IMPROVING THE PREDICTION ACCURACY OF FINANCIAL TIME SERIES BY USING MULTI-NEURAL NETWORK SYSTEMS AND ENHANCED DATA PREPROCESSING

APPROVED: ________________________
Supervising Professor

____

____

RECOMMENDED FOR ACCEPTANCE: ________________________
Graduate Advisor

ACCEPTED: ________________________
Dean

IMPROVING THE PREDICTION ACCURACY OF FINANCIAL TIME SERIES BY USING MULTI-NEURAL NETWORK SYSTEMS AND ENHANCED DATA PREPROCESSING

by

ROY SCHWAERZEL, B.A., M.B.A., B.S.

THESIS

Presented to the Graduate Faculty of
The University of Texas at San Antonio
in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE

THE UNIVERSITY OF TEXAS AT SAN ANTONIO
July 1996

Acknowledgements

My thanks go to my adviser, Professor Bruce E. Rosen, for his guidance and support throughout the work of this thesis. His insights and discussions has been very valuable for this thesis and for myself. I am grateful to Professor Robert Hiromoto, for many helpful discussions that enhanced the application perspective of this thesis. Thanks for his personal and professional support. I would like to sincerely thank all my teachers through the years, especially Professor Samir Das who always had time for a short question. A special acknowledgment is devoted to the Friedrich Naumann Foundation (Germany) who supported my studies in the United States. My studies would have never been possible without their support. The most important acknowledgment belongs to my parents. Their lifelong support and confidence have always strengthened me through many tough times, God bless them.

ROY SCHWAERZEL

The University of Texas at San Antonio

July 29, 1996

Abstract

The objective of this thesis is the evaluation of the performance improvement that can be achieved by utilizing a master neural network design and enhanced data preprocessing methods for the prediction of currency exchange rates. The prediction of financial time series, in general, has been identified as a challenging task since financial time series have a nonlinear and nonstationary behavior.

The master neural network design is a hierarchical neural network system where the master network estimates an optimal weight combination of the corresponding output of several individual trained neural networks. Using financial statistical features, we train master neural networks to predict the behavior of financial time series.

The performance of the system is evaluated by using two metrics, accuracy and profitability. Modeling accuracy is measured by the metric of predicting the direction of the next return. Profitability is measured by performing a real market simulation with and without consideration of trading costs of 0.05% per round-trip (one buy and one sell) transaction. Finally, the performance of our neural network systems is compared with a simple buy-and-hold model and a linear regression based trading model.

Our experiments demonstrate significant improvements over the previous two models in accuracy and profitability on real market data by using the master networks design. Trading based on the predictions of our neural network systems always produced a higher positive return, even in cases where the regression based trading model or the buy-and-hold strategy generated a loss. Our preliminary investigations find that there is a definite benefit in using neural network models for the prediction of financial time series and the trading of currency exchange rates.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Forecasting the behavior on the financial market is a nontrivial task and has been widely discussed [7][16][17]. The forecasting of financial time series relies on the discovery of strong empirical regularities in observations of the system. Because these regularities are often masked by noise and the time series often have nonlinear and nonstationary behavior, it has been suggested that some financial time series are not predictable. To determine whether a time series is predictable we must examine the random walk hypothesis. Applied to daily rates on the financial market, the random walk hypothesis (RWH) states that a financial time series is defined by constant expected price changes between successive trading days and zero correlation between the price change for any pair of different trading days. The RWH means that the time series is Markov[1] and the best forecast of tomorrow's price requires today's price but not previous prices. If this

[1] A stochastic process (i.e. time series) is called Markov, if the probability of a transition from one state to another depends only on the current state.

is actually true, it is not necessary to spend more resources to develop a good forecasting system. All prediction systems based on the previous behavior of the time series are then useless, because the best prediction of tomorrow's price is today's price plus some constant price change. However, the RWH does not appear valid for many financial time series. Using different trend statistics and tests, Taylor [25] concludes several financial time series have non-random behavior. The financial market is not completely efficient and correlation and trends can sometimes be found within data.

1.2 Research Objectives

The thrust of this thesis is to present and evaluate the performance of neural network systems on financial time series. These systems use a master neural network design and enhanced data preprocessing methods. While hierarchical neural network systems (i.e. master networks) have already been successfully applied to several financial time series predictions [7][16][17], the importance of generating extended statistical features has been ignored. The objectives of this research are to perform experimental studies and to study the prediction accuracy that can be achieved by combining the master network design with extended statistical features. Since prediction accuracy is only an estimate for the profitability on the financial market, a second objective is to perform a real market simulation. Based on the results, we will address the question whether neural network can be trained to reliably predict future currency exchange rates.

1.3 Overview of Thesis

The remainder of this thesis falls into three categories: background material (Chapters 2-4), novel contribution (Chapters 5 and 6), and results (Chapters 7-9):

Chapters 2-4 provide a basic introduction into the interdisciplinary topics of the foreign exchange market, traditional statistical methods for time series analysis, and neural network approaches in time series modeling. Chapter 2 describes the organization of the foreign exchange market - the interbank - including the spot, the forward, and the option market for foreign currencies. Chapter 3 gives a systematic overview about traditional statistical models for time series analysis, such as Autoregression, Moving Average, ARIMA, and Box-Jenkins Models. Modeling of time series using neural network approaches is examined in Chapter 4. It further provides a theoretical framework [12][13] for using an ensemble of neural networks (i.e. master network) which is the thrust of this thesis. Chapters 2, 3, and 4 may be skipped in part or in whole by those who already have a good background in the topics addressed.

Chapter 5 analyzes the statistical properties of the four major currencies, the British Pound, the German Mark, the Japanese Yen, and the Swiss Franc, and defines the statistical features utilized in our experiments, their calculation, and their interpretation. The experimental setup is described in Chapter 6. It contains the design of the neural network architecture, the generation of the input-output pattern, and the specification of various parameters and factors.

Chapter 7 presents the results of our experiments using the metric of predicting the right direction of the next day's or next week's return. A more realistic performance evaluation is explored in Chapter 8 using a real market simulation with consideration of trading costs. These results are compared with results achieved from a linear regression trading model and a simple buy-and-hold strategy. Chapter 9 concludes the thesis by discussing and interpreting the results, and proposing areas of future research.

Chapter 2

The Foreign Exchange Market

The trading of currencies has grown enormously within the previous 50 years. The main reason is the general trend of globalization, the increase of the import and export of commodities all over the world, and an increased interest in international investments. International trade would not be possible without the ability of buying and selling foreign currencies in an efficient way. Although the US Dollar is one of the major currencies in the world, it might not be the accepted means of payment in another country. Investors, tourists, exporters, and importers from the United States must exchange their Dollar for foreign currencies, and vice versa. The purpose of this section is to provide a fundamental description about the organization of the foreign exchange market, its participants, its financial instruments, and its trading costs.

2.1 Purpose of the Foreign Exchange Market

The purpose of the foreign exchange market, also referred as to the interbank market, is to permit international transactions that involve at least one foreign currency. The major currencies are the US Dollar, the British Pound, the German Mark, the

Japanese Yen, and the Swiss Franc. The Canadian Dollar and the French Franc are sometimes considered to be a major currency. The interbank market can be divided into the spot market, in which currencies are traded for immediate delivery (within two business days after the transaction has been concluded), and the forward market, in which currencies are traded for future delivery.

Most currency transactions are channeled through the worldwide interbank market, the wholesale market in which major banks trade with one another. The foreign exchange market is not a physical place, it is rather an electronically linked network of major banks, foreign exchange brokers, and dealers whose function is to bring together sellers and buyers of foreign currencies. It can be seen as a distributed system which consists of the world's leading financial centers: London, New York City, Paris, Zürich, Amsterdam, Tokyo, Toronto, Milan, Frankfurt, as well as other cities.

The financial exchange market is by far the largest financial market in the world. According to recent data from the world's central banks, London is the world's largest currency trading market, with daily turnover in 1989 estimated at $187 billion [22]. The United States is second, at about $129 billion, followed by Japan, at $115 billion.

2.2 The Spot Market

The importance of foreign currency exchange rates is emphasized by the fact that all major newspapers have a listing of current daily exchange rates. Each major currency exchange rate is often listed by four different price quotes. The current spot price, as well as the 30-day, the 90-day, and the 180-day forward prices are quoted. These published quotes are used for trades between participants in the interbank market for trading volumes exceeding $1 million[1]. About 60% of the interbank trades involve

[1] The standard transaction amount in the interbank is about $3 million.

the American Dollar. These quotes can be expressed in two different way, in either the American term, which gives the number of US Dollar per unit foreign currency, or European term, which gives the number of foreign currency units per US Dollar[2]. For nonbank customers, banks are using a system of direct quotation, which gives the price in home currency for a certain quantity[3] of the foreign currency.

Transaction costs do not occur, because banks do not charge a commission on their transactions [23]. The bank's profit comes from the spread between the buying and the selling rate (the bid and ask prices). The bid-ask spread is usually stated as a percentage cost of transacting in the foreign exchange market, which is computed as:

$$spread = \frac{ask\,price - bid\,price}{ask\,price} * 100 \ . \tag{2.1}$$

For common traded currencies, such as the Japanese Yen, the German Mark, or the British Pound, the spread is typically 0.1-0.5%, although less heavily traded currencies have a higher spread [23]. When currency trading is volatile the bid-ask spread is higher to offset the additional risk a trader (bank) must take during theses periods.

More significant exchange risks are involved if transactions must be settled in the future, but the decision to buy or to sell a currency must be made now. For example, consider a large importer that supplies several department stores with a foreign product. Suppose that this importer has a collection period with its customers of 30 days. Almost all the time these large importers are short of money, but because of their strong purchasing power they can sign contracts with the manufacturers with due periods of 90 days. These importers face a high exchange risk, since the price for the foreign currency can

[2]Although *The Wall Street Journal* quotes foreign exchange rates in both American and European terms, the European quotation is commonly used for trades involving Dollar.

[3]The quantity is usually 100 units, but it is one unit in the case of the US Dollar and the British Pound.

change dramatically. The forward market for exchange rates is an additional component of the interbank market. The forward exchange operations are described in the following section.

2.3 The Forward Market

The forward exchange market is a highly demanded and fast growing market. This market lowers the exchange risk for trades involving foreign currencies. This risk, of course, includes the possibility of having a loss or gain, but importers might not be interested in speculations, especially if they are not experienced in this area. For example, in a typical forward transaction, if an US importer signs a contract with a German manufacturer with a payment of 1 million DM due in 90 days, he also will sign an additional forward contract with its home bank to buy 1 million DM in 90 days for an already fixed exchange rate. In this case, the US importer knows exactly the cost for the goods he purchases in US Dollar.

A forward contract between a bank and a customer (which could be another bank) calls for delivery, at a fixed future date, of a specified amount of one currency against the home means of payment; the exchange rate is fixed at the time the contract is entered into. Active forward markets exist for all major currencies, where currently the German Mark is the most widely traded currency. The forward market for less-developed countries are either limited or nonexistent. Forward contracts are usually available for one-month, two-months, three-months, six-months, or twelve-months delivery. Banks also offer a maturity which differs from these standards, in order to meet the customer's needs. The forward rate will consider the future expectations of a given exchange rate, and it will include a wider bid-ask spread to cover the bank's risk.

It is important to mention that the forward contract is not an option contract. In the forward contract, both parties must perform the pre-agreed transaction to the specified conditions. Option contracts are discussed in Section 2.4.

The forward market is of interest for different target groups for different reasons. The major participants can be categorized into traders, arbitrageurs, hedgers, and speculators. Traders want to eliminate or cover their exchange risk for future payments in a foreign currency, as described above. Arbitrageurs seek to earn risk-free profits by taking advantage of differences in interest rates among countries. They use forward contracts to eliminate the exchange risk which is involved in transferring one currency into another. Of course, arbitrageurs can only gain profit in these transactions if the interest rate differences are not fully reflected in the forward rates, which occurs only for a relative short period. Hedgers are interested in forward contracts to protect the home currency value of various foreign-currency-dominated assets and liabilities on their balance sheets. This group includes mostly multinational companies. All three groups seek to reduce their exchange risk.

Speculators are mainly interested in large exchange rate fluctuations. Their business is to profit from taking a currency exchange risk which other parties don't want to take. Speculators try to justify current forward rates and their expectations for spot exchange rates in the future. They will buy forward contracts whenever they expect a significant increase in the spot rate, and they will sell them if they are convinced that the future spot rate will be significant less than the current forward rate.

Forward rates in the interbank market are expressed as a discount from the spot rate, or a premium on the spot rate. If the forward rate is above the spot rate, the foreign currency is called to be at a forward premium, whereas the foreign currency is at a forward discount if the forward rate is below the spot rate. Forward premium and forward discount are closely related to the differences in interest rates on the two currencies, and both objective and subjective expectations. The forward premium or discount is defined as an annualized percentage deviation from the spot rate:

$$forward\,premium = \frac{forward\,rate - spot\,rate}{spot\,rate} * \frac{12}{forward\,contract\,length\,in\,months} \quad . \tag{2.2}$$

The exchange risk was already characterized to be more complex. Even, if the spot market is stable, there is no guarantee that future exchange rates will remain the same. The uncertainty will be reflected in the forward market. Dealers will typically increase the bid-ask spread for longer-term forward contracts to compensate their own risks, especially if price fluctuations exist for a specific currency. Higher spreads might reduce the number of participants which further widen the bid-ask spread for this currency. Spreads in the forward market can be seen as a function of the volume of transactions and the risk which is associated with a forward contract.

2.4 The Currency Futures and Options Markets

Foreign currency futures[4] and options are newer financial instruments in the foreign exchange market. Currency futures are contracts for specific quantities of a given currency; the exchange rate is fixed at the time the contract is entered into, and the delivery date is set by the board of directors of the International Monetary Market (IMM). Although the volume in the future market is relative small compared with the foreign currency forward market, it is growing rapidly. Table 2.1 summarizes the specifications for all currently available currency futures. Contracts in Dutch Guilder and Mexican Peso have been dropped because of larger price fluctuations.

[4]Currency futures were introduced in 1972 when the Chicago Mercantile Exchange opened its International Monetary Market (IMM) division.

	Contract Size	Symbol	Margin Requirement		Minimum Price Change	Value of one point
			Initial	Maintenance		
Australian Dollar	A$100,000	AD	$1,200	$900	0.0001	$10.00
British Pound	£62,500	BP	$2,000	$1,500	0.0002	$6.25
Canadian Dollar	C$100,000	CD	$700	$500	0.0001	$10.00
French Franc	FF250,000	FR	$700	$500	0.00005	$2.50
German Mark	DM125,000	DM	$1,400	$1,000	0.0001	$12.50
Japanese Yen	Y12,500,000	JY	$1,700	$1,300	0.000001	$12.50
Swiss Franc	SFr125,000	SF	$1,700	$1,300	0.00001	$12.50
Months traded:		January, March, April, June, July, September, October, December, and spot month				
Last day of Trading:		Third Wednesday of the delivery				

Table 2.1: Contract specification for foreign currency futures.

Future contracts have several advantages compared with forward contracts because of their standardized conditions. These advantages for example are:

1. Currency futures are attractive for many users, because of the smaller contract size.

2. Currency futures are easier to trade, because they are standardized. Every participant should be familiar with this type of contract. Forward contracts, on the other hand, are individual contracts which can contain almost any condition they agree on.

3. The transaction costs for futures are lower than for forward contracts. Instead of using a bid-ask spread, currency futures require a commission which is relative low compared with the forward market. Although the commissions vary, one buy and one sell (round-trip) of one future will cost approximately $15. For example, this equals less than 0.05% of the value of a British Pound contract.

4. Future contracts can be easily liquidated in a well-organized futures market. In fact, less than 1% of all futures contracts are settled by the actual delivery date compared with 90% for forward contracts.

5. The credit risk is reduced because the Exchange Clearing House becomes the opposite partner to each futures contract.

On the other hand, futures contracts have some disadvantages compared with forward contracts. These disadvantages are:

1. The delivery date of currency futures is restricted to a few specific dates a year, whereas any delivery date is possible for forward contracts.

2. There is no margin required for forward contracts, but a margin is required for futures contracts.

3. Futures contract settlements are made daily via the Exchange Clearing House. Daily gains can be withdrawn, but losses are collected also.

Both the forward and the futures contracts protect the contract holder against unfavorable exchange rate movements. The contract members must exchange their currency to the fixed exchange rate on the maturity date. On the other hand, the contract holders lose their chance of gaining an extra profit for favorable spot rate movements.

2.5 Fundamental and Technical Analysis

There exist two principal, model-based approaches to currency prediction, fundamental analysis and technical analysis. Fundamental analysis relies on the examination of the macroeconomic variables and policies that may influence a currency's prospects. Technical analysis, in contrast, focuses exclusively on past price and volume movements while totally ignoring economic and political factors. Technical analysis can further be subdivided into the charting and trend analysis. Charting analyzes past data and tries to find recurring price patterns, whereas trend analysis seeks to identify price trends using various mathematical computations. Statistical models and artificial neural networks can be applied to both areas of the technical analysis. A slight improvement in the accuracy of future currency exchange rate predictions could lead to better financial management decisions, and also could lower the exchange risk.

The primary function of the foreign exchange market is the facilitation of the international trade and investment by transferring the purchasing power in one currency to another. All financial markets are growing, and they are of interest for different target groups for different reasons. The prediction of future currency exchange rates is a major concern for almost every participant in all markets.

The currency exchange market provides different data, such as the instantaneous tic data, daily opening and closing rates. Daily closing rates for currencies are available from several newspapers, and journals, whereas instantaneous tic data are only available from paid online services. Because the goal of this thesis is the prediction of the currency exchange rate of the next trading days (next trading day, next week's trading day), daily closing rates were chosen for our analysis and modeling. Several important statistical models for financial time series are described in the following chapter.

Chapter 3

Statistical Methods for Time Series Analysis

We discussed the foreign currency exchange market in the previous chapter. This chapter introduces statistical methods used to analyze data recorded at regular time intervals (i.e. daily currency exchange rates). This collection of historical data form a time series which is defined as a sequence of data values $x_1, x_2, ..., x_n$ that are separated by equal time intervals. x_t is the notation for the value of the time series a time t. The analysis of time series is primarily the art of estimating the most likely stochastic process that could have generated an observed time series. In the following, we present statistical methods that trader in the financial market can use for forecasting.

3.1 The Box-Jenkins modeling procedure

Modeling of financial time series is very complex and challenging. Box and Jenkins [2] suggested the following three stages of building a model for a time series:

1. Identification,
2. Estimation, and
3. Diagnostic checking.

The **identification** of an appropriate model for a given time series is done through analysis of actual historical data. It is suggested to have at least 50 observations available to identify the appropriate model satisfactorily [9]. The sample autocorrelation function, discussed in Section 3.4, is the primary tool used in the identification process for determining a tentative model. Whatever model is chosen at the identification stage, it is only a candidate for the final model.

After one or more appropriate time series models have been tentatively identified, the available data series needs to be fitted to the chosen models by **estimating** the model parameters (coefficients). These coefficients must be chosen according to some criteria which express a maximum likelihood. Box and Jenkins [2] favor using the least-square criterion because of its attractive statistical properties. Several excellent examples of the estimation algorithm are provided by Box and Jenkins [2].

Once reasonable estimates of the coefficients in the chosen model have been obtained, **diagnostic checking** is used to decide if the estimated model is statistically adequate. Diagnostic checking is closely related to the identification stage for two reasons. First, if it turns out that the tentatively selected model is inadequate, we have to return to the identification stage. Second, the results at the diagnostic checking stage may also indicate how a model could be improved. The cycle of identification, estimation, and diagnostic checking needs to be repeated until a satisfactory model is found. Once an appropriate model is found, it can be used to forecast future values.

3.2 Stability and Stationarity

The goal of any statistical time series analysis is to find an appropriate model which can simulate the data generation process. In general, a model includes a set of assumptions which are made about this mathematical process. We must distinguish between a specific observed time series, called a realization, and the process that is presumed to have generated this realization. Thus, assumptions for the process that generates the observed time series need to be made.

The prediction of a time series is based on the assumption that the behavior of that time series remains fairly stable. It does not mean that the time series itself must remain unchanged, but that some function of the time series has to remain unchanged. For example, it is assumed that an underlying trend will stay the same over time. These stability assumptions define a stationary process. A process that generates data with the same mean, variance, and correlation at time lag k over time is known as a stationary process.

Most of the financial time series are characterized as nonstationary, but the statistical models discussed in this section are restricted to stationary time series. Fortunately, many nonstationary time series can be transformed into stationary ones. Thus, even statistical models used to analyze stationary time series can be used to analyze nonstationary data.

3.3 Differencing

The most common transformation on nonstationary data to stationary data is the differencing method (which is utilized by the ARIMA model described in Section 3.8). Differencing is a simple operation that involves calculating successive changes in the values of a data series. The following example, adapted from Pankratz [10], describes the differencing process in more detail. To difference a data series, a new variable y_t is defined, which is the change in the observed time series data x_t:

$$y_t = x_t - x_{t-1} \ , \ \ t = 2, 3, \cdots, n \ . \tag{3.1}$$

Using the data in Figure 3.1 we get the following results when we difference the observations:

$$\begin{aligned} y_2 &= x_2 - x_1 \\ y_3 &= x_3 - x_2 \\ y_4 &= x_4 - x_3 \\ &\vdots \\ y_{52} &= x_{52} - x_{51} \end{aligned} \quad . \tag{3.2}$$

An example of a time series and its difference is shown in Figure 3.1. The differencing procedure seems to have been successful: the differenced series in Figure 3.1 appears to have a constant mean. Note that we have lost one observation: there is no x_0 available to subtract from x_1 so the differenced series has only 51 observations. Series y_t is called the *first differences* of x_t. If the first differences do not have a constant mean, we can redefine y_t as the first differences of the first differences:

$$y_t = (x_t - x_{t-1}) - (x_{t-1} - x_{t-2}) \ , \ \ t = 3, 4, ..., n \ . \tag{3.3}$$

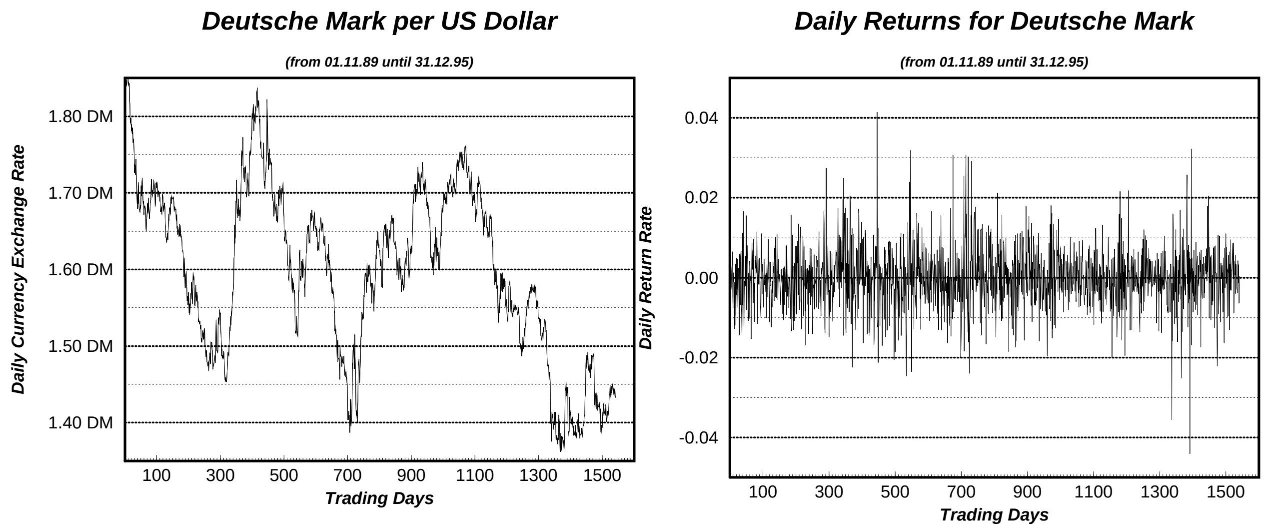


Figure 3.1: Nonstationary and stationary time series.

Series y_t is now called the *second differences* of x_t because it results from differencing x_t twice. Most of the time the first differences are sufficient to generate a data series with a constant mean over historical time. Although the following time series analysis is done for the modified time series of y_t, it is easy to see, that a backward transformation from Equation 3.1 can be performed. Thus, a statistical model for y_t also implies a statistical model for x_t.

3.4 Autocorrelation and Autocovariance

The autocorrelation coefficient ρ_k measures the degree and direction of linear association between the stationary time series random variables x_{t-k} and x_t separated by constant time lag k. The autocorrelation coefficient is defined as

$$\rho_t = \frac{Cov(x_{t-k}, x_t)}{\sqrt{V(x_{t-k}) \cdot V(x_t)}} \ , \tag{3.4}$$

where $Cov(x_{t-k}, x_t)$ is the covariance of x at time lag k and $V(x)$ is the variance of x. Because the theoretical autocorrelation function of the underlying process is unknown, it must be estimated by the sample autocorrelation function. The sample autocorrelation coefficient at lag k, r_k, is a measure of the direction and degree of linear association between an observed time series and the time series lagged by k periods. The sample autocorrelation coefficient can be computed according to the following equation:

$$r_t = \frac{\sum_{t=1}^{N-k}((x_t - \bar{x})(x_{t+k} - \bar{x}))}{\sum_{t=1}^{N}(x_t - \bar{x})^2} \ . \tag{3.5}$$

An individual value r_k for a specific value k is called the sample autocorrelation coefficient at lag k. This coefficient provides information about the probability model of the underlying process. The sample autocorrelation function of a time series is the set of all individual sample autocorrelation coefficients, as defineded in Equation 3.5, where $k = 1,\ 2, \cdots,\ K$. In general, only the first $K \leq N/4$ sample autocorrelations are considered significant for the modeling. In practice the individual sample autocorrelation coefficients are usually calculated by computing the series of autocovariance coefficients, c_k, as defined by

$$c_k = \frac{1}{N}\sum_{t=1}^{N-k}((x_t - \bar{x})(x_{t+k} - \bar{x})) \ . \tag{3.6}$$

The value c_k is called the sample autocovariance coefficient at lag k. Finally, the sample autocorrelation coefficient is computed by

$$r_k = c_k / c_0 \ , \tag{3.7}$$

for $k = 1, 2, \cdots, m$, where $m \leq N - k$. A plot of the set of sample autocorrelation coefficients, r_k, against the lag k is called the **correlogram** of the time series. Similarly, a plot of the set of sample autocovariance coefficients, c_k, against the lag k is called the **covariogram**. Both the correlogram and the covariogram are often a very helpful visual tool. The **estimated partial autocorrelation function** is an additional tool for the representation of the statistical relationship between sets of ordered pairs for a given time series. The basic idea behind the partial autocorrelation analysis is that it measures the relationship (correlation) between the ordered pairs x_{t-k} and x_t, but with accounting for the effects of the intervening values. A more detailed description and definition of the partial autocorrelation function can be found in [5] and [10].

3.5 Autoregressive Model

Most stationary processes can be approximated either as an autoregressive model (AR) or a moving average (MA) model. The autoregressive model of order p is referred to as $AR(p)$. The AR model assumes that the series is predictable from its immediate past values, and it attempts to estimate an observation as a weighted sum of previous observations. An AR model which uses p previous data points in its regression is called an autoregressive model of order p, or simply $AR(p)$[1]. It is usually written as in Equation 3.8, where α_i is an estimated weight for the contribution of a given data point x_{t-p} i time units before the predicted value x_t. The value ϵ_t describes the modeling error:

$$x_t = \sum_{i=1}^{p} \alpha_i x_{t-i} + \epsilon_t \ . \tag{3.8}$$

[1]The AR(1) process is sometimes called the Markov process, after the Russian A. A. Markov.

3.6 Moving Average Model

The moving average method attempts to model an underlying trend of a time series by averaging it for a sufficient period of time. The idea behind this method is to eliminate the influence of shorter term fluctuations. Fitting a moving average process to a given time series turns to the problems of finding the order of the process, and estimating the parameters. A moving average of the values in a time series is found by averaging two or more consecutive values in the series and letting the computed value replace one of the values averaged:

$$x_t = \mu + \epsilon_t - \beta_1\,\epsilon_{t-1} - \beta_2\,\epsilon_{t-2} - \cdots - \beta_q\,\epsilon_{t-q}\ . \tag{3.9}$$

Equation 3.9 describes the MA model of order q, where μ is the mean of the $MA(q)$ process, the ϵ_ts represents the random-shocks (from the averaged value μ), and the β_qs are the estimated model parameter. The negative sign attached to the β_qs is a convention used by Box and Jenkins. It makes no difference whether a negative or positive sign is used, as long it is consistent.

3.7 ARMA Model

The class of ARMA models is formed by combining the AR and MA models discussed in the previous sections. Thus, the mixed autoregressive-moving average model contains p AR terms and q MA terms. The model underlying process is called an ARMA process of order (p, q). The $ARMA(p, q)$ model is represented by

$$x_t = \overbrace{\alpha_0}^{Constant} + \underbrace{\alpha_1\,x_{t-1} + \alpha_2\,x_{t-2} + \cdots + \alpha_p\,x_{t-p}}_{Autoregressive\ terms} + \overbrace{\epsilon_t}^{Noise\ term} \underbrace{-\beta_1\,\epsilon_{t-1} - \beta_2\,\epsilon_{t-2} - \cdots - \beta_q\,\epsilon_{t-q}}_{Moving\ average\ terms}\ , \tag{3.10}$$

where the ϵ_ts are independent.

3.8 ARIMA Model

All previous examples could only be applied to stationary time series, but in practice most time series are nonstationary. By combining differencing, discussed in Section 3.3, and the mixed ARMA models from the previous section, Box and Jenkins were able to propose a parsimonious class of models called the **autoregressive integrated moving-average model** or simply the ARIMA model. In the following, the ARIMA model is referred to as $ARIMA(p, d, q)$ model, and is described in Equation 3.11. The time series y_t is formed from the original series x_t by differencing, as discussed in Section 3.2. The notation p refers to the number of autoregressive terms in the model. The ARMA model, discussed in Section 3.7, is still the underlying model for the ARIMA class of time series, however, the nonstationary time series needs to be transformed into a stationary one. If a time series is not stationary, then changes in the successive time series values, known as the first differences, are often stationary. If the first differences are not stationary, then the differences of the differences (second differences), or the differences of logarithms of the series values are often stationary. The notation d refers to the degree of differencing necessary to obtain a stationary time series. The notation of q refers to the number of lagged error terms that make up the moving average part of the model.

$$y_t = \overbrace{\alpha_0}^{Constant} + \underbrace{\alpha_1\, y_{t-1} + \alpha_2\, y_{t-2} + \cdots + \alpha_p\, y_{t-p}}_{Autoregressive\ terms} + \overbrace{\epsilon_t}^{Noise\ term} - \underbrace{\beta_1\, \epsilon_{t-1} - \beta_2\, \epsilon_{t-2} - \cdots - \beta_q\, \epsilon_{t-q}}_{Moving\ average\ terms} \quad (3.11)$$

ARIMA models are often estimated by using an iterative nonlinear least square estimation procedure. Matrix oriented software packages such as GAUSS, MATLAB, or SAS are usually used for the data analysis and parameter estimation of ARIMA models.

A more generalized model, the Box-Jenkins seasonal model, also referred to as the SARIMA model, has been developed by Box and Jenkins [2] in order to deal with seasonality. The SARIMA model is a general multiplicative seasonal ARIMA model. The time series y_t is formed from the original series x_t not only by simple differencing (removing the trend) but also by seasonal differencing (removing the seasonality). More details are given by Box and Jenkins [2], but there are many computer programs now available to fit a seasonal model to a given time series.

Chapter 4

Neural Networks

4.1 Introduction to Neural Networks

Researchers and even computer scientists are exited about the enormous power of the human brain and the capability in solving problems which are nontrivial or even impossible with the newest computer technology. Neural networks behave, react, and learn rather than execute programs (see Figure 4.1). A new research area, the connectionism, has been opened to explore and simulate the mechanism underlying the biological model. The connectionism developed computation models which were inspired from the biology rather than from conventional computing techniques. Artificial neural networks try to mimic the nature's approach to achieve these properties.

4.1.1 The Biological Model

What makes the biological neural networks so powerful? Although the functionality of a single neuron is very simple and the information transfer is much slower (about

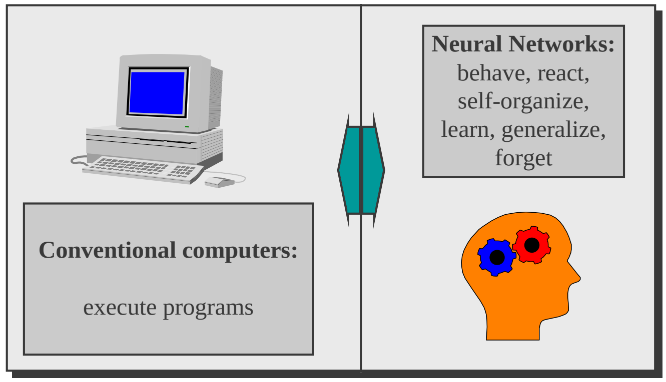


Figure 4.1: Conventional computers versus Neural Networks.

1,000 computations per second) than in today's computers (200 MIPS), their power arrives from the enormous large number of these simple processing units. The human brain consists of 100 to 1,000 billion neurons with about 10,000 connections per neuron.

Neurons are very simple information processing units that gain their power from the large number of connections to other neurons, the ability to process information independently from other neurons, and the ability to self-organize[1]. For a better understanding, some basic terms of the neurobiology will first be explained.

A neuron consist of the **cell body (soma)**, the **axon**, and the **dendrites** (see Figure 4.2). It is important to mention that each neuron has only one axon, but it may have up to 10,000 dendrites. The following tasks are assigned to the three main structures of a nerve cell:

1. Dendrites: Receive information,
2. Cell body: Process information,
3. Axon: Transmit information.

[1]Self-organization is the ability of neural networks to develop their own heuristics to learn the underlying rules of a system.

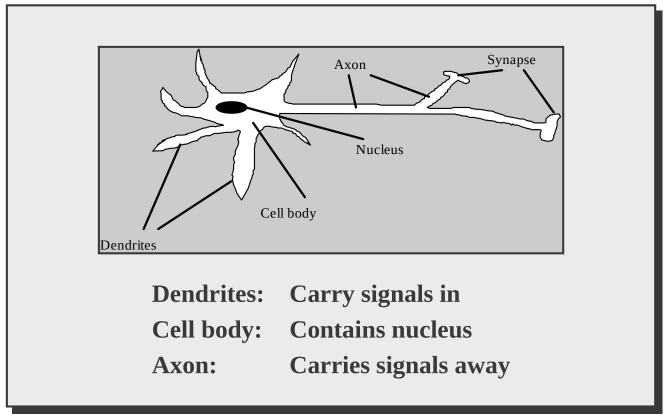


Figure 4.2: Main parts of a simple neuron as found in the retina.

Synapses are the gaps at which a nervous impulse (an electro chemical signal) passes from one neuron to another neuron. Impulses pass in one direction only and the process can be broken down into the following three steps:

1. Signals from other connecting neurons come across the synapses to the neuron's dendrites. The incoming signals (inputs) are weighted depending on the strength and type of synapse. If there is a stronger synaptic connection this input is given a larger weight. Synapses can also be divided into excitatory (positive weight) and inhibitory (negative weight) connections. The effects of all weighted input signals are summed up at the cell body.

2. This weighted sum is compared with an internal threshold for the neuron. If the sum exceeds this threshold, the neuron fires (sends an output signal through its axon). Otherwise the nerve recovers to its original state, and prepares to produce the next new impulse.

3. The response of neurons is rather a dynamic state response than a static one. The synaptic connections or the threshold may change, i.e. frequently used synapses

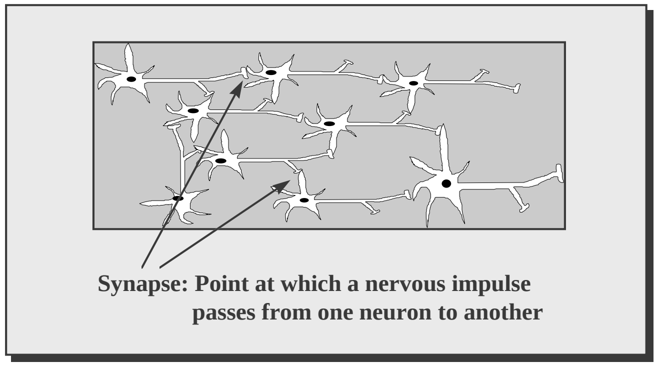


Figure 4.3: Nerve structures formed by a bundle of simple neurons.

can get stronger, and new connections can be generated over time. This ability to adjust signals is a mechanism for learning.

Bundles of neurons form nerve structures as shown in Figure 4.3. In a simplified scenario, nerves structures receive impulses from receptor organs (i.e. eyes), pass them through several layers of nerve cells until they reach effector organs (i.e. muscles).

4.1.2 The Artificial Neural Network

Artificial neural networks, also referred to as neural networks, consist of **simple processing units** (see Figure 4.4) which have the same basic function as the neurons described in the previous section. These basic functions are:

- Receiving several inputs,
- Evaluating the importance of these inputs by assigning appropriate weight coefficients to these inputs,
- Calculating the weighted sum of these inputs,
- Comparing this weighted sum with some threshold, and
- Determining an appropriate output value.

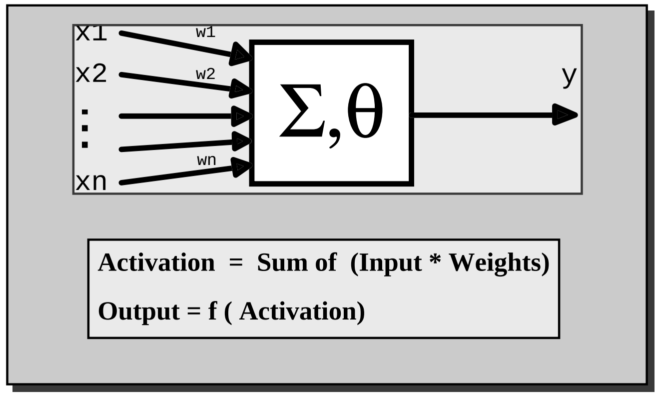


Figure 4.4: Simple processing unit: The basic component of an artificial neural network.

After calculating the weighted sum (activation), the activation is passed through a transfer function. The properties of the transfer function need to mimic the nerve cell which either fires or does not fire. For example, this could be achieved by a simple step function, which returns 1 if some threshold is exceeded, or 0 otherwise. It turns out that a continuous transfer function is needed for some learning methods such as the back-propagation algorithm. Therefore, the most common approach is using a sigmoid[2] transfer function such as

$$y = f(x) = \frac{1}{1+e^{-x}} \quad . \tag{4.1}$$

The exiting features of this function are that both the function and its derivatives are continuous, and that this function introduces nonlinearities into the network.

Combining several of these single processing units into layers and interconnecting several layers generates a neural network. Figure 4.5 shows a fully connected **feed-forward network** consisting of 10 input units in the input layer, 4 hidden units in the

[2]A sigmoid function is an S-shaped function that approaches a minimum and maximum value at the asymptotes.

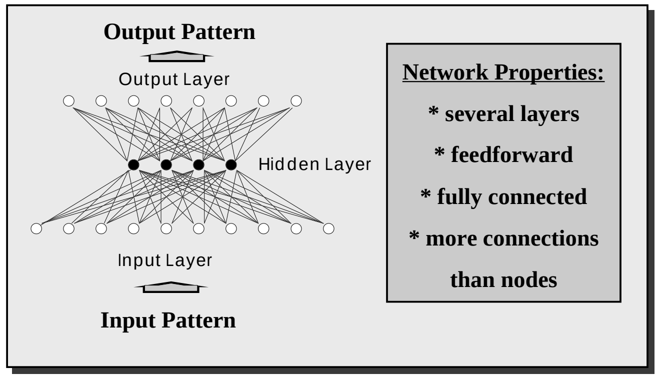


Figure 4.5: Feed-forward network: The combination of many single processing units.

hidden layer, and 8 output units in the output layer. The response or output of the neural network is only determined by the weight values for each of the connections.

Training of a neural network is basically the search of an optimal weight set that minimizes a defined error (i.e. mean square error) between the network's results and the desired results. The most popular used training methods for feed-forward networks are the **back-propagation algorithm** [11][20][31] and its variants [1]. These algorithms give a prescription for changing the weights in any feed-forward network to learn a training set of input-output pairs. Mathematically, the basis is simply gradient descent. These learning algorithms perform a forward and a backward calculation through the networks as shown in Figure 4.6.

In the forward pass, the outputs for a specific input pattern are calculated, and the error at the output units is determined. In the backward pass, the output unit error is used to change the weights on the output unit proportional to the error at that unit. Then the error at the hidden units is calculated and their weights are updated, respectively. One forward and one backward pass is performed for each input-output pattern. Presenting all input and output pairs exactly one time to the neural network is defined as one training cycle. Many training cycles are necessary to train a neural

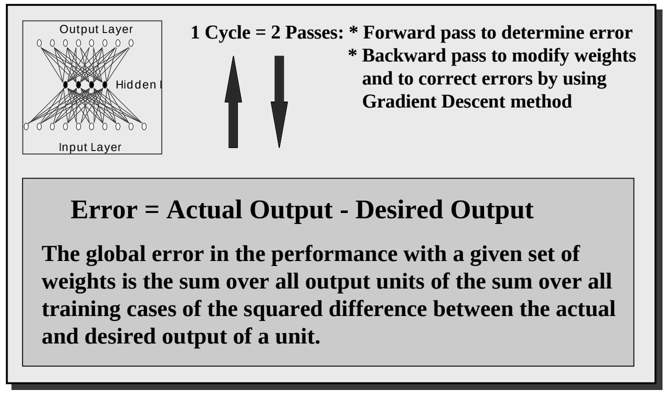


Figure 4.6: The back-propagation algorithm: A very powerful algorithm for supervised training of neural networks.

network. The training is usually stopped when a sufficiently low enough error or a maximum number of training cycles is reached. In general, the methodology "the longer the training, the better the result" does not apply for the training of neural networks, especially for the prediction of financial time series.

4.1.3 Cross-Validation

It is generally true that the generalization error decreases in an early period of training, reaches a minimum and then increases as the training continues, while the training error monotonically decreases. It is considered better to stop training before the neural network starts learning the noise within the training data but after the neural network has discovered the basic model of the data. If the network learns the noise, **overfitting** or **overtraining** can occur. There are several method to avoid overfitting, such as using regularization terms [15], model selection methods [8], or cross-validation stopping [6]. The **cross-validation** method, a simple and effective method, is used for our experiments. In cross-validation, the original training set is divided into two disjoint sets. The first set is used to train the neural network and to minimize the mean square

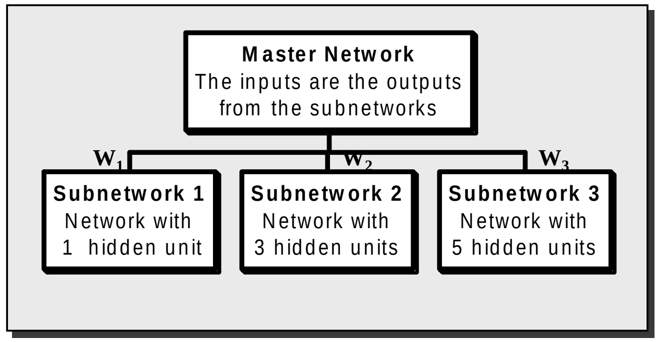


Figure 4.7: Master network: An approach for a hierarchical neural network design.

error (MSE). The second set is used only to test the current MSE performance of the neural networks. After several training cycles using the training set, the training is temporarily stopped and the MSE performance of the neural network is measured for the cross-validation set. If the MSE performance of the cross-validation set is minimized, the training is stopped.

4.1.4 The Master Network Design

Although single neural network systems as introduced in the previous sections, have been used for the prediction of financial time series, neural network approaches (i.e. master network) combining the results of several single neural networks have shown improved performance as shown in [21]. The master network design is a hierarchical neural network architecture consisting of several independently trained neural networks and one master network. The outputs of each subnetwork are the inputs to the master network. Figure 4.7 shows a master network with three subnetworks. The master network is trained to find an optimal weight combination as that produces the minimum of the mean square error between the desired and master network output with respect to the distribution of the training data. The master network design is based on the idea that an ensemble of neural networks will perform better than any individual neural network.

4.2 Ensemble Methods

This section provides a theoretical framework for using an ensemble of neural networks (i.e. master network). Consider the following regression problem:

$$y = f(x) + \eta \ , \tag{4.2}$$

where y is the desired output of a system, $f(x)$ is defined as the expected value of y given x, $f(x) = E[y|x]$, and η is independent with identically distributed zero-mean noise. Thus, y is a random variable with mean $f(x)$. Suppose there exist two finite data sets whose elements are all independent and are chosen randomly from a single distribution. These two sets are the training data set $TR = \{(x_m, y_m)\}$ and the cross-validation data set $CV = (x_l, y_l)$. Further, suppose the training set TR is used to generate a set of neural network functions, $F = f_i(x)$, where each element, f_i is a neural network which approximates the real function $f(x)$. The goal of ensemble learning is to find the best approximation of the desired output y using F.

4.2.1 Naive Estimator

The most common ensemble method uses a naive estimator, $f_{Naive}(x)$, which is the regression function of F with the smallest mean square error (minimum) relative to y. The naive estimator is defined by

$$f_{Naive}(x) \equiv min_i\{MSE[f_i]\} \ , \tag{4.3}$$

where $MSE[f_i]$ is the mean square error of the regression function f_i.

The neural network with the smallest mean square error is the one that performs best for the cross-validation data, but it may not perform very well for the testing data which may have a different distribution. Since the nonoptimal networks probably represent different local minima in the function space, it is possible that some of these networks will perform better than the naive estimator for data other than the previously seen training and cross-validation data. The average of the individual prediction of each network in the population F would be a better choice.

4.2.2 Basic Ensemble Method

The Basic Ensemble Method (BEM) regression function, $f_{BEM}(x)$, averages all regression estimators. It is defined by

$$f_{BEM}(x) \equiv \frac{1}{N}\sum_{i=1}^{N} f_i(x) \ , \tag{4.4}$$

where f_i is an individual function estimator of y, and N is the population size of all function estimators F.

Perrone [13] presents an important proof that averaging regression estimates can reduce the mean square error by a factor of N when compared with the performance of each of the subnetworks,

$$MSE[f_{BEM}] = \frac{1}{N}\overline{MSE} \ . \tag{4.5}$$

The performance improvement can be illustrated by Figure 4.8 on a simple classification problem. Hyperplanes 1 and 3 solve the classification problem for the training data, but hyperplane 2 is the optimal solution. Hyperplane 2 is the average of hyperplanes 1 and 3.

The BEM has a better performance than the naive estimator; however, it cannot find the best possible combination of all estimates in F. Taking the simple average of the network outputs, as shown in Equation 4.4, does not work well when correlated networks produce very similar errors in response to the same input vector. The correlated networks

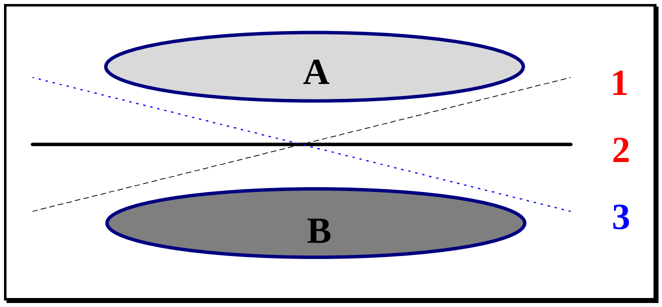


Figure 4.8: Simple classification problem.

will be over-represented using the simple average. One simple extension is to consider the correlation between each function f_i when composing F. This idea leads into the Generalized Ensemble Method.

4.2.3 Generalized Ensemble Method

The Generalized Ensemble Method (GEM) generates a regression estimate which is as low or lower than the naive or the simple averaging estimators. The GEM is the linear combination of the estimators in F based on the empirical mean square error [12]. The GEM regression function, $f_{GEM}(x)$, is defined by

$$f_{GEM}(x) \equiv \sum_{i=1}^{N} \alpha_i f_i(x) \ , \tag{4.6}$$

where the α_is are weighting parameters that satisfy the constraint $\sum \alpha_i = 1$. Each α_i is defined by

$$\alpha_i = \frac{\sum_j C_{ij}^{-1}}{\sum_k \sum_j C_{kj}^{-1}} \ , \tag{4.7}$$

where C_{ij} are the elements of the covariance matrix of the errors from the function estimators f_i and f_j [12].

Perrone's BEM and GEM have shown that by averaging in functional space, a system of neural network estimators can be constructed which will be guaranteed to improve the MSE performance. These ensemble methods can be used with any set of neural networks. The averaging method has the property that it can utilize the local minima [13]. Although ensemble methods are a general approach, our studies focus on the utilization of ensemble methods for financial time series only.

Chapter 5

Statistical Properties of Time Series

5.1 Statistical Features

This section analyzes the statistical properties of the daily rates and returns for four major currencies - the British Pound, the German Mark, the Japanese Yen, and the Swiss Franc - against the US Dollar for the period from November 1, 1989 to December 31, 1995. Because the experiments will be done using daily returns, this analysis focuses on daily returns rather than on daily exchange rates. The reason for using daily returns is that the returns have a stationary behavior whereas the daily exchange rates are nonstationary. Figure 5.1 shows the plots for the daily currency exchange rates and returns for all four major currencies for the time between November 1, 1989 to December 31, 1995. The statistical properties for the daily exchange rates are summarized in Table 5.1, the one for the daily returns can be found in Table 5.2. The following sections give a more detailed descriptions of all utilized statistical features, their calculation, and their interpretation.

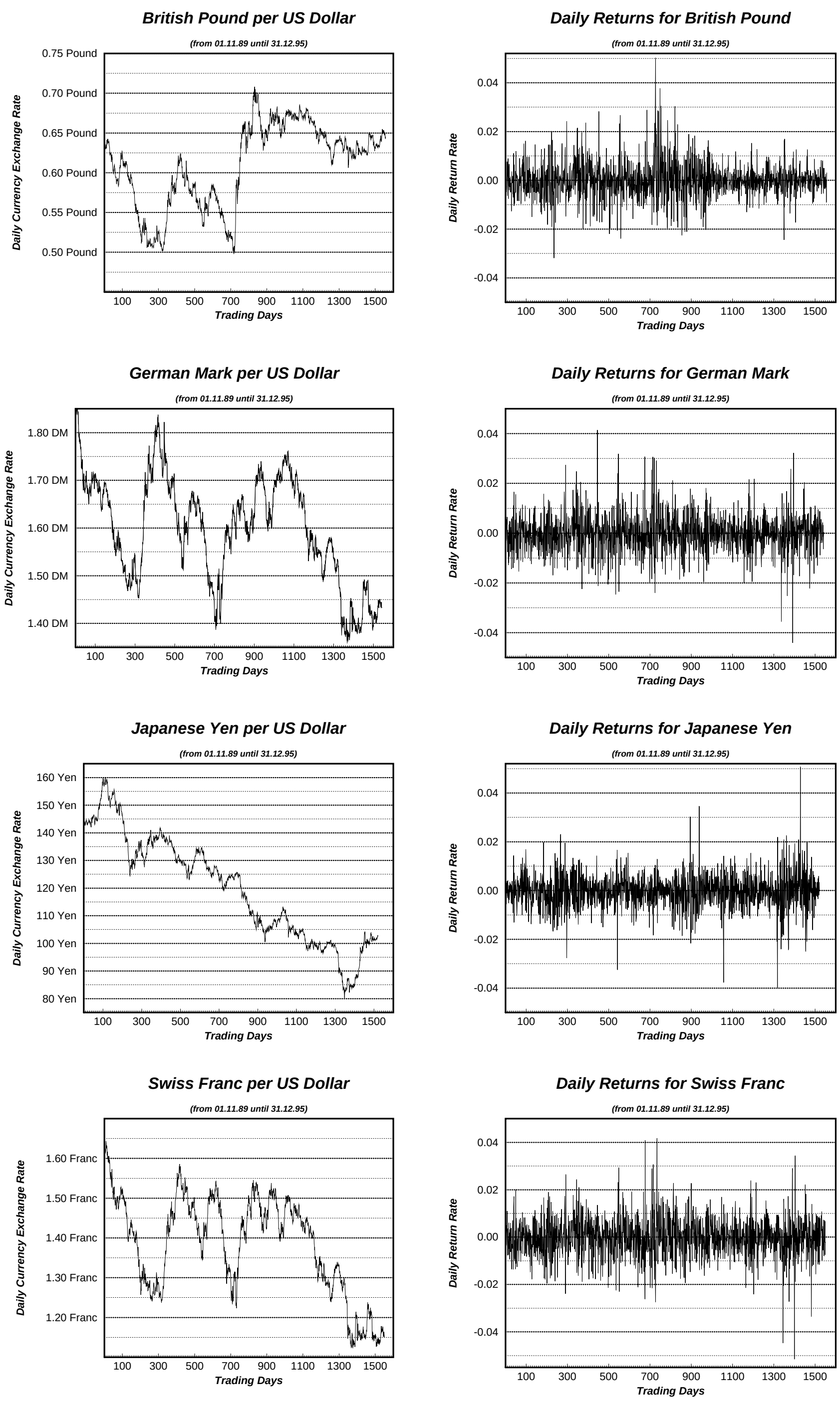


Figure 5.1: Daily currency exchange rates and returns: (a) British Pound, (b) German Mark, (c) Japanese Yen, and (d) Swiss Franc against US Dollar from November 1, 1989 to December 31, 1995.

	Daily currency exchange rates			
	British Pound	German Mark	Japanese Yen	Swiss Franc
Trading days	1,558	1,543	1,522	1,552
Lowest rate	0.4979590	1.36200	80.30	1.12360
Highest rate	0.7081150	1.86370	160.10	1.64350
Average	0.6093671	1.59616	119.54	1.38192

Table 5.1: Statistics of daily rates for major currencies.

	Daily returns			
	British Pound	German Mark	Japanese Yen	Swiss Franc
Arithmetic Mean	0.0000369	-0.0001330	-0.0001924	-0.0001853
Variance	0.0000511	0.0000584	0.0000481	0.0000686
Standard Deviation	0.0071487	0.0076391	0.0069386	0.0082797
Skewness	0.5660350	0.1696712	0.0056861	-0.0200454
Kurtosis	6.8067127	5.6190482	7.3249842	5.7504624
Minimum	-0.0318592	-0.04413	-0.03985	-0.05150
Maximum	0.0502554	0.04144	0.05072	0.04172
Range	0.0821146	0.08557	0.09057	0.09322

Table 5.2: Statistics of daily returns for major currencies.

5.1.1 Mean: Measure for Central Tendency

There are three main types of measure for describing the central tendency for a set of observations:

1. the means: arithmetic, harmonic, geometric,
2. the median, and
3. the mode.

Although all of these measures have their advantages in special cases, the arithmetic mean is the most important one. Therefore, the arithmetic mean will be discussed further in detail. A brief description of the other indices can be found in Section 5.1.5.

As previously stated the arithmetic mean (commonly also abbreviated to "average" or "mean") is the most useful measure of the central tendency for a set of observations $x_1, x_2, ..., x_n$ of a time series. It is the sum of the individual observations x_i divided by the number of observations n. The formula is

$$\bar{x} = \frac{1}{n} \sum_{t=1}^{n} x_t \ . \tag{5.1}$$

A further distinction needs to be made between the mean of a set of observations and the mean of the population. The sample mean $\bar{x}$ is an estimate of the population mean μ, $\bar{x}$ becomes a better estimate if the number of observations n is large.

5.1.2 Standard Deviation: Measure of Dispersion

Any index of central tendency, such as the mean, summarizes only one aspect of the distribution of a time series. It is easy to find two distributions which have the same mean, but have no further similarities. There are other measures that describe the dispersion or variability of a time series, the tendency for observations to depart from the central value. There are five main measures for describing the variability for a set of observations:

1. Range: Minimum and Maximum of observed values,
2. Variance or Standard Deviation,
3. 10- and 90-Percentiles,
4. Semi-Interquartile range, and
5. Mean Absolute Deviation.

Because the variance and standard deviation are the descriptors most used as an index of dispersion, both are described in detail. The other indices are briefly described in Section 5.1.5. The variance of a set of n observations $x_1, x_2, ..., x_n$ is calculated as

$$s^2 = \frac{1}{n-1} \sum_{t=1}^{n} (x_t - \bar{x})^2 \ , \tag{5.2}$$

where $\bar{x}$ is defined by Equation 5.1. The index s^2 is called the sample variance and its positive square root s is called the sample standard deviation. A distinction must be drawn between the population standard deviation σ and the sample standard deviation s. The sample standard deviation s becomes a better estimate of σ with a large number of observations.

5.1.3 Skewness

The distribution of a given time series will probably not be completely symmetrical. The degree of asymmetry is referred to as skewness and is calculated as

$$b = \frac{1}{n-1} \sum_{t=1}^{n} (\frac{(x_t - \bar{x})^3}{s^3}) \ . \tag{5.3}$$

A distribution has a positive skewness if the long tail is on the side of the high values of x. Similarly, the skewness is negative if the long tail is on the side of the low values of x. A symmetric distribution has a skewness of zero.

5.1.4 Kurtosis

The degree of "peakedness" in a distribution is expressed by the kurtosis which is defined as

$$k = \frac{1}{n-1} \sum_{t=1}^{n} (\frac{(x_t - \bar{x})^4}{s^4}) \ . \tag{5.4}$$

The kurtosis is equal to 3 for a Normal distribution. If the kurtosis is greater than 3, the distribution is more sharply peaked than the Normal distribution (i.e. if it has long "tails"). If the kurtosis is less than 3, the distribution is flatter than the Normal distribution.

5.1.5 Other Statistical Properties

Although the more common indices for summarizing the properties of time series have already been discussed in the previous sections, there are several additional indices. They are the median, mode, range, percentile, Semi-Interquartile Range, and Mean Absolute Deviation.

The **median** is the central member of a series, when the observations are arranged in order. For example, there are equal numbers of observations greater than and less than the median. If the number of observations, n, is odd, this definition is complete. If n is an even number, the arithmetic mean of the two central values is usually calculated as the median. The median has the practical advantage as a measure of location, because it is not influenced by extreme observations.

The **mode** is the value of the observation which occurs most frequently, i.e. for which the frequency of occurance is maximum. The mode is not a unique index, because more than one value can occur with the same frequency.

The **range** is the simplest of all measures for dispersion. It is simply the difference between the lowest and highest value of the sample. The larger the difference, the higher the variability. In most cases, the range is not a good measure because it is strongly influenced by "outliers" which are far away from most values in the distribution.

If the data set is arranged in order, the x-**percentile** is the value where x% of the observations are less or equal, and (100-x)% of the observations are greater or equal to this value. For example, the median is the 50-percentile. The percentiles that divide the data set into four parts at 25, 50, and 75% are called quartiles. The range between the third quartile, Q_3 , and the first quartile, Q_1, is called **Semi-Interquartile Range (SIQR)**. It is calculated as

$$SIQR = \frac{Q_3 - Q_1}{2} = \frac{x_{.75} - x_{.25}}{2} \ , \tag{5.5}$$

where $x_{.75}$ is the 75-percentile and $x_{.25}$ is the 25-percentile of the time series.

The **Mean Absolute Deviation (MAD)** is another measure of dispersion, which is easier to calculate than the standard deviation. It is defined as

$$MAD = \frac{1}{n} \sum_{i=1}^{n} |x_i - \bar{x}| \tag{5.6}$$

5.2 Predictability

A time series in general cannot be predicted with certainty since each value is generated by an underlying process and the output of that process is uncertain. Although an observed value of a time series is associated with a random variable, a time series variable in one period is often associated with, or is often influenced by, a variable in a preceding period. These dependencies can be expressed by the **Autocorrelation** and the **Hurst Coefficients** discussed below.

5.2.1 Autocorrelation Coefficient

The autocorrelation coefficient has already been discussed in Section 3.4. The autocorrelation of a stationary time series at lag k, ρ_k, is estimated by the sample autocorrelation coefficient. The sample autocorrelation coefficient at lag k, r_k, is a measure of the direction and degree of linear association between an observed time series and the time series lagged by k periods. The sample autocorrelation coefficient is computed according to Equation 3.5. Table 5.3 summarizes the autocorrelation coefficients for different lags for all major currencies. The correlograms[1] are plotted in Figure 5.2. In all four cases shown in the correlograms, there is a positive autocorrelation at lags 5, 10, and 15. This indicates a day of the week effect, i.e. a weekly seasonal cycle. Further, there is a negative autocorrelation at lags 1, 6, and 11, except for lag 1 of the Japanese Yen which has a low positive autocorrelation coefficient. This indicates alternating returns from one trading day to the next trading day. The autocorrelation coefficients indicate that all four major currences may be predictable because all underlying processes show a seasonal pattern.

	Foreign Currency			
Time lag	British Pound	German Mark	Japanese Yen	Swiss Franc
1	-0.007	-0.039	0.008	-0.031
2	0.035	0.000	0.015	-0.005
3	0.010	0.002	-0.033	0.000
4	0.019	0.006	-0.016	-0.011
5	0.046	0.036	0.009	0.036
6	-0.033	-0.058	-0.040	-0.044
7	-0.024	0.013	0.052	0.011
8	0.016	-0.003	-0.001	0.018
9	-0.009	-0.005	0.006	-0.008
10	0.091	0.049	0.071	0.039
11	-0.039	-0.023	-0.021	-0.019
12	0.019	0.013	0.030	0.008
13	0.027	-0.012	0.022	0.004
14	-0.081	-0.016	0.034	-0.026
15	0.074	0.008	0.034	0.007
16	-0.008	-0.015	0.004	0.011

Table 5.3: Autocorrelation coefficients for various time lags.

[1] A plot of a set of autocorrelation coefficients against various time lags is called the correlogram of the time series.

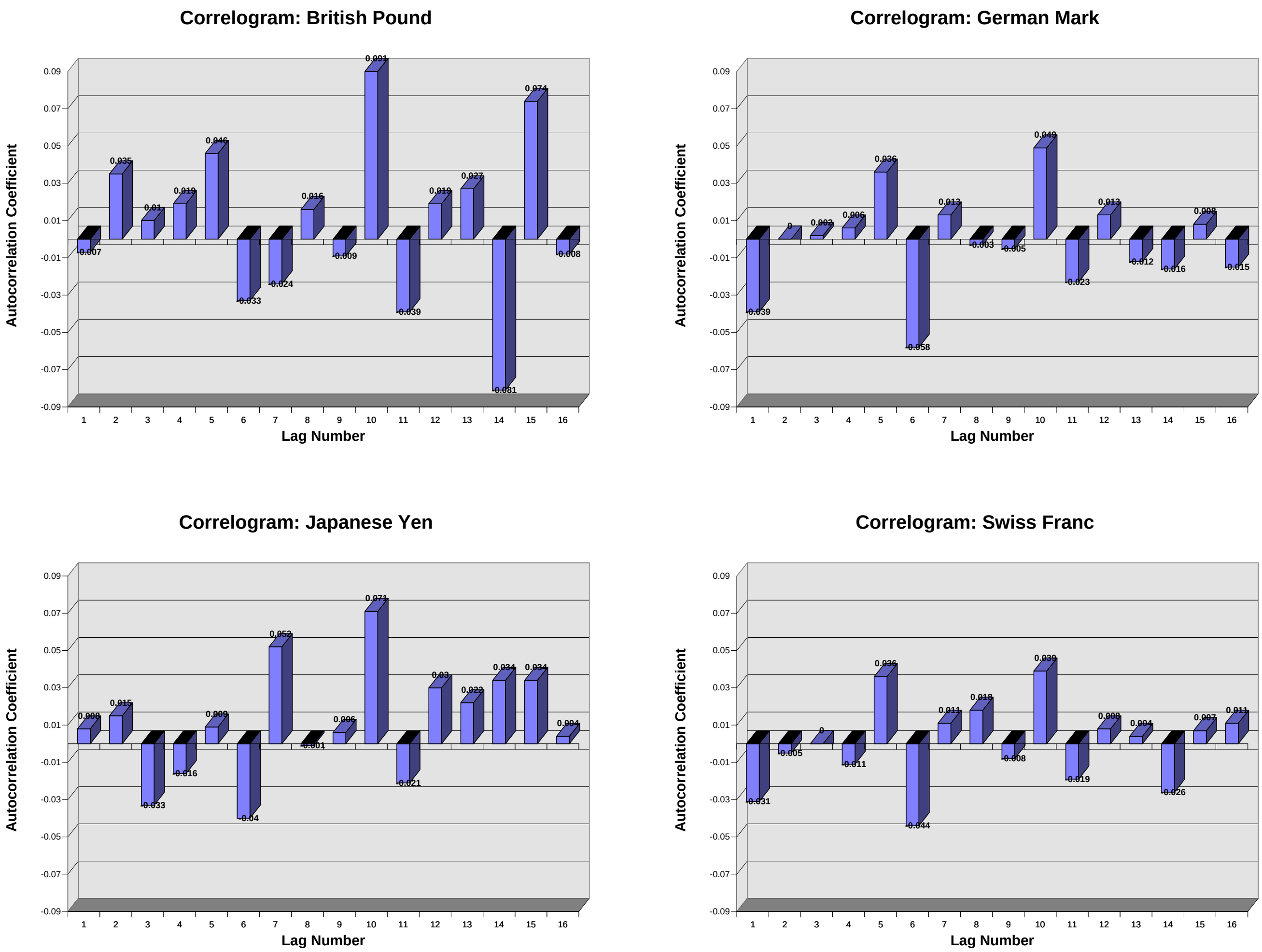


Figure 5.2: Correlogram for daily returns: (a) British Pound, (b) German Mark, (c) Japanese Yen, and (d) Swiss Franc against US Dollar.

5.2.2 Hurst Coefficient

Rescaled range analysis, also called Hurst analysis, is a statistical method to evaluate whether or not memory patterns are present in a time series of data. Rescaled range analysis provides the Hurst coefficient, H, which is a measure of the bias or trend in a time series. Rescaled range analysis follows the following relationship:

$$\frac{R_N}{S_N} = \left(\frac{N}{2}\right)^H \quad , \tag{5.7}$$

where N is the time window, H the Hurst coefficient, and $\frac{R_N}{S_N}$ is the dimensionless ratio of the average range divided by the standard deviation of the daily fluctuations over N. By taking the log of Equation 5.7, we obtain:

$$H = \frac{\log \frac{R_N}{S_N} - \log 2}{\log N} \quad . \tag{5.8}$$

The slope of the logs provides an estimate of the Hurst coefficient. The Hurst coefficient is a value in the range between 0 and 1 and is equal to 0.5 for a random process. A Hurst coefficient greater than 0.5 indicates a memory effect, where there is a bias to enforce the current trend. That is called persistence. A Hurst coefficient less than 0.5 indicates a negative bias, and is called antipersistence. The Hurst coefficients of the daily returns for all four major currencies are summarized in Table 5.4.

	Hurst coefficient for daily returns		
Foreign Currency	first half of time series	second half of time series	entire time series
British Pound	0.518010	0.515229	0.519535
German Mark	0.508209	0.503202	0.509088
Japanese Yen	0.528002	0.520208	0.533237
Swiss Franc	0.514062	0.507526	0.513787

Table 5.4: Hurst coefficients of daily returns for major currencies.

The Hurst coefficients indicate a small but persistent tendency for all four time series. The Hurst coefficient from the second half of each time series is always closer to 0.5 than the one from the first half. This indicates the more current data is more difficult to predict. Further, the Hurst coefficient for the entire time series is slightly larger for all but the Swiss Franc daily return. This indicates that the entire time series contains useful information for a valid prediction. Based on the estimates for the Hurst coefficients for our currency exchange rates, the following experiments are limited to the prediction of the German Mark, which has the smallest Hurst coefficient, and the Japanese Yen, which has the largest Hurst coefficient. The larger Hurst coefficient for the Japanese Yen indicates a better memory effect, which may also result in better predictions.

Chapter 6

Experimental Design

Several network systems each consisting of multiple single networks and a master network were applied to predict the next day's and the next week's return for the German Mark and the Japanese Yen exchange rates. The design of the neural network architecture, the generation of the input and output pattern, and the specification of parameters and factors are described in this chapter.

6.1 Time Windowing

The analysis of the autocorrelation coefficients in Section 5.2.1 indicates that future values are determined not only by the last but by a series of values from the past. In order to capture this dynamic of time series the neural network approach of time windowing has been developed, and was used for our experiments. The idea behind time windowing is the transposition of the original temporal dimension into a spatial vector [16]. This common method partitions the time series data into fixed size sub sequences. As shown in Figure 6.1, two windows W_m^{input} and W_n^{output} of fixed sizes m and n are used to partition the original data set.

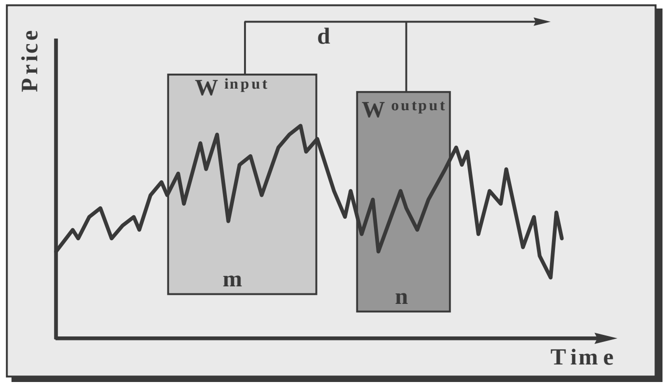


Figure 6.1: Windowing: A method of looking for correlation in the time series.

It is assumed that the sequence in W_m^{input} is correlated to the sequence in W_n^{output}, and that the regularities of a time series can be recognized by moving both windows over the entire data set. Each pair of windows W_m^{input} and W_n^{output}, separated by the distance d, is taken as the input and output vector for our neural networks. If we are predicting only one return at a time, then the window size of W^{output} is one $(n = 1)$. The choice of the window size of W^{input} and of the distance d is more critical, and often determines the quality of the prediction. The analysis of the autocorrelation coefficients in Section 5.2.1 indicates the existence of an underlying day-of-the-week effect[1], because all four major currencies have a positive autocorrelation coefficient for lags 5, 10, and 15. These lags correspond to daily returns exactly one, two, and three weeks previous to the current day[2]. Considering this seasonal effect, two window sizes, W_5^{input} and W_{10}^{input}, were chosen for our simulations. In the case of $m = 5$ and $n = 1$, time windowing will generate a set

[1] The day-of-the-week effect describes a seasonal dynamic of a time series. According to Taylor's analysis [25], Monday returns have slightly higher variance than other returns and sometimes have a negative, but small, mean.

[2] The data set for our currency exchange rates contains only trading days that are weekdays. Therefore, a time lag of five trading days represents the same day of the week.

of pairs, where each pair consists of an input vector of size 5, and an output vector of size 1 separated by distance $d = 1$. More formally, we can write:

$$\begin{aligned}
&input\,pattern_1 := \{x_1, x_2, x_3, x_4, x_5\} && output\,pattern_1 : \{x_{5+d}\} \\
&input\,pattern_2 := \{x_2, x_3, x_4, x_5, x_6\} && output\,pattern_2 : \{x_{6+d}\} \\
&input\,pattern_3 := \{x_3, x_4, x_5, x_6, x_7\} && output\,pattern_3 : \{x_{7+d}\} \\
&\qquad\vdots && \qquad\vdots \\
&input\,pattern_{t-4} := \{x_{t-4}, x_{t-3}, x_{t-2}, x_{t-2}, x_t\} && output\,pattern_{t-4} : \{x_{t+d}\}\ .
\end{aligned} \tag{6.1}$$

To predict the next week's exchange rate, d is set to 5.

The sample exchange rates contain data from November 1, 1989 to December 31, 1995 representing 1,608 weekdays which include about 10 holidays per year[3]. The transformation from daily exchange rates to daily returns reduces the original data set by one to 1,607 data. The 1,607 daily returns for our time windowing method generates a set of 1,602 pairs of input and output patterns for an input window size of five with $d = 1$ and $n = 1$, and a set of 1,597 valid pairs for an input window size of ten, respectively.

6.2 Definition of Input and Output Patterns

6.2.1 Last Trading Days

This time windowing approach has been validated on predictions for certain financial time series [17]. Our experimental design attempts to utilize additional statistical indices describing properties from the immediate past, i.e., from the previous week, and from the previous month.

[3]In our experiments, holidays were not removed from the data set. The previous closing price for the exchange rate was chosen as an approximation for the exchange rate on holidays. This seems to be a valid assumption because the previous trading day's closing price is the latest information from the time series. Therefore, it appears that there is a zero daily return on holidays.

6.2.2 Statistical Features of Time Series

Financial time series exhibit dynamic behavior over time. A statistical description of the previous observations of a given time series can contain useful information that is not easily detected by using the daily returns from the last five or ten trading days. To predict the daily or weekly return, x_{t+d}, the **moving average** $f1_{t,p}$, the **standard deviation** $f2_{t,p}$, the **skewness** $f3_{t,p}$, and the **kurtosis** $f4_{t,p}$ from the previous five ($p = 5$) and twenty ($p = 20$) daily returns x_t, were calculated for every predicted value x_{t+d}. Equations 6.2-6.5 were derived from Equations 5.1-5.4. Further, the **exponential moving average** $f5_{t,\alpha}$ with smoothing constants of $\alpha = 0.5$ and $\alpha = 0.8$ were calculated according to Equation 6.6.

$$f1_{t,p} = \frac{1}{p} \sum_{i=t-p+1}^{t} x_i \tag{6.2}$$

$$f2_{t,p} = \frac{1}{p-1} \sum_{i=t-p+1}^{t} (x_i - \bar{x}_i)^2 \tag{6.3}$$

$$f3_{t,p} = \frac{1}{p-1} \sum_{i=t-p+1}^{t} (\frac{(x_i - \bar{x}_i)^3}{s^3}) \tag{6.4}$$

$$f4_{t,p} = \frac{1}{p-1} \sum_{i=t-p+1}^{t} (\frac{(x_i - \bar{x}_i)^4}{s^4}) \tag{6.5}$$

$$f5_{t,\alpha} = \alpha * x_t + (1 - \alpha) f5_{t-1,\alpha} \tag{6.6}$$

These statistical features more accurately describe the behavior and distribution of the financial time series values.

6.2.3 Output

Although the final goal of the neural network system is an accurate prediction of the next day's or next week's currency exchange rate, the direct output from our networks is a predicted value for the next day's or next week's return. The transformation from a return to a rate is just a simple calculation of the current exchange rate and the predicted return, such as

$$x_t = (1 + r_t) * x_{t-1} \ . \tag{6.7}$$

6.3 Experiment Parameters and Factors

A neural network's performance is affected by a series of parameters such as the definition of the output vector, the choice of data, the initial weight state, and the stopping criteria during the training phase. Section 6.3.1 summarizes the important parameter that will be held fixed through the experiments. Section 6.3.2 will describe parameters which are varied in the experiments.

6.3.1 Fixed Parameters

The outcome of our experiments is heavily influenced by the values of the experiment's fixed parameters. The back-propagation algorithm was chosen for the underlying training of the neural networks, and the network's weights were initialized randomly. Although the initial weights for all individual neural networks are not the same, the random weight initialization as a method is considered a fixed parameter during all experiments. Besides the choice of learning algorithm and weight initialization, the size of the data sets, the number of hidden units, and the number of training cycles are all fixed parameters.

Size of Data Sets

Because statistical information from the previous 20 trading days was used for the definition of the input vector, the original data set of 1,608 observations was reduced to 1,588 input and output pairs to predict the next day's return, and the data set was further reduced to 1,584 pairs to predict the next week's return. Table 6.1 summarizes the partitioning for our time series experiments. The first 1,328 input-output pairs are used for the training and cross-validation set, while the second part was used for testing. The test set contains all input-output pairs from the entire year 1995. The training set consisted of 70% of the randomly chosen pairs from November 1989 until December 1994, while the cross-validation set contained the remaining 30% of the data.

	Predictions of	
Data Set	next day's return	next week's return
Training Set	886	886
Cross-validation Set	442	442
Test Set	260	256

Table 6.1: Partitioning of data for the next day's and the next week's return predictions.

Number of Hidden Units

While the number of the input and output nodes is directly determined by the size of the input and output vectors, the number of hidden layers and hidden nodes per layer is harder to determine. The size of a neural network is determined by the number of connections, also referred to as the degree of free parameters. We will use only one layer of hidden units which is fully connected to both the input and output layer. This is not a serious limitation, because any degree of free parameters can be achieved by using just one hidden layer and an appropriate number of hidden units. For all our experiments, we will train ten networks which differ in the number of hidden units, starting from one hidden unit and increasing by two units for the next largest neural network. Therefore, the smallest networks will have between 5 and 40 input units, 1 hidden unit, and 1 output unit. The largest networks will have between 5 and 40 input units, 19 hidden units, and 1 output unit. A generally accepted rule of thump says the number of weights (degree of free parameters) should be less than one tenth of the number of training pattern. Using a set of networks with between 1 and 19 hidden units guarantees that for all our input vector sizes (between 5 and 40) this rule of thumb is satisfied, while at least some other neural networks will be oversized. Oversized neural networks tend to a condition called overfitting, which was discussed in Section 4.1.3. To deal with this problem, the training must be stopped before the neural networks learn the noise of the time series.

Number of Training Cycles

The cross-validation procedure, discussed in Section 4.1.3, provides a technique to determine when to stop training. Figure 6.2 shows the mean square errors (MSE) for the training, cross-validation, and test sets plotted against the number of training cycles for two typical experiments.

In both cases, a neural network with 15 input units, 9 hidden units, and 1 output units was trained to predict the next day's return for the German Mark and the Japanese Yen. These are typical oversized neural networks, because there are 144 network weights, whereas the number of training pattern is 886. Applying the above, the rule of thumb would require more than 1,440 training pattern. Both plots demonstrate that the training should be stopped after a relative short training phase, i.e., 257 training cycle for the first experiment, and 35 training cycles for the second experiment. Based on these two preliminary findings, the maximum number of training cycles was defined to be 1,000 cycles, because no further improvement occurred in the cross-validation set even after 100,000 training cycles.

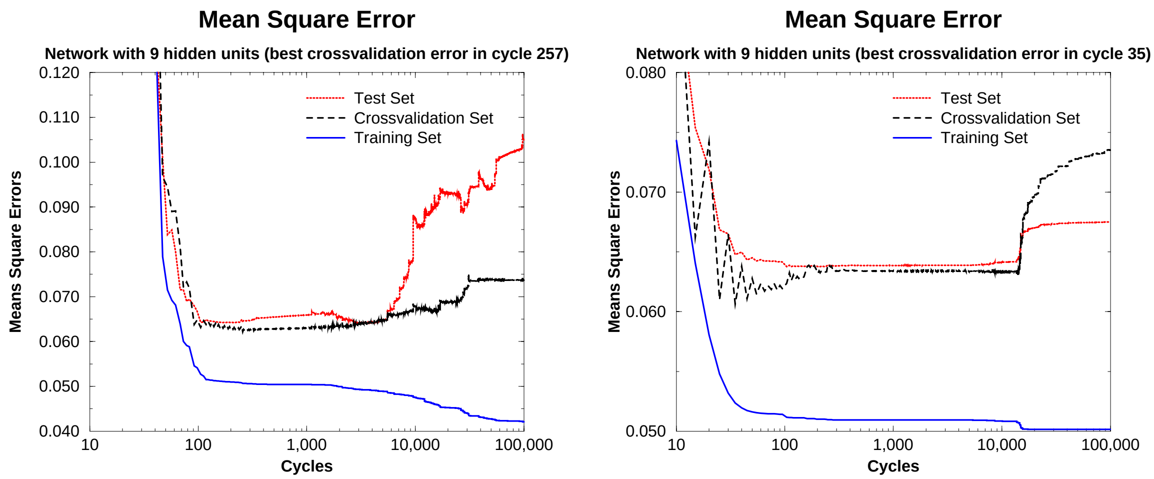


Figure 6.2: Plot of mean square errors for a slightly oversized neural network (15 input units, 9 hidden units, and 1 output unit) against the number of training cycles for the next day's return predictions of the German Mark (left figure), and the Japanese Yen (right figure).

6.3.2 Factors

We have chosen three parameters as factors for our full factorial design. A full factorial design utilizes every possible combination at all levels of all factors. These factors are the two major currencies, the six different levels for the input vector, and the two different output vectors. Therefore, our performance study will require 24 experiments. Table 6.2 summarizes the different levels of each factor.

6.3.3 Performance Metrics

The **Mean Square Error (MSE)** is used directly by the back-propagation algorithm to determine the neural network weight updates, and to determine the stopping point for training. The MSE measures the "misfit" of the regression estimates (calculated output of the neural network) and its desired output. It is a simple performance metric.

The MSE is an artificial measure of performance, because the mean square error is just a quantitative expression of the current system error and it does not measure the

Factor	Levels	Comment
Currency	German Mark	lowest Hurst coefficient
	Japanese Yen	highest Hurst coefficient
Input Vector	Last 5 trading days	last week information
	Last 10 trading days	last two weeks information
	10 statistical features	statistical indices of last 4 weeks
	Last 5 trading days and 10 statistical features	last week information, statistical indices of last 4 weeks
	Last 5 trading days from 4 major currencies	multivariate analysis, last week information
	Last 10 trading days from 4 major currencies	multivariate analysis, last two weeks information
Output Vector	Next day's return	short term predictions
	Next week's return	utilizing day-of-the-week effect

Table 6.2: Definition of the factors and the levels for each factor for the full factorial experimental design.

profitability of the system. The metric of **predicted direction (PD)** is the number of right predictions of the direction for the next daily or weekly return to the total number of predictions. The PD index gives a first estimate of the profitability. In general, if a system can predict the direction of the next trading day's exchange rate with 60% accuracy, this system may be able to produce a profit. However, this is not guaranteed. If a system can predict the direction of the next day's exchange rate with 60% accuracy, it still can produce a loss, e.g. when the system is correct for trades with small returns and is incorrect for trades with larger returns. To capture all these effects, a more realistic market metric is necessary.

The ultimate performance metric is a **gain-loss analysis** (real market simulation) with and without including trading cost. This analysis simulates the real cash flow of investments in the financial market. In the simplest algorithm, a foreign currency is bought or held whenever a positive return is predicted, and is sold, when the predicted return is negative. Simulation with and without including trading costs will be used for the performance evaluation of our neural network systems.

6.4 Neural Network Simulator - FastBep

The experiments were performed on four Sun SPARCstation 20 machines with 64 MB main memory and 134 MB virtual memory in the High Performance Computer Lab at UTSA.. Each station has 4 CPU's and runs SunOS Release 5.4. All four machines were utilized to perform these experiments which run between 15 minutes for the case of 1,000 training cycles and 5 input units (shortest experiment) and 129 hours for the case of 100,000 training cycles and 40 inputs (largest experiment). The experiments were performed by using FastBep[4] a neural network simulator, which especially supports the master network design, discussed in Section 4.1.4. The input to the FastBep simulator is a configuration file that contains specifications about the neural network architecture (i.e. number of input, output and hidden units; number of subnetworks), parameters for the training algorithm (i.e. number of training cycles; acceptable error), and all input-output pattern (training, cross-validation and test set). During its run, FastBep produces several output files containing information about the current weight set for the training and cross-validation data, and the training and cross-validation errors. A summary of all options and parameters can be found in the FastBep manual [19].

[4]FastBep is an experimental neural network simulator, developed by Dr. Bruce Rosen. Thanks for making this software package available for these experiments.

Chapter 7

Results of Simulation

Multi-neural network systems with a different number of subnetworks were used to predict the next day's or next week's return for the German Mark and the Japanese Yen. A full factorial design of 24 experiments (two currency exchange rates, six different input vectors, and two different output vectors) was performed. Each experiment consisted of training 10 single neural networks with different numbers of hidden units, and also combining these successive single networks into 9 separate master networks. Therefore, each experiment generated 19 neural network systems that produced 19 predictions. Each network was trained for 1,000 cycles, and the weight set with the best cross-validation error found during the 1,000 cycles was used for testing. Although the mean square error (MSE) was the performance metric for the training and cross-validation phase, the performance metric we used to evaluate the prediction accuracy for financial time series is the predicted directions (PD) index. The results and the prediction accuracies using the PD index are presented in Section 7.1 for the German Mark and in Section 7.2 for the Japanese Yen. A commonly used technique for the prediction of financial time series is the linear regression. Therefore, a performance evaluation using a linear regression model, and using a gain-loss simulation for 1995 is supplied in Chapter 8.

7.1 Predictions for the German Mark Exchange Rate

7.1.1 Next Day Predictions

Predictions for the next day's return of the German Mark based on the last 5 trading days and the 10 statistical features (mean, standard deviation, skewness, kurtosis, and exponential moving average) gave better results than predictions based on other input vectors, e.g. the previous 10 trading days or even the last 10 trading days from all four major currencies. The average and best neural network performance for the next day's return of the German Mark are summarized in Table 7.1. An "X" indicates whether the best neural network system is a single or a master network. In general, better performance was achieved by master networks, except for the network with its input vector consisting of the last 10 trading days. The best performances were obtained when using the returns of the last 5 trading days and the 10 statistical features. The best results are shown in bold face.

	Average	Best Network		
Input Vector	PD	PD	Master	Single
Last 5 trading days	53.09%	58.0%	X	
Last 10 trading days	53.79%	59.2%		X
10 statistical features	53.26%	57.2%	X	X
Last 5 trading days and 10 statistical features	**54.02%**	**60.4%**	X	
Last 5 trading days from 4 major currencies	51.70%	56.8%	X	
Last 10 trading days from 4 major currencies	52.97%	59.6%	X	

Table 7.1: Predicted direction performance for the next day's return predictions of German Mark for different input vectors.

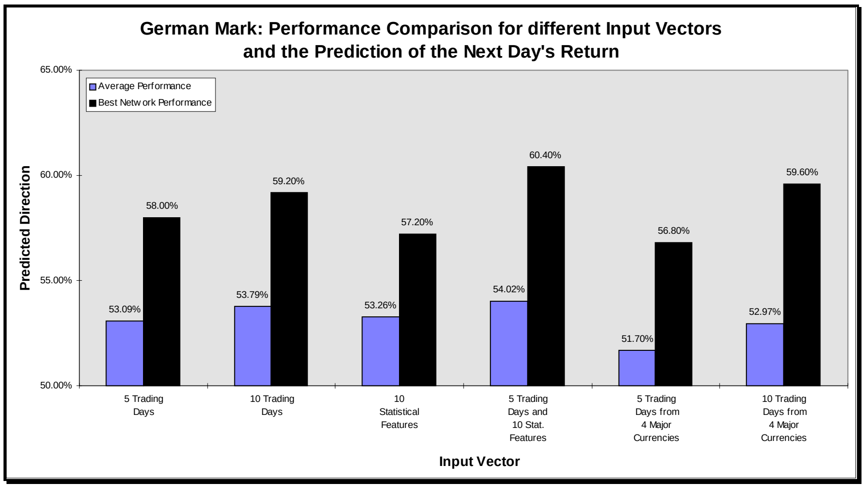


Figure 7.1: Prediction accuracy for the German Mark: Comparison of predicting the direction of the next day's return using different input vectors.

Figure 7.1 is a plot of the previous table. This plot shows that the input vector consisting of the last 5 trading days and the 10 statistical features produced the best network, which is a master network, and also gave the best average performance[1] of 54.02%.

A plot of the results for the experiments with the best input vector (daily returns from last 5 trading days and the 10 statistical features) is shown in Figure 7.2. The master network with five subnetworks provides the highest accuracy (60.40%) when predicting the right direction of the next day's return. The predicted direction (PD) metric is not the best metric for evaluating the profitability of a financial time series, because profits are more dependent on the magnitude of changes. A forecasting system with a lower PD accuracy may be more profitable than a system with a higher PD accuracy if the former is right in cases of large changes, and wrong in cases of smaller changes.

[1]The average performance is calculated as the arithmetic mean of the predicted direction performance of all 10 single neural networks and all 9 master networks.

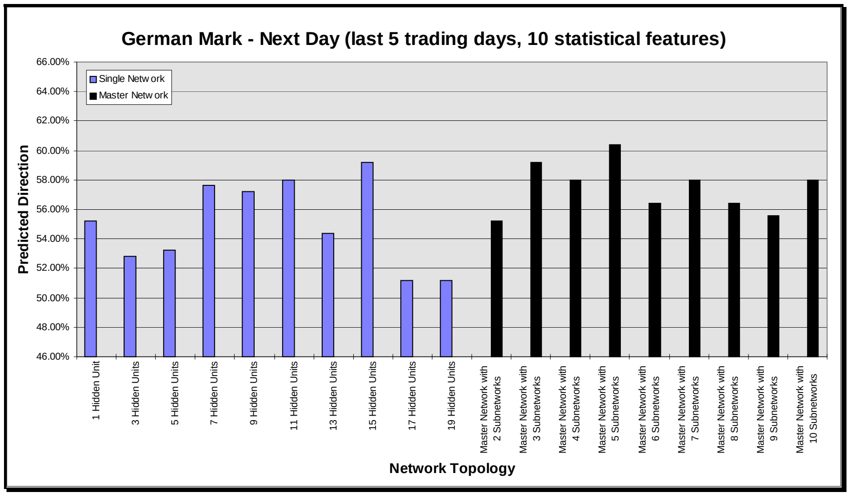


Figure 7.2: Prediction accuracy for the German Mark: Predicting the direction of the next week's return using the returns of the last 5 trading days and the 10 statistical features about the previous returns.

7.1.2 Next Week Predictions

Neural networks trained to predict the next week's return showed better performance than those predicting the next day's return. The average performance of the experiment using only the 10 statistical features and the experiment using the last 5 trading days and the 10 statistical features achieved the best average prediction accuracy (59.61%) as shown in Figure 7.3. The best performing neural network system (64.23%) was a master network consisting of 8 subnetworks trained with the 10 statistical features.

Figure 7.4 shows the PD accuracy of all networks that were trained with just the 10 statistical features. A more detailed analysis shows that the mean of the master networks (60.21%) is higher than the mean of the single neural networks (59.07%). Further, the master networks have a much smaller variance. The worst performance of a single neural network is 52.85% accuracy, whereas the worst performance for a master

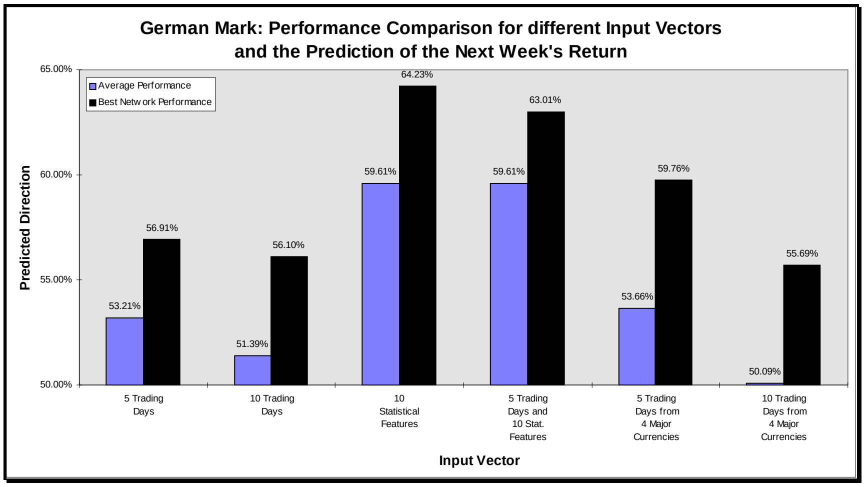


Figure 7.3: Prediction accuracy for the German Mark: Comparison of predicting the direction of the next week's return using different input vectors.

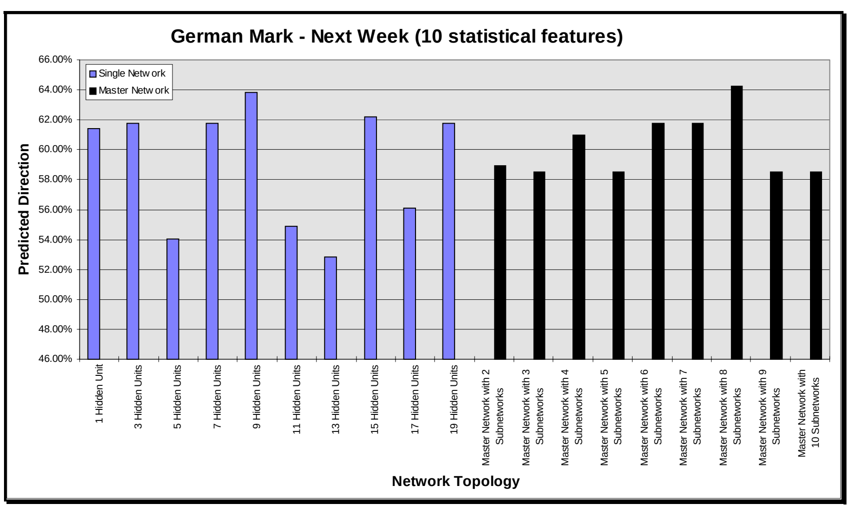


Figure 7.4: Prediction accuracy for the German Mark: Predicting the direction of the next week's return using the 10 statistical features about the previous returns.

network is 58.54%. This is an example of the ability of the master network to improve the prediction accuracy over individual neural networks.

7.2 Predictions for the Japanese Yen Exchange Rate

7.2.1 Next Day Predictions

The results for the Japanese Yen next day predictions differ from the results achieved for the German Mark next day predictions. The best average network performance (57.96%) was achieved using an input vector containing the daily returns of the last 10 trading days of all four major currencies. Two neural networks had the best PD accuracy, a single neural network that was trained with the last 10 trading days, and a master neural network consisting of two single neural networks that were trained with the last 10 daily returns of all four major currencies. Both networks had an accuracy of 60.82% in predicting the right direction of the next day's return. A comparison of the performance for the different input vectors is plotted in Figure 7.5. The plot of the performance of all neural network systems that were trained with the last 10 daily returns of all four major currencies is shown in Figure 7.6. Once again, the average performance of the master networks is much higher (59.18%) compared with the average performance of the single neural networks (56.86%). The worst performance is 57.55% for a master network and 52.24% for a single neural network.

7.2.2 Next Week Predictions

The results of the performance analysis for the different input vectors of the Japanese Yen is very similar to the results achieved for the next week's return predictions

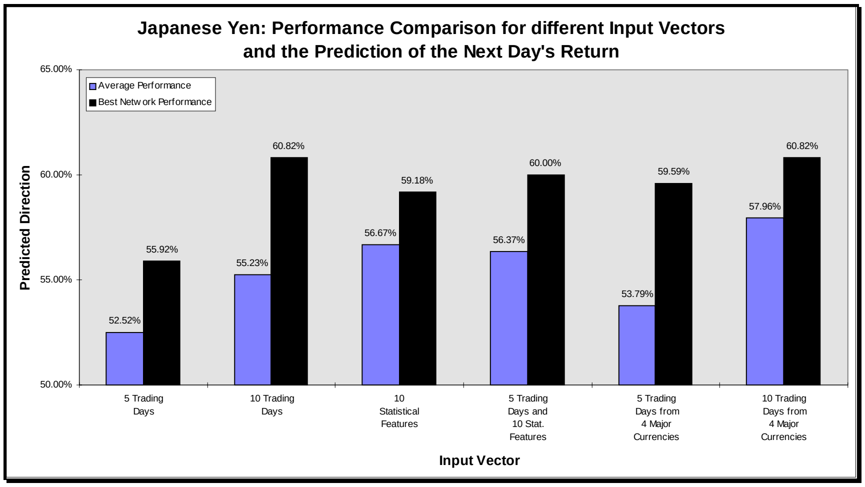


Figure 7.5: Prediction accuracy for the Japanese Yen: Comparison of predicting the direction of the next day's return using different input vectors.

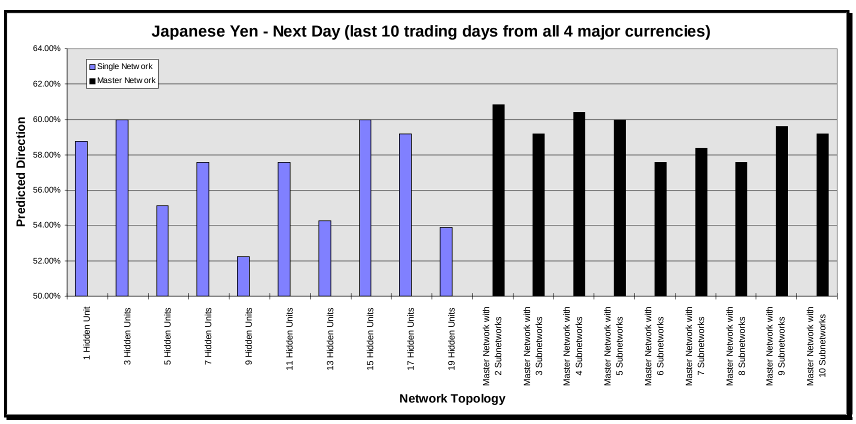


Figure 7.6: Prediction accuracy for the Japanese Yen: Predicting the direction of the next day's return using the returns of the last 10 trading days from 4 major currencies.

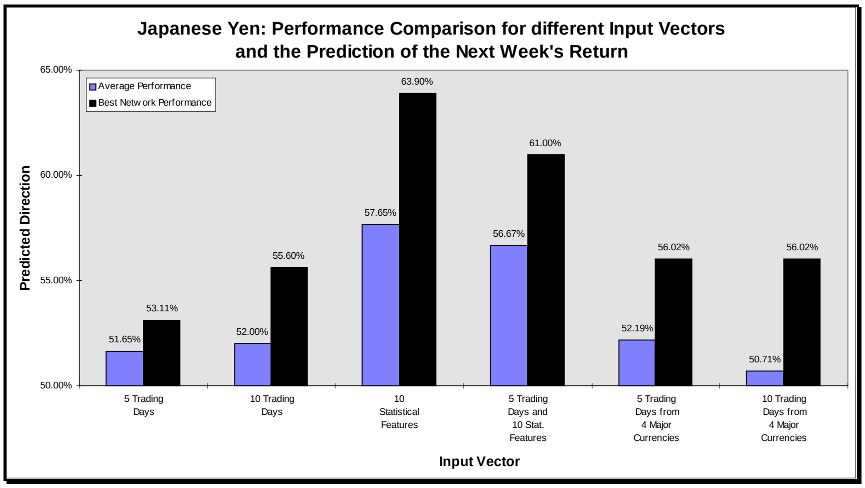


Figure 7.7: Prediction accuracy for the Japanese Yen: Comparison of predicting the direction of the next week's return using different input vectors.

of the German Mark. Predictions based on the 10 statistical features gave a better average and best performance than prediction based on other input vectors as shown in Figure 7.7.

The best neural network system was a single neural network consisting of 17 hidden units, which achieved an exceptional accuracy of 63.90%. Figure 7.8 is a plot of the performance of all single and master networks. Although the best neural network system is a single neural network, the average performance for the master networks (58.41%) is still higher than for the single neural networks (56.97%). A comparison of the performance of the best single neural network (17 hidden units) with similar single neural networks (15 and 19 hidden units) shows that this exceptional performance appears to be an outlier.

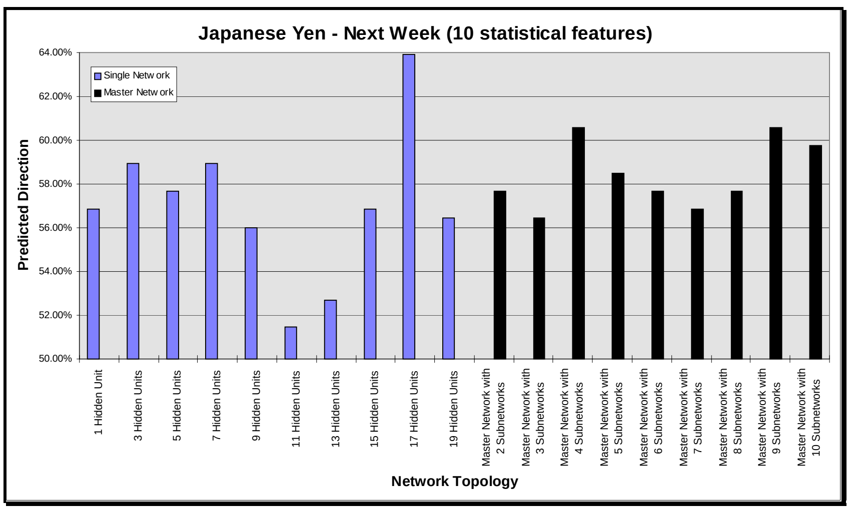


Figure 7.8: Prediction accuracy for the Japanese Yen: Predicting the direction of the next week's return using the 10 statistical features about the previous returns.

Chapter 8

Performance Evaluation

The previous chapter provided us with an argument that our neural networks may produce good predictions, but there is still the question how this good performance is comparable to traditional modeling techniques, and what is their profitability, if any, on the real market. This chapter will discuss these issues and tries to give a real performance evaluation. Although the experiments were performed for predicting the next day's and the next week's returns, the following analysis is focused only on the next day's return predictions. The buy-and-hold strategy, discussed in Section 8.1, simulates a naive investor, who buys a currency and holds it over a period in hope of a positive return at the end of that period. In Section 8.2, a linear regression model for the daily returns of the German Mark and the Japanese Yen is estimated. Finally, in Section 8.3, the predictions of our neural network systems are used for a real market simulation with and without consideration of trading costs.

8.1 Buy-and-Hold Strategy

The buy-and-hold strategy is a very simple trading strategy where a currency is bought at the beginning of a trading period and is held until the end of this period. Trading costs are minimized to just one round-trip and the profitability of the trading algorithm only depends on the difference between the exchange rates at the beginning and the end of the trading period. This strategy will be used as a benchmark for comparisons with a linear regression trading model and our neural network trading systems. We would like to see at least a higher profitability for our neural network trading systems than this simple buy-and-hold strategy. In Figures 8.1-8.4, the current values of the buy-and-hold strategy for 1995 are plotted as an index that was set to 100 on January 1, 1995.

8.2 Linear Regression

A time series linear regression model, also referred to as an autoregressive (AR) model was applied for the prediction of the daily returns of the Japanese Yen and the German Mark. In our experiments, this model used the same data that was used to train the neural networks, i.e. the training data was from November 1989 to December 1994, and the prediction data was for 1995. The autoregressive model used the pre-processed returns of the previous 10 trading days. Using daily returns instead of the original exchange rates transformed the original nonstationary time series into a stationary one. Therefore, the initial linear regression model, AR(10), can be considered as an ARIMA(10,1,0) model (10 autoregressive terms on the first differences of the original data with no (0) moving average (MA) terms):

$$y_t = \alpha_0 + \alpha_1\, y_{t-1} + \alpha_2\, y_{t-2} + \cdots + \alpha_{10}\, y_{t-10} \ , \tag{8.1}$$

where the α_is are the linear regression coefficients and the y_{t-i}s are the daily returns i days prior to the prediction time t. This modeling process is very complex, however, there exist very powerful software packages such as SAS[1] and SPSS[2] that model these methods. The data analysis and parameter estimation for our autoregressive models were performed using the SPSS software. Table 8.1 shows the linear regression coefficients which were estimated using the statistical model as defined in Equation 8.1.

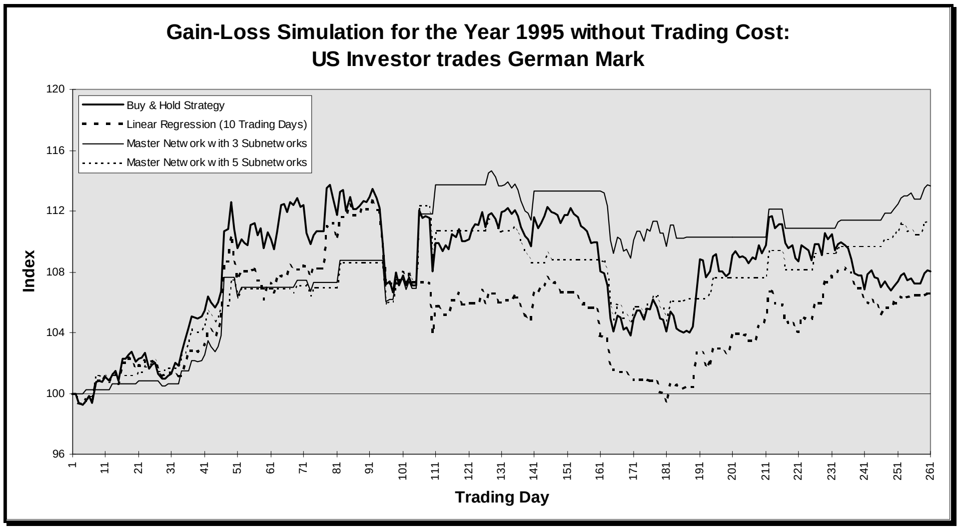
	Next day's return	
Variable	German Mark	Japanese Yen
α_0	-0.000105	-0.000248
α_1	-0.037426	0.032539
α_2	0.002022	0.018302
α_3	0.000127	-0.094425
α_4	0.036667	0.014207
α_5	0.028867	-0.026342
α_6	-0.056079	-0.044767
α_7	0.002827	0.071694
α_8	0.003241	0.000059
α_9	-0.012834	0.002057
α_{10}	0.071347	0.050127

Table 8.1: Variables of the linear regression models for predicting the next day's return of the German Mark and the Japanese Yen.

The predictions of the linear regression model for the next day's return was used with the following simple trading strategy: (1) Buy or hold a currency whenever a positive return is expected, and (2) sell a currency when a negative return is expected. An example for this trading rule is when a German investor buys or holds US Dollar

[1]SAS (Statistical Analysis System), developed by the SAS Institute Inc., is a computer software package used for data management and statistical analysis. Data are usually entered into the SAS program in the form of lines of data (scripts).

[2]SPSS is a complete tool kit of statistics, graphs and reports developed by the SPSS (Statistical Product and Service Solutions) Inc. SPSS offers powerful, state-of-the-art statistical procedures with hundreds of features that make it easy to use.



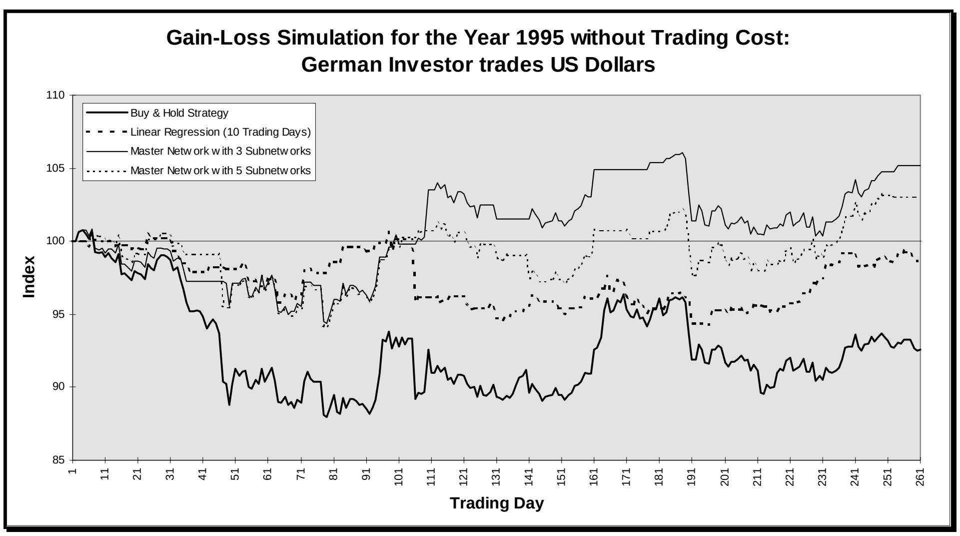
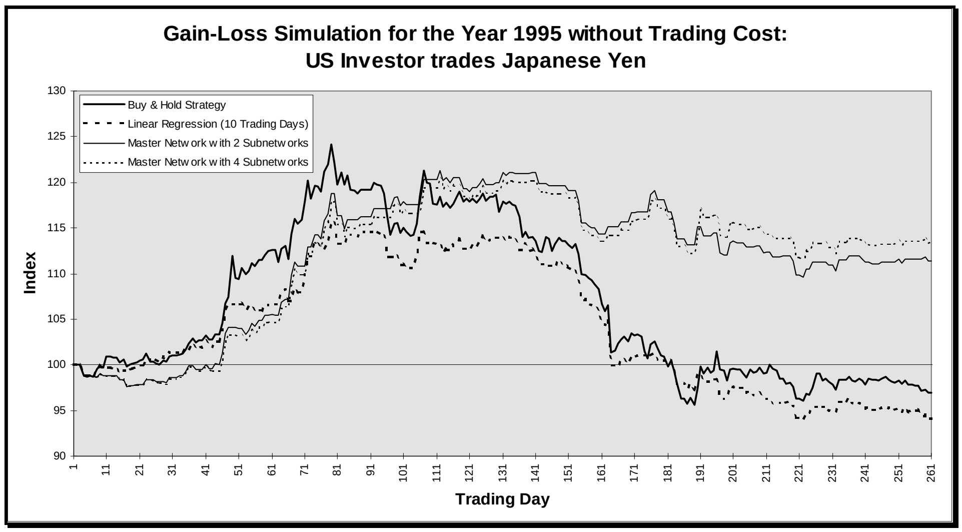


Figure 8.1: Real market simulation without trading costs: A comparison about the profitability between the simple buy-and-hold strategy, the linear regressing model, and two neural network systems for trading (a) German Mark with US Dollar, and (b) US Dollar with German Mark.



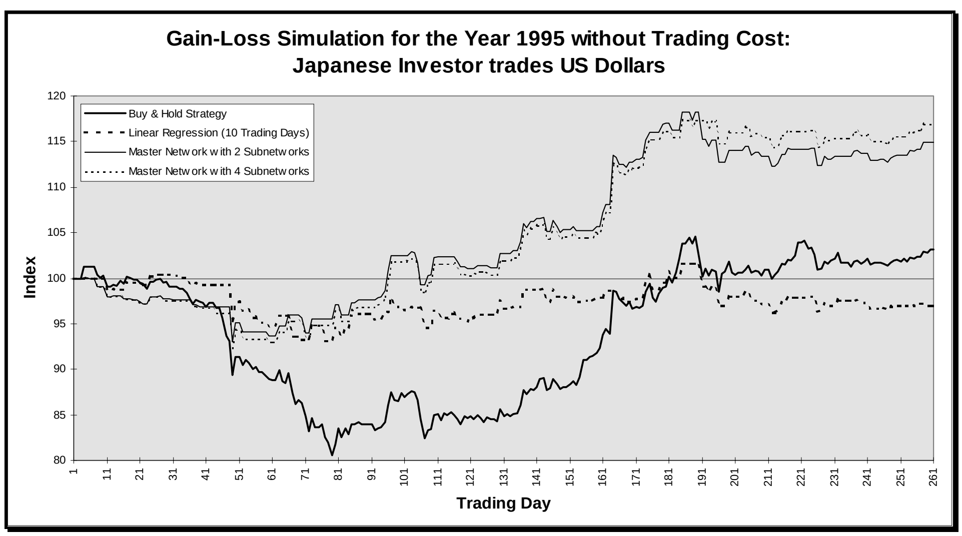


Figure 8.2: Real market simulation without trading costs: A comparison about the profitability between the simple buy-and-hold strategy, the linear regressing model, and two neural network systems for trading (a) Japanese Yen with US Dollar, and (b) US Dollar with Japanese Yen.

whenever a positive return for the German Mark exchange rate is predicted. This gives the investor the opportunity to sell the US Dollar for an expected higher price in the future. The investor sells US Dollar when a negative return is expected. The real market simulation of this simple trading strategy based on predictions of the linear regression model initiated 74 round-trip (buy and sell) transactions for the German Mark and 79 round-trip transactions for the Japanese Yen. Table 8.2 compares the results of the real market simulation for the buy-and-hold strategy with the results of the simple trading strategy based on predictions of the linear regression models.

	Buy-and-Hold Strategy	Simple Trading Strategy based on Linear Regression
US Investor trades German Mark	**108.04%**	106.56%
German Investor trades US Dollar	92.56%	**98.64%**
US Investor trades Japanese Yen	**96.99%**	94.06%
Japanese Investor trades US Dollar	**103.11%**	96.99%

Table 8.2: Comparison of profitability between the buy-and-hold strategy and the simple trading strategy based on predictions of the linear regression models.

The regression model produced a positive return of 6.56% for the US investor who traded German Mark, but the simple buy-and-hold strategy could have brought a positive return of 8.04% (graph *(a)* in Figure 8.1). In the case of the German investor trading US Dollar (graph *(b)* in Figure 8.1), the regression model reduced the loss for the buy-and-hold model from 7.44% to only 1.36% for the regression model. But this is the only case where the simple trading strategy based on predictions of the linear regression model performed better than the buy-and-hold strategy. Trading between the Japanese Yen and the US Dollar based on the regression model produced a negative return for both directions, as shown in charts *(a)* and *(b)* of Figure 8.2.

8.3 Neural Network Predictions

The neural networks with the best PD index for the test data were used for our performance evaluation. Using knowledge of the PD test run to pick the best network system for the trading simulation is a simplification. In real life, the PD indices from a tuning set[3] should be used instead of the PD indices from the test set. The best neural network system for the next day's return predictions of the German Mark was a master network consisting of five subnetworks with an PD accuracy of 60.40%. Further, the best neural network system for the next day's return prediction of the Japanese Yen was a master neural network consisting of two subnetworks with an accuracy of 60.82%. Although these PD indices indicate a good performance, they do not evaluate the return on the investment in the financial market. The best performance metric is a simulation on the real market. The performance of our selected neural network systems is evaluated by utilizing the same simple trading strategy, described in the previous section, and using the predictions of our neural networks. These simulations are performed both with and without consideration of trading costs.

8.3.1 Results of Trading without Trading Costs

A real market simulation without consideration of trading costs provides a first estimate of the profitability of our neural network trading systems. Two neural network systems with the highest PD accuracy for the next day's return prediction were used for our comparison. Table 8.3 summarizes the profitability of the real market simulation for 1995 without consideration of any trading costs for the buy-and-hold strategy, the

[3]The tuning set is a set that is disjoint from the training and test set. The neural network system with the best PD performance for the tuning set should be chosen for testing. The cross-validation set has similar properties and may be used as a tuning set.

trading based on predictions of the linear regression models, and the trading based on predictions of two neural network systems.

	Buy-and-Hold Strategy	Simple Trading Strategy based on Linear Regression	First Neural Network System	Second Neural Network System
US Investor trades German Mark	108.04%	106.56%	**113.66%**	111.30%
German Investor trades US Dollar	92.56%	98.64%	**105.21%**	103.02%
US Investor trades Japanese Yen	96.99%	94.06%	111.42%	**113.44%**
Japanese Investor trades US Dollar	103.11%	96.99%	112.55%	**114.79%**

Table 8.3: Comparison of profitability of the real market simulation for 1995 without consideration of trading costs between the buy-and-hold strategy, the trading based on predictions of the linear regression models, and trading based on predictions of two neural network systems.

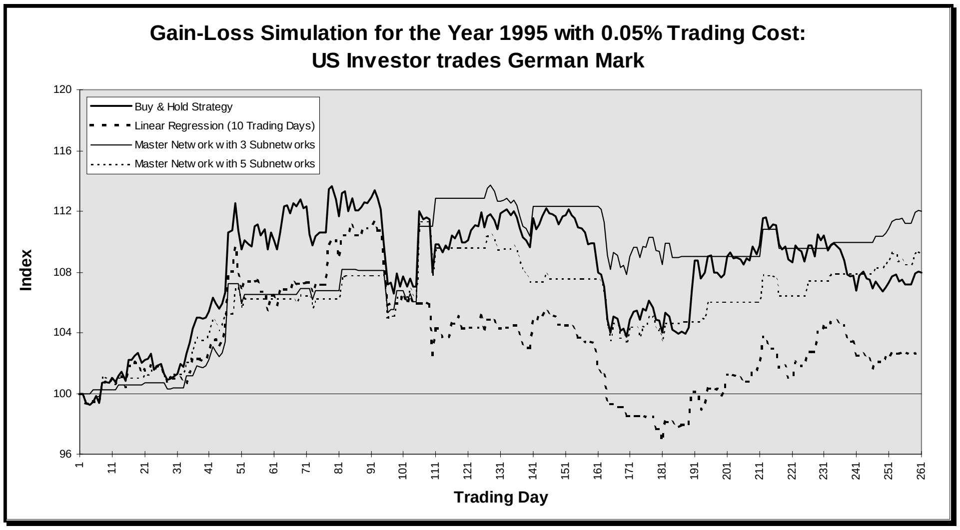
Our two selected neural network systems for the German Mark trading performed very well in the real market simulation as shown in both graphs of Figure 8.1. The best performance (113.66%) for the US investor trading German Mark was achieved by the master neural network consisting of three subnetworks. Even the second neural network performed much better (111.30%) than the buy-and-hold strategy (108.04%) and the trading based on the linear regression model (106.56%). Applying the predictions of the same neural networks to the case of the German investor trading US Dollar, both neural networks produced a consistently better return over the entire year when compared with our two benchmarks. Although both benchmarks, the buy-and-hold strategy and the linear regression, produced a negative return, both neural network systems achieved significant positive returns of 105.21% and 103.02%.

The real market simulations for 1995 for the Japanese Yen and the US Dollar trading are plotted in Figure 8.2. Both neural network systems produced a significant and almost consistent higher return than the buy-and-hold strategy or the trading based on the linear regression model. The master network with four subnetworks achieved a positive return of 114.79% for the case of the Japanese investor trading US Dollar and 113.44% for the US investor trading Japanese Yen. The returns achieved by the two benchmarks were negative, except for the buy-and-hold strategy in the case of our Japanese investor who was trading US Dollar (103.11%).

8.3.2 Results of Trading with Trading Costs

The ultimate index is a real market simulation that includes trading costs. Currency futures contracts have very low trading costs of about 0.05% of the trading value. These trading costs of 0.05% per round-trip (one buy and one sell) transaction was used for our final real market simulations for all trading strategies. Therefore, the trading costs mainly depend on the number of round-trip transactions. Thus, the buy-and-hold strategy produces the lowest trading costs of only 0.05% of the initial trading volume for exactly one round-trip transaction.

The profitability of the real market simulation for 1995 with consideration of 0.05% trading costs for the buy-and-hold strategy, the trading based on predictions of the linear regression models, and the trading based on predictions of two neural network systems is summarized in Table 8.4. Our two selected neural network systems for the German Mark predictions initiated 29 and 36 round-trip transactions, which is much less compared with 74 round-trip transactions for the linear regression trading model. Therefore, the performance difference between the two neural networks and the regression model increases further if compared with the real market simulation without consideration of trading costs, whereas the gain compared with the buy-and-hold strategy decreases. The performance of both neural networks (112.03% and 109.32%) is still higher than the performance of the buy-and-hold strategy (107.98%) and the trading based on the linear regression model (102.69%) for the case of the US investor trading German Mark (graph *(a)* of Figure 8.3). In the case of the German investor trading US Dollar (graph *(b)* of Figure 8.3), the neural network approach still can produce a positive



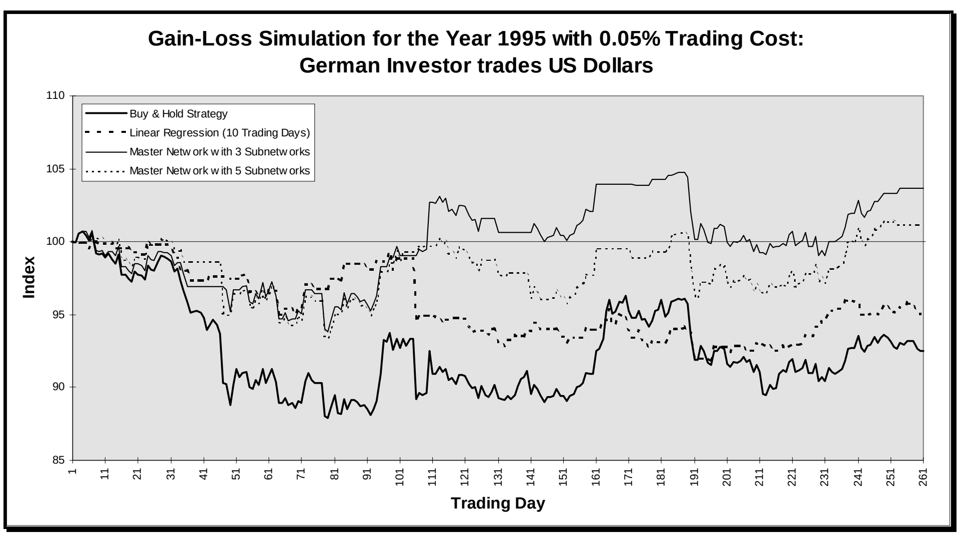
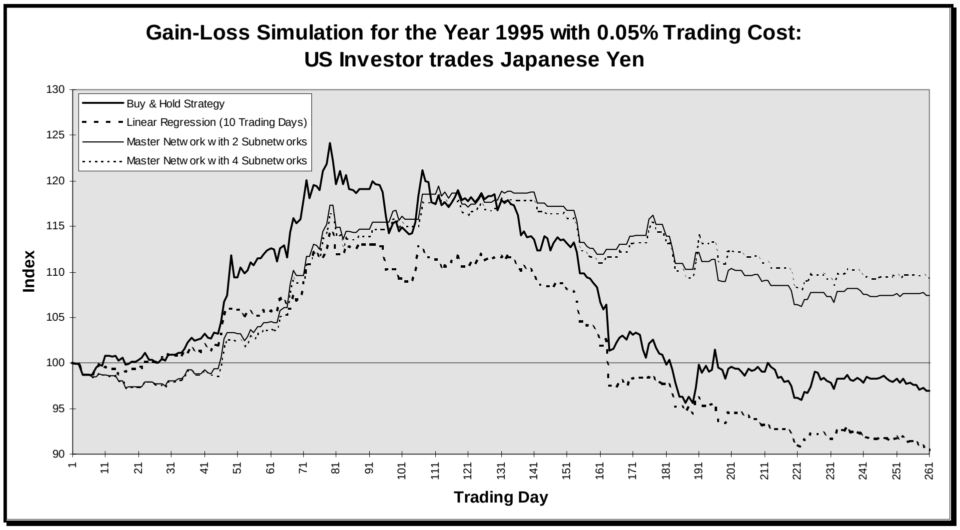


Figure 8.3: Real market simulation with including 0.05% trading costs per buy and sell (one round-trip): A comparison about the profitability between the simple buy-and-hold strategy, the linear regressing model, and two neural network systems for trading (a) German Mark with US Dollar, and (b) US Dollar with German Mark.



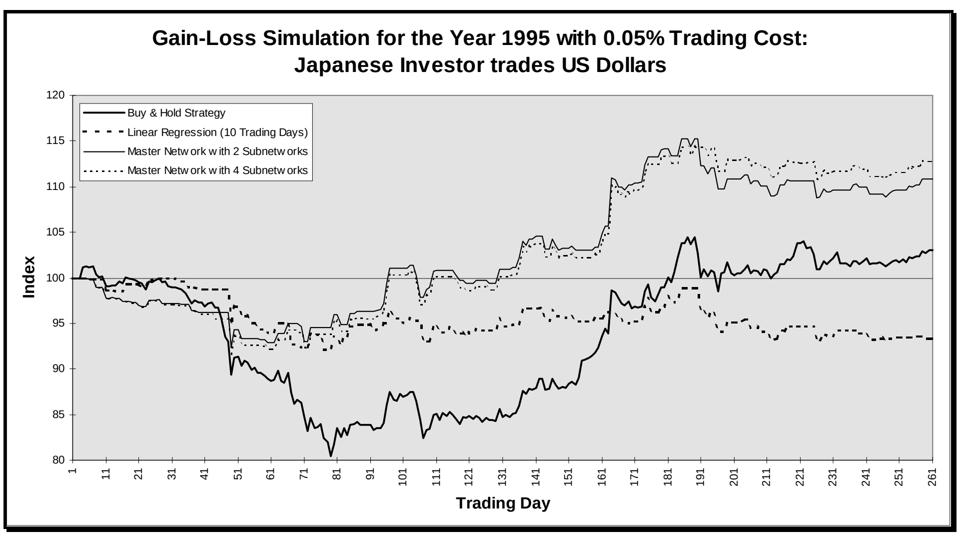


Figure 8.4: Real market simulation with including 0.05% trading costs per buy and sell (one round-trip): A comparison about the profitability between the simple buy-and-hold strategy, the linear regressing model, and two neural network systems for trading (a) Japanese Yen with US Dollar, and (b) US Dollar with Japanese Yen.

return of 103.69% and 101.18% compared with a loss for both benchmarks of 92.51% and 95.05%, respectively.

The real market simulation for the Japanese Yen is plotted in graphs *(a)* and *(b)* of Figure 8.4. Although both neural network approaches initiated 71 and 72 round-trip transactions compared with 79 for the regression model, they still performed much better than both of our benchmarks. The second neural network, a master network consisting of four subnetworks, performed best for both the US investor trading Japanese Yen (109.32%) and the Japanese investor trading US Dollar (112.88%). Our regression models performed poorly with only 90.47% in the first case and 93.32% in the second case. The simple buy-and-hold strategy produced a small gain of 3.05% for the Japanese investor and a loss of 3.06% for the US investor. The results of our first neural network system, a master network with two subnetworks, were very closed to them of our second neural network system. Using the prediction based on the first neural network produced a gain of 7.43% for the US investor and a gain of 10.82% for the Japanese investor.

In general, the neural network approach using enhanced data preprocessing produced better results for both the German Mark and the Japanese Yen currency exchange market. The simple trading mechanism based on linear regression models did not show a better overall prediction performance when compared with the buy-and-hold strategy. The regression model initiated many buy and sell orders which further increased the trading costs and lowered the return. Trading of both currency exchange rates based on predictions of two selected neural networks showed improved performance in all cases if compared with the buy-and-hold strategy and the trading based on the linear regression model. The neural network approaches generated less round-trip transactions, especially for the trading with the German Mark. The predefined goal of achieving a higher profitability for our neural network trading system was fulfilled. Our neural network systems produced the highest returns for all simulations and gave even a positive return if the buy-and-hold strategy produced a loss.

	Buy-and-Hold Strategy	Simple Trading Strategy based on Linear Regression	First Neural Network System	Second Neural Network System
US Investor trades German Mark	107.98%	102.69%	**112.03%**	109.32%
German Investor trades US Dollar	92.51%	95.05%	**103.69%**	101.18%
US Investor trades Japanese Yen	96.94%	90.47%	107.43%	**109.42%**
Japanese Investor trades US Dollar	103.05%	93.32%	110.82%	**112.88%**

Table 8.4: Comparison of profitability of the real market simulation for 1995 with consideration of trading costs of 0.05% for one round-trip transaction between the buy-and-hold strategy, the trading based on predictions of the linear regression models, and trading based on prediction of two neural network systems.

Chapter 9

Conclusions

This chapter presents our conclusions arrived from the previous chapters. Forecasting the currency exchange rate for the German Mark and the Japanese Yen was chosen as a nontrivial application suitable for our studies. Section 9.1 summarizes the results obtained by our experiments from three different viewpoints. A performance comparison and discussion is made between:

1. the master network design vs. its individual neural networks,

2. the utilization of enhanced data preprocessing vs. using data directly, and

3. the neural network approach vs. the linear regression model.

Some suggestions for additional aspects of improving the results and about future research directions are made in Section 9.2.

9.1 Summary of Results

We have shown that a master neural network design lowers the prediction error variance by averaging the forecasts of an ensemble of individual neural networks. It was further shown that the average performance of these master networks was better than the average performance of the individual neural networks. In addition, master neural networks were shown to have a better prediction accuracy than the best individual neural network, except for the next week's return prediction where one individual neural network had an exceptionally high performance. This individual neural network appeared to be an outlier.

Neural network systems utilizing our statistical features from the immediate past achieved a better prediction accuracy for all but the next day's return prediction of the Japanese Yen. The best results for the next week's predictions of the German Mark and the Japanese Yen were achieved using the input vector with the 10 statistical features only. We conclude that besides defining the best indicators the presentation of this data to the neural network is a very important issue to improve the generalization of the neural networks.

Trading of the German Mark and the Japanese Yen based on the prediction of our neural network systems outperformed the trading based on the linear regression model and the buy-and-hold strategy. Our neural network trading systems produced a positive return and performed better than any of our benchmarks even under consideration of trading costs of 0.05%, as suggested by Swingler [24]. We consider our neural network system as a statistical tool that can be relied on to predict currency exchange rates.

9.2 Further Research

The main focus of our study was rather on univariate than multivariate time series predictions. Of course, using the daily returns of one currency without any additional indices is a hard restriction, but it is interesting by itself. Multivariate time series can look at the interdependence between several time series. Two initial experiments utilizing the daily return of the last 5 or 10 trading days of all four major currencies were performed with good results for the next day's return prediction of the Japanese Yen. The cross-correlation between different currency exchange rates and their country's bank interest rates, stock indices, and unemployment rates may further significantly improve the prediction accuracy. Further research needs to be done to identify relevant indicators. However, using many indicators will increase the risk of overfitting and will also increase the training time.

In Section 2.2, we have already mentioned that each major currency exchange rate has actually four different prices. We used the current spot price for our predictions. The 30-day, the 90-day, and the 180-day forward prices reflect some expectations about the future development of an exchange rate. Therefore, these values might have potential use. Utilizing this information was beyond the scope of this thesis.

The task for our neural networks was limited to the prediction of the return of one major currency a fixed time period into the future. Possible extensions for the neural network systems are the prediction of the number of trading days to the next turning point, and the expected exchange rate or return at this turning point. This additional information would be very useful for a more profitable trading on the financial market.

Our experiments have shown an improved performance using the master network design and enhanced data preprocessing. These techniques were used with a very simplified trading strategy. We expect a higher profitability by developing more sophisticated trading mechanisms which use some threshold of the neural network output rather than the neural network's output directly for its buy and sell decisions.

An interesting area of further research is the prediction of minor currency exchange rates. Minor currency exchange rates are characterized by a much higher fluctuation of the exchange rates. This higher fluctuation corresponds to a much higher risk for investors. Therefore, there exist a high demand on predictability. Achieving a predictability of 60%, for example, may result in a much higher profitability than it can be achieved for major currency exchange rates. These techniques may be also applied to other financial instruments such as the currency futures or option market, or even to time series predictions in general. We hope these directions will also provide fruitful incites to the field of time series analysis.

Bibliography

[1] James L. Blue, Patrick J. Grother. Training Feed-forward Neural Networks using Conjugate Gradients. In *SPIE*, Vol. 1661, pages 179-190, 1992.

[2] G. E. P. Box, G. M. Jenkins. *Time Series Analysis: Forecasting and Control.* Holden-Day, San Francisco, CA, 1976.

[3] Christopher Chatfield. *The Analysis of Time Series: An Introduction.* Chapman & Hall, London, 1989.

[4] J. L. Elman. Finding structure in time. In *Cognitive Science*, No.14, pages 179-211, 1990.

[5] John M. Gottman. *Time-series analysis: A comprehensive introduction for social scientists.* Cambridge University Press, Cambridge, MA, 1981.

[6] R. Hecht-Nielsen *Neurocomputing.* Addison-Wesley, Reading, MA, 1989.

[7] Takashi Kimoto, Kazuo Asakawa, Morio Yoda, Masakazu Takeoka. Stock Market Prediction System with Modular Neural Networks. In *Proceedings of the International Joint Conference on Neural Networks*, San Diego, CA, 1990.

[8] J. E. Moody. The effective number of parameters: An analysis of generalization and regularization in nonlinear learning systems. In *Advances in Neural Information Processing Systems*, Morgan Kaufmann, San Mateo, CA, 1992.

[9] Douglas C. Montgomery, Lynwood A. Johnson. *Forecasting and Time Series Analysis.* McGraw-Hill, New York, NY, 1976.

[10] Alan Pankratz. *Forecasting With Univariate Box-Jenkins Models: Concepts and Cases.* John Wiley & Sons, New York, NY, 1983.

[11] B. Parker. *Learning Logic.* Technical Report TR-47, Center for Computational Research in Economics and Management Science, Massachusetts Institute of Technology, Cambridge, MA, 1985.

[12] Michael P. Perrone, Leon N. Cooper. When Networks Disagree: Ensemble Methods for Hybrid Neural Networks. In *Neural Networks for Speech and Image processing*, Chapman & Hall, 1992.

[13] Michael P. Perrone. *Improving Regression Estimation: Averaging Methods for Variance Reduction with Extensions to General Convex Measure Optimization.* PhD thesis, Institute for Brain and Neural Systems, Brown University, May 1993.

[14] Edgar E. Peters. *Chaos and Order in the Capital Markets: A New View of Cycles, Prices, and Market Volatility.* John Wiley & Sons, New York, NY, 1991.

[15] T. Poggio, F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. In *Science*, Vol. 247, pages 978-982, 1990.

[16] A. N. Refenes. Constructive learning and its application to currency exchange rate forcasting. In *Neural Network Applications in Investment and Finance Services*, Probus Publishing, 1991.

[17] A. N. Refenes, M. Azema-Barac, S. A. Karoussos. Currency Exchange Rate Forecasting by Error Backpropagation. In *Preceedings of the 25th Hawaii International Conference on System Siences*, Kauai, Hawaii, January 1992.

[18] Enders A. Robinson, Manuel T. Silva. *Digital Foundations of Time Series Analysis: The Box-Jenkins Approach.* Holden-Day, San Francisco, CA, 1979.

[19] Bruce E. Rosen. Manual of Neural Network Simulator: FastBep Version Beta 2.0. Department of Computer Science, University of Texas at San Antonio, 1996.

[20] David E. Rumelhart, G. E. Hinton, R. J. Williams. Learning Representations by Back-Propagating Errors. In *Nature*, No. 323, pages 533-536, 1986.

[21] Roy Schwärzel. *Prediction of Currency Exchange Rates by Using a Multi-Neural Neural Network System* Technical Report ISS-15 (Information Systems Series), Department of Business Administration, Humboldt Universität Berlin, June 1995.

[22] Michael R. Sesit. Japanese Influence Crows In Global Currency Market. In *The Wall Street Journal*, September 14, pages C1, C11, 1989.

[23] Alan C. Shapiro. *Multinational Financial Management.* Allyn and Bacon, Needham Heights, MA, 1992.

[24] Kevin Swingler. Financial Prediction, Some Pointers, Pitfalls, and Common Errors. Technical report FK9 4LA Center for Cognitive and Computational Neuroscience, Stirling University Stirling, July 1994.

[25] Stephen Taylor. *Modelling Financial Time Series.* John Wiley & Sons, New York, NY, 1986.

[26] Collin J. Watson, Patrick Billingsley, D. James Croft, Davis V. Huntsberger. *Statistics for Management and Economics.* Allyn and Bacon, Needham, MA, 1993.

[27] Andreas S. Weigend, Neil A. Gershenfeld. Results of the Time Series Prediction Competition at the Santa Fe Institute. In *Proceedings of the 1993 IEEE International Conference on Neural Networks*, San Francisco, CA, 1993.

[28] Andreas S. Weigend, Neil A. Gershenfeld. *Time Series Prediction: Forecasting the Future and Understanding the Past.* Addison-Wesley, Reading, MA, 1994.

[29] Andreas S. Weigend, Bernardo A. Huberman, David E. Rumelhart. Predicting the future: a connectionist approach. In *International Journal of Neural Systems*, Vol. 1, No. 3, pages 193-209, 1990.

[30] Andreas S. Weigend, David E. Rumelhart, Bernardo A. Huberman. Generalization by Weight-Elimination applied to Currency Exchange Rate Prediction. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, Singapore, November 1991.

[31] P. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences.* Ph.D. Thesis, Harvard University, 1974.

[32] R. J. Williams, D. Zipser. A learning algorithm for continually running fully recurrent neural networks. In *Neural Computation*, No.1, pages 270-280, 1989.

Vita

Roy Schwärzel was born on October 9, 1966 in Berlin, Germany. He received the B.A. in General Business and the MBA in Finance from the Humboldt Universität Berlin (Germany) in 1991 and 1993, respectively. He was a student in computer science at the Technische Universität Berlin, Germany, from 1991 until December 1993. He received his B.S. degree in computer science from the Technische Universität Berlin in June 1993. During this time, he attended several graduate courses in computer science, primarily in the areas of Artificial Intelligence, and in particular Neural Networks and Genetic Algorithms. His current research interests include time series analysis using neural networks and high-performance computing on networks of workstations.

Permanent address: 9001 Wurzbach Rd. 1411
San Antonio, TX 78240

This thesis was typeset by the author in LaTeX.