Path Imputation Strategies for Signature Models of Irregular Time Series

Michael Moor[1,2], Max Horn[1,2], Christian Bock[1,2], Karsten Borgwardt[1,2], Bastian Rieck[1,2]

[1]Department of Biosystems Science and Engineering, ETH Zurich, Switzerland
[2]SIB Swiss Institute of Bioinformatics, Switzerland
{firstname.lastname@bsse.ethz.ch}

Abstract

The signature transform is a 'universal nonlinearity' on the space of continuous vector-valued paths, and has received attention for use in machine learning on time series. However, real-world temporal data is typically observed at discrete points in time, and must first be transformed into a continuous path before signature techniques can be applied. We make this step explicit by characterising it as an imputation problem, and empirically assess the impact of various imputation strategies when applying signature-based neural nets to irregular time series data. For one of these strategies, Gaussian process (GP) adapters, we propose an extension (GP-PoM) that makes uncertainty information directly available to the subsequent classifier while at the same time preventing costly Monte-Carlo (MC) sampling. In our experiments, we find that the choice of imputation drastically affects shallow signature models, whereas deeper architectures are more robust. Next, we observe that uncertainty-aware *predictions* (based on GP-PoM or indicator imputations) are beneficial for predictive performance, even compared to the uncertainty-aware *training* of conventional GP adapters. In conclusion, we have demonstrated that the path construction is indeed crucial for signature models and that our proposed strategy leads to competitive performance in general, while improving robustness of signature models in particular.

1. Introduction

Originally described by Chen [5, 6, 7] and popularised in the theory of rough paths and controlled differential equations [14, 31, 32], the *signature transform*, also known as the *path signature* or simply *signature*, acts on a continuous vector-valued path of bounded variation, and returns a graded sequence of statistics, which determine a path up to a negligible equivalence class. Moreover, *every* continuous function of a path can be recovered by applying a linear transform to this collection of statistics [3, Proposition A.6]. This 'universal nonlinearity' property makes the signature a promising nonparametric feature extractor in both generative and discriminative learning scenarios. Further properties include the signature's uniqueness [20], as well as factorial decay of its higher order terms [32]. These theoretical foundations have been accompanied by outstanding empirical results when applying signatures to clinical time series classification tasks [34, 40]. Due to their similarities, we may hope that tools that apply to continuous paths can *also* be applied to multivariate time series. But since multivariate time series are not continuous paths, one first needs to construct a continuous path before signature techniques are applicable. Previous work [3, 12, 27] characterised this construction as an embedding problem, and typically considered it a minor technical detail. This is exacerbated by the—perfectly sensible—behaviour of software for computing the signature [22, 39], which commonly considers a continuous piecewise linear path as an input, described by its sequence of knots, i.e. values. Since such sequences resemble a sequence of data, the signature is sometimes interpreted as operating on sequences of data rather than on paths [3, 27]. By contrast, here we show that considering the path construction process is crucial for achieving competitive predictive performance: we reinterpret the task of constructing a continuous path, turning it from an embedding problem to an imputation problem, which we call *path imputation*.

While previous research concerning the signature transform focused on its excellent theoretical properties, such as sampling independence [3, Proposition A.7], our findings show that this does not necessarily correspond to empirical performance. We perform a thorough investigation of multiple imputation schemes in combination with various models that can potentially employ signatures. Furthermore, motivated by the fact that missingness itself can be informative for time series classification [42], we propose a novel imputation strategy: an extension of Gaussian process adapters [16, 29], which exploits uncertainty information during each prediction step and which is beneficial for signature models, but also of independent interest. We make our code anonymously available under `https://osf.io/bg9cw/?view_only=5193e93118d84a5f9be4f261df4c0a06`.

2. Related work

A key motivation for this work is the use of the signature transform in machine learning: recent work [8, 26, 28, 35, 37, 45, 46] typically employed the signature transform as a nonparametric feature extractor, on top of which a model is learnt. A growing body of work has also investigated how to integrate the signature transform more tightly with neural networks; Reizenstein [38], Liao et al. [30], and Bonnier et al. [3] all study how to use the signature transform (or variants thereof) within typical neural network models. Chevyrev and Oberhauser [9], Király and Oberhauser [25] study how the signature transform may be used to define a *kernel*—i.e. a symmetric, positive definite function that is typically used as a similarity measure—on path space, while Toth and Oberhauser [44] show how this kernel may be used to define a Gaussian process. In much of this work, data has been converted into a continuous path via linear interpolation. Some authors [8, 12] have additionally considered 'rectilinear' interpolation, which is similar. Levin et al. [27] present the 'time-joined transformation', which is a hybrid of the two, such that the resulting path exhibits a causal dependence on the data. However, to our knowledge, no prior work has regarded (and empirically investigated) this as an imputation problem.

Imputation schemes The general problem of imputing data is well-known and well-studied, and we will not attempt to describe it here; see for example Gelman and Hill [18, Chapter 25]. Imputation methods typically only fill in missing discrete data points, and do not attempt to impute the underlying continuous path. Gaussian process adapters [29], by contrast, are capable of imputing a *full* continuous path, from which we may sample arbitrarily. Hence, this framework will be considered more closely in this paper. We note that there are also other approaches that perform imputation end-to-end with a downstream classifier [43] and methods that skip the imputation step altogether based on recently-proposed Neural-ODE like architectures [23, 41], variants of recurrent neural networks [4] or set functions [21]. However, the scope of this work is to specifically assess the impact of path imputations for the signature, hence we deem the larger comparison including imputation-free scenarios interesting for future work, while it bypasses the central point of this paper.

3. Background: Signature transform and Gaussian process adapters

Path signatures Let $f = (f_1, \ldots, f_d) \colon [a, b] \to \mathbb{R}^d$ be a continuous, piecewise differentiable path. Then the *signature transform up to depth N* is

$$\mathrm{Sig}^N(f) = \left(\left(\int \cdots \int_{a<t_1<\cdots<t_k<b} \prod_{j=1}^{k} \frac{\mathrm{d}f_{i_j}}{\mathrm{d}t}(t_j)\, \mathrm{d}t_1 \cdots \mathrm{d}t_k \right)_{1 \le i_1, \ldots, i_k \le d} \right)_{1 \le k \le N}. \tag{1}$$

This definition can be extended to paths of bounded variation by replacing these integrals with Stieltjes integrals with respect to each f_{i_j}. In brief, the signature transform may be interpreted as extracting information about *order* and *area* of a path. One may interpret its terms as 'the area/order of one channel with respect to some collection of other channels'. To give an explicit example: first level terms simply describe the increment of the path with respect to one channel, whereas second-order terms are related to the *Levy area* of the path, as shown for a one-dimensional example in Figure 1.

For an exposition on the properties of the signature transform and its use in machine learning, please refer to Chevyrev and Kormilitzin [8] or Bonnier et al. [3, Appendix A]. For building more intuition, in Section A.6 of the appendix, we compare the signature to more well-known transforms.

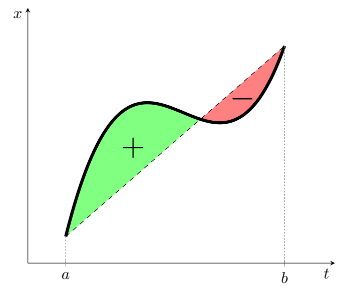


Figure 1: Given a path (bold), its Levy area is its signed area with respect to the chord joining its endpoints.

Computing the signature transform Continuous piecewise linear paths are the paths of choice, computationally speaking, due to the fact that this is the only case for which efficient algorithms for computing the signature transform are known [22].

This is not a serious hurdle when one wishes to compute the signature of a path f that is not piecewise linear—as the signature of piecewise linear approximations to f will tend towards the signature of f as the quality of the approximation increases—but it does enforce this requirement on our imputation schemes. Thus, all of the imputation schemes we examine will first seek to select a collection of points in data space (not necessarily only where we had data before), and for computing the signature we join them up into a piecewise linear path.

Notation We define the space of time series over a set A by

$$\mathcal{S}(A) = \{((t_1, x_1), \ldots, (t_n, x_n)) \,|\, t_i \in \mathbb{R}, x_i \in A, n \in \mathbb{N}, \text{ such that } t_1 \le \cdots \le t_n\}. \tag{2}$$

Furthermore, let $\mathcal{Y}$ be a set and let $\mathcal{X}_j = \mathbb{R}$ for $j \in \{1, \ldots, d\}$ and $d \in \mathbb{N}$. Then we assume that we observe a dataset of labelled time series $(\mathbf{x}_k, y_k)$ for $k \in \{1, \ldots, N\}$, where $\mathbf{x}_k \in \mathcal{S}(\mathcal{X}^*)$ and $y_k \in \mathcal{Y}$, with $\mathcal{X}^* = \prod_{j=1}^{d}(\mathcal{X}_j \cup \{*\})$ and $*$ representing no observation. We similarly define $\mathcal{X} = \prod_{j=1}^{d} \mathcal{X}_j$. Thus, $\mathcal{X}$ is the data space, while $\mathcal{X}^*$ is the data space allowing missing data, and $\mathcal{Y}$ is the set of labels.

Gaussian process adapter Some of the imputation schemes we consider are based on the uncertainty aware-framework of multi-task Gaussian process adapters [16, 29]. Let $\mathcal{W}, \mathcal{H}$ be some sets. Let $\ell \colon \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ be a loss function. Let $F \colon \mathcal{X}^{[a,b]} \times \mathcal{W} \to \mathcal{Y}$, be some (typically neural network) model, with $\mathcal{W}$ interpreted as a space of parameters. Let

$$\mu \colon [a, b] \times \mathcal{S}(\mathcal{X}^*) \times \mathcal{H} \to \mathcal{X}$$
$$\Sigma \colon [a, b] \times [a, b] \times \mathcal{S}(\mathcal{X}^*) \times \mathcal{H} \to \mathcal{X}$$

be mean and covariance functions, with $\mathcal{H}$ interpreted as a space of hyperparameters. The dependence on $\mathcal{S}(\mathcal{X}^*)$ is used to represent conditioning on observed values.

Then the goal is to solve

$$\underset{\mathbf{w}\in\mathcal{W},\eta\in\mathcal{H}}{\arg\min} \sum_{k=1}^{N} \overbrace{\mathbb{E}_{\mathbf{z}_k\sim\mathcal{N}(\mu(\cdot,\mathbf{x}_k,\eta),\Sigma(\cdot,\cdot,\mathbf{x}_k,\eta))}\left[\ell(F(\mathbf{z}_k,\mathbf{w}),y_k)\right]}^{E_k}. \tag{3}$$

As this expectation is typically not tractable, it is estimated by MC sampling with S samples, i.e.

$$E_k \approx \frac{1}{S}\sum_{s=1}^{S} \ell(F(\mathbf{z}_{s,k},\mathbf{w}),y_k), \tag{4}$$

where

$$\mathbf{z}_{s,k} \sim \mathcal{N}\left(\mu(\,\cdot\,,\mathbf{x}_k,\eta),\Sigma(\,\cdot\,,\,\cdot\,,\mathbf{x}_k,\eta)\right). \tag{5}$$

Alternatively, one may forgo allowing the uncertainty to propagate through F by instead passing the posterior mean directly to F; this corresponds to solving

$$\underset{\mathbf{w}\in\mathcal{W},\eta\in\mathcal{H}}{\arg\min} \sum_{k=1}^{N} \ell(F(\mu(\,\cdot\,,\mathbf{x}_k,\eta),\mathbf{w}),y_k). \tag{6}$$

4. Path imputations for signature models

Signatures act on continuous paths. However, in real-world applications, temporal data typically appears as a discretised collection of measurements, potentially irregularly-spaced and asynchronously observed. To apply the signature to this data, it first has to be converted into a continuous path. We believe this step to have a significant impact on the resulting signature, and thus also on models employing the signature. To assess this hypothesis, we explicitly treat this transformation as a *path imputation*, i.e. a mapping of the form $\phi\colon \mathcal{S}(\mathcal{X}^*) \to (\mathbb{R}\times\mathcal{X})^{[a,b]}$.

Task We aim to learn a function $g\colon \mathcal{S}(\mathcal{X}^*) \to \mathcal{Y}$, which decomposes to $g = F\circ\phi$, where F refers to a classifier, mapping from $(\mathbb{R}\times\mathcal{X})^{[a,b]}\times\mathcal{W}$ to $\mathcal{Y}$. Given a loss function ℓ and a set of p path imputation strategies, $\Phi = (\phi_i)_{i=1}^{p}$, we seek to minimise the objective:

$$\underset{\phi_i\in\Phi,\mathbf{w}\in\mathcal{W}}{\arg\min} \quad \mathbb{E}_{(\mathbf{x},y)\sim P(\mathcal{S}(\mathcal{X}^*),\mathcal{Y})}\left[\ell(g(\mathbf{x};\phi_i,\mathbf{w}),y)\right] \tag{7}$$

Even though Equation (7) could be formulated more *implicitly* (i.e. without any explicit imputation step), this formulation enables us to make explicit how the signature transform 'interprets' the raw data for downstream classification tasks. We further motivate this need for assessing the path construction in Section A.5 by showing that a single imputed value can affect the Levy area which is computed with the signature.

Path imputation strategies For our analysis, we consider the following set of strategies for path imputation, namely (1) linear interpolation, (2) forward filling, (3) indicator imputation, (4) zero imputation, (5) causal imputation[1], and (6) Gaussian process adapters (GP). Strategies 1–5 can be seen as a fixed preprocessing step, whereas GP adapters (strategy 6) are optimised end-to-end with the downstream task. For more details regarding these strategies, please refer to Section A.2 in the appendix. As indicated in Section 3, for computing the signature efficiently (i.e. computed in terms of standard tensor operations [3, Proposition A.3]), the imputed time series are transformed into piecewise linear paths beforehand.

[1]This strategy is similar to the time-joined transformation [27]. For more details, please refer to Section A.6 in the appendix.

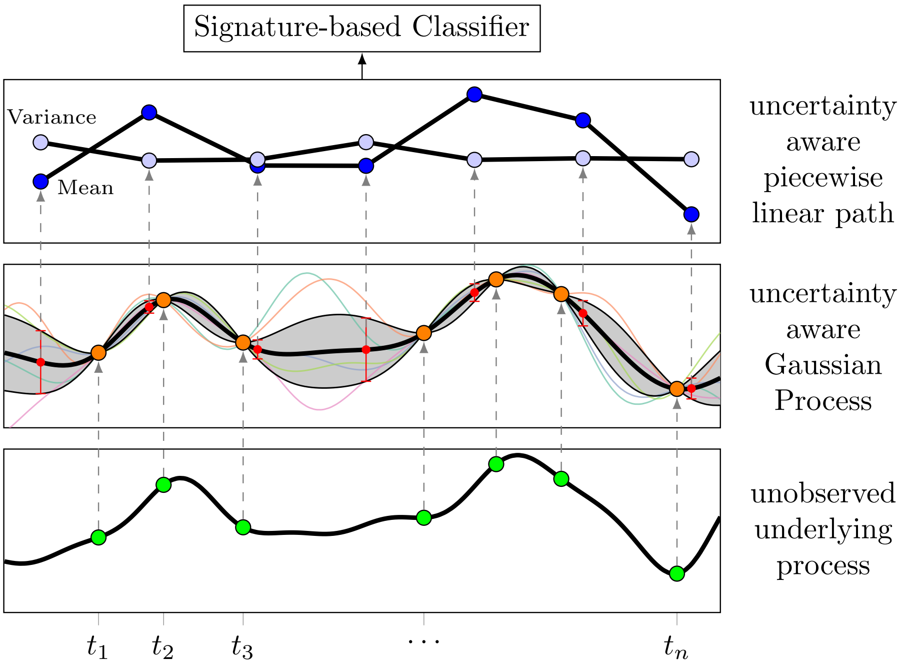


Figure 2: Overview of our proposed extension of GP adapters, GP-PoM, leveraging both posterior moments (mean and variance). In comparison, the conventional GP adapter feeds MC samples (faded colours in the background) drawn from the GP posterior into the classifier.

GP ADAPTER WITH POSTERIOR MOMENTS For conventional GP adapters, one major drawback with the formulations of Li and Marlin [29] and Futoma et al. [16], as described in Equation (3), is that approximating the expectation outside of the loss function with MC sampling is expensive. During prediction, Li and Marlin [29] proposed to overcome this issue by sacrificing the uncertainty in the loss function and to simply pass the posterior mean, as in Equation (6)[2]. To address both points, we propose to instead also pass the posterior covariance of the Gaussian process to the classifier F. This saves the cost of MC sampling whilst explicitly providing F with uncertainty information during the prediction[3]. However, the full covariance matrix may become very large, and it is not obvious that all interactions are relevant to the subsequent classifier. This is why we simplify matters by taking the posterior *variance* at every point, and concatenate it with the posterior mean at every point, to produce a path whose evolution also captures the uncertainty at every point:

$$\tau\colon [a,b] \times \mathcal{S}(\mathcal{X}^*) \times \mathcal{H} \to \mathcal{X} \times \mathcal{X} \tag{8}$$

$$\tau\colon t, \mathbf{x}, \eta \mapsto (\mu(t, \mathbf{x}, \eta), \Sigma(t, t, \mathbf{x}, \eta)). \tag{9}$$

This corresponds to solving

$$\operatorname*{arg\,min}_{\mathbf{w}\in\mathcal{W}, \eta\in\mathcal{H}} \sum_{k=1}^{N} \ell(F(\tau(\,\cdot\,, \mathbf{x}_k, \eta), \mathbf{w}), y_k), \tag{10}$$

where instead now $F\colon (\mathcal{X} \times \mathcal{X})^{[a,b]} \times \mathcal{W} \to \mathcal{Y}$. As this approach leverages information from both posterior moments (mean and variance), we refer to it as posterior moments GP adapter, or short GP-PoM. Figure 2 gives an overview of GP-PoM. In our context of interest, when F is a signature model, it is now straightforward to compute the signature of the Gaussian process, simply by querying many points to construct a piecewise linear approximation to the process. The choice of kernel has

[2] Equations (3) and (6) are of course not in general equal, so following Futoma et al. [16], our standard GP adapter uses MC sampling both in training and testing.

[3] Even if MC sampling is used during prediction, F has no per-sample access to uncertainty about the imputation.

non-trivial mathematical implications for this procedure: for example if a Matérn 1/2 kernel is chosen, then the resulting path is not of bounded variation and the definition of the signature transform given in Equation (1) does not hold, and rough path theory [32] must instead be invoked to define the signature transform. However, in this work we use RBF kernels, and therefore, this caveat does not apply to our case.

5. Experiments

We first introduce our experimental setup (datasets and model architectures) before presenting and discussing quantitative results.

Datasets and preprocessing We classify time series from four real-world datasets: (i) `PenDigits` [11], (ii) `CharacterTrajectories` [11], (iii) `LSST` [1], and (iv) `Physionet2012` [19]. For dataset statistics and necessary filtering steps, please refer to Section A.3 in the appendix. Moreover, to efficiently compute the signature, we sample the imputed path in a *fixed* time resolution[4], resulting in a piecewise linear path. For time series that are not irregularly spaced (this applies to all datasets but `Physionet2012`), we employ two types of random subsampling as an additional preprocessing step, namely (1) 'Random': Missing at random; on the instance level, we discard 50% of all observations. (2) 'Label-based': Missing not at random; for each class, we uniformly sample missingness frequencies between 40% and 60%. Since `PenDigits` consists of particularly short time series (8 steps, 2 dimensions), we use more moderate frequencies of 30% and 20–40%, respectively, for discarding observations. Finally, we standardise all time series channels using the empirical mean and standard deviation as determined on the entire training split.

Models We study the following models: (1) Sig, a simple signature model that involves a linear augmentation, the signature transform (signature block) and a final module of dense layers, (2) RNN, an RNN model using GRU cells [10], (3) RNNSig, which extends the signature transform to a window-based stream of signatures, and where the final neural module is a GRU sliding over the stream of signatures, and (4) DeepSig, a deep signature model sequentially employing two signature blocks featuring augmentation and signature transforms, following Bonnier et al. [3]. Please refer to Supplementary Section A.4 for more details about the architectures and implementations. We use the 'Signatory' package to calculate the signature transform [22], and implemented all GP adapters using the 'GPyTorch' framework [17].

Training and evaluation We use the predefined training and testing splits for each dataset, separating 20% of the training split as a validation set for hyperparameter tuning. For each setting, we run a randomised hyperparameter search of 20 calls and train each of these fits until convergence (at most 100 epochs; we stop early if the performance on the validation split does not improve for 20 epochs). As for performance metrics, for binary classification tasks, we optimise area under the precision-recall curve (as approximated via average precision) and also report AUROC. For multi-class classification, we optimise balanced accuracy (BAC) and additionally report accuracy and weighted AUROC (w-AUROC)[5]. Having selected the best hyperparameter configuration for each setting, we repeat 5 fits; for each fit, we select the best model state in terms of the best validation performance, and finally report mean and standard deviation (error bars) of the performance metrics on the testing split.

[4] For Physionet2012 hourly, for the other datasets once per originally observed time step

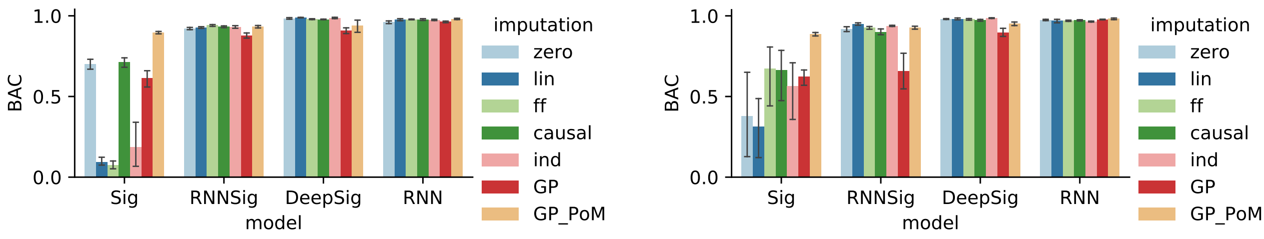
[5] AUROC is computed for each label and averaged with weights according to the support of each class

Table 1: `CharacterTrajectories` dataset under label-based subsampling. The top three methods are highlighted: bold & underlined, bold, underlined. All measures are reported as percentage points. Balanced accuracy (BAC) is the metric we optimised for. We further report accuracy and weighted AUROC (w-AUROC).

Imputation	Model	w-AUROC	BAC	Accuracy
GP-PoM	DeepSig	99.582 ± 0.671	95.155 ± 1.501	94.958 ± 1.716
	RNN	<u>99.973 ± 0.015</u>	**98.161 ± 0.664**	**98.273 ± 0.602**
	RNNSig	99.696 ± 0.089	92.778 ± 1.239	93.231 ± 1.133
	Sig	99.516 ± 0.075	88.627 ± 1.416	89.011 ± 1.319
GP	DeepSig	99.290 ± 0.704	89.545 ± 2.996	89.368 ± 3.123
	RNN	99.970 ± 0.011	97.712 ± 0.266	97.873 ± 0.251
	RNNSig	96.669 ± 2.393	65.717 ± 13.691	67.052 ± 13.182
	Sig	95.283 ± 1.602	62.423 ± 6.110	63.614 ± 5.958
causal	DeepSig	99.940 ± 0.024	97.272 ± 0.709	97.437 ± 0.620
	RNN	99.960 ± 0.010	97.239 ± 0.516	97.409 ± 0.481
	RNNSig	99.523 ± 0.155	89.922 ± 2.301	90.585 ± 2.186
	Sig	95.747 ± 4.957	66.307 ± 21.794	68.259 ± 20.757
forward-filling	DeepSig	99.953 ± 0.041	97.956 ± 0.677	98.078 ± 0.656
	RNN	99.942 ± 0.011	96.942 ± 0.486	97.159 ± 0.444
	RNNSig	99.720 ± 0.071	92.568 ± 1.091	93.148 ± 1.011
	Sig	94.828 ± 8.117	67.169 ± 26.338	68.649 ± 26.125
indicator	DeepSig	<u>**99.988 ± 0.013**</u>	<u>**98.591 ± 0.294**</u>	<u>**98.719 ± 0.263**</u>
	RNN	99.916 ± 0.020	96.414 ± 0.406	96.671 ± 0.367
	RNNSig	99.802 ± 0.032	93.787 ± 0.463	94.234 ± 0.442
	Sig	91.661 ± 10.003	56.423 ± 22.796	58.384 ± 22.932
linear	DeepSig	99.970 ± 0.010	<u>98.051 ± 0.743</u>	<u>98.217 ± 0.671</u>
	RNN	99.880 ± 0.059	96.906 ± 1.314	97.117 ± 1.196
	RNNSig	99.876 ± 0.035	94.848 ± 0.916	95.292 ± 0.842
	Sig	80.442 ± 18.228	31.193 ± 23.962	32.326 ± 24.679
zero	DeepSig	**99.977 ± 0.010**	98.030 ± 0.357	98.189 ± 0.358
	RNN	99.967 ± 0.014	97.428 ± 0.572	97.549 ± 0.596
	RNNSig	99.699 ± 0.132	91.752 ± 1.782	92.368 ± 1.662
	Sig	77.727 ± 23.671	37.992 ± 34.456	38.955 ± 35.232

Table 2: `PenDigits` dataset under label-based subsampling. The top three methods are highlighted: bold & underlined, bold, underlined. All measures are reported as percentage points. Balanced accuracy (BAC) is the metric we optimised for. We further report accuracy and weighted AUROC (w-AUROC)

Imputation	Model	w-AUROC	BAC	Accuracy
GP-PoM	DeepSig	<u>99.930 ± 0.032</u>	**97.403 ± 0.300**	**97.381 ± 0.298**
	RNN	99.901 ± 0.016	96.349 ± 0.297	96.306 ± 0.302
	RNNSig	99.669 ± 0.073	93.022 ± 0.765	92.967 ± 0.763
	Sig	99.150 ± 0.144	88.090 ± 1.493	87.999 ± 1.499
GP	DeepSig	92.885 ± 1.455	60.593 ± 4.092	60.476 ± 4.067
	RNN	95.170 ± 1.438	67.543 ± 4.782	67.426 ± 4.790
	RNNSig	84.501 ± 1.307	42.184 ± 1.977	42.141 ± 1.913
	Sig	80.312 ± 2.655	37.767 ± 3.611	37.725 ± 3.646
causal	DeepSig	99.241 ± 0.075	89.616 ± 0.749	89.514 ± 0.747
	RNN	99.241 ± 0.098	89.496 ± 0.480	89.417 ± 0.501
	RNNSig	99.298 ± 0.041	89.187 ± 0.476	89.137 ± 0.494
	Sig	98.374 ± 0.065	83.205 ± 0.404	83.082 ± 0.426
forward-filling	DeepSig	99.007 ± 0.072	88.205 ± 0.434	88.090 ± 0.428
	RNN	99.333 ± 0.046	89.747 ± 0.406	89.657 ± 0.419
	RNNSig	99.274 ± 0.015	89.788 ± 0.384	89.743 ± 0.392
	Sig	98.310 ± 0.045	83.739 ± 0.421	83.625 ± 0.398
indicator	DeepSig	<u>**99.960 ± 0.013**</u>	<u>**98.068 ± 0.184**</u>	<u>**98.056 ± 0.185**</u>
	RNN	**99.955 ± 0.009**	<u>97.266 ± 0.439</u>	<u>97.238 ± 0.447</u>
	RNNSig	99.747 ± 0.028	93.488 ± 0.616	93.408 ± 0.613
	Sig	99.410 ± 0.031	90.591 ± 0.306	90.492 ± 0.308
linear	DeepSig	99.458 ± 0.052	91.567 ± 0.412	91.452 ± 0.416
	RNN	99.489 ± 0.093	91.608 ± 0.609	91.492 ± 0.608
	RNNSig	99.446 ± 0.039	90.259 ± 0.859	90.143 ± 0.869
	Sig	98.963 ± 0.084	87.254 ± 0.437	87.141 ± 0.458
zero	DeepSig	99.391 ± 0.071	91.121 ± 0.406	91.012 ± 0.403
	RNN	99.551 ± 0.031	91.765 ± 0.283	91.670 ± 0.304
	RNNSig	99.321 ± 0.033	89.543 ± 0.412	89.457 ± 0.417
	Sig	98.544 ± 0.069	84.269 ± 0.445	84.185 ± 0.454



(a) CharacterTrajectories-R

(b) CharacterTrajectories-L

Figure 3: Experimental results visualised for `CharacterTrajectories` dataset. The bars display performance in terms of balanced accuracy (BAC), whereas the panels indicate the subsampling strategy. Left: Random subsampling (R), right: label-based subsampling (L).

Results In Tables 1 and 2, the results for `CharacterTrajectories` and `PenDigits` under label-based subsampling are shown, respectively. For the remaining datasets and subsampling strategies, please refer to Tables 3–7 in the appendix. We observe that both DeepSig as well as the signature-free RNN perform well over many scenarios. In particular, they are impervious to the choice of several imputation schemes in the sense that it does not have a large impact on their predictive performance. However, we also see that certain signature models, in particular Sig, are heavily impacted by the choice of imputation strategy. Figure 3 exemplifies this finding in a barplot visualization; for the remaining visualizations, including the number of parameters of the optimised models, please refer to Supplementary Figures 4 and 5. In the case of `CharacterTrajectories`, Sig was only able to achieve acceptable performance through our novel GP-PoM strategy. In `PenDigits`, we encountered issues of numerical stability for the original GP adapter[6]; not so for GP-PoM. Furthermore, we found that GP-PoM tends to converge faster to a better performance than the original GP adapter, as exemplified in Supplementary Figure 6.

6. Discussion

Our findings suggest that the choice of path imputation strategy can *drastically* affect the performance of signature-based models. We observe this most prominently in 'shallow' signature models, whereas deep signature models (DeepSig) are more robust in tackling irregular time series over different imputations—comparable to non-signature RNNs, yet on average being more parameter-efficient.

Overall, we find that uncertainty-aware approaches (indicator imputation and GP-PoM) are beneficial when imputing irregularly-spaced time series for classification. Crucially, uncertainty information has to be accessible during the *prediction step*. We find that this is indeed not the case for the standard GP adapter (despite the naming of 'uncertainty-aware framework'), since for each MC sample, the downstream classifier has no access to missingness or uncertainty about the underlying imputation. GP-PoM, our proposed end-to-end imputation strategy, shows competitive classification performance, while considerably improving upon the existing GP adapter. As for limitations, GP-PoM sacrifices the GP adapter's ability to be explicitly uncertain *about* its own prediction (due to the variance of the MC sampled predictions), while the subsequent classifier has to be able to handle the doubled feature dimensionality.

Recommendations for the practitioner When dealing with a challenging time series classification task, we recommend to consider signatures as a powerful tool to encode paths with little loss of information. However, we observe that this comes at a certain cost: since the signature describes

[6]They were addressed by jittering the diagonal in the Cholesky decomposition.

continuous paths (and not discrete time series), constructing this path from raw data is a delicate task that can heavily impact the signature and the performance of downstream models. To this end, we recommend using GP-PoM, which explicitly captures uncertainty in the imputed path. Given our findings, indicator imputation is a simple but promising go-to strategy, however we caution its use together with shallow signature models since we observed detrimental effects in terms of predictive performance. Furthermore, when applying signatures in online applications or settings, where during training no data should leak from the future (e.g. in online settings, this could impair performance upon deployment), we recommend to use causal (or time-joined) path imputations: their design specifically prevents leakage from the future, even if the signature interprets the imputations as knots of a piece-wise linear path.

7. Conclusion

The signature transform has recently gained attention for being a promising feature extractor that can be easily integrated to neural networks. As we empirically demonstrated in this paper, the application of signature transforms to real-world temporal data is fraught with pitfalls—specifically, we found the choice of an imputation scheme to be crucial for obtaining high predictive performance. Moreover, by integrating uncertainty to the prediction step, our proposed GP-PoM has demonstrated overall competitive performance and in particular improved robustness in signature models when classifying irregularly-spaced and asynchronous time series.

Broader Impact

Whilst the task of converting observed data into a path in data space is particularly important for signatures, it also arises in the context of, for example, convolutional and recurrent neural networks.

Convolutions are often thought of in terms of discrete sums, but they are perhaps more naturally described as the integral cross-correlation between the underlying data path f and the learnt filter g_θ. Given sample points $t_1, \ldots, t_n \in [0, T]$, this integral is then approximated via numerical quadrature:

$$\frac{1}{T}\int_0^T f(t)g_\theta(t)\mathrm{d}t \approx \frac{1}{n}\sum_{i=1}^{n} f(t_i)g_\theta(t_i),$$

although the $1/n$ scaling is really only justified in the case that the t_i are equally spaced.[7] Thus we see that with convolutions, we are implicitly interpreting the observed data as a path in data space.

Similarly, the connections between dynamical systems and recurrent neural networks are well known [2, 15], and these tend to use a similar setup. For non-signature methods as for signature methods, this implicit usage of data as a path in data space often seems to be swept under the rug, and we have demonstrated that this is deserving further attention.

With respect to ethical considerations, we acknowledge that time series models in general can be used for better or for worse. On top of that, implicit biases underlying models as well as datasets can have unintended harmful consequences. By shedding light on the impact of implicit usage of paths in data space we hope to forward understanding and accountability of black-box models. Even if our work focuses on signature models, we believe that the principle of making underlying assumptions explicit is relevant far beyond this class of models.

[7] The g_θ is typically a step function in 'normal' convolutional layers. Some works exists on replacing it with e.g. B-splines [13] to better handle irregular data. The oddity of scaling by $1/n$ with irregular data has not been explicitly addressed in the literature, at least to our knowledge; indeed quite conversely we have seen it used without remark.

8. Acknowledgements

M.M, M.H, and C.B were supported by the SNSF Starting Grant 'Significant Pattern Mining'. The authors are grateful to Patrick Kidger for valuable discussions, guidance, and code contributions.

References

[1] Allam Jr, T., Bahmanyar, A., Biswas, R., Dai, M., Galbany, L., Hložek, R., Ishida, E. E., Jha, S. W., Jones, D. O., Kessler, R., et al. (2018). The photometric lsst astronomical time-series classification challenge (plasticc): Data set. *arXiv preprint arXiv:1810.00001*.

[2] Bailer-Jones, C., MacKay, D., and Withers, P. (1998). A recurrent neural network for modelling dynamical systems. *Network: Computation in Neural Systems*, 9:531–47.

[3] Bonnier, P., Kidger, P., Perez Arribas, I., Salvi, C., and Lyons, T. (2019). Deep Signature Transforms. In *Advances in Neural Information Processing Systems*, pages 3099–3109.

[4] Che, Z., Purushotham, S., Cho, K., Sontag, D., and Liu, Y. (2018). Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):1–12.

[5] Chen, K. T. (1954). Iterated integrals and exponential homomorphisms. *Proc. London Math. Soc, 4, 502–512*.

[6] Chen, K. T. (1957). Integration of paths, geometric invariants and a generalized Baker-Hausdorff formula. *Ann. of Math. (2), 65:163–178*.

[7] Chen, K. T. (1958). Integration of paths - a faithful representation of paths by non-commutative formal power series. *Trans. Amer. Math. Soc. 89 (1958), 395–407*.

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

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

[10] 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. *arXiv:1406.1078*.

[11] Dua, D. and Graff, C. (2017). UCI machine learning repository.

[12] Fermanian, A. (2019). Embedding and learning with signatures. *arXiv:1911.13211*.

[13] Fey, M., Eric Lenssen, J., Weichert, F., and Müller, H. (2018). SplineCNN: Fast Geometric Deep Learning With Continuous B-Spline Kernels. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

[15] Funahashi, K.-i. and Nakamura, Y. (1993). Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Networks*, 6(6):801 – 806.

[16] Futoma, J., Hariharan, S., and Heller, K. (2017). Learning to Detect Sepsis with a Multitask Gaussian Process RNN Classifier. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1174–1182.

[17] Gardner, J., Pleiss, G., Weinberger, K. Q., Bindel, D., and Wilson, A. G. (2018). GPyTorch: Blackbox matrix-matrix Gaussian Process inference with GPU acceleration. In *Advances in Neural Information Processing Systems 31*, pages 7576–7586.

[18] Gelman, A. and Hill, J. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

[19] Goldberger, A. L., Amaral, L. A., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., and Stanley, H. E. (2000). Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220.

[20] Hambly, B. M. and Lyons, T. J. (2010). Uniqueness for the signature of a path of bounded variation and the reduced path group. *Annals of Mathematics*, 171(1):109–167.

[21] Horn, M., Moor, M., Bock, C., Rieck, B., and Borgwardt, K. (2019). Set functions for time series. *arXiv preprint arXiv:1909.12064*.

[22] 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`.

[23] Kidger, P., Morrill, J., Foster, J., and Lyons, T. (2020). Neural Controlled Differential Equations for Irregular Time Series. *arXiv:2005.08926*.

[24] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[25] Király, F. J. and Oberhauser, H. (2019). Kernels for sequentially ordered data. *Journal of Machine Learning Research*.

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

[27] 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*.

[28] Li, C., Zhang, X., and Jin, L. (2017). LPSNet: a novel log path signature feature based hand gesture recognition framework. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 631–639.

[29] Li, S. C.-X. and Marlin, B. M. (2016). A scalable end-to-end Gaussian process adapter for irregularly sampled time series classification. In *Advances in Neural Information Processing Systems*, pages 1804–1812.

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

[31] Lyons, T. (2014). Rough paths, signatures and the modelling of functions on streams. *arXiv:1405.4537*.

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

[33] Moor, M., Horn, M., Rieck, B., Roqueiro, D., and Borgwardt, K. (2019). Early recognition of sepsis with gaussian process temporal convolutional networks and dynamic time warping. In *Machine Learning for Healthcare Conference*, pages 2–26.

[34] Morrill, J., Kormilitzin, A., Nevado-Holgado, A., Swaminathan, S., Howison, S., and Lyons, T. (2019a). The signature-based model for early detection of sepsis from electronic health records in the intensive care unit. In *2019 Computing in Cardiology (CinC)*, pages Page–1. IEEE.

[35] Morrill, J., Kormilitzin, A., Nevado-Holgado, A., Swaminathan, S., Howison, S., and Lyons, T. (2019b). The Signature-based Model for Early Detection of Sepsis from Electronic Health Records in the Intensive Care Unit. *International Conference in Computing in Cardiology*.

[36] 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., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035.

[37] Perez Arribas, I., Goodwin, G. M., Geddes, J. R., Lyons, T., and Saunders, K. E. A. (2018). A signature-based machine learning model for distinguishing bipolar disorder and borderline personality disorder. *Translational Psychiatry*, 8(1):274.

[38] Reizenstein, J. (2019). *Iterated-integral signatures in machine learning*. PhD thesis, University of Warwick. http://wrap.warwick.ac.uk/131162/.

[39] Reizenstein, J. and Graham, B. (2018). The iisignature library: efficient calculation of iterated-integral signatures and log signatures. *arXiv:1802.08252*.

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

[41] Rubanova, Y., Chen, T. Q., and Duvenaud, D. K. (2019). Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems*, pages 5321–5331.

[42] Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63(3):581–592.

[43] Shukla, S. N. and Marlin, B. (2019). Interpolation-prediction networks for irregularly sampled time series. In *International Conference on Learning Representations*.

[44] Toth, C. and Oberhauser, H. (2019). Variational Gaussian Processes with Signature Covariances. *arXiv:1906.08215*.

[45] Yang, W., Jin, L., Ni, H., and Lyons, T. (2016). Rotation-free online handwritten character recognition using dyadic path signature features, hanging normalization, and deep neural network. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 4083–4088. IEEE.

[46] Yang, W., Lyons, T., Ni, H., Schmid, C., Jin, L., and Chang, J. (2017). Leveraging the path signature for skeleton-based human action recognition. *arXiv:1707.03993*.

A. Appendix

A.1. Further Experiments

Table 3: `CharacterTrajectories`, random subsampling

Imputation	Model	w-AUROC	BAC	Accuracy
GP-PoM	DeepSig	99.698 ± 0.393	94.011 ± 5.037	93.635 ± 5.335
	RNN	$\underline{99.970 \pm 0.011}$	98.011 ± 0.512	98.106 ± 0.508
	RNNSig	99.787 ± 0.074	93.308 ± 0.960	93.844 ± 0.903
	Sig	99.578 ± 0.031	89.570 ± 0.938	89.930 ± 0.914
GP	DeepSig	98.994 ± 1.088	90.821 ± 2.361	90.471 ± 2.347
	RNN	99.909 ± 0.032	96.276 ± 0.691	96.492 ± 0.715
	RNNSig	99.400 ± 0.094	87.587 ± 2.054	88.141 ± 1.959
	Sig	94.862 ± 1.779	61.280 ± 6.440	62.446 ± 6.493
causal	DeepSig	99.963 ± 0.023	97.774 ± 0.228	97.953 ± 0.182
	RNN	99.953 ± 0.023	97.657 ± 0.720	97.813 ± 0.676
	RNNSig	99.814 ± 0.044	93.268 ± 0.730	93.747 ± 0.743
	Sig	96.736 ± 0.578	71.393 ± 3.784	73.245 ± 3.642
forward-filling	DeepSig	99.965 ± 0.030	97.974 ± 0.381	98.120 ± 0.365
	RNN	99.954 ± 0.010	97.786 ± 0.308	97.939 ± 0.281
	RNNSig	99.840 ± 0.047	94.110 ± 0.774	94.596 ± 0.745
	Sig	54.308 ± 4.187	7.387 ± 2.995	7.117 ± 2.417
indicator	DeepSig	99.955 ± 0.033	$\mathbf{98.626 \pm 0.500}$	$\mathbf{98.733 \pm 0.481}$
	RNN	99.953 ± 0.024	97.502 ± 0.527	97.660 ± 0.499
	RNNSig	99.755 ± 0.078	93.091 ± 1.056	93.635 ± 0.952
	Sig	66.917 ± 18.306	18.481 ± 18.692	19.067 ± 19.165
linear	DeepSig	$\underline{\mathbf{99.984 \pm 0.007}}$	$\underline{\mathbf{98.898 \pm 0.205}}$	$\underline{\mathbf{98.997 \pm 0.201}}$
	RNN	99.928 ± 0.043	97.668 ± 0.897	97.786 ± 0.802
	RNNSig	99.767 ± 0.037	92.754 ± 0.662	93.273 ± 0.656
	Sig	55.023 ± 6.655	9.436 ± 3.349	9.958 ± 4.097
zero	DeepSig	$\mathbf{99.980 \pm 0.013}$	$\underline{98.337 \pm 0.644}$	$\underline{98.454 \pm 0.616}$
	RNN	99.887 ± 0.052	96.004 ± 1.074	96.253 ± 1.046
	RNNSig	99.685 ± 0.063	92.154 ± 0.878	92.744 ± 0.820
	Sig	96.997 ± 0.388	69.963 ± 4.208	71.699 ± 4.002

A.2. Imputation strategies

We consider the following set of strategies for path imputation, i.e.

1. linear interpolation: At a given imputation point, the previous and next observed data point are linearly interpolated. Missing values at the start or end of the time series are imputed with 0 which for standardised data also corresponds to the mean.

2. forward filling: At a given imputation point, the last observed value is carried forward. Missing values at the start of the time series are imputed with 0.

Table 4: `PenDigits`, random subsampling

metric	w-AUROC	BAC	Accuracy	
GP-PoM	DeepSig	99.515 ± 0.078	92.151 ± 0.555	92.098 ± 0.548
	RNN	99.564 ± 0.072	92.757 ± 0.735	92.699 ± 0.733
	RNNSig	98.967 ± 0.253	88.148 ± 1.588	88.113 ± 1.579
	Sig	99.028 ± 0.099	87.352 ± 0.898	87.290 ± 0.903
GP	DeepSig	90.509 ± 0.164	54.545 ± 0.426	54.513 ± 0.451
	RNN	91.961 ± 0.856	57.930 ± 2.079	57.900 ± 2.088
	RNNSig	86.740 ± 0.585	46.842 ± 1.255	46.867 ± 1.218
	Sig	83.511 ± 0.485	41.747 ± 0.428	41.809 ± 0.425
causal	DeepSig	99.096 ± 0.116	89.480 ± 0.359	89.434 ± 0.362
	RNN	99.288 ± 0.066	89.526 ± 0.535	89.474 ± 0.539
	RNNSig	99.165 ± 0.067	88.807 ± 0.613	88.759 ± 0.617
	Sig	97.870 ± 0.224	80.065 ± 0.980	80.011 ± 0.971
forward-filling	DeepSig	99.141 ± 0.068	88.974 ± 0.656	88.902 ± 0.644
	RNN	99.311 ± 0.067	90.067 ± 0.247	90.029 ± 0.247
	RNNSig	99.203 ± 0.063	88.930 ± 0.513	88.902 ± 0.528
	Sig	98.425 ± 0.069	84.458 ± 0.468	84.374 ± 0.477
indicator	DeepSig	**99.607 ± 0.059**	**93.156 ± 0.738**	**93.087 ± 0.751**
	RNN	**99.733 ± 0.044**	**94.124 ± 0.412**	**94.071 ± 0.415**
	RNNSig	99.549 ± 0.041	91.604 ± 0.278	91.532 ± 0.268
	Sig	98.708 ± 0.040	84.544 ± 0.538	84.505 ± 0.563
linear	DeepSig	99.407 ± 0.151	91.418 ± 1.075	91.366 ± 1.086
	RNN	99.510 ± 0.041	91.862 ± 0.582	91.812 ± 0.594
	RNNSig	99.591 ± 0.036	91.556 ± 0.518	91.521 ± 0.539
	Sig	99.029 ± 0.094	87.116 ± 0.612	87.038 ± 0.612
zero	DeepSig	99.334 ± 0.077	89.774 ± 0.541	89.686 ± 0.553
	RNN	99.403 ± 0.112	90.729 ± 0.618	90.698 ± 0.620
	RNNSig	99.150 ± 0.046	87.948 ± 0.248	87.879 ± 0.243
	Sig	98.623 ± 0.073	83.935 ± 0.382	83.905 ± 0.375

Table 5: LSST, label-based subsampling

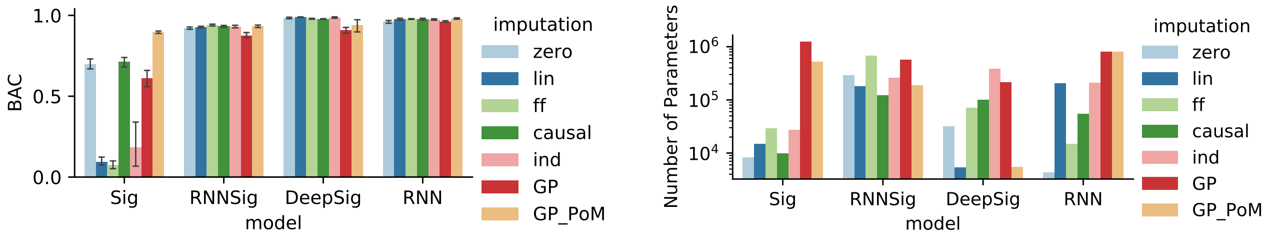
Imputation	Model	w-AUROC	BAC	Accuracy
GP-PoM	DeepSig	64.414 ± 6.770	13.857 ± 2.394	8.743 ± 2.889
	RNN	82.808 ± 4.284	29.483 ± 7.106	29.781 ± 11.520
	RNNSig	82.934 ± 1.495	31.751 ± 2.338	35.442 ± 3.621
	Sig	58.820 ± 1.427	13.779 ± 0.535	10.187 ± 1.418
GP	DeepSig	59.686 ± 3.444	13.144 ± 3.924	28.333 ± 5.465
	RNN	69.269 ± 1.469	16.558 ± 0.465	33.903 ± 0.356
	RNNSig	60.032 ± 0.476	16.779 ± 1.559	33.627 ± 0.213
	Sig	57.381 ± 0.555	14.062 ± 1.174	24.294 ± 2.135
causal	DeepSig	75.336 ± 3.543	25.481 ± 5.868	39.400 ± 5.824
	RNN	84.107 ± 0.606	37.762 ± 1.736	53.009 ± 1.684
	RNNSig	82.343 ± 0.235	33.570 ± 2.776	50.284 ± 0.973
	Sig	58.837 ± 3.205	12.262 ± 2.841	34.161 ± 1.295
forward-filling	DeepSig	77.758 ± 2.562	27.473 ± 4.165	42.238 ± 5.435
	RNN	84.153 ± 0.675	38.621 ± 1.618	52.976 ± 0.667
	RNNSig	82.430 ± 0.439	34.078 ± 1.399	50.560 ± 0.678
	Sig	64.200 ± 0.808	14.779 ± 1.184	35.255 ± 0.520
indicator	DeepSig	**95.351 ± 1.044**	**52.124 ± 2.147**	**77.283 ± 2.231**
	RNN	**98.132 ± 0.242**	**61.893 ± 3.862**	**83.609 ± 1.452**
	RNNSig	82.635 ± 1.816	29.122 ± 3.102	41.152 ± 2.456
	Sig	57.084 ± 0.863	12.806 ± 0.951	32.620 ± 1.113
linear	DeepSig	73.965 ± 3.208	21.356 ± 0.745	36.504 ± 6.349
	RNN	84.931 ± 0.301	40.098 ± 0.968	54.023 ± 0.822
	RNNSig	82.883 ± 0.805	32.553 ± 1.276	49.327 ± 2.426
	Sig	66.687 ± 1.812	17.340 ± 0.777	35.726 ± 0.571
zero	DeepSig	75.783 ± 1.217	23.598 ± 3.268	42.019 ± 1.718
	RNN	87.908 ± 0.952	40.479 ± 1.890	53.268 ± 2.066
	RNNSig	79.432 ± 0.496	34.381 ± 1.176	46.521 ± 0.446
	Sig	52.972 ± 1.080	11.951 ± 1.116	28.532 ± 8.378

Table 6: LSST, random subsampling

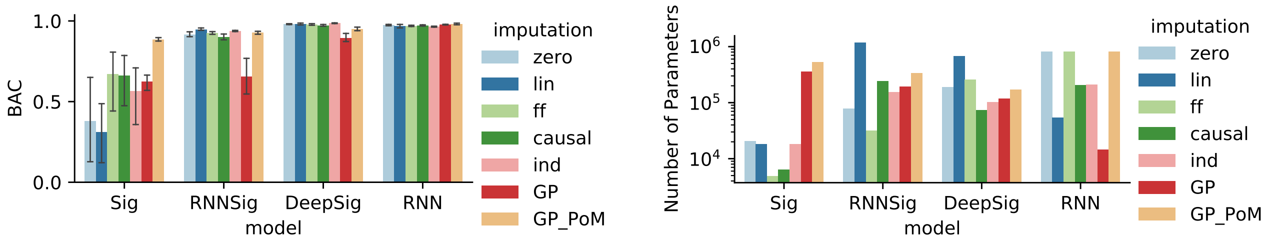
Imputation	Model	w-AUROC	BAC	Accuracy
GP-PoM	DeepSig	70.771 ± 2.903	21.823 ± 2.656	40.114 ± 2.400
	RNN	82.322 ± 0.833	<u>37.372 ± 1.945</u>	52.506 ± 2.559
	RNNSig	75.839 ± 1.576	26.680 ± 1.854	41.792 ± 1.854
	Sig	58.799 ± 0.561	13.200 ± 1.300	34.615 ± 0.802
GP	DeepSig	62.133 ± 0.957	16.191 ± 1.807	34.324 ± 0.161
	RNN	62.638 ± 2.952	17.363 ± 2.249	34.515 ± 1.155
	RNNSig	60.593 ± 0.405	17.138 ± 1.515	33.710 ± 0.651
	Sig	57.978 ± 1.329	13.614 ± 1.147	33.104 ± 0.400
causal	DeepSig	74.266 ± 2.119	22.177 ± 2.683	34.096 ± 6.644
	RNN	**83.938 ± 0.729**	**37.407 ± 2.269**	<u>**54.558 ± 1.074**</u>
	RNNSig	77.195 ± 5.004	31.676 ± 5.485	46.399 ± 5.323
	Sig	52.682 ± 0.817	11.625 ± 0.808	26.026 ± 9.910
forward-filling	DeepSig	78.760 ± 1.204	27.871 ± 2.226	46.853 ± 1.426
	RNN	<u>**84.267 ± 0.570**</u>	<u>**38.320 ± 0.546**</u>	<u>53.236 ± 1.298</u>
	RNNSig	82.291 ± 0.348	33.517 ± 1.103	50.203 ± 1.081
	Sig	56.031 ± 2.311	12.981 ± 1.521	28.516 ± 11.770
indicator	DeepSig	69.863 ± 1.579	20.810 ± 1.165	36.399 ± 3.986
	RNN	76.956 ± 2.996	29.178 ± 2.580	42.182 ± 4.442
	RNNSig	63.700 ± 1.525	18.367 ± 1.625	30.308 ± 1.881
	Sig	53.831 ± 1.467	12.214 ± 2.317	32.668 ± 0.363
linear	DeepSig	75.163 ± 4.039	23.657 ± 4.388	40.324 ± 5.440
	RNN	<u>83.777 ± 0.512</u>	36.819 ± 2.375	**53.439 ± 0.607**
	RNNSig	81.588 ± 1.000	29.814 ± 1.953	49.286 ± 1.578
	Sig	65.499 ± 3.060	17.113 ± 3.513	36.334 ± 0.979
zero	DeepSig	69.513 ± 4.856	15.231 ± 4.668	35.272 ± 4.149
	RNN	81.739 ± 0.416	35.580 ± 2.688	50.073 ± 2.167
	RNNSig	77.597 ± 0.632	31.294 ± 1.615	45.953 ± 1.286
	Sig	52.900 ± 1.487	11.870 ± 2.303	32.612 ± 0.650

Table 7: `Physionet 2012`

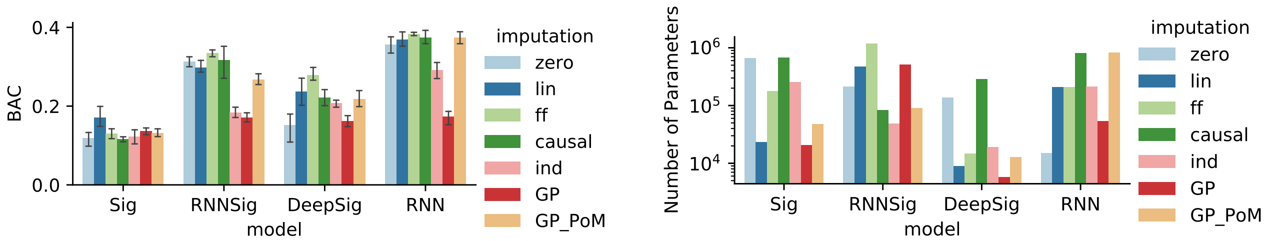
Imputation	Model	AUROC	Average Precision
GP-PoM	DeepSig	82.084 ± 0.836	47.858 ± 1.362
	RNN	83.222 ± 0.570	49.263 ± 1.115
	RNNSig	77.879 ± 1.072	40.157 ± 0.539
	Sig	74.388 ± 2.211	33.909 ± 2.902
GP	DeepSig	82.164 ± 0.245	47.707 ± 0.971
	RNN	81.196 ± 0.953	47.322 ± 1.642
	RNNSig	74.665 ± 1.763	36.585 ± 1.549
	Sig	70.984 ± 1.611	30.886 ± 2.115
causal	DeepSig	83.487 ± 0.574	48.924 ± 0.935
	RNN	$\underline{84.689 \pm 0.325}$	$\underline{\mathbf{52.646 \pm 0.460}}$
	RNNSig	82.074 ± 0.080	47.691 ± 0.214
	Sig	83.499 ± 0.597	47.809 ± 0.783
forward-filling	DeepSig	82.766 ± 0.646	47.971 ± 1.562
	RNN	$\underline{\mathbf{84.954 \pm 0.157}}$	$\mathbf{52.427 \pm 0.521}$
	RNNSig	80.916 ± 0.607	44.979 ± 1.347
	Sig	84.328 ± 0.213	51.043 ± 0.547
indicator	DeepSig	82.332 ± 0.467	47.150 ± 1.020
	RNN	$\mathbf{84.906 \pm 0.211}$	$\underline{51.887 \pm 0.713}$
	RNNSig	83.651 ± 0.199	49.872 ± 0.570
	Sig	81.570 ± 0.473	45.807 ± 1.318
linear	DeepSig	83.168 ± 0.650	48.937 ± 1.291
	RNN	84.367 ± 0.176	50.431 ± 0.340
	RNNSig	77.336 ± 0.469	41.888 ± 0.583
	Sig	83.354 ± 0.223	48.900 ± 0.480
zero	DeepSig	80.101 ± 2.126	44.340 ± 3.076
	RNN	83.571 ± 0.227	49.304 ± 0.597
	RNNSig	76.246 ± 1.436	39.502 ± 2.232
	Sig	80.645 ± 0.097	44.728 ± 0.264



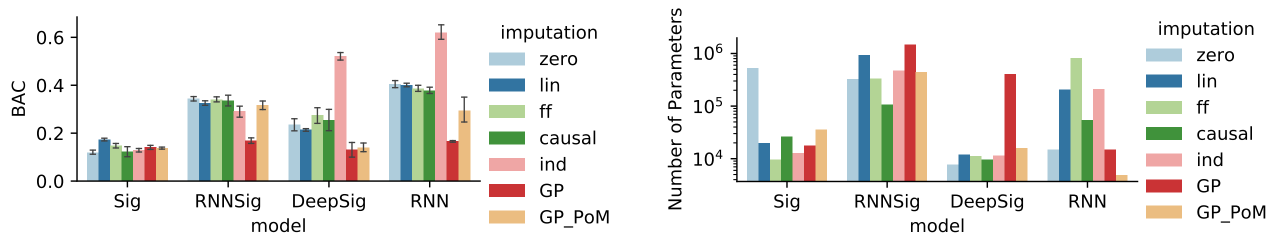
(a) CharacterTrajectories-R



(b) CharacterTrajectories-L

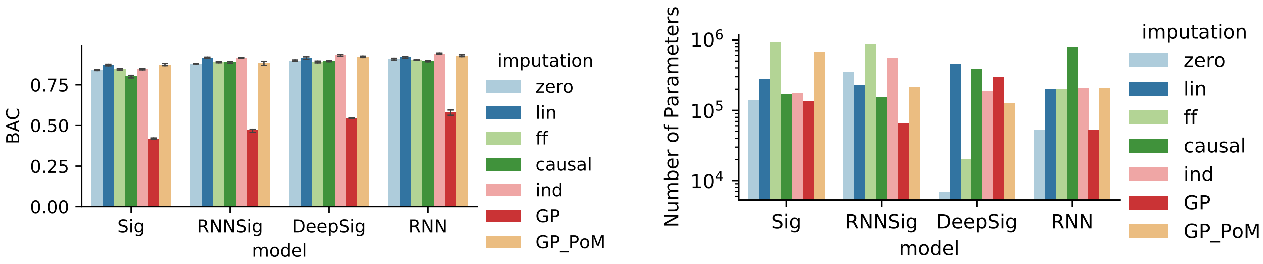


(c) LSST-R

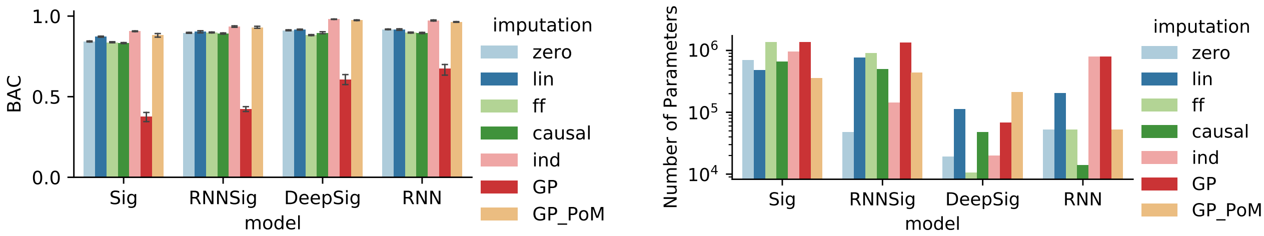


(d) LSST-L

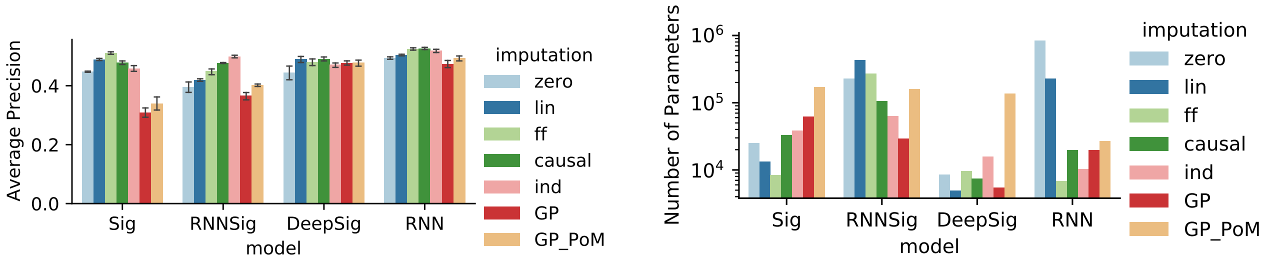
Figure 4: Visualisations for `CharacterTrajectories` and `LSST` . The rows indicate datasets and different subsampling schemes (R for Random, L for Label-based). The left column displays the performance metric which was optimzied for: balanced accuracy (BAC), or average precision. The right column indicates the number of trainable parameters which the best model required (as selected in the hyperparameter search).



(a) PenDigits-R



(b) PenDigits-L



(c) Physionet2012

Figure 5: Visualisations for `PenDigits` and `Physionet` . The rows indicate datasets and different subsampling schemes (R for Random, L for Label-based). The left column displays the performance metric which was optimzied for: balanced accuracy (BAC), or average precision. The right column indicates the number of trainable parameters which the best model required (as selected in the hyperparameter search).

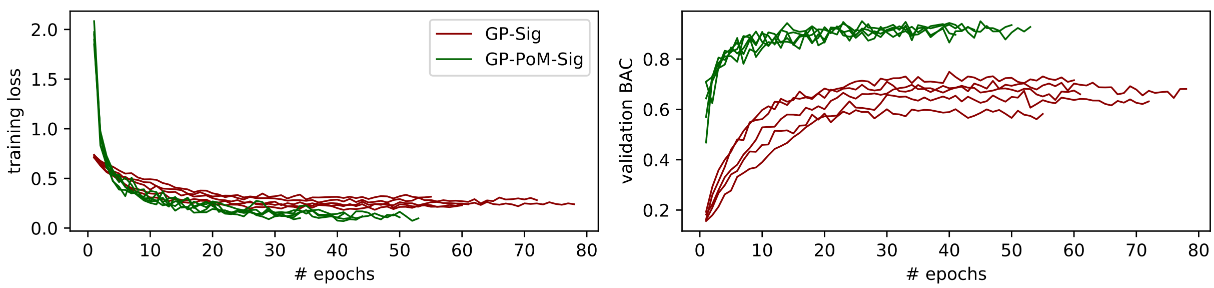


Figure 6: GP-PoM training illustrated for CharacterTrajectories as compared to conventional GP adapter.

3. indicator imputation: At a given imputation point, for each feature dimension, if no observation is available a binary missingness indicator variable is set to 1, 0 otherwise. The missing value is filled with 0.

4. zero imputation: At a given imputation point, missing values are filled with 0.

5. causal imputation: This approach is related to forward filling and motivated by signature theory. As opposed to forward filling, the time and the actual value are updated sequentially. For more details, we introduce causal imputation in Section A.6.

6. Gaussian process adapter: We introduce GP adapters in Section 3, where $\mathbf{z}$ refers to the imputed time series (modelled as Gaussian distribution).

A.3. Dataset statistics and filtering

Physionet2012 As our focus is time series classification, for `Physionet2012` [19], we included the 36 time series variables, and excluded the static covariates (notably, we counted the variable 'weight' as a static covariate). Subsequently, we excluded the following 12 icu stays (here represented by there ids) for having no time series data (but only static covariates): 140501, 150649, 140936, 143656, 141264, 145611, 142998, 147514, 142731, 150309, 155655, 156254, and a single noisy encounter, 135365, which contained much more observations than all other patients. After these filtering steps, we count 11987 instances and a binary class label, whether a patient survives the hospital stay or not.

PenDigits For `PenDigits` [11], we count 10992 samples, featuring 2 channels and 8 time steps, and 10 classes.

LSST LSST [1] contains 4925 instances featuring 6 channel dimensions and 36 time steps. This dataset contains 14 classes.

CharacterTrajectories This dataset contains 2858 instances, featuring 3 channel dimensions, 182 time steps and 20 classes [11].

A.4. Model implementations, architectures and hyperparameters

All models are implemented in Pytorch [36], whereas the GP adapter and GP-PoM are implemented using the GPyTorch framework [17]. Next, we specify the details of the model architectures.

Sig We use a simple signature model that involves one signature block comprising of a linear augmentation followed by the signature transform. Subsequently, a final module of dense layers $(30, 30)$ is used. This is architecture refers to the Neural-signature-augment model [3].

RNNSig This model extends the signature transform to a window-based stream of signatures, where the final neural module is a GRU sliding over the stream of signatures. We allowed window sizes between 3 and 10 steps. For the GRU cell, we allowed any of the following number of hidden units: $[16, 32, 64, 128]$.

RNN Here, we use a standard RNN model using GRU cells. The size of hidden units was chosen as one of the following: $[16, 32, 64, 128, 256, 512]$.

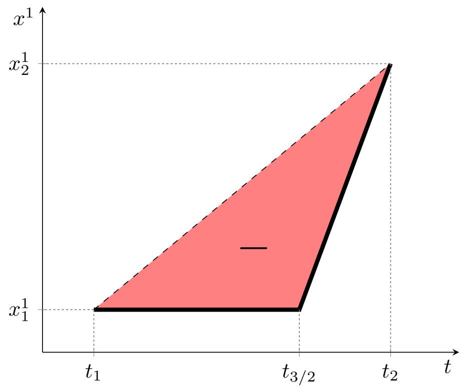


Figure 7: Lévy area of the forward-fill imputed path. By changing $t_{3/2}$ (a *single* unrelated observation!), we can make this disparity greater or smaller.

DeepSig For the deep signature model we employ two signature blocks (each comprising a linear augmentation and the signature calculation) following Bonnier et al. [3].

Hyperparameters

For all signature-based models, we allowed a signature truncation depth of 2–4, as we observed that larger values quickly led to a parameter explosion. All models were optimised using Adam [24]. Both the learning rates and weight decay were drawn log-uniformly between 10^{-4} and 10^{-2}. We allowed for the following batch-sizes: $(32, 64, 128, 256)$. For GP-based models, to save memory, we used virtual batching based on a batch-size of 32. Furthermore, for standard GP adapters we used 10 MC samples, conforming with recent literature [16, 33]. All approaches were constrained to have no more than 1.5 million trainable parameters.

A.5. Fragile dependence on sampling in unrelated channels: example

Suppose that we have observed the (very short) time series

$$\mathbf{x} = ((t_1, x_1^1, x_1^2), (t_2, x_2^1, *)) \in \mathcal{S}(\mathbb{R}^2). \tag{11}$$

Perhaps we now apply, say, forward fill data-imputation, to produce

$$((t_1, x_1^1, x_1^2), (t_2, x_2^1, x_1^2)).$$

Finally we linearly path-impute to create the linear path

$$\begin{aligned} f &\colon [t_1, t_2] \to \mathbb{R} \times \mathbb{R}^2 \\ f &\colon t \mapsto \left(t, x_1^1 \frac{t_2 - t}{t_2 - t_1} + x_2^1 \frac{t - t_1}{t_2 - t_1}, x_1^2\right), \end{aligned}$$

to which we may then apply the signature transform. In particular we will have computed the Lévy area with respect to t and x^1. As this is just a straight line, the Lévy area is zero.

Now suppose we include an additional observation at some time $t_{3/2} \in (t_1, t_2)$, so that our data is instead

$$\mathbf{x} = ((t_1, x_1^1, x_1^2), (t_{3/2}, *, x_{3/2}^2), (t_2, x_2^1, *)). \tag{12}$$

Then the same procedure as before will produce the data

$$\mathbf{x} = ((t_1, x_1^1, x_1^2), (t_{3/2}, x_1^1, x_{3/2}^2), (t_2, x_2^1, x_{3/2}^2)),$$

with corresponding function f. The (t, x^1) components of f and its (t, x^1)-Lévy area are shown in Figure 7. As a result of an unrelated observation in the x^2 channel, the (t, x^1)-Lévy area has been changed. The closer $t_{3/2}$ is to t_2, the greater the disparity. This simple example underscores the danger of 'just forward-fill data-imputing'. Doing so has introduced an undesired dependency on the simple *presence* of an observation in other channels, with the change in our imputed path being determined by the *time* at which this other observation occurred.

Indeed, *any* imputation scheme that predicts something other than the unique value lying on the dashed line in Figure 7, will fail. This means that this example holds for essentially every data-imputation scheme—the only scheme that survives this flaw is the linear data-imputation scheme. This is the unique imputation scheme that coincides with the linear path-imputation that *must* be our concluding step. However, when there is missing data at the start or the end of a partially observed times series, then there is no 'next observation' which linear imputation may use. So in general, we cannot uniformly apply the linear data-imputation scheme, and must choose another scheme or find ad-hoc solutions for missing data at the start or the end of the time series. Furthermore, it is plausible to assume that linear interpolation suffers from low expressivity as an imputation scheme which might empirically mask this benefit.

A.6. Causal signature imputation

In Section A.5 we have spoken about the limitations of traditional data-imputation schemes, and at first glance one may be forgiven for thinking that these are issues are unavoidable. However, it turns out that we need not be limited just to these traditional imputation schemes. The trick is to consider time not as a *parameterisation*, but as a *channel*[8]. This leads to a 'meta imputation strategy', which we refer to as *causal signature imputation*. It will turn any traditional causal data-imputation strategy (for example, feed-forward) into a causal path-imputation strategy for signatures; at the same time it will overcome the issue of a fragile dependence.

Suppose we have $\mathbf{x} \in \mathcal{S}(\mathcal{X}^*)$, and some favourite choice of causal data-imputation strategy $c\colon \mathcal{S}(\mathcal{X}^*) \to \mathcal{S}(\mathcal{X})$. Next, given

$$\mathbf{x} = ((t_1, x_1), \ldots, (t_n, x_n)) \in \mathcal{S}(\mathcal{X}), \tag{13}$$

we define the operation $\Omega\colon \mathcal{S}(\mathcal{X}) \to \mathcal{S}(\mathcal{X})$ by

$$\begin{aligned}\Omega(\mathbf{x}) = (&(t_1, x_1), (t_2, x_1), (t_2, x_2), (t_3, x_2),\\ &\ldots,\\ &(t_i, x_i), (t_{i+1}, x_i), (t_{i+1}, x_{i+1}), (t_{i+2}, x_{i+1}),\\ &\ldots,\\ &(t_{n-1}, x_{n-1}), (t_n, x_{n-1}), (t_n, x_n)).\end{aligned} \tag{14}$$

That is, *first* time is updated, and *then* the corresponding observation in data space is updated. This means that the change in data space occurs instantaneously.

For each $n \in \mathbb{N}$ (and given $a < b$), fix any $s_i^{(n)}$ for $i \in \{1, \ldots, n\}$. (We will see that the exact choice is unimportant in a moment.) Given

$$\mathbf{x} = ((t_1, x_1), \ldots, (t_n, x_n)) \in \mathcal{S}(\mathcal{X}),$$

let $\psi\colon \mathcal{S}(\mathcal{X}) \to (\mathbb{R} \times \mathcal{X})^{[a,b]}$ be the unique continuous piecewise linear path such that $\psi(s_i^{(n)}) = (t_i, x_i)$. Note that this is just a slight generalisation of the linear path-imputation that has already been performed so far; we are simply no longer asking for additional assumptions of the form $s_i^{(n)} = t_i$.[9]

[8] To be clear, using time as a channel is already a well-known trick in the signature literature that we do not take credit for inventing! See for example Bonnier et al. [3, Definition A.3]. It is however pleasing that something commonly used in the theory of signatures is also what allows us to overcome what we identify as some of their limitations.

[9] As in the $'_\theta$ of [44], for example.

Finally, we put this all together, and define the causal signature imputation strategy ϕ_c associated with c to be

$$\phi_c = \psi \circ \Omega \circ c,$$

which will be a map $\mathcal{S}(\mathcal{X}^*) \to (\mathbb{R} \times \mathcal{X})^{[a,b]}$. Thus ϕ_c defines a family of path-imputation schemes, parameterised by a choice of data-imputation scheme.

Before we analyse *why* this works in practice, we repeat a crucial property of the signature transform [3, Appendix A].

Theorem 1 (Invariance to reparameterisation)**.** *Let $f\colon [a,b] \to \mathbb{R}^d$ be a continuous piecewise differentiable path. Let $\psi\colon [a,b] \to [c,d]$ be continuously differentiable, increasing, and surjective. Then $\mathrm{Sig}^N(f) = \mathrm{Sig}^N(f \circ \psi)$.*

Coming back to our analysis, we first note that the previous theorem implies that the signature transform of $\phi_c(\mathbf{x})$ is invariant to the choice of $s_i^{(n)}$. Second, note that holding time between observations fixed is a valid choice, by the definition for $\mathcal{S}$ in equation (2). There should hopefully be no moral objection to our definition of $\mathcal{S}$, as holding time fixed essentially just corresponds to a jump discontinuity; not such a strange thing to have occur. Here, by replacing time as the parameterisation, we are then able to recover the continuity of the path. Third, we claim that ϕ_c is immune to the two major flaws of imputation methods, namely (i) their fragile dependence on sampling in unrelated channels, and (ii) their non-causality. Let us consider the first flaw of dependence on sampling in unrelated channels. For simplicity, take c to be the forward-fill data-imputation strategy. Consider again the $\mathbf{x}$ defined in expression (11). This means that

$$\phi_c(\mathbf{x}) = \psi(\ ((t_1, x_1^1, x_1^2), (t_2, x_1^1, x_1^2), (t_2, x_2^1, x_1^2))\ ). \tag{15}$$

Contrast adding in the extra observation at $t_{3/2}$ as in equation (12). Then

$$\begin{aligned}&\phi_c(\mathbf{x})(s)\\ &= \psi(\ ((t_1, x_1^1, x_1^2), (t_{3/2}, x_1^1, x_1^2), (t_{3/2}, x_1^1, x_{3/2}^2),\\ &\qquad (t_2, x_1^1, x_{3/2}^2), (t_2, x_2^1, x_{3/2}^2))\ ).\end{aligned} \tag{16}$$

Evaluating each ψ will then in each case give a path with three channels, corresponding to t, x^1, x^2. Then it is clear that the (t, x^1) component of the path in equation (15) is just a reparameterisation of the path in equation (16), a difference which is irrelevant by Theorem 1. (And the x^2 component of the second path has been updated to use the new information $x_{3/2}^2$.) Thus the causal path impuation scheme is robust to such issues. For general time series and c taken to be any other causal data-imputation strategy, then much the same analysis can be easily be performed.

Now consider the second potential flaw, of non-causality. The issue previously arose because of the non-causality of the linear path-imputation. We see from equation (14), however, such changes only occur in data space while the time channel is frozen; conversely the time channel only updates with the value in the data space frozen. Provided that c is also causal, then causality will, overall, have been preserved. For example, it is possible to use this scheme in an online setting. There are interesting comparisons to be made between causal signature imputation and certain operations in the signature literature. First is the *lead-lag* transform [8]. With the lead-lag transform, the entire path is *duplicated*, and then each side is alternately updated. Conversely, in causal signature imputation, the path is instead *split* between t and $(x^1, \ldots, x^n)$, and then each side is alternately updated. Second is the comparison to the linear and rectilinear embedding strategies, see for example [12]. It is possible to interpret $\psi \circ \Phi$ as a hybrid between the linear and rectilinear embeddings: it is rectilinear with respect to an ordering of t and $(x^1, \ldots, x^n)$, and linear on $(x^1, \ldots, x^n)$. Furthermore, the time-joined transformation [27] is pursuing a very similar goal to the here described causal signature imputation. This is also why we do not consider this imputation strategy as a novel contribution of this work.

Comparison to the Fourier and wavelet transforms

The signature transform exhibits a certain similarity to the one-dimensional Fourier or wavelet transforms. Both are integrals of paths. However, in reality these transforms are fundamentally different. Both the Fourier and wavelet transforms are linear transforms, and operate on each channel of the input path separately. In doing so they model the path as a linear combination of elements from some basis.

Conversely, the signature transform is a nonlinear transform - indeed, it is a universal nonlinearity - and operates by combining information between different channels of the input path. In doing, the signature transform models *functions of the path*; the universal nonlinearity property says that in some sense it provides a basis for such functions.