Imperial College London

Imperial College London

Department of Mathematics

A Data Driven Approach to Market Regime Classification

Author: Conor McIndoe (CID: 00731087)

A thesis submitted for the degree of

MSc in Mathematics and Finance, 2019-2020

Declaration

The work contained in this thesis is my own work unless otherwise stated.

Abstract

We provide a novel algorithm which attempts to classify market regimes in US equities time series. As far as possible, manual intervention is avoided, preferring a data-driven approach. The path signature is utilised as a central tool; the application of which is justified. We discuss the connection between market regimes and distributions of path signatures, and provide a metric space structure on the latter which allows for a clustering to be formulated. The code both to reproduce the results and to develop further the clustering algorithms presented is provided on GitHub - *mcindoe/regime-detection*

Acknowledgements

I would like to thank my supervisor, Paul Bilokon, for his guidance through this project, as well as my parents for their continuing support.

Contents

List of Figures

List of Tables

Introduction

Financial practicioners have been familiar with the notion of market regimes for decades. Bull markets turn bear, and periods of calm may change abruptly to those of high turmoil. As the saying goes, *liquidity begets liquidity*. The dramatic market shift of early 2020, brought about by the COVID-19 pandemic, brings with it periods of high volatility and low liquidity, which are conditions comparable to the global crisis of 2008. It is said that history does not repeat itself, but it does rhyme[1]. Over the last century there have been several market crashes where a sudden loss of liquidity has been exaggerated by a compounding effect of investors selling off their positions, often targeting the same investments as one another (reduced interest in equities may lead to an increase in fixed income products for example). Explicit examples may be found in the 2015-2016 Chinese stock market crash, the Black Monday crash of 1987, or the Wall Street crash of 1929.

The ability of an investor to recognise the underlying economic and market conditions and, ideally, to estimate the transition probabilities from one market regime to another is a problem which has long sought attention. Kritzman et al. [1] utilised Markov-switching models to characterise regime and dynamically allocate across a portfolio, demonstrating improved performance against statically-assigned weights across a variety of asset classes. This work highlights the importance of regime classification as a trading signal.

Jiltsov [2] demonstrates the use of Hidden Markov Models to classify market data into one of four regimes. The feature set is derived from the S&P 500 index, the VIX index to capture volatility, TED spread[2] to capture market opinion on credit risk of large banks, and several other similar features. Jiltsov's approach is partially data-driven, in that the number of regimes is initially asserted, and the resulting classifications are analysed once fitted.

Typically, in similar literature to this paper, an attempt is made to define some regimes which we expect to find in suitable time series data, and the discussion of how best to characterise the data at hand into these categories is then provided. Nystrup et al. [3] proposed a setup with two market states: the first is one of low variance and positive mean, while the second has negative mean and an increased variance. An online classifier is presented which attempts to categorise market conditions whilst incorporating a penalty term when switching from one classification to another, resulting in a more continuous allocation of regimes across a time series. This algorithm performs well on simulated data and is less prone to misspecification compared with the maximum likelihood estimator.

In much the same way that the strongest chess algorithms used to be trained initially on grandmaster over-the-board play, and have since been surpassed by the AlphaZero algorithm [4] which learns exclusively from self-play, our opinion is that regime identification is sufficiently complex that manual specification or direction is an inheritly limited approach. We would like instead to discuss a framework which is, as much as possible, driven exclusively by the data. This paper proposes such an algorithm, attempting to uncover regimes found in the time series of US equities.

In this paper, we will make repeated use of the *signature* of a path, a tool originating in the study of rough paths [5]. The signature, despite being studied in the literature for decades, has received a lot of attention in recent years, proving itself to be the natural language in which to encode time series data in a form suitable for machine learning tasks. Chapter 1 presents some of the strong mathematical theory supporting the idea that the signature transform is a rich enough object to meaningfully identify and classify market regime. Once the theory is in place, we provide a literature review and highlight some recent developments which have utilised the path signature.

The paper revolves around considering market regimes as clusters of path signature distribu-

[1] Quote often credited to Mark Twain, although some uncertainty exists regarding this attribution

[2] TED spread is the difference between three-month LIBOR and three-month Treasury yield

tions. In Chapter 2 we recall a clustering algorithm presented by Azran and Ghahramani [6], defined on an arbitrary metric space, and shall refer to the algorithm presented in their paper as Azran-Ghahramani clustering. We look at several examples, building up from a synthetic settings where the desired output is clearly understood (and hence the quality of classification may be identified), and building intuition from which the quality of application to real market data can be discussed.

Our examples also steadily build in abstraction. The first example presented is in the familiar setting where points are elements of $\mathbb{R}^2$, and the metric space structure is provided by the Euclidean distance. The second example is that of synthetic market data, with price processes simulated by Geometric Brownian Motion. Here points in our space will be collections of path signatures (or a distribution path signatures), and a distance function will be developed from the two-sample kernel test methodology presented in Gretton et al. [7].

Chapter 3 is dedicated to the application of the clustering algorithm to market data. Here the quality of the output is assessed both in terms of the regimes described by the algorithm, and also by comparing the results to the output of previous examples.

Chapter 4 presents directions for further work. The application of path signatures as a faithful representation of market structure (see Chapter 1) is a powerful approach which is now seeing a surge of attention in the area of quantitative finance. We present some clear next steps for this work as well as some different directions in which to take the framework presented in this paper.

The contribution of this paper to the existing literature is primarily the application of path signatures and clustering algorithms to the problem of regime classification. Whilst the use of signatures to characterise financial time series is not new, and neither is the application of the maximum mean discrepancy statistic to signature spaces (as seen in [8] for example), the attempt to use this framework in a clustering algorithm to detect market regimes is, to our knowledge, a novel contribution.

We will also discuss areas in which the algorithm may be improved. Along with this writeup, we provide a Python library to perform Azran-Ghahramani clustering, which is currently not implemented. This library, and the code to reproduce examples presented in the paper, are provided on GitHub at *mcindoe/regime-detection*[3].

[3]https://github.com/mcindoe/regime-detection

Chapter 1

The Signature Transform

In this first chapter, we develop some knowledge of the path signature transform. This subject boasts a broad mathematical theory dating back at least as far as Chen's work in 1958 [5]. More recently it has seen frequent application as a tool to study rough paths, particularly in the work of Lyons (see [9], [10]), and in 2020 applications to both synthetic financial data generation and quality evaluation demonstrated in [8]. In this chapter, we will first introduce the signature of a path, and then discuss some of the applications it has seen in recent years. The method in which the signature is introduced here is inspired by the work of [11], to which the reader is referred for another entry-level introduction to to this topic.

The signature transform can be defined in a far more general setting than that considered in this paper. Whilst we will only be investigating functions which are piecewise smooth, in fact the results stated here can be shown to hold for the much larger class of continuous functions of bounded variation. Since our application will require only the signature in this simpler setting, a full treatment is beyond the present scope. By presenting only this simpler version, the definition may be presented in the setting of Riemann integrals. We refer the interested reader to [12] for a more general treatment.

Let us begin by recalling some prerequisite knowledge required to state the path signature definition.

1.1 Integration Along a Path

A *path* in $\mathbb{R}$ is a continuous map $X : [a, b] \to \mathbb{R}$. We write $X_t := X(t) \in \mathbb{R}$. We typically think of X as tracing a path from X_a to X_b as 'time' t increases from $t = a$ to $t = b$. If the map $t \mapsto X_t$ is differentiable, the integral of a function $f : \mathbb{R} \to \mathbb{R}$ along the curve X is defined by

$$\int_a^b f(X_t)\, dX_t := \int_a^b f(X_t)\dot{X}_t\, dt := \int_a^b f(X_t)\frac{dX_t}{dt}\, dt$$

We adopt the notation $\dot{X}_t$ for dX_t/dt. We may provide a slightly more general definition. We say a map $X : [a, b] \to \mathbb{R}$ is *piecewise differentiable* curve if there exists a partition $a = x_1 < x_2 < \ldots < x_n = b$ of the interval $[a, b]$ such that over each interval (x_i, x_{i+1}) the map $t \mapsto X_t$ is differentiable. We may define the integral of a function $f : \mathbb{R} \to \mathbb{R}$ over a curve X of this more general class as

$$\int_a^b f(X_t)\, dX_t := \int_{x_1}^{x_2} f(X_t)\, dX_t + \ldots + \int_{x_{n-1}}^{x_n} f(X_t)\, dX_t$$

In this paper we will see applications mainly of this more general type. Typically our paths will be piecwise linear curves which naturally arise out of interpolating market data points (discussed in section 1.4).

Let us expand this concept to the multi-dimensional setting. A *d-dimensional path* is a continuous map $X : [a, b] \to \mathbb{R}^d$. We may denote

$$X_t := X(t) = \left(X_t^1, \ldots, X_t^d\right) \in \mathbb{R}^d$$

where the $X^i : [a, b] \to \mathbb{R}$ are themselves one-dimensional paths. We call the path X^i the i^{th} *coordinate path*. For a d-dimensional path X, and a function $f : \mathbb{R} \to \mathbb{R}$, we may then define the integral of f against any of the coordinate paths.

Example 1.1.1. *Let X be the two-dimensional path $X : [0,1] \to \mathbb{R}^2, X_t = (2t+1, 2t^3)$, and let $f(x) = 3x-2$. The second coordinate path, X^2, is the map $X^2 : [0,1] \to \mathbb{R}$, $X_t^2 = 2t^3$. We compute the integral of f over a portion of the second coordinate path:*

$$\int_{1/2}^{3/4} f(t)\, dX_t^2 = \int_{1/2}^{3/4} f(t)\, \frac{dX_t^2}{dt}\, dt = \int_{1/2}^{3/4} (3t-2)(6t^2)\, dt = -\frac{23}{512}$$

The path signature is simply a collection of integrals and iterated integrals, of the above form, where the function $f(\cdot)$ is taken to be 1. To motivate an iterated integral, let X be a one-dimensional path, and note that if $f(\cdot)$ is a continuous function, then the function $t \mapsto \int_a^t f(X_s)\, ds$ is also a continuous function $[a,b] \to \mathbb{R}$, that is, a one-dimensional path. In particular, we may therefore define the integral of this path over the path X:

$$\begin{aligned}\int_a^b \int_a^t f(X_s)\, dX_s\, dX_t &:= \int_a^b \left(\int_a^t f(X_s)\, dX_s \right) dX_t \\ &= \int_a^b \left(\int_a^t f(X_s)\, dX_s \right) \dot{X}_t\, dt\end{aligned}$$

This is known as an *iterated integral*, or specifically a 2-fold iterated integral. We may continue in this fashion, repeating the process up to some level k:

Definition 1.1.1 (k-fold Iterated Integral)**.** Let $X : [a,b] \to \mathbb{R}^d$ be a d-dimensional piecewise differentiable curve. For any $i \in \{1, \dots, d\}$ we define

$$S(X)_{a,t}^i := \int_{a<s<t} dX_s^i = X_t^i - X_0^i$$

For any $i, j \in \{1, \dots, d\}$ we may then define the *2-fold iterated integral*

$$S(X)_{a,t}^{i,j} := \int_{a<r<s<t} dX_r^i\, dX_s^j := \int_a^t \int_a^s dX_r^i\, dX_s^j = \int_{a<s<t} S(X)_{a,s}^i\, dX_s^j$$

For any $k \in \mathbb{N}$, and collection of indices $i_1, \dots, i_k \in \{1, \dots, d\}$, we define

$$S(X)_{a,t}^{i_1,\dots,i_k} := \int_{a<t_1<\dots<t_k<t} dX_{t_1}^{i_1} \cdots dX_{t_k}^{i_k} = \int_{a<s<t} S(X)_{a,s}^{i_1,\dots,i_{k-1}}\, dX_s^{i_k}$$

Notice that the limits of integration for the 2-fold iterated integral of definition 1.1.1 is over the triangle

$$\mathcal{T} := \{(r,s) : 0 < s < t, 0 < r < s\} \subseteq \mathbb{R}^2 \tag{1.1.1}$$

The 3-fold iterated integral is over a tetrahedron. The generalisation of this shape to higher dimensions is known as a *simplex*.

The indices appearing in the general k-fold expression can be any collection of length k drawn from $\{1, \dots, d\}$, and in particular may have repeated elements. The set of such indices is the k-fold cartesian product $\{1, \dots, d\}^k$ which in this context is often called the set of *words* on the *alphabet* $\{1, \dots, d\}$.

Definition 1.1.2 (Words on an alphabet)**.** For $k, n \in \mathbb{N}$, we define the set of *words of length n on a set of letters* $\{l_1, \dots, l_k\}$ as the set of ordered symbols $x_1 x_2 \cdots x_n$ with each $x_i \in \{l_1, \dots, l_k\}$. In this context, the set of letters is referred to as an *alphabet*. The *set of words* on an alphabet A is the infinite collection of words of any length (including zero), and is denoted $\mathcal{W}(A)$. The *empty word* is the word of length zero, denoted ϵ, and is also considered to be an element of $\mathcal{W}(A)$.

Example 1.1.2. *The set of words of length 3 on the alphabet $A := \{a, b\}$ is the collection*

$$\{aaa, aab, aba, abb, baa, bab, bba, bbb\}$$

The set of all words on the alphabet A is the collection

$$\mathcal{W}(A) = \{\epsilon, a, b, aa, ab, ba, bb, aaa, aab, \dots\}$$

In Definition 1.1.2, by an ordered symbol we mean that, for example, the element *abc* is considered distinct from the element *acb*. We might also say that the symbols are *non-commuting*. If there is an ordering $<$ on a given alphabet A, it may be extended to an ordering on set of words $\mathcal{W}(A)$ in the following way:

Definition 1.1.3 (Lexicographic Ordering Induced by $<$)**.** Let A be a set with an ordering $<$. The *length* of a word $u = x_1 x_2 \dots x_n \in \mathcal{W}(A)$ is denoted by $|u|$ and is the number of letters, n, in the word. The concatenation of two words u and v is denoted uv, and is the word of length $|u|+|v|$ formed by appending the letters of v to the end of u. The ordering $<$ on A is extended inductively to an ordering on $\mathcal{W}(A)$ by defining that $\epsilon < u$ for any non-empty word u, and for letters $a, b \in A$, words $u, v \in \mathcal{W}(A)$, we say $au < bv$ if either $a < b$ (in A), or $a = b$ and $u < v$.

Definition 1.1.4 (The Path Signature)**.** For a piecewise-differentiable curve $X : [a, b] \to \mathbb{R}^{\mathrm{d}}$, the *path signature* $S(X)_{a,b}$ is the infinite collection of iterated integrals $S(X)_{a,b}^{i_1,\dots,i_k}$ where $i_1 \cdots i_k$ is any word on the letters $A := \{1, \dots, d\}$. We take the natural ordering on the alphabet A, and the order of statistics in the path signature is the lexicographic extension of this ordering onto the set of words. The first element is taken to be 1, the reason for which is made clear in section 1.2:

$$S(X)_{a,b} := (1, S(X)_{a,b}^1, \dots, S(X)_{a,b}^d, S(X)_{a,b}^{1,1}, S(X)_{a,b}^{1,2}, \dots)$$

The signature *up to level* n is denoted $S(X)_{a,b}^{\leq n}$ and is the collection of all terms $S(X)_{a,b}^{i_1,\cdots,i_k}$ such that $i_1, \cdots, i_k$ is a word of length at most n.

Later in this chapter we will see that the path signature is an object rich enough to nearly describe the path completely. The first Proposition, however, demonstrates that the starting point of the path is not captured. Indeed, the signature is invariant to translations of the original path.

Proposition 1.1.5 (Translation Invariance of the Path Signature)**.** *Let $X : [a, b] \to \mathbb{R}^d$ be a d-dimensional path, and consider a translation $\tilde{X} : [a, b] \to \mathbb{R}^d$, where $\tilde{X}_t := X_t + c$ for some $c \in \mathbb{R}^d$. Then for any $k \in \mathbb{N}$ and any k-length word $i_1 \cdots i_k \in \mathcal{W}(\{1, \dots, d\})$, we have*

$$S(\tilde{X})_{a,b}^{i_1,\dots,i_k} = S(X)_{a,b}^{i_1,\dots,i_k}$$

The signatures $S(\tilde{X})$ and $S(X)$ are therefore identical.

Proof. Write $c = (c^1, \dots, c^d)$ and take $i \in \{1, \dots d\}$. We have $\tilde{X}_t^i = X_t^i + c^i$, and hence $d\tilde{X}_t^i/dt = dX_t^i/dt$. We therefore have,

$$S(\tilde{X})_{a,b}^i = \int_a^b d\tilde{X}_t^i = \int_a^b \frac{d\tilde{X}_t^i}{dt}\, dt = \int_a^b \frac{dX_t^i}{dt}\, dt = S(X)_{a,b}^i$$

which shows the result for any word of length one. Now suppose the result holds for any k-length word. Let $i_1 \cdots i_k i_{k+1}$ be a $(k+1)$-length word; we have

$$S(\tilde{X})_{a,b}^{i_1,\dots,i_k,i_{k+1}} = \int_a^b S(\tilde{X})_{a,b}^{i_1,\dots,i_k} \frac{d\tilde{X}_t^{i_{k+1}}}{dt}\, dt = \int_a^b S(X)_{a,b}^{i_1,\dots,i_k} \frac{dX_t^{i_{k+1}}}{dt}\, dt = S(X)_{a,b}^{i_1,\dots,i_k,i_{k+1}}$$

So the terms of the signatures of each path agree for any given word in $\mathcal{W}(\{1, \dots, d\})$, and hence for the entire signature. □

1.2 Log Signature

The logsignature, like the signature, is a collection of statistics which together fully describe a path. The logsignature, as we will see, is a more concise representation of the information present in a signature; it therefore has seen much attention in the machine learning literature. In order to introduce the logsignature, we present a slight reformulation of the signature transform from the previous section. We introduce the vector space[1] of *non-commutative formal power series on a basis of symbols* $B = \{e_1, \dots, e_d\}$. We denote this set $\mathcal{V}$ or $\mathcal{V}_B$ if we wish to be explicit about the basis.

[1] Definitions for the algebraic structures not explicitly defined in this paper may be found in the appendices

Formally, $\mathcal{V}$ is the set of elements of the form $\omega = \sum_{i=1}^{k} \lambda_i w_i$, where $k \in \mathbb{N}$, the $\lambda_i \in \mathbb{R}$ are scalars, and w_i are words from the set $\mathcal{W}(B)$ as in Definition 1.1.2. We may equivalently define $\mathcal{V}$ as the collection of infinite sums $\sum_{w \in \mathcal{W}(B)} \lambda_w w$ over all words, but where only finitely many of the λ_w are nonzero. At present, a term λw is to be interpreted as a formal symbol, rather than a multiplication. This is what is meant by a *formal term* in our *formal power series*. This scalar multiplication is what we define next.

We give the set the structure of an $\mathbb{R}$-vector space by defining, for any $\lambda, \mu \in \mathbb{R}, w \in \mathcal{W}(B)$:

- $\lambda \cdot (\mu w) := (\lambda \cdot \mu) w$
- $\lambda w + \mu w := (\lambda + \mu) w$

and extending linearly to an addition and scalar multiplication on $\mathcal{V}$. We have for example

$$2 \cdot (3e_1 + 4e_2 e_1 + e_1 e_2) = 6e_1 + 8e_2 e_1 + 2e_1 e_2$$

Recall that the elements $e_1 e_2$ and $e_2 e_1$ are considered distinct words. This is the meaning of the term *non-commutative*; note however that the addition operator which is implicit here is considered commutative, in that $3e_1 + e_2 = e_2 + 3e_1$.

By providing also a multiplication between elements, we equip the space with the structure of an *algebra*. This operation is usually denoted $\otimes : \mathcal{V} \times \mathcal{V} \to \mathcal{V}$. We define first, for words w and v, the product $w \otimes v := wv$, where wv is the concatenation product between words of Definition 1.1.3. We may then extend this linearly to a product on the set $\mathcal{V}$. For example:

- $2e_1 \otimes (3e_2 + 5e_1 e_4) = 6e_1 e_2 + 10 e_1 e_1 e_4$
- $(3e_1 + 5e_2) \otimes (2e_4 + e_1 e_2) = 6e_1 e_4 + 3e_1 e_1 e_2 + 5e_2 e_4 + 5e_2 e_1 e_2$

The reformulation of the signature is as follows. For a d-dimensional path X, we identify the signature $S(X)_{a,b}$ with the non-commutative formal power series

$$S(X)_{a,b} = 1 + S(X)^1_{a,b} e_1 + \ldots + S(X)^d_{a,b} e_d + S(X)^{1,1}_{a,b} e_1 e_1 + S(X)^{1,2}_{a,b} e_1 e_2 + \ldots$$

over the basis elements $\{e_1, \ldots, e_d\}$, and where the coefficient of the word $e_{i_1} e_{i_2} \cdots e_{i_k}$ is the term $S(X)^{e_{i_1} \cdots e_{i_k}}_{a,b}$. The 1 appearing in the sum is the coefficient of the empty word ϵ. The use of the equals sign is a slight abuse of notation, but is justified in the sense that there is a one-to-one correspondence between d-dimensional signatures and non-commutative formal power series over the basis symbols $\{e_1, \ldots, e_d\}$.

Definition 1.2.1 (Log Signature)**.** On the space of $\mathcal{V}$, we may define the logarithm of certain power series as follows. If $\omega \in \mathcal{V}$ is a power series where the coefficient of the empty word, ϵ, is some nonzero λ_0, then $\log x$ is defined to be the following formal power series:

$$\log x := \log(\lambda_0) + \sum_{n \geq 1} \frac{(-1)^n}{n} \left(1 - \frac{x}{\lambda_0}\right)^{\otimes n} \tag{1.2.1}$$

where the power $\otimes n$ is the n^{th} power with respect to the operation $\otimes$. The logsignature of a path X, written $\log S(X)_{a,b}$, is defined as the power series obtained from taking this logarithm of the signature power series (which we recall has ϵ-coefficient equal to 1), and may be thought of either as that power series or as the corresponding infinite vector describing the coefficients.

In addition to this form of definition for the logsignature, the representation of signatures as non-commuting formal polynomials allows the statement of the following powerful theorem, known as Chen's identity. If we can write a path X as the concatenation of other path segments, and we know the signature of the segements, then Chen's identity provides a tool by which we may calculate the signature of X, and is precisely the multiplication defined in this section. First, let us define formally what is meant by the concatenation of paths:

Definition 1.2.2 (Path Concatenation)**.** Let $X : [a, b] \to \mathbb{R}^d$, $Y : [b, c] \to \mathbb{R}^d$ be two d-dimensional paths. The *concatenation* of X and Y, denoted $X * Y$, is the d-dimensional path on $[a, c]$ given by

$$(X * Y)_t = \begin{cases} X_t & \text{if } a \leq t < b \\ X_b + (Y_t - Y_b) & \text{if } b \leq t \leq c \end{cases}$$

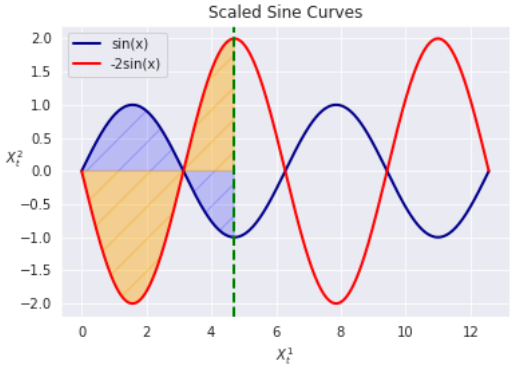

Figure 1.1: Example 1.3.1 - Scaled Sine Curves

Theorem 1.2.3 (Chen's Identity)**.** *Let X, Y be as in Definition 1.2.2. We have*

$$S(X * Y) = S(X) \otimes S(Y) \tag{1.2.2}$$

The intention of this paper will be to use the path signature as an encoding of market time series. The end of this chapter will be devoted to presenting some of the theoretical results which justify encoding paths as their signature transform, and argue that essentially no information is lost in the process. Before then, let us discuss some geometrical interpretations of this transformation to aid in the intuition.

1.3 Geometric Interpretation of the Path Signature

We have already seen in Definition 1.1.1 a geometric interpretation for level-one terms of the signature of a d-dimensional path X. We have

$$S(X)^i_{a,b} = \int_a^b dX_t^i = X_b^i - X_a^i \tag{1.3.1}$$

The level-one term $S(X)^i_{a,b}$ therefore is therefore the displacement in the i^{th} coordinate path X^i between time $t = a$ and $t = b$. Less trivially, we have that this i^{th} displacement in fact completely determines the k-fold iterated integral over the i^{th} index.

Proposition 1.3.1. *Let $X : [a, b] \to \mathbb{R}^d$ be a d-dimensional path. We have*

$$S(X)^{\overbrace{i,\dots,i}^{k \text{ times}}}_{a,b} = \frac{(X_b^i - X_a^i)^k}{k!} \tag{1.3.2}$$

for any $i \in \{1, \dots, d\}$ and any $k \in \mathbb{N}$.

Proof. Let $i \in \{1, \dots, d\}$. The $k = 1$ case is Equation 1.3.1, which is of the required form. Suppose that the result holds for some $k \in \mathbb{N}$. For the $k + 1$ case we have:

$$S(X)^{\overbrace{i,\dots,i}^{k+1 \text{ times}}}_{a,b} = \int_a^b \frac{(X_t^i - X_a^i)^k}{k!} \frac{dX_t^i}{dt}\, dt = \left[\frac{(X_t^i - X_a^i)^{k+1}}{(k+1)!}\right]_a^b = \frac{(X_b^i - X_a^i)^{k+1}}{(k+1)!}$$

□

Let us draw attention at this point to an easy mistake to make. When one thinks of a path such a price process, it is tempting to think of a one-dimensional path. Instead, typically what we are thinking of is the two-dimensional process which is the time-augmentation; that is, the process $\{x_1, \dots, x_n\}$ is recorded instead as $\{(t_1, x_1), \dots, (t_n, x_n)\}$. This is typically the path one imagines when drawing the graph of the process. We stress that the previous result does not imply that the signatures of such paths are trivial.

Example 1.3.1. *We present an example of the computation of path signature. Consider the two-dimensional path $X : [0, 2\pi] \to \mathbb{R}$, $X_t = (t, n \sin t)$, for $n \in \mathbb{R}$. We have seen that the level-1 terms are the increases in each coordinate path. Since both coordinate paths begin at 0 when $t = 0$, we have $S(X)^1_{0,r} = r$, and $S(X)^2_{0,r} = n \sin r$. Equation 1.3.2 gives the terms $S(X)^{1,1}_{0,r} = r^2/2$, and $S(X)^{2,2}_{0,r} = (n \sin r)^2/2$. We compute the remaining two terms for the signature up to level two:*

$$S(X)^{1,2}_{0,r} = \int_0^r S(X)^1_{0,s}\, dX^2_s = \int_0^r ns \cos s\, ds = n(r \sin r + \cos r - 1)$$
$$S(X)^{2,1}_{0,r} = \int_0^r S(X)^2_{0,s}\, dX^1_s = \int_0^r n \sin s\, ds = n(1 - \cos r)$$

The signature of X is therefore as follows:

$$S(X)_{0,r} = \left(1, r, n \sin r, \frac{r^2}{2}, n(r \sin r + \cos r - 1), n(1 - \cos r), \frac{(n \sin r)^2}{2}, \dots\right) \tag{1.3.3}$$

Let us consider the signature of the path X from Example 1.3.1, up to time $t = 3\pi/2$, and in particular the effect which n has on the result. The plots for two such curves are demonstrated in Figure 1.1. First, we have

$$S(X)_{0,3\pi/2} = \left(1, \frac{3\pi}{2}, -n, \frac{9\pi^2}{8}, -n\left(\frac{3\pi}{2} + 1\right), n, \frac{n^2}{2}, \dots\right) \tag{1.3.4}$$

The second and third entry are the displacements of the first and second coordinate paths respectively. Clearly the first of these is independent of n and the second is proportional to n. Shaded in Figure 1.3.1 is the (signed) area equal to the term $S(X)^{2,1}_{0,r}$, as the integral of the displacement of X^2 with respect to X^1. Since one half-period of a sine curve has an integral equal to 2, the sine curve, scaled by n, has an integral up to value $3\pi/2$ equal to n. In a similar fashion, the value of $S(X)^{1,2}_{0,r}$ is also proportional to n. The geometric interpretation of this intergral with respect to the second coordinate is a little less familiar; similar integrals are presented in the upcoming result.

Another well-known result connects second-order signature terms to the product of first-order terms. We provide an understanding of this result in the case where the curve changes direction finitely often, by which we mean that the curve is the concatenation of paths which are piecewise monotone in both coordinates.

Lemma 1.3.2. *Let $X : [a, b] \to \mathbb{R}^d$ be a d-dimensional path. We have*

$$S(X)^{i,j}_{a,b} + S(X)^{j,i}_{a,b} = S(X)^i_{a,b} S(X)^j_{a,b} \tag{1.3.5}$$

Proof. Observe that it suffices to prove the result for curves which have $X^i_0 = X^j_0 = 0$. Indeed, if we have the result for this class of curves, and X is any d-dimensional curve, then if $c \in \mathbb{R}^d$ is such that $\tilde{X} := X + c$ has $\tilde{X}^i_0 = \tilde{X}^j_0 = 0$, then by Proposition 1.1.5 (translation invariance) we have:

$$S(X)^{i,j}_{a,b} + S(X)^{j,i}_{a,b} = S(\tilde{X})^{i,j}_{a,b} + S(\tilde{X})^{j,i}_{a,b} = S(\tilde{X})^i_{a,b} S(\tilde{X})^j_{a,b} = S(X)^i_{a,b} S(X)^j_{a,b}$$

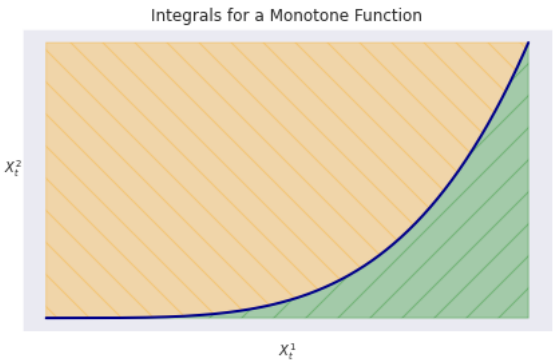

Figure 1.2: Monotone Function - shaded is integral of $X^2\, dX^1$ (green) and $X^1\, dX^2$ (yellow)

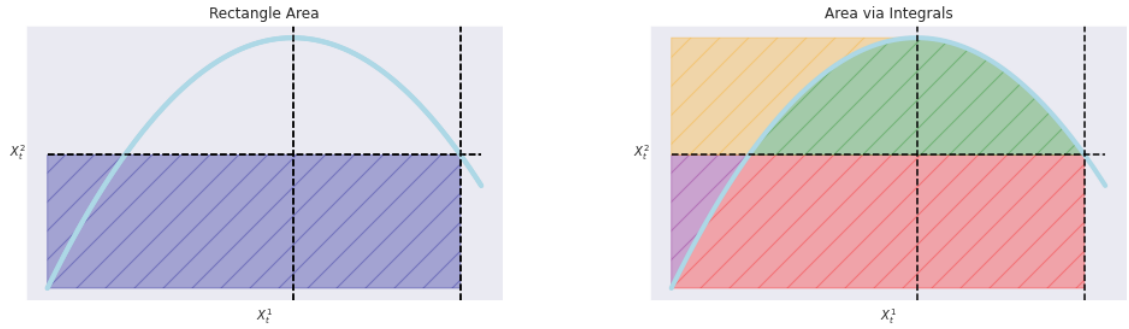

Figure 1.3: Inductive step for Lemma 1.3.2 - Two methods to count the product of the displacements, depicted as the blue rectangle

So assume without losing generality that $X_0^i = X_0^j = 0$. Referring to Figure 1.2, we see that the result is straightforward if the function is monotone. The term on the right-hand side is the area of the bounding rectangle, and the term on the left is the sum of the shaded areas, which are the integrals in both directions.

Next, suppose we can write X as the concatenation of two paths: $X = Y * Z$, where $Y : [a, t] \to \mathbb{R}^d$, $Z : [t, b] \to \mathbb{R}^d$ both satisfy Equation 1.3.5 (for example both are monotone) - we refer to Figure 1.3. Dropping the subscript a, b for brevity of notation, we may write

$$\begin{cases} S(Y)^i S(Y)^j = S(Y)^{i,j} + S(Y)^{j,i} \\ S(Z)^i S(Z)^j = S(Z)^{i,j} + S(Z)^{j,i} \end{cases} \tag{1.3.6}$$

By Chen's identity, we have $S(X) = S(Y * Z) = S(Y) \otimes S(Z)$. Recall the non-commutative formal polynomial of $S(Y), S(Z)$:

$$S(Y) = 1 + S^1(Y)e_1 + \ldots + S^d(Y)e_d + S^{1,1}(Y)e_1e_1 + S^{1,2}(Y)e_1e_2 + \ldots$$

and similarly for $S(Z)$. The coefficient of e_i in the product $S(Y) \otimes S(Z)$ is therefore seen to be $S^i(Y) + S^i(Z)$, that is $S(X)^i = S(Y * Z)^i = S(Y)^i + S(Z)^i$. Note that the geometric interpretation here is simply that the displacement in the i^{th} coordinate path in the concatenation of Y and Z is the sum of displacements in the paths Y and Z. We can also compute $S(Y * Z)^{i,j}$ in a similar fashion, obtaining

$$\begin{aligned} S(Y * Z)^i &= S(Y)^i + S(Z)^i \\ S(Y * Z)^{i,j} &= S(Y)^i S(Z)^j + S(Y)^{i,j} + S(Z)^{i,j} \end{aligned}$$

from which we have

$$\begin{aligned} S(X)^{i,j} + S(X)^{j,i} &= S(Y * Z)^{i,j} + S(Y * Z)^{j,i} \\ &= S(Y)^i S(Z)^j + S(Y)^{i,j} + S(Z)^{i,j} + S(Y)^j S(Z)^i + S(Y)^{j,i} + S(Z)^{j,i} \\ &= S(Y)^i S(Z)^j + S(Y)^i S(Y)^j + S(Y)^j S(Z)^i + S(Z)^i S(Z)^j \\ &= (S(Y)^i + S(Z)^i)(S(Y)^j + S(Z)^j) \\ &= S(Y * Z)^i S(Y * Z)^j \\ &= S(X)^i S(X)^j \end{aligned}$$

Inductively, this shows the desired result for any curve which is composed of semgents which are piecewise-monotone in both coordinates. □

A geometric interpretation of this result may be seen in Figure 1.3. The integral $S(X)^{1,2}$ for example may be considered as the sum of the signed areas over each section where the curve is monotone. The statement then reads that the signed area of the blue rectangle in the left image may be constructed as follows: take the integral of $X^2\, dX^1$, to contribute the positive green and red sections, then the integral of $X^1\, dX^2$ from zero until the maxima (with respect to the X^1 coordinate, the first black dotted line) contributes positive purple and yellow areas; the final

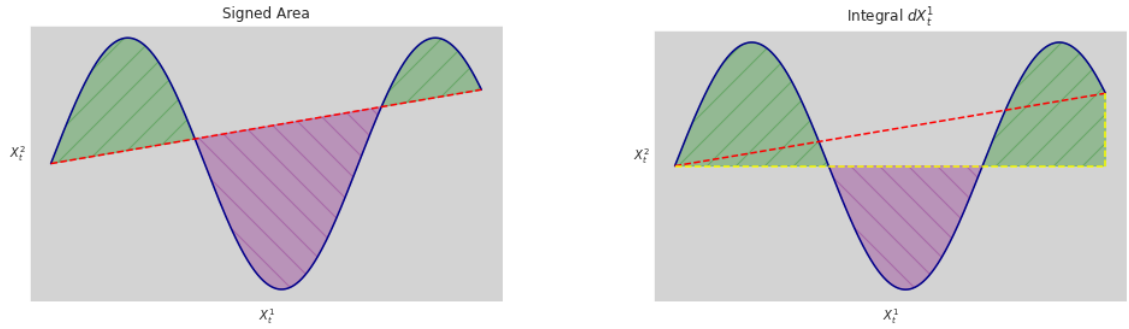

Figure 1.4: Comparison of the Levy area and usual integral

integral from the maxima to the end point (second black dotted line) contributes negative green and negative yellow areas. The resulting sum of all these signed areas is equal to the area of the blue rectangle.

Remark 1.3.3. Lemma 1.3.2 highlights that there is some redundancy in the vector representation of the signature which we have so far discussed. If the terms $S(X)^i, S(X)^j$ and $S(X)^{i,j}$ are known, then $S(X)^{j,i}$ may be inferred without an explicit record in the vector. In the same way, we have seen that the terms corresponding to words in one letter have the representation of Proposition 1.3.1; we note that only one of these terms is required for each letter in order to compute all such terms.

The logsignature, as remarked before, is a more compact representation of the information contained in the signature. Indeed, it can be shown that the logsignature is the most compact representation. Let us recall the notion of a *Lie bracket* operation $[\cdot,\cdot]$. Specifically here we refer to the Lie bracket induced by the product $\otimes$, which means that for $x, y \in \mathcal{V}$, we have $[x, y] := x \otimes y - y \otimes x$. It can be demonstrated that for any path $X : [a, b] \to \mathbb{R}^d$, we can write

$$\log S(X)_{a,b} = \sum_{k\geq 1} \sum_{i_1,\dots,i_k \in \{1,\dots,d\}} \lambda_{i_1,\dots,i_k} [e_{i_1}, [e_{i_2}, \dots, [e_{i_{k-1}}, e_{i_k}] \dots]] \tag{1.3.7}$$

A vector representation would therefore only require entries corresponding to each of the basis elements of the form which appear in Equation 1.3.7. This has considerably fewer terms (up to a given level) than the full signature representation of Definition 1.1.4. A five-dimensional path for example, up to the third level, has 155 terms in its signature and 55 in its logsignature; see [13] for formulas on the sizes of signatures. This makes the logsignature particularly enticing from a computation or machine learning point of view, not just for the reduction of required working memory, but also because the convergence time of algorithms will benefit from the removal of redundancy in the feature set.

We present another geometric interpretation of the second-order signature terms.

Proposition 1.3.4 (Geometric interpretation via the Levy Area)**.** *Let $X : [a, b] \to \mathbb{R}^2$ be a two-dimensional path. The signed area, $A_{0,u}$, between the curve (X_t^1, X_t^2) and the straight line connecting $(X_0^1, X_0^2) \to (X_u^1, X_u^2)$, for some $u \in [a, b]$, has the following form:*

$$A_{0,u} = \frac{1}{2}\left[S(X)_{0,u}^{1,2} - S(X)_{0,u}^{2,1}\right] \tag{1.3.8}$$

This quantity is known as the Levy Area.

Proof. We refer to Figure 1.4 for an illustration. As before, we may assume without this loss of generality that the starting point is $(0, 0)$ by applying a suitable translation to the curve X. Note that this does not alter the Levy area. The terms on the right-hand side are also invariant under this translation by Proposition 1.1.5.

We may compute the Levy area as follows. First, we find the area of the triangle bounded by the following three lines:

i) $X^2 = 0$

ii) $X^1 = X_u^1$

iii) The straight line connecting the start point $(0, 0)$ to the end point (X_u^1, X_u^2).

The Levy area may be computed by first computing the integral of X_t^2 with respect to X_t^1 over $t \in [0, u]$, and then subtracting the area of this triangle. Note that since $X_0^1 = X_0^2 = 0$, we have $S(X)_{0,u}^1 = X_u^1$ and $S(X)_{0,u}^2 = X_u^2$. We compute:

$$\begin{aligned} A_{0,u} = \int_0^u X_t^1 \, dX_t^2 - \frac{1}{2} X_u^1 X_u^2 &= \int_0^u S(X)_{0,t}^1 \, dX_t^2 - \frac{1}{2} S(X)_{0,u}^1 S(X)_{0,u}^2 \\ &= S(X)_{0,u}^{1,2} - \frac{1}{2} S(X)_{0,u}^1 S(X)_{0,u}^2 \\ &= S(X)_{0,u}^{1,2} - \frac{1}{2}\left[S(X)_{0,u}^{1,2} + S(X)_{0,u}^{2,1}\right] \\ &= \frac{1}{2}\left[S(X)_{0,u}^{1,2} - S(X)_{0,u}^{2,1}\right] \end{aligned}$$

□

1.4 Signature of Data Points

Suppose that, instead of a continuous path, we have observations $\{(t_1, x_1), \ldots, (t_n, x_n)\}$ of values x_i at time t_i. These points may, for example, be the closing prices for some stock S_t for every day in some observation window. In order to speak of the signature of a set of data points, we must first choose some piecewise-differentiable path representation of the data and then refer to the signature of that path.

There are several approaches which have been proposed for this purpose. We present the piecewise linear interpolation as well as the *rectilinear* or *axis* interpolation which Levin et al. have used ([14]) to define the signature of a data stream.

Definition 1.4.1 (Path Interpolations of Data Points)**.** Let $\mathcal{X} = \{(t_1, x_1), \ldots, (t_n, x_n)\}$ be a set of observations of some process. The *piecewise linear interpolation* path generated by $\mathcal{X}$ is the path which passes through the (t_i, x_{t_i}) and joins each pair $(t_i, x_{t_i}), (t_{i+1}, x_{t_{i+1}})$, $i \in \{1, \ldots, n-1\}$ with a linear segment. Formally, for each $i = 1, \ldots, n-1$, define the line $L_i : \mathbb{R} \to \mathbb{R}$ through $\{(t_i, x_i), (t_{i+1}, x_{i+1})\}$ by

$$L_i(t) := x_i + \frac{t - t_i}{t_{i+1} - t_i}(x_{i+1} - x_i)$$

then the linear interpolation generated by $\mathcal{X}$ is the map $X : [t_1, t_n] \to \mathbb{R}$ given by

$$X(t) = \sum_{i=1}^{n-1} \mathbb{1}(t \in [t_i, t_{i+1}]) \cdot L_i(t)$$

where $\mathbb{1}(\cdot)$ is the indicator function.

The *rectilinear interpolation* path generated by $\mathcal{X}$ is the path which is horizontal between subsequent data points, and with a jump discontinuity at each $\{1, \ldots, n\}$. The rectilinear interpolation is the path $X' : [t_1, t_n] \to \mathbb{R}$ given by

$$X'(t) := \sum_{i=1}^{n} x_i \cdot \mathbb{1}(t \in [t_i, t_{i+1})) + x_n \mathbb{1}(t = t_n)$$

Figure 1.5 demonstrates piecewise-linear and rectilinear interpolation of the points

$$\mathcal{X} = \{(0, 8), (2, 5), (3, 12), (6, 14)\}$$

In this paper, the piecewise linear interpolation is used to generate piecewise-differentiable paths.

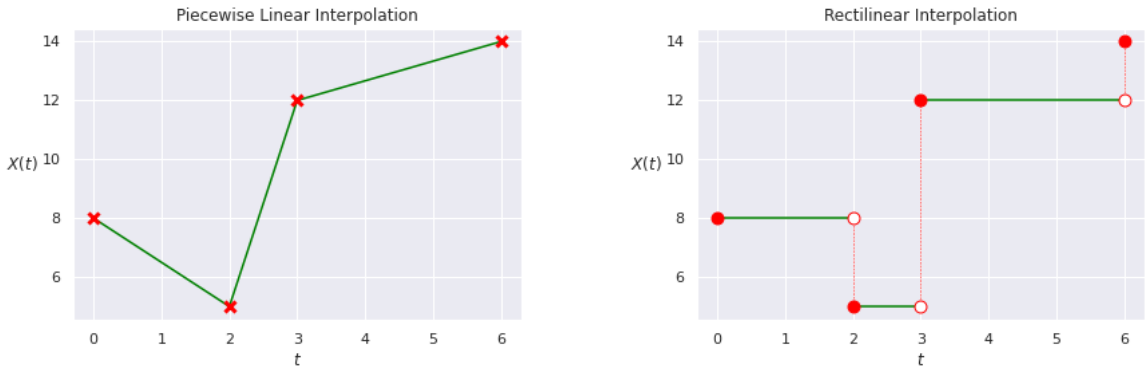

Figure 1.5: Linear and Rectilinear Interpolation

1.5 Motivation for the Path Signature

Having now presented the path signature definition and some geometrical interpretations, we now present some of the strong theoretical results which support this tool's recent surge in popularity. This section is provided both to round off the theoretical portion of this paper with some of these more technical results, a formal discussion of which is beyond present scope, and to discuss some recent applications which have demonstrated the ability of this representation to tackle a wide range of challenges.

The path signature has seen a significant increase in attention in recent years, despite being studied in the literature for decades. It lends itself naturally as a tool to the machine learning practitioner, with its ability to represent, with a series of numbers, the structure of an underlying process such as a price process in a *faithful* manner. The sense in which this embedding is faithful, by which we mean that different paths produce different signatures, is tricky to make precise. We have seen already that the signature is invariant under translations, and therefore different paths may produce the same signature. The signature is also invariant to time reparameterisations; informally we might say that the signature depends only on the curve and the direction in which it was drawn, rather than the speed in which it was drawn. Lyons et al. make this precise by defining an equivalence relation on signatures known as a *tree-like equivalence* ([15]). It says that two paths X and Y are equivalent if the path formed from travelling first along X, and then *backwards* along Y, is indistinguishable in signature from the constant path. The main result is that the signature of two paths coincide if and only if they are equivalent with respect to this restrictive equivalence relation[2].

The previous result may be read that only a very small amount of information is lost when taking the signature transform of a path, in the sense that precision about the generating path is traded for precision about the tree-like equivalence class of the generating path. A further result provided by Lyons et al. ([15]) is that not much information is lost either in considering the signature up to some given level compared to the entire signature. Specifically, the error experienced when approximating the solution to a differential equation *driven by a path* X by a level-k truncation of a path rather than the full signature decays factorially[3] with k. That is to say, the first terms in the signature are the most descriptive.

These two results together suggest something very powerful about the use of the signature transform in applied tasks where the structure of the underlying process is of central importance. In a natural way, we have an encoding of the structure of the path into a series of numbers, naturally included in a wide range of machine learning tasks, with very little loss of information that one had working with the full path, even though one is (clearly) forced to truncate the signature at some level.

[2]The interested reader may consult section 2.2.6 of [15] for a more complete explanation of this result
[3]This is Proposition 2.2 of [15]

1.6 Recent Developments

With the huge increase in attention deep learning has received in the past decade, due in no small way to the recent increases in computing power, GPU support, big data access and other advances, it is natural to wish to include the signature as the feature set in a neural network. Bonnier, et al. in their paper *Deep Signature Transforms* [16] have proposed a flexible methodology for doing this, which extends beyond using the truncated signature as a feature set. Instead, the authors propose a learnable, differentiable selection layer which is then incorporated into the neural network architecture and applied to a variety of problems. The authors demonstrate state of the art performance in a diverse range of tasks including non-supervised generative models for stochastic processes, supervised learning with fractional Brownian motion, and a deep reinforcment learning task. The algorithm is state of the art not just in performance, but in utilising approximately an order of magnitude less memory than the best-competing models.

This work may now be implemented in the Python package *Signatory*, which is the work of Kidger et al. [17]. The module provides a Python wrapper to a C++ implementation of signature transformations on both the CPU and GPU, making the computation of signatures for various applications very straightforward. The module boasts what is currently the fastest implementation of signatures, even before GPU acceleration, and demonstrates improvements over alternative implementations such as *iisignature* ([18]). These improvements are due to recent algorithmic advances in this space, and are detailed in the 2020 paper [17]. The applications we will see in this paper utilise this module for signature computations.

The path signature has also seen recent implementation in the work of Bühler et al. in their 2020 paper [8]. The intended application is the generation of simulated market paths. As noted by Assefa et al. ([19]), the problem of assesing the quality of synthetic data is a particularly challenging one. Typically, only qualititative observations such as the presence of certain *stylized facts* are used in place of some numerical value, for which a natural definition is hard to provide. The contributions in [8] involve not only the application of path signatures to generation of time series via variational autoencoders, but also to providing a performance evaluation metric based on the path signature and maximum mean discrepancy statistic, which we will define in the next chapter.

Chapter 2

Data-Driven Clustering

Having spent the previous chapter developing the theory and motivation to use the signature as our central tool in regime detection, we now work towards the data-driven clustering algorithm which will be applied in the next chapter to our data set of US equities. This task is interpreted in this paper as a clustering task on the space of distributions of signatures. We discuss first a sufficiently general clustering algorithm.

The work of Azran and Ghahramani [6] will be central. In [6], a data-driven clustering algorithm is presented over an arbitrary metric space. The role of this chapter is first to present this algorithm, providing some intuition regarding its outputs, and then to discuss what metric space structure is suitable in our present setting. Motivated by the previous chapter, we will be representing the time series paths as their signature transforms. Note, however, that a natural distance between signatures is not immediately clear.

The algorithm discussed is data-driven, in the sense that no underlying specification of the clusters is assumed. In particular, no preset number of clusters is specified, no minimum or maximum number of elements in each cluster and no specification of cluster shape. The task is to learn this information directly from the data.

This chapter presents several examples of Azran-Ghahramani clustering. The examples are introduced in a framework in which *correct* clusterings do exist (or perhaps multiple options are available). We stress that this is only by construction so as to provide meaningful data on which to examine the algorithm, and investigate its output. None of this information is passed to the algorithm itself.

The first example is that of *Gaussian Clouds*, which will be clouds of points in two dimensions, defined about a centre point with some random noise in each direction generated from independent Normal distributions. We can control both the standard deviation of this random component and the separation of the cluster centres in order to vary the difficulty of the task. This example demonstrates the output of the Azran-Ghahramani algorithm in a setting with familiar notions of points and distances.

The second example looks at the clustering of time series. We will construct synthetic market paths, following a Geometric Brownian Motion, with specified mean and variance. This mean and variance is our specification of a *regime*. The metric space structure here will be similar to the final application of this paper to US equities. To introduce this distance, we turn to the *maximum mean discrepancy*, introduced in [7]. This example is similar in flavour to the final task of this paper, but still in a setting where the correctness of the algorithm may be meaningfully discussed.

2.1 Preliminaries - Metric Spaces

Since this chapter relies heavily on the notion of a metric space, we briefly recall this here.

Definition 2.1.1 (Metric space)**.** A *metric space* is a set X together with a binary operator $d : X \times X \to \mathbb{R}$ such that for any $x, y, z \in X$ the following hold:

i) $d(x, y) = 0 \Leftrightarrow x = y$

ii) $d(x, y) = d(y, x)$ (symmetry)

iii) $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality)

We may either write a metric space as (X, d), or simply X when the metric is implicit.

The classical example of a metric space is that of Eucliedean n-space. This is the metric space we will be using in the *Gaussian Clouds* example.

Example 2.1.1 (Euclidean n-space)**.** *On the set of real numbers $\mathbb{R}$ we have a familiar notion of distance between two points a and b, and denoted $|a-b|$. On $\mathbb{R}^n$, $n \in \mathbb{N}$, we have the following well-known extension (typically referred to as the 'Euclidean distance'): for two points $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$, we define the distance between x and y, denoted $\|x - y\|$, by*

$$\|x - y\| := \left[\sum_{i=1}^{n} (x_i - y_i)^2\right]^{1/2}$$

$\mathbb{R}^n$ may then be equipped with a metric space structure by defining $d(x, y) = \|x - y\|$, for which the axioms of Definition 2.1.1 are readily verified.

Example 2.1.2 (Function Space)**.** *We provide a more abstract example. Let $\mathcal{F}$ be the set of all continuous functions of the form $f : [0, 1] \to \mathbb{R}$. We can equip this set with a metric space structure by defining the following distance between two elements f and g:*

$$d(f, g) := \int_0^1 |f(x) - g(x)| \, dx$$

The first axiom of Definition 2.1.1 follows since this distance is zero exactly when the function $x \mapsto |f(x) - g(x)|$ is almost everywhere zero, which implies that almost everywhere in $[0, 1]$ we have $f(x) = g(x)$. By continuity, we get $f = g$ on $[0, 1]$. The second axiom is immediate, and the third follows from the observation that for any $a, b \in \mathbb{R}$ we have $|a+b| \leq |a| + |b|$. Indeed, for $f, g, h \in \mathcal{F}$, we have

$$\begin{aligned} d(f, h) = \int_0^1 |f(x) - h(x)| \, dx &= \int_0^1 |f(x) - g(x) + g(x) - h(x)| \, dx \\ &\leq \int_0^1 \Big(|f(x) - g(x)| + |g(x) - h(x)| \Big) \, dx \\ &= \int_0^1 |f(x) - g(x)| \, dx + \int_0^1 |g(x) - h(x)| \, dx \\ &= d(f, g) + d(g, h) \end{aligned}$$

The elements of a metric space are referred to as *points*; the functions f, g and h of Example 2.1.2 would be referred to as points of the metric space $\mathcal{F}$.

2.2 Azran-Ghahramani Clustering

We now introduce the data-driven clustering algorithm first presented in [6]. We will refer to this as *Azran-Ghahramani clustering*. We begin with a metric space of points $S = \{s_1, \ldots, s_n\}$ to be clustered, and distance $d(\cdot, \cdot)$ between points.

Conceptually, a good clustering of points involves grouping together *similar* points in some sense. In this setting, the similarity is determined from the distance. We provide, in addition to the distance function, a *similarity function* w, which is a map from distances to similarities between points. The understanding is that as the distance between two points increase, their similarity should decrease. We therefore assert that our choice of w should be limited to functions that are monotonically decreasing. We also take w to be non-negative, so that the minimum similarity is bounded below by zero.

The Azran-Ghahramani algorithm considers particles moving randomly between the points of the space, with the transitional probabilities given in terms of the similarities discussed above. Similar points are perceived as having strong connections or paths between them, and points far apart have weak connections. We consider the motion of n particles, labelled $x_1, \ldots, x_n$, which each begin at the similarly labelled point in the metric space. That is, the particle x_i begins at the point s_i. We allow these particles to move through the space for some t steps, and will see how the

Algorithm 1: K-Prototypes Algorithm

Input: Transition matrix $\Omega \in \mathbb{R}^{n\times n}$, number of clusters K, initial matrix $Q \in \mathbb{R}^{k\times n}$ of prototypes

Initialisation: $Q^{(old)} := Q$

Output: Partition $\mathcal{I}$ of the indexes $\{1, \ldots, n\}$

1. $\mathcal{I}_k^{(\text{new})} := \left\{ m : k = \text{argmin}_{k\in\{1,\ldots,K\}} KL\big(\Omega_m \,\big\|\, Q_k^{(\text{old})}\big) \right\}$
2. For $k \in \{1, \ldots, K\}$, define
$$Q_k^{(\text{new})} := \frac{1}{\big|\mathcal{I}_k^{(\text{new})}\big|} \sum_{m\in\mathcal{I}_k^{(\text{new})}} \Omega_m$$
3. If convereged or stop condition has been met, return the partition $\mathcal{I}$. Otherwise set $Q^{(\text{old})} := Q^{(\text{new})}$ and return to set 1.

probability distribution of the particles' final positions reveal underlying structure and clusters in the space.

Suppose that we have a dataset which can be well-separated into clusters. By this, we mean that there exists a partition of the dataset such that, in each collection, the pairwise similarities between points of the same collection are high, and low between points from different collections. In terms of the random walk, this translates to having a small probability of leaving a given cluster once entered. For such a dataset, the underlying clustering may then be detected by stable equilibria in the random walk.

The choice of similarity function w has a significant impact. Natural candidates may include, for example, the inverse function $w(x) = 1/x$ and the squared inverse function $w'(x) = 1/x^2$ (so long as the similarity of a point with itself is defined separately). We can see that, in choosing the latter, inferred similarity of two points drops off more quickly as distance increases. This will affect the output by preferring smaller, tighter clusters of points. This concept will be explicitly addressed in our second Gaussian Clouds example later in this chapter.

It remains to specify the probability of moving from some point s_i in the space to a point s_j. Let us first clarify one aspect of the similarity between points. For distinct points s_i, s_j, the similarity is given by the composition $(w \circ d)(s_i, s_j)$. In [6], the suggestion is that the similarity of a point with itself should be set manually to zero, to encourage the particles to explore the space. Since we will soon be discussing the ratios of similarities, we instead opt to take the similarity to be 'small'. Since the absolute value of these similarities will vary between problems and choice of similarity functions, the chosen size is the minimum similarity of all the nonzero distances in the space. This corresponds to the similarity of the largest distance observed between any two points in the space.

To define the probability of a particle moving from s_i to s_j, we consider the ratio of the similarity $(w \circ d)(s_i, s_j)$ to the sum of all such similarities. Let $w_{ij} = (w \circ d)(s_i, s_j)$ be the similarity of points s_i and s_j; we take W to be the $n \times n$ matrix with (i, j)-entry equal to w_{ij}. The transition matrix P is defined by scaling the entries of W such that each row sums to 1. The (i, j)-entry defines the probability of moving from point i to point j. We have

$$P = D^{-1}W \tag{2.2.1}$$

where D is the $n \times n$ diagonal matrix of row sums, $D_{ii} = \sum_{j=1}^{n} w_{ij}$, and $D_{ij} = 0$ for $i \neq j$.

The intention is to find structure in the space by investigating the distributions of these particles after some t steps. Let $x_i(a) \in \mathbb{R}^n$ be the distribution of particle x_i after a steps. After zero steps, the particle is at its starting location of point s_i with probability one, so that the probability distribution of its location is given by the vector $x_i(0) = e_i \in \mathbb{R}^n$ with a one in the i^{th} entry and zeros elsewhere. For $a \geq 1$ we have:

$$x_i(a) = Px_i(a-1) = \ldots = P^a x_i(0) = P^a e_i \tag{2.2.2}$$

The probability distribution of particle i after a steps is therefore given by the i^{th} row of the matrix P^a. If a good clustering exists, it may then be inferred from these rows. As previously

discussed, the particles in a well-clustered space are expected to remain in their present cluster. We expect therefore that, after a sufficient number of steps, the distribution of particles which begin in these segregated clusters should be similar; in particular they will have high probability of having remained in the cluster in which they began and a low probability of being elsewhere. We therefore expect the corresponding rows of P^t to be similar. The clustering of the points in the metric space is in this way reduced to a clustering of rows of the matrix P^t.

We have the following lemmas from [6] which help to make this precise:

Lemma 2.2.1 (Properties of P)**.** *Let W, P be as in Equation 2.2.1. If W is full rank, then so is P. Further, let λ_k, $k = 1, \ldots, n$ be the ordered eigenvalues of P, such that $\lambda_k \geq \lambda_{k+1}$ for each $k = 1, \ldots, n-1$. Then every eigenvalue is real, $\lambda_1 = 1$, and $|\lambda_k| \leq 1$ for every $k = 1, \ldots, n$.*

Lemma 2.2.2 (Structure of P^t)**.** *Let W, P, D be as in Equation 2.2.1, and let the eigenvalues λ_k be as in Lemma 2.2.1, ordered so that $\lambda_k \geq \lambda_{k+1}$ as before. Let v_k, $k = 1, \ldots, n$ be the corresponding eigenvectors, chosen to have unit norm[1]. Then, for any $t = 1, 2, \ldots$ we have*

$$P^t = \sum_{k=1}^{n} \lambda_k^t \frac{v_k v_k^T D}{v_k^T D v_k} \tag{2.2.3}$$

The set of matrices $\left\{ \frac{v_k v_k^T D}{v_k^T D v_k} \right\}_{k=1}^{n}$ are idempotent, orthogonal, and form a basis of the vector space generated by $\{P, P^2, P^3, \ldots\}$. That is, if A_i, A_j are elements of this basis with $i \neq j$, then $A_i A_j = 0$, the zero matrix in $\mathbb{R}^{n \times n}$ (orthogonality), and $A_i^2 = A_i$ (idempotency).

The proofs of these lemmas may be found in [6]. Since the eigenvalues are, in absolute value, bounded above by 1, we see in Equation 2.2.3 that eigenvalues closer to 1 correspond to more stable basis elements, whereas small eigenvalues quickly shrink to zero and contribute negligibly to the sum.

In the special case where there are K separated clusters, with no connections between points in different clusters (i.e. zero similarity between such points), it can be shown that the first K eigenvalues are all 1, and $\lambda_k < 1$ for $k > K$. From Equation 2.2.3, this implies

$$\lim_{t \to \infty} P^t = \sum_{k=1}^{K} \frac{v_k v_k^T D}{v_k^T D v_k} \tag{2.2.4}$$

Clustering the initial space into k clusters corresponds to dividing the basis elements up into k elements which are deemed stable; the remaining $n - k$ are deemed unstable. For a given number of steps t, the quantity $\Delta_k(t) := \lambda_k^t - \lambda_{k+1}^t$ captures the separation of the first k eigenvalues from the remaining eigenvalues by t steps. For a target number of clusters, k, the number t_k of steps which *best reveals* k clusters is the value of t which maximises this quantity.

Definition 2.2.3. We say that t_k is the number of steps which *best reveals k clusters* in the underlying data if

$$t_k = \operatorname*{argmax}_t \Delta_k(t) := \operatorname*{argmax}_t (\lambda_k^t - \lambda_{k+1}^t)$$

In total, there are n eigenvalues, since the transition matrix P is assumed to be full rank. The number of clusters could then be anything in $\{1, \ldots, n\}$. We wish to be somewhat selective in the values $k \in \{1, \ldots, n\}$ for which we provide a k-clustering[2] however. We say a number of clusters k' is *better revealed by t steps* than a number of clusters k if $\Delta_{k'}(t) > \Delta_k(t)$. For each k, we can find the number of steps t_k which best reveals k clusters. If there is another value $k' \neq k$ in $\{1, \ldots n\}$ which is better revealed by the number of steps t_k, then a k-clustering of the data is not considered.

The maximum over all $k \in \{1, \ldots, n\}$ of the quantity $\Delta_k(t)$ is the object from which one identifies the suitability of a given number of steps t in identifying clusters in the data. We refer to this as the *maximal eigengap separation* for t steps.

Definition 2.2.4.

$$\Delta(t) := \max_{k \in \{1, \ldots, n\}} \Delta_k(t)$$

[1] Recall that (non-zero) scalar multiples of an eigenvector are also eigenvectors corresponding to the same eigenvalue. In choosing representative eigenvectors, we may therefore select them to have unit norm.

[2] By a k-clustering, we mean a partition of points $\{s_1, \ldots, s_n\}$ into k subsets

Algorithm 2: Multiscale K-Prototypes Algorithm

Input: Metric space (S, d), similarity function w, maximal number of steps T.
Output: Collection $\{(\mathcal{I}_{k_1}, \Delta(t_{k_1})), \ldots, (\mathcal{I}_{k_m}, \Delta(t_{k_m})\})$ of k_i-partitions and eigengap separations for k_i clusters.

1. Compute P according to 2.2.1
2. Compute $\Delta(t)$ for $t \in \{1, \ldots, T\}$. Find the set of local maxima $\mathcal{T} = \{t_1, \ldots, t_m\}$
3. For each $t_i \in \mathcal{T}$, find the number of clusters k_i best revealed by t_i steps; record maximal eigengap separation for t_i steps
4. For each k_i, compute the corresponding k_i-partitioning $\mathcal{I}_{k_i}$. Append the pair $(\mathcal{I}_{k_i}, \Delta(t_{k_i}))$ to the returned list.
5. Return the final collection $\{(\mathcal{I}_{k_1}, \Delta(t_{k_1})), \ldots, (\mathcal{I}_{k_m}, \Delta(t_{k_m}))\}$

For each possible number of clusters k, we may compute the number of steps t_k which best reveals k clusters. This corresponds to a local maxima of the function $\Delta_k(t)$. This number of steps t_k is of interest if, amongst all $k' \in \{1, \ldots, n\}$, k is the number of clusters best revealed by t_k steps. This (typically[3]) will correspond to a local maxima of the function $\Delta(t)$.

Our approach then is to compute local maxima $\mathcal{T}$ of $\Delta(t)$, for each $t \in \mathcal{T}$ compute the number of clusters k best revealed by t, and then to return a k-clustering of the space inferred from the rows of P^t. In order to determine these k-clusterings, we make use of an algorithm which is similar to k-means clustering, called the k-prototypes algorithm. In k-means clustering, a distance between vectors is used to separate points into k clusters. Here we have distributions, and hence a slightly different approach is suggested in [6], making use of the Kullback-Leibler divergence which we now recall.

Definition 2.2.5 (Kullback-Leibler divergence)**.** The *Kullback-Leibler divergence*, usually written as *KL-divergence*, measures the difference between two probability distributions P and Q defined on some set $\mathcal{X}$. When $\mathcal{X}$ is discrete, it is defined by

$$KL(P \parallel Q) := \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)} \right)$$

Notice that $P(x) = Q(x)$ exactly when $P = Q$, although $KL(P \parallel Q) \neq KL(Q \parallel P)$ in general. The KL-divergence is directional; $KL(P \parallel Q)$ is interpreted as the expected value of the difference in $\log P(x)$ and $\log Q(x)$, where the expectation is taken with respect to the measure P. In particular, $KL(\cdot \parallel \cdot)$ is not a distance in the parlance of metric spaces. It does nonetheless give a notion of disparity between two probability distributions, and hence is used in the k-prototypes algorithm of [6] to compare the distribution of particles' location after t steps.

The word *prototype* here is borrowed from the paper of Azran and Ghahramani, and refers to a vector in $\mathbb{R}^n$ representing a distribution in the same way P and Q do in the discussion above. One must specify k prototypes of this form in the initialisation step, about which partitions are formed, in a similar fashion to specifying centres of clusters in the k-means algorithm.

For a fixed prototypes matrix Q, we may compute, for any m, the KL-divergence of the m^{th} row of the transition matrix to each of the different prototypes. By the *closest* prototype, we mean the prototype which has minimal KL-divergence to the m^{th} row. If the m^{th} row is deemed closest to the k^{th} prototype, then index m is recorded in the k^{th} bin of the partition $\mathcal{I}$.

In the second step, the k^{th} prototype is updated to the mean of the distributions corresponding to all indexes in the k^{th} bin of the current partition, and the algorithm is run again until convergence. If the suggested partition does not change for a given iteration, then the algorithm has converged. We may also set some upper limit on the number of iterations to guarantee termination of the algorithm.

The selection of the initial matrix of prototypes has a significant impact on the success of the algorithm. Similar to k-means clustering, if the initialisation is improper then the algorithm may,

[3]One can imagine a situation in which $t_k + 1$ better reveals some other k' number of clusters and hence the maxima of $\Delta_k(t)$ is not a maxima of $\Delta(t)$

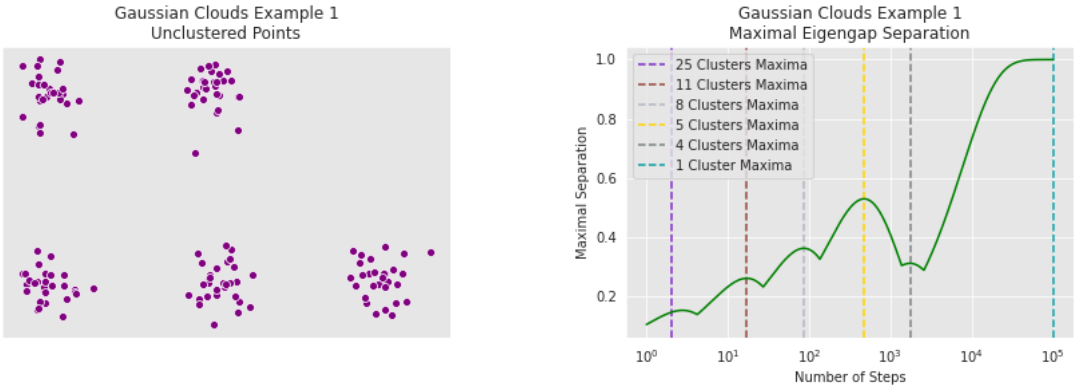

Figure 2.1: Gaussian Clouds Ex. 1. Unclustered points and maximal eigengap separation

for example, output a partition with some empty components. To avoid this, the authors suggest a *star-shaped initialisation* of the prototypes. This algorithm is specified in Algorithm 3 of the appendices. The full Azran-Ghahramani algorithm, referred to as the *Multiscale K-Prototypes Algorithm* in [6], is presented in Algorithm 2.

2.3 Gaussian Clouds

We now turn to applications of the Azran-Ghahramani algorithm. We begin with a simple example in which the points are elements of $\mathbb{R}^2$, and the notions of clusters and distance are straightforward.

Definition 2.3.1. For $x, y \in \mathbb{R}$, positive $\sigma \in \mathbb{R}^+$, and $n \in \mathbb{N}$ we define a *Gaussian Cloud of size* n, *with centre* (a, b) *and standard deviation* σ, denoted $\mathcal{X}^{\sigma}_{a,b}(n)$, to be a collection of n elements $\{a_1, \ldots, a_n\}$ where each $a_i = (x_i, y_i) \in \mathbb{R}^2$ is distibuted according to the following two-dimensional Normal distribution:

$$a_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \backsim \mathcal{N}\left(\begin{bmatrix} a \\ b \end{bmatrix}, \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix} \right)$$

We will see two examples in this Gaussian Clouds setting. The first example will have a straightforward clustering; in the second example more than one clustering is sensible. For each example, we choose $k = 5$ cluster centres in generating the points, and take 30 points per cluster. The first example uses Gaussian noise with a low standard deviation, which results in easily separable clusters. The second example features a higher standard deviation, and as such the resulting clusters are not so easy to define. To be clear, the metric space here is the set $\mathbb{R}^2$ equipped with the Euclidean distance of Example 2.1.1. For the first example, we choose the similarity function $w(x) = 1/x^3$, which was found to produce suitable clusters.

The right-hand image of Figure 2.1 is the graph of the function $\Delta(t)$, the maximal eigengap separation under t steps. Note that as t increases, the number of clusters which is best revealed

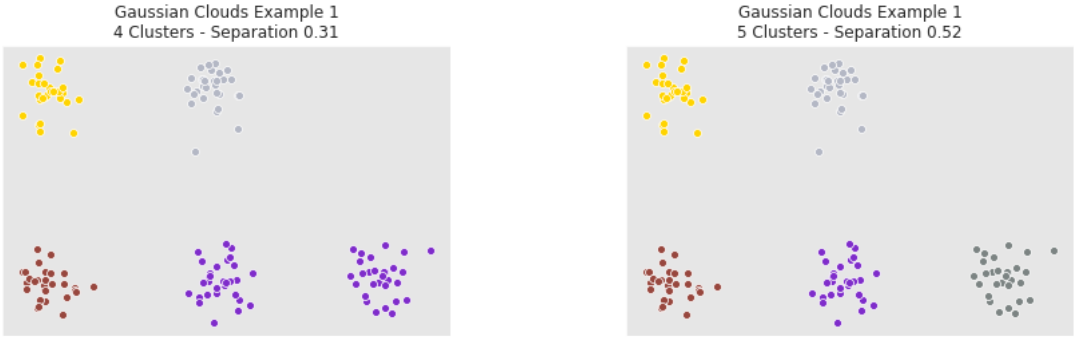

Figure 2.2: Best clustering for four clusters (left), and for five clusters (right)

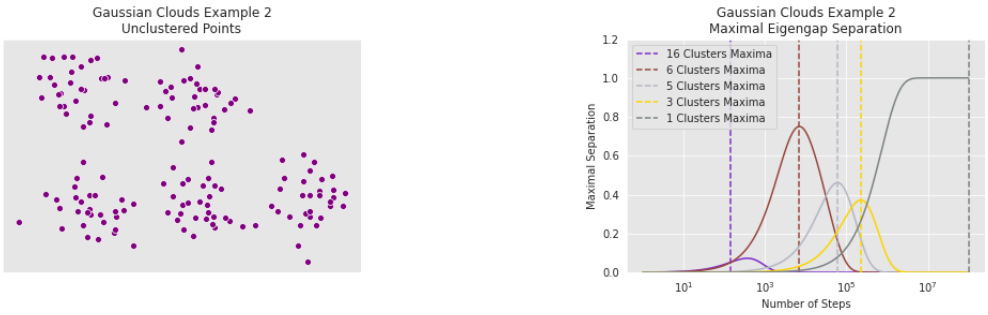

Figure 2.3: Gaussian Clouds Ex. 2. Unclustered points and maximal eigengap separation

by t steps decreases. This is the result of the upper bound $|\lambda| \leq 1$ for every eigenvalue λ. As t increases, the number of basis elements corresponding to eigenvalues which 'survive' t steps (i.e. the t^{th} power has not become negligibly small) decreases. Since all the examples we consider are fully connected, with no two points having zero similarity, eventually the trivial 1-clustering is suggested. We choose to ignore this output throughout.

The non-trivial clustering with the highest eigengap separation is the 5-clustering which reflects the problem's design (right-hand plot of Figure 2.2). The other local maxima correspond to other suggested clusterings of the data; depicted on the left of Figure 2.2 is the suggested 4-clustering. We note that the 4-clustering suggested is somewhat reasonable. In general, we believe that clusterings other than the suggested (highest-eigengap) clustering should also be investigated in situations where the correct clustering is not clear.

2.3.1 Gaussian Clouds of High Standard Deviation

The time series of real market data, however represented, is not expected to be cleanly separable into distinct clusters. For the second example, we consider a similar setting in which the noise has a notably increased standard deviation. The corresponding points are presented in the left-hand plot of Figure 2.4. We take the same metric space structure as before; we alter the similarity function from the previous example, now taking the similarity suggested in [6], which is given by

$$w^{\sigma}(x) := \exp\left(-\frac{x}{\sigma^2}\right) \tag{2.3.1}$$

As suggested in [6], we take a value of σ smaller than 1% of the nonzero distances in the space. The maximal eigengap plot is presented on the right of Figure 2.4. Here, instead of plotting the maximum $\Delta(t)$, we plot the constituent $\Delta_k(t)$ curves for the values of k for which a k-clustering is recommended by Algorithm 2.

Comparing this maximal eigengap separation plot to that of the previous example, we note first that the number of steps required to reveal stable equilibria is around two orders of magnitude higher. In particular, the number of steps before the degenerate 1-clustering is suggested is around 10^7 instead of 10^5 steps; the 5-clustering is suggested in the first example after $10^{2.5}$ steps, here around $10^{4.8}$ steps are required. The interpretation is that, in the second example, the connections between points of distinct clusters (as measured by the similarity and probability of a particle transitioning from one to the other) is significantly stronger than in the first case. In light of Equation 2.2.3, this corresponds to basis elements with higher absolute eigenvalues, which survive for a larger number of steps before contributing only neglibly to the sum.

In this example, the preferred clustering is the 6-clustering on the right of Figure 2.4, rather than a 5-clustering as in the problem specification. Here, the point generated on the bottom-left of the image, which is the point furthest from its nearest neighbour, is sufficiently far from the other points as to warrent its own cluster.

2.3.2 Stability of the Output

Let us use this example to investigate the impact of the algorithm to the problem specificiation. In particular, we are interested in quantifying how stable the output is to the following two changes:

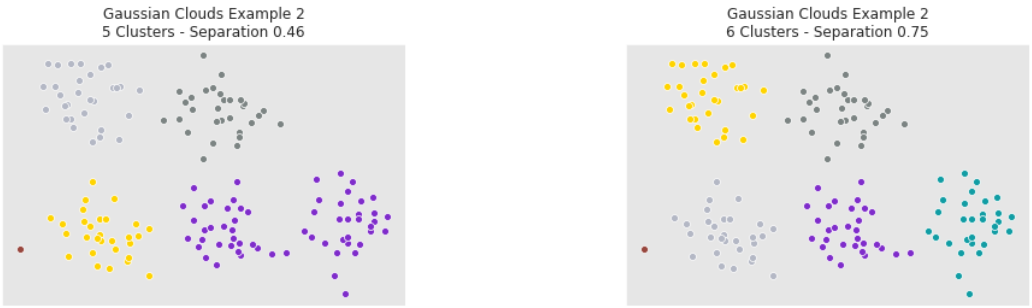

Figure 2.4: Best clustering for two clusters (left), and for six clusters (right)

i) Removal of the isolated point which is depicted as a singleton cluster in both of the clusterings of Figure 2.4

ii) Altering the similarity function

We may answer the first question by executing the clustering algorithm against the same subset of points with the outlier point removed. The corresponding maximal eigengap plot is presented in Figure 2.5. Note that an almost identical plot is produced. Indeed, the clusterings suggested are identical to the clusterings suggested in the unaltered example, now with the singleton cluster removed. The number of steps required to identify each cluster is also similar. The 5-clustering is best-revealed by 7142 steps when the outlier point is removed, compared to 6915 steps to reveal the corresponding 6-clustering in the unaltered example.

The largest seven eigenvalues of the transition matrix resulting from the unaltered (original) framework are presented in Table 2.1, along with the eigenvalues of the transition matrix once the outlier point has been removed. We demonstrate numerically the eigengap separation after 6915 steps, which corresponds, by Equation 2.2.3, to raising each eigenvalue to the power 6915. To compare the eigenvalues of the altered transition matrix, we raise the equivalent eigenvalues to the same power rather than the adjusted number of steps best revealing the corresponding cluster. Of particular interest is that, in the third column, the first six eigenvalues have a clear separation from the remaining eigenvalues, whereas in the fourth coumn the corresponding gap is after five eigenvalues. The result is that the clustering algorithm suggests a 5-clustering instead of a 6-clustering.

For the second question, we present the impact of altering the similarity function in Figure 2.6. We consider setting the value of sigma at the 0.5^{th} and 0.1^{st} percentiles in the similarity function of Equation 2.3.1. The left eigengap plot of Figure 2.6 is the clustering suggested when the 0.1-percentile value is chosen (resulting in $\sigma \approx 0.417$). The impact of the smaller σ value is that similarities between points are increased, resulting in recommendations involving a larger number of clusters. We see that, overall, the number of steps required to reveal a given k-clustering is

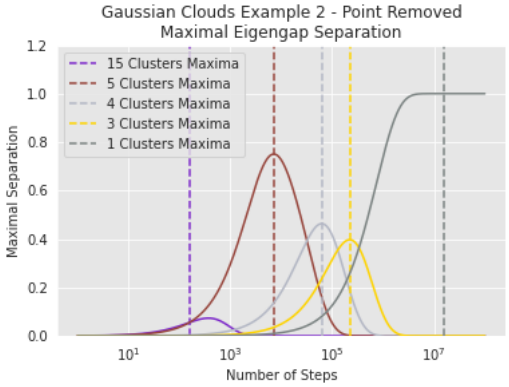

Figure 2.5: Gaussian Clouds Example 2 (Outlier Point Removed Variation) - Impact on the Maximal Eigengap Separation

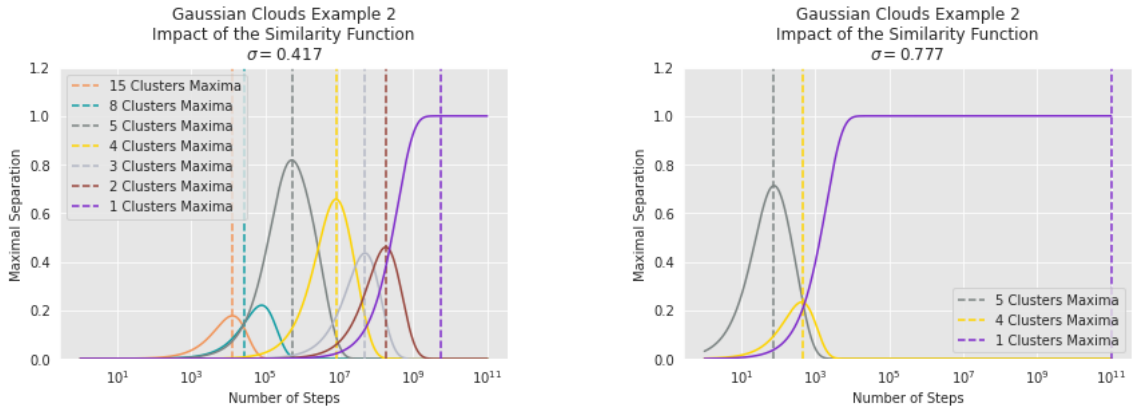

Figure 2.6: Gaussian Clouds Ex. 2. Impact of the Similarity Function - Varying Sigma

increased in the setting with larger similarities, corresponding to larger eigenvalues.

The only nontrivial clusterings of the left-hand plot of Figure 2.6 which remain after the change of similarity function are the 5-clustering and the 4-clustering. Increasing σ further removes the 4-clustering, and in the limit as σ increases the similarity between all points tends to zero, yielding the trivial 1-clustering. In this way the clusterings with higher eigengap separation correspond to equilibria in the random walks with higher stability.

We intend in this paper to discuss a data-driven approach to regime classificaiton as far as possible; this example however highlights the importance of the similarity function as well as the metric space structure on the set of points to be clustered. The understanding is that well-clustered data will resist small changes to the similarity function and slight alterations to the dataset, although this choice of structure on the space remains an important manual aspect to the algorithm which much be acknowledged.

	Original Zero Steps	Point Removed Zero Steps	Original 6915 Steps	Point Removed 6915 Steps
λ_1	1.00000000	1.00000000	1.00000000	1.00000000
λ_2	0.99999864	0.99999866	0.99065273	0.99077991
λ_3	0.99999751	0.99999758	0.98291351	0.98338968
λ_4	0.99999282	0.99999245	0.95157578	0.94910946
λ_5	0.99999219	0.99997074	0.94743909	0.81679964
λ_6	0.99996992	0.99960787	0.81217599	0.06639451
λ_7	0.99959761	0.99950583	0.06184723	0.03277786

Table 2.1: Largest eigenvalues for the transition matrix induced by the original set of points, and the set of points with the outlier removed

2.4 Synthetic Time Series

We now work towards the clustering of market time series data. In this example we work with simulated price paths, each belonging to one of a predefined number of regimes. The simulations used are realisations of a Geometric Brownian Motion process.

Definition 2.4.1 (Standard Brownian Motion)**.** A continuous-time stochastic process W_t is called a *Standard Brownian Motion* if the following hold:

i) $W_0 = 0$

ii) $t \mapsto W_t$ is almost-surely continuous

iii) W_t has independent increments: any increment $W_{t+u} - W_u$ is independent of W_s for any past value $s \leq t$

iv) The increments are normally distributed: $W_{t+u} - W_t \backsim \mathcal{N}(0, u)$

Definition 2.4.2 (Geometric Brownian Motion)**.** A continuous-time stochastic process S_t is called a *Geometric Brownian Motion* if there exists $\mu, \sigma \in \mathbb{R}$ with $\sigma > 0$ such that S_t satisfies the stochastic differential equation

$$dS_t = \mu S_t \, dt + \sigma S_t \, dW_t \tag{2.4.1}$$

where W_t is a standard Brownian motion.

The solution to equation 2.4.1 is given by

$$S_t = S_0 \exp\left(\left(\mu - \frac{\sigma^2}{2}\right) t + \sigma W_t\right) \tag{2.4.2}$$

the variables μ, σ of Equation 2.4.2 are called the *drift* and *volatility* respectively. The drift term governs the long-term expected value of S_t, whereas the volatility indicates how far away from this expected value the value of S_t is likely to be.

In this example, a regime corresponds to a choice of the parameters (μ, σ). We demonstrate the clustering of Brownian paths by selecting five regimes according to the parameters in Table 2.2. This example will serve as a simplified version of the main target of the paper. The next section explains how points of a metric space are generated once this choice has been made.

	μ	σ
Regime 1	0.001	0.008
Regime 2	0.001	0.003
Regime 3	-0.001	0.008
Regime 4	-0.001	0.003
Regime 5	0	0.008

Table 2.2: Parameters for Sample Brownian Paths of Figure 2.7

2.4.1 Regime Points and Point Elements

The understanding is that, over small time horizons, market data will have an approximately constant regime. For each regime (μ, σ), we generate fixed-length paths representing the evolution of a price process over some observation period. All paths generated will begin at the same starting point and the observed displacements, volatilities and higher order terms which appear in the path signature will be used to identify regime.

In the market data, for a given observation window (one week, for example), many paths are available to suggest the underlying regime for that interval. There is a choice to be made about how these paths are chosen. One could, for instance, take individual stock price paths and generate a collection of signatures. Another choice would be to consider multiple stocks together as a single multi-dimensional path, allowing the signatures to reflect cross-correlation terms, which may provide insight into regimes also.

Let us take, for example, a one-week observation window. A point in the metric space of our current setting will be a representation of a given week's regime. In this setting, the week's regime may be represented by a collection of path signatures or logsignatures. Due to the discussion of Section 1.2, we choose to represent the regime by the collection of logsignatures. In order to consider Azran-Ghahramani clustering on this space, we therefore require a metric between collections of logsignatures.

In real market data, due to data availability, the number of paths available varies from one week to the next. The size of the logsignature collections representing the observation periods therefore vary from one window to the next. We therefore require a metric capable of comparing collections of different sizes.

Algorithm 3 outlines the construction of points in our space. A point, generated under a regime with drift μ and volatility σ, is denoted by $\mathcal{X}^{\mu,\sigma}$. The number of elements n in a given point $\mathcal{X}$ is chosen at random from some predetermined subset of $\mathbb{N}$. The subset is chosen to approximately match the size of collections available when working with real data in the next chapter. The signature level l is constant between points and hence is not included in the notation.

Repeating Algorithm 3 k times we obtain the collection of points $\{\mathcal{X}_1^{\mu,\sigma}, \ldots, \mathcal{X}_k^{\mu,\sigma}\}$, which are the points of the space corresponding to the regime with parameters (μ, σ). Repeating this for each regime in Table 2.2, we obtain the set of points in our space. In the next section, we discuss the distance function which equips this set with the structure of a metric space.

Algorithm 3: Generation of a regime-point

Input: Maximum signature level l, path dimension d, regime drift μ and volatility σ. Minimum and maximum number of elements $N_{\min}$ and $N_{\max}$, path length m

Output: Single metric space point

1. Select number of elements n uniformly at random from the set $\{N_{\min}, \ldots, N_{\max}\}$
2. For each $i \in \{1, \ldots, n\}$, simulate a sample one-dimensional geometric Brownian motion path[4] $s_i = (s_i^1, \ldots, s_i^m)$, of length m, with drift μ and volatility σ
3. For each i, time-augment the path to obtain a two-dimensional path $\{(1, s_i^1), \ldots, (m, s_i^m)\}$
4. Linearly interpolate the points of step 3 to form a piecewise-differentiable function $\tilde{s}_i : [1, m] \to \mathbb{R}^d$
5. Let $x_i = S(\tilde{s}_i)_{1,m}^{\leq l}$ be the signature transform of the path $(\tilde{s}_i)$ up to level l
6. Return $\mathcal{X}_{\mu,\sigma} := \{x_1, \ldots, x_n\}$

2.4.2 A Distance between Collections of Signatures

The approach will be to consider the points $\mathcal{X} = \mathcal{X}^{\mu,\sigma}$ and $\tilde{\mathcal{X}} = \tilde{\mathcal{X}}^{\tilde{\mu},\tilde{\sigma}}$, generated according to Algorithm 3, as random samples from some distributions p and $\tilde{p}$ of path signatures. The distance between the two collections may then be understood as a distance between these two distributions. Indeed, in this paper a market regime is seen as synonymous with a cluster in the space of distributions of path signatures.

We seek then an estimate on the distance between the distributions p and $\tilde{p}$ given the samples $\mathcal{X}$ and $\tilde{\mathcal{X}}$. A *two-sample hypothesis test* is a general framework employed to tackle these problems. Given a collection $X = \{x_1, \ldots, x_n\}$ of independent samples from a population p and a collection $Y = \{y_1, \ldots, y_m\}$ of samples from a population $\tilde{p}$, a two-sample hypothesis test is used to determine whether there is sufficient evidence at some significance level to reject the null hypothesis that $p = \tilde{p}$.

The *maximum mean discrepancy* test statistic is a two-sample hypothesis test statistic introduced in [7] which provides us with a statistic by which we will quantify how different the distributions generating our two collections are. In order to define the maximum mean discrepancy statistic, we first recall the definition of a *Reproducing Kernel Hilbert Space*, the definition of which is adapted from [20].

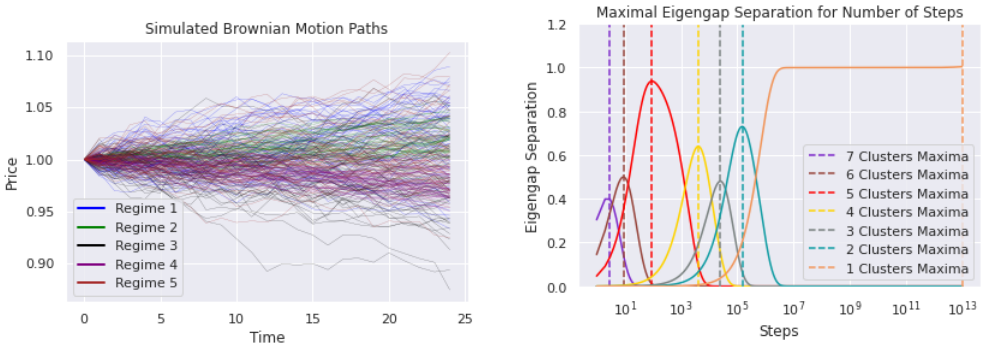

Figure 2.7: Synthetic Paths Example - Generated Paths and Eigengaps Plot

Definition 2.4.3 (Reproducing Kernel Hilbert Space)**.** Let X be a set, and let $\mathcal{H}$ be a Hilbert space[5] of functions $X \to \mathbb{R}$. For each $x \in X$, the *evaluation functional* is the function $\mathcal{L}_x : \mathcal{H} \to \mathbb{R}$ defined by

$$\mathcal{L}_x(f) := f(x) \tag{2.4.3}$$

We say that $\mathcal{H}$ is a *Reproducing Kernel Hilbert Space* if, for every $x \in X$, the functional $\mathcal{L}_x$ is continuous (with respect to the standard metric on $\mathbb{R}$ and the metric induced by the inner product on $\mathcal{H}$).

If $\mathcal{H}$ is a Reproducing Kernel Hilbert Space then, for each $x \in X$, there exists a *reproducing kernel* $K_x \in \mathcal{H}$ such that the evaluation functional $\mathcal{L}_x$ can be computed by the inner product with K_x. That is, for any $f \in \mathcal{H}$:

$$\mathcal{L}_x(f) := f(x) = \langle f, K_x \rangle \tag{2.4.4}$$

where the inner product is provided by the Hilbert space structure on $\mathcal{H}$. The *reproducing kernel for* $\mathcal{H}$ is the function $k : X \times X \to \mathbb{R}$ given by

$$k(x, y) := \langle K_x, K_y \rangle \tag{2.4.5}$$

From [21], the reproducing kernel hibert space $\mathcal{H}$ is said to be *universal* if $k(x, \cdot)$ is continuous for all x and $\mathcal{H}$ is dense[6] in $C(X)$, the space of continuous functions $X \to \mathbb{R}$.

We have the following formulation of the maximum mean discrepancy statistic, due to [7]:

Definition 2.4.4 (Maximum Mean Discrepancy)**.** Let $\mathcal{F}$ be a class of functions $f : X \to \mathbb{R}$, and let p, q be distributions on X. The *Maximum Mean Discrepancy* over p, q over $\mathcal{F}$ is defined to be

$$\text{MMD}[\mathcal{F}, p, q] := \sup_{f \in \mathcal{F}} \left(\mathbb{E}_{x \backsim p} [f(x)] - \mathbb{E}_{y \backsim q} [f(y)] \right) \tag{2.4.6}$$

If, instead of observing the distributions p and q, we have independent observations $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_n\}$ from the distributions p, q respectively, then an empirical estimate of the maximum mean discrepancy statistic is given by

$$\text{MMD}[\mathcal{F}, X, Y] := \sup_{f \in \mathcal{F}} \left(\frac{1}{m} \sum_{i=1}^{m} f(x_i) - \frac{1}{n} \sum_{i=1}^{n} f(y_i) \right)$$

Let $\mathcal{H}$ be a universal reproducing kernel Hilbert space with associated kernel k. If we choose the class of functions $\mathcal{F}$ to be the unit ball in $\mathcal{H}$, then an empirical estimate for the maximum mean discrepancy may be computed in terms of the kernel $k(\cdot, \cdot)$ by

$$\text{MMD}[\mathcal{F}, X, Y] = \left[\frac{1}{m^2} \sum_{i,j=1}^{m} k(x_i, x_j) - \frac{2}{mn} \sum_{i,j=1}^{m,n} k(x_i, y_j) + \frac{1}{n^2} \sum_{i,j=1}^{n} k(y_i, y_j) \right]^{1/2} \tag{2.4.7}$$

[5]See the appendices for the definition of a Hilbert Space
[6]See the appendices for the definition of a dense subset

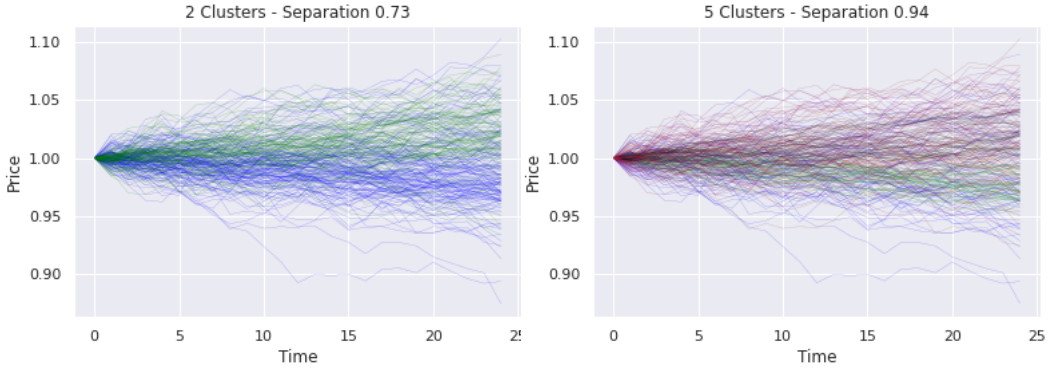

Figure 2.8: Synthetic Paths Example - Best 2-Clusterings and 5-Clusterings

In [21], it is shown that in particular the reproducing kernel Hilbert space induced by the Gaussian kernel

$$k^{\sigma}(x, y) := \exp\left(-\frac{1}{2\sigma^2}\|x - y\|^2\right) \tag{2.4.8}$$

on compact subsets of $\mathbb{R}^d$ is universal, for fixed $\sigma \in \mathbb{R}$. It is this kernel and this space which we will be utilising in this example. Note that the maximum mean discrepancy in particular is not a distance function, since Equation 2.4.6 is dependent on the order of p and q. Nevertheless, in [7] it is demonstrated that if the induced Hilbert space $\mathcal{H}$ is a universal reproducing kernel Hilbert space then the maximum mean discrepancy statistic has the property that $\text{MMD}[\mathcal{F}, p, q] = 0$ if and only if $p = q$. Furthermore, the empirical estimate provided in 2.4.7 is symmetric in its arguments; is it this measure of disparity between the collections X and Y which we take as a distance function in the Azran-Ghahramani algorithm. We demonstrate in the next section that this choice of metric is sufficient to allow for the classification of regimes from Brownian paths.

2.4.3 Results

The parameters (μ, σ) chosen to represent regimes have been presented in Table 2.2. For each regime, 10 samples were generated according to Algorithm 3. For each sample, a random number of path innovations between 75 and 100 were generated, simulating the differing quantity of data available when working with real market data. Each generated path was chosen to start at value 1, and continue for 25 steps. A subset of the resulting paths (with 50 paths per regime) is presented in the left-hand plot of Figure 2.7. For each point in the space, the logsignatures up to level 3 are computed. The second-order terms in the signature vectors represent areas, as seen in Chapter 1, and hence are much smaller in absolute value than the displacement terms, due to the small values the paths take in this setting. Since the Gaussian kernel of Equation 2.4.8 considers the Euclidean distance between points, we found it helpful to scale the signature terms so that, for each entry in the vector, most[7] of the population has a value between 0 and 1. The distance function is taken as the MMD estimate of Equation 2.4.7, and we take the similarity function of Equation 2.3.1 from the previous Gaussian Clouds example. From this setup, we may proceed with a similar analysis as in the Gaussian Clouds examples.

The maximal eigengap separation plot is presented as the right-hand plot of Figure 2.7. As in the previous example, this metric space structure is deemed successful at clustering the underlying paths in the sense that the nontrivial clustering with the highest eigengap separation is the 5-clustering, which is presented on the right of Figure 2.8. For comparison, the suggested 2-clustering is presented on the left. The clusters suggested by the algorithm also coincide with the problem specification, as we discuss next.

From Figure 2.8, it is hard to determine the quality of the output of the algorithm. Let us denote by $\mathcal{R}_1, \ldots, \mathcal{R}_5$ the point indices corresponding to the regimes of Table 2.2, with the same numberings. We separate the regime $\mathcal{R}_3 = \mathcal{R}_3^1 \cup \mathcal{R}_3^2$, and $\mathcal{R}_5 = \mathcal{R}_5^1 \cup \mathcal{R}_5^2$; these subsets are of sizes $|\mathcal{R}_3^1| = 4$, $|\mathcal{R}_3^2| = 6$, $|\mathcal{R}_5^1| = 5$, and $|\mathcal{R}_5^2| = 5$, and appear in the suggested 7-clustering in the output of the Azran-Ghahramani clustering when run over this space.

[7] We scale each term in the signature vector by the 95^{th} percentiles of the population values for this entry

The resulting clusters are presented in Table 2.3. The 6-clustering output suggested contains an empty partition, and is otherwise identical to the preferred 5-clustering suggested.

	Partition
7 Clusters	$\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3^1, \mathcal{R}_3^2, \mathcal{R}_4, \mathcal{R}_5^1, \mathcal{R}_5^2$
6 Clusters	$\emptyset, \mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4, \mathcal{R}_5$
5 Clusters	$\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4, \mathcal{R}_5$
4 Clusters	$\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3 \cup \mathcal{R}_5, \mathcal{R}_4$
3 Clusters	$\mathcal{R}_1 \cup \mathcal{R}_2, \mathcal{R}_3 \cup \mathcal{R}_5, \mathcal{R}_4$
2 Clusters	$\mathcal{R}_1 \cup \mathcal{R}_2, \mathcal{R}_3 \cup \mathcal{R}_4 \cup \mathcal{R}_5$

Table 2.3: Synthetic Data - Azran Ghahramani Algorithm Suggested Clusterings

This example has validated the concept of understanding market regimes as clusters in a space of (truncated) signature vectors. We have used the maximum mean discrepancy statistic to represent the distance between two such distributions, and have identified the desired clustering in the synthetic time series data. In the final chapter, we attempt to apply the same algorithm to actual market paths.

Chapter 3

Regime Classification

In this chapter, we present the results of the Azran-Ghahramani clustering algorithm when applied to real market data. The distance between collections of signature vectors is the same distance identified in the Synthetic Data example. We work with one-minute data, and attempt to classify the regime over one-day observation windows. The data used in this example was collected from *Yahoo! Finance*. The data set spans a four-month time period beginning on April 27$^{\text{th}}$ 2020 and concluding August 28$^{\text{th}}$ 2020. We refer to this period as the *observation period*.

For each day in the observation period, price paths were collected for as many tickers as possible from the SP500 index. In order to compare signature terms, we apply a preprocessing step of normalising each price path by the first component of the day. Price paths are therefore recorded as relative prices with the opening price scaled to have a value of 1. The paths over which signatures are computed are the time-augmented, two-dimensional paths encoding the price and time at which the price was recorded. Logsignatures up to level 4 for each path were computed. A subset[1] of the resulting price paths is presented as the left-hand plot of Figure 3.1.

The most suitable results were found when the signature of the *absolute price path* were considered. For a path, we define the *absolute price path* as the path $\tilde{X}$ defined inductively by $\tilde{X}_0 := X_0$ and for $i \geq 1$ we define $\tilde{X}_i := \tilde{X}_{i-1} + |X_i - X_{i-1}|$. That is, differences from one minute to the next compound, and a measure of volatility may be captured in the displacement of this absolute path over the day. We found that these terms gave a clearer indication of the volatility than the signature terms, which reflect signed areas and result in less separable points. The distance between days in the dataset is then given by the distance between *combination signatures*, which are signature-style vectors where the first two entries are the displacements in each axis, which come from the signature vector, and the remaining entries are taken from the signature of the absolute path.

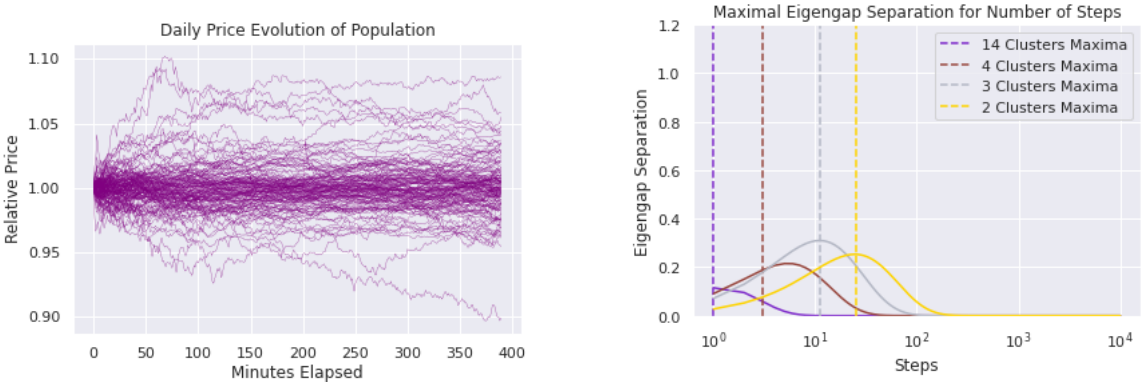

Figure 3.1: Market Data - Relative Price Paths and Resulting Eigengaps Plot

[1]The figure is presented with only a subset of the paths for clarity. The full set of paths is too large to be meaningfully displayed on one plot in this way.

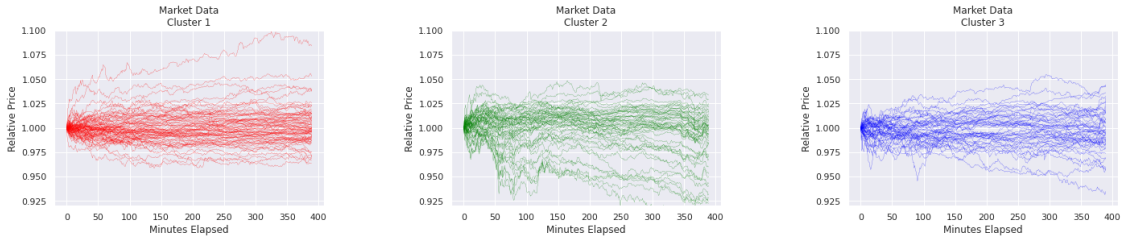

Figure 3.2: Market Data - Recommended 3-Clustering

3.1 Results

The maximal eigengap separation plot is presented on the right-hand side of Figure 3.1; the curve corresponding to the 1-clustering is not displayed. We note first that, in contrast to the synthetic examples, the eigengap separation corresponding to the recommended clusterings are much lower that the successful clusterings of the previous sections. Indeed, we found that the distances between points were much more evenly distributed than in the previous examples. Natural clusters are therefore harder to identify. As we have seen in the second Gaussian Clouds example, this low eigengap separation corresponds to less stable clusterings, which are sensitive to the choice of similarity function for example. Indeed, for this example the similarity function is taken to be the function $w(x) := 1/x^5$, rather than the similarity functions which produced strong results in the previous chapter. This choice of similarity function was found to produce the most convincing results.

We demonstrate the suggested 3-clustering in Figure 3.2. Cluster 2 contains only a single day, similar to the outlier point discussed in the second Gaussian Clouds example. We can discuss several characteristics which have been identified as meaningful attributes by which to separate the data. Comparing Cluster 1, the left-hand plot of Figure 3.2, with Cluster 3, the right-hand plot of the same Figure, we see that neither collection displays significant drift overall across a one-day period. The volatility of the constituent paths in Cluster 1 however appear to have notably lower volatility than those of Cluster 3. Cluster 2 on the other hand, a singleton cluster, displays rather different characteristics of paths experiencing sharp selloffs. Indeed many paths display a 5% loss over a one-day interval. For this reason, the MMD distance from other points is deemed significant enough to warrant a singleton cluster.

To make this precise, we compute the mean closing price and mean quadratic variation for the full collection of paths which make up each cluster. Recall the following formulation of the quadratic variation of a discrete path $X = \{x_1, x_2, \ldots, x_n\}$:

$$QV(X) := \sum_{i=2}^{n} (x_i - x_{i-1})^2 \tag{3.1.1}$$

For each path in the cluster, we may compute the quadratic variation according to Equation 3.1.1. A notion of volatility for each cluster is given by taking the mean of all such quadratic variations across all constituent paths. We present this statistic, along with the cluster's mean closing price, in Table 3.1. We note that the quadratic variation of Cluster 3 is indeed around double the quadratic variation of Cluster 1. Cluster 2 also exhibits high quadratic variation. The clusters identified in the data would be characterised as zero mean, low variance; negative mean, high variance; zero mean, high variance.

	Average Closing Value	Average Quadratic Variation
Cluster 1	1.0027	$1.202e^{-6}$
Cluster 2	0.9878	$1.997e^{-6}$
Cluster 3	0.9995	$2.246e^{-6}$

Table 3.1: Market Data - Means and Quadratic Variations of Recommended Clusters

Chapter 4

Conclusion

In this paper, we have presented a data-driven algorithm to classify market regimes. We have seen how market conditions may be thought of as probability distributions on path signatures, and that market regimes may be thought of as clusters of such distributions. We have attempted to present an algorithm which is as free from manual specification as possible. We discuss in this chapter both the shortcomings of the current algorithm and possible directions for future work.

Firstly, the framework presented has only proposed a method in which market regimes may be identified, but does not provide transition probabilities between the regimes. Indeed, the algorithm is backwards-looking, and makes no attempt to predict market conditions. Future work could include adapting the current algorithm to take recent market conditions into account, rather than treating each sample as independent.

The regimes indicated in the final example are fairly straightforward, being identified mainly by volatility. With higher-order signature terms, we found it difficult to generate meaningful clusters. Firstly, we would like to investigate this market application further with a larger dataset. In addition, however, we note that the clusters suggested are highly sensitive to the choice of similarity function and metric put on the space. We have seen the motivation to use empirical estimates of maximum mean discrepancy as a distance function between distributions of signatures, and this seems quite a natural description. The choice of distance function then is reduced to a choice of kernel. Whilst kernels associated with *universal* reproducing kernel hilbert spaces are well-motivated by the theoretical results surrounding their MMD distance, discussed in Chapter 2, we have a wide choice of such kernels to choose from (see [21]). It is not clear which choice of kernel results in a distance function most faithfully encoding market regimes, or if such a question can be answered in a definitive fashion. A choice of kernel which separates the data more convincingly would allow for subtler signature terms to be incorporated into the regime specification and produce stronger results.

The eigengap separations of the final example are much lower that in the successful clusterings of the previous chapter. Our opinion is that this cluster instability is due to a more continuous spread of distances between points, as discussed in Chapter 3. This example suggests that the clustering algorithm discussed performs poorly when clearly-separable clusters are not available. Whilst we have investigated this issue in the Gaussian Clouds example, potential further work could include clustering the signature distributions using a *fuzzy* or *soft* clustering algorithm, in which points may belong to multiple clusters at once. We expect higher stability in such an algorithm's output and more convincing descriptions of market regime.

It remains also to apply the algorithm to other markets. The algorithm presented here is sufficiently general that we expect to find similar results. Nevertheless, this is another clear avenue for future work.

Appendix A

Supplementary Definitions and Algorithms

A.1 Algebraic Structures

Definition A.1.1 (Vector Space)**.** A *vector space over a field* F is a set V, equipped with an addition $+ : V \times V \to V$, and a scalar multiplication $\cdot : F \times V \to V$ such that, for all $x, y, z \in V$ and $\lambda, \mu \in F$:

i) $x + y = y + x$

ii) $(x + y) + z = x + (y + z)$

iii) There exists an element $0 \in V$ such that $0 + x = x$ (for any $x \in V$)

iv) For every $x \in V$ there exist an element $-x \in V$ such that $x + (-x) = 0$

v) $\lambda \cdot (\mu \cdot x) = (\lambda\mu) \cdot x$

vi) $(\lambda + \mu) \cdot x = (\lambda \cdot x) + (\mu \cdot x)$

vii) $\lambda \cdot (x + y) = (\lambda \cdot x) + (\lambda \cdot y)$

viii) The element $1 \in F$, the multiplicative identity of F, satisfies $1 \cdot x = x$

Definition A.1.2 (Inner Product Space)**.** Let V be a vector space over a field F. An *inner product* on V is a map $\langle\,\rangle : V \times V \to F$, written $(x, y) \mapsto \langle x, y\rangle$, which satisfies, for all $x, y \in V$ and $\lambda \in F$:

i) $\langle x + y, z\rangle = \langle x, z\rangle + \langle y, z\rangle$

ii) $\langle x, y\rangle = \langle y, x\rangle$

iii) $\langle \lambda \cdot x, y\rangle = \lambda \cdot \langle x, y\rangle$

iv) $\langle x, x\rangle \geq 0$ and $\langle x, x\rangle = 0 \Leftrightarrow x = 0$

The pair $(V, \langle \cdot, \cdot \rangle)$ of a vector space equipped with an inner product on V is called an *inner product space*.

Definition A.1.3 (Algebra over a field)**.** Let V be a vector space over a field F. We say that V is an *algebra over* F if, in addition to the vector space structure, there is a multiplication $\times : V \times V \to V$ such that, for any $x, y, z \in V$ and $\lambda, \mu \in F$ we have

i) $(x + y) \times z = (x \times z) + (y \times z)$

ii) $z \times (x + y) = (z \times x) + (z \times y)$

iii) $(\lambda \cdot x) \times (\mu \cdot y) = (\lambda\mu) \cdot (x \times y)$

Definition A.1.4 (Complete Metric Space)**.** Let (X, d) be a metric space[1]. Let $(x_n)_{n=1}^{\infty} \subset X$ be a *sequence in* X.

i) The sequence $(x_n)_{n=1}^{\infty}$ in X *converges to a point* $x^* \in X$ if, for all $\epsilon > 0$ there exists a number $N \in \mathbb{N}$ such that $n > N \Rightarrow d(x_n, x^*) < \epsilon$.

ii) The sequence $(x_n)_{n=1}^{\infty}$ is *Cauchy* if, for all $\epsilon > 0$ there exists a number $N' \in \mathbb{N}$ such that $n, m > N \Rightarrow d(x_n, x_m) < \epsilon$.

The metric space (X, d) is *complete* if these concepts coincide. That is, every Cauchy sequence converges (the converse always holds).

Definition A.1.5 (Hilbert Space)**.** Let $(V, \langle \cdot, \cdot \rangle)$ be an inner product space. The inner product $\langle \cdot, \cdot \rangle$ induces a *norm* on V by $|x| := \langle x, x \rangle$. We can in turn define a metric space structure on V by defining that $d(x, y) := |x - y|$. The tuple (V, d) is then a metric space. If the metric space (V, d) defined in this way is a *complete metric space*, then we say that the inner product space $(V, \langle \cdot, \cdot, \rangle)$ is a *Hilbert space*.

Definition A.1.6 (Dense Subset)**.** Let $(X, \mathcal{T})$ be a topological space. We say that a subset $A \subset X$ is *dense in* X if the closure $\bar{A}$ of A is the whole set X: $\bar{A} = X$. In the context of the universal reproducing kernel hilbert space of definition 2.4.3, density of a subset of a metric space refers to density in the topology induced by the metric (the topology generated by all open balls around points of the space). As in definition A.1.5, the metric space structure on a hilbert space is itself induced from the norm $\langle \cdot, \cdot \rangle$.

A.2 Algorithms

Algorithm 4: Star-Shaped Prototypes Initialisation

Input: Transition matrix P, number of clusters k
Output: Matrix of prototypes Q

i) Set $Q_1 := \frac{1}{n} \sum_{i=1}^{n} P_n$

ii) For $j = 2, \ldots, k$, set $Q_j := P_m$, where $m = \text{argmax}_a \min_{l=1,\ldots,j-1} KL\big(P_a \,\big\|\, Q_j\big)$

That is, after initialising the first row, each subsequent row is chosen from the transition matrix P so as to maximise the minimum divergence to all previously-selected rows. One can think of choosing the next row of P so as to maximise the distance to all other prototypes.

Algorithm 5: Generation of a one-dimensional Geometric Brownian Motion path

Input: Time steps $\mathcal{T} = \{t_1, \ldots, t_n\}$, drift μ, volatility σ
Output: Sequence $\{W_0, W_1, \ldots, W_n\}$ with $W_i \in \mathbb{R}$, realisations of a Brownian Path with required drift, volatility at time steps $t \in \mathcal{T}$

i) Set $W_0 = 0$

ii) For each $i = 1, \ldots, n$, generate independent $Z_n \sim \mathcal{N}(\mu, \sigma^2)$

iii) Set $W_1 := \sqrt{t_1} Z_1, W_2 := \sqrt{t_2 - t_1} Z_2, \ldots, \sqrt{t_n - t_{n-1}} Z_n$

iv) Return $\{W_0, W_1, \ldots, W_n\}$

[1] See Definition 2.1.1 for the definition of a metric space

Bibliography

[1] M. Kritzman, S. Page, and D. Turkington. Regime shifts: Implications for dynamic strategies. *Financial Analysts Journal*, 68:22–39, 2012.

[2] A. Jiltsov. Market regimes: How to spot them and how to trade them. 2020. FX Markets. https://www.fx-markets.com/comment/7665631/market-regimes-how-to-spot-them-and-how-to-trade-them.

[3] P. Nystrup, P. N. Kolm, and E. Lindström. Greedy online classification of persistent market states using realized intraday volatility features. *The Journal of Financial Data Science*, 2.3, 2020.

[4] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. Preprint, arXiv:1712.01815, 2017.

[5] K. T. Chen. Integration of paths - a faithful representation of paths by noncommutative formal power series. *Transactions of the American Mathematical Society*, 89(2):395–407, 1958.

[6] A. Azran and Z. Ghahramani. A new approach to data driven clustering. *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

[7] A. Gretton, K. M. Borgwardt, M. Rasche, B. Schölkopf, and A. J. Smola. A kernel method for the two-sample problem. Preprint, arXiv:0805.2368, 2008.

[8] H. Bühler, B. Horvath, T. Lyons, I. P. Arribas, and B. Wood. A data-driven market simulator for small data environments. Preprint, arXiv:2006.14498, 2020.

[9] H. Boedihardjo, X. Geng, T. Lyons, and D. Yang. The signature of a rough path: Uniqueness. Preprint, arXiv:1406.7871, 2015.

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

[11] I. Chevyrev and A. Kormilitzin. A primer on the signature method in machine learning. Unpublished, arXiv:1603.03788, 2016.

[12] P. Friz and M. Hairer. *A Course on Rough Paths with an Introduction to Regularity Structures.* Springer, 2015.

[13] J. Reizenstein. Calculation of iterated-integral signatures and log signatures. Unpublished, ResearchGate 321665636, 2017.

[14] D. Levin, T. Lyons, and H Ni. Learning from the past, predicting the statistic for the future, learning an evolving system. Preprint, arXiv:1309.0260, 2013.

[15] T. Lyons, M. Caruana, and T. Lévy. *Differential Equations Driven by Rough Paths.* Springer-Berline Heidelberg, 2007.

[16] P. Bonnier, P. Kidger, I. P. Arribas, C. Salvi, and T. Lyons. Deep signature transforms. *Advances in Neural Information Processing Systems*, 32:3082–3092, 2019.

[17] P. Kidger and T Lyons. Signatory: differentiable computations of the signature and logsignature transforms, on both CPU and GPU. Preprint, arXiv:2001.00706, 2020.

[18] J. Reizenstein and B. Graham. The iisignature library: Efficient calculation of iterated-integral signatures and logsignatures. Preprint, arXiv:1802.08252, 2018.

[19] S. Assefa, D. Dervovic, M. Mahfouz, T. Balch, P. Reddy, and M. Veloso. Generating synthetic data in finance: Opportunities, challenges and pitfalls. Preprint, SSRN:3634235, 2020.

[20] Reproducing kernel hilbert space. https://en.wikipedia.org/wiki/Reproducing_kernel_Hilbert_space.

[21] I. Steinward. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2001.