DOCTOR OF PHILOSOPHY

FORECASTING FOREIGN EXCHANGE RATES AND VOLATILITY WITH ARTIFICIAL NEURAL NETWORKS

Wang, Guan

Award date:
2021

Awarding institution:
Bangor University

GUAN WANG

FORECASTING FOREIGN EXCHANGE RATES AND VOLATILITY WITH ARTIFICIAL NEURAL NETWORKS

March 2021

Bangor University

Declaration

Yr wyf drwy hyn yn datgan mai canlyniad fy ymchwil fy hun yw'r thesis hwn, ac eithrio lle nodir yn wahanol. Caiff ffynonellau eraill eu cydnabod gan droednodiadau yn rhoi cyfeiriadau eglur. Nid yw sylwedd y gwaith hwn wedi cael ei dderbyn o'r blaen ar gyfer unrhyw radd, ac nid yw'n cael ei gyflwyno ar yr un pryd mewn ymgeisiaeth am unrhyw radd oni bai ei fod, fel y cytunwyd gan y Brifysgol, am gymwysterau deuol cymeradwy.

I hereby declare that this thesis is the results of my own investigations, except where otherwise stated. All other sources are acknowledged by bibliographic references. This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree unless, as agreed by the University, for approved dual awards.

Abstract

The foreign exchange (FX) market is long established as the largest and most important global financial market. While a large number of research papers focus on forecasting in the FX market, there are still gaps in the literature. First, very few papers focus on improving the parameter estimation process in the forecasting context. Second, artificial neural networks (ANN) with large sizes have not been applied to FX forecasting with the recently-fast-developed GPU techniques. Third, forecasting for trading purposes in the FX market has been limited to either building forecasting models or analysing technical indicators. A combination of ANN forecasting models with technical indicators is rare in the existing literature. The use of more-accurate parameter estimation algorithms and GPU techniques also makes the thesis unique in the methodological sense.

The thesis uses three types of ANN models, namely GARCH-ANN, large Multilayer Perceptron (MLPNN) and Long Short Term Memory (LSTM), to forecast volatility, the direction of price movements and price patterns in the FX market. Research is conducted at three data frequencies, namely monthly, daily and hourly as the analysis goes from the macro-perspective to the micro-perspective.

In the first empirical chapter, a Recursive Simulation Algorithm (RSGA) is proposed for estimating the parameters of a volatility forecasting model using GARCH-ANN. The proposed algorithm significantly improves the stability and accuracy of the estimation process by dealing with the local-optimum and convergence problems. The second empirical chapter utilises a large MLPNN model with GPU implementation to forecast the price direction of different FX pairs, with over 40 macro-economic indicators as input variables. Highly profitable out-of-sample results are observed for some of the currency pairs, which challenges the semi-strong form of the Efficient Market Hypothesis (EMH). Significant efficiency improvement is achieved with the GPU implementation. The third empirical chapter proposes the use of the Relative Strength Indicator (RSI) as a measure

of the extent of trend-following and mean-reversion patterns of FX rates. A LSTM model is utilised to forecast price movement patterns (measured by RSI). The trading strategy based on forecasting results of price movement patterns generates more stable profits than the benchmark Moving Average (MA) or RSI implemented on their own. However, the overall low profitability over time for the four currency pairs fails to challenge the weak-form EMH.

Overall, with the novel methodologies and technologies implemented within different models, this thesis finds evidence on some extent of inefficiency of the FX market at lower trading frequency (e.g. monthly) and less inefficiency of the FX market at higher trading frequency (e.g. hourly). One possible explanation is that at higher frequencies, the large number of daily (or higher frequency) traders and high-frequency trading algorithms reduce both the number of mis-pricing opportunities and the length of time that any mis-pricing opportunity may last.

Acknowledgements

I would like to thank my supervisors, Professor Owain ap Gwilym and Dr Chrysovalantis Vasilakis, for their continuous support throughout the period of my PhD. They have been constantly giving me invaluable guidance, comments, and encouragement, without which this thesis would never have been completed in any reasonable time.

I am also so blessed to have my family with all their encouragement, support and love.

Contents

List of Figures

List of Tables

1 Introduction

The foreign exchange (FX) market has expanded dramatically in its size and global influence since the collapse of the Bretton Woods System[1]. According to the most recent Bank for International Settlements (BIS) survey, the daily average turnover in the FX market has increased from less than 2 trillion USD in 2001 to 6.6 trillion USD in 2019. The increasing extent of fluctuation of prices since major countries started to adopt free-floating FX rates has also marked significantly more opportunities for profit making and an increasing necessity for risk management.

Historically, the extent of fluctuation (termed as "volatility") of FX rates becomes greater during periods of significant economic, political or social changes. As an example, on 15 January 2015, the Swiss National Bank abandoned its defence against the appreciation of the Swiss Franc (CHF) against the Euro (EUR) by dropping the 1.20 floor level of EUR/CHF. Within a day of the announcement, EUR/CHF dropped from 1.20 to 0.85 (roughly 41%). A more recent example is the US dollar (USD) depreciation during the coronavirus pandemic since early 2020. As a result of fast increasing coronavirus cases in the US (which leads to lock-down policies and therefore significant deterioration in economic performances), the Dow Jones FXCM Dollar Index[2] has dropped by over 7% since March (until mid-August) 2020. A significant economic, political or social change not only affects the concerned currency, but also other currencies as well. Dao et al. (2019) observe a significant appreciation in safe heaven currencies such as Japanese Yen (JPY) and CHF, during the UK EU-membership referendum period in June 2016. An increase in the number of trades and trading volume is also observed for these currencies.

From the trading perspective, as the volatility of FX rates increases, more traders are

[1]For a brief description of the earlier evolution of the FX market, refer to Subsection 2.1.1

[2]The Dow Jones FXCM Dollar Index measures the value of the US dollar against a basket of four currencies: the Euro, the British pound, the Japanese yen and the Australian dollar.

attracted to participate in the FX market because of the increased potential for making more profits. This, however, does not mean increased profitability for every trader in the market. In fact, the Efficient Market Hypothesis (EMH) argues that all information is reflected in the current price of FX rates. An implication of EMH is that one cannot make consistent abnormal profits in the market.[3] In this thesis, two of the three forms of EMH (weak form and semi-strong form) will be tested based on empirical evidence in the FX market.

Just as any other financial market, larger volatility not only means more potential for profitable trading but also more potential risks. Therefore, speculators are not the only group of participants who care about volatility. Households/businesses whose investments/assets are closely related to FX rates (e.g. international investors, trading companies) must also closely monitor FX volatility. The activity of FX trading in order to reduce FX rates risks is called FX hedging. Speculation and hedging are not the only areas where volatility forecasting is significant.

Another case where volatility forecasting plays a crucial role is option pricing. Being able to accurately forecast volatility will increase profitability in option trading. Angelidis and Degiannakis (2008), Bandi et al. (2008), Enke and Amornwattana (2008), and Yang et al. (2019) show that as the performance of their volatility forecasting model improves, significant abnormal profits can be generated in option trading. Therefore, the significant abnormal profits made in the option market as a result of successful volatility forecasting challenge the (weak form) EMH, based on the empirical evidence from these papers.

For the FX market, volatility forecasting[4] has drawn researchers' attention even before FX trading became popular from the late 1980s onwards. Pioneering research papers in chronological order include Engle (1982), Bollerslev (1986), and Heston (1993). More

[3]For more details of the EMH and its three forms, refer to Subsections 2.1.2 and 2.1.4.

[4]For more details of volatility forecasting models relevant to this thesis, refer to Subsection 2.2.1.

recent developments on these pioneering papers are Klein and Walther (2016), Diebold et al. (2017), and Ma and Ji (2019).

The main focus of all of the above papers is model development and financial interpretation of the empirical results. None of them discusses in detail about the parameter estimation process of the models they build.

Models from the above papers are dynamic models which contain linear/non-linear combinations of lagged historical values plus a time-dependent error term. The error term is assumed to follow a random distribution, i.e. normal-distribution, t-distribution. To estimate parameter values, estimation algorithms are used.

Most of the above papers (and especially in the early years - 1980s and 1990s) use gradient-based algorithms (e.g. Broyden (1970), Fletcher (1970), Goldfarb (1970) and Shanno (1970), abbreviated as the BFGS algorithm) to estimate parameters[5]. For these algorithms, at each step the parameter values are varied by a small amount in a direction that makes the cost function (such as negative likelihood) decrease fastest. However, as the algorithm constantly involves taking derivatives of the likelihood function, the indifferentiability problem might occur, especially for models with more complex likelihood functions.[6]

The Genetic Algorithm (GA) was originally developed in the field of biological science. It is a type of algorithm that does not involve taking derivatives of the likelihood functions. It mimics the gene selection process in which each generation aims at preserving only the "best" genes (in terms of, for example, healthiness, strength, ability to adapt to the environment) for them to be passed onto the next generation. In parameter estimation terms, parameter values are selected and varied such that the "best" (in terms of generating

[5]Gradient-based algorithms are still one of the most widely used estimation algorithms even nowadays.

[6]For more details of parameter estimation algorithms, refer to Subsection 2.2.4.

the lowest cost function) parameter values are more likely to be passed to the next iteration. Syarif et al. (2016), Zeng et al. (2017), and Zhang et al. (2017) discuss the use of GA in the parameter estimation process.

Similar to a gradient-based algorithm, a GA estimates parameter values through a large number of iterations until the estimated parameter values reach a stable level (this is termed "parameter convergence"). Despite the large number of iterations, both GA and gradient-based algorithms face the possibility of trapping into a local minimum (of the cost function), i.e. the estimated values reaching convergence makes further iterations useless because among all of their neighbours the estimated values already produce the smallest cost function. This is called a local-optimum problem.[7]

In the first empirical chapter (Chapter 3), the main research question is how to deal with the local-optimum problem for estimating models with complex likelihoods. The complex likelihoods are linked with volatility forecasting models built for FX volatility forecasting. The concept of repetitive computation is used to produce the Recursive Simulation Genetic Algorithm (RSGA). The RSGA recursively utilises previously obtained well-performing parameter values and repeats the optimisation process a large number of times. With the simulation algorithm, the variation of forecasting performance is reduced hence the performance stability is increased. The recursive algorithm increases the efficiency of simulation by discarding poorly performing candidate values. Six currency pairs are used for volatility forecasting, including less volatile pairs such as USD/JPY, EUR/USD and more volatile pairs such as USD/RUB, USD/ZAR. With this methodology, although it is not guaranteed that a global optimum is reached, significant improvements

[7]Throughout this thesis, the terms "local-maximum" and "local-minimum" are used on different occasions, depending on whether the target function needs to be maximised (i.e. for the likelihood function) or minimised (i.e. for the cost function). The term "local-optimum" is used to refer to the general case where the target function is not specifically defined.

are made in terms of accuracy for the forecasting of volatility of less volatile pairs as well as more volatile pairs.

The fact that the relatively more volatile pairs (e.g. USD/RUB and USD/ZAR) generate less than expected errors suggests that these two emerging market pairs are easier to forecast than the other four developed market pairs. In terms of market efficiency, the evidence in this chapter verifies the emerging markets as less (weak form) efficient than the developed markets. This is in line with the nature of the two types of markets.

Volatility forecasting is conducted and researched mainly by market participants who place more focus on financial derivatives pricing or risk management. For investors (speculators) whose target is to maximise the return/risk ratio, volatility forecasting is also crucial. Apart from volatility forecasting, return forecasting is a key task for speculators, in the FX market.

For speculators, return forecasting is central to all tasks as the success or failure of return forecasting ultimately determines the level of profitability. The significance of return forecasting has increased since the start of internet trading in the 1990s. As more and more institutions and individuals are able to trade FX rates online, the FX market turnover increases dramatically. The fast development of internet speed and data storage technologies has also made possible the availability of FX data to the general public. The 24-hour trading property of the FX market and development in artificial intelligence trading bots push forward the necessity of provision of various data frequencies of FX rates, from monthly, daily (for long-term researchers and investors, or governments) to hourly, minute, second, and even tick data (for short-term researchers and high frequency traders).

The second empirical chapter (Chapter 4) utilises macro-economic indicators to forecast

monthly FX rates[8], with an Artificial Neural Network (ANN) model[9]. The research questions of this chapter are (i) whether an increase in the number of neurons in the input and hidden layers will improve forecasting performance, and (ii) by how much computation efficiency will be improved with the help of GPU as the number of neurons increases. The modelling and forecasting processes are conducted on four FX pairs with the objective of maximising their out-of-sample annualised returns.

A typical ANN model mimics the structure of a human brain. It has several layers, each containing multiple individual cells to hold values called neurons. The advantage of an ANN model is that by varying the number of neurons per layer and the number of layers, an ANN theoretically is able to depict data patterns at any level of complexity, see Zhang (2003). However, in practice, it is not possible to increase the size of an ANN indefinitely due to the limits of computation power.

ANN models have been applied to forecast FX rates for the past few decades. Some of the selected papers include Kamruzzaman and Sarker (2004), Choudhry et al. (2012), Erdogan and Goksu (2014), Galeshchuk and Mukherjee (2017), and Liu et al. (2017). For all of these papers, the number of neurons is usually less than 10 and the number of layers is usually 3-5 (one input layer, one output layer and one to three hidden layers). In the early days, ANN models with this level of sizes typically take hours even days to train. However, with the development of improved Graphics Processing Unit (GPU) in the last five years, training ANN with GPU (instead of the traditional use of Central Processing Unit - CPU) has dramatically improved efficiency and therefore enhanced the computer's ability to train much larger ANN models within a shorter time. In addition, fast ANN libraries have been developed to boost the training process even more by improving

[8]The time frame for FX rates is chosen such that it is consistent with the highest data frequency of most macro-economic indicators.

[9]For more technical details on the ANN models, refer to Subsection 2.2.2.

the efficiency of codes. Keras is an open-source library created by Google for training large ANN models. It has both CPU and GPU implementations and supports multiple programming languages (such as R and Python).

Chapter 4 utilises Keras-GPU to training large ANN models[10] (with number of parameters over 10 million). The concept of repetitive computation from Chapter 3 is re-applied in this chapter to improve the performance of the parameter estimation process. With historical FX rates and macro-economic indicators as input variables, the ANN aims at forecasting the direction of FX rates movement in the next forecasting period. With a simple buy-low-sell-high trading rule, the large ANN models achieve significant profits for all of the four currency pairs (GBP/USD, EUR/USD, USD/JPY and USD/CHF). These results provide some evidence to challenge the semi-strong form of the Efficient Market Hypothesis (EMH) in the above markets for the underlying research period[11].

One special form of return forecasting is price movement pattern forecasting, which forecasts the patterns formed by a series of consecutive days of returns. Chapter 5 discusses the topic of pattern forecasting. But before moving onto the details of pattern forecasting, one question needs to be answered: Since being able to accurately forecast the return of a future period (a month, a day, or even a minute) can already make the forecaster significant profits, why bother forecasting patterns?

The reason is because forecasting exact future FX rates (or returns) at higher frequencies (hourly, minute or higher) is much more challenging than at lower frequencies (annually, quarterly or monthly). The challenge arises from two aspects: first, higher frequency means more data points and more combinations of possible patterns therefore the model needs to be able to depict more patterns to give an accurate forecast of a single future

[10]The ANN model adopted in this chapter refers to the Multilayer Perceptron Neural Network (MLPNN). Refer to Subsection 2.2.2 for more details of this kind of ANN.

[11]Refer to Subsections 2.1.2 and 2.1.4 for more details of EMH.

value. Second, for higher frequency forecasting, there is no reference to many indicators such as macroeconomic indicators which are typically published at a monthly, quarterly or an even lower frequency.

One real-world example is that for most FX traders (many of whom are daily or hourly traders), very few of them actually base their trading decisions on a forecast value (or direction of price movement) of FX rates in the future because it would be too risky, see Fung and Hsieh (1997) and Edwards (2014) for a discussion of different trading styles. Instead they usually form a brief idea on the overall price pattern and make decisions based on this idea. For example, if the market is currently trend following then it is more likely to continue the trend than fluctuating and if the market is moving around (mean-reverting) then it would bring higher odds to assume that the market is not following trends.[12]

As is discussed above, pattern forecasting plays a crucial role in FX trading. However, there is no consensus on how to define a particular pattern. One way to describe a pattern (as is shown in the example in the previous paragraph) is by classifying price patterns into the trend-following pattern and the mean-reversion pattern, see Serban (2010), Wu (2011), and Chaves and Viswanathan (2016).

The research questions of Chapter 5 are (i) How does a forecasting-trading algorithm's (such as the proposed LSTM-RSI) performance differ from a well-established, widely used trading strategy algorithm (such as an MA or RSI)? (ii) Are trend-following and mean-reversion patterns (measured by a pre-defined metric) related to different forecasting horizons? (iii) How much variation is observed in price movement patterns for different currency pairs? (iv) What is the implication of the market status being trend-following or mean-reverting for forecastability/profitability of a given period?

[12]One key nature of most financial markets (including the FX market) is that almost nothing happens at certain and everything is about probability.

Chapter 5 uses a Long Short Term Memory (LSTM) model[13] to forecast price movement patterns measured by the Relative Strength Indicator (RSI). To the author's best knowledge, this thesis is the first application of RSI to measure the extent of the trend-following pattern and the mean-reversion pattern. The LSTM-RSI forecasting-trading algorithm is tested against two most widely used technical trading rules - the Moving Average (MA) trading rule and the RSI trading rule[14]. The proposed LSTM-RSI generates relatively more stable trading performance than both MA and RSI. However, in terms of absolute profitability, none of the three trading rules is able to generate consistent profits for all currency pairs, which shows evidence in favour of the weak-form EMH.

Table 1.1 shows a summary and inter-relations of the three empirical chapters.

[13]The LSTM model is trained with Keras-GPU as is used in Chapter 4. The concept of repetitive computation from Chapter 3 is applied again for the parameter estimation process in this chapter.

[14]The RSI trading rule uses the same RSI indicator but the purpose of using this indicator and the logic of trading is completely different from the LSTM-RSI trading rule.

	Chapter 3	Chapter 4	Chapter 5
Main focus	Parameter estimation & volatility forecasting	Fundamental analysis & return forecasting	Technical analysis & pattern forecasting
Key methodological literature	Donaldson and Kamstra (1997) Monfared and Enke (2014) Junghans and Darde (2015)	Zhang and Hu (1998) Guresen et al. (2011) Bai and Koong (2018) Engel et al. (2019)	Taylor et al. (2001) Raza et al. (2014) Sang and Pierro (2019) Cohen (2020)
EMH context	Draw some implications for the weak form EMH from volatility forecasting	Test the semi-strong form EMH with fundamental data	Test the weak form EMH with historical FX rates data
Key EMH literature	Fama (1970), Jensen (1978), Fama (1991), and Timmermann and Granger (2004)		
Model	GARCH-ANN	Large MLPNN	LSTM-RSI
Model inputs	FX returns	FX rates and macroeconomic indicators	FX price patterns
Model outputs	FX volatility	Direction of FX rates movement	FX price patterns
Estimation tool	RSGA	Keras-GPU	Keras-GPU
Data source	Datastream	Datastream	MetaQuotes history centre
Data frequency	Daily	Monthly	Hourly
Data period	01/01/08 - 31/12/17	01/01/99 - 01/12/18	04/01/99 - 06/04/20

Table 1.1: A summary table of the three empirical chapters.

The three empirical chapters focus on parameter estimation, large-MLPNN modelling and LSTM-technical trading. Three key topics in the FX market are studied. These topics are volatility forecasting, return forecasting and price pattern forecasting. Implications of the empirical results on different forms EMH are also discussed.[15] Chapter 3 proposes an estimation algorithm that significantly improves the estimation accuracy of a GA. A comparison is made between the (weak form) efficiency of emerging markets and the (weak form) efficiency of developed markets. Chapter 4 extends the number of hidden neurons from 5-10 (a typical size used by previous researchers) to 2048. With the use of Keras-GPU to speed up the process of training a large MLPNN model (using the concept of repetitive training from Chapter 3), highly satisfactory out-of-sample trading performance is achieved with the model. The significant abnormal profits obtained in this chapter challenge the semi-strong EMH. Chapter 5 proposes the use of RSI as a measure of the extent of the market status being trend-following or mean-reversion. A LSTM model is trained to forecast price patterns (measured by RSI) and the LSTM-RSI trading rule generates overall more stable trading performance than traditional trading strategies. However, the proposed algorithm fails to generate consistent abnormal profits across all currency pairs. This empirical evidence supports the (weak form) EMH.

The remainder of the thesis is organised as follows. Chapter 2 discusses the general background to help the reader understand more thoroughly about the FX market, the EMH and technical methodologies used later in the thesis. Chapters 3-5 are the main empirical chapters. Chapter 6 presents the thesis conclusions.

[15]For a more thorough discussion of the three forms of EMH and different approaches to test EMH, refer to Subsection 2.1.2.

2 Conceptual and methodological background for the thesis

In this chapter, general concepts, theories and methodologies related to the three empirical chapters are introduced. The aim is to provide basic background knowledge upon which the empirical chapters are based.

The sections are organised such that the conceptual background section provides background on the foreign exchange (FX) market, the Efficient Market Hypothesis (EMH) and return/volatility forecasting in the FX market. The methodological background section discusses models and methodologies that are closely related or directly used in the empirical chapters. The conclusion section summarises the chapter and bridges the gap between the background and the three empirical chapters.

2.1 Conceptual background

2.1.1 The foreign exchange market

In the early 19th century, countries began to adopt the gold standard. Countries could convert foreign currencies they receive in trading into gold. This system started to break down during World War I as European countries started to print more money to pay for the war. In 1944, the Bretton Woods Conference Meeting was held and the Bretton Woods system was established. For countries other than the US, the currency of one currency was fixed against the currency of the other country. Countries could adjust their foreign exchange (FX) rate against the US dollar, which was pegged to gold. Under this system, the exchange rates usually experienced much smaller variations in a given time (compared with nowadays) and were almost completely determined by governments.

In 1971, the Bretton Woods system was abandoned because there was not enough gold reserve to back the amount of US dollars in circulation. From then on, a free-floating FX

system started to form. In 1985, G-5 (US, UK, France, Japan and West Germany) held a secret meeting at the Plaza Hotel in New York. News of the meeting was leaked, forcing the governments to make an announcement to encourage the appreciation of non-US dollar currencies. Since then, traders started to realise that despite relatively high level of government interventions during certain periods, FX rates might still have significant fluctuations in other periods, i.e. after the government already adjusted the FX rates. In fact the fluctuations could be so large that more and more traders (also called speculators) were attracted to trade FX because more fluctuations means more potential profitability.

Individual currencies are referred to as three-letter codes agreed through the International Organisation for Standardisation (ISO).[16] Table 2.1 displays the three-letter codes of the currencies to be used in this thesis. Since currencies are always traded in pairs, they are quoted as the relative price of one currency to another called exchange rates, i.e. the GBP/USD rate represents how much US dollar is needed to buy one British pound and the USD/JPY rate shows how much Japanese Yen one US dollar can buy.

Three-letter code	Currency
USD	US dollar
GBP	British pound
EUR	Euro
JPY	Japanese yen
CHF	Swiss franc
RUB	Russian ruble
ZAR	South African rand

Table 2.1: Three-letter representation of the currencies studied in this thesis.

[16] A full list of three letter currency letters agreed by ISO can be found on `www.iso.org/iso-4217-currency-codes.html`.

USD, GBP, EUR, JPY and CHF are called major currencies as they are traded most heavily across the globe. RUB and ZAR are much less frequently traded currencies (named minors) than the major currencies and usually exhibit larger volatility throughout time.[17] The reason for selecting RUB and ZAR among other volatile currencies is that USD/RUB and USD/ZAR are two of the minor pairs that have longer data availability, i.e. data of some of the other minor currency pairs is only available from 2010s.

The FX market facilitates trading for governments, companies and individuals to exchange currencies for different purposes. For example, governments may exchange currencies to buy or sell a foreign government's bond or to control its currency value against another currency within a target range. Companies which imports or exports goods and services need to exchange currencies between their own country and the country they transact with. Individuals who travel or study abroad also have to exchange their own currency into the currency used in the country they are going to visit. For more descriptive details on the FX market refer to Chapters 4 and 5 of Weithers (2006).

Apart from the above purposes, institutions and individuals may also use the FX market for hedging and speculative purposes. Hedging in the FX market refers to exchanging currencies spot or currency derivatives in order to reduce the risk of future FX rates moving in the unfavourable direction.[18] Speculation is an activity conducted by market participants in order to make profits based on correct forecasts of future FX rates movements. Research papers have shown that speculation increases the volatility of FX rates, see Aliber (1964), and Driskill and McCafferty (1980). Rothig et al. (2007) find that depending on whether hedging or speculative strategies are adopted, magnitudes of recessions and booms are decreased or amplified.

One structural difference between the FX market and the stock market is that the stock

[17] See Figure 3.3 and Table 3.2 in Chapter 3 for a visualisation and summary.

[18] The term "FX rates" will be used in this thesis specifically for the spot rate.

market is centralised, meaning there is only one price for any particular stock at any given time, determined by a central agency, normally the exchange that the stock is listed on, i.e. New York Stock Exchange, London Stock Exchange. The FX market, however, is decentralised as it is controlled by not only one but multiple agencies such as major banks, governments, hedge funds, FX brokers and so on. Therefore for any currency pair at any given time every agency may have different quote prices. This is one of the main reasons why an overview-type-of-study of the relatively "chaotic" FX market is needed.

The Bank of International Settlements (BIS) Triennial Central Bank Survey, conducted every three years since 1986, provides information on the global FX market and over-the-counter (OTC) derivatives market. According to the BIS survey in 2019, average daily turnover of the FX market increased by nearly 30% from 5.1 trillion US dollars in 2016 to 6.6 trillion US dollars in 2019. The US dollar remains the dominant currency in the market, being on one side of 88% of all trades. Currencies of emerging market economies reach 25% of global turnover.[19]

The above paragraphs demonstrate the significance of the FX market in terms of its functions and development. To improve performance especially for speculative purposes, forecasting the FX rates has been focused on and researched for decades. However, the success of forecasting techniques highly depends on the market being (at least partially) inefficient. The next subsection discusses the Efficient Market Hypothesis (EMH) and research papers in this area.

[19]More details on results from the BIS survey are provided in 3.1.1 and 4.1.2.

2.1.2 Efficient market hypothesis

The concept of "efficient market" was originally developed in the stock market. As one of the pioneering papers, Fama (1970) describes an efficient market as "a market in which prices always fully reflect all available information". He considers three subsets of market efficiency, the weak form efficiency, the semi-strong form efficiency and the strong form efficiency. More specifically, a stock market is weak form efficient if the stock price at any time fully reflects all information from historical prices of the stock up until that given time. A stock market is semi-strong form efficient if the stock price at any given time fully reflects all publicly available information (e.g. announcements of earnings, stock splits, etc.) of the stock up until that given time. Finally, a stock market is strong form efficient if the stock price at any given time fully reflects all private information relating to the stock up until that given time. One noteworthy nature of the three forms of efficient market is the cumulative nature, i.e. if a market is semi-strong form efficient then it is also weak form efficient and if a market is strong form efficient then it is both semi-strong form efficient and weak form efficient. In terms of inefficiency, the order of cumulation is reversed, i.e. if a market is semi-strong form inefficient then it is also strong form inefficient and if a market is weak form inefficient then it is both semi-strong form inefficient and strong form inefficient.

Although the difference in the three subsets of efficiency is made clear, it is a challenging task to numerically test the term "fully reflect". As is mentioned in the previous paragraph, the EMH was originally proposed in the area of the stock market. However, as the key concept of EMH is whether different kinds of information is reflected in current price of the underlying security, the EMH is also applicable in other financial markets where the security price is affected by different sources of information, e.g. the FX market. While there is a rich vein of academic papers which discuss/test EMH for the stock market, the number of papers based on the FX market is relatively smaller.

One reason for the limited discussion is the relatively high efficiency of the FX market. As is discussed in Subsection 2.1.1, the large trading volume in the FX market and global trading activities around the clock have made the FX market arguably one of the most liquid and efficient markets in the world. More recently, the efficiency has been further enhanced with a larger number of trading algorithms implemented in the past two decades. These algorithms have significantly reduced both the number of mispricing opportunities and the time any mispricing opportunity may last. However, despite the higher extent of efficiency, does it mean the FX market is efficient in all senses, i.e. for different currency pairs and under different forms of tests? This will be the main question the three empirical chapters of the thesis aim to answer.

Although the EMH has been widely studied in academia, there is no consensus on how to define quantitatively (i.e. with a formulae), an efficient market. In other words, there is a diverse range of choices of methodologies adopted by researchers to test EHM. One approach is to apply statistical tests on historical FX rates, examples include Burt et al. (1977), Timmermann and Granger (2004), Karuppiah and Los (2005), Popovic and Durovic (2014), Makovsky (2014), Narayan et al. (2016), and Caporale and Plastun (2020).[20] In all of the above papers, evidence of different extents of inefficiency is observed for the FX (or stock) market.

However, statistical tests typically involve only one time series, i.e. historical FX rates, for the purpose of testing EMH. Therefore, all of the above papers essentially test the weak-form EMH because no other information than historical FX rates is used to test EMH. To test the semi-strong and strong-forms of EMH, an approach which involves multiple time series (e.g. historical FX prices and macroeconomic indicators) is needed.

According to Fama (1991), he considers the Jensen (1978) version of the EMH as an

[20]Caporale and Plastun (2020) also discuss the use of trading strategies for EMH testing, which is a second approach to test EMH.

"economically more sensible" version of the EMH. This version of the EMH states that "prices reflect information to the point where the marginal benefits of acting on information (the profits to be made) do not exceed the marginal costs".[21] From this point onwards, the term "EMH" will be used to refer to the Jensen (1978) version of the EMH unless otherwise specified.

Testing EMH with the performance of trading strategies is one of such approaches to test the other two forms of EMH because a trading strategy can be built by taking information from multiple time series. The theoretical basis of this methodology has been discussed in the previous paragraph. The practical intuition of this approach is discussed as follows. Under the EMH, an implication is made that no consistent above-normal profit (after considering transaction costs) can be achieved in the long term. Therefore, being able to build up persistently profitable trading strategies serves as a sufficient (but not necessary) condition to challenge the EMH, i.e. if a trading strategy generates significant profits over the long term then the EMH is challenged but if such a trading strategy does not generate significant profits, then the conclusion that EMH is verified cannot be drawn. This is because there could be a numerous number of other models that generate significant profits and one failed strategy cannot represent the whole world of strategies. This brings one characteristic of the abnormal-trading-profits approach to test EMH, namely it can be only used to challenge the EMH but not to verify the EMH, unless the number of strategies is large enough to fully represent the whole world of trading strategies.

Research papers that utilise the abnormal-trading-profits approach to test EMH in the FX market include Sweeney (1986), Katusiime et al. (2015) and Zarrabi et al. (2017) and Caporale and Plastun (2020). Among these papers, Sweeney (1986) looks into the DEM (Deutsche Mark)/USD rate from 1973 to 1980. Katusiime et al. (2015) test weak-form

[21]This version of the EMH by Jensen (1978) forms a basis for testing EMH with the abnormal-trading-profits approach, as is discussed in the following paragraph.

efficiency of the Ugandan foreign exchange market from 1994 to 2012. Zarrabi et al. (2017) study six currency pairs, namely GBP/USD, USD/CAD, USD/JPY, USD/NOK, USD/SEK and USD/CHF from 1994 to 2014. Caporale and Plastun (2020) also study six currency pairs, namely EUR/USD, USD/JPY, USD/CAD, AUD/USD and EUR/JPY, from 2008 to 2018. In all of these papers, some extent of weak form market inefficiency is observed.

One limitation of the above papers is that all of them study only one time frequency - the daily. There is no logic defect in using only one frequency. However, with multiple time frequencies, one is able to draw a more thorough conclusion with the comparison among different time frequencies. This thesis studies EMH in three time frequencies, hourly, daily and monthly.

In Subsection 2.1.3, two types of forecasting are discussed, namely return forecasting and volatility forecasting. In Subsection 2.1.4, two types of trading techniques (fundamental and technical) are discussed and the link between these two techniques and the three forms of EMH is illustrated.

2.1.3 Foreign exchange rates and volatility forecasting

Foreign exchange (FX) rates (returns) and volatility are two target time series that forecasting can be based upon. This subsection discusses the significance of forecasting FX rates (returns) and volatility and developments in forecasting these two time series.

Price or return forecasting is one of the key financial forecasting tasks because from a speculator's perspective forecasting future price movements with high accuracy delivers significant profits. The returns are typically calculated as percentage price changes or log-returns. Glaser et al. (2019) study the difference in expectations of return forecasting and price forecasting. They find that investors hold higher expectations of return forecasting than price forecasting. This higher expectation is caused by different attitudes of the subjects towards return forecasting and price forecasting. Most subjects consider price forecasting as more challenging than return forecasting.

However, based on the quantity of information to be forecast, it makes no difference to forecast return or forecast price, i.e. you can get the forecast return with the forecast price and vice versa. In this thesis, the reason for forecasting returns rather than prices from the modelling perspective is that using returns makes the models easier to generalise with respect to different value levels, i.e. for the GBP/USD pair the rates are usually between 1 and 2 while for the USD/JPY pair the rates are larger than 100.

When return forecasting is concerned, one of two different types of returns is typically focused on, log-return or simple return. Hudson and Gregoriou (2015) make a comparison between log- and simple returns. They show that there is not a one-to-one relationship between the log-return and the simple return. In this thesis, since FX pairs are studied, log-returns are more advantageous than percentage price changes because log-returns are symmetric with respect to one currency pair and its reverse pair. As an illustrating example, GBP/USD and USD/GBP are reverse pairs which should be considered essentially as the same pair. The factors (i.e. macro or micro-economic news) that affect the

price of one pair are exactly the same as the factors that affect the other pair. Therefore, the extent of price change (caused by the same factor) should also match for the reverse pairs.

Table 2.2 illustrates that for a reverse pair, using simple return generates different absolute changes while using log-returns generates the same absolute changes.

Currency pair	Month 1	Month 2	Monthly simple return	Monthly log-return
GBP/USD	1.25	1.28	2.04%	2.37%
USD/GBP	0.80	0.78	-2.34%	-2.37%

Table 2.2: A hypothetical example illustrating simple returns and log-returns.

In the FX market, due to the availability of high leverage[22] and 24-hour non-stop (except weekends) trading, there is opportunity to make huge profits (or losses) in a shorter time period. To be more specific, with the availability of high leverage, FX traders can place trades that valued more than their account balance and the 24-hour non-stop (except weekends) trading not only presents more trading opportunities within a given time but also makes trading algorithms overwhelmingly powerful because unlike humans they can trade 24 hours a day. This puts price return forecasting at a more significant place in the FX market than in, for example, the stock market. Tenti (1996), Diebold et al. (1999), Panda and Narasimhan (2007), Villanueva (2007), Coakley et al. (2016), and Rundo et al. (2019) demonstrate that significant profits (over buy-and-hold or random walk) can be made in the FX market.[23]

[22]Many FX brokers offer leverage levels from 10 up to 500 while for the stock market for example, stocks traded through the exchange usually cannot be leveraged.

[23]More methodological details on the forecasting models are presented in Section 2.2.

A subclass of return (price) forecasting is called pattern forecasting. It is considered a subclass because a pattern is formed by a series of consecutive days of returns. Essentially, the information contained in a pattern is always a proportion of the information contained in the return series from which the pattern is derived.

Volatility, a measure of price deviation over time, is a key metric of the variance of FX rates. It is a crucial metric because of the two following reasons.

First, it serves as a measure of risks. From a hedger's perspective, being able to forecast volatility of different FX rates with high accuracy helps identify where hedging is most needed. For example, if the volatility of a currency pair is forecast to be extremely high in the next few months, this implies a potentially urgent need for hedging. On the contrary, if the volatility of a currency pair is going to be so low that even the most unfavourable FX rates movement occurs, the loss is still lower than the cost of hedging then there is no reason for hedging.

Second, the volatility of different currency pairs in different time periods can be used as an indicator of whether the underlying pair should be traded in the next forecasting period, i.e. speculators may find a FX pair less attractive if its forecast volatility is too low and likewise they may also give up trading one FX pair because the forecast volatility is too high for their risk tolerance. Dunis and Miao (2005) find that traditional Moving Average Convergence Divergence (MACD) trading rules perform poorly in volatile periods compared with less-volatile periods. They implement a volatility filter[24] (i.e. no trade takes place in certain volatility conditions) on traditional Moving Average Convergence Divergence (MACD) trading rules and significantly improve trading performances in stock, bond and commodity markets.

[24]Dunis and Miao (2007) also explore this volatility filter trading rule and confirm its improved performance in the FX market.

In addition, one recent real-life example is that during the negotiation between UK and EU for a deal/no-deal outcome in late 2019, many investment funds anticipated the volatility of British pound to increase sharply and therefore decided to lower the maximum leverage allowed for any trades on GBP pairs and others even completely gave up trading GBP pairs during that time.

One advantage of studying returns (patterns) is that by implementing a trading strategy based on the forecast returns (patterns), it is possible to test the EMH via the abnormal-trading-profits approach discussed in Subsection 2.1.2. On the other hand, although it is technically possible to implement a trading strategy based on the forecast volatility alone, it does not make much practical sense to do so. For example, a volatility surge could arise from a significant price increase as well as a significant price decrease. Therefore, making buy or sell decisions solely based on the forecast volatility does not seem reasonable. Hence, the abnormal-trading-profits approach cannot be used with forecast volatilities, yet the forecasting performance can still act as an indication of how efficient the market is. For example, in a highly efficient market, it would be more difficult to forecast accurately future volatilities based on historical data alone, than in a less efficient market. Given that volatility forecasting only gives an indication of market efficiency to a restricted extent,[25] the task of testing EMH is mainly conducted in the return (or pattern) forecasting processes in Chapters 4 and 5.

In the next subsection, two types of trading techniques (fundamental and technical), which are adopted in Chapter 4 and Chapter 5 respectively, are discussed in the FX market context.

[25]The term "restricted extent" is used to contrast the return (pattern) forecasting case where the abnormal-trading-profits approach can be adopted to test EMH (the Jensen (1978) version).

2.1.4 Foreign exchange trading: fundamental, technical analysis, and three types of EMH-related information

To test EMH, the essential question that needs answering is whether different kinds of information is reflected in the price of a security (stocks, FX rates, etc.). As is discussed in Subsection 2.1.2, depending on the type of information concerned, the EMH has three subsets of forms, namely weak form, semi-strong form and strong form. This subsection discusses the three types of information, namely fundamental, technical and private information as well as how these types of information are utilised in two of the most widely applied trading techniques in practice - fundamental analysis and technical analysis. The move from information-based analysis to trading-based analysis helps facilitate the tests of different forms of EMH (the Jensen (1978) version EMH).

Fundamental information

Fundamental information refers to the information that is publicly available and also closely related to the price of an underlying security. Therefore, fundamental information is closely related to the semi-strong form EMH. For stocks, fundamental information includes company earnings results, stock split decisions, stock dividend decisions, etc. For FX rates, however, fundamental information exists more on a larger scale, i.e. in the macro-economic sense rather than micro-economic sense. Interest rate is one of such examples of fundamental information to FX rates.

According to the Interest Rate Parity (IRP) condition, under the non-arbitrage[26] assumption, the interest rate differential between two countries is equal to the differential between the forward FX rate and the spot FX rate. The IRP acts as a connection of interest rate and FX rate (forward and spot), i.e. forward FX rate is determined by interest

[26] An arbitrage opportunity refers to the situation where simultaneous purchase and sale of an asset are conducted to make profits as a result of the price difference.

rate differential and spot FX rate. Although this thesis does not focus on forward FX rate pricing or interest rate modelling, the IRP suggests that interest rate should also be a crucial factor affecting future FX spot rate, as the forward FX rate is a reasonable forecast of future FX spot rate.

The inflation rate (typical measures of inflation rate include Consumer Price Index - CPI and Retail Prices Index - RPI) is another factor closely related to FX rates. According to the Purchasing Power Parity (PPP) relationship, there is a link between prices in two countries and the exchange rate between the currencies of the two countries. Therefore, it is reasonable to assume that the inflation rate has some extent of forecasting power on FX rates. Qiu et al. (2011) forecast FX rates of eight currency pairs, based on historical deviations (dataset lasts from 1974 to 2007) of the nominal FX rates from the FX rates as suggested by PPP. They find significant out-of-sample correction patterns of the nominal FX rates towards the FX rates calculated by PPP. Ca'Zorrzi and Rubaszek (2020) forecast FX rates based on the mean-reversion patterns of FX rates as implied by PPP. They show that their forecast of FX rates outperforms the random walk. This paper not only supports the inflation rate as a significant factor in fundamental analysis, but also links with Chapter 5 via the forecasting methodology, i.e. making forecast based on price patterns.

Other examples of fundamental information to FX rates include (but not restricted to) government fiscal (i.e. government budget and spending) and monetary policy (i.e. controlled by interest rates), balance of trade (i.e. exports and imports), economic growth (i.e. GDP), etc.[27]

From the trading perspective, utilising fundamental information to help make trading

[27]However, since most of those indicators are published at a low frequency, i.e. monthly, quarterly or annually, it is practically infeasible to conduct fundamental analysis on a daily (or a higher frequency) basis.

decisions is referred to as fundamental analysis (or fundamental trading). Research papers that study the relationship between macro-economic indicators (fundamental information) and FX rates include MacDonald (1999), Galati and Ho (2003), Mariano et al. (2016) and Aka (2020). For these papers, statistical tests (e.g. based on regression models) are conducted on data to justify the relationship (between fundamental information and FX rates). Compared with the number of papers that discuss the relationship between fundamental information and FX rates, there is a smaller number of papers that test the EMH with the abnormal-trading-profits approach. Examples of these papers include Dunis and Williams (2002), Eng et al. (2008) and Yildirim et al. (2021). With the abnormal-trading-profits approach, trading strategies (using fundamental information) are implemented and a strategy that generates long-lasting abnormal returns serves as a challenge to the semi-strong form EMH. In all of these three papers, significant out-of-sample abnormal profits are observed and they therefore challenge the semi-strong form EMH (within the time period concerned).

Technical information

Technical information includes historical prices (or volumes) of a security or any metrics directly derived from historical prices (or volumes). Examples of technical information include returns, trading volumes, direction of price movement, values of technical indicators, e.g. Moving Average (MA), Relative Strength Indicator (RSI), etc. Unlike fundamental information for which stocks (company-specific news and statistics) and FX (macro-economic indicators) differ in terms of the sub-types of information, the sub-types of technical information are the same for stocks and FX, i.e. historical prices and derived values from historical prices. Therefore, the use of technical information as a forecasting tool is closely related to the weak form EMH across different asset classes.

The practice of trading based on technical information is referred to as technical analysis

or technical trading. Unlike fundamental analysis, technical analysis does not aim at forecasting the intrinsic value of the underlying security (e.g. stocks or FX rates) but forecasting future prices based on past price trends or patterns. For example, the MA indicator utilises moving averages calculated for different lengths of period, e.g. 10-day, 20-day. If the short MA (10-day) crosses above the long MA (20-day) then it is likely that an upward trend may start to form and if the short MA (10-day) crosses down the long MA (20-day) then a downward trend may start to form. MA (200-day) is another widely adopted indicator used for identifying long-term trends. Taylor (2014) provides a review of technical trading rules (MA and trading range breakout) applied to all members of the Dow Jones Industrial Average (DJIA) stock index over the period 1928–2012. He finds that abnormal returns are confined to particular time periods (from the mid-1960s to mid- 1980s), i.e. no consistent abnormal profits can be obtained.

Other papers that find evidence supporting the weak form EMH include Fyfe et al. (1999), Hoffmann and Shefrin (2014), and Urquhart et al. (2015). These papers build up technical trading rules and discover empirically that none of these trading rules generate consistent abnormal profits. Therefore, these papers provide evidence to support (not verify)[28] the weak form EMH.

There are also papers that provide empirical evidence to challenge the weak form EMH. In addition to the papers mentioned in Subsection 2.1.2,[29] where they use non-technical trading rules, Menkhoff et al. (2012) implement technical trading rules (momentum-based strategies) to trade 16 currency pairs from 1976 to 2010. They obtain significant out-of-sample abnormal profits therefore the weak form EMH is challenged based on their empirical evidence.

[28] As is discussed in Subsection 2.1.2, the failure of certain trading rules cannot be used as a means of verifying the weak form EMH.

[29] See Burt et al. (1977), Timmermann and Granger (2004), Karuppiah and Los (2005), Popovic and Durovic (2014), Makovsky (2014), Narayan et al. (2016), and Caporale and Plastun (2020).

Private information

Private information is a type of information that is available only to a small group of people. Essentially all sub-types of the fundamental information (regardless of whether it is in the stock market or the FX market) can be considered as private information before they are published, e.g. unpublished earnings results, unrevealed central bank decisions on interest rate cut, etc.

The strong form EMH is related to private information and it is considered as the strongest form because it can hardly be achieved in reality. For example, for someone who always has company earnings results beforehand (such as a company leader), it would be much easier for him to achieve significant abnormal profits by trading on the private information before the information is made public, than someone who trades the news after it is released. This is also why trading based on private information by specific people (i.e. who are given the right to know the private information in advance), also termed as insider trading, is legally forbidden in most countries. There are, instances where getting private information is legally permitted, e.g. hedge funds paying for private exit polls (as they did in the 2016 UK-EU membership referendum). With this type of information (which is unavailable to the general public), individuals or institutions are more likely to make more successful (in terms of profitability) investment decisions than most people who do not have access to the private information.

The strong form EMH is therefore considered least likely to hold compared with the other two forms. For this reason, the number of papers discussing the strong form EMH is much smaller than the number of papers discussing the other two forms, e.g. Finnerty (1976), Givoly and Palmon (1985), Rozeff and Zaman (1988), and Bashir et al. (2020) find that insiders achieve significantly higher abnormal profits than normal investors, which provides strong empirical evidence against the strong form EMH.

Due to the relatively strong consensus in academia on whether strong form EMH holds

(i.e. not holding) and the practical difficulty in obtaining private information that is actually used for insider trading,[30] the strong form EMH is not focused on in this thesis.

Fundamental and technical analysis in trading practices

Although fundamental and technical analysis differ significantly in terms of source of information and logic of trading, most trading practitioners adopt a combination of the two approaches for trading decision making.

Oberlechner (2001) presents results from a questionnaire and interview survey on fundamental and technical analysis by traders and financial journalists in the FX market from several cities across Europe. The results show that most traders use both fundamental and technical analysis and as time frame goes lower technical analysis are more dominant. Compared with traders, financial journalists generally place more emphasis on fundamental analysis. His results also show that there has been an increasing popularity of technical analysis since 1990. The survey conducted by CitiFX Pro (2010) finds that over 50 percent traders adopt a combination of fundamental and technical analysis.

A similar research is conducted by Lui and Mole (1998) with FX market participants in Hong Kong. They also find that most traders use both technical and fundamental analysis and the usage of fundamental analysis decreases as time frame decreases. In addition, they find that technical analysis is slightly more useful than fundamental analysis in forecasting trends but significantly more powerful in forecasting turning points.

To provide an overview of the three empirical chapters in the conceptual sense, Chapter 3 utilises daily FX data to forecast FX volatility. With forecast volatility, the EMH cannot be tested via the abnormal-trading-profits approach. However, the relative easiness of

[30] How likely is someone who actually performs insider trading willing to reveal it to someone doing the research?

forecasting among different pairs and the difference in training and test samples give some implication of (weak form) EMH in the comparative sense. Chapter 4 employs monthly data in order to test EMH from the macro-economic perspective. Both historical FX rates and macro-economic indicators are utilised therefore it will test the semi-strong form EMH. Chapter 5 studies FX rates at the hourly frequency. The target is to test EMH in high-frequency settings. Only historical FX rates are used as inputs of the models therefore the weak-form EMH is tested. The strong-form EMH which involves private information is not studied in this the thesis, due to the "seemingly" unnecessity in testing it (i.e. there is arguably no dispute in academia on whether the strong form EMH holds - it does not hold). Another reason for not testing the strong form EMH is because of the difficulty in collecting relevant data, as the term "private information" suggests. The following subsection discusses models that are commonly used to forecast the two types of target series (return and volatility).

In Section 2.2, time series, Artificial Neural Networks (ANN) and their hybrid models are discussed in the context of return/volatility forecasting.

2.2 Methodological background

2.2.1 Time series models (GARCH-type)

Time series models, as the name suggests, aims at modelling and forecasting time series data, such as financial datasets including FX rates. One of the most well-known and widely used time series for volatility forecasting is the Generalised Autoregressive Conditional Heteroskedasticity (GARCH) model proposed by Bollerslev (1986) defined as:

$$\sigma_t^2 = \alpha + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2 + \sum_{j=1}^{q} \gamma_j \epsilon_{t-j}^2, \tag{2.1}$$

where α, β_i and γ_j all need to be non-negative to guarantee that the conditional variance σ_t^2 is non-negative. The advantage of the GARCH model is that the conditional variance of the next period tends to be larger if its nearest past conditional variances are large and it tends to be smaller if its nearest past conditional variances are small. This forms a mimic of the financial prices, which typically exhibit the volatility clustering pattern, i.e. high-volatility periods are more likely to be followed by high-volatility periods and low-volatility periods are more likely to be followed by low-volatility periods.

Variations of the GARCH model include EGARCH, IGARCH, NAGARCH, GJR-GARCH and among others, all of which are called GARCH-type models.

Although all of the variations of the GARCH model may have good performance under certain circumstances, there is no consensus on which model may perform relatively well most of the time. However, research papers have demonstrated that the original GARCH model of the GARCH-type family, even in its simplest form, GARCH(1,1), has a satisfactory forecasting power in general cases.

McMillan and Speight (2004) point out that the failure of GARCH does not arise from the model itself but from the failure of specify an appropriate measure of the "true volatility". With the cumulative daily squared returns instead of daily squared returns

as the measure of "true volatility", the GARCH model, in a dataset of 17 FX rates, outperforms smoothing and moving average techniques which have been previously shown to produce superior results. The same authors (McMillan and Speight (2012)) later compare the performance of the GARCH with other GARCH-type models for forecasting volatility of higher frequency (intra-day) FX rates. They find that although the GARCH model can be beaten by others in forecasting volatility of daily FX rates, it produces the best result in forecasting volatility of intra-day FX rates, based on R^2.

Hansen and Lunde (2005) are the earlier explorers of the GARCH(1,1) model in comparison with other GARCH-type models. They compare 330 GARCH-type models for their performance in forecasting conditional variance with FX rates. They find no evidence that any of the more sophisticated models can significantly outperform GARCH(1,1) in their analysis on FX rates.

A later paper by Miah and Rahman (2016) examines the performance of the GARCH model with different specifications (i.e. different lags of order) on the stock market. Despite the findings that the stock market are non-normal, negatively skewed and with high excess kurtosis, the GARCH(1,1) model produces significantly improved results over other specifications in terms of AIC and BIC.

Although evidence of good performance in forecasting volatility by the GARCH model (together with its simplest form GARCH(1,1)) has been presented by a number of research papers, the comparisons are made mainly with other GARCH-type models. This means any improved performance by the GARCH model could be restricted within the GARCH-type family while the common limitations of the GARCH-type family are not revealed in those comparisons.

One limitation of the GARCH-type family is that once the specific model is determined, the architecture of the model (hence the type of data patterns that the model is best at

forecasting) cannot be changed. For example, once we decide to use the GARCH model (with linear increments in past conditional variances and residuals), the fact that the model might not perform well for datasets which exhibit significant non-linear patterns will remain the issue. There are, certainly, GARCH-type models that have non-linear components (such as EGARCH, with logarithmic increments). However, if the dataset for which volatility is to be forecast has movement patterns significantly different from the type of increments of the selected GARCH-type family, then performance may be poor, as a result of inflexibility of the GARCH-type family.

One solution to the inflexibility problem is by using an artificial neural network (ANN). The next subsection discusses this approach.

2.2.2 Artificial neural networks (ANN)

The term "neural network" origins from the area of information processing in biological systems. The concept is first proposed by McCulloch and Pitts (1943). One of the earliest-created and most fundamental ANN structures is the multilayer perceptron neural network (MLPNN). Many other ANN structures are modified versions based on MLPNN. Because of its wide popularity, strong forecasting ability and pioneering status, MLPNN is the main focus of this subsection.[31] As an illustrating example, a simple three-layered network is displayed in Figure 2.1. Each circular node represents an artificial neuron and an arrow represents a connection from the output of one neuron to the input of another.

Given an array of input $\mathbf{x} = (x_1, x_2, ..., x_D)$, named input units, compute the first layer $\mathbf{a} = (a_1, ..., a_M)$ by

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}, j = 1, ..., M \tag{2.2}$$

[31] In Chapter 5, time-focused ANN - namely recurrent neural networks (RNN) with its developed version long-short-term-memory (LSTN) will be discussed.

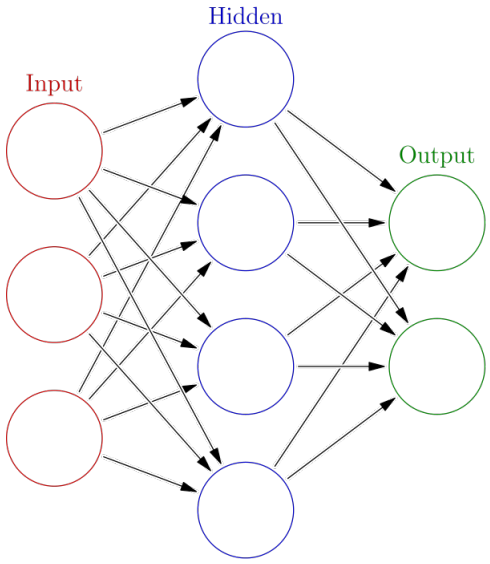


Figure 2.1: A three-layer neural network

$w_{ji}^{(1)}$ are referred to as weights of the first layer and $w_{j0}^{(1)}$ as biases of the first layer. Then each of the a_j is transformed using a differentiable, nonlinear function $h(\cdot)$, called activation function

$$z_j = h(a_j) \tag{2.3}$$

Then the output units (before activation) are given as linear combinations of $z_1, ..., z_M$.

$$a_k = \sum_{i=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)}, k = 1, ..., K \tag{2.4}$$

Finally each of the a_k is activated via an activation function $\sigma(\cdot)$ to give the final output $\mathbf{y} = (y_1, ..., y_K)$, where

$$y_k = \sigma(a_k). \tag{2.5}$$

Combine the previous steps together, a three-layered MLPNN has a structure in the following format.

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma \left(\sum_{j=1}^{M} w_{kj}^{(2)} h \left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \right) + w_{k0}^{(2)} \right) \tag{2.6}$$

Two commonly used activation functions are the hyperbolic tangent $\sigma(a_k) = \tanh(a_k)$and the logistic function $\sigma(a_k) = (1 + e^{-a_k})^{-1}$. Both functions are sigmoid, i.e. S-shaped bounded and differentiable functions.[32] Zhang (2015) shows that using a combination of different activation functions improves forecasting performance (in terms of Mean Squared Error - MSE) over using a single type of activation function.

There is a critique on ANN that only what enters and leaves the structure are clear while what has been done within the hidden layers seems a "black box", see Castelvecchi (2016). In fact, this is similar to how a brain works. For example, after years and years of training, a top professional footballer may develop some habitual behaviours (such as dribbling the ball with high skills). In the training process, his development in dribbling skills do not arise from knowing in detail why each small body movement works best but as a result of the brain memorising the overall body movements which have the highest rate of avoiding opponent tackling, after thousands of training practices.

In modelling with ANN structures, the number of neurons per hidden layer and the number of hidden layers determine the complexity of the model, i.e. as the number of neurons per hidden layer and the number of hidden layers increase, the model's ability to capture data patterns increases, for example, from linear to quadratic, from cubic to quartic, and goes up indefinitely to any level of pattern complexity.[33] However, as the numbers of neurons and layers increase, computation time increases dramatically, see Alvarez and Salzmann (2016). Computation time can be reduced by (1) combining an ANN with a GARCH-type model (refer to Chapter 3). This approach is able to restrict the number of neurons and layers without sacrificing too much forecasting power because of the inclusion of a GARCH-type model. (2) utilising high-efficiency training libraries and GPU computing to reduce computation time (refer to Chapter 4).

[32]The estimation of parameters of ANN (as well as GARCH-type models) is discussed in 2.2.4.

[33]The exploration of MLPNN with a larger structure for FX forecasting is the topic of Chapter 4.

Overfitting as a result of the massive number of parameters is another problem not to be neglected but it should be considered later in the training process. In the past few decades, its capability of adapting to and modelling with complex patterns has earned ANN models a significant amount of attention and research efforts in the academic world.

One attempt to test the weak-form EMH is made by Yao et al. (1996). They take past FX rates and technical indicators calculated from past FX rates as inputs of an ANN model. With empirical evidence from five currency pairs, they conclude that useful information can be made with historical FX rates (and their derived technical indicators) and forecasts based on the ANN models generate significant profits with the use of simple technical indicators.

Another effort to challenge the semi-strong EMH is by Eng et al. (2008), who use fundamental indicators such as CPI, GDP, trade balance and interest rates together with historical FX rates as inputs of ANN to forecast FX rates. They find that the inclusion of macro-economic indicators does not significantly improve the forecasting performance, despite the close economic relationship between those factors and FX rates. One possible reason they point out is that the macro-economic indicators they use are given at a quarterly frequency. Another potential reason is that the ANN they build has only 4-6 neurons in the hidden layer. It is possible that the relationship between the macro-economic indicators and FX rates are more complex than a simple ANN is able to capture.[34]

A research paper comparing ANN with time series models within the FX market context is Zhang and Hu (1998). They find that ANN models outperform linear models such as GARCH in terms of Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). They also focus on the effects of the number of neurons and layers of ANN models. The

[34]In Chapter 4, monthly collected macro-economic indicators are used and the number of hidden layer neurons is widely tested from 2 to 4096.

number of neurons is found to have more impact on the forecasting performance than the number of layers.[35]

Erdogan and Goksu (2014) also implement an ANN to forecast FX rates. However, ANN is used in different forecasting horizons. They show that there are significant differences in forecasting performance (measured by Mean Squared Error (MSE) and R^2) for different forecasting horizons. This result helps form one of the research questions of Chapter 5, in which the impact of forecasting horizons on the trading (instead of forecasting) performance is studied.

Subsections 2.2.1 and 2.2.2 explore the two popular types of forecasting models, time series models (GARCH-type) and ANN. It is certain that both types of models have advantages and drawbacks. To cover the other type of model's drawback while preserving its advantage, researchers have proposed the use of Time series and ANN hybrid models.

2.2.3 Time series and ANN hybrid models

As is discussed in Subsection 2.2.2, the combination of time series and ANN models increases forecasting capability of time series models and avoids expanding the size of the ANN model too much so that computation time can be controlled within reasonable range.

Two directions in which the combination of time series and ANN models can be made are discussed below.

The first direction is to use the estimated conditional variance from a time series model as one of the inputs of an ANN. This is essentially an ANN model with inbuilt time series specifications. Research papers in this direction include Guresen et al. (2011) and Pathberiya et al. (2017).

[35]Therefore in both Chapter 3 and 4, the number of layers keeps unchanged once determined.

The second direction adds an ANN directly to a time series model. This type of model is essentially a time series model with an ANN component.[36] Examples of research papers following this direction include Donaldson and Kamstra (1997), Zhang (2003), Aladag et al. (2009), Babu and Reddy (2014), and Kristjanpoller and Minutolo (2015).

Although almost all of the above papers in either direction succeed in building hybrid (time series - ANN) models that outperform time series/ANN models alone, none of the papers have discussed in detail how the coefficients of the hybrid models are estimated. The question of parameter estimation is of particular interest to hybrid models because they typically have much more parameters than traditional time series models. High-dimension parameter estimation is a big challenge for traditional parameter estimation algorithms (discussed in 2.2.4). Although high-efficiency libraries and GPU computation techniques (discussed in Chapter 4) can be applied to large ANN models to increase accuracy and speed up computation, their usage is restricted to a small number of inbuilt ANN models (e.g. MLPNN, RNN, LSTM). Therefore the hybrid models discussed above cannot utilise these estimation approaches.

In Chapter 3, an estimation algorithm (RSGA) is proposed to estimate parameters of the hybrid models which have more parameters than time series models but are not implemented within high-efficiency libraries.

2.2.4 Parameter estimation algorithms

Following Equation 2.6, the parameters $\mathbf{w}$ can be obtained via minimising the error function

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} \|\mathbf{y}(\mathbf{x}_n, \mathbf{w}) - \mathbf{t}_n\|^2, \tag{2.7}$$

where $\mathbf{y}(\mathbf{x}_n, \mathbf{w})$ is a collection of $\{y_k(\mathbf{x}, \mathbf{w}), k = 1, ..., K\}$ defined in Equation 2.6.

[36]Chapter 3 follows this direction in forecasting FX rates volatility.

By setting $\nabla E(\mathbf{w}) = 0$, $\mathbf{w}$ may minimise the error function $E(\mathbf{w})$. This may corresponds to a local minimum rather than a global minimum. However, for the application of neural networks finding a local minimum suffices. On the other hand, comparing several local minima may be needed to give a sufficiently good solution.

Since most of the time finding the analytic solution is impossible, iterative numerical procedures are carried out. Normally an initial value $\mathbf{w}_0$ is needed and $\mathbf{w}$ is updated through

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta\mathbf{w}^{(\tau)}, \tag{2.8}$$

where τ represents the iteration step.

One simple approach of the numerical procedure is called gradient descent. It chooses the weight update to comprise a small step in the direction of the negative gradient so that

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta\nabla\mathbf{w}^{(\tau)}, \tag{2.9}$$

where $\eta > 0$ is called the learning rate. At each step the weight vector moves in the direction of the greatest rate of decrease of the error function.

The main problem of any gradient descent-based algorithm is that it involves calculation of gradients, i.e. the differentiability of the error function in the parameter search space is a crucial assumption. When this assumption is violated, the singularity (differentiability) problem occurs and poorly estimated results may be obtained by the algorithm.

One type of non-gradient-descent algorithm is called Genetic Algorithm (GA). It mimics the biological gene passing process in which parents' genes are passed onto their children. In parameter estimation, a GA takes an initial sample of parameters to form the parental population and perform three types of operations on each generation (crossover, mutation and selection) to produce an offspring generation. In computation terminology, each generation represents an iteration. After each iteration, the candidate parameter

coefficients should produce improved results in terms of a pre-defined metric, i.e. the error function defined by Equation 2.7.

Andreou et al. (2002) apply GA to estimate parameters of a MLPNN model. Their estimated MLPNN structures have up to 46 neurons from the hidden layer. Compared with its preceding papers (normally using 5-10 hidden neurons), the use of GA for parameter estimation proves successful from the empirical evidence in FX rates forecasting. The MLPNN model estimated in this manner outperforms time series models (MA, AR) in terms of RMSE.

A different GA approach applied also in forecasting FX rates is conducted by Nag and Mitra (2002). Instead of estimating parameters of a given ANN structure, they use GA to select among different ANN structures (with different number of neurons, layers and types of activation functions)[37]. Their GA-selected ANN structures produce improved performance over selected GARCH-type models (ARCH, GARCH, AGARCH, EGARCH and GARCH-M) as well as fixed-structure ANN models.

Hansen et al. (2010) conduct a comparison analysis on the performance of 31 algorithms in an optimisation task. The task contains optimisations of a collection of functions with different extent of complexity. According to their empirical results, GA gives improved performance (in terms of percentage of successfully optimised functions) over BFGS, especially in optimising more complex functions.

In Chapter 3, therefore, GA is used as the basis of the estimation algorithm in order to perform a more accurate parameter estimation task of the more complex GARCH-ANN model.

[37]This is also considered as parameter estimation because the number of neurons, layers and type of activation function are also parameters called "hyperparameters".

2.3 Conclusion

In this chapter, firstly the significance of the FX market in terms of its size and global presence is demonstrated. This brings the initial motivation of research in the FX market.

While the FX market has a long history and a large number of market participants, the debate surrounding whether and to what extent the market demonstrates informational efficiency never disappears. A large number of papers have tested the EMH. It is a crucial topic because the extent of market efficiency directly determines the forecastability of FX return and volatility, two of the most significant metrics of FX rates from both the hedging perspective and the speculation perspective.

In this thesis, two forms of EMH are tested, namely weak form and semi-strong form. A fundamental approach is used to test the semi-strong form EMH with macro-economic indicators and a technical approach is utilised to test the weak form EMH.

To test EMH in either the fundamental or technical approach, a model needs to be built and trained. Two of the most widely used types of models are time series and ANN models. This chapter discusses the advantages and disadvantages of these two types of models in terms of their ease-of-training, forecasting power and flexibility.

The training process (estimation of model parameters) is such a crucial but frequently overlooked step in the training process. Two types of parameter estimation algorithms (gradient-based and genetic algorithm) are discussed. While both algorithms have been developed and extensively used for decades, they have the disadvantage of instability problem especially in estimating models with a relatively large number of parameters and complex formats.

In Chapter 3, a Recursive Simulation Genetic Algorithm (RSGA) is proposed for forecasting FX volatility. The RSGA increases estimation stability and accuracy by adding a recursively simulating process to the traditional GA. Volatility, as one of the two

significant metrics (along with return), is forecast based on historical FX rates. As the parameter estimation process is central in the modelling process, Chapter 3 serves a preparation purpose for the other two empirical chapters, although some implications for the weak form EMH are also drawn from the volatility forecasting results. The key concept of the RSGA (repetitive computation) is re-applied in the parameter estimation process of Chapter 4 and 5.

Chapter 4 focuses on return forecasting under the fundamental direction. With the implementation of a multilayer perceptron neural network (MLPNN) this chapter will test the semi-strong form of the EMH. The use of highly efficient ANN library (Keras) with the assistance of graphics processing unit (GPU) significantly increases computation efficiency.

Chapter 5 uses a long short term memory (LSTM) together with a technical trading rule to trade in the FX market. The relative strength indicator (RSI) is utilised to depict price movement patterns and the trading rule is based on the forecast of future price movement patterns by the LSTM model. As the input of the LSTM model contains only historical FX rates (or technical indicators calculated directly from historical FX rates), this chapter tests the weak-form of the EMH.

3 A Recursive Simulation Genetic Algorithm with a GARCH-ANN model for foreign exchange volatility forecasting

3.1 Introduction

In this chapter, a Recursive Simulation Genetic Algorithm (RSGA) is proposed to improve the forecasting performance on foreign exchange (FX) volatility by a given forecasting model (GARCH-ANN). To improve forecasting performance, recent literature such as Bollerslev et al. (2016), Kristjanpoller and Minutolo (2018), Chatziantoniou et al. (2019), Wang (2019), and Wu and Wang (2019) focus on forecasting model development and Bakas and Triantafyllou (2019) and Ma et al. (2019) focus on model input selection. Model development and input selection are two important steps in a general modelling process, e.g. return forecasting, volatility forecasting, hence they certainly deserve more focus.

However, the next step of modelling, namely the parameter estimation, draws much less attention in terms of the number of research papers discussing the relevant topic, e.g. Xu (2017) and Ding et al. (2019). Therefore this chapter aims at improving forecasting performance by designing an algorithm to enhance the parameter estimation process. The research questions focus on whether and to what extent can forecasting performance be improved by searching more widely for better coefficient values of the forecasting model and how to improve computation efficiency in the process.

Because volatility forecasting, together with return forecasting (to be discussed in the next chapter) are two important forecasting tasks in finance, this chapter will focus on improving modelling performance in the volatility forecasting process. In the following subsections, more details will be discussed on volatility forecasting, forecasting models, research direction and main results.

3.1.1 Volatility forecasting in the foreign exchange market

Volatility measures the dispersion of returns for a given asset. It is considered as one of the key metrics of describing the price behaviour of a financial instrument. On the buy-side, investors can estimate their investment risks by calculating volatility of their current assets holdings. They can also estimate their future investment risks by forecasting future volatility of their current assets or any assets to be purchased in the future. On the sell-side, investment banks which design and sell options need accurately estimated volatility to price the options. Policy makers such as governments, central banks and financial regulators also depend on accurately forecasting volatility to oversee how the market will perform in any future period.

Of all the different financial markets, the foreign exchange (FX) market is the largest and most heavily traded. According to the Bank of International Settlements (BIS) 2019 survey, the average daily trading volume increased by 30%, from 5.1 trillion US dollars in 2016 to 6.6 trillion US dollars in 2019. This daily trading volume of FX is 24 times the size of the daily trading volume of the global stock market.[38] Volatility plays such an important role in the FX market because it can behave so differently for different currency pairs during different time periods. For example, most of the major currency pairs underwent a high-volatility period in the financial crisis around 2008. The GBP/USD pair also moved with great volatility after the 2016 UK EU-membership referendum and since early 2019 as Brexit proceeded. For different currency pairs, volatility over the same period can vary by a large amount. Detailed literature on this topic is discussed in Section 3.2.

[38] According to the World Bank database (`https://data.worldbank.org/indicator/CM.MKT.TRAD.CD?end=2018&start=1975&view=chart`), the total trading volume of stocks in 2018 globally was 68.212 trillion US dollars, making the daily traded volume $68.212/253 = 0.27$ trillion US dollars.

3.1.2 Forecasting models and parameter estimation

To forecast volatility, the most frequently used type of model is the time series model. Examples of popular time series models are ARIMA (Box and Jenkins, 1970), ARCH (Engle, 1982), GARCH (Bollerslev, 1986) and many GARCH-type models such as EGARCH (Nelson and Cao, 1991) and GJR-GARCH (Glosten, Jagannathan and Runkle, 1993). The GARCH-family time series models listed above have the advantage of being capable of modelling the volatility clustering pattern.[39] More recently, another type of model called Artificial Neural Networks (ANN) has been introduced and used in financial modelling, e.g. forecasting FX volatility. Compared with time series models, ANN models have the advantage of being more flexible and able to capture more complex features especially non-linear patterns, see Donaldson and Kamstra (1997), Al-hnaity and Abbod (2016), Hajirahimi and Khashei (2016), and Jothimani and Shankar (2017). Literature on specific models is discussed in Section 3.2.

As is discussed above, one way to improve forecasting performance is to apply different types of models and find the best performing model for a specific modelling task. Another direction is to pursue the optimal parameter vector for a given model. This is an important yet often neglected aspect because most statistical packages with inbuilt parameter estimation functionality use the "set seed" function [40] in the parameter estimation process. The fact that every time the estimation process is conducted (on the same dataset) the same estimated coefficients are returned gives false impression of the estimated coefficients being optimal. However, the estimated coefficients are normally far from

[39]The volatility clustering pattern in finance refers to the empirical evidence that high volatile periods are more likely to be followed by high volatile periods and low volatile periods are more likely to be followed by low volatile periods.

[40]The "set seed" function stores a random result under a predetermined number called "seed" and every time when the random process is run the same output result will be produced as long as the same "seed" is quoted.

being optimal. Reasons are described below.

To optimise the parameter vector, a metric called likelihood is formulated based on the assumed distribution (Gaussian distribution, t-distribution, etc.), parameters of the model and the dataset. The method to find the "optimal" parameter vector is to choose it such that the likelihood can be as large as possible, i.e. maximised. The word "optimal" is used in a comparative sense because it is usually not possible to confirm whether the chosen parameter vector produces the highest likelihood.[41] Because the search space is usually so large, [42] it is not possible to experiment through all possibilities and find the optimum. Therefore, estimation algorithms are proposed to accelerate the search.

Traditional algorithms to estimate parameter vectors are usually gradient-based,[43] including Fletcher and Reeves (1964), Nelder and Mead (NM) (1965), Broyden, Fletcher, Goldfarb and Shanno (BFGS) (1970), and Byrd et. al. (1995). These methods may seem dated, but they are still being dominantly used in almost all areas of research whenever parameter estimation is involved, e.g. time series modelling, ANN forecasting, regression modelling.

More recently, research papers such as Grefenstette (1986), Cook (2000), Chang (2006), Elsayed et al. (2014), Thakur et al. (2014), Junghans and Darde (2015), Si et al. (2017) and Yalcinkaya et al. (2018) use a non gradient-based algorithm called Genetic Algorithm (GA) for parameter optimisation. GA, as the term "genetic" suggests, is a

[41] One special case in which the optimal parameter vector can be confirmed is when the likelihood function is a smooth function (differentiable everywhere), such as a simple quadratic function. However, none of the likelihood functions is smooth for the distributions discussed above.

[42] For example, assume we have a model that has six parameters to estimate and each of them ranges from 0 to 1. Then a search space from 0 to 1 in steps of 0.01, i.e. 0.01, 0.02, ... 0.99, 1, has a size of $100^6 = 1$ Trillion possibilities.

[43] The most popular gradient-based algorithm is called gradient descent with which the parameter vector moves in a way such that the gradient changes fastest.

search algorithm inspired by Charles Darwin's theory of natural evolution. Under the theory, the best possible gene combinations tend to be generated and passed onto future generations by operators such as selection, crossover and mutation. Detailed discussion on GA is provided in Section 3.2 and 3.3. Compared with gradient-based algorithms, GA has the advantage of avoiding the indifferentiability problem that a typical gradient-based method generally faces. Moreover, it produces similar, if not better estimation within a shorter time than a typical gradient-based method.[44] This chapter is unique in using an extension of the GA approach in the FX volatility prediction context.

All of the above algorithms start with randomly generating a parameter vector and change its components so that likelihood is maximised as fast as possible. Therefore, different random starting parameter vectors usually lead to different final estimated parameter vectors. There is no guarantee (in fact it is almost impossible) that a random starting parameter vector will generate an optimal estimated parameter vector. In fact, without comparing to estimation results generated by other starting parameter vectors, it is not possible even to tell whether the estimated parameter vector is one of the better ones.

However, as has been discussed, statistical estimation tools use the "set seed" function so that each time the user estimates the parameter vector of a model given a dataset, the starting parameter vector is randomly generated and set seeded. Regardless of who runs the estimation or how many times the estimation is conducted, the same estimated parameter vector is returned every time. This is problematic because the estimated coefficients of the parameters may be very poor, in terms of the modelling performance, and there is hardly any room for improvement because the estimated result is fixed once the dataset enters the model. The poor performance of the search algorithm becomes

[44]In an experiment conducted in this chapter, the GA outperforms the BFGS algorithm in terms of MAE and RMSE in a sample of 100 randomly simulated time series. Performances of BFGS and NM in estimating conditional variance of currency pairs are also significantly inferior to GA, see Subsection 3.5.4.

more significant as the estimated model gets more complex. Detailed reasons for this are explained in the next subsection.

3.1.3 Research direction and main results

Since GARCH-family models have a relatively simple likelihood function, algorithms (typically GA) may produce good estimated coefficients. However, as the complexity of the model increases, so does the complexity of the likelihood function, as well as the increased search space due to more parameters to be estimated. This leads to an increased possibility of reaching a local optimum, and very likely to be a poor local optimum. To be more clear about the problem that the mentioned algorithms (GA as a good performer) face, elaboration is required. As is mentioned previously, it is an unfortunate fact that, for parameter estimation, especially with a complex likelihood function, one can never know for sure whether or not a parameter vector generates the largest likelihood globally. However, one can know for sure that a parameter vector is not such a good estimation by finding, without much difficulty, other parameter vectors that generate larger likelihoods. With a single run of a GA, it is almost for sure that one always gets a not-so-good parameter vector because of the extremely large search space (difficult to get a good starting parameter vector) and complex likelihood function (very likely to trap into a local optimum and stop searching for better candidates). This is further illustrated and visualised in Section 3.4 and 3.5 where both types of estimation methods are applied to estimate an illustrative example (the GARCH-ANN) on several selected datasets and simulated datasets as well.

The intuition for this chapter is that instead of relying the estimation accuracy solely on a single good run of GA, an appropriate number of GA runs should be conducted in a relatively time-efficient way. The so-called Recursive Simulation (RS) Algorithm is designed and can be jointly used with GA, hence the name RSGA. The target is not just to improve estimation results by repeatedly running the estimation algorithm but also to

design the algorithm such that it is able to reach a globally "good" estimation in a fast way. Here the word "good" does not mean that the estimated parameter vector is necessarily the global optimum but it should be "comparatively accurate and time-efficient". A trade-off exists, i.e. in order to obtain a better estimated parameter vector, the extra computation time would have been excessive relative to the actual improvement and therefore not worth further excessive runs.

In short, the RSGA is designed in a way such that after the first iteration all the previous best candidates are selected into the initial population of the next iteration together with randomly selected candidates. This will make sure the performance never becomes inferior as the iterations proceed and thus speeds up the simulation process.

The research results of this chapter are previewed below. For the 100 series simulated with 100 randomly simulated coefficient vectors, the RSGA generates a 24% improvement over GA in terms of the Mean Absolute Distance (MAD) and a 16% improvement over GA in terms of the Mean Ratio Distance (MRD). This improvement is further supported by the estimation of a GARCH-ANN model with six currency pairs (GBP/USD, EUR/USD, USD/JPY, USD/CHF USD/RUB and USD/ZAR) from 2008 to 2017. The RS Algorithm again improves significantly the estimation accuracy of conditional variance in terms of two metrics (Mean Absolute Error and Root Mean Squared Error) for all six currencies. The average in-sample improvements of RSGA over GA are 74.6% and 74.4% in terms of MAE and RMSE respectively and the out-sample improvements are 74.5% and 70.2% in terms of MAE and RMSE respectively.

The empirical results also have some implications for the EMH. Based on the forecasting errors of volatility of different currency pairs, two main findings are drawn as below. First, the emerging market pairs USD/RUB and USD/ZAR are easier to forecast than the four developed market pairs. Therefore, developed markets do show more efficiency than emerging markets, as expected. Second, for most of the pairs, the extent of difficulty for

forecasting varies as time goes on. USD/CHF is the pair for which the extent of difficulty for forecasting is most consistent. The fact that USD/CHF is consistently more difficult to forecast (indicated by high errors) indicates that the Swiss market is one of the most (weak form) efficient markets.

The improvement of the RS Algorithm over the original methods is more significant in two cases. First, the RS Algorithm provides more significant improvements when the target model to be estimated has a complex likelihood function. Second, the RS Algorithm excels in the situation where the search space is large (the number of parameters is large and the range of each parameter is wide) because in this situation the number of possibilities from a simulation of the starting values is so large that it's much less possible for the randomly-chosen starting value to lie close to the global optimum value. Therefore traditional single-run algorithms tend to perform poorly on this type of problem.

The remainder of the chapter is organised as follows. Section 3.2 reviews relevant literature on the FX market, volatility forecasting models and parameter optimisation algorithms. Section 3.3 describes models and algorithms used (or built and used) in this chapter, including the GARCH-ANN model, the Genetic Algorithm and the RS Algorithm. Section 3.4 simulates 100 series and compares accuracy and timings of estimating the coefficients between GA and RSGA. Section 3.5 utilises a real-world dataset (six FX time series), provides estimation results and compares accuracy of modelling the conditional variance between the GA and RSGA. Section 3.6 concludes this chapter together with providing potential further research directions.

3.2 Literature review

3.2.1 The foreign exchange market

According to the Bank for International Settlements' (BIS) foreign exchange (FX) turnover survey 2019, the US dollar retained its dominant currency status, being on one side of 88% of all trades. Emerging market currencies again gained market share, reaching 25% of global turnover. The dominance of USD in the FX market makes it the most researched currency (being on one side of most researched currency pairs) and the increasing trading share of emerging market currencies earns them more attention from the academic world as well as the financial world.[45]

Chronologically, the FX market liquidity varies within the day based on trading hours in different areas. King et al. (2011) discuss that throughout a trading day overall market liquidity is highest when both London and New York are open and liquidity is low between 19:00 and 22:00 GMT when most New York traders end their trading day and most Sydney traders are on their way to work. In terms of a given currency, the liquidity tends to be higher during its local trading hours.

Geographically, Cheung et al. (2019) study the ongoing diffusion of Renminbi (RMB) trading across the globe. They find that RMB trading is converging to the geographical pattern of all currencies. Other emerging market currencies such as Mexican peso (MXN),Hong Kong dollar (HKD), Indian rupee (INR) and Korean won (KRW) show a similar pattern, with a slower convergence rate.

In the short term, FX markets are affected highly by macroeconomic announcements. Chaboud et al. (2007) study the FX market reaction to U.S. macroeconomic announcements. They find that sharp pickup in trading volume (and usually volatility as well)

[45]For more descriptions on the FX market refer back to 2.1.1.

generally occurs in the minutes following news announcements. The rise in trading volume happens even if the data release is entirely in line with market expectations.

A similar research by Piccotti (2018) shows that before jump and cojump events, exchange rate quote volume, liquidity, signed order flow, and informed trades are at high levels and following jump and cojump events, quote volume and return variance remain at high levels while liquidity, informed trade, and signed order flow drop to low levels.

In the long term, Karnaukh et al. (2015) show that FX liquidity is mainly affected by funding constraints and global risk dynamics. They suggest that supply side factors are key drivers of FX liquidity and FX traders are exposed to cross-market factors, e.g. FX liquidity tends to decrease with volatility of global stock and bond markets. These effects are stronger for developed country currency pairs.

In addition to liquidity, volatility is also researched for the FX market. As is discussed in Section 3.1, volatility plays an important role in risk management. Christoffersen and Diebold (2000) show that although volatility forecasting is of significant relevance for risk management at the short horizons, it may be much less important at longer horizons. With empirical evidence they show that in the FX market volatility forecasting provides useful information for risk management under five days of forecasting horizon.

As has been discussed in 3.1 and 3.2.1, the FX market is arguably the most influential financial market and volatility is one of the key metrics of FX rates. Therefore, volatility forecasting models are of particular interest to many researchers and practitioners.

3.2.2 Volatility forecasting models

In terms of volatility forecasting models, various types of models have been proposed by researchers.[46] This subsection will focus on discussing models developed/utilised for

[46]Refer to Chapter 2 for more general background on volatility forecasting models.

volatility forecasting. All papers discussed in this subsection focus on either developing the model or selecting input variables, in order to improve forecasting performance. None of them focus on parameter estimation, which is the aim of this chapter. Therefore, the purpose of discussing these models is not to analyse the details (benefits/shortcomings) of their model development but to provide a general introduction to some relevant models in the area of volatility forecasting and then introduce the model (GARCH-ANN) to be used for developing the parameter estimation algorithm. [47]

Wang et al. (2001) forecast the FX volatility, with a flexible parametric GARCH model. It is flexible in the sense that the error distribution is based on the exponential generalized beta (EGB) family of distributions, rather than the Gaussian or t-distribution. Goodness-of-fit tests favour the proposed model over the GARCH-t distribution for six currency pairs with the USD.

Although modified versions of the GARCH model are proposed by researchers and empirical evidence of better performance of the proposed models is presented, a different conclusion has also been drawn regarding the effectiveness of more "advanced" GARCH-family models. Hansen and Lunde (2005) compare 330 GARCH-family models in terms of their performance in forecasting the conditional variance of FX rates. They find no evidence of a consistent outperformance from any of the more sophisticated models over a GARCH (1,1). Their research supports the effectiveness of the GARCH (1,1) model in terms of both simplicity and performance in forecasting FX volatility.

Hafner (1998) is one of the earlier examples of applying modified versions of GARCH-family models for volatility modelling. It is particularly noteworthy because it uses a non-parametric ARCH model to forecast the volatility of FX rates. The model differs from a traditional GARCH-family model by using a non-parametric function [48] instead

[47] For more details on general time series (GARCH) models refer back to 2.2.1.

[48] A non-parametric function is one that at least some part of the function is not determined by pre-defined

of a deterministic function such as a weighted sum of the past errors and conditional variances. Most of the traditional GARCH-family models are parametric. By including non-parametric components in the model the flexibility of the model will be enhanced. This idea of using non-parametric function is inspiring because other forms of non-parametric/non-linear function may also be applied to aim at producing good forecasting results.

In addition to time series model such as GARCH-family models, researchers also implement ANN models together with a GARCH-family model. The idea is similar to Hafner (1998) in the sense that non-parametric/non-linear components are included in the model. The inclusion of an ANN component also aims at increasing the flexibility and therefore enhancing potential forecasting performance of the model. Examples of the hybrid GARCH-ANN models include the following.

Donaldson and Kamstra (1997) are one of the earliest to implement a hybrid model that combines a GARCH model with an ANN. They evaluate the model's ability to model and forecast stock return volatilities in several countries. They show evidence that the ANN captures volatility effects overlooked by GARCH-type models. They also mention in their research that without the ANN component, the GARCH model on its own, despite its simplicity, lacks the ability to adequately capture non-linear relationship between the conditional variance and the lagged errors. Donaldson and Kamstra (1997) is considered as pioneering research. It is not the only paper that utilises the GARCH-ANN model or any other time series-ANN hybrid models. Later research papers that adopt a similar approach are briefly presented below.

Monfared and Enke (2014) implement ANN models to improve the performance of a GARCH-GJR model and show that in high-volatility periods the ANN component improves the forecasting ability of GARCH-GJR more than it does in low-volatility

parameters but determined from data.

periods. Kristjanpoller and Minutolo (2015) apply an ANN to a GARCH model to forecast the gold price (spot and future) volatility. An average reduction of 25% in mean average percentage error is realised using the GARCH and ANN hybrid model. Bildirici and Ersin (2009) utilise an ANN with different GARCH-type models to evaluate the volatility of daily returns at the Istanbul Stock Exchange from 1987 to 2008. There is at least a 5% reduction in RMSE with different GARCH models. The largest reduction of more than 23% is with a TGARCH model. The GARCH-ANN reduces the RMSE of a GARCH model by over 8%. Lu et al. (2016) apply an ANN to different GARCH-type models to forecast volatility of log-returns of the Chinese energy index. The EGARCH-ANN model outperforms other models (GARCH-GJR-ANN and those without ANN) in terms of RMSE.

3.2.3 Parameter optimisation algorithms

The parameter optimisation algorithms discussed below are applicable to modelling problems in not only finance but also many other areas (as long as parameter estimation is involved). This subsection selects literature which is most pertinent to the focus of this chapter.[49]

The traditional and most widely used estimation algorithms are called "gradient-based" algorithms. Typical examples include Fletcher and Reeves (1964), Nelder and Mead (NM) (1965), Broyden, Fletcher, Goldfarb and Shanno (BFGS) (1970), and Byrd et. al. (1995). These algorithms adopt the "gradient descent" approach which aims at optimising parameter vectors in the direction of fastest decreasing likelihood function. However, because this type of algorithm involves calculating gradients of a large number of different points, problems may very often occur when a certain number of points are non-differentiable. This problem arises much more often in high dimensional spaces or

[49]For more details on general parameter estimation algorithms refer to 2.2.4.

for complex models, i.e. models with over five parameters to estimate or with complex likelihood functions. No further focus will be concentrated on this type of algorithm (gradient-descent) because of these problems and limited recent literature exists on developing those algorithms.

To help solve the differentiability problem typically faced by the gradient-based algorithms, another type of optimisation algorithm which does not involve gradient calculation was introduced, namely the GA. GA became popular through the work of John Holland in the early 1970s, and particularly his book Adaptation in Natural and Artificial Systems (1992). This subsection focuses on a discussion of the literature. Details on how a basic GA actually works are presented in Section 3.3.

Later research papers have developed the GA approach in various directions. Elsayed et al. (2014) propose a GA with a new multi-parent crossover. They argue that while biologically offspring is produced by two parents, this need not be the case in parameter optimisation. The inclusion of a "third parent" enables their algorithm to visit a wider searching space within a shorter time.

Similarly, Thakur et al. (2014) focus on improving the crossover operator. They utilise a bounded exponential crossover, i.e. always create offspring within the variable bounds. The modified GA is able to produce a larger number of successful runs within a shorter time than a traditional GA.

A different approach is implemented by Junghans and Darde (2015) for a modified GA. They introduce a hybrid GA and Simulated Annealing (SA) algorithm. [50] Empirical evidence shows that their hybrid algorithm can provide more reliable optimisation results than those provided by the GA alone within a relatively short period of time. Another

[50] A SA algorithm attempts to avoid trapping into a local optimum by sometimes moving to a neighbourhood that decreases the value of the likelihood function.

key point they also illustrate is that GA does not always provide solutions close to the global optimum because of the randomness in the search process. This is an important result because despite being able to improve optimisation performance, none of the pre-mentioned GA algorithms deal directly with the problem due to randomness in the estimation process. This brings out the incentive of the RSGA algorithm proposed in this chapter because statistically speaking, the best way to reduce the randomness problem is through repetitive simulation.

As is discussed in the previous paragraph, all of the pre-mentioned research papers concentrate on improving the crossover or redesigning the mutation process. None of them focus on the initial parent population formation process which is the first step of GA. The lack of literature on this matter may be due to the fact that the formation of initial population is a completely random process, i.e. randomly simulate an initial parent population (a certain number of coefficient vectors in the case of a parameter estimation task). However, as Junghans and Darde (2015) suggest, it is this randomness that generates the instability in the estimation performance of a GA because whether the randomly simulated initial parent population is "good" (some members are close to the optimum) or "bad" (none of the population members is anywhere close to the optimum) will affect the estimation result and sometimes the difference between the estimation performances from different initial parent populations can be very large.

Evidence of the instability of a GA's performance is shown in Section 3.5 (see Figure 3.4). The figure shows the significance of the work within this chapter in addition to previous research on the improvement of GA because improving GA via crossover or mutation will help improve the performance in some cases but even the best-performing GA may sometimes fail to obtain a good estimation if it starts with an inappropriate initial parent population. Therefore the RSGA is proposed to reduce the effect of randomness in the initial parent formation process, in a time-efficient way.

The RS algorithm (with appropriate[51] programming) can in fact be applied to any of the pre-mentioned GAs to improve the estimation performance on top of the improvement they make over the most basic type of GA. However, in this chapter the GA that the RS works on is chosen to be the most basic type of GA for the purpose of easy illustration and computation time reduction.

In the next section, descriptions on the estimation model, the basic GA and the proposed RSGA will be discussed in detail.

[51] "Appropriate" means not too much extra programming but some adaptation to different models is needed in order for the RS algorithm to work on other types of GAs.

3.3 Model description

3.3.1 Model to estimate: GARCH-ANN

The model used for volatility modelling in this chapter is the GARCH-ANN model, proposed by Donaldson and Kamstra (1997). As has been discussed in Section 3.2, they evaluate the model's ability to model and forecast stock return volatilities in several countries. It's a hybrid model that combines a GARCH model with an ANN model. They present evidence that the ANN captures volatility effects overlooked by GARCH-type models. The model is displayed in Equations 3.1 - 3.4.

$$\sigma_t^2 = \alpha + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2 + \sum_{j=1}^{q} \gamma_j \epsilon_{t-j}^2 + \sum_{h=1}^{s} \xi_h \Psi(z_t \lambda_h), \tag{3.1}$$

$$\Psi(z_t \lambda_h) = \left[1 + \exp\left(\lambda_{h,0,0} + \sum_{d=1}^{\nu} \left[\sum_{w=1}^{m} (\lambda_{h,d,w} z_{t-d}^{w}) \right] \right) \right]^{-1}, \tag{3.2}$$

$$z_{t-d} = [\epsilon_{t-d} - E(\epsilon)] / \sqrt{E(\epsilon^2)}, \tag{3.3}$$

$$\frac{1}{2} \lambda_{h,d,w} \sim \text{uniform}[-1, +1]. \tag{3.4}$$

In Equation 3.1, the first 3 terms on the right-hand-side of the equation form a GARCH model while the last term corresponds to the ANN part. Engle (1982) and Bollerslev (1986) discuss in their papers that the parameters α, β_i and γ_j all need to be non-negative to guarantee that the conditional variance σ_t^2 is non-negative. For this GARCH-ANN model, however, even if the GARCH component is non-negative there is no guarantee that the ANN component and the sum of the two components are also non-negative. Therefore all restrictions on the GARCH parameters to guarantee non-negative σ_t^2 are removed. Instead, an extra step is added to the parameter estimation process such that only parameter vectors generating non-negative σ_t^2 are preserved for further steps.

Donaldson and Kamstra (1997) argue that without the ANN component, the GARCH model on its own, despite its simplicity, lacks the ability to adequately capture non-linear relationships between the conditional variance σ_t^2 and the lagged innovations ϵ_{t-j}.

Although there are variations of the Donaldson and Kamstra's model (refer back to subsection 3.2.2), they share a similar methodology - to include non-parameter/non-linear components in the model. There are two main reasons for using the GARCH-ANN model as the target model for the parameter estimation algorithm build-up in this chapter. The first reason is that it is able to capture both linear (with the GARCH[52] component) and non-linear (with the ANN component) patterns. This is very important because financial time series are one of the most complicated time series to analyse and it is expected that they would exhibit both linear and non-linear elements. One advantage of having an ANN component in the model is that by adjusting the number of hidden layers and neurons it can theoretically capture any level of complexity in the dataset.[53] The possibility of over-parameterisation and potential overfitting is one aspect to be considered but it is not discussed in detail because the focus of this chapter is on parameter estimation of the GARCH-ANN model. The second reason for using the GARCH-ANN is that it has a relatively complex likelihood function with multiple parameters to estimate. It is a situation where some traditional estimating algorithms start to suffer and therefore the RS algorithm and the idea behind it are likely to be well suited to this problem and application.

[52]There are GARCH-family models (such as Exponential GARCH and Realised GARCH) which are non-linear or non-parametric. However, the GARCH model in this chapter refers to the traditional GARCH approach proposed by Bollerslev (1986).

[53]Refer to Chapter 2 and Chapter 4 for more details of ANN.

3.3.2 Basic estimating algorithm: GA

As has been discussed in Section 3.2, a gradient-based algorithm such as BFGS or NM estimates the coefficients of a target model, by maximising the model's likelihood function. This works very well for most of the time and for most models. However, as the complexity of the target model increases so does its likelihood function and chance of getting to a point where the function is indifferentiable increases. An indifferentiable point may affect the optimisation process of a gradient-based algorithm as it involves calculating gradients at different points.

This potential problem is especially worrisome in this case because of the complex likelihood function of the GARCH-ANN model. As is discussed in Section 3.2, the GA is a non-gradient-based algorithm therefore does not have the potential differentiability problem. A GA typically involves the following steps:

1. formulating an initial population P_0 (a collection of parameter vectors);
2. evaluating the performance of each individual parameter vector p_i by using a fitness function;[54]
3. selecting individuals for reproduction of a new individual;
4. applying the two genetic operators: crossover and mutation;
5. iterating Step 2-4 until a stopping criterion is reached.

Because the first step of GA is simulating an initial population there is an element of randomness in the process. The problem, however, is that by formulating an initial population, an initial "guess" is made and then evolves to improve itself through reproduction and mutation. Since this initial "guess" is made randomly, in cases where it's very far away from the global optimum point, the possibility that it will finally reach the global optimum point or somewhere near the global optimum is very low. Another problem is

[54] As is discussed in Subsection 3.3.1, only parameter vectors generating non-negative σ_t^2 are preserved.

pointed out by Ono et al (2003) that GA may fail to generate offspring in some cases such as when the population size is small relative to the search space and that they also have difficulty in finding an optimal estimation which is near the boundaries of the search space.

3.3.3 Proposed estimating algorithm: RSGA

One solution to this problem is to run a reasonable number of times of GA each time with a new initial population generated. In this way it is more likely to include an initial population somewhere closer to the global optimum point. The Recursive Simulation (RS) algorithm is therefore proposed in addition to the GA. The RSGA is divided into the following steps:

1. Choose the number of successful simulations needed for the loop to end ($n.sim$) and the maximum number of iterations ($n.max$). Set $i = j = 0$;
2. While $i < n.sim$ and $j < n.max$, compute the GA with all the previous selected successful simulation coefficients (known as suggestions) together with simulated population;
3. If the estimated GA[55] generates larger likelihood L (this is considered as a successful simulation), add the estimated coefficients to the suggestions for the simulation and increase i and j by one. Else only increase j by one;
4. Go to Step 2.

A flowchart is displayed in Figure 3.1. The addition of the successful coefficients to suggestions makes a consecutive of simulations not parallel but recursive because it ensures every new simulation is based on previous successful simulations, i.e. new results can only offer improvement (or unchanged) but not inferior to the previous. In

[55]Only simulations which generate convergent estimations (the estimated coefficients stablise in values) enter the flowchart.

this way the optimisation process accelerates as time goes, which makes the algorithm more time efficient and reduces computation resources.

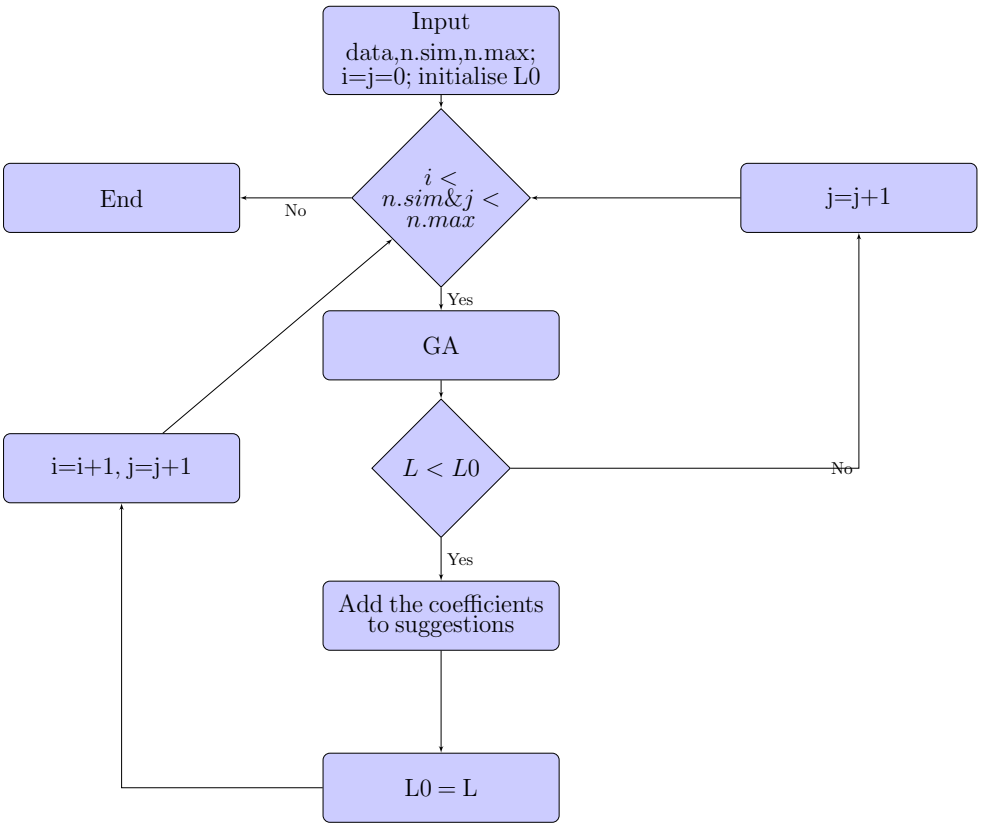


Figure 3.1: A flowchart of the Recursive Simulation Algorithm for GA. (This flowchart is a visual representation of the four steps in Subsection 3.3.3.)

3.4 Estimation with simulated series

With a typical real-world dataset, even if we estimate coefficients of a model, we would never know how close the estimated coefficients are from the "real" values of the coefficients. With simulated datasets, we can compare the estimated coefficients with the coefficients that were used to simulate the dataset. The closer these coefficients are to each other, the more successful the estimation algorithm is in terms of parameter estimation accuracy.

In this subsection, 100 coefficient vectors are randomly simulated from a Uniform(-1,1) distribution and then these coefficient vectors are used to generate 100 simulated datasets with the GARCH-ANN model.[56] The length of each of these 100 series is chosen to be 200 data points because it's neither too short to be informative nor too long to be computationally efficient. The error terms ϵ_t are generated from a normal distribution, similar to how the error terms are generated in a traditional GARCH model. After the series are generated and estimated with RSGA, the estimated coefficient vectors can be compared with the original coefficient vectors. Figure 3.2 shows a random sample of 6 of the 100 simulated series. The reason for limiting the range of the variables to (-1,1) is to exclude those "explosive" series. [57] This range contains the range of a typical financial log-return series.[58]

To measure the difference between the estimated coefficients and original coefficients used to simulate the series, two metrics are used. The first metric (one of the more widely

[56] Refer to Equation 3.1 as the data generating process.

[57] "Explosive" means after the daily return reaches an extreme level, it does not come back to the average level, i.e. 0, whereas for most financial assets the return level almost never stays at an extreme level indefinitely.

[58] A daily log-return within (-1,1) corresponds to a daily return within (-271.8%, 271.8%), within which lie almost all observed financial returns.

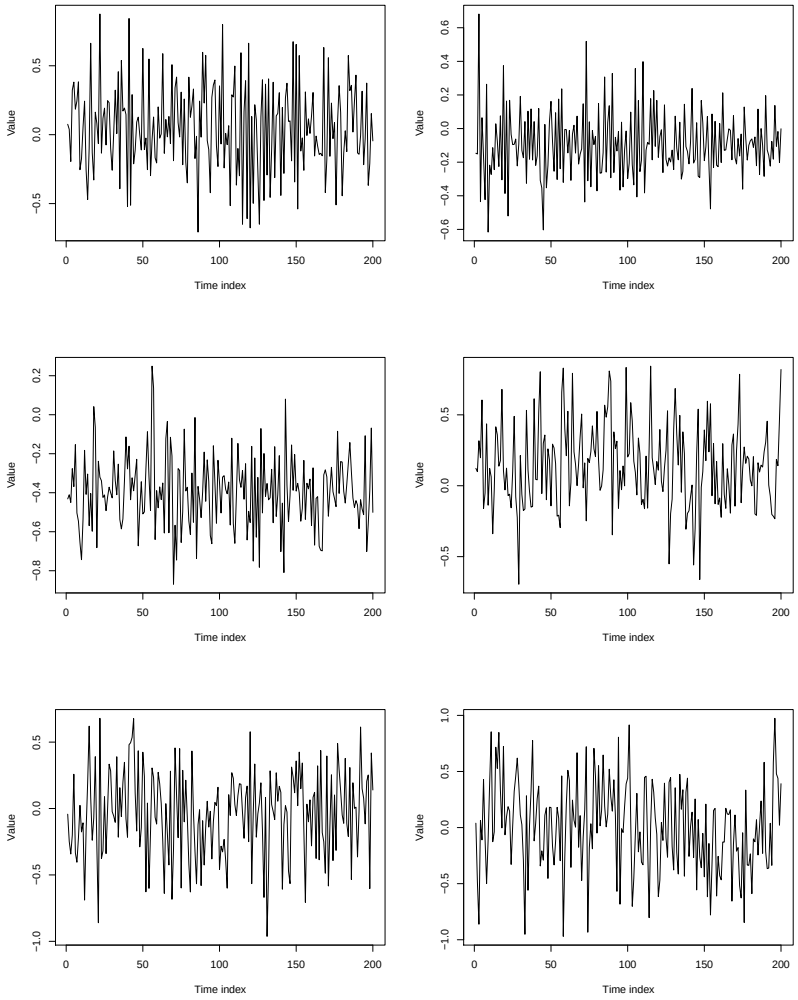


Figure 3.2: A random sample of 6 of the 100 simulated series with length of 200 data points.

used metrics) d (also called Mean Absolute Distance) measures the average Euclidean distance between the estimated coefficients and the coefficients used for simulating the time series. A formal definition is given in Equation 3.5.

$$d = \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{m} \left(p'_{ij} - p_{ij} \right)^2 \right)^{\frac{1}{2}}, \tag{3.5}$$

where n is the number of simulated time series to be estimated, in this case 100, m is the length of the parameter vector, for the GARCH-ANN we use, m is 6. p' is the estimated parameter vector and p is the parameter vector used for simulating the time series.

The Euclidean distance d measures directly how far the estimated coefficients are from the original coefficients. However, d weighs all differences equally, regardless of the relative sizes of different coefficients. This could result in false advantage/disadvantage when comparing two estimated parameter vectors.The following example discusses this particular problem and the solution to it. Assume we only have one series with a 2-D parameter vector to estimate, i.e. $n = 1$, $m = 2$ in Equation 3.5. If the two parameter values (used to generate the series) vary a great amount in size, the Euclidean distance d may suffer in comparing estimated parameter vectors. Below is a hypothetical example to illustrate a case where the Euclidean distance fails to perform comparison well enough. If the target parameter vector is $p = (0.11, 0.00002)$ while the first estimation is $p'_1 = (0.12, 0.003)$ and the second estimation is $p'_2 = (0.13, 0.00001)$. The Euclidean distance for the first estimation d_1 is calculated to be 0.01 while d_2 is 0.02. Based on d, p'_1 is more accurate than p'_2. However, one might argue that p'_2 in fact is a better estimation than p'_1 because p'_2's estimation on the second parameter is much more accurate. This is because the Euclidean distance measures the errors in the absolute sense. To be able to compare two estimations after taking into account the relative magnitudes of the coefficients, another metric r (Mean Ratio Distance) is introduced and defined in

Equation 3.6.

$$r = \frac{1}{mn} \sum_{i=1}^{n} \sum_{j=1}^{m} |(p'_{ij} - p_{ij})/p_{ij}| \tag{3.6}$$

For the same task as above, r_1 and r_2 is calculated to be 75.04 and 0.34 respectively. This time p'_2 is more accurate than p'_1 based on r. While r takes into account the relative values of the coefficients, it may sometimes exaggerate the outperformance of one estimation over another, i.e. the goodness of p'_2 relative to p'_1 is not reflected accurately by the sizes of the values 75.04 and 0.34. Therefore both metrics are computed and considered so that both absolute and relative comparisons can be made.

Table 3.1 displays a summary of the comparison between GA and RSGA, based on d, r, probability of getting convergent estimates (p) and computation time [59] in seconds per series estimated (t). From the table it can be seen that while estimation time increases significantly, RSGA makes significant improvements over GA in terms of d, r and p. At 96%, the original probability of convergence may not cause concern sometimes. However, in a regular modelling process, when a convergence problem occurs it usually means a "stop-and-check" to find a solution and sometimes even a new model is needed to potentially resolve the model. The extra time (and work) needed in those situations are usually much more significant than the extra computation time used by RSGA.[60]

[59] All computations were conducted on a laptop with a typical Intel Core i7 processor.

[60] With a more advanced processor, the extra time that RSGA consumes is expected to decrease significantly. The extent of time decrease is more significant than that of a normal GA (because hardware upgrades generate greater efficiency gains for complex tasks than for less complex tasks).

	d	r	p	t
GA	1.46	1.34	96%	26.21
RSGA	1.11	1.12	100%	465.92

Table 3.1: Comparison of parameter estimation accuracy (d and r), probability of getting convergent series (p) and estimation time in seconds per series (t) between GA and RSGA.

3.5 Modelling and forecasting volatility of foreign exchange rates

With simulated series, it is possible to measure how different algorithms perform in estimating parameter values in a model by comparing the estimated parameter values with the values used to generate the series.[61] However, most financial time series (including FX rates) move significantly differently from simulated series. In fact, one might argue that financial time series never follow any distribution, at least not in the longer term. One example is that financial time series typically have much fatter tails than a normal distribution (meaning that extreme returns occur more frequently than a normal distribution). Even if a fatter-tailed distribution such as a t-distribution is considered, there are other characteristics of financial time series that a t-distribution can never adequately capture, which in some cases may put the t-distribution at an inferior position than the normal distribution. Therefore, modelling with financial time series is very different from modelling with simulated series.

One difference in performance evaluation between the two types of series is that with real-world time series, since they are not simulated from a predetermined parameter vector, it is not possible to directly compare the estimated coefficients with the predetermined parameter vector. Under those circumstances, one way to measure performance of

[61] Because the error distribution of the GARCH-ANN model is assumed to be normally distributed, the simulated series also have the normal distribution specification.

the algorithm is to compare the forecast conditional variance with the real conditional variance of the dataset which can be approximated by the square of conditional residuals, see Donaldson and Kamstra (1997).

In the following subsection, FX time series used for this chapter are described and summary statistics of the FX log-returns are listed.

3.5.1 Data description

In this subsection, six FX return series are used for the purpose of modelling and forecasting time-varying volatilities with the GARCH-ANN model. The six currency pairs include GBP/USD, EUR/USD, USD/JPY, USD/CHF, USD/RUB and USD/ZAR. Daily prices of the six currency pairs are obtained for the time period 1 January 2008 to 31 December 2017 (a total of 2609 data points spanning 10 years). All datasets are collected from Thomson Reuters Datastream. For the rest of this chapter, GBP, EUR, JPY, CHF, RUB and ZAR will be used to stand for the daily log-returns (2608 data points) of the six pairs and "USD" is neglected because it is the counterparty of all other currencies.

To split the dataset for modelling and testing purposes, it is common to create a training set and a test set. A training set is a subset of the dataset used for building up the model and a test set is the rest of the dataset to test how the model built from the training set works. Following the way in which Sermpinis et al. (2012) split their training and test set (ratio of number of data points is approximately 2:1 between training and test set), the datasets (six FX log-returns) in this research are split in to a training set containing the first 1739 observations and a test set containing the rest 869 observations.

Figure 3.3 shows a time series plot of the six currency pairs. The scale of the y-axis is set to be the same for all pairs so that a rough visualisation of volatilities can be made. One

can observe by eyes that GBP, EUR, JPY and CHF have smaller volatilities although GBP and CHF have some very high-volatility periods while EUR and JPY are more stable throughout the period. RUB and ZAR, on the other hand, are more volatile during the period and RUB experienced arguably one of the most volatile periods in year 2015. This difference in volatilities is also revealed in the summary statistics in Table 3.2. The reason for choosing FX pairs with significant differences in volatility patterns is that it will enable us to test whether the model and the algorithm are only good at modelling certain time series, or they are capable of modelling time series with a wider range of characteristics. RUB and ZAR are selected among other volatile currencies because USD/RUB and USD/ZAR have longer data availability than other minor pairs.

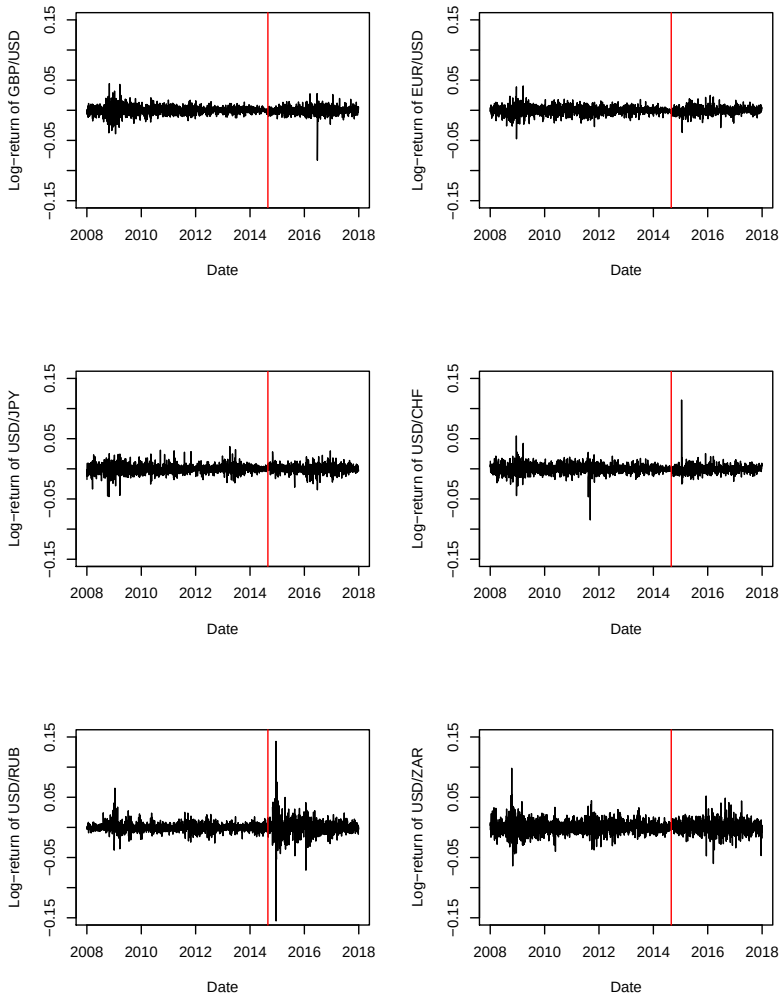


Figure 3.3: Daily log-returns of GBP, EUR, JPY, CHF, RUB and ZAR from 1 January 2008 to 31 December 2017. All currency pairs are against USD. The red line represents a separation of the training set from the test set. Source of data: Thomson Reuters Datastream.

Currency	Mean	Med	Min	Max	Std. Dev
GBP	-1.5e-04	0	-0.0829	0.0443	6.6e-03
EUR	-7.9e-05	0	-0.0474	0.0404	6.4e-03
JPY	2.7e-06	0	-0.0460	0.0370	6.9e-03
CHF	5.8e-05	-7.2e-05	-0.0848	0.1142	7.7e-03
RUB	3.3e-04	0	-0.1552	0.1427	1.0e-02
ZAR	2.3e-04	0	-0.0639	0.0981	1.1e-02

Table 3.2: Summary statistics of the daily log-returns of the six currency pairs from 1 January 2008 to 31 December 2017. The median values of 0 in the table are accurate to 1×10^{-6}. All currency pairs are against USD. Source of data: Thomson Reuters Datastream.

3.5.2 Parameter estimation with GA and RSGA

Each time a GA is run over the same dataset the estimation result is different due to the random choice of initial parent population. Therefore it is reasonable to compute multiple times of GA and use their average performance as the overall performance of the GA. In this chapter, GA is computed 100 times for each of the six currency pairs.

Tables 7.1 and 7.2 in the Appendix show estimated coefficients by one randomly selected run of GA (out of 100 runs), together with the coefficients estimated by RSGA (in brackets). Although most coefficients have the same sign for the two algorithms, the sizes of the coefficients for GA and RSGA are generally unequal. This suggests the potential difference in estimation performance of the two algorithms.

3.5.3 Investigating in-sample and out-of-sample performances

The Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), defined in Equations 3.7 and 3.8, are used to measure the modelling and forecasting performance of the conditional variance from the estimated model. According to Willmott (1982), RMSE and MAE are among the best overall measures of model performance, and he recommends the inclusion of both measures for performance comparison in researches.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|x_i - y_i|, \tag{3.7}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2}, \tag{3.8}$$

where n is the number of forecasting points, x_i is the forecast value and y_i is the benchmark value. In this research, x_i is the one-day ahead forecast conditional variance σ_i^2 and y_i is the real conditional variance which is unobservable but can be approximated by the square of residuals u_i^2, refer to Donaldson and Kamstra (1997).

One-day ahead in-sample and out-of-sample MAE and RMSE for the six currency pairs are computed for each of the 100 rounds of GA. Box-plots of MAE and RMSE of the 100 rounds are displayed in Figure 3.4 to help visualise the level deviation in performance of a single GA and therefore explain on its own the advantage of the RSGA. As can be seen from Figure 3.4 that there is a wide range of MAE and RMSE values across the 100 rounds of GA and a number of significant outliers (bad performers) are observed. This shows us how unstable a single run of GA is in terms of forecasting performance.

Tables 3.3 and 3.4 display the MAE and RMSE values of GA, calculated as the mean of the error values obtained from the 100 rounds in comparison with the error values from the RSGA (in brackets). For all currency pairs, the RSGA produces smaller MAE and RMSE than GA. The percentage improvements, calculated as the percentage difference

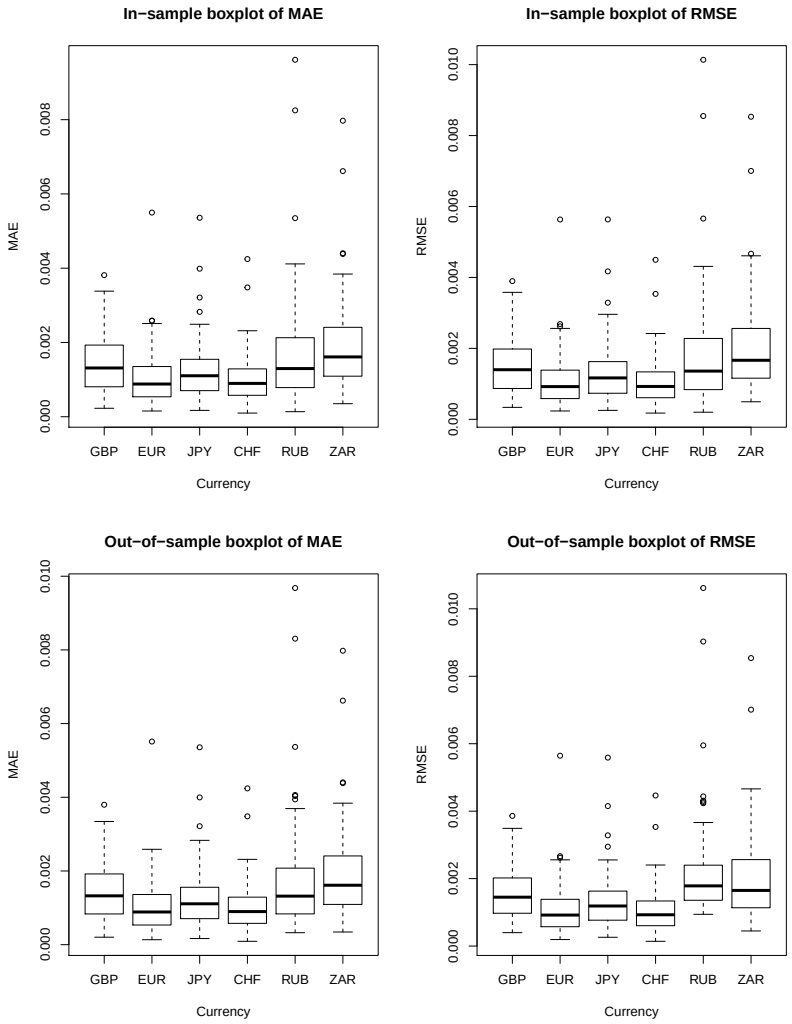


Figure 3.4: Box-plots of MAE and RMSE for the 100 rounds of GA for the six currency pairs. Source of data: Thomson Reuters Datastream.

in the errors of the RSGA from GA, are displayed in Table 3.5.[62]

	MAE ($\times 10^{-4}$)	RMSE ($\times 10^{-4}$)
GBP	12.22 (3.26)	12.81 (3.46)
EUR	10.05 (2.23)	10.52 (2.56)
JPY	9.92 (5.22)	10.46 (5.91)
CHF	14.01 (6.38)	14.80 (7.11)
RUB	16.57 (1.27)	17.63 (1.78)
ZAR	18.77 (1.66)	20.11 (4.38)

Table 3.3: In-sample MAE and RMSE of GA (mean error values out of the 100 rounds) and RSGA (in brackets) of the one-day ahead conditional variance forecast of the six currency pairs. All currencies are against USD.

	MAE ($\times 10^{-4}$)	RMSE ($\times 10^{-4}$)
GBP	12.29 (3.29)	12.96 (4.15)
EUR	10.07 (2.26)	10.46 (2.50)
JPY	9.92 (5.20)	10.39 (5.88)
CHF	14.06 (8.28)	15.30 (8.28)
RUB	16.87 (2.42)	21.37 (12.49)
ZAR	18.78 (1.59)	20.03 (3.13)

Table 3.4: Out-of-sample MAE and RMSE of GA (mean error values out of the 100 rounds) and RSGA (in brackets) of the one-day ahead conditional variance forecast of the six currency pairs. All currencies are against USD.

[62]This percentage improvement is calculated such that it lies between 0 and 100. A 0% improvement suggests there is no improvement of RSGA over GA and a 100% improvement suggests the RSGA reduces errors to zero (practically impossible to happen but it is always the ultimate target).

	Improvement (MAE)	Improvement (RMSE)
GBP	73.3% (73.2%)	73.0% (67.9%)
EUR	77.8.4% (77.6%)	75.6% (76.1%)
JPY	47.3% (47.6%)	43.5% (43.4%)
CHF	54.4% (53.4%)	52.0% (45.9%)
RUB	92.3% (85.6%)	89.9% (41.6%)
ZAR	91.2% (91.6%)	78.2% (84.4%)
Average	74.6% (74.5%)	74.4% (70.2%)

Table 3.5: In-sample and out-of-sample (in brackets) MAE and RMSE improvement of RSGA over GA (a weaker performance will be represented by a negative percentage change which never happens in this case). All currencies are against USD.

Overall the percentage improvement of RSGA over GA is significant although the extent of improvement varies from currency pair to currency pair. The improvement is more significant for the more volatile currency pairs such as RUB and ZAR and less significant for the currency pairs with lower volatilities such as JPY and CHF. Due to its instability in performance, GA can perform very well sometimes and very badly other times. This leads to another aspect where a comparison can be made between GA and RSGA, which is to compute the percentage of the 100 rounds in which the GA is outperformed by RSGA.

To make the comparison, three cases are split up: (1) when the error from a GA is larger than the error from a RSGA by over 10% it is classified as a "RSGA better" case; (2) when the error from a GA is smaller than the error from a RSGA by over 10% it is classified as a "GA better" case; (3) when the difference between the errors is between -10% and 10% it is classified as an "Indifference" case. Table 3.6 and Table 3.7 show the performance comparison with respect to MAE and RMSE. While GA does show some

good performance in the modelling process of JPY and CHF, overall speaking the RSGA outperforms GA for the majority of time and for the two more volatile currency pairs RUB and ZAR, the RSGA has a dominant performance over GA.

	RSGA better	GA better	Indifference
GBP	93% (93%)	6% (6%)	1% (1%)
EUR	94% (92%)	3% (6%)	3% (2%)
JPY	74% (75%)	21% (21%)	5% (4%)
CHF	79% (78%)	13% (14%)	8% (8%)
RUB	99% (100%)	0% (0%)	1% (0%)
ZAR	100% (100%)	0% (0%)	0% (0%)
Average	89.8% (89.7%)	7.2% (7.8%)	3.0% (2.5%)

Table 3.6: In-sample and out-of-sample (in brackets) MAE comparison of RSGA with GA. All currencies are against USD.

	RSGA better	GA better	Indifference
GBP	94% (92%)	4% (6%)	2% (2%)
EUR	96% (95%)	0% (3%)	4% (2%)
JPY	71% (70%)	21% (21%)	8% (9%)
CHF	80% (77%)	14% (14%)	6% (9%)
RUB	100% (72%)	0% (9%)	0% (19%)
ZAR	100% (100%)	0% (0%)	0% (0%)
Average	90.2% (84.4%)	6.5% (8.8%)	3.3% (6.8%)

Table 3.7: In-sample and out-of-sample (in brackets) RMSE comparison of RSGA with GA. All currencies are against USD.

3.5.4 Comparison with gradient-based algorithms

In this subsection, comparisons are made between the RSGA and two previously-mentioned "gradient-based" algorithms in Subsection 3.1.4, namely BFGS and NM.[63] The motivation for this is to substantiate the use of the GA (hence also RSGA) over "gradient-based" algorithms in this chapter. Tables 3.8 and 3.9 show that RSGA generates significantly smaller errors than the BFGS, NM and GA. Across the six currency pairs, all four algorithms generate more disappointing results for CHF than for other currency pairs. This indicates that CHF may be the most challenging currency pair to forecast in the time period of interest. This is consistent with Kamruzzaman and Sarker (2004) and Bakhach et al. (2016). In both papers, the CHF pairs (not only against USD but also against other currencies) turn to be one of the most difficult to accurately forecast.

While BFGS, NM and GA all suffer in forecasting volatile pairs such as RUB and ZAR, RSGA is able to produce far superior performance results than the other three algorithms. This shows that RSGA can forecast different types of datasets (regardless of whether they are volatile or less-volatile) with satisfying performance. The main reason for this improvement is that through the process of repetitive simulation, RSGA has a higher chance to move the estimation further away from local optimum points where other algorithms can easily become trapped.

BFGS and NM, being efficient algorithms in terms of computation time, suffer greatly in this task of estimating a relatively complex model (GARCH-ANN) for forecasting one of the most unforecastable financial assets (FX rates). The GA (and RSGA), however, with the ability to adjust the size and members of its initial parent population, is able to generate significantly more satisfactory performance results. One should be aware that the contribution of this chapter rests in the improvement of RSGA over GA. The

[63] BFGS and NM are two "gradient-based" algorithms that are fast to implement but may sometimes suffer from the "indifferentiability" problem.

disappointing performance of BFGS and NM compared with GA is anticipated and the inclusion of their performance in this subsection is for the purposes of completeness and robustness checking.

It is also expected that the extra capability of adjustment from the GA and RSGA also significantly increases computation time. Table 3.10 displays the average computation time (in seconds) per run out of 100 runs for BFGS, NM and GA, together with the computation time used by RSGA, rounded over the six currency pairs. Unlike the other three algorithms for which 100 runs are computed to see the average time (in reality only one run is computed to get the result), the RSGA runs only once. The GA algorithm utilises much longer computation time per run than BFGS and NM and the RSGA takes even longer. However, with more advanced hardware, including using upgraded CPU (or even GPU for parallelising complex tasks)[64] and potentially more efficient programmes (i.e. redesigning functions within RSGA so that it is more computationally efficient), this extent of extra computational cost could be significantly reduced.

	BFGS ($\times 10^{-4}$)	NM ($\times 10^{-4}$)	GA ($\times 10^{-4}$)	RSGA ($\times 10^{-4}$)
GBP	510.22 (510.87)	94.45 (99.57)	12.22 (12.81)	3.26 (3.46)
EUR	689.65 (699.91)	113.10 (117.56)	10.05 (10.52)	2.23 (2.56)
JPY	845.72 (858.24)	70.26 (74.24)	9.92 (10.46)	5.22 (5.91)
CHF	1216.38 (1228.78)	196.24 (217.28)	14.01 (14.80)	6.38 (7.11)
RUB	954.45 (970.72)	240.37 (273.11)	16.57 (17.63)	1.27 (1.78)
ZAR	1114.28 (1137.48)	130.10 (143.87)	18.77 (20.11)	1.66 (4.38)

Table 3.8: In-sample MAE and RMSE (in brackets) comparison of RSGA with GA and gradient-based algorithms. All currencies are against USD.

[64]GPU computation is one of the main topics of Chapter 4.

	BFGS ($\times 10^{-4}$)	NM ($\times 10^{-4}$)	GA ($\times 10^{-4}$)	RSGA ($\times 10^{-4}$)
GBP	510.59 (522.69)	94.78 (100.30)	12.29 (12.96)	3.2 (4.15)
EUR	691.35 (703.62)	113.36 (119.27)	10.07 (10.46)	2.26 (2.50)
JPY	846.72 (860.97)	70.68 (75.10)	9.92 (10.39)	5.20 (5.88)
CHF	1217.17 (1231.59)	196.31 (217.30)	14.06 (15.30)	8.28 (8.28)
RUB	960.63 (1004.73)	255.53 (319.42)	16.87 (21.37)	2.42 (12.49)
ZAR	1115.07 (1138.72)	130.20 (144.62)	18.78 (20.03)	1.59 (3.13)

Table 3.9: Out-of-sample MAE and RMSE (in brackets) comparison of RSGA with GA and gradient-based algorithms. All currencies are against USD.

	BFGS	NM	GA	RSGA
Time (s)	4.14	1.52	233.27	4193.32

Table 3.10: Average computation time (in seconds) per run out of 100 runs for BFGS, NM and GA, together with the computation time used by RSGA. All times are averaged through six currency pairs. All currencies are against USD.

3.5.5 Implications for EMH

From Table 3.2, it can be seen that the emerging market currency pairs USD/RUB and USD/ZAR have significantly higher volatilities than the four developed market currency pairs. Therefore it would be natural to presume that in the tasks of forecasting volatility, the forecast errors of USD/RUB and USD/ZAR should be larger (in value, not necessarily in percentage) than the other pairs. However, based on the results from Tables 3.8 and 3.9, USD/RUB and USD/ZAR actually have smaller errors (with the RSGA algorithm) than some of the developed market pairs.

This empirical finding indicates that the emerging market currency pairs are easier to forecast than the developed market pairs (in terms of volatility forecasting). In other words, the developed markets are more efficient than the emerging markets. This is in line with what the terms "developed" and "emerging" suggest. According to the definitions of three forms of EMH by Fama (1970) in Subsection 2.1.2, the term "efficient" in this paragraph refers to the weak form efficiency. This is because all information used to forecast volatility in this chapter is historical FX rates.

Another finding comes from the comparison of forecasting results for the in-sample data points with the forecasting results for the out-of-sample data points.[65] For a specific time series, it is common to see out-of-sample errors greater than in-sample errors. This is because parameters of the model have been tuned based on the in-sample data points in the training process while the model has never seen any data points from the out-of-sample dataset before.

Based on the results from Tables 3.8 and 3.9, only the in-sample error of USD/CHF is smaller than its out-of-sample error. For the other pairs, at least one of the metrics

[65] At this point, only the errors generated by RSGA are considered because the concern is to see how consistent the performances of the best possible algorithm are.

(RMSE or MAE) shows the opposite, i.e. in-sample error larger than out-of-sample error. USD/JPY is the only pair that the in-sample error is larger than the out-of-sample error for both metrics. For those pairs whose in-sample error is larger than out-of-sample error, it is a possible explanation that the out-of-sample period is easier to forecast than the in-sample period, i.e. the underlying market is less (weak form) efficient during the out-of-sample period than the in-sample period. Therefore, this chapter provides evidence that the Japanese market has gone through some extent of efficiency decrease since the start of the out-of-sample period (mid 2014). The Swiss market shows consistency in the extent of difficulty in forecasting during different periods and the high extent of (weak form) market efficiency (suggested by relatively large forecasting errors even with RSGA) also makes it arguable the most difficult pair to forecast in the given period, out of the six pairs.

3.6 Conclusion

The FX market, one of the most important financial markets in the world, has been extensively researched and a significant number of papers focus on modelling the volatility of FX rates. Most of these papers either propose new models or improve existing models. Very few of them concentrate on parameter estimation in the modelling process, where problems such as the convergence problem, the local-maximum problem and the instability problem occur frequently.

In this chapter, the RSGA is proposed for parameter estimation in order to help deal with the convergence problem, the local-maximum problem and improve the stability of the estimation process. The RSGA is applied to estimate coefficients of a GARCH-ANN model for forecasting volatility of, first of all, 100 simulated time series. Simulated series are considered first because they provide a large replicable sample for performance evaluation. Comparisons are made between the traditional GA and the proposed RSGA based on accuracy, convergence rate and computation timing. With a reasonable extent of extra computation time, the RSGA is able to provide higher estimation accuracy while dealing with the convergence problem.

The algorithm is also utilised to model the volatility of six FX pairs: GBP, EUR, JPY, CHF, RUB and ZAR, all against USD. Among the six pairs, RUB and ZAR (from emerging economies) are more volatile than the other four pairs (from developed economies). While GA as expected provides smaller errors for the less-volatile currency pairs than volatile pairs, RSGA sometimes generates even smaller errors for volatile pairs than less-volatile pairs. This is confirmed by a significant extent of percentage improvement. The RSGA algorithm generates higher percentage improvement for high-volatile pairs, e.g. RUB and ZAR, than for less-volatile pairs. This shows potential superiority of RSGA in modelling high-volatility emerging market pairs, which can be a desirable advantage because high-volatility emerging market currency pairs are sometimes more

profitable due to lower level of speculative activities. In terms of EMH, this can also be explained by the relatively lower (weak form) efficiency of the emerging markets than developed markets.

Another finding is that, despite its relatively low volatility, the performance of CHF volatility forecasting remains one of the poorest across all algorithms. This indicates the low forecastability of CHF during the period of interest. The consistent difficulty in forecasting the CHF also marks the Swiss market a relatively informationally efficient one.

This agrees with the results of Chapter 4 where the ANN models for CHF generate the lowest direction-of-price-movement accuracy rates and annualised returns. Overall, the RSGA algorithm provides satisfactory results in modelling volatility of these currency pairs with smaller errors overall compared with a GA algorithm (and other two "gradient-based" algorithms).[66]

In this chapter, the RSGA algorithm is applied in order to estimate the GARCH-ANN model, i.e. find the parameters that ideally maximise the likelihood function associated with the model. However, the usefulness of the algorithm in improving estimation results is not limited to this particular likelihood function. Therefore, one advantage of the algorithm is that it can be easily migrated to estimate other models.[67]

The RSGA increases estimation accuracy and deals with the convergence problem, at the cost of computation time. Considering the relatively small datasets and even smaller simulation datasets used in this chapter, computation speed-up is a challenging yet

[66] As is discussed in Subsection 3.5.4, even the GA provides much more accurate forecasting results than "gradient-based" algorithms such as BFGS and NM.

[67] The extent of improvement over traditional algorithms will increase as the complexity of the target model increases.

rewarding task for further implementation. In particular, it is of great benefit to speed up the computation such that the RSGA algorithm can be applied to estimate even more complex models and with much larger datasets, in a time-efficient manner. Three ways that can be employed to improve computation efficiency (listed in the order of increasing efficiency improvement) include programming with a faster language such as Python or C++, upgrading the CPU, and utilising Graphic Processing Units (GPU), given the fast development in GPU over the past few years.[68]

Another aspect that may draw further attention is that in this research the errors ϵ_t of all models are assumed to follow a normal distribution, which is by no means a perfect mimic considering the nature of the FX data. This problem will always exist as long as the coefficients are estimated with a likelihood-based algorithm which involves making an assumption on the underlying distribution. One way to deal with this problem is by using utility/cost functions other than likelihood. With non-likelihood utility/cost functions, it avoids the necessity of assuming any inappropriate distributions. For modelling with exchange rates or other financial assets, one example of such a utility/cost function would be the trading profits with a given trading strategy. Related research on this includes Alvarez-Diaz and Alvarez (2003), Hirabayashi et al. (2009), and Abreu et al. (2018). Other types of algorithms such as Particle Swarm Optimisation (PSO) also utilise non-likelihood utility/cost functions therefore can also act as a potential alternative, see Eberhart & Kennedy (1995) and Sermpinis et al. (2013). Overall, there is a rich agenda for other researchers to build on the work of this chapter.

[68] GPU computing is one of the main topics in Chapter 4.

4 Using GPU with large neural networks for forecasting directions of foreign exchange rate movements

4.1 Introduction

4.1.1 The role and approaches of forecasting

Forecasting is one of the most challenging yet important tasks in financial markets. Volatility forecasting and return forecasting are two of the main directions of research in the area of forecasting. Volatility forecasting, the main topic discussed in Chapter 3, plays a key role in risk management, portfolio management, and derivative pricing. Research in these areas includes Christoffersen and Diebold (2000), Brooks and Persand (2003), Chen et al. (2017), Kongslip and Mateus (2017), Bollerslev et al. (2018). Since volatility forecasting is not the main focus of this chapter, the mentioned literature is not discussed in detail. Details of general literature in the area of volatility forecasting can be found in Chapter 2.

The main focus of this chapter is return forecasting which contributes to success in hedging and speculation. Three types of frequently-used tools are time series models (the most popular being Box-Jenkins models, such as the Auto Regressive Integrated Moving Average Model, or ARIMA), neural network models and modified or hybrid models built based on both types of models.

Time series models have been developed and applied to the area of finance for nearly four decades. Wide-ranging research focuses on forecasting in financial markets with time series models. In recent examples, Tripathy (2017) uses an ARIMA (0,1,1) model to forecast gold prices and Alkhazaleh (2018) applies an ARIMA (2,0,2) for forecasting stock returns. Garikai et al. (2019) forecast foreign direct investment with an ARIMA (0,1,2) model.

The problem with the time series approach, however, is that the number of variables for the models is very small (rarely more than five) and normally only lagged values are utilised as inputs of the model. In other words, future time series values are forecast with only a few values in the past from the same time series. The time series to be forecast may indeed contain some important information (internal or external) but many other external factors which might significantly affect future values of the time series are not absorbed into the model. In fact, with the time series approach they are not even considered. Therefore, potential over-simplification of the real situation arises, which can lead to poor forecast of future values of the time series. Artificial Neural Networks (ANN) is an alternative approach which takes into account as many variables as the researchers choose and therefore has the capability of lowering the extent of over-simplification.

ANN models, despite having been studied much earlier, have been implemented for financial modelling for only about two decades. This is mainly due to high complexity and low predictability of the financial markets, i.e. forecasting the movement of a given stock price is far more challenging than forecasting the weather or the electricity usage of the whole country with a neural network model. Several research papers have demonstrated the superiority of ANN models over time series models in forecasting both financial and non-financial series. Details of these papers will be discussed in the Literature Review section of this chapter. Another reason for using ANN in this chapter is the fast development of Graphics Processing Units (GPU) in recent years. GPU, with the help of parallel processing, greatly increases computation speed and enables the implementation of large neural networks (more informative and therefore more predictive) within a reasonable time frame.

A common question might arise on the use of predictive models for forecasting future prices of financial assets based on past prices and values of financial assets and economic indicators. In the case of a major political or economic event, how possibly can a future

price be forecast based on the above-mentioned past values? A short answer to this question is that it cannot. However, to the author's best knowledge none of the predictive models aims at modelling even the direction of price movement (less challenging than modelling the actual prices) at 100% accuracy, nor is it possible to do so. In fact using a model with just above 50% accuracy can be sufficient in making profits, as long as a reasonable stop-loss [69] level is set to make sure losses are under control when a false prediction occurs.

One example is the British pound depreciation after the EU-membership referendum of the UK in 2016. The final voting result could not be forecast based solely on past prices of financial assets and values of economic indicators. Without a large and reliable public opinion poll it can be very difficult to assess what the result might be. In fact even after a public opinion poll, many agencies, companies and casinos thought it was going to be a non-exit result, which turned out to be wrong. A similar story happened later in the same year when Donald Trump was elected as the president of the USA, contrary to many experts' predictions. In both cases the actual results may cause potential financial losses due to high level of unpredictability of the outcome of the political events. These potential losses can be controlled with a reasonable stop-loss level or used as an opportunity to invest in the derivative market, i.e. options which benefit from high volatilities in financial asset prices. Potential losses can be completely avoided by choosing not to involve in investing in these circumstances where a nearly-unpredictable significant event is due to happen.

As a result of the above arguments, the key usage of these predictive models is to forecast buying and selling pressure based on past financial and economic conditions, with an accuracy rate of over 50% (preferably over 55% or higher), to benefit as much as possible

[69] A stop-loss is a pre-set level when reached in the case of false prediction of the direction of price movement, the transaction is terminated to prevent further losses.

from the correct predictions and control losses with a reasonable stop-loss level for the false predictions. According to the Law of Large Numbers,[70] as the number of transactions becomes large the actual accuracy rate tends to get closer to the target rate (the accuracy rate obtained by the selected best model) and with losses controlled by a good stop-loss level, significant opportunities in the economic sense can potentially be identified in the FX, stock or commodity markets.

4.1.2 The foreign exchange market as a forecasting context

The foreign exchange (FX) market is the largest and most heavily traded financial market. According to the Bank of International Settlements (BIS) 2019 survey, the average daily trading volume increased by 30%, from 5.1 trillion US dollars in 2016 to 6.6 trillion US dollars in 2019. This daily trading volume is 24 times the size of the daily trading volume of the global stock markets, [71] making the FX market arguably the most important financial market in the world. [72]

Figure 4.1 from the BIS Triennial Central Bank Survey (2019) suggests that the US dollar remains the dominant currency, being on one side of 88.3% of all trades in April 2019, an increase by 0.7% since April 2016. In terms of geographical distribution of FX turnover, trading continues to be concentrated in a few large financial centres. In April 2019, 79% (increased from 77% in 2016) of all FX trading took place in five countries - the UK, the US, Singapore, Hong Kong SAR and Japan.

[70] Law of Large Numbers: the average of the results obtained from a large number of trials should be close to the expected value, and will tend to become closer as more trials are performed.

[71] According to the World Bank database (`https://data.worldbank.org/indicator/CM.MKT.TRAD.CD?end=2018&start=1975&view=chart`), the total trading volume of stocks in 2018 globally was 68.212 trillion US dollars, making the daily traded volume $68.212/253 = 0.27$ trillion US dollars.

[72] For more general detail on the FX market refer back to Subsection 2.1.1.

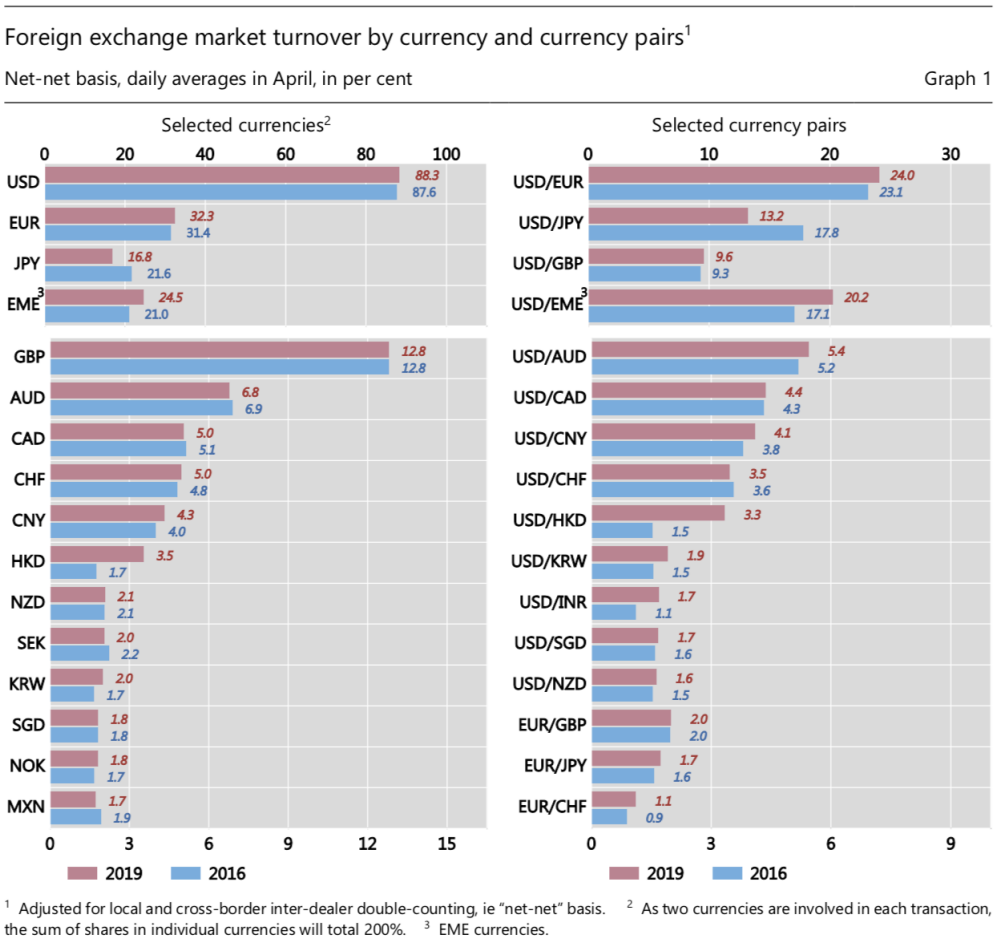


Figure 4.1: Foreign exchange market turnover by currency and currency pairs in April 2019 and 2016. Source: BIS Triennial Central Bank Survey (2019, 2016).

In addition to its size and significance, the FX market is also one of the most challenging financial assets to be researched due to the high complexity of its price change patterns, the high level of speculation activities (reducing the level of any potential trading profits) and a large number of influential factors, including both macro and micro indicators. Two examples are described below.

After two consecutive cuts in deposit facility interest rates in the Euro Zone in June and September 2014, from 0% to -0.1% and then -0.2%, the EUR/USD exchange rate dropped by a maximum of 21% in the 8 months following September 2014. However, after a further cut in deposit facility interest rates to -0.4% in March 2016, the Euro appreciated almost 5% against USD in just one month. These two similar moves (cutting interest rates) of the European Central Bank (ECB) caused completely opposite direction of movement of the exchange rate. One reason for this contradiction could be that investors held different expectations about the ECB's movement before the actual cut took place, resulting in different outcomes after the cut. Another possible reason is that there were other conditions happening that affected the strength of the currencies. These conditions could be affecting only the Eurozone, affecting only the USA or affecting both areas. For this reason, including only the interest rates and even only a few more indicators might not be enough for a good prediction of future FX rates. More factors need to be taken into account when building a model.

The UK currency GBP, depreciated against USD by a maximum of 30% in the following several years since the 2008 financial crisis and depreciated over 17% against USD in just 4 months during the period immediately after the UK referendum on EU membership in June 2016. Forecasting price movements immediately after a major event is a challenging task. However, once the immediate effect is gone and weeks or even months of aftershocks arrive, it is time for predictive models to shine, after taking into account massive information including financial and economic conditions and market reactions.

As Brexit proceeded, GBP fluctuated up and down as volatile as an emerging market currency, leaving tremendous opportunities of making profits for investors. With a good stop-loss level controlling risks, the correct predictions would generate much more profits than during those less volatile periods.

Other researchers have also demonstrated that profits of economic significance can be made within the FX market via different approaches, see Lee and Loh (2002), Qi and Wu (2006), Gradojevic (2007), Corte et al (2016), and Hsu et al. (2016). These approaches include the currency volatility premium approach, ANN and hybrid ANN models, the Genetic Algorithm approach and the technical trading methodology.

4.1.3 Research questions, challenges and main focus

As is discussed in Section 4.1.1, researchers can select as many variables (known as input neurons) as they desire for an ANN model. Therefore, they can base their forecast on more information about the market - the very challenging and complicated FX market. Another advantage of an ANN is that apart from input and output neurons, it also has one or more in-between levels of variables called hidden layers. The addition of the hidden layers (originally designed to mimic the way human brain works) greatly increases the maximum complexity the model can reach hence significantly boosts the interpretability of the model. However, as the number of input and hidden neurons increases, the total number of coefficients to be estimated increases at a much faster speed. This will cause the slow computation problem.

The research questions of this chapter are (i) whether an increase in the number of neurons in the input and hidden layers will improve forecasting performance and the implications of the forecasting performance for the EMH, and (ii) by how much computation efficiency will be improved with the help of GPU as the number of neurons increases and potential problems arise with the GPU technique in the training process. There are three focuses

within these research questions. The first focus is the prediction of financial asset prices, typically FX rates. The second focus is a discussion and development on the prediction models - ANN with a large number of input and hidden neurons. The third focus considers computation power with the employment of GPU for the purpose of efficiency enhancement and computation time reduction.

To achieve the target there are several challenges to overcome. The first challenge is that a financial asset has far more features (potential input variables) than many other areas that ANN is frequently applied in (such as speech or photo recognition) therefore selecting the input variables is a challenging task. Secondly, a challenge arises because the GPU works in a way such that it utilises a different language from the underlying programming language (R in this research) and requires special communication with the computer's main system. Several programmes including Anaconda, CUDA, Cudnn, all in the right versions without being incompatible with each other, need to be installed and maintained routinely to make sure there are no version conflicts due to version update. The third challenge lies in the programming process whereby in addition to programming in the main language (R), in order to facilitate GPU and utilise the open-source library (Keras[73]), a new language (created specifically for Keras based on Python) needs to be fluently mastered.

Readers should be aware that the main focus of this research is to explore a newly developed area (GPU computing) and facilitate its use for predicting FX rates with large ANN models. In terms of published research, this has not been done as widely in the financial market as in other areas. In addition, programmes and procedures will be designed and enhanced such that they complete the heavy computation tasks within a reasonable time. Performance evaluation metrics such as forecasting accuracy and

[73]Keras is an open-source library aiming at improving training efficiency in ANN models by using more efficient codes and allowing for the use of GPU.

annualised returns are used as key factors to evaluate the model performance and build up a reasonably profitable model but maximising these metrics to a highly profitable level is not the main target of this research.

To briefly preview the chapter's findings, the key outcomes of this research include: (1) For the four currency pairs researched (GBP/USD, EUR/USD, USD/JPY, USD/CHF), the MLPNN models with 2^8 up to 2^{10} neurons are able to produce significantly better out-of-sample forecasting results in terms of accuracy (a maximum accuracy of 62.5%) and annualised return (a maximum annualised return of 6.9%) than traditional MLPNN models built for FX modelling with 5-10 neurons and the benchmark logistic regression model. The significant out-of-sample abnormal returns provide evidence against the semi-strong EMH. Out of the four markets, the Swiss market exhibits the least extent of (semi-strong form) inefficiency; (2) Computation with GPU saves computation time by over 23% on average and can save up to nearly 33% when the GPU memory is not fully occupied; (3) GPU computation time increases dramatically (faster than exponential increment) after its inbuilt memory is full but still saves more than 20% time compared with CPU computation even when the GPU is at its worst performance.

The remainder of the chapter is organised as follows. Section 4.2 includes contextual literature on ANN, the FX market, and ANN applied in the FX market. Section 4.3 describes the methodology (and related empirical literature) applied in this research including the MLPNN[74] model used for classification problems, the Keras library and GPU computing. In Section 4.4, data used in this research are described in terms of their source, structure and variable selection. Section 4.5 contains the main modelling process presented in the order of the process (parameter tuning, training, testing). The final section 4.6 concludes the chapter and suggests potential future research directions.

[74]MLPNN refers to the Multilayer Perceptron Neural Network. It is a type of ANN that contains multiple layers (input, output and hidden) of neurons.

4.2 Literature review

A few frequently mentioned performance metrics are described and defined below. All of these metrics measure the deviation of the predicted value from the actual value, and they are useful in evaluating how well a model performs, regardless of whether it's a time series model or an ANN model.

Sum Squared Error (SSE): the sum of the squared errors.
Mean Squared Error (MSE): the average of the squared errors.
Root Mean Squared Error (RMSE): the square root of the average of the squared errors.
Mean Absolute Error (MAE): the average of the absolute values of the errors.
Mean Absolute Percentage Error (MAPE): the average of the absolute percentage errors.

$$SSE = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2, \tag{4.1}$$

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2, \tag{4.2}$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2}, \tag{4.3}$$

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |\hat{y}_i - y_i|, \tag{4.4}$$

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{\hat{y}_i - y_i}{y_i} \right|, \tag{4.5}$$

where $\hat{y}_i$ is the predicted value of the ith observation and y_i is the actual value of the ith prediction. n is the total number of observations to be predicted.

4.2.1 ANN models

ANN in financial forecasting

As a recent overview on the research topic of machine learning, Emerson et al. (2019) provide a summary of the literature on the machine learning technique in the area of quantitative finance. According to their research, return forecasting is the most frequently researched theme of machine learning, followed by portfolio construction, ethics, fraud detection, decision-making, language processing, and sentiment analysis. In terms of machine learning techniques, MLPNN is the most frequently utilised and researched technique, followed by Support Vector Machine (SVM), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), Random Forests/Decision Trees (RF/DT), Gaussian Process Regression (GPR) and Logistic Regression (LR). Both the most frequently researched theme (return forecasting) and the most popular technique (MLPNN) are pursued within the research direction and technique of this chapter. [75]

In addition to the widely used ANN models mentioned in the previous paragraph, researchers have also implemented modified versions of these models in order to improve forecasting performance. Cao et al. (2019) combine a CEEMDAN (complete ensemble empirical mode decomposition with adaptive noise) algorithm with a LSTM neural network. An EMD (empirical mode decomposition) is an adaptive signal time-frequency processing method that decomposes the series into a finite number of intrinsic mode functions. A CEEMDAN is a modified version of EMD that aims at reducing reconstruction errors of an EMD. The hybrid CEEMDAN-LSTM model is compared against the benchmark LSTM model. Empirical results show that their proposed model gives better forecasting results on 4 stock indices (S&P 500, HSI, DAX and SSE) in terms of MAE, MAPE and RMSE.

[75] For more general details on ANN (MLPNN) refer back to Subsection 2.2.2.

Two other recent research papers also focus on forecasting stock indices. The first is Tran et al. (2018), who forecast stock prices with a temporal attention augmented bilinear network. The layer structure they propose includes the formulation of an attention mechanism, which encourages the competition between neurons representing the same feature at different time instances. Based on forecasting results of 5 different stocks in NASDAQ Nordic coming from different industrial sectors, they show the effectiveness of their proposed model over SVM, CNN and LSTM in terms of forecast accuracy, even with only two hidden layers. Their research suggests that the ANN can work well even with very few hidden layers. This result helps in deciding to use a small number of hidden layers in this research in order to save computation time.

The second recent paper aiming at forecasting stock indices is Mo and Wang (2018). They propose a stochastic time strength neural network (STSNN) model to forecast stock indices from China and USA. They utilise a stochastic time strength function, giving a weight to each of the data points, to make the model exhibit some level of randomness. The empirical results show that the STSNN model outperforms the benchmark back propagation neural network (BPNN) in terms of RMSE, MAE and MAPE.

Apart from the "model-design" research direction (building up different types of ANN models), another direction is the "parameter-optimisation" (looking for the optimal number of parameters to be used in a given model). Kaastra and Boyd (1996) discuss that the large number of parameters to be estimated in a neural network means that the modelling process involves much trial and error and the trade-off between estimation performance and computation time exists as in many other types of models. A few other papers overlook the power of a large ANN with many neurons by pointing out the overfitting problem. While this paper puts forward the computation time problem that a large neural network generally faces, it confirms the extra performance improvement a large ANN can bring. This helps in forming the motivation of exploring large ANN in

this chapter.

A more precise research on the optimal number of input and hidden neurons is Zhang and Hu (1998). They examine the effects of the number of input and hidden neurons in an ANN for the prediction of the GBP/USD currency pair. With RMSE, MAE, and MAPE as the performance measures, they show that the ANN becomes more powerful in modelling the data as the number of hidden nodes increases. As the number of input nodes increases, the performance from the test set enhances first and then worsens, suggesting that too many input nodes may reduce the ability of generalisation of the model. Similar to Kaastra and Boyd (1996), the conclusion of this research that an increase in the number of input and hidden neurons makes the ANN more powerful provides support and motivation for this chapter. On the other hand, Zhang and Hu (1998) only compare ANN with hidden neurons from 4 up to 20, possibly due to the limitation on computation power at the time of their research. It is now reasonable for researchers today to expand this range to test the performance of neural networks of much larger sizes.

Unlike all pre-mentioned papers which aim at forecasting the exact values of a financial asset, Chen et al. (2003) model and predict the direction of returns on the Taiwan Stock Exchange with the Probabilistic Neural Network (PNN) model. Comparison is made with the Generalized Methods of Moments (GMM) and the random walk model. Given a buy-and-hold trading strategy, the PNN model obtains higher returns than the other two models. One advantage of predicting the direction of return rather than the value of return is that a trading strategy can be easily implemented and therefore trading returns can be obtained as a practical performance metric, in addition to statistical metrics such as MAE, RMSE and MAPE.

Principal Component Analysis (PCA)[76] is a frequently used tool in machine learning

[76]PCA is a procedure that uses an orthogonal transformation to convert a number of (possibly) correlated

for the purpose of dimensionality reduction. It helps to reduce the number of variables without losing too much information, by absorbing some of the useful information into the transformed components. Wang and Wang (2015) apply PCA on the input data and then forecast several stock indices with a Stochastic Time Effective Function Neural Network (STNN). Comparison is made of the proposed model with a traditional back-propagation ANN model. The proposed model achieves better results than the benchmark model in terms of MAE, RMSE and MAPE.

The same authors (Wang and Wang (2016)) use a recurrent neural network for forecasting crude oil prices. The model they use, Elman Recurrent Neural Network (ERNN), is a time-varying predictive control system which has the ability to keep memory of recent information for future prediction. They use Complexity Invariant Distance (CID) instead of traditional metrics such as MAE, RMSE and MAPE as the performance measure. The proposed model outperforms traditional back-propagation ANN models in terms of CID. A similar paper by Wang et al. (2016) utilises the same model with the same performance measure CID. They apply their model on stock indices such as SSE, TWSE, KOSPI and Nikkei 225 and their model again produces better forecasting results than the benchmark model (back-propagation ANN) in terms of CID.

Repetitive training, discussed in detail in Chapter 3 and also to be implemented in this chapter, is an essential yet often omitted process in the training of ANN models (and many other models which involve parameter estimation). Without the repetitive training process, the training performance can be very unstable, due to the randomness in choosing initial values of the parameters to be estimated. Chen et al. (2017) propose a novel double-layer ANN for forecasting prices of 100 stocks with the largest capitalisation from the S&P 500. Their proposed model allows updating over time to achieve the best goodness-of-fit. They also train their model several times to increase the stability of

variables into a (smaller) number of uncorrelated variables called principal components.

the model. Their trained model produces significantly better results than the normal double-layer ANN, in terms of prediction accuracy and returns.

ANN compared with time series models

With the fast development of GPU computing in recent years, more attention has been drawn to ANN models. Compared with time series models, ANN models have greater interpretability of data due to their ability in adding a large number of input and hidden neurons.

Researchers have made performance comparisons of the ANN models and time series models within several circumstances. Kaytez et al. (2015) utilise both regression models and ANN models to predict electricity consumption in Turkey and ANN models significantly outperform regression models in terms of Maximum Error (ME), Mean Absolute Percentage Error (MAPE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Sum Squared Error (SSE) and Receiver Operating Characteristic (ROC) analysis.

Another application is also carried out on non-financial time series. Oshodi et al. (2017) compare the performance of Box-Jenkins models and ANN models in forecasting tender price index[77] of Hong Kong. Their empirical results suggest that the ANN outperforms the Box-Jenkins model in terms of MAPE.

Here are two examples of performance comparisons based on financial datasets. Cocianu and Grigoryan (2015) compare the performance of an ANN model against standard ARIMA models on predicting 300 weekly stock prices. The ANN model significantly outperforms standard ARIMA models in terms of MSE. Singh and Mishra (2015) model commodity prices with an ARIMA model and an ANN model. Criteria for comparing

[77]Tender price index is a measure that tracks movements in construction costs over time.

forecasting accuracy include MAE, MAPE, MSE and RMSE. They use monthly prices of groundnut oil in Mumbai from April 1994 to July 2010. The ANN model outperforms the ARIMA model significantly. They attribute the outperformance to the chaotic behaviour the data exhibits which cannot be fully captured by the linear ARIMA model.

Large ANN models

A typical ANN has an input layer, an output layer and one or more hidden layers, all of which contain nodes called neurons. While the number of neurons in the input and output layers will be fixed once the input variables and output variable(s) are determined, the number of neurons in the hidden layers is fully decided by the researcher's choice. Around one or two decades ago most researches use 3-10 neurons in the hidden layers, or 5-15 neurons in total for ANN models.

For example, Yao and Tan (2000) apply ANN models with the number of input neurons ranging from 9 to 11 to forecast foreign exchange rates. They show that useful prediction can be made without the use of extensive market data or knowledge. Dunis and Williams (2002) use an ANN with 9 input neurons to model and forecast foreign exchange rates and confirm its potential in forecasting EUR/USD returns for the period investigated. Guresen et al. (2011) model and forecast stock market indices with several ANN models (with 4-6 input neurons) including MLP and dynamic artificial neural network (DAN).

Some believe that a smaller number of neurons works better because they couldn't investigate that many neurons due to limitations on computation power. Especially during 2016-19, computation power has strengthened dramatically, in terms of both software (open-source libraries built specifically for ANN modelling) and hardware (high-end General Purpose Graphics Processing Units - GPU).

Large ANN models refer to those with a large number of neurons, typically over 100 neurons in each layer. These large ANN models have been built in a wide range of areas

such as text classification, image recognition, sound recognition and many others. Recent papers in the above-mentioned areas include Lai et al. (2015), Piczak (2017), Maggiori et al. (2017), Salamon and Bello (2017), Ju et al. (2018), Le et al. (2018), and Kum and Nam (2019). While large ANN models have been applied in a wide range of areas mentioned above, they have not been applied nearly as much in the finance area. Just as ANN was applied in finance almost three decades after it had been applied in biological and computational areas, there is a time lag (maybe not as long as three decades) for large ANN models to be applied in finance as well because it is very difficult to select a large pool of influential variables from an ocean of variables consisting of all kinds of economic, financial, political and market information.

4.2.2 The foreign exchange market

Since the asset class researched in this chapter is foreign exchange (FX) rates, some recent literature on the FX market forecasting context is reviewed and discussed below.

During periods when major events happen, jumps in the FX rates are very likely to occur. Piccotti (2018) discusses how the FX market changes before and after an exchange rate jump event happens. By conducting event studies, he finds that before an exchange rate jump event happens, quote volume, illiquidity, signed order flow, and informed trades are at higher levels and after an exchange rate jump event happens, quote volume and return variance remain at higher levels while illiquidity, informed trade, and signed order flow are at lower levels. The result reveals that jump events are consistent with rational dealer quoting behaviour. This research suggested that variables that contain information on volume, order flow, liquidity can be used for modelling price changes, especially during price jumps.

Cheung, Fatum and Yamamoto (2019)[78] focus on what news could have caused those

[78] A full list of authors is provided instead of Cheung et al. (2019) to differentiate from another Cheung

jumps. They conduct a study on whether the effects of macro news on FX rates are time and state- dependent. They compare the influence of US and Japanese macro news on USD/JPY quotes before, during and after the financial crisis from 1 August 1999 to 31 August 2016. Their results suggest that while the US news became more important than before the crisis, Japanese maco news became near irrelevant. While it's a great idea to forecast FX rates with news on its own by using structural models, it is practically difficult to include both news information and variable values in a single ANN model. This is because some news is too important to exclude but also too difficult to quantify and to be used as input neurons of an ANN model.

A few other recent papers focus on exploring the relationship between the FX rates and prices of other financial assets such as oil price and stock prices, and macroeconomic indicators such as risk-free interest rate and inflation. Utilising multivariate Markov switching vector autoregressive (MS-VAR) models, Roubaud and Arouri (2018) find significant non-linear interrelations between currency, oil and stock markets and interrelations are stronger in volatile periods.

Bai and Koong (2018) show that the US dollar index is negatively correlated with oil price shocks and positively correlated with stock price shocks, by using a diagonal BEKK (Baba, Engle, Kraft, and Kroner (1991)) model. With different models (VARs and GARCH), Mollick and Sakaki (2019) find similar results on the correlation between changes in the US dollar rates with oil price shocks and stock price shocks. In addition to also suggesting the negative relationship between exchange rates (based on USD) and oil prices, Yang et al. (2018), with the dynamic conditional correlation-mixed data sampling (DCC-MIDAS) model, present research on the relationship between exchange rates, risk-free interest rate and inflation. They find that the risk-free interest rate has a positive effect on FX rates while inflation is negatively correlated with FX rates.

---

paper in the same year with different co-authors.

In this context, the inflation rate (one of the most widely used measures of inflation is the Consumer Price Index - CPI) is very important. Engel et al. (2019) study the interest parity puzzle and FX rates forecasting. Their research results suggest that the US inflation variable is highly significant in forecasting changes in the US exchange rates (high inflation in one month forecasts appreciation in USD in the next month) while the interest rate differential is not significant in forecasting the US exchange rates. This research guides us on how macroeconomic indicators affect FX rates and equally importantly the effects are shown to exist on a monthly basis.

All of the above papers on the relationship between macroeconomic indicators and FX rates provide useful suggestions on what variables could be chosen as input neurons for an ANN in predicting FX rates. Examples of influential variables suggested by these papers include oil prices, stock prices, interest rates and inflation rates.

One of the important purposes of forecasting FX rates is trading. Cao et al. (2019) explore the FX rate predictability by using a generalised (term structure) model to capture dynamics between the risk premium component of FX rates and a wide range of variables. They also evaluate statistical and economic significance of the model and find that the model generates performance returns at 6.5% per annum.

Apart from forecasting the actual FX rates one can also forecast the direction of change of FX rates. By forecasting the direction of change it's easier to implement a simple Buy-Low-Sell-High trading strategy and thereby to use trading performance as a new evaluation metric for a wider assessment. In Cheung, Chin, Pascual and Zhang (2019), forecasting the direction of change is one of their research questions. They expand traditional exchange rate models e.g. Meese and Rogoff (1983) and Cheung et al. (2005) to include Taylor rule fundamentals, yield curve factors, and to incorporate shadow rates, risk and liquidity factors. They compare the performance of the proposed models with that of a random walk benchmark. The models are examined at different forecast

horizons (1 quarter, 4 quarters and 20 quarters) using several metrics (MSE and direction of change). They find that with MSE, no model consistently outperforms a random walk with statistical significance. However, with the direction of change metric, certain structural models do outperform a random walk with statistical significance. This research supports this chapter's decision to forecast the direction of change rather than the actual value/return value of FX rates because models should be implemented to forecast what they are best at modelling rather than what they are bad at forecasting.

4.2.3 ANN in forecasting foreign exchange rates

In the finance world, although ANN models have been introduced for over 15 years, the reported sizes of the ANN models remains small relative to those models used in the above-mentioned non-financial areas. This is true especially in the modelling and predicting of foreign exchange (FX) rates. ANN models are utilised for modelling FX and other financial assets as a result of their strong interpretation power. Designed as a mimic to the human brain, although the complexity of an ANN with current computing power is nowhere near that of a human brain, it is able to learn from a large dataset (tens of millions of observations) in days or even hours whereas it would take at least years, if not longer for a human to learn from a dataset of a similar size. Existing literature on modelling FX rates with ANN models is discussed here.

Among the earliest research on using an MLPNN to forecast FX rates is Yao and Tan (2000). They use a traditional rescaled range (R/S) analysis to test the "efficiency" of several currency markets, and they show that without the use of extensive market data or knowledge, useful prediction can be made for out-of-sample data with simple technical indicators. However, the data frequency they adopt is the weekly data which means the number of observations they have for both the training and test sets is small. The ANN they applied all have only three layers (one input layer, one hidden layer and one output layer) with an average of only 9 neurons in total in each model. To estimate datasets of

larger sizes and increase estimation accuracy in the longer term, a more complex model with more neurons is needed.

Unlike Yao and Tan (2000) who use technical indicators as the inputs of the neural networks, Dunis and Williams (2002) choose lagged values of different indices such as stock market price indices, 3-month interest rates, 10-year benchmark bond yields, Brent Crude oil price and gold bullion price. To model and forecast the currency pair EUR/USD, they build up a MLPNN with some of the above-mentioned indices as input neurons against benchmark models such as the Naive strategy, the Moving Average Convergence Divergence (MACD) strategy, ARMA and Logit models. The MLPNN produces the highest correct directional change forecast rate and also the highest annualised return. Unlike most other research where the main focus is on predicting the future values of FX they focus on forecasting the moving direction of future prices. The key methodology difference between these two directions is that a value-based research uses regression-type models while a direction-based research utilises classification-type models, in which the two types of models differ from each other mainly by the loss function. More details related to this issue will be discussed in the methodology section. Cheung, Chin, Pascual and Zhang (2019) support the use of direction-based models by finding that certain structural models outperform a random walk with statistical significance along a direction-of-change dimension.

In addition to modelling FX rates, FX volatilities are also modelled with ANN. Nag and Mitra (2002) build up a Genetic Algorithm Neural Network (GANN) to forecast GBP/USD and EUR/USD. Lagged returns are inserted into the model as input neurons. A significant improvement is achieved over traditional time series models (for volatility modelling) such as ARCH(1), GARCH(1,1), AGARCH(1,1) EGARCH(1,1) and GARCH(1,1)-M, according to MAPE, MSE, Max AE and R-SQ.

All of these ANN models applied in FX modelling adopt a relatively small number (fewer

than 20) of neurons compared to other areas. Smaller ANN models have the advantage of being easy to interpret, fast to train and simple to improve. However, since most would agree that the FX price patterns are no less trivial than a voice recognition problem or a photo classification problem, it is reasonable to think that modelling the FX prices with larger ANN models would produce better forecasting results. Another perspective from which we can see this topic is that an ANN essentially mimics the human brain in the learning process. Therefore a larger ANN represents a more powerful brain and stronger learning abilities.

Table 4.1 presents a summary of key papers including their year of publication, research contents and link with this chapter. Support from previous research papers on the use of large ANN models (or more specifically large MLPNN models) has been discussed, together with the advantage of using ANN to predict directions of price movement. In Section 4.3, the detailed methodology of MLPNN models for classification problems,[79] packages for computing with ANN, and GPU computing will be discussed.

[79]Predicting direction of price movement is a special case of a classification problem where only two classes exist - the price either goes up or down. The situation when no price change occurs is not considered because no transaction will be placed in this case anyway.

Name & Year	Research Content	Link
Emerson et al. (2019)	Significance of return forecasting and MLPNN	Provides guide on motivation and research direction
Tran et al. (2018)	Impact of fewer hidden layers	Supports the use of fewer hidden layers
Kaastra and Boyd (1996)	Large ANN improves forecasting	Supports the motivation of implementing large ANN
Zhang and Hu (1998)	Effects of large number of hidden neurons	Supports adding more hidden neurons
Chen et al. (2003)	Direction of return and trading returns	Supports forecasting direction of return
Yao and Tan (2000)	ANN with 5-10 neurons	Provides a gap of large ANN for this research
Dunis and Williams (2002)	ANN with 9 neurons	Provides a gap of large ANN for this research
Lai et al. (2015)	Large ANN for text classification	Provides a gap of large ANN for the financial market
Salamon and Bello (2017)	Large ANN for sound classification	Provides a gap of large ANN for the financial market
Roubaud and Arouri (2018)	Relationship between currency, oil and stock prices	Supports using of oil and stock prices as input neurons
Engel et al. (2019)	Exchange rate, interest rate and inflation rate	Supports using interest and inflation rate as input neurons

Table 4.1: A summary of key papers related to Chapter 4.

4.3 Methodology and related empirical literature

4.3.1 MLPNN for classification problems

An MLPNN model usually contains at least three layers: one input, one output and one or more hidden layers. All input neurons are computed such that a weighted sum of the neurons are passed into the next layer. This process goes on until the output layer is reached. During the process each weighted sum is transformed using a function called the activation function.[80] An MLPNN, as the name suggests, distinguishes itself from a linear perceptron by its multiple layers and non-linear activation functions.[81] Typical activation functions include the hyperbolic tangent function, the logistic function and the Rectifier Linear Unit (ReLU) function. These three types of activation functions are defined as below.

Hyperbolic tangent function:

$$f(x) = tanh(x), \tag{4.6}$$

Logistic function:

$$f(x) = (1 + e^{-x})^{-1}, \tag{4.7}$$

ReLU function:

$$f(x) = max(0, x). \tag{4.8}$$

Both the hyperbolic tangent and the logistic function are called a sigmoid function (a function with "S-shape"). Despite being smooth functions (differentiable everywhere), sigmoid activation functions have the advantage of less complex implementation. Chung et al. (2016) make a comparison between the performance of the ReLU and sigmoid functions in handwritten digit recognition. In their empirical research, the model with

[80] Activation function: a function that transforms the weighted sum (could be a very large or very small number) into a value within a smaller range, i.e. (0,1) with the logistic function.

[81] Refer to Chapter 2 for more details on MLPNN.

sigmoid functions predicts at an error rate of 7.93% while the model with ReLU functions predicts at an error rate of 1.99%. Their handwritten digit recognition task is essentially a classification problem. For example, if English is the underlying language then the number of classes is 26 (or 52 if upper and lower cases are considered differently) because there are 26 English letters. Therefore the ReLU function is used as the activation function for this chapter to predict direction of price movement, which is also a classification task.

For the prediction of the direction of FX rates movement with classification models, the y-variable has two levels (classes). It is equal to 1 if the price goes up in the next forecasting period and 0 if the price goes down or stays unchanged. For each prediction made the forecast class is either right or wrong. Therefore researchers often use the overall accuracy rate as the measure of performance and generally accuracy greater than 50% suggests potential profitability and accuracy exceeding 60% may very likely generate stable and significant profits in the forecasting period.[82]

To justify the decision of predicting direction (of price movement) rather than values of FX rates, another type of model called a regression model is introduced. A regression model, despite potentially being able to predict actual values of FX rates, encounters difficulty in measuring its trading performance. The most widely used performance metrics for regression models include Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE). These metrics are straightforward to calculate and have the ability to measure the overall performance to a certain extent. However, none of these types of metrics is able to differentiate the correctly-predicted-direction errors and wrongly-predicted-direction errors. To be more precise, a proper performance metric should work in a

[82]The trading strategy used is very simple: buy if the FX rate is predicted to increase and sell if the FX rate is predicted to decrease. More sophisticated trading strategies could be studied in the future but will not be focused on in this chapter.

way such that models generating similar values of the performance metric should be able to produce a similar level of potential profitability given a certain trading strategy. However, since all the above-mentioned metrics treat errors the same way (by summing up, squaring, averaging or taking square root), regardless of whether the direction of movement of the FX rate is predicted correctly. For example, if the direction of FX rates movement is predicted wrongly then a higher error means more losses while if the direction is predicted correctly then a higher error actually means more profits. Being incapable of considering the difference between these two situations limits the practical value of the regression-type models, especially from the trading perspective. For this reason, classification-type models will be considered in this research.

Different types of ANN have been widely applied in areas such as text classification e.g. Lai et al. (2015), Le et al. (2018), image classification, e.g. Maggiori et al. (2017), Ju et al. (2018) and voice/sound classification, e.g. Piczak (2017), Salamon and Bello (2017), Kum and Nam (2019). In the area of finance, the published research is much more limited. Examples include Tang et al. (2018), in which a neural network model is applied in credit classification analysis, i.e. whether a bank should approve a loan application from someone, given the information the bank collected from the person (for example employment, wage, age, education, and other factors). Sezer and Ozbayoglu (2018) apply a Convolution Neural Network (CNN) to image analysis (graphs of technical trading indicators) for the purpose of classifying trading decisions (Buy/Sell).

To determine which type of ANN to use, one aspect to consider is what information is used as inputs and what result is to be predicted. If the complexity of output is far more than that of the inputs then the model could be over-simplified and any good predicting result could be due to pure luck. This brings the limit of a Recurrent Neural Network (RNN) and a Convolution Neural Network (CNN) in the world of finance, which arguably generates the most complicated patterns of data. RNN typically works with time series

data and the information it contains as inputs are past values of the dataset so that nothing other than the time series itself is included. The CNN, although it is capable of analysing very complex graphs, the number of indicators on which the graphs are based is normally very small. In both types of models, the information used as inputs is too limited to produce a well-performing and long-lasting model. An MLPNN model, despite being longer established compared with some other neural networks, has the advantage of being able to include as many variables as the model builder can justify for inclusion. Therefore with an MLPNN model, much more information can be inserted as inputs of the model to increase its prediction power. The potential overfitting problem due to the inclusion of irrelevant variables is addressed in the modelling process in a later subsection.

4.3.2 An introduction to Keras for neural network modelling

As is discussed in the previous section, if the overfitting problem can be dealt with properly, then the more variables to include as inputs the better because one cannot be worse off by holding more information about the market. A reasonable question may arise: if the use of more input variables can generate potentially better models, why haven't previous researchers used a 30-variable MLPNN? Instead they typically only used 5-10 variables as inputs. One reason is that selecting a large number of possibly influential variables from a pool of a vast number of variables is a difficult and time-consuming task. Another reason is the limitation on computation power. The computation power of even just 3-5 years ago, is non-comparable to what has been achievable in the past two years. The term computation power refers to both software and hardware aspects. In this section, the software aspect is discussed and the hardware aspect will be focused on in the next section.

To understand why computation power matters so much in this kind of research problem,

a simple calculation is made on a 3-layer MLPNN in which the input layer has 5 variables (n_1), the output layer has two classes (n_3) and the only hidden layer has 3 neurons (n_2). Then even for this very simple MLPNN, the number of parameters to be estimated is:

$$n_1 \times n_2 + n_2 \times n_3 = 5 \times 3 + 3 \times 2 = 21 \tag{4.9}$$

The estimation process developed in the first chapter takes around 3 hours in R to estimate a model with just 6 parameters. Turning to a faster programming language such as C++ or Java could boost the computation speed. According to Aruoba and Fernandez-Villaverde (2014), C++ runs the fastest among Fortran, Java, Julia, Python, Matlab, Mathematica and R [83] in solving a stochastic maximisation problem, about 3.66 to 5.41 times faster than R and 1.24 to 1.64 times faster than Matlab. However, not many research papers use C++ as the programming language because it does not have inbuilt packages such as ANN, regression, and third-party libraries such as Tensorflow and Keras (described below), all of which are implemented in R and Python.

Open-source libraries accelerate the estimation process dramatically. Keras, an open-source neural network library, was initially released in 2015 and its stable version was released in late 2018. It is a very new tool designed to enable fast modelling with neural networks. Keras is based on a neural network modelling tool called Tensorflow (developed by Google) and can be implemented in several languages such as python and R.[84]

Because of the way Keras and Tensorflow are programmed the training time of large models is reduced significantly. This speed-up therefore enables us to work with large

[83] Matlab, Mathematica and R are technically not traditional programming languages as the others. They are software with many user-friendly mathematical/statistical packages. The term "programming language" is used because all these software have their own grammar of programming and compile at different speeds, just like the traditional programming languages.

[84] Keras-GPU in R: `https://tensorflow.rstudio.com/tools/local_gpu.html`

models which are impossible to be estimated within a reasonable time a few years ago. Choi et al. (2017) utilise the Keras library for audio and music signal preprocessing, with a 5-layer neural network (around 160,000 parameters to be estimated in total) and achieved an average training time of only around 450 seconds.

4.3.3 CPU vs GPU computation

The Keras library has two versions: a CPU version and a GPU one. The GPU version needs an NVIDIA GPU together with a parallel computing programme developed by NVIDIA for general computing on graphical processing units called Compute Unified Device Architecture (CUDA).

Computations done in most of the previous researches, without specifically pointing out, are conducted on computer CPUs. While the development of CPU has been constantly fast, the number of cores (reflecting the ability of conducting parallelisable tasks) has only increased from a typical single-core 20 years ago to a 32-core one as one of the top-end CPUs today. Considering the performance boost of each single core during the years, the improvement is a great achievement in terms of computation power for data researchers.

However, this core-number based boost is not even close to what the GPU has gone through. Up to today, a top-end GPU can easily have over 5000 cores. To see how this core boost makes the difference, imagine a 32-core CPU as a team of 32 professors and the 5000-core GPU as a team of 5000 high school students. Which team performs better depends on the type of task to be completed. If the task is to publish 100 academic papers then the team made of professors should excel because of the complexity of the task and non-parallelism of the task, meaning that all 100 papers are different in terms of level of complexity and also time to complete. However, if the task is to conduct

10000 addition and subtraction calculations within the range 1-100, then the team of high school students definitely has an upper end because the task is parallelisable.

A parallelisable task refers to one which can be split into parallel and simpler tasks. For estimating parameters of neural networks, the task is highly parallelisable because it simply involves trying a large number of values for the parameters to achieve a smaller loss function. While the difference between a CPU and GPU in completing a non-parallelisable or even lowly-parallelisable task is small, the difference could be very significant as the level of parallelisation goes up.

Several research papers make a comparison of CPU and GPU computation speeds in different circumstances. Li et al. (2015) utilise a CNN for face detection and conduct the training process on both a CPU and a GPU. The GPU completes the training process 7 times faster than the CPU in estimating their CNN. Han et al. (2016) make a comparison on the performance of a CPU and GPU on the same deep neural network and report a 14.5 times speed-up by the GPU over CPU. Coelho et al. (2017) examines the power of GPU computing on time series forecasting.Their GPU strategy appears to be scalable as the number of time series training rounds increases. An average of 15 times speed up is achieved using GPU and a maximum of 45 times speed up is observed with GPU as the number of time series training rounds increases. McNally et al. (2018) apply a Recurrent Neural Network (RNN) and a Long Short Term Memory (LSTM) to predict the moving direction of Bitcoin prices. The training process takes 67.7% less time on a GPU than on a CPU. The reason for this smaller improvement of GPU over CPU than Coelho et al. (2017) is that the RNN and LSTM estimated in this research are path-dependent therefore much less parallelisable than other types of neural network and time series models. For all of the above research, the GPU leads to a speed-up in the training process. However, the extent of the speed-up varies according to the usage of different CPU/GPU devices and different levels of parallelism of the tasks.

In Section 4.4, the data used for this research will be discussed in detail, including the source of data, how the data is re-structured to work for MLPNN and summary statistics of the data.

4.4 Data explanation

4.4.1 Data source

All data mentioned in this subsection are collected from Thomson Reuters Datastream. The initial response variables (y) include four monthly foreign exchange rates GBP/USD, EUR/USD, USD/JPY, USD/CHF from 1 January 1999 to 1 December 2018, a total of 240 observations per currency pair. Although the number of observations is not large, the time period covers less volatile periods as well as extremely volatile periods (for example financial crisis in 2008, European sovereign debt crisis between 2010 and 2012 and Brexit referendum in 2016). Any strong performance of the model is very unlikely to arise due to pure luck because sustaining good performance would be achieved even in the toughest and most unpredictable economic conditions.

42 explanatory variables (x) are selected and collected at the same frequency within the same period as the response variables. A list of the explanatory variables is provided in the Appendix. There is no missing value for the whole dataset, due to the high reporting disclosure in the relevant economies.

The main reason for using monthly data (not higher frequencies such as weekly or daily) is that many of the explanatory variables are macroeconomic indicators collected at monthly or even quarterly frequency. Utilising higher-frequency data would exclude many of these explanatory variables from the model and therefore deviate from one of the main targets of this chapter - working with large ANN models in terms of the number of input and hidden neurons.

4.4.2 Generation of the response variable

Since it is the direction of price movement rather than actual prices to be modelled, the initial response variables (FX rates) are transformed such that two classes are created to represent price moving up or down (or staying unchanged) in the following month. Since it is monthly data to be worked with, the likelihood of an unchanged price from month to month is very low. Therefore a third "unchanged" class is not created but combined with the "down" class.[85]

4.4.3 Selection of the explanatory variables

The selection of variables as inputs of neural network or time series models can be informed by prior literature including Zhang and Hu (1998), Panda and Narasimhan (2007), and Sermpinis et al. (2012), all of which utilise lagged FX values as inputs for their time series models. Dunis and Williams (2002) take several financial indices include FTSE100, DAX30, S&P500, Nikkei225, CAC40, Gold Bullion, Brent Crude as input variables for a neural network for the modelling and prediction of EUR/USD.

Stock indices suggest, to a certain extent, how well the economy is doing and a good stock market condition usually comes as a result of a booming economy. A similar situation exists for CPI and interest rates as normally a strong economy accompanies with higher CPI and reducing interest rates is often used as a tool to boost a bad economy. It is usually the case that during bad economic times the currency tends to be weaker and during good economic times the currency tends to be stronger. Therefore stock indices, CPI and interest rates, all of which are indicative of the economic condition of a country, should also be suggestive of the strength of the currency.

As for the oil price, due to the fact that different economies rely on oil at different levels,

[85] For the time period researched in this chapter, none of the 4 currency pairs has any zero-movement on a monthly basis.

currencies might react differently to a change in oil prices. Gold and government bonds are usually considered as an alternative to the stock and FX markets and also as the safe haven when economic condition is very bad or highly risky, which causes high volatility in the FX and stock markets. Therefore some level of correlation may exist between gold prices, interest rates[86] and FX rates, see Sjaastad and Scacciavillani (1996), Baur and McDermott (2010), and Hameed and Rose (2016).

The 42 variables selected for this research include main stock indices of the countries involved[87], lending/borrowing rates of countries involved, bond prices, CPI, Crude Oil prices, Gold Bullion and volatility indices. A list of Datastream variable ID is provided in the Appendix. All these variable are collected at the same frequency for the same period of time as the FX rates mentioned above. Summary statistics of the 42 variables are provided in Tables 7.3 - 7.4 in the Appendix.

All of the explanatory variables enter the model in levels (after standardisation) rather than differences. For ANN models, what goes into the model as inputs is essentially information. Levels are used as inputs in this chapter because they contain all information that differences hold.[88] Another reason for using levels as inputs is that unlike time series models which usually require the data be stationary[89] (and differencing data increases the level of stationarity), ANN models do not impose such restrictions on the input data.

With a large number of input variables, it is reasonable to implement feature selection on the dataset, i.e. only include those input variables which have more dominant effects on

[86]For a discussion of the relationship between FX rates and interest rates as a result of the Interest Rate Parity (IRP), refer to Subsection 2.1.4.

[87]For example when modelling GBP/USD as the focus of investigation then FTSE100, Dow Jones and NASDAQ are the indices included.

[88]Because differences are obtained from levels, levels contain no less information than differences.

[89]Refer back to Chapter 2 for more discussion on stationarity with time series models.

the response variable and exclude those which play a less important role. For a regression model, for example, feature selection can help improve statistical significance of the estimated coefficients and save computation power. Previously proposed feature selection algorithms include Fisher Score (Duda et al., 2012), Chi-square (Liu and Setiono, 1995), Low Variance (Pedregosa et al., 2011) and others. The problem, however, is that with feature selection every candidate input variable is either included in or excluded from the model, depending on some criteria from past data. There is no "in-between" state for a candidate input variable. However, this can be problematic if a variable, for example, significantly affects the response variable in certain periods and has a much weaker effect in other periods. Under those circumstances, it would be unwise to simply include or exclude the candidate input variable. With ANN models, all potential input variables are included (unless there is an obvious reason not to) and then interpreted by layers of hidden neurons like human brains. In this way, each potential input variable is learned more thoroughly. For this reason, feature selection is not conducted in this chapter to avoid excluding potentially useful variables.

4.4.4 Standardisation of the explanatory variables

Standardisation in data analysis refers to the process of transforming a group of variables (with different mean and variance) into variables of mean 0 and variance 1. It is a widely applied methodology in many areas of data analysis, e.g. regression analysis, ANN training, time series analysis. Since the response variable (FX rates) has been transformed into classes depending on whether they increase or decrease in the next period, only the explanatory variables need standardisation. To standardise a variable X, each of its n observations x_i ($i = 1, ...n$) needs to be rescaled into the standardised value s_i:

$$s_i = \frac{x_i - \bar{x}}{sd(x)}, \tag{4.10}$$

where

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \tag{4.11}$$

and

$$sd(x) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2. \tag{4.12}$$

4.4.5 Split of the training, validation and test sets

In terms of the sizes of training and test sets, there is no consensus on what the most appropriate training/test ratio would be.[90] For example, Donaldson and Kamstra (1997) use a training/test ratio of 1:1 and Sermpinis et al. (2012) adopt a training/test ratio of 2:1. In this chapter, since the training set is going to be further split to form a separate validation set, the ratio of 2:1 is used so that the size of the test set would not exceed that of the training set. 20% of the observations from the training set is then used as a validation set, i.e. the training/validation/test ratio is 1.6:0.4:1, or 8:2:5. In terms of the timespan, the training set starts in January 1999 and ends in January 2009. The validation set lasts from February 2009 to June 2012. The test set starts in July 2012 and ends in December 2018. The reason for having an extra validation set is to be able to perform overfitting detection and model selection without using the test set. For more details of detecting and reducing the overfitting problem, refer to Subsection 4.5.3.

One key difference between a time-series-type dataset and other types of non-time-series datasets is that in the process of splitting the training and test set with a time series dataset the selection cannot be random because otherwise the model would use information from the future in the training set to predict the test set and exaggerate the test set prediction performance. Instead, researchers usually split the training and test set in time order[91],

[90] Except the fact that it is rare to see a smaller training set than the test set, i.e. usually more emphasis is placed on obtaining a well-performing model with the training set.

[91] See Yao and Tan (2000), Nag and Mitra (2002), Panda and Narasimhan (2007), Sermpinis et al.

i.e. in this case the training set takes up the first two-thirds of observations and the test uses the rest one third of the observations.

After splitting the training and test sets, cross-validation[92] can be applied in the training process to increase the stability of the model estimation process. However, in this chapter, because monthly data is used the number of data points from the training set is too small for a further split into smaller validation samples. Instead, a repetitive training process is introduced to increase stability. This is further discussed in the Modelling section (Section 4.5).

4.4.6 Descriptive plots and summary statistics of the FX rates

Figures 4.2 - 4.4 display the monthly FX rates, log-returns and monthly variation[93] of the four currency pairs: GBP/USD, EUR/USD, USD/JPY and USD/CHF. Unlike realised volatility, a standard measure of volatility, monthly variation does not need to utilise daily data to compute monthly volatility. The objective of using monthly variation is simply for visualisation of volatility, instead of addressing a research question on measuring volatility. From Figure 4.4, it can be observed that although all currency pairs experienced a volatility increment in the 2008 global financial crisis, GBP/USD and USD/CHF have much greater volatilities than USD/JPY and EUR/USD. Despite having lower volatilities in the crisis, EUR/USD and USD/JPY underwent a period of higher volatilities in the late 90s and early 2000s. High volatilities in different periods generate great opportunities for investors to take advantage of the price movement in those periods.

(2012), and Rehman et al. (2014).

[92]Cross-validation is the process of splitting the training set into smaller validation samples. Each validation sample can be used on its own to estimate coefficients of the model and estimated coefficients from different validation samples can be compared with each other.

[93]Monthly variation is calculated as the monthly change of the square root of the cumulative squared monthly log-returns.

Such great opportunities exist because of the large price movement and continuing price movement, also known as "momentum",[94] in those extremely volatile periods.

A few papers research on the momentum pattern in several financial markets. For example, Jegadeesh and Titman (1993) show that buying or selling stocks following their previous performance generates significant positive returns over 3-12 month holding periods. Apart from the stock market, Moskowitz and Pedersen (2012) find persistence in returns from one to 12 months in equity index, currency, commodity, and bond futures. Both Jegadeesh and Titman (1993) and Moskowitz et al. (2012) show that after 12 months the momentum pattern turns into a reversal pattern, i.e. positive returns are more likely to be followed by negative returns and vice versa. In addition to time horizon, He and Li (2015) conclude that market dominance of momentum traders also determines the performance of momentum trading strategy. The literature of the momentum pattern discussed here does not aim at illustrating the trading models they use but showing that the existence of the momentum pattern might affect performance of trading models.

[94]The term "momentum" refers to the extent to which the sign of price movement continues, i.e. positive changes follow previous positive changes and negative changes follow previous negative changes.

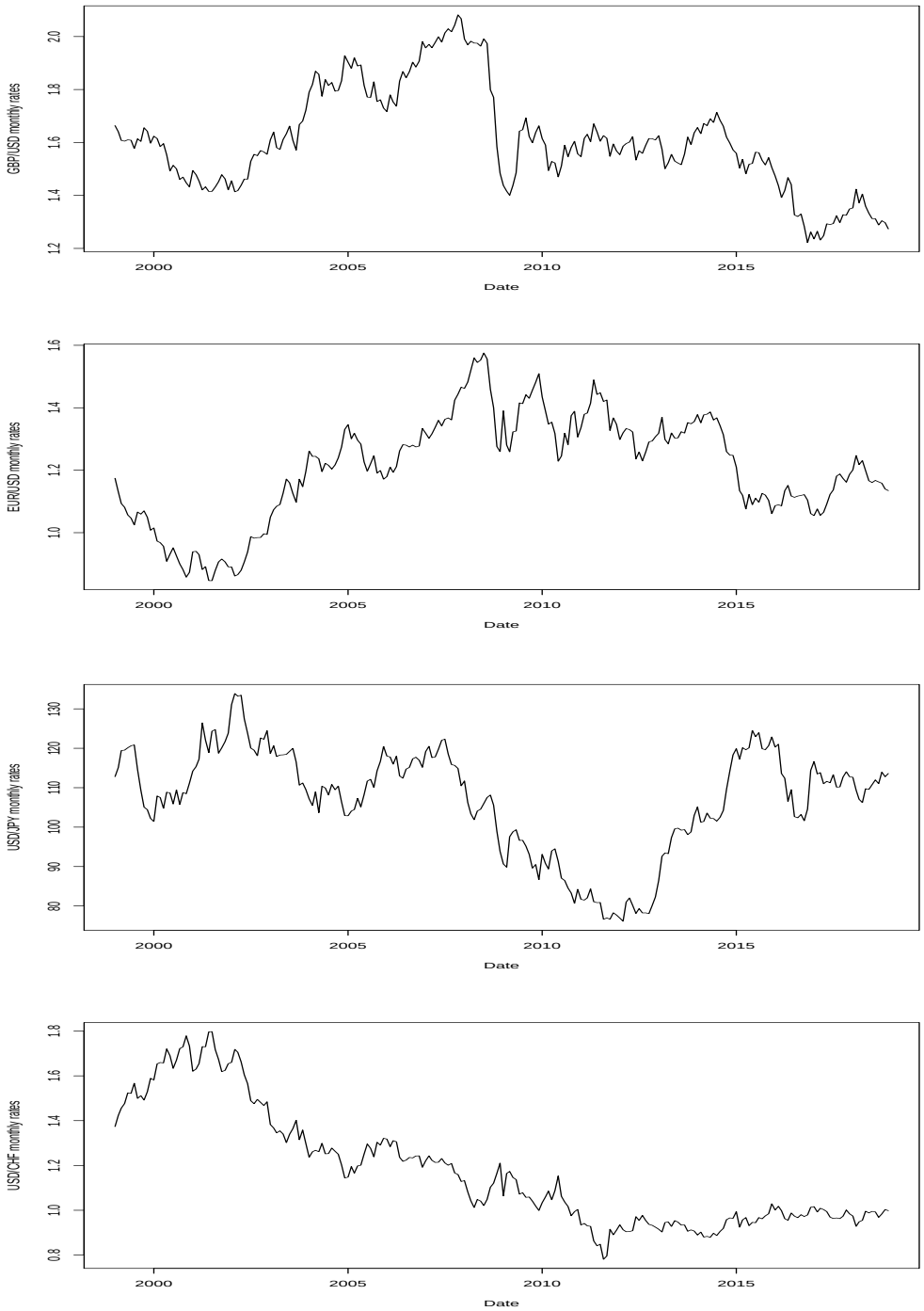


Figure 4.2: Monthly rates of the four currency pairs: GBP/USD, EUR/USD, USD/JPY and USD/CHF.

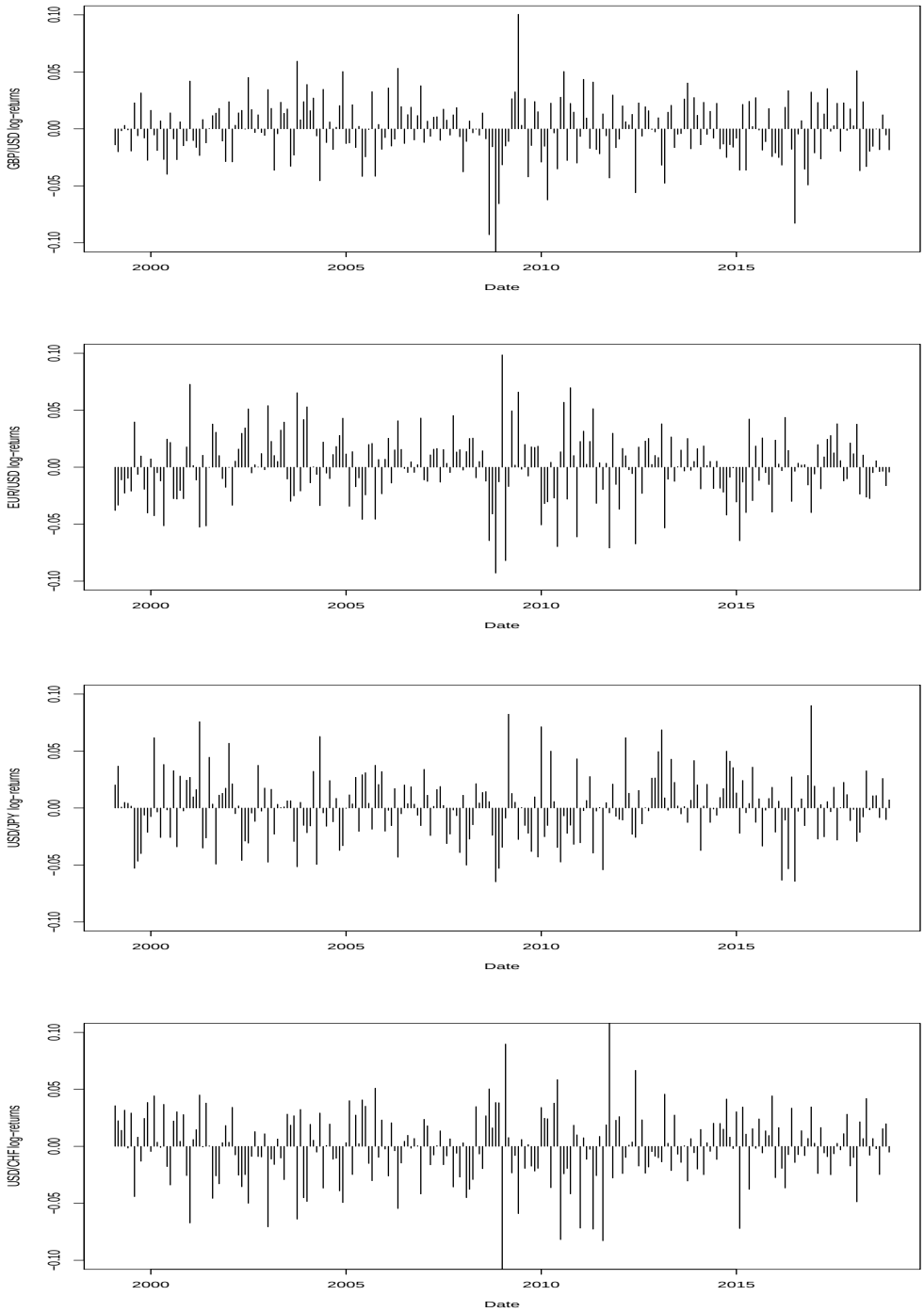


Figure 4.3: Monthly log-returns of the four currency pairs: GBP/USD, EUR/USD, USD/JPY and USD/CHF.

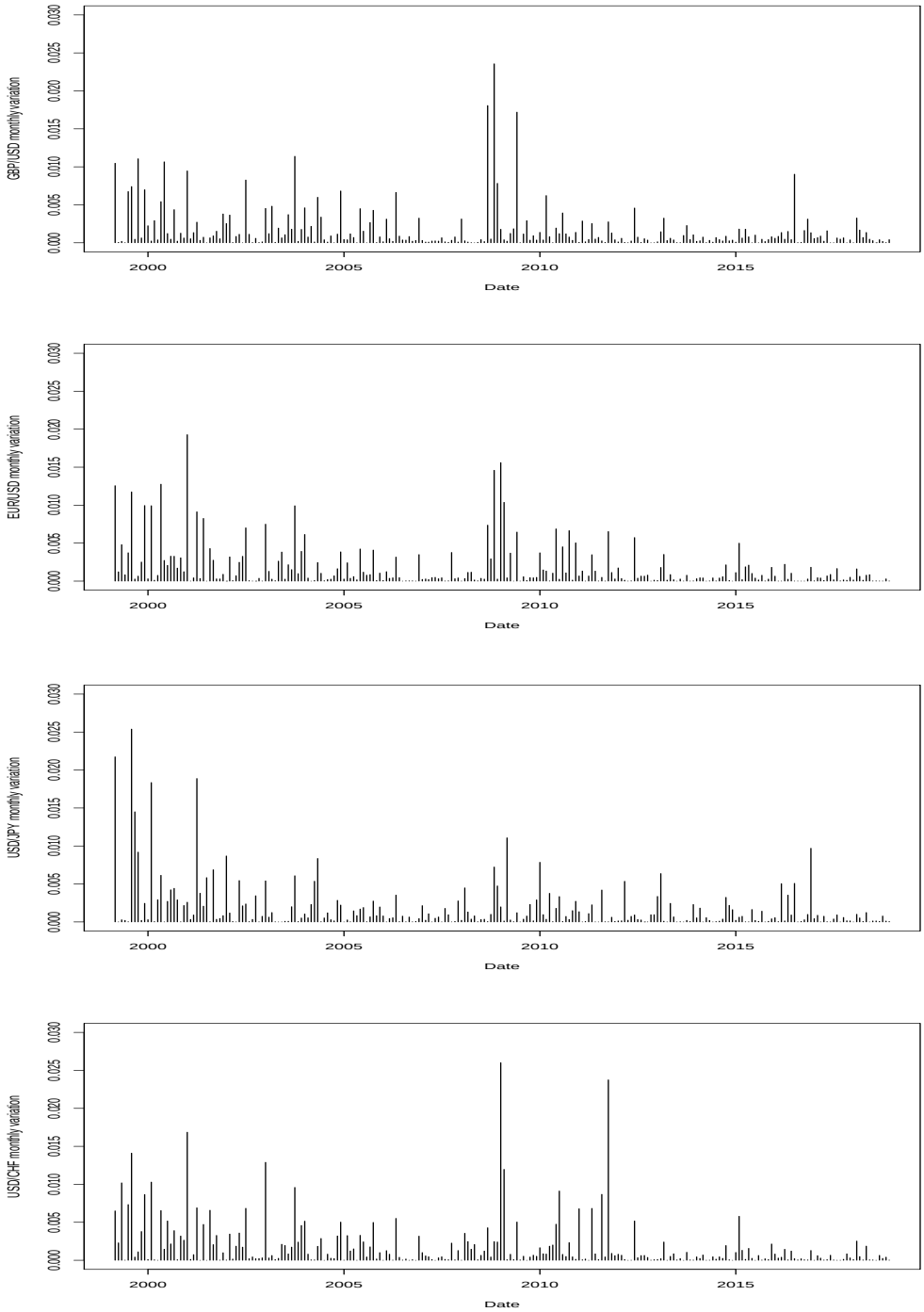


Figure 4.4: Monthly variation of the four currency pairs: GBP/USD, EUR/USD, USD/JPY and USD/CHF.

Table 4.2 shows summary statistics including maximum, minimum, mean, median, standard deviation, percentage of positive changes and percentage of negative changes of log-returns of the FX rates. Summary statistics tables of the 42 explanatory variables are displayed in the Appendix. From Table 4.2, GBP/USD has the highest level of imbalance

	GBP	EUR	JPY	CHF
Maximum	0.101	0.099	0.090	0.140
Minimum	-0.111	-0.093	-0.065	-0.129
Mean	-0.001	-0.0001	0.00003	-0.001
Median	-0.002	0.001	0.0004	-0.001
Standard deviation	0.026	0.029	0.028	0.031
Positive changes (%)	46	51	51	49
Negative changes (%)	54	49	49	51

Table 4.2: Summary statistics of log-returns of FX rates: GBP, EUR, JPY, CHF. All currencies are against USD.

[95] in terms of percentage of positive changes and negative changes, this could potentially increase the predictability of GBP/USD over the other three currency pairs.

As has been discussed briefly earlier, momentum is an important aspect to consider regarding to positive and negative changes. One way to measure momentum is by using the percentage of price movement continuation.[96] Table 4.3 shows the percentage of price movement continuation of the four currency pairs for the whole dataset, the training set and the test set. Comparing the percentage of price movement continuation of the

[95]The "highest level of imbalance" of GBP/USD means that the absolute difference (8%) between percentage of positive changes and negative changes is larger than other currency pairs.

[96]The percentage of price movement continuation is calculated as the ratio of price movement where the sign of movement remains unchanged for the next period, to the total number of price movement.

training set with the test set provides a measure of the similarity between the training and test set and hence the predictability of the test set. In Table 4.3, the log ratio is calculated by taking logarithm of the ratio of percentage of price movement continuation for the training set and the test set. A positive log ratio indicates that the training set has greater momentum than the test set and a negative ratio indicates the opposite. The absolute value of the log ratio, on the other hand, shows the deviation of the test set from the training set in terms of percentage of price movement continuation. A larger (absolute) log ratio indicates a larger deviation of the test set from the training set in terms percentage of price movement continuation. USD/JPY has the highest train-test log ratio of the four currency pairs. GBP/USD and EUR/USD have significantly lower train-test log ratio than the other two currencies. This will be confirmed by better forecasting performance of these two pairs in the later modelling section.

	GBP	EUR	JPY	CHF
Whole dataset (%)	44.54	51.26	48.73	44.96
Training set (%)	44.94	50.63	53.16	48.10
Test set (%)	43.04	51.90	40.50	37.97
Train-test log ratio	0.04	-0.02	0.27	0.23

Table 4.3: Percentage of price movement continuation for the four currency pairs. All pairs are against USD.

4.5 Modelling

4.5.1 Overall process

A single currency pair (GBP/USD) is focused on initially to identify any problems in the modelling process. The reason for this is that as long as the problems are not related to a certain currency pair, it is not necessary to spend time on repeating the computation once the problems are already identified. These problems will then be investigated and if an improvement is made the improved model will be used for modelling with other currency pairs for best possible performance. All graphics and tables in this section, without specifying which currency pair is being modelled, refer to the GBP/USD pair.

A key step in the training of a neural network is parameter tuning and parameter estimation. Parameter tuning refers to the process of deciding for example how many neurons to include per layer and how many layers to use. Once these two aspects are fixed the number of parameters to be estimated is known and the parameters can be estimated. However, researchers have found that the number of neurons is far more important than the number of hidden layers. According to Donaldson and Kamstra (1997), if a sufficient number of nodes are placed on the first hidden layer, higher layers are not usually needed to establish satisfactory connection between the initial input and final output. Stathakis (2009) points out that any continuous function can be represented by a neural network that has only one hidden layer. Therefore the main tuning process in this research only involves the tuning of the number of neurons.

In the estimation process two key aspects are illustrated: the loss function and the overfitting problem.

4.5.2 Loss function

The rule to estimate parameters is that we first define a loss function and then aim to find the parameters that minimise the loss function. A commonly used loss function for

classification problems is the cross-entropy error function, which is defined as follows. Assume we have a 2-class classification problem, i.e. the output variable $y \in \{0, 1\}$. If the predicted output for one prediction is denoted as $\hat{y}$ then the cross-entropy error function for this prediction is:

$$H = -y \log \hat{y} - (1 - y) \log(1 - \hat{y}), \tag{4.13}$$

and for multiple (N) predictions to calculate the cross-entropy loss the error functions for all predictions are averaged:

$$J = -\frac{1}{N} \sum_{i=1}^{N} \big(- y_i \log \hat{y}_i - (1 - y_i) \log(1 - \hat{y}_i) \big), \tag{4.14}$$

where J is also called the cost function.

Unlike a simple quadratic function where the global maximum/minimum is known to exist and easy to find, a function like the cross-entropy loss, especially after plugging in as $\hat{y}$ the expressions of numerous neurons in different layers, generates great difficulty in finding the global optimum[97]. Therefore, in machine learning it is usually the local optimum that is sought. A traditional training process starts with random initial values of the parameters to be estimated and the subsequent values (called "optimiser") develop such that the loss function reduces as fast as possible until convergence (stability) is reached.

4.5.3 The overfitting problem

Another important issue in deep learning is the overfitting problem where the number of parameters is so large that the model essentially remembers the dataset rather than learns from it and therefore loses the ability to generalise. As this chapter works with a large number of input variables, the overfitting problem should not be neglected.

[97] In a case where there is one or more singularities (undefined points that lead to infinity values), the global optimum might not even exist.

In the training process of ANN models, one way to deal with the overfitting problem is called early stopping. To perform an early stopping process, a random small fraction of the training set is selected as a validation set (not used in the initial training process), see Subsection 4.4.5. The training process will stop when the error on the validation set starts to increase. Figure 4.5 helps visualise the early stopping idea.

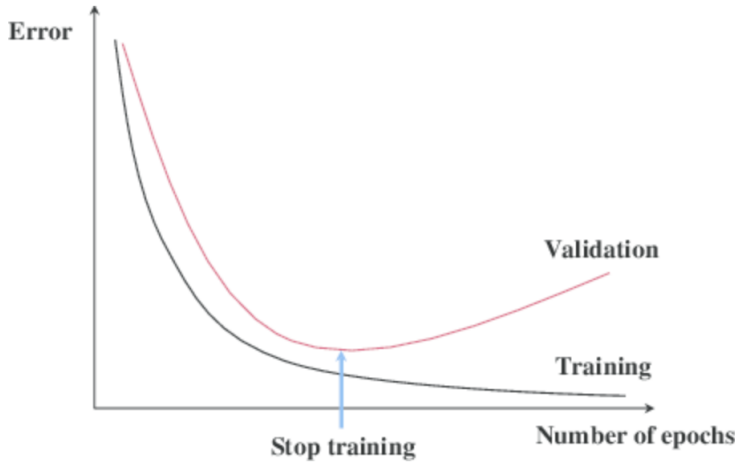


Figure 4.5: A plot showing early stopping in the training process of ANN models.

The reason for selecting a separate portion of the training set as the validation set (instead of using the test set for selecting non-overfitting models) is to avoid the use of the test set in the model selection process, which would cause the problem of "utilising the future data to forecast past data". Larsen et al. (1996), Bylander and Tate (2006), Guresen et al. (2011), Alvarez and Salzmann (2016), Lever et al. (2016), and Xu and Goodacre (2018) use a validation set for overfitting detection and model selection. All papers produce evidence that with the extra validation set, the problem of overfitting has been significantly mitigated.

Another recently developed method to deal with the overfitting problem is the adoption of drop-out layers. The word "drop-out" refers to randomly ignoring some neurons (along with their connections) during the training phase. Figure 4.6 visualises how drop-out

works. Srivastava et al. (2014) conduct a detailed discussion on this methodology and show that the drop-out methodology significantly improves the performance of neural networks on supervised learning tasks in vision, speech recognition, document classification and computational biology.

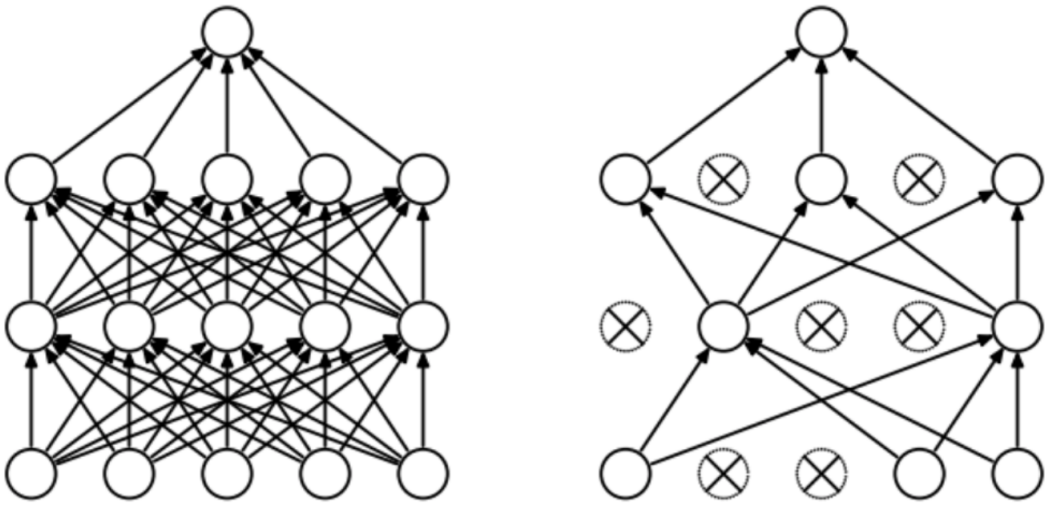


Figure 4.6: A plot showing a drop-out structure in the training process of ANN models. Source of figure: Srivastava et al. (2014).

4.5.4 Estimation performance improvement

Due to the randomness in selection of initial values, a single run of the network training process very often generates unstable estimation results and unluckily most of the time a single run will generate a bad result rather than a good one as good estimated values for parameters are only a small proportion of the whole parameter search space. Another reason which supports the use of multiple runs is that (as mentioned before) different number of neurons will be compared with each other in terms of their forecasting

performance and then the most appropriate number of neurons will be used for modelling in the next stage. Therefore using the performance of a single run is inappropriate and unfair. One way to deal with this problem and therefore improve performance is to run the training process multiple times. This methodology has been adopted in the machine learning area recently. Fischetti and Jo (2018) run their deep neural network 100 times in order to compare two models based on a few metrics. Namin and Namin (2018) also discuss the improvement of multiple runs of a LSTM over a single run in forecasting financial indices and economic indicators.

4.5.5 Evaluation metrics

Once the model is trained the next step is to evaluate the model using different evaluation metrics. The cross-entropy loss function used for estimating the parameters on the training set is one option. However, as the datasets to be modelled and predicted in this chapter are all financial time series (currency pairs), the cross-entropy loss function is not the best choice in terms of practical value. Therefore two financial metrics are introduced: prediction accuracy and annualised return with a simple buy-low-sell-high trading strategy. The prediction accuracy measures the percentage of correctly predicted[98] observations and annualised return is calculated on a monthly basis.

Since multiple outcomes of the estimation algorithm are run every time, the mean accuracy and annualised return (together with their corresponding standard deviations) are displayed. This will not only enable the location of the best performing model with the most appropriate number of neurons but also identify how stable models with a certain number of neurons are performing. For both accuracy and annualised return, their standard deviation is also considered important, as a measure of stability in performance. Therefore, it is possible to obtain two models, say Model A and Model B, such that

[98] A correct prediction refers to one that we predict the exchange rate to move up/down during the next month and it does move in that direction.

Model A produces a higher accuracy percentage than Model B while Model B generates results more stable than Model A. A similar situation can happen for the annualised return metric. In both cases it is not clear which model performs better. It is therefore reasonable to construct a combination of the performance metric value and its standard deviation. One possible combination is an average of the metric value and its standard deviation. However, as the magnitude of the metric value and its standard deviation are not at the same level, taking averages is not a proper choice. An alternative combination is the ratio of the metric value and its standard deviation. The underlying idea for this combination to be effective is that the metric value (accuracy or annualised return) is positive-directional, i.e. the larger the value the better the model, while the standard deviation is negative-directional, i.e. the smaller the value the better the model. The ratio of the metric value and its standard deviation will increase if either the metric value increases or its standard deviation decreases, i.e. a high ratio reflects good performance of the model due to high accuracy/annualised return or low standard deviation or both and a low ratio reflects bad performance of the model due to low accuracy/annualised return or high standard deviation or both. The ratio of accuracy and its standard deviation, denoted as R_A and the ratio of annualised return and its standard deviation, denoted as R_P will therefore be constructed and used for comparing models whose metric values and standard deviations disagree on which model is better.

A time metric showing how long it takes on average for a single run is also introduced, for the purpose of evaluating computational cost.

4.5.6 Training

The GPU used for this research is NVIDIA RTX 2080TI with 11GB memory. As has been explained the number of layers is not such an important factor, a traditional 3-layer network is constructed with one input layer, one hidden layer and one output layer, together with the drop-out structure. The number of neurons to be tested on ranges from

$(2, 2^2, 2^3, ..., 2^{12})$. For the maximum value 2^{12} neurons, the total number of parameters becomes over 16 million. Together with the optimiser indices and other types of data, a single run of the programme is close to taking up the memory limit of the GPU.

One problem encountered while training models with different numbers of neurons is that as each trained model gets larger the GPU memory gets full very quickly and starts to utilise the computer's memory which has significantly slower data transfer speed than that of the GPU. This will dramatically slow down the training process.

Table 4.4 shows that as the number of parameter increases there is not an increasing pattern in computation time. In fact some of the smaller models take a longer time to train. This is mainly because of the randomness in the training process that causes different convergence times. This effect is not evident in the 30-run training process since the computation time increases as the number of parameters increases. However, if we compare horizontally in the table, the time difference between the two situations is so large. To be more precise, with 4096 neurons if we restart R (the easiest way to clear GPU memory) every time a single run is complete and conduct 30 runs, this would take around 352 seconds (close to 6 minutes) while if the 30 runs are conducted one by one without restarting R, i.e. within a for-loop, it would take almost an hour of computation time. Unfortunately it is impractical to restart every time by hand before starting the next run because it limits the number of runs we can take and after the number of neurons is fixed it would take ideally no less than 100 runs for the training process to generate a good result.

Attempts have been made to put a restart command right after each iteration within the for-loop so that all runs can be conducted in one go and GPU clears its memory after each run. However, this does not work due to data loading problems. In fact, after extensive investigation it is discovered that a restarting command within a for-loop is impossible to implement.

	N	T_1	T_{30}
1	2	9.25	11.09
2	4	15.93	15.49
3	8	7.45	19.15
4	16	7.37	25.26
5	32	7.50	31.08
6	64	7.72	37.06
7	128	7.86	45.54
8	256	7.85	53.76
9	512	8.00	63.81
10	1024	8.18	76.12
11	2048	8.88	92.31
12	4096	11.72	112.53

Table 4.4: A table showing computation time per run measured in seconds for a single run and 30 runs with different numbers of neurons. N is the number of neurons per hidden layer, T_1 is the computation time of a single run and T_{30} is the computation time per run as an average of 30 runs for the 12 levels of N. Note: An attempt to repeat the computation in this table is unlikely to generate the exact same computation time in seconds. However, the increase in computation time per run as the number of runs increases, i.e. from T_1 to T_{30}, remains significant, especially when the model includes more neurons (larger N). This is what the table aims to show.

Another direction in which this problem can be dealt with is to compromise the number of runs in the parameter tuning process and increase it to the ideal amount once the parameter tuning finishes and the finalised model is to be trained. After a few experiments, the number of runs for the parameter tuning process is set to be 10, providing relatively reliable performances for comparison within a reasonable time.

The annualised return (P) is defined as:

$$P = \big((1 + r_1)(1 + r_2)...(1 + r_n)\big)^{1/n} - 1, \tag{4.15}$$

where r_i represents the return of the i'th period and n is the number of periods.

Table 4.5 presents the performance metrics from 10 runs of the 12 specifications. Since there are 4 financial metrics on which the comparison is based, it is unlikely that all 4 would agree on the best performer, hence a ratio of accuracy to its standard deviation and a ratio of annualised return to its standard deviation are computed. As has been discussed in the "Evaluation metrics" section, these ratios are considered because we want accuracy and annualised return as high as possible and the standard deviations as low as possible. The ratios of these two pairs will serve as an overall measurement and help compare models whose metric values and standard deviations disagree on which model is better. The better-performing model will always have higher ratios.

Based on R_A and R_P, the two best performers are when $N = 1024$ and $N = 2048$, one with higher R_A and the other with higher R_P. Given that the difference between these two specification are very small, the smaller specification $N = 1024$ is chosen due to shorter computation time.

The process of neuron number selection in this chapter not only serves a purpose of modelling but also supports the use of large MLPNN over traditional MLPNN with much fewer neurons.

N	A (%)	SD_A	P (%)	SD_P	T_{10} (s)	R_A	R_P
2	50.50	0.0633	-0.32	0.0381	8.12	7.9785	-0.0843
4	49.50	0.0757	-0.49	0.0435	10.59	6.5421	-0.1126
8	56.12	0.0473	1.95	0.1383	12.31	11.8776	0.1408
16	52.88	0.0696	0.10	0.0577	12.97	7.5961	0.1644
32	54.87	0.0535	2.13	0.1563	13.48	10.2650	0.1365
64	50.88	0.0716	0.66	0.0738	14.56	7.1054	0.0897
128	55.75	0.0363	2.56	0.1379	16.12	15.3520	0.1857
256	56.87	0.0528	3.85	0.2359	17.26	10.7694	0.1633
512	56.38	0.0449	3.11	0.1925	18.64	12.5617	0.1614
1024	58.50	0.0146	3.89	0.2351	20.75	40.1306	0.1653
2048	59.13	0.0148	3.93	0.2318	22.83	39.8337	0.1694
4096	56.12	0.0479	2.63	0.1392	26.10	11.7148	0.1889

Table 4.5: A table displaying performance evaluation metrics on the test set for different numbers of neurons with 10 runs, where N is the number of neurons, A represents the mean accuracy rate from the 10 runs for a given number of neurons, SD_A denotes the standard deviation of accuracy, P denotes Annualised return, SD_P denotes the standard deviation of annualised return, T_{10} denotes the average computation time in seconds per run out of the 10 runs, R_A denotes the ratio of accuracy to its standard deviation and R_P denotes the ratio of annualised return to its standard deviation. The underlying currency pair is GBP/USD. Computation results for EUR/USD, USD/JPY and USD/CHF are displayed in Tables 7.15 - 7.17 in the Appendix.

4.5.7 Test set prediction with 100 runs of the 1024-neuron MLPNN

After the number of neurons is selected to be 1024 the next step is to run the ANN a significant number of times so that the best possible results can be obtained. Table 4.6 displays part of the 100 runs of the network and their corresponding performance evaluation metrics. Out of the 100 runs the best accuracy achieved is 62.5% and the highest annualised return achieved is 6.7%.

The choice of 100 runs is checked by plotting accuracy/annualised return as the number of runs increases. Figure 4.7 shows the highest accuracy obtained as the number of runs increases and Figure 4.8 displays the highest return obtained as the number of runs increases. In addition, Figure 4.9 plots the computation time of each of the 100 runs. Both the accuracy and return series display a convergence patten. While accuracy completely stablises before 20 runs, return still has some improvement after even 90 runs, supporting the decision of choosing 100 as the number of runs. Figure 4.9 shows an increasing trend (certainly faster than linear increment) of computation time as the number of runs increases. To see how fast the slowdown speeds up, the start and end points are used to estimate the following functions: linear, exponential and factorial. Figure 4.10 shows a comparison of the computation time with the other three types of increments. As can be seen from Figure 4.10, the computation time increment speed is faster than (due to larger second-order derivatives) linear and exponential but slower than factorial increment. This is mainly due to the GPU reaching its maximum memory and, as a result of the fast time increment speed, if we used a 100-run approach for each of the 12 specifications in the parameter tuning process, computation time needed to complete the task would have been enormous.

Run number	Accuracy (%)	Annualised return (%)	Time (seconds)
1	57.50	2.75	9.53
2	57.50	2.75	16.58
3	60.00	4.43	23.75
4	57.50	2.75	31.36
5	60.00	4.43	39.20
6	58.75	4.02	46.75
7	60.00	4.41	55.06
8	60.00	4.62	62.89
9	62.50	5.27	70.80
10	61.25	5.70	79.03
		.....	
91	57.50	3.89	1184.73
92	60.00	4.43	1204.08
93	57.50	4.53	1223.66
94	60.00	4.10	1243.27
95	45.00	-1.85	1263.03
96	62.50	5.73	1283.16
97	61.25	6.67	1303.52
98	58.75	4.68	1324.20
99	57.50	2.75	1344.92
100	58.75	5.31	1365.83

Table 4.6: The GBP/USD test set prediction performance from 100 runs of a 1024-neuron MLPNN. Only the first and last 10 runs are listed due to space limit. The full results are provided in Tables 7.5 - 7.9 in the Appendix.

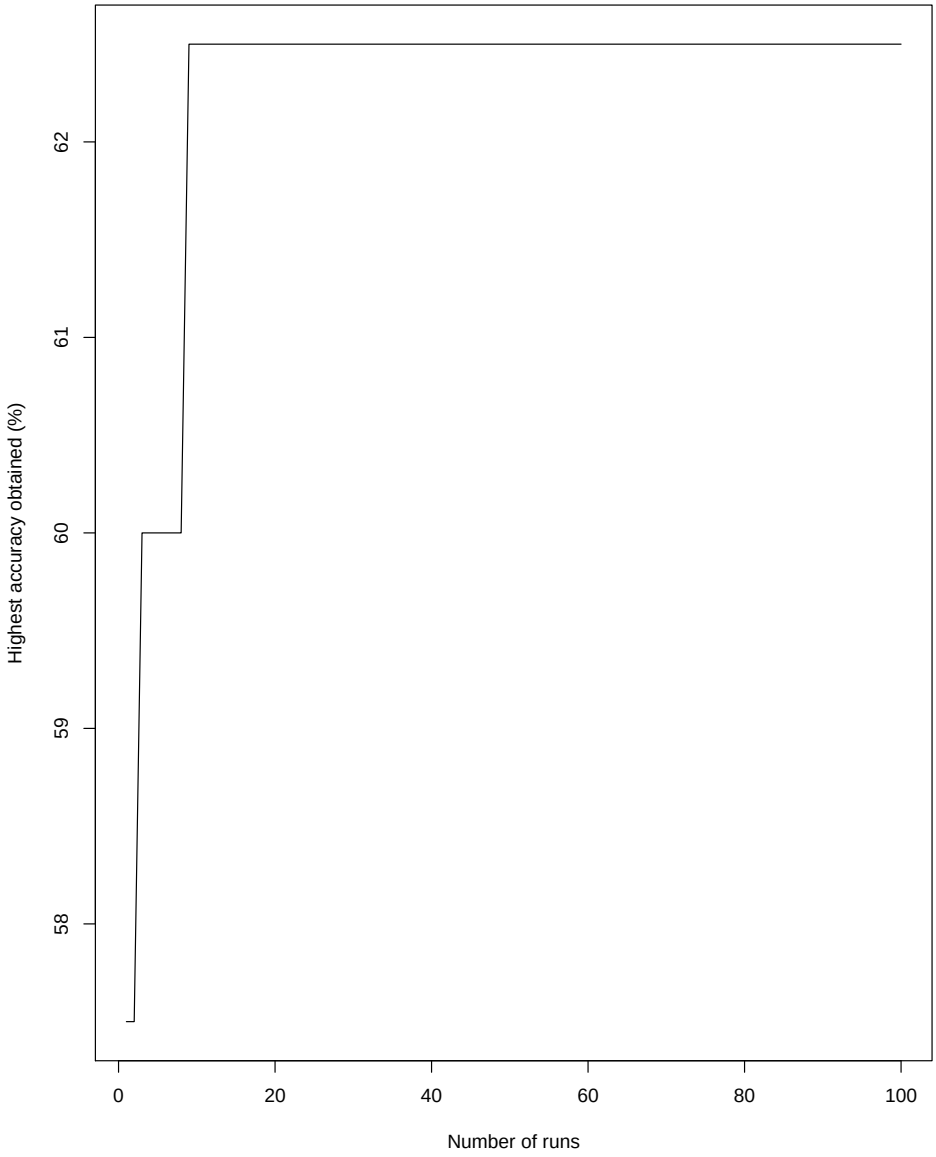


Figure 4.7: A plot showing the highest accuracy obtained as the number of runs increases.

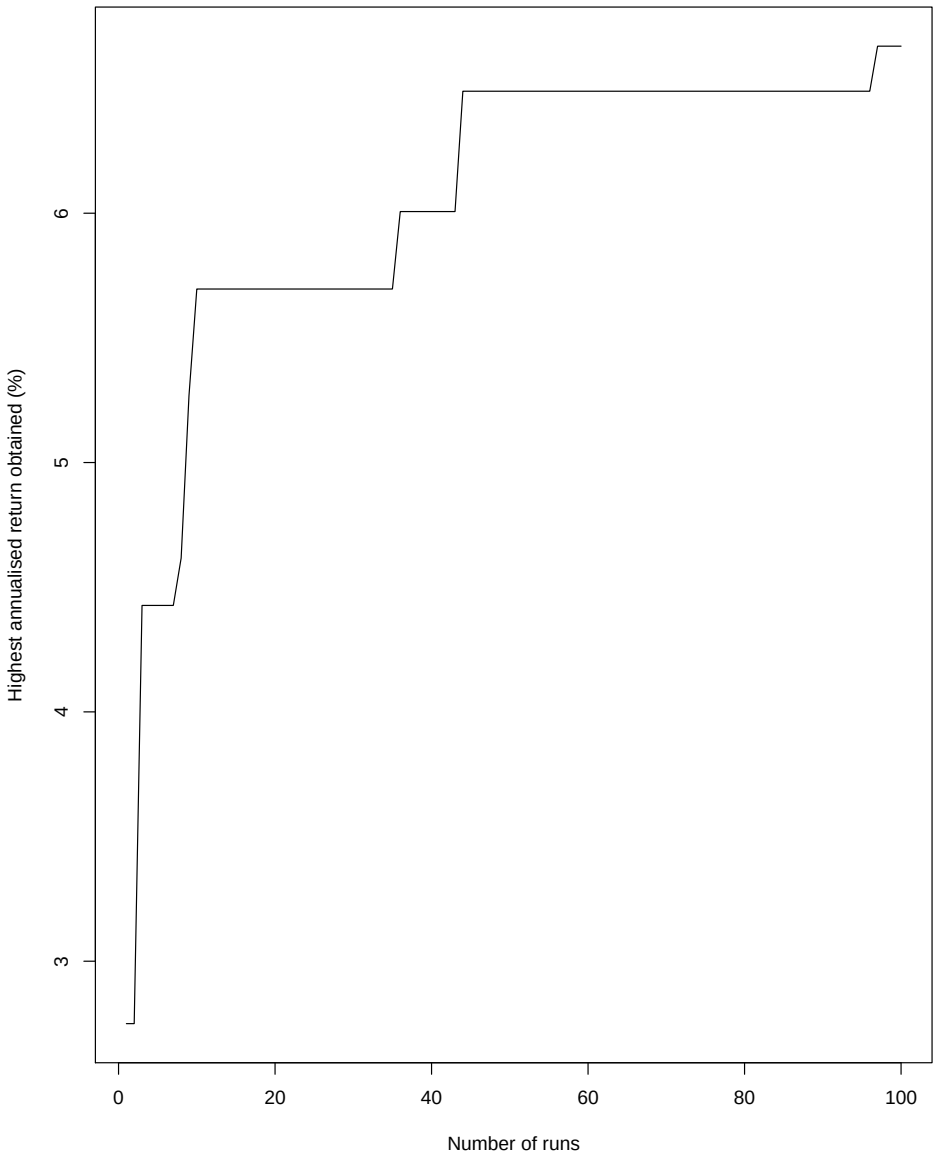


Figure 4.8: A plot showing the highest return obtained as the number of runs increases

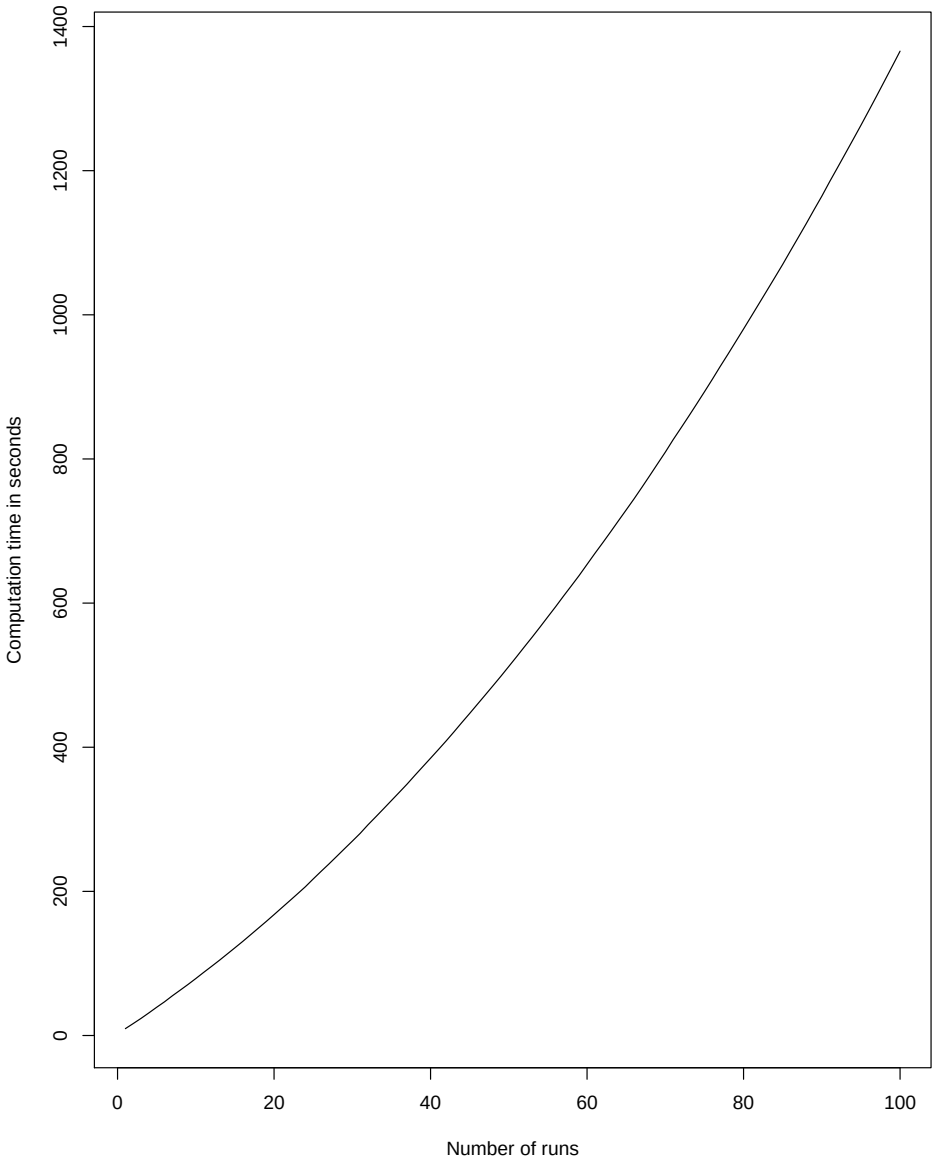


Figure 4.9: A plot showing the computation time for the 100 runs.

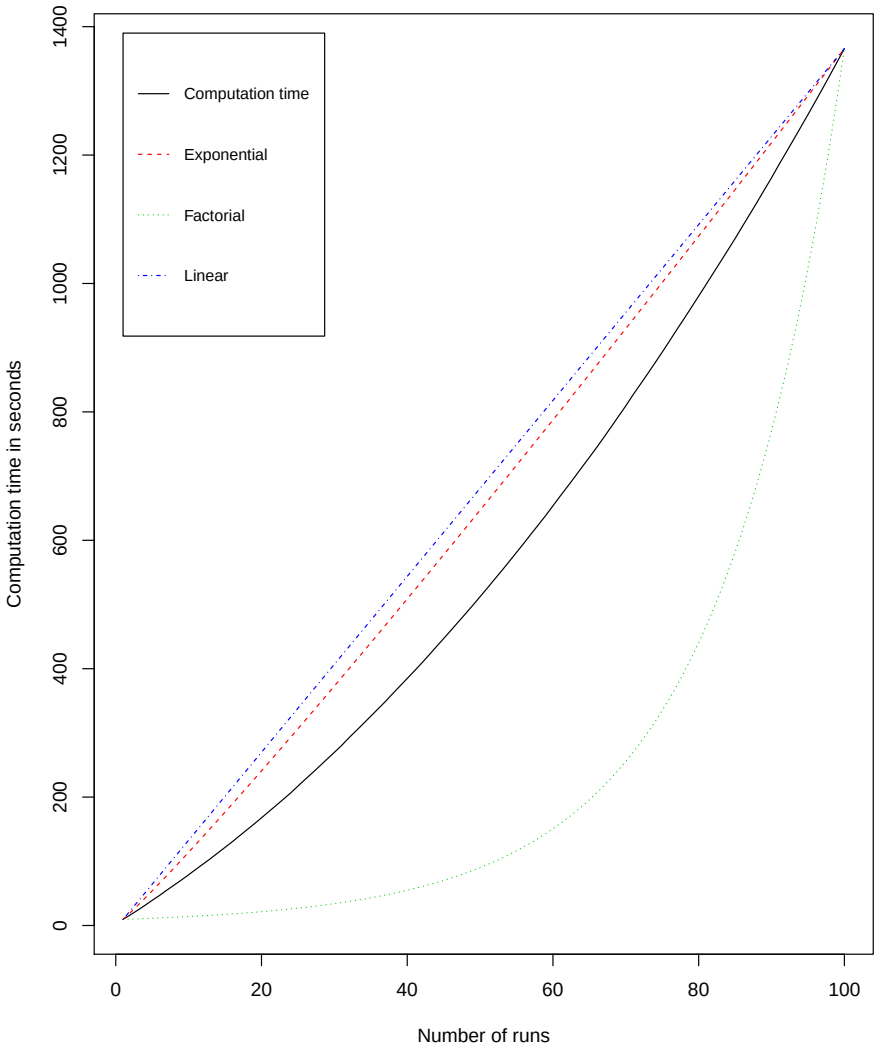


Figure 4.10: A plot showing the computation time for the 100 runs compared with linear, exponential, and factorial increments.

4.5.8 Comparison of CPU and GPU computation time

As reported computations are conducted via GPU. This subsection reports results for the same computations using CPU[99] so that any difference in computation time can be compared. The same task (test set prediction with 100 runs of the 1024-neuron MLPNN) is conducted and the computation time of each of the 100 runs is recorded. Table 4.11 shows a comparison of the computation time of the 100 runs between CPU and GPU.[100] From Table 4.7 it can be seen that the GPU always finishes the task faster than the CPU. The mean speed-up is almost 24%. However, the actual speed-up percentage varies. Figure 4.11 plots the GPU speed-up percentage for the 100 runs. The graph shows that GPU speed-up percentage increases first and then decreases. The reason for the increment initially is that GPU takes longer time to initialise than CPU and the decrement at a later stage is mainly due to the maximum memory of the GPU being reached. The maximum speed-up occurs at around the 6th run, meaning that after the 6th run the GPU memory exceeds its limit and starts to utilise the computer's memory which is much slower than the GPU's memory. Another interesting fact is that after the peak speed-up is reached, the speed-up decrement slows down and has a tendency to converge as the number of runs increases. The convergence level of speed-up refers to the level GPU speed-up over CPU when the maximum memory of the computer is almost reached, after which both CPU and GPU will stop working as a result of no available memory.

[99] The CPU used in this research is an AMD Ryzen Threadripper 1950x 16-core processor.

[100] Again only the first and last 10 runs are displayed which is enough to observe the pattern.

	CPU time (s)	GPU time (s)	GPU speed-up (%)
1	10.81	9.53	13.43
2	20.98	16.58	26.54
3	30.89	23.75	30.06
4	41.15	31.36	31.22
5	51.87	39.20	32.32
6	62.12	46.75	32.88
7	72.57	55.06	31.80
8	83.25	62.89	32.37
9	94.07	70.80	32.87
10	104.78	79.03	32.58
		.....	
91	1430.86	1184.73	20.78
92	1453.20	1204.08	20.69
93	1475.67	1223.66	20.59
94	1498.36	1243.27	20.52
95	1521.15	1263.03	20.44
96	1544.53	1283.16	20.37
97	1567.79	1303.52	20.27
98	1591.64	1324.20	20.20
99	1615.67	1344.92	20.13
100	1639.73	1365.83	20.05
Mean	711.46	576.24	23.47

Table 4.7: A table showing the CPU computation time against the GPU computation time and the GPU speed-up for part of the 100 runs. Full table is provided in Tables 7.10 - 7.14 in the Appendix.

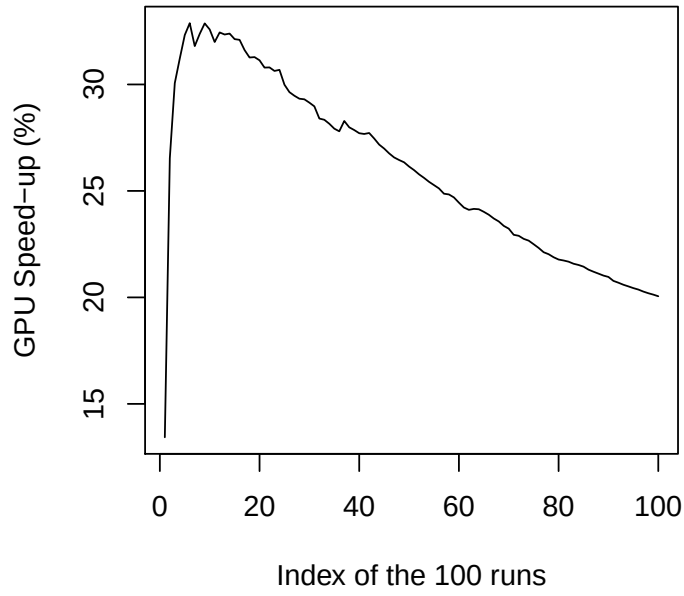


Figure 4.11: A plot showing the GPU speed-up percentage for the 100 runs.

4.5.9 Modelling and predicting with selected currencies

Another three currency pairs (EUR/USD, USD/JPY and USD/CHF) are modelled and predicted with the same procedure as for GBP/USD. Table 4.8 displays the test set prediction results from the best performing model for all three currency pairs together with GBP/USD.

Currency	N_1	N_2	Best A (%)	Highest P(%)	P_{BH} (%)
GBP	2^{10}	1102850	62.50	6.67	-1.05
EUR	2^8	79106	58.75	5.56	-0.77
JPY	2^9	289282	61.25	6.92	1.60
CHF	2^7	23170	53.75	4.45	0.51

Table 4.8: Test set prediction results for the four currency pairs. N_1 - number of neurons, N_2 - total number of parameters. Best A - best accuracy. Highest P - highest annualised return. P_{BH} - annualised return with a simple "buy and hold" strategy. All currency pairs are against USD. Note: transaction costs are not considered because the estimated annual transaction costs are negligible compared with the annualised returns. Refer to Table 4.9 of Subsection 4.5.10 for more details on the estimation of transaction costs.

From Table 4.8, we can observe that models with more parameters (for modelling GBP and JPY) produce better accuracy and higher annualised return as well, while models with fewer parameters (for modelling EUR and CHF) generate lower accuracy and annualised return. This, however, does not mean that more neurons should simply be added to the model but the amount of information the ANN can learn from the given information (dataset) is more limited than for the other two more profitable currency pairs.

With the simple "Buy and hold" strategy, the base currency (for example GBP is the base

currency for the pair GBP/USD) is bought and held till the end of the relevant period. It serves as a benchmark strategy as opposed to the "Buy-Low-Sell-High" strategy. The ANNs for all four currency pairs generate significant improvement over the "Buy and hold" strategy.

4.5.10 Transaction costs

Whenever trading profits are concerned, transaction costs are a crucial factor not to be neglected. However, the effect of including transaction costs not only depends on the size of the bid/ask spread but also on the frequency of trading as well. Below is an illustrative example.

If trader A is a long-term trader who places only one trade per year and trader B is a daily trader who places one trade per trading day (or approximately 253 trades per year). Assume both make an annual return (not considering bid/ask spread) of 10%. Assume the average transaction cost per trade is 0.02%. Because trader A only makes one trade, the transaction cost is deducted once. Therefore, trader A makes a 9.98% profit after considering transaction costs. However, for trader B who makes 253 trades, the transaction costs are incurred 253 times. If we assume that the 10% annual return is distributed evenly among 253 trades and profit compounding is not considered, then the average profit per trade (before transaction costs) is approximately 0.04%. After accounting for 0.02% of transaction cost per trade, trader B makes a profit of 0.02% per trade, or 7.06% in a year. This is over 2% less than trader A makes, although both makes the same amount of profit in that year before transaction is considered. An astounding fact is that if the transaction cost per trade goes up to 0.04%, then trader A makes 9.96% in that year while trader B makes 0%.

A concluding remark on this hypothetical example is that transaction costs affect high frequency traders significantly more than low frequency traders.

Bid-ask spreads are generally not available for non-tick data. Therefore for Chapters 4 and 5 where the data frequencies are monthly and hourly, it is practically infeasible to obtain a spread for every data point and therefore account for transaction costs in the most accurate way.

However, transaction costs can be estimated with data provided by some FX brokers. OANDA Corporation publishes recent spreads up to 3 months for most major currency pairs.[101] The median spread values in Table 4.9 as summarised from OANDA are in line with McGroarty et al. (2007).[102] The median spread values are used as an estimate of the percentage transaction cost per trade. Since trades are made on a monthly basis, i.e. the annual number of trades is fixed at 12, the annual percentage transaction cost is estimated by multiplying the estimated percentage transaction cost per trade by 12. Table 4.9 displays the estimated annual transaction costs for the four currencies. Based on a trading frequency of 12 times a year (as the trading strategy in this chapter works), transaction costs are negligible and therefore neglected in this chapter.[103]

FX pair	Median spread	Average percentage cost (%)	Annual percentage cost (%)
GBP	0.00018	0.015	0.18
EUR	0.00015	0.012	0.14
JPY	0.013	0.012	0.14
CHF	0.00018	0.015	0.18

Table 4.9: A table showing an estimate of annual transaction costs for the four currency pairs. The "Median spread" and "Average percentage cost" are calculated as per trade. Since in Chapter 4, trades are placed exactly once per month, the "Annual percentage cost" is estimated by multiplying the "Average percentage cost" by 12. All pairs are against USD.

[101] Visit `https://www1.oanda.com/forex-trading/markets/recent` for a link to the website.

[102] For example, for the currency pair that both this thesis and McGroarty et al. (2007) study - USD/CHF, the estimated bid-ask spread is 0.000180 in this thesis and 0.000177 by McGroarty et al. (2007).

[103] None of the four pairs incurs more than 0.2% transaction costs per year and 0.2% is negligible compared with all of the annual profits, i.e. in Table 4.8, the lowest annual profit is 4.45%.

4.5.11 Comparing large MLPNN with a benchmark model

A benchmark model, logistic regression is also applied on the same dataset. Logistic regression is a frequently used technique for dealing with classification problems. To predict the class of a variable Y, logistic regression computes the probability of Y given a set of explanatory variables X_i, $i = 1, ..., n$ as:

$$P(Y|X_1, ...X_n) = \frac{exp(\beta_0 + \sum_{i=1}^{n} \beta_i X_i)}{1 + exp(\beta_0 + \sum_{i=1}^{n} \beta_i X_i)}, \tag{4.16}$$

The right-hand-side of the equation is always between 0 and 1, making sure that the probability never lies outside this range.

Currency	Accuracy (%)	Annualised return (%)
GBP	51.90	2.42
EUR	49.37	1.59
JPY	49.37	1.12
CHF	54.43	3.06

Table 4.10: Test set prediction results for the four currency pairs with the benchmark model - logistic regression. All currency pairs are against USD. Note: transaction costs are not considered because the estimated annual transaction costs are negligible compared with the annualised returns. Refer to Table 4.9 of Subsection 4.5.10 for more details on the estimation of transaction costs.

Estimated coefficients of the logistic regression model for the four currency pairs are displayed in Tables 7.18 - 7.29 in the Appendix. Test set prediction results with logistic regression are displayed in Table 4.10. Both GBP/USD and USD/CHF generate accuracy over 50% while EUR/USD and USD/JPY generate lower than 50% accuracy rates. In terms of annualised return, USD/JPY pair has the lowest annualised return at 1.12% and

USD/CHF has the highest annualised return at 3.06%. As has been discussed in Section 4.4 (see Table 4.3), one main reason for the low annualised return of USD/JPY with the logistic regression is the large deviation of the test set from the training set in terms of momentum measured by percentage of price movement continuation.

Comparing the performance of logistic regression with MLPNN, MLPNN provides better forecasting results in terms of both accuracy (except USD/CHF for which logistic regression slightly outperforms MLPNN) and annualised return for all four currency pairs. Extra attention needs to be paid to the surprisingly good performance of MLPNN in forecasting USD/JPY, even with the large deviation of the test set from the training set in terms of momentum. This empirical result shows MLPNN is able to perform better forecasting than logistic regression even when the test data is very different from the training data, i.e. better at learning and adapting to newly-revealed features.

To compare this chapter with other research papers using ANN, Yao and Tan (2000) predicts GBP, JPY and CHF against USD. For GBP they obtain a percentage of correct directional change prediction at 54.74% and a 2.30% return. For JPY the percentage of correct directional change prediction is 53.40% and the return is 3.00%. Finally, for CHF the percentage of correct directional change is 56.00% and the return is 8.40%. According to Yao and Tan (2000), CHF generates the highest success rate with highest return. This differs from the result of this chapter and other chapters of this thesis where CHF is the most challenging currency to forecast and make profit on. Possible explanations for the difference include data frequency difference and difference in the period of interest. For example, some time periods include market interventions by the Swiss National Bank (i.e. the Swiss National Bank decided to drop the CHF/EUR limit of 1.2 in 2015) while other periods do not.

The performance of ANN on modelling other financial assets include Sezer and Ozbayoglu (2018) where their MLPNN model (similar to the model in this chapter but with

much fewer variables) achieve an average of 5.45% annualised return over 28 traded stocks compared with the RSI (5.01%), SMA (3.78%) and LSTM (6.48%).

4.5.12 Implications for EMH

In this chapter, the Jensen (1978) version of the EMH is tested,[104] i.e. utilising significant out-of-sample abnormal returns as a challenge to the EMH. In terms of the three forms of EMH as defined in Subsection 2.1.2, it is the semi-strong form EMH that is tested. This is because information used for forecasting not only includes historical FX rates, but also publicly available macro-economic indicators.

Empirical results of this chapter show that although all of the four currency pairs are from developed markets, some extent of (semi-strong form) market inefficiency does exist. This is suggested by the significant out-of-sample abnormal returns generated by the trained MLPNNs, see Table 4.8. There is not a dramatic difference in the level of (semi-strong form) inefficiency across the four markets. The Swiss market, indicated by the lowest abnormal return, shows the least extent of (semi-strong form) inefficiency. In brief, the results provide empirical evidence which challenges the semi-strong EMH for all four currency pairs.

[104] For more details of the Jensen (1978) version of the EMH, refer to Subsection 2.1.2.

4.6 Conclusion

In this chapter, large MLPNN models have been constructed and trained in order to perform modelling and prediction of FX rates. In order to test the semi-strong form of the EMH,[105] macro-economic indicators as well as historical FX rates are used as model inputs. Significant out-of-sample profits for all of the four currency pairs are present and therefore considered as evidence against the semi-strong form EMH, during the sample period of 20 years studied in this chapter.

Due to the large number of neurons hence a massive number of total parameters, traditional training algorithms with CPU computing can hardly complete the task in a reasonable time. However, with GPU computation built into the Keras library, the whole training and testing process is done within half an hour. GPU computation power is tested and compared with CPU computation power and one conclusion from this research is that although GPU computation has the slow-down problem as its maximum memory is reached it still stands as a better choice than CPU computation especially when conducting highly parallelisable tasks such as training models of large sizes. By comparing performance of models made up of different number of neurons, larger models show superiority in predicting complex patterns (such as the price pattern in the financial world) with as many inputs as we can possibly obtain. Research papers have shown that models perform better with more inputs (hence more information) entering them.

The problem, however, is that since MLPNN of this size has not been applied to financial series (at least not widely) such as FX rate before, the input variable selection process is challenging and time-consuming due to limited literature as suggestions and the large amount of candidate variables in the search pool.

Although this research has expanded the size of the MLPNN model used in financial time

[105] For more details on the EMH refer back to Subsection 2.1.2.

series by a large amount, more efforts are still needed to further expand the space for input variables as the 44 variables used in this research represent a significant yet small portion of the whole information space available in the market. As Hsu et al. (2016) discuss, substantial predictability and excess profitability can be achieved by technical analysis. This research result suggests the use of technical indicators as potential input variables of a large MLPNN model.

Another direction worthy of efforts is to make a step increment (in steps of 10 or 20) instead of an exponential increment of the number of neurons in the parameter tuning process. This will help boost the prediction performance by exploring more possibilities. However, this will certainly be at the cost of computation time and extra programming efforts.

One way to help facilitate the above direction of research is to deal with the GPU slow-down problem. The following three solutions would all help eliminate or at least reduce the slow-down problem: (1) to increase GPU memory by upgrading the GPU or adding more GPUs; (2) to upgrade or increase computer memory so that when GPU memory is filled up computer memory is used and the upgrade/increment would speed up data transfer rate therefore reduce slow-down; (3) to experiment with other programming languages to see if restarting (therefore refreshing GPU memory) is possible in the computation process. Among these three solutions, the first two are the most straight forward but financially expensive and even the upgrade is made it will still face the slow-down problem when the model to be trained gets large enough. While the final solution requires most effort (i.e. learning another language) and might still not work (for example in Python it is also impossible to restart within the process), the slow-down problem would be almost (if not completely) solved should it prove to be working.

In terms of the financial aspect, this chapter shows that the balance of the percentages of positive and negative FX rates movements affects the predictability of the currency

pair. The more unbalanced the percentages of positive and negative FX rates movements are, the more predictable (hence more profitable) the underlying currency pair is. One potential further research direction is to split the whole time span into different shorter periods and utilise higher frequency data to ensure a reasonable number of data points in each sub-sample. The predictability of FX rates in each sub-sample (representing periods under different economic conditions) can then be studied in detail.

Another factor that can possibly affect the predictability of a currency pair is whether this currency pair is an emerging market pair (at least one side of the currency pair is an emerging market currency) or a developed country pair (both sides of the currency pair are currencies from developed countries). Hsu et al. (2016) show that their emerging market currency pairs portfolio generates an annualised return of 7.32% while their developed country currency pairs portfolio only generates an annualised return of 1.35%. Since all currency pairs discussed in this chapter are developed country pairs, it raises the research direction of exploring emerging market pairs. One large difference between a developed country pair and an emerging market pair is liquidity.[106] Generally speaking, developed country currency pairs have higher liquidity than emerging market pairs, due to more significant amount of trading activities. It can be expected that more significant amount of trading activities reduces the number of, as well as shorten the lasting time of, trading opportunities due to mis-pricing. However, the extent to which these effects take place for different FX pairs during different time periods is worth further research.

Apart from predictability, profitability is another important aspect in this chapter since specifications of all models are optimised in order to maximise annualised return (and forecast accuracy). For simplicity, an one-step-ahead Buy-Low-Sell-High strategy is implemented. With this strategy, a Buy/Sell market order is placed if prediction for the

[106]In the FX market, liquidity refers to the extent to which one currency can be exchanged into another without causing a large change in the FX rate.

direction of FX rate movement next month is up/down and will be closed by the end of next month. However, if the strategy is used for speculation (profit making) rather than hedging (risk management), then it may sometimes close the order too early to profit from continuing price movement in the correctly predicted direction and sometimes close the order too late to stop at a reasonable loss in the case of a wrongly predicted price movement. Under those circumstances, more sophisticated trading strategies such as trend following and mean reversion can be implemented so that more promising trading performance can be achieved.

5 Using a LSTM-RSI trading algorithm for technical trading strategies in the foreign exchange market

5.1 Introduction

5.1.1 From forecasting to trading

In Chapter 4, the topic of financial forecasting is discussed and a large ANN model is built to forecast the direction of price movement of FX rates with the assistance of GPU in the training process. However, forecasting in general, is by no means the ultimate target by itself. One of such ultimate targets is maximising profits in trading (others include hedging, policy making and education) - the process of making decisions on whether to open/close a buy/sell order.[107] In this process, a set of pre-defined rules (called a trading strategy/rule) will need to be built such that it specifies explicitly different actions to be taken under different market conditions.

5.1.2 Technical trading

To help depict market conditions numerically, values derived from market prices called technical indicators are introduced. Using technical indicators to analyse the market and make trading decisions is called technical analysis and technical trading. Chang (2019), Gerritsen et al. (2019), Psaradellis et al. (2019), and Grobys et al. (2020) study the performance of technical trading rules in several markets including the FX market, the cryptocurrency market, and the crude oil market. In all of these papers, significant returns are obtained, and they therefore provide evidence of some extent of inefficiency

[107] In trading practice, an order can be placed at the current market price. This is called a market order. Traders sometimes also place an order with a pre-defined entry level and the order will be automatically triggered later only when the price hits the pre-defined level. This is called a limit order. In this chapter, only the market order is considered.

in the above markets. More details of literature on technical trading can be found in Section 5.2. While a technical trading rule generally makes a forecast on future prices, this forecast is usually more qualitative and can be difficult to quantify.

For example, a very popular technical trading rule is called Moving Average (MA). It calculates the average of the past i prices until the present. i is a configurable integer that can be modified to reflect the periodicity of the trading instrument, i.e. MA(5) on a daily chart shows the moving average of the price of an instrument for the past 5 trading days/one week. The MA line created by connecting consecutive moving averages helps to show the overall price trend. The MA trading rule uses the MA line as the indicator. It triggers a buy order when the market price just rises above the MA line and places a sell order when the market price just falls below the MA line.[108] As is discussed at the end of the previous paragraph, decision-making of the MA rule is not determined by a numerical forecast of future prices. Instead, a technical trading rule forms an overall belief of the price trend in the near future. This is different from, for example, using the forecast value of a large MLPNN model to trade FX rates in Chapter 4. Cases when technical trading rules succeed and fail are discussed as follows.

The MA trading rule holds a belief that when price breaks the trend line, it is a signal of a strong trend-following period in which the price will move in the same direction as the breakout. This belief makes an assumption on the market being "trend-following", i.e. a strong price increase is more likely to be followed by a further price increase and a strong price drop is more likely to be followed by a further price drop. However, problems could occur when the assumption on the trend-following market is violated. In fact, in almost all financial markets, there are periods of time when the market is not trend-following. In such times, trend-following trading rules usually suffer from great losses.

On the other hand, when a market is not trend-following, i.e. it is fluctuating up and

[108] In trading terminology, a scenario when a price penetrates a trend line is called a breakout.

down, such a price movement pattern is called "mean-reversion".[109] A typical technical trading rule based on the assumption of the market being mean-reverting is the Relative Strength Indicator (RSI).[110] The RSI indicator measures the relative price increase size to price decrease size. Bhargavi et al. (2017), Pedirappagari and Babu (2019), and Cohen (2020) have investigated trading strategies based on the RSI indicator to invest in the cryptocurrency market and the stock market.

A formal definition of the RSI indicator will be given in Section 5.3. In general, a high RSI shows that the market is experiencing many upward movements and a low RSI shows that the market is experiencing many downward movements. The RSI trading rule triggers a buy order when the RSI is high and places a sell order when the RSI is low. The assumption made by the RSI trading rule is that when the market has been going up for a long time (identified by a high RSI), it is likely to be overbought and a sell order should be placed in anticipation of the price falling, and vice versa for the low RSI case.

Similar to the trend-following based trading rules, the performance of mean-reversion based rules is also highly dependent upon the market condition. And once again, the market condition usually switches between trend-following and mean-reversion patterns rather than remaining in one of the patterns. Therefore, mean-reversion based trading rules may also suffer from significant losses in certain times. One solution to deal with the problem is to implement a forecasting model (which forecasts the state of the market in the near future, i.e. whether it's trend-following or mean-reverting) and form a trading rule based on this forecast.

---

[109] Trend-following and mean-reversion are two types of the overall price trend, which is a characteristic of price movement patterns. More details and literature on price movement patterns will be discussed in Subsection 5.2.2.

[110] Just like the term MA which can represent either the trading rule or the indicator, the term RSI can be used to represent either the RSI trading rule or the RSI indicator that the trading rule uses.

The problem, however, is that it is difficult to define specifically and differentiate with certainty between trend-following and mean-reversion patterns. This is because the concept of trend-following and mean-reversion is in the relative sense, i.e. there is no such case as absolute trend-following or mean-reversion. Instead, the price movements behave to a greater or lesser extent of being trend-following or mean-reversion.

This chapter utilises the RSI indicator as a measure of the extent of the market status being trend-following and mean-reversion. This is the first usage of the RSI indicator for this purpose, to the author's best knowledge. To forecast future RSI values, a forecasting model is needed. Subsection 5.1.3 discusses briefly the forecasting model used in this chapter and more details of the forecasting model are presented in Section 5.2 and Section 5.3.

5.1.3 Forecasting-trading algorithms

A forecasting-trading algorithm has a forecasting model implemented upon which a trading rule is based. The main difference between a forecasting-trading algorithm and a simple trading algorithm is that instead of using indicators together with assumption about the market (whether the market is trend-following or mean-reverting), a forecasting-trading algorithm utilises the implemented model to forecast information (price or price movement patterns) on the market in the future.

One advantage of using a forecasting-trading algorithm is that the performance of the algorithm does not rely on the assumption about the market being trend-following or mean-reverting. Instead, the performance of the forecasting-trading algorithm is highly dependent upon the the performance of the forecasting model. Sang and Pierro (2019) use a Long Short Term Memory (LSTM) model[111] to improve performance of

[111] A LSTM model is a type of ANN models that specialises in dealing with time series datasets. More technical details of this type of model will be provided in Section 5.2 and Section 5.3.

technical trading rules. In this chapter, the LSTM model is also used as the forecasting model because of its great capability of working with time series datasets. Similar to the MLPNN model as discussed in Chapter 4, LSTM models are implemented in the Keras-GPU framework as well, for faster training on large datasets.

5.1.4 Research questions

The research questions of this chapter are defined in the context of the modelling framework. They are (i) How does the trading performance of a forecasting-trading algorithm (such as the proposed LSTM-RSI) differ from a well-established, widely used trading algorithm (such as an MA or RSI)? (ii) Are trend-following and mean-reversion patterns related to forecasting horizon? (iii) How much variation is observed in price movement patterns for different currency pairs? (iv) What is the implication of the market status being trend-following or mean-reverting for forecastability/profitability of a given period?

Below is a brief preview of some of the research outcomes. A LSTM-RSI trading algorithm is built to first forecast the RSI value of the next period and then use training results to determine the best trading rule given the forecast RSI. Key results include (1) The proposed forecasting-trading algorithm, LSTM-RSI provides more stable profitability than the most widely used trading rules (MA and RSI) across time among different currency pairs. The LSTM-RSI algorithm has the advantage of controlling the number of trades therefore reducing transaction cost, over MA and RSI; (2) By working with four major currency pairs, GBP/USD, EUR/USD, USD/JPY and USD/CHF from 1999 to 2019, mean-reversion patterns exist more frequently in shorter time frames (e.g. on a weekly basis) while trend-following patterns appear more in longer terms (e.g. on a monthly basis); (3) USD/JPY and USD/CHF exhibit more mean-reversion patterns than GBP/USD and EUR/USD. USD/CHF is the most mean-reverting pair of all four pairs during the period of interest; (4) The forecastability therefore profitability of

the four currency pairs are higher for the pairs with higher level of trend-following pattens. Forecastability and profitability are lower for the pairs with higher level of mean-reversion patterns; (5) None of the four currency pairs generate consistent abnormal profits, especially after transaction costs are deducted. This empirical evidence supports the weak form EMH.

The remainder of the chapter is organised as follows. Section 5.2 reviews literature on technical trading, price movement patterns, and the LSTM model applied for technical trading. Section 5.3 describes the methodology used in this chapter including using RSI to measure price movement patterns and using LSTM to forecast RSI values. Section 5.4 discusses the source of data used in this chapter, the process of splitting the training/test sets and preparation of the dataset as LSTM inputs and outputs. Section 5.5 contains the main modelling and trading process. Section 5.6 concludes the chapter and suggests potential future research directions.

5.2 Literature review

5.2.1 Technical trading and trading rules

Technical trading is a combinatorial use of technical indicators and technical trading rules.[112] A number of recent research papers on technical indicators and technical trading rules are presented below.

The question of whether technical trading generates significant profits in financial markets has been debatable in the academic world. Chang (2019) utilises a variable-length moving averages (VMA) to trade the daily exchange rates of New Taiwan Dollars (NTD) against USD. The VMA has the advantage of flexible length of moving average period over a simple MA. The results show that the VMA significantly outperforms the buy and hold strategy.

In addition to the FX market, other financial markets have also been explored with technical trading rules. Psaradellis et al. (2019) investigate different types of trading rules on the crude oil market. Their results suggest that contrary to the in-sample outstanding results, no persistent and significant profits can be obtained with the technical trading rules.

A different conclusion is drawn for another market - the Cryptocurrency market - by Gerritsen et al. (2019).[113] They analyse the performance of MA, RSI, MACD and trading range breakout strategies on Bitcoin prices. Significant profits can be made with the trading range breakout strategy. Their findings provide evidence against market efficiency of Bitcoin.

A similar paper on testing technical trading performance on the Bitcoin market is Cohen

[112] For more description on technical trading refer back to Subsection 2.1.4.

[113] Since the price surge of Bitcoin (and other Cryptocurrencies) in 2017, it has drawn the most attention from the academic world based on the number of recent publications.

(2020). He explores three types of trading strategies - RSI, MACD and pivot reversal (PR). His results demonstrate that the RSI produces the poorest results of the three strategies and both MACD and PR generate greater profits than the buy and hold strategy.

Despite the poor performance of the RSI strategy in the Bitcoin market, Bhargavi et al. (2017) provide evidence of good performance of RSI in the stock market. An important contribution of their research is that instead of using RSI to trade according to the traditional rule they use the RSI as a filter to select stocks to form a portfolio that contains undervalued (overpriced) stocks with which buy (sell) decisions can be made. Therefore, the usage of an indicator may not be restricted to the traditional usage for which the indicator was initially designed.

Although all the above papers investigate performance of technical trading rules, their main focus is on whether and how much a trading rule makes profits for a financial instrument in a given period. The questions of, why one trading rule makes profits while the other rule suffers from losses, why a trading rule performs much better in certain periods than in other times, and why even the same trading rule's performances differ significantly with respect to different financial instruments, have not been explored in these papers.

5.2.2 Price movement patterns

The key cause for the variations in performances as described in the previous paragraph is the difference in price movement patterns. Strictly speaking, the same pattern almost never happens again because not only the direction of price movement but also the value of price movement makes the price movement pattern of every period unique. In reality, it is not necessary nor possible to depict every single characteristic of the price movement pattern within a period. Therefore, more crucial characteristics should deserve more attention. Overall trend is one of those more crucial characteristics. The

trend characteristic can be classified into two types, trend-following (momentum) and mean-reversion.

Two directions of description on trend-following/mean-reversion patterns are (1) statistically testing the existence of momentum/mean-reversion patterns; and (2) determining trend patterns with performances of trend-following/mean-reversion based trading rules.

For the first direction, there are much more papers testing mean-reversion patterns than trend-following pattern. The main reason for this imbalance is that there are more statistical tools to test mean-reversion than trend-following patterns. The most widely used test is the Unit Root test, e.g. augmented Dicky-Fuller test, Phillips-Perron test, KPSS test and others. Testing a trend-following pattern, however, is much more difficult because of the uncertainty in the speed of price increase/decrease, i.e. even if we can visually tell a price series is in an up-trend, it's difficult to test whether it's a linear up-trend, a quadratic up-trend or some other types of increase that cannot even be mathematically modelled.

Mukherji (2011) indicates that significant mean-reversion patterns exist in the US stock market from 1926 to 2007. Although the mean-reversion pattern weakens in the past decade prior to 2007, it persists for small company stocks. Despite studying a different asset class from this chapter, his paper suggests that price trend patterns vary, for different periods as well as for different sub-classes, i.e. stocks of different companies.

Taylor et al. (2001) imply that real dollar FX rates over the post-Bretton Woods period not only exhibits mean-reversion patterns but also non-linear mean-reversion patterns. Their research result supports the use of non-linear models to forecast FX price patterns.

In addition to statistically testing trend patterns, researchers have also implemented different trading rules (trend-following trading rules or mean-reversion trading rules) for several asset classes. The performance of the implemented trading rule reflects

the trend pattern in the underlying period. For example, if using a trend-following strategy generates higher returns than using a mean-reversion strategy, this suggests the underlying period exhibits more trend-following patterns than mean-reversion patterns, and vice versa.

Szakmary et al. (2010) use trend-following strategies to trade the commodity futures market from 1996 to 2007. They find that the MA strategies with all MA lengths yield positive excess returns in 22 out of the 28 markets. Another paper on the commodity futures market is Lubnau and Todorova (2015). They implement mean-reversion strategies on Crude Oil, Natural Gas, Gasoline and Heating Oil futures from 1992 to 2013. Their mean-reversion strategies, with appropriate strategy parameters, also gain significant profits in the given time.

Since the market does not always exhibit trend-following or mean-reversion patterns (instead the two patterns usually happen interchangeably), there are papers that combine the two types of trend-following and mean-reversion strategies. Both Serban (2010) and Wu (2011) suggest a strategy that switches between a trend-following strategy and a mean-reversion strategy. The strategy by Serban (2010) generates significant abnormal returns for both the stock market and the FX market. The combined strategy by Wu (2011) generates greater profits in the stock market than either the trend-following or mean-reversion strategy implemented alone.

Not only do time period and asset classes, but also time horizon affects price trend patterns. Raza et al. (2014) investigate whether momentum (trend-following) or reversal pattern is dominant in short horizon (1-4 weeks) FX rate returns.Their empirical results show that momentum based strategies generate much larger profits than reversal based strategies, based on a sample of 63 emerging and developed market currency pairs. This result suggests that the momentum pattern is more prevalent than the reversal pattern on a 1-4 weeks horizon.

All the previous trading rules in Subsections 5.2.1 and 5.2.2 base their forecast of the future price on technical indicators, i.e. MA and RSI. However, indicators such as MA are lagging, as suggested by Ellis and Parbery (2005). For example, a typical MA line goes up by a significant amount only after the underlying asset price has already increased for some time. Another approach of setting up a trading strategy is to base the trading rule on the forecast results given by a forecasting model. The advantage of this approach is that non-lagging indicators can be used as inputs of the forecasting model in order to make a precise forecast about the future price/trend of an asset.

5.2.3 LSTM as a forecasting model for technical trading

Within the universe of forecasting models, Artificial Neural Networks (ANN) are one of the most widely used and extensively researched models. As is discussed in Chapter 4, ANN has the advantage of being highly adaptive, able to model non-linear patterns and fast to train with the rapid-developing GPU computation techniques. The Multi-Layer Perceptron Neural Network (MLPNN) model used in Chapter 4 is good at forecasting non-time series datasets with multiple explanatory variables (multi-dimensional), e.g. GDP, interest rates, CPI. In this chapter, price movement patterns which are single-dimensional time series, are the focus. Therefore, it would be advantageous to use an ANN capable of working with time series datasets to forecast price movement patterns.

The LSTM model, proposed by Hochreiter and Schmidhuber (1997), is one of the ANN models that takes time series data as input variables. It has the capability of storing short-term sequential information and retrieving the information many time steps later. In this subsection, literature on LSTM applied in a trading context will be explored.[114]

Although there is a wide range of literature utilising ANN models to forecast financial asset prices (as is discussed in Section 4.2), very few research papers focus on ANN

[114]More technical details of the LSTM model are given in Section 5.3.

models for the purpose of technical trading.

Sang and Pierro (2019) apply LSTM to help trade in the US stock market (stocks are divided into 9 sectors). They use the LSTM model as a filter of trading decisions made by technical trading rules (MA, RSI and MACD), i.e. place a buy order only when the MA strategy suggests a buy and the LSTM model also forecasts future price to increase. By implementing the LSTM model, a trading performance improvement (over the trading rules applied without the LSTM filtration) is achieved for almost all sectors for all three trading rules.

The number of papers on LSTM models for the purpose of technical trading within the FX market is very limited as LSTM has been introduced to the financial world for a relatively short time. This brings motivation as well as novelty of implementing LSTM in the FX market for technical trading purposes.

To present a brief summary of the literature, the popularity of ANN models in financial asset price forecasting has increased over the past few years. However, ANN models have not been extensively applied in the area of technical trading, i.e. very few papers have focused on forecasting-trading algorithms. Therefore, this chapter will explore forecasting-trading algorithms in the FX market. Section 5.3 will discuss details of the measurement of price patterns, the forecasting model and performance metrics.

5.3 Methodology

In this section, a brief work-flow of the modelling and trading process is discussed, based on the different parts of a trading algorithm. Both the trading rule and the forecasting model (LSTM) are discussed. More details on the training process of LSTM are presented in Section 5.5.

A traditional trading algorithm takes one or more indicators such as MA and RSI and uses a set of trading rules based on the (up-till-current) values of the indicators to suggest potential entrance or exit price levels. As is discussed in Section 5.1 and 5.2, these algorithms lack the ability to adjust to the current market price pattern and may suffer great losses from a change in the market price pattern. For example, a trend-following trading algorithm (such as MA) performs poorly in a period with significant mean-reversion patterns. With these traditional algorithms, it is not possible to foresee the upcoming poor performance until a loss has occurred.

This is why a trading algorithm with more powerful forecasting ability is desired. In this chapter, such algorithms are termed forecasting-trading algorithms. A forecasting-trading algorithm consists of two components, a forecasting component and a trading component. The forecasting component builds up (trains and tests) a model that takes past data (or metrics derived from past data) as inputs and delivers output values as trading indicators. The trading component pre-defines a set of trading rules (criteria to trigger or close a buy/sell order) based on the trading indicators obtained from the forecasting component.

5.3.1 Measurement of price movement patterns

In this chapter, I utilise a LSTM model as the modelling component, to forecast future price movement patterns based on past price movement patterns. In order to quantify price movement patterns, this chapter proposes the use of RSI as a measure of movement

patterns. This is different from previous research papers, all of which use RSI as the indicator for making trading decisions. The RSI is defined as:

$$RSI_n = 100 - \frac{100}{1 + \frac{AG_n}{AL_n}}, \tag{5.1}$$

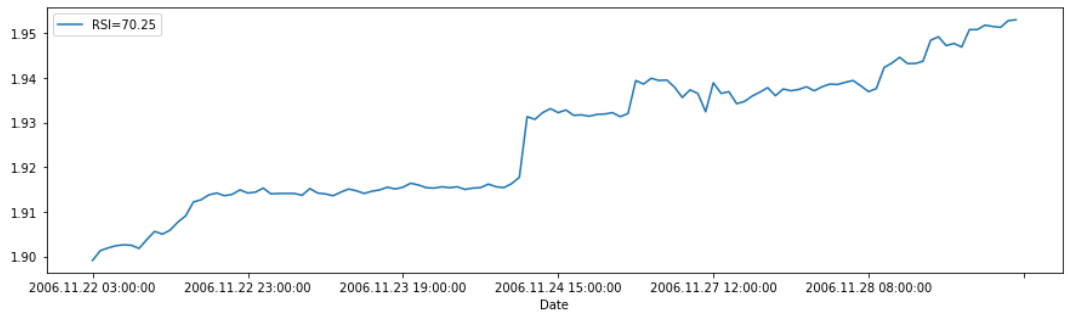
where RSI_n represents the RSI value of a period lasting n units of time. AG_n measures the average gain over the period lasting n units of time and AL_n calculates the average loss over the same period.

To help visualise how price movement patterns are closely related to the RSI values, Figures 5.1 - 5.3 show 9 randomly selected sub-periods of GBP/USD with three levels of RSI: above 70, below 30 and around 50.[115] It can be clearly observed that all sub-periods with high RSI values follow an up-trend price movement, all sub-periods with low RSI values follow an down-trend price movement and all sub-periods with medium RSI values move sideways and exhibit much greater fluctuation patterns.

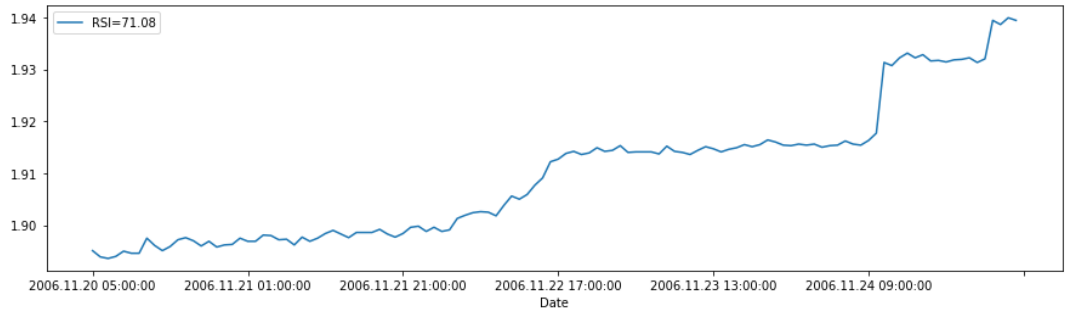
To calculate a RSI value, a sub-period needs to be specified. In this chapter, the dataset (with a large number of data points) will be split into sub-periods of equal length. The RSI for each sub-period will then be calculated and used as the inputs and outputs of the LSTM model, i.e. RSI values of a number of past sub-periods are used as inputs to forecast a RSI value (output of the LSTM model) of a future sub-period.

After a forecast RSI value for the next sub-period is obtained, and depending on its value, a trading decision to open/close a buy/sell order can be made.

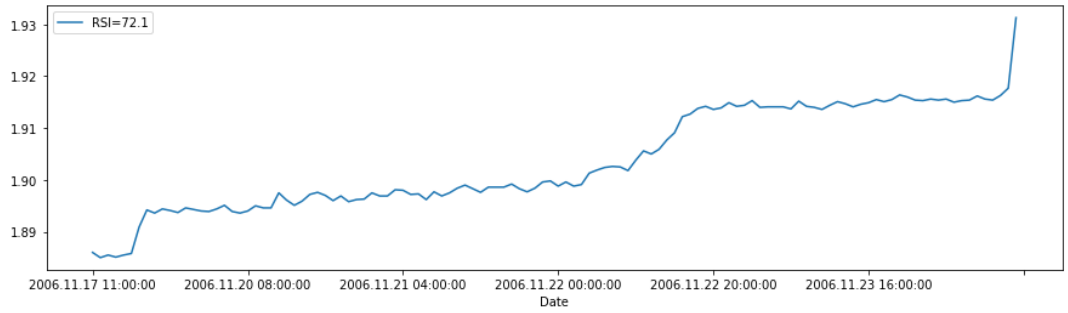
[115]Traditionally, a RSI value greater than 70 is recognised as an indication of the financial asset being overbought and a RSI value less than 30 is considered as an indication of the financial asset being oversold.



(a) A randomly selected period with a large RSI value (>70). Currency pair: hourly GBP/USD.

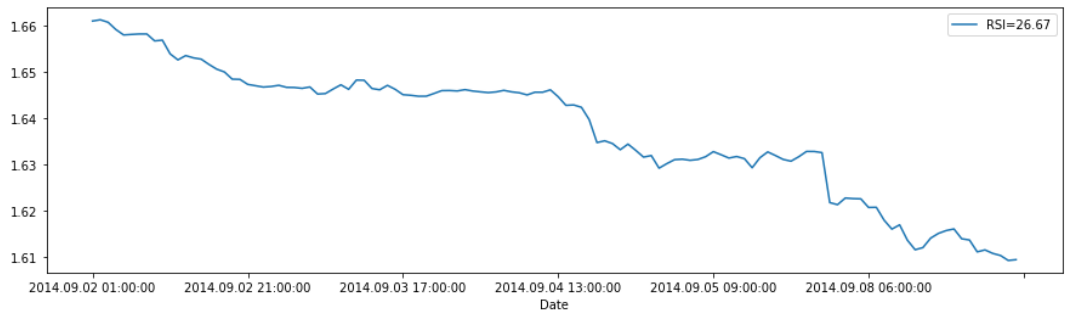


(b) A randomly selected period with a large RSI value (>70). Currency pair: hourly GBP/USD.

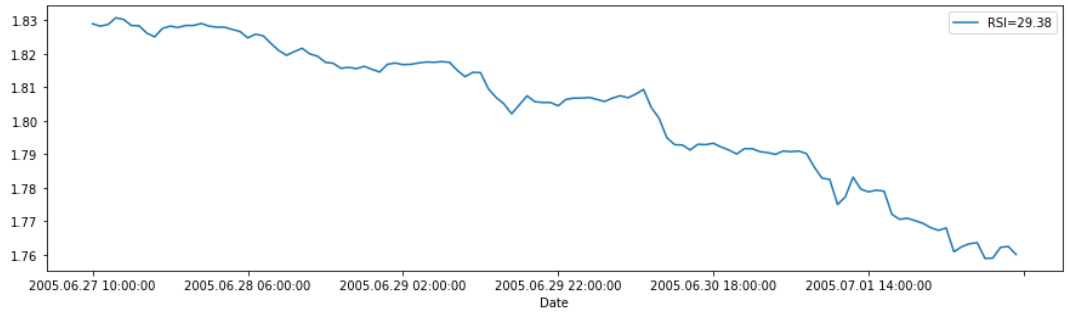


(c) A randomly selected period with a large RSI value (>70). Currency pair: hourly GBP/USD.

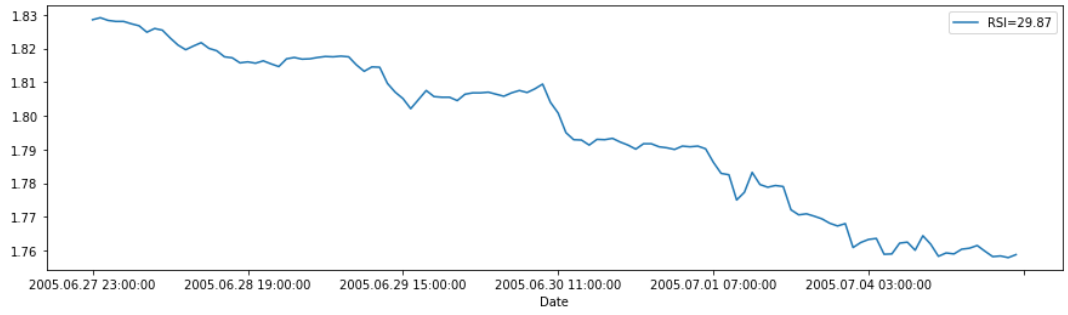
Figure 5.1: Three randomly selected periods with large RSI (>70) values. Currency pair: hourly GBP/USD



(a) A randomly selected period with a small RSI value (<30). Currency pair: hourly GBP/USD.

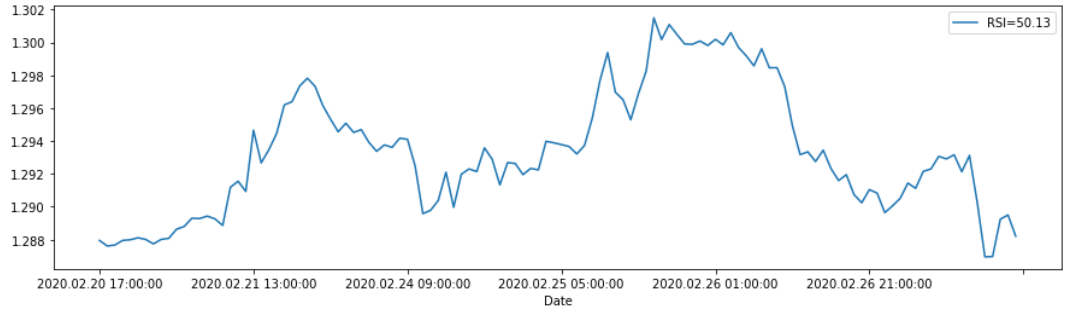


(b) A randomly selected period with a small RSI value (<30). Currency pair: hourly GBP/USD.

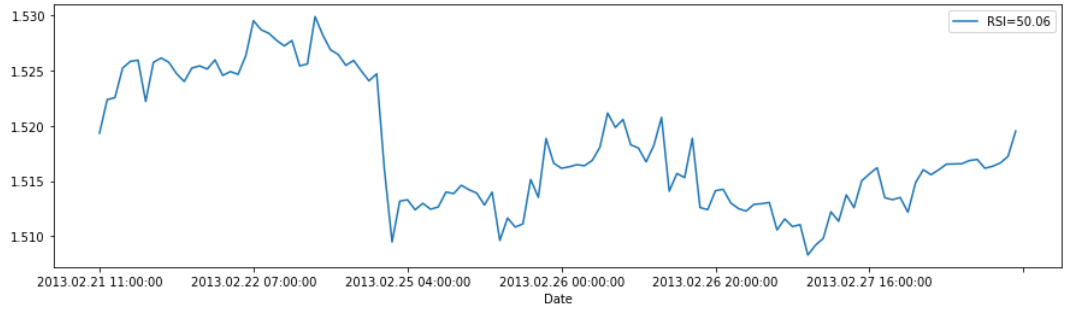


(c) A randomly selected period with a small RSI value (<30). Currency pair: hourly GBP/USD.

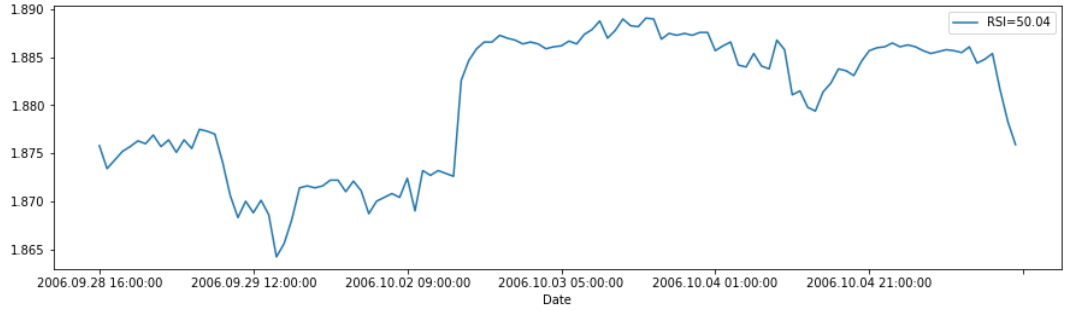
Figure 5.2: Three randomly selected periods with small RSI (<30) values. Currency pair: hourly GBP/USD



(a) A randomly selected period with a medium RSI value (around 50). Currency pair: hourly GBP/USD.



(b) A randomly selected period with a medium RSI value (around 50). Currency pair: hourly GBP/USD.



(c) A randomly selected period with a medium RSI value (around 50). Currency pair: hourly GBP/USD.

Figure 5.3: Randomly selected periods with medium RSI (around 50) values. Currency pair: hourly GBP/USD

5.3.2 LSTM to forecast price movement patterns

The LSTM model is implemented after the RSI value (which measures the trend patterns) of each sub-period is calculated. To formulate a LSTM structure, following the symbol usage from Zhang et al. (2017), symbols are denoted and defined as below.

At each time t, the LSTM has an input vector (e.g. RSI values of sub-periods) $\mathbf{x_t}$, a memory state vector $\mathbf{c_t}$, a hidden state vector $\mathbf{h_t}$ and the output vector $\mathbf{o_t}$. Each of these vectors is defined as:

$$\mathbf{i_t} = \sigma(\mathbf{W_i x_t} + \mathbf{U_i h_{t-1}} + \mathbf{b_i}), \tag{5.2}$$

$$\mathbf{f_t} = \sigma(\mathbf{W_f x_t} + \mathbf{U_f h_{t-1}} + \mathbf{b_f}), \tag{5.3}$$

$$\tilde{\mathbf{c}}_\mathbf{t} = tanh(\mathbf{W_c x_t} + \mathbf{U_c h_{t-1}} + \mathbf{b_c}), \tag{5.4}$$

$$\mathbf{c_t} = \mathbf{i_t} \circ \tilde{\mathbf{c}}_\mathbf{t} \tag{5.5}$$

$$\mathbf{o_t} = \sigma(\mathbf{W_o x_t} + \mathbf{U_o h_{t-1}} + \mathbf{V_o c_t} + \mathbf{b_o}), \tag{5.6}$$

$$\mathbf{h_t} = \mathbf{o_t} \circ tanh(\mathbf{c_t}), \tag{5.7}$$

where $\mathbf{W}_*$ and $\mathbf{U}_*$ are the weight matrices and $\mathbf{b}_*$ are the bias vectors (all of the weight matrices and bias vectors to be estimated in the training process). The $\sigma()$ represents the sigmoid function, as is defined in Subsection 2.2.2. $\mathbf{i_t}$, $\mathbf{f_t}$ and $\mathbf{o_t}$ are the input gate, forget gate and output gate respectively. The input modulation $\tilde{\mathbf{c}}_\mathbf{t}$ and output $\mathbf{h_t}$ take the hyperbolic tangent function $tanh()$ as the activation function. The $\circ$ sign denotes point-wise multiplication.

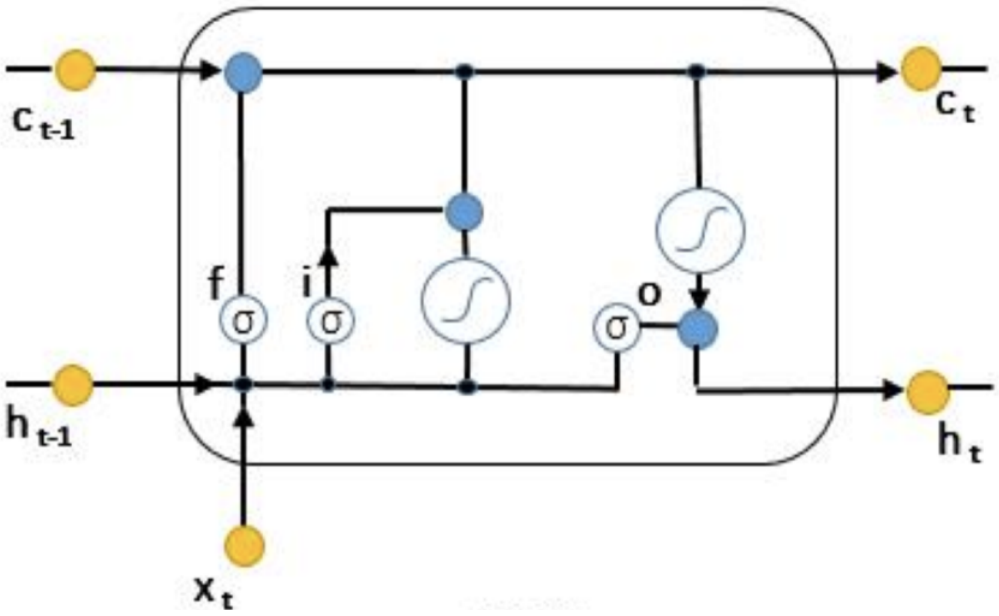


Figure 5.4: A plot showing the structure of a LSTM model. Source of graph: Zhang et al. (2017).

The input gate controls information (e.g. RSI values of sub-periods) that enters the model and the output gate allows processed information (forecast RSI value for the next period) to form the output. The forget gate decides what information is kept in the model and what information should be (temporarily) discarded. Therefore the LSTM model has the ability to balance long and short term memories of a time series.

The technical details of how LSTM works are not discussed in detail because the purpose of discussing LSTM in this subsection is to use it as a tool for forecasting in later sections, rather than developing or improving LSTM as a methodology.

5.3.3 Performance metrics

To measure how a forecasting-trading algorithm performs, two performance metrics (one for each of the forecasting and trading component) are utilised. (1) To evaluate the performance of the LSTM model the Mean Squared Error (MSE) is used.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2, \tag{5.8}$$

where n is the total number of forecasts made, Y_i is the real value, and $\hat{Y}_i$ is the forecast value.

(2) To measure the trading performance, the Annualised Return (AR, in %) is used.

$$AR = \Big(\big((1+r_1)(1+r_2)...(1+r_n)\big)^{1/n} - 1\Big) \times 100, \tag{5.9}$$

where r_i represents the return of the i'th period and n is the number of periods.

5.4 Data description

5.4.1 Data source

Four currency pairs are researched in this chapter, namely GBP/USD, EUR/USD, USD/JPY and USD/CHF. Similar to Chapter 4, only the major pairs are considered because they provide a longer time period of high frequency data availability.[116]

The datasets for the four currency pairs (close price) are collected at an hourly frequency from January 1999 to April 2020. The source of the data is the MetaQuotes Software Corp. history centre. Data is accessible and downloadable from an MetaTrader 4 platform supported by most FX brokers.

The reason for using hourly data is because it provides a sufficient number of data points for training the LSTM model within a reasonable amount of time. With lower frequency data such as daily data, the number of data points is insufficient to train the model and with higher frequency data such as minute data, the computation time would be too long.

5.4.2 Split of training, test and validation sets

With the same approach of splitting training/test sets as the previous two empirical chapters, a 2:1 training/test ratio is adopted. A further 20% of observations from the training set is used as a validation set to reduce overfitting.

The training/test set split is useful in both components of the forecasting-trading algorithm. For the forecasting component, the training set is used to train the LSTM model (i.e. optimise the parameters of the model) and afterwards for the trading component, the training set is used to configure the trading rule[117] to generate best trading performance.

[116] From MetaQuotes Software Corp. history centre, hourly data of the above four major pairs are available from 1999. However, for minor pairs, hourly data is only available from the 2010s.

[117] Refer to Subsection 5.5.2 for more discussion of the trading rules.

5.4.3 Preparation of the raw data as LSTM inputs and outputs

As is described in Section 5.3, the dataset needs to be spilt into sub-periods in order to calculate RSI values. The length of each sub-period is an important factor to consider. In this chapter, four different lengths are considered, 120, 240 360 and 480. Given that the data frequency is hourly and FX markets are usually open 24 hours per day and 5 days per week. The four lengths last 1 week, 2 weeks, 3 weeks and 4 weeks, respectively. Shorter lengths are not considered because it would make the number of observations per sub-period too small to observe significant price movement patterns. On the other hand, the length of each sub-period is also restricted by an upper bound because a longer length leads to a smaller number of sub-periods available (each of which forms a potential input neuron for the LSTM model), i.e. for the maximum length of 480 hours used in this chapter, the number of sub-periods is around 274, making it 274 individual data points for the LSTM model as inputs. As the length increases, the number of sub-periods becomes so small that the trained LSTM may fail to generalise.

After the sub-periods division, the RSI value, which describes the price movement pattern of the underlying period, is calculated for each sub-period. These RSI values form the input dataset for the LSTM to train. To forecast the RSI value for the next sub-period, the RSI values of previous sub-periods are used as inputs of the LSTM. A range of numbers of look-back sub-periods are considered, from 1 to 10. This look-back number essentially represents the input size of the LSTM model, i.e. how long in the past does the model take to forecast the future.

5.5 Modelling and forecasting

5.5.1 The iterative training process

Given one combination of the sub-period length and number of look-back sub-periods, the best LSTM model can be obtained by iteratively training on the training set. The concept of iterative training is the main topic of Chapter 3 and is also introduced in Chapter 4. To briefly remind the reader of the concept, the training process generally involves taking random initial values of the parameters and improve the performance metric by adjusting the initial values. By repeating the training process multiple times (instead of taking the first training result), it is almost sure that an improved model can be obtained.

While it is true that improved performance will almost always be achieved as the number of iterations increases, the time it takes for a new improvement to be made tends to increase over time, i.e. the marginal time cost increases. Therefore it is important to find an optimal number of iterations. The word "optimal" means it should aim for a good balance between performance improvement and computation time.

An experiment is conducted on the GBP/USD pair for the 40 combinations of length of sub-periods and number of look-back sub-periods, in order to find an optimal number of iterations. In the experiment, a maximum of 100 iterations are carried out.[118] Table 5.1 shows the minimum number of iterations needed for the model to achieve at least 99% of the performance improvement (in terms of MSE) out of the 100 iterations.[119] None

[118] It takes nearly 72 hours to run 100 iterations for the 40 combinations of settings for a single currency pair. Therefore, the experiment is restricted to one currency pair only.

[119] As a hypothetical illustration, suppose that after 100 iterations, the decrease of MSE is 0.01. If a decrease of MSE by 0.0099 (99% of 0.01) is achieved right after, for example, 70 iterations, then it can be said that using 70 iterations will generate a 99% improvement of the total improvement made by 100 iterations.

of the combinations of settings require more than 10 iterations to achieve a satisfactory percentage of performance improvement.

Therefore, in this chapter, the number of iterations is set to be 20 for all four currency pairs. The number is increased from 10 to 20 to allow for possible deviation among different currency pairs, i.e. some currency pairs might require more iterations to obtain optimal performance than others.

	120	240	360	480
1	8	6	5	5
2	8	6	5	4
3	7	6	5	4
4	7	5	4	4
5	7	5	4	4
6	7	5	4	4
7	7	5	4	4
8	7	5	4	4
9	7	5	4	4
10	7	5	4	4

Table 5.1: A table showing the minimum number of iterations needed for the model to achieve at least 99% of the performance improvement (in terms of MSE) out of 100 iterations for different sub-period length and number of look-back sub-periods combinations. Currency pair: GBP/USD.

5.5.2 Trading rules

Based the forecasting value of a future RSI, decisions of whether to open/close a buy/sell order can be made to maximise trading profits. As a result of the RSI nature, i.e. lower (than 50) RSI values usually accompany down-trends and higher (than 50) RSI values usually represent up-trends, the trading rule associated with this model, is set to be opening a sell order (and closing an existing buy order) when the forecast RSI value is below a certain value (called a "Low RSI") and placing a buy order (and closing an existing sell order) when the forecast RSI is above a certain value (called a "High RSI").

Empirical analysis is conducted on the training set for a Low RSI ranging from 30 to 50 and a High RSI from 50 to 70 to find the best RSI trading rule for each of the 40 "sub-period length & number of look-backs" combinations. The RSI value 50 overlaps because it is possible for both Low RSI and High RSI to be 50. In this case, a buy order is placed (existing sell order closed) for higher-than-50 forecast RSI and a sell order is placed (existing buy order closed) for lower-than-50 forecast RSI.

The gap in values between the Low RSI and the High RSI is an important indicator of whether the market is trending or mean-reverting. In the above example when the Low RSI and High RSI overlaps (at 50), orders are constantly placed as long as the Low and High RSIs are not equal to 50. This represents a trending period (upward or downward) in which trading is strongly encouraged to maximise trading profits. On the other hand, when the Low RSI and High RSI have a large gap in values, i.e. very small Low RSI and very large High RSI, trading is discouraged with the strict trade-entering criteria. This type of trading rule (i.e. a large-gap between Low and High RSIs) typically appears within a fluctuating period. In such a period, the market direction is usually indecisive. No trade will take place unless there is evidence of a strong trend, indicated by an extremely high or low forecast RSI value.

5.5.3 Transaction costs

As is discussed in Chapter 4, the effect of transaction costs is largely dependent upon the frequency of trading. In this chapter, since data frequency has been increased from monthly to hourly data, trading frequency is also likely to increase. Therefore, the average annual transaction cost is expected to increase. In Chapter 4, the number of trades is fixed at once per month because in each month the simple Buy-Low-Sell-High trading rule always makes either a buy or sell order. However, in Chapter 5, the number of trades depends on the trading decisions made by the trading rule (LSTM-RSI and the two benchmarks MA and RSI), which varies across different periods, as illustrated in the next paragraph. Therefore, the annual transaction cost cannot be estimated by multiplying the number of trades placed per year by the average percentage transaction cost per trade as in Chapter 4.[120] Instead the transaction cost is estimated on a trade-by-trade basis, i.e. each time the underlying trading strategy places a trade, the average percentage transaction cost (as estimated in Subsection 4.5.10 of Chapter 4) is deducted.

As an illustrative example, suppose that transaction cost is to be estimated in the process of trading the USD/JPY pair with the best in-sample specification for LSTM-RSI being [3, 120, 50, 52]. The specification [3, 120, 50, 52] means that the past 3 sub-periods (each lasting 120 hours) are used to forecast the RSI value of the next sub-period. If the forecast RSI is greater than 52, open a buy order and if the forecast RSI if less than 50, open a sell order. If the forecast RSI is between 50 and 52, no trade is placed. Based on the above information, for each sub-period, the algorithm may or may not instigate an order. Therefore, as is discussed in the previous paragraph, it is not possible to estimate the (annual) transaction cost by multiplying the number of trades placed per year by the average percentage transaction cost per trade. Given this circumstance, the calculation

[120]The average percentage transaction cost is calculated with the spread values from OANDA as in Table 4.9 of Subsection 4.5.10.

of transaction cost is programmed into the trading algorithm such that whenever the algorithm instigates a trade (i.e. forecast RSI is above 52 or below 50 for USD/JPY as an example), the estimated transaction cost per trade (0.012% for USD/JPY as estimated in Table 4.9 of Subsection 4.5.10) is deducted, and when no trade is opened (i.e. forecast RSI is between 50 and 52 for USD/JPY), no transaction cost is incurred. Transaction costs are estimated in a similar way for the benchmark strategies, MA and RSI, both of which place a varied number of trades from year to year.

5.5.4 Training/test results and interpretation

In Tables 5.3, 5.6, 5.9 and 5.12, the best (in terms of annualised return) RSI rules are displayed for each of the 40 combinations of settings. Table 5.2 summarises the mean RSI differences between the Low RSI and the High RSI for the four currency pairs in different lengths of forecasting periods.

From Table 5.2, it can be seen that across different lengths of forecasting periods, the gap between the Low RSI and the High RSI almost always decreases as the length of forecasting period increases. This provides evidence to answer the question of whether the market is in trend-following or mean-reversion status: in the short term (i.e. on a weekly basis), the four FX pairs tend to be mean-reverting while in the long term (i.e. on a monthly basis), the four FX pairs tend to be trend-following.

To compare across different currency pairs, the USD/JPY and USD/CHF pairs have larger RSI gaps than the other two pairs, meaning that USD/JPY and USD/CHF are more mean-reverting than GBP/USD and EUR/USD. Especially for the USD/CHF pair, which has almost twice the RSI gap values of the second largest RSI gap pair - USD/JPY. Out of the four pairs, the EUR/USD pair has the smallest and most stable RSI gap values, making it the most trend-following pair in all time frames.

	120	240	360	480
GBP/USD	7	2.4	3.5	1.5
EUR/USD	2.8	2	1.9	1.3
USD/JPY	8.8	4.2	3.4	2.5
USD/CHF	16	9.7	7.7	3.6

Table 5.2: A table showing the mean RSI differences between the Low RSI and the High RSI for the four currency pairs in different lengths of forecasting periods.

	120	240	360	480
1	[48, 65]	[50, 56]	[50, 58]	[48, 51]
2	[50, 59]	[50, 56]	[48, 53]	[49, 51]
3	[50, 56]	[50, 55]	[48, 52]	[49, 52]
4	[50, 58]	[50, 52]	[49, 53]	[49, 52]
5	[50, 55]	[50, 52]	[49, 52]	[49, 52]
6	[50, 54]	[50, 50]	[49, 52]	[50, 50]
7	[50, 53]	[50, 50]	[49, 54]	[50, 50]
8	[49, 55]	[50, 50]	[49, 51]	[49, 51]
9	[49, 56]	[49, 51]	[49, 50]	[50, 50]
10	[48, 53]	[49, 50]	[50, 50]	[50, 50]
Mean-Diff	7	2.4	3.5	1.5

Table 5.3: A table showing the best (in terms of annualised return) in-sample trading rules with 10 iterative runs for different period lengths and numbers of look-back periods. Mean-Diff is calculated as the mean of the differences between the High RSI and the Low RSI of each column. Currency pair: GBP/USD.

	120	240	360	480	R-Mean
1	2.94	1.35	1.15	-0.34	1.28
2	2.26	0.56	0.85	1.31	1.24
3	2.14	1.12	2.24	3.68	2.30
4	2.89	2.06	1.61	2.46	2.25
5	1.68	2.58	2.04	3.19	2.37
6	2.35	5.04	1.90	3.66	3.24
7	1.58	7.93	1.71	0.90	3.03
8	2.36	5.12	3.25	3.11	3.46
9	2.00	3.69	2.20	1.38	2.32
10	1.10	5.98	3.92	3.50	3.62
C-Mean	2.13	3.54	2.09	2.28	

Table 5.4: A table showing the highest in-sample annualised returns (%) for different period lengths and numbers of look-back periods. "R-Mean" calculates the mean of each row and "C-Mean" calculates the mean of each column. Currency pair: GBP/USD.

	120	240	360	480	R-Mean
1	-2.28	1.50	-0.41	2.58	0.35
2	-1.18	1.61	1.51	3.64	1.39
3	-0.12	0.45	0.69	1.94	0.74
4	-0.54	2.59	0.88	2.01	1.23
5	-0.83	1.35	2.23	1.69	1.11
6	-1.09	-0.58	1.71	1.77	0.45
7	-1.20	0.94	1.96	3.79	1.37
8	-0.75	3.52	2.16	1.74	1.67
9	-0.49	1.89	0.60	3.55	1.39
10	0.14	0.01	1.33	2.90	1.10
C-Mean	-0.84	1.33	1.27	2.56	

Table 5.5: A table showing the highest test annualised returns (%) for different period lengths and numbers of look-back periods. "R-Mean" calculates the mean of each row and "C-Mean" calculates the mean of each column. Currency pair: GBP/USD.

	120	240	360	480
1	[43, 54]	[50, 50]	[49, 53]	[49, 51]
2	[50, 56]	[49, 53]	[49, 50]	[47, 51]
3	[49, 50]	[49, 51]	[49, 50]	[50, 50]
4	[49, 50]	[50, 50]	[50, 52]	[50, 50]
5	[50, 50]	[49, 50]	[49, 52]	[50, 50]
6	[49, 50]	[49, 51]	[49, 52]	[50, 50]
7	[49, 54]	[48, 51]	[50, 50]	[50, 51]
8	[49, 53]	[49, 51]	[49, 51]	[49, 52]
9	[49, 50]	[48, 51]	[50, 50]	[50, 51]
10	[50, 50]	[48, 52]	[49, 52]	[49, 51]
Mean-Diff	2.8	2	1.9	1.3

Table 5.6: A table showing the best (in terms of annualised return) in-sample trading rules with 10 iterative runs for different period lengths and numbers of look-back periods. Mean-Diff is calculated as the mean of the differences between the High RSI and the Low RSI of each column. Currency pair: EUR/USD.

	120	240	360	480	R-Mean
1	0.53	5.30	3.63	2.52	2.99
2	1.89	3.65	5.24	2.67	3.36
3	6.35	3.67	4.47	4.17	4.67
4	5.08	3.42	3.40	4.19	4.02
5	7.04	3.84	3.52	5.54	4.98
6	5.78	3.04	2.93	4.58	4.08
7	3.87	3.30	4.93	3.37	3.86
8	3.84	4.74	2.80	2.23	3.40
9	5.47	3.95	4.59	3.29	4.33
10	5.02	1.66	3.64	4.14	3.62
C-Mean	4.49	3.66	3.92	3.67	

Table 5.7: A table showing the highest in-sample annualised returns (%) for different period lengths and numbers of look-back periods. "R-Mean" calculates the mean of each row and "C-Mean" calculates the mean of each column. Currency pair: EUR/USD.

	120	240	360	480	R-Mean
1	1.13	0.54	1.15	3.41	1.56
2	-0.62	1.82	-0.12	0.89	0.49
3	-2.16	2.08	0.29	-0.83	-0.15
4	0.04	0.86	1.67	1.35	0.98
5	0.60	-0.44	2.84	-0.45	0.64
6	-0.03	2.02	2.62	0.18	1.20
7	2.31	1.83	-0.14	1.21	1.30
8	1.52	0.77	2.22	2.56	1.77
9	0.31	1.77	0.34	1.85	1.07
10	3.08	2.43	2.79	0.90	2.30
C-Mean	0.62	1.37	1.37	1.11	

Table 5.8: A table showing the highest test annualised returns (%) for different period lengths and numbers of look-back periods. "R-Mean" calculates the mean of each row and "C-Mean" calculates the mean of each column. Currency pair: EUR/USD.

	120	240	360	480
1	[42, 52]	[47, 53]	[45, 52]	[50, 50]
2	[42, 53]	[50, 55]	[50, 53]	[49, 52]
3	[50, 52]	[50, 54]	[49, 53]	[50, 51]
4	[30, 52]	[49, 54]	[49, 53]	[49, 52]
5	[50, 51]	[50, 52]	[49, 52]	[49, 52]
6	[48, 52]	[50, 53]	[49, 52]	[48, 51]
7	[50, 55]	[48, 54]	[49, 54]	[49, 52]
8	[50, 54]	[49, 52]	[49, 51]	[48, 52]
9	[30, 52]	[49, 52]	[50, 52]	[50, 52]
10	[46, 53]	[48, 53]	[50, 51]	[49, 52]
Mean-Diff	8.8	4.2	3.4	2.5

Table 5.9: A table showing the best (in terms of annualised return) in-sample trading rules with 10 iterative runs for different period lengths and numbers of look-back periods. Mean-Diff is calculated as the mean of the differences between the High RSI and the Low RSI of each column. Currency pair: USD/JPY.

	120	240	360	480	R-Mean
1	2.58	2.87	0.92	1.88	2.06
2	2.27	2.35	2.50	1.56	2.17
3	3.71	2.24	1.96	2.18	2.52
4	1.38	3.00	1.58	2.28	2.06
5	2.93	2.17	2.34	2.92	2.59
6	1.81	2.40	2.91	2.37	2.37
7	2.34	1.83	2.87	2.49	2.38
8	3.01	2.56	2.82	1.78	2.54
9	1.35	3.49	2.51	1.62	2.24
10	2.14	3.19	1.80	0.73	1.96
C-Mean	2.35	2.61	2.22	1.98	

Table 5.10: A table showing the highest in-sample annualised returns (%) for different period lengths and numbers of look-back periods. "R-Mean" calculates the mean of each row and "C-Mean" calculates the mean of each column. Currency pair: USD/JPY

	120	240	360	480	R-Mean
1	0.95	-0.84	2.66	5.24	2.00
2	-0.12	0.56	0.84	3.30	1.14
3	-1.37	2.61	1.30	2.09	1.16
4	0.64	0.99	-0.29	1.91	0.81
5	0.88	1.79	0.95	2.21	1.46
6	0.12	0.75	1.04	2.34	1.06
7	-0.76	1.56	1.54	3.15	1.37
8	-1.64	0.79	1.29	2.24	0.67
9	1.97	1.66	1.98	3.49	2.28
10	0.55	0.68	1.55	3.72	1.62
C-Mean	0.12	1.06	1.29	2.97	

Table 5.11: A table showing the highest out-of-sample annualised returns (%) for different period lengths and numbers of look-back periods. "R-Mean" calculates the mean of each row and "C-Mean" calculates the mean of each column. Currency pair: USD/JPY.

	120	240	360	480
1	[43, 64]	[43, 54]	[30, 58]	[50, 57]
2	[46, 61]	[44, 57]	[45, 54]	[50, 51]
3	[46, 58]	[47, 57]	[50, 54]	[50, 53]
4	[45, 59]	[47, 55]	[47, 53]	[50, 54]
5	[46, 59]	[50, 57]	[50, 53]	[50, 54]
6	[46, 59]	[48, 57]	[50, 54]	[50, 54]
7	[47, 58]	[46, 57]	[47, 55]	[50, 55]
8	[30, 59]	[49, 57]	[50, 55]	[49, 51]
9	[46, 60]	[47, 57]	[47, 54]	[50, 55]
10	[42, 60]	[47, 57]	[50, 53]	[50, 51]
Mean-Diff	16	9.7	7.7	3.6

Table 5.12: A table showing the best (in terms of annualised return) in-sample trading rules with 10 iterative runs for different period lengths and numbers of look-back periods. Mean-Diff is calculated as the mean of the differences between the High RSI and the Low RSI of each column. Currency pair: USD/CHF.

	120	240	360	480	R-Mean
1	1.61	0.97	0.00	3.04	1.41
2	1.92	0.39	0.69	6.04	2.26
3	1.49	1.28	4.16	4.23	2.79
4	0.05	0.30	0.88	4.07	1.32
5	0.96	3.24	3.10	3.14	2.61
6	0.56	0.66	3.42	4.56	2.30
7	1.78	0.13	1.81	3.35	1.77
8	0.00	2.21	3.01	3.84	2.27
9	0.64	1.50	1.13	3.38	1.66
10	0.19	0.73	4.51	4.53	2.49
C-Mean	0.92	1.14	2.27	4.02	

Table 5.13: A table showing the highest in-sample annualised returns (%) for different period lengths and numbers of look-back periods. "R-Mean" calculates the mean of each row and "C-Mean" calculates the mean of each column. Currency pair: USD/CHF

	120	240	360	480	R-Mean
1	-1.37	-0.73	0.00	-2.47	-1.14
2	-1.71	0.00	0.43	-4.86	-1.53
3	-0.73	-0.61	-4.26	-2.65	-2.06
4	0.07	0.10	-0.05	-2.54	-0.60
5	-0.53	-2.40	-2.56	-2.79	-2.07
6	-0.36	-0.59	-2.99	-3.10	-1.76
7	-2.00	0.07	-0.78	-3.37	-1.52
8	0.00	-1.89	-3.38	-4.53	-2.45
9	-0.58	-0.42	0.29	-1.95	-0.67
10	0.00	-0.19	-3.80	-5.21	-2.30
C-Mean	-0.72	-0.67	-1.71	-3.35	

Table 5.14: A table showing the highest out-of-sample annualised returns (%) for different period lengths and numbers of look-back periods. "R-Mean" calculates the mean of each row and "C-Mean" calculates the mean of each column. Currency pair: USD/CHF.

Tables 5.4, 5.7, 5.10 and 5.13 display the annualised returns (%) generated from the best RSI rules for the training set. Tables 5.5, 5.8, 5.11 and 5.14 show the annualised returns (%) generated from the best RSI strategies for the test set. The mean annualised return for each look-back size (mean of each row) and the mean annualised return for each sub-period length (mean of each column) are calculated. These means are only presented to serve as an indicator of how a given specification (a particular length or size) generally performs for a given currency pair. However, these mean values are not of great practical significance because in trading practice only the best model with the specification (different combinations of settings) and trading rule that generate highest return will be focused on.

The performances of both the training and test sets of each currency pair show in general the forecastability of the currency pair during the underlying period. Meanwhile, the difference in the training and test set performances indicates how much price movement patterns change from the training period to the test period.

From Tables 5.4, 5.7, 5.10 and 5.13, the USD/CHF and USD/JPY pairs generate lower annualised returns than the other two pairs (GBP/USD and EUR/USD), especially during the test period. The USD/CHF pair generates the lowest returns. Given the results of the gaps in values of the Low and High RSI of different pairs, the poor performance of the USD/CHF pair is associated with a large gap in its Low and High RSI values. For GBP/USD and EUR/USD which are more trend-following pairs (indicated by a small gap in values of the Low and High RSI) from the above results, the mean annualised return is higher overall across different lengths of forecasting periods. The above results show a negative relationship between the gap in values (of Low and High RSI) and profitability. According to Subsection 5.5.2, gaps in values (of Low and High RSI) are closely related to price patterns, i.e. large gaps in values (of Low and High RSI) usually accompany mean-reversion patterns and small gaps in values (of Low and High RSI)

usually accompany trend-following patterns. Therefore, this answers one of the research questions of how does market status being mean-reversion or trend-following affect market forecastability/profitability. Markets which exhibit more trend-following patterns have higher forecastability/profitability than markets which exhibit more mean-reversion patterns.

Therefore, it is essential for an investor to be able to forecast whether the next period of interest will be trend-following or not. This links with the contribution of the novel usage of RSI as a measure of patterns (the extent of the market status being trend-following or mean-reversion) and forecasting RSI with the help of a forecasting model, e.g. LSTM in this chapter.

5.5.5 A comparison with the widely adopted MA and RSI rules

The two widely used trading rules MA and RSI differ from the proposed LSTM-RSI in the sense that the LSTM-RSI algorithm is a forecasting-trading algorithm while the MA and RSI do not have a forecasting model associated with them.[121] In this section, comparisons will be made on these two types of trading rules to trade the four currency pairs.

One difference needs to be made clear between the term RSI as in LSTM-RSI and the RSI as in the RSI strategy. The LSTM-RSI uses LSTM to forecast RSI values as a measure of price patterns while the RSI strategy is a trading strategy based on over-bought and over-sold patterns identified by RSI values. So essentially the proposed LSTM-RSI differs from the traditional RSI strategy by the use of LSTM to forecast RSI values. Therefore, benchmark comparisons are made between LSTM-RSI and other strategies without LSTM, e.g. MA and RSI.

Tables 5.15 - 5.18 display the best performance from the MA and RSI rules, along with the best performance from the LSTM-RSI algorithm. For the MA rule, the best setting represents the number of data points used to calculate the MA that generates the highest annualised return for the training set. The RSI setting consists of an array of three numbers including the number of data points used to calculate RSI, the Lower and Higher RSI values as triggers of buy/sell order placement. The setting for the LSTM-RSI algorithm contains four numbers including the number of look-back periods, the length of each sub-period, the Lower and Higher RSI values as triggers of buy/sell order placement.

For the MA and RSI rules, the training set is used to configure the best setting and then the best setting is applied to the test set. For the LSTM model, the training set is used

[121] Refer to Subsection 5.1.2 for more details of the MA and RSI trading rules.

not only to find the best setting of the trading rule, but also to estimate parameters for the LSTM model.

This makes the performance of MA and RSI fixed results, i.e. once the setting is fixed, the trading performance of any given period is determined. Similarly, the trading component of the LSTM-RSI is also fixed once the trading rule is determined. However, because the LSTM-RSI algorithm has a forecasting model implemented, the trading performance is also highly dependent on the accuracy of the forecasting model. Therefore, there is always a potential to improve LSTM-RSI's performance, i.e. by adjusting the complexity of the LSTM or varying the parameter values.

The forecastability of the LSTM model is highly dependent upon the training process, whose efficiency is determined by the power of hardware equipment (computer CPU, GPU and memory size).[122] Therefore, readers should be aware that it is not the main focus of this chapter to obtain a model with extremely high forecastability and achieve significant returns. Instead, it is of interest in this chapter to introduce a new forecasting model - trading rule combination to explore the potential of such approaches in practice.

From Tables 5.15 - 5.18, it can be seen that, in general, the LSTM-RSI approach is able to give more stable performances than the MA and RSI trading rules. The phrase "more stable" is interpreted in two aspects. First, the LSTM-RSI suffers from less extreme losses than both MA and RSI, e.g. RSI suffers from an out-of-sample loss of 6.29% for EUR/USD and MA suffers from an out-of-sample loss of 12.26% for USD/CHF. Second, LSTM-RSI generally benefits from smaller annual transaction costs than MA and RSI. This is because the LSTM-RSI strategy controls (with a fixed forecasting horizon) the number of trades per year within a limit. However, with MA and RSI, there is no such limit, i.e. under the trading rule a trade is always placed whenever the criteria is met. In this way, a large number of trades which generate very small profits are frequently

[122] A supercomputer can train a model within seconds that a normal desktop would need months to train.

placed while after deducting transaction cost these trades will become profitless or even bring losses.

GBP/USD	MA	RSI	LSTM-RSI
Best setting	4320	[120, 45, 55]	[7, 240, 50, 50]
Training return (%)	2.10	3.34	7.93
Training return after transaction cost (%)	1.64	3.13	7.82
Test return (%)	2.88	7.43	0.94
Test return after transaction cost (%)	2.66	7.20	0.87

Table 5.15: A table showing the highest in-sample and out-of-sample annualised returns (%) for the MA and RSI trading rules compared with the LSTM-RSI trading algorithm for the GBP/USD pair. For the MA rule, the best setting represents the number of data points used to calculate the MA that generates the highest annualised return for the training set. For the RSI rule, the best setting consists of an array of three numbers including the number of data points used to calculate RSI, the Lower and Higher RSI values as triggers for buy/sell order placement. For the LSTM-RSI algorithm, the best setting contains an array of four numbers including the number of look-back periods, the length of each sub-period, the Lower and Higher RSI values as triggers for buy/sell order placement.

EUR/USD	MA	RSI	LSTM-RSI
Best setting	1080	[240, 50, 50]	[5, 120, 50, 50]
Training return (%)	5.57	5.90	7.04
Training return after transaction cost (%)	4.80	4.28	6.68
Test return (%)	-3.28	-4.40	0.60
Test return after transaction cost (%)	-4.46	-6.29	0.23

Table 5.16: A table showing the highest in-sample and out-of-sample annualised returns (%) for the MA and RSI trading rules compared with the LSTM-RSI trading algorithm for the EUR/USD pair. For the MA rule, the best setting represents the number of data points used to calculate the MA that generates the highest annualised return for the training set. For the RSI rule, the best setting consists of an array of three numbers including the number of data points used to calculate RSI, the Lower and Higher RSI values as triggers for buy/sell order placement. For the LSTM-RSI algorithm, the best setting contains an array of four numbers including the number of look-back periods, the length of each sub-period, the Lower and Higher RSI values as triggers for buy/sell order placement.

USD/JPY	MA	RSI	LSTM-RSI
Best setting	360	[120, 35, 55]	[3, 120, 50, 52]
Training return (%)	0.89	3.75	3.71
Training return after transaction cost (%)	-0.94	1.89	3.60
Test return (%)	0.17	-0.21	-1.37
Test return after transaction cost (%)	-1.63	-1.64	-1.52

Table 5.17: A table showing the highest in-sample and out-of-sample annualised returns (%) for the MA and RSI trading rules compared with the LSTM-RSI trading algorithm for the USD/JPY pair. For the MA rule, the best setting represents the number of data points used to calculate the MA that generates the highest annualised return for the training set. For the RSI rule, the best setting consists of an array of three numbers including the number of data points used to calculate RSI, the Lower and Higher RSI values as triggers for buy/sell order placement. For the LSTM-RSI algorithm, the best setting contains an array of four numbers including the number of look-back periods, the length of each sub-period, the Lower and Higher RSI values as triggers for buy/sell order placement.

USD/CHF	MA	RSI	LSTM-RSI
Best setting	840	[600, 45, 50]	[2, 480, 50, 51]
Training return (%)	3.01	5.77	6.04
Training return after transaction cost (%)	1.72	3.85	5.96
Test return (%)	-10.49	2.81	-4.86
Test return after transaction cost (%)	-12.26	-1.77	-4.93

Table 5.18: A table showing the highest in-sample and out-of-sample annualised returns (%) for the MA and RSI trading rules compared with the LSTM-RSI trading algorithm for the USD/CHF pair. For the MA rule, the best setting represents the number of data points used to calculate the MA that generates the highest annualised return for the training set. For the RSI rule, the best setting consists of an array of three numbers including the number of data points used to calculate RSI, the Lower and Higher RSI values as triggers for buy/sell order placement. For the LSTM-RSI algorithm, the best setting contains an array of four numbers including the number of look-back periods, the length of each sub-period, the Lower and Higher RSI values as triggers for buy/sell order placement.

5.5.6 Implications for EMH

In this chapter, the Jensen (1978) version of the EMH is tested,[123] i.e. utilising significant out-of-sample abnormal returns as a challenge to the EMH. In terms of the three forms of EMH as defined in Subsection 2.1.2, it is the weak form EMH that is tested. This is because information used for forecasting FX rates only includes historical FX rates and technical indicators which can be directly derived from historical FX rates.

Based on the results from Tables 5.15 - 5.18, it can be seen that although LSTM-RSI gives more stable performance than the benchmark MA and RSI. However, no currency pair generates significant out-of-sample profits after transaction cost is considered. Therefore, there is not enough evidence to challenge the weak form EMH for the four markets where the currencies are based.

A noteworthy fact is that across the four markets, the Swiss market once again (as in Chapter 3 and 4) shows most (weak form) efficiency because it is most difficult to generate abnormal returns (across all three trading strategies) in the Swiss FX market.

[123] For more details of the Jensen (1978) version of the EMH, refer to Subsection 2.1.2.

5.6 Conclusion

In this chapter, a price movement pattern-focused approach is proposed. Traditionally, the RSI indicator is used in the RSI trading rule in which the indicator value identifies overbought/oversold status of the current market. This chapter adopts the novel use of RSI as a trend indicator for past, present and future sub-periods. A forecasting-trading algorithm is built to forecast future price patterns and make trading decisions based on the forecast values.

With the fast development in multi-core GPU, there has been a greater tendency to apply ANN models to forecast financial asset prices, see Roondiwala et al. (2017), Baek and Young (2018), and Rundo (2019). However, in terms of technical trading, the number of research papers that apply ANN models for technical trading is limited.

This chapter utilises a LSTM model to forecast future price movement patterns (measured by the RSI indicator) and make trading decisions based on the forecast RSI values. A wide range of parameter specifications (number of look-back periods and length of sub-periods) are tested for four currency pairs on the LSTM model. The target is to explore the trend-following and mean-reversion patterns under different time horizons and for different FX pairs.

This methodology differs from the traditional RSI strategy mainly in the extra forecasting power provided by the LSTM model. More specifically, the traditional RSI strategy makes decisions on past over-bought or over-sold status suggested by RSI values. However, the LSTM-RSI model, firstly uses RSI for a different purpose (price pattern measuring) and secondly, has more forecasting power of future price patterns, as well as more flexibility (from a large number of adjustable parameters in the LSTM model), with the use of the LSTM. RSI and MA (as two of the most widely adopted technical trading strategies) without LSTM are used as benchmark strategies.

The proposed LSTM-RSI algorithm provides more stable trading performance than the benchmark MA and RSI rules (without LSTM) during the given period. It is observed that mean-reversion patterns happen more frequently in shorter time horizons (shorter than 2 weeks) and trend-following patterns are more prevalent in longer time horizons (2-4 weeks). Among the four currency pairs, USD/CHF is the most difficult to forecast. This is consistent with the conclusions from Chapters 3 and 4. In terms of forecasting horizon, GBP/USD and USD/CHF generate more profits with longer-horizon strategies and USD/JPY is more profitable with short-term strategies. The research results from Olson (2004) suggest that profits gained from trend-following strategy have diminished from 1971 to 2000. However, with empirical evidence in all of the four currency pairs from 1999 to 2019, this chapter shows that potential profit still exists in certain times and the profit level goes up as the extent of trend-following patterns increases. Therefore, trend-following strategies may continue to be profitable and worth further research.

In terms of the long-term overall performance of all strategies for the four pairs, none of the strategies generate significant profits (especially out-of-sample) for all currencies. This empirical evidence supports the weak-form EMH.[124]

The main research results of this chapter has potential practical usefulness to trading practitioners, government research departments, as well as academic researchers in understanding the significance of price movement patterns in forecasting/trading in the FX market and encouraging more efforts in developing other types of forecasting models to improve the technical trading process.

The main target of this chapter is not to develop a highly profitable trading algorithm but explore the not-so-widely adopted approach of forecasting-trading algorithms. However, this does not mean the proposed algorithm cannot be further improved to increase forecastability hence profitability. In fact, although there is some empirical evidence of

[124]For more details on the EMH refer back to Subsection 2.1.2.

performance convergence for the trained model, it is not possible to prove it is a global convergence rather than a local convergence. Therefore, given enough computation power (i.e. with high-end supercomputers), there is still a possibility that a significant improvement can be made on the LSTM model.

As Daniel and Moskowitz (2016) have suggested, despite persistent returns, momentum (trend-following) strategies suffer from significant losses in certain periods. This points to a further research direction of focusing on a shorter period of time in which one or more critical events happens, i.e. the 2008 global financial crisis, the Swiss National Bank's decision to drop the CHF/EUR limit of 1.2 in 2015, the UK EU-membership referendum in 2016. By researching in this direction, improved understanding will be formed on price movement patterns under extreme conditions so that not only will it help control risk but also suggest more significantly profitable trading opportunities.

6 Conclusions

6.1 Contributions to knowledge

Since the breakdown of the Bretton Woods system in 1971, the FX market has attracted increased attention from both financial practitioners and academic researchers, see Zhang and Hu (1998), Wang et al. (2001), Monfared and Enke (2014), Bai and Koong (2018), and Mollick and Sakaki (2019). The topics of volatility and return forecasting (including price movement pattern) remain central in terms of their significance from both speculation and hedging perspectives.

This thesis approaches the FX market from three directions: volatility forecasting, return forecasting and price movement pattern forecasting (as a special case of return forecasting). Several major (and minor) currency pairs are studied, at different frequencies - monthly, daily and hourly. Chapter 3, a methodology-focused chapter, develops a parameter estimation algorithm for a model that aims at forecasting FX volatility and some implications for the weak form EMH are drawn from the volatility forecasting results. In Chapter 4 and 5, both fundamental and technical analysis techniques are used, in order to explore profitability levels due to macro- and micro-structures of the FX market and test the semi-strong and weak forms of the EMH.

6.1.1 A review of Chapter 3

Donaldson and Kamstra (1997), Lu et al. (2016), and Kristjanpoller and Minutolo (2018) utilise GARCH-ANN models to forecast FX volatility. According to Hansen et al. (2010), the Genetic Algorithm (GA) gives improved performance over gradient-based algorithms (such as BFGS). Xu (2017) and Ding et al. (2019) focus on developing parameter estimation algorithms for gradient-based algorithms in the modelling and forecasting processes. Chapter 3 builds an algorithm that improves the performance of a

non-gradient-based algorithm (GA).

Chapter 3 aims at solving the local-optimum problem in the parameter estimation process for models with especially complex likelihood functions. The target model for estimation is GARCH-ANN, a model with linear and non-linear components which may increase the possibility of trapping into a local optimum in the estimation process.

The proposed Recursive Simulation Genetic Algorithm (RSGA) algorithm applies the concept of repetitive - recursive computation to a GA to estimate the GARCH-ANN models, which are implemented for forecasting daily volatility of GBP/USD, EUR/USD, USD/JPY, USD/CHF, USD/RUB and USD/ZAR. The six currency pairs selected include major FX pairs with relatively lower volatility such as USD/JPY and EUR/USD, medium level of volatility such as GBP/USD and USD/CHF and minor FX pairs with higher volatility such as USD/RUB and USD/ZAR.

The RSGA has made significant improvements in forecasting the volatility of all six currency pairs, in terms of MAE and RMSE. An average of over 70% improvement is made for the six pairs. Despite its medium-sized volatility level over the underlying period, the USD/CHF pair generates the highest overall errors (for both the benchmark algorithm and RSGA) among all pairs, making it the most difficult pair for volatility forecasting. Higher percentage improvement is achieved for higher-volatility pairs (USD/RUB and USD/ZAR). This empirical evidence shows that the proposed RSGA algorithm is able to significantly improve the parameter estimation results for modelling both lower-volatility and higher-volatility currency pairs.

The empirical results also provide some implications for the EMH. First, the emerging markets (Russia and South Africa) are less efficient than the developed markets (UK, Eurozone, Japan and Switzerland). This is consistent with the nature of these two types of markets. It also agrees with the empirical evidence from Hsu et al. (2016)

(see Subsection 4.1.2 and Section 4.6) where significantly less profits are obtained in developed markets than in emerging markets, suggesting the more efficient nature of developed markets. Second, based on the high extent of consistency of performance for in-sample and out-of-sample datasets and the high difficulty in forecasting the USD/CHF pair, the Swiss market demonstrates the most (weak form) efficiency out of the six markets in this chapter.

6.1.2 A review of Chapter 4

Huang et al. (2004), Erdogan and Goksu (2014), Bakhach et al. (2016), and Galeshchuk and Mukherjee (2017) apply ANN models with relatively small sizes (3-10 neurons per layer) in FX return forecasting. Chapter 4 extends the number of neurons per layer up to over 4000 to check whether the seemingly over-fitted models would improve forecasting performance. With the implementation of high-efficiency GPU, the computation time is controlled within an acceptable range.

Chapter 4 focuses on return forecasting on a monthly basis for four currency pairs (GBP/USD, EUR/USD, USD/JPY, USD/CHF) with a large ANN model. 42 input variables including historical FX rates and macroeconomic indicators are used. Due to limitations of computation power, previous researchers commonly use 5-20 neurons for the hidden layer. With the help of the fast developing GPU and highly-efficient training libraries (Keras), this chapter is able to implement a wide range (from 2 to 4096 increasing in power of 2) of the number of neurons in the hidden layer to build ANNs with various sizes.

By comparing performance of different ANN sizes, this chapter finds that as the number of neurons in the hidden layer increases up to 256 (a size that has been rarely worked with in previous similar research papers), both in-sample and out-of-sample profits start to increase. There is a sign that after the number of neurons reaches a certain amount

(dependent on the currency pair[125]), profits start to decline. However, due to the limitation in computation power (maximum number of neurons for the GPU in use is 4096), it is not clear whether this performance decline is permanent or temporary, i.e. if it is temporary, then after the number of neurons is increased beyond a certain level (somewhere over 4096), performance will start to rise again. Based on the fact that previous researchers with ANN models of much smaller sizes also obtained the "best" result for the number of neurons they finally decide to used after comparing with other numbers (within their computation limit), it is shown by empirical evidence in this chapter that their (e.g. results in previous research papers with small ANNs) result is unlikely to be the best. Likewise there is also no obvious reason to believe that the "best" result in this chapter is actually the best, i.e. the performance decline after the number of neurons reaches a certain amount is permanent.

Historically, it has been a consensus that the number of parameters should be limited as to reduce overfitting, i.e. too many parameters might over-interpret the data pattern and make people falsely believe their model is working well by generating nearly perfect in-sample results. When applied to an out-of-sample dataset, i.e. the model has never seen and learned in the training process, the overfitted model usually gives poor results, see Panchal et al. (2011), and Bilbao and Bilbao (2017).

In contrast to the previous literature on overfitting, the findings of this chapter point to a different direction of thinking about the overfitting problem. It is undeniably true that there are cases where the forecasting datasets have so simple patterns (i.e. linear, polynomial) that using too many parameters will increase the risk of overfitting. However, for almost all financial datasets (FX being one of the most complicated), the data patterns may never be deemed as simple as any hypothetical examples. Therefore, a further expansion of the number of parameters is suggested even if overfitting has already been

[125]For example, for GBPUSD, the optimum number of neurons is 1024.

observed. In Chapter 4, with the inclusion of drop-out layers, models with significantly larger neuron sizes do not suffer from the overfitting problem. This is verified by the good performance of the validation set (a small portion of the training set). A separate portion of the training set is used to avoid the use of the test set in the model selection process, which would cause the problem of "utilising the future data to forecast past data". Refer to Larsen et al. (1996), Bylander and Tate (2006), Guresen et al. (2011), Alvarez and Salzmann (2016), Lever et al. (2016), and Xu and Goodacre (2018) for justification of the use of a validation set for overfitting detection and model selection.

As a benchmark model, logistic regression is implemented with the same number of input variables as the large MLPNN. The large MLPNN outperforms logistic regression in terms of both accuracy rate and profit for all four currency pairs. This confirms that even with the same number of input variables the MLPNN model has more adaptive power to model with datasets of various levels of pattern complexity than fixed size models such as logistic regression.

From the finance perspective, although the four currency pairs all generate positive profits, USD/CHF has the lowest accuracy rate as well as lowest profit level. This is consistent with the conclusion of the volatility forecasting task in Chapter 3. The relative success of the USD/JPY pair is partially due to its higher profit achieved by adopting the buy-and-hold strategy. This is because the buy-and-hold strategy is more profitable when the FX rate is trend following and less profitable when the FX rate is mean reverting. The effect of price movement patterns indicated by being trend-following or mean-reversion on profitability is further discussed in Chapter 5.

The implications of the results on the EMH in this chapter are summarised below. First, there is evidence to challenge the semi-strong EMH for all four markets where the currencies are based. Second, overall the difference in the extent of inefficiency in the four markets is not significantly large. Third, the Swiss market exhibits relatively less

inefficiency than the other three markets.

6.1.3 A review of Chapter 5

Lui and Mole (1998), Oberlechner (2001), Coakley et al. (2016), and Hsu et al. (2016) discuss the application of technical analysis in the FX market. Yao and Tan (2000), Sang and Pierro (2018) utilise ANN models to improve technical trading. Gil-alana (2000), Serban (2010), and Lubnau and Todorova (2015) study the trend-following and mean-reversion patterns in trading practices. To the author's best knowledge, this thesis (Chapter 5) is the first to use an ANN model to forecast trend-following and mean-reversion patterns. The following paragraph explains the reason for studying these patterns.

The trend-following and mean-reversion patterns are valued highly by financial practitioners (e.g. FX traders) but have not been thoroughly studied by academic researchers. One difficulty is that the two states of patterns (trend-following and mean-reversion) are not categorical but numerical, i.e. for any period of interest, there is an extent of being trend-following or mean-reversion for FX rates, instead of just being in one state or the other. Therefore being able to measure the extent of trend-following or mean-reversion is essential in pattern forecasting.

This thesis (Chapter 5) utilises the Relative Strength Indicator (RSI) as a measure of the extent of being trend-following or mean-reversion. To the best knowledge of the author, this is the first attempt of the methodology in this area. Despite the fact that RSI has been used and researched extensively in both the financial and academic world, it has been mainly used as an indicator to suggest over-bought and over-sold status. By using RSI as an indicator to measure the extent of being trend-following or mean-reversion, this chapter then uses the Long Short Term Memory (LSTM) model to forecast future RSI values and make trading decisions based on the forecast RSI values.

Higher frequency hourly FX rates of four currency pairs (GBP/USD, EUR/USD, USD/JPY, USD/CHF) are used for forecasting. By comparing performance of two most widely-used technical trading rules Moving Average (MA) and RSI[126], the proposed LSTM-RSI strategy is able to give more stable trading performances than MA and RSI. While MA and RSI may generate satisfactory profits in certain times or for certain currency pairs, they give poor performance at other times or for other currency pairs. Overall, while LSTM-RSI increases performance stability, none of the three strategies is able to consistently generate profits across currency pairs throughout time.

One noteworthy fact is that the USD/CHF pair once again generates the lowest profits (or even losses) among the four currency pairs. Although the economic or structural reasons for the under-performance of the USD/CHF is beyond the researching scope of this thesis, these empirical findings suggest that even with relatively more sophisticated models and trading strategies, the performances of these models and strategies are still highly dependent upon the price pattern and nature of each specific currency pair. Therefore for financial practitioners such as FX traders, on top of a mature trading rule/strategy, choosing the most appropriate currency pair to trade is equivalently crucial towards successful trading performances.

As is discussed above, the out-of-sample profits for none of the four pairs are significant. This helps demonstrate that there is no evidence that any of the four markets is inefficient (in the sense of weak-form EMH). Moreover, the especially poor performance of the USD/CHF pair once again confirms the Swiss market as one of the most efficient markets in the world.[127]

---

[126]Here the term RSI refers to a trading rule based on the RSI indicator.

[127]Evidence in Chapters 3-5 supports that the Swiss market is the most (weak and semi-strong form) efficient among even the three highest developed markets. Therefore it would be fair to conclude that, after including less developed markets, the Swiss market is still one of the most efficient of all markets.

6.2 Limitations of the thesis

One limitation of the research methodologies of this thesis, which is also the limitation of most model-based methodologies, arises from the use of fixed forecasting horizons. For example, in the training stage, the model learns from a fixed number of input variables (lasting the same length of time) and forecasts a fixed length of time in the future. This problem is intrinsic with any model-based forecasting methodology because the model's structure needs to be pre-defined and fixed (although the parameter values may subject to changes in the training process). There is a problem of fixed-horizon forecasting because the success of fixed-horizon forecasting is highly dependent on assumption that the target dataset has a fixed length of periodical patterns, which is seldom the case for financial time series.

Another limitation arises from the low number of split training and test sets. In this thesis, due to the limit in the number of data points, the datasets are split into a training set and a test set. If the number of training and test sets increases, the number of data points in each set would be too small. The following example illustrates the problem as a result of using only one training and test set.

If a model needs to first determine the number of days GBP/USD typically fluctuates before it breaks out the range and make an upward or downward trend, and then make trading decisions based on whether the fluctuation period has ended. In the training process, the model learns from the whole period of the training set and determine the "most appropriate" number of fluctuation days based on a cost function. However, the problem is that for financial datasets like FX, the patterns are constantly changing and therefore taking a single value to represent the overall situation for the whole period may cause problems when patterns of the test set are changing and the overall pattern for the test set is significantly different from that of the training set.

6.3 Future research directions

One potential approach to partially solve the problem arising from adopting fixed-length horizons is to use varied-length models, see Ouyang and Ying (2018). With varied-length models, both the length of the data used to forecast the future and the length of the future data to be forecast can vary within a certain range of their nearest past forecasting settings. This will increase the flexibility of the model in terms of its capability of self-adapting changes in market conditions throughout time. However, the word "partially" is used in the first sentence because even with varied-length models, the varied lengths may only change in one (or more) pre-defined ways designed by the model builder. It is highly unlikely that the pre-defined variation specifications of varied-length models will cover even a small proportion of the total variation specifications of the market over a longer period.

To solve the problem caused by having only one training and one test set, it would be reasonable to split the datasets in to multiple pairs of training and test sets. Models are trained and updated on a rolling basis to promptly adapt to the changes in financial markets. This approach requires significantly large datasets for each training and test to have enough data points, i.e. very high frequency (5-min, 1-min, or even tick) FX data.

In this thesis, the information used to forecast future FX volatility, return or pattern comes from the financial (e.g. past FX rates and patterns) and economic (e.g. GDP, interest rates, inflation rates) aspects. However, because the trading of FX is essentially a game of humans, human behaviour is another crucial aspect not to be neglected. Therefore, another direction of future research is to include opinion-based information in the forecasting model, see Iwantoro and Koesrindartoto (2015), and Semiromi et al. (2020). In addition to the above factors, political factors (such as election and policy uncertainties) also play a crucial role in affecting market volatility, see Goodell et al. (2020).

The models implemented in this thesis are also related to the some of the further research directions. For example, LSTM (as in Chapter 5) can be used to read, summarise and learn from different opinions (i.e. whether a currency is going to appreciate or depreciate against another currency in the next few days) on social media following recent big news. Large ANN models (as in Chapter 4) can be used to forecast future FX movements based on the information learned from the LSTM, together with macro-economic indicators and past FX rates as in Chapter 4. The GARCH-ANN model (as in Chapter 3) can be used for identifying trading opportunities (periods with higher volatility), forecasting the effect of political uncertainties on market volatility, and managing risks.

In conclusion, this thesis focuses on improving forecasting performance for FX volatility, return and price patterns. The implemented algorithms and models may also facilitate further research directions as described above.

References

Abreu, G., Neves, R. & Horta, N. (2018), 'Currency exchange prediction using machine learning, genetic algorithms and technical analysis', *arXiv preprint arXiv* **1805**(11232).

Aka, K. (2020), 'The effect of selected macroeconomic indicators on foreign exchange rate: an application on Turkey's economy', *Journal of BRSA Banking and Financial Markets* **14**(1), 99–117.

Al-hnaity, B. & Abbod, M. (2016), 'Predicting financial time series data using hybrid model', *Intelligent Systems and Applications* **650**, 19–41.

Aladag, C. H., Egrioglu, E. & Kadilar, C. (2009), 'Forecasting nonlinear time series with a hybrid methodology', *Applied Mathematics Letters* **22**, 1467–1470.

Aliber, R. Z. . (1964), 'Speculation and price stability once again', *Journal of Political Economy* **72**(6), 607–609.

Alkhazaleh, M. M. H. (2018), 'Forecasting banking volatility in Amman stock exchange by using ARIMA model', *British Journal of Management* **29**(3), 1–9.

Alvarez-Diaz, M. & Alvarez, A. (2003), 'Forecasting exchange rates using genetic algorithms', *Applied Economics Letters* **10**(6), 319–322.

Alvarez, J. M. & Salzmann, M. (2016), 'Learning the number of neurons in deep networks', *Advances in Neural Information Processing Systems* **29**, 2270–2278.

Andreou, A. S., Georgopoulos, E. F. & Likothanassis, S. D. (2002), 'Exchange rates forecasting: a hybrid algorithm based on genetically optimized adaptive neural networks', *Computational Economics* **20**(3), 191–210.

Angelidis, T. & Degiannakis, S. (2008), 'Volatility forecasting : Intra-day versus inter-day models', *International Financial Markets, Institutions and Money* **18**, 449–465.

Babu, C. N. & Reddy, B. E. (2014), 'A moving-average filter based hybrid ARIMA – ANN model for forecasting time series data', *Applied Soft Computing* **23**, 27–38.

Baek, Y. & Young, H. (2018), 'ModAugNet: a new forecasting framework for stock market index value with an overfitting prevention LSTM module and a prediction LSTM module', *Expert Systems With Applications* **113**, 457–480.

Bai, S. & Koong, K. S. (2018), 'Oil prices , stock returns , and exchange rates: Empirical evidence from China and the United States', *North American Journal of Economics and Finance* **44**, 12–33.

Bakas, D. & Triantafyllou, A. (2019), 'Volatility forecasting in commodity markets using macro uncertainty', *Energy Economics* **81**, 79–94.

Bakhach, A., Tsang, E. P. K. & Jalalian, H. (2016), 'Forecasting directional changes in the FX markets', *IEEE Symposium Series on Computational Intelligence (SSCI)* pp. 1–8.

Bandi, F. M., Russell, J. R. & Yang, C. (2008), 'Realized volatility forecasting and option pricing', *Journal of Econometrics* **147**(1), 34–46.

Bank for International Settlements - Monetary and Economic Department (2016), 'Foreign exchange turnover in April 2016', *Triennial Central Bank Survey* .

Bank for International Settlements - Monetary and Economic Department (2019), 'Foreign exchange turnover in April 2019', *Triennial Central Bank Survey* .

Bashir, T., Khan, K. I. & Urooge, S. (2020), 'Assumptions of making a good deal with bad person: empirical evidence on strong form market efficiency', *Global Social Sciences Review* **5**(November), 154–162.

Baur, D. G. & McDermott, T. K. (2010), 'Is gold a safe haven? International evidence', *Journal of Banking and Finance* **34**(8), 1886–1898.

Bhargavi, R., Gumparthi, S. & R, A. (2017), 'Relative strength index for developing effective trading strategies in constructing optimal portfolio', *International Journal of Applied Engineering Research* **12**(19), 8926–8936.

Bilbao, I. & Bilbao, J. (2017), 'Overfitting problem and the over-training in the era of data: particularly for Artificial Neural Networks', *International Conference on Intelligent Computing and Information Systems (ICICIS)* pp. 173–177.

Bildirici, M. & Ersin, O. O. (2009), 'Improving forecasts of GARCH family models with the artificial neural networks: an application to the daily returns in Istanbul Stock Exchange', *Expert Systems with Applications* **36**(4), 7355–7362.

Bollerslev, T. (1986), 'Generalized autoregressive conditional heteroskedasticity', *Journal of Econometrics* **31**, 307–327.

Bollerslev, T., Hood, B., Huss, J. & Pedersen, L. H. (2018), 'Risk everywhere: modeling and managing volatility', *The Review of Financial Studies I* **31**(7), 2730–2773.

Bollerslev, T., Patton, A. J. & Quaedvlieg, R. (2016), 'Exploiting the errors: a simple approach for improved volatility forecasting', *Journal of Econometrics* **192**(1), 1–18.

Borges, M. R. (2010), 'Efficient market hypothesis in European stock markets', *The European Journal of Finance* **16**(7), 711–726.

Brooks, C. & Persand, G. (2003), 'Volatility forecasting for risk management', *Journal of Forecasting* **22**(1), 1–22.

Broyden, C. G. (1970), 'The convergence of a class of double-rank minimization algorithms', *IMA Journal of Applied Mathematics* **6**(3), 76–90.

Burt, J., Kaen, F. R. . & Booth, G. . G. (1977), 'Foreign exchange market efficiency under flexible exchange rates', *The Journal of Finance* **32**(4), 1325–1330.

Bylander, T. & Tate, L. (2006), Using validation sets to avoid overfitting in AdaBoost, *in* 'Flairs conference', pp. 544–549.

Byrd, R. H., Lu, P. & Nocedal, J. (1995), 'A limited-memory algorithm for bound-constrained optimization', *SIAM Journal on Scientific Computing* **16**(5), 1190–1208.

Cao, J., Li, Z. & Li, J. (2019), 'Financial time series forecasting model based on CEEMDAN and LSTM', *Physica A* **519**, 127–139.

Cao, S., Huang, H., Liu, R. & Macdonald, R. (2019), 'The term structure of exchange rate predictability : commonality , scapegoat , and disagreement', *Journal of International Money and Finance* **95**, 379–401.

Caporale, G. M. & Plastun, A. (2020), 'Daily abnormal price changes and trading strategies in the FOREX', *Journal of Economic Studies* **48**(1), 211–222.

Castelvecchi, D. (2016), 'Can we open the black box of AI?', *Nature* **538**(23), 1–4.

Ca'Zorrzi, M. & Rubaszek, M. (2020), 'Exchange rate forecasting on a napkin', *Journal of International Money and Finance* **104**(102168), 1–13.

Chaboud, A. P., Wright, J. H. & Chernenko, S. V. (2007), 'Trading activity and exchange rates in high-frequency EBS data', *International Finance Discussion Papers* (903), 1–28.

Chang, W. D. (2006), 'An improved real-coded genetic algorithm for parameters estimation of nonlinear systems', *Mechanical Systems and Signal Processing* **20**(1), 236–246.

Chang, Y.-h. (2019*a*), 'Cross-market information spillover and the performance of technical trading in the foreign exchange market', *Journal of Economics and Finance* **43**, 211–227.

Chang, Y.-h. (2019*b*), 'Cross-market information spillover and the performance of

technical trading in the foreign exchange market', *Journal of Economics and Finance* **43**(2), 211–227.

Chatziantoniou, I., Degiannakis, S. & Filis, G. (2019), 'Futures-based forecasts: how useful are they for oil price volatility forecasting?', *Munich Personal RePEc Archive* (96446).

Chaves, D. B. & Viswanathan, V. (2016), 'Momentum and mean-reversion in commodity spot and futures markets', *Journal of Commodity Markets* **3**(1), 39–53.

Chen, A.-s., Leung, M. T. & Daouk, H. (2003), 'Application of neural networks to an emerging financial market: forecasting and trading the Taiwan stock index', *Computers and Operations Research* **30**, 901–923.

Chen, H., Xiao, K., Sun, J. & Wu, S. (2017), 'A double-layer neural network framework for high-frequency forecasting', *ACM Transactions on Management Information Systems* **7**(4), 1–15.

Chen, J., Jiang, F., Liu, Y. & Tu, J. (2017), 'International volatility risk and Chinese stock return predictability', *Journal of International Money and Finance* **70**, 183–203.

Cheung, Y.-w., Chinn, M. D. & Pascual, A. G. (2005), 'Empirical exchange rate models of the nineties: are any fit to survive?', *Journal of International Money and Finance* **24**(7), 1150–1175.

Cheung, Y.-w., Chinn, M. D., Pascual, A. G. & Zhang, Y. (2019), 'Exchange rate prediction redux: new models, new data, new currencies', *Journal of International Money and Finance* **95**, 332–362.

Cheung, Y.-w., Fatum, R. & Yamamoto, Y. (2019), 'The exchange rate effects of macro news after the global financial crisis', *Journal of International Money and Finance* **95**, 424–443.

Cheung, Y.-w., McCauley, R. N. & Shu, C. (2019), 'Geographic spread of currency trading: the Renminbi and other emerging market currencies', *China and World Economy* **27**(5), 25–36.

Choi, K., Joo, D. & Kim, J. (2017), 'Kapre: On-GPU audio preprocessing layers for a quick implementation of deep neural network models with Keras', *arXiv preprint* .

Choudhry, T., McGroarty, F., Peng, K. & Wang, S. (2012), 'High-frequency exchange-rate prediction with an artificial neural network', *Intelligent Systems in Accounting, Finance and Management* **19**, 170–178.

Christoffersen, P. F. & Diebold, F. X. (2000), 'How relevant is volatility forecasting for financial risk management?', *Review of Economics and Statistics* **82**(1), 12–22.

Chung, H., Lee, S. J. & Park, J. G. (2016), 'Deep neural network using trainable activation functions', *International Joint Conference on Neural Networks* pp. 348–352.

CitiResearch (2010), 'CitiFX Pro forex traders survey'.

Coakley, J., Marzano, M. & Nankervis, J. (2016), 'How profitable are FX technical trading rules?', *International Review of Financial Analysis* **45**, 273–282.

Cocianu, C.-L. & Grigoryan, H. (2015), 'An artificial neural network for data forecasting purposes', *Information Economica* **19**(2), 34–45.

Coelho, I. M., Coelho, V. N., Eduardo, J., Luz, S., Ochi, L. S. & Guimarães, F. G. (2017), 'A GPU deep learning metaheuristic based model for time series forecasting', *Applied Energy* **201**, 412–418.

Cohen, G. (2020), 'Optimizing algorithmic strategies for trading Bitcoin', *Computational Economics* **Springer**, 1–16.

Cook, D., Ragsdale, C. & Major, R. (2000), 'Combining a neural network with a genetic

algorithm for process parameter optimization', *Engineering Applications of Artificial Intelligence* **13**(4), 391–396.

Corte, P. D., Ramadorai, T. & Sarno, L. (2016), 'Volatility risk premia and exchange rate predictability', *Journal of Financial Economics* **120**(1), 21–40.

Daniel, K. & Moskowitz, T. J. (2016), 'Momentum crashes', *Journal of Financial Economics* **122**, 221–247.

Dao, T. M., McGroarty, F. & Urquhart, A. (2019), 'The Brexit vote and currency markets', *Journal of International Financial Markets, Institutions and Money* **59**, 153–164.

Diebold, F. X., Hahn, J. & Tay, A. S. (1999), 'Multivariate density forecast evaluation and calibration in financial risk management : high-frequency returns on foreign exchange', *The Review of Economics and Statistics* **81**(4), 661–673.

Diebold, F. X., Schorfheide, F. & Shin, M. (2017), 'Real-time forecast evaluation of DSGE models with stochastic volatility', *Journal of Econometrics* **201**(2), 322–332.

Ding, F., Pan, J., Alsaedi, A. & Hayat, T. (2019), 'Gradient-based iterative parameter estimation algorithms for dynamical systems from observation data', *Mathematics* **7**(5), 428–442.

Donaldson, Glen, R. & Kamstra, M. (1997), 'An artificial neural network-GARCH model for international stock return volatility', *Journal of Empirical Finance* **4**, 17—-46.

Driskill, R. & McCafferty, S. (1980), 'Speculation, rational expectations, and stability of the foreign exchange market', *Journal of International Economics* **10**, 91–102.

Duda, R. O., Hart, P. E. & Stork, D. G. (2012), *Pattern classification*, John Wiley & Sons.

Dunis, C. L. & Miao, J. (2005), 'Optimal trading frequency for active asset management: evidence from technical trading rules', *Journal of Asset Management* **5**(5), 305–326.

Dunis, C. L. & Miao, J. (2007), 'Trading foreign exchange portfolios with volatility filters: the carry model revisited', *Applied Financial Economics* **17**(3), 249–255.

Dunis, C. L. & Williams, M. (2002), 'Modelling and trading the EUR / USD exchange rate: do neural network models perform better?', *Derivatives use, trading and regulation* **8**(3), 211–239.

Eberhart, R. & Kennedy, J. (1995), 'A new optimizer using particle swarm theory', *Proceedings of the Sixth International Symposium on Micro Machine and Human Science* **MHS'95**, 39–43.

Edwards, D. W. (2014), *Risk management in trading*, John Wiley & Sons.

Ellis, C. A. & Parbery, S. A. (2005), 'Is smarter better? A comparison of adaptive , and simple moving average trading strategies', *Research in International Business and Finance* **19**, 399–411.

Elsayed, S. M., Sarker, R. A. & Essam, D. L. (2014), 'Engineering applications of artificial intelligence: a new genetic algorithm for solving optimization problems', *Engineering Applications of Artificial Intelligence* **27**, 57–69.

Emerson, S., Kennedy, R., Shea, L. O. & Brien, J. O. (2019), 'Trends and applications of machine learning in quantitative finance', *8th International Conference on Economics and Finance Research* .

Eng, M. H., Li, Y., Wang, Q.-g. & Lee, T. H. (2008), 'Forecast forex with ANN using fundamental data', *International Conference on Information Management, Innovation Management and Industrial Engineering* (1), 279–282.

Engel, C., Lee, D., Liu, C., Liu, C., Pak, S. & Wu, Y. (2019), 'The uncovered interest parity puzzle , exchange rate forecasting , and Taylor rules', *Journal of International Money and Finance* **95**, 317–331.

Engle, R. F. (1982), 'Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation', *Econometrica* **50**(4), 987–1007.

Enke, D. & Amornwattana, S. (2008), 'A hybrid derivative trading system based on volatility and return forecasting', *The Engineering Economist* **53**(3), 259–292.

Erdogan, O. & Goksu, A. (2014), 'Forecasting Euro and Turkish Lira Exchange Rates with Artificial Neural Networks (ANN)', *International Journal of Academic Research in Accounting, Finance and Management Sciences* **4**(4), 307–316.

Fama, E. F. . (1970*a*), 'Efficient capital markets: a review of theory and empirical work', *The Journal of Finance* **25**(2), 383–417.

Fama, E. F. . (1970*b*), 'Efficient capital markets: a review of theory and empirical work', *The Journal of Finance* **25**(2), 383–417.

Fama, E. F. (1991), 'Efficient capital markets: II', *The Journal of Finance* **46**(5), 1575–1617.

Finnerty, J. E. (1976), 'Insiders and market efficiency', *The Journal of Finance* **31**(4), 1141–1148.

Fischetti, M. & Jo, J. (2018), 'Deep neural networks and mixed integer linear optimization', *Constraints* **23**, 296–309.

Fletcher, R. (1970), 'A new approach to variable metric algorithms', *The Computer Journal* **13**(3), 317–322.

Fletcher, R. & Reeves, C. M. (1964), 'Function minimization by conjugate gradients', *The Computer Journal* **7**(2), 149–154.

Fung, W. & Hsieh, D. A. (1997), 'Empirical characteristics of dynamic trading strategies: the case of hedge funds', *The Review of Financial Studies* **10**(2), 275–302.

Fyfe, C., Marney, J. P. & Tarbert, H. F. E. (1999), 'Technical analysis versus market efficiency - a genetic programming approach', *Applied Financial Economics* **9**, 183–191.

Galati, G. & Ho, C. (2003), 'Macroeconomic news and the Euro/Dollar exchange rate', *Economic Notes* **32**(3), 371–398.

Galeshchuk, S. & Mukherjee, S. (2017), 'Deep networks for predicting direction of change in foreign exchange rates', *Intelligent Systems in Accounting Finance & Management* **24**(4), 100–110.

Garikai, B. W. & Nyoni, T. (2019), 'Predicting net foreign direct investment in Nigeria using Box-Jenkins ARIMA approach', *Journal of Economics and Finance* **4**(2), 30–37.

Gerritsen, D. F., Bouri, E., Ramezanifar, E. & Roubaud, D. (2019), 'The profitability of technical trading rules in the Bitcoin market', *Finance Research Letters* (August), 1–10.

Gil-alana, L. A. (2000), 'Mean reversion in the real exchange rates', *Economics Letters* **69**, 285–288.

Givoly, D. & Palmon, D. (1985), 'Insider trading and the exploitation of inside information: some empirical evidence', *The Journal of Business* **58**(1), 69–87.

Glaser, M., Iliewa, Z. & Weber, M. (2019), 'Thinking about prices versus thinking about returns in financial markets', *The Journal of Finance* **74**(6), 2997–3039.

Goldfarb, D. (1970), 'A family of variable-metric methods derived by variational means', *Mathematics of Computation* **24**(109), 23–26.

Goodell, J. W., McGee, R. J. & McGroarty, F. (2020), 'Election uncertainty, economic policy uncertainty and financial market uncertainty: a prediction market analysis', *Journal of Banking & Finance* **110**, 105684.

Gradojevic, N. (2007), 'Non-linear, hybrid exchange rate modeling and trading profitability in the foreign exchange market', *Journal of Economic Dynamics and Control* **31**, 557–574.

Grefenstette, J. J. (1986), 'Optimization of control parameters for genetic algorithms', *IEEE Transactions on Systems, Man and Cybernetics* **16**(1), 122–128.

Grobys, K., Ahmed, S. & Sapkota, N. (2020), 'Technical trading rules in the cryptocurrency market', *Finance Research Letters* **32**(101396).

Guresen, E., Kayakutlu, G. & Daim, T. U. (2011), 'Using artificial neural network models in stock market index prediction', *Expert Systems with Applications* **38**(8), 10389–10397.

Hafner, C. (1998), 'Estimating high frequency foreign exchange rate volatility with nonparametric ARCH models', *Journal of Statistical Planning and Inference* **68**, 247–269.

Hajirahimi, Z. & Khashei, M. (2016), 'Improving the performance of financial forecasting using different combination architectures of ARIMA and ANN models', *Journal of Industrial Engineering and Management Studies* **3**(2), 17–32.

Hameed, A. & Rose, A. K. (2016), 'Exchange rate behavior with negative interest rates: some early negative observations', *Pacific Economic Review* **23**(1), 27–42.

Han, S., Liu, X., Mao, H., Pu, J., Pedram, A., Horowitz, M. A. & Dally, W. J. (2016), 'EIE: efficient inference engine on compressed deep neural network', *ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)* **44**(3), 243–254.

Hansen, N., Auger, A., Ros, R., Finck, S. & Posik, P. (2010), 'Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009', *Proceedings*

of the 12th Annual Conference Companion on Genetic and Evolutionary Computation pp. 1689–1696.

Hansen, P. R. & Lunde, A. (2005*a*), 'A forecast comparison of volatility models: does anything beat a GARCH (1,1)?', *Journal of Applied Econometrics* **20**(7), 873–889.

Hansen, P. R. & Lunde, A. (2005*b*), 'A forecast comparison of volatility models: Does anything beat a GARCH(1,1)?', *Journal of Applied Econometrics* **20**(7), 873–889.

He, X.-z. & Li, K. (2015), 'Profitability of time series momentum', *Journal of Banking and Finance* **53**, 140–157.

Heston, S. L. (1993), 'A closed-form solution for options with stochastic volatility with applications to bond and currency options', *The Review of Financial Studies* **6**(2), 327–343.

Hirabayashi, A., Aranha, C. & Iba, H. (2009), 'Optimization of the trading rule in foreign exchange using genetic algorithm', *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation* pp. 1529–1536.

Hochreiter, S. & Schmidhuber, J. (1997), 'Long short-term memory', *Neural Computation* **9**(8), 1735–1780.

Hoffmann, A. O. I. & Shefrin, H. (2014), 'Technical analysis and individual investors', *Journal of Economic Behavior and Organization* **107**, 487–511.

Holland, J. H. (1992), 'Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence', *MIT press* .

Hsu, P.-h., Taylor, M. P. & Wang, Z. (2016), 'Technical trading: is it still beating the foreign exchange market ?', *Journal of International Economics* **102**, 188–208.

Huang, W., Lai, K. K., Nakamori, Y. & Wang, S. (2004), 'Forecasting foreign exchange

rates with artificial neural networks: a review', *International Journal of Information Technology and Decision Making* **3**(1), 145–165.

Hudson, R. S. & Gregoriou, A. (2015), 'Calculating and comparing security returns is harder than you think: a comparison between logarithmic and simple returns', *International Review of Financial Analysis* **38**, 151–162.

Iwantoro, T. & Koesrindartoto, D. P. (2015), 'Exchange rate directional forecasting using sentiment analysis on social media in Indonesia 2015', *Australian Academy of Accounting and Finance Review* **1**(2), 132–147.

Jegadeesh, N. & Titman, S. (1993), 'Returns to buying winners and selling losers: implications for stock market efficiency', *Journal of Finance* **48**(1), 65–91.

Jensen, M. C. (1978), 'Some anomalous evidence regarding market efficiency', *Journal of Financial Economics* **6**(2/3), 95–101.

Jeon, B. N. & Lee, E. (2002), 'Foreign exchange market efficiency , cointegration, and policy coordination', *Applied Economics Letters* **9**(1), 61–68.

Jothimani, D. & Shankar, R. (2017), 'Ensemble of non-classical decomposition models and machine learning models for stock index prediction', *Proceedings of the Twelfth Midwest Association for Information Systems Conference* **6**.

Ju, C., Bibaut, A. & Laan, M. J. V. D. (2018), 'The relative performance of ensemble methods with deep Convolutional Neural Networks for image classification', *Journal of Applied Statistics* **45**(15), 2800–2818.

Junghans, L. & Darde, N. (2015), 'Hybrid single objective genetic algorithm coupled with the simulated annealing optimization method for building optimization', *Energy & Buildings* **86**, 651–662.

Kaastra, I. & Boyd, M. (1996), 'Designing a neural network for forecasting financial and economic time series', *Neurocomputing* **10**, 215–236.

Kamruzzaman, J. & Sarker, R. A. (2004), 'ANN-based forecasting of foreign currency exchange rates', *Neural Information Processing - Letters and Reviews* **3**(2), 49–58.

Karnaukh, N., Ranaldo, A. & Söderlind, P. (2015), 'Understanding FX liquidity', *Review of Financial Studies* **28**(11), 3073–3108.

Karuppiah, J. & Los, C. A. (2005), 'Wavelet multiresolution analysis of high-frequency Asian FX rates , Summer 1997', *International Review of Financial Analysis* **14**, 211–246.

Katusiime, L., Shamsuddin, A. & Agbola, F. W. (2015), 'Foreign exchange market efficiency and profi tability of trading rules: evidence from a developing country', *International Review of Economics and Finance* **35**, 315–332.

Kaytez, F., Taplamacioglu, M. C., Cam, E. & Hardalac, F. (2015), 'Forecasting electricity consumption: a comparison of regression analysis, neural networks and least squares support vector machines', *International Journal of Electrical Power and Energy Systems* **67**, 431–438.

King, M. R., Osler, C. & RIme, D. (2011), 'Foreign exchange market structure, players and evolution', *Norges Bank Economic Bulletin* **10**, 1–45.

Klein, T. & Walther, T. (2016), 'Oil price volatility forecast with mixture memory GARCH', *Energy Economics* **58**, 46–58.

Kongsilp, W. & Mateus, C. (2017), 'Volatility risk and stock return predictability on global financial crises', *China Financial Review International* **7**(1), 33–66.

Kristjanpoller, W. & Minutolo, M. C. (2015), 'Gold price volatility: a forecasting

approach using the Artificial Neural Network - GARCH model', *Expert Systems with Applications* **42**, 7245–7247.

Kristjanpoller, W. & Minutolo, M. C. (2018), 'A hybrid volatility forecasting framework integrating GARCH, artificial neural network, technical analysis and principal components analysis', *Expert Systems With Applications* **109**, 1–11.

Kum, S. & Nam, J. (2019), 'Joint detection and classification of singing voice melody using Convolutional Recurrent Neural Networks', *Applied Sciences* **9**(7), 1–17.

Lai, S., Xu, L., Liu, K. & Zhao, J. (2015), 'Recurrent Convolutional Neural Networks for text classification', *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence Recurrent* pp. 2267–2273.

Larsen, J., Hansen, L., Svarer, C. & Ohlsson, M. (1996), 'Design and regularization of neural networks: the optimal use of a validation set', *Neural Networks for Signal Processing* **6**, 62–71.

Le, H. T., Cerisara, C. & Denis, A. (2018), 'Do Convolutional Networks need to be deep for text classification ?', *The Workshops of the Thirty-Second AAAI Conference on Artificial Intelligence* pp. 29–36.

Lee, C. S. & Loh, K. Y. (2002), 'GP-based optimisation of the technical trading indicators and profitability in FX market', *Proceedings of the 9th International Conference on Neural Information Processing* **3**, 1159–1163.

Lever, J., Krzywinski, M. & Altman, N. (2016), 'Model selection and overfitting', *Nature Methods* **13**(9), 703–704.

Liu, C., Hou, W. & Liu, D. (2017), 'Foreign exchange rates forecasting with Convolutional Neural Network', *Neural Processing Letters* **46**(3), 1095–1119.

Liu, H. & Setiono, R. (1995), 'Chi2: feature selection and discretization of numeric

attributes', *IEEE 7th International Conference on Tools with Artifcial Intelligence* pp. 2–5.

Lu, X., Que, D. & Cao, G. (2016), 'Volatility forecast based on the hybrid artificial neural network and GARCH-type models', *Procedia Computer Science* **91**, 1044–1049.

Lubnau, T. & Todorova, N. (2015), 'Trading on mean-reversion in energy futures markets', *Energy Economics* **51**, 312–319.

Lui, Y.-h. & Mole, D. (1998), 'The use of fundamental and technical analyses by foreign exchange dealers: Hong Kong evidence', *Journal of International Money and Finance* **17**, 535–545.

Ma, F., Liao, Y., Zhang, Y. & Cao, Y. (2019), 'Harnessing jump component for crude oil volatility forecasting in the presence of extreme shocks', *Journal of Empirical Finance* **52**, 40–55.

Ma, Y. & Ji, Q. (2019), 'Oil financialization and volatility forecast: evidence from multidimensional predictors', *Journal of Forecasting* **38**, 564–581.

MacDonald, R. (1999), 'Exchange rate behaviour: are fundamentals important?', *The Economic Journal* **109**(459), 673–691.

Maggiori, E., Tarabalka, Y., Charpiat, G., Alliez, P., Neural, C., Maggiori, E., Tarabalka, Y., Charpiat, G. & Alliez, P. (2017), 'Convolutional Neural Networks for large-scale remote sensing image classification', *IEEE Transactions on Geoscience and Remote Sensing* .

Makovsky, P. (2014), 'Modern approaches to efficient market hypothesis of FOREX – the Central European case', *Procedia Economics and Finance* **14**, 397–406.

Malkiel, B. G. (2005), 'Reflections on the Efficient Market Hypothesis: 30 years later', *The Financial Review* **40**, 1–9.

Mariano, C. N. Q., Sablan, V. F., Sardon, J. R. C. & Paguta, R. B. (2016), 'Investigation of the factors affecting real exchange rate in the Philippines', *Review of Integrative Business and Economics Research* **5**(4), 171–202.

McCulloch, W. S. & Pitts, W. (1943), 'A logical calculus of the ideas immanent in nervous activity', *Bulletin of Mathematical Biophysics* **5**, 115–133.

McGroarty, F., ap Gwilym, O. & Thomas, S. (2007), 'The components of electronic inter-dealer spot FX bid-ask spreads', *Journal of Business Finance and Accounting* **34**(9-10), 1635–1650.

McMillan, D. G. & Speight, A. E. H. (2004), 'Daily volatility forecasts: reassessing the performance of GARCH models', *Journal of Forecasting* **23**, 449–460.

McMillan, D. G. & Speight, A. E. H. (2012), 'Daily FX volatility forecasts: can the GARCH (1,1) model be beaten using high-frequency data ?', *Journal of Forecasting* **31**, 330–343.

McNally, S., Roche, J. & Caton, S. (2018), 'Predicting the price of bitcoin using machine learning', *26th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing* .

Meese, R. & Rogoff, K. (1981), 'Empirical exchange rate models of the seventies: Do they fit out of sample ?', *Journal of International Economics* **14**(1983), 3–24.

Menkhoff, L., Sarno, L., Schmeling, M. & Schrimpf, A. (2012), 'Currency momentum strategies', *Journal of Financial Economics* **106**(3), 660–684.

Miah, M. & Rahman, A. (2016), 'Modelling volatility of daily stock returns: is GARCH (1,1) enough?', *American Scientific Research Journal for Engineering, Technologt, and Sciences* **18**(9), 29–39.

Mo, H. & Wang, J. (2018), 'Return scaling cross-correlation forecasting by stochas-

tic time strength neural network in financial market dynamics', *Soft Computing* **22**(9), 3097–3109.

Mollick, A. V. & Sakaki, H. (2019), 'Exchange rates, oil prices and world stock returns', *Resources Policy* **61**, 585–602.

Monfared, S. A. & Enke, D. (2014), 'Volatility forecasting using a hybrid GJR-GARCH neural network model', *Procedia Computer Science* **36**, 246–253.

Moskowitz, T. J., Ooi, Y. H. & Pedersen, L. H. (2012), 'Time series momentum', *Journal of Financial Economics* **104**(2), 228–250.

Mukherji, S. (2011), 'Are stock returns still mean-reverting?', *Review of Financial Economics* **20**(1), 22–27.

Nag, A. K. & Mitra, A. (2002), 'Forecasting daily foreign exchange rates using genetically optimized neural networks', *Journal of Forecasting* **21**(7), 501–511.

Namin, S. S. & Namin, A. S. (2018), 'Forecasting economic and financial time series: ARIMA vs. LSTM', *arXiv preprint arXiv:1803.06386* pp. 1–19.

Narayan, P. K., Liu, R. & Westerlund, J. (2016), 'A GARCH model for testing market efficiency', *Journal of International Financial Markets, Institutions and Money* **41**, 121–138.

Nelder, J. A. & Mead, R. (1965), 'A simplex method for function minimization', *The Computer Journal* **7**(4), 308–313.

Oberlechner, T. (2001), 'Importance of technical and fundamental analysis in the European foreign exchange market', *International Journal of Finance and Economics* **6**, 81–93.

Olson, D. (2004), 'Have trading rule profits in the currency markets declined over time?', *Journal of banking & Finance* **28**(1), 85–105.

Ono, I., Kita, H. & Kobayashi, S. (2003), 'A real-coded genetic algorithm using the unimodal normal distribution crossover', *Advances in Evolutionary Computing* pp. 203–237.

Oshodi, O. S., Ejohwomu, O. A., Famakin, I. O. & Cortez, P. (2017), 'Comparing univariate techniques for tender price index forecasting: Box-Jenkins and neural network model', *Construction Economics and Building* **17**(3), 109–123.

Ouyang, Y. & Yin, H. (2018), 'Multi-step time series forecasting with an ensemble of varied length mixture models', *International Journal of Neural Systems* **28**(4), 1750053.

Panchal, G., Ganatra, A., Shah, P. & Panchal, D. (2011), 'Determination of over-learning and over-fitting problem in back propagation neural network', *International Journal on Soft Computing* **2**(2), 40–51.

Panda, C. & Narasimhan, V. (2007), 'Forecasting exchange rate better with artificial neural network', *Journal of Policy Modeling* **29**, 227–236.

Pathberiya, H. A., Tilakaratne, C. D. & Hansen, L. L. (2017), 'An intelligent system for forex trading: hybrid ANN with GARCH and intrinsic mode functions', *Intelligent Systems Conference* **7-8**(September), 436–445.

Pedirappagari, V. R. & Babu, C. H. (2019), 'Validating relative strength index for developing productive trading strategies in Indian stock market', *International Journal of Applied Engineering Research* **14**(3), 717–731.

Pedregosa, F., Weiss, R. & Brucher, M. (2011), 'Scikit-learn: machine learning in Python', *Journal of Machine Learning Research* **12**, 2825–2830.

Piccotti, L. R. (2018), 'Jumps, cojumps, and efficiency in the spot foreign exchange market', *Journal of Banking and Finance* **87**, 49–67.

Piczak, K. J. (2017), 'Environmental sound classification with Convolutional Neural

Networks', *IEEE International Workshop on Machine Learning for Signal Processing* pp. 17–20.

Popovic, S. & Durovic, A. (2014), 'Intraweek and intraday trade anomalies: evidence from FOREX market', *Applied Economics* **46**(32), 3968–3979.

Psaradellis, I., Laws, J. & Pantelous, A. A. (2019), 'Performance of technical trading rules: evidence from the crude oil market', *The European Journal of Finance* **25**(17), 1793–1815.

Qi, M. & Wu, Y. (2006), 'Technical trading-rule profitability, data snooping, and reality check: evidence from the foreign exchange market', *Journal of Money, Credit and Banking* **38**(8), 2135–2158.

Qiu, M., Pinfold, J. F. & Rose, L. C. (2011), 'Predicting foreign exchange movements using historic deviations from PPP', *International Review of Economics and Finance* **20**(4), 485–497.

Raza, A., Marshall, B. R. & Visaltanachoti, N. (2014), 'Is there momentum or reversal in weekly currency returns?', *Journal of International Money and Finance* **45**, 38–60.

Rehman, M., Khan, G. M. & Mahmud, S. A. (2014), 'Foreign currency exchange rates prediction using CGP and Recurrent Neural Network', *International Conference on Future Inform mation Engineering* **10**, 239–244.

Roondiwala, M., Patel, H. & Varma, S. (2017), 'Predicting stock prices using LSTM', *International Journal of Science and Research* **6**(4), 1754–1756.

Rothig, A., Semmler, W. & Flaschel, P. (2007), 'Hedging, speculation, and investment in balance-sheet triggered currency crises', *Australian Economic Papers* **46**(3), 224–233.

Roubaud, D. & Arouri, M. (2018), 'Oil prices, exchange rates and stock markets under uncertainty and regime-switching', *Finance Research Letters* **27**, 28–33.

Rozeff, M. S. & Zaman, M. A. (1988), 'Market efficiency and insider trading: new evidence', *The Journal of Business* **61**(1), 25–44.

Rundo, F. (2019), 'Deep LSTM with reinforcement learning layer for financial trend prediction in FX high frequency trading systems', *Applied Sciences* **9**(4460), 1–18.

Rundo, F., Trenta, F., Luigi, A. & Battiato, S. (2019), 'Grid trading system robot (GTSbot): a novel mathematical algorithm for trading FX Market', *Applied Sciences* **9**(1796), 1–15.

Salamon, J. & Bello, J. P. (2017), 'Deep Convolutional Neural Networks and data augmentation for environmental sound classification', *IEEE Signal Processing Letters* **24**(3), 279–283.

Sang, C. & Pierro, M. D. (2019), 'Improving trading technical analysis with TensorFlow Long Short-Term Memory (LSTM) Neural Network', *The Journal of Finance and Data Science* **5**(1), 1–11.

Semiromi, H. N., Lessmann, S. & Peters, W. (2020), 'News will tell: forecasting foreign exchange rates based on news story events in the economy calendar', *North American Journal of Economics and Finance* **52**, 101181.

Serban, A. F. (2010), 'Combining mean reversion and momentum trading strategies in foreign exchange markets', *Journal of Banking and Finance* **34**(11), 2720–2727.

Sermpinis, G., Dunis, C., Laws, J. & Stasinakis, C. (2012), 'Forecasting and trading the EUR/USD exchange rate with stochastic neural network combination and time-varying leverage', *Decision Support Systems* **54**(1), 316–329.

Sermpinis, G., Theofilatos, K., Karathanasopoulos, A., Georgopoulos, E. F. & Dunis, C. (2013), 'Forecasting foreign exchange rates with adaptive neural networks using radial-

basis functions and Particle Swarm Optimization', *European Journal of Operational Research* **225**(3), 528–540.

Sezer, O. B. & Ozbayoglu, M. (2018), 'Algorithmic financial trading with deep Convolutional Neural Networks: time series to image conversion approach', *Applied Soft Computing* **70**, 525–538.

Shanno, D. F. (1970), 'Conditioning of quasi-Newton methods for function minimization', *Mathematics of Computation* **24**(111), 647–656.

Si, X., Lu, J., Zhang, X. & Zhang, J. (2017), 'An improved RNA genetic algorithm for the parameter estimation multiple solutions of ordinary differential equations', *Procedia Engineering* **174**, 477–481.

Singh, A. & Mishra, G. C. (2015), 'Application of Box-Jenkins method and Artificial Neural Network procedure for time series forecasting of prices', *Statistics in Transition* **16**(1), 83–96.

Sjaastad, L. A. & Scacciavillani, F. (1996), 'The price of gold and the exchange rate', *Journal of International Money and Finance* **15**(6), 879–897.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014), 'Dropout: A simple way to prevent Neural Networks from overfitting', *Journal of Machine Learning Research* **15**, 1929–1958.

Stathakis, D. (2009), 'How many hidden layers and nodes?', *International Journal of Remote Sensing* **30**(8), 2133–2147.

Sweeney, R. J. . (1986), 'Beating the foreign exchange market', *The Journal of Finance* **41**(1), 163–182.

Syarif, I., Prugel-Bennett, A. & Wills, G. (2016), 'SVM parameter optimization using

grid search and genetic algorithm to improve classification performance', *Telkomnika* **14**(4), 1502–1509.

Szakmary, A. C., Shen, Q. & Sharma, S. C. (2010), 'Trend-following trading strategies in commodity futures: a re-examination', *Journal of Banking and Finance* **34**(2), 409–426.

Tang, Y., Ji, J., Gao, S., Dai, H., Yu, Y. & Todo, Y. (2018), 'A pruning Neural Network Model in credit classification analysis', *Computational Intelligence and Neuroscience* **2018**, 1–22.

Taylor, M. P., Peel, D. A. & Sarno, L. (2001), 'Nonlinear mean-reversion in real exchange rates: Toward a solution to the purchasing power parity puzzles', *International Economic Review* **42**(4), 1015–1042.

Taylor, N. (2014), 'The rise and fall of technical trading rule success', *Journal of Banking and Finance* **40**, 286–302.

Tenti, P. (1996), 'Forecasting foreign exchange rates using recurrent neural networks', *Applied Artificial Intelligence ISSN:* **10**(6), 567–582.

Thakur, M., Meghwani, S. S. & Jalota, H. (2014), 'A modified real coded genetic algorithm for constrained optimization', *Applied Mathematics and Computation* **235**, 292–317.

Timmermann, A. & Granger, C. W. J. (2004), 'Efficient market hypothesis and forecasting', *International Journal of Forecasting* **20**, 15–27.

Tran, D. T., Iosifidis, A., Kanniainen, J. & Gabbouj, M. (2018), 'Temporal attention augmented bilinear network for financial time-series data analysis', *IEEE transactions on neural networks and learning systems* **30**(5), 1407–1418.

Tripathy, N. (2017), 'Forecasting gold price with Auto Regressive Integrated Moving

Average model', *International Journal of Economics and Financial Issues* **7**(4), 324–329.

Urquhart, A., Gebka, B. & Hudson, R. (2015), 'How exactly do markets adapt? Evidence from the moving average rule in three developed markets', *Journal of International Financial Markets, Institutions and Money* **38**, 127–147.

Villanueva, O. M. (2007), 'Forecasting currency excess returns: can the forward bias be exploited?', *The Journal of Financial and Quantitative Analysis* **42**(4), 963–990.

Wang, H. (2019), 'VIX and volatility forecasting: A new insight', *Physica A* **533**(121951).

Wang, J. & Wang, J. (2015), 'Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks', *Neurocomputing* **156**, 68–78.

Wang, J. & Wang, J. (2016), 'Forecasting energy market indices with recurrent neural networks: case study of crude oil price fluctuations', *Energy* **102**, 365–374.

Wang, J., Wang, J., Fang, W. & Niu, H. (2016), 'Financial time series prediction using Elman Recurrent Random Neural Networks', *Computational Intelligence and Neuroscience* **2016**, 1–15.

Wang, K. L., Fawson, C., Barrett, C. B. & McDonald, J. B. (2001), 'A flexible parametric GARCH model with an application to exchange rates', *Journal of Applied Econometrics* **16**(4), 521–536.

Weithers (2006), *Foreign exchange markets: a practical guide to the FX markets*, Wiley Finance.

Willmott, C. J. (1982), 'Some comments on the evaluation of model performance', *Bulletin of the American Meteorological Society* **63**(11), 1309–1313.

Worthington, A. C. & Higgs, H. (2004), 'Random walks and market efficiency in European equity markets', *Global Journal of Finance and Economics* **1**(1), 59–78.

Wu, X. & Wang, X. (2019), 'Forecasting volatility using realized stochastic volatility model with time-varying leverage effect', *Finance Research Letters* **34**, 1–7.

Wu, Y. (2011), 'Momentum trading, mean reversal and overreaction in Chinese stock market', *Review of Quantitative Finance and Accounting* **37**(3), 301–323.

Xu, L. (2017), 'The parameter estimation algorithms based on the dynamical response measurement data', *Advances in Mechanical Engineering* **9**(11), 1–12.

Xu, Y. & Goodacre, R. (2018), 'On splitting training and validation set: a comparative study of cross - validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning', *Journal of Analysis and Testing* **2**(3), 249–262.

Yalcinkaya, A., Senoglu, B. & Yolcu, U. (2018), 'Maximum likelihood estimation for the parameters of skew normal distribution using genetic algorithm', *Swarm and Evolutionary Computation* **38**, 127–138.

Yang, H., Kutan, A. M. & Ryu, D. (2019), 'Volatility information trading in the index options market: an intraday analysis', *International Review of Economics and Finance* **64**, 412–426.

Yang, L., Cai, J. J. & Hamori, S. (2018), 'What determines the long-term correlation between oil prices and exchange rates?', *North American Journal of Economics and Finance* **44**, 140–152.

Yao, J., Poh, H.-l. & Jasic, T. (1996), 'Foreign exchange rates forecasting with neural networks', *International Conference on Neural Information Processing* pp. 754–759.

Yao, J. & Tan, C. L. (2000), 'A case study on using neural networks to perform technical forecasting of forex', *Neurocomputing* **34**, 79–98.

Yildirim, D. C., Toroslu, I. H. & Fiore, U. (2021), 'Forecasting directional movement of Forex data using LSTM with technical and macroeconomic indicators', *Financial Innovation* **7**(1), 1–36.

Zarrabi, N., Snaith, S. & Coakley, J. (2017), 'FX technical trading rules can be profitable sometimes', *International Review of Financial Analysis* **49**, 113–127.

Zeng, D., Wang, S., Shen, Y. & Shi, C. (2017), 'A GA-based feature selection and parameter optimization for selection and parameter optimization for support tucker machine', *Procedia Computer Science* **111**, 17–23.

Zhang, G. & Hu, M. Y. (1998), 'Neural network forecasting of the British Pound/US Dollar exchange rate', *International Journal of Management Science* **26**(4), 495–506.

Zhang, G. P. (2003), 'Time series forecasting using a hybrid ARIMA and neural network model', *Neurocomputing* **50**, 159–175.

Zhang, J., Zheng, C.-h., Xia, Y., Wang, B. & Chen, P. (2017), 'Optimization enhanced genetic algorithm-support vector regression for the prediction of compound retention indices in gas chromatography', *Neurocomputing* **240**, 183–190.

Zhang, L., Aggarwal, C. & Qi, G.-j. (2017), 'Stock price prediction via discovering multi-frequency trading patterns', *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining* **August**, 2141–2149.

Zhang, L. M. (2015), 'Genetic deep neural networks using different activation functions for financial data mining', *IEEE International Conference on Big Data* **IEEE**, 2849–2851.

7 Appendix

Chapter 3

	GBP	EUR	JPY
a_0	-2.18e-02 (4.42e-03)	-6.28e-04 (4.87e-03)	-1.63e-03 (9.91e-04)
a_1	-2.28e-02 (3.08e-03)	-2.37e-01 (2.55e-01)	-1.67e-01 (-1.39e-01)
α	2.36e-03 (8.49e-04)	2.58e-03 (7.79e-04)	2.19e-03 (1.65e-03)
β	-2.04e-01 (-3.40e-01)	-2.29e-01 (-3.65e-01)	-9.97e-02 (-1.62e-01)
γ	4.49e-02 (3.05e-01)	3.46e-02 (2.34e-01)	-3.20e-02 (2.82e-01)
ξ	-1.83e-03 (-5.51e-04)	-7.75e-04 (-6.33e-04)	-1.87e-03 (-1.54e-03)

Table 7.1: Appendix: Chapter 3 - GARCH-ANN estimated coefficients with RSGA (in brackets) and one randomly selected GA out of 100 rounds of GA for log-returns of GBP, EUR and JPY. 1 Jan 2008 to 31 Dec 2017 (daily). All currencies are against USD.

	CHF	RUB	ZAR
a_0	9.07e-03 (-3.87e-03)	-1.45e-02 (-4.05e-04)	1.71e-02 (-2.85e-03)
a_1	6.16e-02 (-2.05e-01)	-7.68e-02 (1.51e-01)	4.34e-02 (-3.02e-01)
α	6.40e-03 (2.07e-03)	3.49e-03 (2.62e-04)	3.26e-03 (2.73e-04)
β	-2.18e-01 (-4.26e-01)	-1.31e-01 (-2.35e-01)	-2.05e-01 (2.51e-01)
γ	9.06e-02 (1.36e-01)	2.15e-02 (5.51e-02)	1.78e-01 (3.09e-01)
ξ	-5.52e-03 (-1.64e-03)	-3.54e-03 (-1.29e-04)	-3.13e-03 (-3.06e-04)

Table 7.2: Appendix: Chapter 3 - GARCH-ANN estimated coefficients with RSGA (in brackets) and one randomly selected GA out of 100 rounds of GA for log-returns of CHF, RUB and ZAR. 1 Jan 2008 to 31 Dec 2017 (daily). All currencies are against USD.

Chapter 4

A list of the Datastream ID's and their brief description for the variables selected:

USDOLLR - GBPUSD close prices

ECURRS$ - EURUSD close prices

JAPAYE$ - USDJPY close prices

SWISSF$ - USDCHF close prices

DJINDUS - Dow Jones Industrials

NASCOMP - NASDAQ Composite

S&PCOMP - S&P 500 Composite

FTSE100 - FTSE 100 Composite

JPIBO3M - Japan Interbank 3 months offshore

SWIBK3M- Swiss Interbank 3 months offshore

BBGBP3M - IBA GBP Interbank 3 months

BBUSD3M - IBA USD Interbank 3 months

BBJPY3M - IBA JPY Interbank 3 months

BBEUR3M - IBA EUR Interbank 3 months

BBCHF3M - IBA CHF Interbank 3 months

SP5LVIN - S&P 500 Low Volatility

CBOEVIX - CBOE SPX Volatility VIX

MSWDMVL - MSCI World Minimum Volatility

MSURMVE - MSCI Europe Minimum Volatility

FTSEGL$ - FTSE Global 100

CRUDOIL - Crude Oil Prices

GOLDBLN - Gold Bullion LBM

GSCITOT - S&P Commodities (GSCI)

DJUBSTR - Bloomberg Commodity Index

UKPRATE. - Bank of England Base Rate

USPRATE. - Federal Funds Target Rate

EMIBOR3. - Euro Interbank offered Rate 3 months

UKGBOND. - UK Gross Redemption Yield on 20-Year Gilts

USGBOND. - US Treasury Yield Adjusted to Constant Maturity 20-Year

USGBILL3 - US Treasury Bill Rate 3 months

EMGBOND. - Euro Government Bond Yield 10-Year

JPGBOND. - Japan Interest-Bearing Government Bonds 10-Year

SWGBOND. - Swiss Confederation Bond Yield 10-Year

USCONPRCE - US CPI

UKCONPRCF - UK CPI

EMCPHARMF - Eurozone HICP

JPCONPRCF - Japan CPI

SWCONPRCF - Swiss CPI

UKEXPGDSA - UK Exports

UKIMPGDSA - UK Imports

USEXPGDSB - US Exports

USIMPGDSB - US Imports

JPEXPGDSA - Japan Exports

JPIMPGDSA - Japan Imports

SWEXPGDSA - Swiss Exports

SWIMPGDSA - Swiss Imports

Chapter 4: Type of data of the explanatory variables

(1) Index: Dow Jones, Nasdaq, S&P 500, FTSE 100, S&P 500 Low Volatility, World Minimum Volatility, Europe Minimum Volatility, FTSE Global 100, S&P Commodity Total Return, Bloomberg Commodity Total Return, US CPI, UK CPI, Euro HICP, Japan CPI, Swiss CPI.

(2) Percentage: Japan Interbank Offshore, Swiss Interbank, GBP Interbank, USD Interbank, JPY Interbank, EUR Interbank, CHF Interbank, CBOE Volatility, Bank of England Base Rate, US Federal Target Rate, Euro Interbank Offered Rate, UK Gross Redemption Yield, US Treasury Yield, US Treasury Bill Rate, Euro Government Bond Yield, Japan Government Bond Yield, Swiss Confederation Bond Yield.

(3) Price (in USD): Crude Oil, Gold Bullion.

(4) Others: all Exports and Imports are values in local currency. (Japan - Billions JPY, UK - Millions GBP, US - Millions USD, Swiss - Millions CHF)

	Max	Min	Mean	Median	SD
Dow Jones	26651.21	6763.29	13222.80	11350.40	4395.22
Nasdaq	8109.54	1213.72	3201.29	2575.11	1601.29
S&P 500	2924.59	700.82	1494.43	1331.33	498.91
FTSE 100	7701.77	3625.83	5841.62	5957.82	970.23
Japan Interbank offshore	0.88	0.05	0.27	0.18	0.23
Swiss Interbank	3.51	-1.53	0.64	0.23	1.25
GBP Interbank	6.74	0.28	2.88	3.12	2.28
USD Interbank	6.87	0.22	2.22	1.34	2.06
JPY Interbank	1.02	-0.12	0.22	0.13	0.26
EUR Interbank	5.29	-0.39	1.84	2.06	1.74
CHF Interbank	3.57	-0.86	0.74	0.28	1.25
S&P 500 Low Volatility	8126.19	1949.57	4164.22	3632.20	1607.60
CBOE Volatility	68.51	9.45	19.93	17.93	8.23
World Minimum Volatility	2295.25	819.03	1346.09	1220.74	381.07
Europe Minimum Volatility	1820.05	755.26	1273.49	1187.15	296.70
FTSE Global 100	1816.62	586.57	1067.51	1030.78	267.87
Crude Oil	141.06	12.14	60.07	58.41	28.05
Gold Bullion	1826.35	254.00	876.90	914.98	473.45
S&P Commodity Total Return	10636.48	1888.23	4351.78	4250.95	1631.55
Bloomberg Commodity TR	468.63	108.59	236.66	247.78	70.65
Bank of England Base Rate	6.00	0.25	2.64	1.75	2.23
US Federal Target Rate	6.50	0.25	1.99	1.25	1.99
Euro Interbank Offered Rate	5.11	-0.33	1.85	2.04	1.72
UK Gross Redemption Yield	5.18	1.27	3.80	4.32	1.09

Table 7.3: Appendix: Chapter 4 - Summary statistics of the explanatory variables.

	Max	Min	Mean	Median	SD
US Treasury Yield	6.86	1.82	4.17	4.39	1.27
US Treasury Bill Rate	6.37	-0.01	1.78	1.05	1.92
Euro Government Bond Yield	5.70	0.61	3.47	3.91	1.39
Japan Government Bond Yield	2.02	-0.23	1.05	1.25	0.59
Swiss Confederation Bond Yield	4.19	-0.54	1.83	2.15	1.32
US CPI	252.79	164.70	210.90	214.76	25.82
UK CPI	107.14	71.37	87.17	85.80	11.39
Euro HICP	104.68	73.76	90.33	91.70	9.21
Japan CPI	102.00	95.70	98.12	97.50	1.57
Swiss CPI	103.56	91.92	99.27	100.43	3.12
UK Exports	36739.93	11766.75	21474.85	20759.19	5425.55
UK Imports	46498.74	14919.16	29241.58	29780.69	8156.43
US Exports	143464.00	55522.00	98042.14	98905.50	29362.09
US Imports	217875.00	77781.00	153178.89	161428.00	39948.69
Japan Exports	7681.69	3450.56	5504.20	5521.85	1039.76
Japan Imports	8047.03	2662.08	5214.83	5441.69	1356.82
Swiss Exports	35715.38	8392.20	17867.78	16724.21	5866.06
Swiss Imports	30665.08	7894.72	16482.65	15245.39	4883.06

Table 7.4: Appendix: Chapter 4 - Continued - Summary statistics of the explanatory variables.

Run number	Accuracy (%)	Annualised return (%)	Time (seconds)
1	57.50	2.75	9.53
2	57.50	2.75	16.58
3	60.00	4.43	23.75
4	57.50	2.75	31.36
5	60.00	4.43	39.20
6	58.75	4.02	46.75
7	60.00	4.41	55.06
8	60.00	4.62	62.89
9	62.50	5.27	70.80
10	61.25	5.70	79.03
11	61.25	4.91	87.55
12	57.50	2.75	95.83
13	57.50	2.75	104.22
14	57.50	2.75	112.97
15	46.25	-0.30	121.73
16	58.75	4.62	130.55
17	60.00	5.59	139.78
18	58.75	4.02	149.02
19	56.25	3.30	158.23
20	57.50	2.75	167.73

Table 7.5: Appendix: Chapter 4 - Table of the GBP/USD test set prediction performances from 100 runs of a 1024-neuron MLPNN.

Run number	Accuracy (%)	Annualised return (%)	Time (seconds)
21	57.50	2.75	177.39
22	58.75	4.02	186.95
23	55.00	5.41	196.61
24	57.50	2.75	206.39
25	56.25	4.56	217.16
26	43.75	-2.34	227.64
27	57.50	4.53	237.88
28	57.50	2.75	248.48
29	61.25	5.68	259.13
30	42.50	-3.38	269.69
31	57.50	3.28	280.41
32	57.50	2.60	292.34
33	57.50	3.98	303.41
34	57.50	2.75	314.66
35	57.50	4.18	326.03
36	58.75	6.01	337.38
37	57.50	4.31	348.84
38	43.75	-1.63	361.02
39	57.50	2.75	372.92
40	60.00	4.77	384.78

Table 7.6: Appendix: Chapter 4 - Continued - Table of the GBP/USD test set prediction performances from 100 runs of a 1024-neuron MLPNN.

Run number	Accuracy (%)	Annualised return (%)	Time (seconds)
41	56.25	3.75	396.72
42	57.50	2.75	408.84
43	57.50	2.75	421.41
44	61.25	6.49	434.31
45	56.25	4.56	446.91
46	58.75	4.02	459.80
47	60.00	4.43	472.66
48	57.50	2.75	485.61
49	57.50	2.75	498.67
50	60.00	5.02	512.14
51	53.75	3.55	525.61
52	58.75	4.34	539.41
53	42.50	-3.38	552.98
54	57.50	3.98	566.91
55	56.25	4.22	581.13
56	58.75	4.29	595.25
57	57.50	2.75	609.86
58	57.50	2.75	624.06
59	60.00	4.55	638.47
60	57.50	4.53	653.69

Table 7.7: Appendix: Chapter 4 - Continued - Table of the GBP/USD test set prediction performances from 100 runs of a 1024-neuron MLPNN.

Run number	Accuracy (%)	Annualised return (%)	Time (seconds)
61	55.00	5.70	668.91
62	57.50	2.66	683.66
63	57.50	2.75	698.72
64	58.75	4.12	714.06
65	61.25	6.02	729.27
66	57.50	2.75	744.58
67	57.50	2.75	760.50
68	57.50	2.98	776.72
69	62.50	5.09	793.11
70	57.50	2.75	809.42
71	60.00	4.32	826.72
72	57.50	2.75	842.95
73	57.50	2.75	859.42
74	57.50	3.61	876.20
75	53.75	4.38	893.14
76	60.00	5.46	910.39
77	57.50	2.75	928.11
78	57.50	2.75	945.23
79	58.75	4.62	962.78
80	58.75	4.02	980.34

Table 7.8: Appendix: Chapter 4 - Continued - Table of the GBP/USD test set prediction performances from 100 runs of a 1024-neuron MLPNN.

Run number	Accuracy (%)	Annualised return (%)	Time (seconds)
81	55.00	3.93	998.13
82	62.50	6.15	1016.00
83	58.75	5.08	1033.89
84	57.50	2.75	1051.88
85	57.50	4.91	1069.97
86	58.75	4.74	1088.89
87	58.75	4.02	1107.52
88	57.50	2.75	1126.31
89	57.50	4.13	1145.66
90	58.75	4.30	1164.58
91	57.50	3.89	1184.73
92	60.00	4.43	1204.08
93	57.50	4.53	1223.66
94	60.00	4.10	1243.27
95	45.00	-1.85	1263.03
96	62.50	5.73	1283.16
97	61.25	6.67	1303.52
98	58.75	4.68	1324.20
99	57.50	2.75	1344.92
100	58.75	5.31	1365.83

Table 7.9: Appendix: Chapter 4 - Continued - Table of the GBP/USD test set prediction performances from 100 runs of a 1024-neuron MLPNN.

Run number	CPU time (s)	GPU time (s)	GPU speed-up (%)
1	10.81	9.53	13.43
2	20.98	16.58	26.54
3	30.89	23.75	30.06
4	41.15	31.36	31.22
5	51.87	39.20	32.32
6	62.12	46.75	32.88
7	72.57	55.06	31.80
8	83.25	62.89	32.37
9	94.07	70.80	32.87
10	104.78	79.03	32.58
11	115.56	87.55	31.99
12	126.92	95.83	32.44
13	137.93	104.22	32.35
14	149.56	112.97	32.39
15	160.84	121.73	32.13
16	172.45	130.55	32.09
17	183.97	139.78	31.61
18	195.61	149.02	31.26
19	207.73	158.23	31.28
20	219.95	167.73	31.13

Table 7.10: Appendix: Chapter 4 - A table showing the CPU computation time against the GPU computation time and the GPU speed-up for 100 runs. Currency pair is GBP/USD.

Run number	CPU time (s)	GPU time (s)	GPU speed-up (%)
21	232.01	177.39	30.79
22	244.53	186.95	30.80
23	256.84	196.61	30.63
24	269.72	206.39	30.68
25	282.28	217.16	29.99
26	295.11	227.64	29.64
27	307.98	237.88	29.47
28	321.36	248.48	29.33
29	335.06	259.13	29.30
30	348.29	269.69	29.14
31	361.65	280.41	28.97
32	375.36	292.34	28.40
33	389.40	303.41	28.34
34	403.25	314.66	28.15
35	417.07	326.03	27.92
36	431.17	337.38	27.80
37	447.51	348.84	28.29
38	462.06	361.02	27.99
39	476.81	372.92	27.86
40	491.40	384.78	27.71

Table 7.11: Appendix: Chapter 4 - Continued - A table showing the CPU computation time against the GPU computation time and the GPU speed-up for 100 runs. Currency pair is GBP/USD.

Run number	CPU time (s)	GPU time (s)	GPU speed-up (%)
41	506.51	396.72	27.67
42	522.17	408.84	27.72
43	537.14	421.41	27.46
44	552.34	434.31	27.18
45	567.53	446.91	26.99
46	582.86	459.80	26.76
47	598.26	472.66	26.57
48	614.06	485.61	26.45
49	630.04	498.67	26.34
50	646.04	512.14	26.15
51	662.15	525.61	25.98
52	678.47	539.41	25.78
53	694.61	552.98	25.61
54	711.07	566.91	25.43
55	728.00	581.13	25.27
56	744.76	595.25	25.12
57	761.53	609.86	24.87
58	779.04	624.06	24.83
59	796.14	638.47	24.69
60	813.51	653.69	24.45

Table 7.12: Appendix: Chapter 4 - Continued - A table showing the CPU computation time against the GPU computation time and the GPU speed-up for 100 runs. Currency pair is GBP/USD.

Run number	CPU time (s)	GPU time (s)	GPU speed-up (%)
61	830.93	668.91	24.22
62	848.50	683.66	24.11
63	867.50	698.72	24.16
64	886.43	714.06	24.14
65	904.43	729.27	24.02
66	922.40	744.58	23.88
67	940.78	760.50	23.71
68	959.76	776.72	23.57
69	978.37	793.11	23.36
70	997.43	809.42	23.23
71	1016.36	826.72	22.94
72	1035.93	842.95	22.89
73	1054.93	859.42	22.75
74	1074.79	876.20	22.66
75	1094.06	893.14	22.50
76	1113.62	910.39	22.32
77	1133.43	928.11	22.12
78	1153.45	945.23	22.03
79	1173.50	962.78	21.89
80	1193.84	980.34	21.78

Table 7.13: Appendix: Chapter 4 - Continued - A table showing the CPU computation time against the GPU computation time and the GPU speed-up for 100 runs. Currency pair is GBP/USD.

Run number	CPU time (s)	GPU time (s)	GPU speed-up (%)
81	1215.07	998.13	21.73
82	1236.25	1016.00	21.68
83	1257.01	1033.89	21.58
84	1278.28	1051.88	21.52
85	1299.42	1069.97	21.44
86	1320.86	1088.89	21.30
87	1342.39	1107.52	21.21
88	1364.15	1126.31	21.12
89	1386.54	1145.66	21.03
90	1408.68	1164.58	20.96
91	1430.86	1184.73	20.78
92	1453.20	1204.08	20.69
93	1475.67	1223.66	20.59
94	1498.36	1243.27	20.52
95	1521.15	1263.03	20.44
96	1544.53	1283.16	20.37
97	1567.79	1303.52	20.27
98	1591.64	1324.20	20.20
99	1615.67	1344.92	20.13
100	1639.73	1365.83	20.05
Mean	711.46	576.24	23.47

Table 7.14: Appendix: Chapter 4 - Continued - A table showing the CPU computation time against the GPU computation time and the GPU speed-up for 100 runs. Currency pair is GBP/USD.

N	A (%)	SD_A	P (%)	SD_P	T_{10} (s)	R_A	R_P
2	49.25	0.0778	-0.0100	0.0014	8.59	6.3302	-0.0726
4	50.38	0.0596	-0.2200	0.0347	10.85	8.4534	-0.0634
8	50.62	0.0470	1.1700	0.1047	10.98	10.7683	0.1118
16	50.25	0.0935	2.6200	0.1934	11.78	5.3767	0.1355
32	49.75	0.0485	2.4300	0.1900	13.62	10.2648	0.1279
64	51.88	0.0391	1.9900	0.1455	15.52	13.2789	0.1368
128	48.88	0.0386	3.0200	0.2089	15.92	12.6572	0.1446
256	55.75	0.0184	3.5700	0.2215	17.43	30.3618	0.1612
512	52.45	0.0237	3.1400	0.2059	19.75	22.1670	0.1525
1024	53.15	0.0301	2.7900	0.2085	21.55	17.6521	0.1338
2048	51.02	0.0361	2.8200	0.2090	23.35	14.1307	0.1349
4096	52.33	0.0342	2.7000	0.2116	26.62	15.2804	0.1276

Table 7.15: Appendix: Chapter 4 - A table displaying performance evaluation metrics on the test set for different number of neurons with 10 runs, where N is the number of neurons, A represents the mean accuracy rate from the 10 runs for a given number of neurons, SD_A denotes the standard deviation of accuracy, P denotes Annualised return, SD_P denotes the standard deviation of annualised return, T_{10} denotes the average computation time in seconds per run out of the 10 runs, R_A denotes the ratio of accuracy to its standard deviation and R_P denotes the ratio of annualised return to its standard deviation. The underlying currency pair is EUR/USD.

N	A (%)	SD_A	P (%)	SD_P	T_{10} (s)	R_A	R_P
2	50.88	0.0675	0.3400	0.0937	9.62	7.5413	0.0363
4	51.00	0.0802	-0.4100	0.0326	10.79	6.3627	-0.1259
8	53.75	0.0402	-1.9300	0.1961	12.01	13.3656	-0.0984
16	53.25	0.0822	2.8800	0.1535	12.72	6.4757	0.1876
32	46.12	0.0370	3.5200	0.2423	13.21	12.4508	0.1453
64	50.12	0.0323	2.2000	0.1354	13.93	15.5240	0.1625
128	52.12	0.0340	2.5300	0.1626	15.21	15.3429	0.1556
256	55.75	0.0294	2.2100	0.1143	16.57	18.9834	0.1933
512	58.55	0.0187	4.3200	0.2163	18.18	31.3284	0.1997
1024	56.12	0.0341	3.2700	0.2012	20.22	16.4387	0.1625
2048	53.87	0.0290	3.5900	0.2276	22.88	18.5823	0.1577
4096	51.25	0.0231	3.4800	0.2269	26.37	22.2198	0.1534

Table 7.16: Appendix: Chapter 4 - A table displaying performance evaluation metrics on the test set for different number of neurons with 10 runs, where N is the number of neurons, A represents the mean accuracy rate from the 10 runs for a given number of neurons, SD_A denotes the standard deviation of accuracy, P denotes Annualised return, SD_P denotes the standard deviation of annualised return, T_{10} denotes the average computation time in seconds per run out of the 10 runs, R_A denotes the ratio of accuracy to its standard deviation and R_P denotes the ratio of annualised return to its standard deviation. The underlying currency pair is USD/JPY.

N	A (%)	SD_A	P (%)	SD_P	T_{10} (s)	R_A	R_P
2	49.88	0.0777	-0.0042	0.0820	8.60	6.4197	-0.0512
4	50.75	0.0812	-0.0031	0.0681	10.84	6.2467	-0.0455
8	50.25	0.0533	-0.0015	0.0210	10.98	9.4214	-0.0713
16	51.25	0.0617	0.0076	0.2420	11.78	8.3120	0.0314
32	50.75	0.0503	0.0129	0.1491	13.62	10.0812	0.0865
64	51.25	0.0385	0.0216	0.2111	15.50	13.3109	0.1023
28	52.75	0.0266	0.0299	0.2038	15.93	19.8271	0.1467
256	50.12	0.0313	0.0271	0.1804	17.44	16.0289	0.1502
512	51.25	0.0363	0.0226	0.1768	19.76	14.1292	0.1278
1024	50.50	0.0355	0.0189	0.1921	21.57	14.2318	0.0984
2048	51.75	0.0420	0.0163	0.1641	23.37	12.3127	0.0993
4096	51.88	0.0341	0.0141	0.1452	26.62	15.2103	0.0971

Table 7.17: Appendix: Chapter 4 - A table displaying performance evaluation metrics on the test set for different number of neurons with 10 runs, where N is the number of neurons, A represents the mean accuracy rate from the 10 runs for a given number of neurons, SD_A denotes the standard deviation of accuracy, P denotes Annualised return, SD_P denotes the standard deviation of annualised return, T_{10} denotes the average computation time in seconds per run out of the 10 runs, R_A denotes the ratio of accuracy to its standard deviation and R_P denotes the ratio of annualised return to its standard deviation. The underlying currency pair is USD/CHF.

	Estimate	SE	z value	p value
Intercept	-0.3601	0.2520	-1.43	0.1531
GBP	-1.4011	1.9471	-0.72	0.4718
EUR	4.5003	3.0942	1.45	0.1458
JPY	-0.4963	1.1447	-0.43	0.6646
CHF	4.8494	3.2986	1.47	0.1415
DowJones	-0.3070	2.3336	-0.13	0.8953
Nasdaq	-1.2761	1.6305	-0.78	0.4338
SP500	2.1673	5.0181	0.43	0.6658
FTSE100	1.9967	2.9216	0.68	0.4943
JapanInterbankoffshore	-5.2053	3.2579	-1.60	0.1101
SwissInterbank	8.0408	5.2630	1.53	0.1266
GBPInterbank	0.0413	4.5069	0.01	0.9927
USDInterbank	-4.0739	4.8058	-0.85	0.3966
JPYInterbank	5.8031	3.7018	1.57	0.1170
EURInterbank	4.9349	5.4199	0.91	0.3625
CHFInterbank	-8.8017	5.8826	-1.50	0.1346
SP500LowVolatility	-3.0824	3.7101	-0.83	0.4061
CBOEVolatility	-1.6148	1.0336	-1.56	0.1182
WorldMinimumVolatility	-6.1272	4.6750	-1.31	0.1900
EuropeMinimumVolatility	5.9057	4.7488	1.24	0.2136

Table 7.18: Appendix: Chapter 4 - Estimated coefficients of the logistic regression model. Underlying currency pair is GBP/USD.

	Estimate	SE	z value	p value
FTSEGlobal100	-3.5889	3.9231	-0.91	0.3603
CrudeOil	-4.6242	2.9510	-1.57	0.1171
GoldBullion	-2.0886	3.8842	-0.54	0.5908
SPCommodityTotalReturn	2.3434	3.4852	0.67	0.5013
BloombergCommodityTR	-1.6597	3.3458	-0.50	0.6199
BankofEnglandBaseRate	3.2327	4.7912	0.67	0.4999
USFederalTargetRate	1.2073	5.1403	0.23	0.8143
EuroInterbankOfferedRate	-7.1028	6.0639	-1.17	0.2415
UKGrossRedemptionYield	-0.2483	0.8887	-0.28	0.7799
USTreasuryYield	1.5292	2.0176	0.76	0.4485
USTreasuryBillRate	1.8509	3.8626	0.48	0.6318
EuroGovernmentBondYield	0.2912	1.1144	0.26	0.7939
JapanGovernmentBondYield	-1.1082	0.6726	-1.65	0.0994
SwissConfederationBondYield	2.3095	1.7522	1.32	0.1875
USCPI	4.9593	8.0109	0.62	0.5359
UKCPI	7.5490	9.1025	0.83	0.4069
EuroHICP	3.0397	9.7089	0.31	0.7542
JapanCPI	0.8760	1.0938	0.80	0.4232
SwissCPI	-6.5274	3.4281	-1.90	0.0569

Table 7.19: Appendix: Chapter 4 - Continued - Estimated coefficients of the logistic regression model. Underlying currency pair is GBP/USD.

	Estimate	SE	z value	p value
UKExports	-0.9345	1.0623	-0.88	0.3790
UKImports	0.7229	1.9251	0.38	0.7073
USExports	2.1992	4.3632	0.50	0.6142
USImports	4.3059	3.8523	1.12	0.2637
JapanExports	1.0556	0.8850	1.19	0.2330
JapanImports	-2.4425	1.5242	-1.60	0.1090
SwissExports	2.4917	1.3741	1.81	0.0698
SwissImports	-1.4962	1.0944	-1.37	0.1716

Table 7.20: Appendix: Chapter 4 - Continued - Estimated coefficients of the logistic regression model. Underlying currency pair is GBP/USD.

	Estimate	SE	z value	p value
Intercept	0.1422	0.2074	0.69	0.4927
GBP	-1.3459	1.8399	-0.73	0.4645
EUR	0.1462	2.7721	0.05	0.9579
JPY	-1.2278	1.1271	-1.09	0.2760
CHF	3.0638	2.9116	1.05	0.2927
DowJones	0.2129	2.2972	0.09	0.9261
Nasdaq	-4.0299	1.7548	-2.30	0.0216
SP500	5.8985	5.2102	1.13	0.2576
FTSE100	-2.3930	2.6834	-0.89	0.3725
JapanInterbankoffshore	2.0799	2.6595	0.78	0.4342
SwissInterbank	1.5806	3.7515	0.42	0.6735
GBPInterbank	-1.0719	3.7557	-0.29	0.7753
USDInterbank	0.9468	3.1726	0.30	0.7654
JPYInterbank	-1.3919	3.1162	-0.45	0.6551
EURInterbank	5.3716	4.6215	1.16	0.2451
CHFInterbank	-4.3502	4.3422	-1.00	0.3164
SP500LowVolatility	-7.6346	3.6727	-2.08	0.0376
CBOEVolatility	-1.0610	0.8598	-1.23	0.2172
WorldMinimumVolatility	-1.9383	4.5856	-0.42	0.6725
EuropeMinimumVolatility	6.8630	4.4194	1.55	0.1204

Table 7.21: Appendix: Chapter 4 - Estimated coefficients of the logistic regression model. Underlying currency pair is EUR/USD.

	Estimate	SE	z value	p value
FTSEGlobal100	-2.2303	3.9553	-0.56	0.5728
CrudeOil	0.5414	2.6853	0.20	0.8402
GoldBullion	-0.7370	3.2281	-0.23	0.8194
SPCommodityTotalReturn	-4.0918	3.1887	-1.28	0.1994
BloombergCommodityTR	3.2827	3.2220	1.02	0.3083
BankofEnglandBaseRate	7.3591	4.0438	1.82	0.0688
USFederalTargetRate	-0.7483	4.1235	-0.18	0.8560
EuroInterbankOfferedRate	-8.1471	5.1560	-1.58	0.1141
UKGrossRedemptionYield	-0.6692	0.9130	-0.73	0.4636
USTreasuryYield	-0.1934	1.8452	-0.10	0.9165
USTreasuryBillRate	-1.8993	3.1550	-0.60	0.5472
EuroGovernmentBondYield	1.3791	1.0878	1.27	0.2049
JapanGovernmentBondYield	-1.0532	0.6348	-1.66	0.0971
SwissConfederationBondYield	1.8672	1.6353	1.14	0.2535
USCPI	5.1334	7.1829	0.71	0.4748
UKCPI	3.9129	9.4859	0.41	0.6800
EuroHICP	-0.7237	9.4157	-0.08	0.9387
JapanCPI	1.8198	1.0709	1.70	0.0893
SwissCPI	-1.4875	3.1050	-0.48	0.6319

Table 7.22: Appendix: Chapter 4 - Continued - Estimated coefficients of the logistic regression model. Underlying currency pair is EUR/USD.

	Estimate	SE	z value	p value
UKExports	0.2281	1.0573	0.22	0.8292
UKImports	0.7798	1.8573	0.42	0.6746
USExports	-3.3375	4.0979	-0.81	0.4154
USImports	8.2087	3.9245	2.09	0.0365
JapanExports	0.6685	0.7519	0.89	0.3739
JapanImports	-2.9781	1.5084	-1.97	0.0483
SwissExports	-0.6073	1.2458	-0.49	0.6259
SwissImports	0.2959	0.9653	0.31	0.7592

Table 7.23: Appendix: Chapter 4 - Continued - Estimated coefficients of the logistic regression model. Underlying currency pair is EUR/USD.

	Estimate	SE	z value	p value
Intercept	-0.1101	0.2011	-0.55	0.5839
GBP	1.7834	1.7692	1.01	0.3134
EUR	-1.8971	2.6989	-0.70	0.4821
JPY	-1.7248	1.1179	-1.54	0.1229
CHF	-2.7842	2.8017	-0.99	0.3203
DowJones	-5.2004	2.4846	-2.09	0.0363
Nasdaq	-3.6445	2.0239	-1.80	0.0717
SP500	11.6276	5.4535	2.13	0.0330
FTSE100	2.5558	2.6838	0.95	0.3409
JapanInterbankoffshore	2.5315	2.5611	0.99	0.3229
SwissInterbank	3.1903	3.8157	0.84	0.4031
GBPInterbank	-3.0133	3.8252	-0.79	0.4308
USDInterbank	-3.6318	3.8107	-0.95	0.3406
JPYInterbank	-2.2930	3.0476	-0.75	0.4518
EURInterbank	2.9061	4.5927	0.63	0.5269
CHFInterbank	-4.6351	4.4483	-1.04	0.2974
SP500LowVolatility	-3.3989	3.3982	-1.00	0.3172
CBOEVolatility	0.5419	0.7810	0.69	0.4878
WorldMinimumVolatility	1.1561	4.2987	0.27	0.7880
EuropeMinimumVolatility	-1.1996	4.2752	-0.28	0.7790

Table 7.24: Appendix: Chapter 4 - Estimated coefficients of the logistic regression model. Underlying currency pair is USD/JPY.

	Estimate	SE	z value	p value
FTSEGlobal100	-3.1075	3.7417	-0.83	0.4062
CrudeOil	-2.0478	2.6991	-0.76	0.4480
GoldBullion	3.4541	3.0276	1.14	0.2539
SPCommodityTotalReturn	3.8449	3.2986	1.17	0.2438
BloombergCommodityTR	-4.3412	3.1448	-1.38	0.1675
BankofEnglandBaseRate	1.0306	4.0433	0.25	0.7988
USFederalTargetRate	-5.0556	4.3662	-1.16	0.2469
EuroInterbankOfferedRate	0.4161	5.1338	0.08	0.9354
UKGrossRedemptionYield	1.7569	0.9152	1.92	0.0549
USTreasuryYield	-1.5068	1.8227	-0.83	0.4084
USTreasuryBillRate	8.4660	3.4861	2.43	0.0152
EuroGovernmentBondYield	-0.5094	1.0375	-0.49	0.6234
JapanGovernmentBondYield	-0.9810	0.5965	-1.64	0.1001
SwissConfederationBondYield	1.9359	1.5896	1.22	0.2233
USCPI	5.1138	7.3564	0.70	0.4870
UKCPI	-2.3116	8.6415	-0.27	0.7891
EuroHICP	6.1600	8.6535	0.71	0.4766
JapanCPI	-0.0688	0.9757	-0.07	0.9438
SwissCPI	-1.9718	2.9831	-0.66	0.5086

Table 7.25: Appendix: Chapter 4 - Continued - Estimated coefficients of the logistic regression model. Underlying currency pair is USD/JPY.

	Estimate	SE	z value	p value
UKExports	0.8984	0.9135	0.98	0.3253
UKImports	-1.3515	1.6248	-0.83	0.4055
USExports	-4.5442	3.9760	-1.14	0.2531
USImports	-1.7676	3.5148	-0.50	0.6150
JapanExports	0.8623	0.7629	1.13	0.2583
JapanImports	0.5361	1.4047	0.38	0.7027
SwissExports	-0.3424	1.2231	-0.28	0.7795
SwissImports	1.0367	0.9953	1.04	0.2976

Table 7.26: Appendix: Chapter 4 - Continued - Estimated coefficients of the logistic regression model. Underlying currency pair is USD/JPY.

	Estimate	SE	z value	p value
Intercept	0.0671	0.2357	0.28	0.7759
GBP	0.3747	2.1903	0.17	0.8642
EUR	-6.8978	3.4694	-1.99	0.0468
JPY	0.4457	1.3055	0.34	0.7328
CHF	-14.5361	4.1379	-3.51	0.0004
DowJones	-2.0451	2.5486	-0.80	0.4223
Nasdaq	5.2226	1.9064	2.74	0.0062
SP500	-0.8977	6.1033	-0.15	0.8831
FTSE100	-0.4017	3.1787	-0.13	0.8994
JapanInterbankoffshore	2.0097	3.0894	0.65	0.5154
SwissInterbank	8.6532	4.5695	1.89	0.0583
GBPInterbank	1.1494	4.7210	0.24	0.8077
USDInterbank	1.6026	4.4573	0.36	0.7192
JPYInterbank	-3.7833	3.5748	-1.06	0.2899
EURInterbank	-3.0485	5.2313	-0.58	0.5601
CHFInterbank	-6.1354	5.0550	-1.21	0.2248
SP500LowVolatility	10.2636	4.2546	2.41	0.0159
CBOEVolatility	2.4019	1.1250	2.13	0.0328
WorldMinimumVolatility	9.9977	5.7615	1.74	0.0827
EuropeMinimumVolatility	-12.9533	5.3959	-2.40	0.0164

Table 7.27: Appendix: Chapter 4 - Estimated coefficients of the logistic regression model. Underlying currency pair is USD/CHF.

	Estimate	SE	z value	p value
FTSEGlobal100	-0.3865	4.4802	-0.09	0.9312
CrudeOil	-1.7895	3.0924	-0.58	0.5628
GoldBullion	6.4349	4.2306	1.52	0.1283
SPCommodityTotalReturn	3.8132	3.6142	1.06	0.2914
BloombergCommodityTR	-3.9049	3.7188	-1.05	0.2937
BankofEnglandBaseRate	-11.0641	4.8413	-2.29	0.0223
USFederalTargetRate	-1.0666	5.2168	-0.20	0.8380
EuroInterbankOfferedRate	8.8859	5.8877	1.51	0.1312
UKGrossRedemptionYield	1.1519	1.0652	1.08	0.2795
USTreasuryYield	2.0425	2.2673	0.90	0.3677
USTreasuryBillRate	3.7325	3.3184	1.12	0.2607
EuroGovernmentBondYield	-3.2976	1.2672	-2.60	0.0093
JapanGovernmentBondYield	0.2966	0.6793	0.44	0.6624
SwissConfederationBondYield	-0.0337	1.8830	-0.02	0.9857
USCPI	11.1917	8.6136	1.30	0.1938
UKCPI	-23.5105	10.9588	-2.15	0.0319
EuroHICP	-7.9534	11.2797	-0.71	0.4807
JapanCPI	-0.4142	1.1691	-0.35	0.7231
SwissCPI	10.4579	3.9448	2.65	0.0080

Table 7.28: Appendix: Chapter 4 - Continued - Estimated coefficients of the logistic regression model. Underlying currency pair is USD/CHF.

	Estimate	SE	z value	p value
UKExports	-0.7068	1.1967	-0.59	0.5548
UKImports	-2.6555	2.2690	-1.17	0.2419
USExports	4.5833	4.6010	1.00	0.3192
USImports	-13.5702	4.7943	-2.83	0.0046
JapanExports	0.3719	0.9045	0.41	0.6810
JapanImports	3.3014	1.7561	1.88	0.0601
SwissExports	1.2067	1.5227	0.79	0.4281
SwissImports	-1.4560	1.1690	-1.25	0.2129

Table 7.29: Appendix: Chapter 4 - Continued - Estimated coefficients of the logistic regression model. Underlying currency pair is USD/CHF.