RoughPy: Streaming data is rarely smooth

Sam Morley (University of Oxford)

DataSig

A rough path between mathematics and data science

Rough path theory and non-commutative algebra

Rough path theory arose from the study of controlled differential equations, which aim to study the effect of a non-linear system relative to some driving path. In this theory, one particular controlled differential equation plays a key role. For a path X taking values in $\mathbb{R}^d$, consider the equation

$$\mathrm{d}S_t = S_t \otimes \mathrm{d}X_t \qquad S_0 = 1.$$

The solution to this equation is called the *signature* of X. It is a high order description of the evolution of X and is one of the main tools of rough path theory.

When the path X is sufficiently smooth, the signature can be computed directly as a sequence $(1, S^1, S^2, \ldots)$ where each S^n is given by an iterated integral

$$S^n = \int_{0<u_1<u_2<\cdots<u_n<t} \mathrm{d}X_{u_1} \otimes \mathrm{d}X_{u_2} \otimes \cdots \otimes \mathrm{d}X_{u_n}.$$

However, not all paths are sufficiently smooth. In those cases we need a rough path, where higher order information is provided along with the values of the path.

Free tensor algebra

The signature is an element of the free tensor algebra $\mathrm{T}((\mathbb{R}^d))$. This is the space of all sequences $(x_0, x_1, \ldots)$ where x_0 is a scalar and each x_j belongs to the space of jd-tensors $(\mathbb{R}^d)^{\otimes j}$ for $j > 0$. This space has a multiplication given by the tensor product, making it a non-commutative algebra.

Shuffle algebra

Dual to the free tensor algebra is the shuffle algebra $\mathrm{Sh}(\mathbb{R}^d)$. This consists of finite sequences from the tensor algebra equipped with the shuffle product. This is a commutative algebra and, via their action on signatures, elements of $\mathrm{Sh}(\mathbb{R}^d)$ represent continuous functions defined on paths.

Free Lie algebra

The free tensor algebra also contains the Lie algebra $\mathrm{L}(\mathbb{R}^d)$. First define the Lie bracket operator on $\mathrm{T}((\mathbb{R}^d))$ by

$$[\boldsymbol{x}, \boldsymbol{y}] = \boldsymbol{x} \otimes \boldsymbol{y} - \boldsymbol{y} \otimes \boldsymbol{x} \qquad (\boldsymbol{x}, \boldsymbol{y} \in \mathrm{T}((\mathbb{R}^{\mathrm{d}}))).$$

Then we define recursively subspaces $L_0 = \{\mathbf{0}\}$, $\mathrm{L}_1 = \mathbb{R}^d$ and for each $m \geq 1$

$$L_{m+1} = \mathrm{span}\{[\boldsymbol{x}, \boldsymbol{y}] : \boldsymbol{x} \in \mathbb{R}^{\mathrm{d}}, \ \boldsymbol{y} \in L_m\}.$$

The Lie algebra contains those sequences $(l_0, l_1, \ldots)$ where $l_j \in L_j$ for each j. If $\mathrm{Sig}(X)$ is the signature of a path X, then there is an element of the Lie algebra $\mathrm{LogSig}(X)$ that encodes the same information about X as $\mathrm{Sig}(X)$. This log-signature is obtained by taking the tensor logarithm of $\mathrm{Sig}(X)$. The log-signature does not enjoy the same approximation of continuous functions.

Truncated tensor and Lie algebras

The tensor and Lie algebras contain infinite sequences, so for practical computations we truncate at a finite level, usually called the *depth*. The tensor algebra truncated at depth n is denoted $\mathrm{T}^n(\mathbb{R}^d)$ and has dimension equal to

$$\sum_{i=0}^{n} d^i = \frac{d^{n+1}-1}{d-1}$$

when $d > 1$. The truncated exponential of $\mathbf{x} \in \mathrm{T}^n(\mathbb{R}^d)$ with scalar term 0 can be computed using the truncated power series

$$\exp_n(\boldsymbol{x}) = \sum_{j=0}^{n} \frac{\boldsymbol{x}^{\otimes j}}{j!}$$

The truncated Lie algebra is defined in a similar way, and has a smaller dimension than the tensor algebra.

Streaming data is everywhere and is rarely smooth on normal scales

Text, vital signs, lab results, stocks, trades, electricity, radio signals, light intensity, a person signing, audio and speech, ...

RoughPy Library

Algebra

RoughPy is not just a library for streaming data, it is also a library for abstract algebra. RoughPy provides classes for elements of the free tensor algebra, shuffle algebra, and Lie algebra. This means interacting with RoughPy is much closer to the mathematics than with other libraries.

Intervals

RoughPy provides a rich system of tools for working with intervals. Intervals are used to query streams and to understand the results of such queries. The intervals module includes tools such as dyadic segmentation and dissection routines.

Differentiation

One of the design goals of RoughPy is to support fully differentiable operations on streams and algebras. This will allow RoughPy objects to appear within machine learning pipelines and fully support back propagation and training.

Streams

RoughPy is all about changing the way we think about streaming data. The stream class is an abstraction of streaming data, viewed through the lens of rough path theory. A stream is an object that can be queried over an interval to obtain a (log-)signature. The stream object also enables perform intelligent caching of intermediate results, which greatly reduces the number of computations that are necessary for a query.

A very simple constructor for a stream takes a sequence of Lie increments, which represent the "jumps" in Lie space in the value of the path. This can be the changes in values that occur, but it can also include higher order changes in the process. RoughPy provides several other kinds of stream constructors, including piecewise Abelian streams and Brownian motion streams, which provides an approximation of Brownian motion.

Data

RoughPy can consume data from a variety of sources. This includes loading directly from audio files and from data in array form. Our plan is to expand this to include other sources such as querying a database for data and loading direct from online sources such as IOT devices via MQTT.

Interoperability

RoughPy is designed to be interoperable with array implementations such as NumPy, PyTorch, Tensorflow, etc. RoughPy understands the DLPack interchange format for communicating arrays without copying, where possible. This means RoughPy objects can be passed back and forth seamlessly as part of a machine learning pipeline.

Computation

RoughPy provides a flexible mechanism for performing computations based on kernels. This means we can provide high-performance, efficient algorithms for computing signatures with floating point scalars but also support infinite precision rational arithmetic, polynomial scalars, and (in the future) user defined scalars.

https://roughpy.org
https://github.com/datasig-ac-uk/RoughPy

This work was partially funded by EPSRC under Grant EP/S026347/1, and by The Alan Turing Institute EPSRC Data Centric Engineering Programme Grant EP/N510129/1. This work was partially supported by The Alan Turing Institute's Defense and Security Programme, funded by the UK Government.

https://datasig.ac.uk

Data Science

The Signature Transform

A simple use of the signature is as a feature map that can be used in conjunction with other techniques. The signature captures the order of events and compresses the fine data into a single vector of fixed size; this size needs to be chosen per application. The high dimensionality of the signature is a representation of the internal complexity of the signal. The log-signature can be used instead for a more compact representation, but this sacrifices the linear approximation of functions. The signature transform has been used in a variety of applications including tracking mood via online posts,[1] predicting battery cell degradation,[2] and predicting sepsis based on intensive care data.[3]

[1] Tseriotou, T., Tsakalidis, A., Foster, P., Lyons, T. and Liakata, M., 2023, July. Sequential path signature networks for personalised longitudinal language modeling. In *Findings of the Association for Computational Linguistics*: ACL 2023 (pp. 5016-5031).

[2] Ibraheem, R., Wu, Y., Lyons, T. and Dos Reis, G., 2023. Early prediction of Lithium-ion cell degradation trajectories using signatures of voltage curves up to 4-minute sub-sampling rates. *Applied Energy*, 352, p.121974.

[3] Morrill, J.H., Kormilitzin, A., Nevado-Holgado, A.J., Swaminathan, S., Howison, S.D. and Lyons, T.J., 2020. 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(10), pp.e976-e981.

The Signature Kernel

The signature kernel is a kernel induced on paths by an inner product defined on signatures. Informally, the signature kernel is a measure of the similarity between two streams. It can be used in a number of "kernel methods" such as support vector machines. The signature kernel can be approximated using truncated signatures, but it can also be computed directly as the (numerical) solution of a partial differential equation of Goursat type.[4] In fact, Lemercier has considerably generalised this to make use of the high order data using a system of partial differential equations.[5] This breakthrough was made using RoughPy.

An important application of the signature kernel is to determine whether a stream is outlying compared to a corpus of streams. DataSig has developed a robust framework for this task that better respects the structure of the streams and has fewer hyperparameters than other techniques.[6] Notably, distance and coordinate chart are not hyperparameters. Recently, this framework has been applied in the domain of radio astronomy. The task is to detect radio frequency interference in radio astronomy data.[7]

[4] Salvi, C., Cass, T., Foster, J., Lyons, T. and Yang, W., 2021. The Signature Kernel is the solution of a Goursat PDE. *SIAM Journal on Mathematics of Data Science*, 3(3), pp.873-899.

[5] Lemercier, M. and Lyons, T., 2024. A high order solver for signature kernels. arXiv preprint arXiv:2404.02926.

[6] Shao, Z., Chan, R.S.Y., Cochrane, T., Foster, P. and Lyons, T., 2020. Dimensionless anomaly detection on multivariate streams with variance norm and path signature. arXiv preprint arXiv:2006.03487.

[7] Arrubarrena, P., Lemercier, M., Nikolic, B., Lyons, T. and Cass, T., 2024. Novelty Detection on Radio Astronomy Data using Signatures. arXiv preprint arXiv:2402.14892.

Neural Controlled Differential Equations

Neural controlled differential equations (NCDEs) are of the form

$$\mathrm{d}Y_t = f_\theta(Y_t)\,\mathrm{d}X_t$$

where f_θ is a neural network and X_t is a data stream. We treat Y_t as hidden state, which results in trainable systems that are a continuous time analogue of recurrent neural networks.[8] The resulting model is robust to irregular sampling and missingness. It has been through several iterations, but older iterations are no longer competitive with current state-of-the-art state space models like S5.

The most recent iteration of NCDEs, termed Log-NCDEs, make use of the Log-ODE solving method from the theory of controlled differential equations. Early results show that this method is far more competitive with highly-tuned state space models.[9]

[8] Kidger, P., Morrill, J., Foster, J. and Lyons, T., 2020. Neural controlled differential equations for irregular time series. *Advances in Neural Information Processing Systems*, 33, pp.6696-6707.

[9] Walker, B., McLeod, A.D., Qin, T., Cheng, Y., Li, H. and Lyons, T., 2024. Log Neural Controlled Differential Equations: The Lie Brackets Make a Difference. arXiv preprint arXiv:2402.18512.