SIGNATURE TRADING

Research Report

December 15, 2023

Badiss Ben Abdallah, Gaetan Ecrepont

CONTENTS

Abstract

In this paper, we will discuss, reproduce and try to extend the work of Futter, Horvath and Wiese on Signature Trading [1]. In Section 1, we will give a broad overview of the Signature object. In Section 2, we will show how one can use to the Signature to build quantitative trading strategies. In particular, we will implement the Sig Trader proposed in [1] and analyze its performance in various scenarios.

INTRODUCTION

Portfolio optimization is a vast theory that comprises a wealth of approaches. It is usually formulated as a problem of optimization under constraints of the form *maximize gains while minimizing risk*, the complexity thus lies in choosing the right proxies for gains and risk while keeping the problem tractable. In his seminal paper [2], Markowitz proposed a static solution to the portfolio optimization problem under the mean-variance criterion (i.e. maximize expected returns with an upper bound on the variance of returns). This marked the beginning of Modern Portfolio Theory, which has blossomed since then.

However, many of the models proposed by Modern Portfolio Theory rely on oversimplistic and indeed erroneous markets assumptions such as the (notoriously false) gaussianity of returns or the (largely disproved) markovianity of price signals. Besides, these models often fail to capture the stylized facts displayed by financial times series [3] and only take into account traditional factors. This is the price to pay to have clear and oftentimes closed-form formulas for optimal portfolios. On the other hand, the recent joint explosion of machine learning, big data and high performance computing has prompted many traders to favor a data-driven approach to investing. This new framework makes little to no assumptions on the market's dynamics (i.e. agnostic or *model-free* approach) and puts heavy emphasis on data, especially financial time series. Trading algorithms built using machine learning are often more sophisticated and less prone to interpretation than traditional models. They can thus capture more complex signals but unlike more classical models, they can rarely be explained by precise formulas.

We thus see that there is a trade-off between classical models which offer concise formulation but fail to model the markets accurately and data-driven approaches which integrate the idiosyncrasies of the markets but offer no clear theoretical explanation.

Signature Trading takes the best of both worlds by using signature terms to capture complex signals while being mathematically tractable enough to offer a closed-form formulation for the optimal portfolio. Signature Trading is *model-free* and naturally incorporates path-dependencies in the asset dynamics while offering an implicit drawdown control by design.

1
THE SIGNATURE OBJECT

1.1 Motivation

The key idea in Signature Trading is to use the signature to build trading signals. But how so?

To make things simpler, let's imagine we're trading a single asset X with no outside information. In this case, our position ξ_t in asset X at time t will be a function of the stopped process $X^t = (X_s)_{0\leq s\leq t}$, i.e. $\xi_t = \phi(X^t)$ where ϕ essentially represents our trading strategy. For instance, if we're using a trend-following strategy then ϕ could be a weighted average of the last τ returns, where the weights W and window size τ could be optimized using cross-validation for instance.

Of course there are infinitely many choices for ϕ. It is therefore common to restrict oneself to a certain class of strategies and then find optimal strategies within this subclass, thus reducing the problem to a simpler one. Signature Trading proposes strategies of the form $\xi_t = \langle \ell, \mathbb{X}_{0,t}\rangle$ where $\mathbb{X}_{0,t} \in T(\mathbb{R})$ is the signature of stopped process X^t and $\ell \in T(\mathbb{R}^*)$ is a linear functional which accepts signature objects like $\mathbb{X}_{0,t}$ as input. In other words, we express ξ_t as a linear combination of terms of the signature of the stopped process. Note that in practice, the signature is infinite in size so we use the N-truncated signature $\mathbb{X}^{\leq N}_{0,t} \in T^{\leq N}(\mathbb{R})$ and likewise we take $\ell \in T^{\leq N}(\mathbb{R}^*)$ the dual space of the N-truncated tensor algebra of $\mathbb{R}$.

A natural question is: how *rich* is the class of Signature Trading strategies? In other words, can most strategies be loosely replicated by Signature Trading strategies? The answer is yes, thanks to the signature's capability to approximate continuous functions on paths. More precisely, the *Universal Approximation theorem* states that continuous functions on paths can be approximated arbitrarily well by linear functionals on the signature, provided that we restrict ourselves to a compact subset of continuous paths.

Theorem 1 (Universal Approximation)**.** *Let $K \subset C^1([0,T],\mathbb{R}^d)$ a compact subset of paths. $\forall \phi \in C(K,\mathbb{R}), \forall \varepsilon > 0, \exists \ell \in T(\mathbb{R}^*)$ such that*

$$\sup_{X\in K} ||\phi(X) - \langle \ell, \mathbb{X}_{0,T}\rangle|| < \varepsilon$$

1.2 Preliminaries

Now that we have motivated the use of the signature for trading, let's delve into its mathematical foundations. Before even defining the signature, it's important to introduce the concept of tensor algebra, which is the space in which the signature object lives.

Definition 1 (Tensor power)**.** *Let E be a vector space and k a non-negative integer. We define the k-th tensor power of E as:*

$$E^{\otimes k} = \underbrace{E \otimes E ... E \otimes E}_{k}$$

where $\otimes$ denotes the tensor product between two vector spaces. By convention, $E^{\otimes 0} = K$ where K is the field over which E is defined.

Intuitively, if E is a vector space of finite dimension d with basis $\mathcal{B} = (e_1, ..., e_d)$, then $E^{\otimes k}$ is the vector space with basis $\mathcal{B}_k = (e_{i_1,...,i_k})_{1 \leq i_1,...,i_k \leq d}$, such that $\dim(E^{\otimes k}) = \dim(E)^k$.

$x \in E^{\otimes k}$ is called a tensor, and its rank is k. Tensors with dimension 0 (resp. 1, 2) are called scalars (resp. vectors, matrices).

Definition 2 (Tensor algebra)**.** *Let E be a vector space. The tensor algebra of E is the algebra of tensors (of any rank) on E, where the product between two elements is the tensor product $\otimes$, although we will denote it $\boxtimes$ when dealing with elements of the tensor algebra (because they can be seen as sums of tensors of different ranks, or "inhomogenous" tensors). In other words, if we denote $T(E)$ the tensor algebra of E, then we have*

$$T(E) = \bigoplus_{k=0}^{\infty} E^{\otimes k}$$

Intuitively, if E is a vector space of finite dimension d with basis $\mathcal{B} = (e_1, ..., e_d)$, then $x \in T(E)$ can be written $x = (x^0, x^1, x^2, ..., x^k, ...)$ where $x^k = (a_{i_1,...,i_k})_{1 \leq i_1,...,i_k \leq d} \in E^{\otimes k}$ is a tensor of rank k. Thus x can be seen as a series of increasingly large tensors.

Likewise, for $x, y \in T(E)$, their product $x \boxtimes y$ can be seen as a Cauchy product between infinite real series:

$$\begin{aligned}
x \boxtimes y &= \sum_{k=0}^{+\infty} \Big(\sum_{1 \leq i_1,...,i_k \leq d} x^{i_1,...,i_k} e_{i_1,...,i_k} \Big) \boxtimes \sum_{l=0}^{+\infty} \Big(\sum_{1 \leq j_1,...,j_l \leq d} y^{j_1,...,j_l} e_{j_1,...,j_l} \Big) \\
&= \sum_{m=0}^{+\infty} \sum_{k+l=m} \sum_{\substack{1 \leq i_1,...,i_k \leq d \\ 1 \leq j_1 ... j_l \leq d}} (x^{i_1,...,i_k} e_{i_1,...,i_k}) \otimes (y^{j_1,...,j_l} e_{j_1,...,j_l}) \\
&= \sum_{m=0}^{+\infty} \sum_{k+l=m} \sum_{\substack{1 \leq i_1,...,i_k \leq d \\ 1 \leq j_1 ... j_l \leq d}} x^{i_1,...,i_k} y^{j_1,...,j_l} (e_{i_1,...,i_k} \otimes e_{j_1,...,j_l}) \\
&= \sum_{m=0}^{+\infty} \sum_{k+l=m} \sum_{1 \leq i_1,...,i_k, j_1 ... j_l \leq d} x^{i_1,...,i_k} y^{j_1,...,j_l} e_{i_1,...,i_k, j_1,...,j_l}
\end{aligned}$$

Definition 3 (Truncated tensor algebra)**.** *Let E be a vector space and N a non-negative integer. The N-truncated tensor algebra of E is the subset of the tensor algebra of E containing the elements that have null tensors for any rank $k > N$. In other words, if we denote $T^{(N)}(E)$ the N-truncated tensor algebra of E, then we have*

$$T^{(N)}(E) = \{x = (x^0, x^1, x^2, ..., x^k, ...) \in T(E), x^k = 0 \ \forall k > N\}$$

.

Intuitively, $T^{(N)}(E)$ contains elements of the form $x = (x^0, x^1, x^2, \ldots, x^N, 0, \ldots)$, such that if E is of finite dimension d with basis $\mathcal{B} = (e_1, \ldots, e_d)$, then $\mathcal{B}^N = \bigvee_{k=0}^{N} \mathcal{B}_k$ is a basis of $T^{(N)}(E)$, where $\mathcal{B}_k = (e_{i_1,\ldots,i_k})_{1 \le i_1,\ldots,i_k \le d}$. Thus $\dim\left(T^{(N)}(E)\right) = 1 + d + d^2 + \cdots + d^N = \frac{d^{N+1}-1}{d-1}$.

Likewise, for $x, y \in T^{(N)}(E)$, their product $x \boxtimes y$ can be seen as a Cauchy product between finite real series:

$$
\begin{aligned}
x \boxtimes y &= \sum_{k=0}^{N} \Big(\sum_{1 \le i_1,\ldots,i_k \le d} x^{i_1,\ldots,i_k} e_{i_1,\ldots,i_k} \Big) \boxtimes \sum_{l=0}^{N} \Big(\sum_{1 \le i_1,\ldots,i_l \le d} y^{j_1,\ldots,j_l} e_{j_1,\ldots,j_l} \Big) \\
&= \sum_{m=0}^{2N} \sum_{k+l=m} \sum_{\substack{1 \le i_1,\ldots,i_k \le d \\ 1 \le j_1 \ldots j_l \le d}} x^{i_1,\ldots,i_k} e_{i_1,\ldots,i_k} \otimes y^{j_1,\ldots,j_l} e_{j_1,\ldots,j_l} \\
&= \sum_{m=0}^{2N} \sum_{k+l=m} \sum_{\substack{1 \le i_1,\ldots,i_k \le d \\ 1 \le j_1 \ldots j_l \le d}} x^{i_1,\ldots,i_k} y^{j_1,\ldots,j_l} e_{i_1,\ldots,i_k} \otimes e_{j_1,\ldots,j_l} \\
&= \sum_{m=0}^{2N} \sum_{k+l=m} \sum_{1 \le i_1,\ldots,i_k, j_1 \ldots j_l \le d} x^{i_1,\ldots,i_k} y^{j_1,\ldots,j_l} e_{i_1,\ldots,i_k,j_1,\ldots,j_l} \in T^{(2N)}(E)
\end{aligned}
$$

Note that if we consider the N-truncature of the above element of $T^{(2N)}(E)$ i.e. if we restrict ourselves to tensors of rank less or equal to N in the sum, we have:

$$
x \boxtimes y = \sum_{m=0}^{N} \sum_{k+l=m} \sum_{1 \le i_1,\ldots,i_k, j_1 \ldots j_l \le d} x^{i_1,\ldots,i_k} y^{j_1,\ldots,j_l} e_{i_1,\ldots,i_k,j_1,\ldots,j_l} \in T^{(N)}(E) \tag{1}
$$

Tensor indexing Let E be a vector space of finite dimension d with basis $\mathcal{B} = (e_1, \ldots, e_d)$, and let $x \in T(E)$. Then $x^k = (a_{i_1,\ldots,i_k})_{1 \le i_1,\ldots,i_k \le d} \in E^{\otimes k}$ is a tensor of rank k. We need multi-indices to index x^k and we will use words $\mathbf{w}$ to represent these multi-indices. For instance, $x^3_{\mathbf{211}} = a_{211}$ and $x^5_{\mathbf{02111}} = a_{02111}$. In fact we can use the same indexing for x itself since the length of the word $\mathbf{w}$ will indicate which tensor we want to access. For instance if $\mathbf{w} = \mathbf{02}$ we are accessing tensor x^2 of rank 2, and inside this tensor we are taking coordinate 0 along the first dimension and coordinate 2 along the second dimension. We will denote W_Z^L (resp. $W_Z^{\le L}$) the set of words of length L (resp. length $\le L$) in the alphabet $\mathcal{A}_Z = \{\mathbf{0}, \mathbf{1}, \ldots, \mathbf{Z\text{-}1}\}$. For instance,

$$
W_3^{\le 2} = \{\mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbf{00}, \mathbf{01}, \mathbf{02}, \mathbf{10}, \mathbf{11}, \mathbf{12}, \mathbf{20}, \mathbf{21}, \mathbf{22}\}
$$

1.3 The Signature

We can now formally define the signature object.

1.3.1 • DEFINITION

Definition 4 (Signature)**.** *Let $X \in C([0,T],\mathbb{R}^d)$ be a smooth path. Then the signature transform of X between time s and t where $0 \leq s \leq t \leq T$ is given by:*

$$\mathbb{X}_{s,t} = (1, \mathbb{X}^1_{s,t}, ..., \mathbb{X}^k_{s,t}, ...) \in T(\mathbb{R}^d)$$

where

$$\mathbb{X}^k_{s,t} = \int \cdots \int_{s<u_1<\cdots<u_k<t} dX_{u_1} \otimes \ldots \otimes dX_{u_k} \in \left(\mathbb{R}^d\right)^{\otimes k}$$

i.e.

$$\mathbb{X}^k_{s,t} = (\mathbb{X}^{i_1,\ldots,i_k}_{s,t})_{1\leq i_1,\ldots,i_k \leq d}$$

where

$$\mathbb{X}^{i_1,\ldots,i_k}_{s,t} = \int \cdots \int_{s<u_1<\cdots<\ldots<u_k<t} dX^{i_1}_{u_1} \ldots dX^{i_k}_{u_k} \in \mathbb{R}$$

Intuitively, $\mathbb{X}^n_{s,t}$ is a tensor of rank n containing the scalars $\mathbb{X}^{i_1,\ldots,i_n}_{s,t}$ for all n-tuples $(i_1,\ldots,i_n) \in \{1,\ldots,d\}^n$.

Note that the signature $\mathbb{X}_{s,t}$ can also be written $S(X)_{s,t}$ for convenience. Also, note that when we refer to the signature of X without explicitly mentioning s and t, we are referring the signature over $[0,T]$. In this case we can simply write $\mathbb{X}$ or $S(X)$ without specifying the time interval over which we're computing the signature.

One can also define the signature recursively, which will prove useful to demonstrate some of its properties.

Definition 5 (Signature, recursive definition)**.** *For any index $i \in \{1,...,d\}$, we define the quantity:*

$$\mathbb{X}^i_{s,t} = \int_s^t dX^i_u = X^i_t - X^i_s$$

which is simply the increment of the i-th coordinate of the path between time s and time t. This allows us to define recursively $\mathbb{X}^{i_1,\ldots,i_k}_{s,t}$ for a k-tuple of indices $(i_1,...,i_k) \in \{1,...,d\}^k$:

$$\begin{aligned}
\mathbb{X}^{i_1,\ldots,i_k}_{s,t} &= \int \cdots \int_{s<u1<\cdots<u_k<t} dX^{i_1}_{u_1} \ldots dX^{i_k}_{u_k} \\
&= \int_{s<u_k<t} \Big(\int \cdots \int_{s<u_1<\cdots<u_{k-1}<u_k} dX^{i_1}_{u_1} \ldots dX^{i_{k-1}}_{u_{k-1}} \Big) dX^{i_k}_{u_k} \\
&= \int_{s<u_k<t} \mathbb{X}^{i_1,\ldots,i_{k-1}}_{s,u_k} dX^{i_k}_{u_k}
\end{aligned}$$

Definition 6 (Truncated signature)**.** *Let $X \in C([0,T],\mathbb{R}^d)$ be a smooth path. Then its N-truncated signature, denoted $\mathbb{X}^{\leq N}$ or $S(X)^{\leq N}$, is the truncation of its signature up to the tensor of rank N, i.e.*

$$\mathbb{X}^{\leq N} = (1, \mathbb{X}^1, \mathbb{X}^2, \ldots, \mathbb{X}^N) \in T^{(N)}(\mathbb{R}^d)$$

1.3.2 • First Intuitions

The signature can be an intimidating object at first. We can study its behavior in small dimensions and with linear paths in order to get a better understanding of it.

Case $d = 1$ Let's first consider the case where $d = 1$ i.e. we are looking at a one-dimensional path. In this case, note that $\mathbb{R}^{\otimes k} \simeq \mathbb{R}$ such that $T(\mathbb{R}) \simeq \mathbb{R}^{\mathbb{N}}$ i.e. the signature of $X \in C([0,T],\mathbb{R})$ is simply a numeric series indexed by $\mathbb{N}$. The first terms are easy to compute:

$$\mathbb{X}^1 = \int_0^T dX_u = X_T - X_0$$

$$\mathbb{X}^2 = \iint_{0<u<v<T} dX_u dX_v = \int_0^T dX_u \int_u^T dX_v = \int_0^T dX_u(X_T - X_u) = X_T(X_T - X_0) - \frac{X_T^2 - X_0^2}{2}$$

$$\mathbb{X}^2 = \frac{(X_T - X_0)^2}{2}$$

Thus, we see that the first term $\mathbb{X}^1$ can be thought of as the total variation of X while the second term $\mathbb{X}^2$ can be thought of as the quadratic variation of X. There is some similarity with the moments of a random variation but the comparison doesn't extend further and higher order terms of the signature are harder to interpret. What matters crucially is that these higher order terms encode complex, nonlinear geometric properties of the path. (hence they are good candidates to capture sophisticated signals hidden in a stochastic process)

The same interpretation applies to $d > 1$, as shown below.

Case $d > 1$ Likewise, when $d > 1$, we can interpret the vector $\mathbb{X}^1$ as the total variation of X along each dimension and the matrix $\mathbb{X}^2$ can be thought of as the covariation of X between each dimension. Indeed, we have that:

$$\mathbb{X}^1 = \int_0^T dX_u = X_T - X_0 = \left(X_T^i - X_0^i\right)_{1\le i\le d} \in \mathbb{R}^d$$

$$\mathbb{X}^2 = \iint_{0<u<v<T} dX_u \otimes dX_v = \left(\iint_{0<u<v<T} dX_u^i dX_v^j\right)_{1\le i,j\le d} \in \mathcal{M}_d(\mathbb{R})$$

Case of a linear path Let's now consider a linear path $X : t \in [0,T] \mapsto a + bt \in \mathbb{R}^d$ where $a, b \in \mathbb{R}^d$. In this case, a simple induction shows that

$$\mathbb{X}_{s,t}^{i_1,\dots,i_k} = b_{i_1} \cdots b_{i_k} \cdot \frac{(t-s)^k}{k!} \tag{2}$$

In particular, the $k!$ guarantees that signature terms decay as the tensor rank k increases. In other words, the signature "converges".

Note that if $d = 1$, then

$$\mathbb{X}_{s,t} = \left(1, b\frac{t-s}{1!}, b^2\frac{(t-s)^2}{2!}, b^3\frac{(t-s)^3}{3!}, \dots, b^k\frac{(t-s)^k}{k!}, \dots\right)$$

which corresponds to the series expansion of $\exp\big(b(t-s)\big)$.

1.3.3 • Key Properties

Although a complex object, the signature has a lot of interesting properties.

Invariance by translation Since the signature depends only on iterated integrals of differentials of X along each of its dimension, it is invariant by translation i.e. for any smooth path $X \in C([0,T], \mathbb{R}^d)$ and for any $a \in \mathbb{R}^d$, $S(X) = S(X + a)$.

Invariance by time reparametrisation Given a nondecreasing bijection $\varphi : [T_1, T_2] \to [0,T]$, we may define the time-reparametrized path $X^\varphi = X \circ \varphi : [T_1, T_2] \to \mathbb{R}^d$. Then $S(X^\varphi) = S(X)$. This comes from the fact that for any paths $X, Y \in C([0,T], \mathbb{R})$, $\int_{T_1}^{T_2} X_t^\varphi dY_t^\varphi = \int_0^T X_t dY_t$, combined with the recursive definition of the signature.

Chen's identity Let us consider two paths $X^1 : [a,b] \to \mathbb{R}^d$ and $X^2 : [b,c] \to \mathbb{R}^d$ where $a < b < c$. We define their concatenation $X = X^1 * X^2$ by $X_t = X_t^1$ $\forall t \in [a,b]$ and $X_t = X_b^1 + (X_t^2 - X_b^2)$ $\forall t \in [b,c]$. Then Chen's identity states that

$$S(X) = S(X^1) \otimes S(X^2) \tag{3}$$

In particular, Chen's identity allows us to compute the signature of any discrete path, and since any path stored on a computer must be discrete, it effectively allows us to compute any signature. Indeed, a discrete path has timestamps $(t_i)_{0 \le i \le p}$ and each subpath $X^i = (X_t)_{t_i \le t \le t_{i+1}}$ is linear, so we can easily compute its signature, and then Chen's identity yields $S(X) = S(X^0) \boxtimes S(X^1) \boxtimes \cdots \boxtimes S(X^{p-1})$ such that we can compute the final signature as a tensor product of several simple signatures.

Time complexity of computing the signature We can estimate the time complexity of Chen's signature computation method. Note that signature libraries also use Chen's method but with slight tweaks to speed up computation, including parallelization [4]. Let's have a look at the time complexity of computing $\mathbb{X}_{0,T}^{\le N}$

- For a **linear path** $X : t \in [0,T] \mapsto a + bt \in \mathbb{R}^d$: complexity is $O(d^N)$.

  Proof. Using equation (2), we have that $\mathbb{X}^{i_1,\ldots,i_k} = b_{i_k} \frac{T-0}{k} \cdot \mathbb{X}^{i_1,\ldots,i_{k-1}}$ such that $\mathbb{X}^k$ can be computed in $O(d^k)$ from $\mathbb{X}^{k-1}$ since each of its d^k term is computed in $O(1)$. Therefore $\mathbb{X}^{\le N}$ is recursively computed in $O(1 + d + d^2 + \cdots + d^N) = O(\frac{d^{N+1}-1}{d-1}) = O(d^N)$. □

- For a **discrete path** (i.e. piecewise-linear) $X : t \in \{t_i\}_{1 \le i \le p} \mapsto X_t \in \mathbb{R}^d$: complexity is $O(\log_2(p) d^{2N})$.

  Proof. Indeed, using equation (3) we have $S(X)^{\le N} = S(X^0)^{\le N} \boxtimes S(X^1)^{\le N} \boxtimes \cdots \boxtimes S(X^{p-1})^{\le N}$ where $S(X^i)^{\le N}$ is the N-truncated signature of $X^i = (X_t)_{t_i \le t \le t_{i+1}}$. Using fast multiplication this amounts to computing $\log_2(p)$ products of the form $x \boxtimes y$ where $x, y \in T^{(N)}(\mathbb{R}^d)$. Such products are computed in $O(1 + d^2 + d^4 + \cdots + d^{2N}) = O(\frac{d^{2(N+1)}-1}{d^2-1}) = O(d^{2N})$ according to equation (1). Thus computing $\mathbb{X}^{\le N}$ is in $O(\log_2(p) d^{2N})$. □

To conclude, Chen's method allows us compute the signature of discrete paths in $O(\log_2(p)d^{2N})$ where d is the path dimension, p is the number of time steps, and N is the signature depth. We note that the dependence on p is satisfying, as is the dependence on d (no curse of dimensionality). However the complexity is exponential in N. This last remark is of high importance as we'll see in the next section: the exponential complexity in N creates a computational bottleneck when attempting Signature Trading at high depth levels.

1.3.4 • Path Transformations

Prior to computing the signature of a path, one may apply certain transform operations. Certain judicious transforms can indeed make the subsequent signature more insightful. We present two such transforms which we use for Signature Trading: lead-lag and time-augmentation.

Lead-Lag transform Given a discrete path $X : t \in \{t_i\}_{1\leq i\leq p} \mapsto X_t \in \mathbb{R}^d$, we can define its lead-lag transform $(X^{LL}_{t_i})_{1\leq i\leq 2p-1}$ where

$$X^{LL}_{t_i} = \begin{cases} (S_{t_i}, S_{t_i}) & \text{when } i \text{ is even} \\ (S_{t_{i+1}}, S_{t_i}) & \text{when } i \text{ is odd} \end{cases}$$

Therefore $X^{LL} = (X^{\text{lead}}, X^{\text{lag}}) \in \mathbb{R}^d \times \mathbb{R}^d$ contains both a leading and a lagging signal for any given timestep. In particular, note that if X is a discrete path in dimension d with p steps, then X^{LL} is a discrete path in dimension $2d$ with $2p-1$ steps.

Example For the sake of simplicity, let's consider a one-dimensional path with just 4 steps: $X = (1, 3, 7, 5)$. The lead-lag transform of X is simply given by

$$X^{LL} = \left(\begin{pmatrix}1\\1\end{pmatrix}, \begin{pmatrix}3\\1\end{pmatrix}, \begin{pmatrix}3\\3\end{pmatrix}, \begin{pmatrix}7\\3\end{pmatrix}, \begin{pmatrix}7\\7\end{pmatrix}, \begin{pmatrix}5\\7\end{pmatrix}, \begin{pmatrix}5\\5\end{pmatrix} \right)$$

Below is an illustration of the lead-lag transform of a more complex signal.

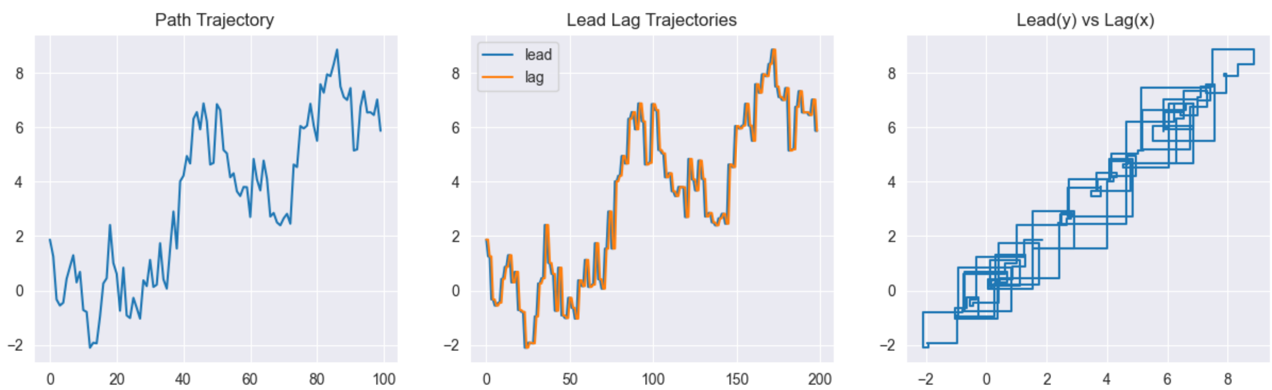


Figure 1: Lead-lag of a random price trajectory

Utility The lead-lag transform naturally gives access to the *variation* of the path at hand since we have a lead and a lag component. This can be interesting to capture certain signals. For instance in a financial setting, capturing the quadratic variation of a price process can be insightful if we model it as a random walk. Likewise, in the field of machine learning, feature engineering using the signature of the lead-lag transformed signal dramatically improves performance compared to the signature of the vanilla path [6].

Time-augment Given a process $X \in (\mathbb{R}^d)^{[0,T]}$, we can define the time-augmented process $\hat{X} \in (\mathbb{R} \times \mathbb{R}^d)^{[0,T]}$ with $\hat{X}_t = (t, X_t)\ \forall t \in [0,T]$.

Example If we consider again the discrete path $X = (1,3,7,5)$, then the corresponding time-augmented process is given by

$$\hat{X} = \left(\begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 3 \\ 7 \end{pmatrix}, \begin{pmatrix} 4 \\ 5 \end{pmatrix} \right)$$

Utility One nice feature of time-augmented paths is that their signature is unique. Indeed, it can be shown that signature objects are unique only up to tree-like equivalence, a notion defined in details in [7]. However, it can also be shown that if the path has a monotone component, then its signature is unique. Therefore, adding the time guarantees uniqueness of the signature object.

1.3.5 • Use Cases

As an object which encodes geometric properties of path, the signature is a natural candidate for feature engineering when dealing with path data. Various original applications have emerged in recent years, from Chinese character recognition to sound compression to financial time series classification [8].

2
SIGNATURE TRADING

2.1 Signature Trading framework

We consider a very general trading framework where we have d tradable assets whose prices at time t are given by $X_t \in \mathbb{R}^d$, such that if we denote ξ_t^m our position in the mth asset at time t, then our PnL writes:

$$PnL = \sum_{m=1}^{d} \int_0^T \xi_t^m dX_t^m$$

The hypothesis in Signature Trading is that we restrict ourselves to strategies which are expressed as linear functionals over terms of the signature, i.e. $\xi_t^m = \langle \ell_m, \hat{\mathbb{Z}}_{0,t} \rangle$. We thus have

$$PnL = \sum_{m=1}^{d} \int_0^T \langle \ell_m, \hat{\mathbb{Z}}_{0,t} \rangle dX_t^m$$

The problem with the above expression is that the Itô integral is quite unconvenient to manipulate if we want to optimize our PnL with respect to some constraint.

Interestingly, this integral could be explicitly computed as a higher order term of the signature if it was a Stratonovich integral. The trick is to go from Itô to Stratonovich using the lead-lag transform.

The proof is quite technical, but the basic idea is the following:

$$\int_0^T \langle \ell_m, \hat{\mathbb{Z}}_{0,t} \rangle dX_t^m \simeq \int_0^T \langle \ell_m, \hat{\mathbb{Z}}_{0,t}^{\text{lag}} \rangle dX_t^{\text{lead},m} = \langle \ell_m \mathbf{f}(m), \hat{\mathbb{Z}}_{0,T}^{LL} \rangle \tag{4}$$

where $\mathbf{f}(m)$ is a "shift operator" which basically allows us to go one level higher in the signature term i.e. capture an element of $\hat{\mathbb{Z}}_{0,T}^{LL}$ belonging to the $(k+1)$-th tensor of the signature instead of the kth.

Note that the approximation we made becomes increasingly true as our mesh size $\pi = \sup_i |t_{i+1} - t_i|$ converges towards 0 (in the limit $\pi = 0$ we have that $S^{\text{lead}} = S^{\text{lag}} = S$ for any signal S).

This new expression of the PnL is much nicer to work with since it is just a sum of linear functionals over the signature $\hat{\mathbb{Z}}_{0,T}^{LL}$, i.e. it is a linear combination of the terms in $\hat{\mathbb{Z}}_{0,T}^{LL}$. We can thus easily compute $\mathbb{E}[PnL]$ and $\text{Var}(PnL)$ in terms of just the expected signature $\mathbb{E}[\hat{\mathbb{Z}}_{0,T}^{LL}]$.

Finally, we can now setup our optimization problem under the classical mean-variance criterion i.e. we want to solve the following problem:

$$\max_{\ell_1,\dots,\ell_d \ \text{s.t.} \text{Var}(PnL) \leq \Delta} \mathbb{E}[PnL]$$

Since $\mathbb{E}[PnL]$ and $\text{Var}(PnL)$ are linear in $\hat{\mathbb{Z}}_{0,T}^{LL}$, this seemingly complex optimization problem can in fact be solved explicitly i.e. we have closed-formed formulas for the functionals, which look very familiar to Markowitz's mean-variance optimization portfolio ($w^* = \frac{\Sigma^{-1}\mu}{2\lambda}$). Indeed we have that:

$$\langle \ell_m, e_{\mathbf{w}} \rangle = \frac{\left((\Sigma^{\text{sig}})^{-1}\mu^{\text{sig}}\right)_{\mathbf{wf}(m)}}{2\lambda} \quad \forall 1 \leq m \leq d,\ \forall \mathbf{w} \in W^{\leq N}_{N+d+1}$$

In this expression, μ^{sig} and Σ^{sig} are a vector and a matrix populated using terms of $\mathbb{E}[\hat{\mathbb{Z}}^{LL,\leq 2(N+1)}_{0,T}]$, and λ is inversely proportional to $\sqrt{\Delta}$ and represents the risk aversion, just like in Markowitz's solution. The derivation of this formula is rigorously demonstrated in the original paper [1] but essentially it use the Lagrangian method, just like in Markowitz's paper.

Note that fitting in practice, fitting an Order N Sig Trader requires to estimate $\mathbb{E}[\hat{\mathbb{Z}}^{LL,\leq 2(N+1)}_{0,T}]$ i.e. to compute signatures at depth $2(N+1)$. Given the cost $O(\log_2(p)d^{2N}$ to compute a signature with p points and d dimension at depth N, we see that there is an exponential dependency on N and therefore having to compute signatures at depth $2(N+1)$ creates a computational bottleneck. For this reason, we were not able to train our Sig Traders above $N = 2$, unlike in the paper which goes up to $N = 3$.

2.2 Sig-Trader Algorithm

Definition 7 (Time-augmented market factor process)**.** *Let X be a process in $\mathbb{R}^d$ representing the prices of d tradable assets and f a process in $\mathbb{R}^N$ representing an exogenous signal related to X. (X, f) is called the market factor process and we define the time-augmented market factor process as :*

$$\hat{Z}_t = (t, X_t, f_t) \in \mathbb{R}^{1+d+N}$$

We have seen in the above subsection that within our very general trading framework, we can explicitly compute the optimal functionals ℓ^*_m $\forall 1 \leq m \leq d$ under the approximation in equation (4). We have expressed these functionals in terms of vector μ^{sig} and matrix Σ^{sig}, which are populated with elements of $\mathbb{E}[\hat{\mathbb{Z}}^{LL,\leq N}_{0,T}]$. Therefore, in order to "fit" the Sig Trader i.e. find the optimal functionals, we need to estimate this expected signature. The most natural estimator is simply the empirical mean, although it requires us to have several independent samples of the same underlying process $\hat{Z}$ we want to trade. We thus obtain the following training algorithm.

$\square$

Once the Sig Trader is fitted, we can easily deploy it since we only need to compute $\xi^m_t = \langle \ell_m, \hat{\mathbb{Z}}^{\leq N}_{0,t} \rangle$ for all time step t on which we trade. More precisely, we obtain the following trading algorithm.

Algorithm 2: Trading at time t using the Sig Trader

Input: The previous stopped underlying asset and signal paths $(X_s)_{s\in[0,t]}, (f_s)_{s\in[0,t]}$, linear functionals $\ell_m \in T\left((\mathbb{R}^{N+d+1})^*\right) \quad \forall 1 \leq m \leq d\}$.
Output: $\xi_t^m \quad \forall m \in \{1,\dots,d\}$: The optimal trading strategy

1. Concatenate t, X, f to obtain the $(N+d+1)$-dimensional time-augmented market factor process $(\hat{Z}_s)_{s\in[0,t]}$.
2. Compute the N-truncated signature of the stopped market factor process, $\hat{\mathbb{Z}}_{0,t}^{\leq N}$.
3. For each asset $m \in \{1,\dots,d\}$, compute the optimal strategy as $\xi_t^m = \left\langle \ell_m, \hat{\mathbb{Z}}_{0,t}^{\leq N} \right\rangle$
4. Return $\xi_t^m \quad \forall m \in \{1,\dots,d\}$

2.3 Applications

Let's now review several use cases for the Sig Trader. We will give various types of "easy" processes to the Sig Trader and see if it learns to capture the underlying signal.

2.3.1 • Markowitz

Before delving into more complex signals, let's observe the following fact: an Order 0 Sig Trader is static because $\xi_t^m = \left\langle \ell_m, \hat{\mathbb{Z}}_{0,t}^{\leq 0} \right\rangle = \langle \ell_m, 1\rangle = \ell_m$ and therefore for each asset m, $\ell_m \in \mathbb{R}$ is the weight of that asset in the Sig Trader portfolio. The Order 0 Sig Trader portfolio is thus represented by the vector $w_{\text{SigTrader0}} = \left(\ell_1, \dots \ell_n\right)$ where n is the number of assets. Thus, if we trade n assets following lognormal prices, one should expect the Sig Trader to find the optimal solution given by Markowitz since we are optimizing the same process under the same mean-variance criterion.

In order words, if we feed samples of prices processes $X \in \mathbb{R}^n$ with log returns following $\ln(X) \sim \mathcal{N}(\mu, \Sigma)$, then we expect to have

$$w_{\text{Markowitz}} = w_{\text{SigTrader0}} = \frac{\Sigma^{-1}\mu}{2\lambda}$$

where λ is the risk aversion parameter, which is inversely proportional to $\sqrt{\Delta}$ for the Sig Trader.

Experiment We thus trained the Sig Trader on $M = 1000$ i.i.d. samples of the same lognormal price process $\ln(X) \sim \mathcal{N}(\mu, \Sigma)$ with $n = 10$ tradable assets. The returns were computed using daily frequency $f_{\text{daily}} = 252$ over $T = 1$ year and μ and Σ were generated arbitrarily. The weights obtained after fitting to Sig Trader and those given by Markowitz are are almost exactly the same, demonstrating that the Order 0 Sig Trader is indeed able to learn the optimal static strategy under the mean-variance criterion.

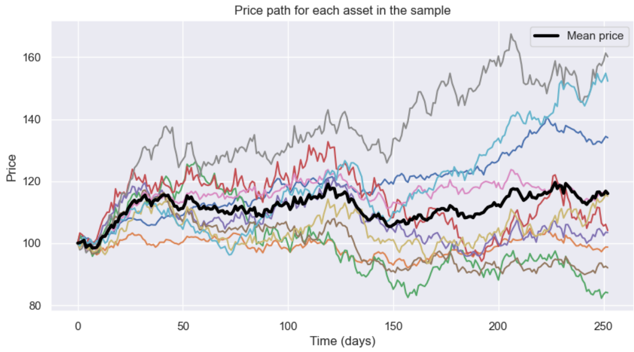


Figure 2: Simulated price process X with $n = 10$ assets and $\ln(X) \sim \mathcal{N}(\mu, \Sigma)$

Figure 3: Optimal weights $w_{\text{Markowitz}}$ and $w_{\text{SigTrader0}}$

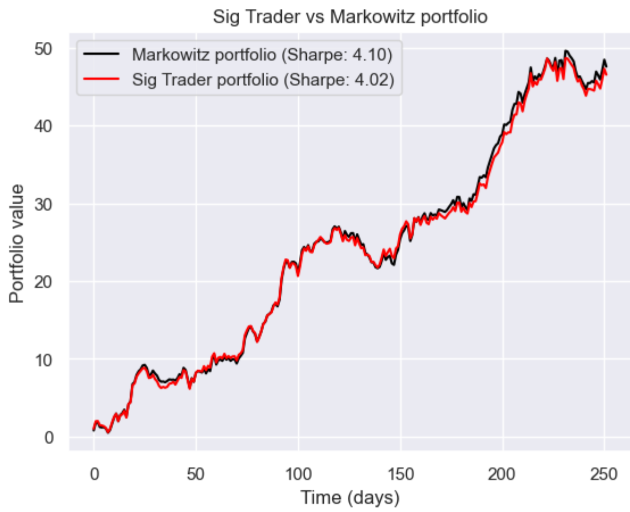


Figure 4: Markowitz and SigTrader0 portfolio growth over $T = 1$ year

NB: Note that there is no exogenous signal here, i.e. $f = 0$.

2.3.2 • Mean-reversion

Let's now see if the SigTrader can trade on a mean-reverting signal, which is one of the most classical example in the financial literature for algorithmic trading. Let us consider a one-dimensional mean-reverting time series $(X_t)_{t\in[0,T]}$. It will act as the main asset we want to trade on. In this case, the market process factor is particularly simple : $\hat{Z}_t = (t, X_t) \in \mathbb{R}^2$. The dynamic of X is given by the Ornstein-Uhlenbeck process

$$dX_t = \kappa(X_0 - X_t)dt + \sigma^X dWt$$

The simplicity of this first model leads us to consider only the first order of the signature to avoid useless time-consuming computations : $\hat{\mathbb{Z}}^{\leq 1}_{0,t} = \left(1, \begin{bmatrix} t \\ X_t - X_0 \end{bmatrix}\right)$ and the Sig Trader strategy is as follows :

$$\boxed{\langle \ell, \hat{\mathbb{Z}}^{\leq 1}_{0,t} \rangle = \ell^0 \cdot 1 + \ell^1 \cdot t + \ell^2 \cdot (X_t - X_0)}$$

Since we're dealing with a stationary process i.e. no dependence on t, we expect $\ell^1 \simeq 0$. Besides, since X is mean-reverting, the Trader ought to *buy* when $X_t - X_0 < 0$, and *sell* when $X_t - X_0 > 0$. Consequently ℓ^2 should be negative. ℓ_0 can be positive or negative depending on X_0 as it serves to recenter the spread $X_t - X_0$ around 0.

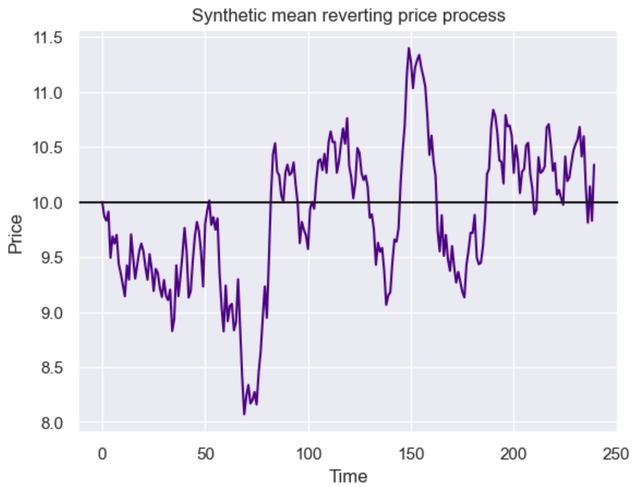


Figure 5: Example of simulated mean-reverting signal $X_0 = 10, \sigma^X = 10, \kappa = 100, T = 1/12$

Experiment We use parameters $X_0 = 10, \sigma^X = 10, \kappa = 100, M = 1000, f_{\text{hourly}} = 252 \times 8, T = 30 \times \frac{1}{252}$. This means that we have 1000 independent mean-reverting (price) paths centered around $X_0 = 10$, with (annualized) volatility σ^X and pullback strength κ. We use hourly frequency and trade over 1 month.

Upon fitting the Order 1 Sig Trader, we find

$$\ell = \left(\underbrace{0.21}_{\text{Level 0}}, \underbrace{\begin{bmatrix} -0.020 \\ -1.57 \end{bmatrix}}_{\text{Level 1}} \right)$$

Which is coherent with what we expected.

2.3.3 • Pairs Trading

Let us suppose we don't have any additional signal to improve our strategy. In this section, we will work with the time series of two correlated underlying assets, a method commonly known as **Pairs Trading**. The main idea is that the spread should offer some clues about the future co-movements of the assets. If we assume the spread is somewhat mean-reverting, our strategy should *buy* when the spread drops below a certain threshold and *sell* when the spread exceeds that limit.

Let us consider two correlated underlying assets following :

$$\begin{aligned} dX_t &= \sigma^X dW_t^X \\ dY_t &= \kappa\,(X_t - Y_t)\,dt + \sigma^Y dW_t^Y \end{aligned}$$

- X is a zero-drift Brownian motion and Y is a mean-reverting process with a drift proportional to the difference between X and Y.
- σ^X and σ^Y adjust the amount of noise in the data, while κ modulates the correlation between the two assets's price movements.

Figure 6: Example of simulated pair trading assets with $X_0 = Y_0 = 100, \sigma^X = 0.2, \sigma^Y = 0.3, \kappa = 50, T = 1/12$

The linearity of the relation should once again enable the *alpha* to be contained only in the first level of the signature. Thus we limited our computations to $\hat{\mathbb{Z}}^{\leq 1}$, which can still capture the mean-reverting dynamic of the spread. We applied the same reasoning as for the mean-reverting process:

$$\hat{\mathbb{Z}}_{0,t}^{\leq 1} = \left(1, \begin{bmatrix} t \\ X_t - X_0 \\ Y_t - Y_0 \end{bmatrix}\right)$$

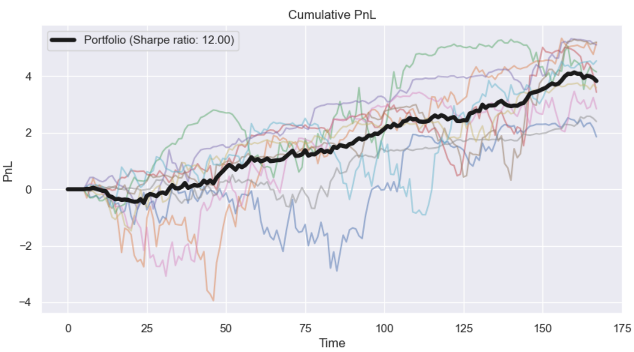
Then our strategy becomes for each asset $\mathcal{A} \in \{X, Y\}$ $\langle \ell^{(\mathcal{A})}, \hat{\mathbb{Z}}_{0,t}^{\leq 1} \rangle = \ell_0^{(\mathcal{A})} \cdot 1 + \ell_1^{(\mathcal{A})} \cdot t + \ell_2^{(\mathcal{A})} \cdot (X_t - X_0) + \ell_3^{(\mathcal{A})} \cdot (Y_t - Y_0)$. We expect $\langle \ell^{(\mathcal{A})}, \hat{\mathbb{Z}}_{0,t}^{\leq 1} \rangle$ and $X_t - Y_t$ to be colinear as mentioned before. Hence we must have for both assets:

- $\ell_0^{(\mathcal{A})} \simeq 0$ because there is no need to recenter the spread $(X_t - X_0) - (Y_t - Y_0)$ since we have $X_0 = Y_0$
- $\ell_1^{(\mathcal{A})} \simeq 0$ since time doesn't matter (we model stationary processes)
- $\ell_2^{(\mathcal{A})} \simeq -\ell_3^{(\mathcal{A})}$ since we want to capture the spread. The absolute magnitude of these two terms will determine the size of the positions we take and therefore will depend on the variance criterion Δ of the Sig Trader

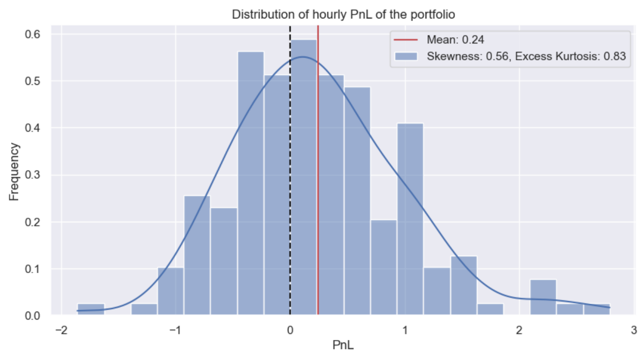
Experiment These ideas are confirmed by the numerical experiment which gives the following functionals:

$$\ell^{(X)} = \left(-0.021, \begin{bmatrix} -0.012 \\ -0.078 \\ 0.081 \end{bmatrix}\right) \quad \text{and} \quad \ell^{(Y)} = \left(0.093, \begin{bmatrix} -0.089 \\ 0.083 \\ -0.098 \end{bmatrix}\right)$$

Besides, we note consistent performance across both in-sample and out-sample data. This demonstrates that the Sig Trader does learn to trade mean-reverting signals and does not overfit on the training samples. Therefore, the Order 1 Sig Trader can effectively do pairs trading.



(a) Cumulative PnL for a 10-asset portfolio



(b) Distribution of hourly PnL of the portfolio

Figure 7: Parameters: $\kappa = 50$, $\sigma^X = 0.2$, $\sigma^Y = 0.3$, $T_{\text{trade}} = 1/12$

2.3.4 • Exogenous Signal

In this new scenario, we will assume the asset is driven by the dynamic of an exogenous signal f enclosed in a given function ϕ such that $dX_t = \phi(t, (f_s)_{s\in[0,t]})dt + \sigma^X dW_t^X$.

We made the following assumptions :

- The observable signal f follows a zero-centered Ornstein-Uhlenbeck process: $df_t = -\kappa f_t + \sigma^f dW_t^f$.
- Z denotes a latent process containing the decaying information of f between instant 0 and t: $Z_t = \int_0^t K(t-s)df_s$ with $K(t,s) = \exp(-\alpha(t-s))$ our decay kernel.
- X is the integral of Z with some added noise: $dX_t = Z_t dt + \sigma^X dW_t^X$. It behaves like an arithmetic Brownian motion with long term zero drift, but short term momentum based on the local sign of Z
- σ^X and σ^f parameters control the influence of the Brownian noise over the signal we're trying to capture.

Figure 8: Simulated path for the exogenous model with : $\kappa = 5$, $\sigma^X = 0.2$, $\sigma^f = 0.3$, $\alpha = 25$, $X_0 = 1$, $T = 10$

Such a relation is clearly non-linear, and we opt for a level 2 signature truncation to better capture the relation between the asset and the exogenous signal :

$$\hat{\mathbb{Z}}^{\leq 2}_{(s,t)} = \left(1, \begin{bmatrix} t-s \\ X_t - X_s \\ f_t - f_s \end{bmatrix}\right), \Sigma^{\hat{\mathbb{Z}}}(s,t)\right)$$

where $\Sigma^{\hat{\mathbb{Z}}}(s,t)$ is a 3×3 matrix and

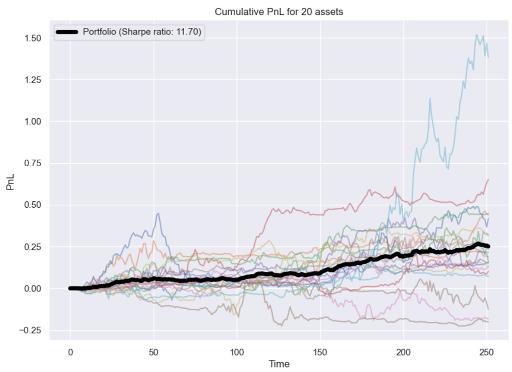
$$\Sigma^{\hat{\mathbb{Z}}}_{(i,j)}(s,t) = \int_{s<u_1<u_2<t} dZ^{(i)}_{u_1} dZ^{(j)}_{u_2}$$

In particular, the terms with $i = 3$ or $j = 3$ contains the information related to the exogenous signal, which is our only source of *alpha* when trading X. Hence, we expect higher functional values in absolute terms to properly this signal.

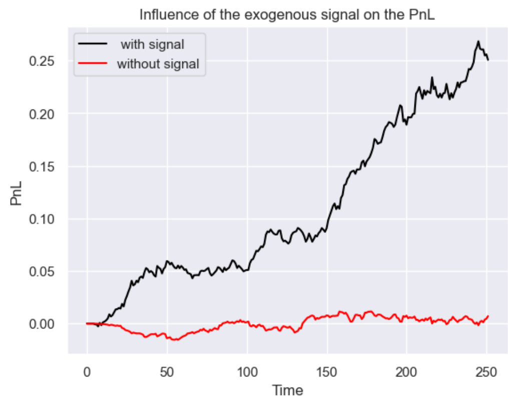
Experiment In our simulation, for $M = 10,000$ samples, daily frequency $f_{\text{daily}} = 252$ and trading over $T = 1$ year, we obtained the following functionals:

$$\left(-0.02, \begin{bmatrix} -0.26 \\ 4.20 \\ \boxed{12.7} \end{bmatrix}, \begin{bmatrix} -0.57 & 0.17 & -1.42 \\ -8.36 & -0.80 & -1.56 \\ \boxed{-28.3} & \boxed{-3.78} & \boxed{2.95} \end{bmatrix} \right)$$

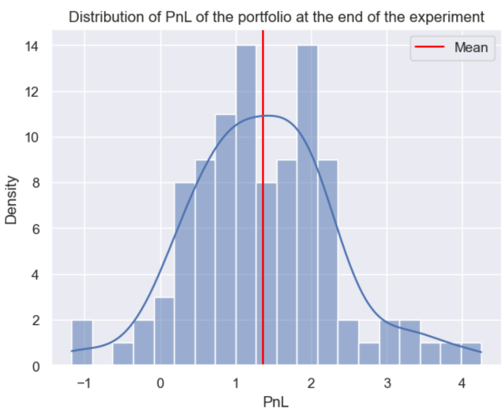
The boxed terms illustrates the explanatory power of the signal f. Then we tested our learned functionals on a 10-asset portfolio during a one year timeframe. The results clearly show a steadily growing portfolio value. If we instead train the Order 2 Sig Trader without any information ($f = 0$), we see that as expected, it does not find any signals and therefore the cumulative PnL hovers around 0, as show in 9b.



(a) Cumulative PnL for a 10-asset portfolio



(b) Influence of the exogenous signal



(c) Distribution of the daily PnL of the portfolio

Figure 9: Parameters: $\kappa = 5$, $\sigma^X = 0.2$, $\sigma^f = 0.3$, $\alpha = 25$, $X_0 = 1$, $T_{\text{trade}} = 1$

CONCLUSION

Financial time series, far from being random as claimed by the Efficient Market Hypothesis, often exhibit complex statistical properties. Trading strategies seek to capture these statistical anomalies, also known as signals or *alphas*, in order to profit from them. Yet, traditional portfolio optimization methods struggle to capture such complex signals. In contrast, Signature Trading is purely data-driven and can theoretically capture arbitrarily complex signals, given a sufficient depth N, as shown by the *Universal Approximation Theorem*. In this paper, we reproduced results from [1] and indeed we find that Sig Traders are able to learn to capture certain signals. However, our implementations revealed two drawbacks to Signature Trading, which are common to data-driven trading strategies.

1. Need for large amounts of data to train the Sig Trader: Order 1 Sig Traders required around $M = 1000$ samples to reach stable functionals, whereas Order 2 Sig Traders required at least $M = 10,000$

2. Computational bottleneck: since an Order N Sig Trader needs to compute signatures at depth $2(N + 1)$ during the fitting process and the cost of computing a signature is exponential in the depth, we could not trade at depths greater than 2. However, since we only need to compute signatures at depth N when trading, this bottleneck is not necessarily a problem in production and we could imagine firms deploying large compute power for the fitting part in depth $(2(N + 1)$ and then trading low-latency in depth N. Depth up to $N = 5$ appears feasible under this scenario, but we emphasize that the Sig Trader already performs remarkably well at depth 3.

Overall, Sig Trading remains a very promising trading scheme and anyone with enough computing power and easy access to large amounts of financial data can probably obtain solid performance from it. The affiliations of some prominent researchers in this niche area suggest some trading firms might indeed already be using Sig Trading in production.

REFERENCES

[1] Futter, Owen and Horvath, Blanka and Wiese, Magnus. *Signature Trading: A Path-Dependent Extension of the Mean-Variance Framework with Exogenous Signals* (2023).

[2] Markowitz, Harry. *Portfolio Selection* (1952).

[3] Cont, Rama. *Empirical Properties of Asset Returns: Stylized Facts and Statistical Issues* (2001).

[4] Kidger, Patrick and Lyons, Terry. *Signatory: differentiable computations of the signature and logsignature transforms, on both CPU and GPU* (2021).

[5] Borwein, Peter. *On the complexity of calculating factorials* (1985).

[6] Fermanian, Adeline. *Embedding and learning with signatures* (2021).

[7] Hambly, Ben and Lyons, Terry. *Uniqueness for the signature of a path of bounded variation and the reduced path group* (2006).

[8] Chevyrev, Ilya and Kormilitzin, Andrey. *A Primer on the Signature Method in Machine Learning* (2016).