In Search of Structure: Unsupervised Learning in Foreign Exchange

Jonathan Millin

Master of Science

School of Informatics

University of Edinburgh

2010

Abstract

The search for structure in financial time-series has recently become a topic of great interest to economists, but has mainly focused on structures existing between time-series to optimise portfolios of stocks and manage risk. This study takes different approach aimed at finding structures within a single financial time-series and exploiting these structures for financial gain. An information-rich feature space was created, and unsupervised clustering techniques were applied to find internal structures. Three techniques were employed to utilise these structures for decision making: dimensionality reduction, market state as cluster membership, and market state as cluster transitions. The latter two were found to perform better than a baseline trader on foreign exchange data, and have potential to generate profits greater than a fair return. When these strategies were applied to artificially generated data, it was found that these profitable structures were missing.

Acknowledgements

I would like to thank the friends who took the time to proofread this thesis.

My supervisor, Dr. Subramanian Ramamoorthy, for his input, guidance and most importantly his patience with me throughout the year.

I would also like to thank the Informatics department at The University of Edinburgh for the resources that made this research possible.

Last but not least I would like to thank my parents: Roger and Wendy Howson for their sponsorship of my academic career and continued support in all aspects of my life.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Jonathan Millin*)

I dedicate the thesis to my mother, Wendy – who instilled in me a quest for knowledge and the value of personal research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Statement

Historically, economists have believed that markets are perfectly efficient, and profits could not be obtained from asset price information alone [37]. Recent studies into the randomness of financial time-series propose that repetitions do occur, and the degree of predictability can be estimated [46, 17, 51].

This suggests that inefficiencies exist in financial markets which can be exploited for financial gain. While much research has gone into finding structures between different financial time-series to optimise portfolios of stocks, little effort has been put into the identification of structures within a single financial time-series.

This project aims to identify such structures through the application of unsupervised learning techniques on an information rich feature space. Once identified, these structures are to be used in a variety of autonomous trading systems to identify whether such structures point to inefficiencies that can be exploited for financial gain.

1.2 Background

In the past, the Random Walk Hypothesis (RWH) and Efficient Market Hypothesis (EMH) have been the cornerstones to economists understanding financial markets. RWH and EMH hypothesise that financial markets are perfectly efficient, and profits cannot be found on price information alone [37].

RWH was initially disproved in 1988 by Lo and MacKinlay [34] and their results have subsequently been confirmed by many others [35]. More recently the attack has turned towards the EMH, in which extensive analysis of financial data is empirically

leading the way to the breakdown of the EMH in its weak-form [22, 39, 46]. Findings such as these suggest that recurring patterns and underlying structures exist within financial time-series.

The ability to detect and predict such structures and patterns would not only engender profitability in trading, but would also lead to a deeper understanding of the way financial time-series work. The search for structures in financial data has primarily been directed towards finding correlations between different stocks or currencies in order to create optimal portfolios, detect changes in a stock or currency, and to forecast volatility [16, 21, 41]. The fact that these efforts are being lead by Nobel prize winning economists bears testament to the importance of this field.

Technical analysis is a collection of algorithms and mechanical rules which attempt to aid investors in forecasting future market movements and understanding the current market state, using only historic data [43]. Academia however, has found mixed results from technical analysis, arguing that such attempts find no sufficient basis for profitability [19]. However, recent works in Approximate and Sample Entropy suggest that there are inefficiencies and repetitions in financial time-series, and they are measurable through the entropy of the time-series [46, 17, 51]. This provides reason to investigate the possibility of finding structures and patterns within a single financial time-series.

1.3 Specific Objectives

In order to deem this project successful, four main objectives need to be fulfilled.

Creation of an Information Rich Feature Space The first objective is to identify and implement methods of creating an information-rich feature space that characterises the current state of a financial time-series. This involves taking a single time-series of asset prices, and expanding it into a joint feature space containing a number of interesting statistics about the time-series which could be used to find structure. Loosely speaking, this is similar to the kernel trick, in that the time-series is mapped into a higher dimensional space in which machine learning techniques can be applied with greater success than can be achieved on the time-series alone [60].

Technical Analysis will provide some of the characterising statistics, and others will be drawn from common descriptions used to portray financial time-series in the literature.

Finding Structure using Unsupervised Learning Once a feature space has been created, unsupervised learning techniques will be applied to the it. The aim of this is to find structures in the feature space which are representative of the current market state. Structure will be sought after in historical foreign exchange data.

Identifying Ways of Exploiting Structure After structures have been identified, they need to be explored in further detail to understand how best to utilise them. This exploration will be performed on historical foreign exchange data which is not in either the test or training sets.

Building Systems to Harness Structures for Financial Gain A number of autonomous trading systems are to be developed in order to exploit the structures found in generated and historical data. This will prove empirically if one can capitalise on these structures, and whether generated and historical data contain similar inefficiencies. Profit gained above a fair return[1] in these systems will be used as an indicator of whether structures and inefficiencies have been found.

1.4 Scope of Research

This thesis is primarily concerned with the identification and exploitation of structure in a single financial time-series. Although technical trading tools (as used in technical analysis) are utilised to characterise the current market state, this is by no means intended as a comparison of technical trading rules, rather, a select few will be identified through literature and implemented based on their ability to describe the market well.

This project is rather meant to be a proof of concept in identifying if structures exist within a single financial time-series. This is not a project to compare the efficacy of different unsupervised learning techniques on foreign exchange data, nor is it a project to compare optimisation techniques. Therefore only a single unsupervised learning technique will be presented, and only a handful of optimisation techniques are shown. It is out of the scope of this project to investigate many different optimisation and unsupervised learning techniques.

The autonomous traders developed in this project are intended to be instruments to show whether or not any structures found could be used to identify inefficiencies in the market. Although profitability and risk are used as metrics of success, they are not

[1] Fair returns are those afforded by a risk free strategy such as a US treasury bill

intended to be exceptionally profitable. No other information about the currencies or economies of the respective countries is analysed. Consequently, the system will only analyse asset price information.

1.5 Contributions

This thesis finds that profitable structures may exist in real foreign exchange data. Furthermore, profits were not found in generated data. This suggests that the models investigated do not accurately capture all of the underlying structures present in real-world data. This implies that structures other than volatility clustering and trending exist in the original data, and these are being exploited to render profit. Another interesting observation made in this dissertation is that when the daily movements in the real-world data are randomly permuted, the profitable structures are destroyed, even when randomly permuting contiguous segments of these daily movements none of the profitable structure is retained.

These results were produced by creating an information rich feature space upon which unsupervised learning techniques were applied. The structures identified through the unsupervised learning were used in the designing of several autonomous trading strategies which were tested on generated and historical foreign exchange data.

This project identifies a possible method of finding and exploiting structure in foreign exchange data through the application of unsupervised learning. Although these results are not empirical proof of the existence of underlying profitable structures, they provide motivation for further research in this area, and equip future researchers with a solid foundation upon which to build.

1.6 Thesis Structure

In order to clearly address requirements, this thesis is structured around the aforementioned objectives. Chapter 2 introduces important concepts and previous works related to this study. Chapter 3 draws on these concepts in order to create an information rich feature space. Statistics which are used to characterise the market are described fully in this chapter, however exploration of this feature space is left until Chapter 4. Chapter 4 also describes the structures identified, as well as possible strategies to utilise the structures found. Chapter 5 presents the trading strategies developed to capitalise

on the identified structures; the results of which discussed and analysed in Chapter 6. Finally, the thesis is concluded in Chapter 7.

Chapter 2

Introduction to Concepts and Related Works

As little work has been done in the current field of research, a very brief overview of structural learning in financial time series is provided. This section find agglomerative hierarchical clustering as being a common method of unsupervised learning in financial data, thus this method is explained. Following this, a brief introduction to the Efficient Market Hypothesis is provided for the readers interest. The majority of this chapter is spent focusing on the characterising statistics to be used in building the information-rich feature space, and justification for each of these methods is given. Following these statistics, the models used for data generation in Section 5.3 will be explained. The explanation of the GARCH model is quite detailed as much insight can be gained as to the structure of financial data by understanding what the generative models are attempting to approximate. Finally the chapter is concluded with a brief introduction into genetic algorithms as an optimisation function.

2.1 Structure and Unsupervised learning in Financial time-series

Finding structure within financial markets is of great interest to investors. If a structure can be found, it can be exploited. Such exploitation could render greater profit margins with lower risk. Up till now, much research has gone into finding structure between financial time-series in order to create optimal portfolios, manage risk and forecast volatility [16, 21, 41]. Little research, however, explores structure within a single

time-series.

The most common technique used is by finding correlations between time-series based on correlation matrices [16, 41]. This however, does not prove to be a good method, as Ormerod and Mounfield [49] provide significant evidence that the correlation matrix of the change in asset prices is dominated by noise, and contains little genuine information. They showed this by applying random matrix theory to asset portfolios, and illustrate the ineptitude of correlation matrices on daily changes in foreign exchange rates. Thus suggests a change in direction. The direction taken by Ormerod and Mounfield [49] was rooted in Econophysics. They used hierarchical agglomerative clustering based on ultra-metric distances between currencies in order to identify currencies of greatest similarity.

Focardi [21] applied the unsupervised learning technique of clustering to financial time-series as a means of studying dependencies between variables.

2.2 Agglomerative Hierarchical Clustering

Agglomerative Hierarchical Clustering is a popular unsupervised learning technique. One of the reasons for its popularity is that it is non-parametric. This means that no parameter need be set in order to make use of this method [68].

This method works by assigning each data point as being its own cluster. It then merges clusters in order to minimise some distance function. The clusters and labels are recorded at each point in a dendrogram. By parsing through the dendrogram, one can retrieve the cluster labels for all points for any number of clusters. This makes the agglomerative hierarchical clustering technique ideal for exploratory data analysis in situations where one is not sure as to the correct number of clusters required [68].

One can use any distance metric to build up the tree, on one of a number of distance functions. Common distance functions are:

- **Average** - This tries to minimise the average distance between all points within the cluster. Adding new points with this method tends to build up spherical clusters.
- **Single** - This minimises the single smallest distance. This method tends to create long string-like clusters.
- **Complete** - This method minimises the furthest distance between any two points in the clusters.

2.3 Efficient Market Hypothesis

Part of the reason why little research has been dedicated towards identifying structures within a single financial time series is because of the *Efficient Market Hypothesis* (EMH).

The EMH states that the current market price fully reflects all of the available information. In short, this suggests that the trading of any stock or currency is a game of chance, and that no information exists which can guarantee profit [37]. EMH was independently developed by Samuelson [58] and Fama [18] in the 1960's, and is firmly ingrained in the ethos of economics [35].

There are three major versions of the EMH.

- **Weak** - The weak-form asserts that all publicly available information is already reflected in the price of an asset.
- **Semi-strong** - The semi-strong form claims that not only is all publicly available information reflected in the price of an asset, but the prices change instantly to reflect new publicly available information.
- **Strong** - In its strong-form, the EMH claims the same as the semi-strong, but adds that the price of an asset will also instantly reflect hidden information.

2.4 Technical Analysis

Technical trading tools are mechanical rules which identify good positions in which to buy and sell in financial times series. These, together with Fundamental Analysis, are amongst the two broad categories of tools used to guide practising investors on when to buy, hold or sell their assets. While Fundamental Analysis uses information about markets (such as price-earnings ratios and interest rates) to gain an understanding of a market, Technical Analysis disregards all of this information, only reviewing trading data: such as price levels and volumes [45].

One can use signals generated by fundamental analysis however, technical analysis has orders-of-magnitude more data available, which makes it much more suitable for Machine Learning, and thus is pursued in this project.

In essence, Technical Analysis attempts to forecast the movements in future markets, using historic data as a basis. According to Brock, Lakonishok and Lebaron [5]:

> "Technical analysis is considered by many to be the original form of investment analysis, dating back to the 1800s. It came into widespread use before the period of extensive and fully disclosed financial information, which in turn enabled the practise of fundamental analysis to develop. In the United States, the use of trading rules to detect patterns in stock prices is probably as old as the stock market itself."

Technical analysis is still widely employed in the market by practising investors, who believe that inefficiencies exist in the short-run, and that these are exploitable by technical strategies in order to earn profit [47].

In spite of this, academics have treated Technical Analysis with scepticism. According to Park and Irwin [50], this scepticism can be linked to the *Efficient Market Hypothesis* [19], which postulates that attempting to make profits by exploiting currently available information is futile, as any possible opportunity would already have been exploited due to the efficiency of the market. Early studies of technical analysis in the stock market from Fama and Blume (1966) [20], Van Horne and Parker (1967, 1968) [65, 66] and Jensen and Bennington (1970) [30], amongst others, support this statement, with the conclusion that prices fluctuate randomly and hence cannot be predicted.

More recent literature however, opposes these conclusions. In a review of the profitability of technical analysis, Park and Irwin [50] find that, of 95 modern studies: 56 find positive results, 20 obtained negative results, and 19 indicated mixed results.

The technical trading system used to find these positive results consists of a set of trading rules. These rules generate long-term and/or short-term trading signals, according to various parameter values. Popular technical trading tools include trending strategies, filters and oscillators. Practitioners generally use them in a practise known as charting; plotting them with the actual price levels, to determine market trends. However, the information is equally, if not more valuable if employed as raw values [50]. These raw values are of interest, as they contain information about the strength, momentum and and direction in the trends of an assets price, thus making them useful statistics in characterising the current market state.

2.4.1 Different Methods of Technical Analysis

There are many different tools which fall under the umbrella of technical analysis, but this is not the primary concern of this study, and hence only five rules have been investigated in this thesis. These rules were chosen because of their proven usefulness

in papers such as Park and Irwin [50] and Sweeney [63], and also their diversity - capturing different views of the same data. The reasoning behind, and explanation of, each tool is discussed below.

2.4.1.1 Filter Rules

Filter rules encourage a trader to buy when an asset value rises a certain percentage above a previous local high, and sell if the price declines a certain percentage below a local low [20]. The raw values in a filter rules output the current price in relation to its local maxima and minima within a predefined time window. This proves a useful characterising statistic as it is able to identify the current price in relation to maxima and minima within the local window.

These were first introduced by Alexander in 1961, who reported returns significantly higher than a simple buy and hold strategy. Although this was refuted by Fama and Blume [20] It has been subsequently found to deliver significant profitability by both Sweeney [63] and Levich and Thomas [31].

2.4.1.2 Trending

Trending generates trading signals at the intersection of a short- and a long-run moving average. A buy signal is indicated when a the short-run moving average intersects the long-run moving average from below, indicating that the asset value is trending upwards. The converse is true for sell signals [43].

Trending has also seen mixed success when tested empirically as a trading rule, but Levich and Thomas [31], Pukthuanthong-Le, [54] and Okunev and White [47] find that trending is profitable to a significant level. Trending is identified as a useful characterising statistic because it is sensitive to the changes in the markets current trend.

2.4.1.3 Relative Strength Index (RSI)

The Relative Strength Index (RSI) indicates the Relative Strength of an asset as a value between 0 and 100. If the RSI value is low, it indicates that an asset has been over-sold, and thus it is a good time to buy the asset. If the RSI value is high, it indicates that an asset has been over-bought, and thus it is a good time to sell the asset [32]. RSI is a good characterising statistic because of its different perspective of the time-series - using oscillators to identify current value against expected values

Figure 2.1: *An example of the RSI rule. This example is useful in that it shows how RSI could be used to gain profit, and how it could lose money. If the second set of buy signals were used, and the only set of sell signals were used - significant profit is gained. However, if the first set of buy signals are used no significant increase in price between the two is evident. This shows the need to identify when different signals should and should not be listened to. Used with permission [48]*

Levy [32] finds success in using RSI to predict opportune points to buy and sell in the stock market. This is exemplified in Figure 2.1.

2.4.1.4 Moving Average Convergence / Divergence (MACD)

The Moving Average Convergence / Divergence (MACD) shows the difference between a fast and slow exponential moving average (EMA) of an asset's closing price information.

MACD is used to identify changes in duration, direction, strength and momentum in the trend of an assets price. This makes it a useful characterising statistic as it highlights changes in trends.

The mechanics of MACD are thus: two moving averages are compared with one another, to produce the MACD (typically the 12 and 26 period moving averages). The MACD is then compared to a *signal* line, which is also an EMA of the price (typically a 9 period moving average). Once the two are compared, the result is treated in a similar fashion to that of trending, however it is . A buy is signalled when the MACD line intersects the signal line from below, and a sell is signalled when the MACD line intersects the signal line from above [53]. This can be seen in Figure 2.2. Little empirical research has been done on the efficacy of this rule, but it is has been suggested as

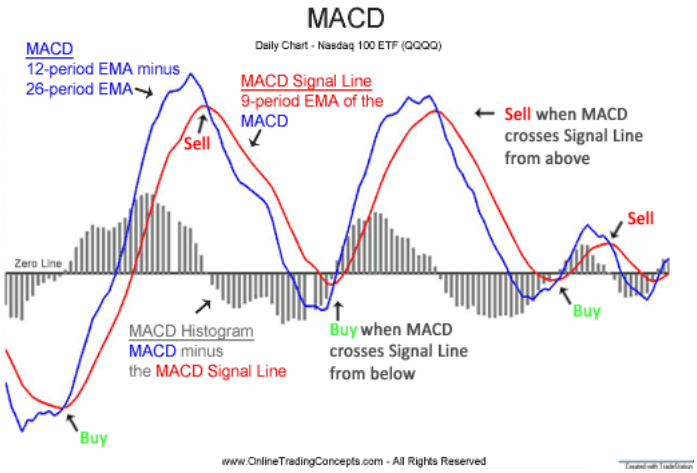

Figure 2.2: *An example of the MACD rule. This shows significant profits can be gained by buying when the MACD line intersects the Signal line from below, and selling when it intersects from above. This is because it identifies subtle changes in the trend of a market. Used with permission [48]*

a worthwhile technical trading rule [50].

2.4.1.5 Momentum (MOM)

Momentum (MOM) shows the momentum of the movements of an asset price. This is calculated by looking at the change in asset price from the current day against that of a few days prior. MOM is a useful statistic in characterising the current market state, as it gives an indication of the current market trend, and the rate at which it is following that trend.

MOM is also a useful technical trading rule. If the MOM returns a negative value, this means that the asset price is downward trending, which is signalled as a sell; whereas if the MOM is positive, the asset price has an upwards trend and is signalled as a buy [53]. An example of this can be seen in Figure 2.3. MOM was found to be a profitable indicator by Jegadeesh, and Titman [29].

2.4.2 Summary and Application

Although technical trading is frowned upon by academia, practitioners advocate the above rules to be profitable. The rules can be useful as signals, as well as raw values, and these rules are mechanical in nature, and hence can be programmed on a computer.

Above and beyond the possibilities of profitability, these rules also give insight into

Figure 2.3: *An example of the MOM rule. The raw value shows the current direction and rate of the asset price movement. As a technical trading rule, MOM indicates buys when crossing the x-axis from below, and sells when crossing from above. Used with permission [48]*

current market trends which is useful in characterising the market at any given time.

2.5 Other Characterising Statistics

Technical Analysis provides good insight into possibly profitable times to trade, however there are many other statistics which provide insight into the state of the market at any given time. This section identifies a number of key characterising statistics which will prove useful in creating an information rich feature space.

2.5.1 Sample Entropy

Approximate Entropy (ApEn) which was developed by Pincus [52] to measure regularity and system complexity in time-series data. More recently, ApEn has been applied to financial time-series analysis to quantify the degree of market efficiency and the predictability of future market price [46, 17, 51]. Pincus and Kalman [51] found that a drastic change in the ApEn may foreshadow substantial changes in the state of the market, and substantiated this by identifying such phenomena before the market crash represented by Black Monday and the Asian Currency Crisis.

Sample Entropy (SampEn) is an extension of ApEn, and was originally developed by Richman and Moorman[55] for the analysis of physiological time-series. SampEn has fewer dependencies on parameters, and proves more accurate on known and generated test cases and is seen as an improvement on ApEn.

Sugisaki and Ohmori [62] extended SampEn to create Variable Weighted Sample Entropy (FSampEn) to allow for time-varying systems. FSampEn was successfully applied to financial time-series and confirmed that a rapid increase in SampEn may foreshadow market crashes.

A time-series with a high degree of randomness will have a much higher SampEn than one which has a low degree of randomness. This is the basis behind SampEn as a measure of the market efficiency and stability[46].

As the system is to be developed and tested in fixed time windows, SampEn is explored further as FSampEn is unnecessarily complex for this application.

2.5.2 Volatility

Volatility is the standard metric for the risk of an investment [9]. If the volatility is high, one can expect large gains or losses by investing in the asset.

Subsection 2.6.2 explains the concepts of conditional heteroskedasticity, in particular the idea that there are volatility clusters within financial time-series, and that volatility changes over time. This is also evident by looking at Figure 2.5. As volatility varies, one needs to keep track of it in order to know how risky the current asset is. This makes the current volatility of an asset a very useful statistic to know.

2.5.3 Sample Variance

Similar to volatility, the sample variance is also concerned with the anticipated deviations. Volatility is concerned with the deviations in the log returns, where as sample variance is concerned with the deviations in the assets price.

2.5.4 Multi-order Derivatives

People are much better at identifying patterns than computers. Thus it makes sense to mathematically characterise the current trend of the asset. This can be done by looking at the rate of change of the asset price, or the first order derivative of an asset price.

The first order derivative tells us of the 'velocity' of the asset price - the rate at which it is currently increasing or decreasing. One may also be interested in the rate at which the velocity is changing; the 'acceleration'. Acceleration is the second order derivative and would be useful in realising that the asset price is approaching a peak (when acceleration becomes zero).

In rare cases, one might want to know if the asset price is accelerating at an accelerating rate, requiring the third order derivative, however any further than that the values become too noisy to be useful.

2.5.5 Summary

Sample Entropy is used to estimate the randomness and predictability of a time-series and may have the ability to foreshadow market crashes . Volatility is a measure of risk in a time-series which allows one to estimate the possible extent of future gains and losses. Multi-order derivatives are able to give a picture about the shape and trend of the time-series by using single sets of numbers.

All of the above are useful statistics for characterising the current state of the time-series.

2.6 Generative Models

When testing the aforementioned technical trading tools, they are often bench-marked on generative models as well as on real-world data [50]. This chapter provides an introduction into two generative models: random walks, and GARCH/ARMA. Both of these models will be used as test data for the final system as they both have interesting characteristics.

2.6.1 Random Walks

A *random walk* is defined as the sequence of partial sums ($\mathcal{S}_n$) of independently and identically distributed (i.i.d) random variables (X).

$$\mathcal{S}_n = \sum_{i=1}^{n} X_i$$

The study of random walks date as far back as the sixteenth century, and have been studied by some of the greatest mathematicians of all time [26]. Their popularity in economics dates back to 1964 in which Cootner's book *The Random Character of Stock Market Prices* proposed the *Random Walk Hypothesis* (RWH), which states that the future path of a stock is no more predictable than a random walk [10]. Although the random walk hypothesis has been rejected many times since its proposal, random walks can still provide some use. They are often used as test data in order to identify

whether a system is truthful or not, as one should not be able to generate profit on random walks.

2.6.2 Generalised Autoregressive Conditional Heteroskedasticity (GARCH)

In econometrics, AutoRegressive Conditional Heteroskedastic models (ARCH) as proposed by Engle[14], are often used to model and characterise variance in financial time series. They were developed to overcome the general assumption of a constant one-period forecast variance. Thus in an ARCH model, the recent past provides information about the forecast variance, and this changes with time.

Bollerslev generalised the ARCH model to form the GARCH model in 1986 [2]. This was done to reduce the computation required by the model, but reducing the number of parameters. GARCH also offers more flexibility in that the the current variance can be conditioned on a number of previous days, rather than just the day before [2]. GARCH models differ to the popular Exponentially Weighted Moving Average (EWMA) models [56], because GARCH has an extra term for mean reversion. This creates some persistence to the variance, so that if it strays too far from a long run average, it will be drawn towards back towards the long run average by the extra term.

2.6.2.1 Understanding GARCH

The GARCH equation is as follows:

$$\sigma_t^2 = \kappa + \sum_{i=1}^{P} G_i \sigma_{t-i}^2 + \sum_{j=1}^{Q} A_j \varepsilon_{t-j}^2 \tag{2.1}$$

Breaking down the complex name into individual terms makes it easier to understand the process and the benefits it produces.

Heteroskedasticity Heteroskedasticity is a statistical term which describes the situation in which a sequence of random variables have different variance. In other words, the variance in the generative model changes over time, and thus produces some samples which vary wildly, and some samples which are relatively stable. This is exhibited in Equation 2.1, by the fact that σ^2 changes over time.

Conditional The conditional aspect of the GARCH model is suggestive that the system is conditional on the past. Thus today's observations are functions of yesterdays observations. This is characterised by the third term in Equation 2.1, as the current variance is conditional on the past innovations, or past realisations of the variance, ε.

Autoregressive The Autoregressive component is the feedback mechanism which links the past to the present. Thus given the above, the variance of today's observations is a function of the variance of yesterdays observations (amongst other terms). This is what enforces the volatility clustering, as previous spikes in volatility are used to construct future volatility.

Generalised As mentioned above, Bollerslev generalised the ARCH model in order to incorporate more than one days lagged observations, and to also produce a more parsimonious model. This is evident in the summing over a number of days. The variables P and Q are used to describe the number of lagged days are taken into account, and thus GARCH models are written as $GARCH(P,Q)$ to show the time dependencies. The other addition GARCH brought to ARCH was the influence of past variances as well as sample variances (the innovations). Thus a $GARCH(P,Q)$ model where $P = 0$ is the equivalent of an $ARCH(Q)$ model.

2.6.2.2 Why is GARCH important?

GARCH takes two of the most important characteristics of financial time series into account: excess kurtosis and volatility clustering [2]. Excess kurtosis describes the probability distribution from which innovations (the difference between the previous and current asset price) are drawn. Financial time series appear to be drawn from distributions which have fatter tails than a the standard Gaussian distribution, as illustrated in Figure 2.4.

Financial time series also show volatility clustering [9]. This means that periods of high volatility tend to follow periods of high volatility, and vice versa. Thus even though the direction of these changes is unpredictable, and uncorrelated volatility clustering suggests that successive disturbances are serially dependent. This volatility clustering can be observed in Figure 2.5.

Through their ability to model time-varying conditional variances, GARCH models provide accurate forecasts of variances and covariances of asset returns. This makes GARCH models applicable to not only single asset returns, but a wide array of financial

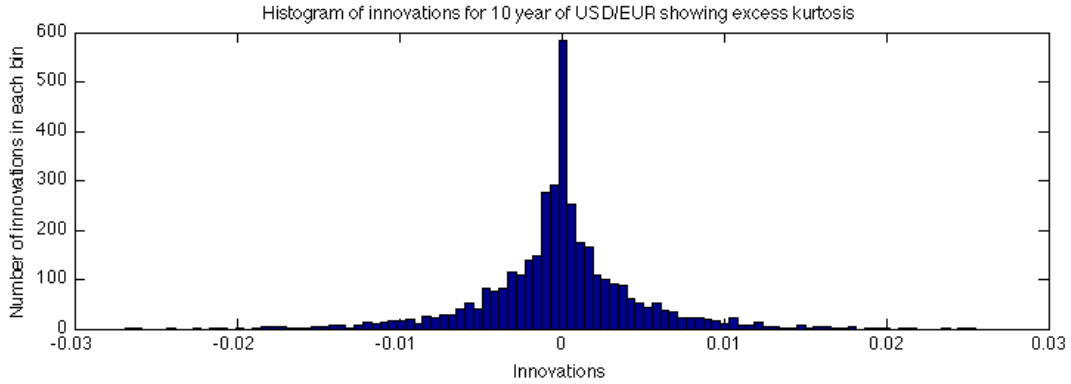

Figure 2.4: *A histogram of ten years worth of innovations between the US Dollar and the Euro. This clearly shows excess kurtosis (fat tails).*

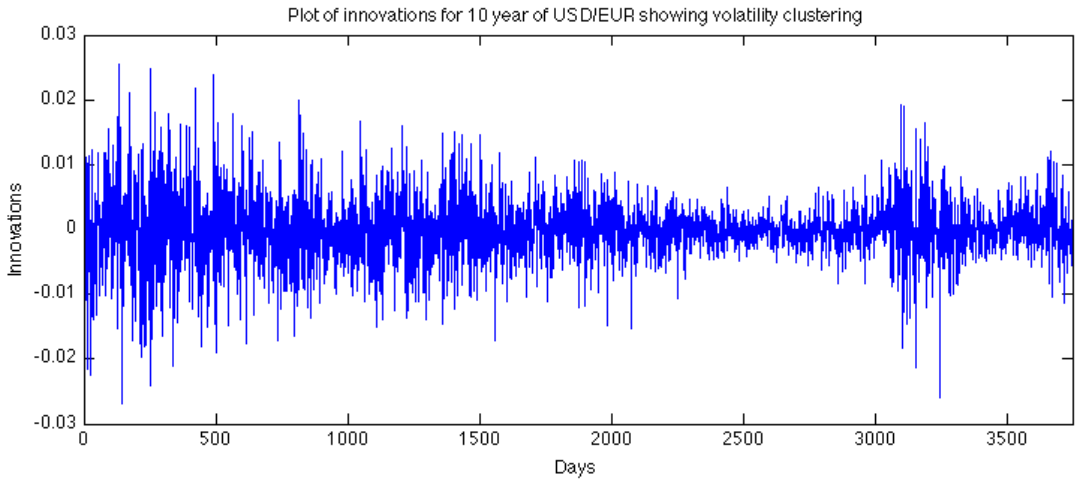

Figure 2.5: *A plot of ten years worth of innovations between the US Dollar and the Euro. This clearly shows volatility clustering in the noisy areas and areas of relative calm.*

systems from portfolios of stocks to equity futures markets. These effects are important many different fields, from risk management to asset allocation and term structure of interest rates. Foreign exchange markets which couple exhibit persistent periods of volatility clustering and excess kurtosis are particularly well suited for GARCH modelling [3].

Engle's survey observed that 'most investigators have found that the amazing $GARCH(1,1)$ model is a generally excellent model for a wide range of financial data' [15], and Bollerslev *et al.* detailed several dozen successful applications of $GARCH(1,1)$. Thus $GARCH(1,1)$ is the specification used in the application of this research.

2.6.2.3 Limitations to the GARCH model

GARCH models are not always able to fully capture the excess kurtosis in innovations. The heteroskedasticity accounts for a large portion of this but typically not all of it [67].

GARCH models operate best under stable market conditions [**?**], and although they are designed to capture fluctuations in variance over time, they are known to fail in capturing highly irregular activity such as market crashes and regime switches [27].

GARCH only models volatility - although useful, this is not enough information to base financial decisions on.

2.6.3 Autoregressive Moving Average (ARMA)

As GARCH models the variance of a financial system as it changes over time, ARMA models the actual price movements [4]. The combination of ARMA and ARCH like models have been shown to accurately characterise and model financial time series [67].

2.6.3.1 Understanding ARMA

$$y_t = C + \sum_{i=1}^{R} AR_i y_{t-i} + \varepsilon_t + \sum_{j=1}^{M} MA_j \varepsilon_{t-j} \tag{2.2}$$

Similar to the GARCH model, the name breaks down the function into its various components: Autoregressive (AR), Moving Average (MA) with Regression (X).

Autoregressive The autoregressive component suggests that today's price y_t is a function of a long run constant C, new innovations ε_t, and the past R days prices, $\sum_{i=1}^{R} AR_i y_{t-i}$. This links the present price to that of the past in an autoregressive manner, such that the price today is going to be similar to the price yesterday [4]. The AR model makes use of the first three terms in Equation 2.2 above.

Moving Average The moving average component is a function of the innovations ε. It suggests that the price is to be affected by the current innovation ε_t, but also the last M innovations, $\sum_{j=1}^{M} MA_j \varepsilon_{t-j}$. This helps the model with trending - moving in a similar direction for some time before changing direction [4]. The MA model makes use of the third and fourth components of Equation 2.2 above.

2.6.4 Summary

Random walks have been researched since the early sixteenth century. They prove useful as test data in this study, as there should not be any underlying structures detectable in a random walk model.

GARCH is a powerful tool for modelling the variance in financial time-series, while ARMA models the price movements in a financial time-series. Both GARCH and ARMA can be used to characterise the variance and price movements of a given time-series, but also to generate them. The variance of the innovations in an ARMA model is generally modelled using a GARCH model, and thus by combining the two one is able to either model or generate realistic financial time-series. Literature suggests that $GARCH(1,1)$ and $ARMA(1,1)$ models are sufficient for most applications. These models are used as test data in this study because of the properties of volatility clustering and trending – both characteristics of real-world data.

2.7 Genetic Algorithms

Genetic Algorithms (GA) are a popular meta heuristic optimisation method. Meta heuristics operate by iteratively improving candidate solutions with few assumptions about the problem, but cannot guarantee optimality [36]. GAs were first developed by Holland and colleagues in the 1960's and 1970's at the University of Michigan [40]. His initial intentions were to explain the adaptive processes exhibited by natural systems, and to harness these techniques to solve engineering problems [6].

This meta heuristic is most commonly used to optimise real valued objective functions and combinatorial problems. The terminology used to describe GAs follows their biological inspiration, and thus the parameter to be optimised (a bitstring or combinatorial string) is called the *chromosome* [23]. A *population* is comprised of a number of candidate solutions, each of which is represented as a chromosome. Genetic operators of selection, *crossover* and *mutation* are applied to successive *parent* populations in order to create a *child* population. In the standard case, this results in two parent chromosomes being selected based on some *selection* criteria, and recombined in order to form two child chromosomes. The children are then mutated in order to provide some genetic variation. The process of selection, recombination and mutation is repeated to successive child populations until a *termination* criteria is met - generally a maximum number of epochs being reached or a lack of improvement in the best performing solu-

tion [40]. By selecting parents based on their performance, and performing operations on their chromosomes, similarities between high-performing chromosomes can be exploited. The term for the performance of an individual chromosome is its *fitness* and the better it performs the fitter it is. More information about Genetic Operators can be found in Appendix A.

2.7.1 GAs as a means of parameter optimisation

Genetic Algorithms have been successfully used in a wide variety of optimisation tasks, including numerical and combinatorial optimisation problems, circuit layout and scheduling tasks [40]. De Jong [13] finds that genetic algorithms out-perform local search algorithms, especially in multimodal and noisy search spaces. In the context of this research, GAs have been used successfully in the optimisation of technical trading tools by: Allen and Karjalainen [1], Neely, Weller and Dittmar [42], Shazly and Shazly [61], Chen [8] and Li Lin, Cao, Wang and Zhang [33] to name a few.

2.7.2 Summary

Genetic Algorithms are a meta heuristic optimisation method which mimics natural selection in order to find good solutions. They encode the parameters of an objective function into a chromosome, and perform genetic operators of selection, crossover and mutation in order to improve a population of candidate solutions. They cannot guarantee an optimal solution, but require no knowledge of the objective function, and are known for successfully finding adequate solutions in noisy and disjointed functions.

GAs have been used successfully for many optimisation tasks, including computational finance, and thus are deemed an appropriate method of optimisation for the combinatoric problems faced.

2.8 Summary

This chapter provides a brief overview of the current state of research into structures of financial time-series. It is found that although most efforts are towards finding correlation matrices between financial series, clustering techniques are also a fairly common method of extracting information from the time-series in an unsupervised fashion. A brief overview of agglomerative hierarchical clustering was presented, followed by an introduction into the Efficient Market Hypothesis.

The bulk of this introduction to concepts is spent identifying ideal statistics which can be used to characterise the market state, as is the first objective of this project. After discussing methods from technical analysis which can be used to capture different perspectives of the current market state, other characterising statistics were investigated.

Some time was also spent on the generative models which will be used to test the strategies created. GARCH was paid a particularly large amount of attention because it is instructive of many phenomena present in financial time-series.

Finally a brief overview of genetic algorithms was presented, as it has proven very useful in the past for optimising autonomous trading systems.

Chapter 3

Creating an Information-rich Feature Space

This chapter addresses the first objective of the project: the design and implementation of characterising statistics identified in Chapter 2. These are used to create an information-rich feature space in which unsupervised learning can occur. Following the literature review in Chapter 2, this chapter splits these characterising statistics into two groups: Technical Trading Tools as used in Technical Analysis, and Other Characterising Statistics employed in literature.

3.1 Technical Trading Tools

Section 2.4 justified the usefulness of technical trading tools, and explained the way in which investors often use them as guide for when to buy, hold or sell an asset. For these reasons, they are expected to provide differing and interesting perspectives into the current state of the market.

Each tool requires a number of parameters as well as the asset price information. Each tool requires the asset price information as well as a number of other parameters in order to perform their analysis. These analyses result in sets of buy, hold, and sell signals. This section details the mechanical nature of each of the trading rules, indicates the parameters they require, and illustrates how they were implemented.

3.1.1 Filter Rules

As stated in Section 2.4.1.1, filter rules encourage a trader to buy when an asset value rises by a certain percentage above a previous local high, and sell if the value declines by a certain percentage below a local low.

This is implemented by searching in a given *window size* before each data point for local maxima and minima in that interval. The maxima and minima are then stored in an indexed array of the same size as the data. The data at each point is then compared to the indexed maxima and minima for each point.

A *filter size* is then chosen as the threshold by which a data value needs to exceed its local boundary point. If the data value is higher than its local maxima by a factor of *filter size*, it becomes a buy signal; and if it is lower than its local minima by a factor of *filter size*, it becomes a sell signal. For example: if the *filter size* was 10%, the data point would have to be 10% higher than a local maxima to register as a buy signal. The recommended value for the *window size* is 15 days [50].

3.1.2 Trending

As previously discussed in Section 2.4.1.2, trending generates trading signals at the intersection of a short- and a long-run moving average. A buy is indicated when the short-run moving average intersects the long-run moving average from below, indicating that the asset value is trending upwards. The converse is true for sell signals.

These signals are generated by the distance between the two moving averages. The closer they become, the stronger the signal. If the short-run is above the long-run, and they are close together, then a sell signal is generated. If the short-run is below the long-run, then a buy signal is generated. The recommended values for the window sizes of the long- and short-run moving averages are five and 20 days respectively [50].

3.1.3 Relative Strength Index (RSI)

Subsection 2.4.1.3 noted that RSI indicates the Relative Strength of an asset as a value between 0 and 100. If the RSI value is low, it indicates that an asset has been over-sold, and thus it is a good time to buy the asset. If the RSI value is high, it indicates that an asset has been over-bought, and it is thus a good time to sell the asset.

This is calculated by first finding the upward and downward changes.

Let Up be the set of all upward changes and $Down$ be the set of all downward

changes described by Equations 3.1, 3.2 and 3.3.

$$change = close_t - close_{t-1} \tag{3.1}$$

$$Up = change \tag{3.2}$$

$$Down = -change \tag{3.3}$$

The ratio of the averages is termed the *Relative Strength* (*RS*), and is calculated as shown in Equation 3.4, where $EMA(n)$ is the Exponential Moving Average with an n-day smoothing factor.

$$RS = \frac{EMA(n)\,of\,Up}{EMA(n)\,of\,Down} \tag{3.4}$$

This is then converted to a *Relative Strength Index* (*RSI*) between 0 and 100 to be interpreted as buy and sell signals if above or below certain thresholds, as shown in Equation 3.5.

$$RSI = 100 - \frac{100}{1+RS} \tag{3.5}$$

This can be done in a single step, as the RSI expresses the upward movements as a proportion of the total upward and downward movements. It can also be modified further to convert the numbers to an index between -1 and 1 as illustrated in Equation 3.6.

$$RSI = 2 * \frac{EMA(n)\,of\,Up}{(EMA(n)\,of\,Up) + (EMA(n)\,of\,Down)} - 1 \tag{3.6}$$

This produces buy and sell signals based on the RSI of the asset value. The recommended value for n is five days [50].

3.1.4 Moving Average Convergence Divergence (MACD)

As mentioned in Section 2.4.1.4, the MACD shows the difference between a fast and slow Exponential Moving Average (EMA) of an asset's closing price information. Two moving averages are compared to each other: an s-day short run moving average, and an l-day long run moving average. The distance between the two is the *MACD*, calculated as shown in Equation 3.7.

$$MACD = EMA(s)\,of\,price - EMA(l)\,of\,price \tag{3.7}$$

This must then be compared with the signal line, calculated as indicated in Equation 3.8, which is an m-day moving average of the *MACD* line.

$$signal = EMA(m)\,of\,MACD \tag{3.8}$$

Trending is then applied to the *MACD* and *signal* values. Thus a buy is signalled when the *MACD* line intersects the *signal* line from below, and a sell is signalled when the *MACD* line intersects the *signal* line from above. The recommended values for s, l, and m, are 12, 26 and 9 respectively [50].

3.1.5 Momentum (MOM)

Section 2.4.1.5 explained that MOM shows the momentum of an asset price. This is calculated by looking at the change in asset price from the current day against n-days ago. This is calculated as shown in Equation 3.9.

$$momentum = close_t - close_{t-n} \tag{3.9}$$

If the *momentum* returns a negative value, this means that the asset price is downward trending which is signalled as a sell., If the *momentum* value is positive, it means that the asset price has an upwards trend and this is signalled as a buy. The recommended value for n is 12 days [50].

3.1.6 Summary

In this subsection, the mechanical nature of the technical trading tools is highlighted by describing the manner in which they are implemented in the system.

Each of these tools uses a number of parameters as inputs, and outputs a set of continuous values which characterise the current market state. These signals could independently act as buy, hold and sell indicators by thresholding[1] them. In general, a large, negative number represents a strong sell signal, while a large positive number represents a strong buy signal.

[1]Thresholding is when a threshold, or decision boundary is used to convert a continuous value into a discrete class.

3.2 Other Characterising Statistics

Coupled with outputs from the technical trading tools described above, the statistics mentioned in this section will be used to characterise the current market state.

3.2.1 Sample Entropy

Sample Entropy (SampEn) has recently been used as a measure of market efficiency by detecting the level of randomness or predictability in a time-series [46]. This makes SampEn a useful statistic as it gives insight into the randomness of the current market state.

SampEn is defined as follows[55]:

$$SampEn(m,r,N) = -ln\frac{\sum_{i=1}^{L-1}\sum_{j=i+1}^{L-1} s_{i,j}^{m+1}}{\sum_{i=1}^{L-1}\sum_{j=i+1}^{L-1} s_{i,j}^{m}} \tag{3.10}$$

where

$$\begin{aligned} s_{i,j}^{m} &= \begin{cases} 1 & d[\mathbf{x}_m(i),\mathbf{x}(j)] \leq r \\ 0 & d[\mathbf{x}_m(i),\mathbf{x}(j)] > r \end{cases} \\ d[\mathbf{x}_m(i),\mathbf{x}(j)] &= \max_{k\in 0,m-1} |u(i+k)-u(j+k)| \end{aligned}$$

Where N is the length of the time series, m is the length of the sequences to be compared, and r is the tolerance for accepting matches. $\{u(g) : 1 \leq g \leq N\}$ forms the $N-m$ vectors $\mathbf{x}_m(i)$ for $\{i|1 \leq i \leq N-m\}$, where $\mathbf{x}_m(i) = \{u(i),\cdots u(i+m-1)\}$ is the vector of m points.

A time-series with a high degree of randomness will have a much higher SampEn than one which has a low degree of randomness.

Values used for N, m and r were those suggested by Costa, Goldberger and Peng [11], such that $N = 300$, $m = 2$ and $r = 0.25 \times \sigma$ (where σ is the standard deviation for the time-series).

3.2.2 Volatility

Volatility is often used as a measure of risk in finance, and is calculated as the standard deviation of the daily periodic log returns, where the daily periodic log returns are calculated as:

$$R_t = \ln\left(\frac{y_t}{y_{t-1}}\right)$$

Thus the volatility is calculated as:

$$\sigma_t = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(R_{t-i} - \bar{R})^2}$$

Volatility is known to changes over time, and is also known to have some auto-correlation. This suggests that tomorrow's volatility is some function of today's volatility. This makes the volatility of the current market state valuable to know, as it is indicative of the current risk of investing in the market, and some loose indicator of the future risk of investing in a market.

A high volatility indicates large increases and decreases in the recent history of the market. Volatility clusters in financial time-series, thus a high volatility is indicative of future risk.

3.2.3 Sample Variance

Sample variance is calculated as the

$$s^2 = \frac{\sum_{i=1}^{n}\left(y_{t-i}^2\right) - \frac{\left(\sum_{i=1}^{n} y_{t-i}\right)^2}{n}}{n-1}$$

3.2.4 Multi-order Derivatives

Multi-order derivatives can be used to provide a rough picture of the shape of a curve in the immediate vicinity. They describe the curve's direction, the rate at which it is moving, and the speed at which that rates changes.

The first derivative is calculated as $m = \frac{\Delta y}{\Delta x}$. However, as the time steps are fixed at 1, the derivative can be calculated as $m = \Delta y$, where $\Delta y = y_t - y_{t-1}$. In the current application, multiple currencies are investigated, each of which has a different rate of exchange, and some of which differ by an order of magnitude. The prices within a single currency also fluctuate, sometimes growing or shrinking by 50% within a single year. It therefore makes sense calculate Δy as the geometric difference, rather than the arithmetic difference. This records the changes in percentages (making the changes relative), rather than as the absolute change in value. Thus the first derivative of the asset price is calculated as:

$$y'_t = \frac{y_t}{y_{t-1}}$$

If the rate of change is positive: $(y'_t > 0)$, it implies that the asset price is increasing, and vice versa. This is not the only useful rate to know, as one has to examine a series of derivatives in order to ascertain whether the asset price is changing at an accelerating or decelerating rate. Thus it is useful to know the second derivative of the asset price, calculated as:

$$y''_t = \frac{y'_t}{y'_{t-1}}$$

This process can be repeated many more times to get even higher order derivatives, but the usefulness of such a statistic diminishes quickly, as the higher the order statistics become increasingly noisy.

Asset prices are inherently stochastic, moving up and down from minute to minute. This makes the rates of change based on the raw values very noisy. Derivatives are therefore calculated from an exponential moving average of the asset price in order to reduce the presence of noise.

3.3 Multiple Parameter Settings

The parameters described above are based on those recommended by practitioners. However, these are not necessarily optimal for all time-series. This was empirically proven by Lin, Cao, Wang, and Zhang [33] where genetic algorithms were successfully used to optimise parameters of technical trading rules to near optimal values found through an exhaustive search. Both of these methods produced profits almost four times greater than when set manually.

An intuitive take on this is that one would rather use a longer moving average in a highly variant system, as this provides much-needed stability for the erratic behaviour of the time-series. Adversely, this would slow down the reaction speed of the trading tool because current changes would take longer to reflect due to the moving average.

Conversely, in a stable system with low variance less of a smoothing effect is needed. Shorter moving averages would therefore be beneficial because the trading tools would have a faster response rate. This is not only relevant for different time-series but also within a single time-series. As described in Subsection 2.6.2, financial time-series are known to be heteroskedastic (i.e. they have variances which change

over time). From the arguments presented above, it becomes clear that it is necessary to provide the trading rules with options as to the parameters which work best in the current situation.

Therefore, two *additional* sets of characterising statistics are generated from the same rules but with different parameters: a shorter set, targeted towards stable series and periods; and a longer set, targeted towards unstable series and periods. It should be noted that the different time periods will also capture different aspects of the market state in short-, medium- and long-term perspectives.

The exact parameters used for all forms of the trading rules can be found in Appendix B.

3.4 Creating the Feature Space

The first objective of this project is the creation of an information-rich feature space in which structure can be found. The aforementioned characterising statistics were shown to be information-rich in the current context, and all that remained was to combine them to form a joint feature space.

A simple approach was taken in creating the feature space: each trading rule and statistic was treated as its own dimension in Euclidean space. Given the above trading rules and descriptive statistics, and generating them for perspectives in the short-, medium- and long-term, the feature space becomes a sizable 31 dimensions. This means that a single entry in time has be transformed from being a single price to a point which contains information from many perspectives with many time horizons - truly an information-rich feature space.

3.5 Summary

This chapter explained the methodology used to create each of the descriptive statistics, and provided an indication of what these statistics represent. Each of the statistics requires parameters of a time-window, over which the statistics can be generated.

...

Each of the statistics require a time-window length parameter. In order to characterise the market state in short- medium- and long-term perspectives, a number of time frames were chosen for each.

The statistics were combined to form an information-rich feature space by treating each as its own dimension in Euclidean space. The result is a sparse, but information-rich representation in which a single point is described by over 30 different statistics, providing much information in which unsupervised learning techniques can be applied.

Chapter 4

Finding Structure through Unsupervised Learning

This chapter investigates structures within the feature space defined in Chapter 3. This involves taking a single time-series of asset prices, and expanding it into a joint feature space containing a number of interesting statistics about the time-series within which structure might be found. Loosely speaking, this is similar to the kernel trick, in that the time-series is mapped into a higher dimensional space in which machine learning techniques can be applied with greater success than on the time-series itself.

4.1 Exploring the feature space

The feature space was created by assigning each of the descriptive statistics their own dimension in Euclidean space. This makes the Euclidean distance between points an ideal metric for exploration. The Euclidean distance between two points, x and y in n dimensions is calculated as follows:

$$d(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\| = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Six years of USD/EUR price indices were compared with a number of variations of the same series in order to test whether the Euclidean distance was sufficient for identifying similar positions. A number of variations were generated based on this data. Figure 4.1 shows all of the paths and their distances from the original path. These distances were calculated by adding the Euclidean distances between each point of the index in question and the original series in order to calculate the total distance.

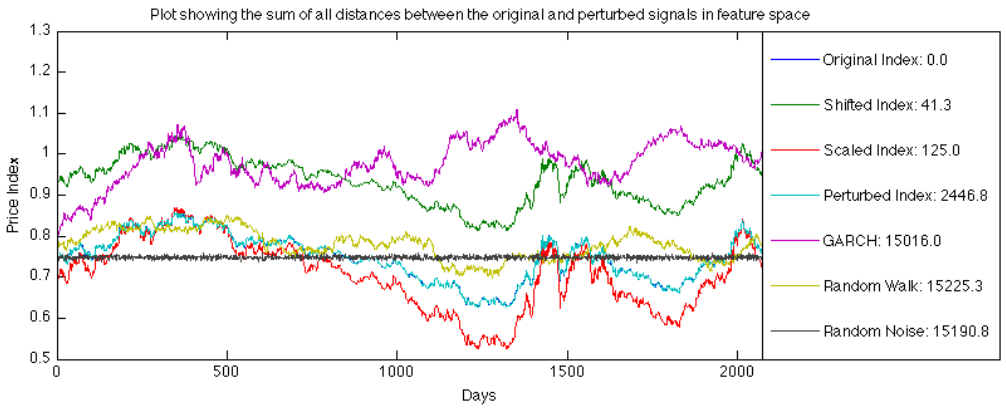

Figure 4.1: *Plot showing the sum of all distances between six years of USD/EUR, the original index but shifted upwards, the original index but with scaled innovations, the original indices which have been perturbed by adding Gaussian noise to each point, a GARCH model with parameters learnt from the original index, and a random walk and random noise both with the same standard deviation as the original series' innovations.*

It is evident that the original index is not going to vary when compared to itself, and thus has a distance of 0.

In order to create baselines for comparisons, two metrics were used. The first was created by comparing the original line to a horizontal line located at the mean of the original index.

The second was created by taking the additive inverse $(-y_t)$ of the series, yielding a series with exactly the opposite trends at every point in time. This should generate signals opposite to those of the original index which would be close to the maximum distance for this system.

Interestingly the difference between the original and the mean is 10140. This is seen as a bench mark, in that if the distance is lower than 10140, the two series share similarities, while if it is over 10140 the series exhibit large differences in trends, tending towards the additive inverse. The additive inverse has a distance of 18174 from the original data, indicating that 18174 is approaching the maximum distance from the original sequence.

The *shifted index* was created by adding an arbitrarily chosen constant to the original index. In this case the constant was $0.25 \times \bar{y}$ (20% of the mean for the series). The distance between the shifted and original indices is only 41. Bearing in mind that there are over 2000 points in over 30 dimensions being compared, this is a very low number. This result is not surprising, because the majority of the characteristic statist-

ics are calculated geometrically in order to render them indifferent to actual values but sensitive to relative change.

The *scaled index* is created by differencing the original data ($i_t = y_t - y_{t-1}$) in order to get the price innovations (also known as daily movements). These innovations are then magnified by 50%, making each increase and decrease 50% larger. The series was then recreated by taking the cumulative sum of all of the differences, and adding the original price on the first day to each point. Once again the total distance is fairly small (125) which is a good indicator that the similar points describe similar situations. Although the variances and volatility may differ somewhat, the trends are identical in the short-, medium- and long-terms.

The *perturbed index* is generated by adding some Gaussian noise to each point in the original index. The noise was generated from a normal distribution which had a standard deviation of 25% of the standard deviation of the innovations. This made the perturbed series much noisier in the short-term which can be seen by the large increase in its distance from the original series. This change leaves the medium- and long-term statistics relatively similar, but creates havoc in the short term statistics and the sample entropy.

The Generalised AutoRegressive Conditional Heteroskedastic (*GARCH*) index was created using a simulation. The coefficients for the GARCH model were learnt from the original data using maximum likelihood estimates. These coefficients were then used to generate a time-series with variances similar to those in the original data. It is immediately evident that the differences are orders of magnitude larger than the shifted and scaled versions, suggesting that large distances do exist.

The *random walk* and *random noise* indices were generated from a Gaussian distribution with mean 0 and the same standard deviation as the innovations in the original data. Both exhibit numbers similar to the GARCH model, showing the difference when similar random numbers are generated, but with none of the same trends.

4.1.1 Summary and conclusions

In this section, the points in feature-space generated by a number of different time-series were examined. All of these time-series had some similarities to an original time-series, but were transformed in some way. The points of a number of generative models were also examined. By comparing the feature spaces of series obtained from transforms of the original to different types of random times-series, it was found that

points close to each other in the feature space, are representative of similar market conditions. Although not entirely impervious to transformation, they do still retain some similarity when scaled, shifted and perturbed.

These results show that when two time-series are in similar states, they have points near to each other in the feature space. Conversely, when two time-series have different states, their points in the feature space are far apart.

Extrapolating from these results, it was hypothesised that two points being near to each other in the feature space is indicative of similar market conditions in the time-series with generated these points.

4.2 Choosing an Unsupervised Learning Technique

Based on the findings and hypothesis above, an ideal unsupervised learning technique should identify groups of similar points in order to find areas of similar market conditions.

An autonomous trader could then make decisions based on the current state, by identifying ideal decision from past data that occupies a similar place in the feature space. This suggests that clustering techniques would be an appropriate area of unsupervised learning to explore.

A criticism of a number of unsupervised learning techniques is that their success is dependent on good parametrisation. Due to the nonlinear interaction between these parameters, it is seen as more of an art form than a science, requiring deep knowledge and experience with both the learning techniques and the data [57]. The purpose of this project is the exploration of an unknown feature-space – thus it was decided that a non-parametric approach would be most suitable. This rules out several methods that have assumptions as to the shape of the clusters (e.g. Gaussian Mixture Models trained through Expectation Maximisation). The exact number of clusters required is also unknown and would have to be explored, making methods such as k-means clustering unsatisfactory.

Hierarchical agglomerative clustering is a method which, when performed once, finds all numbers of clusters between 1 and n, where there are n points being clustered. The hierarchy is created by starting with each point as its own cluster. In every iteration, two clusters are merged in a way that minimises some distance metric. This creates all possible numbers of clusters in an efficient and nonparametric method by minimising the distance within clusters. This approach appears to satisfy all the con-

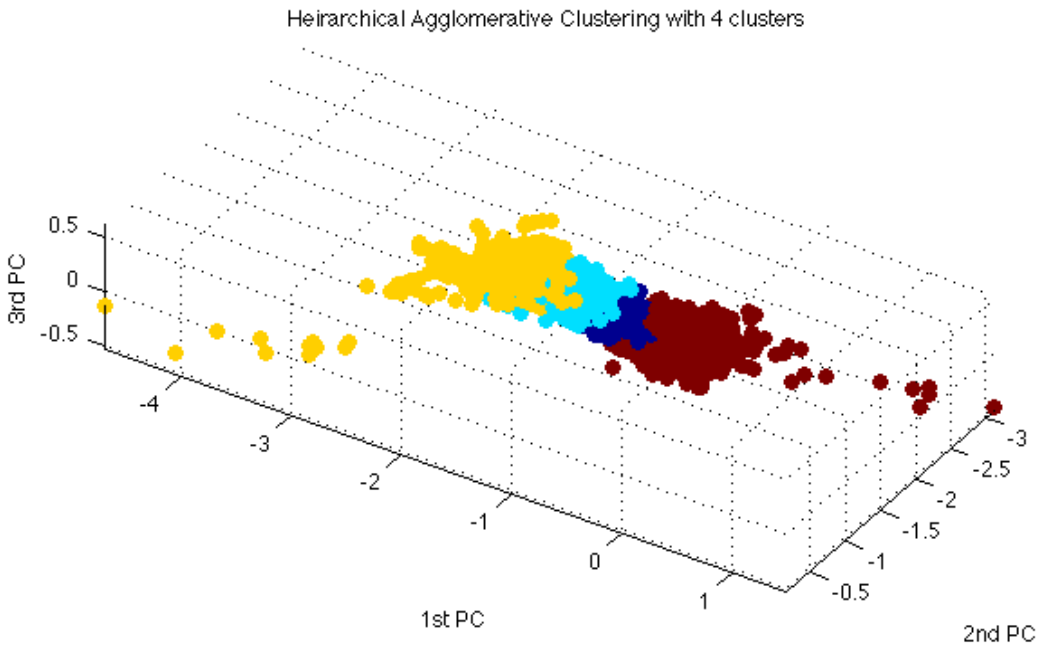

Figure 4.2: *Plot showing the first four clusters generated in 31 dimensions using Wards method on Euclidean distance. This is visualised by showing the first three Principal Components. Note the separation along the principal component.*

straints required for the current application.

4.3 Examining the Clusters

Clusters were created by using the hierarchical agglomerative clustering technique explained above, using Euclidean distance as the distance metric. Clusters of different size were visualised and select sizes are presented below to show notable features. This investigation was done data not present in the testing and training sets in order to mitigate against criticisms of data snooping[1].

Although the clustering is done within the high dimensional feature space, the dimensionality has been reduced down to three dimensions by Principal Component Analysis (PCA) in order to visualise the clusters.

Figure 4.2 shows an example of the clusters generated. A notable feature is that the clusters appear to separate the data along the principal component. When more clusters are used, they appear to break down the clustering along the second principal component as visualised in Figure 4.3.

[1]Data snooping occurs when one limits the data sets so that only favourable data is tested

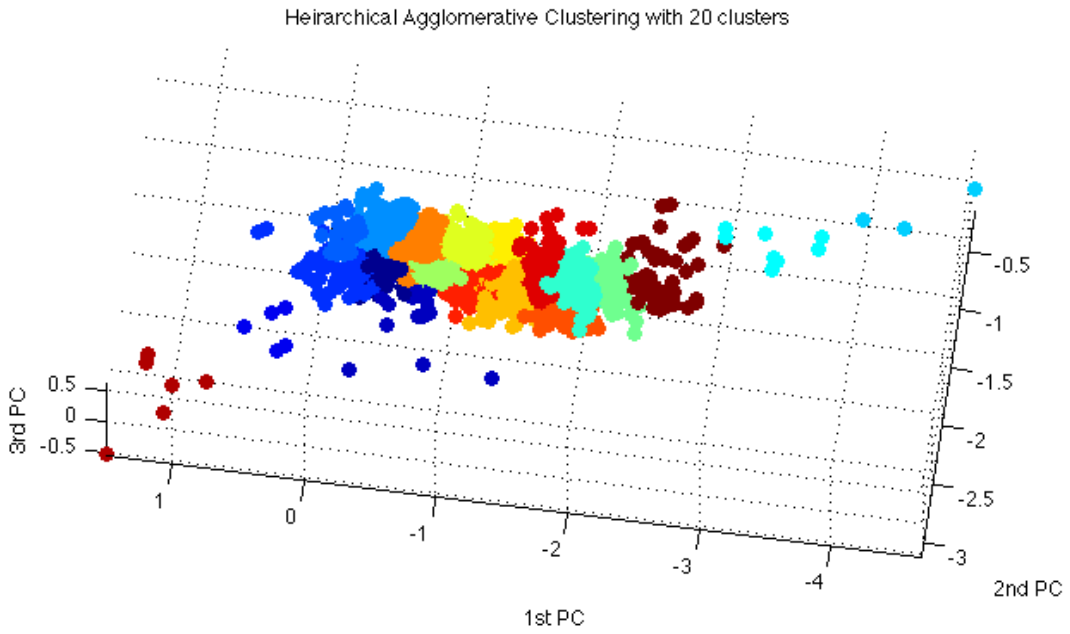

Figure 4.3: *Plot showing the first 20 clusters generated in 31 dimensions using Wards method on Euclidean distance. This is visualised by showing the first three Principal Components. Note the separation along the first and second principal components.*

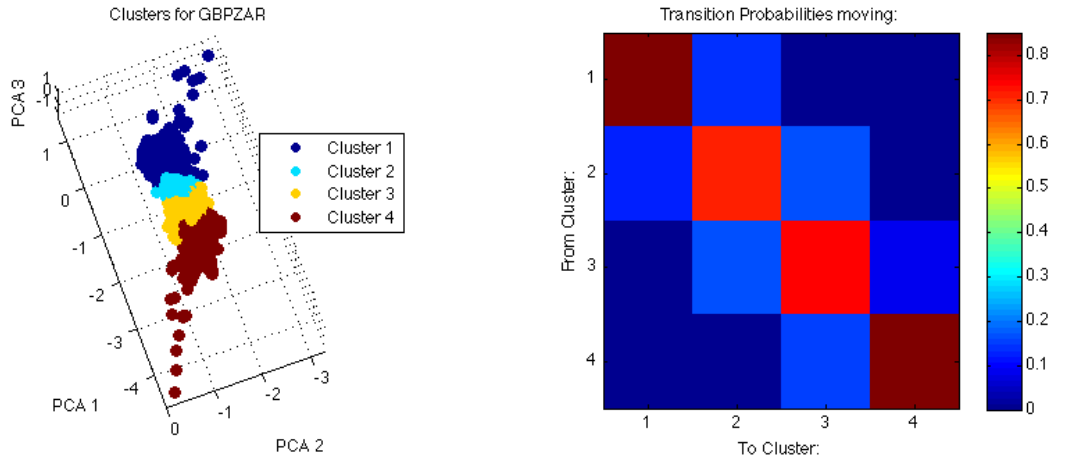

Figure 4.4: *Plots showing the first four clusters, and the transition probabilities between them. Notice how transitions between clusters are more rare than transitions within the same cluster, and transitions occur between neighbouring clusters, seldom skipping a neighbour.*

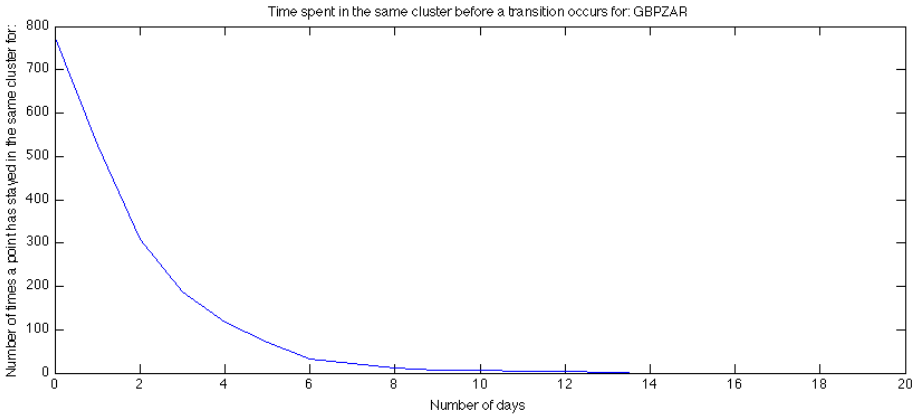

Figure 4.5: *Plot showing the amount of time spent in the same cluster before transitioning to a neighbouring cluster. The exponential decay indicates that transitions between clusters occur fairly frequently.*

In order to understand the dynamics between the clusters, the transition probabilities were calculated. A transition is defined as the movement from one cluster to another (when the current point is no longer in the same cluster is the point before it). This transition view-point treats the cluster model similarly to a finite state automata in that the market is currently in a single state (represented by a cluster) and moves to another state (a different cluster) through a transition. The clusters and their transition probabilities are shown in Figure 4.4.

The clusters are randomly allocated, thus the cluster labels had to be shuffled in the order of their centroid's position along the principal component to get the ordering which is illustrated in Figure 4.4. This shows that subsequent points reside within a single cluster before transitioning to another. Figure 4.4 also shows that points tend move from one cluster to a neighbouring cluster rather than skipping neighbours.

These visualisations sparked an initial hypothesis that the points resided within a single cluster until a major market shift, and could possibly indicated regime switches. If this were the case, this structure could potentially be exploited by creating different algorithms that specialise in different clusters, and switching between them as the changes in clusters were found. However, when investigating this possibility it was found that although transactions generally occur within a cluster, subsequent points do not stay within any given cluster for very long. Rather, they follow an exponential decay, as shown in Figure 4.5. Therefore clusters are not indicative of regime switches.

The movement of points through the space was then visualised through time to understand how the movements related to the principal components. These visualisations

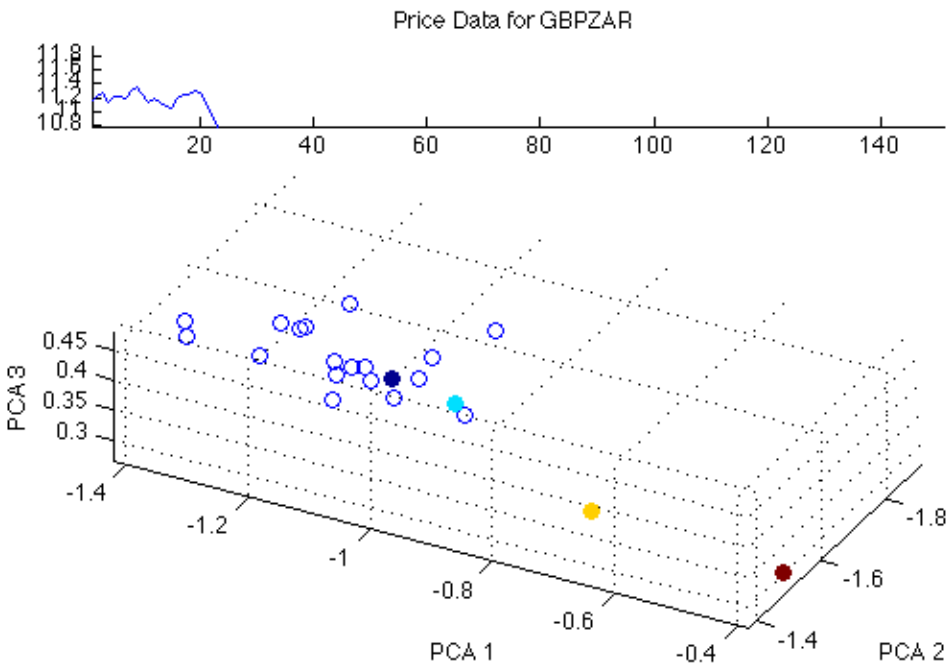

Figure 4.6: *Plot showing the movement of points through PCA space over time. Dark blue represents y_{t-3}, cyan is y_{t-2}, yellow is y_{t-1} and red is the current point: y_t. As the price drops, the points move further to the right - this indicates that movement towards the right side of the principal component is indicative of a sell.*

can be seen in Figures 4.6 and 4.7. In each of these figures, the colour of the filled-in point relates to its position in time, where as the empty circles represent historical points. The dark blue dot represents the point as it was three time steps ago (y_{t-3}). The cyan dot is the point two steps ago (y_{t-2}), yellow is the step before the current point (y_{t-1}) and red is the current point (y_t).

Figure 4.6 shows the movements of the last four positions in time as the price drops. This visualisation indicates that movements to the right in the principal component are indicative of good selling positions. Conversely, Figure 4.7 shows the movements as the price rises, indicating that movements to the left in the principal component signify good buying positions.[2]

Based on these structures and movements it is hypothesised that the position and movement in feature space could be a good indicator of when to buy or sell in a market.

[2]It should be noted that the examples in Figures 4.6 and 4.7 have been specifically chosen because they show the movements in ideal situations. The results are therefore not always as clear-cut.

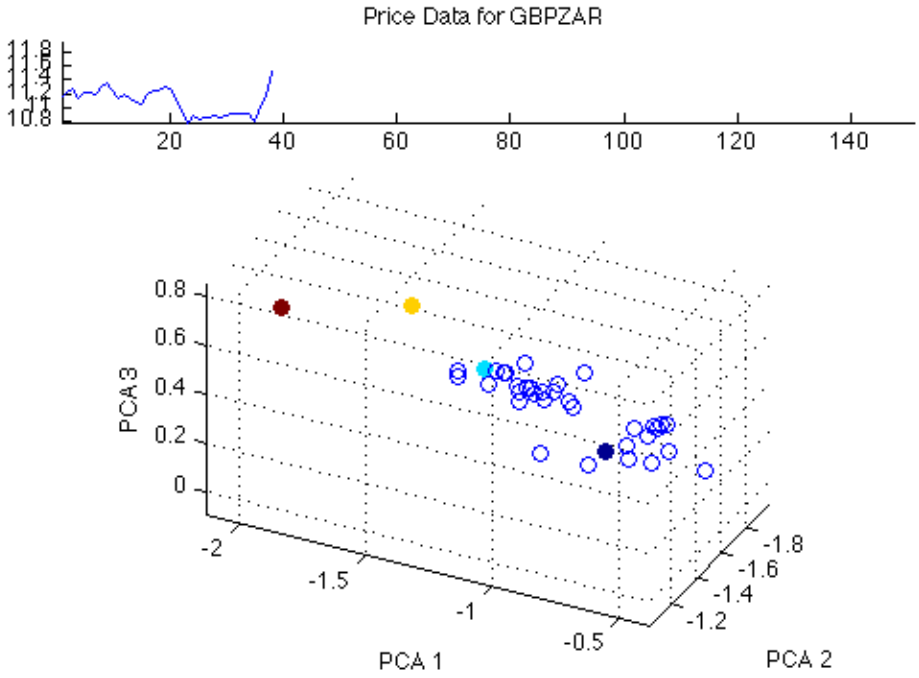

Figure 4.7: *Plot showing the movement of points through PCA space over time. Dark blue represents y_{t-3}, cyan is y_{t-2}, yellow is y_{t-1} and red is the current point: y_t. As the price rises, the points move further to the left - this indicates that movement towards the left side of the principal component is indicative of a buy.*

4.4 Summary

The findings above show that the position of a point in the feature space is indicative not only of the market state (with similar market states occupying similar positions), but their positions could also be indicative of good times to buy and sell on the market. When these points are clustered, they separate the feature space such that points generally reside within the same cluster and switch between clusters less often, but consecutive points seldom reside within the same cluster for long, following an exponential decay.

It was also noted that the movement through the feature space, registered as transitions between clusters, could indicate good positions to buy and sell.

These results could be used to create autonomous trading strategies which exploit the location and movement of points in the feature space. Strategies and their implementations are discussed in the following chapter.

Chapter 5

Exploiting Structures for Financial Gain

This chapter identifies strategies to exploit the structures found in the previous chapter and details their implementation. After the strategies have been discussed, the simulation environment used to determine their profitability is explained. The chapter ends with a discussion on the data used to test the system.

5.1 Autonomous Trading Strategies

Several autonomous trading strategies were designed and implemented to fully test the structures found in Chapter 4. This section details the design and implementation of a naïve baseline competitor, as well as three intelligent strategies which utilise the structures identified in Section 4.3 above.

The naïve strategy uses a mixture of experts based on the Weighted Majority Algorithm. In this strategy, each trading rule represents an expert. This strategy is intended to be a baseline against which the more intelligent strategies can be compared. Its mechanics and implementation are discussed in Subsection 5.1.1.

The second model, detailed in Subsection 5.1.2, is slightly more intelligent, and uses dimensionality reduction techniques on the feature space. By doing this, it can take advice from all of the experts simultaneously, before converting them into buy, hold and sell signals. A decision boundary is then optimised in the training set before being tested.

The third model produced is the first one to use the structures found in Chapter 4. It takes a different approach to dimensionality reduction, treating a single cluster as a

buy cluster, and treating another cluster as a sell cluster. Profitable buy and sell clusters are identified in a training set, and utilised in the test set. This is further explained in Subsection 5.1.3.

The final trading strategy, explained in Subsection 5.1.4, is based on the cluster strategy above. However, instead of identifying profitable clusters, it identifies the transitions between clusters which are indicative of profitable movements through the feature space.

The cluster and transition techniques described above use only a single buy and a single sell cluster or transition. This is because an exhaustive search through all combinations of buy and sell clusters is unfeasibly large. Subsection 5.1.5 looks at using Genetic Algorithms as a means of finding profitable combinations of clusters or transitions in this large search space.

5.1.1 A Naïve Mixture of Experts

Combining the decisions from a number of experts is non-trivial problem, particularly when the mixture is required to operate within a dynamical system. This generally means that the weighting between the experts would need to change over time and in different situations which are usually unpredictable.

Taking a mixture of the experts decisions would be a naïve approach to solving this problem. This can be accomplished by assigning each expert a vote, and making a decision based on the majority rule. This approach can be optimised by strategically swaying the process. In other words, one could afford experts who perform better a greater influence on the final decision by weighting their votes more highly. In this way, a linear combination of the experts decisions are taken.

A popular ways to solve this problem in the context of autonomous traders is by taking a non-linear combination of the expert's decisions. The most common method for combining this information is via Artificial Neural Networks (ANNs) [64]. The ANNs typically use back-propagation in order to learn the weights within a training set, which can then be applied in a test set. The naïve approach is intended as a comparative baseline, thus a non-linear combinations of experts adds unnecessary complexity to the problem and so is deemed unsuitable in the current context.

In this situation, taking a linear combination of the decisions is not as simple as it appears either as each expert, or trading rule, is generating a continuous series of values. These values can be thresholded into discrete buy, hold or sell signals. If a

value is above the buy threshold, it is converted into a buy signal. If the value is below the sell threshold, it is regarded as a sell signal. Lastly, if the value is between the buy and sell thresholds, it is converted into a hold signal. Because each trading rule will differing points at which the thresholds are optimal, these thresholds need to be learnt from a training set. Learning optimal weights for each expert becomes an added learning task. Once combined, another threshold needs to be learnt to decide upon the point at which the majority vote should be taken.

To avoid this complexity, a variant of the weighted majority algorithm was proposed. Let each expert learn its own optimal thresholds in the training set, and record the profit earned. Each of these experts is then given a share of the initial asset according to how profitable it was in the training set. The experts then all trade in the test set, and the profits generated by the experts are then pooled.

5.1.1.1 Implementation

First a greedy approach is taken to finding the optimal thresholds for each of the technical trading tools individually, by iteratively raising the buy threshold from the minimum value to the maximum value in the training set. For each change in the buy signal, the sell signal is iteratively raised from the minimum value in the training set to the buy threshold. After each change is made, the profit is calculated for that rule over the entire training set with those thresholds. If this profit is greater than the previously identified maximum profit, both the profit and the threshold values are stored.

This process will generate buy, hold, and sell signals for every valid combination of buy and sell thresholds, storing the maximum profits and associated thresholds for each trading rule. The distribution of assets is calculated by adding up the maximum profits for each of the rules, and storing the proportion of the total profit that each rule contributed. The optimised thresholds are then used in the test set, and the profit for each trading rule is calculated.

Finally, the overall test profit is calculated by multiplying the profits contributed by each of the rules in the test set by the proportional of contribution to the total profit in the training set. The sum of all these profit-fractions renders the end result of the pooled profits.

5.1.1.2 Changes to the Technical Trading Tools

The trading rules were implemented as described in Section 3.1. Most of these rules use raw values (a single, continuous output variable) that show the degree to which they represent buy or sell signals in their magnitude, but, a few of them require additional effort to convert them to this form.

Filter Rules, Trending and MACD all use intersection of lines as buying and selling signals. Encoding these into hard signals of buying and selling is as simple as taking their sign. However it is non-trivial to encode these into continuous values due to the discrete nature of the time periods in the data (daily returns). The following equations were designed in order to do this conversion, such that the closer the value to zero, and the faster the value is approaching zero, the greater the magnitude of the rule's output. If it is approaching from below, it is given a positive value, and if it is approaching from above, it is given a negative value.

Let y be the raw signal value. Define dy_t as $y_t - y_{t-1}$. The signals can be inverted by applying the following function:

$$\text{intersect}(y_t, dy_t) = \frac{\text{thld}\left[\log\left(|\frac{dy_t}{y_t}|\right)\right]}{\max\left[\log\left(|\frac{dy_t}{y_t}|\right)\right]} \times \text{sign}\,(dy_t)$$

where

$$\text{thld}(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$$

and

$$\text{sign}(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases}$$

This gives a positive value when signals approach the x-axis from below, and a negative value when approaching from above. It also takes the rate of change into account, giving values of greater magnitude to signals which are closer to the x-axis, and to those approaching the x-axis at a higher rate. The results of this can be seen in Figure 5.1.

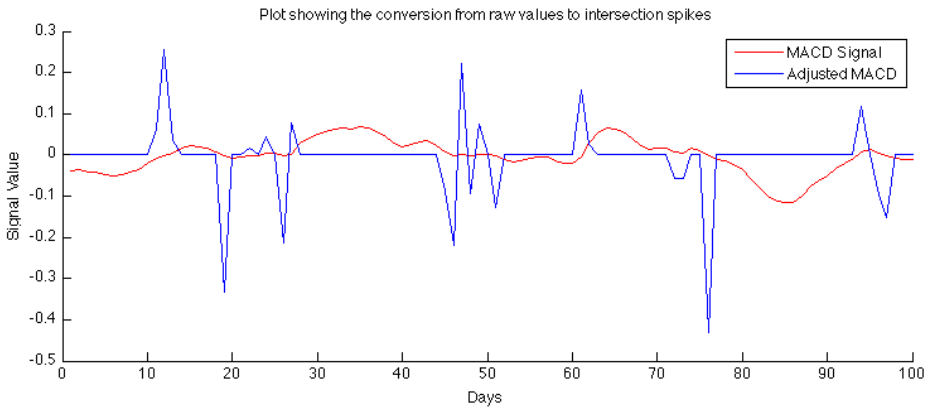

Figure 5.1: *Plot showing the result of converting the raw signals into the intersection spikes. A positive spike indicates a buy signal, and a negative spike indicates a sell signal.*

5.1.1.3 Summary

The strategy described above provides a naïve way of combining the advice from all of the experts. Risk can be reduced by distributing and sharing the profits generated amongst the experts. Thus instead of hedging all bets on a single decision, the profits and losses are spread over a large number of experts. This gives the mixture of experts an advantage over the other methods because of this dispersion of risk.

This was implemented by using a greedy search to find optimal decision boundaries for each rule, allocating assets to each expert according to their profitability in the training set, and pooling the profits generated in the test set.

Some of the rules required a transformation in order to make the representation of their values contextual.

5.1.2 Dimensionality Reduction

An interesting way of solving the problem of a mixture of experts is to treat each expert as its own dimension (the feature space described in Chapter 3), instead of treating each expert as an individual. Thus a decision is taken by aggregating across decisions instead of the naïve mixture's method of aggregating profits and losses.

This new technique takes information from all of the experts into account at the same time by performing dimensionality reduction techniques on the feature space to reduce the decision space to a single dimension. Once reduced in dimensionality, thresholding can be applied to create buy, hold and sell signals. This is the approach taken by the second trading strategy.

5.1.2.1 Implementation

All of the experts are given their own dimension to make decisions. This multidimensional feature space is then reduced down to a single dimension by using non-parametric dimensionality reduction techniques.

Principal Component Analysis (PCA) was chosen as the dimensionality reduction technique of choice for a number of reasons. Firstly, PCA is linear in nature. This means that it keeps the movements to scale, and the directions constant. PCA is also very simple to implement, and easy to understand. One of the criticisms of PCA, is that following the principal component, subsequent components are forced to be perpendicular to the principal component. As the dimensionality is being reduced to a single dimension, this criticism is irrelevant. Another of the fundamental criticisms against PCA is that it does not take the task at hand into account - rather, it blindly pics the direction of most variance [28]. This prompted a novel tweak to the data before PCA in order to adjust for its insensitivity to context.

In order to treat each expert similarly, the data was standardised by subtracting the mean and dividing by the standard deviation. This placed the data about the origin mark, and scaled all the values such that the standard deviation in each dimension occurred at one.

Once standardised, each expert was allowed to perform a greedy search on the training data, in a similar way to the method used for the mixture of experts above, in order to identify ideal decision boundaries, and identify their maximum profit achievable in the training period. The standardised signals were then scaled (multiplicatively) by the maximum profit they achieved. This produces more variance in the more profitable signals, and thus biases them in PCA. The end result is that PCA is forced to take the context into account by treating more profitable signals more favourably due to their increased variance. The magnitude of this effect can be adjusted in the exponent of the scalar (in this case the maximum profit). Raising the scalar to the power of something greater than one will increase the effect of this contextualisation, and raising the scalar to the power of something less than one (but greater than zero) will decrease the effect.

Once the signals are in a single dimension, the same greedy search used for the naïve method above is employed to find optimal decision boundaries on the training set, which are then applied on the test set in order to determine profitability for this strategy.

5.1.2.2 Summary

Instead of taking a mixture of the experts decisions, this strategy takes all of the experts decisions into account simultaneously by operating in the feature space. The multi-dimensional feature space was then simplified through dimensionality reduction, and optimal decision boundaries were learnt by greedy search.

5.1.3 Clusters

Based on the structural analysis in Chapter 4, an autonomous trading strategy was created which addresses the dimensionality reduction problem described above in an very different way. Each cluster is assigned a label of whether it is a buy, hold or sell cluster. The presence of a data point in a buy clusters is then indicative of a buy signal. The cluster labels are learnt in the training set and applied in the test based on the labels of the test point's nearest neighbours.

5.1.3.1 Implementation

First, the clusters were identified using the unsupervised learning techniques described in Chapter 4: hierarchical agglomerative clustering, using Euclidean distance as the distance function. Not only did this allow grouping of neighbours by minimising the total distance in any one cluster, but it also enabled investigations into different numbers of clusters efficiently without having to recompute the clusters every time.

A greedy search algorithm was then used to identify a single buy cluster, and a single sell cluster for a given number of clusters which maximised profit in the training set. This was implemented by iterating through all clusters, giving each cluster a turn as the buy cluster. For each change of the buy cluster, every other cluster was given a turn at being the sell cluster, and profits were recorded for each combination. If the profit for a combination was greater than the maximum profit seen before, both the profit and the combination of buy and sell clusters were recorded. This produced both the maximum possible profit in the training set for a single buy and sell cluster, as well as the clusters used to generate this profit.

Once the clusters had been identified and classified as signals, these cluster labels and hence signals were propagated through to the test set by using the K-Nearest Neighbours (KNN) approach. The KNN approach was chosen because of its similarity to the clustering method used, and the five nearest neighbours were chosen in order to give each point a representative sample of neighbours, while not requiring so many

samples that the sparseness of the feature space would introduce distant neighbours. These signals were then used to calculate the profitability on the test set, and hence the profitability of the strategy.

5.1.3.2 Reasons for limiting the number of active clusters

The decision to allow for only a single buy and a single sell cluster was made in order to significantly reduce the search space. Given that there are three possible states for any one cluster (buy, hold or sell), an exhaustive search of all combinations of clusters labels would amount to 3^n possibilities, where n is the number of clusters. This is feasible for small numbers of clusters, but becomes intractable quite quickly. By reducing the problem to only two active clusters, the search space was reduced to n^2.

5.1.3.3 Choosing the number of clusters

The number of clusters to use is not a question that is easily resolved. The system – as implemented – is only allowed one cluster for each of its signals, therefore having too few clusters will result in large clusters - which are likely to create some unnecessary transactions, which will include some bad decisions, along with any good ones. Conversely, having too many clusters will become very restrictive, because the system can only buy and sell in one of these clusters. Thus, the strategy must optimise the necessary number of clusters, ranging between two and ten. This increases the complexity to $\sum_{n=l}^{u} n^2$ where l is the lower bound on the number of clusters tested , and u is the upper bound.

Thus instead of setting the actual number of clusters used, the only parameter required is the maximum number of clusters. The value of this parameter was ultimately decided by performing greedy searches as described above, on a variety of different cluster sizes. The results of this can be seen in Figure 5.2.

This figure shows that an increase in number of clusters will generate greater profit in the training set, as unnecessary transitions are whittled out. In the test set the results are not so clear. There is a variable amount of profit with a peak appearing at 8 clusters, and steadily decreases after 11 clusters. This shows that the cluster size should not be prescribed, allowing the number of clusters to be chosen at training time, however, after 11 clusters there is a steady decrease in the profits rendered due to over-fitting. Thus 11 is a good number at which to set the upper bound for clusters.

Figure 5.2: *Plot showing the profits rendered for a variety of cluster sizes. The larger the number of clusters in the training set, the more profitable it becomes. In the test set, the increase in cluster size engenders an increase in profits up to a point, where over-training occurs. The peak point appears to be about 7 or 8 clusters, and after about 10 or 11 clusters the profit decreases.*

5.1.3.4 Summary

This trading strategy identified profitable buy and sell clusters in the training set by iterating through all combinations of a single buy and a single sell cluster. The decision to use only two active clusters was made in order to dramatically reduce the size of the search space. This was repeated for different numbers of clusters in order to find the most profitable number of clusters for the given training set. Once the optimal number of clusters and the optimal active clusters were identified, these signals were applied to the test set. The points in the test set were then classified by using the K-Nearest Neighbours approach.

5.1.4 Transitions

The final trading strategy developed is based on the cluster strategy above - but instead of identifying profitable clusters, it identifies profitable transitions between clusters. These are then marked as buy or sell transitions. This is based on the hypothesis that transitions should prove more profitable than clusters as they take the direction of movement between points, and also takes temporal ordering into account. This strategy is based on the findings of investigations such as the ones shown in Figures 4.6 and 4.7.

5.1.4.1 Implementation

As above, the clusters must first be identified by using hierarchical agglomerative clustering. A similar greedy search algorithm was used to iterate through all possible transitions between clusters. This was accomplished by iterating through all of the clusters, letting each one have a turn at being the *buy from* cluster. For each *buy from* cluster, every other cluster was given a chance at being the *buy to* cluster. The *buy from* and *buy to* pair signified the buy transition. The same principle applied to the remaining clusters in order to identify profitable *sell from* and *sell to* clusters.

KNN was once again used to propagate the signals to the training set for the buy and sell transitions, and profitability was then calculated on the test set in order to determine profitability for this strategy.

5.1.4.2 Motivation for restricting active transitions

It should be noted that in the case of an exhaustive search of all possible combinations of transitions, the search space increases to 3^{n^2} for a fixed number of n clusters[1]. Using the greedy search as above complexity is reduced to n^4, and when identifying an ideal number of clusters, it is increased to $\sum_{n=l}^{u} n^4$ where l is the lower bound on the number of clusters tested, and u is the upper bound.

Similarly to in the Cluster strategy, the upper bound needs to be set. Figure 5.3 shows the profits generated when greedy searching through a variety of cluster sizes.

This figure shows that an increase in number of clusters will generate greater profit in the training set up until a maximum of 13 clusters when the profit starts to decline. This is because unnecessary transitions are whittled out (as an increase in number of clusters decreases the average number of points in a cluster) until the point where valid transitions are being removed.

The results in the test set are less structured. There is a peak at 3 clusters but no clear trend follows thereafter. It was therefore decided that for simplicity's sake, the same number of clusters would be used for both the Cluster and Transition strategies.

5.1.4.3 Summary

The final trading strategy identified transitions between clusters which were profitable buy and sell indicators in the training. These transitions were learnt by iterating

[1]To put this into perspective: at 15 clusters, there are $3^{15^2} = 3^{225} \approxeq 2.3 \times 10^{107}$ different transition combinations. Where as $15^4 = 50625$; many orders of magnitude smaller.

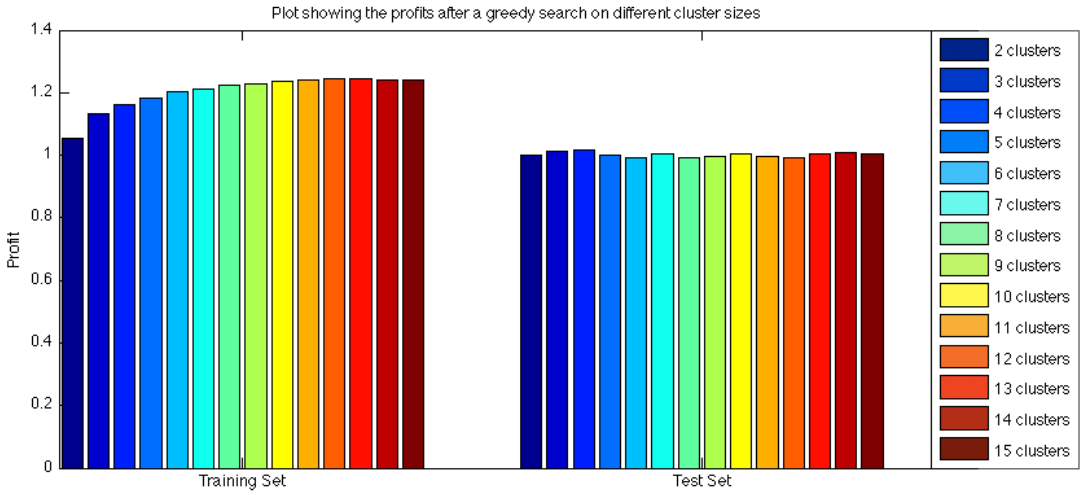

Figure 5.3: *Plot showing the profits rendered for a greedy search on the transitions for a variety of cluster sizes. As with the greedy search on clusters, the larger the number of clusters in the training set, the more profitable it becomes, but it appears to peak at 13 clusters (*13^2 *transitions). However, in the test set, this is not so clear. The peak point appears to be 3 clusters (and thus 9 transitions), but no clear trend in decreasing number of transitions is evident.*

through all combinations of a single buy and a single sell transition. The decision to use only two active transitions was made in order to dramatically reduce the size of the search space, and the number of clusters between which transitions occur was included as part of the optimisation problem.

Intuitively, identifying transitions should be more useful than clusters, because transitions are able to capture the direction of movement as well as the region. This starts to incorporate temporal ordering into the decision space which is hypothesised to be useful based on the explorations done in Section 4.3.

5.1.5 Using Genetic Algorithms to Find Optimal Positions

Subsections 5.1.3 and 5.1.4 described the need for restricting the number of active clusters. Intuitively having many more clusters and transitions from which to choose, and identifying multiple clusters and transitions to act as buy or sell signals should improve results dramatically. This is because having increased options effectively creates a highly non-linear decision boundary in the feature space with much finer granularity than a single buy and sell cluster or transition could afford. It is also expected that good buy and sell clusters may change over time pending non-stationarity and regime switches. Thus selecting only a single buy and sell cluster or transition limits the sys-

tem's potential tremendously.

Subsections 5.1.3 and 5.1.4 also outlined how large the search space becomes when multiple clusters and transitions are investigated. For a fixed number of n clusters the search space for all combinations of clusters is 3^n, and for transitions it is 3^{n^2}, where n is the number of clusters. To put this in perspective: At 15 clusters, there are approximately 2×10^{26} times more transition possibilities than the estimated number of particles in the observable universe!

Numbers of this size are far beyond the reach of standard optimisation techniques, and we must therefore enter the domain of meta heuristics in order to optimise these strategies. Due to the size and combinatoric nature of the search space, genetic algorithms were identified as the ideal metahueristic for the task.

5.1.5.1 Chromosome

A chromosome was created, with each position in the chromosome corresponding to a specific cluster or transition. These chromosomes had the options of being either -1 (registering a sell signal), 0 (registering a hold signal) or 1 (registering a buy signal). The cluster chromosome contained n positions and the transitions chromosome contained n^2 positions, where n is the number of clusters.

Although the transition chromosome is a single long string, it can be thought of as a square matrix where each row corresponds to a *from* cluster, and each column corresponds to a *to* cluster as described in Subsection 5.1.4 above.

5.1.5.2 Genetic Operators

An initial population of 20 individuals was randomly generated, and the optimisation ran for a maximum of 500 generations, or until there was less than a 1% improvement in the best solution over more than 50 generations. Roulette wheel selection was used to select parents except for a single member which was chosen by elitism in order to retain the best solution through subsequent generations. Single point crossover was chosen as a means of recombining parents to create the child population, and the child population was perturbed by uniform mutation. The crossover and mutation rates were set to 0.75 and 0.05 respectively as suggested by Mitchell[40].

5.1.5.3 Summary

When dealing with all possible combinations of clusters and transitions, the search space becomes too large to perform a greedy search for optimal combinations. Thus genetic algorithms were applied to the problem to allow for multiple buy and sell transitions or clusters.

5.1.6 Summary

In this section, several strategies for autonomous traders were presented. A naïve approach was taken, allowing each expert to learn optimal decision boundaries and allowed each expert operate in the test with a proportion of the asset corresponding to its profitability in the training set. The second strategy took information from all experts into account in a single decision by performing dimensionality reduction on the feature space. Once the feature space was reduced in dimensionality, a greedy search was performed to identify optimal decision boundaries. The final two strategies presented made use of the structures found in Chapter 4 by associating clusters and transitions between clusters with buy and sell signals. Increasing the number of clusters results in a combinatoric explosion, thus the greedy search algorithm were only allowed to choose a single buy and a single sell cluster or transition. Finally a genetic algorithm was created in order to find optimal clusters and transitions from the space of all possible combinations.

5.2 The Simulation Environment

The Simulation Environment (SE) is responsible for simulating trades and calculating profits. It needs to track all of the buy and sell requests, ensure that they are legal (i.e. that one is not selling an asset that they don't have), and ensure fairness towards the Trading Strategies. This section describes how the SE accomplishes these tasks.

5.2.1 Mechanics of the System

This SE calculates profit based on the geometric differences between the buy and sell points. The end results of all trades can then be calculated as the product of all the geometric differences.

The system is intended to operate on a single stream of foreign exchange data. As an example, this means that we are buying US Dollars (USD) with British Pounds (GBP) in order to form the USDGBP pair. Thus, a buy signal in the USD stream means a sell signal on the GBP stream as we are trading GBP for USD. As this system is based on geometric differences, a buy registers buying into the indicated currency with *all* of one's wealth (which is currently invested in another currency), and correspondingly, a sell registers selling *all* of the current currency for another. Not only is this intuitive, but it also simplifies the implementation, as the system only has to keep track of one asset instead of a portfolio of assets, enabling focus on the Trading Strategies themselves.

Based on the above, one can only buy into a currency if one is not currently holding all ones money in that currency, and so to ensure fairness, one is only able to enter the market on a buy signal. This mitigates against any penalties that the trading strategy may incur by simply starting in a bad market state.

5.2.2 The Profit Finder

Once a single set of signals has been generated, it is necessary to test them for profitability in the appropriate currency and period. The *Profit Finder* (PF) is used to perform this function. The PF was built to take trading costs and slippage[2] into account to mitigate against criticisms shown towards previous autonomous traders [20].

5.2.2.1 Cleaning Signals

The PF takes in a single set of crisp signals. These crisp signals are either -1s, 0s or 1s, representing sell, hold and buy signals respectively. It is then the job of the PF to calculate the profit generated by these signals for the given time period in the given asset.

Entering and Exiting the Market One of the constraints placed on transactions was mentioned above: one has to enter the market with a buy signal before one can sell in that market (one cannot sell what one does not have). Thus the first signal registered needs to be that of a buy signal. The PF resolves this issue by replacing all signals before the first buy signal with hold signals (0s).

[2]Slippage is the time taken between receiving the information, and making a decision. As even the fastest processors can't process information instantaneously, this needs to be taken into account – especially when using tick data at intervals of a few minutes.

Similarly, the final signal needs to be a sell signal so that we can fairly calculate the profits generated for decisions the trading strategy actually made. Thus all signals after the final sell signal are replaced with hold signals.

Matching Transactions The trading rules are governed by inequalities, thus one seldom sees a single buy or sell signal on its own. These signals generally come in waves, registering the same signal for the entire time the inequality holds. Due to the nature of the system, in which it buys and sells *all* assets with each transaction, duplicate signals are redundant. This means that we have to pair each buy signal with a single sell signal, converting all duplicate buy signals in between into hold signals. The same is true for the space between sell and the next buy signal.

This is accomplished by finding the indices of non-zero elements, and checking if neighbouring non-zero elements are the same. If this is the case, only the first in the sequence of duplicate signals is kept, and the rest are converted to hold signals.

Summary This process converts an arbitrary string of crisp signals into one which has leading hold signals until the first buy signal, and ensures that the final non-hold signal is a sell signal. The spaces between consecutive buy and sell signals are all set to hold signals, because only the first buy or sell signal in a set of duplicate signals is relevant. This makes it easy to calculate profit in the given system.

5.2.2.2 Calculating Profit

Profits in the PF are all calculated by calculating the geometric difference between the data for each pair of buy and sell signals. This is done in the following manner.

Let y_t be the price of an asset at time t, the time at which a buy signal occurs be described by b, and a sell signal by s. The returns R for the period $s-b$ is then calculated as follows:

$$R_{b,s} = \frac{y_s}{y_b} \tag{5.1}$$

thus if the selling price y_s were greater than the buying price y_b, a return greater than 1 is observed.

Previous systems have been criticised for not taking trading costs, interest rates and slippages into account [50]. This system was therefore built to take all of these technicalities into account.

The data for both direct and indirect prices for all currencies was gathered from the electronic foreign exchange service OANDA[44]. Analysis on the the data is provided in Section 5.3 below. These price indices were published at the interbank rates (those faced by large institutions), and so does have some inherent trading costs. These trading costs C are isolated by calculating the geometric difference between the direct price and the inverse of the indirect price. For example, if there were no trading costs, the price of USD in EUR (the buying price) would be the same as $\frac{1}{price\,of\,EUR\,in\,USD}$ (the selling price), thus we can find the geometric cost of trading by taking the geometric difference between the two. This cost is increased by adding an additional percentage to each of the prices before calculating the geometric costs.

When transacting, these costs are applied to Equation 5.1 multiplicatively such that:

$$R_{b,s} = \frac{y_s}{y_b} C_b C_s \tag{5.2}$$

Interest effects, $\imath$, can be taken into account by using the average overnight interest rate $\bar{i_{b,s}}$ for each currency for the duration of the time spent in that currency:

$$\imath = (1 + \bar{i}_{b,s})^{\frac{s-b}{365}} \tag{5.3}$$

The selling price in Equation 5.1 can then be inflated by the interested generated over that time, such that:

$$R_{b,s} = \frac{\imath y_s}{y_b} \tag{5.4}$$

Slippage can be accounted for by incrementing both b and s by the number of periods, n, required, such that Equation 5.1 becomes:

$$R_{b,s} = \frac{y_{s+n}}{y_{b+n}} \tag{5.5}$$

The total profit for each transactions τ can be calculated by continuously compounding the returns for each transaction:

$$R = \prod_{\tau=1}^{T} R_{b_\tau, s_\tau} \tag{5.6}$$

In order to make comparisons fair between different systems which trade for different lengths of time, the returns must be scaled to be over an entire year. Thus:

$$\mathcal{R} = R^{\frac{D}{365}} \tag{5.7}$$

where

$$D = \sum_{\tau=1}^{T} s_\tau - b_\tau \tag{5.8}$$

Placing all of the above in a single formula yields:

$$\mathcal{R} = \left(\prod_{\tau=1}^{T} \frac{(1+\bar{i}_{b_\tau+n,s_\tau+n})^{\frac{s_\tau-b_\tau}{365}} y_{s_\tau+n}}{y_{b_\tau+n}} C_{b_\tau+n} C_{s_\tau+n} \right)^{\frac{\sum_{\tau=1}^{T} b_\tau - s_\tau}{365}} \tag{5.9}$$

5.2.3 Exclusion of Interest Rates

Although Equation 5.9 is built to include interest rates, it was decided against including them into the system. This decision was made in order to make the returns rendered indicative of the returns over and above a fair return. This would also allow the profits rendered on real data comparable to profits rendered on generated data (where a fair interest rate is not obvious).

5.2.4 Summary

Through cleaning signals and processing them with the Profit Finder (PF), the Simulation Environment (SE) is able to calculate the transactional profits each signal makes, taking interest effects, slippage and trading costs into account. These transactional profits can then be combined to return the overall, fair, and comparable profit per annum for all of the inputted signals. Although it is possible to include them, interest rates were ignored in order to identify the profit over and above a fair return.

5.3 Test Data

The trading strategies described above were tested on a number of different data types. Firstly they were tested on generative models before being trialed on real-world data. The generative models used are of interest because they follow the evolution of models used to understand financial markets. They are: random noise, random walks, and GARCH/ARMA. The financial data chosen represents ten years of foreign exchange data from ten very different currencies in order to mitigate criticisms of data snooping.

5.3.1 Random Noise and Random Walks

Random walks were, until fairly recently, believed to represent the behaviour of financial markets, and so are an interesting data set upon which to test. This is also a fair test of the system, as one theoretically should not be able to earn profits on a random walk due to its unpredictability.

In order to be representative of the real data, a random walk was generated for each of the tested currencies. Data was generated for the same number of days (10 years), and the standard deviations for each of the walk's innovations (or daily movements) were directly taken from the standard deviations of each of the currency's innovations. The first day was then set to be the same as that of the currency, so that when the cumulative sum was taken, the random walks were in the same region as the original currency data, whether it be below one or above ten.

Table 5.1 shows that all of the randomly generated currencies appear to be Gaussian in nature. Although the standard deviations of the data are quite different, their means and skewness are all about 0, and the kurtosis of each of the series is about 3. These are all classic characteristics of the Gaussian distribution. Figure 5.4 shows the distributions of the innovations for each of the curves which are quite clearly Gaussian. When comparing these to the actual foreign exchange data in Table 5.3 and Figure 5.6, it becomes obvious that this model does not capture the excess kurtosis present in real world data [9, 14, 25] (also discussed in Sections 2.6.2 and 5.3.3), nor does it capture the skewness.

5.3.2 GARCH/ARMA

GARCH is a popular method of modelling the variance in a financial time-series. It is able to account for many of the phenomena observed in real time-series, particularly those of excess kurtosis and volatility clustering. ARMA provides a way of generating innovations from the GARCH variances while allowing for phenomena such as trending to occur. As with the random walks, a GARCH simulation was run for each of the currencies, using coefficients learnt from the original currency data. These coefficients were learnt from the original currency data using maximum likelihood estimation, and simulated using GARCH(1,1) and ARMA(1,1) models as suggested by literature [2, 3, 24].

Table 5.2 shows the statistical results of these generations. Although the kurtosis is more representative of the real world data than the random walk model (in Table 5.1

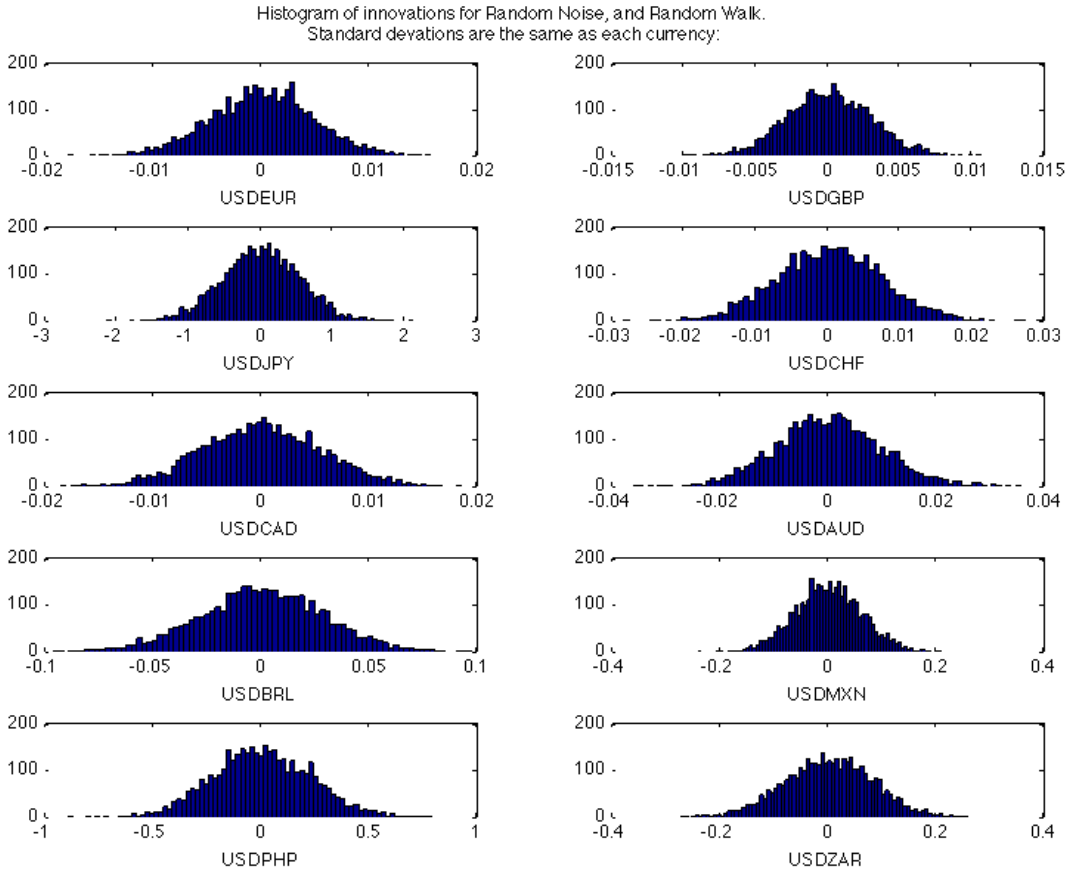

Figure 5.4: *A histogram showing the distributions of innovations for each of the currencies generated by a Random Walk. Notice the Gaussian shape of the distributions.*

	Random Walk models of Foreign Exchange Data			
Currency:	Mean:	STD:	Skewness:	Kurtosis:
RW_USDEUR	0.0001	0.0047	-0.0507	2.9374
RW_USDGBP	0	0.0029	0.0109	2.9318
RW_USDJPY	-0.0019	0.5382	0.0304	3.1005
RW_USDCHF	0.0001	0.0074	-0.0198	3.0397
RW_USDCAD	0	0.0055	0.013	2.8952
RW_USDAUD	0	0.0097	0.107	3.1357
RW_USDBRL	0.0002	0.0285	-0.0507	2.9971
RW_USDMXN	-0.0007	0.0618	-0.0108	2.9731
RW_USDPHP	0.0029	0.2263	0.0841	2.9648
RW_USDZAR	0.0005	0.0822	-0.0551	2.9172

Table 5.1: *Table showing descriptive statistics of the average each of the random walk models trained on the currencies.*

and Figure 5.4), it is still nowhere near the level of excess kurtosis observed in the real world data (in Table 5.3 and Figure 5.6). It is also evident that the skewness is not representative of the levels found in real world data.

5.3.3 Foreign Exchange

> "As for the foreign exchange, it is almost as romantic as young love, and quite as resistant to formulae." - H. L. Mencken (1880-1950)

In order to fully test the aforementioned system and avoid criticism based on data snooping, a number of different currencies were chosen over an extended period of time.

This data was obtained via the OANDA electronic foreign exchange service [44] as direct quotes relative to the USD at the daily interbank rate for each day for over ten years, dating from 1st January 2000 to 5th August 2010.

The currency set includes: four major currencies (majors), the Euro (EUR), Japanese Yen (JPY), British Pound (GBP) and Swiss Franc (CHF); two minor currencies (minors), Canadian Dollar (CAD), and Australian Dollar (AUD); and four emerging market currencies, including the Brazilian Real (BRL), Mexican Peso (MXN), Philippine Peso (PHP) and the South African Rand (ZAR). These test currencies were chosen

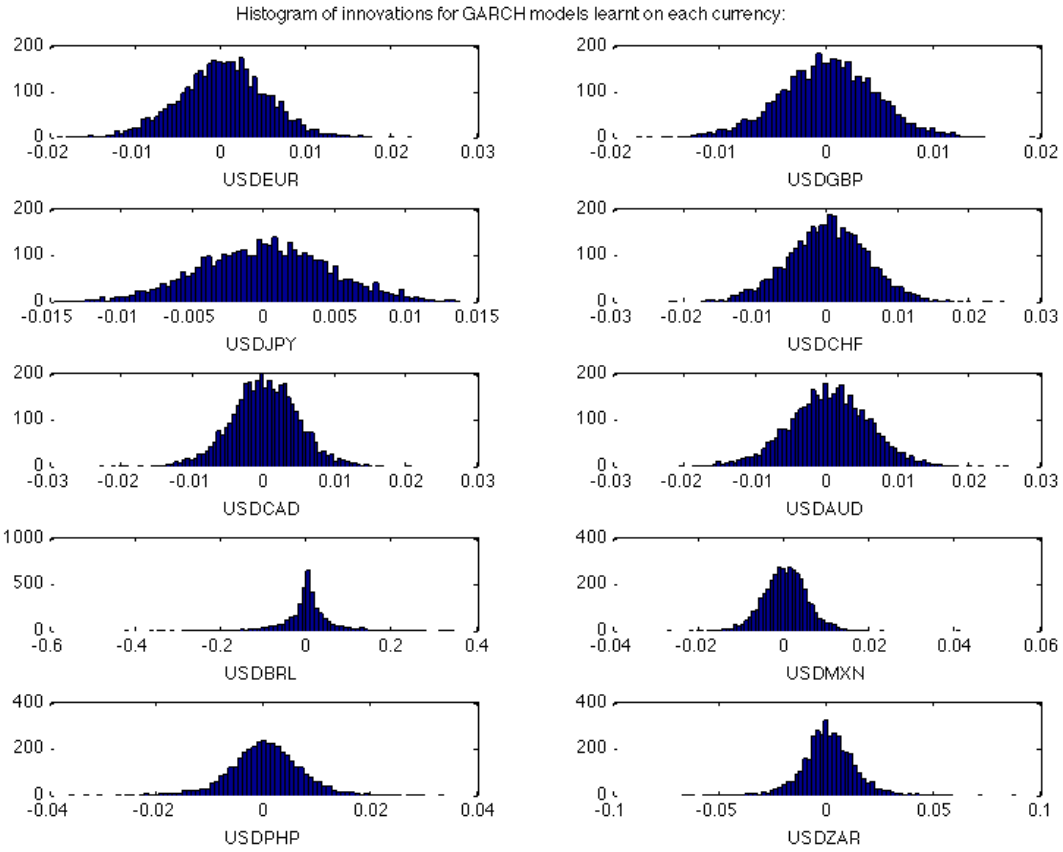

Figure 5.5: *A histogram showing the distributions of innovations for each of the currencies generated by GARCH models. The coefficients for these models were learnt from real foreign exchange data. Notice that although exhibiting some fatter tails (particularly in the more variant currencies such as the USDBRL and USD ZAR), the distributions are still fairly Gaussian in shape.*

	GARCH models of Foreign Exchange Data			
Currency:	Mean:	STD:	Skewness:	Kurtosis:
GARCH_USDEUR	0.0001	0.0051	-0.0505	3.3163
GARCH_USDGBP	0	0.0045	-0.0367	3.258
GARCH_USDJPY	0	0.0046	-0.0019	2.9005
GARCH_USDCHF	0	0.0056	-0.0038	3.3985
GARCH_USDCAD	-0.0001	0.0047	-0.0818	3.6847
GARCH_USDAUD	0.0001	0.0057	-0.005	3.3441
GARCH_USDBRL	-0.0018	0.0654	-0.2393	8.1301
GARCH_USDMXN	0	0.0052	0.1003	4.7138
GARCH_USDPHP	0.0001	0.007	-0.0538	4.8891
GARCH_USDZAR	0.0001	0.0133	0.0263	5.8302

Table 5.2: *Table showing descriptive statistics of each of the GARCH models trained on the currencies.*

from the majors, minors and emerging markets because all exhibit different characteristics. As such, they all have different transaction costs, interest rates, and some are more volatile than others.

Transaction Costs were determined as the rates faced by large institutions, with the ability to magnify them by a percentage of the currency traded [44]. Although interest rates can be factored into the model, they were ignored for a number of reasons. Firstly the project is not concerned with the maximum profit a system could earn, but rather the ability to earn more profit than fair returns would govern. Returns thus represent the returns over and above a standard US Treasury bill. Secondly, hypothetical interest rates for the generative models would be unsubstantiated, thus interest rates were ignored in order to make returns between the data sets comparable.

Table 5.3 provides more information on each of the currencies. Most notable of these is the huge variations in kurtosis, all of which are more than double that of a standard Gaussian distribution. There also appear to be some currencies with large skewness, indicating asymmetric distributions.

	Foreign Exchange Data			
Currency:	Mean:	STD:	Skewness:	Kurtosis:
USDEUR	-0.0001	0.0047	-0.0356	6.8913
USDGBP	0	0.0029	0.4437	10.2898
USDJPY	-0.0043	0.5496	-0.2407	6.6451
USDCHF	-0.0001	0.0074	-0.1378	7.089
USDCAD	-0.0001	0.0056	0.0798	8.9663
USDAUD	-0.0001	0.01	0.3848	12.3818
USDBRL	0	0.0283	-0.9299	28.1658
USDMXN	0.0008	0.0631	2.2754	46.4433
USDPHP	0.0013	0.2272	-2.6099	75.9474
USDZAR	0.0003	0.0815	0.4103	42.8147

Table 5.3: *Table showing descriptive statistics of each of the currencies.*

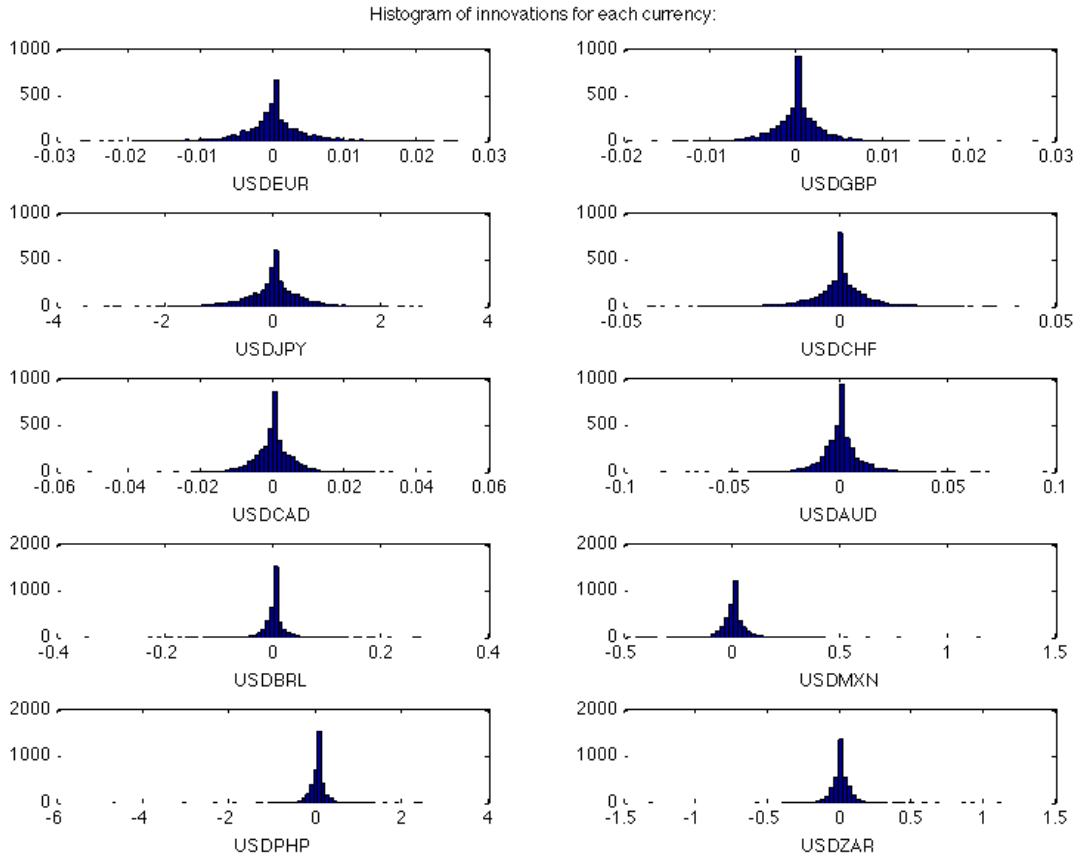

Figure 5.6: *A histogram showing the distributions of innovations for each of the currencies. Notice the excess kurtosis (fat tails and high peaks).*

5.3.4 Permuted data

Because of the deviations between the real world data and the generative models, it is worth investigating data which has similar statistical properties to the real world data. This semi-realistic data can be generated from the real data by taking a random permutation of the innovations. Then, just as with the random walk model the cumulative sum of permuted innovations and the first point of the original data set forms a new data set. This quite obviously produces exactly the same histograms and statistical properties as found in Table 5.3 and Figure 5.6 because it is comprised of exactly the same innovations, but would not have the same volatility clustering or any long-memory effects.

The the effect of the permutation on volatility clustering and long-memory can be controlled by changing the granularity at which the data is permuted. By permuting at the daily price level (the granularity of the data used), all volatility clustering and long-memory effects are discarded. By permuting chunks of innovations, the presence of volatility clustering and long-memory is controlled by the size of the chunks. The size of the training and test sets are 365 days each, thus a middle ground of permutation was chosen to have chunks of about 50 days in length. This would allow some effects to persist, but would create enough disruption to be evident in every training and testing period. To ensure fairness, the exact size of each chunk was randomly generated between 40 and 60 days.

5.4 Summary

This chapter identified three potential trading strategies which made use of the structures identified in Chapter 4, and outlined the implementations for each of them. The first of which used dimensionality reduction techniques on the feature space in order to listen to all experts simultaneously, before greedily finding a decision boundary for the reduced data. The second strategy made use of the clusters found in Chapter 4. This strategy identified the clusters as profitable times to buy, hold and sell in a market. The exact combination of buy and sell clusters was learnt in a greedy fashion. The final strategy used similar principles to the cluster-based strategy, but instead identified transitions between clusters as being profitable or not, thus using the temporal ordering and movements between points in the feature space. The cluster- and transition-based strategies both created a combinatoric explosion when the number of cluster was in-

creased. The effects of this were mitigated by allowing two clusters or transitions to be active (one for a buy signal and one for a sell signal), leaving the rest as hold signals.

This chapter also presented an optimisation method in order to deal with the combinatoric explosion created when extending the search space to include all possible combinations of clusters and transitions. Genetic algorithms were chosen as a suitable meta-heuristic search method to optimise the combination of clusters and transitions.

The simulation environment used to calculate returns was also described, detailing the mechanisms which govern the calculation of a strategy's profitability.

Finally, the data used to test the system was investigated. This included three generative models (random noise, random walk and GARCH), as well as ten years of foreign exchange data across ten very different currencies. The generative models do not fully capture the excess kurtosis or skew present in the foreign exchange data and so a fourth method of randomly perturbing innovations of real data was also discussed.

Chapter 6

Results

The original hypothesis of this thesis is that structures exist within a single financial time-series, and that these structures can be exploited for financial gain. In order to test for the above, several strategies were designed, implemented, and tested on several data sets. This generates results which would be overwhelming if presented all at once. Thus the presentation of results is broken down into two sections before being critically analysed.

Section 6.1 presents the comparison between different trading strategies, using real foreign exchange data as a basis. This shows that the strategies which make use of the structures identified were profitable, where as those that do not use the structures were not profitable.

Section 6.2 compares the results of the different strategies on different data sets. This shows that while profits are rendered in the real foreign exchange data by exploiting structures, no such profitable structures are found in the generative models. Furthermore, when randomly permuting the daily movements of the time-series no profit is found, whilst when randomly permuting chunks of daily movements, some profit is found but less so than on the full unpermuted data set. This shows that randomly permuting the innovations breaks down the structure being exploited by the advanced techniques.

Section 6.3 provides a critical analysis of the results.

6.1 Comparisons between Strategies

This section explains the results of an investigation into the profitability of each of the strategies on real foreign exchange data. The data, as described in Section 5.3, com-

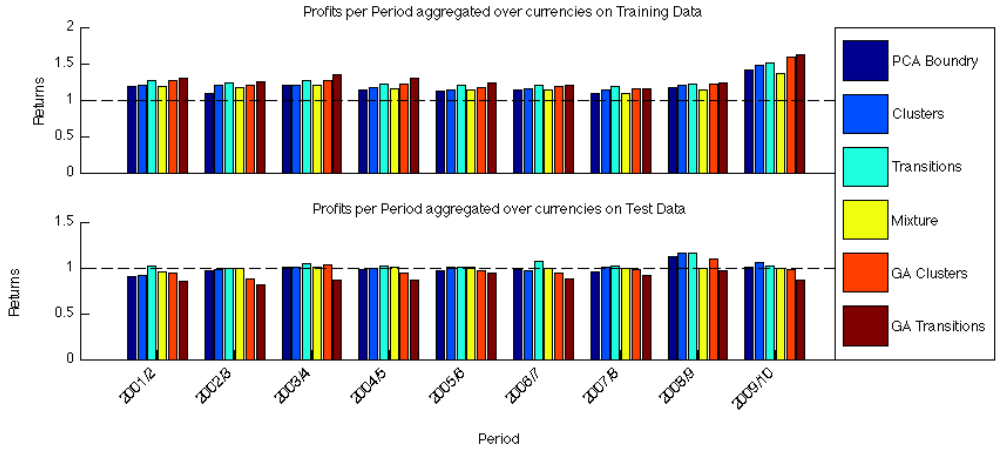

Figure 6.1: *A bar chart, showing the profits per period (aggregated over currencies) on foreign exchange data. The dotted black line represents a fair return, as one would get from an investment bank. As the training and test data is segregated, the dates for the training periods are one year behind those of the test periods. Notice how the profits in the training data increase with each change of strategy (barring the mixture of experts baseline), however on the the test sets, the genetically optimised cluster and transition strategies under perform. This is most likely due to over-fitting on the test set. Also notice how the 2008/2009 period yields a higher returns than the other periods. This might indicate that there is more profit to be made in times of instability (due to the financial crisis).*

prises of ten years and eight months worth of daily interbank rates from ten different currencies against the US Dollar. When testing, the eight months was discarded because all of the signals require time to settle, some of which require over one hundred days (i.e. SampEn). In the tests – one year was dedicated to training, and the following year was dedicated to testing. This separation meant that ten years of data yields nine years of results. The results for each of the years (aggregated over currencies) is shown in Figure 6.1[1].

The most noticeable feature of Figure 6.1 is that in the training set, the more complex the system, the more profit it generally earns. This does not, however, translate directly to the test set. It is clear that the Cluster strategy performs better than the Dimensionality Reduction approach (using PCA to combine experts' opinions), and the Transition strategy out performs the Cluster strategy.

A large difference occurs between the genetically optimised versions of the cluster

[1]The data used to generate Figures 6.1 and 6.2, can be found in Appendix C, in Tables C.2 and C.1 respectively.

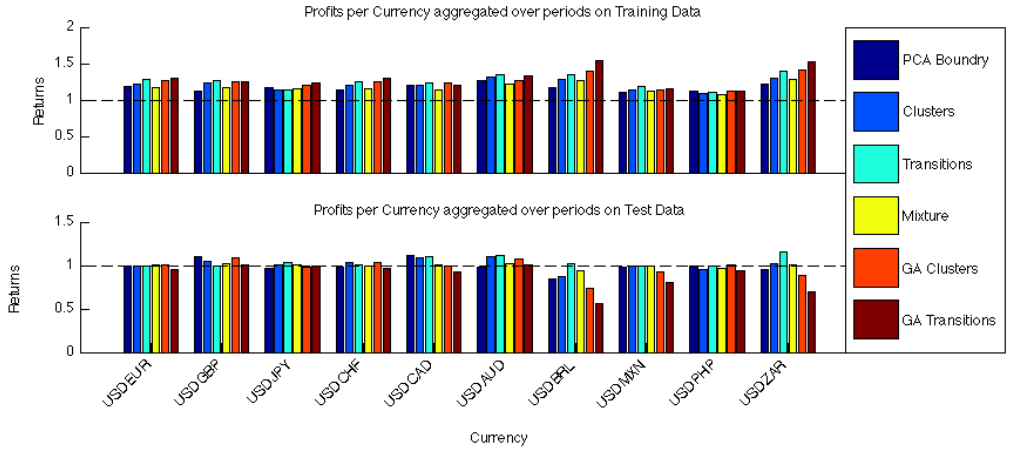

Figure 6.2: *A bar chart, showing the profits per currency (aggregated over periods) on foreign exchange data. The dotted black line represents a fair return, as one would get from an investment bank. Notice how the profits in the training data increase with each change of strategy (barring the mixture of experts baseline), however on the the test sets, the genetically optimised cluster and transition strategies under perform. This is most likely due to over-fitting on the test set. Also notice how the two least stable currencies (BRL and ZAR) have a greater variability in their returns.*

and transition strategies which perform much worse than their greedy counterparts, and even worse than the most naïve systems.

This shows that the more we make use of structures found, the more profitable the system becomes. The most profitable of the systems examined used clusters, temporal ordering of clusters and movements between clusters, but was restricted to choosing only the best single cluster in which to buy, and the best single cluster in which to sell. Allowing the system to have any number of buy and sell clusters results in a combinatoric explosion. When the chosen meta-heuristics attempt to solve the massive search space, they either cannot find optimal solutions (the space is too large), or they over fit the solution to the training set, such that it becomes ineffective on the test set.

Figure 6.2 shows the profit per currency (when aggregated over all periods). An interesting feature observed in this figure is that not all currencies are equal. The majors (EUR, GBP, JPY and CHF) afford very little profits, while the minors and emerging markets on the other hand, appear to have more exploitable inefficiencies. This observation agrees with Pukthuanthong *et al.* [54] who suggest that greater opportunities exist in emerging and exotic currencies, despite the higher trading costs. It is also evident that some of the biggest differentiating factors exist in the highly volatile markets

such as the BRL and ZAR.

Overall statistics about the profitability of the different strategies (aggregated over all currencies and all periods) are provided in Table 6.1. When comparing the different strategies which make use of greedy search and structures (Dim Red[2], Cluster, Transition), one can see that as more of the structure is used, the system improves in almost every single way. When examining the test set, not only does the profit increase, but the variance decreases. Quite importantly, the minimum loss experienced by the systems differs wildly: the Dimensionality Reduction strategy lost 94.3% of its assets in its worst year, while the Cluster strategy lost 74.3%. The mixture of experts strategy, which is supposed to be very risk averse, happened to lose 47.6% in its worst year, while the Transition based strategy only lost a maximum of 22.2%. These minimum losses have a huge effect on the compounded returns. The compounded returns tells the story of what happens if, starting with a wealth of 1, one invested according to a particular strategy for all the periods and all the currencies. The Transition strategy is the only one to record a profit after 90 years of trading, growing over 15 times in value, above and beyond that of a fair return (investing in a risk free strategy). This equates to a profit of just over 3% per annum above that of a fair return.

Another interesting feature is the skew of the distribution of returns. The The Dimensionality Reduction strategy appears to have a fairly even distribution of profits, while the Cluster strategy has a slight positive skew. The Transition strategy appears to have quite a large positive skew, indicating that the tails are definitely heavier on the positive side. Positive skew is a good characteristic to have as it indicates outliers in the profitable (positive) direction but not in the negative direction. The mixture of experts has a large negative skew, but this is mainly due to a few outliers. These effects can be seen in Figure 6.3.

6.1.1 Summary and conclusions

In summary, the more complex the structures used by the decision strategy, the more profitable, and less volatile the returns. The baseline strategy of a mixture of experts is not profitable, but is not very risky due to its dispersion of risk across the experts. The genetic algorithms appear to be over-fitting during the training period, and thus under performing in the test period. They also have a much greater volatility than the greedy structures, which makes them unsuitable for real-world applications of these structures.

[2]Dim Red is a contraction for Dimensionality Reduction, used in order to conserve space in the tables.

	Dim Red	Cluster	Transition	Mixture	GA Cluster	GA Transition
Training Set (Foreign Exchange data)						
Mean Profit:	1.16888	1.20863	1.25487	1.17347	1.25027	1.29436
Variance of Profit:	0.02441	0.0296	0.02993	0.01527	0.03466	0.06015
Minimum Profit:	1	1.0088	1.0399	1.01224	1	0.99228
Median Profit:	1.14096	1.16104	1.22048	1.14643	1.19558	1.22875
Maximum Profit:	1.89796	2.10207	2.11788	1.62486	1.79355	2.47318
Sharpe Ratio:	1.0808	1.21257	1.47315	1.40394	1.34433	1.20017
Test Set (Foreign Exchange data)						
Mean Profit:	0.98920	1.01171	1.03985	0.99445	0.97364	0.88447
Variance of Profit:	0.03443	0.02529	0.02242	0.00405	0.04122	0.04642
Minimum Profit:	0.05746	0.25754	0.7782	0.52409	0.01002	0.06074
Median Profit:	1	1	1.01152	0.99772	0.99254	0.9181
Maximum Profit:	1.79751	1.7344	1.74749	1.11161	1.75079	1.27882
Sharpe Ratio:	-0.05819	0.07363	0.26612	-0.0873	-0.12983	-0.53621
Skew of Returns:	-0.0344	0.3246	2.1891	-4.6053	-0.7901	-1.4098
Compounded Returns	0.0214	0.75433	15.15027	0.47738	0.00057	0
Profit p.a. over 90yrs	0.9582	0.9969	1.0307	0.9918	0.9203	0.8365

Table 6.1: *Table showing statistical analysis of the different strategies on foreign exchange data. Notice the increase in profitability in the training set when using more complex structures and optimisation techniques, however the genetically optimised strategies perform poorly on the test data, indicating possible over-fitting. The Transition strategy is the only one to achieve a profit over the entire 90 year test period, and had the highest mean profit. It also has a positive skew on the distribution of returns. This shows that there is merit in using structures, and in particular the transitions between clusters, as an indicator of when to buy and sell in foreign exchange markets.*

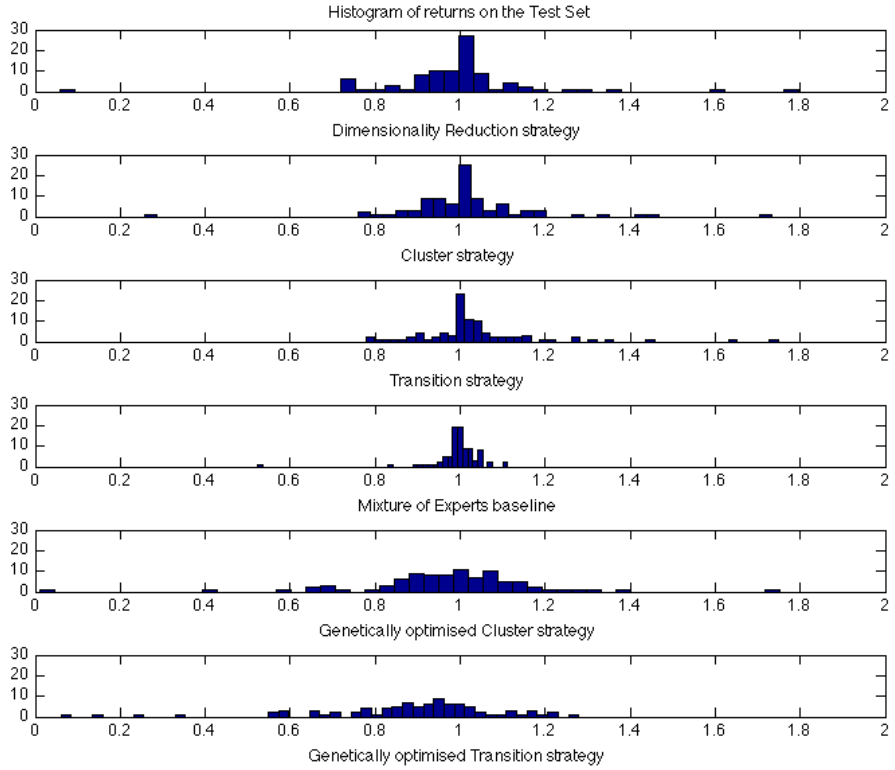

Figure 6.3: *A Histogram showing the distribution of profits for each of the trading strategies on the test set of foreign exchange data. Notice how the Dimensionality Reduction and Cluster strategies appear to have an even distribution over profits. On the other hand, the Transition strategy has a positive skew, and the mixture of experts has a negative skew. Both of the genetically optimised strategies have a much larger variance in their returns.*

Only the final Transition strategy using greedy search, appears to be profitable above a fair return when taking all 90 years of test data into account.

These results show merit in using the kind of structures analysed in this paper, and how they could possibly be used as a profitable trading strategy in foreign exchange markets.

6.2 Comparisons between Models

Section 6.1 compared the different strategies used, and identified the Transition strategy as being the best strategy for foreign exchange markets. It also identified genetic optimisation as being counter productive due to over-fitting. This section takes a closer look at the results of the different strategies on the different data sets (as described in Section 5.3). Because of their stochastic nature, observed ineffectiveness, and the extra space required to display their results, the genetically optimised strategies have been omitted from the results in this section.

Section 5.3 described the different data sets that the strategies would be tested against, namely: a random walk model, a GARCH model, real foreign exchange data, foreign exchange data where daily movements are permuted and foreign exchange data in which chunks of daily movements are permuted. These data sets are used to ensure that the strategies are not cheating in any way (mitigating against bugs in code), as well as to determine where profitable structures can be found.

6.2.1 Random Walks

Starting with the most simple model, Table 6.2 provides the profitability statistics for the random walk data. The same tests were run 50 times in order to get overall statistics. Although profitable in the training set, none of the systems are profitable in the test set. This is as expected, as there should not be any structure in randomly generated data. It is interesting, however, that the Transition based system still out performs the others, and is less risky and has a higher minimum than the other strategies using structure.

6.2.2 GARCH models

The next model to investigate is that of GARCH. These models are interesting because they maintains the structures of volatility clustering and trending as is present in fin-

	Random Walk Training Set			
	Dim Red	Cluster	Transition	Mixture
Mean Profit:	1.21196	1.24355	1.36231	1.24906
Variance of Profit:	0.07582	0.06187	0.11906	0.06967
Minimum Profit:	0.9656	1.00862	1.06309	1.05973
Median Profit:	1.13361	1.15414	1.22621	1.14067
Maximum Profit:	2.8209	2.15048	2.65325	2.43191
Sharpe Ratio:	0.76976	0.97912	1.05004	0.9436
	Random Walk Test Set			
Mean Profit:	0.98421	0.97133	0.99794	0.99440
Variance of Profit:	0.02495	0.02831	0.0223	0.00373
Minimum Profit:	0.57638	0.51637	0.62473	0.76823
Median Profit:	0.97931	0.96507	0.99482	0.9939
Maximum Profit:	1.63595	1.92319	1.58248	1.20349
Sharpe Ratio:	-0.09997	-0.17043	-0.01379	-0.09168
Skew of Returns:	1.5788	1.7749	0.9726	0.4324
Compounded Returns	0.08371	0.02108	0.31672	0.51068
Profit p.a. over 90yrs	0.9728	0.9580	0.9873	0.9926

Table 6.2: *Table showing the profitability statistics for the different strategies on the data generated by a random walk, with standard deviations taken from the original foreign exchange data.*

	GARCH Training Set			
	Dim Red	Cluster	Transition	Mixture
Mean Profit:	1.08887	1.10856	1.16271	1.09342
Variance of Profit:	0.02789	0.04043	0.06313	0.03467
Minimum Profit:	1	0.97741	1	0.99799
Median Profit:	1.03514	1.06317	1.11621	1.06101
Maximum Profit:	4.07394	5.38642	5.97338	6.28454
Sharpe Ratio:	0.53215	0.53993	0.64761	0.50172
	GARCH Test Set			
Mean Profit:	0.98073	0.97847	0.99568	0.98867
Variance of Profit:	0.00803	0.00605	0.00722	0.00074
Minimum Profit:	0.482	0.47198	0.53099	0.83203
Median Profit:	1	0.99689	1	0.99267
Maximum Profit:	1.60528	1.6079	2.84236	1.16721
Sharpe Ratio:	-0.21506	-0.27686	-0.05085	-0.41788
Skew of Returns:	0.6489	-0.0732	6.9522	-0.2293
Compounded Returns	0.39192	0.102	0.47328	0.62049
Profit p.a. over 90yrs	0.9766	0.9753	0.9926	0.9883

Table 6.3: *Table showing the profitability statistics for the different strategies on the data generated by a GARCH model with coefficients learnt from the original foreign exchange data.*

	Completely Permuted Training Set			
	Dim Red	Cluster	Transition	Mixture
Mean Profit:	1.19006	1.20948	1.28269	1.18071
Variance of Profit:	0.03064	0.02588	0.04359	0.01878
Minimum Profit:	0.96902	0.97492	1	0.98751
Median Profit:	1.15596	1.18294	1.24066	1.15143
Maximum Profit:	2.92384	3.25334	3.88129	3.14799
Sharpe Ratio:	1.08573	1.30212	1.35397	1.31855
	Completely Permuted Test Set			
Mean Profit:	0.97559	0.9859	0.99549	0.98799
Variance of Profit:	0.01653	0.01312	0.01213	0.00322
Minimum Profit:	0.27885	0.44847	0.43425	0.61967
Median Profit:	0.98926	1	1	0.98988
Maximum Profit:	1.87994	1.85055	2.1427	1.35981
Sharpe Ratio:	-0.18988	-0.12314	-0.04091	-0.21156
Skew of Returns:	0.0401	0.5831	0.8977	-0.2283
Compounded Returns	0.2315	0.20648	0.3091	0.38298
Profit p.a. over 90yrs	0.9666	0.9792	0.9894	0.9863

Table 6.4: *Table showing the profitability statistics for the different strategies on the data generated by randomly permuting the daily price movements of foreign exchange data.*

ancial time-series [2, 14]. The same tests were run 50 times in order to get overall statistics. Again, none of the strategies make any profits. This suggests that the profitable structures found on real data are not only due to volatility clustering or trending, as no profits were rendered on generated data with these features. It is possible that the profits arise from the large skews and excess kurtosis observed in real foreign exchange data.

6.2.3 Permuted Data

To take the excess kurtosis and skew into account with a generative model, data was generated by permuting the daily movements, and taking the cumulative sum of these movements and the initial data points. This produces a series with the same statistical properties and with the same start and end locations as the original data, but with

	Permuted Segmented Training Set			
	Dim Red	Cluster	Transition	Mixture
Mean Profit:	1.21036	1.22637	1.29904	1.21959
Variance of Profit:	0.19251	0.05833	0.26082	0.96762
Minimum Profit:	0.97072	0.85605	1	0.99164
Median Profit:	1.16059	1.17902	1.23717	1.14711
Maximum Profit:	16.66563	6.1028	18.68283	40.20839
Sharpe Ratio:	0.47944	0.93727	0.58555	0.22323
	Permuted Segmented Test Set			
Mean Profit:	0.98107	0.98203	0.99837	0.98611
Variance of Profit:	0.01958	0.01887	0.0138	0.00516
Minimum Profit:	0.00841	0.13752	0.18628	0.42162
Median Profit:	1	1	1	0.98856
Maximum Profit:	2.32319	2.49915	2.16977	2.4921
Sharpe Ratio:	-0.13528	-0.13084	-0.01384	-0.19323
Skew of Returns:	-0.4896	0.2139	1.6062	4.6458
Compounded Returns	0.18732	0.03948	1.01648	0.35541
Profit p.a. over 90yrs	0.9673	0.9705	0.9915	0.9836

Table 6.5: *Table showing the profitability statistics for the different strategies on the data generated by randomly permuting segments of daily price movements of foreign exchange data.*

different trends in between. The same tests were run 50 times in order to get overall statistics. As one might expect, this has the ability to destroy all volatility clustering present in the original data, as well as any auto-correlations or long-memory if they exist. The results show that no profits are earned after training on this kind of permuted data. This indicates that whatever profitable structures were found on the original data were destroyed when permuting every innovation.

In order to detect whether any form of permutation destroyed these structures, a final data set was created. This separated the innovations into segments of between 40 and 60 days in length, and permuted the segments randomly before taking the cumulative sum to once again find a series that appears to be financial time-series. The same tests were run 50 times in order to get overall statistics. Interestingly, none of the profits observed in the original data are present in this test data. This suggests that none of the structures which were exploited in the original data are being preserved when

trained on this kind of segment-wise permutation, and thus any permutations which disrupt the test and training sets destroy the profitable structures.

6.2.4 Summary and conclusions

The four models presented in Chapter 5 were tested on the data sets explained in Section 5.3. None of the strategies were profitable on any of the generated data, but some showed signs of profitability on the original foreign exchange data. This suggests that the generative models are not able to fully capture the dynamics of the real data. To address this issue, the systems were tested on the original data in which the innovations were randomly permuted. No profits were rendered on the full permuted data, and no profits were found on the data in which chunks of innovations were permuted. This suggests that none of the structures which are found to be profitable in the original data are retained in the permuted chunks, or when permuted on the daily level. The profitable structures found are not due to trending or volatility clustering, as no profits were found on the GARCH models.

6.3 Critical Analysis of Results

Although the sections above shed interesting light into the idea of exploitable structures within financial time-series, there are a number of criticisms which must be addressed.

The most noticeable criticism that needs to be made is that the results and conjectures mentioned above are based on differences of a few percent in mean profits and annual returns realised between strategies and models. Although this is tested on over 32000 days of price movements across very different currencies, the time-series are inherently stochastic. These can be mitigated against by introducing further data sets, and treating the current data as a validation set. As this project is intended to be a proof-of-concept, ten years of data over ten different currencies was perceived as sufficient.

The same criticism is due for the testing of genetic algorithms. The size of the search space, and the time required for the GA to optimise a single solution, made it impossible to generate the number of trials required to get statistically significant results as to the efficacy of genetic algorithms as optimisers of the strategies. That being said, the huge variance in the trials which were run, showed that they were unacceptable for real world applications of a system. This is especially true if the

strategies were to operate in as near to real-time as possible. This project is also not intended to generate the maximum profit possible – the GAs were included to provide some insight into the use of meta-heuristics for this combinatoric problem, but in no way suggests that these were implemented in an optimal fashion.

It is suggested that the feature-space is 'information-rich'. No metrics or tests from information theory were used to justify this claim because the amount of information within a financial time-series is unknown.

The Simulation Environment, as it stands, calculates profits based on the time spent in the market. This makes it possible for situations to occur in which no trades are made, rendering neither a loss nor a profit. Based on the profitability of the strategies observed, not playing the game is a fairly competitive strategy. The other concern this method raises is that trading a single time in an entire year, and being marginally profitable over that time results in massive annual profits, as the profits are converted to an annual return on the time spent in the market. That said, if the system is able to identify single and highly profitable jumps, it is not seen as a behaviour which should be discouraged. These effects could have been avoided in a number of ways. Firstly, by not scaling the profits to be annualised, and taking the profit as though the strategies had traded over the entire year. This was decided against, as it then makes clean calculations of profits per annum difficult to calculate. Secondly, by forcing the strategies into the USD at the start of the training and testing periods, and forcing the final signal to be a sell on which ever currency it stayed on. This was decided against because it often engendered a buy-hold strategy, which did not show the merits of the systems ability to exploit all types of situations. Finally, the strategies could have been forced in the the USD at the start of the training period, which ever market the training period ended on, would be the starting position for the market in the test period. Unfortunately this strategy was only considered in the final week of the project, after data had been generated, and with little time to implement and test the system thoroughly.

Further to this, the systems were only trained for a year and tested for a year. There are many different combinations of training and test lengths which could show other interesting features, or provide more concrete results. There were not fully investigated due to the already overwhelming number of results, and the limited time-frame in which to finish the entire project.

Although other methods were investigated, the Euclidean distance was the only distance metric investigated with rigour. It is quite possible that the cosine, Mahalan-

obis or one of the many other distance metrics might be better suited to this task. Under the given time constraints and due to the exploratory nature of this project it did not seem feasible to vigorously test all possibilities. The same argument applies to the unsupervised learning tasks. There was good justification for using hierarchical agglomerative clustering, but other methods such as Latent Dirichlet Allocation would have been interesting to investigate, and could possibly have generated further profits, however, time was short.

Finally, it took over 5000 lines of MATLAB code (excluding toolboxes) to implement and test the system and to generate the presented results. Although the utmost care was taken in ensuring no errors were made, this is always a possibility that bugs creep in when programmes get this large and complex.

6.4 Summary

This chapter presented the results of testing the strategies on different data sets. Although not entirely conclusive due to the small differences in margins, the results suggest that the kinds of structures analysed in this paper, and could possibly be used as a profitable trading strategy in foreign exchange markets. More specifically, the results show that the more complex the structures used by a strategy, the more profitable the strategy becomes. The most profitable strategy used the location and movements of points through the feature space in order to generate an overall profit of 3% per annum above a fair return, when compounded on the returns from tests on the entire set of foreign exchange data. None of the other strategies were able to generate profits over the entire data set. Genetic algorithms were used to deal with the combinatoric explosion created when extending the search space to include all possible combinations of clusters and transitions. When tested, however, the GAs appeared to over fit to the training data, resulting in lower returns than any of the other strategies. The GAs also displayed a much greater volatility in the test set. This volatility, reduced performance and the additional computational effort required to use GAs as optimise techniques, making them impractical for real-world scenarios.

Results of the different strategies on different types of data were also presented in this chapter. None of the generative models appeared to have any inefficiencies, as no profits were rendered on any of them. This indicates that inefficiencies are not created because of volatility clustering or trending, as GARCH models take these phenomena into account. It was then investigated whether these inefficiencies were due to the

excess kurtosis or large skew present in some of the foreign exchange data. To test this hypothesis, the daily movements of the original data were permuted in order to generate a number of different time-series with the same statistical properties. These did not prove to have any inefficiencies when permuted at the daily level, or at the segment level as no profits were rendered. This result suggests that permuting the movements in segments or daily movements destroys all of the structure utilised to earn profits on the original data. Incorporating these results and those generated by the GARCH tests, it appears as though some other structures besides the auto-correlation of variance and trending of financial data, and these were exploited in order to generate profits which the Efficient Market Hypothesis dictates shouldn't be there.

Finally the chapter provided some criticisms for the system as it currently stands.

Chapter 7

Conclusion

7.1 Problem Statement Revisited

This study set out to find potential structure within a single financial time-series, and to exploit any such discoveries for financial gain. In the process of realising these objectives a number of intermediate goals were achieved.

An information-rich feature space was created which was able to sufficiently characterise the current market state. It was deemed successful after measuring the similarity between it's representation of historical foreign exchange data, and that same data after being shifted, scaled and perturbed.

The feature space was then studied and explored in order to identify structures. Certain regularities were found in the feature space through hierarchical agglomerative clustering. There was also some structure evident in the time spent in any one cluster (exponential decay), and the movements of consecutive points between clusters (the transition probabilities).

Following this, three different methods of exploiting the structures were identified and detailed. The first of which used dimensionality reduction techniques on the feature space to take information from all experts into account simultaneously. This dimensionality reduction was also contextualised by scaling the standardised data by the experts' profitability in the training set. The second method used the clusters themselves as indicators of whether to buy, hold or sell, based on the proximity of the current data point to one of these identified clusters. The final method took the temporal ordering and movement of consecutive points between clusters as possible buy, hold and sell indicators. These three strategies were compared to a naïve baseline strategy which made a simple portfolio of investments by letting each expert follow its own

rules, with a share of the total asset proportional to their profitability in the training set.

Finally the implementation of these systems was explained before testing them thoroughly on ten years of foreign exchange data across ten very different currencies. The models were also tested on generated data based on the historical data.

The general results suggest that profitable structures do exist in real foreign exchange data as the most profitable system consistently performed well enough to average a profit of 3% per annum above that of a risk-free fair return one that would get from a bank or US treasury bill.

Additionally, no such profits were found in the generated data. This suggests that the generated data used in this study does not capture all of the structures present in real-world data. A further test set was created from the original financial data by randomly permuting the daily movements. When permuted at the daily level and at the segmented level (limiting permutation of the data to discrete segments), no profits were found. This suggests that all profitable structure was destroyed by the permutation process.

Genetically optimised variants of the most profitable strategies were also tested in order to see if profitability could be optimised through the application of meta-heuristics. It was found that the GAs were excellent at identifying good fits in the training data, but these optimisations (of combinations of clusters or transitions) did not generalise to the test set. The strategies optimised by the GAs were both less profitable and also had a much larger variance, suggesting that they are unsuitable for the task of optimising cluster and transition combinations.

Based on these results and criteria, it can be concluded that this project is a success, and motivates future research into this area of computational finance.

7.2 Future Work

The results found in this project suggest that sustained returns can be rendered which are sufficient enough to be profitable when continually compounded after 90 different tests across 10 very different currencies. This shows merit in the current system, and thus is an avenue worth exploring in the future. As mentioned in the Critical Analysis of Results, many of the potential flaws were due to a lack of time in which to fully address these problems. This coupled with the interesting nature of the results leaves much room for future work. The rest of this section describes some of the possible additions that could be made to this project.

7.2.1 Optimisation Problems

There is much to be optimised in the system as it stands. Firstly one could follow in the footsteps of Lin, Cao, Wang, and Zhang [33], and identify methods of optimising the technical trading tools (as used in technical analysis) to be best suited to the current data set by finding optimal time windows for each of the rules. Once the trading rules have been optimised, further work can be done in intelligently optimising the total number of clusters to use, the number of active clusters, and an efficient method of labelling the active clusters as buy or sell clusters. The results of any such optimisations would be directly transferable to the transition space, however it should be noted that solutions based on transitions between cluster rather than clusters themselves will have a much higher complexity than the Cluster implementation.

The complexity of the transitions based strategy could be reduced by not taking the direction of the transition into account, making the transition matrix symmetric, but it is unknown whether the loss of this directional information will hamper the efficacy of the solution.

7.2.2 Structural Problems

The methods used in this project are by no means perfect. There are a host of different characterising statistics which could be used to describe the current market state. Econophysics provides an interesting arena to explore for such descriptive statistics [38].

Above and beyond the characterising statistics, there are a number of different distance functions which could be applied to the feature space. Ideally a distance function would minimise the distances between similar trending systems. The cosine and Mahalanobis distances were briefly explored empirically, but it they did not provide noticeable improvements to the system. They would need to be tested and analysed thoroughly before completely ruling them out.

Bespoke distance metrics could also be designed for this specific problem. Some very different structures would emerge if one were to try and minimise the distance between objects in feature space as well as time. Alternatively clusters could be generated which minimise the transitions between clusters in an iterative procedure similar to the Expectation Maximisation algorithm. Both of these would yield clusters which are more persistent in times than the clusters identified above (not affording as many transitions), and could be used to detect regime switches or drift in the time-series.

7.2.3 Learning Problems

The simple dimensionality reduction techniques used in this project were applicable and justified, but many other potentially applicable dimensionality reduction techniques were not examined.

Furthermore, this project only investigated hierarchical agglomerative clustering as the means of performing unsupervised learning. This was justified in that a non-parametric approach was required for exploration, and the hierarchical nature of this technique allowed for easy iteration between differing numbers of clusters. However, there are many other forms of unsupervised learning which could be applicable to this problem. An example of this would be using Latent Dirichlet Allocation to identify possible distributions over clusters [59]. Another interesting idea would be to expand on the concept of building finite state automata from the cluster and transition investigations and using graphical models to represent the movements of points between different clusters over time. Methods from graphical models could also be applied to this situation.

7.2.4 Exploiting Structures Found

The current project identified clusters and transition probabilities between clusters. Although the transitions themselves were utilised, little effort was made in using the transition probabilities themselves. These transition probabilities are fairly symmetric when looking at Figure 4.4, however in higher numbers of clusters the transition probabilities become quite asymmetric. This suggests that predictions as to the next cluster could be made by identifying the probability of making such a transition. Building some game theoretic ideas into such a prediction system, the possible regrets for each space could be calculated in order to make preemptive decisions. This was a possible extension to the project given enough time, and the curious reader is pointed towards Cesa-Bianchi and Lugosi [7] for more information on how this could be implemented.

7.2.5 Variations of Simulation Environment

Another one of the criticisms made about this system was in the way in which profits were calculated. The system currently makes use of the time spent in the market in order to calculate annual returns for each strategy. This is not necessarily the correct way to do this. Three alternatives were suggested to the current implementation, all of

which should be investigated thoroughly in order to find a profit calculating strategy with engenders an ideal trading strategy. To avoid repetition, the reader is pointed towards Section 6.3 for more information on these ideas.

7.2.6 Other Interesting Ideas

Each of the technical trading tools makes use of moving averages in order to smooth the stochastic noise in the time-series. It was suggested in Chapter 3 that different window sizes should be used in order to capture short- medium- and long-term perspectives of the data. It was also suggested that intuitively a longer moving average is necessary for highly variant systems, as there is more noise to filter out than in stable, low variance systems. Chapter 2, however, mentioned that volatility (and hence standard deviation) of innovations changes through out time, but tends to cluster such that there are periods of high volatility and periods of low volatility. It therefore might make sense to have an adaptive window size for each of the trading rules. Basing a window size on the sample variance and/or sample entropy of the system could provide a more flexible trading rule, which is able to adapt and exploit changes in volatility more efficiently.

Appendix A

Genetic Operators

This appendix provides further information on the details of genetic operators.

Encoding is the mechanism by which the parameters of the objective function are converted into a the chromosome. Binary encoding is most common among encoding methods, in which a real value is converted into a bitstring, however this project uses GAs to solve a combinatorial problem, and so no encoding is necessary. In general combinatoric problems (such as the travelling salesman), one has to ensure that no constraints are violated (such as visiting the same city twice), if this is the case, then the chromosome needs to be *repaired* [40]. There are no such constraints in the optimisation problem at hand, and so methods of repairing our out of the scope of this project.

Initialisation is done at the beginning of the optimisation routine. Traditionally the initial population is randomly generated to span the entirety of the search space, however one can 'seed' the population to generate candidate solutions in an area which is known to perform well.

Selection is the process by which the parent chromosomes are selected from the population according to their fitness. The function used to calculate the fitness of each individual is intuitively called the *fitness function*. The fitness function decodes the chromosome (if necessary), and the objective function is evaluated with these parameters. The performance of the chromosome is that chromosomes fitness. The process of Natural Selection [12] is emulated by allocating parent chromosomes with a higher fitness a higher probability of being selected. This mimics the natural selection, in which a creature's fitness enables it to survive and therefore be more prolific in its current environment.

One can perform the selection operation in a number of different manners, the most

popular of which listed below:

- *Roulette Wheel Sampling* is a computer-friendly implementation of the Stochastic Universal Sampling, the original method proposed by Holland. Each chromosome is assigned a probability proportionate to its fitness against the total fitness. A random number between 0 and 1 is then generated, and all the elements of the population are iterated through, adding their respective fitnesses together until the random number is reached. The individual whose fitness is the last to be added is then selected. This is representative of the expected selection probabilities [40].

- *Tournament Selection* pits two randomly selected individuals from the population against each other. The fittest of the two will be selected as a parent for the next generation, but both will be returned to the population to be selected again [40].

- *Rank Selection* ranks individuals based on their fitness, and then selects them based on their rank, instead of their actual fitness. This prevents the GA from converging too quickly [40].

- *Elitism* retains the same number of best individuals at each generation, to ensure that best solutions are not randomly lost. Elitism can be used in conjunction with any of the other selection mechanisms [40].

Crossover is the process in which parent chromosomes are recombined in order to form offspring in the next generation. This is achieved by swapping corresponding elements between parent chromosomes in order to form child chromosomes [23]. The points at which the swapping occurs are randomly selected by one of the following methods:

- *Single point crossover* is when all elements before a randomly selected point on the parent chromosome are combined with all the elements after the crossover point from the other parent, and *vice versa*. This method has its shortcomings, as not all possible combinations can be produced this way [40].

- *Two point crossover* is when the chromosomes cross over at two points along the bitstring, and effectively swap a chunk of information, which closely mimics mitosis (the recombination of chromosomes in sex cells). This allows for considerably more combinations than single point crossover, but still does not allow for every possible combination to occur [40].

- Some practitioners are advocates for *parametrised uniform crossover* or *scattered crossover*, where a probability is calculated for each bit to swap over. However, having too many crossover points can be disruptive when trying to find an optimal solution [40].

- It is also possible to create *crossover hotspots* where crossover is far more likely to occur at certain hotspots than at any other point, which is useful to retain dependencies [40].

Mutation is used to maintain genetic diversity in the population. This prevents the entire population converging towards the single fittest individual, and helps explore search space by perturbing existing solutions.

This is achieved by randomly changing values in the child chromosome. If the chromosome represents real valued parameters, mutation will add a normally distributed value to the element in the vector. The parameters of the distribution can be dynamic, thus one can allow variance to fluctuate depending on certain population statistics. If the chromosome is a combinatoric in nature, then mutation will replace the vector with an element randomly selected from a pool of possibilities.

The most common forms of mutation operate as follows:

- *Uniform Mutation* is a two-step process. First a vector (equal in length to that of the chromosome) is randomly generated, containing numbers between 0 and 1. If the value of any element is less than that of the *mutation rate*, then those elements will be mutated. The second step is applying the mutation to those elements, be it adding a random number, or replacing a combinatorial possibility.

- *Swapping Genes* is when two randomly selected points within in the chromosome are selected, and swapped. This is not always viable, particularly in real valued problems, however it has proven useful in solving the travelling salesman problem.

- *Inversion* is another of Holland's original propositions. This mimics biological mechanisms in which a chunk of the chromosome (between two randomly selected points), is extracted, inverted, and replaced in the chromosome. As with Swapping Genes, this is not always viable in real valued problems.

Appendix B

Exact Trading Rule Specifications

This appendix describes the exact parameters for the trading rules used.

Short-term: 'MACD (4:6:12)';
'Momentum (6)';
'RSI (2)';
'Filter (6)';
'Trending (2:5)';
'Moving Ave (Dist:6)';
'Moving Ave (Rate:6)';
'2nd Deriv Moving Ave (Rate:6)';
'3rd Deriv Moving Ave (Rate:6)';
'Sample Variance (6)';

Medium 'MACD (9:12:26)';
'Momentum (12)';
'RSI (5)';
'Filter (15)';
'Trending (5:20)';
'Moving Ave (Dist:12)';
'Moving Ave (Rate:12)';
'Sample Variance (12)';
'2nd Deriv Moving Ave (Rate:12)';
'3rd Deriv Moving Ave (Rate:12)';

'Sample Entropy (100)';

Long 'MACD (15:26:64)';
'Momentum (26)';
'RSI (12)';
'Filter (26)';
'Trending (20:50)';
'Moving Ave (Dist:26)';
'Moving Ave (Rate:26)';
'2nd Deriv Moving Ave (Rate:26)';
'3rd Deriv Moving Ave (Rate:26)';
'Sample Variance (26)';

Appendix C

Additional Results

This appendix provides additional results at levels of granularity too fine to be presented in the dissertation.

	Profitability per currency					
	Dim Red	Cluster	Transition	Mixture	GA Cluster	GA Transition
USDEUR	0.988	0.9962	0.9902	1.0019	1.001	0.9555
USDGBP	1.0952	1.0441	0.9976	1.0206	1.0942	1.005
USDJPY	0.972	1.0043	1.0345	1.0093	0.9843	0.9922
USDCHF	0.9854	1.0388	1.0111	0.988	1.0306	0.9696
USDCAD	1.1119	1.0874	1.1012	1.0074	0.9898	0.9252
USDAUD	0.9817	1.1002	1.1187	1.0169	1.0789	1.0015
USDBRL	0.8409	0.8755	1.0198	0.9352	0.7315	0.5622
USDMXN	0.979	0.9895	0.9868	0.9916	0.9275	0.7988
USDPHP	0.9874	0.9586	0.9886	0.9717	1.0122	0.935
USDZAR	0.9505	1.0225	1.1501	1.002	0.8863	0.6996
Mean Profit:	0.9892	1.01171	1.03986	0.99446	0.97363	0.88446

Table C.1: *Table showing the mean profitability per currency (aggregated over periods) on the test sets of foreign exchange data. This data is visualised in Figure 6.2.*

	Profitability per period					
	Dim Red	Cluster	Transition	Mixture	GA Cluster	GA Transition
2001 - 2002	0.9014	0.9168	1.0235	0.9596	0.9464	0.8495
2002 - 2003	0.9698	0.975	0.9931	0.9969	0.8796	0.8087
2003 - 2004	1.006	1.0028	1.0482	1.0123	1.0303	0.8624
2004 - 2005	0.9791	0.9944	1.0203	1.0107	0.9456	0.8715
2005 - 2006	0.9733	1.0077	1.0113	1.0031	0.9673	0.9464
2006 - 2007	0.9798	0.9722	1.0703	0.9895	0.9388	0.876
2007 - 2008	0.9603	1.0085	1.0203	0.9967	0.9858	0.9134
2008 - 2009	1.1258	1.1656	1.1546	0.9905	1.0946	0.9628
2009 - 2010	1.0074	1.0623	1.0172	0.9908	0.9744	0.8696
Mean Profit:	0.9892	1.01171	1.03986	0.99446	0.97363	0.88446

Table C.2: *Table showing the mean profitability per period (aggregated over currencies) on the test sets of foreign exchange data. This data is visualised in Figure 6.1.*

Bibliography

[1] James F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843, 1983.

[2] T. Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

[3] T. Bollerslev, R.Y. Chou, and K.F. Kroner. ARCH modeling in finance: A review of the theory and empirical evidence. *Journal of Econometrics*, 52(1-2):5–59, 1992.

[4] G.E.P. Box, G.M. Jenkins, and G.C. Reinsel. *Time series analysis: forecasting and control*. Holden-day San Francisco, 1970.

[5] W. Brock, J. Lakonishok, and B. LeBaron. Simple technical trading rules and the stochastic properties of stock returns. *Journal of Finance*, 47(5):1731–1764, 1992.

[6] Thomas Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, USA, 1996.

[7] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[8] S.H. Chen. *Genetic algorithms and genetic programming in computational finance*. Kluwer Academic Pub, 2002.

[9] R. Cont. Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance*, 1(2):223–236, 2001.

[10] P.H. Cootner et al. *The random character of stock market prices*. MIT press Cambridge, MA, 1964.

[11] M. Costa, A.L. Goldberger, and C.K. Peng. Multiscale entropy analysis of biological signals. *Physical Review E*, 71(2):21906, 2005.

[12] C. Darwin. The origin of species. 1859.

[13] K.A. De Jong. Analysis of the behavior of a class of genetic adaptive systems. 1975.

[14] R.F. Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica: Journal of the Econometric Society*, 50(4):987–1007, 1982.

[15] R.F. Engle. Statistical models for financial volatility. *Financial Analysts Journal*, 49(1):72–78, 1993.

[16] RF Engle. *Anticipating Correlations*. Princeton University Press. Forthcoming, 2008.

[17] C. Eom, G. Oh, and W.S. Jung. Relationship between degree of efficiency and prediction in stock price changes. *Quantitative Finance Papers*, 2007.

[18] E.F. Fama. The Behavior of Stock-Market Prices. *Journal of Business*, 38(1):34–105, 1965.

[19] E.F. Fama. Efficient capital markets: A review of theory and empirical work. *Journal of Finance*, 25(2):383–417, 1970.

[20] E.F. Fama and M.E. Blume. Filter rules and stock-market trading. *Journal of Business*, 39(S1):226, 1966.

[21] S.M. Focardi. Clustering economic and financial time series: Exploring the existence of stable correlation conditions. *The Intertek Group*, 2005.

[22] S. Ghashghaie, W. Breymann, J. Peinke, P. Talkner, and Y. Dodge. Turbulent cascades in foreign exchange markets. *Nature*, 381(6585):767–770, 1996.

[23] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1989.

[24] C. Gouriéroux. *ARCH models and financial applications*. Springer Verlag, 1997.

[25] C.W.J. Granger. Some properties of time series data and their use in econometric model specification. *Journal of econometrics*, 16(1):121–130, 1981.

[26] A. Hald. *A History of Probability and Statistics and their Applications before 1750*. Wiley-IEEE, 2003.

[27] J.D. Hamilton and R. Susmel. Autoregressive conditional heteroskedasticity and changes in regime. *Journal of Econometrics*, 64(1-2):307–333, 1994.

[28] D.J. Hand, H. Mannila, and P. Smyth. *Principles of data mining*. The MIT press, 2001.

[29] N. Jegadeesh and S. Titman. Cross-sectional and time-series determinants of momentum returns. *Review of Financial Studies*, 15(1):143, 2002.

[30] M.C. Jensen and G. Bennington. Random walks and technical theories: Some additional evidence. *Journal of Finance*, 25(2):469–482, 1970.

[31] R.M. Levich and L.R. Thomas. The merits of active currency risk management: evidence from international bond portfolios. *Financial Analysts Journal*, 49(5):63–70, 1993.

[32] R.A. Levy. Relative strength as a criterion for investment selection. *Journal of Finance*, 22(4):595–610, 1967.

[33] L. Lin, L. Cao, J. Wang, and C. Zhang. The applications of genetic algorithms in stock market data mining optimization. In *Proceedings of Fifth International Conference on Data Mining, Text Mining and their Business Applications*, pages 273–280, 2004.

[34] A.W. Lo and A.C. MacKinlay. Stock market prices do not follow random walks: Evidence from a simple specification test. *Review of financial studies*, 1(1):41–66, 1988.

[35] A.W. Lo and A.C. MacKinlay. *A non-random walk down Wall Street*. Princeton Univ Press, 2001.

[36] Sean Luke. *Essentials of Metaheuristics*. 2009. available at http://cs.gmu.edu/∼sean/book/metaheuristics/.

[37] B.G. Malkiel. The efficient market hypothesis and its critics. *The Journal of Economic Perspectives*, 17(1):59–82, 2003.

[38] R.N. Mantegna and H.E. Stanley. *An introduction to econophysics: correlations and complexity in finance*. Cambridge Univ Pr, 2000.

[39] T.D. Matteo, T. Aste, and M.M. Dacorogna. Long-term memories of developed and emerging markets: Using the scaling analysis to characterize their stage of development. *Journal of Banking & Finance*, 29(4):827–851, 2005.

[40] M. Mitchell. *An Introduction to Genetic Algorithms*. Bradford Books, 1996.

[41] M.C. Münnix, R. Schäfer, and O. Grothe. Estimating correlation and covariance matrices by weighting of market similarity. *Arxiv preprint arXiv:1006.5847*, 2010.

[42] C. Neely, P. Weller, and R. Dittmar. Is technical analysis in the foreign exchange market profitable? A genetic programming approach. *Journal of Financial and Quantitative Analysis*, 32(04):405–426, 2009.

[43] C.J. Neely. Technical analysis in the foreign exchange market: a layman's guide. *Review*, (Sep):23–38, 1997.

[44] OANDA.com. Historical foreign exchange rates, August 2010.

[45] T. Oberlechner. Importance of technical and fundamental analysis in the european foreign exchange market. *International Journal of Finance & Economics*, 6(1):81–93, 2001.

[46] G. Oh, S. Kim, and C. Eom. Market efficiency in foreign exchange markets. *Physica A: Statistical Mechanics and its Applications*, 382(1):209–212, 2007.

[47] J. Okunev and D. White. Do momentum-based strategies still work in foreign currency markets. *Journal of Financial and Quantitative Analysis*, 38(2):425–447, 2003.

[48] OnlineTradingConcepts.com. Technical analysis, 2008.

[49] P. Ormerod and C. Mounfield. Localised structures in the temporal evolution of asset prices. In *New Approaches to Financial Economics. Santa Fe Conference*, 2000.

[50] C.H.O. Park and S.H. Irwin. The profitability of technical analysis: A review. *Urbana*, 51:61801, 2004.

[51] S. Pincus and R.E. Kalman. Irregularity, volatility, risk, and financial market time series. *Proceedings of the National Academy of Sciences of the United States of America*, 101(38):13709, 2004.

[52] S.M. Pincus. Approximate entropy as a measure of system complexity. *Proceedings of the National Academy of Sciences of the United States of America*, 88(6):2297, 1991.

[53] M.J. Pring. *Introduction to technical analysis*. McGraw-Hill Companies, 1997.

[54] K. Pukthuanthong, R.M. Levich, and L.R. Thomas. Do Foreign Exchange Markets Still Trend? *Journal of Portfolio Management*, 2006.

[55] J.S. Richman and J.R. Moorman. Physiological time-series analysis using approximate entropy and sample entropy. *American Journal of Physiology- Heart and Circulatory Physiology*, 278(6):H2039, 2000.

[56] SW Roberts. Control Chart Tests Based on Geometric Moving Averages. *Technometrics*, pages 239–250, 1959.

[57] S.J. Russell and P. Norvig. *Artificial intelligence: a modern approach*. Prentice hall, 2009.

[58] P.A. Samuelson. Proof that properly anticipated prices fluctuate randomly. *Management Review*, 6(2), 1965.

[59] S. Saria, D. Koller, and A. Penn. Discovering shared and individual latent structure in multiple time series. *Arxiv preprint arXiv:1008.2028*, 2010.

[60] B. Scholkopf and A.J. Smola. *Learning with kernels*. Citeseer, 2002.

[61] M.R.E. Shazly and H.E.E. Shazly. Forecasting currency prices using a genetically evolved neural network architecture. *International review of financial analysis*, 8(1):67–82, 1999.

[62] K. Sugisaki and H. Ohmori. Realtime estmation of the degree of Market Efficiency using Variable Weighted Sample Entropy. In *SICE Annual Conference*, 2008.

[63] R.J. Sweeney. Beating the foreign exchange market. *Journal of Finance*, 41(1):163–182, 1986.

[64] E. Turban and R.R. Trippi. *Neural networks in finance and investment: using artificial intelligence to improve real-world performance*. McGraw-Hill, Inc. New York, NY, USA, 1995.

[65] J.C. Van Horne and G.G.C. Parker. The random walk theory: an empirical test. *Financial Analysts Journal*, 23:87–92, 1967.

[66] J.C. Van Horne and G.G.C. Parker. Technical trading rules: A comment. *Financial Analysts Journal*, 24(4):128–32, 1968.

[67] A.A. Weiss. ARMA models with ARCH errors. *Journal of Time Series Analysis*, 5(2):129–143, 1984.

[68] I.H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann Pub, 2005.