = \ell^{-1} \sum_{k=1}^{i} \|\beta_k\|(t_k - t_{k-1}).$$

For any $i \in \{1, \dots, M\}$, $u \in [u_{i-1}, u_i[$, we have

$$\phi^{-1}(u) = \frac{\ell}{\|\beta_i\|}(u - u_{i-1}) + t_{i-1},$$

and

$$\overline{X}_u = \alpha_i + \beta_i \phi^{-1}(u) = \alpha_i + \frac{\ell}{\|\beta_i\|} \beta_i (u - u_i) + \beta_i t_{i-1}.$$

The path $\overline{X}$ is therefore piecewise linear and its slopes have a constant norm, equal to ℓ. We see that recovering X up to translations and reparametrizations amounts to recovering the direction of each linear segment up to its norm, that is, ${}^{\beta_1}/_{\|\beta_1\|}, \dots, {}^{\beta_M}/_{\|\beta_M\|}$.

From now on, we therefore assume that X is piecewise linear on a partition $t_0 = 0 < t_1 < \cdots < t_M = 1$ and that $\|\beta_i\| = \ell$ for any $i \in \{1, \dots, M\}$:

$$(A_2) \quad \forall i \in \{1, \dots, M\},\ \|\beta_i\| = \ell.$$

The general idea of the algorithm is the following. Signatures of order n and $n+1$ are linked through the insertion of the derivatives of the path into a tensor product that is then integrated. Therefore, retrieving the derivative of the path boils down to finding a vector in $\mathbb{R}^d$ such that, once inserted into the tensor product of the signature of order n, it will minimize the distance between this modified signature and the signature of order $n+1$. This problem can be written as a linear minimization problem, which can be solved very efficiently. Solving this problem allows to retrieve the derivatives of the path on different intervals and to reconstruct the path by integration.

The first ingredient of the algorithm is the insertion map, which is a linear map defined as a tensor product between a vector $y \in \mathbb{R}^d$ and the signature of X of order n along the dimension p.

Definition 6.7. *Let $n \geq 1$, $p \in \{1, \dots, n+1\}$, and $X \in BV(\mathbb{R}^d)$. The insertion map is denoted by $\mathscr{L}^n_{p,X} : \mathbb{R}^d \to (\mathbb{R}^d)^{\otimes(n+1)}$ and defined, for any $y \in \mathbb{R}^d$, by*

$$\mathscr{L}^n_{p,X}(y) = \int_{(u_1,\dots,u_n)\in\Delta_n} dX_{u_1} \otimes \cdots \otimes dX_{u_{p-1}} \otimes y \otimes dX_{u_p} \otimes \cdots \otimes dX_{u_n} \in (\mathbb{R}^d)^{\otimes n+1}.$$

A first observation is that this map is linear, depends only on the signature of X truncated at order n, and is Lipschitz continuous for any admissible tensor norm (see Definition 6.3).

Proposition 6.4. *Let $X \in BV(\mathbb{R}^d)$. Then, for any $n \geq 1, p \in \{1, \dots, n+1\}$, $\mathscr{L}^n_{p,X}$ is linear, Lipschitz-continuous, and its Lipschitz constant is $\|\mathbf{X}^n\|$.*

Proof. The linearity is a direct consequence of the linearity of the tensor product. Let $y, z \in \mathbb{R}^d$, then

$$\begin{aligned}
\|\mathscr{L}^n_{p,X}(y) - \mathscr{L}^n_{p,X}(z)\| &= \Big\| \int_{\Delta_n} dX_{u_1} \otimes \cdots \otimes dX_{u_{p-1}} \otimes (y-z) \otimes dX_{u_p} \otimes \cdots \otimes dX_{u_n} \Big\| \\
&= \Big\| \int_{\Delta_n} dX_{u_1} \otimes \cdots \otimes dX_{u_n} \otimes (y-z) \Big\| \qquad \text{(by Definition 6.3, } (i)\text{)} \\
&= \Big\| \int_{\Delta_n} dX_{u_1} \otimes \cdots \otimes dX_{u_n} \Big\| \|y-z\| \qquad \text{(by Definition 6.3, } (ii)\text{)} \\
&= \|\mathbf{X}^n\| \|y-z\|. \qquad (6.7)
\end{aligned}$$

□

The idea of the insertion algorithm is to minimize the following quantity

$$\min_{y\in\mathbb{R}^d} \|\mathscr{L}^n_{p,X}(y) - (n+1)\mathbf{X}^{n+1}\|. \qquad (6.8)$$

If p is well-chosen, that is, ${}^{p}/_{n+1} \in [t_{i-1}, t_i[$ far enough from the limits of the interval (see Figure E.2), then we can show that the minimizer of (6.8) is close to β_i, which is stated in the next

theorem.

Theorem 6.5. *Let X be a piecewise linear path as defined by (6.5), following assumptions (A_1) and (A_2). Let*

$$K(\omega) = \log\Big(\frac{2}{1-\cos(|\omega|/2)}\Big), \quad \text{and} \quad n_1 = \lfloor 4e^{2(M-1)K(\omega)}\rfloor, \tag{6.9}$$

where ω is defined by (6.6). For any $1 \le i \le M$, $n \ge \max(n_1, 2/(t_i - t_{i-1}))$, and $p = \lfloor (3t_i + t_{i-1})(n+1)/4\rfloor$, let

$$y^*_{p,n} \in \operatorname*{arg\,min}_{y\in\mathbb{R}^d} \|\mathscr{L}^n_{p,X}(y) - (n+1)\mathbf{X}^{n+1}\|. \tag{6.10}$$

Then there exists $k_n \in [n - n^{3/4}, n + n^{3/4}]$ such that

$$\|y^*_{p,k_n} - \beta_i\| \le 4\ell e^{(M-1)K(\omega)}\Big(\frac{1}{\sqrt{k_n+1}}\sqrt{\frac{1-(t_i-t_{i-1})}{t_i - t_{i-1}}} + 4\exp\Big(-\frac{k_n}{16}(t_i - t_{i-1})^2\Big)\Big).$$

Given that the intervals $[n - n^{3/4}, n + n^{3/4}]$ go rightward and that their width grows slower than n, we have the immediate corollary

Corollary 6.6. *There exists a strictly increasing subsequence $(k_n)_{n\ge n_1}$ such that y^*_{p,k_n} converges to β_i as n increases.*

The proof of Theorem 6.5 relies on two results which we state below. We postpone their proofs to Appendix E.

Theorem 6.7. *Let X be a piecewise linear path as defined by (6.5), following assumptions (A_1) and (A_2). If $1 \le i \le M$, $n \ge 2/(t_i - t_{i-1})$ and $p = \lfloor (3t_i + t_{i-1})(n+1)/4\rfloor$, then*

$$\|\mathscr{L}^n_{p,X}(\beta_i) - (n+1)\mathbf{X}^{n+1}\| \le \frac{\ell^{n+1}}{n!}\Big(\frac{1}{\sqrt{n+1}}\sqrt{\frac{1-(t_i-t_{i-1})}{t_i - t_{i-1}}} + 4\exp\Big(-\frac{n}{16}(t_i - t_{i-1})^2\Big)\Big).$$

Theorem 6.8. *Let X be a piecewise linear path as defined by (6.5), following assumptions (A_1) and (A_2). For any $n > n_1$ (n_1 being defined by (6.9)), there exists $k_n \in [n - n^{3/4}, n + n^{3/4}]$ such that*

$$\|\mathbf{X}^{k_n}\|_\pi \ge \ell^{k_n}\frac{e^{-(M-1)K(\omega)}}{2k_n!},$$

where $\|\cdot\|_\pi$ is the projective tensor norm defined by (6.2).

With these two results at hand, the proof of Theorem 6.5 is straightforward.

Proof of Theorem 6.5. The identity (6.7) is true for any admissible norm so it is true in particular for the projective norm. It gives, for any $n \ge 0$,

$$\|\mathscr{L}^n_{p,X}(y^*_{p,n}) - \mathscr{L}^n_{p,X}(\beta_i)\|_\pi = \|\mathbf{X}^n\|_\pi \, \|y^*_{p,n} - \beta_i\|.$$

On the one hand, Theorem 6.8 gives the existence of $k_n \in [n - n^{3/4}, n + n^{3/4}]$ such that

$$\|\mathbf{X}^{k_n}\|_\pi \ge \ell^{k_n}\frac{e^{-(M-1)K(\omega)}}{2k_n!}.$$

On the other hand, Theorem 6.7 (which is true for any tensor norm) yields

$$\|\mathscr{L}^{k_n}_{p,X}(\beta_i) - (k_n+1)\ \mathbf{X}^{k_n+1}\|_\pi \leq \frac{\ell^{k_n+1}}{k_n!}\Big(\frac{1}{\sqrt{k_n+1}}\sqrt{\frac{1-(t_i-t_{i-1})}{t_i-t_{i-1}}} + 4\exp\Big(-\frac{k_n}{16}(t_i-t_{i-1})^2\Big)\Big),$$

and, by definition of $y^*_{p,n}$,

$$\|\mathscr{L}^{k_n}_{p,X}(y^*_{p,k_n}) - (k_n+1)\ \mathbf{X}^{k_n+1}\| \leq \|\mathscr{L}^{k_n}_{p,X}(\beta_i) - (k_n+1)\ \mathbf{X}^{k_n+1}\|.$$

Combining these inequalities, we obtain

$$\begin{aligned}
\|y^*_{p,k_n} - \beta_i\| &= \frac{1}{\|\mathbf{X}^{k_n}\|_\pi}\|\mathscr{L}^{k_n}_{p,X}(y^*_{p,k_n}) - \mathscr{L}^{k_n}_{p,X}(\beta_i)\| \\
&\leq \frac{1}{\|\mathbf{X}^{k_n}\|_\pi}\big(\|\mathscr{L}^{k_n}_{p,X}(y^*_{p,k_n}) - (k_n+1)\ \mathbf{X}^{k_n+1}\| + \|(k_n+1)\ \mathbf{X}^{k_n+1} - \mathscr{L}^{k_n}_{p,X}(\beta_i)\|\big) \\
&\leq \frac{2}{\|\mathbf{X}^{k_n}\|_\pi}\|(k_n+1)\ \mathbf{X}^{k_n+1} - \mathscr{L}^{k_n}_{p,X}(\beta_i)\| \\
&\leq 4\ell e^{(M-1)K(\omega)}\Big(\frac{1}{\sqrt{k_n+1}}\sqrt{\frac{1-(t_i-t_{i-1})}{t_i-t_{i-1}}} + 4\exp\Big(-\frac{k_n}{16}(t_i-t_{i-1})^2\Big)\Big).
\end{aligned}$$

□

6.3.2 Algorithm

The last step necessary to have a complete algorithm is to solve (6.10). Let $A_p \in \mathbb{R}^{d^{n+1}\times d}$ denote the matrix representing the linear application $\mathscr{L}^n_{p,X}(\cdot)$ in the canonical basis of $\mathbb{R}^d$, where we identify $(\mathbb{R}^d)^{\otimes n+1}$ with $\mathbb{R}^{d^{n+1}}$. Then, for any $y \in \mathbb{R}^d$, $\mathscr{L}^n_{p,X}(y) = A_p y$. Recall that the singular values of a linear operator A are the square roots of the eigenvalues of the operator $A^\top A$. The following lemma tells us the form of the singular values of $\mathscr{L}^n_{p,X}$, which is a crucial step to find an explicit solution of (6.10).

Lemma 6.9. *For any $p \in \{1,\dots,n+1\}$, all the singular values of A_p are identical and equal to $\|\mathbf{X}^n\|$.*

Proof. Let $e_1,\dots,e_d$ denote the canonical basis of $\mathbb{R}^d$. Then, each column of A_p corresponds to $\mathscr{L}^n_{p,X}(e_j)$. By definition, for any index $(i_1,\dots,i_{n+1}) \in \{1,\dots,d\}^{n+1}$,

$$(\mathscr{L}^n_{p,X}(e_j))_{i_1,\dots,i_{n+1}} = \begin{cases} S^{(i_1,\dots,i_{p-1},i_{p+1},\dots,i_{n+1})} \text{ if } i_p = j, \\ 0 \text{ otherwise.} \end{cases}$$

Therefore, each row of A_p, indexed by a tuple $(i_1,\dots,i_{n+1})$, contains only one non-zero element: the one in the column $j = i_p$. Let $a_{i_1,\dots,i_{n+1},j}$ denote the element in column j corresponding to the row $(i_1,\dots,i_{n+1})$. Then, for any $q,r \in \{1,\dots,d\}$,

$$\begin{aligned}
(A_p^\top A_p)_{q,r} &= \sum_{(i_1,\dots,i_{n+1})\in\{1,\dots,d\}^{n+1}} a_{i_1,\dots,i_{n+1},q}\, a_{i_1,\dots,i_{n+1},r} \\
&= \sum_{i_p=1}^{d} \sum_{(i_1,\dots,i_{p-1},i_{p+1},\dots,i_{n+1})\in\{1,\dots,d\}^n} a_{i_1,\dots,i_{n+1},q}\, a_{i_1,\dots,i_{n+1},r}
\end{aligned}$$

For both $a_{i_1,\dots,i_{n+1},q}$ and $a_{i_1,\dots,i_{n+1},r}$ to be non-zero, q and r must both be equal to i_p. Therefore, $A_p^\top A_p$ is diagonal. Then, the terms corresponding to $i_p = q = r$ are non-zero and equal to

$$(A_p^\top A_p)_{q,q} = \sum_{(i_1,\dots,i_{p-1},i_{p+1},i_{n+1})\in\{1,\dots,d\}^n} (S^{(i_1,\dots,i_{p-1},i_{p+1},\dots,i_{n+1})})^2 = \|\mathbf{X}^n\|^2.$$

Therefore, all diagonal entries of $A_p^\top A_p$ are equal to $\|\mathbf{X}^n\|^2$. By definition, the singular values of A_p are therefore all equal to $\|\mathbf{X}^n\|$. □

It is then immediate to compute the solution to (6.10).

Proposition 6.10. *For any $p \in \{1,\dots,n+1\}$, Problem (6.10) has a unique solution, equal to*

$$y^*_{p,n} = \frac{A_p^\top \mathbf{X}^{n+1}}{\|\mathbf{X}^n\|^2}.$$

Proof. We can rewrite (6.10) as

$$y^*_{p,n} \in \operatorname*{arg\,min}_{y\in\mathbb{R}^d} \|\mathscr{L}^n_{p,X}(y) - (n+1)\mathbf{X}^{n+1}\|^2.$$

This is exactly the same minimization problem than in least-squars regression. It is then well known that

$$y^*_{p,n} = (A_p^\top A_p)^{-1} A_p^\top \mathbf{X}^{n+1}$$

By Lemma 6.9, $A_p^\top A_p = \|\mathbf{X}^n\|^2 I_d$, and

$$y^*_{p,n} = \frac{A_p^\top \mathbf{X}^{n+1}}{\|\mathbf{X}^n\|^2}.$$

□

Algorithme 2 : Insertion algorithm

Data : n: truncation order of the signature; $S^n(X)$: signature truncated at order n; d: dimension of the underlying path; X_0: starting point of the path.

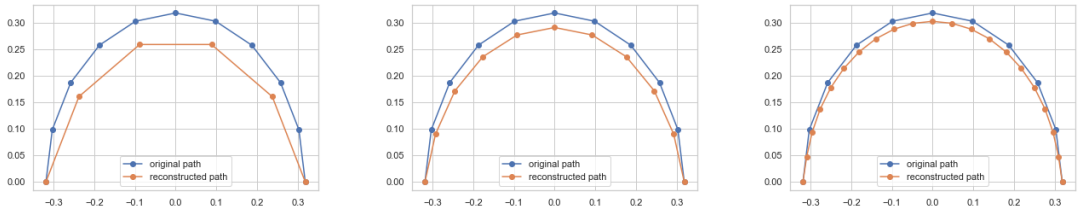
Result : $\widetilde{X} = (X_0, \widetilde{X}_{1/n}, \dots, \widetilde{X}_1) \in \mathbb{R}^{(n+1)\times d}$: array of the $n+1$ positions in $\mathbb{R}^d$ of the reconstructed path.

Extract $\mathbf{X}^n$ and $\mathbf{X}^{n-1}$, the terms of order n and $n-1$ of $S^n(X)$.
$\widetilde{X}_0 = X_0$
for $p \in \{1,\dots,n\}$ **do**
    Compute A_p, the matrix representing the linear map $\mathscr{L}^{n-1}_{p,X}(\cdot)$, function of $\mathbf{X}^{n-1}$.
    $y^*_{p,n} = \frac{A_p^\top \mathbf{X}^n}{\|\mathbf{X}^{n-1}\|^2}$,
    $\widetilde{X}_{p/n} = \widetilde{X}_{p-1/n} + \frac{y^*_{p,n}}{n}$.

Algorithm 2 summarizes the insertion algorithm. Given a signature of order n, we solve (6.10) for any p varying in $\{1,\dots,n\}$. Then, we reconstruct a continuous piecewise linear path having

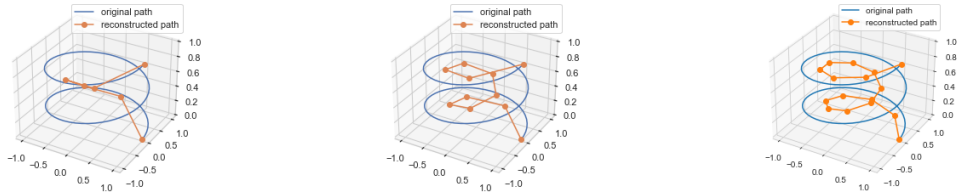
a slope of $y^*_{p,n}$ on $[(p-1)/n, p/n]$. This algorithm is very cheap computationally: inside the loop, the most expensive operation is the matrix multiplication $A_p^\top \mathbf{X}^n$.

It is straightforward to parallelize this algorithm so that it takes as input a batch of signatures, that is, an array $I \in \mathbb{R}^{N \times (d^{n+1}-1)/(d-1)}$, and outputs a tensor of multiple paths $O \in \mathbb{R}^{N \times (n+1) \times d}$. We refer the reader to the code included in the Signatory package (Kidger and Lyons, 2020) for more details. All operations in Algorithm 2 can indeed be performed on tensors of size $N \times \cdots$ instead of matrices.



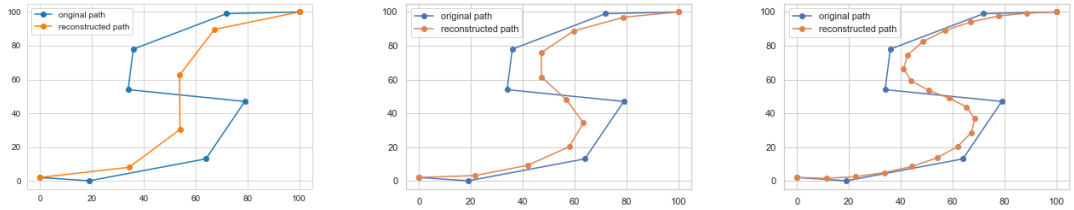
(a) $n = 5$ (b) $n = 10$ (c) $n = 20$

Figure 6.2 – Signature inversion of a half circle



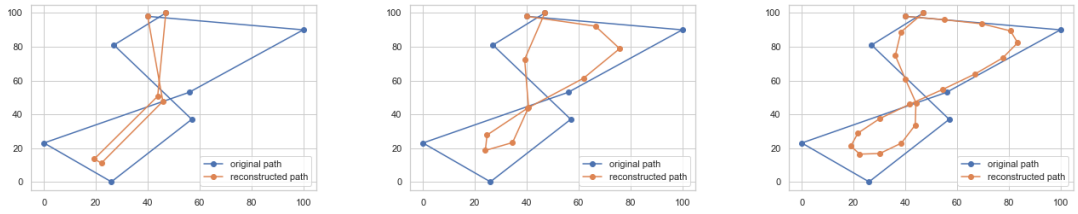
(a) $n = 5$ (b) $n = 10$ (c) $n = 15$

Figure 6.3 – Signature inversion of a spiral in 3d.



(a) $n = 5$ (b) $n = 10$ (c) $n = 20$

Figure 6.4 – Signature inversion of a sample of the class "5" from the Pendigits dataset

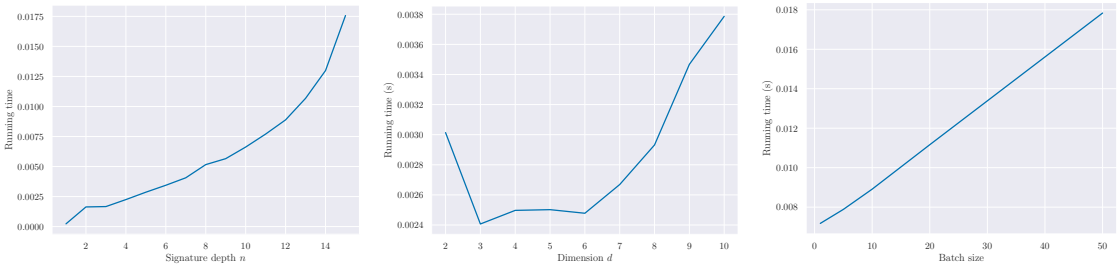


(a) $n = 5$ (b) $n = 10$ (c) $n = 20$

Figure 6.5 – Signature inversion of a sample of the class "8" from the Pendigits dataset

6.4 Experimental results

Examples of inversion To conclude, from a signature truncated at order n, we are able to reconstruct n positions of the path with basic linear algebra tools. We show in Figures 6.2, 6.3, 6.4, and 6.5 some examples of reconstruction for paths in 2 and 3 dimensions. Figures 6.2 and 6.3 are simulations of a half circle and a spiral, while Figures 6.4 and 6.5 are two samples of the Pendigits dataset from the UCI Machine Learning Repository (Dua and Graff, 2017). We can see that in all the figures, for $n = 20$ the path reconstructed is smooth and close to the original path.



(a) Run time when n varies. (b) Run time when d varies (c) Run time when N varies

Figure 6.6 – Running time in seconds to invert the signature of several paths, when we let some hyperparameters vary: the depth of the signature n, the dimension of the path d, and the number of signatures that are inverted.

Running times We present in Figure 6.6 the running time of Algorithm 2 when we let several hyperparameters vary. The paths are randomly generated as piecewise linear paths with $M = 10$ pieces, where the ending point of each linear part is generated uniformly at random in $[0, 1]^d$. The experiments were run on a standard laptop computer.

In Figure 6.6a, we invert one path in dimension $d = 2$ for different values of n. In Figure 6.6b, we set $n = 4$ and let d vary. In Figure 6.6c, we set $n = 10$ and $d = 2$, and let the number of paths N vary. We can see that when n increases, the running time increases exponentially, while the dependence to the dimension d is polynomial, and the dependence to the number of paths N is linear. Overall, we can invert 50 signatures of depth 10 in approximately 0.02s.

Bibliography

Chang, J., Duffield, N., Ni, H., and Xu, W. (2017). Signature inversion for monotone paths. *Electronic Communications in Probability*, *22*, 1–11.

Chang, J., and Lyons, T. (2019). Insertion algorithm for inverting the signature of a path. *arXiv:1907.08423*.

Chen, K.-T. (1957). Integration of paths, geometric invariants and a generalized baker-hausdorff formula. *Annals of Mathematics*, 163–178.

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

Dua, D., and Graff, C. (2017). UCI machine learning repository. http://archive.ics.uci.edu/ml

Fermanian, A. (2021). Embedding and learning with signatures. *Computational Statistics & Data Analysis*, *157*, 107148.

Friz, P. K., and Victoir, N. B. (2010). *Multidimensional stochastic processes as rough paths: theory and applications* (Vol. 120). Cambridge University Press.

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

Kidger, P., Bonnier, P., Perez Arribas, I., Salvi, C., and Lyons, T. (2019). Deep signature transforms. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 3099–3109). Curran Associates, Inc.

Kidger, P., and Lyons, T. (2020). Signatory: differentiable computations of the signature and logsignature transforms, on both CPU and GPU. *arXiv:2001.00706*. https://github.com/patrick-kidger/signatory

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

Levin, D., Lyons, T., and Ni, H. (2013). Learning from the past, predicting the statistics for the future, learning an evolving system. *arXiv:1309.0260*.

Li, C., Zhang, X., and Jin, L. (2017). LPSNet: a novel log path signature feature based hand gesture recognition framework. *2017 IEEE International Conference on Computer Vision Workshop*, 631–639.

Liao, S., Lyons, T., Yang, W., and Ni, H. (2019). Learning stochastic differential equations using RNN with log signature features. *arXiv:1908.08286*.

Lyons, T., Caruana, M., and Lévy, T. (2007). *Differential equations driven by rough paths* (Vol. 1908). Springer.

Lyons, T., Ni, H., and Oberhauser, H. (2014). A feature set for streams and an application to high-frequency financial tick data. *Proceedings of the 2014 International Conference on Big Data Science and Computing*, 5.

Lyons, T., and Xu, W. (2018). Inverting the signature of a path. *Journal of the European Mathematical Society*, *20*, 1655–1687.

Lyons, T. J., and Xu, W. (2017). Hyperbolic development and inversion of signature. *Journal of Functional Analysis*, *272*, 2933–2955.

Morrill, J., Kormilitzin, A., Nevado-Holgado, A., Swaminathan, S., Howison, S., and Lyons, T. (2019). The signature-based model for early detection of sepsis from electronic health records in the intensive care unit. *International Conference in Computing in Cardiology*.

Morrill, J. H., Kormilitzin, A., Nevado-Holgado, A. J., Swaminathan, S., Howison, S. D., and Lyons, T. J. (2020c). Utilization of the signature method to identify the early onset of sepsis from multivariate physiological time series in critical care monitoring. *Critical Care Medicine*, *48*, e976–e981.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 8024–8035). Curran Associates, Inc.

Perez Arribas, I. (2018). Derivatives pricing using signature payoffs. *arXiv:1809.09466*.

Purbhoo, K. (2012). Notes on tensor products and the exterior algebra.

Toth, C., and Oberhauser, H. (2020). Bayesian learning from sequential data using Gaussian processes with signature covariances. In H. Daumé III and A. Singh (Eds.), *Proceedings of the 37th international conference on machine learning* (pp. 9548–9560). PMLR.

Wang, B., Liakata, M., Ni, H., Lyons, T., Nevado-Holgado, A. J., and Saunders, K. (2019). A path signature approach for speech emotion recognition. *Interspeech 2019*, 1661–1665.

Wilson-Nunn, D., Lyons, T., Papavasiliou, A., and Ni, H. (2018). A path signature approach to online arabic handwriting recognition. *Proceedings of the 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)*, 135–139.

Yang, W., Jin, L., and Liu, M. (2016a). DeepWriterID: An end-to-end online text-independent writer identification system. *IEEE Intelligent Systems*, *31*, 45–53.

Yang, W., Lyons, T., Ni, H., Schmid, C., Jin, L., and Chang, J. (2017). Developing the path signature methodology and its application to landmark-based human action recognition. *arXiv:1707.03993*.

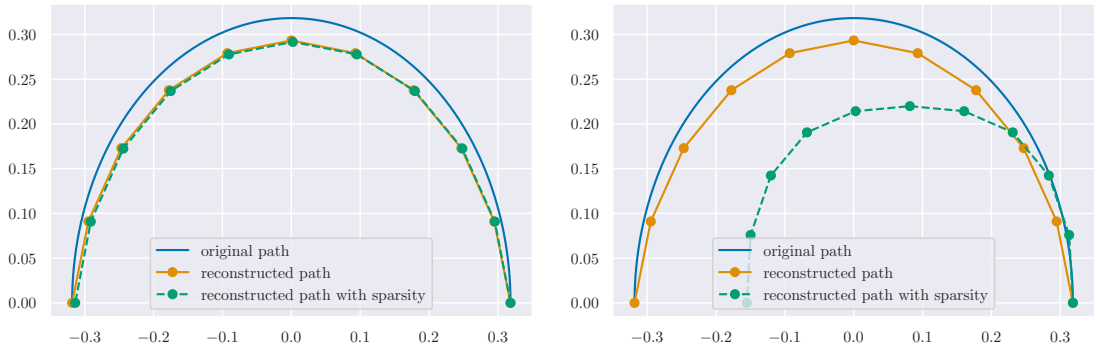
Conclusion

In this thesis, we have presented several contributions in the growing field of learning with signatures. First, we have shown in Chapter 2 that the choice of embedding, that is, the way discretely sampled data is modeled as a continuous path, is a critical factor. Indeed, this choice impacts drastically the performance of the whole method, regardless of the algorithm subsequently used. We pursue this study in Chapter 3 by an extensive study of the different existing variations of signature features. This study enables us to identify a generic algorithm that shows competitive performance for time series classification. We go on in Chapter 4 by studying the estimation problem in a signature linear regression setting. The crucial parameter in this context is the choice of truncation of the signature. We define an estimator of this truncation parameter, give its rate of convergence, and show that the resulting model is competitive with classical functional regression models with basis expansions such as Fourier or splines. In Chapter 5, we take a slightly different perspective and use the signature kernel to study recurrent neural networks (RNN). We show that RNN can be rewritten, in the continuous-time limit, as a kernel method on signatures. This approach leads to the derivation of generalization bounds valid for a large class of RNN and new regularization techniques. We conclude in Chapter 6 by contributions to the problem of inverting the signature with the insertion algorithm. We propose a modification of this algorithm, provide an implementation in a Python package, and show theoretical guarantees.

These contributions are only the first steps in a field where much remains to be discovered, and open up several questions. A first research direction would be to address the problem of the high dimension of truncated signatures—recall that for a path in $\mathbb{R}^d$, the size of the signature truncated at order n is of magnitude d^n. The exponential dependence on the truncation parameter n forces it to remain small in practical applications. It seems reasonable to think that specific treatment of this dimension issue could improve existing algorithms. We can think of different approaches to this dimension issue, detailed below.

On the one hand, logsignatures provide a lower-dimensional representation of signatures but do not inherit all their good properties, in particular the linear approximation property (Theorem 1.13). A study of the statistical properties of logsignatures would be valuable, for example with tools from topological data analysis or algebraic statistics. On the other hand, a natural approach to dealing with high dimension problems is to use sparsity, which can be deployed in different directions. First, a lasso-type model could improve the signature linear model of Chapter 4. The proper choice of an L1 tensor norm in this context is an open question; both an empirical and theoretical analysis would be valuable to make progress on this issue. An extension of the group lasso, exploiting the tensor structure of the signature, seems a reasonable direction.

The question of sparsity may also be addressed outside the supervised learning context. More specifically, it would be interesting to study how one should induce sparsity on signatures so that not too much information is lost. Figure 6.7 shows the result of a naive approach to this problem: we have computed the signature of order 10 of a half-circle, set to zero the smallest coefficients of the signature, and inverted this sparse signature. We can see in Figure



(a) Largest 66% coefficients are kept

(b) Largest 39% coefficients are kept

Figure 6.7 – Inversion of the signature truncated at order $n = 10$ of a half circle, with the complete signature (orange curves) or with a sparse signatures, where the smallest coefficients are set to zero (green curves)

6.7a that when keeping only 66% of the signature coefficients, we get the same reconstruction as with the full set of coefficients. In Figure 6.7b we keep 39% of the coefficients and the reconstruction is much worse. A lot remains to be explored in this direction: how can we exploit the algebraic properties of signatures, for example, the shuffle product identity (Theorem 1.12) in the sparsifying mechanism? Should we favor some orders, for example, the smallest? Can we derive theoretical guarantees on the reconstructed signal, in the same spirit as compressed sensing results in the area of signal processing?

On the other hand, several extensions to the analysis of RNN performed in Chapter 5 are worth mentioning. First, leveraging this theoretical work to practical applications would be a valuable extension. Moreover, the neural ODE point of view on neural networks is not restricted to recurrent neural networks. In particular, using our tools of controlled differential equations and signatures to the analysis of deep residual networks seems particularly relevant. In this case, the weights between each layer are allowed to differ. In the continuous limit, an object of interest is then a path in the space of weights. This may allow us to study empirically the regularity and the geometry of the network weights, but also to analyze generalization properties of deep residual networks—-which achieve state-of-the-art performance in computer vision.

Finally, applications of the insertion algorithm of Chapter 6 have yet to be explored. For example, its use for trend estimation seems very promising: given a potentially high-frequency irregularly sampled multivariate time series, this algorithm is an efficient way to reconstruct its trend based on its signature, with very few assumptions on the structure of the time series. It would also be valuable to refine this algorithm, which suffers from the fact that a signature of order n is needed to reconstruct n points (in other words, d^n coefficients are needed to output nd values). Using sparse signatures as mentioned above would be a potential direction.

Appendix A

Supplementary material of Chapter 1

Contents

A.1 Proof of Proposition 1.2

Let $N > 0$ and consider the projection at order N of $a \otimes \sum_{n\geq 0}(\mathbf{1}-a)^{\otimes n}$. We have

$$\pi_N\Big(a \otimes \sum_{n\geq 0}(\mathbf{1}-a)^{\otimes n}\Big) = \pi_N\Big(a \otimes \sum_{n=0}^{N}(\mathbf{1}-a)^{\otimes n} + a \otimes \sum_{n\geq N+1}(\mathbf{1}-a)^{\otimes n}\Big)$$
$$= \pi_N\Big(a \otimes \sum_{n=0}^{N}(\mathbf{1}-a)^{\otimes n}\Big),$$

since the second term contains only elements of order at least $N+1$. Moreover,

$$a \otimes \sum_{n=0}^{N}(\mathbf{1}-a)^{\otimes n} = \big(\mathbf{1}-(\mathbf{1}-a)\big) \otimes \sum_{n=0}^{N}(\mathbf{1}-a)^{\otimes n} = \sum_{n=0}^{N}(\mathbf{1}-a)^{\otimes n} - \sum_{n=0}^{N}(\mathbf{1}-a)^{\otimes n+1}$$
$$= \mathbf{1} - (\mathbf{1}-a)^{\otimes N+1}.$$

Therefore,

$$\pi_N\big(a \otimes \sum_{n=0}^{N} (\mathbf{1}-a)^{\otimes n}\big) = \mathbf{1}.$$

This is true for any N and similar calculations give the same result for $\sum_{n\geq 0}(\mathbf{1}-a)^{\otimes n} \otimes a$. Therefore,

$$\sum_{n\geq 0}(\mathbf{1}-a)^{\otimes n} \otimes a = a \otimes \sum_{n\geq 0}(\mathbf{1}-a)^{\otimes n} = \mathbf{1}.$$

$\tilde{T}((E))$ is an affine subspace of $T((E))$ so it is a smooth manifold. The operations $\otimes$ and $^{-1}$ being smooth maps (they are polynomials in the coordinates), we conclude that $\tilde{T}((E))$ is a Lie group.

A.2 Proof of Theorem 1.3

Let $Z = X * Y : [s,u] \to \mathbb{R}^d$. The Nth order term of its signature is

$$\begin{aligned}
\mathbf{Z^N} &= \int\cdots\int_{s<u_1<\cdots<u_N<u} dZ_{u_1} \otimes \cdots \otimes dZ_{u_N} \\
&= \sum_{n=0}^{N} \int\cdots\int_{s<u_1<\cdots<u_n<t<u_{n+1}<\cdots<u_N<u} dZ_{u_1} \otimes \cdots \otimes dZ_{u_N} \\
&= \sum_{n=0}^{N} \Big(\int\cdots\int_{s<u_1<\cdots<u_n<t} dX_{u_1} \otimes \cdots \otimes dX_{u_n}\Big) \otimes \Big(\int\cdots\int_{t<u_{n+1}<\cdots<u_N<u} dY_{u_{n+1}} \otimes \cdots \otimes dY_{u_N}\Big) \\
&= \sum_{n=0}^{N} \mathbf{X}^n \otimes \mathbf{Y}^{N-n}.
\end{aligned}$$

Therefore, $S_{[s,u]}(Z) = S_{[s,t]}(X) \otimes S_{[t,u]}(Y)$.

A.3 Proof of Proposition 1.4

The proof is based on Lemma 1.8. Let $Z = X * \overleftarrow{X}$ and V a Banach space. Denote by $\mathscr{L}(E,V)$ the space of linear functions from E to V. Then, for any function $f : V \to \mathscr{L}(E,V)$, if $Y : [0,1] \to V$ is the solution of

$$dY_t = f(Y_t)dX_t, \quad Y_0 = \xi, \quad Y_1 = \eta,$$

then $\overleftarrow{Y}$ is solution of

$$dY_t = f(Y_t)d\overleftarrow{X}_t, \quad Y_0 = \eta, \quad Y_1 = \xi.$$

Therefore, any solution of

$$dY_t = f(Y_t)dZ_t, \quad Y_0 = \xi$$

satisfies $Y_2 = \xi$. Take f equal to the function of Lemma 1.8, then both $\mathbf{1}$ and $S^N(Z)_{[0,t]}$ are solutions. By uniqueness, for any $N \geq 0$, $S^N(Z)_{[0,t]} = \mathbf{1}$ and $S(Z) = S(X) \otimes S(\overleftarrow{X}) = \mathbf{1}$.

A.4 Proof of Theorem 1.6

We refer the reader to Hambly and Lyons (2010) for a proof of the first part of the theorem. The second part is a direct consequence of Chen's identity 1.3 and Proposition 1.4. We know that

$$S(X * \overleftarrow{Y}) = S(X) \otimes S(\overleftarrow{Y}) = S(X) \otimes S(Y)^{-1}.$$

Therefore,

$$S(X) = S(Y) \Leftrightarrow S(X) \otimes S(Y)^{-1} = \mathbf{1} \Leftrightarrow S(X * \overleftarrow{Y}) = \mathbf{1} \Leftrightarrow X * \overleftarrow{Y} \text{ is tree-like } \Leftrightarrow X \sim Y.$$

A.5 Proof of Lemma 1.7

By Theorem 1.6, it is clear that $\sim$ is an equivalence relation. Then, Hambly and Lyons (2010) show that there exists a unique element of minimal length in each equivalence class. It is therefore sufficient to show that if X has a strictly monotone coordinate, then it is this element of minimal length, which is then entirely determined by its signature.

Let $Y \in BV(\mathbb{R}^d)$ be a path of minimal length in the equivalence class of X, we show that Y is equal to X up to a reparametrization. Let

$$Z = X * \overleftarrow{Y} : [0, 2] \to \mathbb{R}^d.$$

By definition Z is tree-like and we denote by h a height function for Z. With the same argument as in Example 1.6, h is strictly increasing on $[0, 1]$. We next show that Y being of minimal length implies that h is strictly decreasing on $[1, 2]$.

By contradiction, if it was not the case, there would exist an interval $[s, t] \subset [1, 2]$ such that $h(s) = h(t)$, $h(u) \leq h(s)$, $\forall u \in [s, t]$, and h has a unique minimum on $[s, t]$. Consider the function $\tilde{h} : u \mapsto h(s) - h(u)$. It satisfies $\tilde{h}(s) = \tilde{h}(t) = 0$, $\tilde{h}(u) \geq 0$ on $[s, t]$ and for any $[u, v] \subset [s, t]$,

$$\|Z_v - Z_u\| \leq \tilde{h}(u) + \tilde{h}(v) - 2 \inf_{w \in [u,v]} \tilde{h}(w).$$

To prove this inequality, let m denote the point in $[s, t]$ where h is minimal, we have to treat the cases $v \leq m$, $u < m < v$, and $m \leq u$ separately. Let us assume that $v \leq m$, then,

$$\inf_{w \in [u,v]} h(w) = h(v), \quad \text{and} \quad \inf_{w \in [u,v]} \tilde{h}(w) = h(s) - h(u),$$

so

$$\begin{aligned} \|Z_v - Z_u\| \leq h(u) + h(v) - 2 \inf_{w \in [u,v]} h(w) &\leq h(u) - h(v) \\ &\leq \big(h(s) - h(u)\big) + \big(h(s) - h(v)\big) - 2\big(h(s) - h(u)\big) \\ &\leq \tilde{h}(u) + \tilde{h}(v) - 2 \inf_{w \in [u,v]} \tilde{h}(w). \end{aligned}$$

The other cases are treated in a similar way. We conclude that $\tilde{h}$ is a height function for $Z|_{[s,t]}$, which implies that $Z|_{[s,t]}$ is tree-like and its signature is $\mathbf{1}$. Therefore, removing this portion of the path would not change the signature. This would yield a path with strictly shorter length in the equivalence class of X, which contradicts the minimality of the length of Y. So h is strictly

decreasing on $[1,2]$. Let

$$h_1 : [0,1] \to [0, h(1)],\ h_1 = h|_{[0,1]}, \quad \text{and} \quad h_2 : [1,2] \to [h(1), 0],\ h_2 = h|_{[1,2]}.$$

We have shown that h_1 and h_2 are continuous isomorphisms. The function $\sigma = h_2^{-1} \circ h_1 : [0,1] \to [1,2]$ is then also a continuous isomorphism. Moreover, we have $\sigma(0) = h_2^{-1}(h(0)) = h_2^{-1}(0) = 2$, and $\sigma(1) = h_2^{-1}(h(1)) = 1$, so σ is strictly decreasing, and it satisfies $h(s) = h(\sigma(s))$ for any $s \in [0,1]$. Then, for any $s \in [0,1]$:

$$\|Z_s - Z_{\sigma(s)}\| \leq h(s) + h(\sigma(s)) - 2 \inf_{t \in [\sigma(s), s]} h(t) = 2h(s) - 2h(s) = 0,$$

which implies that for any $s \in [0,1]$, $Z_s = Z_{\sigma(s)} \Leftrightarrow X_s = Y_{2-\sigma(s)}$. We conclude that Y is equal to X up to a reparametrization.

A.6 Proof of Lemma 1.8

Existence and uniqueness are a consequence of Picard-Lindelöf theorem (Lyons et al., 2007, Theorem 1.3). It only remains to check that the truncated signature is indeed a solution. For any $n \geq 0$, $t \in [0,1]$,

$$\begin{aligned}
\mathbf{X^n}_{[0,t]} &= \int_{0<u_1<\cdots<u_n<t} \cdots \int dX_{u_1} \otimes \cdots \otimes dX_{u_n} \\
&= \int_0^t \Big(\int_{0<u_1<\cdots<u_{n-1}<u_n} \cdots \int dX_{u_1} \otimes \cdots \otimes dX_{u_{n-1}} \Big) \otimes dX_{u_n} = \int_0^t \mathbf{X^{n-1}}_{[0,u]} \otimes dX_u.
\end{aligned}$$

Thus,

$$S^N(X)_{[0,t]} = (1, \mathbf{X^1}, \ldots, \mathbf{X^N}) = \mathbf{1} + \int_0^t \pi_N(S^N(X)_{[0,u]} \otimes dX_u).$$

A.7 Proof of Proposition 1.10

By the triangle inequality,

$$\|\mathbf{X}^n\|_{(\mathbb{R}^d)^{\otimes n}} = \Big\| \int_{\Delta_n} X_{u_1} \otimes \cdots \otimes X_{u_n} \Big\|_{(\mathbb{R}^d)^{\otimes n}} \leq \int_{\Delta_n} \|dX_{u_1} \otimes \cdots \otimes dX_{u_n}\|_{(\mathbb{R}^d)^{\otimes n}}$$

Moreover, classical properties of tensor norms (Ryan, 2002) yield

$$\|dX_{u_1} \otimes \cdots \otimes dX_{u_n}\|_{(\mathbb{R}^d)^{\otimes n}} = \|dX_{u_1}\| \ldots \|dX_{u_n}\|.$$

Therefore, by partitioning $[0,1]^n$ by the $n!$ possible reorderings of the variables $u_1, \ldots, u_n$, we have

$$\int_{\Delta_n} \|dX_{u_1} \otimes \cdots \otimes dX_{u_n}\|_{(\mathbb{R}^d)^{\otimes n}} \leq \int_{\Delta_n} \|dX_{u_1}\| \ldots \|dX_{u_n}\| = \frac{1}{n!} \int_{u_1,\ldots,u_n \in [0,1]} \cdots \int \|dX_{u_1}\| \ldots \|dX_{u_n}\|.$$

This gives

$$\|\mathbf{X}^n\|_{(\mathbb{R}^d)^{\otimes n}} \leq \frac{1}{n!} \int\cdots\int_{u_1,\cdots,u_n\in[0,1]} \|dX_{u_1}\|\cdots\|dX_{u_n}\| = \frac{1}{n!}\Big(\int_{u\in[0,1]} \|dX_u\|\Big)^n = \frac{1}{n!}\|X\|^n_{1\text{-var}}.$$

Finally, we have

$$\begin{aligned}\|S(X)\|_{T(\mathbb{R}^d)} = \sqrt{1+\sum_{n\geq 0}\|\mathbf{X}^n\|^2_{(\mathbb{R}^d)^{\otimes n}}} &\leq 1+\sum_{n\geq 0}\|\mathbf{X}^n\|_{(\mathbb{R}^d)^{\otimes n}} \\ &\leq 1+\sum_{n\geq 0}\frac{1}{n!}\|X\|^n_{1\text{-var}} = \exp(\|X\|_{1\text{-var}}).\end{aligned}$$

A.8 Proof of Theorem 1.12

The result comes by partitioning the integration domain. Indeed, we have

$$\begin{aligned}S^I(X)S^J(X) &= \int\cdots\int_{0<u_1<\cdots<u_n<1} dX^{i_1}_{u_1}\dots dX^{i_n}_{u_n} \int\cdots\int_{0<t_1<\cdots<t_m<1} dX^{j_1}_{t_1}\dots dX^{j_m}_{u_m} \\ &= \sum_{\sigma\in\text{Shuffles}(n,m)} \int\cdots\int_{0<v_1<\cdots<v_{n+m}<1} dX^{r_{\sigma(1)}}_{v_1}\dots dX^{r_{\sigma(n+m)}}_{v_{n+m}} \\ &= \sum_{K\in I\sqcup J} S^K(X),\end{aligned}$$

with $(r_1,\dots,r_{n+m}) = (i_1,\dots,i_n,j_1,\dots,j_m)$.

A.9 Proof of Theorem 1.13

This is a consequence of Stone-Weierstrass theorem. Indeed, let us consider the space

$$\mathscr{A} = \text{Span}\big\{f_I : X\mapsto \langle e_I, S(X)\rangle_{T(\mathbb{R}^d)} \,|\, n\geq 0, I\subset\{1,\dots,d\}^n\big\}.$$

$\mathscr{A}$ is a linear subspace of the set of continuous functions from D to $\mathbb{R}$, denoted by $\mathscr{C}(D,\mathbb{R})$. Endowed with the shuffle product, Theorem 1.12 ensures that it is also a sub-algebra. To apply Stone-Weierstrass theorem we must check that it contains a non-zero constant function and that it separates points. The first condition is met because the first term of the signature is one: $\langle e_0, S(X)\rangle_{T(\mathbb{R}^d)} = 1$.

The second condition is met by Theorem 1.6. Let $X, Y\in D, X\neq Y$. By assumption, X and Y are not tree-like equivalent, and therefore $S(X)\neq S(Y)$, which means that at least one of their coordinates differ. By taking the corresponding basis there exists a function f in $\mathscr{A}$ such that $f(X)\neq f(Y)$. By Stone-Weitherstrass theorem, we conclude that $\mathscr{A}$ is dense in $\mathscr{C}(D,\mathbb{R})$, which proves the theorem.

Bibliography

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

Lyons, T., Caruana, M., and Lévy, T. (2007). *Differential equations driven by rough paths* (Vol. 1908). Springer.

Ryan, R. A. (2002). *Introduction to tensor products of banach spaces* (Vol. 73). Springer.

Appendix B

Supplementary material of Chapter 3

Contents

B.1 Augmentations

We recall that an augmentation is a map

$$\phi\colon \mathcal{S}(\mathbb{R}^d) \to \mathcal{S}(\mathbb{R}^e)^p$$

We give below the precise definition of the different augmentations considered in the study, which are summarized in Table B.1. These augmentations were not typically introduced using such language, so this serves as a reference for how the existing literature may be interpreted through the generalised signature method.

Throughout the section, we consider a sequence $\mathbf{x} = (x_1, \dots, x_n) \in \mathcal{S}(\mathbb{R}^d)$ and timestamps $\mathbf{t} = (t_1, \dots, t_n) \in \mathcal{S}(\mathbb{R})$. We recall that if $\mathbf{x}$ is regularly sampled then $\mathbf{t}$ is usually set to $\mathbf{t} = (1, \dots, n)$.

Time augmentation We recall the definition of the time augmentation:

$$\phi(\mathbf{x}) = \big((t_1, x_1), \dots, (t_n, x_n)\big) \in \mathcal{S}(\mathbb{R}^{d+1}).$$

It ensures uniqueness of the signature transformation and removes the parametrization invariance (Levin et al., 2013).

Invisibility-reset augmentation First introduced by Yang et al. (2017), the invisibility-reset augmentation consists in adding a coordinate to the sequence $\mathbf{x}$ that is constant equal to 1 but drops to 0 at the last time step, i.e.,

$$\phi(\mathbf{x}) = \big((1, x_1), \dots, (1, x_{n-1}), (1, x_n), (0, x_n), (0, 0)\big) \in \mathcal{S}(\mathbb{R}^{d+1}).$$

This augmentation adds information on the initial position of the path, which is otherwise not included in the signature as it is a translation-invariant map.

Basepoint augmentation Introduced by Kidger and Lyons (2020), the basepoint augmentation has the same goal as the invisibility-reset augmentation: removing the translation-invariant property of the signature. It simply adds the point 0 at the beginning of the sequence:

$$\phi(\mathbf{x}) = (0, x_1, \dots, x_n) \in \mathcal{S}(\mathbb{R}^d).$$

The main difference compared to the invisibility-reset augmentation is that the signature of $\mathbf{x}$ is contained in the signature of the invisibility-reset augmented path, whereas it is not in the signature of the basepoint augmented path. The price paid is that the invisibility-reset augmentation introduces redundancy into the signature, and is more computationally expensive due to the additional channel. (Recall that the signature method scales as $\mathcal{O}(d^N)$, where d is the input channels and N is the depth of the (log)signature.)

Lead-lag augmentation The lead-lag augmentation, introduced by Chevyrev and Kormilitzin (2016a) and Flint et al. (2016) has been used in several applications (see for example Lyons et al. (2014), Kormilitzin et al. (2016), and Yang et al. (2017)). It adds lagged copies of the path as new coordinates. This then explicitly captures the quadratic variation of the underlying process (Flint et al., 2016). As many different lags as desired may be added. If there is a single lag of a single timestep, then this corresponds to

$$\phi(\mathbf{x}) = ((x_1, x_1), (x_2, x_1), (x_2, x_2), \dots, (x_n, x_n)) \in \mathcal{S}(\mathbb{R}^{2d}).$$

Coordinate projections For multidimensional streams, one may want to compute the signature of a subset of coordinates individually, rather than the signature of the whole stream; doing so restricts the interaction considered by the signature to just those between the projected coordinates. Let $\mathbf{x^1}, \dots, \mathbf{x^d} \in \mathcal{S}(\mathbb{R})$ denote the different coordinates of $\mathbf{x} \in \mathcal{S}(\mathbb{R}^d)$.

Then we define the singleton coordinate projection as

$$\phi(\mathbf{x}) = \big((\mathbf{t}, \mathbf{x^1}), (\mathbf{t}, \mathbf{x^2}), \dots, (\mathbf{t}, \mathbf{x^d})\big) \in \mathcal{S}(\mathbb{R}^2)^d,$$

whilst considering all possible pairs of coordinates yields the augmentation

$$\phi(\mathbf{x}) = \big((\mathbf{t}, \mathbf{x^1}, \mathbf{x^2}), (\mathbf{t}, \mathbf{x^1}, \mathbf{x^3}), \dots, (\mathbf{t}, \mathbf{x^d}, \mathbf{x^{d-1}})\big) \in \mathcal{S}(\mathbb{R}^3)^{d(d-1)},$$

and all possible triples yields the augmentation

$$\phi(\mathbf{x}) = \big((\mathbf{t}, \mathbf{x^1}, \mathbf{x^1}, \mathbf{x^2}), (\mathbf{t}, \mathbf{x^1}, \mathbf{x^1}, \mathbf{x^3}), \dots, (\mathbf{t}, \mathbf{x^d}, \mathbf{x^d}, \mathbf{x^{d-1}})\big) \in \mathcal{S}(\mathbb{R}^4)^{d(d^2-1)}.$$

Table B.1 – Summary of the different augmentations

	e	p	Property
Fixed augmentations			
None	d	1	
Time	$d+1$	1	sensitivity to parametrization, uniqueness of the signature map
Invisibility-reset	$d+1$	1	sensitivity to translation
Basepoint	d	1	sensitivity to translation
Lead-lag	$2d$	1	information about quadratic variation, uniqueness of the signature map
Coordinates projection			dimensionality reduction
with singletons	2	d	
with pairs	3	$d(d-1)$	
with triplets	4	$d(d^2-1)$	
Random projections	e	p	dimensionality reduction
Learnt augmentations			
Learnt projections	e	p	data-dependent and linear
Stream-preserving neural network	e	1	data-dependent
Multi-headed stream-preserving NN	e	p	data-dependent

The decision to always include a time dimension is a somewhat arbitrary one, and it may alternatively be excluded if desired. (This is done so as to make sense of singleton coordinate projections; otherwise the result is a collection of univariate time series, for which the signature extracts only the increment due to the tree-like equivalence property.)

Random projections When the dimension of the input path is very large, Lyons and Oberhauser (2017) have proposed to project it into a smaller space by taking multiple random projections. Let $e < d$ and let $A_i : \mathbb{R}^d \to \mathbb{R}^e$ be random affine transformations indexed by $i \in \{1, \dots, p\}$. Then ϕ is defined as

$$\phi(\mathbf{x}) = ((A_1 x_1, \dots, A_1 x_n), \dots, (A_p x_1, \dots, A_p x_n)) \in \mathcal{S}(\mathbb{R}^e)^p.$$

Learnt projections Rather than taking random projections, Liao et al. (2019) learn it from the data. This takes exactly the same form as the random projections, except that the A_i are learnt.

Stream-preserving neural network Kidger et al. (2019) introduce arbitrary learnt sequence-to-sequences maps prior to the signature transform, and refer to such maps, when parameterised as neural networks, as stream-preserving neural networks. For example these may be standard convolutional or recurrent architectures. In general this may be any learnt transformation

$$\phi \colon \mathcal{S}(\mathbb{R}^d) \to \mathcal{S}(\mathbb{R}^e).$$

Multi-headed stream-preserving neural network A straightforward extension of stream-preserving neural networks is to use multiple such networks, so as to avoid a potential bottleneck

through the single signature map that it is eventually used in. Letting $\phi^1, \dots, \phi^p$ be p different stream-preserving neural networks, then this gives an augmentation

$$\phi(\mathbf{x}) = (\phi^1(\mathbf{x}), \dots, \phi^p(\mathbf{x})) \in (\mathcal{S}(\mathbb{R}^e))^p.$$

B.2 Rescaling

The signature transform can be written as a sequence of tensors, indexed by $k \in \{1, \dots, N\}$. The k-th term is of size $\mathcal{O}(1/k!)$, as it is computed by an integral over a k-dimensional simplex. It is typical that rescaling these terms to be $\mathcal{O}(1)$ will aid subsequent learning procedures.

One option is to simply multiply the k-th term by $k!$, which we call *post-signature* scaling.

However, it is possible that the previous option may suffer from numerical stability issues. Thus we also explore the performance of an option, called *pre-signature scaling*, which may alleviate this, which is to multiply the input $\mathbf{x}$ by some scaling factor $\alpha \in \mathbb{R}$. Then the k-th term will be of size $\mathcal{O}(\alpha^k/k!)$, and so by taking $\alpha = (N!)^{1/N}$ the N-th term in the signature will be $\mathcal{O}(1)$; the trade-off is that Stirling's approximation then shows that the $N/2$-th term will be of size $\mathcal{O}(2^{N/2})$.

B.3 Implementation details

B.3.1 General notes

Code All the code for this project is available at https://github.com/jambo6/generalised-signature-method.

Libraries The machine learning framework used was PyTorch (Paszke et al., 2019) version 1.3.1. Signatures and logsignatures were computed using the Signatory library (Kidger and Lyons, 2020) version 1.1.6. Scikit-learn (Pedregosa et al., 2011) version 0.22.1 was used for the logistic regression and random forest models. The experiments were tracked using the Sacred framework (Greff et al., 2017) version 0.8.1.

Normalisation Every dataset was normalised so that each channel has mean zero and unit variance.

Architectures Two different GRU models were used on every dataset; a 'small' one with 32 hidden channels and 2 layers, and a 'large' one with 256 hidden channels and 3 layers.

Likewise, two different Residual CNN models were considered. The 'small' one used 6 blocks, each composed of batch normalisation, ReLU activation, convolution with 32 filters and kernel size 4, batch normalisation, ReLU activation, and a final convolution with 32 filters and kernel size 4, so that there are also 32 channels along the 'residual path'. A final two-hidden-layer neural network with 256 neurons was placed on the output. The 'large' is similar, except that it used 128 filters in both the blocks and the residual path, had 8 blocks, used a kernel size of 8, and the final neural network had 1024 neurons.

The logistic regression was performed three times with different amounts of L^2 regularisation, with scaling hyperparameters of 0.01, 0.2 and 1; for every experiment the regularization hyperparameter achieving the best accuracy on the test set was used.

The random forest used the default Scikit-learn implementation with a maximum depth of 6 and 100 trees.

Optimiser The GRU and CNN were optimised using Adam (Kingma and Ba, 2015). The learning rate was 0.01 for the GRU, and 0.001 for the residual CNN. The small models were trained for a maximum of 500 epochs; the large models were trained for a maximum of 1000 epochs. The learning rate was decreased by a factor of 10 if validation loss did not improve over a plateau of 10 epochs. Early stopping was used if the validation loss did not improve for 30 epochs. After training the parameters were always rolled back to those that demonstrated the best validation loss over the course of training. The batch size used varied by dataset; in each it was taken to be the power of two that meant that the number of batches per epoch was closest to 40.

Computing infrastructure Experiments were run on an Amazon AWS G3 Instance (g3.16xlarge) equipped with 4 Tesla M60s, parallelized using GNUParallel (Tange, 2011).

B.3.2 Analysis of variations of the signature method

Splits The UEA archive comes with a pre-defined train-test split, which we respect. We take an 80%/20% train/validation split in the training data, stratified by class label. For the Human Activities and Postural Transitions dataset, we take a 60%/15%/25% train/validation/test split from the whole dataset. For the Speech Commands dataset, we take a 68%/17%/15% train/validation/test split from the whole dataset. (These somewhat odd choices corresponding to taking either 25% or 15% of the dataset as test, and then splitting the remaining 80%/20% between train and validation.) These train/validation splits are only used for the training of the GRU and CNN classifiers.

Combinations In total we tested 8569 different combinations.

The variations tested are divided into groups. The first group consists of the sensitivity-adding augmentations, namely time, basepoint and invisibility-reset. Relative to the baseline model, we test every possible combination of these. (Including using none of them.)

The second group consists of those other augmentations, namely the lead-lag, singleton coordinate projection, pair coordinate projection, triplet coordinate projection, random projections, learnt projections, and multi-headed stream preserving neural networks, and finally also the case of no additional augmentation.

For the random projections, we consider four possibilities, with $e \in \{3, 6\}$ and $p \in \{2, 5\}$, all relative to the baseline model.

For no additional augmentation, lead-lag, coordinate projections, learnt projections, and the multi-headed stream preserving neural networks, we compose them with the time, time+basepoint and time+invisibility-reset augmentations (the clear best three from the first group), all relative to the baseline model.

For the learnt projections, we consider four different possibilities corresponding to $e \in \{3, 6\}$ and $p \in \{2, 5\}$; together with the time/time+basepoint/time+invisibility-reset cases this yields a total of twelve possibilities.

For the multi-headed stream-preserving neural networks, we again consider four different possibilities corresponding to $e \in \{3, 6\}$ and $p \in \{2, 5\}$, for a total of twelve possible augmentation strategies. In each the neural network operates elementwise, so as to map one sequence to another, and is given by a feedforward neural network of three hidden layers separated by ReLU activation functions. When $e = 3$ the hidden layers have 16 neurons each, and when $e = 6$ they have 32 neurons each.

For both the learnt projections and multi-headed stream-preserving neural networks, training these requires backpropagating through the model, so these were only considered for the GRU

Table B.2 – Summary of the number of combinations considered and omitted.

Variations	**# Variations**	**# Classifiers**	**# Omitted Combinations**	**# Total Combinations**
Basic augmentations (Figure 3.2)	6	6	54	936
Other augmentations (Figure 3.3)				
Lead-lag/ None	3	6	100/27	468
Coordinates projection (1)/(2)/(3)	3	6	12/12/54	468
Random projections	4	6	32	624
Learnt projections / MHSP	12	4	348/176	1248
Windows (Figure 3.4)	8	6	227	1248
Signature/ Logsignature transform	12	6	361	1872
Rescalings (Figure B.2)	3	6	12	468
Total			1415	9984

and residual CNN model. (The logistic regression model would in principle be possible as well, except that we ended up implementing this through Scikit-learn rather than PyTorch.)

We note that there are a great many possible ways of doing stream preserving neural networks, of which these are a small fraction. Their relatively weak performance here may likely be improved upon with greater tuning on an individual task, or the selection of better final models than were considered here.

The third group consisted of the different windows. Recall that the baseline model used a global window; we then consider varying this to two possible sliding windows, two possible expanding windows, and three possible dyadic windows. The two possible sliding/expanding windows are chosen so that either 5 or 20 windows are applied across the full length of the dataset. The three possible dyadic windows are depths 2, 3, 4. Thus in total there are 8 possible window combinations we consider.

The fourth group consists of rescaling options, namely no rescaling, pre-signature rescaling, and post-signature rescaling.

Omissions For the empirical study on the variations on the signature method, we excluded those UEA datasets with a dimension d over 60, so as to reduce the computational cost. This results removes 6 of the 30 datasets from the study, namely DuckDuckGeese, FaceDetection, Heartbeat, InsectWingbeat, MotorImagery, and PEMS-SF. These were nonetheless used in the demonstration of performance of the canonical signature method in Figure 3.6. Furthermore those combinations of dataset/variation/model which produced more than 10^5 signature features were omitted, to keep the computation managable. See Table B.2.

B.3.3 The canonical signature pipeline

For each dataset, we implement the following steps. First, the sequences are augmented with time and basepoint augmentations. Then, we consider every combination of signature depth in $\{1, 2, 3, 4, 5, 6\}$ and hierarchical dyadic window depth in $\{2, 3, 4\}$. For each of these choices, we perform a randomized grid search on a random forest classifier to optimize its number of trees

Table B.3 – Average run time (in seconds) for various experiments. mean (std), averaged over all UEA datasets.

	Classifier			
	CNN	GRU	Logistic regression	Random forest
Time augment & Global window (Baseline)	69.8 (98.0)	22.2 (31.8)	2.67 (7.09)	2.23 (4.84)
Augmentation				
None	48.1 (63.5)	16.8 (33.6)	3.55 (9.91)	66.3 (321)
Lead-lag	48.58 (69.99)	15.2 (18.1)	5.76 (11.7)	3.35 (6.04)
Coordinates projection (1)	32.8 (31.49)	13.4 (17.8)	1.37 (4.2)	12.2 (59.3)
Coordinates projection (2)	41.5 (51.4)	22.6 (62.3)	3.01 (8.54)	42.3 (203)
Coordinates projection (3)	41.3 (39.9)	19.1 (24.5)	5.41 (9.76)	6.3 (14.1)
Random projection	62.2 (70.1)	21.1 (31.2)	0.86 (1.25)	1.4 (2.47)
Learnt projection	917 (1288)	752 (972)	–	–
Multi-headed stream-preserving	1051 (1677)	1758 (4442)	–	–
Window				
Sliding	90.6 (120)	79.4 (175)	10.1 (27.4)	6.4 (16.0)
Expanding	102 (133)	68.7 (115)	9.98 (27.2)	7.17 (19.0)
Dyadic	725 (868)	56.9 (65.1)	12.5 (33.2)	7.59 (18.3)

and maximal depth parameters. We test 20 combinations randomly sampled from the following grids:

$$\begin{aligned} \text{n\_trees} &= [50, 100, 500, 1000], \\ \text{max\_depth} &= [2, 4, 6, 8, 12, 16, 24, 32, 45, 60, 80, \text{None}]. \end{aligned}$$

Note that a maximal depth set to 'None' means that the trees are expanded until all leaves contain exactly one sample. Finally, we choose the combination of signature and hierarchical dyadic window depths which maximise the out-of-bag score.

B.4 Additional results

B.4.1 Analysis of variations of the signature method

Running time To get a sense of the cost of each augmentation or window, we present the run times of each augmentation/model combination, and each window/model combination. (The times for varying between signature and logsignature, and between different rescalings, are largely insignificant.) See Table B.3.

The run times are averaged over every UEA dataset. As the datasets are of very different sizes this thus represents quite a crude statistic, and in particular produces very large variances, so these are most meaningful simply with respect to each other.

Sensitivity-inducing augmentations broken down by dataset type Table B.4 shows the average rank of each of the first group of augmentations (that add sensitivity to certain kinds of

Table B.4 – Average ranks for different augmentations by type of data. Lower is better.

	Augmentation					
Data type	None	Time	Basepoint	Invisibility-reset	Time + Basepoint	Time + Invisibility-reset
EEG	3.88	3.50	4.00	**2.00**	4.00	3.63
HAR	5.00	2.95	4.85	3.65	**2.00**	2.55
MOTION	5.25	2.75	5.75	3.88	**1.50**	1.88
OTHER	4.43	3.31	4.88	3.19	2.87	**2.31**

perturbation) by dataset type, where the types are taken from Bagnall et al. (2018). (This may be regarded as a companion to Table B.5.)

It is interesting to note that for EEG data, it seems better not to consider the time augmentation, whereas it is the case for other applications. In particular the combination of time and basepoint augmentations achieve the best ranks for human action and motion recognition (HAR and MOTION in Table B.4). Recognizing an action may not be translation-invariant nor invariant by time reparametrization.

Table B.5 – Average ranks for different augmentations by dataset characteristics. Lower is better.

	Augmentation							
	None	Lead-lag	Coordinates projection (1)	(2)	(3)	Random Projection	Learnt Projection	MHSP
Data type								
EEG	4.88	4.83	6.50	3.13	5.67	4.38	**2.75**	**2.75**
HAR	2.25	**1.78**	7.20	3.50	2.90	4.75	6.50	6.50
MOTION	2.63	**1.75**	7.00	4.50	2.13	5.00	7.33	5.00
OTHER	2.88	3.92	5.44	**2.63**	3.29	4.69	6.00	5.21
Series length								
<50	3.20	**2.20**	7.40	3.20	2.70	5.10	7.00	5.20
50-100	2.20	**1.33**	6.00	4.10	2.75	6.20	4.80	5.10
100-500	**2.28**	2.57	7.28	3.50	2.63	4.17	6.63	5.33
>500	4.00	4.00	5.28	**2.64**	4.57	4.07	4.40	5.60
Dimension d								
2	4.67	3.5	6.33	4.33	4.0	**2.83**	6.67	3.67
3-5	2.5	**2.21**	6.36	3.43	3.14	4.64	6.67	6.83
6-8	3.25	**2.5**	6.94	3.0	3.56	5.0	5.29	6.0
>8	**2.25**	3.75	6.31	3.19	2.5	5.19	5.29	4.19

Other augmentations broken down by dataset characteristichs Table B.5presents the average ranks of the other augmentations borken down by some characteristics of the datasets.

Here we see that there is generally a better choice than doing nothing at all, but that this better choice. For example on long or high-dimensional datasets, coordinate projections often perform well, whilst multi-headed stream preserving transformations do substantially better on EEG datasets. Lead-lag remains a strong choice in many cases.

Depth study on the signature transform In the main text we focused on the difference between the signature and logsignature transforms, and stated that larger depths must be chosen by a bias-variance tradeoff. Here we consider varying the depth together with the choice of signature or logsignature, and taking the best transform for each depth. See Figure B.1. We see that larger depths do indeed generally correspond to increased performance, up to a point. The optimal depth will depend on the complexity of the task, as the number of features increases exponentially with the depth.

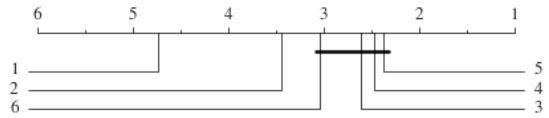


Figure B.1 – Critical differences plot for the depth study on the UEA datasets.

Rescaling critical difference diagram In Figure B.2, we see that pre-signature rescaling performs significantly worse than the other two options and that no significant difference between post-rescaling and no rescaling is found.

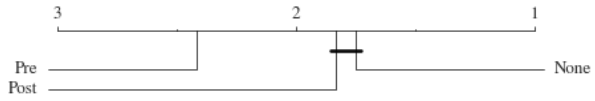


Figure B.2 – Performance of different rescalings

B.4.2 Complete results

We present in Tables B.6, B.7, B.8, B.9, B.10 and B.11 the performance of the different signature variations on each dataset. The tables were obtained by maximizing the test accuracy of the signature method over the different classifiers considered. Recall that some values are omitted due to the large number of signature features that would be obtained.

B.4.3 Canonical signature method

In Table B.12 we give the full results for our canonical signature method on all UEA datasets, together with the results of Ruiz et al. (2020) used in Figure 3.6.

Finally, we give in Table B.13 the hyperparameters that were selected for each dataset in the signature pipeline model.

Table B.6 – Accuracy of sensitivity-inducing augmentations per dataset

Dataset	Augmentation					
	None	Time	Basepoint	Invisibility-reset	Time + Basepoint	Time + Invisibility-reset
ArticularyWordRecognition	96.0	96.3	95.7	96.3	97.7	97.0
AtrialFibrillation	46.7	46.7	40.0	33.3	40.0	40.0
BasicMotions	100.0	100.0	100.0	100.0	100.0	100.0
CharacterTrajectories	88.3	93.2	86.4	88.7	93.8	93.7
Cricket	91.7	94.4	94.4	97.2	97.2	95.8
ERing	80.0	92.6	77.4	89.6	91.9	92.2
EigenWorms	72.5	79.4	74.8	76.3	87.0	81.7
Epilepsy	84.8	89.9	91.3	91.3	97.1	94.9
EthanolConcentration	27.8	29.3	33.5	41.8	34.6	41.4
FingerMovements	55.0	52.0	57.0	58.0	55.0	56.0
HandMovementDirection	29.7	33.8	32.4	33.8	36.5	32.4
Handwriting	21.9	30.8	23.6	24.6	30.6	28.7
JapaneseVowels	85.4	85.1	97.3	98.1	97.3	98.1
LSST	42.0	47.4	44.0	44.4	50.9	48.7
Libras	72.8	84.4	65.0	75.0	80.0	77.2
NATOPS	81.7	88.3	79.4	79.4	91.1	92.2
PenDigits	91.1	97.1	88.3	93.1	96.8	97.1
PhonemeSpectra	4.7	8.2	4.3	5.7	10.0	8.1
RacketSports	78.9	80.3	78.9	82.9	82.9	81.6
SelfRegulationSCP1	81.6	83.3	76.8	84.0	75.4	85.0
SelfRegulationSCP2	57.2	56.7	56.1	56.7	56.1	55.0
SpokenArabicDigits	82.5	85.5	80.5	88.0	85.1	90.1
StandWalkJump	60.0	46.7	40.0	46.7	40.0	46.7
UWaveGestureLibrary	84.1	87.5	79.7	82.8	87.5	83.4
Human Activity	73.0	76.6	92.3	92.2	93.0	93.8
Speech Commands	71.4	75.9	74.7	74.9	79.7	79.5
Average rank	4.69	3.12	4.87	3.29	**2.5**	2.54

Table B.7 – Accuracy of other augmentations per dataset

Dataset	Augmentation							
	None	Lead-lag	Coordinates projection (1)	(2)	(3)	Random projection	Learnt projection	MHSP
ArticularyWordRecognition	97.7	96.3	83.3	95.7	97.0	95.3	73.7	80.3
AtrialFibrillation	46.7	40.0	53.3	53.3	46.7	66.7	46.7	53.3
BasicMotions	100.0	100.0	80.0	100.0	100.0	100.0	97.5	87.5
CharacterTrajectories	93.8	95.3	43.9	93.2	93.8	93.3	89.6	91.1
Cricket	97.2	98.6	90.3	97.2	95.8	88.9	69.4	56.9
ERing	92.6	94.8	79.3	89.3	91.9	74.4	62.2	61.1
EigenWorms	87.0	87.8	50.4	84.0	89.3	78.6	–	–
Epilepsy	97.1	97.1	55.8	95.7	95.7	81.9	67.4	65.9
EthanolConcentration	41.4	39.9	42.2	43.3	42.2	30.4	32.3	30.0
FingerMovements	56.0	–	59.0	58.0	–	55.0	60.0	65.0
HandMovementDirection	36.5	31.1	31.1	40.5	37.8	37.8	33.8	44.6
Handwriting	30.8	33.5	11.3	27.8	30.0	21.6	12.6	13.2
JapaneseVowels	98.1	97.6	94.1	97.8	97.6	84.1	95.4	95.9
LSST	50.9	55.6	43.5	51.7	52.8	43.7	34.4	39.8
Libras	84.4	86.7	47.8	83.9	85.0	86.7	73.3	81.1
NATOPS	92.2	–	33.3	90.6	91.1	85.6	83.9	81.7
PenDigits	97.1	98.3	60.2	96.8	97.2	96.7	96.5	97.4
PhonemeSpectra	10.0	–	4.5	9.4	10.6	8.9	7.2	7.7
RacketSports	82.9	82.2	53.3	85.5	84.2	75.7	73.7	75.0
SelfRegulationSCP1	85.0	86.0	61.8	85.0	84.0	81.6	86.7	84.6
SelfRegulationSCP2	56.7	57.2	55.6	58.9	55.6	60.6	59.4	57.8
SpokenArabicDigits	90.1	96.6	58.5	86.0	90.0	83.0	88.0	86.0
StandWalkJump	46.7	40.0	40.0	53.3	40.0	53.3	–	–
UWaveGestureLibrary	87.5	88.8	50.6	85.6	87.5	86.2	74.1	75.6
Human Activity	93.8	93.6	75.8	93.2	93.6	69.2	91.3	91.5
Speech Commands	79.7	–	14.9	77.1	–	70.2	–	76.1
Average ranks	2.9	**2.77**	6.52	3.33	3.23	4.71	5.83	5.31

Table B.8 – Accuracy of windows per dataset

	Window			
Dataset	Global	Sliding	Expanding	Dyadic
ArticularyWordRecognition	96.3	89.3	99.0	99.0
AtrialFibrillation	46.7	46.7	46.7	60.0
BasicMotions	100.0	100.0	100.0	100.0
CharacterTrajectories	93.2	94.6	96.9	97.1
Cricket	97.2	93.1	97.2	95.8
ERing	90.7	88.5	91.9	94.8
EigenWorms	80.2	74.8	78.6	76.3
Epilepsy	89.9	92.8	92.0	94.2
EthanolConcentration	30.4	38.8	30.0	35.7
FingerMovements	50.0	–	–	–
HandMovementDirection	33.8	33.8	36.5	33.8
Handwriting	30.1	21.5	30.2	27.2
JapaneseVowels	85.1	76.5	88.1	89.2
LSST	47.8	43.1	48.3	46.6
Libras	83.3	85.6	91.1	90.0
NATOPS	93.3	84.4	90.6	–
PenDigits	95.8	–	–	97.6
PhonemeSpectra	8.6	9.2	9.6	10.4
RacketSports	82.2	80.3	84.2	88.2
SelfRegulationSCP1	82.6	87.4	84.0	86.7
SelfRegulationSCP2	56.7	60.6	54.4	56.1
SpokenArabicDigits	85.5	91.5	93.7	96.6
StandWalkJump	46.7	53.3	46.7	53.3
UWaveGestureLibrary	86.6	79.4	89.1	89.7
Human Activity	76.1	73.0	80.4	81.7
Speech Commands	75.9	76.5	82.3	83.0
Average ranks	2.83	3.04	2.17	**1.73**

Table B.9 – Accuracy of signature and logsignature transforms per dataset.

	Transform	
Dataset	Signature	Logsignature
ArticularyWordRecognition	97.7	97.3
AtrialFibrillation	60.0	53.3
BasicMotions	100.0	100.0
CharacterTrajectories	93.8	93.8
Cricket	100.0	100.0
ERing	90.0	89.3
EigenWorms	79.4	81.7
Epilepsy	93.5	91.3
EthanolConcentration	31.9	30.0
FingerMovements	59.0	56.0
HandMovementDirection	40.5	40.5
Handwriting	35.3	24.5
JapaneseVowels	85.9	86.8
LSST	52.0	46.4
Libras	90.6	87.8
NATOPS	89.4	91.7
PenDigits	97.8	97.5
PhonemeSpectra	8.9	7.6
RacketSports	85.5	84.9
SelfRegulationSCP1	84.0	83.3
SelfRegulationSCP2	57.2	56.1
SpokenArabicDigits	87.5	85.8
StandWalkJump	53.3	53.3
UWaveGestureLibrary	90.0	86.9
Human Activity	78.7	78.3
Speech Commands	75.9	76.3
Average ranks	**1.25**	1.75

Table B.10 – Accuracy of rescaling choices per dataset

	Rescaling		
Dataset	None	Post	Pre
ArticularyWordRecognition	97.3	97.0	97.7
AtrialFibrillation	53.3	53.3	46.7
BasicMotions	100.0	100.0	100.0
CharacterTrajectories	94.6	94.6	94.6
Cricket	98.6	97.2	97.2
ERing	93.7	93.7	93.0
EigenWorms	80.9	80.9	79.4
Epilepsy	92.0	92.0	91.3
EthanolConcentration	31.2	31.6	30.0
FingerMovements	54.0	54.0	50.0
HandMovementDirection	35.1	32.4	29.7
Handwriting	36.6	36.4	37.1
JapaneseVowels	87.3	85.9	85.7
LSST	55.8	55.6	55.4
Libras	85.0	86.1	84.4
NATOPS	92.8	92.8	91.7
PenDigits	96.6	96.7	96.7
PhonemeSpectra	8.0	8.1	8.2
RacketSports	84.2	84.2	83.6
SelfRegulationSCP1	79.5	83.3	84.6
SelfRegulationSCP2	56.1	57.2	56.7
SpokenArabicDigits	90.5	90.5	90.2
StandWalkJump	46.7	53.3	46.7
UWaveGestureLibrary	87.5	87.2	87.2
Human Activity	85.0	84.6	85.1
Speech Commands	77.0	75.7	75.9
Average ranks	**1.73**	1.92	2.35

Table B.11 – Accuracy of (log)signature depth per dataset.

	Depth					
Dataset	1	2	3	4	5	6
ArticularyWordRecognition	83.3	96.0	97.3	97.7	95.3	–
AtrialFibrillation	40.0	40.0	60.0	33.3	40.0	53.3
BasicMotions	70.0	100.0	100.0	100.0	100.0	92.5
CharacterTrajectories	42.3	88.0	93.2	93.8	92.9	93.8
Cricket	30.6	93.1	97.2	98.6	100.0	–
ERing	77.0	89.6	90.0	89.3	88.9	84.8
EigenWorms	46.6	81.7	79.4	79.4	–	–
Epilepsy	50.7	78.3	89.9	93.5	93.5	93.5
EthanolConcentration	25.5	30.8	30.0	31.2	31.9	27.4
FingerMovements	57.0	58.0	59.0	–	–	–
HandMovementDirection	40.5	36.5	37.8	39.2	32.4	–
Handwriting	7.3	22.4	32.4	33.3	35.3	32.7
JapaneseVowels	78.9	85.9	86.8	84.3	81.4	–
LSST	40.9	45.6	47.6	50.6	52.0	44.7
Libras	51.7	77.2	85.0	87.8	88.9	90.6
NATOPS	35.0	86.7	91.7	–	–	–
PenDigits	60.0	90.4	96.9	97.7	97.4	97.8
PhonemeSpectra	4.1	7.6	8.9	–	–	–
RacketSports	44.1	77.0	78.9	84.9	85.5	82.2
SelfRegulationSCP1	53.6	80.2	84.0	83.3	81.9	–
SelfRegulationSCP2	56.1	55.0	56.7	54.4	57.2	–
SpokenArabicDigits	52.1	85.8	85.5	87.5	–	–
StandWalkJump	46.7	46.7	46.7	46.7	53.3	46.7
UWaveGestureLibrary	49.4	83.1	86.6	87.8	90.0	88.1
Human Activity	47.7	78.3	76.0	78.7	78.6	–
Speech Commands	14.8	69.6	76.3	–	–	–
Average ranks	4.73	3.44	2.62	2.48	**2.38**	3.04

Dataset	Classification method								
	DTWD	DTWA	DTWI	HIVE COTE	MLCN	MUSE	TapNet	gRSF	Signature Pipeline
AWR	98.7	98.7	98.0	99.0	95.7	99.3	95.7	98.3	97.7
AtrialFibrillation	20.0	26.7	26.7	13.3	33.3	40.0	20.0	26.7	46.7
BasicMotions	97.5	100.0	100.0	100.0	87.5	100.0	100.0	100.0	100.0
Cricket	100.0	100.0	98.6	98.6	91.7	98.6	100.0	98.6	95.8
Epilepsy	96.4	97.8	97.8	100.0	73.2	99.3	95.7	97.8	95.7
EConcentration	32.3	31.6	30.4	79.1	37.3	47.5	30.8	34.6	43.3
ERing	91.5	92.6	91.9	97.0	94.1	97.4	90.4	95.2	94.8
FaceDetection	52.9	52.8	51.3	65.6	55.5	63.1	60.3	54.8	61.4
FingerMovements	53.0	51.0	52.0	55.0	58.0	55.0	47.0	58.0	52.0
HMD	18.9	20.3	29.7	44.6	52.7	36.5	33.8	41.9	20.3
Handwriting	60.7	60.7	50.9	48.2	30.9	52.2	28.1	37.5	37.9
Heartbeat	71.7	69.3	65.9	72.2	38.0	71.2	79.0	76.1	69.8
Libras	87.2	88.3	89.4	90.0	85.0	89.4	87.8	69.4	93.9
LSST	55.1	56.7	57.5	57.5	52.8	64.0	51.3	58.8	56.9
NATOPS	88.3	88.3	85.0	88.9	90.0	90.6	81.1	84.4	92.2
PenDigits	97.7	97.7	93.9	93.4	97.9	96.7	85.6	93.5	97.4
Racketsports	80.3	84.2	84.2	88.8	84.2	92.8	87.5	88.2	90.8
SCP1	77.5	78.5	76.5	85.3	90.8	69.6	93.5	82.3	78.8
SCP2	53.9	52.2	53.3	46.1	50.6	52.8	48.3	51.7	50.6
StandWalkJump	20.0	33.3	33.3	33.3	40.0	26.7	13.3	33.3	46.7
UWGL	90.3	90.0	86.9	89.1	85.9	93.1	90.0	89.7	90.9
Average Ranks	5.6	5.2	5.9	4.0	5.6	3.2	6.4	4.8	4.3

Table B.12 – Results of the signature canonical pipeline along with a selection of classifiers from Ruiz et al. (2020) (including the top performing MUSE algorithm) with a Random Forest for the UEA archive.

Dataset	Signature hyperparmeters		RF hyperparameters		Other
	Depth	Dyadic depth	Max depth	Num estimators	Training time (s)
ArticularyWordRecognition	2	2	45	500	60.3
AtrialFibrillation	1	2	None	50	35.9
BasicMotions	2	2	24	100	19.3
CharacterTrajectories	4	2	80	500	181.4
Cricket	2	4	6	500	249.0
DuckDuckGeese	1	2	16	100	140.9
ERing	2	3	8	1000	16.7
EigenWorms	3	3	12	100	250.1
Epilepsy	2	3	8	1000	42.8
EthanolConcentration	2	4	24	1000	454.2
FaceDetection	1	4	8	1000	1816.2
FingerMovements	1	2	4	100	30.8
HandMovementDirection	2	2	None	50	66.3
Handwriting	6	2	32	1000	280.3
Heartbeat	1	4	None	50	45.1
InsectWingbeat	1	3	45	1000	5367.5
JapaneseVowels	2	3	6	1000	95.4
LSST	4	2	60	1000	1590.5
Libras	6	2	None	100	28.4
MotorImagery	1	3	24	50	347.1
NATOPS	2	3	32	1000	37.8
PEMS-SF	1	3	80	1000	252.3
PenDigits	3	2	80	1000	302.3
PhonemeSpectra	2	4	45	1000	2188.7
RacketSports	3	2	None	500	13.9
SelfRegulationSCP1	3	2	None	100	186.6
SelfRegulationSCP2	3	2	6	50	138.1
SpokenArabicDigits	2	3	45	1000	1204.0
StandWalkJump	1	3	2	50	101.5
UWaveGestureLibrary	2	2	60	500	21.8

Table B.13 – Hyperparameters used for each dataset in the signature pipeline model.

Bibliography

Bagnall, A., Dau, H. A., Lines, J., Flynn, M., Large, J., Bostrom, A., Southam, P., and Keogh, E. (2018). The uea multivariate time series classification archive, 2018. *arXiv:1811.00075*.

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

Flint, G., Hambly, B., and Lyons, T. (2016). Discretely sampled signals and the rough Hoff process. *Stochastic Processes and their Applications*, *126*, 2593–2614.

Greff, K., Klein, A., Chovanec, M., Hutter, F., and Schmidhuber, J. (2017). The sacred infrastructure for computational research. In K. Huff, D. Lippa, D. Niederhut, and M. Pacer (Eds.), *Proceedings of the 16th python in science conference* (pp. 49–56).

Kidger, P., Bonnier, P., Perez Arribas, I., Salvi, C., and Lyons, T. (2019). Deep signature transforms. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 3099–3109). Curran Associates, Inc.

Kidger, P., and Lyons, T. (2020). Signatory: differentiable computations of the signature and logsignature transforms, on both CPU and GPU. *arXiv:2001.00706*. https://github.com/patrick-kidger/signatory

Kingma, D. P., and Ba, J. (2015). Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*.

Kormilitzin, A., Saunders, K., Harrison, P., Geddes, J., and Lyons, T. (2016). Application of the signature method to pattern recognition in the cequel clinical trial. *arXiv:1606.02074*.

Levin, D., Lyons, T., and Ni, H. (2013). Learning from the past, predicting the statistics for the future, learning an evolving system. *arXiv:1309.0260*.

Liao, S., Lyons, T., Yang, W., and Ni, H. (2019). Learning stochastic differential equations using RNN with log signature features. *arXiv:1908.08286*.

Lyons, T., Ni, H., and Oberhauser, H. (2014). A feature set for streams and an application to high-frequency financial tick data. *Proceedings of the 2014 International Conference on Big Data Science and Computing*, 5.

Lyons, T., and Oberhauser, H. (2017). Sketching the order of events. *arXiv:1708.09708*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 8024–8035). Curran Associates, Inc.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Ruiz, A. P., Flynn, M., and Bagnall, A. (2020). Benchmarking multivariate time series classification algorithms. *arXiv:2007.13156*.

Tange, O. (2011). Gnu parallel - the command-line power tool. *The USENIX Magazine*, *36*, 42–47.

Yang, W., Lyons, T., Ni, H., Schmid, C., Jin, L., and Chang, J. (2017). Developing the path signature methodology and its application to landmark-based human action recognition. *arXiv:1707.03993*.

Appendix C

Supplementary material of Chapter 4

Contents

C.1 Proof of Theorem 4.4

This section is devoted to the proof of Theorem 4.4. We will use extensively results from van Handel (2014). The next two lemmas first show that it is sufficient to obtain a uniform tail bound on the risk to control the convergence of $\widehat{m}$.

Lemma C.1. *For any $m \in \mathbb{N}$,*

$$\left|\widehat{L}_n(m) - L(m)\right| \leq \sup_{\beta \in B_{m,\alpha}} \left|\widehat{\mathcal{R}}_{m,n}(\beta) - \mathcal{R}_m(\beta)\right|.$$

Proof. Introducing $\widehat{\mathcal{R}}_{m,n}(\beta_m^*)$ yields

$$\widehat{L}_n(m) - L(m) = \widehat{\mathcal{R}}_{m,n}(\widehat{\beta}_m) - \mathcal{R}_m(\beta_m^*) = \widehat{\mathcal{R}}_{m,n}(\widehat{\beta}_m) - \widehat{\mathcal{R}}_{m,n}(\beta_m^*) + \widehat{\mathcal{R}}_{m,n}(\beta_m^*) - \mathcal{R}_m(\beta_m^*).$$

Since $\widehat{\beta}_m$ minimises $\widehat{\mathcal{R}}_{m,n}$ over $B_{m,\alpha}$, $\widehat{\mathcal{R}}_{m,n}(\widehat{\beta}_m) - \widehat{\mathcal{R}}_{m,n}(\beta_m^*) \leq 0$, which gives

$$\widehat{L}_n(m) - L(m) \leq \widehat{\mathcal{R}}_{m,n}(\beta_m^*) - \mathcal{R}_m(\beta_m^*) \leq \sup_{\beta \in B_{m,\alpha}} |\widehat{\mathcal{R}}_{m,n}(\beta) - \mathcal{R}_m(\beta)|.$$

In the same manner,

$$L(m) - \widehat{L}_n(m) \leq \sup_{\beta \in B_{m,\alpha}} |\widehat{\mathcal{R}}_{m,n}(\beta) - \mathcal{R}_m(\beta)|,$$

which proves the lemma. □

Lemma C.2. *For any $m > m^*$,*

$$\mathbb{P}(\widehat{m} = m) \leq \mathbb{P}\big(2 \sup_{\beta \in B_{m,\alpha}} |\widehat{\mathcal{R}}_{m,n}(\beta) - \mathcal{R}(\beta)| \geq \mathrm{pen}_n(m) - \mathrm{pen}_n(m^*)\big).$$

Proof. For any $m \in \mathbb{N}$,

$$\begin{aligned}\mathbb{P}(\widehat{m} = m) &\leq \mathbb{P}\left(\widehat{L}_n(m) + \mathrm{pen}_n(m) \leq \widehat{L}_n(m^*) + \mathrm{pen}_n(m^*)\right) \\ &= \mathbb{P}\left(\widehat{L}_n(m^*) - \widehat{L}_n(m) \geq \mathrm{pen}_n(m) - \mathrm{pen}_n(m^*)\right).\end{aligned}$$

Recall that, by the definition of the model (4.5), $m \mapsto L(m)$ is a decreasing function and that its minimum is attained at $m = m^*$. Therefore, for any $m \in \mathbb{N}$, $L(m^*) \leq L(m)$, and Lemma C.1 yields

$$\begin{aligned}\widehat{L}_n(m^*) - \widehat{L}_n(m) &= \widehat{L}_n(m^*) - L(m^*) + L(m^*) - L(m) + L(m) - \widehat{L}_n(m) \\ &\leq \widehat{L}_n(m^*) - L(m^*) + L(m) - \widehat{L}_n(m) \\ &\leq \sup_{\beta \in B_{m^*,\alpha}} |\widehat{\mathcal{R}}_{m,n}(\beta) - \mathcal{R}_m(\beta)| + \sup_{\beta \in B_{m,\alpha}} |\widehat{\mathcal{R}}_{m,n}(\beta) - \mathcal{R}_m(\beta)|.\end{aligned}$$

For $m > m^*$, $B_{m^*,\alpha} \subset B_{m,\alpha}$, which gives

$$\widehat{L}_n(m^*) - \widehat{L}_n(m) \leq 2 \sup_{\beta \in B_{m,\alpha}} |\widehat{\mathcal{R}}_{m,n}(\beta) - \mathcal{R}_m(\beta)|,$$

and the proof is complete. □

From now on, we denote by $Z_{m,n}$ the centered empirical risk for signatures truncated at m: for any $\beta \in B_{m,\alpha}$,

$$Z_{m,n}(\beta) = \widehat{\mathcal{R}}_{m,n}(\beta) - \mathcal{R}_m(\beta) = \frac{1}{n} \sum_{i=1}^{n} \big(Y_i - \langle \beta, S^m(X_i) \rangle\big)^2 - \mathbb{E}\big(Y - \langle \beta, S^m(X) \rangle\big)^2.$$

We will now derive a uniform tail bound on $Z_{m,n}(\beta)$, which is the main result needed to prove Theorem 4.4. In a nutshell, we show that $(Z_{m,n}(\beta))_{\beta \in B_{m,\alpha}}$ is a subgaussian process for some appropriate distance, and then use a chaining tail inequality (2014, Theorem 5.29) on $Z_{m,n}$.

Lemma C.3. *Under the assumptions (H_α) and (H_K), for any $m \in \mathbb{N}$, the process $(Z_{m,n}(\beta))_{\beta \in B_{m,\alpha}}$ is subgaussian for the distance*

$$D(\beta, \gamma) = \frac{K}{\sqrt{n}} \|\beta - \gamma\|, \tag{C.1}$$

where the constant K is defined by (4.6).

Proof. By definition, it is clear that $\mathbb{E} Z_{m,n}(\beta) = 0$ for any $\beta \in B_{m,\alpha}$. Let $\ell_{(X,Y)} \colon B_{m,\alpha} \to \mathbb{R}$ be given by

$$\ell_{(X,Y)}(\beta) = \big(Y - \langle \beta, S^m(X) \rangle\big)^2.$$

We first prove that $\ell_{(X,Y)}$ is K-Lipschitz. For any $\beta, \gamma \in B_{m,\alpha}$,

$$\begin{aligned}
|\ell_{(X,Y)}(\beta) - \ell_{(X,Y)}(\gamma)| &= \left|\left(Y - \langle \beta, S^m(X)\rangle\right)^2 - \left(Y - \langle \gamma, S^m(X)\rangle\right)^2\right| \\
&\leq 2\max\Big(\left|Y - \langle \beta, S^m(X)\rangle\right|, \left|Y - \langle \gamma, S^m(X)\rangle\right|\Big) \times \left|\langle \beta - \gamma, S^m(X)\rangle\right| \\
&\quad \text{(because } |a^2 - b^2| \leq 2\max(|a|,|b|)|a-b|\text{)} \\
&\leq 2\max\Big(\left|Y - \langle \beta, S^m(X)\rangle\right|, \left|Y - \langle \gamma, S^m(X)\rangle\right|\Big) \times \left\|S^m(X)\right\| \|\beta - \gamma\| \\
&\quad \text{(by the Cauchy-Schwartz inequality).}
\end{aligned}$$

Moreover, by the triangle inequality and Cauchy-Schwartz inequality,

$$\left|Y - \langle \beta, S^m(X)\rangle\right| \leq |Y| + \left\|S^m(X)\right\|\|\beta\| \leq K_Y + \alpha\left\|S^m(X)\right\|,$$

and, by Proposition 4.3,

$$\left\|S^m(X)\right\| \leq e^{\|X\|_{TV}} \leq e^{K_X}.$$

Consequently, $\left|Y - \langle \beta, S^m(X)\rangle\right| \leq K_Y + \alpha e^{K_X}$, and

$$\left|\ell_{(X,Y)}(\beta) - \ell_{(X,Y)}(\gamma)\right| \leq 2\big(K_Y + \alpha e^{K_X}\big)e^{K_X}\|\beta - \gamma\| = K\|\beta - \gamma\|.$$

Therefore, by Hoeffding's lemma (2014, Lemma 3.6), $\ell_{(X,Y)}(\beta) - \ell_{(X,Y)}(\gamma)$ is a subgaussian random variable with variance proxy $K^2\|\beta - \gamma\|^2$, which gives, for $\lambda \geq 0$,

$$\mathbb{E}\exp\left(\lambda\Big(\ell_{(X,Y)}(\beta) - \ell_{(X,Y)}(\gamma) - \mathbb{E}\big(\ell_{(X,Y)}(\beta) - \ell_{(X,Y)}(\gamma)\big)\Big)\right) \leq \exp\left(\frac{\lambda^2 K^2\left\|\beta - \gamma\right\|^2}{2}\right).$$

From this, it follows that

$$\begin{aligned}
\mathbb{E}e^{\lambda\left(Z_{m,n}(\beta) - Z_{m,n}(\gamma)\right)} &= \mathbb{E}\exp\left(\frac{\lambda}{n}\sum_{i=1}^n \ell_{(X_i,Y_i)}(\beta) - \ell_{(X_i,Y_i)}(\gamma) - \mathbb{E}\Big(\ell_{(X_i,Y_i)}(\beta) - \ell_{(X_i,Y_i)}(\gamma)\Big)\right) \\
&= \prod_{i=1}^n \mathbb{E}\exp\left(\frac{\lambda}{n}\Big(\ell_{(X_i,Y_i)}(\beta) - \ell_{(X_i,Y_i)}(\gamma) - \mathbb{E}\big(\ell_{(X_i,Y_i)}(\beta) - \ell_{(X_i,Y_i)}(\gamma)\big)\Big)\right) \\
&\leq \exp\left(\frac{\lambda^2 K^2\left\|\beta - \gamma\right\|^2}{2n}\right) = \exp\left(\frac{\lambda^2 D(\beta,\gamma)^2}{2}\right),
\end{aligned}$$

where $D(\beta,\gamma) = \frac{K\|\beta-\gamma\|}{\sqrt{n}}$, which completes the proof. □

We can now derive a maximal tail inequality for $Z_{m,n}(\beta)$.

Proposition C.4. *Under the assumptions (H_α) and (H_K), for any $m \in \mathbb{N}$, $x > 0$, $\beta_0 \in B_{m,\alpha}$,*

$$\mathbb{P}\left(\sup_{\beta \in B_{m,\alpha}} Z_{m,n}(\beta) \geq 108\sqrt{\pi}K\alpha\sqrt{\frac{s_d(m)}{n}} + Z_{m,n}(\beta_0) + x\right) \leq 36\exp\Big(-\frac{x^2 n}{144K^2\alpha^2}\Big),$$

where the constant K is defined by (4.6).

Proof. By Lemma C.3, $Z_{m,n}$ is a subgaussian process for D, defined by (C.1). So, we may apply

Theorem 5.29 of van Handel (2014) to $Z_{m,n}$ on the metric space $(B_{m,\alpha}, D)$:

$$\mathbb{P}\Big(\sup_{\beta\in B_{m,\alpha}} Z_{m,n}(\beta) - Z_{m,n}(\beta_0) \geq 36\int_0^\infty \sqrt{\log(N(\varepsilon, B_{m,\alpha}, D))}d\varepsilon + x\Big)$$
$$\leq 36\exp\Big(-\frac{x^2 n}{36\times 4K^2\alpha^2}\Big),$$

where $N(\varepsilon, B_{m,\alpha}, D)$ is the ε-covering number of $B_{m,\alpha}$ with respect to D, and where we use that

$$\operatorname{diam}(B_{m,\alpha}) = \frac{2K\alpha}{\sqrt{n}}.$$

Moreover, $N(\varepsilon, B_{m,\alpha}, D) = N(\frac{\sqrt{n}}{K}\varepsilon, B_{m,\alpha}, \|\cdot\|)$, and so, by Lemma 5.13 of van Handel (2014),

$$N(\varepsilon, B_{m,\alpha}, D) \leq \left(\frac{3K\alpha}{\sqrt{n}\varepsilon}\right)^{s_d(m)} \text{ if } \varepsilon < \frac{K\alpha}{\sqrt{n}},$$

and

$$N(\varepsilon, B_{m,\alpha}, D) = 1 \text{ otherwise.}$$

Therefore,

$$\begin{aligned}\int_0^\infty \sqrt{\log(N(\varepsilon, B_{m,\alpha}, D))}d\varepsilon &= \int_0^{\frac{K\alpha}{\sqrt{n}}} \sqrt{\log(N(\varepsilon, B_{m,\alpha}, D))}d\varepsilon \\ &\leq \int_0^{\frac{K\alpha}{\sqrt{n}}} \sqrt{s_d(m)\log\left(\frac{3K\alpha}{\sqrt{n}\varepsilon}\right)}d\varepsilon \\ &\leq 3K\alpha\sqrt{\frac{s_d(m)}{n}}\int_0^\infty 2x^2\exp\left(-x^2\right)dx = 3K\alpha\sqrt{\frac{s_d(m)}{n}}\sqrt{\pi}, \quad \text{(C.2)}\end{aligned}$$

where in the second inequality we use the change of variable $x = \sqrt{\log\left(\frac{2K\alpha}{\sqrt{n}\varepsilon}\right)}$. □

Since $\mathbb{P}(\widehat{m} \neq m^*) = \mathbb{P}(\widehat{m} > m^*) + \mathbb{P}(\widehat{m} < m^*)$, we divide the proof into two cases. Let us first consider $m > m^*$ in the next proposition.

Proposition C.5. *Let $0 < \rho < \frac{1}{2}$, and $\operatorname{pen}_n(m)$ be defined by (4.7):*

$$\operatorname{pen}_n(m) = K_{\text{pen}} n^{-\rho}\sqrt{s_d(m)}.$$

Let n_1 be the smallest integer satisfying

$$n_1 \geq \left(\frac{432\sqrt{\pi}K\alpha\sqrt{s_d(m^*+1)}}{K_{\text{pen}}(\sqrt{s_d(m^*+1)} - \sqrt{s_d(m^*)})}\right)^{1/(\frac{1}{2}-\rho)}. \quad \text{(C.3)}$$

Then, under the assumptions (H_α) and (H_K), for any $m > m^*$, $n \geq n_1$,

$$\mathbb{P}\left(\widehat{m} = m\right) \leq 74\exp\left(-C_3(n^{1-2\rho} + s_d(m))\right),$$

where the constant C_3 is defined by

$$C_3 = \frac{K_{\text{pen}}^2 d^{m^*+1}}{128 s_d(m^*+1)(72K^2\alpha^2 + K_Y^2)}.$$

Proof. Let

$$u_{m,n} = \frac{1}{2}\big(\text{pen}_n(m) - \text{pen}_n(m^*)\big) = \frac{K_{\text{pen}}}{2} n^{-\rho}\Big(\sqrt{s_d(m)} - \sqrt{s_d(m^*)}\Big).$$

As $m \mapsto \text{pen}_n(m)$ is increasing in m, it is clear that $u_{m,n} > 0$ for any $m > m^*$. From Lemma C.2, we see that

$$\begin{aligned}\mathbb{P}(\widehat{m} = m) &\le \mathbb{P}\Big(\sup_{\beta \in B_{m,\alpha}} |Z_{m,n}(\beta)| > u_{m,n}\Big) \\ &= \mathbb{P}\Big(\sup_{\beta \in B_{m,\alpha}} Z_{m,n}(\beta) > u_{m,n}\Big) + \mathbb{P}\Big(\sup_{\beta \in B_{m,\alpha}} (-Z_{m,n}(\beta)) > u_{m,n}\Big).\end{aligned}$$

We focus on the first term of the inequality, the second can be handled in the same way since Proposition C.4 also holds when $Z_{m,n}(\beta)$ is replaced by $-Z_{m,n}(\beta)$. Let β_0 be a fixed point in $B_{m,\alpha}$ that will be chosen later, we have

$$\begin{aligned}\mathbb{P}\Big(\sup_{\beta \in B_{m,\alpha}} Z_{m,n}(\beta) > u_{m,n}\Big) &= \mathbb{P}\Big(\sup_{\beta \in B_{m,\alpha}} Z_{m,n}(\beta) > u_{m,n},\, Z_{m,n}(\beta_0) \le \frac{u_{m,n}}{2}\Big) \\ &\quad + \mathbb{P}\Big(\sup_{\beta \in B_{m,\alpha}} Z_{m,n}(\beta) > u_{m,n},\, Z_{m,n}(\beta_0) > \frac{u_{m,n}}{2}\Big) \\ &\le \mathbb{P}\Big(\sup_{\beta \in B_{m,\alpha}} Z_{m,n}(\beta) > \frac{u_{m,n}}{2} + Z_{m,n}(\beta_0)\Big) \\ &\quad + \mathbb{P}\Big(Z_{m,n}(\beta_0) > \frac{u_{m,n}}{2}\Big). \qquad \text{(C.4)}\end{aligned}$$

We treat each term separately. The first one is handled by Proposition C.4. To this end, we need to ensure that $\frac{u_{m,n}}{2} - 108K\alpha\sqrt{\frac{\pi s_d(m)}{n}}$ is positive. By definition,

$$\begin{aligned}\frac{u_{m,n}}{2} - 108K\alpha\sqrt{\frac{\pi s_d(m)}{n}} &= \frac{K_{\text{pen}}}{2} n^{-\rho}\Big(\sqrt{s_d(m)} - \sqrt{s_d(m^*)}\Big) - 108K\alpha\sqrt{\frac{\pi s_d(m)}{n}} \\ &= \sqrt{s_d(m)} n^{-\rho} \frac{K_{\text{pen}}}{2}\left(1 - \sqrt{\frac{s_d(m^*)}{s_d(m)}} - \frac{2 \times 108\sqrt{\pi}K\alpha}{K_{\text{pen}}} n^{\rho - \frac{1}{2}}\right). \\ &\ge \sqrt{s_d(m)} n^{-\rho} \frac{K_{\text{pen}}}{2}\left(1 - \sqrt{\frac{s_d(m^*)}{s_d(m^*+1)}} - \frac{216\sqrt{\pi}K\alpha}{K_{\text{pen}}} n^{\rho - \frac{1}{2}}\right).\end{aligned}$$

Let $n_1 \in \mathbb{N}$ be such that

$$\begin{aligned}&1 - \sqrt{\frac{s_d(m^*)}{s_d(m^*+1)}} - \frac{216\sqrt{\pi}K\alpha}{K_{\text{pen}}} n_1^{\rho - \frac{1}{2}} > \frac{1}{2}\left(1 - \sqrt{\frac{s_d(m^*)}{s_d(m^*+1)}}\right) \\ \Leftrightarrow \quad &n_1 > \left(\frac{432\sqrt{\pi}K\alpha\sqrt{s_d(m^*+1)}}{K_{\text{pen}}(\sqrt{s_d(m^*+1)} - \sqrt{s_d(m^*)})}\right)^{1/(\frac{1}{2}-\rho)},\end{aligned}$$

then, for any $n \geq n_1$,

$$\frac{u_{m,n}}{2} - 108K\alpha\sqrt{\frac{\pi s_d(m)}{n}} \geq \sqrt{s_d(m)}n^{-\rho}\frac{K_{\text{pen}}}{4}\left(1 - \sqrt{\frac{s_d(m^*)}{s_d(m^*+1)}}\right) > 0.$$

Hence, Proposition C.4 applied to $x = \frac{u_{m,n}}{2} - 108\sqrt{\pi}K\alpha\sqrt{\frac{s_d(m)}{n}}$ now shows that, for $n \geq n_1$,

$$\begin{aligned}
&\mathbb{P}\Big(\sup_{\beta\in B_{m,\alpha}} Z_{m,n}(\beta) > \frac{u_{m,n}}{2} + Z_{m,n}(\beta_0)\Big) \\
&\quad\leq 36\exp\Big(-\frac{n}{144K^2\alpha^2}\Big(\frac{u_{m,n}}{2} - 108K\alpha\sqrt{\frac{\pi s_d(m)}{n}}\Big)^2\Big) \\
&\quad\leq 36\exp\Big(-\frac{s_d(m)n^{1-2\rho}K_{\text{pen}}^2}{144K^2\alpha^2\times 16}\Big(1 - \sqrt{\frac{s_d(m^*)}{s_d(m^*+1)}}\Big)^2\Big) \\
&\quad= 36\exp\Big(-\kappa_1 s_d(m)n^{1-2\rho}\Big), \qquad \text{(C.5)}
\end{aligned}$$

where

$$\kappa_1 = \frac{K_{\text{pen}}^2}{2304K^2\alpha^2}\Big(1 - \sqrt{\frac{s_d(m^*)}{s_d(m^*+1)}}\Big)^2.$$

We now turn to the second term of (C.4). Since

$$\big|Y - \langle\beta_0, S^m(X)\rangle\big|^2 \leq \big(K_Y + \|\beta_0\|e^{K_X}\big)^2 \quad \text{a.s. ,}$$

Hoeffding's inequality yields, for $n \geq n_1$,

$$\begin{aligned}
\mathbb{P}\Big(Z_{m,n}(\beta_0) > \frac{u_{m,n}}{2}\Big) &\leq \exp\Big(-\frac{nu_{m,n}^2}{8\big(K_Y + \|\beta_0\|e^{K_X}\big)^2}\Big) \\
&= \exp\Big(-\frac{n^{1-2\rho}K_{\text{pen}}^2\Big(\sqrt{s_d(m)} - \sqrt{s_d(m^*)}\Big)^2}{32\big(K_Y + \|\beta_0\|e^{K_X}\big)^2}\Big) \\
&\leq \exp\Big(-\frac{n^{1-2\rho}K_{\text{pen}}^2 s_d(m)}{32\big(K_Y + \|\beta_0\|e^{K_X}\big)^2}\Big(1 - \sqrt{\frac{s_d(m^*)}{s_d(m^*+1)}}\Big)^2\Big) \\
&= \exp\big(-\kappa_2 n^{1-2\rho}s_d(m)\big), \qquad \text{(C.6)}
\end{aligned}$$

where

$$\kappa_2 = \frac{K_{\text{pen}}^2}{32\big(K_Y + \|\beta_0\|e^{K_X}\big)^2}\Big(1 - \sqrt{\frac{s_d(m^*)}{s_d(m^*+1)}}\Big)^2.$$

Combining (C.5) with (C.6), we obtain

$$\begin{aligned}
\mathbb{P}\Big(\sup_{\beta\in B_{m,\alpha}} Z_{m,n}(\beta) > u_{m,n}\Big) &\leq 36\exp\big(-\kappa_1 n^{1-2\rho}s_d(m)\big) + \exp\big(-\kappa_2 n^{1-2\rho}s_d(m)\big) \\
&\leq 37\exp\big(-\kappa_3 n^{1-2\rho}s_d(m)\big) \leq 37\exp\Big(-\frac{\kappa_3}{2}\big(n^{1-2\rho} + s_d(m)\big)\Big),
\end{aligned}$$

where $\kappa_3 = \min(\kappa_1, \kappa_2)$. The same proof works for the process $(-Z_{m,n}(\beta))$, and consequently

$$\mathbb{P}\left(\widehat{m} = m\right) \leq 2 \times 37 \exp\Big(- \frac{\kappa_3}{2}\big(n^{1-2\rho} + s_d(m)\big)\Big).$$

We are left with the task of choosing an optimal β_0. Since

$$\kappa_3 = \min(\kappa_1, \kappa_2) = \frac{K_{\text{pen}}^2}{32}\left(1 - \sqrt{\frac{s_d(m^*)}{s_d(m^*+1)}}\right)^2 \min\left(\frac{1}{72K^2\alpha^2}, \frac{1}{\big(K_Y + \|\beta_0\| e^{K_X}\big)^2}\right),$$

it is clear that κ_3 is maximal at $\beta_0 = 0$, which yields

$$\kappa_3 = \frac{K_{\text{pen}}^2}{32}\left(1 - \sqrt{\frac{s_d(m^*)}{s_d(m^*+1)}}\right)^2 \min\left(\frac{1}{72K^2\alpha^2}, \frac{1}{K_Y^2}\right).$$

Noting that

$$\sqrt{s_d(m^*+1)} - \sqrt{s_d(m^*)} = \sqrt{d^{m^*+1} + s_d(m^*)} - \sqrt{s_d(m^*)} \geq \sqrt{\frac{d^{m^*+1}}{2}},$$

where we have used the fact that for $a, b \geq 0$, $\sqrt{a} + \sqrt{b} \geq \sqrt{2}\sqrt{a+b}$, letting

$$C_3 = \frac{1}{2} \times \frac{K_{\text{pen}}^2 d^{m^*+1}}{64 s_d(m^*+1)(72K^2\alpha^2 + K_Y^2)}$$

completes the proof. □

To treat the case $m < m^*$, we need a rate of convergence of $\widehat{L}_n$. This can be obtained with arguments similar to the previous proof.

Proposition C.6. *For any $\varepsilon > 0$, $m \in \mathbb{N}$, let $n_2 \in \mathbb{N}$ be the smallest integer such that*

$$n_2 \geq \frac{432^2 K^2 \pi \alpha^2 s_d(m)}{\varepsilon^2}. \tag{C.7}$$

Then, for any $n \geq n_2$,

$$\mathbb{P}\big(\big|\widehat{L}_n(m) - L(m)\big| > \varepsilon\big) \leq 74 \exp\big(-C_4 n \varepsilon^2\big),$$

where the constant C_4 is defined by

$$C_4 = \frac{1}{2(1152K^2\alpha^2 + K_Y^2)}. \tag{C.8}$$

Proof. By Lemma C.1,

$$\begin{aligned}\mathbb{P}\big(|\widehat{L}_n(m) - L(m)| > \varepsilon\big) &\leq \mathbb{P}\Big(\sup_{\beta \in B_{m,\alpha}} |Z_{m,n}(\beta)| > \varepsilon\Big) \\ &= \mathbb{P}\Big(\sup_{\beta \in B_{m,\alpha}} Z_{m,n}(\beta) > \varepsilon\Big) + \mathbb{P}\Big(\sup_{\beta \in B_{m,\alpha}} (-Z_{m,n}(\beta)) > \varepsilon\Big).\end{aligned}$$

Let us fix $\beta_0 \in B_{m,\alpha}$, we can now proceed as in Proposition C.5. Since, for $n \geq n_2$,

$$\frac{\varepsilon}{2} - 108K\alpha\sqrt{\frac{\pi s_d(m)}{n}} > \frac{\varepsilon}{4} > 0,$$

Hoeffing's inequality and Proposition C.4 show that

$$\begin{aligned}
\mathbb{P}\Big(\sup_{\beta\in B_{m,\alpha}} Z_{m,n}(\beta) > \varepsilon\Big) &\leq \mathbb{P}\Big(\sup_{\beta\in B_{m,\alpha}} Z_{m,n}(\beta) > \frac{\varepsilon}{2} + Z_{m,n}(\beta_0)\Big) + \mathbb{P}\Big(Z_{m,n}(\beta_0) > \frac{\varepsilon}{2}\Big) \\
&\leq 36\exp\Big(-\frac{n}{144K^2\alpha^2}\Big(\frac{\varepsilon}{2} - 108K\alpha\sqrt{\frac{\pi s_d(m)}{n}}\Big)^2\Big) \\
&\quad + \exp\Big(-\frac{n\varepsilon^2}{2\big(K_Y + \|\beta_0\|e^{K_X}\big)^2}\Big) \\
&\leq 36\exp\Big(-\frac{n\varepsilon^2}{2304K^2\alpha^2}\Big) + \exp\Big(-\frac{n\varepsilon^2}{2\big(K_Y + \|\beta_0\|e^{K_X}\big)^2}\Big) \\
&\leq 37\exp\left(-\kappa_4 n\varepsilon^2\right),
\end{aligned}$$

where

$$\kappa_4 = \min\left(\frac{1}{2304K^2\alpha^2}, \frac{1}{2\big(K_Y + \|\beta_0\|e^{K_X}\big)^2}\right).$$

The same analysis can be done to $(-Z_{m,n}(\beta))$, and so

$$\mathbb{P}\left(|\widehat{L}_n(m) - L(m)| > \varepsilon\right) \leq 74\exp\left(-\kappa_4 n\varepsilon^2\right).$$

Moreover, taking $\beta_0 = 0$ gives

$$\kappa_4 = \min\left(\frac{1}{2304K^2\alpha^2}, \frac{1}{2\big(K_Y + \|\beta_0\|e^{K_X}\big)^2}\right) \geq \frac{1}{2(1152K^2\alpha^2 + K_Y^2)} = C_4,$$

which completes the proof. □

This allows us to treat the case $m < m^*$.

Proposition C.7. *Let $0 < \rho < \frac{1}{2}$ and $\mathrm{pen}_n(m)$ be defined by (4.7). Let n_3 be the smallest integer satisfying*

$$n_3 \geq \left(\frac{2\sqrt{s_d(m^*)}}{L(m^*-1) - \sigma^2}\big(432K\alpha\sqrt{\pi} + K_{\mathrm{pen}}\big)\right)^{1/\rho}. \tag{C.9}$$

Then, under the assumptions (H_α) and (H_K), for any $m < m^*$, $n \geq n_3$,

$$\mathbb{P}\left(\widehat{m} = m\right) \leq 148\exp\Big(-n\frac{C_4}{4}\big(L(m) - L(m^*) - \mathrm{pen}_n(m^*) + \mathrm{pen}_n(m)\big)^2\Big),$$

where the constant C_4 is defined by (C.8).

Proof. This is a consequence of Proposition C.6. For any $m < m^*$,

$$\begin{aligned}
\mathbb{P}(\widehat{m} = m) &\leq \mathbb{P}\left(\widehat{L}_n(m) - \widehat{L}_n(m^*) \leq \mathrm{pen}_n(m^*) - \mathrm{pen}_n(m)\right) \\
&= \mathbb{P}\Big(\widehat{L}_n(m^*) - L(m^*) + L(m) - \widehat{L}_n(m) \geq L(m) - L(m^*) - \big(\mathrm{pen}_n(m^*) - \mathrm{pen}_n(m)\big)\Big) \\
&\leq \mathbb{P}\Big(\big|\widehat{L}_n(m) - L(m)\big| \geq \frac{1}{2}\big(L(m) - L(m^*) - \mathrm{pen}_n(m^*) + \mathrm{pen}_n(m)\big)\Big) \\
&\quad + \mathbb{P}\Big(\big|\widehat{L}_n(m^*) - L(m^*)\big| \geq \frac{1}{2}\big(L(m) - L(m^*) - \mathrm{pen}_n(m^*) + \mathrm{pen}_n(m)\big)\Big).
\end{aligned}$$

In order to apply Proposition C.6, we first need to ensure that $L(m) - L(m^*) - \mathrm{pen}_n(m^*) + \mathrm{pen}_n(m)$ is strictly positive. Recall that $m \mapsto L(m)$ is a decreasing function, minimal at $m = m^*$ and then bounded by σ^2. Recall also that $m \mapsto \mathrm{pen}_n(m)$ is strictly increasing. This gives, for $m < m^*$:

$$L(m) - L(m^*) - \mathrm{pen}_n(m^*) + \mathrm{pen}_n(m) > L(m^* - 1) - \sigma^2 - K_{\mathrm{pen}} n^{-\rho} \sqrt{s_d(m^*)}.$$

This implies that it is enough that

$$L(m^* - 1) - \sigma^2 - K_{\mathrm{pen}} n^{-\rho} \sqrt{s_d(m^*)} > \frac{1}{2}(L(m^* - 1) - \sigma^2) \tag{C.10}$$

to ensure that $L(m) - L(m^*) - \mathrm{pen}_n(m^*) + \mathrm{pen}_n(m) > 0$. This yields a first condition on n_3:

$$n_3 \geq \left(\frac{2K_{\mathrm{pen}}\sqrt{s_d(m^*)}}{L(m^* - 1) - \sigma^2}\right)^{\frac{1}{\rho}}. \tag{C.11}$$

However, to apply Proposition C.6, we also need n_3 to satisfy (C.7) , which writes

$$n_3 \geq \frac{432^2 K^2 \pi \alpha^2 s_d(m)}{(L(m) - L(m^*) - \mathrm{pen}_n(m^*) + \mathrm{pen}_n(m))^2}.$$

If n_3 satisfies (C.11), we can bound the right-hand side uniformly in m:

$$\begin{aligned}
\frac{432^2 K^2 \pi \alpha^2 s_d(m)}{\big(L(m) - L(m^*) - \mathrm{pen}_n(m^*) + \mathrm{pen}_n(m)\big)^2} &\leq \frac{4 \times 432^2 K^2 \pi \alpha^2 s_d(m^*)}{(L(m^* - 1) - \sigma^2)^2} \\
&= \left(\frac{2 \times 432 K \alpha \sqrt{\pi s_d(m^*)}}{L(m^* - 1) - \sigma^2}\right)^2.
\end{aligned}$$

We can assume that this quantity is larger than 1, as otherwise the condition on n_3 will be trivially satisfied. Then, as $\rho < \frac{1}{2}$, it is enough for n_3 to satisfy

$$n_3 \geq \max\left(\frac{2K_{\mathrm{pen}}\sqrt{s_d(m^*)}}{L(m^* - 1) - \sigma^2}, \frac{2 \times 432 K \alpha \sqrt{\pi s_d(m^*)}}{L(m^* - 1) - \sigma^2}\right)^{1/\rho},$$

or in a more compact form that

$$n_3 \geq \left(\frac{2(K_{\mathrm{pen}} + 432 K \alpha \sqrt{\pi})\sqrt{s_d(m^*)}}{L(m^* - 1) - \sigma^2}\right)^{1/\rho}.$$

We conclude by applying Proposition C.6 to both terms with

$$\varepsilon = \frac{1}{2}\big(L(m) - L(m^*) - \mathrm{pen}_n(m^*) - \mathrm{pen}_n(m)\big)$$

. □

We are now in a position to prove Theorem 4.4.

Proof of Theorem 4.4. The result is a consequence of Propositions C.5 and C.7. For this, we first need to ensure that the conditions on n (C.3) and (C.9) are satisfied. Thus, we need to bound

$$M = \max\left(\left(\frac{2\sqrt{s_d(m^*)}}{L(m^*-1)-\sigma^2}\big(432K\alpha\sqrt{\pi}+K_{\mathrm{pen}}\big)\right)^{1/\rho}, \left(\frac{432\sqrt{\pi}K\alpha\sqrt{s_d(m^*+1)}}{K_{\mathrm{pen}}(\sqrt{s_d(m^*+1)}-\sqrt{s_d(m^*)})}\right)^{1/(\frac{1}{2}-\rho)}\right).$$

If $\tilde{\rho} = \min(\rho, \frac{1}{2}-\rho)$, then

$$\begin{aligned} M &\leq \left((432K\alpha\sqrt{\pi}+K_{\mathrm{pen}})\sqrt{s_d(m^*+1)}\times \right. \\ &\qquad \left.\max\left(\frac{2}{L(m^*-1)-\sigma^2}, \frac{1}{K_{\mathrm{pen}}\big(\sqrt{s_d(m^*+1)}-\sqrt{s_d(m^*)}\big)}\right)\right)^{1/\tilde{\rho}} \\ &\leq \left((432K\alpha\sqrt{\pi}+K_{\mathrm{pen}})\sqrt{s_d(m^*+1)}\Big(\frac{2}{L(m^*-1)-\sigma^2}+\frac{\sqrt{2}}{K_{\mathrm{pen}}\sqrt{d^{m^*+1}}}\Big)\right)^{1/\tilde{\rho}}. \end{aligned}$$

Therefore, condition (4.8) implies that (C.3) and (C.9) are satisfied. Splitting the probability $\mathbb{P}(\widehat{m} \neq m^*)$ into two terms now gives

$$\mathbb{P}(\widehat{m} \neq m^*) = \mathbb{P}(\widehat{m} > m^*) + \mathbb{P}(\widehat{m} < m^*) \leq \sum_{m>m^*} \mathbb{P}(\widehat{m} = m) + \sum_{m<m^*} \mathbb{P}(\widehat{m} = m).$$

On the one hand, Theorem C.5 shows that, for $n \geq n_0$,

$$\sum_{m>m^*} \mathbb{P}(\widehat{m} = m) \leq 74 e^{-C_3 n^{1-2\rho}} \sum_{m>m^*} e^{-C_3 s_d(m)},$$

and, on the other hand, Proposition C.7 gives

$$\begin{aligned} \sum_{m<m^*} \mathbb{P}(\widehat{m} = m) &\leq 148 \sum_{m=0}^{m^*-1} \exp\Big(-\frac{C_4}{4} n\big(L(m) - L(m^*) - \mathrm{pen}_n(m^*) + \mathrm{pen}_n(m)\big)\Big) \\ &\leq 148 m^* \exp\big(-\frac{C_4}{8} n(L(m^*-1)-\sigma^2)\big), \end{aligned}$$

where we have used that for $n \geq n_0$, (C.10) is true. Letting

$$\kappa_5 = \min\Big(C_3, \frac{C_4(L(m^*-1)-\sigma^2)}{8}\Big)$$

yields

$$\mathbb{P}\left(\widehat{m} \neq m^*\right) \leq 74e^{-\kappa_5 n^{1-2\rho}} \sum_{m>0} e^{-C_3 s_d(m)} + 148 m^* e^{-\kappa_5 n} \leq C_1 e^{-\kappa_5 n^{1-2\rho}},$$

where

$$C_1 = 74 \sum_{m>0} e^{-C_3 s_d(m)} + 148 m^*.$$

To complete the proof, it remains to find a lower bound on κ_5:

$$\begin{aligned}
\kappa_5 &= \min\left(C_3, \frac{C_4(L(m^*-1)-\sigma^2)}{8}\right) \\
&= \min\left(\frac{K_{\text{pen}}^2 d^{m^*+1}}{128 s_d(m^*+1)(72K^2\alpha^2 + K_Y^2)}, \frac{L(m^*-1)-\sigma^2}{16(1152K^2\alpha^2+K_Y^2)}\right) \\
&\geq \frac{1}{16(1152K^2\alpha^2+K_Y^2)} \min\left(\frac{K_{\text{pen}}^2 d^{m^*+1}}{8 s_d(m^*+1)}, L(m^*-1)-\sigma^2\right) = C_2.
\end{aligned}$$

□

C.2 Proof of Corollary 4.5

First, let us note that

$$\mathbb{E}\big(\langle \widehat{\beta}_{\widehat{m}}, S^{\widehat{m}}(X)\rangle - \langle \beta^*_{m^*}, S^{m^*}(X)\rangle\big)^2 = \mathbb{E}\big(\mathcal{R}_{\widehat{m}}(\widehat{\beta}_{\widehat{m}})\big) - \mathcal{R}_{m^*}(\beta^*_{m^*}).$$

Moreover, we have a.s.

$$\begin{aligned}
&\mathcal{R}_{\widehat{m}}(\widehat{\beta}_{\widehat{m}}) - \mathcal{R}_{m^*}(\beta^*_{m^*}) \\
&\quad = \mathcal{R}_{\widehat{m}}(\widehat{\beta}_{\widehat{m}}) - \mathcal{R}_{\widehat{m}}(\beta^*_{\widehat{m}}) + \mathcal{R}_{\widehat{m}}(\beta^*_{\widehat{m}}) - \mathcal{R}_{m^*}(\beta^*_{m^*}) \\
&\quad = \mathcal{R}_{\widehat{m}}(\widehat{\beta}_{\widehat{m}}) - \widehat{\mathcal{R}}_{\widehat{m},n}(\widehat{\beta}_{\widehat{m}}) + \widehat{\mathcal{R}}_{\widehat{m},n}(\widehat{\beta}_{\widehat{m}}) - \widehat{\mathcal{R}}_{\widehat{m},n}(\beta^*_{\widehat{m}}) \\
&\qquad + \widehat{\mathcal{R}}_{\widehat{m},n}(\beta^*_{\widehat{m}}) - \mathcal{R}_{\widehat{m}}(\beta^*_{\widehat{m}}) + \mathcal{R}_{\widehat{m}}(\beta^*_{\widehat{m}}) - \mathcal{R}_{m^*}(\beta^*_{m^*}) \\
&\quad \leq \mathcal{R}_{\widehat{m}}(\widehat{\beta}_{\widehat{m}}) - \widehat{\mathcal{R}}_{\widehat{m},n}(\widehat{\beta}_{\widehat{m}}) + \widehat{\mathcal{R}}_{\widehat{m},n}(\beta^*_{\widehat{m}}) - \mathcal{R}_{\widehat{m}}(\beta^*_{\widehat{m}}) + \mathcal{R}_{\widehat{m}}(\beta^*_{\widehat{m}}) - \mathcal{R}_{m^*}(\beta^*_{m^*}) \\
&\quad \leq 2 \sup_{\beta \in B_{\widehat{m},\alpha}} |\widehat{\mathcal{R}}_{\widehat{m},n}(\beta) - \mathcal{R}_{\widehat{m}}(\beta)| + \mathcal{R}_{\widehat{m}}(\beta^*_{\widehat{m}}) - \mathcal{R}_{m^*}(\beta^*_{m^*})
\end{aligned}$$

We decompose the proof into two lemmas.

Lemma C.8.

$$\mathbb{E}\Big(\sup_{\beta \in B_{\widehat{m},\alpha}} |\widehat{\mathcal{R}}_{\widehat{m},n}(\beta) - \mathcal{R}_{\widehat{m}}(\beta)|\Big) = O\Big(\frac{1}{\sqrt{n}}\Big).$$

Proof. From Corollary 5.25 of van Handel (2014) and (C.2), for any $m \in \mathbb{N}$,

$$\begin{aligned}
\mathbb{E}\Big(\sup_{\beta \in B_{m,\alpha}} |\widehat{\mathcal{R}}_{m,n}(\beta) - \mathcal{R}_m(\beta)|\Big) &\leq 12 \int_0^\infty \sqrt{\log(N(B_{m,\alpha}, D, \varepsilon))} \\
&= 36 K\alpha \sqrt{s_d(m)} \sqrt{\frac{\pi}{n}},
\end{aligned}$$

where $N(B_{m,\alpha}, D, \varepsilon)$ is the ε-covering number of $B_{m,\alpha}$ with respect to the distance D, defined by (C.1). This gives, for $m = \widehat{m}$,

$$\mathbb{E}\Big(\sup_{\beta \in B_{\widehat{m},\alpha}} |\widehat{\mathcal{R}}_{\widehat{m},n}(\beta) - \mathcal{R}_{\widehat{m}}(\beta)| \Big) \leq 36 K \alpha \sqrt{\frac{\pi}{n}} \mathbb{E}\Big(\sqrt{s_d(\widehat{m})} \Big).$$

To compute this expectation, Proposition C.5 yields

$$\begin{aligned}
\mathbb{E}\Big(\sqrt{s_d(\widehat{m})} \Big) &= \sum_{m \leq m^*} \sqrt{s_d(m)} \mathbb{P}(\widehat{m} = m) + \sum_{m > m^*} \sqrt{s_d(m)} \mathbb{P}(\widehat{m} = m) \\
&\leq (m^* + 1) \sqrt{s_d(m^*)} + \sum_{m > m^*} \sqrt{s_d(m)} 74 \exp\big(- C_3 (n^{1-2\rho} + s_d(m)) \big) \\
&\leq (m^* + 1) \sqrt{s_d(m^*)} + e^{-C_3 n^{1-2\rho}} \sum_{m > m^*} \sqrt{s_d(m)} 74 \exp\big(- C_3 s_d(m) \big) \\
&= O(1),
\end{aligned}$$

which completes the proof. □

Lemma C.9.

$$\mathbb{E}\big(\mathcal{R}_{\widehat{m}}(\beta^*_{\widehat{m}}) - \mathcal{R}_{m^*}(\beta^*_{m^*}) \big) = O(e^{-C_2 n^{1-2\rho}}),$$

where the constant C_2 is defined by (4.10).

Proof. Since, for any $m \in \mathbb{N}$,

$$\langle \beta^*_m, S^m(X) \rangle^2 \leq \|\beta^*_k\|_2^2 \|S^m(X)\|_2^2 \leq \alpha^2 e^{K_X},$$

it follows that

$$\begin{aligned}
\mathbb{E}\big(\mathcal{R}_{\widehat{m}}(\beta^*_{\widehat{m}}) - \mathcal{R}_{m^*}(\beta^*_{m^*}) \big) &= \mathbb{E}\Big(\big(Y - \langle \beta^*_{\widehat{m}}, S^{\widehat{m}}(X) \rangle \big)^2 - \big(Y - \langle \beta^*_{m^*}, S^{m^*}(X) \rangle \big)^2 \Big) \\
&= \mathbb{E}\Big(\big(\langle \beta^*_{m^*}, S^{m^*}(X) \rangle + \varepsilon - \langle \beta^*_{\widehat{m}}, S^{\widehat{m}}(X) \rangle \big)^2 - \varepsilon^2 \Big) \\
&= \mathbb{E}\Big(\big(\langle \beta^*_{m^*}, S^{m^*}(X) \rangle - \langle \beta^*_{\widehat{m}}, S^{\widehat{m}}(X) \rangle \big)^2 \Big) \\
&\leq 2\alpha^2 e^{K_X} \mathbb{P}(\widehat{m} \neq m^*).
\end{aligned}$$

By Theorem 4.4, this yields

$$\mathbb{E}\big(\mathcal{R}_{\widehat{m}}(\beta^*_{\widehat{m}}) - \mathcal{R}_{m^*}(\beta^*_{m^*}) \big) \leq 2\alpha^2 e^{K_X} C_1 e^{-C_2 n^{1-2\rho}} = O(e^{-C_2 n^{1-2\rho}}),$$

where C_1 and C_2 are defined by (4.9) and (4.10).

□

Combining these lemmas, we conclude that

$$\mathbb{E}\big(\langle \widehat{\beta}_{\widehat{m}}, S^{\widehat{m}}(X) \rangle - \langle \beta^*_{m^*}, S^{m^*}(X) \rangle \big)^2 = O\Big(\frac{1}{\sqrt{n}} \Big) + O(e^{-C_2 n^{1-2\rho}}) = O\Big(\frac{1}{\sqrt{n}} \Big).$$

Bibliography

van Handel, R. (2014). *Probability in high dimension* (tech. rep.). Princeton University.

Appendix D

Supplementary material of Chapter 5

Contents

D.1 Mathematical details

D.1.1 Writing the GRU and LSTM in the neural ODE framework

GRU. Recall that the equations of a GRU take the following form: for any $1 \leq j \leq T$,

$$\begin{aligned} r_{j+1} &= \sigma(W_r x_{j+1} + b_r + U_r h_j) \\ z_{j+1} &= \sigma(W_z x_{j+1} + b_z + U_z h_j) \\ n_{j+1} &= \tanh\big(W_n x_{j+1} + b_n + r_{j+1} * (U_n h_j + c_n)\big) \\ h_{j+1} &= (1 - z_{j+1}) * h_j + z_{j+1} * n_{j+1}, \end{aligned}$$

where σ is the logistic activation, tanh the hyperbolic tangent, $*$ the Hadamard product, r_j the reset gate vector, z_j the update gate vector, W_r, U_r, W_z, U_z, W_n, U_n weight matrices, and b_r, b_z, b_n, c_n biases. Since r_{j+1}, z_{j+1}, and n_{j+1} depend only on x_{j+1} and h_j, it is clear that these equations can be rewritten in the form

$$h_{j+1} = h_j + f(h_j, x_{j+1}).$$

We then obtain equation (5.1) by normalizing f by $1/T$.

LSTM. The LSTM networks are defined, for any $1 \leq j \leq T$, by

$$\begin{aligned} i_{j+1} &= \sigma(W_i x_{j+1} + b_i + U_i h_j) \\ f_{j+1} &= \sigma(W_f x_{j+1} + b_f + U_f h_j) \\ g_{j+1} &= \tanh(W_g x_{j+1} + b_g + U_g h_j) \\ o_{j+1} &= \sigma(W_o x_{j+1} + b_o + U_o h_j) \\ c_{j+1} &= f_{j+1} * c_j + i_{j+1} * g_{j+1} \\ h_{j+1} &= o_{j+1} * \tanh(c_{j+1}), \end{aligned}$$

where σ is the logistic activation, tanh the hyperbolic tangent, $*$ the Hadamard product, i_j the input gate, f_j the forget gate, g_j the cell gate, o_j the output gate, c_j the cell state, W_i, U_i, W_f, U_f, W_g, U_g W_o, U_o weight matrices, and b_i, b_f, b_g, b_o biases. Since i_{j+1}, f_{j+1}, g_{j+1}, o_{j+1} depend only on x_{j+1} and h_j, these equations can be rewritten in the form

$$\begin{aligned} h_{j+1} &= f_1(h_j, x_{j+1}, c_{j+1}) \\ c_{j+1} &= f_2(h_j, x_{j+1}, c_j). \end{aligned}$$

Let $\tilde{h}_j = (h_j^\top, c_j^\top)^\top$ be the hidden state defined by stacking the hidden and cell state. Then, clearly, $\tilde{h}$ follows an equation of the form

$$\tilde{h}_{j+1} = f(\tilde{h}_j, x_{j+1}).$$

We obtain (5.1) by subtracting $\tilde{h}_j$ and normalizing by $1/T$.

D.1.2 Picard-Lindelöf theorem

Consider a CDE of the form (5.8). We recall the Picard-Lindelöf theorem as given by Lyons et al. (2007, Theorem 1.3), and provide a proof for the sake of completeness.

Theorem D.1 (Picard-Lindelöf theorem)**.** *Assume that $X \in BV([0,1], \mathbb{R}^d)$ and that $\mathbf{F}$ is Lipschitz-continuous with constant $K_{\mathbf{F}}$. Then, for any $H_0 \in \mathbb{R}^e$, the differential equation* (5.8)

admits a unique solution $H : [0,1] \to \mathbb{R}^e$.

Proof. Let $\mathscr{C}([s,t]), \mathbb{R}^e)$ be the set of continuous functions from $[s,t]$ to $\mathbb{R}^e$. For any $[s,t] \subset [0,1]$, $\zeta \in \mathbb{R}^e$, let Ψ be the function

$$\begin{aligned}\Psi : \mathscr{C}([s,t]), \mathbb{R}^e) &\to \mathscr{C}([s,t], \mathbb{R}^e) \\ Y &\mapsto \big(v \mapsto \zeta + \int_s^v \mathbf{F}(Y_u) dX_u\big).\end{aligned}$$

For any $Y, Y' \in \mathscr{C}([s,t]), \mathbb{R}^e)$, $v \in [s,t]$,

$$\begin{aligned}\|\Psi(Y)_v - \Psi(Y')_v\| &\leq \int_s^v \big\|\big(\mathbf{F}(Y_u) - \mathbf{F}(Y'_u)\big) dX_u\big\| \\ &\leq \int_s^v \|\mathbf{F}(Y_u) - \mathbf{F}(Y'_u)\|_{\mathrm{op}} \|dX_u\| \\ &\leq \int_s^v K_{\mathbf{F}} \|Y_u - Y'_u\| \|dX_u\| \\ &\leq K_{\mathbf{F}} \|Y - Y'\|_\infty \int_s^v \|dX_u\| \\ &\leq K_{\mathbf{F}} \|Y - Y'\|_\infty \|X\|_{TV;[s,t]}.\end{aligned}$$

This shows that the function Ψ is Lipschitz-continuous on $\mathscr{C}([s,t]), \mathbb{R}^e)$ endowed with the supremum norm, with Lipschitz constant $K_{\mathbf{F}} \|X\|_{TV;[s,t]}$. Clearly, the function $t \mapsto \|X\|_{TV;[0,t]}$ is non-decreasing and uniformly continuous on the compact interval $[0,1]$. Therefore, for any $\varepsilon > 0$, there exists $\delta > 0$ such that

$$|t - s| < \delta \Rightarrow \big|\|X\|_{TV;[0,t]} - \|X\|_{TV;[0,s]}\big| < \varepsilon.$$

Take $\varepsilon = 1/K_{\mathbf{F}}$. Then on any interval $[s,t]$ of length smaller than δ, one has $\|X\|_{TV;[s,t]} = \|X\|_{TV;[0,t]} - \|X\|_{TV;[0,s]} < 1/K_{\mathbf{F}}$, so that the function Ψ is a contraction. By the Banach fixed-point theorem, for any initial value ζ, Ψ has a unique fixed point. Hence, there exists a solution to (5.8) on any interval of length δ with any initial condition. To obtain a solution on $[0,1]$ it is sufficient to concatenate these solutions. □

A corollary of this theorem is a Picard-Lindelöf theorem for initial value problems of the form

$$dH_t = f(H_t, X_t)dt, \quad H_0 = \zeta, \tag{D.1}$$

where $f : \mathbb{R}^e \times \mathbb{R}^d \to \mathbb{R}^e$, $\zeta \in \mathbb{R}^e$.

Corollary D.2. *Assume that f is Lipschitz continuous in its first variable. Then, for any $\zeta \in \mathbb{R}^e$, the initial value problem (D.1) admits a unique solution.*

Proof. Let $f_X : (h,t) \mapsto f(h, X_t)$. Then the solution of (D.1) is solution of the differential equation

$$dH_t = f_X(H_t, t)dt.$$

Let $d = 1$, $\bar{e} = e + 1$, and $\mathbf{F}$ be the vector field defined by

$$\mathbf{F} : h \mapsto \begin{pmatrix} f_X(h^{1:e}, h^{e+1}) \\ 1 \end{pmatrix},$$

where $h^{1:e}$ denotes the projection of h on its first e coordinates. Then, since f_X is Lipschitz, so is the vector field $\mathbf{F}$. Theorem D.1 therefore applies to the differential equation

$$dH_t = \mathbf{F}(H_t)dt, \quad H_0 = (\zeta^\top, 0)^\top.$$

Projecting this differential equation on the last coordinate gives $dH_t^{e+1} = dt$, that is, $H_t^{e+1} = t$. Projecting on the first e coordinates exactly provides equation (D.1), which therefore has a unique solution, equal to $H^{1:e}$. □

D.1.3 Operator norm

Definition D.1. *Let $(E, \|\cdot\|_E)$ and $(F, \|\cdot\|_F)$ be two normed vector spaces and let $f \in \mathscr{L}(E, F)$, where $\mathscr{L}(E, F)$ is the space of linear functions from E to F. The operator norm of f is defined by*

$$\|f\|_{\mathrm{op}} = \sup_{u \in E, \|u\|_E = 1} \|f(u)\|_F.$$

Equipped with this norm, $\mathscr{L}(E, F)$ is a normed vector space.

This definition is valid when f is represented by a matrix.

D.1.4 Tensor Hilbert space

Let us first briefly recall some elements on tensor spaces. If $e_1, \dots, e_d$ is the canonical basis of $\mathbb{R}^d$, then $(e_{i_1} \otimes \cdots \otimes e_{i_k})_{1 \le i_1, \dots, i_k \le d}$ is a basis of $(\mathbb{R}^d)^{\otimes k}$. Any element $a \in (\mathbb{R}^d)^{\otimes k}$ can therefore be written as

$$a = \sum_{1 \le i_1, \dots, i_k \le d} a^{(i_1, \dots, i_k)} e_{i_1} \otimes \cdots \otimes e_{i_k},$$

where $a^{(i_1, \dots, i_k)} \in \mathbb{R}$. The tensor space $(\mathbb{R}^d)^{\otimes k}$ is a Hilbert space of dimension d^k, with scalar product

$$\langle a, b \rangle_{(\mathbb{R}^d)^{\otimes k}} = \sum_{1 \le i_1, \dots, i_k \le d} a^{(i_1, \dots, i_k)} b^{(i_1, \dots, i_k)}$$

and associated norm $\|\cdot\|_{(\mathbb{R}^d)^{\otimes k}}$.

We now consider the space $\mathscr{T}$ defined by (5.6). The sum, multiplication by a scalar, and scalar product on $\mathscr{T}$ are defined as follows: for any $a = (a_0, \dots, a_k, \dots) \in \mathscr{T}$, $b = (b_0, \dots, b_k, \dots) \in \mathscr{T}$, $\lambda \in \mathbb{R}$,

$$a + \lambda b = (a_0 + \lambda b_0, \dots, a_k + \lambda b_k, \dots) \quad \text{and} \quad \langle a, b \rangle_{\mathscr{T}} = \sum_{k=0}^{\infty} \langle a_k, b_k \rangle_{(\mathbb{R}^d)^{\otimes k}},$$

with the convention $(\mathbb{R}^d)^{\otimes 0} = \mathbb{R}$.

Proposition D.3. *$(\mathscr{T}, +, \cdot, \langle \cdot, \cdot \rangle_{\mathscr{T}})$ is a Hilbert space.*

Proof. By the Cauchy-Schwartz inequality, $\langle \cdot, \cdot \rangle_{\mathscr{T}}$ is well-defined: for any $a, b \in \mathscr{T}$,

$$\begin{aligned} |\langle a, b \rangle_{\mathscr{T}}| \le \sum_{k=0}^{\infty} |\langle a_k, b_k \rangle_{(\mathbb{R}^d)^{\otimes k}}| &\le \sum_{k=0}^{\infty} \|a_k\|_{(\mathbb{R}^d)^{\otimes k}} \|b_k\|_{(\mathbb{R}^d)^{\otimes k}} \\ &\le \Big(\sum_{k=0}^{\infty} \|a_k\|^2_{(\mathbb{R}^d)^{\otimes k}} \Big)^{1/2} \Big(\sum_{k=0}^{\infty} \|b_k\|^2_{(\mathbb{R}^d)^{\otimes k}} \Big)^{1/2} < \infty. \end{aligned}$$

Moreover, $\mathscr{T}$ is a vector space: for any $a, b \in \mathscr{T}$, $\lambda \in \mathbb{R}$, since

$$a + \lambda b = (a_0 + \lambda b_0, \dots, a_k + \lambda b_k, \dots),$$

and

$$\begin{aligned}\sum_{k=0}^{\infty} \|a_k + \lambda b_k\|^2_{(\mathbb{R}^d)^{\otimes k}} &= \sum_{k=0}^{\infty} \|a_k\|^2_{(\mathbb{R}^d)^{\otimes k}} + \lambda^2 \sum_{k=0}^{\infty} \|b_k\|^2_{(\mathbb{R}^d)^{\otimes k}} \\ &\quad + 2\lambda \sum_{k=0}^{\infty} \langle a_k, b_k \rangle_{(\mathbb{R}^d)^{\otimes k}} \\ &\leq \sum_{k=0}^{\infty} \|a_k\|^2_{(\mathbb{R}^d)^{\otimes k}} + \lambda^2 \sum_{k=0}^{\infty} \|b_k\|^2_{(\mathbb{R}^d)^{\otimes k}} + 2\lambda \langle a, b \rangle_{\mathscr{T}} < \infty,\end{aligned}$$

we see that $a + \lambda b \in \mathscr{T}$. The operation $\langle \cdot, \cdot \rangle_{\mathscr{T}}$ is also bilinear, symmetric, and positive definite:

$$\langle a, a \rangle_{\mathscr{T}} = 0 \Leftrightarrow \sum_{k=0}^{\infty} \|a_k\|^2_{(\mathbb{R}^d)^{\otimes k}} = 0 \Leftrightarrow \forall k \in \mathbb{N}, \|a_k\|^2_{(\mathbb{R}^d)^{\otimes k}} = 0 \Leftrightarrow \forall k \in \mathbb{N}, a_k = 0 \Leftrightarrow a = 0.$$

Therefore $\langle \cdot, \cdot \rangle_{\mathscr{T}}$ is an inner product on $\mathscr{T}$. Finally, let $(a^{(n)})_{n \in \mathbb{N}}$ be a Cauchy sequence in $\mathscr{T}$. Then, for any $n, m \geq 0$,

$$\|a^{(n)} - a^{(m)}\|^2_{\mathscr{T}} = \sum_{k=0}^{\infty} \|a_k^{(n)} - a_k^{(m)}\|^2_{(\mathbb{R}^d)^{\otimes k}},$$

so for any $k \in \mathbb{N}$, the sequence $(a_k^{(n)})_{n \in \mathbb{N}}$ is Cauchy in $(\mathbb{R}^d)^{\otimes k}$. Since $(\mathbb{R}^d)^{\otimes k}$ is a Hilbert space, $(a_k^{(n)})_{n \in \mathbb{N}}$ converges to a limit $a_k^{(\infty)} \in (\mathbb{R}^d)^{\otimes k}$. Let $a^{(\infty)} = (a_0^{(\infty)}, \dots, a_k^{(\infty)}, \dots)$. To finish the proof, we need to show that $a^{(\infty)} \in \mathscr{T}$ and that $a^{(n)}$ converges to $a^{(\infty)}$ in $\mathscr{T}$. First, note that there exists a constant $B > 0$ such that for any $n \in \mathbb{N}$,

$$\|a^{(n)}\|_{\mathscr{T}} \leq B.$$

To see this, observe that for $\varepsilon > 0$, there exists $N \in \mathbb{N}$ such that for any $n \geq N$, $\|a^{(n)} - a^{(N)}\|_{\mathscr{T}} < \varepsilon$, and so $\|a^{(n)}\|_{\mathscr{T}} \leq \varepsilon + \|a^{(N)}\|_{\mathscr{T}}$. Take $B = \max(\|a^{(1)}\|_{\mathscr{T}}, \dots, \|a^{(N)}\|_{\mathscr{T}}, \varepsilon + \|a^{(N)}\|_{\mathscr{T}})$. Then, for any $K \in \mathbb{N}$,

$$\sum_{k=0}^{K} \|a_k^{(n)}\|^2_{(\mathbb{R}^d)^{\otimes k}} \leq \|a^{(n)}\|_{\mathscr{T}} \leq B.$$

Letting $K \to \infty$, we obtain that $\|a^{(\infty)}\|_{\mathscr{T}} \leq B$, and therefore $a^{(\infty)} \in \mathscr{T}$. Finally, let $\varepsilon > 0$ and let $N \in \mathbb{N}$ be such that for any $n, m \geq N$, $\|a^{(n)} - a^{(m)}\|_{\mathscr{T}} < \varepsilon$. Clearly, for any $K \in \mathbb{N}$,

$$\sum_{k=0}^{K} \|a_k^{(n)} - a_k^{(m)}\|^2_{(\mathbb{R}^d)^{\otimes k}} < \varepsilon^2.$$

Letting $m \to \infty$ leads to

$$\sum_{k=1}^{K} \|a_k^{(n)} - a_k^{(\infty)}\|^2_{(\mathbb{R}^d)^{\otimes k}} < \varepsilon^2,$$

and letting $K \to \infty$ gives

$$\|a^{(n)} - a^{(\infty)}\|_{\mathscr{T}} < \varepsilon,$$

which completes the proof. □

D.1.5 Bounding the derivatives of the logistic and hyperbolic tangent activations

Lemma D.4. *Let σ be the logistic function defined, for any $x \in \mathbb{R}$, by $\sigma(x) = 1/(1+e^{-x})$. Then, for any $n \geq 0$,*

$$\|\sigma^{(n)}\|_\infty \leq 2^{n-1} n!\,.$$

Proof. For any $x \in \mathbb{R}$, one has (Minai and Williams, 1993, Theorem 2)

$$\sigma^{(n)}(x) = \sum_{k=1}^{n+1} (-1)^{k-1}(k-1)! \left\{ {n+1 \atop k} \right\} \sigma(x)^k,$$

where $\left\{ {n \atop k} \right\}$ stands for the Stirling number of the second kind (see, e.g., Riordan, 1958). Let

$$u_n = \sum_{k=1}^{n+1} (k-1)! \left\{ {n+1 \atop k} \right\}$$

for $n \geq 1$ and $u_0 = 1$. Since $0 \leq \sigma(x) \leq 1$, it is clear that $|\sigma^{(n)}(x)| \leq u_n$. Using the fact that the Stirling numbers satisfy the recurrence relation

$$\left\{ {n+1 \atop k} \right\} = k \left\{ {n \atop k} \right\} + \left\{ {n \atop k-1} \right\},$$

valid for all $0 \leq k \leq n$, we have

$$\begin{aligned} u_n &= \sum_{k=1}^{n} (k-1)! \left(k \left\{ {n \atop k} \right\} + \left\{ {n \atop k-1} \right\} \right) + n! = \sum_{k=1}^{n} k! \left\{ {n \atop k} \right\} + \sum_{k=0}^{n-1} k! \left\{ {n \atop k} \right\} + n! = 2 \sum_{k=1}^{n} k! \left\{ {n \atop k} \right\} \\ &\qquad \text{(since } \left\{ {n \atop 0} \right\} = 0\text{)} \\ &\leq 2n \sum_{k=1}^{n} (k-1)! \left\{ {n \atop k} \right\} = 2n u_{n-1}. \end{aligned}$$

Thus, by induction, $u_n \leq 2^{n-1} n!$, from which the claim follows. □

Lemma D.5. *Let* tanh *be the hyperbolic tangent function. Then, for any $n \geq 0$,*

$$\|\tanh^{(n)}\|_\infty \leq 4^n n!\,.$$

Proof. Let σ be the logistic function. Straightforward calculations yield the equality, valid for any $x \in \mathbb{R}$,

$$\tanh(x) = 2\sigma(2x) - 1.$$

But, for any $n \geq 1$,

$$\tanh^{(n)}(x) = 2^{n+1}\sigma^{(n)}(2x),$$

and thus, by Lemma D.4,

$$\|\tanh^{(n)}\|_\infty \leq 2^{n+1}\|\sigma^{(n)}\|_\infty \leq 4^n n!\,.$$

The inequality is also true for $n = 0$ since $\|\tanh\|_\infty \leq 1$. □

D.1.6 Chen's formula

First, note that it is straightforward to extend the definition of the signature to any interval $[s,t] \subset [0,1]$. The next proposition, known as Chen's formula (Lyons et al., 2007, Theorem 2.9), tells us that the signature can be computed iteratively as tensor products of signatures on subintervals.

Proposition D.6. *Let $X \in BV([s,t], \mathbb{R}^d)$ and $u \in (s,t)$. Then*

$$S_{[s,t]}(X) = S_{[s,u]}(X) \otimes S_{[u,t]}(X).$$

Next, it is clear that the signature of a constant path is equal to $\mathbf{1} = (1, 0, \dots, 0, \dots)$ which is the null element in $\mathscr{T}$. Indeed, let $Y \in BV([s,t], \mathbb{R}^d)$ be a constant path. Then, for any $k \geq 1$,

$$\mathbb{Y}^k_{[s,t]} = k! \int\limits_{s \leq u_1 < \cdots < u_k \leq t} \cdots \int dY_{u_1} \otimes \cdots \otimes dY_{u_k} = k! \int\limits_{s \leq u_1 < \cdots < u_k \leq t} \cdots \int 0 \otimes \cdots \otimes 0 = 0.$$

Now let $X \in BV([0,1], \mathbb{R}^d)$ and consider the path $\tilde{X}_{[j]}$ equal to the time-augmented path $\bar{X}$ on $[0, j/T]$ and then constant on $[j/T, 1]$—see Figure D.1. We have by Proposition D.6

$$S_{[0,1]}(\tilde{X}_{[j]}) = S_{[0,j/T]}(\tilde{X}_{[j]}) \otimes S_{[j/T,1]}(\tilde{X}_{[j]}) = S_{[0,j/T]}(\bar{X}) \otimes \mathbf{1} = S_{[0,j/T]}(\bar{X}).$$

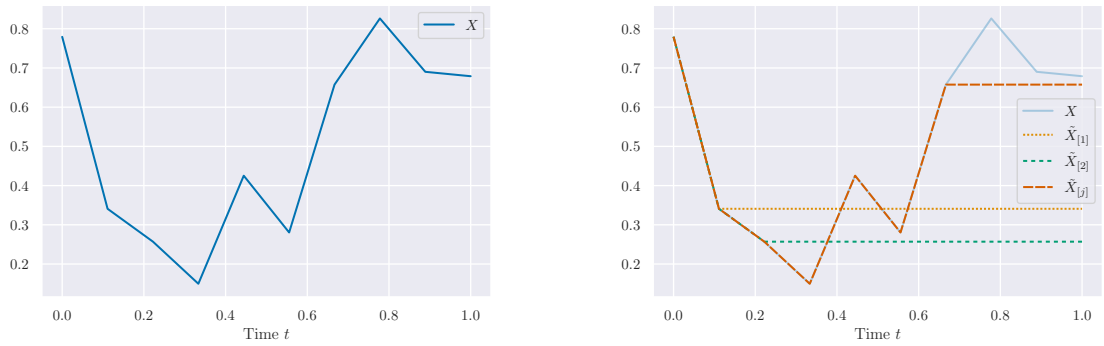


Figure D.1 – Example of a path $X \in BV([0,1], \mathbb{R})$ (left) and its corresponding paths $\tilde{X}_{[j]}$, plotted against time, for different values of $j \in \{1, \dots, T\}$ (right)

D.2 Proofs

D.2.1 Proof of Proposition 5.1

According to Assumption (A_1), for any $h_1, h_2 \in \mathbb{R}^e, x_1, x_2 \in \mathbb{R}^d$, one has

$$\|f(h_1, x_1) - f(h_2, x_1)\| \leq K_f \|h_1 - h_2\| \quad \text{and} \quad \|f(h_1, x_1) - f(h_1, x_2)\| \leq K_f \|x_1 - x_2\|.$$

Under assumption (A_1), by Corollary D.2, the initial value problem (5.3) admits a unique solution H. Let us first show that for any $t \in [0,1]$, H_t is bounded independently of X. For any $t \in [0,1]$,

$$\begin{aligned}
\|H_t - H_0\| = \Big\| \int_0^t f(H_u, X_u) du \Big\| &\leq \int_0^t \|f(H_u, X_u)\| du \\
&= \int_0^t \|f(H_u, X_u) - f(H_0, X_u) + f(H_0, X_u)\| du \\
&\leq \int_0^t \|f(H_u, X_u) - f(H_0, X_u)\| + \int_0^t \|f(H_0, X_u)\| du \\
&\leq K_f \int_0^t \|H_u - H_0\| du + t \sup_{\|x\| \leq L} \|f(H_0, x)\|.
\end{aligned}$$

Applying Grönwall's inequality to the function $t \mapsto \|H_t - H_0\|$ yields

$$\|H_t - H_0\| \leq t \sup_{\|x\| \leq L} \|f(H_0, x)\| \exp\Big(\int_0^t K_f du \Big) \leq \sup_{\|x\| \leq L} \|f(H_0, x)\| e^{K_f} := M.$$

Given that $H_0 = h_0 = 0$, we conclude that $\|H_t\| \leq M$.

Next, let

$$\|f\|_\infty = \sup_{\|x\| \leq L, \|h\| \leq M} f(h, x).$$

By similar arguments, for any $[s,t] \subset [0,1]$, Grönwall's inequality applied to the function $t \mapsto \|H_t - H_s\|$ yields

$$\|H_t - H_s\| \leq (t-s) \|f\|_\infty e^{K_f}.$$

Therefore, for any partition $(t_0, \ldots, t_k)$ of $[s,t]$,

$$\sum_{i=1}^k \|H_{t_i} - H_{t_{i-1}}\| \leq \|f\|_\infty e^{K_f} \sum_{i=1}^k (t_i - t_{i-1}) \leq \|f\|_\infty e^{K_f} (t-s),$$

and, taking the supremum over all partitions of $[s,t]$, $\|H\|_{TV;[s,t]} \leq \|f\|_\infty e^{K_f}(t-s)$. In other words, H is of bounded variation on any interval $[s,t] \subset [0,1]$. Let $(t_0, \ldots, t_T)$ denote the regular partition of $[0,1]$ with $t_j = j/T$. For any $1 \leq j \leq T$, we have

$$\begin{aligned}
\|H_{t_j} - h_j\| &= \Big\| H_{t_{j-1}} + \int_{t_{j-1}}^{t_j} f(H_u, X_u) du - h_{j-1} - \frac{1}{T} f(h_{j-1}, x_j) \Big\| \\
&\leq \|H_{t_{j-1}} - h_{j-1}\| + \int_{t_{j-1}}^{t_j} \Big\| f(H_u, X_u) - f(h_{j-1}, x_j) \Big\| du.
\end{aligned}$$

Writing

$$\begin{aligned}\big\|f(H_u,X_u)-f(h_{j-1},x_j)\big\| &= \big\|f(H_u,X_u)-f(H_u,x_j)+f(H_u,x_j)-f(h_{j-1},x_j)\big\|\\ &\le \big\|f(H_u,X_u)-f(H_u,x_j)\big\|+\big\|f(H_u,x_j)-f(h_{j-1},x_j)\big\|\\ &\le K_f\big\|X_u-x_j\big\|+K_f\big\|H_u-h_{j-1}\big\|,\end{aligned}$$

we obtain

$$\begin{aligned}\|H_{t_j}-h_j\| &\le \|H_{t_{j-1}}-h_{j-1}\|+K_f\int_{t_{j-1}}^{t_j}\|H_u-h_{j-1}\|du+K_f\int_{t_{j-1}}^{t_j}\|X_u-x_j\|du\\ &\le \|H_{t_{j-1}}-h_{j-1}\|+K_f\int_{t_{j-1}}^{t_j}\big(\|H_u-H_{t_{j-1}}\|+\|H_{t_{j-1}}-h_{j-1}\|\big)du\\ &\quad+\frac{K_f}{T}\|X\|_{TV;[t_{j-1},t_j]}\\ &\le \big(1+\frac{K_f}{T}\big)\|H_{t_{j-1}}-h_{j-1}\|+\frac{K_f}{T}\big(\|H\|_{TV;[t_{j-1},t_j]}+\|X\|_{TV;[t_{j-1},t_j]}\big).\end{aligned}$$

By induction, we are led to

$$\begin{aligned}\|H_{t_j}-h_j\| &\le \frac{K_f}{T}\sum_{k=0}^{j-1}\Big(1+\frac{K_f}{T}\Big)^k\big(\|H\|_{TV;[t_k,t_{k+1}]}+\|X\|_{TV;[t_k,t_{k+1}]}\big)\\ &\le \frac{K_f}{T}\Big(1+\frac{K_f}{T}\Big)^T\big(\|X\|_{TV;[0,1]}+\|H\|_{TV;[0,1]}\big)\\ &\le \frac{K_f e^{K_f}}{T}\big(L+\|f\|_\infty e^{K_f}\big),\end{aligned}$$

which concludes the proof.

D.2.2 Proof of Proposition 5.2

Let $\bar{h}\in\mathbb{R}^{\bar{e}}$ and let $\bar{h}^{i:j}=(\bar{h}^i,\dots,\bar{h}^j)$ be its projection on a subset of coordinates. It is sufficient to take $\mathbf{F}$ defined by

$$\mathbf{F}(\bar{h})=\begin{pmatrix}0_{e\times d} & \frac{2}{1-L}f(\bar{h}^{1:e},\bar{h}^{e+1:e+d})\\ I_{d\times d} & 0_{d\times 1}\end{pmatrix},$$

where $I_{d\times d}$ denotes the identity matrix and $0_{\times\cdot}$ the matrix full of zeros. The function $\bar{H}$ is then solution of

$$d\bar{H}_t=\begin{pmatrix}0_{e\times d} & \frac{2}{1-L}f(\bar{H}_t^{1:e},\bar{H}_t^{e+1:e+d})\\ I_{d\times d} & 0_{d\times 1}\end{pmatrix}\begin{pmatrix}dX_t\\ \frac{1-L}{2}dt\end{pmatrix}.$$

Note that under assumption (A_1), the tensor field $\mathbf{F}$ satisfies the assumptions of the Picard-Lindelöf theorem (Theorem D.1) so that $\bar{H}$ is well-defined. The projection of this equation on the last d coordinates gives

$$d\bar{H}_t^{e+1:e+d}=dX_t,\quad \bar{H}_0^{e+1:e+d}=X_0,$$

and therefore $\bar{H}_t^{e+1:e+d} = X_t$. The projection on the first e coordinates gives

$$d\bar{H}_t^{1:e} = \frac{2}{1-L} f(\bar{H}_t^{1:e}, X_t) \frac{1-L}{2} dt = f(\bar{H}_t^{1:e}, X_t) dt, \quad \bar{H}_0^{1:e} = h_0,$$

which is exactly (5.3).

D.2.3 Proof of Proposition 5.3

According to Lyons (2014, Lemma 5.1), one has

$$\|\bar{\mathbb{X}}^k_{[0,t]}\|_{(\mathbb{R}^d)^{\otimes k}} \leq \|\bar{X}\|^k_{TV;[0,t]}.$$

Let $(t_0, \dots, t_k)$ be a partition of $[0,t]$. Then

$$\begin{aligned}
\sum_{j=1}^k \|\bar{X}_{t_j} - \bar{X}_{t_{j-1}}\| &= \sum_{j=1}^k \sqrt{\|X_{t_j} - X_{t_{j-1}}\|^2 + \Big(\frac{1-L}{2}\Big)^2 (t_j - t_{j-1})^2} \\
&\leq \sum_{j=1}^k \|X_{t_j} - X_{t_{j-1}}\| + \frac{1-L}{2} \sum_{j=1}^k (t_j - t_{j-1}) \\
&= \sum_{j=1}^k \|X_{t_j} - X_{t_{j-1}}\| + \frac{1-L}{2} t.
\end{aligned}$$

Taking the supremum over any partition of $[0,t]$ we obtain

$$\|\bar{X}\|_{TV;[0,t]} \leq \|X\|_{TV;[0,t]} + \frac{1-L}{2} t \leq L + \frac{1-L}{2} = \frac{1+L}{2} < 1,$$

and thus $\|\bar{\mathbb{X}}^k_{[0,t]}\|_{(\mathbb{R}^d)^{\otimes k}} \leq \Big(\frac{1+L}{2}\Big)^k$. It is then clear that

$$\|S_{[0,t]}(\bar{X})\|_{\mathscr{T}} = \Big(\sum_{k=0}^\infty \|\bar{\mathbb{X}}^k_{[0,t]}\|^2_{(\mathbb{R}^d)^{\otimes k}}\Big)^{1/2} \leq \sum_{k=0}^\infty \|\bar{\mathbb{X}}^k_{[0,t]}\|_{(\mathbb{R}^d)^{\otimes k}} \leq \sum_{k=0}^\infty \Big(\frac{1+L}{2}\Big)^k = 2(1-L)^{-1}.$$

D.2.4 Proof of Proposition 5.4

We first recall the fundamental theorem of calculus for line integrals (also known as gradient theorem).

Theorem D.7. *Let $g : \mathbb{R}^e \to \mathbb{R}$ be a continuously differentiable function, and let $\gamma : [a,b] \to \mathbb{R}^e$ be a smooth curve in $\mathbb{R}^e$. Then*

$$\int_a^b \nabla g(\gamma_t) d\gamma_t = g(\gamma_b) - g(\gamma_a),$$

where ∇g denotes the gradient of g.

The identity above immediately generalizes to a function $g : \mathbb{R}^e \to \mathbb{R}^e$:

$$\int_a^b J(g)(\gamma_t) d\gamma_t = g(\gamma_b) - g(\gamma_a),$$

where $J(g) \in \mathbb{R}^{e\times e}$ is the Jacobian matrix of g. Let us apply Theorem D.7 to the vector field F^i between 0 and t, with $\gamma = H$. We have

$$\begin{aligned} F^i(H_t) - F^i(H_0) &= \int_0^t J(F^i)(H_u)dH_u = \int_0^t J(F^i)(H_u)\sum_{j=1}^d F^j(H_u)dX_u \\ &= \sum_{j=1}^d \int_0^t J(F^i)(H_u)F^j(H_u)dX_u = \sum_{j=1}^d \int_0^t F^j \star F^i(H_u)dX_u. \end{aligned}$$

Iterating this procedure $(N-1)$ times for the vector fields $F^1, \dots, F^d$ yields

$$\begin{aligned} H_t &= H_0 + \sum_{i=1}^d \int_0^t F^i(H_u)dX_u^i \\ &= H_0 + \sum_{i=1}^d \int_0^t F^i(H_0)dX_u^i + \sum_{i=1}^d \int_0^t \sum_{j=1}^d \int_0^u F^j \star F^i(H_v)dX_v^j dX_u^i \\ &= H_0 + \sum_{i=1}^d F^i(H_0)S^{(i)}(X)_{[0,t]} + \sum_{1\le i,j\le d} \int_{0\le v\le u\le t} F^j \star F^i(H_v)dX_v^j dX_u^i \\ &= \cdots \\ &= H_0 + \sum_{k=1}^N \sum_{1\le i_1,\dots,i_k\le d} F^{i_1}\star\cdots\star F^{i_k}(H_0)\frac{1}{k!}S^{(i_1,\dots,i_k)}_{[0,t]}(X) \\ &\quad + \sum_{1\le i_1,\dots,i_{N+1}\le d} \int_{\Delta_{N+1;[0,t]}} F^{i_1}\star\cdots\star F^{i_{N+1}}(H_{u_1})dX_{u_1}^{i_1}\cdots dX_{u_{N+1}}^{i_{N+1}}, \end{aligned}$$

where $\Delta_{N;[0,t]} := \{(u_1,\cdots,u_N) \in [0,t]^N \,|\, 0 \le u_1 < \cdots < u_N \le t\}$ is the simplex in $[0,t]^N$. The first $(N+1)$ terms equal H_t^N. Hence,

$$\begin{aligned} &\|H_t - H_t^N\| \\ &= \Big\| \sum_{1\le i_1,\dots,i_{N+1}\le d} \int_{\Delta_{N+1;[0,t]}} F^{i_1}\star\cdots\star F^{i_{N+1}}(H_{u_1})dX_{u_1}^{i_1}\cdots dX_{u_{N+1}}^{i_{N+1}} \Big\| \\ &\le \sum_{1\le i_1,\dots,i_{N+1}\le d} \int_{\Delta_{N+1;[0,t]}} \|F^{i_1}\star\cdots\star F^{i_{N+1}}(H_{u_1})\||dX_{u_1}^{i_1}|\cdots|dX_{u_{N+1}}^{i_{N+1}}| \\ &\le \sum_{1\le i_1,\dots,i_{N+1}\le d} \int_{\Delta_{N+1;[0,t]}} \sup_{1\le i_1,\dots,i_{N+1}\le d, \|h\|\le M} \|F^{i_1}\star\cdots\star F^{i_{N+1}}(h)\||dX_{u_1}^{i_1}|\cdots|dX_{u_{N+1}}^{i_{N+1}}| \\ &\le \Lambda_{N+1}(\mathbf{F}) \sum_{1\le i_1,\dots,i_{N+1}\le d} \int_{\Delta_{N+1;[0,t]}} |dX_{u_1}^{i_1}|\cdots|dX_{u_{N+1}}^{i_{N+1}}|. \end{aligned}$$

Thus,

$$\begin{aligned}\|H_t - H_t^N\| &\leq \Lambda_{N+1}(\mathbf{F}) \sum_{1\leq i_1,\dots,i_{N+1}\leq d} \int_{\Delta_{N+1;[0,t]}} |dX_{u_1}^{i_1}| \cdots |dX_{u_{N+1}}^{i_{N+1}}| \\ &\leq \Lambda_{N+1}(\mathbf{F}) \sum_{1\leq i_1,\dots,i_{N+1}\leq d} \int_{\Delta_{N+1;[0,t]}} \|dX_{u_1}\| \cdots \|dX_{u_{N+1}}\| \\ &= \Lambda_{N+1}(\mathbf{F}) \frac{d^{N+1}}{(N+1)!} \int_{[0,t]^{N+1}} \|dX_{u_1}\| \cdots \|dX_{u_{N+1}}\| \\ &= \Lambda_{N+1}(\mathbf{F}) \frac{d^{N+1}}{(N+1)!} \Big(\int_0^t \|dX_u\| \Big)^{N+1} \\ &= \Lambda_{N+1}(\mathbf{F}) \frac{d^{N+1}}{(N+1)!} \|X\|_{TV;[0,t]}^{N+1} \leq \Lambda_{N+1}(\mathbf{F}) \frac{d^{N+1}}{(N+1)!}.\end{aligned}$$

D.2.5 Proof of Proposition 5.5

For simplicity of notation, since the context is clear, we now use the notation $\|\cdot\|$ instead of $\|\cdot\|_{(\mathbb{R}^e)^{\otimes k}}$. According to Proposition 5.1, the solution $\bar{H}$ of (5.4) verifies $\|\bar{H}_t\| \leq M + L := \bar{M}$. We therefore place ourselves in the ball $\mathscr{B}_{\bar{M}}$. Recall that for any $1 \leq i_1, \dots, i_N \leq d$, $\bar{h} \in \mathscr{B}_{\bar{M}}$,

$$F^{i_1} \star \cdots \star F^{i_N}(\bar{h}) = J(F^{i_2} \star \cdots \star F^{i_N})(\bar{h}) F^{i_1}(\bar{h}). \tag{D.2}$$

Linear case. We start with the proof of the linear case before moving on to the general case. When σ is chosen to be the identity function, each F_{RNN}^i is an affine vector field, in the sense that $F_{\mathrm{RNN}}^i(\bar{h}) = W_i \bar{h} + b_i$, where $W_i = 0_{\bar{e}\times\bar{e}}$, b_i is the $i+d$th vector of the canonical basis of $\mathbb{R}^{e+d}$, and

$$W_{d+1} = \begin{pmatrix} \frac{2}{1-L} W \\ 0_{d\times\bar{e}} \end{pmatrix} \quad \text{and} \quad b_{d+1} = \begin{pmatrix} \frac{2}{1-L} b \\ 0_d \end{pmatrix}.$$

Since $J(F_{\mathrm{RNN}}^i) = W_i$, we have, for any $\bar{h} \in \mathbb{R}^{e+d}$ and any $1 \leq i_1, \dots, i_k \leq d$,

$$F_{\mathrm{RNN}}^{i_1} \star \cdots \star F_{\mathrm{RNN}}^{i_k}(\bar{h}) = W_{i_k} \cdots W_{i_2}(W_{i_1}\bar{h} + b_{i_1}).$$

Thus, for any $\bar{h} \in \mathscr{B}_{\bar{M}}$,

$$\|F_{\mathrm{RNN}}^{i_1} \star \cdots \star F_{\mathrm{RNN}}^{i_k}(\bar{h})\| \leq \|W_{i_k}\|_{\mathrm{op}} \cdots \|W_{i_2}\|_{\mathrm{op}} (\|W_{i_1}\|_{\mathrm{op}} \bar{M} + \|b_{i_1}\|).$$

For $i \neq d+1$, $\|W_{i_1}\|_{\mathrm{op}} = 0$, and so

$$\Lambda_k(\mathbf{F}_{\mathrm{RNN}}) \leq C \|W_{d+1}\|_{\mathrm{op}}^{k-1},$$

with $C = \|W_{d+1}\|_{\mathrm{op}} \bar{M} + \max(1, 2(1-L)^{-1}\|b\|)$. Therefore,

$$\sum_{k=1}^{\infty} \frac{d^k}{k!} \Lambda_k(\mathbf{F}_{\mathrm{RNN}}) \leq Cd \sum_{k=0}^{\infty} \frac{1}{k!} \big(2d(1-L)^{-1}\|W\|_{\mathrm{op}}\big)^{k-1} < \infty.$$

General case. In the general case, the proof is two-fold. First, we upper bound (D.2) by a function of the norms of higher-order Jacobians of $F^{i_1}, \dots, F^{i_N}$. We then apply this bound to the specific case $\mathbf{F} = \mathbf{F}_{\mathrm{RNN}}$. We refer to Appendix D.3 for details on higher-order derivatives in

tensor spaces. Let $F : \mathbb{R}^e \to \mathbb{R}^e$ be a smooth vector field. If $F(h) = (F_1(h), \dots, F_e(h))^\top$, each of its coordinates F_i is a function from $\mathbb{R}^e$ to $\mathbb{R}$, $\mathscr{C}^\infty$ with respect to all its input variables. We define the derivative of order k of F as the tensor field

$$\begin{aligned} J^k(F) : \mathbb{R}^e &\to (\mathbb{R}^e)^{\otimes k+1} \\ h &\mapsto J^k(F)(h), \end{aligned}$$

where

$$J^k(F)(h) = \sum_{1 \le j, i_1, \dots, i_k \le e} \frac{\partial^k F_j(h)}{\partial h_{i_1} \dots \partial h_{i_k}} e_j \otimes e_{i_1} \otimes \dots \otimes e_{i_k}.$$

We take the convention $J^0(F) = F$, and note that $J(F) = J^1(F)$ is the Jacobian matrix, and that $J^k(J^{k'}(F)) = J^{k+k'}(F)$.

Lemma D.8. *Let $A^1, \dots, A^k : \mathbb{R}^e \to \mathbb{R}^e$ be smooth vector fields. Then, for any $h \in \mathbb{R}^e$*

$$\left\| A^k \star \dots \star A^1(h) \right\| \le \sum_{n_1 + \dots + n_k = k-1} C(k; n_1, \dots, n_k) \|J^{n_1}(A^1)(h)\| \dots \|J^{n_k}(A^k)(h)\|,$$

where $C(k; n_1, \dots, n_k)$ is defined by the following recurrence on k: $C(1; 0) = 1$ and for any $n_1, \dots, n_{k+1} \ge 0$,

$$\begin{aligned} C(k+1; n_1, \dots, n_{k+1}) &= \sum_{\ell=1}^{k} C(k; n_1, \dots, n_\ell - 1, \dots, n_k) && \text{if} \quad n_{k+1} = 0, \qquad \text{(D.3)} \\ C(k+1; n_1, \dots, n_{k+1}) &= 0 && \text{otherwise.} \end{aligned}$$

Proof. We refer to Appendix D.3 for the definitions of the tensor dot product $\odot$ and tensor permutations, as well as for computation rules involving these operations. We show in fact by induction a stronger result, namely that there exist tensor permutations π_p such that

$$A^k \star \dots \star A^1(h) = \sum_{n_1 + \dots + n_k = k-1} \sum_{1 \le p \le C(k; n_1, \dots, n_k)} \pi_p \left[J^{n_1}(A^1)(h) \odot \dots \odot J^{n_k}(A^k)(h) \right]. \quad \text{(D.4)}$$

Note that we do not make explicit the permutations nor the axes of the tensor dot operations since we are only interested in bounding the norm of the iterated star products. Also, for simplicity, we denote all permutations by π, even though they may change from line to line.

We proceed by induction on k. For $k = 1$, the formula is clear. Assume that the formula is true at order k. Then

$$\begin{aligned} &J(A^k \star \dots \star A^1) \\ &\quad = \sum_{n_1 + \dots + n_k = k-1} \sum_{1 \le p \le C(k; n_1, \dots, n_k)} J\Big[\, \pi_p[\, J^{n_1}(A^1) \odot \dots \odot J^{n_k}(A^k)\,] \, \Big] \\ &\quad = \sum_{n_1 + \dots + n_k = k-1} \sum_{1 \le p \le C(k; n_1, \dots, n_k)} \pi_p\Big[\, J[\, J^{n_1}(A^1) \odot \dots \odot J^{n_k}(A^k)\,] \, \Big] \\ &\quad = \sum_{n_1 + \dots + n_k = k-1} \sum_{1 \le p \le C(k; n_1, \dots, n_k)} \sum_{\ell=1}^{k} \pi_p \circ \pi_\ell \Big[J^{n_1}(A^1) \odot \\ &\qquad\qquad \dots \odot J^{n_\ell + 1}(A^\ell) \odot \dots \odot J^{n_k}(A^k) \Big]. \end{aligned}$$

In the inner sum, we introduce the change of variable $p_i = n_i$ for $i \neq \ell$ and $p_\ell = n_\ell + 1$. This yields

$$\begin{aligned}
&J(A^k \star \cdots \star A^1) \\
&\quad = \sum_{p_1+\cdots+p_k=k} \sum_{\ell=1}^{k} \sum_{1\le p\le C(k;p_1,\ldots,p_\ell-1,\ldots,p_k)} \pi_p \circ \pi_\ell \Big[J^{n_1}(A^1) \odot \\
&\qquad\qquad \cdots \odot J^{n_\ell+1}(A^\ell) \odot \cdots \odot J^{n_k}(A^k) \Big] \\
&\quad = \sum_{p_1+\cdots+p_{k+1}=k} \sum_{1\le q\le C(k+1;p_1,\ldots,p_{k+1})} \pi_q \Big[J^{n_1}(A^1) \odot \cdots \odot J^{p_k}(A^k) \Big],
\end{aligned}$$

where in the last sum the only non-zero term is for $p_{k+1} = 0$. To conclude the induction, it remains to note that

$$A^{k+1} \star \cdots \star A^1 = J(A^k \star \cdots \star A^1) \odot A^{k+1} = J(A^k \star \cdots \star A^1) \odot J^0(A^{k+1}).$$

Hence,

$$\begin{aligned}
&A^{k+1} \star \cdots \star A^1 \\
&\quad = \sum_{p_1+\cdots+p_{k+1}=k} \sum_{1\le q\le C(k+1;p_1,\ldots,p_{k+1})} \pi_q \left[J^{n_1}(A^1) \odot \cdots \odot J^{p_k}(A^k) \right] \odot J^{p_{k+1}}(A^{k+1}) \\
&\quad = \sum_{p_1+\cdots+p_{k+1}=k} \sum_{1\le q\le C(k+1;p_1,\ldots,p_{k+1})} \pi_q \left[J^{n_1}(A^1) \odot \cdots \odot J^{p_k}(A^k) \odot J^{p_{k+1}}(A^{k+1}) \right].
\end{aligned}$$

The result is then a consequence of (D.4) and of Lemma D.11. □

We now restrict ourselves to the case $\mathbf{F} = \mathbf{F}_{\mathrm{RNN}}$ as defined by (5.5) and give an upper bound on the higher-order derivatives of the tensor fields $F^{i_1}, \ldots, F^{i_N}$.

Lemma D.9. *For any $i \in \{1, \ldots, d+1\}$, $\bar{h} \in \mathscr{B}_{\bar{M}}$, for any $k \geq 0$,*

$$\|J^k(F^i_{\mathrm{RNN}})(\bar{h})\| \leq \Big(\frac{2}{1-L}\|W\|_F\Big)^k \|\sigma^{(k)}\|_\infty.$$

Proof. For any $1 \leq i \leq d$, $F^i_{\mathrm{RNN}}(\bar{h})$ is constant, so $J^k(F^1_{\mathrm{RNN}}) = \cdots = J^k(F^d_{\mathrm{RNN}}) = 0$. For $i = d+1$, we have, for any $1 \leq j \leq e$,

$$\frac{\partial^k F^{d+1}_{\mathrm{RNN},j}(\bar{h})}{\partial \bar{h}_{i_1} \ldots \partial \bar{h}_{i_k}} = \Big(\frac{2}{1-L}\Big)^k W_{ji_1} \cdots W_{ji_k} \sigma^{(k)}(W_{j\cdot}\bar{h} + b),$$

where $W_{j\cdot}$ denotes the jth row of W and for $e+1 \leq j \leq \bar{e}$, $F^{d+1}_j = 0$. Therefore,

$$\begin{aligned}
\|J^k(F^{d+1}_{\mathrm{RNN}})(\bar{h})\|^2 &\leq \Big(\frac{2}{1-L}\Big)^{2k} \sum_{1\le j,i_1,\ldots,i_k\le e} |W_{ji_1} \cdots W_{ji_k} \sigma^{(k)}(W_{j\cdot}\bar{h} + b)|^2 \\
&= \Big(\frac{2}{1-L}\Big)^{2k} \|\sigma^{(k)}\|^2_\infty \sum_j \Big(\sum_i |W_{ji}|^2\Big)^k \\
&\leq \Big(\frac{2}{1-L}\Big)^{2k} \|\sigma^{(k)}\|^2_\infty \|W\|^{2k}_F.
\end{aligned}$$

□

We are now in a position to conclude the proof using condition (5.10). By Lemma D.8 and D.9, for any $1 \le i_1, \dots, i_N \le d+1$,

$$\begin{aligned}
&\big\| F^{i_1}_{\mathrm{RNN}} \star \cdots \star F^{i_N}_{\mathrm{RNN}}(\bar{h}) \big\| \\
&\quad \le \sum_{n_1+\cdots+n_N=N-1} C(N; n_N, \dots, n_1) \| J^{n_N}(F^{i_N}_{\mathrm{RNN}})(\bar{h}) \| \cdots \| J^{n_1}(F^{i_1}_{\mathrm{RNN}})(\bar{h}) \| \\
&\quad \le \Big(\frac{2}{1-L} \|W\|_F \Big)^{N-1} \sum_{n_1+\cdots+n_N=N-1} C(N; n_N, \dots, n_1) a^{n_1+1} n_1! \cdots a^{n_N+1} n_N! \\
&\quad \le a \Big(\frac{2}{1-L} a^2 \|W\|_F \Big)^{N-1} \sum_{n_1+\cdots+n_N=N-1} C(N; n_N, \dots, n_1) n_1! \cdots n_N! \,.
\end{aligned}$$

Assume for the moment that $C(N; n_N, \dots, n_1)$ is smaller than the multinomial coefficient $\binom{N}{n_N, \dots, n_1}$. Then, using the fact that there are $\binom{n+k-1}{k-1}$ weak compositions of n in k parts and Stirling's approximation, we have

$$\begin{aligned}
\Lambda_N(\mathbf{F}) &\le a \Big(\frac{2}{1-L} a^2 \|W\|_F \Big)^{N-1} N! \times \ \mathrm{Card}\big(\{ n_1 + \cdots + n_N = N-1 \} \big) \\
&\le a \Big(\frac{2}{1-L} a^2 \|W\|_F \Big)^{N-1} N! \binom{2N-2}{N-1} \\
&\le \frac{a}{2} \Big(\frac{2}{1-L} a^2 \|W\|_F \Big)^{N-1} N! \binom{2N}{N} \\
&\le a \frac{\sqrt{2}e}{\pi} \Big(\frac{8}{1-L} a^2 \|W\|_F \Big)^{N-1} \frac{N!}{\sqrt{N}}.
\end{aligned}$$

Hence, provided $\|W\|_F < {}^{(1-L)}\!/_{8a^2 d}$,

$$\sum_{k=1}^{\infty} \frac{d^k}{k!} \Lambda_k(\mathbf{F}) \le ad \frac{\sqrt{2}e}{\pi} \sum_{k=1}^{\infty} \Big(\frac{8da^2 \|W\|_F}{1-L} \Big)^{k-1} \frac{1}{\sqrt{k}} < \infty,$$

and (A_2) is verified.

To conclude the proof, it remains to prove the following lemma.

Lemma D.10. *For any $k \ge 1$ and $n_1, \dots, n_k \ge 0$, $C(k; n_1, \dots, n_k) \le \binom{k-1}{n_1, \dots, n_k}$.*

Proof. The proof is done by induction, by comparing the recurrence formula (D.3) with the following recurrence formula for multinomial coefficients:

$$\binom{k}{n_1, \dots, n_{k+1}} = \sum_{\ell=1}^{k+1} \binom{k-1}{n_1, \dots, n_\ell - 1, \dots, n_{k+1}}.$$

More precisely, for $k = 1$, $C(1;0) = 1 \le \binom{0}{0} = 1$ and $C(1;1) = 0 \le \binom{0}{1} = 0$. Assume that the formula is true at order k. Then, at order $k+1$, there are two cases. If $n_{k+1} \neq 0$,

$C(k+1;n_1,\dots,n_{k+1})=0$, and the result is clear. On the other hand, if $n_{k+1}=0$,

$$\begin{aligned}C(k+1;n_1,\dots,n_k,0)&=\sum_{\ell=1}^{k}C(k;n_1,\dots,n_\ell-1,\dots,n_k)\\&\le\sum_{\ell=1}^{k}\binom{k-1}{n_1,\dots,n_\ell-1,\dots,n_k}\\&\le\sum_{\ell=1}^{k+1}\binom{k-1}{n_1,\dots,n_\ell-1,\dots,n_{k+1}}\le\binom{k}{n_1,\dots,n_{k+1}}.\end{aligned}$$

□

D.2.6 Proof of Theorem 5.6

First, Propositions 5.1 and 5.2 state that if $\bar{H}$ is the solution of (5.4) and Proj denotes the projection on the first e coordinates, then

$$\big|z_T-\psi\big(\mathrm{Proj}(\bar{H}_1)\big)\big|=\big|\psi(h_T)-\psi\big(\mathrm{Proj}(\bar{H}_1)\big)\big|\le\|\psi\|_{\mathrm{op}}\big\|h_T-\mathrm{Proj}(\bar{H}_1)\big\|\le\|\psi\|_{\mathrm{op}}\frac{c_1}{T}.$$

For any $1\le k\le N$, we let $\mathscr{D}^k(\bar{H}_0):(\mathbb{R}^d)^{\otimes k}\to\mathbb{R}^e$ be the linear function defined by

$$\mathscr{D}^k(\bar{H}_0)(e_{i_1}\otimes\cdots\otimes e_{i_k})=F^{i_1}\star\cdots\star F^{i_k}(\bar{H}_0),\tag{D.5}$$

where $e_1,\dots,e_d$ denotes the canonical basis of $\mathbb{R}^{\bar{d}}$. Then, under assumptions (A_1) and (A_2), if $\bar{\mathbb{X}}^k$ denotes the signature of order k of the path $\bar{X}_t=(X_t^\top,\frac{1-L}{2}t)^\top$, according to Propositions 5.4 and 5.5,

$$\bar{H}_1=\bar{H}_0+\sum_{k=1}^{\infty}\frac{1}{k!}\sum_{1\le i_1,\dots,i_k\le d}S^{(i_1,\dots,i_k)}_{[0,t]}(X)F^{i_1}\star\cdots\star F^{i_k}(\bar{H}_0)=\sum_{k=1}^{\infty}\frac{1}{k!}\mathscr{D}^k(\bar{H}_0)(\mathbb{X}^k_{[0,t]}),$$

and

$$\psi\circ\mathrm{Proj}(\bar{H}_1)=\psi\circ\mathrm{Proj}\Big(\sum_{k=0}^{\infty}\frac{1}{k!}\mathscr{D}^k(\bar{H}_0)(\bar{\mathbb{X}}^k)\Big)=\sum_{k=0}^{\infty}\frac{1}{k!}\psi\circ\mathrm{Proj}\big(\mathscr{D}^k(\bar{H}_0)(\bar{\mathbb{X}}^k)\big),$$

by linearity of ψ and Proj. Since the maps $\mathscr{D}^k(\bar{H}_0):(\mathbb{R}^d)^{\otimes k}\to\mathbb{R}^e$ are linear, the above equality takes the form

$$\psi\circ\mathrm{Proj}(\bar{H}_1)=\sum_{k=0}^{\infty}\langle\alpha^k,\bar{\mathbb{X}}^k\rangle_{(\mathbb{R}^d)^{\otimes k}},\tag{D.6}$$

where $\alpha^k\in(\mathbb{R}^d)^{\otimes k}$ is the coefficient of the linear map $\frac{1}{k!}\psi\circ\mathrm{Proj}\circ\mathscr{D}^k(\bar{H}_0)$ in the canonical basis. Let $\alpha=(\alpha^0,\dots,\alpha^k,\dots)$. Under assumption (A_2),

$$\begin{aligned}\sum_{k=0}^{\infty}\|\alpha^k\|^2_{(\mathbb{R}^d)^{\otimes k}}&\le\sum_{k=0}^{\infty}\sum_{1\le i_1,\dots,i_k\le d}\Big(\frac{1}{k!}\Big)^2\|\psi\|^2_{\mathrm{op}}\|F^{i_1}\star\cdots\star F^{i_k}(\bar{H}_0)\|^2\\&\le\|\psi\|^2_{\mathrm{op}}\sum_{k=0}^{\infty}\sum_{1\le i_1,\dots,i_k\le d}\Big(\frac{1}{k!}\Big)^2\Lambda_k(\mathbf{F})^2\le\|\psi\|^2_{\mathrm{op}}\sum_{k=0}^{\infty}\Big(\frac{d^k}{k!}\Lambda_k(\mathbf{F})\Big)^2<\infty.\end{aligned}$$

This shows that $\alpha \in \mathscr{T}$, and therefore, using (D.6), we conclude

$$\|z_T - \langle \alpha, S(\bar{X})\rangle_{\mathscr{T}}\| \leq \|\psi\|_{\mathrm{op}} \frac{c_1}{T}.$$

D.2.7 Proof of Theorem 5.7

Let

$$\mathscr{G} = \Big\{ g_\theta : (\mathbb{R}^d)^T \to \mathbb{R} \,|\, g_\theta(\mathbf{x}) = z_T, \theta \in \Theta \Big\}$$

be the function class of (discrete) RNN and

$$\mathscr{S} = \Big\{ \xi_{\alpha_\theta} : \mathscr{X} \to \mathbb{R} \,|\, \xi_{\alpha_\theta}(X) = \langle \alpha_\theta, S(\bar{X})\rangle_{\mathscr{T}}, \theta \in \Theta \Big\},$$

be the class of their RKHS embeddings, where α_θ is defined by (D.6). For any $\theta \in \Theta$, we let

$$\mathscr{R}_{\mathscr{G}}(\theta) = \mathbb{E}[\ell(\mathbf{y}, g_\theta(\mathbf{x}))], \quad \text{and} \quad \mathscr{R}_{\mathscr{S}}(\theta) = \mathbb{E}[\ell(\mathbf{y}, \xi_{\alpha_\theta}(\bar{X}))],$$

and denote by $\widehat{\mathscr{R}}_{n,\mathscr{G}}$ and $\widehat{\mathscr{R}}_{n,\mathscr{S}}$ the corresponding empirical risks. We also let $\theta^*_{\mathscr{G}}$, $\theta^*_{\mathscr{S}}$, $\widehat{\theta}_{n,\mathscr{G}}$, and $\widehat{\theta}_{n,\mathscr{S}}$ be the corresponding minimizers. We have

$$\begin{aligned}
&\mathbb{P}\big(\mathbf{y} \neq g_{\widehat{\theta}_{n,\mathscr{G}}}(\mathbf{x})\big) - \widehat{\mathscr{R}}_{n,\mathscr{G}}(\widehat{\theta}_{n,\mathscr{G}}) \\
&\quad \leq \mathbb{E}\big[\ell(\mathbf{y}, g_{\widehat{\theta}_{n,\mathscr{G}}}(\mathbf{x}))\big] - \widehat{\mathscr{R}}_{n,\mathscr{G}}(\widehat{\theta}_{n,\mathscr{G}}) \\
&\quad = \mathscr{R}_{\mathscr{G}}(\widehat{\theta}_{n,\mathscr{G}}) - \widehat{\mathscr{R}}_{n,\mathscr{G}}(\widehat{\theta}_{n,\mathscr{G}}) \\
&\quad = \mathscr{R}_{\mathscr{G}}(\widehat{\theta}_{n,\mathscr{G}}) - \mathscr{R}_{\mathscr{S}}(\widehat{\theta}_{n,\mathscr{G}}) + \mathscr{R}_{\mathscr{S}}(\widehat{\theta}_{n,\mathscr{G}}) - \widehat{\mathscr{R}}_{n,\mathscr{S}}(\widehat{\theta}_{n,\mathscr{G}}) + \widehat{\mathscr{R}}_{n,\mathscr{S}}(\widehat{\theta}_{n,\mathscr{G}}) - \widehat{\mathscr{R}}_{n,\mathscr{G}}(\widehat{\theta}_{n,\mathscr{G}}) \\
&\quad \leq \sup_\theta |\mathscr{R}_{\mathscr{G}}(\theta) - \mathscr{R}_{\mathscr{S}}(\theta)| + \sup_\theta |\mathscr{R}_{\mathscr{S}}(\theta) - \widehat{\mathscr{R}}_{n,\mathscr{S}}(\theta)| + \sup_\theta |\widehat{\mathscr{R}}_{n,\mathscr{G}}(\theta) - \widehat{\mathscr{R}}_{n,\mathscr{S}}(\theta)|.
\end{aligned}$$

Using Theorem 5.6, we have

$$\begin{aligned}
\sup_\theta |\mathscr{R}_{\mathscr{G}}(\theta) - \mathscr{R}_{\mathscr{S}}(\theta)| &= \sup_\theta \big|\mathbb{E}\big[\ell(\mathbf{y}, g_\theta(\mathbf{x})) - \ell(\mathbf{y}, \xi_{\alpha_\theta}(\bar{X}))\big]\big| \\
&\leq \sup_\theta \mathbb{E}\big[|\phi(\mathbf{y} g_\theta(\mathbf{x})) - \phi(\mathbf{y}\xi_{\alpha_\theta}(\bar{X}))|\big] \\
&\leq \sup_\theta \mathbb{E}\big[K_\ell |\mathbf{y}| \times |g_\theta(\mathbf{x}) - \xi_{\alpha_\theta}(\bar{X})|\big] \\
&\leq K_\ell \sup_\theta (\|\psi\|_{\mathrm{op}} c_{1,\theta}) \frac{1}{T} := \frac{c_2}{2T},
\end{aligned}$$

where $c_{1,\theta} = K_{f_\theta} e^{K_{f_\theta}} \big(L + \|f_\theta\|_\infty e^{K_{f_\theta}}\big)$ (the infinite norm $\|f_\theta\|_\infty$ is taken on the balls $\mathscr{B}_L$ and $\mathscr{B}_M$). One proves with similar arguments that

$$\sup_\theta |\widehat{\mathscr{R}}_{n,\mathscr{G}}(\theta) - \widehat{\mathscr{R}}_{n,\mathscr{S}}(\theta)| \leq \frac{c_2}{2T}.$$

Under the assumption of the theorem, there exists a ball $\mathscr{B} \subset \mathscr{H}$ of radius B such that $\mathscr{S} \subset \mathscr{B}$. This yields

$$\sup_\theta |\mathscr{R}_{\mathscr{S}}(\theta) - \widehat{\mathscr{R}}_{n,\mathscr{S}}(\theta)| \leq \sup_{\alpha \in \mathscr{T}, \|\alpha\|_{\mathscr{T}} \leq B} |\mathscr{R}_{\mathscr{B}}(\alpha) - \widehat{\mathscr{R}}_{n,\mathscr{B}}(\alpha)|,$$

where

$$\mathscr{R}_{\mathscr{B}}(\alpha) = \mathbb{E}[\ell(Y, \xi_\alpha(\bar{X}))] \quad \text{and} \quad \widehat{\mathscr{R}}_{n,\mathscr{B}}(\alpha) = \frac{1}{n}\sum_{i=1}^{n} \ell(Y^{(i)}, \xi_\alpha(\bar{X})).$$

We now have reached a familiar situation where the supremum is over a ball in a RKHS. It is known (see, e.g., Bartlett and Mendelson, 2002, Theorem 8) that with probability at least $1-\delta$,

$$\sup_{\alpha\in\mathscr{T}, \|\alpha\|_{\mathscr{T}}\leq B} |\mathscr{R}_{\mathscr{B}}(\alpha) - \widehat{\mathscr{R}}_{n,\mathscr{B}}(\alpha)| \leq 4K_\ell \mathbb{E}\mathrm{Rad}_n(\mathscr{B}) + 2BK_\ell(1-L)^{-1}\sqrt{\frac{\log(1/\delta)}{2n}},$$

where $\mathrm{Rad}_n(\mathscr{B})$ denotes the Rademacher complexity of $\mathscr{B}$. Observe that we have used the fact that the loss is bounded by $K_\ell B(1-L)^{-1}$ since, for any $\xi_\alpha \in \mathscr{B}$, by the Cauchy-Schwartz inequality,

$$\ell(\mathbf{y}, \xi_\alpha(\bar{X})) = \phi(\mathbf{y}\langle \alpha, S(\bar{X})\rangle_{\mathscr{T}}) \leq K_\ell |\mathbf{y}\langle \alpha, S(\bar{X})\rangle_{\mathscr{T}}| \leq K_\ell \|\alpha\|_{\mathscr{T}} \|S(\bar{X})\|_{\mathscr{T}} \leq 2K_\ell B(1-L)^{-1}.$$

Finally, the proof follows by noting that Rademacher complexity of $\mathscr{B}$ is bounded by

$$\mathrm{Rad}_n(\mathscr{B}) \leq \frac{2B}{n}\sqrt{\sum_{i=1}^{n} K(X^{(i)}, X^{(i)})} = \frac{2B}{n}\sqrt{\sum_{i=1}^{n} \|S(\bar{X}^{(i)})\|_{\mathscr{T}}^2} \leq \frac{4B(1-L)^{-1}}{\sqrt{n}}.$$

D.2.8 Proof of Theorem 5.8

Let

$$\mathscr{G} = \left\{ g_\theta : (\mathbb{R}^d)^T \to (\mathbb{R}^p)^T \,|\, g_\theta(\mathbf{x}) = \big(z_1, \ldots, z_T\big), \theta \in \Theta \right\}$$

be the function class of discrete RNN in a sequential setting. Let

$$\mathscr{S} = \left\{ \Gamma_\theta : \mathscr{X} \to (\mathbb{R}^p)^T \,|\, \Gamma_\theta(X) = \big(\Xi_\theta(\tilde{X}_{[1]}), \ldots, \Xi_\theta(\tilde{X}_{[T]})\big) \right\},$$

be the class of their RKHS embeddings, where $\tilde{X}_{[j]}$ is the path equal to X on $[0, j/T]$ and then constant on $[j/T, 1]$ (see Figure D.1). For any $X \in \mathscr{X}$,

$$\Xi_\theta(a) = \begin{pmatrix} \langle \alpha_{1,\theta}, S(\bar{X})\rangle_{\mathscr{T}} \\ \vdots \\ \langle \alpha_{p,\theta}, S(\bar{X})\rangle_{\mathscr{T}} \end{pmatrix} = \begin{pmatrix} \xi_{\alpha_{1,\theta}}(X) \\ \vdots \\ \xi_{\alpha_{p,\theta}}(X) \end{pmatrix} \in \mathbb{R}^p,$$

where $(\alpha_{1,\theta}, \ldots, \alpha_{p,\theta})^\top \in (\mathscr{T})^p$ are the coefficients of the linear maps $\frac{1}{k!}\psi \circ \mathrm{Proj} \circ \mathscr{D}^k(\bar{H}_0) : (\mathbb{R}^d)^{\otimes k} \to \mathbb{R}^p$, $k \geq 0$, in the canonical basis, where $\mathscr{D}^k$ is defined by (D.5).

We start the proof as in Theorem 5.7, until we obtain

$$\begin{aligned}\mathscr{R}_{\mathscr{G}}(\widehat{\theta}_{n,\mathscr{G}}) - \widehat{\mathscr{R}}_{n,\mathscr{G}}(\widehat{\theta}_{n,\mathscr{G}}) \leq \sup_\theta |\mathscr{R}_{\mathscr{G}}(\theta) - \mathscr{R}_{\mathscr{S}}(\theta)| + \sup_\theta |\mathscr{R}_{\mathscr{S}}(\theta) - \widehat{\mathscr{R}}_{n,\mathscr{S}}(\theta)| \\ + \sup_\theta |\widehat{\mathscr{R}}_{n,\mathscr{G}}(\theta) - \widehat{\mathscr{R}}_{n,\mathscr{S}}(\theta)|.\end{aligned}$$

By definition of the loss, for any $\theta \in \Theta$,

$$\begin{aligned}
|\mathscr{R}_{\mathscr{G}}(\theta) - \mathscr{R}_{\mathscr{S}}(\theta)| &= \Big|\mathbb{E}\big[\ell\big(\mathbf{y}, g_\theta(\mathbf{x})\big) - \ell\big(\mathbf{y}, \Gamma_\theta(X)\big)\big]\Big| \\
&\leq \mathbb{E}\Big[\Big|\frac{1}{T}\sum_{j=1}^{T}\big(\|y_j - z_j\|^2 - \|y_j - \Xi_\theta(\tilde{X}_{[j]})\|^2\big)\Big|\Big] \\
&\leq \mathbb{E}\Big[\frac{1}{T}\sum_{j=1}^{T}\big|\big\langle z_j + \Xi_\theta(\tilde{X}_{[j]}) - 2y_j, z_j - \Xi_\theta(\tilde{X}_{[j]})\big\rangle\big|\Big] \\
&\leq \mathbb{E}\Big[\frac{1}{T}\sum_{j=1}^{T}\|z_j + \Xi_\theta(\tilde{X}_{[j]}) - 2y_j\| \times \|z_j - \Xi_\theta(\tilde{X}_{[j]})\|\Big] \\
&\qquad \text{(by the Cauchy-Schwartz inequality).}
\end{aligned}$$

According to inequality (5.13), one has

$$\|z_j - \Xi_\theta(\tilde{X}_{[j]})\| \leq \|\psi\|_{\mathrm{op}}\frac{c_{1,\theta}}{T},$$

where $c_{1,\theta} = K_{f_\theta}e^{K_{f_\theta}}\big(L + \|f_\theta\|_\infty e^{K_{f_\theta}}\big)$. Moreover,

$$\big\|\Xi_\theta(\tilde{X}_{[j]})\big\|^2 = \sum_{\ell=1}^{p}\big|\langle\alpha_{\ell,\theta}, S(\tilde{X}_{[j]})\rangle_{\mathscr{T}}\big|^2 \leq \sum_{\ell=1}^{p}\|\alpha_{\ell,\theta}\|_{\mathscr{T}}^2\|S(\tilde{X}_{[j]})\|_{\mathscr{T}}^2 \leq pB^2\big(2(1-L)^{-1}\big)^2,$$

since $\|S(\tilde{X}_{[j]})\|_{\mathscr{T}} = \|S_{[0,j/T]}(\bar{X})\|_{\mathscr{T}} \leq \|S(\bar{X})\|_{\mathscr{T}}$. This yields

$$\begin{aligned}
\|z_j + \Xi_\theta(\tilde{X}_{[j]}) - 2y_j\| &\leq \|z_j\| + \|\Xi_\theta(\tilde{X}_{[j]})\| + 2\|y_j\| \\
&\leq \|\psi\|_{\mathrm{op}}\|f_\theta\|_\infty + 2\sqrt{p}B(1-L)^{-1} + 2K_y.
\end{aligned}$$

Finally,

$$\sup_\theta|\mathscr{R}_{\mathscr{G}}(\theta) - \mathscr{R}_{\mathscr{S}}(\theta)| \leq \frac{c_3}{2T},$$

where $c_3 = \sup_\theta\big(c_{1,\theta} + \|\psi\|_{\mathrm{op}}\|f_\theta\|_\infty\big) + 2\sqrt{p}B(1-L)^{-1} + 2K_y$. One proves with similar arguments that

$$\sup_\theta|\widehat{\mathscr{R}}_{n,\mathscr{G}}(\theta) - \widehat{\mathscr{R}}_{n,\mathscr{S}}(\theta)| \leq \frac{c_3}{2T}.$$

We now turn to the term $\sup_\theta|\mathscr{R}_{\mathscr{S}}(\theta) - \widehat{\mathscr{R}}_{n,\mathscr{S}}(\theta)|$. We have

$$\begin{aligned}
&\mathscr{R}_{\mathscr{S}}(\theta) - \widehat{\mathscr{R}}_{n,\mathscr{S}}(\theta) \\
&\quad = \mathbb{E}[\ell(\mathbf{y}, \Gamma_\theta(X))] - \frac{1}{n}\sum_{i=1}^{n}\ell(\mathbf{y}^{(i)}, \Gamma_\theta(X^{(i)})) \\
&\quad = \frac{1}{T}\sum_{j=1}^{T}\Big(\mathbb{E}[\|y_j - \Xi_\theta(\tilde{X}_{[j]})]\|^2 - \frac{1}{n}\sum_{i=1}^{n}\big\|y_j^{(i)} - \Xi_\theta(\tilde{X}_{[j]}^{(i)})\big\|^2\Big).
\end{aligned}$$

Therefore,

$$\sup_\theta |\mathscr{R}_{\mathscr{S}}(\theta) - \widehat{\mathscr{R}}_{n,\mathscr{S}}(\theta)| \leq \frac{1}{T}\sum_{j=1}^{T} \sup_\theta \Big| \mathbb{E}[\|y_j - \Xi_\theta(\tilde{X}_{[j]})]\|^2 - \frac{1}{n}\sum_{i=1}^{n} \|y_j^{(i)} - \Xi_\theta(\tilde{X}_{[j]}^{(i)})\|^2 \Big|.$$

Note that for a fixed j, the pairs $(\tilde{X}_{[j]}^{(i)}, y_j^{(i)})$ are i.i.d. Under the assumptions of the theorem, there exists a ball $\mathscr{B} \subset \mathscr{H}$ such that for any $1 \leq \ell \leq p$, $\theta \in \Theta$, $\xi_{\alpha_{\ell,\theta}} \in \mathscr{B}$. We denote by $\mathscr{B}_p$ the sum of p such spaces, that is,

$$\mathscr{B}_p = \big\{ f_\alpha : \mathscr{X} \to \mathbb{R}^p \,|\, f_\alpha(X) = (f_{\alpha_1}(X), \ldots, f_{\alpha_p}(X))^\top, f_{\alpha_\ell} \in \mathscr{B} \big\}.$$

Clearly, $\Xi_\theta \in \mathscr{B}_p$, and it follows that

$$\begin{aligned} &\sup_\theta \Big| \mathbb{E}[\|y_j - \Xi_\theta(\tilde{X}_{[j]})]\|^2 - \frac{1}{n}\sum_{i=1}^{n} \|y_j^{(i)} - \Xi_\theta(\tilde{X}_{[j]}^{(i)})\|^2 \Big| \\ &\qquad \leq \sup_{f_\alpha \in \mathscr{B}_p} \Big| \mathbb{E}\big[\|y_j - f_\alpha(\tilde{X}_{[j]})\|^2\big] - \frac{1}{n}\sum_{i=1}^{n} \|y_j^{(i)} - f_\alpha(\tilde{X}_{[j]}^{(i)})\|^2 \Big|. \end{aligned}$$

We have once again reached a familiar situation, which can be dealt with by an easy extension of Bartlett and Mendelson (2002, Theorem 12). For any $f_\alpha \in \mathscr{B}_p$, let $\tilde{\phi} \circ f_\alpha : \mathscr{X} \times \mathbb{R}^p : (X, y) \mapsto \|y - f_\alpha(X)\|^2 - \|y\|^2$. Then, $\tilde{\phi} \circ f_\alpha$ is upper bounded by

$$\begin{aligned} |\tilde{\phi} \circ f_\alpha(X, y)| = \big| \|y - f_\alpha(X)\|^2 - \|y\|^2 \big| &\leq \|f_\alpha(X)\| \big(\|f_\alpha(X)\| + 2\|y\| \big) \\ &\leq 2\sqrt{p}B(1-L)^{-1}(2\sqrt{p}B(1-L)^{-1} + 2K_y) \\ &\leq 4pB(1-L)^{-1}(B(1-L)^{-1} + K_y). \end{aligned}$$

Let $c_4 = B(1-L)^{-1} + K_y$ and $c_5 = 4pB(1-L)^{-1}c_4 + K_y^2$. Then with probability at least $1-\delta$,

$$\sup_{f_\alpha \in \mathscr{B}_p} \Big| \mathbb{E}\big[\|y_j - f_\alpha(\tilde{X}_{[j]})\|\big] - \frac{1}{n}\sum_{i=1}^{n} \|y_j^{(i)} - f_\alpha(\tilde{X}_{[j]}^{(i)})\| \Big| \leq \mathrm{Rad}_n(\tilde{\phi} \circ \mathscr{B}_p) + \sqrt{\frac{2c_5 \log(1/\delta)}{n}},$$

where $\tilde{\phi} \circ \mathscr{B}_p = \big\{ (X, y) \mapsto \tilde{\phi} \circ f_\alpha(X, y) | f_\alpha \in \mathscr{B}_p \big\}$. Elementary computations on Rademacher complexities yield

$$\mathrm{Rad}_n(\tilde{\phi} \circ \mathscr{B}_p) \leq 2pc_4 \mathrm{Rad}_n(\mathscr{B}) \leq \frac{8pc_4 B(1-L)^{-1}}{\sqrt{n}},$$

which concludes the proof.

D.3 Differentiation with higher-order tensors

D.3.1 Definition

We define the generalization of matrix product between square tensors of order k and ℓ.

Definition D.2. *Let $a \in (\mathbb{R}^e)^{\otimes k}$, $b \in (\mathbb{R}^e)^{\otimes \ell}$, $p \in \{1, \ldots, k\}$, $q \in \{1, \ldots, \ell\}$. Then the tensor*

dot product along (p,q), denoted by $a \odot_{p,q} b \in (\mathbb{R}^e)^{\otimes(k+\ell-2)}$, is defined by

$$(a \odot_{p,q} b)_{(i_1,\dots,i_{k-1},j_1,\dots,j_{\ell-1})} = \sum_{j=1}^{e} a_{(i_1,\dots,i_{p-1},j,i_p,\dots,i_{k-1})} b_{(j_1,\dots,j_{q-1},j,j_q,\dots,j_{\ell-1})}.$$

This operation just consists in computing $a \otimes b$, and then summing the pth coordinate of a with the qth coordinate of b. The $\odot$ operator is not associative. To simplify notation, we take the convention that it is evaluated from left to right, that is, we write $a \odot b \odot c$ for $(a \odot b) \odot c$.

Definition D.3. *Let $a \in (\mathbb{R}^e)^{\otimes k}$. For a given permutation π of $\{1,\dots,k\}$, we denote by $\pi(a)$ the permuted tensor in $(\mathbb{R}^e)^{\otimes k}$ such that*

$$\pi(a)_{(i_1,\dots,i_k)} = a_{(i_{\Pi(1)},\dots,i_{\Pi(k)})}.$$

Example D.1. *If A is a matrix, then $A^T = \pi(A)$, with π defined by $\pi(1) = 2, \pi(2) = 1$.*

D.3.2 Computation rules

We need to obtain two computation rules for the tensor dot product: bounding the norm (Lemma D.11) and differentiating (Lemma D.12).

Lemma D.11. *Let $a \in (\mathbb{R}^e)^{\otimes k}$, $b \in (\mathbb{R}^e)^{\otimes \ell}$. Then, for all p, q,*

$$\|a \odot_{p,q} b\|_{(\mathbb{R}^e)^{\otimes k+\ell-2d}} \leq \|a\|_{(\mathbb{R}^e)^{\otimes k}} \|b\|_{(\mathbb{R}^e)^{\otimes \ell}}.$$

Proof. By the Cauchy-Schwartz inequality,

$$\begin{aligned}
&\|a \odot_{p,q} b\|^2_{(\mathbb{R}^e)^{\otimes k+\ell-2}} \\
&\quad= \sum_{1\leq i_1,\dots,i_{k-1},j_1,\dots,j_{\ell-1}\leq e} (a \odot_{p,q} b)^2_{(i_1,\dots,i_{k-1},j_1,\dots,j_{\ell-1})} \\
&\quad= \sum_{1\leq i_1,\dots,i_{k-1},j_1,\dots,j_{\ell-1}\leq e} \Big(\sum_{1\leq j\leq e} a_{(i_1,\dots,i_{p-1},j,i_p,\dots,i_{k-1})} b_{(j_1,\dots,j_{q-1},j,j_q,\dots,j_{\ell-1})} \Big)^2 \\
&\quad\leq \sum_{i_1,\dots,i_{k-1},j_1,\dots,j_{\ell-1}} \Big(\sum_j a^2_{(i_1,\dots,i_{p-1},j,i_p,\dots,i_{k-1})} \Big) \Big(\sum_j b^2_{(j_1,\dots,j_{q-1},j,j_q,\dots,j_{\ell-1})} \Big) \\
&\quad\leq \sum_{i_1,\dots,i_{k-1},j} a^2_{(i_1,\dots,i_{p-1},j,i_p,\dots,i_{k-1})} \sum_{j_1,\dots,j_{\ell-1},j} b^2_{(j_1,\dots,j_{q-1},j,j_q,\dots,j_{\ell-1})} \\
&\quad\leq \|a\|^2_{(\mathbb{R}^e)^{\otimes k}} \|b\|^2_{(\mathbb{R}^e)^{\otimes \ell}}.
\end{aligned}$$

□

Lemma D.12. *Let $A : \mathbb{R}^e \to (\mathbb{R}^e)^{\otimes k}$, $B : \mathbb{R}^e \to (\mathbb{R}^e)^{\otimes \ell}$ be smooth vector fields, $p \in \{1,\dots,k\}$, $q \in \{1,\dots,\ell\}$. Let $A \odot_{p,q} B : \mathbb{R}^e \to (\mathbb{R}^e)^{\otimes k+\ell-2}$ be defined by $A \odot_{p,q} B(h) = A(h) \odot_{p,q} B(h)$. Then there exists a permutation π such that*

$$J(A \odot_{p,q} B) = \pi(J(A) \odot_{p,q} B) + A \odot_{p,q} J(B).$$

Proof. The left-hand side takes the form

$$(J(A \odot_{p,q} B))_{i_1,\dots,i_{k-1},j_1,\dots,j_{\ell-1},m} = \sum_j \Big[\frac{\partial A}{\partial h_m}_{(i_1,\dots,i_{p-1},j,i_p,\dots,i_{k-1})} B_{(j_1,\dots,j_{q-1},j,j_q,\dots,j_{\ell-1})} + A_{(i_1,\dots,i_{p-1},j,i_p,\dots,i_{k-1})} \frac{\partial B}{\partial h_m}_{(j_1,\dots,j_{q-1},j,j_q,\dots,j_{\ell-1})} \Big].$$

The first term of the right-hand side writes

$$(J(A) \odot_{p,q} B)_{i_1,\dots,i_{k-1},m,j_1,\dots,j_{\ell-1}} = \sum_j \Big[\frac{\partial A}{\partial h_m}_{(i_1,\dots,i_{p-1},j,i_p,\dots,i_{k-1})} B_{(j_1,\dots,j_{q-1},j,j_q,\dots,j_{\ell-1})} \Big],$$

and the second one

$$(A \odot_{p,q} J(B))_{i_1,\dots,i_{k-1},j_1,\dots,j_{\ell-1},m} = \sum_j \Big[A_{(i_1,\dots,i_{p-1},j,i_p,\dots,i_{k-1})} \frac{\partial B}{\partial h_m}_{(j_1,\dots,j_{q-1},j,j_q,\dots,j_{\ell-1})} \Big].$$

Let us introduce the permutation π which keeps the first $(k-1)$ axes unmoved, and rotates the remaining ℓ ones such that the last axis ends up in kth position. Then

$$\pi(J(A) \odot_{p,q} B)_{i_1,\dots,i_{k-1},j_1,\dots,j_{\ell-1},m} = \sum_j \Big[\frac{\partial A}{\partial h_m}_{(i_1,\dots,i_{p-1},j,i_p,\dots,i_{k-1})} B_{(j_1,\dots,j_{q-1},j,j_q,\dots,j_{\ell-1})} \Big].$$

Hence $J(A \odot_{p,q} B) = \pi(J(A) \odot_{p,q} B) + A \odot_{p,q} J(B)$, which concludes the proof. □

The following two lemmas show how to compose the Jacobian and the tensor dot operations with permutations. Their proofs follow elementary operations and are therefore omitted.

Lemma D.13. *Let $A : \mathbb{R}^e \to (\mathbb{R}^e)^{\otimes k}$ and π a permutation of $\{1,\dots,k\}$. Then there exists a permutation $\tilde{\pi}$ of $\{1,\dots,k+1\}$ such that*

$$J(\pi(A)) = \tilde{\pi}(J(A)).$$

Lemma D.14. *Let $a \in (\mathbb{R}^e)^{\otimes k}$, $b \in (\mathbb{R}^e)^{\otimes \ell}$, $p \in \{1,\dots,k\}$, $q \in \{1,\dots,\ell\}$, π a permutation of $\{1,\dots,k\}$. Then there exists $\tilde{p} \in \{1,\dots,k\}$, $\tilde{q} \in \{1,\dots,\ell\}$, and a permutation $\tilde{\pi}$ of $\{1,\dots,k+\ell-2\}$ such that*

$$\pi(a) \odot_{p,q} b = \tilde{\pi}(a \odot_{\tilde{p},\tilde{q}} b).$$

The following result is a generalization of Lemma D.12 to the case of a dot product of several tensors.

Lemma D.15. *For $\ell \in \{1,\dots,k\}$, $n_\ell \in \mathbb{N}$, let $A_\ell : \mathbb{R}^e \to (\mathbb{R}^e)^{\otimes n_\ell}$ be smooth tensor fields. For any $(p_\ell)_{1\le\ell\le k-1}$ and $(q_\ell)_{1\le\ell\le k-1}$ such that $p_\ell \in \{1,\dots,n_\ell\}$, $q_\ell \in \{1,\dots,n_{\ell+1}\}$, there exist k permutations $(\pi_\ell)_{1\le\ell\le k}$ such that*

$$J(A_1 \odot_{p_1,q_1} A_2 \odot_{p_2,q_2} \cdots \odot_{p_{k-1},q_{k-1}} A_k) = \sum_{\ell=1}^{k} \pi_\ell \left[A_1 \odot A_2 \odot \cdots \odot J(A_\ell) \odot \cdots \odot A_k \right],$$

where the dot products of the right-hand side are along some axes that are not specify for simplicity.

Proof. The proof is done by induction on k. The formula for $k = 1$ is straightforward. Assume that the formula is true at order k. As before, we do not specify indexes for tensor dot products as we are only interested in their existence. By Lemma D.14, we have

$$
\begin{aligned}
&J(A_1 \odot \cdots \odot A_{k+1}) \\
&\quad = J((A_1 \odot \cdots \odot A_k) \odot A_{k+1}) \\
&\quad = \pi(J(A_1 \odot \cdots \odot A_k) \odot A_{k+1}) + A_1 \odot \cdots \odot A_k \odot J(A_{k+1}) \\
&\quad = \pi\left[\sum_{\ell=1}^{k} \pi_\ell \left[A_1 \odot A_2 \odot \cdots \odot J(A_\ell) \odot \cdots \odot A_k\right] \odot A_{k+1}\right] + A_1 \odot \cdots \odot A_k \odot J(A_{k+1}) \\
&\quad = \pi\left[\sum_{\ell=1}^{k} \tilde{\pi}_\ell \left[A_1 \odot A_2 \odot \cdots \odot J(A_\ell) \odot \cdots \odot A_k \odot A_{k+1}\right]\right] + A_1 \odot \cdots \odot A_k \odot J(A_{k+1}) \\
&\quad = \sum_{\ell=1}^{k} \hat{\pi}_\ell \left[A_1 \odot A_2 \odot \cdots \odot J(A_\ell) \odot \cdots \odot A_k \odot A_{k+1}\right] + A_1 \odot \cdots \odot A_k \odot J(A_{k+1}) \\
&\qquad (\text{where } \hat{\pi} = \pi \circ \tilde{\pi}) \\
&\quad = \sum_{\ell=1}^{k+1} \hat{\pi}_\ell \left[A_1 \odot A_2 \odot \cdots \odot J(A_\ell) \odot \cdots \odot A_k \odot A_{k+1}\right].
\end{aligned}
$$

□

D.4 Experimental details

All the code to reproduce the experiments is available on GitHub at https://github.com/afermanian/rnn-kernel. Our experiments are based on the PyTorch (Paszke et al., 2019) framework. When not specified, the default parameters of PyTorch are used.

Convergence of the Taylor expansion. For Figure 5.1, 10^3 random RNN with 2 hidden units are generated, with the default weight initialization. The activation is either the logistic or the hyperbolic tangent. In Figure 5.1b, only the results with the logistic activation are plotted. The process X is taken as a 2-dimensional spiral. The reference solution to the ODE (5.3) is computed with a numerical integration method from SciPy (Virtanen et al., 2020, `scipy.integrate.solve_ivp` with the 'LSODA' method). The signature in the step-N Taylor expansion is computed with the package Signatory (Kidger and Lyons, 2020).

The step-N Taylor expansion requires computing higher-order derivatives of tensor fields (up to order N). This is a highly non-trivial task since standard deep learning frameworks are optimized for first-order differentiation only. We refer to, for example, Kelly et al. (2020), for a discussion on higher-order differentiation in the context of a deep learning framework. To compute it efficiently, we manually implement forward-mode higher-order automatic differentiation for the operations needed in our context (described in Appendix D.3). A more efficient and general approach is left for future work. Our code is optimized for GPU.

Penalization on a toy example. For Figure 5.2, the RNN is taken with 32 hidden units and hyperbolic tangent activation. The data are 50 examples of spirals, sampled at 100 points and labeled ± 1 according to their rotation direction. We do not use batching and the loss is taken as the cross entropy. It is trained for 200 epochs with Adam (Kingma and Ba, 2015)

with an initial learning rate of 0.1. The learning rate is divided by 2 every 40 epochs. For the penalized RNN, the RKHS norm is truncated at $N = 3$ and the regularization parameter is selected at $\lambda = 0.1$. Earlier experiments show that this order of magnitude is sensible. We do not perform hyperparameter optimization since our goal is not to achieve high performance. The initial hidden state h_0 is learned (for simplicity of presentation, our theoretical results were written with $h_0 = 0$ but they extend to this case). The accuracy is computed on a test set of size 1000. We generate adversarial examples using 50 steps of projected gradient descent (following Bietti et al., 2019). The whole methodology (data generation + training) is repeated 20 times. The average training time on a Tesla V100 GPU for the RNN is 8.5 seconds and for the penalized RNN 12 seconds.

Figure 5.3 is obtained by selecting randomly one run among the 20 of Figure 5.2.

Libraries. We use PyTorch (Paszke et al., 2019) as our overall framework, Signatory (Kidger and Lyons, 2020) to compute the signatures, and SciPy (Virtanen et al., 2020) for ODE integration. We use Sacred (Greff et al., 2017) for experiment management. The links and licences for the assets are given in the following table:

Name	Homepage link	License
PyTorch	GitHub repository	BSD-style License
Sacred	GitHub repository	MIT License
SciPy	GitHub repository	BSD 3-Clause "New" or "Revised" License
Signatory	GitHub repository	Apache License 2.0

Table D.1 – Links and licences of the libraries

Bibliography

Bartlett, P. L., and Mendelson, S. (2002). Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, *3*, 463–482.

Bietti, A., Mialon, G., Chen, D., and Mairal, J. (2019). A kernel perspective for regularizing deep neural networks. In K. Chaudhuri and R. Salakhutdinov (Eds.), *Proceedings of the 36th international conference on machine learning* (pp. 664–674).

Greff, K., Klein, A., Chovanec, M., Hutter, F., and Schmidhuber, J. (2017). The sacred infrastructure for computational research. In K. Huff, D. Lippa, D. Niederhut, and M. Pacer (Eds.), *Proceedings of the 16th python in science conference* (pp. 49–56).

Kelly, J., Bettencourt, J., Johnson, M. J., and Duvenaud, D. K. (2020). Learning differential equations that are easy to solve. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), *Advances in neural information processing systems* (pp. 4370–4380). Curran Associates, Inc.

Kidger, P., and Lyons, T. (2020). Signatory: differentiable computations of the signature and logsignature transforms, on both CPU and GPU. *arXiv:2001.00706*. https://github.com/patrick-kidger/signatory

Kingma, D. P., and Ba, J. (2015). Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*.

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

Lyons, T., Caruana, M., and Lévy, T. (2007). *Differential equations driven by rough paths* (Vol. 1908). Springer.

Minai, A. A., and Williams, R. D. (1993). On the derivatives of the sigmoid. *Neural Networks*, *6*, 845–853.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* (pp. 8024–8035). Curran Associates, Inc.

Riordan, J. (1958). *An introduction to combinatorial analysis*. John Wiley & Sons.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ... SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272.

Appendix E

Supplementary material of Chapter 6

Contents

E.1 Proof of Theorem 6.7

Before beginning the proof, we prove a relation similar to Chen's relation (Proposition 6.1, (i)) for the insertion operator. First, for any $n \geq 1$, $p \in \{1, \dots, n+1\}$, $X \in BV(\mathbb{R}^d)$, the insertion operator restricted to an interval $[s,t] \subset [0,1]$ is defined by

$$\mathscr{L}^n_{p,X;[s,t]}(y) = \int_{(u_1,\dots,u_n)\in\Delta_{n;[s,t]}} dX_{u_1} \otimes \cdots \otimes dX_{u_{p-1}} \otimes y \otimes dX_{u_p} \otimes \cdots \otimes dX_{u_n}.$$

Lemma E.1. *Let $n \geq 1$, $p \in \{1, \dots, n+1\}$, $X \in BV(\mathbb{R}^d)$, $u, v, w \in [0,1]$ such that $u < v < w$. Then,*

$$\mathscr{L}^n_{p,X,[u,w]}(y) = \sum_{k=0}^{p-1} \mathbf{X}^k_{[u,v]} \otimes \mathscr{L}^{n-k}_{p-k,X;[v,w]}(y) + \sum_{k=p}^{n} \mathscr{L}^k_{p,X;[u,v]}(y) \otimes \mathbf{X}^{n-k}_{[v,w]}.$$

Proof. This formula is obtained by splitting the integration domain in two parts, with a running index k corresponding to the number of integration variables in each part of the interval:

$$\begin{aligned}
\mathscr{L}^n_{p,X;[u,w]}(y) &= \int\cdots\int_{u\leq u_1\leq\cdots\leq u_n\leq w} dX_{u_1} \otimes \cdots \otimes dX_{u_{p-1}} \otimes y \otimes \cdots \otimes dX_{u_n} \\
&= \sum_{k=0}^{n} \int\cdots\int_{u\leq u_1\leq\cdots\leq u_k\leq v\leq u_{k+1}\leq\cdots\leq u_n\leq w} dX_{u_1} \otimes \cdots \otimes dX_{u_{p-1}} \otimes y \otimes \cdots \otimes dX_{u_n} \\
&= \sum_{k=0}^{n} \int_{(u_1,\dots,u_k)\in\Delta_{k;[u,v]}} \int_{(u_{k+1},\dots,u_n)\in\Delta_{n-k;[v,w]}} dX_{u_1} \otimes \cdots \otimes dX_{u_{p-1}} \otimes y \otimes \cdots \otimes dX_{u_n}.
\end{aligned}$$

To simplify this expression, we need to separate the cases when $k \leq p-1$ from the cases when $k > p$, which gives

$$
\begin{aligned}
&\mathscr{L}^n_{p,X;[u,w]}(y) \\
&\quad = \sum_{k=0}^{p-1} \int_{(u_1,\dots,u_k)\in\Delta_{k;[u,v]}} \int_{(u_{k+1},\dots,u_n)\in\Delta_{n-k;[v,w]}} dX_{u_1}\otimes\cdots\otimes dX_{u_{p-1}}\otimes y\otimes\cdots\otimes dX_{u_n} \\
&\qquad + \sum_{k=p}^{n} \int_{(u_1,\dots,u_k)\in\Delta_{k;[u,v]}} \int_{(u_{k+1},\dots,u_n)\in\Delta_{n-k;[v,w]}} dX_{u_1}\otimes\cdots\otimes dX_{u_{p-1}}\otimes y\otimes\cdots\otimes dX_{u_n} \\
&\quad = \sum_{k=0}^{p-1} \Big(\int_{(u_1,\dots,u_k)\in\Delta_{k;[u,v]}} dX_{u_1}\otimes\cdots\otimes dX_{u_k} \Big) \\
&\qquad\qquad \otimes \Big(\int_{(u_{k+1},\dots,u_n)\in\Delta_{n-k;[v,w]}} dX_{u_{k+1}}\otimes\cdots\otimes dX_{u_{p-1}}\otimes y\otimes\cdots\otimes dX_{u_n} \Big) \\
&\qquad + \sum_{k=p}^{n} \Big(\int_{(u_1,\dots,u_k)\in\Delta_{k;[u,v]}} dX_{u_1}\otimes\cdots\otimes dX_{u_{p-1}}\otimes y\otimes\cdots\otimes dX_{u_k} \Big) \\
&\qquad\qquad \otimes \Big(\int_{(u_{k+1},\dots,u_n)\in\Delta_{n-k;[v,w]}} dX_{u_{k+1}}\otimes\cdots\otimes dX_{u_n} \Big) \\
&\quad = \sum_{k=0}^{p-1} \mathbf{X}^k_{[u,v]} \otimes \mathscr{L}^{n-k}_{p-k,X;[v,w]}(y) + \sum_{k=p}^{n} \mathscr{L}^k_{p,X;[u,v]}(y) \otimes \mathbf{X}^{n-k}_{[v,w]}.
\end{aligned}
$$

□

The first step of the proof is to use the two Chen's relations (Proposition 6.1, (i), and Lemma E.1) to split both the signature $\mathbf{X}^{n+1}$ and the insertion operator $\mathscr{L}^n_{p,X}(y)$ on the intervals $[0, t_{i-1}]$, $[t_{i-1}, t_i]$, and $[t_i, 1]$. Concerning signatures, a straightforward extension of Proposition 6.1 yields

$$
\mathbf{X}^{n+1} = \sum_{\substack{n_1+n_2+n_3=n+1 \\ 0\leq n_1,n_2,n_3\leq n+1}} \mathbf{X}^{n_1}_{[0,t_{i-1}]} \otimes \mathbf{X}^{n_2}_{[t_{i-1},t_i]} \otimes \mathbf{X}^{n_3}_{[t_i,1]}. \tag{E.1}
$$

Concerning the insertion operator, it is necessary to split the sum depending on the location of p, in a similar way to the proof of Lemma E.1. Let $\mathscr{S}$ denotes the set

$$
\mathscr{S} = \big\{(n_1,n_2,n_3) \in \{0,\dots,n+1\} | n_1+n_2+n_3 = n+1\big\}.
$$

For a fixed $p \in \{1,\dots,n+1\}$, we use the following partition, illustrated in Figure E.1:

$$
\begin{aligned}
\mathscr{S} = &\big\{(n_1,n_2,n_3) \in \mathscr{S} | n_1 \geq p\big\} \cup \big\{(n_1,n_2,n_3) \in \mathscr{S} | n_1 < p,\ n_1+n_2 \geq p\big\} \\
&\cup \big\{(n_1,n_2,n_3) \in \mathscr{S} | n_1+n_2 < p\big\}.
\end{aligned}
$$

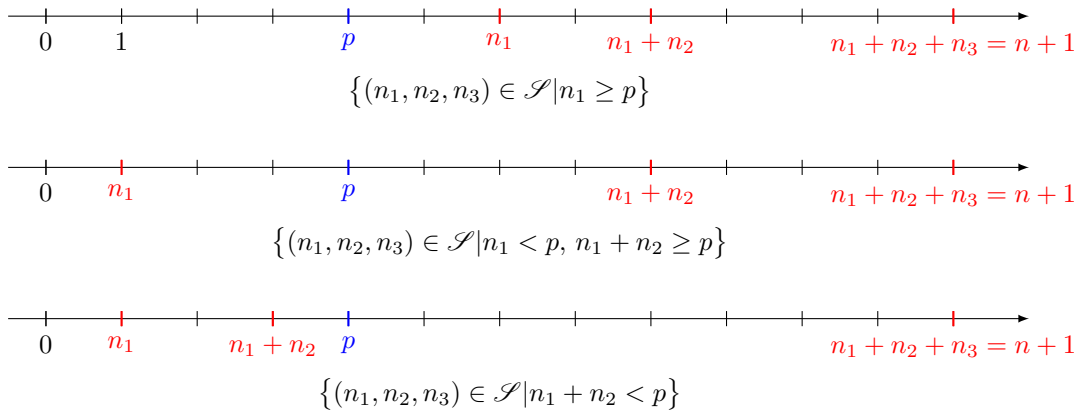


Figure E.1 – Partition of $\mathscr{S}$.

We then obtain

$$\begin{aligned}\mathscr{L}^n_{p,X}(y) = &\sum_{\substack{(n_1,n_2,n_3)\in\mathscr{S}\\ p\leq n_1}} \mathscr{L}^{n_1-1}_{p,X;[0,t_{i-1}]}(y)\otimes\mathbf{X}^{n_2}_{[t_{i-1},t_i]}\otimes\mathbf{X}^{n_3}_{[t_i,1]}\\ &+\sum_{\substack{(n_1,n_2,n_3)\in\mathscr{S}\\ n_1<p\leq n_1+n_2}} \mathbf{X}^{n_1}_{[0,t_{i-1}]}\otimes\mathscr{L}^{n_2-1}_{p-n_1,X;[t_{i-1},t_i]}(y)\otimes\mathbf{X}^{n_3}_{[t_i,1]}\\ &+\sum_{\substack{(n_1,n_2,n_3)\in\mathscr{S}\\ n_1+n_2<p}} \mathbf{X}^{n_1}_{[0,t_{i-1}]}\otimes\mathbf{X}^{n_2}_{[t_{i-1},t_i]}\otimes\mathscr{L}^{n_3-1}_{p-n_1-n_2,X;[t_i,1]}(y)\\ =&\sum_{k=p-1}^{n}\mathscr{L}^k_{p,X;[0,t_{i-1}]}(y)\otimes\mathbf{X}^{n-k}_{[t_{i-1},1]}+\sum_{\substack{(n_1,n_2,n_3)\in\mathscr{S}\\ n_1<p\leq n_1+n_2}}\mathbf{X}^{n_1}_{[0,t_{i-1}]}\otimes\mathscr{L}^{n_2-1}_{p-n_1,X;[t_{i-1},t_i]}(y)\otimes\mathbf{X}^{n_3}_{[t_i,1]}\\ &+\sum_{k=0}^{p-1}\mathbf{X}^k_{[0,t_i]}\otimes\mathscr{L}^{n-k}_{p-k,X;[t_i,1]}(y).\end{aligned}\tag{E.2}$$

Using the same decomposition on (E.1), we have

$$\begin{aligned}\mathbf{X}^{n+1} = &\sum_{k=p-1}^{n}\mathbf{X}^{k+1}_{[0,t_{i-1}]}\otimes\mathbf{X}^{n+1-k}_{[t_{i-1},1]}+\sum_{\substack{(n_1,n_2,n_3)\in\mathscr{S}\\ n_1<p\leq n_1+n_2}}\mathbf{X}^{n_1}_{[0,t_{i-1}]}\otimes\mathbf{X}^{n_2}_{[t_{i-1},t_i]}\otimes\mathbf{X}^{n_3}_{[t_i,1]}\\ &+\sum_{k=0}^{p-1}\mathbf{X}^k_{[0,t_i]}\otimes\mathbf{X}^{n+1-k}_{[t_i,1]}.\end{aligned}\tag{E.3}$$

It follows that

$$\begin{aligned}
&n!(\mathscr{L}^n_{p,X}(\beta_i) - (n+1)\mathbf{X}^{n+1}) \\
&\quad = n!\big(\mathscr{L}^n_{p,X}(\beta_i) - (n+1)\mathbf{X}^{n+1}\big) \\
&\quad = n!\sum_{k=p-1}^{n} (\mathscr{L}^k_{p,X;[0,t_{i-1}]}(\beta_i) - (n+1)\mathbf{X}^{k+1}_{[0,t_{i-1}]}) \otimes \mathbf{X}^{n-k}_{[t_{i-1},1]} \\
&\qquad + n! \sum_{\substack{(n_1,n_2,n_3)\in\mathscr{S} \\ n_1<p\le n_1+n_2}} \mathbf{X}^{n_1}_{[0,t_{i-1}]} \otimes (\mathscr{L}^{n_2-1}_{p-n_1,X;[t_{i-1},t_i]}(\beta_i) - (n+1)\mathbf{X}^{n_2}_{[t_{i-1},t_i]}) \otimes \mathbf{X}^{n_3}_{[t_i,1]} \\
&\qquad + n!\sum_{k=0}^{p-1} \mathbf{X}^{k}_{[0,t_i]} \otimes (\mathscr{L}^{n-k}_{p-k,X;[t_i,1]}(\beta_i) - (n+1)\mathbf{X}^{n+1-k}_{[t_i,1]}) \\
&\quad := A_1 + A_2 + A_3, \qquad \text{(E.4)}
\end{aligned}$$

where we denote by A_1 (respectively A_2 and A_3) the first (respectively second and third) sum. We now bound A_1, A_2, and A_3 separately. Note that, by Proposition 6.3,

$$\|\mathscr{L}^n_{p,X;[s,t]}(y)\| = \|\mathbf{X}^n\| \cdot \|y\| \le \frac{\|X\|^n_{TV;[s,t]}}{n!}\|y\|.$$

This yields, together with the triangle inequality,

$$\begin{aligned}
\|A_1\| &= \Big\| n! \sum_{k=p-1}^{n} (\mathscr{L}^k_{p,X;[0,t_{i-1}]}(\beta_i) - (n+1)\mathbf{X}^{k+1}_{[0,t_{i-1}]}) \otimes \mathbf{X}^{n-k}_{[t_{i-1},1]} \Big\| \\
&\le n! \sum_{k=p-1}^{n} \big\|\mathscr{L}^k_{p,X;[0,t_{i-1}]}(\beta_i)\big\| \big\|\mathbf{X}^{n-k}_{[t_{i-1},1]}\big\| + (n+1)! \sum_{k=p-1}^{n} \big\|\mathbf{X}^{k+1}_{[0,t_{i-1}]}\big\| \big\|\mathbf{X}^{n-k}_{[t_{i-1},1]}\big\| \\
&\le n! \sum_{k=p-1}^{n} \|\beta_i\| \frac{\|X\|^k_{TV;[0,t_{i-1}]}}{k!} \frac{\|X\|^{n-k}_{TV;[t_{i-1},1]}}{(n-k)!} + (n+1)! \sum_{k=p-1}^{n} \frac{\|X\|^{k+1}_{TV;[0,t_{i-1}]}}{(k+1)!} \frac{\|X\|^{n-k}_{TV;[t_{i-1},1]}}{(n-k)!}.
\end{aligned}$$

Recall that our choice of parametrization of X ensures that for any $j \in \{1,\dots,M\}$, $\|\beta_j\| = \ell$, and therefore

$$\|X\|_{TV;[0,t_{i-1}]} = \sum_{j=1}^{i-1} \|\beta_j\|(t_j - t_{j-1}) = \ell \sum_{j=1}^{i-1} (t_j - t_{j-1}) = \ell t_{i-1}.$$

Similarly, $\|X\|_{TV;[t_{i-1},1]} = \ell(1-t_{i-1})$. It follows that

$$\|A_1\| \le \ell^{n+1} \sum_{k=p-1}^{n} \binom{n}{k} t^k_{i-1}(1-t_{i-1})^{n-k} + \ell^{n+1} \sum_{k=p}^{n+1} \binom{n+1}{k} t^k_{i-1}(1-t_{i-1})^{n+1-k}. \qquad \text{(E.5)}$$

Bounding A_3 in a similar way, we obtain

$$\|A_3\| \le \ell^{n+1} \sum_{k=0}^{p-1} \binom{n}{k} t^k_{i}(1-t_{i})^{n-k} + \ell^{n+1} \sum_{k=0}^{p-1} \binom{n+1}{k} t^k_{i}(1-t_{i})^{n+1-k}. \qquad \text{(E.6)}$$

We now turn to the term A_2. We have

$$\begin{aligned}\|A_2\| &= \Big\| n! \sum_{\substack{(n_1,n_2,n_3)\in\mathscr{S}\\ n_1<p\le n_1+n_2}} \mathbf{X}^{n_1}_{[0,t_{i-1}]} \otimes (\mathscr{L}^{n_2-1}_{p-n_1,X;[t_{i-1},t_i]}(\beta_i) - (n+1)\mathbf{X}^{n_2}_{[t_{i-1},t_i]}) \otimes \mathbf{X}^{n_3}_{[t_i,1]} \Big\| \\ &\le n! \sum_{\substack{(n_1,n_2,n_3)\in\mathscr{S}\\ n_1<p\le n_1+n_2}} \big\|\mathbf{X}^{n_1}_{[0,t_{i-1}]}\big\| \big\|\mathscr{L}^{n_2-1}_{p-n_1,X;[t_{i-1},t_i]}(\beta_i) - (n+1)\mathbf{X}^{n_2}_{[t_{i-1},t_i]}\big\| \big\|\mathbf{X}^{n_3}_{[t_i,1]}\big\| \\ &\le n! \sum_{(n_1,n_2,n_3)\in\mathscr{S}} \frac{t_{i-1}^{n_1}\ell^{n_1}}{n_1!} \|\mathscr{L}^{n_2-1}_{p-n_1,X;[t_{i-1},t_i]}(\beta_i) - (n+1)\mathbf{X}^{n_2}_{[t_{i-1},t_i]}\| \frac{(1-t_i)^{n_3}\ell^{n_3}}{n_3!}.\end{aligned}$$

Since X is linear on $[t_{i-1},t_i]$,

$$\mathbf{X}^{n_2}_{[t_{i-1},t_i]} = \frac{(t_i-t_{i-1})^{n_2}}{n_2!}\beta_i^{\otimes n_2}, \quad \mathscr{L}^{n_2-1}_{p-n_1,X;[t_{i-1},t_i]}(\beta_i) = \frac{(t_i-t_{i-1})^{n_2-1}}{(n_2-1)!}\beta_i^{\otimes n_2}.$$

From this, we have

$$\begin{aligned}\|\mathscr{L}^{n_2-1}_{p-n_1,X;[t_{i-1},t_i]}(\beta_i) - (n+1)\mathbf{X}^{n_2}_{[t_{i-1},t_i]}\| &= \Big\| \frac{(t_i-t_{i-1})^{n_2-1}}{(n_2-1)!}\beta_i^{\otimes n_2} - \frac{(n+1)(t_i-t_{i-1})^{n_2}}{n_2!}\beta_i^{\otimes n_2}\Big\| \\ &= \frac{(t_i-t_{i-1})^{n_2}}{n_2!}\Big|\frac{n_2}{t_i-t_{i-1}} - (n+1)\Big| \|\beta_i^{\otimes n_2}\| \\ &= \frac{(t_i-t_{i-1})^{n_2}\ell^{n_2}}{n_2!}\Big|\frac{n_2}{t_i-t_{i-1}} - (n+1)\Big|,\end{aligned}$$

and

$$\begin{aligned}\|A_2\| &\le n! \sum_{(n_1,n_2,n_3)\in\mathscr{S}} \frac{t_{i-1}^{n_1}\ell^{n_1}}{n_1!}\frac{(t_i-t_{i-1})^{n_2}\ell^{n_2}}{n_2!}\Big|\frac{n_2}{t_i-t_{i-1}} - (n+1)\Big|\frac{(1-t_i)^{n_3}\ell^{n_3}}{n_3!} \\ &= \ell^{n+1} \sum_{(n_1,n_2,n_3)\in\mathscr{S}} \frac{(n+1)!}{n_1!n_2!n_3!} t_{i-1}^{n_1}(t_i-t_{i-1})^{n_2}(1-t_i)^{n_3}\Big|\frac{n_2}{(n+1)(t_i-t_{i-1})} - 1\Big| \\ &= \ell^{n+1} \sum_{k=0}^{n+1} \Bigg(\frac{1}{k!}\Big(\sum_{n_1=0}^{k} \frac{k!}{n_1!(k-n_1)!} t_{i-1}^{n_1}(1-t_i)^{k-n_1}\Big) \\ &\qquad \times \frac{(n+1)!}{(n+1-k)!}(t_i-t_{i-1})^{n+1-k}\Big|\frac{n+1-k}{(n+1)(t_i-t_{i-1})} - 1\Big|\Bigg) \\ &= \ell^{n+1} \sum_{k=0}^{n+1} \frac{(n+1)!}{k!(n+1-k)!}(1-(t_i-t_{i-1}))^k(t_i-t_{i-1})^{n+1-k}\Big|\frac{n+1-k}{(n+1)(t_i-t_{i-1})} - 1\Big| \\ &= \ell^{n+1} \sum_{k=0}^{n+1} \binom{n+1}{k}(1-(t_i-t_{i-1}))^{n+1-k}(t_i-t_{i-1})^k\Big|\frac{k}{(n+1)(t_i-t_{i-1})} - 1\Big|. \qquad \text{(E.7)}\end{aligned}$$

The right hand sides of (E.5), (E.6), and (E.7) correspond to probability mass functions of binomial random variables. Indeed, let

$$Y_{1,n} \sim \text{Binomial}(n, t_{i-1}), \quad Y_{2,n} \sim \text{Binomial}(n, t_i - t_{i-1}), \quad \text{and} \quad Z_{3,n} \sim \text{Binomial}(n, t_i).$$

Then, we have

$$\begin{aligned}
\|A_1\| &\le \ell^{n+1}\big(\mathbb{P}(Y_{1,n} \ge p-1) + \mathbb{P}(Y_{1,n+1} \ge p)\big),\\
\|A_3\| &\le \ell^{n+1}\big(\mathbb{P}(Y_{3,n} \le p-1) + \mathbb{P}(Y_{3,n+1} \le p-1)\big),\\
\|A_2\| &\le \ell^{n+1}\mathbb{E}\Big[\Big|\frac{Y_{2,n+1}}{(n+1)(t_i-t_{i-1})} - 1\Big|\Big].
\end{aligned}$$

First, since $\mathbb{E}[Y_{2,n+1}] = (n+1)(t_i - t_{i-1})$, and $\mathrm{Var}(Y_{2,n+1}) = (n+1)(t_i-t_{i-1})(1-(t_i-t_{i-1}))$, by Hölder's inequality,

$$\begin{aligned}
\mathbb{E}\Big[\Big|\frac{Y_{2,n+1}}{(n+1)(t_i-t_{i-1})} - 1\Big|\Big] &= \frac{1}{(n+1)(t_i-t_{i-1})}\mathbb{E}\big[|Z_{1,n+1} - \mathbb{E}Z_{1,n+1}|\big]\\
&\le \frac{1}{(n+1)(t_i-t_{i-1})}\mathbb{E}\big[|Z_{1,n+1} - \mathbb{E}Z_{1,n+1}|^2\big]^{1/2}\\
&\le \frac{1}{(n+1)(t_i-t_{i-1})}\sqrt{(n+1)(t_i-t_{i-1})(1-(t_i-t_{i-1}))}\\
&\le \frac{1}{\sqrt{n+1}}\sqrt{\frac{1-(t_i-t_{i-1})}{t_i-t_{i-1}}}.
\end{aligned}$$

The other terms decay exponentially fast if p is well chosen. We give the details for the bound on $\|A_1\|$ below but the one on $\|A_3\|$ is treated in the same way. First, we have

$$\begin{aligned}
\|A_1\| &\le \ell^{n+1}\big(\mathbb{P}(Y_{1,n} \ge p-1) + \mathbb{P}(Y_{1,n+1} \ge p)\big),\\
&\le \ell^{n+1}\big(\mathbb{P}(Y_{1,n} - \mathbb{E}Y_{1,n} \ge p-1-nt_{i-1}) + \mathbb{P}(Y_{1,n+1} - \mathbb{E}Y_{1,n+1} \ge p-(n+1)t_{i-1})\big).
\end{aligned}$$

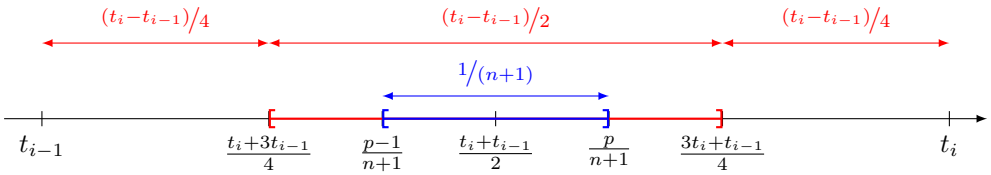


Figure E.2 – Illustration on the choice of p.

Recall that p is chosen as $p = \lfloor (n+1)(3t_i+t_{i-1})/4 \rfloor$. Then, if $n \ge 2/(t_i-t_{i-1})$, we have

$$\frac{t_i+3t_{i-1}}{4} \le \frac{p}{n+1} < \frac{3t_i+t_{i-1}}{4}$$

This is summarized in Figure E.2. In particular, we have

$$p-(n+1)t_{i-1} = (n+1)\Big(\frac{p}{n+1} - t_{i-1}\Big) \ge (n+1)\Big(\frac{t_i+3t_{i-1}}{4} - t_{i-1}\Big) = (n+1)\frac{t_i-t_{i-1}}{4} > 0,$$

and

$$p-1-nt_{i-1} = n\Big(\frac{p}{n+1} - t_{i-1} + \frac{p}{n+1} - \frac{p-1}{n}\Big) \geq n\Big(\frac{t_i - t_{i-1}}{4} + \frac{n+1-p}{n(n+1)}\Big) > n\frac{t_i - t_{i-1}}{4} > 0.$$

By Hoeffding's inequality, we obtain

$$\begin{aligned}\|A_1\| &\leq \ell^{n+1}\big(\mathbb{P}(Y_{1,n} - \mathbb{E}Y_{1,n} \geq p-1-nt_{i-1}) + \mathbb{P}(Y_{1,n+1} - \mathbb{E}Y_{1,n+1} \geq p-(n+1)t_{i-1})\big)\\ &\leq \ell^{n+1}\Big(\exp\Big(-n\Big(\frac{p-1}{n} - t_{i-1}\Big)^2\Big) + \exp\Big(-(n+1)\Big(\frac{p}{n+1} - t_{i-1}\Big)^2\Big)\Big)\\ &\leq \ell^{n+1}\Big(\exp\Big(-\frac{n(t_i - t_{i-1})^2}{16}\Big) + \exp\Big(-\frac{(n+1)(t_i - t_{i-1})^2}{16}\Big)\Big)\\ &\leq 2\ell^{n+1}\exp\Big(-\frac{n(t_i - t_{i-1})^2}{16}\Big).\end{aligned}$$

With similar arguments, we obtain

$$\|A_3\| \leq 2\ell^{n+1}\exp\Big(-\frac{n(t_i - t_{i-1})^2}{16}\Big).$$

Combining these inequalities, we obtain finally

$$\begin{aligned}\|\mathscr{L}^n_{p,X}(\beta_i) - (n+1)\mathbf{X}^{n+1}\| &\leq \frac{\ell^{n+1}}{n!}\Big(\frac{1}{\sqrt{n+1}}\sqrt{\frac{1-(t_i - t_{i-1})}{t_i - t_{i-1}}} + 4\exp\Big(-\frac{n(t_i - t_{i-1})^2}{16}\Big)\Big)\\ &= \mathscr{O}\Big(\frac{\ell^{n+1}}{n!\sqrt{n+1}}\Big).\end{aligned}$$

E.2 Proof of Theorem 6.8

The proof is based on several ingredients of Hambly and Lyons (2010). Their core idea is to move the path to a hyperbolic space, which is called the "development" of the path. First, note that if $Y \in BV(\mathbb{R}^d)$ is defined by $Y_t = {}^{X_t}/_{\ell}$, then $\|Y\|_{TV;[0,1]} = 1$, and

$$\mathbf{Y}^n = \int_{(u_1,\dots,u_n)\in\Delta_n} dY_{u_1}\otimes\cdots\otimes dY_{u_n} = \int_{(u_1,\dots,u_n)\in\Delta_n} \frac{1}{\ell}dX_{u_1}\otimes\cdots\otimes\frac{1}{\ell}dX_{u_n} = \frac{1}{\ell^n}\mathbf{X}^n.$$

Without loss of generality, we therefore restrict ourselves to the case $\ell = 1$, and the lower bound obtained will have to be multiplied by ℓ^n to go back to the general case.

We begin the proof with a series of definitions.

Definition E.1. *The hyperboloid model is the subspace of $\mathbb{R}^{d+1}$ defined by*

$$\mathbb{H} = \{y \in \mathbb{R}^{d+1} \,|\, B(y,y) = -1\},$$

where, for any $x, y \in \mathbb{R}^{d+1}$,

$$B(x,y) = \sum_{i=1}^{d} x_i y_i - x_{d+1}y_{d+1}. \tag{E.8}$$

This hyperbolic space has several well-known good properties (Cannon et al., 1997; Paupert, 2016; Loustau, 2020). The main one is that the hyperbolic distance between two points $y, z \in \mathbb{H}$, denoted by d, can be easily computed as

$$d(y, z) = \operatorname{arcosh}(-B(y, z)).$$

Then, we define the space of bounded linear operators between vector spaces and their associated operator norm.

Definition E.2. *The set of bounded linear operators between two normed vector spaces $(E, \|\cdot\|_E)$ and $(F, \|\cdot\|_F)$ is denoted by $\mathscr{L}(E, F)$. Equipped with the operator norm, defined for any $f \in \mathscr{L}(E, F)$ by*

$$\|f\|_{\mathscr{L}(E,F)} = \sup_{x \in E, \|x\|_E = 1} \|f(x)\|_F,$$

it is itself a normed vector space.

We will use bounded linear maps on $\mathbb{R}^d$ equipped with the Euclidean norm, and regard linear maps as matrix multiplication. Therefore, we will denote fx instead of $f(x)$ for function evaluation. In particular, for any $y \in \mathbb{R}^d$, let $F : \mathbb{R}^d \to \mathscr{L}(\mathbb{R}^{d+1}, \mathbb{R}^{d+1})$ be defined by

$$Fy = \begin{pmatrix} 0 & \cdots & 0 & y_1 \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & y_d \\ y_1 & \cdots & y_d & 0 \end{pmatrix}.$$

Lemma E.2. *F is a bounded linear map and its operator norm is $\|F\|_{\mathscr{L}(\mathbb{R}^d, \mathscr{L}(\mathbb{R}^{d+1}, \mathbb{R}^{d+1}))} = 1$.*

Proof. For any $y \in \mathbb{R}^d$, $z \in \mathbb{R}^{d+1}$, we first have

$$Fyz = \begin{pmatrix} 0 & \cdots & 0 & y_1 \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & y_d \\ y_1 & \cdots & y_d & 0 \end{pmatrix} \begin{pmatrix} z_1 \\ \vdots \\ z_{d+1} \end{pmatrix} = \begin{pmatrix} y_1\, z_{d+1} \\ \vdots \\ y_d\, z_{d+1} \\ \sum_{i=1}^d y_i\, z_i \end{pmatrix}.$$

Then,

$$\begin{aligned} \|Fyz\|^2 &= z_{d+1}^2 \sum_{i=1}^d y_i^2 + \Big(\sum_{i=1}^d y_i z_i\Big)^2 \\ &\leq z_{d+1}^2 \|y\|^2 + \|y\|^2 \sum_{i=1}^d z_i^2 \qquad \text{(by the Cauchy-Schwartz inequality)} \\ &\leq \|y\|^2 \|z\|^2. \end{aligned}$$

This yields

$$\|Fy\|_{\mathscr{L}(\mathbb{R}^{d+1}, \mathbb{R}^{d+1})} = \sup_{z \in \mathbb{R}^{d+1}, \|z\|=1} \|Fyz\| \leq \sup_{z \in \mathbb{R}^{d+1}, \|z\|=1} \|y\| \|z\| = \|y\|.$$

Moreover the inequality becomes an equality when $z = e_{d+1}$ (where $(e_1, \ldots, e_{d+1})$ is the canonical

basis of $\mathbb{R}^{d+1}$), so

$$\|Fy\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} = \|y\|, \tag{E.9}$$

and

$$\|F\|_{\mathscr{L}(\mathbb{R}^d,\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1}))} = \sup_{y\in\mathbb{R}^d,\|y\|=1} \|Fy\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} = \sup_{y\in\mathbb{R}^d,\|y\|=1} \|y\| = 1.$$

□

We are now in a position to define the development of a path $X \in BV(\mathbb{R}^d)$ to the hyperbolic space. Let $t \in [0,1]$ and $\Gamma_t : \mathbb{R}^{d+1} \to \mathbb{R}^{d+1}$ be defined as follows. For any $y \in \mathbb{R}^{d+1}$, $\Gamma_t y$ is the solution at time t of the controlled differential equation

$$dY_t = F(dX_t)Y_t, \quad Y_0 = y. \tag{E.10}$$

Lemma E.3. *For any $t \in [0,1]$, Γ_t is a (well-defined) linear map preserving the function B: for any $y, \widetilde{y} \in \mathbb{R}^{d+1}$,*

$$B(\Gamma_t y, \Gamma_t \widetilde{y}) = B(y, \widetilde{y}).$$

Proof. Equation (E.10) is a linear controlled differential equation and by Picard's theorem, for any $y \in \mathbb{R}^d$, it has a unique solution. Therefore Γ_t is well-defined. Moreover, it is clearly a linear operator: if $y, \widetilde{y} \in \mathbb{R}^d$, and $Y_t = \Gamma_t y$, $\widetilde{Y}_t = \Gamma_t \widetilde{y}$, then $Y_t + \widetilde{Y}_t$ follows equation (E.10) with initial point $y + \widetilde{y}$, so by uniqueness of the solution $\Gamma_t(y + \widetilde{y}) = \Gamma_t y + \Gamma_t \widetilde{y}$.

□

The differential equation (E.10) may also be rewritten as a differential equation on Γ, which can be thought of as a $(d+1) \times (d+1)$ matrix:

$$d\Gamma_t = F(dX_t)\Gamma_t, \quad \Gamma_0 = I_{d+1}, \tag{E.11}$$

where I_{d+1} is the identity matrix. Now let $y_0 = (0, \dots, 0, 1)^\top \in \mathbb{R}^{d+1}$. The development of X in $\mathbb{H}$ is defined as the path $Y : [0,1] \to \mathbb{H}$, $Y_t = \Gamma_t y_0$. Since $B(y_0, y_0) = -1$, then $y_0 \in \mathbb{H}$ and by Lemma E.3 $Y_t \in \mathbb{H}$ for any $t \in [0,1]$. So equation (E.11) maps a path X in $\mathbb{R}^d$ into a new path Y in $\mathbb{H}$, in a way that preserves a number of properties of the path X:

- Any linear piece of X is mapped to a geodesic in $\mathbb{H}$. So if X satisfies (6.5), then Y is geodesic on each $[t_{i-1}, t_i]$, and $d(Y_{t_i}, Y_{t_{i-1}}) = \|\beta_i\|(t_i - t_{i-1}) = \|X_{t_i} - X_{t_{i-1}}\|$.
- Γ preserves the angles between linear segments. If 2ω is the angle between the linear pieces $[X_{t_{i-1}}, X_{t_i}]$ and $[X_{t_i}, X_{t_{i+1}}]$, then the angle between the geodesics $[Y_{t_{i-1}}, Y_{t_i}]$ and $[Y_{t_i}, Y_{t_{i+1}}]$ is also equal to 2ω.

Now we state two results from Hambly and Lyons (2010) that are the key ingredients of our proof.

Lemma E.4.

(*i*) *Let $Y : [0,1] \to \mathbb{H}$ be a continuous path, geodesic on the intervals $[t_{i-1}, t_i]$, where $0 = t_0 < t_1 < \cdots < t_M = 1$ is a partition of $[0,1]$. If 2Ω is the smallest angle between two geodesic segments and each geodesic segment has length at least $K(\Omega)$ (defined by (6.9)), then*

$$0 \le \sum_{i=1}^{M} d(Y_{t_{i-1}}, Y_{t_i}) - d(y_0, Y_{t_M}) \le (M-1)K(\Omega).$$

(*ii*) *Let $SO(B)$ denote the group of $(d+1)\times(d+1)$ matrices with positive determinant preserving the isometry B: if $G \in SO(B)$, then for any $x, y \in \mathbb{R}^{d+1}$, $B(Gx, Gy) = B(x, y)$. Then, for any $G \in SO(B)$,*

$$\|G\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} \geq e^{d(y_0, Gy_0)},$$

where $y_0 = (0, \dots, 0, 1)^\top \in \mathbb{R}^{d+1}$.

Proof. We refer the reader to Lemma 3.7 and Proposition 3.13 of Hambly and Lyons (2010) for proofs. □

We now have all the ingredients necessary to prove Theorem 6.8. Without loss of generality, we can assume that X has a total variation of 1. Let α be any strictly positive real number, we define the renormalized path $X^\alpha = \alpha X$ and its corresponding hyperbolic developments Γ^α and Y^α. Let D be the length of the shortest linear segment of X, and let assume that $\alpha > K(\Omega)/D$. Then, Y^α satisfies the hypothesis of (i) in Lemma E.4, and we have

$$0 \leq \sum_{i=1}^{M} d(Y^\alpha_{t_{i-1}}, Y^\alpha_{t_i}) - d(y_0, Y^\alpha_1) \leq (M-1)K(\Omega).$$

Since X is linear on each segment $[t_{i-1}, t_i]$, $d(Y^\alpha_{t_{i-1}}, Y^\alpha_{t_i}) = \|X^\alpha_{t_i} - X^\alpha_{t_{i-1}}\|$, and

$$\sum_{i=1}^{M} d(Y^\alpha_{t_{i-1}}, Y^\alpha_{t_i}) = \sum_{i=1}^{M} \|X_{t_i} - X_{t_{i-1}}\| = \alpha,$$

therefore

$$d(y_0, Y^\alpha_1) = d(y_0, \Gamma^\alpha_1 y_0) \geq \alpha - (M-1)K(\Omega).$$

Now lemma E.4 yields

$$\|\Gamma^\alpha_1\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} \geq e^{d(y_0, \Gamma^\alpha_1 y_0)} \geq e^{\alpha-(M-1)K(\Omega)}. \tag{E.12}$$

By definition of Γ as the solution to the linear controlled differential equation (E.11), we can actually write Γ as a linear function of the signature of X:

$$\begin{aligned}
\Gamma^\alpha_1 &= I_{d+1} + \int_0^1 F(dX^\alpha_t)\Gamma_t \\
&= I_{d+1} + \int_0^1 F(dX^\alpha_t)\big(I_{d+1} + \int_0^t F(dX^\alpha_s)\Gamma_s\big) \\
&= I_{d+1} + \int_0^1 F(dX^\alpha_t) + \int_0^1 \int_0^t F(dX^\alpha_t)F(dX^\alpha_s)\Gamma_s \\
&= \dots.
\end{aligned}$$

Iterating this procedure gives

$$\Gamma^\alpha_1 = I_{d+1} + \sum_{k=1}^{\infty} \int_{\Delta_k} F(dX^\alpha_{t_1}) \dots F(dX^\alpha_{t_k}). \tag{E.13}$$

Using tensor notations, we denote by $F^{\otimes k}$ the linear map $F^{\otimes k} : (\mathbb{R}^d)^{\otimes k} \to \mathscr{L}(\mathbb{R}^{d+1}, \mathbb{R}^{d+1})$ which,

to any tensor $u = \sum_{j=1}^{\ell} u_{1,j} \otimes \cdots \otimes u_{k,j}$ associates

$$F^{\otimes k}(u) = \sum_{j=1}^{\ell} F(u_{1,j}) \cdots F(u_{k,j}).$$

Note that we then have

$$\begin{aligned}
\|F^{\otimes k}(u)\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} &\leq \sum_{j=1}^{p} \|F(u_{1,j}) \cdots F(u_{k,j})\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} \\
&\leq \sum_{j=1}^{p} \|F(u_{1,j})\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} \cdots \|F(u_{k,j})\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} \\
&\qquad \text{(because the operator norm is sub-multiplicative)} \\
&\leq \sum_{j=1}^{p} \|u_{1,j}\| \ldots \|u_{k,j}\|.
\end{aligned}$$

Taking the infimum over any representation of u yields $\|F^{\otimes k}(u)\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} \leq \|u\|_{\pi}$. Note that this is not true for any tensor norm. Then, (E.13) becomes

$$\begin{aligned}
\Gamma_1^{\alpha} &= I_{d+1} + \sum_{k=1}^{\infty} F^{\otimes k}\Big(\int_{\Delta_k} dX_{t_1}^{\alpha} \otimes \cdots \otimes dX_{t_k}^{\alpha} \Big) \\
&= I_{d+1} + \sum_{k=1}^{\infty} \alpha^k F^{\otimes k}\Big(\int_{\Delta_k} dX_{t_1} \otimes \cdots \otimes dX_{t_k} \Big) \\
&= I_{d+1} + \sum_{k=1}^{\infty} \alpha^k F^{\otimes k}(\mathbf{X}^k).
\end{aligned}$$

Endowing $(\mathbb{R}^d)^{\otimes k}$ with the projective norm yields

$$\begin{aligned}
\|\Gamma_1^{\alpha}\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} &\leq \|I_{d+1}\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} + \Big\| \sum_{k=1}^{\infty} \alpha^k F^{\otimes k}(\mathbf{X}^k) \Big\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} \\
&\leq 1 + \sum_{k=1}^{\infty} \alpha^k \big\| F^{\otimes k}(\mathbf{X}^k) \big\|_{\mathscr{L}(\mathbb{R}^{d+1},\mathbb{R}^{d+1})} \\
&\leq 1 + \sum_{k=1}^{\infty} \alpha^k \|\mathbf{X}^k\|_{\pi}
\end{aligned}$$

Note that, by Proposition 6.3, the right hand side is a converging series so (E.13) is well-defined. Let, for any $k \geq 1$, $b_k = k! \|\mathbf{X}^k\|_{\pi}$, and $b_0 = 1$. Note that since we have taken X of norm 1, $b_k \leq 1$. Combining the last inequality with (E.12) gives

$$e^{-(M-1)K(\Omega)} \leq e^{-\alpha} \sum_{k=0}^{\infty} \frac{\alpha^k}{k!} b_k. \tag{E.14}$$

The last step in our proof is to show that there exists infinitely many k such that $b_k \geq e^{-(M-1)K(\Omega)}/2$. To this end, let us take α a non-zero integer, denoted by n, and $Z \sim \text{Poisson}(n)$ a random variable following a Poisson distribution with parameter α. The right-hand-side of (E.14) is then exactly $\mathbb{E}[b_Z]$. Let us show that

$$\mathbb{P}\big(b_Z \geq \frac{1}{2}e^{-(M-1)K(\Omega)} \text{ and } Z \in J_n\big) > 0.$$

where J_n denotes the interval $[n - n^{3/4}, n + n^{3/4}]$. It follows that there exists $k \in J_n$ such that $b_k \geq \frac{1}{2}e^{-(M-1)K(\Omega)}$ (otherwise the probability above would be zero). Since for any $k \geq 0$, $b_k \leq 1$, we have

$$\begin{aligned} e^{-(M-1)K(\Omega)} \leq \mathbb{E}[b_Z] &= \mathbb{E}[b_Z \mathbf{1}_{b_Z \geq \frac{1}{2}e^{-(M-1)K(\Omega)}}] + \mathbb{E}[b_Z \mathbf{1}_{b_Z < \frac{1}{2}e^{-(M-1)K(\Omega)}}] \\ &\leq \mathbb{P}(b_Z \geq \frac{1}{2}e^{-(M-1)K(\Omega)}) + \frac{1}{2}e^{-(M-1)K(\Omega)}, \end{aligned}$$

which yields

$$\mathbb{P}(b_Z \geq \frac{1}{2}e^{-(M-1)K(\Omega)}) \geq \frac{1}{2}e^{-(M-1)K(\Omega)} > 0.$$

Moreover, by Chebyshev's inequality,

$$\mathbb{P}(b_Z \notin J_n) = \mathbb{P}(|Z - n| > n^{3/4}) \leq \frac{1}{\sqrt{n}}.$$

Let n_0 be the smallest integer such that $n_0 > 4e^{2(M-1)K(\Omega)}$. Then, for any $n \geq n_0$,

$$\begin{aligned} \mathbb{P}\big(b_Z \geq \frac{1}{2}e^{-(M-1)K(\Omega)} \text{ and } Z \in J_n\big) &\geq \mathbb{P}(b_Z \geq \frac{1}{2}e^{-(M-1)K(\Omega)}) - \mathbb{P}(b_Z \notin J_n) \\ &\geq \frac{1}{2}e^{-(M-1)K(\Omega)} - \frac{1}{\sqrt{n}} \\ &\geq \frac{1}{2}e^{-(M-1)K(\Omega)} - \frac{1}{\sqrt{n_0}} > 0. \end{aligned}$$

We conclude that for any $n > n_0$, there exists $k_n \in J_n$ such that $b_{k_n} \geq 2^{-1}e^{-(M-1)K(\Omega)}$, that is,

$$\|\mathbf{X}^{k_n}\|_\pi \geq \frac{e^{-(M-1)K(\Omega)}}{2k_n!}.$$

Bibliography

Cannon, J. W., Floyd, W. J., Kenyon, R., Parry, W. R., et al. (1997). Hyperbolic geometry. *Flavors of geometry*, *31*, 59–115.

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

Loustau, B. (2020). Hyperbolic geometry. *arXiv:2003.11180*.

Paupert, J. (2016). Introduction to hyperbolic geometry.

Learning time-dependent data with the signature transform

Résumé

Les applications modernes de l'intelligence artificielle amènent à travailler avec des données temporelles multivariées de grande dimension qui posent de nombreux défis. Par une approche géométrique des flux de données, la notion de signature, représentation d'un processus en un vecteur infini de ses intégrales itérées, est un outil prometteur. Ses propriétés développées dans le cadre de la théorie des chemins rugueux en font en effet un bon candidat pour jouer le rôle de features, ensuite injectées dans des algorithmes d'apprentissage. Si la définition de la signature remonte aux travaux de Chen (1960), son utilisation en apprentissage est récente et de nombreuses questions théoriques et méthodologiques restent à explorer. Nous nous intéressons donc à l'utilisation de la signature pour développer des algorithmes génériques et performants pour les données temporelles de grande dimension, ainsi que de leur fournir des garanties théoriques. Ce but se déploie principalement dans deux directions : d'une part, développer de nouveaux algorithmes prenant en entrée la signature des données, d'autre part utiliser la signature comme un outil théorique pour étudier les algorithmes existants d'apprentissage profond, via la notion récente de *neural ordinary differential equation* qui fait le lien entre apprentissage profond et équations différentielles.

Mots clés : signatures, données temporelles, apprentissage séquentiel, réseaux de neurones récurrents

---

Abstract

Modern applications of artificial intelligence lead to high-dimensional multivariate temporal data that pose many challenges. Through a geometric approach to data flows, the notion of signature, a representation of a process as an infinite vector of its iterated integrals, is a promising tool. Its properties, developed in the context of rough path theory, make it a good candidate to play the role of features, then injected in learning algorithms. If the definition of the signature goes back to the work of Chen (1960), its use in machine learning is recent. Many theoretical and methodological questions remain to be explored. We are therefore interested in using the signature to develop generic and efficient algorithms for high-dimensional temporal data, with theoretical guarantees. This goal is mainly deployed in two directions: on the one hand, to develop new algorithms taking the signature of the data as input, and, on the other hand, to use the signature as a theoretical tool to study existing deep learning algorithms, via the recent notion of neural ordinary differential equation which makes the link between deep learning and differential equations.

Keywords: signatures, temporal data, sequential learning, recurrent neural networks

---

Laboratoire de Probabilités, Statistique et Modélisation
Sorbonne Université – Campus Pierre et Marie Curie – 4 place Jussieu – 75005 Paris – France