A feature set for streams and an application to high-frequency financial tick data*

Terry Lyons
Oxford-Man Institute,
University of Oxford
Eagle House, Walton Well Rd
Oxford, UK
terry.lyons@oxford-man.ox.ac.uk

Hao Ni
Oxford-Man Institute,
University of Oxford
Eagle House, Walton Well Rd
Oxford, UK
Hao.Ni@maths.ox.ac.uk

Harald Oberhauser
Oxford-Man Institute,
University of Oxford
Eagle House, Walton Well Rd
Oxford, UK
harald.oberhauser@oxford-man.ox.ac.uk

ABSTRACT

We propose a set of features to study the effects of data streams on complex systems. This feature set is called the *the signature representation* of a stream. It has its origin in pure mathematics and relies on a relationship between non-commutative polynomials and paths. This representation had already signifcant impact on algebraic topology, control theory, numerics for PDEs, stochastic analysis and the theory of rough paths; more recently first steps have been taken to apply such methods to the study of big data streams. We show that the signature representation can provide an efficient summary of a stream and its effects. We then show that it can be combined with standard tools from machine learning. After introducing the signature for streams and some theoretical background, we apply this approach to a challenging real-world example: high-frequency financial data streams. In this context, the streams are tick-by-tick market data of a stock traded at the New York stock exchange NYSE and the effect of the stream is the profit and loss of complex investment strategies (i.e. a nonlinear functional of the stream). Our numerical results (applied to Thomson–Reuters tick data of several full trading days for IBM stocks) show that the signature of the price stream efficiently captures the necessary information to learn the return of an investment strategy. However, we emphasize that the underlying ideas are not limited to financial data streams and have the potential to be applied to many other areas in data mining where the non-commutative nature of streams is of importance, like text mining, bioinformatics or click history.

Categories and Subject Descriptors

H.2.8 [**Database applications**]: Data mining; G.3 [**Probability and Statistics**]: [time series analysis, robust regression, nonparametric statistics]; J.1 [**Computer Applications**]: Financial

General Terms

Summarizing streams and their effects, the signature representation, time series analysis, data mining, nonlinear functionals of data streams, mathematical finance

Keywords

Time series, data streams, signature, rough paths, finance

1. INTRODUCTION

The last decade has seen an enormous rise in the need for analysing and mining data streams. Motivated by new applications, algorithms have been developed which for example allow to find the most frequent elements, obtain representative samples, estimate moments, etc.; see for example [16, 7].

A problem that is related but not directly addressed by this recent progress is that very often we are not only interested in streams per se (like counting the most frequent values it takes, etc.) but instead, we want to understand the effects of a stream on a complex system. On such an abstract level, examples include advertisements (a stream of characters, sounds or images) that affects the decision of a person to buy a product, the sequence of nucletoides that appear in DNA strings and affects their biological function (e.g. the folding of protein encoded in the DNA string) or the sequence of daily or even intra-day financial data that affects the profit and loss of an investor resp. her trading strategy, etc. In all these examples it is clear that the information in the stream that we have to extract to understand its effects has a *non-commutative structure* (e.g. a drop in price followed by an increase in price does in general not have the same effect on the outcome of an investment strategy compared to this sequence of events happening in the reversed order; the order in which a customer on a video streaming site like Netflix has watched movies is important for predicting his next movie choice; etc.). It is not easy to provide features for a stream that reliably *capture such "non-commutative order information" in a quantitative way* but this is exactly the goal of this paper.

To be precise, the contribution of this paper is to

*DOI:http://dx.doi.org/10.1145/2640087.2644157

BigDataScience '14 Beijing, China

- introduce ideas from pure mathematics about the representation of paths in the context of data mining. These ideas had already huge impact in mathematics (algebraic topology, control theory, stochastic analysis, probability and rough path theory). In mathematics it is known as the so-called *signature* representation, and we show that this presentation of a path/stream[1] can be turned into an *effective feature set for describing sequentially ordered data, i.e. streams*,

- show that this representation/feature set becomes a powerful tool to *approximate nonlinear functionals of multi-dimensional streams*; in fact, the situation is analogous to the approximation of a continuous function by polynomials as guaranteed by the Stone–Weierstrass theorem,

- show that the signature representation of a stream can be combined with tools from machine learning — a topic that has recently attracted attention of several researchers [9, 11],

- apply these abstract concepts to a challenging real-world data set: the *electronic order book* of a modern stock exchange. Here new data arrives with high-frequency[2] and traders resp. algorithms buy and sell the underlying asset based on the past of the data stream.

The feature set we propose is to associate with a stream a sequence of tensors that are built from the iterated integrals of the path over a simplex; this is the so-called *signature* representation. The signature summarizes the stream and its effects on a large class of systems (i.e. functionals of streams) to arbitrary precision. Moreover, it has a rich algebraic structure and can be thought of as a non-commutative polynomial that describes the underlying stream [3, 2, 10, 17].

In Section 2 we briefly survey the mathematical background. In Section 3 we show that such a representation allows to *turn nonlinear relationships on the stream level into linear relationships on the signature level*. In Section 4 we apply these new methods to a challenging real-world data set: a complex functional of the market data stream is the return of an investment strategy of a high-frequency trader. We use above methods to show that already simple tools like linear regression applied to above features can learn such complex functionals.

2. FEATURES FOR STREAMS

On an abstract level, a stream x is a map from a totally ordered set I (like the integers or the positive reals) into some state space, in other words a path defined on some domain and taking values in some state space E, i.e.

$$x : I \to E. \tag{1}$$

To develop ideas and to keep things concrete, let us briefly specialise on systems that can be modelled by differential equations controlled by paths i.e. the stream is a real-valued path evolving in continuous time $x : [0,T] \to R^d$ and the evolution of the complex system it affects is modelled by $y : [0,T] \to R^e$ via a differential equation of the form

$$dy_t = Ay_t dx_t. \tag{2}$$

Here, $Ay_t dx_t$ stands for $A(dx_t)(y_t)$ and $A : R^d \to L(R^e, R^e)$ is a map into the space of linear maps $L(R^e, R^e)$. Such differential equations driven by the path x are called linear and a formal (we neglect convergence questions!) iteration of (2) ("Picard iteration") then leads to the representation

$$\begin{aligned} dy_t &= y_0 + Ay_0 \int_0^t dx_{t_1} + A^2 y_0 \int_{0<t_1<t_2<t} dx_{t_1} \otimes dx_{t_2} + \dots \\ &= \sum_{k=0}^{\infty} A^k y_0 \int_{0<t_1<\dots<t_k<t} dx_{t_1} \otimes \dots \otimes dx_{t_k}. \end{aligned} \tag{3}$$

In above series we used the standard notation $\otimes$ to denote tensor products, e.g. $\int_{0<t_1<t_2<t} dx_{t_1} \otimes dx_{t_2}$ is a 2-tensor that can be spelled out in components as the Riemann–Stieltjes integrals

$$\left(\int_{0<t_1<t_2<t} dx^i_{t_1} dx^j_{t_2} \right)_{i,j=1,\dots,d}.$$

A few remarks: firstly, the expression (3) shows that the response y_t is described by two clearly separated quantities: one that describes the stream x over the time interval $[0,t]$, namely as an element in the space of graded tensors $\bigoplus_{k\geq 1}(R^d)^{\otimes k}$,

$$\left(\int_{0<t_1<t} dx_t, \dots, \int_{0<t_1<\dots<t_n<t} dx_{t_1} \otimes \dots \otimes dx_{t_n}, \dots \right), \tag{4}$$

and another that describes the intrinsic properties of the system (2) that determine the response to any input path, namely the the sequence $(A^n)_{n\geq 0}$. Secondly, above derivation can be easily extended to nonnlinear vector fields instead of linear vector fields A (using the usual identification of vector fields as directional derivatives). Finally, since differential equations of the type (2) (resp. with a nonlinear vector field) model a large class of complex systems [1], we can hope that a "reasonably large class" of nonlinear functionals of streams can be approximated by linear combinations of such iterated integrals of the stream.

2.1 The signature of a path (stream)

It turns out that the sequence of iterated integrals has a rich algebraic structure and for algebraic reasons (multiplication and inversion) it is useful to include an additional coordinate which we define to be $1 \in R$.

Definition 1. The signature $S(x)_{s,t}$ of a bounded variation [3] path $x : [0,T] \to R^d$ over some domain $[s,t] \subset [0,T]$ is the sequence of tensors[4]

$$\begin{aligned} &\bigoplus_{k\geq 0} (R^d)^{\otimes k} \ni S(x)_{s,t} := \\ &\left(1, \int_{s<t_1<t} dx_t, \dots, \int_{s<t_1<\dots<t_n<t} dx_{t_1} \otimes \dots \otimes dx_{t_k}, \dots \right) \end{aligned}$$

[1] We do not make a technical distinction between paths and streams; however we use stream when we want to emphasize more the data arrives in discrete time.

[2] The arms race is currently fought at orders of micro-seconds

[3] Recall that the path x is said to be of bounded variation if $\sup \sum_i |x_{t_{i+1}} - x_{t_i}| < \infty$ where the supremum is taken over all partitions $s \leq t_1 \leq \dots < t_n \leq t$ of $[s,t]$

[4] We use the convention $(R^d)^0 \equiv 1$

We have already encountered a first motivation to consider the signature as feature set by looking at the differential equation (2). Let us now give another motivation that is due to deep results going back to work of Chen [3, 2] and subsequent refined quantitative estimates of Hambly–Lyons [10]. It shows that there exists "nearly" a one-to-one relation between a path $x : [s,t] \to E$ and its signature, i.e. the sequence of tensors $S(x)_{s,t}$. Before we make this precise, recall that the graded space of tensors $\bigoplus_{k\geq 0}(R^d)^{\otimes k}$ forms an algebra and has rules of calculus similar to power series: scalar multiplication and addition is clear and element multiplication and inversion follow in analogy to power series, i.e. for $a, b \in \bigoplus_{k\geq 0}(R^d)^{\otimes k}$, $c \equiv (1, c_1, c_2, \ldots) = a + b$ is given by $c_k = a_k + b_k$; $c \equiv (1, c_1, c_2, \ldots) = a \otimes b$ is given by $c_k = \sum_{i=0}^{k} a_i b_{k-i}$ and the inverse $a^{-1} = \sum_{n\geq 0}(1-a)^{\otimes n}$. Chen understood that geometric operations on paths like concatentation or running the path backwards, have an algebraic interpretation in terms of their signature.

THEOREM 1 (CHEN [3]). *Let $x : [s,t] \to R^d$ and $y : [t,u] \to R^d$ be paths of bounded variation. If we denote with $x \star y$ their concatenation, then*

$$S(x \star y)_{s,u} = S(x)_{s,t} \otimes S(y)_{t,u} \tag{5}$$

Similarly to concatenation, time-reversal corresponds to inversion of the signature.

THEOREM 2. *Let $x : [s,t] \to R^d$ be a path of bounded variation. Define $\overleftarrow{x} : [s,t] \to R^d$ as $\overleftarrow{x}_r = x_{t+s-r}$*

$$S(x)_{s,t} = S(\overleftarrow{x})^{-1}_{s,t} \tag{6}$$

2.2 Representing paths as signatures

Another motivation besides the expansion (3) for taking the signature (i.e. the sequence of iterated integrals) as a collection of features to represent an element of (the infinite-dimensional) path-space is that there exists *"nearly" a one-to-one correspondence between a path $x : [s,t] \to E$ and its signature $S(x)_{s,t}$*; indeed, we cannot expect a true one-to-one correspondence between x and $S(x)_{s,t}$ since for example $S(x)_{[s,t]}$ is invariant under time-reparametrisation [5] of the domain $[s,t]$; similar "tree-like" paths (see below for the definition) are not distinguishable from their signature. However, modulo these two classes the correspondence is one-to-one.

Definition 2. A path $x : [s,t] \to R^d$ is tree-like if there exists a continuous function $h : [s,t] \to [0,\infty)$ such that

$$|x_t - x_s| \leq h_t + h_s - 2 \inf_{u\in[s,t]} h_u \tag{7}$$

Let y be another path of bounded variation. We define an equivalence relation on the set of bounded variation paths by saying that $x \sim y$ if $S(x)_{s,t} = S(y)_{s,t}$.

We can now state the precise characterisation of the set paths for which the signature representation is unique.

THEOREM 3 (CHEN,HAMBLY–LYONS [3, 10]). *Let $x, y : [s,t] \to R^d$ be paths of bounded variation. Then*

- *x is tree-like if and only if $S(x) = 1$.*
- *$x \sim y$ if and only if $x \star y^{-1}$ is tree-like.*
- *Among all paths with the same signature as x there exists a path of minimal length that is unique (up to time-reparametrisation).*

Two remarks. Firstly, the fact that the signature is invariant under time-reparametrisation is usually an advantage in the treatment of streams since it leads to a dimensionality reduction (if time-parametrisation actually matters one can simply add a 0th "time-coordinate" to the stream). Secondly, above shows that tree-like paths can be thought of as "null-sets"; most "real-world" paths do not have such tree-like structure (and certainly not the financial streams of Section 4).

2.3 Discrete time, truncated signatures, Lead-lag time series

To apply above tools to real-world data several issues arise.

One is of course that we typically do not observe a path/stream x in continuous time but only have discrete observations of x at a finite number of times $D = (t_i)_{i=1,\ldots,n}$ available, i.e. the time-series $(x_{t_i})_{i=1,\ldots,n}$. A natural way to use the signature in this context is to work with piecewise linear interpolations, that is we define a new path $x^D : [t_1, t_n] \to R^d$,

$$x^D_t := x_{t_i} + \frac{t - t_i}{t_{i+1} - t_i}(x_{t_{i+1}} - x_{t_i}) \text{ for } t \in [t_i, t_{i+1}]$$

and then work with $S(x^D)$. As we will see in Section 4 storing a few elements the signature is typically *(much) more efficient* compared to storing many of the samples (x_{t_i}).

Another issue is that the signature has a recursive nature: the $(k+1)$th iterated integral is given by integrating the path x against the kth iterated integral which suggests that the state space $\bigoplus_{k\geq 0}(R^d)^{\otimes k}$ is too big. This is indeed the case and can show that one can work with strict subset of $\bigoplus_{k\geq 0}(R^d)^{\otimes k}$ that altough not a linear space, still has (Lie-)group structure[6]. Moreover, in practice it is of course not possible to store the infinite series of tensors and we work with the projection of $S(x)$ to $\bigoplus_{k=0}^{N}(R^d)^{\otimes k}$ where N is a natural number (this is comparable to the Laplace transform where the number of coefficients needed for an accurate description depends on the underlying function).

Further it turns out that while the signature $S(x^D)$ provides (nearly) a one-to-one (see Theorem 3) representation of paths and thus allows to recover the lagged values, it is advantageous for many applications to include the lag-1 time series as another coordinate, i.e. to work with $\tilde{x}_{t_i} = (x_{t_i}, x_{t_{i-1}})$ and calculate the signature $S(\tilde{x}^D)$. The reason is that especially for noisy streams several quantities are more efficiently expressed as iterated integrals of x against its lagged value[7].

3. NONLINEAR FUNCTIONALS AS LINEAR COMBINATIONS OF THE SIGNATURE

Classic regression methods allow to infer a functional relationship between an observed quantity (output of a complex

[5] If $\phi : [s,t] \to [s,t]$ is a strictly increasing function then $S(x)_{s,t} = S(x \circ \phi)_{s,t}$

[6] We do not elaborate on this but refer to [13]

[7] For example, if the underlying process is a semi-martingale on can show that this captures the quadratic variation which is an important quantity for financial applications; see [6]

system) and explanatory variables (input to the system). An important special case in engineering and economics is the case when the explantory variables are streams.

Assume there exists a functional f that takes an input path $x : [s,t] \to R^d$ to a (for simplicity) scalar output y and that we are subject to noisy observations, i.e.

$$y = f(x) + \epsilon. \tag{8}$$

In general it is a hard task to infer truly nonlinear relations between the stream x and the value y. For example, one of the simplest situations is given when y is proportional to the signed area surpassed by x, i.e. assume $d = 2, e = 1$ and

$$y_t - y_s = f(x) \sim \int_s^t x_r^1 dx_r^2 - \int_s^t x_r^2 dx_r^1. \tag{9}$$

However, motivated by the previous sections of representing a stream via its signature, we can restate the problem in signature space as of finding a function $\tilde{f}$ such that

$$y = \tilde{f}(S(x)) + \epsilon \tag{10}$$

Seen this way, in the above example (9) the nonlinear relationship in terms of the path x reduces to a simple linear relationship in terms of the signatures, i.e. if we denote $\boldsymbol{x} = (1, \boldsymbol{x}^1, \boldsymbol{x}^2, \ldots) = S(x)_{s,t}$, and the scalar outcome by y, then

$$y = c\boldsymbol{x}^{2;1,2} - c\boldsymbol{x}^{2;2,1}. \tag{11}$$

Recall that $\boldsymbol{x}^2$ is the set of iterated integrals

$$\left(\int_{0<t_1<t_2<t} dx_{t_1}^i dx_{t_2}^j \right)_{i,j=1,2} \in \left(R^2\right)^{\otimes 2}$$

and with $\boldsymbol{x}^{2;1,2}$ we denote the $(i,j) = (1,2)$ coordinate

$$\int_{0<t_1<t_2<t} dx_{t_1}^1 dx_{t_2}^2 \in R.$$

In other words, *nonlinear relationships between an observation and an explanatory stream are after transformation into signatures reduced to linear relationships.* The general procedure can be summarized as

- Given are N independent observations of (stream,scalar)-tuples
$$(x_i, y_i)_{i=1,\ldots,N}$$
,
- Calculate the (truncated) signatures of the steams (x_i) to transform this into (signature,scalar)-tuples
$$(\boldsymbol{x}_i, y_i)_{i=1,\ldots,N},$$
- Use simple *linear*(!) methods (e.g. regression) to approximate
$$y_i - \epsilon = f(\boldsymbol{x}_i) \sim c_0 + \sum_k \sum_{i_1,\ldots,i_k} c_{i_1,\ldots,i_k} \boldsymbol{x}^{k;i_1,\ldots,i_k}$$

Above approach is only of interest if we expect that a sufficiently rich class of functionals f of streams x can be approximated by linear combinations of iterated integrals. We have already seen that functionals that are the solutions of differential equations fall into this class, see equation (3). However, a much more general result holds: since the signatures form an algebra (Section 2.1) one can use a variation of the Stone–Weierstrass theorem to show that *every functional f that depends continuously on the underlying stream x can be uniformly approximated by such linear combinations of iterated integrals of x.*

4. FINANCIAL DATA STREAMS

Electronic limit order books are nowadays the prevalent mode of exchange on financial markets. Participants send their buy/sell orders via a continuous time auction system to the server of an exchange. This has led to one of the biggest revolutions in finance namely the rise of algorithmic trading at high frequencies (HFT) which is an arms at the hardware (latency) level as well as on the algorithmic/modelling front. In contrast to traditional low-frequency trading strategies where an investor holds a trading position for several weeks or months to generate return, the high-frequency paradigm is to execute a large number of trades per day/ hour/ minute/ second or even milli or micro-seconds where each trade only generates a tiny return on average but the large number of trades makes up for the small margin per trade. (Several research reports estimate that for example in 2009, 60-70% of the US equity trading volumne was due to HFT). The available literature of succesfull "real-world" HFT trading strategies is (not surprisingly) rather sparse but what is common knowledge is that many of the usual mathematical approaches that are used by quants in investment banks for low-frequency trading (e.g. the link between no-arbitrage and martingale theory) are much less relevant on a high-freqency/tick data scale.

One is hard-pressed to come up with a model from top-down economic arguments that provides a good fit and *non-parametric/data-driven models* are needed. The lack of statistical models that capture the whole order book makes high frequency financial data streams a challenging problem. We believe that the methods we have presented in the previous section can provide a robust way to study phenomena that appear in HFT data streams. In the context of low-frequency financial data they have been recently sucessfully applied for classification purpose (i.e. given a thirty minute sample decide whether it belongs to a time interval in the morning or afternoon; see [8].) Here we go further and show that the signature does not only provide features for classification purposes but provides an *efficient description* of the whole (financial data) stream that allows to learn a nonlinear functional of the stream (the investment strategy).

To keep things simple and due to space limitations we focus here on a basic but non-trivial question:

Can we learn the outcome of a (to us unknown) trading strategy over a given investment period based on previous performance of this trading strategy?

In other words the stream x is a sequence of prices, the effects of this stream are given by a functional f that transforms the stream of prices to the return $y = f(x)$ made by an investor who follows a certain trading strategy. Our task is to predict $f(x)$ for any given x, based on the observations $(x_i, f(x_i))_{i=1,\ldots,N}$, i.e. a supervised learning problem where the explanatory variables $(x_i)_i$ are elements in an infinite-dimensional space.

Our approach is to use the signature representation of the price stream and then apply the regression outlined in Section 3 to approximate the functional f.

4.1 The electronic limit order book

A typical (so-called Level 1) order book of an electronic exchange contains the following market data for every tick:

- *Bid and Ask prices*: the lowest resp. highest price another trader is willing to buy or sell,
- *Bid and Ask sizes*: the number of stocks available to buy or sell at corresponding Bid/Ask price,
- *Last price*: the last price at which the most recent trade was executed,
- *Last size*: the last size at which the most recent trade was executed.

In other words, we have a four dimensional stream of data arriving at very high frequencies (the time difference between ticks is typical on the scale of microseconds) and there are even more detailed data sets available (e.g. the so-called Level 2 order book data usually provides a break down of bid/ask offers into market participants, etc.). Electronic order book data is usually not freely available and our dataset comes from a commerical provider (we use Thomson–Reuters tick data via the subscription of the Oxford-Man Institute of Quantitative Finance; summaries of the data we use is publicly available [14]) that provides API access to 2 petabytes of microsecond, time-stamped historical electronic order book data; see Figure 1 for the price evolution over the whole day and Figure 2 for a high-frequency plot of 1000 subsequent trading times plus the traded volume.

Table 1: Three subsequent time ticks from the (reduced) electronic order book for IBM shares traded at the NYSE on 9th April 2014. The change of periods of slow trading to periods with lots of activity (price changes in microseconds) is typical.

Time	GMT offset	Type	Price	Volume
13:31:39.024993	-4	Trade	194.66	100
13:31:47.454290	-4	Trade	194.636	100
13:31:47.454314	-4	Trade	194.64	400
⋮	⋮	⋮	⋮	⋮

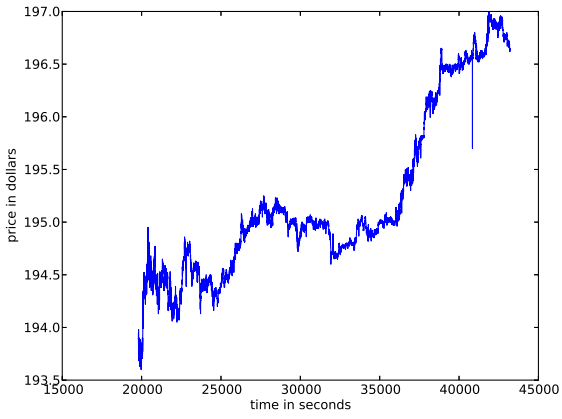

Figure 1: Exectuted trading prices of IBM share at the NYSE on the 9th April 2014

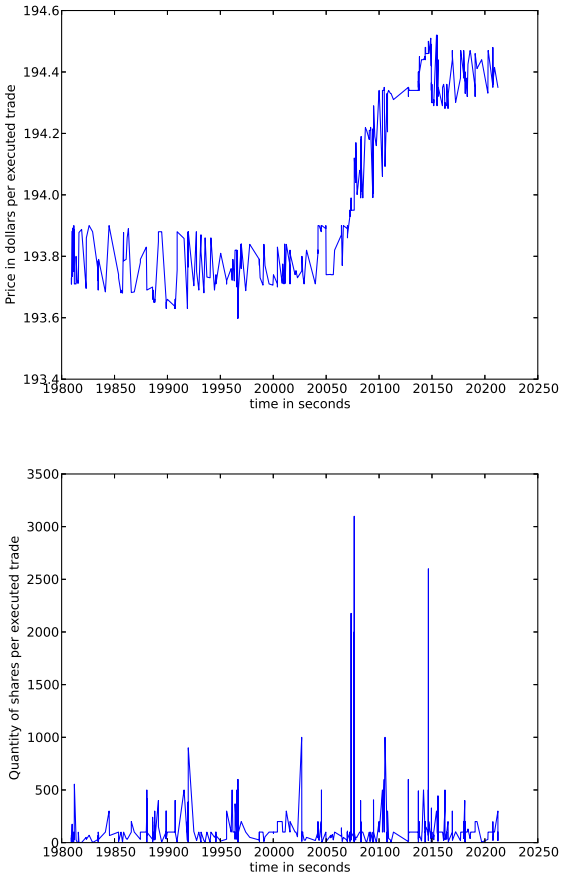

Figure 2: The first 1000 executed trading prices and the associated volumes of IBM shares at the NYSE on the 9th April 2014

4.2 Trading strategies as functionals of streams

To keep things simple for our demonstration we use the reduced electronic order book that provides us with sequential data $(x_{t_i})_i$ where each x_{t_i} is a vector consisting of (p_{t_i}, v_{t_i}) where p_i denotes the executed trading price and v_i denotes the volume (quantity of shares) of some risky asset. A trader is allowed to buy and sell the risky asset (e.g. IBM shares) or to invest in a riskless asset (e.g. a fixed rate cash bank account). We therefore describe a trading strategy over the time period $[0, t_n]$ by a stream of vectors $(h_{t_i})_{i=1,\dots,n}$ where each $h_{t_i} = (h^0_{t_i}, h^1_{t_i})$ and $h^0_{t_i}$ denotes the amount of units held by the investor between time t_{i-1} and t_i of the riskless asset and $h^1_{t_i}$ denotes the amount of units of the risky asset held by the investor between these times. The investor takes the investement decision at time t_{i-1}, hence each h_{t_i} is allowed to depend only on data from the past (an *adapted process*), that is h_{t_i} can be only a function of $\{x_{t_k} : k = 1, \dots, i-1\}$. Moreover, we are only interested in trading strategies that are *self-financing* ("no rich uncle helps out"), i.e. if we assume that the riskless asset is our numéraire (e.g. a cash account in dollars and the we trade IBM shares in dollars), this reads as the requirement that

for all $i = 1, \dots, n$:

$$h^0_{t_i} + h^1_{t_i} p_{t_i} = h^0_{t_{i+1}} + h^1_{t_{i+1}} p_{t_i} \tag{12}$$

The expression (12) has a natural interpretation as the wealth of the trader at time t_i. A simple calculation shows that if we start with an initial wealth w_0 at time t_0 then the wealth at time t_n is given by

$$w_{t_n} = w_0 + \sum_{i=1}^{n} h^1_{t_i}(p_{t_i} - p_{t_{i-1}}) \tag{13}$$

Moreover, for any given trading strategy $h^1_{t_i}$ we can find a sequence of corresponding investements $h^0_{t_i}$ into the riskless asset such that $(h^0_{t_i}, h^1_{t_i})_i$ is self-financing; wlog we can even assume that $h^0_{t_1} = 0$. This setup allows for very complex strategies since the dependence of each h_{t_i} on the past of the stream x can be (nearly) arbitrary complex (any adapted, measurable process). To keep the discussion focused on the main ideas we assume zero interest rate and neglect issues like transaction costs, bid-ask spreads, limit vs market orders, travel time of order from the trader to the exchange server, etc. (For more details see any textbook on financial mathematics, e.g. [5]). All these can be incorporated to the approach we outline in the next section but obfuscates the main point we want to make in this context, namely that the signature of the stream x provides an *effective summary of the market and its effects*.

To put this in the context of Section 2 and 3: we have N observations of a stream x (the price and trading volume of a risky asset) and an associated scalar value y (the outcome of a trading strategy), i.e. our observation are tuples of paths and scalars (x_i, y_i). Based on this, we want to learn a nonlinear functional f such that $y \sim f(x)$.

5. NUMERICAL RESULTS: LEARNING A TRADING STRATEGY

The family of trading strategies we have choosen is a classic but non-trivial family of strategies: the so-called *constant proportion of wealth strategy*, see [15]. At each new tick the investor rebalances the allocation of her wealth into the risky and riskless asset by following the principle that she always aims to have a constant proportion of her current overall wealth invested into the risky asset. Such trading strategies are self-financing, only requires a non-zero initial investment and despite the simple investment rule it gives rise to many interesting questions [15, 4, 18]. When they are applied in the context of HF intra-day trading, the investors usually take highly leveraged positions[8].

The dataset we use is the Thomson–Reuters tick data for the IBM stock on the three days of the 7th, 8th and 9th of April 2014 and consisted of about 110000 time ticks. The strategy we implemented in python consisted in trading over 400 subsequent time ticks with the constant proportion of wealth strategy. We considered a trader with an initial capital of 100000$ and a leverage factor of 100 (i.e. the investor is allowed to invest 100 times the wealth she posseses at a current time in stocks). That is our data set consists of tuples $(p_i, r_i)_{i=1,\dots,N}$ where each p_i is a stream of 400 subsequent stock prices and r_i denotes the return of this investment strategy. Our goal was to learn the relation $r_i = f(p_i)$ where f denotes above trading strategy. We divided the data into a training set (70% of the data) and a testing set (30% of the data) by randomly choosing the elements of the training and testing set.

The first test was a standard linear regression of the normalized return r of the trading strategy against the increment of the stock price over the 400 time ticks. The second test was to use additional higher levels of the signature (note that the level 1 signature is just the path increment, i.e. $\int_0^T dp = p_T - p_0$).

The numerical results show that already when we truncate the signature at level two (i.e. we consider only $(\int d\tilde{p}^D, \int d\tilde{p}^D \otimes d\tilde{p}^D)$ where $\tilde{p}^D$ denotes the linear interpolation between ticks of the two-dimensional lead-lag price process) the regression consistenly outperforms regression against just the price increments.

Explanatory variables	R^2	R^2 adjusted	std. dev.
Mean and Variance	0.983	0.982	0.0140
Six price increments	0.874	0.867	0.0354
Signature level 2	0.991	0.990	0.0105
Signature level 3	0.997	0.997	0.00625
Signature level 4	0.998	0.998	0.00481

Table 2: OLS regression against different sets of explanatory variables

To calculate the signature we used a simple python wrapper from the open source C++ library for computational rough paths [12] and for the regression we used MATLAB. We ran the same tests with different trading periods (100, 200, 300 time ticks instead of 400) and the results are consistent with above table.

Of course, these tests are just a first proof of concept that shows that by combining the iterated integrals of the lead-lag stream with one of the arguably simplest regression method one can gain already strong improvements. Extensions are of course possible and in work, for example for the two-dimensional lead-lag price path it is on a standard laptop feasible to go up to the level of 16 iterated integrals however we already see in Table 2 that overfitting the learning set becomes an issue. To counter this one can combine the above with the usual regularisation methods. Moreover, our methodology can be applied to the full electronic order book (i.e. breakdown into bid/ask offers and market participants), one can incorporate transaction costs etc.

5.1 Regression against price and mean-variance

We ran two numerical tests via OLS regression: first against price increments then against the mean and variance as explanatory variables. We divided each trading interval into 6 smaller time intervals $T_{\text{start}} = t_0 < t_1 < \cdots < t_6 = T_{\text{end}}$ and regressed against the price increments over the interval $[t_{i-1}, t_i]$.

$$y_T = c_0 + \sum_{i=1}^{6} c_i(p_{t_i} - p_{t_{i-1}}) + \epsilon \tag{14}$$

The residuals on the testing set are shown in Figure 3(the line "Lagged 6").

For our second test, we used as explantory variables mean $\mu = \frac{1}{N}\sum_{i=1}^{N} p_{t_i}$ and variance $\sigma^2 = \sum_{i=1}^{N}(p_{t_{i+1}} - p_{t_i})^2$. The results are shown in the figures 4 and 5.

[8]Since the stock is not expected to change very much during one trading day, the exchange can always take the stocks the investors owns as a collateral

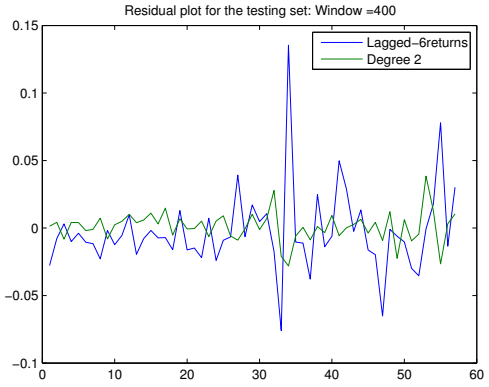

Figure 3: Each point in the plot shows the difference between the return of the trading strategy and the prediction of the regression with the regression done either against the six price increments or level-2 signature; the trading strategy was run over a trading window of subsequent 400 time ticks. The number of explanatory variables in both regressions is the same, i.e. six.

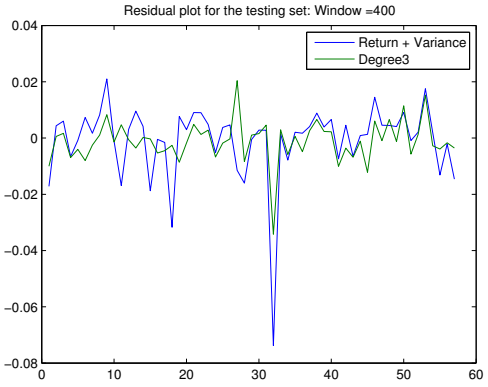

Figure 4: The plot compares the residuals on the testing set of the regression of the profit and loss (after trading periods of 400 ticks) against the mean and variance with the regression against level 3 signature.

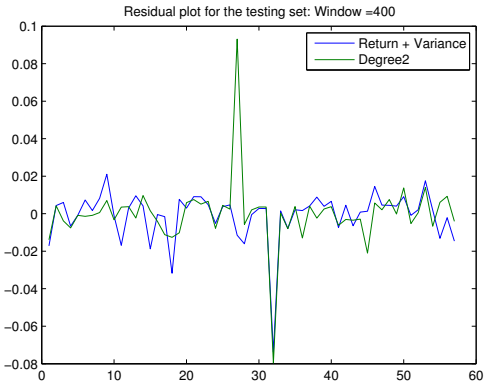

Figure 5: The same as figure 4 but with level 2 signature.

5.2 Regression against the signature

We again use OLS to find the coffecients

$$c_0, c_{1;1}, c_{1;2}, c_{2;11}, c_{2;12}, c_{2;21}, c_{2;22}$$

for the linear regression

$$\begin{aligned} y_T =& c_0 + c_{1;1}\boldsymbol{p}^{1;1}_{0,T} + c_{1;2}\boldsymbol{p}^{1;2}_{0,T} \\ &+ c_{2;11}\boldsymbol{p}^{2;11}_{0,T} + c_{2;12}\boldsymbol{p}^{2;12}_{0,T} + c_{2;21}\boldsymbol{p}^{2;21}_{0,T} + c_{2;22}\boldsymbol{p}^{2;22}_{0,T} + \epsilon \end{aligned} \tag{15}$$

Here, $\boldsymbol{p} = (1, \boldsymbol{p}^1, \boldsymbol{p}^2) \in \bigoplus_{k=0,1,2}(R^2)^{\otimes k}$ denotes the signature (truncated at level 2) of the two-dimensional stream given by linear interpolation between price points $(p_{t_i})_i$ and adding their lagged values (as detailed in Section 2.3). Figures 4 and 5 show the residuals of the linear regression of the P&L against the (level 2 and level 3) signature of the underlying price process and compare them with the residuals of the regression against mean and variance.

Note especially that the *complexity* of the regression against the signature (15) and the regression against the price increments (14) *is the same* (an intercept plus six explanatory variables: either six price increments or the first six elements of the signature); however regression against the signature significantly outperforms (15) regression against increments (14): the standard deviation on the testing set for the signature is less than a third(!) of that of the increments; on the learning set R^2 of the signature regression is more than 12% bigger than that of the increments, see Table 2.

5.3 Conclusion

We introduced a new set of features to describe a data stream and its effects. These features are the sequence of iterated integrals of paths that connect piecewise linear the sample points in the (possibly multidimensional) data stream. Classic results from pure mathematics guarantee a one-to-one corresponds of this graded sequence of finite dimensional tensors and the underlying path. We then applied this feature to describe a real-world data stream (high frequency financial data) and its effects (the profit and loss of an investor) by OLS-regression against the truncated signature. The numerical results indicate that the signature can be a powerful nonparametric method; in the regression context of our example it outperformed standard explanatory variables (e.g. mean and variance are usual explanatory variables in finance). However, more tests are needed and interesting questions remain that we hope to address in the future; for example: how to deal with overfitting that occurs by using higher signature levels; can we study more complicated trading strategies, possibly the full limit order book; what is the financial interpretation of the terms we regress against; how does our approach perform on other (non-finance) streams;

6. REFERENCES

[1] Agrachev, A. A. Introduction to optimal control theory. In *Mathematical control theory, Part 1, 2 (Trieste, 2001)*, ICTP Lect. Notes, VIII. Abdus Salam Int. Cent. Theoret. Phys., Trieste, 2002, pp. 453–513 (electronic).

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

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

[4] Cover, T. M. Universal portfolios. *Mathematical finance 1*, 1 (1991), 1–29.

[5] Delbaen, F., and Schachermayer, W. A general version of the fundamental theorem of asset pricing. *Math. Ann. 300*, 3 (1994), 463–520.

[6] Flint, G., Hambly, B., and Lyons, T. Discretely sampled signals and the rough Hoff process. *ArXiv e-prints* (Oct. 2013).

[7] Gaber, M. M., Zaslavsky, A., and Krishnaswamy, S. Mining data streams: a review. *ACM Sigmod Record 34*, 2 (2005), 18–26.

[8] Gergely Gyurkó, L., Lyons, T., Kontkowski, M., and Field, J. Extracting information from the signature of a financial data stream. *ArXiv e-prints* (July 2013).

[9] Graham, B. Sparse arrays of signatures for online character recognition. preprint.

[10] Hambly, B., and Lyons, T. Uniqueness for the signature of a path of bounded variation and the reduced path group. *Ann. of Math. (2) 171*, 1 (2010), 109–167.

[11] Levin, D., Lyons, T., and Ni, H. Learning from the past, predicting the statistics for the future, learning an evolving system. *ArXiv e-prints* (Sept. 2013).

[12] Lyons, T. CoRoPa: a C++ package for computational rough paths. `http://coropa.sourceforge.net/`.

[13] Lyons, T. J., Caruana, M., and Lévy, T. Differential equations driven by rough paths, 2007. Lectures from the 34th Summer School on Probability Theory held in Saint-Flour, July 6–24, 2004, With an introduction concerning the Summer School by Jean Picard.

[14] Oxford-Man. Oxford-man institute realised library. `http://realized.oxford-man.ox.ac.uk/`.

[15] Perold, A. F., and Sharpe, W. F. Dynamic strategies for asset allocation. *Financial Analysts Journal* (1988), 16–27.

[16] Rajaraman, A., and Ullman, J. D. *Mining of massive datasets*. Cambridge University Press, 2012.

[17] Reutenauer, C. *Free Lie algebras*. The Clarendon Press Oxford University Press, New York, 1993. Oxford Science Publications.

[18] Rotando, L. M., and Thorp, E. O. The kelly criterion and the stock market. *American Mathematical Monthly 99* (1992), 922–922.