2005

Multi-layer feed forward neural networks for foreign exchange time series forecasting

Bina R. Setyawati
West Virginia University

MULTI-LAYER FEED FORWARD NEURAL NETWORKS FOR FOREIGN EXCHANGE TIME SERIES FORECASTING

Bina R. Setyawati

Dissertation submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy
in
Industrial Engineering

Robert C. Creese, Ph.D., P.E., CCE, Chair
Majid Jaraiedi, Ph.D.
Wafik Iskander, Ph.D., P.E.
Powsiri Klinkhachorn, Ph.D.
James Denton, Ph.D.

Department of Industrial and Management Systems Engineering

Morgantown, West Virginia
2005

Keywords: NEURAL NETWORKS, FOREIGN EXCHANGE, TIME SERIES FORECASTING

ABSTRACT

MULTI-LAYER FEED FORWARD NEURAL NETWORKS FOR FOREIGN EXCHANGE TIME SERIES FORECASTING

Bina R. Setyawati

This dissertation examines the forecasting performance of multi-layer feed forward neural networks in modeling five weekly foreign exchange rates: Australian Dollars /U.S. Dollars (AUS/USD), Euro/U.S. Dollar (EUR/USD), Swiss Franc/U.S. Dollar (CHF/USD), British Pound sterling/U.S. Dollars (GBP/USD), and Japanese Yen/U.S. Dollars (JPY/USD). There are five objectives to accomplish. The first is to determine the key modeling factors that should be considered in topology determination. The second is to compare the performances of Genetic Algorithm (GA) and Modified Tabu Search (TS) in choosing the topology for Neural Networks (NN) implementation. The third is to investigate the suitable learning algorithm for NNs for time series forecasting by comparing Back Propagation (BP) with GAs and TS. The fourth is to conduct computational studies for multi-step ahead forecasting for GBP/USD and EUR/USD, as well as to study other accuracy forecasting issues. The last is to study the implementation of multivariate time series forecasting using NNs.

The results of the experiments performed indicate that one should examine the correct topology, especially the three most important factors (number of input nodes, hidden nodes, learning rate) prior to using NNs for time series forecasting.

The comparison performance of topology suggested using GA, TS, and benchmark led to the conclusion that neither GA nor TS is guaranteed to provide better results, especially in terms of percentage of true directional changes (DIR). However, if there is no prior knowledge of the problem, GA searches for topology determination are favored and provide reasonably good performances. GA is also preferred for NN training. Compared to BP, GA guarantees better performance in term of Mean of Absolute Percentage Error (MAPE) and, most of the time, performs better in terms of Mean of Square Error (MSE).

Caution should be taken in adopting the results, since the study of time periods indicated that the best topology for forecasting a specific foreign exchange is “data specific”; hence the best for a certain period is not always the best to forecast other periods. However, the chosen topology is reasonably useful for up to three steps ahead forecasting.

The trivariate time series, which incorporate interest rates of the two countries involved, did improve the results. Multivariate time series forecasts for monthly JPY/USD, as well as for monthly EUR/USD, produced a higher level of success than the one achieved in the previous experiment. The NNs were programmed using MATLAB®.

ACKNOWLEDGEMENT

This dissertation bears the imprint of many people who have made an important impact on my thinking. I am deeply indebted to many people for their part in making this dissertation possible. However, I particularly would like to acknowledge the advice and support of Dr. Robert C. Creese, Dr. Majid Jaraiedi, Dr. Wafik Iskander, Dr. Powsiri Klinkhachorn, and Dr. James W. Denton.

I wish to express my sincere appreciation and heartfelt gratitude to my advisor, Dr. Robert C. Creese, for showing extreme patience, support, and guidance throughout my research. I also would like to express my appreciation for the help and support that I received throughout my course of study at Industrial Engineering and Management System Engineering Department at West Virginia University from Dr. Majid Jaraiedi and Dr. Wafik Iskander. Thank you for being my mentors and for providing me with insightful suggestions not only in my study but also in nurturing my baby. I am also indebted to Jennifer McIntosh for her love and support for my family and me.

My overriding debt is to my husband, Sidharta Sahirman, and my son, Morgan, who provided me with the love, time, support, inspiration, and motivation needed to finish this dissertation. It's truly our dissertation.

Mom and Dad, I don't know how I can say my thank you for your love and continued prayers for us. I am very fortunate to have wonderful parents like you.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ANOVA = Analysis of Variance
AR = Auto Regression
ARIMA = Auto Regressive Integrated Moving Average
ARMA = Mixed Auto Regressive / Moving Average
AUS = Australian Dollars
BCD = Binary Coded Decimal
BP = Back Propagation
CHF = Swiss Franc
DIR = Percentage True Directional Changes
EUR = Euro
FIR = Finite Impulse Response
Forex = Foreign Exchange
GA = Genetic Algorithm
GBP = British Pound Sterling
JPY = Japanese Yen
MAPE = Mean Absolute Percentage Error
MLP = Multi-Layer Perceptron
MSE = Mean Square Error
NAR = Nonlinear Auto Regressive
NH = Number of Neighborhood
NMSE = Normalized Mean Squared Error
NNs = Neural Networks
NS = Number of local Searches
NVAR = Number of variables
RMS = Root Mean Square
SA = Simulated Annealing
Sig = Sigmoid
SPI = Swiss Performance Index
SSR = Stochastic Sampling with Replacement
Tanh = Hyperbolic Tangent
TC = Tabu Criterion
TL = Tabu Length
TS = Modified Tabu Search
USD = United States Dollars

Chapter 1

Introduction and Research Objectives

1.1 Introduction

In recent years, forecasting has gained more and more attention because of its critical role in making timely decisions in the face of uncertainty about the future. A good forecast needs to identify appropriate information and an appropriate forecasting method.

Forecasting methods are categorized into two groups, as presented in Figure 1.1, the more popular being the quantitative method. This method includes two different forecasting approaches: causal forecasting and time series forecasting. Causal forecasting considers a number of variables that are related to the variable that is to be predicted. Regression analysis is one type of causal forecasting. Time series forecasting uses a sequence of numerical values reflecting the time evolution of a certain magnitude to study the patterns of the data and then reflect the future values based on those patterns. Time series forecasting will be investigated in this study.

The procedures used to perform time series forecasting can be classified into linear and nonlinear methods. The most common linear methods are (1) Smoothing methods, which include Moving Averages and Exponential Smoothing, (2) Decomposition, and (3) Autoregressive (Box and Jenkins Methods). These methods work well for linear time series but fail to model complicated nonlinearity and trends in time series. Unfortunately, nonlinear time series forecasting is not straightforward and the theory does not guide the model building process by suggesting a functional

relationship between relevant lags and the response variable. Conventional nonlinear models such as threshold autoregressive models (Tong, 1990) and time-varying parameter models (Nicholls and Pagan, 1985) generally pre-specify a special nonlinear function to be used. The difficulty in choosing the nonlinear function has made these models ineffective for modeling nonlinear time series.

As the attention to forecasting processes grow, new techniques such as Neural Networks (NNs) are being developed. This method is a potential alternative tool in overcoming the problem of pre-specifying the nonlinear function. NNs are a good candidate for this task because of their universal approximation properties. They do not require inside knowledge about the process under investigation. It also has been shown that NNs outperformed Autoregressive Integrated Moving Averages (ARIMA).

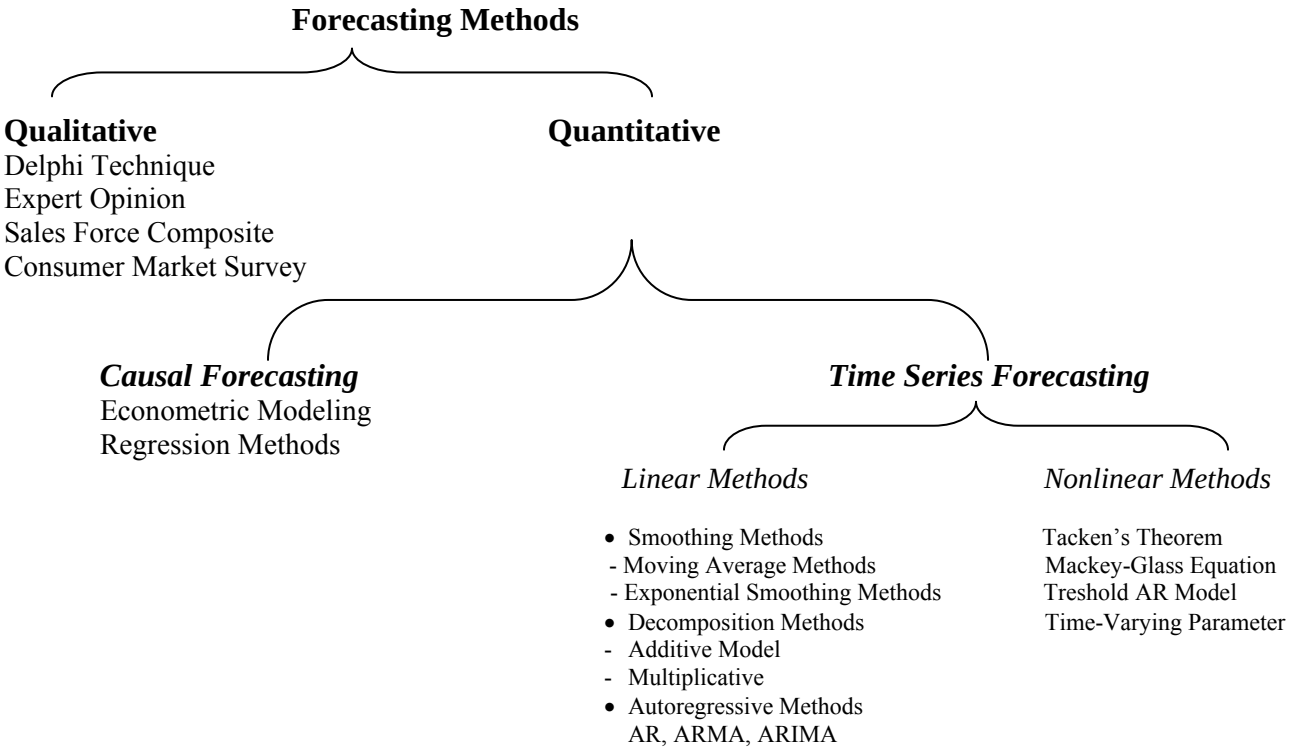


Figure 1.1 Classification of Forecasting Methods

Quantitative forecasting methods include causal and time series. Likewise, NNs can be implemented correspondingly based on these two categories and is a promising tool. NNs are extremely useful for causal forecasting when one does not have any idea of the functional relationship between the dependent and independent variables. NNs can replace the function of common statistical methods such as linear regression. As explained by Warner and Misra (1996), generalized linear regression is equivalent to a single layer neural network. Krishnaswamy et al. (2000) suggested that NNs for causal forecasting are best applied where:

(1) one can specify particular influences on a phenomenon whose outcome is known with certainty
(2) the relationship cannot be described
(3) the relationship is not necessarily linear, and
(4) there are no known models.

Additionally, Denton (1995) has shown NNs outperformed linear regressions especially under less than ideal conditions (when there are outliers in the data, when two independent variables are highly correlated, or when the model is mis-specified). Furthermore, NNs avoided some of the pitfalls common to the regression approach.

In regard to time series forecasting, Yao et al. (1996) stated that time series forecasting is the most exciting application of NNs. NNs, which are generalized nonlinear forecasting models, are reliable for modeling nonlinear time series. NNs are non-parametric data driven approaches and Linear Autoregressive is a special case of the model in which there are no hidden nodes. From a statistician's point of view, NNs are analogous to nonparametric, nonlinear regression models. Some major advantages of

NNs are that they: (1) are reliable for modeling nonlinear time series; (2) do not require any assumptions for underlying data to be forecast as opposed to normality assumptions commonly found in statistical methods; and (3) are able to learn even in the case of noise and or missing data; and (4) have the ability to handle discontinuities (Paik, 2000).

There are some drawbacks for NNs for forecasting that require special attention. These drawbacks must be addressed to improve NN performance for time series forecasting.

Since the time series forecasting cases are very broad, one must focus on a specific implementation. It has been known that among different kinds of time series forecasting problems, foreign exchange (forex) forecasting is one of the most interesting problems in practice. The forex market is the largest financial market in the world and trades enormous amounts of money, estimated at several trillion dollars daily.

Several authors claimed that exchange rates are rather unpredictable and that a random walk model is often a better predictor than nonlinear models. However, Medeiros et al. (2001) have shown that there are some supportive results in favor of linear and nonlinear models against the simple random walk concerning the predictability of exchange rates when mean absolute error is used as the performance measured. Similarly, Leung et al. (2000) concluded that neural networks have a higher degree of forecast accuracy than random walk for three monthly exchange rate studied.

The forex forecasting problem is difficult because of two reasons: (1) the time series is noisy and non-stationary, and (2) many factors leading to forex fluctuation cannot be captured precisely or are too numerous to be modeled. Forex time series data are chaotic system problems. They are inherently complex, thereby generating complex

error surfaces. These time series are most challenging in terms of the difficulty in obtaining good forecasts. Based upon those arguments, this research will focus on optimization of NNs for foreign exchange time series forecasting.

1.2 Needs for Research

A few topics must be investigated to improve NN performance for time series analysis. The first topic is NN topology. As stated by Adya and Collopy (1996), the most important drawback of neural networks is that mixed results have been reported in the literature. Extensive literature studies suggest that different NN structures and parameters lead to different results for the same problem. Unfortunately there is a lack of systematic approaches to NN model building, and the effect of key modeling factors on performance has not been thoroughly examined. The common method employed to determine NN topology is trial and error. For that reason, one must obtain a tool to select the most appropriate NN topology (i.e. number of input nodes and hidden nodes) and related information on system inputs (i.e. learning rate, momentum, activation function) in an attempt to produce the most accurate forecast. Hence, optimization of network topology is needed. Rather than using the trial and error approach, optimization methods should be applied to enhance the performance by making certain that the correct topology has been used.

Considering the importance of the topology to NN performance, this research topic should be explored extensively. This includes the determination of NN architecture and the parameter values to obtain the best results for each data set. The choice of NN

architectures were decided based on experiments using Back Propagation (BP) as the learning algorithm.

The second area of emphasis is that the most common learning algorithm used (BP) is based on the gradient descent method. It is well known that this method is not the best non-linear optimization technique, but it was chosen for its suitability for network problems. The serious problem with BP is that it may reach a local rather than global minimum. The modifications of BP will capture this property. Global maximization methods are good alternates for NN learning. Although some existing literature demonstrates the superiority of global search to BP, it provides limited information about the relative performance of global search algorithms for real time series data. Additionally, no study used more than one time series in its conclusions. It is believed that there is insufficient data to make general conclusions of the comparative performances.

In addition, regardless of the large number of publications of NN applications, there are only a few papers that have examined multivariate time series. From all papers reviewed, none of them studied the performances of multivariate time series for forex forecasting. Considering the significance of this topic, research in NNs for multivariate time series is necessary.

NN applications for either univariate time series or multivariate time series forecasting have concerned only single-step ahead forecasting. No literature for NN performance for multi-step ahead time series forecasting performances was found. Experiments will be performed to gain the working knowledge of multivariate time series

application for foreign exchange forecasting, as well as to study the performances of multi step ahead forecasting for these particular applications.

1.3 Research Objectives and Organization

1.3.1 Research Objectives

The five major research objectives are:

1. To study the effects of architecture and parameter values on NNs forecasting performance to (1) determine the key modeling factors that should be considered in topology optimization, and (2) provide the forecasting benchmark for each time series under investigation by constructing good NNs for this specific implementation.
2. To compare the performances of two Meta heuristics (Genetic Algorithm (GA) and Tabu Search (TS)) in choosing the number of hidden nodes and time-delayed inputs of NNs for time series forecasting. The implementation will be rated as good if the network topology suggested provides an equivalent or better result as the benchmark for the given problem.
3. To investigate the suitable learning algorithm for NNs for time series forecasting by comparing BP with GA and TS.
4. To conduct computational studies for multi-step ahead forecasting for GBP/USD and EUR/USD based on the neural network selected from the previous objectives.
5. To implement multivariate time series forecasting using NNs.

1.3.2 **Organization**

The scope of this research is to analyze a time series forecasting system in which the forecasts are accrued from an NN model. The main focus is to minimize the forecast error by determining the appropriate topology and weight matrices using NNs, GAs, and TS.

This dissertation is organized into nine Chapters. Chapter 1 is the introduction, while Chapter 2 gives a systematic literature review of the topics related to the research. Chapter 3 describes the methodology used in this research. Chapter 4 discusses the design of experiments conducted, the topology chosen, and the benchmarks selected for each case. In Chapter 5 the results of trials to determine appropriate topology for NN forecaster are presented. Chapter 6 describes the methods and related results to determine NN weight matrices. Chapter 7 shows computational accuracy issues using the results from Chapter 4 and 5 for certain foreign exchange forecasting problems, giving the numerical results of the comparison between the developed model and its benchmarks. Furthermore, it discusses the results of multi-step ahead forecasting along with the forecasting accuracy issues. Applications of the method for multivariate time series forecasting are explained in Chapter 8. In Chapter 9, conclusions are drawn and recommendations for future research directions are outlined.

Chapter 2

Background and Research Survey

2.1 Time Series Forecasting Methods

A time series is defined as any univariate or multivariate data collected over time and arranged in temporal order. It can consist of either continuous or discontinuous values. Both linear and non-linear models are available to model time series behavior. The application of time series forecasting method includes two basic steps: (1) Analyze the data series; and (2) Select the forecasting method that best fits the data series. There are several methods available for time series forecasting, particularly for linear time series. Some important methods are grouped based on the Makridakis and Wheelwright (1989) classification.

2.1.1 Smoothing Methods

Smoothing methods consist of two subclasses: moving average methods and exponential smoothing methods. Moving average methods conform to the conventional definition of an average – equal weighting of the number of values included in the average. The larger the values included in the averages the higher the smoothing effect. The exponential smoothing methods apply an unequal set of weights to past data. These weights decay in an exponential manner from the most recent observation to the most distant observation. Other methods in this group include double exponential smoothing, higher order smoothing, Winters' method, and double moving average.

The technique of one-step ahead forecasting with single moving averages can be represented as follows:

$$F_{t+1} = S_t = (X_t + X_{t-1} + \ldots\ldots\ldots\ldots\ldots + X_{t-N+1}) / N \qquad (2\text{-}1)$$

$F_{t+1} = S_t$ = Forecast for time t+1

X_i = Actual value at time i

i = Time period

N = Number of values included in average

The general form used in computing a one-step ahead forecast using the single exponential smoothing method is as follow:

$$F_{t+1} = \alpha X_t + (1-\alpha) F_t \quad \text{or} \qquad (2\text{-}2)$$

$$F_{t+1} = F_t + \alpha (X_t - F_t) = F_t + \alpha e_t \qquad (2\text{-}3)$$

Where:

α = smoothing parameter; $0 \leq \alpha \leq 1$

e_t = error of previous forecast

The smaller the value of α, the smoother (less fluctuation) the forecasts.

Single Exponential Smoothing does not work well in following the data when there is a trend. In that situation, one should use Exponential Smoothing which introduces a second equation with a second constant, γ ($0 \leq \gamma \leq 1$). The value of γ must be chosen in conjunction with α. The two equations associated with Double Exponential Smoothing are:

$$F_t = \alpha X_t + (1-\alpha) (F_{t-1} + b_{t-1}) \qquad (2\text{-}4)$$

$$b_t = \gamma (S_t - S_{t-1}) + (1-\gamma) b_{t-1} \qquad (2\text{-}5)$$

Where:

γ = double exponential smoothing parameter; $0 \leq \gamma \leq 1$

2.1.2 Decomposition Methods

Decomposition methods identify three separate components of the basic underlying pattern that characterize the time series. The decomposition assumes that data consist of pattern and error, and that the pattern is made up of trend, cycle, and seasonality. Decomposition methods are among the oldest forecasting approaches (Makridakis and Wheelwright, 1989).

The general mathematical representation of the decomposition approach is as follows: $X_t = f(S_t, T_t, C_t, E_t)$ (2-6)

Where:

X_t = time series value at period t

S_t = seasonal component at period t

T_t = trend component at period t

C_t = cyclical component at period t

E_t = error component at period t

This approach includes two methods: (1) Additive Decomposition Method, and (2) Multiplicative Decomposition Method. In the first method, the function is addition ($X_t = S_t + T_t + C_t + E_t$), while in the second, the effects of seasonality are considered to be multiplicative, that is, growing (or decreasing) over time, i.e. $X_t = S_t \times T_t \times C_t + E_t$.

2.1.3 Autoregressive Methods

Autoregressive methods, unlike the smoothing methods and decomposition methods, can deal with any data pattern. The basic tool employed by this method is correlation. Autocorrelation among successive values of a time series is a key tool in identifying the basic pattern and determining an appropriate model for the time series (Box and Jenkins, 1976). The degree of the relationship is measured by the correlation coefficient. The correlation coefficient of strong seasonal or cyclical character will be high. Three general types of auto correlation are: Auto Regressive (AR), Moving Average (MA), and Mixed Auto Regressive / Moving Average (ARMA). The auto regressive model is of the form:

$$Y_t = f_1 Y_{t-1} + f_2 Y_{t-2} + f_3 Y_{t-3} + \ldots\ldots + f_p Y_{t-p} + e_t \qquad (2\text{-}7)$$

Where f_p is the autocorrelation coefficient between original time series data and the same time series with p lack periods, where p is the number of terms to be included and e is the residual term.

Among the ARMA methods, the Auto Regressive - Integrated - Moving Average (ARIMA) methodology is one important group of linear statistical models that finds extensive use in industrial applications. The wide acceptance of standard ARIMA models and their recent statistical refinements make them a good reference for performance comparison for NNs.

A nonseasonal ARIMA model is classified as an "ARIMA (p,d,q)" model, where:

p = the number of autoregressive terms,

d = the number of nonseasonal differences, and

q = the number of lagged forecast errors in the prediction equation.

ARMA is a special class of ARIMA in which d=0. Often the pattern of the data may be described best by the mixed process of Auto Regressive and Moving Averages elements. Wold described ARMA models in 1954. In spite of such early work, the development and application were severely limited during the early phases. Recently, ARMA development tools are widely available through commercial statistical software, which increase the utilization of the ARMA models.

ARIMA models theoretically are the most general class of models for forecasting a time series, which can be stationarized by transformations such as differencing and logging. The easiest way to think of ARIMA models is as a fine-tuned version of random-walk and random-trend models: the fine-tuning consists of adding lags of the different series and/or lags of the forecast errors to the prediction equation to remove any traces of autocorrelation from the forecast errors. The best short horizon forecasting techniques (Makridakis et al., 1982) are all special cases of ARIMA models. ARIMA development tools are widely available through most commercial statistical software, such as SAS and E-view.

2.2 Neural Networks

2.2.1 Basic Principles

NNs, also referred to as neurocomputers, connectionist networks, or parallel-distributed processors, are massively parallel-distributed processors that have a natural propensity for storing experiential knowledge and making it available for use. NNs are networks of artificial neurons grouped in sets of layers to form a network. There are three types of layers: an input layer, a hidden layer, and an output layer. The input layer

receives the inputs and directs them to the hidden layers. Neurons in the hidden layers perform the computations of the network and add degrees of freedom to the network, sending the results to the neurons in the output layer (Adya and Collopy, 1998).

The three basic elements of the neuron model include (Haykin, 1994):

1. A set of synapses or connecting links, each of which is characterized by a weight or strength of its own.
2. An adder for summing the input signals weighted by the respective synapses of the neuron.
3. An activation function for limiting the amplitude of the output of a neuron.

The manner in which the neurons of an NN are structured is intimately linked with the learning algorithm used to train the network (Balkin, 2000). In general there are four different classes of network architectures:

(1) Single layer feed forward networks,

(2) Multi-layer feed forward networks,

(3) Recurrent networks, and

(4) Lattice structures.

Examples of NNs classified into each of those classes are shown in Figure 2.1.

A network is classified as feed forward if it does not contain directed cycles, and is classified as recurrent if it does contain such directed cycles. The most popular and widely used network paradigm is the multi-layer feed forward network (Dougherty and Cobbett, 1997). The idea of a feed forward NN is that the inputs feed into the functions in the hidden layer, and there is no feedback. Also, the functions in the hidden layer do not feed sideways into each other. Instead, they feed onward to the output layer. And

there is no feedback, delayed or otherwise, from the output layer to the hidden layer. An NN is called a feed forward system because of the absence of feedback and the absence of interaction between hidden-layer functions. Multi-Layer Perceptron (MLP) is used in a variety of problems in forecasting because of their inherent capability of arbitrary input/output mapping (Zhang et al., 1998).

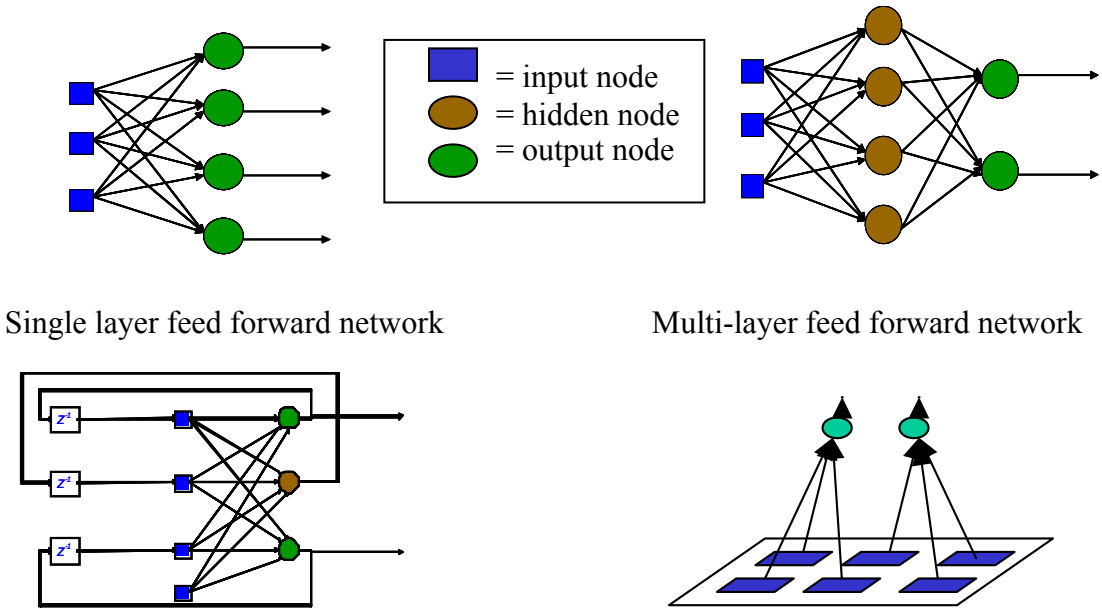


Figure 2.1 Four Classes of Network Architectures

In order to be useful, an NN must be trained before being applied. Training is an estimation of an NN model. This process can be accomplished using either a supervised or unsupervised method. The goal of the training process is to find the NN parameters, which are called connection strengths or network weights, that make the model errors small. Because the model parameters are nonlinear, it is necessary to use a nonlinear search algorithm. The most common search algorithm used is the back propagation (BP)

method. It has been estimated that 70% of NN applications report the implementation of the BP method (Nguyen, 2000). The disadvantage of this method is its learning speed (Balkin, 2000).

There are two basic types of NN training methods: supervised and unsupervised. Supervised NNs are typically used when the desired output is known and input-output data are available. Supervised NNs learn by forming an association between example inputs and associated correct outputs. In other word, NNs are trained by humans to perform specific tasks. During the training, the teacher evaluates whether the NN output is correct or not. If the output is correct, the weightings that produced output are reinforced. If not they are diminished. This is often used for cognitive research and problem-solving applications. MLP that employs BP is an example of a supervised NN. Conversely, unsupervised NNs don't require output for training. They learn in response to the arrangement of inputs, organize them into clusters during training, and then classify future inputs according to the cluster in which they occur. A self-organizing NN (called a Kohonen – the name of its inventor) is an example of an unsupervised NN. It is often used to analyze experimental data. An unsupervised NN is exposed to a large amount of data and tends to discover patterns and relationships in that data.

NNs are not programmed but trained. They do not require any assumptions for underlying data to be forecast. Also, they can develop models from incomplete or imperfect data (Abid et al., 2001).

2.2.2 **Mathematical Description**

A three-layer feed forward NN with three inputs, two hidden nodes, and one output node in Figure 2.2 will be used to illustrate and describe the NN process in mathematical terms.

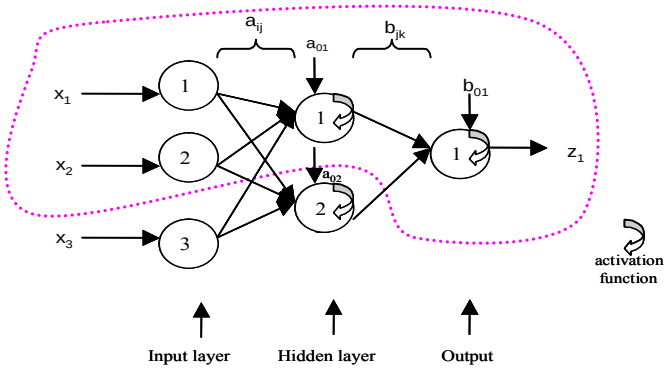


Figure 2.2 A 3-2-1 Feed Forward Neural Network

The generic weights are separated into two groups: the hidden node weights, a_{ij}, and the output node weights, b_{jk}. From Figure 2.2, consider a network with two inputs, x_1 and x_2, and one hidden node numbered 1 that computes the response or hidden output, y_1 and one output node numbered 1 that generates the output of the network, z_1 (as shown inside the dotted lines).

The propagation rule adds the bias weights, a_{01}, to the linear combination of input and weights generating u_1. Then the activation function, g, the logistic, generates the sigmoid-shaped response, y_1. That is:

$$u_1 = a_{01} + a_{11}\, x_1 + a_{21}\, x_2$$

$$y_1 = g(u_1).$$

This output of hidden node 1, y_1, is transmitted as input to the output node 1 in which the propagation role generates Nu_1 and the logistic, g, computes the output of the network, $z_1 = g(Nu_1)$. In symbols:

$$Nu_1 = b_{01} + b_{11}y_1$$

$$z_1 = g(Nu_1).$$

If one generalizes to a network with I inputs, J hidden nodes, J hidden outputs y_j, K output nodes, and k outputs z_k, then the following relations can be written. The hidden outputs are:

$$y_j = g(u_j);\ j = 1,..., J$$

$$u_j = a_{0j} + \Sigma_i\, a_{ij}\, x_i,$$

where a_{ij} are the weights from input i to hidden node j and a_{0j} are the bias weights for hidden node j. The response of hidden node j, y_j, is sigmoid-shaped of dimension I. These j responses are received by the k output nodes as input and compute k outputs, z_k, as the logistic transformation of Nu_k. That is:

$$Nu_k = b_{0k} + \Sigma_{\,j}\, b_{jk}\, y_j,$$

and

$$z_k = g(Nu_k);\ k = 1,..., K,$$

where b_{jk} are the weights from hidden node j to output node k and b_{0k} are the bias weights of output node k.

2.2.3 **Back Propagation**

Back Propagation (BP) is a well-known learning method for multi-layer perceptron training. The BP training algorithm is an iterative gradient algorithm designed to minimize the mean squared error between the actual output of the NN and the desired output.

BP involves two steps, a forward propagating step and a backward propagating step. In the forward propagating step, the training data set is presented to the input layer, which will propagate through the hidden layers until it reaches the output layer. In the backward propagating step, the calculated error (the difference between the actual output and the desired output) is propagated back to change the assigned weights. The magnitude or the error value indicates how large an adjustment must be made and the sign of the error provides the direction of the change. The recursive formula is the key to BP learning. It allows the error signal of a lower layer to be computed as a linear combination of the error signal of the upper layer. In this manner, the error signals are back propagated through all the layers from the top (output nodes) to the bottom (input nodes).

Learning rate controls how fast the weights are updated along the computed error gradient. It should generally be less than 1. The momentum parameter determines how much of the previous weight change will be retained in the present weight change computation. Thus, weight changes can build up momentum over time if they all head in the same direction, which can speed up learning.

2.2.4 **Applications**

NNs have been used in a wide range of applications. Some of the well-known applications have been in the areas of:

(1) Signal Processing

(2) Control

(3) Pattern Recognition

(4) Medicine

(5) Speech Production

(6) Business Applications

(7) Estimating

In time series study, NNs have been used to forecast a single variable as well as multiple variables.

2.3 Metaheuristic Algorithms

2.3.1 Genetic Algorithm

2.3.1.1 Basic Idea

Genetic Algorithms (GAs) are parallel, adaptive search algorithms inspired by the mechanisms of biological evolution. There are 5 basic components in GA: a method for encoding of chromosomes, a fitness (or objective) function, an initial population, a set of operations to perform evolutions between two chromosome populations, and working parameters (Osmera, 1995).

The main idea of GA is to start with a population of solutions to a problem and then attempt to produce new generations of solutions that are better than the previous ones. It includes four stages (Dorsey and Mayer, 1994):

(1) Initialization,

(2) Selection,

(3) Crossover, and

(4) Mutation.

In the initialization stage, a population of genetic structures, which are randomly distributed in the solution space, is selected as the starting point of the search. During the selection stage, each structure is evaluated using a set of user-defined criteria. The best structures are then selected for reproduction. These structures are combined using crossovers to allow existing structures to be tested further and also to introduce new structures into the population for evaluation. The final stage, mutation, functions as a background operator with a very low probability of application. It is used to alter one or more components of a selected structure, providing the means of introducing new information into the population (Hua, 2000). The GA is terminated when some criteria are satisfied, e.g. a certain number of generations, a mean deviation in the population, or when a particular point in the search space is encountered. GAs do not guarantee to find the optimal solution. However, the ability of GAs to find a near optimal solution is important

2.3.1.2 Encoding

Each generation of GA includes a certain number of individuals, commonly between 20-50 individuals. These individuals are encoded as chromosomes, so that the genotypes (chromosome values) are uniquely mapped onto the decision variable (phenotype). The search process operates on these encoding variables. When the search is finished, the variables are decoded into their phenotype values that can be applied to the problem.

Choice of encoding (representation) has a major impact on the performance of the GA in terms of accuracy and computation time. There are two common representation methods for numerical optimization problems: binary string and real number. The accuracy of the former depends on the number of bits used to represent a parameter. The latter uses a vector of real numbers, with each real number representing a single parameter.

Pham and Karaboga argued that binary string is the preferred method because it offers the maximum number of schemata per bit compared to other coding techniques. Furthermore, Yao (1999) stated that the binary representation is simple and straightforward to apply with classical crossover and mutation. Various binary coding schemes can be found in the literature, including Uniform coding and Gray scale coding.

On the other side, real values representation increases GA efficiency as there is no need to convert chromosomes to phenotypes before each function evaluation. Less memory is required as efficient floating-point internal computer representations can be used directly. And there is no loss in precision by converting to binary or other values.

The usual way of expressing a decimal number in terms of a binary number is known as pure binary coding. There are various binary codings available, two common ones, the standard binary code (8421 binary-coded decimal (BCD)) and gray binary code are discussed further. In standard BCD code, each bit is weighted by 8, 4, 2 and 1 respectively. In other words, each decimal digit is converted to its 4-bit pure binary equivalent. For example: 55 decimal = 0011 0111 BCD. Gray code is a non-weighted reflected binary code. Gray coding is an important code and is used for its speed. It is also relatively free from errors. In pure binary coding or 8421 BCD, counting from 7 (0111) to 8 (1000) requires 4 bits to be changed simultaneously. If this does not happen, then various numbers could be momentarily generated during the transition, creating spurious numbers that could be read. Gray coding avoids this since only one bit changes between subsequent numbers. The first 16 standard binary coded numbers, as well as Gray coded numbers, are indicated in Figure 2.3.

STANDARD BINARY CODE					GRAY CODE			
0	0	0	0	0=16	0	0	0	0
0	0	0	1	1	0	0	0	1
0	0	1	0	2	0	0	1	1
0	0	1	1	3	0	0	1	0
0	1	0	0	4	0	1	1	0
0	1	0	1	5	0	1	1	1
0	1	1	0	6	0	1	0	1
0	1	1	1	7	0	1	0	0
1	0	0	0	8	1	1	0	0
1	0	0	1	9	1	1	0	1
1	0	1	0	10	1	1	1	1
1	0	1	1	11	1	1	1	0
1	1	0	0	12	1	0	1	0
1	1	0	1	13	1	0	1	1
1	1	1	0	14	1	0	0	1
1	1	1	1	15	1	0	0	0

Figure 2.3 Binary Coding

2.3.1.3 Selection

In GAs there are two main selection procedures: proportional selection (roulette wheel) and ranking-based selection. In roulette wheel, fitness values of individuals represent the widths of slots on the wheel. Using this method, individuals are mapped into a wheel in which the size of each individual interval corresponds to the fitness value of the associated individual. The basic roulette wheel selection method is Stochastic Sampling with Replacement (SSR), which means the segment size and selection probability remain the same throughout the selection phase. SSR gives a zero bias but a potentially unlimited spread. Any individual with a segment size greater than zero could entirely fill the next population.

In the ranking-based selection, each individual is assigned a rank based on its fitness. The best individual in a population ranks first and the probability of selecting an individual is calculated as follows:

$P_i=(n_{max}-(n_{max}-n_{min})(i-1)/(N-1))/N,$

where N= number of individuals

$n_{max} + n_{min}=2$ and

$n_{max} \geq n_{min} \geq 0.$

Then a parent is selected by going through the following steps:

a. Generate a random value r between 0 and 1.
b. Set sum=0;
c. for i=1 to N do
   begin
     sum=sum+P_i;
     if (sum>=r)
     return i;
   end

2.3.1.4 Crossover

Crossover is the basic operator for producing new chromosomes in a genetic algorithm. It is considered the step that makes the GA different from other algorithms. It is used to create two new individuals (children) from two existing individuals (parents) picked from the current population by the selection procedure. Crossover produces new individuals that inherit parts of both parents' genetic material.

Genetic operators manipulate the characters (genes) of the chromosomes directly, using the assumption that certain individual's gene codes, on average, produce fitter individuals. The recombination operator is used to exchange genetic information between pairs or larger groups of individuals. Some common crossover operations are one-point crossover, two-point crossover, uniform crossover, and intermediate recombination crossover. The simplest form of crossover is single-point crossover.

The idea behind multi-point, and indeed many of the variations on the crossover operator, is that the parts of the chromosome representation that contribute the most to the performance of a particular individual may not necessarily be contained in adjacent sub-strings. Further, the disruptive nature of multi-point crossover appears to encourage the exploration of the search space, rather than favoring the convergence to highly fit individuals early in the search, thus making the search more robust.

Given a real-valued encoding of the chromosome structure, intermediate recombination is a method of producing new phenotypes around and between the values of the parents' phenotypes. Offspring are produced according to the rule $O1=P1+\alpha (P2-P1)$, where α is a scaling factor chosen uniformly at random over some interval. Line

recombination is similar to intermediate recombination, except that only one value of α is used in the recombination.

2.3.1.5 Mutation

Unlike crossover, in mutation a child string is produced from a single parent string. The purpose of the mutation is to randomly check the appropriateness of the path by applying occasional random alteration of a string associated with the current generation (Dorsey and Mayer, 1994). The mutation operator forces the algorithm to search new areas and helps avoid premature convergence.

2.3.2 Tabu Search

Tabu Search is an iterative procedure designed for the solution of optimization problems. It was invented by Fred Glover and has been used to solve a wide range of hard optimization problems, such as job shop scheduling, graph coloring (related), the Traveling Salesman Problem (TSP) and the capacitated arc routing problem. Most of the applications using this technique have been combinatorial problems, and only a few attempt have been made to use it for continuous problems (Glover and Laguna, 1997).

A Tabu Search consists of several ingredients, including a number neighborhood searches, number of local searches, tabu list, and aspiration level, whose combination in adequate proportions will make it an efficient heuristic procedure. A simple tabu search algorithm consists of three main strategies: forbidding strategy, freeing strategy, and short-term strategy (Glover, 1990). Apart from these strategies, there also can be a learning strategy, such as intensification and diversification.

The forbidding strategy controls what enters the tabu list. The main idea is to prevent the search from becoming stuck by making 'backward' moves tabu. In other words, it is employed to avoid cycling problems by forbidding certain moves (classifying them as tabu). This is usually achieved by constructing a tabu list with a specified length. The length of a tabu list is called the tabu list length (TL). The appropriate value of TL is important. A too small value will lead to high probability of cycling, while a too large value will drive away the search from good solution regions before these regions are completely explored. An additional feature, called an aspiration function, can be included in the model to cancel the tabu status of a certain solution, if it is desired. This feature is important to improve the fact that too many solutions may be forbidden.

The freeing strategy controls what exits the tabu list and when. The strategy deletes the tabu restrictions of the solutions so that they can be reconsidered in further steps of the search. A tabu solution remains on the tabu list for a duration of TS iterations.

The short-term strategy manages the interplay between the forbidding and freeing strategies to select trial solutions.

Tabu Search requires tracking the best solution so far. For each feasible solution, an appropriate neighborhood has to be defined. A stopping criterion terminates the tabu search procedure. The common stopping criterion is a lower bound. If it is not available, the iterations can be stopped by setting the number of iterations that have been performed in total or the one that has been performed without improving the value of the best solution (De Werra and Hertz, 1989; Hertz and De Werra, 1990). There is an ad hoc

element in choosing the various tabu rules. The method is often faster at finding good solutions than simulated annealing.

Sexton et al. (1998) claimed to apply Tabu Search for neural network training. In their work, the neighborhoods are defined as randomly drawn points from a uniform distribution while the local searches are random searches within the neighborhood. The neighborhood was a region restricted to $\pm$ 1 % for each weight in the initial neighborhood point. Other parameters used are the size of Tabu List (TL) and Tabu Criterion (TC). An aspiration level condition was also included.

An explanation of their algorithm that is adapted for this research is as follows: An initial solution Xo (weights of NNs) is randomly drawn from a uniform distribution in the range of [-10 10]. The function value (MSE of the NNs) is then calculated. Since Xo is the current minimum solution and f(Xo) is the minimum function value, both are used to set the best solution and function value parameters, X_{best} and f_{best} as well as entered into separate Tabu Lists.

The next solution is then randomly drawn from the neighborhood (Xo $\pm$ 1%). Then, the function value is calculated and checked for acceptance by checking the aspiration level and tabu conditions. The aspiration level is checked by directly checking the new function value with $f_{best.}$ If the new value is less than f_{best} then the point is automatically accepted and both X_{best} and f_{best} are updated, as well as entering both in their respective TLs. Otherwise the tabu conditions are tested for solution acceptance.

The tabu conditions are twofold, with the first condition predetermining the need for the second condition. If new function value (f_{new}) is within $\pm$ 0.01 of any value in the function value TL, the second tabu condition would then be applied, otherwise, the point

is accepted and the respective TLs are updated. However, if the new function value is within this tolerance, a check for specific solution weights of X_{new} and the weights for the point which corresponds with the tabu list function value is performed. If all of the weights in X_{new} are within the tolerate from this tabu list point, the point is rejected, otherwise the point is accepted and X_{new} and f_{new} are entered into TLs, dropping the oldest solution and function value from the list. This local search continues for NS number of iterations, unless there is no improvement for 10 times (maximum number of iterations without any improvement). This process is repeated for NH number of neighborhood searches.

2.3.3 Simulated Annealing

Simulated Annealing (SA) was introduced by Metropolis and is used to approximate the solution of very large combinatorial optimization problems (e.g. NP-hard problems). SA is a global optimization heuristic. The technique originates from the theory of statistical mechanics and is based upon the analogy between the annealing of solids and solving optimization problems.

Simulated annealing is a Monte Carlo approach for minimizing multivariate functions. The term simulated annealing derives from the roughly analogous physical process of heating and then slowly cooling a substance to obtain a strong crystalline structure. In simulation, a minima of the cost function corresponds to this ground state of the substance. The simulated annealing process lowers the temperature by slow stages until the system "freezes" and no further changes occur. At each temperature the simulation must proceed long enough for the system to reach a steady state or

equilibrium. This is known as thermalization. The time required for thermalization is the decorrelation time and correlated microstates are eliminated. The sequence of temperatures and the number of iterations applied to thermalize the system at each temperature comprise an annealing schedule. To apply simulated annealing, the system is initialized with a particular configuration. A new configuration is constructed by imposing a random displacement. If the energy of this new state is lower than that of the previous one, the change is accepted unconditionally and the system is updated. If the energy is greater, the new configuration is accepted probabilistically. This is the Metropolis step, the fundamental procedure of simulated annealing. This procedure allows the system to move consistently toward lower energy states, yet still "jump" out of local minima due to the probabilistic acceptance of some upward moves. Because it consistently moves toward the minimum, if the temperature is decreased logarithmically, simulated annealing guarantees an optimal solution.

This principle can be employed as an optimization technique in computer science. More specifically, in Artificial Intelligence, simulated annealing is used to help an NN avoid local minima in its energy function. It is suggested that because of its highly iterative (thus computationally slow) nature, simulated annealing should be completely avoided, combined with the gradient descent algorithm, or only used when speed is not of the essence. Since simulated annealing can deal with highly dimensional minimization problems, or problems with many false minima, it should always be considered as an alternative when other methods fail.

2.4 Neural Network Applications

2.4.1 Neural Network Applications for Forecasting

Hill and O'Connor (1996) compared the performance of NNs to six statistical time series methods generated in major forecasting competition (Makridakis et al., 1982). The methods included deseasonalized single exponential smoothing, Box Jenkins, deseasonalized Holts Exponential smoothing, graphical forecasting approach, combine forecasting, and Carbone Longini filter method. Beside those, Naïve method was employed. Based on several different architectures examined, it is suggested that 3-2-1 be used for annual data and 9-4-1 for monthly data. The results showed that NN did significantly better than traditional methods. Variance of the NN model forecast errors was almost always smaller than those of traditional models and of the reference average.

White (1998) studied the usefulness of NN applications for economic time series for IBM daily stock returns. A three-layer, fully connected NN with five input nodes and five hidden nodes was used. The training data included the daily stock returns from the second quarter of 1974 to the first quarter of 1978. The network considered failed in finding evidence against the simple efficient market hypothesis. White suggested that the problem might arise from the training algorithm employed (BP) in this case study. A global optimization method such as simulated annealing or the genetic algorithm would be preferable.

Kolarik and Rudorfer (1994) showed the application of NNs for time series forecasting. The data used were IBM common stock daily closing price for the period May 1961-November 1962 and monthly totals of international airline passengers from 1949-1960. A three-layer network with BP algorithm was used. The learning rate and

momentum terms were varied from 0 to 1 with 0.1 steps. The numbers of epochs used were varied between 60-138. Based on the results, it was concluded that NNs with more than 50 hidden nodes were not suited for time series forecasting. For the first data set, the best architecture found was 70 input nodes, 45 hidden nodes. For second data set, the best architecture found was 80 input nodes and 30 hidden units. For both data sets, the learning rate used was 0.1, and the momentum term was 0.9. He also suggested two working architectures for time series forecasting: 8-8-1 and 6-6-1.

Zhang et al. (2001) implemented NNs for eight nonlinear univariate time series data generated from regressive and moving average models that are replicated 30 times. It was concluded that NN models were more competent compared to Box Jenkins models. Wikowska (1995) compared back propagation NNs with econometric models to predict the Polish stock exchange. He used back propagation NNs with the following topology: 5-3-3, 3-2-1, 2-1-1, 5-3-1, and linear activation function in the output layer. He concluded that NNs performed better. By comparing NNs with 6 other techniques for predicting inflation, Aiken (1999) indicated that an NN may be able to forecast the inflation fairly accurately. Moshiri and Cameron (2000) had the same opinions. They found that back propagation NNs out performed traditional econometric approaches (ARIMA and VAR) in some cases based on the same variables used.

Walczak (2001) explained that financial time series, particularly foreign exchange rate forecasts, are difficult to model. He suggested general heuristics for NN design in financial domains: The more knowledge that is available to NN for forming its model, the better the ultimate performance of the NN, with a minimum of two years training data as a nominal starting point. It has been shown that NNs provide better fit compared to

linear regression and random walk for this purpose (Balkin and Ord, 2000). Mehta (1995) suggested the applicability of NNs to foreign exchange forecasts and concluded that NNs are currently the best problem-solving tool available for non linear time series. He cautions, however, that it takes quite a lot of understanding, experience, and plenty of experiments to achieve a stable set of networks (Khrisnaswamy et al., 2001).

Setyawati (2004) studied the performance of GA learning for time series forecasting using NNs. The case study was the foreign exchange rate predictions for JPY/USD. In this comparison study, the two network architectures applied were: 3-2-1 and 5-3-1. When BP was used, back propagation plus momentum term was employed. The learning rate was set at 0.9, and the momentum term was set at 0.03. The activation function was tanh. The learning was stopped after 1000 iterations. The architectures and parameters used were based on the results of previous experiments (Setyawati et al., 2003, Yao and Tan, 2000). For the 5-3-1 model, the average correctness of gradient predictions from three replications for the BP and GA were 50.72% and 54.33% respectively. Besides the better average, GA results gave much lower standard deviation compared to BP results. Yao and Tan (2000) reported an average correctness of 53.40 % using the same architecture but different set of weekly data, while that obtained using ARIMA was 44.32 %.

A summary of some previous researchers of financial time series forecasting using NNs is given in Table 2.1. NNs always performed better if a certain condition (i.e. correct parameters are employed) were satisfied. Otherwise, the results usually are different.

Table 2.1 Research Studies on Time Series Forecasting of Forex

Researchers:	Yao and Tan, 2000	Yao, Li, and Tan	Yao,Poh, and Jasic, 1996
Application:	univariate time series for weekly exchange rate forecasts	univariate time series for weekly exchange forecasts	univariate time series for weekly exchange rate forecasts
	for Yen, Mark, Pound, Franc, AUS $ vs US $	for Franc vs US $	for Yen, Mark, Pound, Franc, AUS $ vs US $
Data:	Daily rates of each currency for period	Daily rates for period March 1, 1983 to Nov. 3, 1995	Daily rates of each currency for period
	18 May 1984 to 7 July 1995	Segmented into 13 different data sets, each with span of	18 May 1984 to 7 July 1995
	(weekly forecasts over 7 daily data were adopted)	6.5 years with overlapping period of 0.5 years	(weekly forecasts over 7 daily data were adopted)
Source of data:	N/A	Singapore Forex Market	Singapore Foreign Market
Number of data:	1910 data		510 weekly data
Type of NNs:	3 layers feed forward NNs	Singapore Forex Market	3 layers feed forward NNs
Learning algorithm:	Back propagation	Back propagation	Back propagation
Architecture:	5-3-1,6-3-1 (for exp 1: time delay models) , 5-3-1, 6-4-1	The best architecture of each segment varies,	5-3-1,6-3-1 (for exp 1: time delay models) , 5-3-1, 6-4-1
	(for exp 2: using indicators and hybrid), 6-2-1, 6-3-1, 6-4-1	that consist of 6-2-1, 6-3-1, and 6-4-1	(for exp 2: using indicators and hybrid), 6-2-1, 6-3-1, 6-4-1
	(for exp 3: consistency test of different sets)		
Activation function:		hyperbolic tangent	
Input:	1. Time delay of Predictors	Technical indicator : Moving averages	1. Time delay of Predictors
	2. MA5, MA10, MA20, MA60, MA120, (One time delay)	(MA5, MA10, MA20, MA60, MA120, One time delay)	2. MA5, MA10, MA20, MA60, MA120, (One time delay)
Input Pre-processing	-	Normalized within [-1 1]	
Size of training and test	Six year training data and half year test data, totally 12 sets	Six year for training and validation, half year test data	2/3 for training, 2/15 for validation, 3/15 for test set
Stopping criteria:	-	Treshold is 0.0005 with max. 10,000 epochs	
Statistical performance measured:	NMSE, sign statistics, directional change statistics	NMSE, correctness of gradient	NMSE, correctness of gradient predictions
Methods compared:	ARIMA		ARIMA
Results:	Exp 1: NMSE average is .1499, gradient 54.03 %	NMSE is [0.03 0.66]; gradient is [42.31 69.23]	Exp 1: NMSE is [0.05-0.32], gradient is [51% 56%]
	Exp 2: Hit rates 70% of AU$& BPS, 60% of CHF & DEM,	Average paper profits of 11.36 % - 27.59% achieved	Exp 2 (Except for Yen) NMSE is [0.03-0.06], gradient is
	50% for JPY, ARIMA 55.86 %	for different trading strategies over different time horizons	[61% 75%]; For Yen the values are [1.2-2.0], 46%. Gradient
			for ARIMA(1,0,1) is [39% 53%] ; (2,0,2) is [44% 56%]
Other important information:	In respect of NMSEW, gradient or profit, NNs model	Number of hidden layer used is approx. half of inputs	In respect of NMSEW, gradient or profit, NNs model
	using simple technical indicator (MA) better than ARIMA	Learning rate used is 0.9, momentum term 0.0003-0.02	using simple technical indicator (MA) better than ARIMA

Table 2.1 Research Studies on Time Series Forecasting of Forex (continued)

Researchers:	El Shazly et al., 1999	Walczak, 2001	Ankenbrand and Tomassini, 1995
Application:	univariate time series for 3 month spot rate exchange	univariate time series for daily exchange rate forecasts	multivariate time series NNs for montly forecasting of SPI
	for Pound, Mark, Yen, Franc vs. US $	for Yen, Mark, Pound vs US $	
Data:	-	Daily rates of each currency for period	SPI period January 1987-December 1994 (predictor)
		1 March 1973 to 30 June 1995	1987-1994 data of S&P 500, DEM/USD exchange rate
			average bond interest rate of CHF and US (indicators)
Source of data:	N/A	N/A	N/A
Number of data:	-	vary from 1 year to 21.75 years data	96 for each series
Type of NNs:	3 layers feed forward NNs	3 layers feed forward NNs	3 layers feed forward NNs
Learning algorithm:	-	Back propagation	Back propagation with changes in parameters
Architecture:	5-4-1	3-5-1 (Pound and Mark), 2-3-1 (Yen)	3 input, 2 hidden, and 1 output
Activation function			
Input:	original data	1,2,5 lags (Pound), 1,2,3 lags (Mark)	monthly differences of S&P 500, DEM/USD exchange rate
		1,2 lags (Yen)	monthly differences of interest rate CHF
Input Pre-processing	-	-	Predictor scaled [.2 .8], indicators scaled [-1 1]
Size of training and test sets:	-	-	84 data for training and 10 data for test
Stopping criteria:	-	50,000 epochs	If the error of the validation set increases
Statistical performance measured:	total absolute errors, mean absolute errors,	percentage of correct direction of change	NMSE and trend forecasting accuracy
	% correct direction of change		
Methods compared:	-	-	-
Results:	hybrid NNs+GAs seem to be well suited for	a maximum of two years of training data produces	
	the forecasting of financial data	the best NNs forecasting performances: 62.4 %	
		(Pound), 61.6% (Mark), 59.2%(Yen)	
Other important information:		Previous research by Walczak suggested that single	
		hidden layer outperforms two hidden layers	
		Methodology to find the best lag combination obtained	

2.4.2 Choices of Neural Network Architecture Classes for Time Series

Feed forward and recurrent multi-layer perceptron are popular NNs for complex time series tasks, such as forecasting financial time series. Hallas and Dorffner (1998) conducted a comparative study on several linear and nonlinear feed forward and recurrent NNs trained on artificially created time series. Several of the time series were generated by some of the NN models, in order to test whether they could learn to predict a time series that they could theoretically perfectly model. All networks used BP as the training method. The results showed that recurrent networks (Jordan and Elman types) did not adequately predict the sample time series. These networks also resulted in significantly sub-optimal predictions whenever a time series was sufficiently described by a linear model.

A simple feed forward network (a nonlinear autoregressive model) performs better than other models for most of the nonlinear time series. Hallas and Dorffner (1998) concluded that for unknown underlying nonlinear characteristics of a time series, the feed forward Nonlinear Auto Regressive (NAR) model appears to be most likely to lead to satisfying results. NAR is a two-layer perceptron with an input window of size 20 and 10 nonlinear hidden units with sigmoid activation functions.

Koskela et al. (1994) compared time series prediction with multi-layer perceptron (MLP), Finite Impulse Response (FIR), and Elman NNs in four different time series prediction tasks. All prediction tasks were one-step ahead forecasting. The performance measured was Normalized Mean Squared Error (NMSE). Simulations were done with an MLP network that had one hidden layer and one nonlinear output neuron, an Elman network that had one linear output neuron, and a FIR network that had one hidden layer

and one nonlinear output neuron. The comparisons were based on the best architecture of each network. The results showed that the MLP network outperforms in 3 out of 4 time series studied. In the other time series, MLP performed better in the training set and did nearly as good as the Elman network in the test set. The FIR network provided the worst results for all cases. In addition to the poor statistical performance, Elman and FIR training were slower than MLP. The training of the Elman networks were three to ten times slower than for MLP, depending on the training data size and the number of network parameters, while training for the FIR network was five to twenty times slower than for the MLP network.

2.5 Neural Network Building and its Performance

The development of high-quality NN models is difficult. Selecting the best NN architecture is crucial to the success of NN modeling (Hill and O'Connor, 1996). Several design factors, include selection of input variables, architecture of the network, and quantity of training data significantly impact the accuracy of NN forecasts (Denton and Hung, 1996).

Choosing the correct NN topology for use in a particular domain (e.g. corporate bankruptcy) with optimum generalization performance is not a trivial problem. It involves the daunting task of constructing a large number of NN topologies with different structures and parameter values before arriving at an acceptable model (Chen et al., 2001). The main problem is that there is no fixed rule to determine the appropriate architecture or its parameter values. Very few researchers have studied the effect of key factors in NN modeling. Important factors to be considered in NN building are the

number of hidden layers and nodes, the number of input nodes, the activation function, the learning parameters, the size of the training set, data preparation, weight initialization, the cost function, the learning algorithm, and the stopping criterion.

2.5.1 Number of Hidden Layers, Hidden Nodes, and Input Nodes

It has been shown that one hidden layer is sufficient to approximate a continuous function and is appropriate for time series forecasting (Chan et al., 2000, Davey et al. 2000, Zhang et al., 1998, Zhang et al., 2001). The three-layer network is widely used in empirical work.

Hidden nodes are abstract constructs whose function is to introduce and control the nonlinearity of the NNs. The more hidden nodes, the more parameters and even more nonlinear equations resulted. Therefore they can represent a wider variety of functions. It is necessary to include enough hidden units so that the network can detect at least simple nonlinear regularities. If there are too few hidden nodes, the NN may not be able to generate a function that reflects the underlying problem. Having more hidden nodes than necessary will result in over-fitting of the training set and decreasing the ability to generalize the out-of-sample data. The magnitude of the weights is another quantity that affects the complexity of an NN, but to a much lesser degree.

There are a number of techniques for determining the optimal number of hidden nodes, but a popular method is bootstrapping (Efron and Tibshirani, 1993). Examples of rules of thumb to determine the number of hidden nodes are: (1) The number of hidden nodes should be 75 % of the number of input nodes (Salchenberger et al., 1992); (2) The number of hidden nodes is approximately the square root of the product of the number of

inputs and the number of outputs. The use of these rules, however, does not guarantee the effectiveness of the architecture. If there are very few inputs and outputs and the problem is complex, this method may underestimate the actual number required (Hansen et al., 1999). A rule of thumb, based on statistical classification theory, suggests that the number of connections in NNs should be less than 10% of the sample size.

Results from empirical works are varied. Some of the important clues to determine the number of hidden nodes are as follows: (1) One or two hidden nodes typically give the best forecasting performances in terms of MSE and MAPE (Zhang et al., 2001); (2) Two to five hidden nodes are recommended in terms of Bayesian Information Criterion (BIC) (McMenamin, 1997); (3) Balkin and Ord (2000) suggested using the number of hidden nodes in the range of one to the total number of inputs nodes; and (4) NNs with more than 50 hidden nodes are not suited for the task of time series forecasting (Dougherty and Cobbett, 1997).

One of two important issues for NNs in time series forecasting is the number of data points (Davey et al., 2000) that should be used in the forecast. It is perhaps the most important issue since it corresponds to the number of lagged observations used to discover the underlying patterns and or autocorrelation structures of time series.

Zhang et al. (2001) studied the effect of 3 main factors: input nodes (from one to five), hidden nodes (from one to 10), and training sample size (100, 200, 400). In their work, they were primarily concerned with nonlinear univariate time series forecasting using one hidden layer fully connected feed forward NN, one output node, and employed BP as the training algorithm. One-step ahead forecasts were performed, and MSE and MAPE of the results were measured. The time series investigated were generated from

eight nonlinear equations. The research was done using a simulated computer experiment based on a factorial experimental design. They found that the number of input nodes was more important to the performance of NN than the number of hidden nodes, while a large training sample was helpful. For most of the time series investigated, one or two hidden nodes typically gave the best forecasting performance. It was suggested that NN models were more competent than ARIMA.

Based on the obtained results using a three layer NN and one to eight hidden nodes in the network, McMenamin (1997) suggested employing the number of hidden nodes between two and five for time series data. He also showed that NN performances are better than regression models.

Setyawati (2002) compared the performance of NN for time series forecasting with smoothing methods. She showed that NNs outperformed moving average and single exponential methods for two data sets investigated (i.e. IBM common stock daily closing price and sales of new one-family houses). Setyawati (2003) compared NNs with Random Walk and ARMA models for commodity price forecasting. Results indicated that NN models generally outperformed those two models in predicting monthly zinc, copper, and tin prices.

Hansen et al. (1999) designed a Genetic Algorithm (GA)-guided selection of NN architectures. The results showed that impressive increases in forecasting accuracy could be achieved through applying NN techniques. The application of GA to determine the architecture allowed NNs to outperform statistical models on the Box Jenkins data sets (i.e. ARIMA models). Nasir et al. (2001) suggested the number of neurons for each

layer is related to the complexity of the input data and to the properties of the non-linear mapping from input to output.

2.5.2 Activation Function

Each neuron takes in the output from many other neurons. Once inside the neuron, the weighted signals are summed to a net value. The neuron calculates its output by finding the net value and then applying an activation function that produces an activation level inside the neuron. The activation is passed through an output, or transfer function, which produces the actual output for that neuron for that time. The activation function specifies what the neuron is to do with the signals after the weights have had their effect.

There are three basic types of activation functions (Haykin, 1994): (1) Threshold function, (2) Piecewise-linear function and (3) Sigmoid function. In the threshold activation function, a unit fires if the weighted sum of the inputs reaches or exceeds the threshold value. Paik (2000) stated that a nonlinear activation function does not guarantee that NN can represent any nonlinear function; for that, it is necessary to introduce hidden nodes into NN. Commonly used nonlinear activation functions are the sigmoid function and the tangent function.

2.5.3 Learning parameters

A learning rate determines the magnitude of a correction term applied to adjust the weight of each node. A large value causes the network to learn quickly, but it may

cause the training to be unstable or no learning occurs. Momentum term is the percentage of previous errors applied to weight adjustment in each training case.

2.5.4 Size of Training Set

It is necessary to divide the available data to develop an NN into two sets, one set for training purposes, and the other for test set. The size of the training sample set is an important issue for NN applications. Balkin and Ord (2000) concluded that a sufficiently long series is required to detect the nonlinearity and necessary for NNs to outperform simple methods. Walczak (2001) studied the effects of different sizes of training sample sets on forecasting currency exchange rates. The selection of input variables, the NN training algorithm, and the design of the NN architecture were not discussed. In his work, three foreign exchange rates were predicted, each using 11 different NN models. Results of the research showed that for most exchange rate predictions, a maximum of two years of training data produced the best NN forecasting model performance, which was 58% accurate for trading the pound sterling. This result contradicts the current financial NN development heuristics, which suggest using more data for training purposes (Zhang et al., 1998). Walczak (2001) noted his previous conclusion based on the extension of his research. His study on CHF/USD daily trading cautioned that a cutoff of two years of training data may not always be appropriate.

2.5.5 Preprocessing Data

Data preparation is crucial to the NN performance. For time series forecasting using NNs, transformation of input points to interval [0 1] is recommended. Beside the

transformation of input points to interval [0 1], it may be necessary to perform other kinds of data transformation. Kolarik and Rudorfer (1997) e.g., have shown that the log time series performed better than the usual [0 1] interval transformation.

Preprocessed-data may result in better forecasting performances. Ankenbrand and Tomassini (1996) used monthly differences of the SPI (Swiss Performance Index), rather than the original value of the SPI as a predictor, since it detrends the predictor. The predictor was scaled in the range of [0.2, 0.8].

2.5.6 Weight Initialization

The most widely used initialization method is the random initialization, either between [-0.5 0.5] or [-1 1]. Initialization influences the time required to converge, and whether the NN reach a global or local minima. Another famous initialization is Nguyen-Widrow method. This method is designed based on an analysis using the hyperbolic tangent activation function. The procedure is as follow:

(1) Initialize the weight between the input and the hidden layer as a random number [-0.5 0.5]. The initialization weight is called $w_{ij}(old)$.

(2) For each hidden unit (j=1,2, …,p), compute $||w_j(old)||$

(3) Reinitialize weight $wij= \beta\ w_{ij}(old)/|| wj(old)||$

(4) Set bias w_{oj} using random values $[-\beta\ \beta]$

where β = scale factor = $0.7\ (p)^{1/n}$, p = number of hidden nodes, n = number of input nodes.

Chen (1992) suggested that randomization of initial weights is inappropriate. He proposed the Forward Estimation Algorithm, with the basic idea to set the initial weight

space as close as possible to a global minimum before training to reduce the training time. In this method, weights between the input layer and the hidden layer are still initialized randomly, but weights between the hidden layer and the output layer are obtained through several matrix multiplications. Based on his findings for classification problems, he concluded that the proposed weight initialization algorithm improved the training speed. Moreover, it was shown that the weight space could be easily computed if the number of hidden units is equal to or greater than the number of training patterns minus one.

Chan et al. (2000) suggested a new weight initialization, which they believed resulted in better performance of BP. In their method, weights between the input layer and the hidden layer were still initialized randomly, but weights between the hidden layer and the output layer were obtained by multiple linear regressions. Koza and Rice (1991) illustrated how to find weights and architecture for an NN (including the number of layers and nodes in each layer) using a genetic algorithm for the one-bit adder. Unfortunately, neither the implementation program nor the detail algorithm was provided in his paper.

Setyawati (2002) compared three initialization methods (random, forward initialization method, and multi regression method) suggested that there is significant difference of the NN performances. The range test showed that the forward initialization method was worse compared to the random and multi regression methods. There was little difference between the two other models, but random initialization performed better.

2.5.7 Cost function

The BP cost function is the squared error, while the objective is minimizing the squared error value (MSE). Other cost functions commonly used are Normalized Mean Squared Error (NMSE) and Root Mean Squared Error (RMSE).

2.5.8 Learning Algorithm

Training an NN is a search in the so-called weight-space; that is, the space spanned by all weights in the NN. The goal of the search is to find a point in weight-space that minimizes a certain error criterion. Hence, it is equivalent to performing a minimization procedure in weight-space with respect to the error criterion. One of the most prominent and widely used algorithms is the BP method, which is based on the gradient descent minimization method, discovered independently by Werbos, Parker, and Rumelhart.

BP is probably one of the most important NN paradigms because it is reasonably simple to implement and works well for a wide variety of applications. BP is an iterative algorithm to minimize the mean squared error. It requires the use of a continuous differentiable activation function. It employs a nonlinear function, because a linear function achieves no advantage from the hidden units. Basically, BP is a first-order stochastic gradient descent method. This algorithm consists of two phases: the forward phase where the activations are propagated from the input to the output layer, and the backward phase, where the error between the observed actual and the requested nominal value in the output layer is propagated backwards in order to modify the weights and bias values.

In spite of its popularity, BP has several limitations. BP converges slowly and has difficulty in determining the network parameters (Chan et al., 2000, El Shazly et al., 1999, Hansen et al., 1999). Plus, it is sensitive to parameters such as learning rate and momentum rate and doesn't guarantee global minima (Chan et al., 2000).

A common approach to deal with BP drawbacks is to use second-order algorithms. In regard to training speed, the conjugate gradient method is promising. The conjugate gradient learning algorithm, which has a second order convergence property, is expected to reach convergence faster than the first order steepest descent approach (Chan et al., 2000). Besides this method, many modifications of BP have been implemented in past NN literature to overcome the BP problems. Sexton et al. (1999) demonstrated that such modifications are unnecessary if a sufficiently complex initial architecture and an appropriate global search algorithm is used for network training.

BP has unquestionably been a major factor for the success of NN applications. Nevertheless, some researchers concluded that its inconsistent and unpredicted performances in some applications were rooted from the use of BP as the learning method. BP is often trapped to local optima. Obtaining a global solution using this method is often dependent on the choice of starting values. Many modifications have been implemented in past ANN literature to overcome this problem, but they have had limited success. A more effective approach might be to replace the gradient base search technique with a global search technique, such as Genetic Algorithm (GA), Tabu Search (TS), or Simulated Annealing (SA).

2.6 Applications of Metaheuristic to Neural Network Design and Learning

2.6.1 Application of Genetic Algorithms to Neural Networks

The application of genetic algorithms to NNs has followed two separate but related paths: (1) genetic algorithms to find the optimal network architectures for specific tasks, (2) genetic algorithms to optimize weights in NNs.

Some previous studies have used the genetic search instead of gradient descent learning to establish the appropriate weight matrices in fixed architectures. They suggest that genetic algorithms are a promising learning method for NNs. Marti (1992) successfully employed genetic algorithm for training a recurrent network for a classification task. In this study, small fix-sized networks were treated with two genetic operations: mutation with probability of 0.03 and crossover with probability of 1.00 for 400 generations. El Shazly and El Shazly (1999) and Hung (2000) found that this algorithm resulted in better performance (in terms of total absolute forecast errors, mean absolute forecast errors, and the ability to correctly forecast the direction of the change in the exchange rate movement) and was suitable for networks. Aiken and Bsat (1999) stated that GAs overcome the local minimum problem. The application of GAs for NNs appears to be well suited for the forecasting of financial data (El Shazly and El Shazly, 1999, Hansen et al., 1999), as well as for forecasting construction demand (Hua, 2000). For the latter application, it has been shown that GAs outperform BP by reducing the average MAPE from about 6% to 1% for the case of the Singapore residential sector construction demand.

Dorsey, Johnson, and Mayer (1994) examined the performance of the genetic algorithm for training three-layer feed forward NN with six hidden nodes on six

problems: the exclusive OR problem, parity 4, T&C (a recognition problem), binary addition, mapping a line to itself and the meshed region problem. The inputs and the outputs were binary, while the weights were also represented as binary. The genetic algorithm used a population of 20 with a probability of mutation of 0.02. Weights values were drawn randomly from the interval [-1000 1000].

While applications of genetic algorithms to NN models have been used in some areas, only a few of them were applied for multi-layer feed forward NN. Furthermore, most of those works dealt with classification problems or causal forecasting as indicated in Table 2.2. The possibility of necessary modifications of the existing GA to enhance the multi-layer perceptron for time series forecasting has not been thoroughly examined.

Dodd (1991) presented arguments that it is necessary to optimize NN structure for classification problems. He used genetic algorithms by applying pure crossover for this purpose. The genetic algorithm was chosen based on three reasons: (1) GA is insensitive to correlation between the network parameters, (2) It is desirable to explore several different regions of search in parallel; and (3) It is required to have an optimization technique that is tolerant of noise present on estimation fitness due to different starting weights. Hansen et al. (1999) used a Genetic Algorithm to determine the number of hidden layers (1 or 2), the number of hidden nodes and the activation function in the hidden layers (choice of logistic and tanh), and the output layer (choice of linear and logistic). They suggested that the application of GA to determine its architecture allow NNs to outperform statistical models on the Box Jenkins data sets.

Based upon the early results, El Shazly et al. (1999) believed that the application of hybrid systems (three-layer feed forward + GA) seems to be well suited for forecasting of financial data. A hybrid NN 5-4-1 was employed to support this statement.

Gupta and Sexton (1999) have shown that the use of a genetic algorithm can provide better results for training a feed forward NN than BP. Their conclusion was based on a single application example, an artificial chaotic time series derived from the Mackey and Glass Equation. The training data was normalized to the range of [-1 1]. The initial five values of the time series were generated by drawing random values from a uniform distribution from the range [0 1]. In this comparison study, three network architectures were used, but the only difference was the number of hidden nodes (2,4, or 6), while the number of inputs was fixed at 5. The statistical performance used was Root Mean Squared Error. The momentum value was manipulated from 0.0 to 0.9 in 0.1 increments.

Sexton et al. (1999a) compared BP with simulated annealing via an intensive Monte Carlo study on seven test-functions. The SA algorithm implemented in this study was the one suggested by Goffe et al. (1994). The networks were compared on the basis of the Root Mean Squared (RMS) forecast error and six hidden nodes were used. It was shown that simulated annealing might indeed be a superior search alternative.

Sexton et al. (1999b) compared the performance of two global search techniques, Simulated Annealing and Genetic Algorithm. The three layer feed forward NNs with six hidden nodes were chosen. Six experiment problems were examined. The last problem was the Mackey-Glass equation. For this problem, five lagged values of the dependent variable were used as the input variables. They showed that GA, which was terminated

Table 2.2 Genetic Algorithms for Multi-Layer Feed Forward Neural Network Training

Source	Method	Model Applied	Neural Network Type	Topology	Task	Objective	Parameter Settings
Sexton et al., 1999	GA	Dorsey et al., 1994	Multilayer feedforward neural network	6 hidden nodes	Data sets generated from six different equations	SSE	Population = 20 Weight values were drawn randomly from the interval [-600 600] 10 replications Stopping criteria= 5,000 generations
Gupta and Sexton, 1999	GA	Dorsey et al., 1994	Multilayer feedforward neural network	2, 4, 6 hidden nodes	Data set drawn from Mackey and Glass Equation	RMSE	Population = 20 10 replications Stopping criteria= 100,000 generations
Sarkar and Yegna-narayana, 1997	GA		Multilayer feedforward neural network	30 hidden nodes	Classification Problems (4 classes of 400 data)	MSE	Stopping criteria= 3,000 generations
Osmera, 1995	GA	Osmera, 1995	N/A	N/A	Travelling Salesman Problem	N/A	N/A
Dorsey et al., 1994	GA	Dorsey et al., 1994	Multilayer feedforward neural network	6 hidden nodes	Exclusive OR (parity2) Parity 4, T&C, Binary Addition, Mapping a line, meshed region	SSE	Population = 20 Probability of mutation = 0.02 Probability of cross over = 1.0 Weight values were drawn randomly from the interval [-1000, 1000]

prematurely, provided solutions that typically dominated the SA results. Furthermore, by examining four other architectures (varying in the number of hidden nodes: 2, 4, 8, and 10), they have suggested that the best results were all obtained by the GA. Besides those, Sexton et al. conducted a comparison of three learning methods (GA, SA, and BP) using a time series problem, i.e. daily S&P 500 index closing price. Eight input nodes and six hidden nodes were used. The momentum and learning parameters were set to 0.9 and 0.5 respectively. It was concluded that GA was superior to SA and BP, with average RMSE to be 0.0071. SA failed on average to outperform.

2.6.2 Applications of Tabu Search to Neural Networks

Tabu Search has been applied to the network design problem and investigated as an alternative to the BP algorithm. Cannas et al. (2000) attempted to obtain an optimally designed Locally Recurrent NN architecture by implementing a Tabu Search (TS) Algorithm. In their work, the problem of choosing the number of hidden neurons and the number of taps and delays in the Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) network synapses was formalized as an optimization problem whose cost function to be minimized was the network error calculated on a validation data set. Cannas et al. applied the proposed approach to a specific Continuous Stir Tank Reactor application. The system had two inputs, two outputs, and one hidden layer. By training the network for 3,000 epochs, it was demonstrated that the optimal NN found was able to make accurate predictions for data not used in the training phase.

Battiti and Tecchiolli in 1993 have suggested the Reactive Tabu Search (RTS) approach for training multi-layer feed forward NN for classification tasks. De Werra and

Hertz (1989) proposed tabu search for associative memory problems, specifically the Hopfield network (Hertz and De Werra, 1990). More recent papers examined these possible alternatives to the problematic BP approach for forecasting applications.

Sexton et al. (1998) conducted comparison analysis between Tabu Search and BP for training three-layer feed forward NNs with 6 hidden nodes. Six problems were examined. Two Tabu Search algorithms were used, the preliminary and the extended Tabu Search. Different combinations of BP parameters were studied and included the alternate learning rate of 0.5 and 1, momentum of 0.3 and 0.9, and epoch size of 1 and 50. In all problems, the best epoch size for BP was one and momentum was 0.9. The preliminary found superior solutions for four out of seven test problems. It was not found superior to the last problem, the generated time series data. In regard to the extended algorithm, the results suggested that the Tabu Search solutions were significantly superior to those of BP solutions for all seven test functions.

Chapter 3

Overview of Methodology

The scope of this research is to analyze a time series forecasting system in which the forecasts, i.e. foreign exchange forecasts, are accrued from an NN model. The main focus is to minimize the forecast error through some attempts to determine the appropriate topology and weight matrices using various methods. Problematic issues related to the development of the best possible NN model for given inputs, i.e. selection of the NN training algorithm and design of the NN structures, are emphasized. Other problematic issues such as weight initialization, selection of input variables, and the amount of training data are not discussed. Random weight initialization was chosen for this research. All experiments based on pure time-delayed inputs.

There are three main things studied in this research: (1) optimization of NN topology; (2) searching for the best learning algorithm for the NN time series forecaster; (3) implementing NNs for multi-step ahead time series forecasting. Beside those, an attempt to implement an NN for multivariate time series forecasting is also incorporated.

3.1 Foreign Exchange Forecasting using Back Propagation Method

The NN model used in this research, depicted in Figure 3.1, is a three-layer feed forward NN with n inputs, p hidden nodes, and one output. The underlying forecasting procedure is a two-stage procedure, in which the first stage is to identify the model parameters, while the second stage uses those parameters in the training process to

generate forecasts and determine the adequacy of the model parameters in the test process. The flowcharts of the training process using BP learning is shown in Figure 3.2.

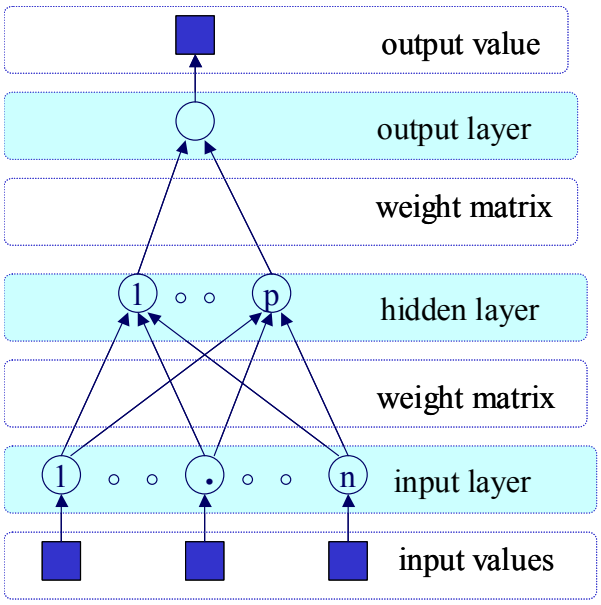


Figure 3.1 Three-layer Feed Forward Neural Network

The objective of this experiment is to study the effects of architecture and parameter values on its forecasting performance (i.e. Percentage of Correct Forecast Direction of Change (DIR), Mean Squared Error (MSE) and Mean of Absolute Percentage Error). The effect of modeling factors in Three-Layer Feed Forward Neural Networks are studied thoroughly to accomplish two goals: (1) determine the key modeling factors that should be considered in topology optimization, and (2) provide the forecasting benchmark for each time series under investigation by constructing good NNs for this specific implementation. To achieve these goals, a series of experiments was performed for an in-depth understanding of the various issues that have significant influence on the performance of NNs for time series prediction.

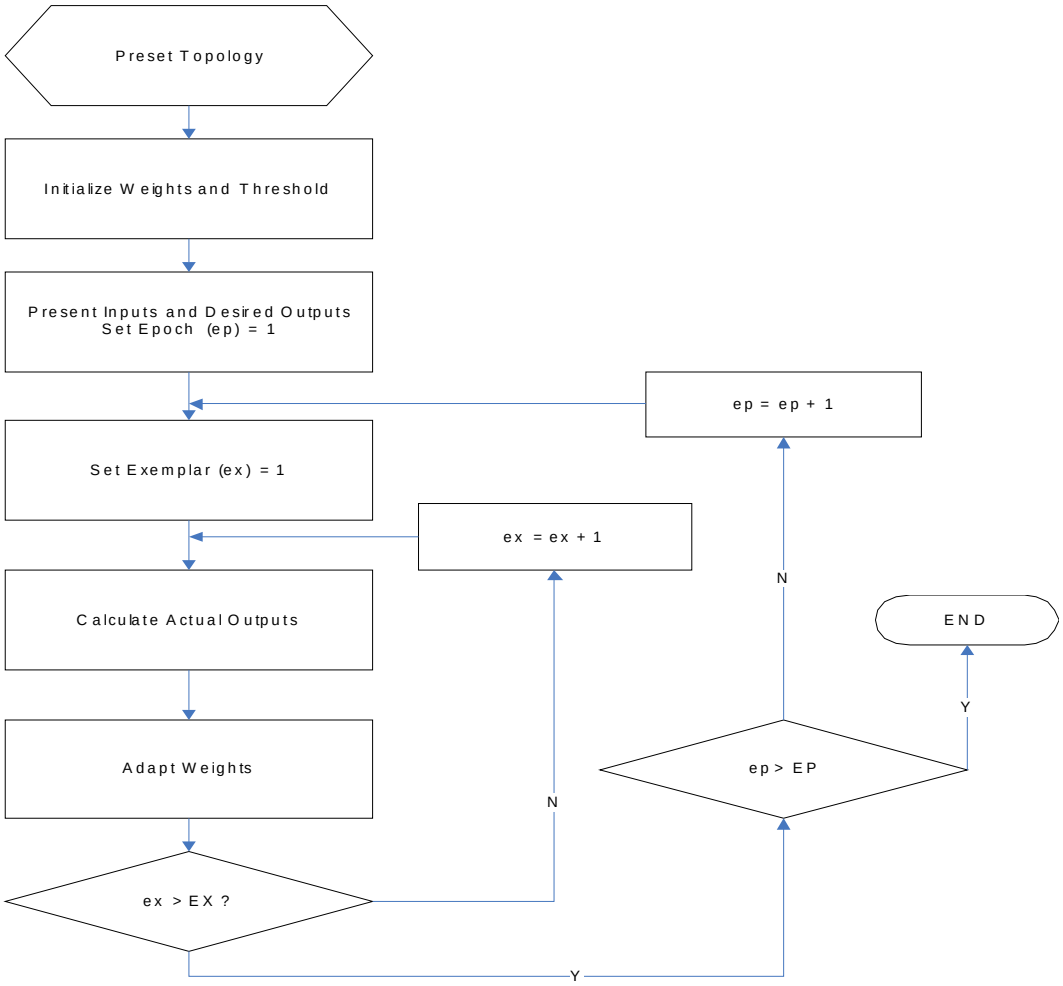


Figure 3.2 The Training Process

The effects of the following variables were investigated:

a) Size of input

b) Number of hidden neurons

c) Activation function of hidden nodes and output nodes

d) Learning rate

e) Momentum term

The experiments were conducted in two steps, involving two and four variables respectively. All experiments were applied for each data set in performing one-step ahead forecasting. To estimate the error term, three replications were used. In the first

experiment, the effects of the two most important factors (number of input nodes and hidden nodes) were studied while others were kept constant, i.e. learning rate=0.9, momentum term=0.02, activation function= tanh (Based on the values used by Yao, Li, and Tan for forex forecasting). Seven fixed levels of hidden nodes (3, 5, 6, 10, 12, 16, 20) and ten fixed levels of input nodes (1, 2, 3, 4, 5, 6, 7, 8, 9, 10) were examined. A total of 1050 data sets of one-step ahead forecasting were performed, each involving 1000 iterations.

The linear statistical model for this experiment is as follows:

$$y_{ijk} = \mu + \tau_i + \beta_j + (\tau\beta)_{ij} + \varepsilon_{ijk} \qquad (3\text{-}1)$$

where μ is the overall mean, τ_i is *i*th number of input node effect, β_j is *j*th number of hidden node effect, $(\tau\beta)_{ij}$ is the interaction effect of *i*th number of input nodes and *j*th number of hidden nodes and ε_{ijk} is a random error.

The best result found in the first experiment was the base for the second experiment, which was performed to study the effects of learning rate, momentum term, and activation function in hidden layer and output layer. A fractional factorial with 16 runs was employed for this purpose. The four factors were examined: (1) the activation function 1 (2 levels), (2) the activation function 2 (2 levels), (3) the learning rate (3 levels) and the momentum terms (3 levels). The array used is provided in Table 3.1.

The experiments were conducted using univariate time series with pure time delayed inputs. MATLAB® programming was used to facilitate the experiment. Basically, the MATLAB® program performs the following steps:

Table 3.1 The Experimental Array

runs	act fn 1	act fn 2	l rate	m term
1	T	T	0.1	0.1
2	S	T	0.1	0.3
3	T	S	0.1	0.9
4	S	S	0.1	0.3
5	T	T	0.3	0.9
6	S	T	0.3	0.3
7	T	S	0.3	0.1
8	S	S	0.3	0.3
9	T	T	0.3	0.3
10	S	T	0.3	0.9
11	T	S	0.3	0.3
12	S	S	0.3	0.1
13	T	T	0.9	0.3
14	S	T	0.9	0.1
15	T	S	0.9	0.3
16	S	S	0.9	0.9

1. The program takes the values of parameters inputted by the user (i.e. number of input nodes, number of hidden nodes, learning parameter, momentum term, activation function).
2. Input the chosen data using the 'switch' argument.
3. Transform the input data into [0 1] interval.
4. Map the input vector into the appropriate input data using sliding window techniques. Figure 3.3 provides an illustration of the technique employed for this assignment.
5. Initialize the weight based on the desired method: random, Nguyen Widrow.
6. Perform BP training with the desired activation function. The number of training patterns depends on the number of input nodes and type of forecast performed (one-step or multiple ahead forecasting), since here dynamic input data was used rather than static input data.

7. Transform back the values obtained; plot the values and obtained the statistical performance for the training and test sets.

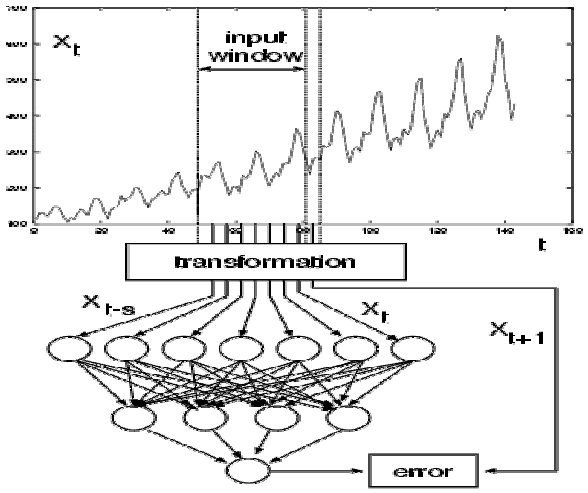


Figure 3.3 The Sliding Window Technique for Neural Network Time Series Forecaster

Based upon the experiment results for one-step ahead forecasting and the analysis performed (Analysis of Variance (ANOVA)), conclusions were drawn. The factors that gave significant difference were included in the topology optimization. The best results for each case were compared with the findings from previous research results, and the best were used as benchmarks for all future comparison purposes.

The accuracy of forecasting was used as the basis for selecting the appropriate method. It is widely accepted that no single accuracy measure can capture all the differences among various methods. Three performance measures were examined in this research: Mean Squared Error (MSE), Mean Absolute Percentage Error (MAPE), and The Percentage of true directional changes (DIR). The last performance measure was used because the case study is foreign exchange forecasting. The reason being that for

foreign exchange forecasting, measuring only the mean standard error of the NN may produce misleading evaluations of the NN capabilities, since even a very small error that incorrectly predicts the direction of change will result in a capital loss (Walczak, 2001). Instead or measuring the mean standard error of a forecast, many researchers argue that a better method for measuring the performance of NNs is to analyze the direction of change. The percentage of correct direction of change forecasts is therefore equivalent to the percentage of profitable trades enabled by the NN system.

3.2 Neural Networks Topology

Considering the importance of the topology to the NNs' performances, this research topic was explored intensively. This would include the optimization of NN architecture and the parameter values to obtain the best result for each data set. The choice of NN architectures was made based on experiments using BP as the learning algorithm.

The problem of choosing the number of hidden neurons and time-delay was formalized as an optimization problem in which the cost function was the network error calculated on a data set. Two metaheuristic were compared for this purpose: Genetic Algorithm and Modified Tabu Search. The topology identified in this stage along with the benchmarks, was used in the next stage to generate forecasts. The implementation is rated as good if the network topology suggested provides a better or at least the same result as the benchmark for the given problem.

The flowcharts of Genetic Algorithm and Modified Tabu Search for NN topology determination are given in Figure 3.4 and 3.5, respectively.

3.3 Neural Network Training

This research addresses one of the main drawbacks of NNs for time series forecasting. The goal is comparing the performance of a BP algorithm with a Genetic Algorithm and a Modified Tabu Search (TS) in an attempt to get an appropriate learning algorithm for a multi-layer feed forward NN for one-step ahead time series forecasting. In this experiment, GA and TS were employed to optimize the weight parameters of the three-layer feed forward NN so that the network mean squared error was minimized.

A rigorous comparison among BP, TS, and GA was proposed. Simulated Annealing was not examined, despite its superiority compared to BP. The reason was that in nearly all cases where TS has been compared with SA, the superiority of TS was quite apparent (Hertz and de Werra, 1990). BP has multiple user-determined parameters that may significantly impact the solution; thus, the exploration of the parameters was done before the comparisons took place. The various GA and TS models used in the literature were adapted and implemented to the chosen NNs architecture.

Since the network consists of n number of inputs, p number of hidden nodes, and o output nodes and the neuron thresholds, the genetic algorithm requires chromosomes with [p*(n+o+1) + ot] genes and real encoding was applied. The process of genetic

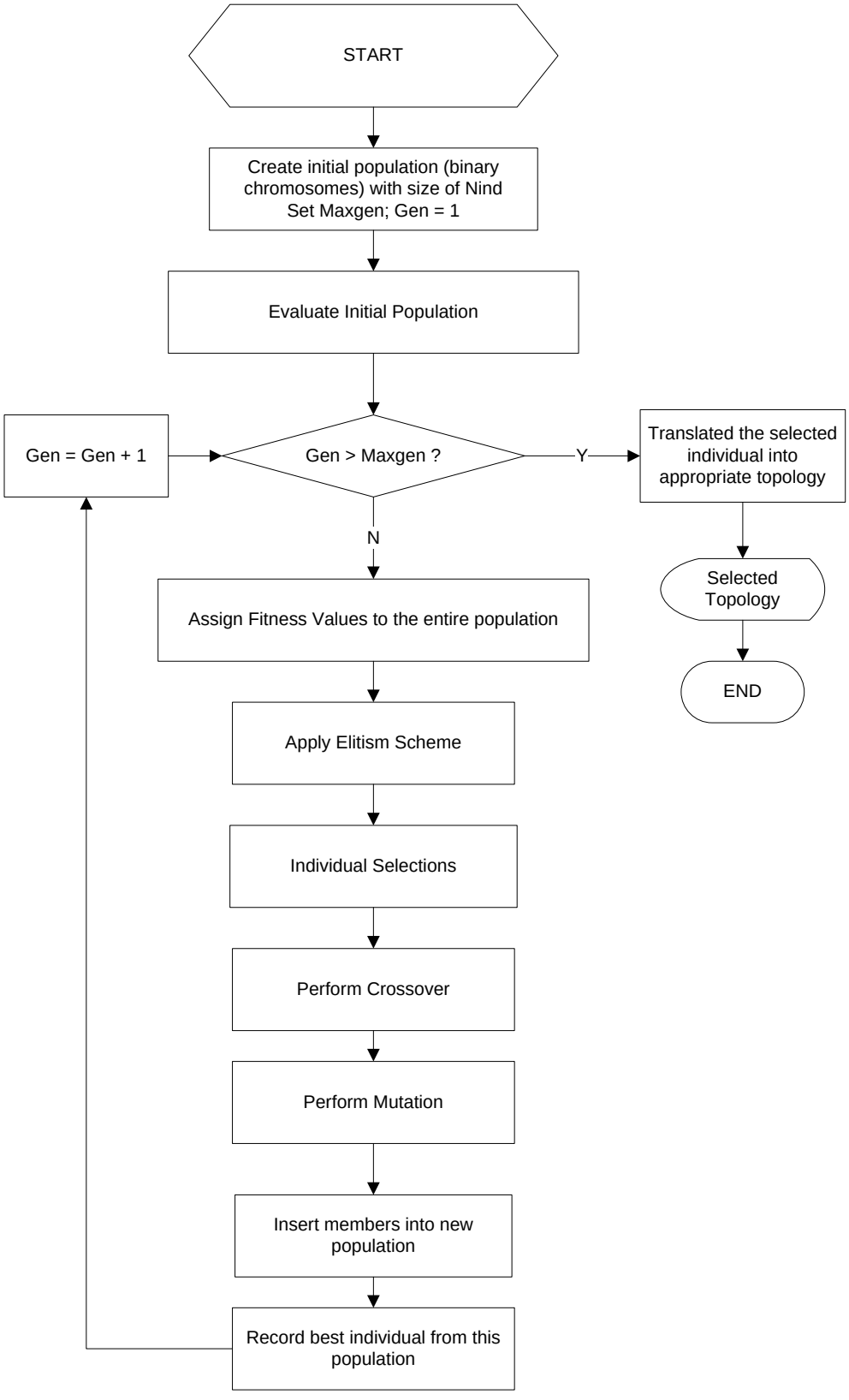


Figure 3.4 Genetic Algorithm for Topology Determination

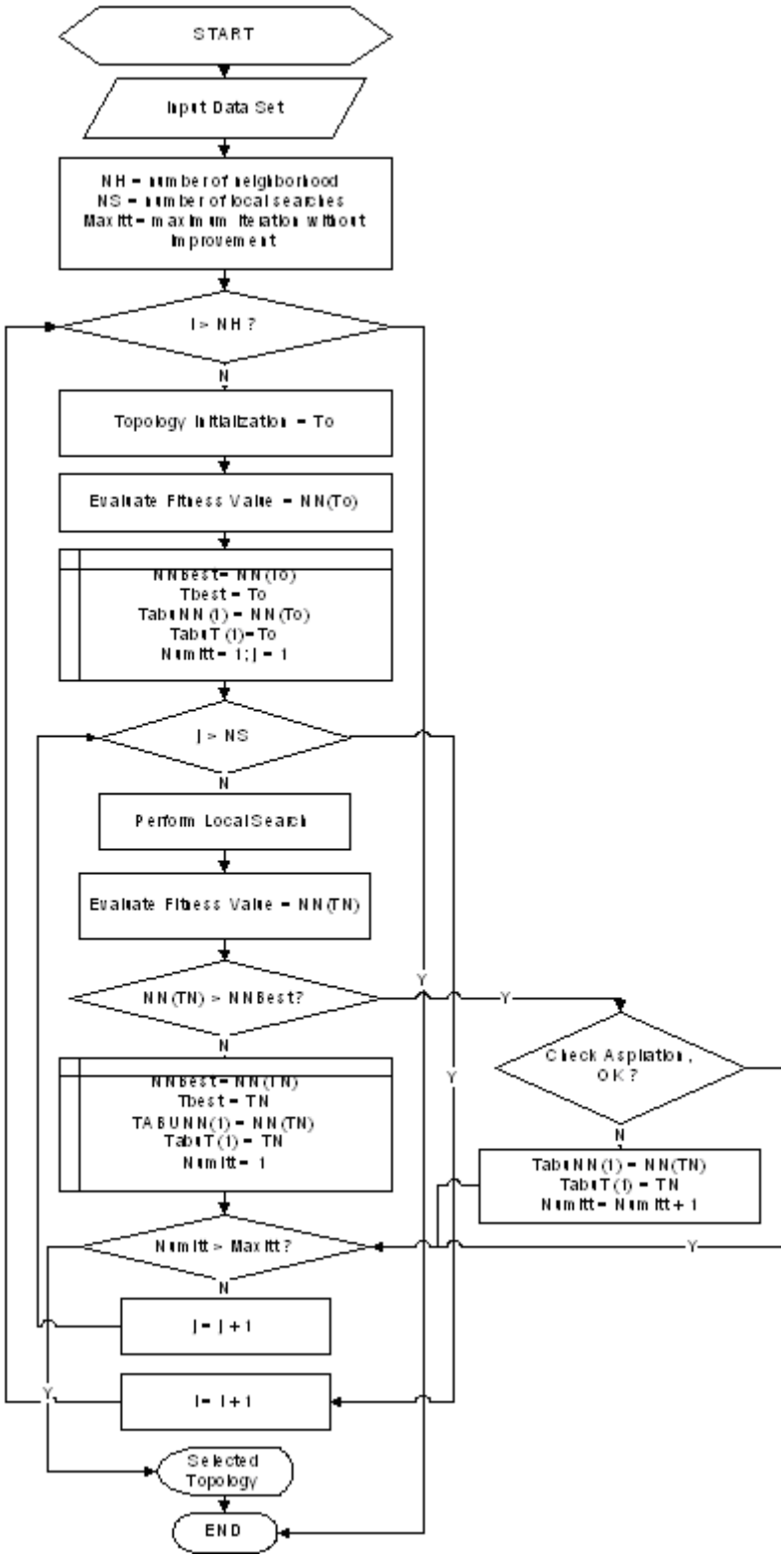


Figure 3.5 Modified Tabu Search for Topology Determination

evolution works on the neuron connections by applying two procedures: mutation and crossover. The GA for this purpose was constructed based on outlines provided by Dorsey et al. (1994) and Rooij et al. (1996). The flowchart of a Genetic Algorithm is in Figure 3.6. The Tabu Search Heuristic used is based on the one suggested by Sexton et al. (1998) and its flowchart is in Figure 3.7.

3.4 Computational Studies for Multi-Step Ahead Forecasting

In this phase, the best NN selected for two data sets (GBP/USD and EUR/USD) was used to conduct multi-step ahead forecasting. Two- to five- step ahead forecasting were investigated. The number of input nodes and the number of hidden nodes were specified the same as the ones chosen for one-step ahead forecasting. The quality of the forecasts obtained for these time series data was discussed.

3.5 Neural Networks for Multivariate Time Series Forecasting

Based on the results, a similar approach was applied for multivariate time series forecasting. The modifications were mainly on the inputs. Rather than using one time series, the inputs came from multi time series. Inputs initially considered in this study are the ones used by previous researchers to conduct multivariate time series models or the ones indicated in the literature, which include log of exchange rate index, three months

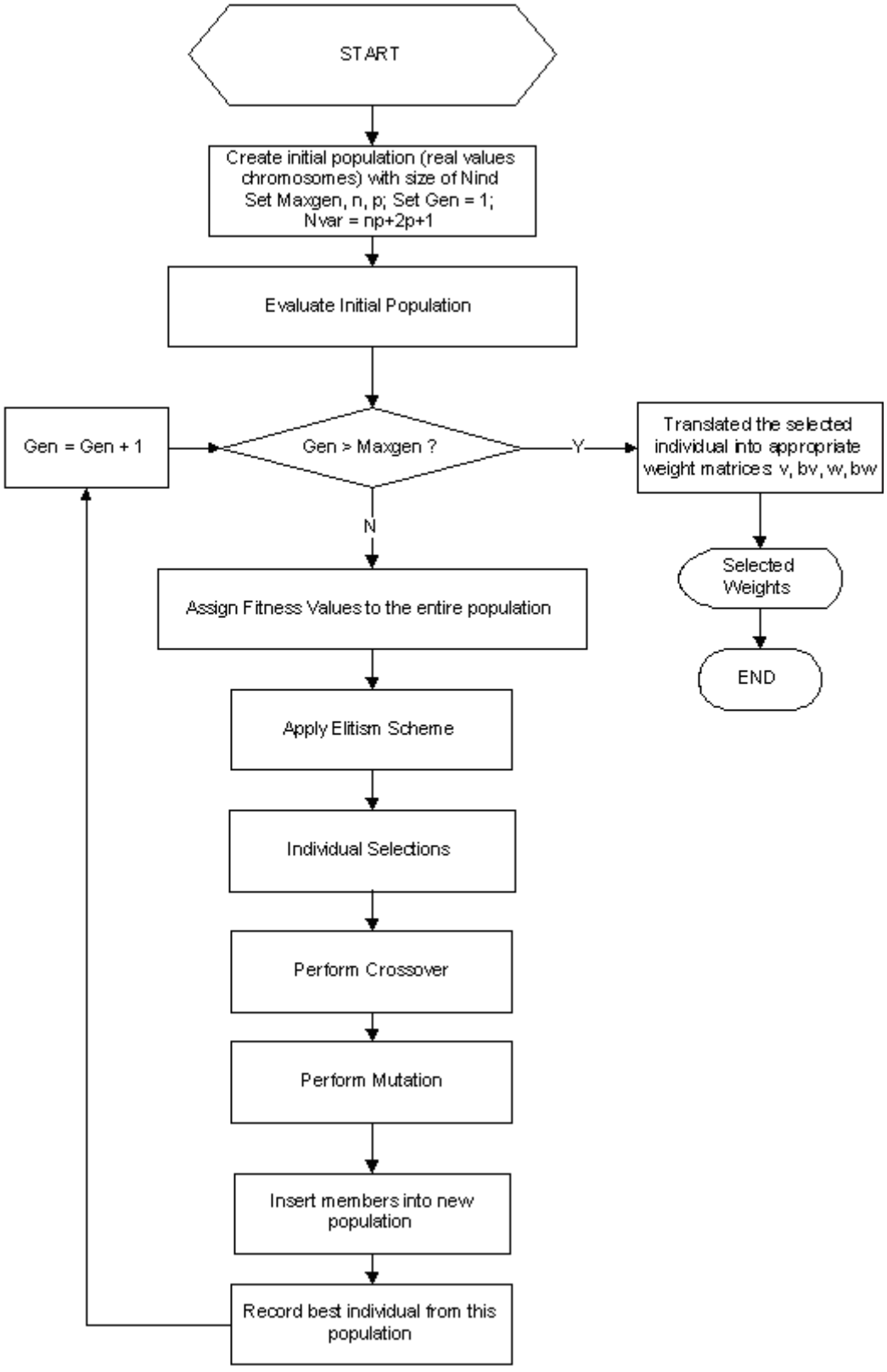


Figure 3.6 Genetic Algorithm for Neural Network Weights Training

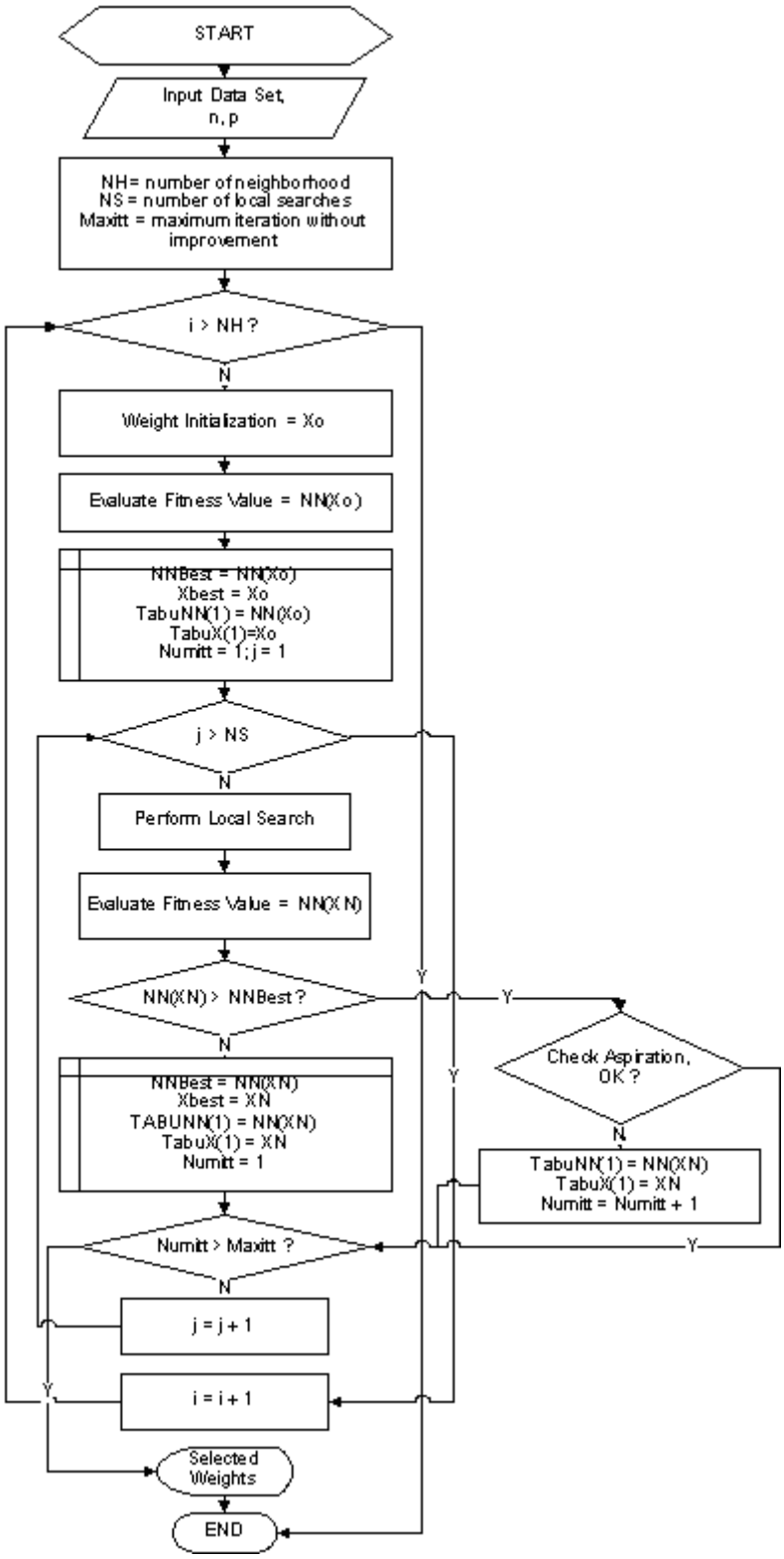


Figure 3.7 Modified Tabu Search for Neural Network Weights Training

forward rate, current account balance, quarterly relative interest rate, log of quarterly relative price (Hoque and Latif, 1993), inflation rate, flows of funds between two countries, export/import values of the two countries, and the exchange rate of other currencies.

Since the case studies are weekly closing data of foreign exchange rates, most of the above inputs were not applicable and or available except the log of exchange rate index and the exchange rate of other currencies. The quality of the forecasts obtained for EUR/USD using this approach is presented in this section.

3.6 The Data

The following time series data were used in this experiment:

(1). Weekly and monthly time series data of Euro/US$ (EUR /USD) from period 2000 – 2004

(2). Weekly and monthly time series data of Japanese Yen/US$ (JPY /USD) from period 1991 – 2004

(3). Weekly time series data of Swiss Franc /US$ (CHF/USD) from period 1991 – 2004

(4). Weekly time series data of British Pound sterling /US$ (GBP/USD) from period 1991 – 2004

(5). Weekly time series data of Australian$ /US$ (AUS/USD) from period 1991 – 2004

(6). Weekly and monthly interest rate of Japan from period 1997-2004

(7). Weekly and monthly interest rate of Euro from period 2000-2004

(8). Weekly and monthly interest rate of US from period 1997-2004

Interest rate data were obtained from *Economagic* (economagic.com). Historical exchange rates are available from some web sites in the form of daily time series data. The daily data for this research was obtained from the PACIFIC exchange rate services, which is maintained by Professor Werner Antweiler at The University of British Columbia, Vancouver BC, Canada. The data was cleaned by conducting the required preprocessing. The experiment used Friday closing prices as the inputs as well as the prediction target. In the event of Friday being a holiday, the most recently available closing price for the currency was used. The complete data are depicted in Appendix 1.

For each time series, the training set is required to be larger than the testing set. As a default setting for the system, six years data are used as the training set and the next one-year data are used as the testing set. Unless specified otherwise, the data employed were 1997-2003 data.

Chapter 4
Forecasts Using Back Propagation

4.1 The Back Propagation Model

The NN model used in this research is a three-layer feed forward NN with n input nodes and p hidden nodes. The model parameters, including the values of n and p, have to be identified prior to using the NN for time series forecasting. Unless otherwise specified, five inputs nodes, three hidden nodes, and tanh activation functions were employed. The learning rate was set at 0.9, while the momentum term was set at 0.02. Three years of training data (2000-2002) and one year of the test data (2003) were used for EUR/USD. Six years of training data (1997-2002) and one year of test data (2003) were used for other foreign exchange rates. The data statistics of weekly foreign exchange rates employed (four years of data for EUR/USD and seven years of data for others) are given in Table 4.1. The searches initialized using random weights, was stopped after 1,000 iterations.

Table 4.1 Data Statistics of Weekly Foreign Exchange Rate

Variable	Mean	Median	Max	Min	Std Dev
AUS$/US$	1.6465	1.6207	2.0629	1.2525	0.2014
EURO/US$	1.0384	1.0646	1.1929	0.8077	0.1016
FRANC/US$	1.5265	1.4983	1.8091	1.2562	0.1322
POUND/US$	0.6381	0.6241	0.7236	0.5657	0.0375
YEN/US$	120.7242	120.8200	146.3500	102.1400	9.1149

4.2 Experiment I

The objective of this experiment was to study the effects of the number of input and hidden nodes to the NN performances. The number of input nodes and hidden nodes

were set based on the corresponding unit experiment, and other parameters were kept constant. For each currency, 70 unit experiments with 3 replications were performed. For the five currencies, 1050 data points were collected and analyzed.

The Analysis of Variance (ANOVA) performed for JPY/USD is depicted in Table 4.2 thru Table 4.4. The ANOVA suggests that there is significant difference (based on $\alpha = 0.05$) either for MSE, MAPE, or percentage true directional changes on the number of input nodes, the number of hidden nodes, and their interactions.

Table 4.2 Analysis of Variance for MSE

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	165418.2	6	27569.7	**9.872754**	4.53E-09	2.163929
H. Nodes	69391.72	9	7710.191	**2.761032**	0.00529	1.947349
Interaction	232764.2	54	4310.448	**1.543578**	0.022709	1.429385
Within	390950.5	140	2792.504			
Total	858524.6	209				

Table 4.3 Analysis of Variance for MAPE

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	320.9708	6	53.49513	**15.88311**	7.18E-14	2.163929
H. Nodes	100.0885	9	11.12095	**3.301894**	0.001109	1.947349
Interaction	322.4482	54	5.971264	**1.772914**	0.004034	1.429385
Within	471.5271	140	3.36805			
Total	1215.035	209				

Table 4.4 Analysis of Variance for Percentage of True Directional Changes

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	752.1187	6	125.3531	**4.224177**	0.000613	2.163929
H. Nodes	1068.505	9	118.7228	**4.000748**	0.000143	1.947349
Interaction	3222.053	54	59.66764	**2.010693**	0.000587	1.429385
Within	4154.522	140	29.67516			
Total	9197.199	209				

The same analysis was performed for other currencies. The results show that there are significant effects on the number of input nodes to the NN performances or the interaction between the number of input nodes and hidden nodes. The complete analyses are given in Appendix 2.

The significance of the number of input nodes are as expected. As mentioned by Davey et al. (2000) and Zhang et al. (2001), the number of input nodes is the most important issue for NNs as time series forecasters, since it corresponds to the number of lagged observations used to discover the underlying patterns and / or autocorrelation structures of time series. It has a much stronger effect than the number of hidden nodes.

Besides that, there is an interaction between the number of hidden nodes and number of input nodes. This suggests that the effect of the number of hidden nodes must be considered, since it will affect the result through the interaction effect with the number of input nodes.

4.3 Experiment 2

The objective of this experiment was to study the effects of activation function in hidden nodes, activation function in output nodes, learning rate and momentum term. For this experiment the number of input nodes and hidden nodes were pre-set (3-5, 8-3, 3-3, 2-3, and 2-12 respectively for AUS/USD, EUR/USD, CHF/USD, GBP/USD, and JPY/USD), while the values of other parameters were set correspondingly to the unit experiment. Two 2-level factors (activation function 1 and activation function 2) and two 3-level factors (learning rates, momentum term) were examined using fractional factorials with 16 run. The design is given in Table 4.5 and the design generators used were E=ABC and F=BCD. The defining relations were I=ABCE=BCDF=ADEF. The

design is a resolution IV design. In this design, every main effect was aliased by three factor and five factor interactions, whereas two factor interactions were aliased with each other and with higher order interactions. Hence, if three and higher factor interactions are negligible, this design gives a clear estimate of the main effects. Analysis of the effects, as well as the percent contribution of each factor, were analyzed using Design Expert® by StatEase.

The linear combinations of the observations used to estimate the main effects of the four factors are as follows:

$l_A = 1/8^* (-r_1 + r_2 - r_3 + r_4 - r_5 + r_6 - r_7 + r_8 - r_9 + r_{10} - r_{11} + r_{12} - r_{13} + r_{14} - r_{15} + r_{16})$

$l_B = 1/8^* (-r_1 - r_2 + r_3 + r_4 - r_5 - r_6 + r_7 + r_8 - r_9 - r_{10} + r_{11} + r_{12} - r_{13} - r_{14} + r_{15} + r_{16})$

$l_C = 1/8^* (-r_1 - r_2 - r_3 - r_4 + r_5 + r_6 + r_7 + r_8 - r_9 - r_{10} - r_{11} - r_{12} + r_{13} + r_{14} + r_{15} + r_{16})$

$l_D = 1/8^* (-r_1 - r_2 - r_3 - r_4 - r_5 - r_6 - r_7 - r_8 + r_9 + r_{10} + r_{11} + r_{12} + r_{13} + r_{14} + r_{15} + r_{16})$

$l_{learning\_rate} = ½^* (l_C + l_D)$

$l_E = 1/8^* (-r_1 + r_2 + r_3 - r_4 + r_5 - r_6 - r_7 + r_8 - r_9 + r_{10} + r_{11} - r_{12} + r_{13} - r_{14} - r_{15} + r_{16})$

$l_F = 1/8^* (-r_1 - r_2 + r_3 + r_4 + r_5 + r_6 - r_7 - r_8 + r_9 + r_{10} - r_{11} - r_{12} - r_{13} - r_{14} + r_{15} + r_{16})$

$l_{momentum\_term} = ½^* (l_E + l_F)$

Table 4.5 The Experiment II Design

runs	act fn 1	act fn 2	l rate	m term	A	B	C	D	=ABCE	=BCDF
1	T	T	0.1	0.1	-	-	-	-	-	-
2	S	T	0.1	0.3	+	-	-	-	+	-
3	T	S	0.1	0.9	-	+	-	-	+	+
4	S	S	0.1	0.3	+	+	-	-	-	+
5	T	T	0.3	0.9	-	-	+	-	+	+
6	S	T	0.3	0.3	+	-	+	-	-	+
7	T	S	0.3	0.1	-	+	+	-	-	-
8	S	S	0.3	0.3	+	+	+	-	+	-
9	T	T	0.3	0.3	-	-	-	+	-	+
10	S	T	0.3	0.9	+	-	-	+	+	+
11	T	S	0.3	0.3	-	+	-	+	+	-
12	S	S	0.3	0.1	+	+	-	+	-	-
13	T	T	0.9	0.3	-	-	+	+	+	-
14	S	T	0.9	0.1	+	-	+	+	-	-
15	T	S	0.9	0.3	-	+	+	+	-	+
16	S	S	0.9	0.9	+	+	+	+	+	+

The experiment results for JPY/USD are showed in Table 4.6. The results for other currencies are available in Appendix 3.

Table 4.6 Experiment II for JPY/USD

Runs	MSE	MAPE	Dir
1	15.94	2.75	43.58
2	1.80	0.95	48.71
3	7.79	2.06	41.02
4	22.92	3.97	48.71
5	22.21	3.85	51.28
6	1.97	0.99	53.84
7	6.00	1.82	48.71
8	2.18	1.03	43.58
9	21.69	3.85	53.84
10	2.28	1.09	56.41
11	6.36	1.88	51.28
12	2.15	1.02	46.10
13	14.00	2.91	53.80
14	19.20	3.63	64.10
15	20.60	3.74	51.20
16	19.60	3.64	51.20

The linear combinations (effects) as well as the percent contribution of each factor in this experiment are given as follows (Table 4.7).

Table 4.7 Effects and Percent Contribution for JPY/USD

	Effect			**% Contribution**		
Activation Function 1	-5.31	-0.82	2.24	10.29	12.06	4.28
Activation Function 2	-1.44	-0.11	-5.47	0.75	0.21	**25.47**
Learning rate	3.12	0.52	4.79	3.55	4.96	**20.88**
Momentum term	1.08	0.18	-0.64	10.88	9.99	1.48
Statistical Performance	MSE	MAPE	Dir	MSE	MAPE	Dir

Based on Partial ANOVA, only two factors are significantly different: activation function output node and learning rate. While sigmoid and tanh activation functions

worked well for this application, the hyperbolic tanh performed better. The results also indicated that learning rate is more sensitive than the momentum term. A higher learning rate is more appropriate for this problem.

The complete results for all five foreign exchange rates proved that the above conclusions are problem dependent. Percent contribution as well as partial ANOVA for GBP/USD, for example, demonstrated that the significant factor was the activation function hidden node. In general, the hyperbolic activation function performed better. In all cases, three different momentum term values (0.1, 0.3 or 0.9) did not have a significant effect to the NN performance.

Learning rate should not be excluded in the topology optimization in determining the correct parameters for different problems. The hyperbolic tangent activation function seems more appropriate compared to sigmoid function. It should be noted that learning rate and momentum term are only applicable for BP learning. Hence, if one uses either Genetic Algorithm or Tabu Search as the learning method for NN for time series forecasting, only two out of the four parameters above must be included in the topology modeling.

4.4 Obtaining the Benchmarks

A set of parameters for NN forecasting for each currency was determined by comparing all the best results from the experiments using BP, as well as the ones given in the literature, and was used as a benchmark for further analysis. Whenever available, the variances of the forecasting performance, i.e. the variance of MSE, MAPE, and DIR for a certain problem are provided to informatively describe the risk associated with the forecast parameters.

Considering the noise nature of BP evaluations, the forecasting performance, complete with the corresponding variance, is preferred. The selected parameter values, which are used for the benchmark, are given in Table 4.8, while the complete comparisons are listed in Table 4.9.

Table 4.8 Parameter Values For Comparing AUS, EUR, CHF, GBP, JPY with USD

	AUS/USD	**EUR/USD**	**CHF/USD**
Number of Input Nodes	3	8	3
Number of Hidden Nodes	5	3	3
Learning Rate	0.9	0.9	0.9
Momentum Term	0.02	0.02	0.02
Activation Functions	tanh	tanh	tanh
Performances	MSE: [0.0017, 0.0006] MAPE: [2.64, 0.42] Dir: [64.00, 5.29]	MSE: [0.0008, 0.0010] MAPE: [3.15, 1.69] Dir: [58.13, 6.15]	MSE: [0.0006, 0.0044] MAPE: [2.17, 0.46] Dir: [58.33, 4.17]

Note: Dir= directional change statistics; the performance values are given as [mean, variance]

	GBP/USD	**JPY/USD**
Number of Input Nodes	2	2
Number of Hidden Nodes	3	12
Learning Rate	0.9	0.9
Momentum Term	0.02	0.02
Activation Functions	tanh	tanh
Performances	MSE: [0.0005, 0.0003] MAPE: [3.02, 0.88] Dir: [62.58, 1.18]	MSE: [57.26, 11.79] MAPE: [4.53, 1.05] Dir: [61.90, 1.18]

Table 4.9 Comparison of Research Studies on Forex Time Series Forecasting

AUS/USD

input nodes	hidden nodes	l rate	m term	act function	MSE	MAPE	Dir	Data Freq	Source:
					(average, standard deviation)				
3	3	0.9	0.02	tanh	[0.009, 0.0009]	[3.51, 0.97]	[63.00, 8.19]	weekly	Experiment I
3	5	0.9	0.02	tanh	[0.0017, 0.0006]	[2.64, 0.42]	[64.00, 5.29]	weekly	Experiment I
3	5	0.1	0.3	sig/tanh	[0.001]	[1.99]	[69.38]	weekly	Experiment II
5	3	0.9	-	-	-	-	[55.00]	weekly	Yao et al. 1996, Yao and Tan, 2000
5	ARIMA	-	-	-	-	-	[54.32]	weekly	Yao et al. 1996, Yao and Tan, 2000

EUR/USD

input nodes	hidden nodes	l rate	m term	act function	MSE	MAPE	Dir	Data Freq	Source:
					(average, standard deviation)				
4	3	0.9	0.02	tanh	[0.0020, 0.0032]	[4.48, 0.22]	[57.43, 0.02]	weekly	Experiment I
8	3	0.9	0.02	tanh	[0.0008, 0.0010]	[3.15, 1.69]	[58.13, 6.15]	weekly	Experiment I
8	3	0.3	0.3	tanh	[0.001]	[3.22]	[58.13]	weekly	Experiment II

CHF/USD

input nodes	hidden nodes	l rate	m term	act function	MSE	MAPE	Dir	Data Freq	Source:
					(average, standard deviation)				
3	3	0.9	0.02	tanh	[0.0006, 0.0004]	[2.17, 0.46]	[58.33, 4.17]	weekly	Experiment I
8	12	0.9	0.02	tanh	[0.0060, 0.0044]	[5.10, 2.06]	[58.25, 4.48]	weekly	Experiment I
3	3	0.3	0.9	tanh	[0.001]	[2.30]	[60.41]	weekly	Experiment II
3	6	-	-	-	-	-	[57.60]	Daily (4 yrs)	Walczak, 2001
5	3	0.9	-	-	-	-	[56.00]	weekly	Yao et al. 1996, Yao and Tan, 2000
5	ARIMA	-	-	-	-	-	[55.86]	weekly	Yao et al. 1996, Yao and Tan, 2000

Table 4.9 Comparison of Research Studies on Forex Time Series Forecasting (continued)

GBP/USD

input nodes	hidden nodes	l rate	m term	act function	MSE	MAPE	Dir	Data Freq	Source:
					[average, standard deviation if available]				
2	3	0.9	0.02	tanh	[0.0005, 0.0003]	[3.02, 0.88]	[62.58, 1.18]	weekly	Experiment I
2	6	0.9	0.02	tanh	[0.0005, 0.0003]	[2.73, 0.97]	[61.90, 2.36]	weekly	Experiment I
2	3	0.9	0.3	tanh	[0.0001]	[1.36]	[61.22]	weekly	Experiment II
3	6	-	-	-	-	-	[62.40]	daily (2 yrs)	Walczak, 2001
6	3	0.9	-	-	-	-	[54.74]	weekly	Yao et al. 1996, Yao and Tan, 2000
6	ARIMA	-	-	-	-	-	[53.41]	weekly	Yao et al. 1996, Yao and Tan, 2000

JPY/USD

input nodes	hidden nodes	l rate	m term	act function	MSE	MAPE	Dir	Data Freq	Source:
					[average, standard deviation if available]				
2	12	0.9	0.02	tanh	[57.26, 11.79]	[4.53, 1.05]	[61.90, 1.18]	weekly	Experiment I
4	10	0.9	0.02	tanh	[33.65, 3.74]	[5.27, 1.29]	[58.15, 2.46]	weekly	Experiment I
2	12	0.9	0.1	sig/tanh	[19.2]	[3.63]	[64.1]	weekly	Experiment II
5	3	0.9	-	-	-	-	[53.4]	weekly	Yao et al. 1996, Yao and Tan, 2000
5	ARIMA	-	-	-	-	-	[44.32]	weekly	Yao et al. 1996, Yao and Tan, 2000
2	6	-	-	-	-	-	[59.20]	daily (2 yrs)	Walczak, 2001
3	2	-	-	-	-	-	[60.59]	weekly	Setyawati et al., 2003

Chapter 5

Neural Network Topology Determination

The topology used has a very significant influence to the NNs performances. The NN architecture and its parameter values should be set up appropriately to obtain a good forecast for each different problem.

The common method used to determine the NN topology is trial and error, but this method is unstructured in nature and tedious. Design of the experiment is an alternate method for finding out the appropriate topology for time series forecasting using NNs. However, there are two problems encountered when applying this method. First, to be able to apply this method using a reasonable number of experimental units, one should have preliminary information about the possible working architectures for a certain application. Secondly, this method is not a trivial one as knowledge of statistics is required. It is not possible for someone with no prior knowledge of NNs for a certain application and no experimental design skill to apply this method. Therefore, a simpler application method is necessary.

In this research phase, the problem of choosing the number of hidden neurons and time-delayed was formalized as an optimization problem in which the cost function was the network error of the data set. The choice of NN architectures was made based on experiments using back propagation as the learning algorithm.

Two possible methods were examined, i.e. Genetic Algorithm (GA) and Modified Tabu Search (TS) Algorithm. Having the GA and TS tools for a topology search allows

the user to determine the correct topology for time series forecasting using an NN without any difficulty.

It should be noted that GA or TS global search procedures are usually computationally expensive. It is, however, beneficial to introduce this method when there is little prior knowledge available, and the performance of the NNs is required to be high. Trial and error is very ineffective in such circumstances.

5.1 Genetic Algorithms for Neural Network Topology Determination

The Genetic Algorithm used was written in the MATLAB® environment. Using this method, an initial population consisting of various network architectures was represented and was mapped directly into a bit-string genotype. For each architecture, six variables were examined: the number of input nodes, the number of output nodes, activation function for hidden nodes, activation function for output nodes, learning rate, and momentum term. The genotype and phenotype representations are illustrated in Figure 5.1. Genetic operations were then used to act upon populations of these genotypes to produce a higher fitness level.

Gray binary encoding was chosen for its speed. As has been known, Gray coding uses one-bit changes between subsequent numbers, which is not the case in pure binary coding. Changing for 7 (0111) to 8 (1000) using the pure binary coding, for example, requires 4 bits to be changed simultaneously.

For each of the five foreign exchanges examined, a Genetic Algorithm with back propagation-based fitness evaluation was employed to obtain the NNs topology. A BP-based objective value evaluation was employed. This objective value evaluation is a noisy fitness evaluation with weight initialization as the major source of noise. To reduce

the noisy nature of BP, each architecture was trained two times using different random initial weights. The average value was used for the decision-making.

A ranking mechanism was used that ranks the chromosomes in order of fitness. For selection, the standard roulette selection mechanism was used. In every generation all chromosomes were replaced, except for the best performing chromosome, which was retained and inserted into the next generation without change (elitism scheme). The crossover operator employed was two points crossover as illustrated in Figure 5.2 with a crossover rate of 0.7, and the mutation rate was set at 0.01.

The normal parameters, including the population size, were set so that the time required for the topology search was approximately the same as the time required to conduct experiment 1. Hence, the two methods are comparable. Three different configurations were tried involving combinations of numbers of generations and population sizes of 50, 20, 10 and 4, 10, 20, respectively. The pseudo code of the GA is given in Appendix 4, while the results of the best configuration (based on population size of 10 and 20 generations) are depicted in Table 5.2. The performances (MSE, MAPE, DIR) were based on an average of five replications.

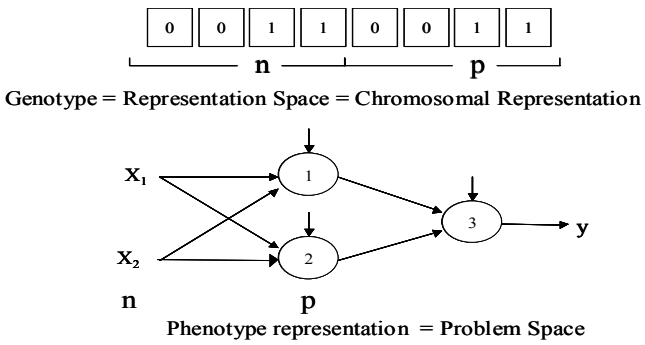


Figure 5.1 The Genotype and Phenotype Representations

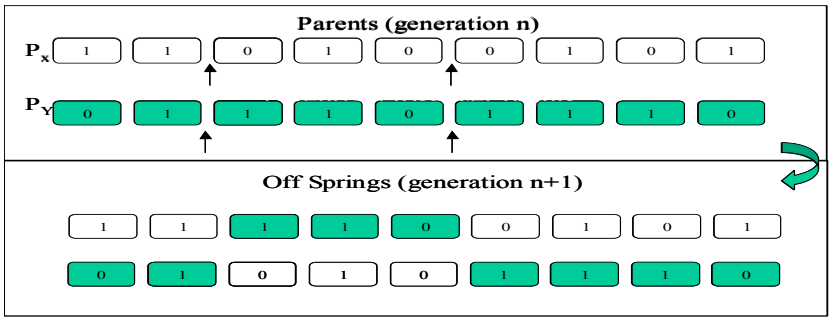


Figure 5.2 Two Point Crossover

Table 5.1 Topology Suggested by GA

AUS/USD	TOPOLOGY	MSE	MAPE	DIR
GA_Topology	8-1-0.59,0.02,1,1	0.0009	1.6810	61.36
Benchmark	3-5-0.90,0.02,1,1	0.0017	2.6400	64.00
EUR/USD	**TOPOLOGY**	**MSE**	**MAPE**	**DIR**
GA_Topology	3-3-1,0.54,1,1	0.0008	2.7390	61.25
Benchmark	8-3-0.90,0.02,1,1	0.0008	3.1467	58.13
CHF/USD	**TOPOLOGY**	**MSE**	**MAPE**	**DIR**
GA_Topology	3-4-0.96,0.50,1,1	0.0007	2.0620	60.00
Benchmark	3-3-0.90,0.02,1,1	0.0006	2.1700	58.33
GBP/USD	**TOPOLOGY**	**MSE**	**MAPE**	**DIR**
GA_Topology	2-3-0.30, 0.25, 1,1	0.0001	1.3626	61.22
Benchmark	2-3-0.90,0.02,1,1	0.0005	3.0167	62.58
JPY/USD	**TOPOLOGY**	**MSE**	**MAPE**	**DIR**
GA_Topology	3-1-0.88,0.02,1,1	3.9820	1.4710	54.17
Benchmark	2-12-0.90,0.02,1,1	57.2633	4.5333	61.90

The above results indicated that the topologies suggested by Genetic Algorithm were not as good as expected. They are indeed better in terms of MAPE for all cases, and generally perform better in terms of MSE, but failed to show superiority in terms of DIR. However, except for JPY/USD, all DIR were above 60%. These values are

acceptable in practice. As is widely known, different criteria exist for academics and industry. In industry, 60% accuracy is typically aimed for (Yao and Tan, 2000).

It is apparent that there is a completely opposite choice suggested for JPY/USD. The benchmark provides good performance in DIR but very low performance in terms of MSE and MAPE. The GA suggested a topology that guarantees a very good performance in terms of MSE and MAPE, but somewhat low performance in terms of DIR. Undoubtedly, the JPY/USD exchange rate is the most difficult one to forecast. The 2003 data, for example, has 27 turning points, as opposed to 16-22 turning points for other data. That implies that more than 50% turning points in one year should be predicted. The difficulty of this exchange rate forecasting is also reported by Yao and Tan (2000). Consequently; more rigorous effort should be devoted for this foreign exchange forecasting.

5.2 Modified Tabu Search for Neural Network Topology Determination

A Modified Tabu Search (TS) algorithm was applied for this purpose. The pseudo code of the algorithm for NN topology search is given in Appendix 4. The maximum number of iterations = 10, with no improvement over the best solution so far, was used to terminate the search process. The number of neighborhood searches (NH) and the number of local searches (NS) were set to ensure the time to complete the search was comparable with that for Experiment 1. From the three configurations tested, i.e. NH = 50, 20, 10 and NS = 4, 10, 10, respectively, the second configuration (NH = 20 and NS = 10) was chosen for further discussion. The size of the Tabu List was set equal to 100. An aspiration level condition was included to allow for overriding the rejection of a new

solution that meets the Tabu Criterion if the new solution is better than the best solution so far. The best configuration was used for comparison purposes.

For each architecture, six variables were examined: the number of input nodes, the number of output nodes, activation function for hidden nodes and output nodes, learning rate, and momentum term. As implemented for this problem, the neighborhoods were simply randomly drawn points, which include random integers for the number of input nodes and hidden nodes, random numbers from 0 to 1 for learning rate, and value of 0 or 1 for activation functions. The local searches were conducted as random searches within the neighborhood. The neighborhood was formulated as follows:

1. For the number of hidden nodes, the neighborhood is restricted within 2 numbers from the original value.
2. For the learning rate and momentum term, the neighborhood is region-restricted within 0.1 from the original values.

A BP-based objective value evaluation was employed. Each architecture was trained two times using different random initial weights. The average value was used for the decision making. The algorithm described above was implemented in the MATLAB® environment. The pseudo code of the modified Tabu Search used for the NN topology search is given in Appendix 4. The chosen topologies for each currency using this method are given in Table 5.2.

Table 5.2 Topology Suggested by Modified Tabu Search

AUS/USD	TOPOLOGY	MSE	MAPE	DIR
TS	9-1-0.90,0.02,1,1	0.0011	1.8600	60.46
benchmark	3-5-0.9,0.02,1,1	0.0017	2.6400	64.00
EUR/USD	**TOPOLOGY**	**MSE**	**MAPE**	**DIR**
TS	9-4-0.90,0.25,1,1	0.0031	5.3006	60.47
benchmark	8-3-0.9,0.02,1,1	0.0008	3.1467	58.13
CHF/USD	**TOPOLOGY**	**MSE**	**MAPE**	**DIR**
TS	8-4-0.97,0.31,1,1	0.0029	3.3434	61.16
benchmark	3-3-0.9,0.02,1,1	0.0006	2.1700	58.33
GBP/USD	**TOPOLOGY**	**MSE**	**MAPE**	**DIR**
TS	3-3-0.29,0.32,1,1	0.0001	1.1194	60.41
benchmark	2-3-0.9-0.02,1,1	0.0005	3.0167	62.58
JPY/USD	**TOPOLOGY**	**MSE**	**MAPE**	**DIR**
TS	5-3-0.27,0.89,1,1	5.9705	1.6588	53.80
benchmark	2-12-0.9,0.02,1,1	57.2633	4.5333	61.90

Again, the above table indicated that the topologies suggested by Modified Tabu Search were not as good as expected. The results are worse than those for GA. Compared to benchmarks, they are only better in terms of MSE and MAPE for three cases. TS is also failed to show superiority in terms of DIR. However, except for JPY/USD, all DIR are above 60%. As discussed earlier, these values are acceptable in practice.

Similar to the problem encountered in topology determination using GA, a completely opposite choice was suggested for JPY/USD. The benchmark provides good performance in DIR, but TS suggested a topology that guarantees a very good performance in terms of MSE and MAPE, with a somewhat lower performance in terms of DIR. This phenomenon emphasizes the need for a more thorough experiment for JPY/USD.

Table 5.3 Comparisons of MSE, MAPE, and DIR Performances

M S E

Topology Determination	AUS/USD	EUR/USD	CHF/USD	GBP/USD	JPY/USD
Benchmark	0.0017	0.0008	0.0006	0.0005	57.2633
GA	0.0009	0.0008	0.0007	0.0001	3.9820
TS	0.0011	0.0031	0.0029	0.0001	5.9705

M A P E

Topology Determination	AUS/USD	EUR/USD	CHF/USD	GBP/USD	JPY/USD
Benchmark	2.6400	3.1467	2.1700	3.0167	4.5333
GA	1.6810	2.7390	2.0620	1.3626	1.4710
TS	1.8600	5.3006	3.3434	1.1194	1.6588

D I R

Topology Determination	AUS/USD	EUR/USD	CHF/USD	GBP/USD	JPY/USD
Benchmark	64.00	58.13	58.33	62.58	61.90
GA	61.36	61.25	60.00	61.22	54.17
TS	60.46	60.47	61.16	60.41	53.80

The comparison performance of topology suggested using GA, TS, and benchmark is given in Table 5.3. The results from GA or TS for topology determination show that neither GA nor TS guarantee better results than the benchmarks, especially in terms of DIR. Compared to benchmark, both GA and TS only provide better performance for two cases. Furthermore, GA performs better than the benchmark for all three performances measured only for one case, i.e. EUR/USD, while TS provides none.

GA-based topology did result in the lowest MAPE for four out of five cases, and the lowest MSE for three out of five cases. The only benchmark to yield the lowest MSE was CHF/USD, while none of the benchmarks result in lowest MAPE.

For that reason, if a range of possible input nodes and hidden nodes are known, design of the experiment is recommended. However, if there is no prior knowledge of

the problem, or the user has no statistical background and would like to get relatively good topology automatically, a GA or TS search for topology determination is better.

By comparing the performance of GA-based and TS-based topologies for the five foreign exchanges, it can be concluded that in general the GA-based topology performs better. GA generally prevails over TS in terms of MSE, MAPE, and DIR. Only once out of five cases GA-based topology is defeated by TS-based topology either in MSE, MAPE, or in terms of DIR. For that reason, GA is recommended for topology determination.

Chapter 6

Neural Network Weight Determination

6.1 Training with Back Propagation

The BP network is based on a multi-layer feed forward topology with supervised learning. The network is fully connected with every node in the lower layer linked to every node in the next higher layer. These linkages are attached with weight values. The learning of BP network is actually an error minimization procedure. A BP optimized NN learns by using the generalized delta rule. A random initialization weight is a common approach to initialize the search. The weights are changed according to an error function, which compares the NN outputs with the targets. The direction to change the weights is determined based on the negative of the gradient with respect to the weights. The objective (error) function must be differentiable. The error function applied is the sum of squared errors.

BP is a local search algorithm (Shang and Wah, 1996), and tends to become trapped in local optima. If the initial weights are located on local grated, the algorithm will likely become trapped at a local optimum. Since obtaining an optimal solution is the goal of NN training, a global search technique seems more suitable for this difficult nonlinear optimization problem (Shang and Wah, 1996).

The BP with momentum term was used in this study. The program was written in the Mat Lab® environment. For each of five currencies, two NN architectures were used for comparison purposes. The parameter settings are given in Table 6.1. Each configuration was trained for 1000 iterations and random initialization was applied. The

trained networks were then used to generate one-step ahead forecasts for the next one-year period. Each replication differed only in the random initialization of the network.

It is important to note that the average time to perform GA or TS training using the same number of iterations is twice the one to perform BP training. To guarantee the fair comparisons, the better of two subsequent BP trainings was used for performance evaluation.

Table 6.1 Parameter Settings for Each Currency

Code	Topology
AUS[1]	3-5-1, tanh, tanh, 0.9, 0.02
AUS[2]	3-3-1, tanh, tanh, 0.9, 0.02
EURO[1]	8-3-1, tanh, tanh, 0.9, 0.02
EURO[2]	4-3-1, tanh, tanh, 0.9, 0.02
FRANC[1]	3-3-1, tanh, tanh, 0.9, 0.02
FRANC[2]	8-12-1, tanh, tanh, 0.9, 0.02
POUND[1]	2-3-1, tanh, tanh, 0.9, 0.02
POUND[2]	2-6-1, tanh, tanh, 0.9, 0.02
YEN[1]	2-12-1, tanh, tanh, 0.9, 0.02
YEN[2]	4-10-1, tanh, tanh, 0.9, 0.02

6.2 Training with Genetic Algorithms

The version of Genetic Algorithm used in this comparison was a modification of Dorsey et al. (1994) and Rooij et al. (1996). The program was written in the Mat Lab® environment. Using this method and the pre-selected network architecture, initial population weight matrices were presented, which were mapped directly into genotype representations. Genetic operations were then used to act upon populations of these genotypes to produce a higher fitness level. Real valued encoding was used. As an

example, the genotype and phenotype representations for a network with two input nodes and two hidden nodes are depicted in Figure 6.1.

A GA requires no pre-selection of parameters other than the number of input nodes and the number of hidden nodes. In training the five foreign exchanges using Genetic Algorithm, two NN architectures were used for comparison purposes. Table 6.1. illustrates the specific parameter settings for each currency. Each configuration was trained with five replications, each for maximum 1000 generations. Each replication differed only in the random initialization of the network. Since the optimal solutions are not known apriori, the size of the search space was set between -10 and 10.

The normal parameters, including the population size, were set in the common pattern of most researchers, making it easier to compare results with other reported research, if desired. The population size was set to 20 points per generation. A ranking mechanism was used that ranks the chromosomes in order of fitness. For selection, the standard roulette selection mechanism was used. In every generation all chromosomes were replaced, except for the best performing chromosome, which is retained and inserted into the next generation without change (elitism scheme).

The crossover operator employed was linear with a crossover rate of 0.7. The mutation type employed was Gaussian addition, i.e., adding a random number to the mutated value. The standard deviation of this number is equal to 1 with a mean of 10. The mutation rate was 0.02.

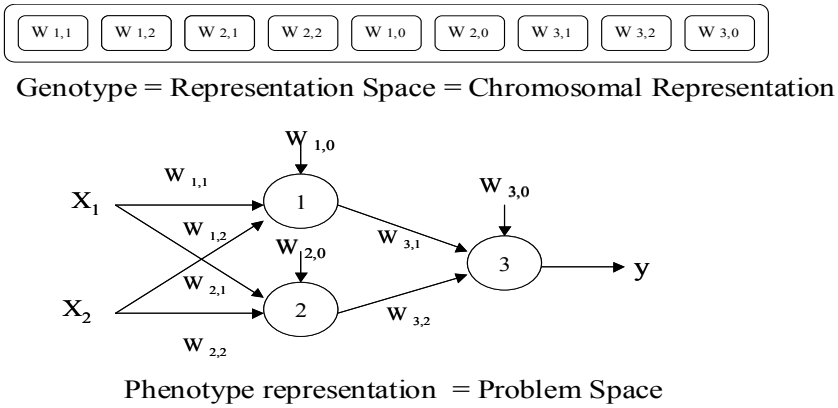


Figure 6.1 The Genotype and Phenotype Representations

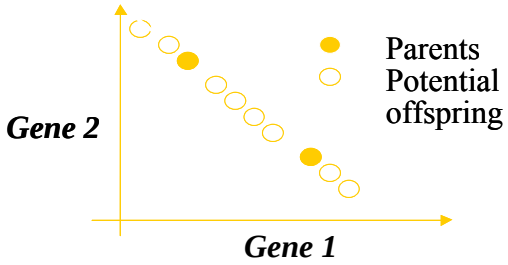


Figure 6.2 Linear Recombination Crossover

6.3 Training with Modified Tabu Search

The Modified Tabu Search (TS) used for comparison with either BP trained networks or genetic algorithm trained networks was the one suggested by Sexton et al. (1998). In this study, the maximum number of iterations was used to terminate the process. When the number of iterations has taken place 10 times without improvement over the best solution, the algorithm will terminate. The number of neighborhood searches (NH), and the number of local searches (NS) were both set at 100.

As implemented for this problem, the neighborhoods are simply randomly drawn points from uniform distribution, while the local searches were conducted as random

searches within the neighborhood. The neighborhood was a region restricted to its original value $\pm$ one percent for each parameter value in the initial neighborhood point. The size of the Tabu List was set equal to 100.

Since the NN weights deal with real values, the likelihood of finding solutions that are identical is extremely small. For that reason, a proximity criterion, i.e., Tabu Criterion (TC) was applied to relax the strict comparison of the new solution to the present solutions in the Tabu List. The TC was set at level 0.01 %. Besides those, an aspiration level condition was also included for allowing overriding the rejection of a new solution that meets the Tabu Criterion if the new solution is better than the best solution so far.

The Modified Tabu Search algorithm described above was implemented in Mat Lab® environment. For each problem, the algorithm was run five times each with new random seed.

6.4 Comparisons

In this comparison, the parameter was set as to what BP did best in previous research. This was done to generate confidence that the optimal values were used for BP. The comparison of the algorithms were based on true directional changes (DIR), mean squared error (MSE), and mean absolute percentage error (MAPE). Tables 6.2-6.4 show the relative performances of GA and TS compared to BP for five foreign exchange rates, each with two different architectures.

Table 6.2 Mean Squared Error (MSE) Performances of BP, GA, and TS

MSE

	Mean			Variance		
	BP	**GA**	**TS**	**BP**	**GA**	**TS**
AUS[1]	0.0017	***0.0008***	0.0012	0.0000	0.0000	0.0000
AUS[2]	0.0091	***0.0011***	0.0015	0.0000	0.0000	0.0000
EURO[1]	0.0008	***0.0003***	0.0005	0.0000	0.0000	0.0000
EURO[2]	0.0020	***0.0002***	0.0003	0.0000	0.0000	0.0000
FRANC[1]	0.0006	***0.0006***	0.0007	0.0000	0.0000	0.0000
FRANC[2]	0.0060	***0.0008***	0.0028	0.0000	0.0000	0.0000
POUND[1]	0.0005	0.0002	***0.0001***	0.0000	0.0000	0.0000
POUND[2]	0.0005	***0.0001***	0.0002	0.0000	0.0000	0.0000
YEN[1]	57.2600	***1.7862***	2.8930	139.0041	***0.0072***	0.9852
YEN[2]	33.6500	***2.7274***	4.3766	13.9876	***0.3773***	2.2625

Table 6.3 Mean Absolute Percentage Error (MAPE) Performances of BP, GA, and TS

MAPE

	Mean			Variance		
	BP	**GA**	**TS**	**BP**	**GA**	**TS**
AUS[1]	2.6400	***1.4840***	1.9098	0.1764	***0.0307***	0.0877
AUS[2]	3.5100	***1.6960***	2.0824	0.9409	***0.0655***	0.1728
EURO[1]	3.1500	***1.6680***	2.0490	2.8561	***0.0276***	0.1171
EURO[2]	4.4800	***1.3640***	1.4424	0.0484	***0.0152***	0.0160
FRANC[1]	2.1700	***1.5100***	1.6584	0.2116	***0.0210***	0.0281
FRANC[2]	5.1000	***1.7100***	3.3406	4.2436	***0.0245***	0.8636
POUND[1]	3.0200	1.1680	***1.1516***	1.3924	0.0230	***0.0177***
POUND[2]	2.7300	***1.0280***	1.0762	0.9409	***0.0069***	0.0126
YEN[1]	4.5300	***0.9000***	1.1744	1.1025	***0.0007***	0.0561
YEN[2]	5.2700	***1.1240***	1.4756	1.6641	***0.0154***	0.0970

Table 6.4 The True Directional Changes (DIR) of BP, GA, and TS

DIR

	Mean			Variance		
	BP	**GA**	**TS**	**BP**	**GA**	**TS**
AUS[1]	64.00	***64.07***	61.67	***27.98***	30.44	40.40
AUS[2]	63.00	61.22	***63.75***	67.08	62.46	***18.67***
EURO[1]	58.13	56.27	***58.62***	37.82	***14.59***	28.00
EURO[2]	***57.02***	53.18	52.61	29.48	18.11	***3.30***
FRANC[1]	58.33	***58.71***	53.61	17.39	15.98	***14.48***
FRANC[2]	***58.25***	53.02	50.47	***20.07***	36.24	35.17
POUND[1]	62.58	63.65	***67.08***	1.39	***0.86***	3.04
POUND[2]	61.90	***63.26***	62.08	5.57	12.50	***3.04***
YEN[1]	***61.90***	50.61	52.50	***1.39***	7.10	7.37
YEN[2]	***58.15***	46.81	53.47	***6.05***	18.12	20.29

Comparing BP and GA, the GA algorithm found better solutions for 10 out of 10 problems in terms of MSE and MAPE. It is worth to note that GA solution variance was significantly lower for those two statistical performances. However, GA is only better for four out of 10 problems in terms of DIR, while overpowering BP for five out of 10 problems in terms of the variance.

Comparing BP and TS, the TS algorithm found superior solutions for nine out of 10 problems in terms of MSE and for 10 out of 10 problems in terms of MAPE. It is also interesting to note that BP solution variance was significantly higher. It showed that the TS algorithm was less likely to get stuck in bad local solutions. In terms of DIR, the TS algorithm found superior solutions in four out of 10 test problems. For five problems, the TS variance was lower compared to that of BP.

It is interesting to find out the relative performances of GA and TS. In terms of MSE and MAPE, the GA seems to perform better than TS but shows indifferent performances in terms of DIR.

Overall, BP found inferior solutions for all problems compared to either GA or TS in both means and variance of MSE and MAPE. In terms of DIR, BP found superior solutions for four out of ten problems for means. On the other hand, compared to BP, GA guarantees better performance in term of MAPE and most of the time performs better in term of MSE. It should be noted, however, that there is no uncertainty in the relative performance of DIR.

Due to the fact that the time to perform GA is almost twice as the one for BP (on average 1.79 times), and that BP was run two times in order to avoid being trapped in a poor local optimum, GA is quite competitive. Furthermore, it is apparent that GA is much easier to use. In contrast to BP, which required extensive search for topology determination, there was no need to search for optimal parameter configurations for the GA to outperform BP. For that reason, further discussions will be based on GA training. Figure 6.3 - 6.7 are provided to graphically illustrate the difference in BP, GA, and TS for the test data forecasts for each currency. From the graphs, one can easily learn that GA performed better compared to BP for all cases.

The residuals plot of the training set for the above forecasting problems using GA training were shown in Figure 6.8 – 6.12. It can be seen that for all cases, the residuals fall within a horizontal band centered on zero with constant variance. They display no systematic tendencies to be either negative or positive, and therefore present desired patterns.

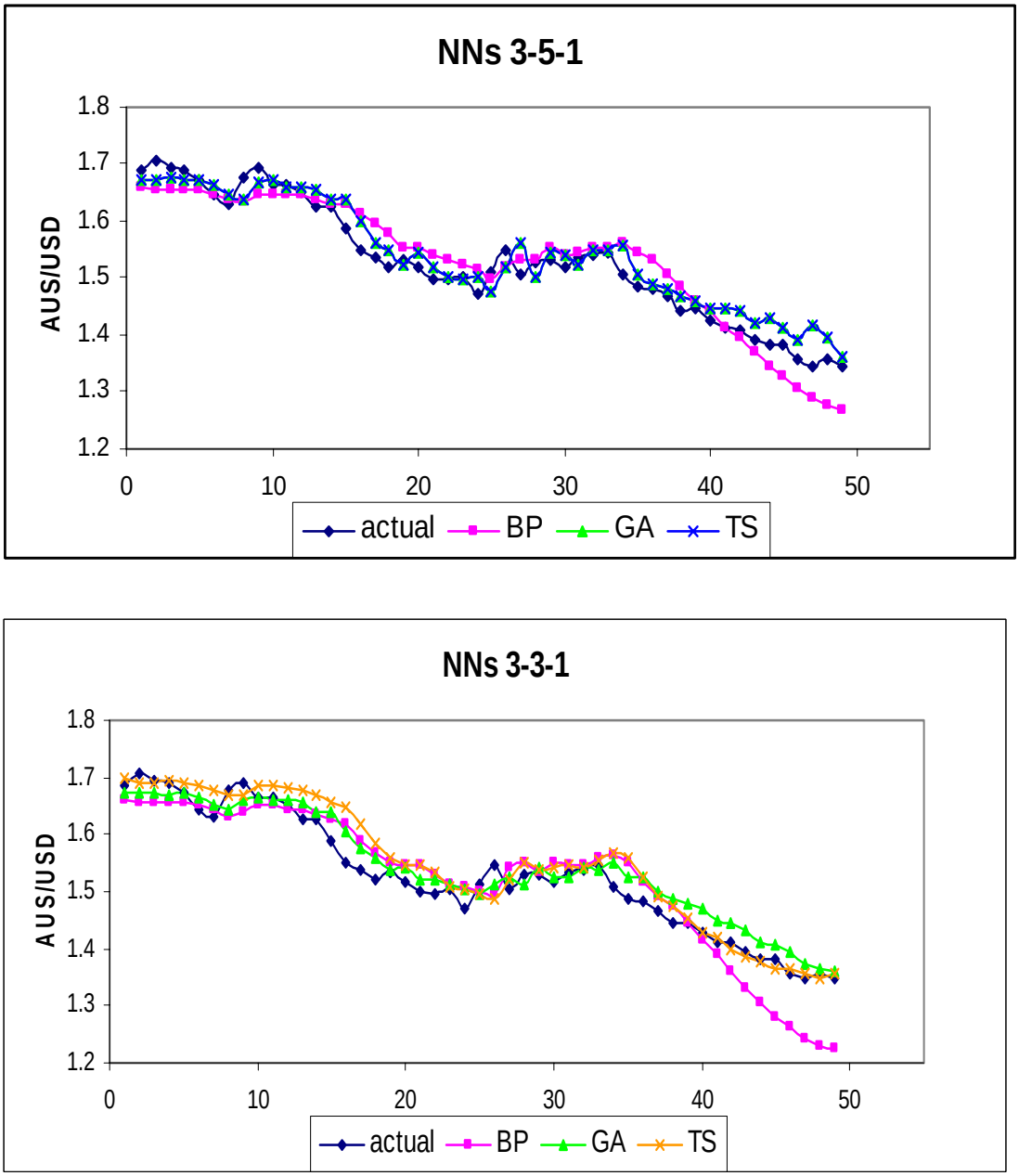
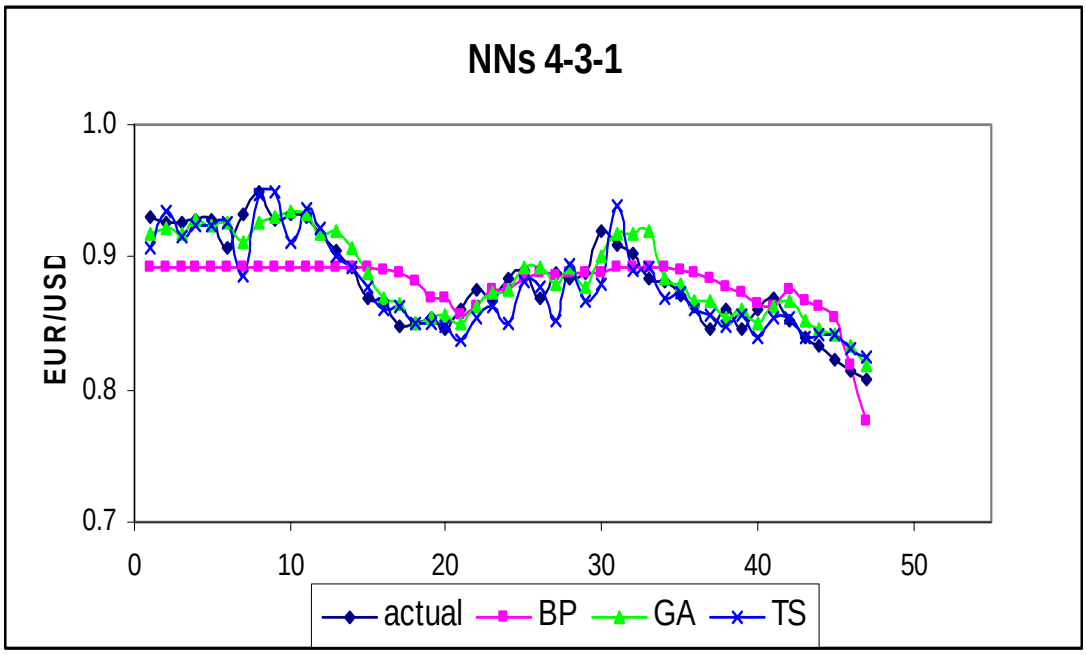


Figure 6.3 AUS/USD forecasts using three different training methods



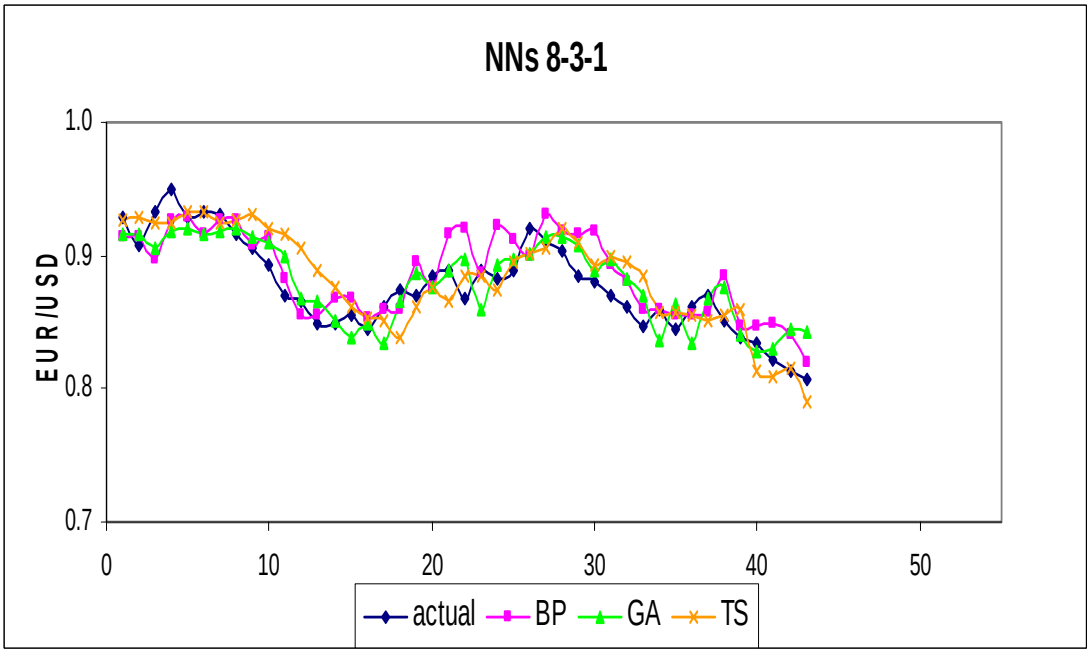


Figure 6.4 EUR/USD forecasts using three different training methods

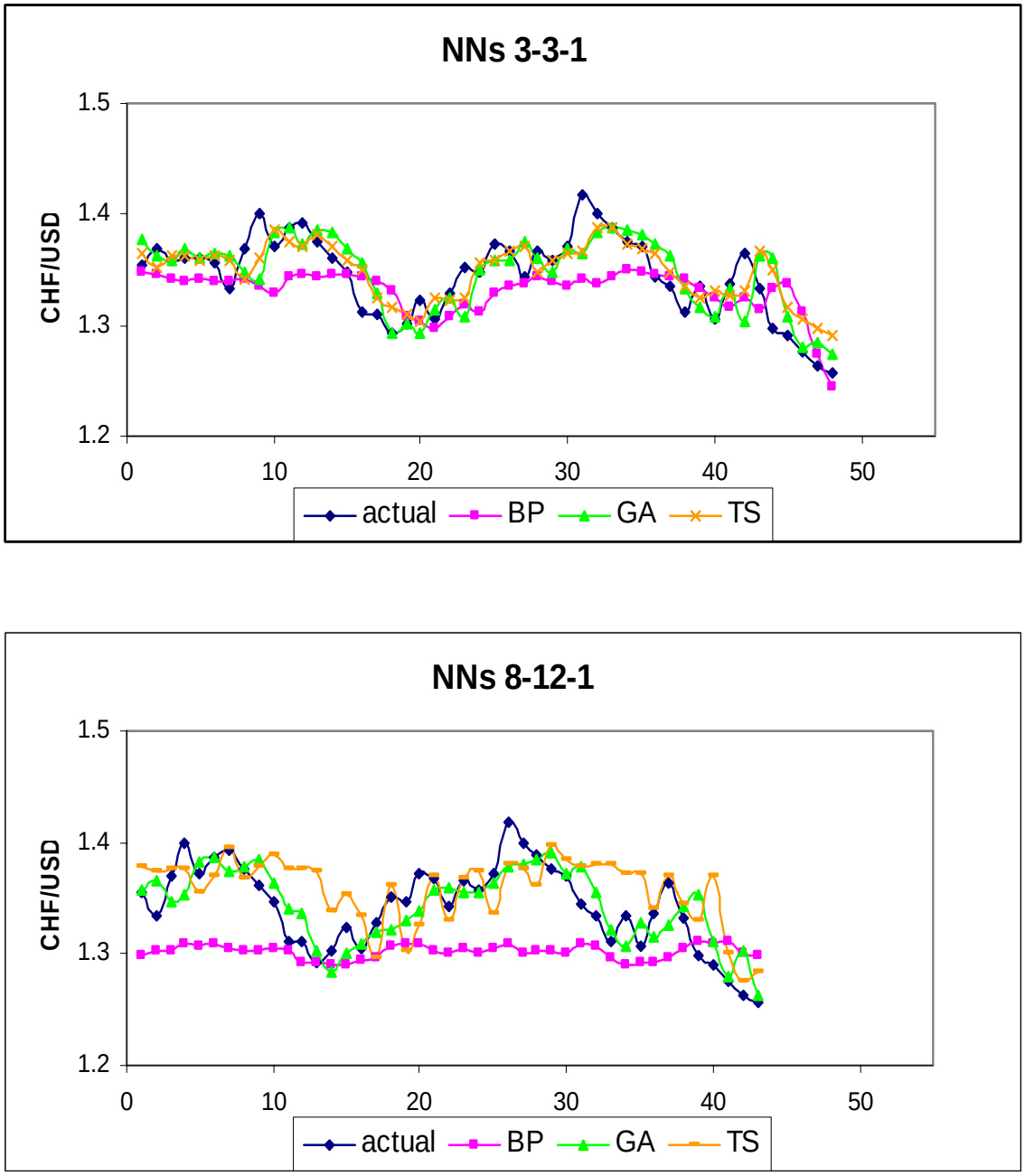


Figure 6.5 CHF/USD forecasts using three different training methods

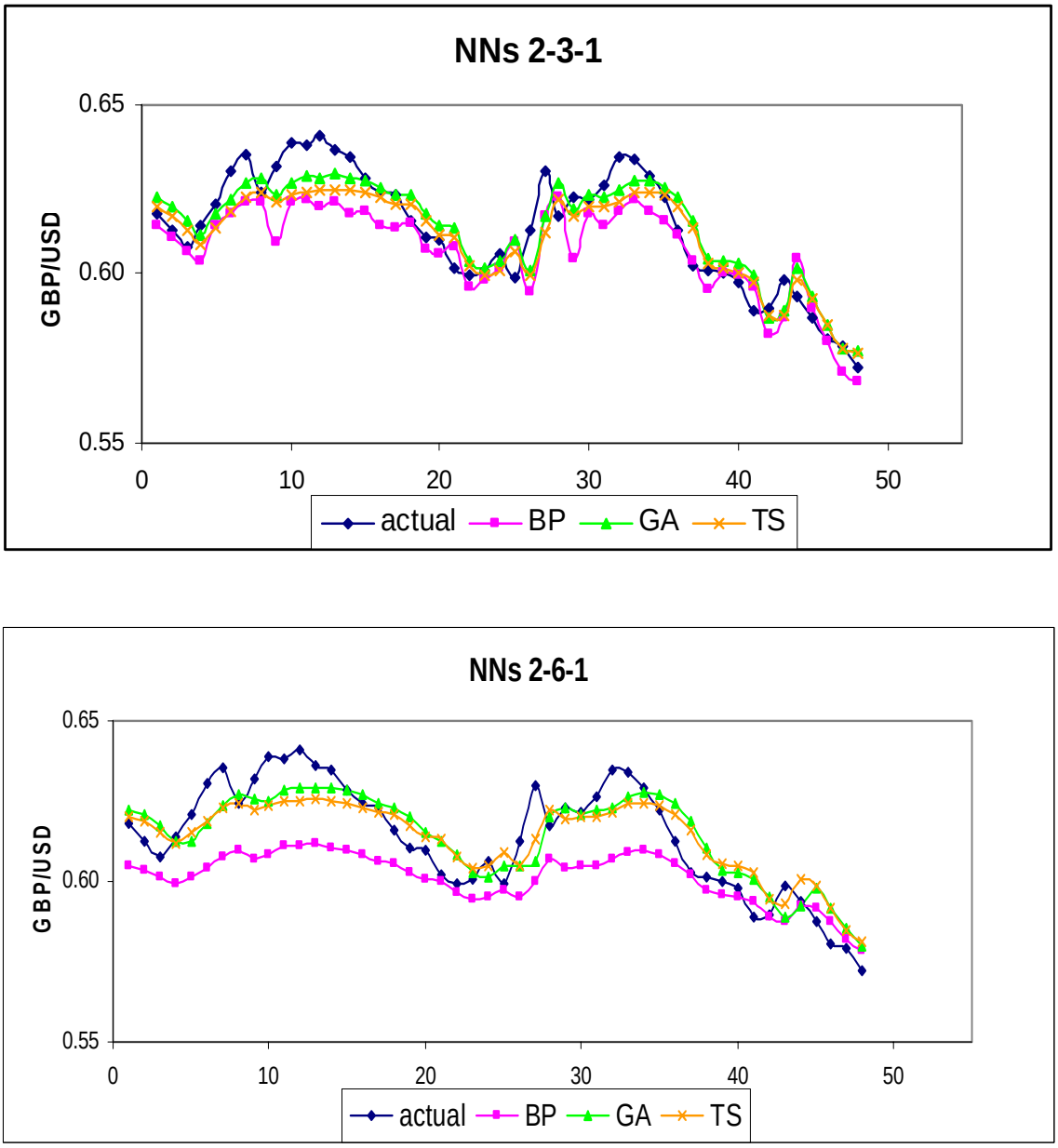


Figure 6.6 GBP/USD forecasts using three different training methods

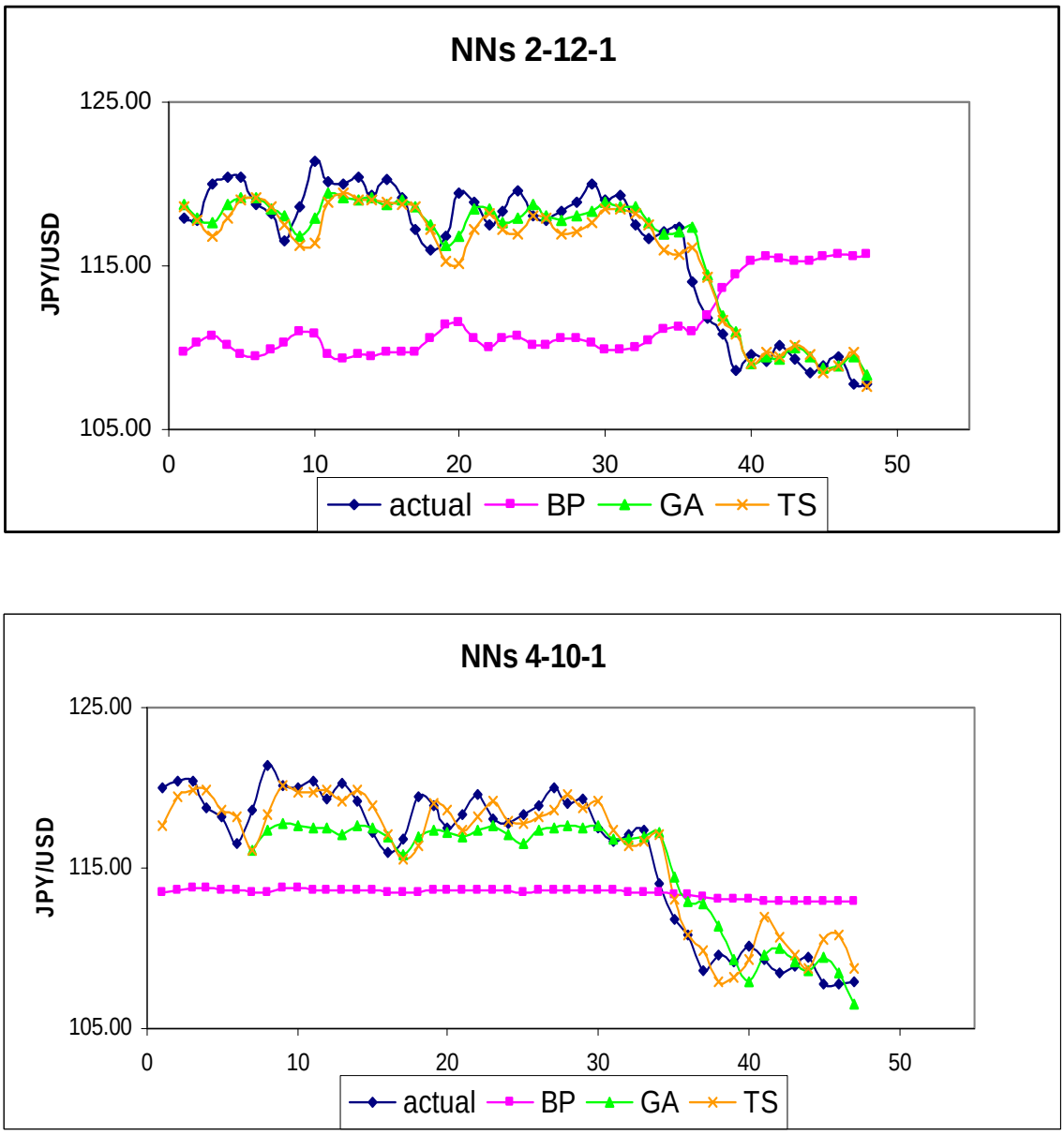


Figure 6.7 JPY/USD forecasts using three different training methods

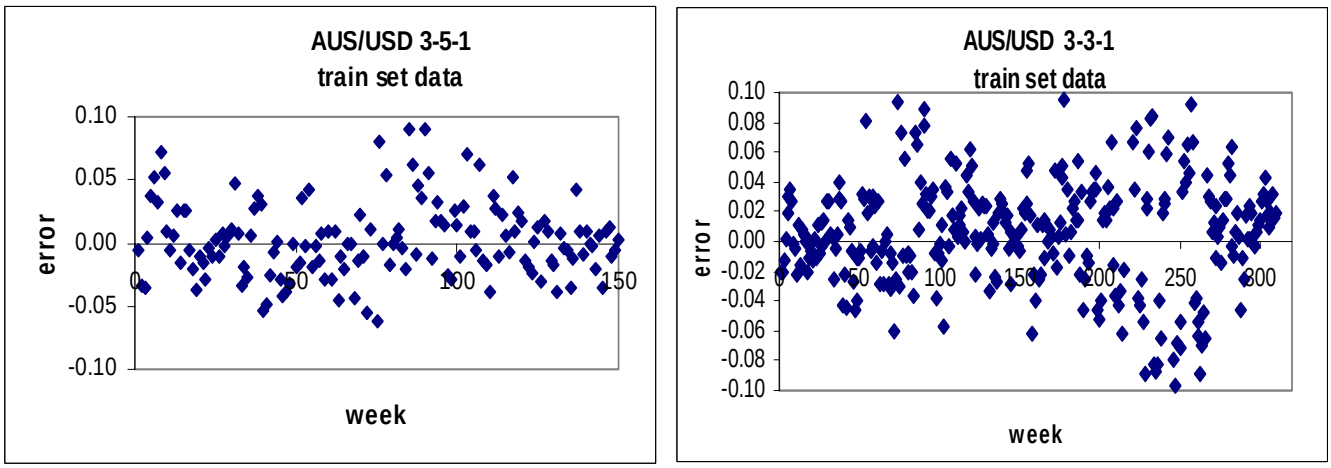


Figure 6.8. Plot of residuals for GA training (AUS/USD data)

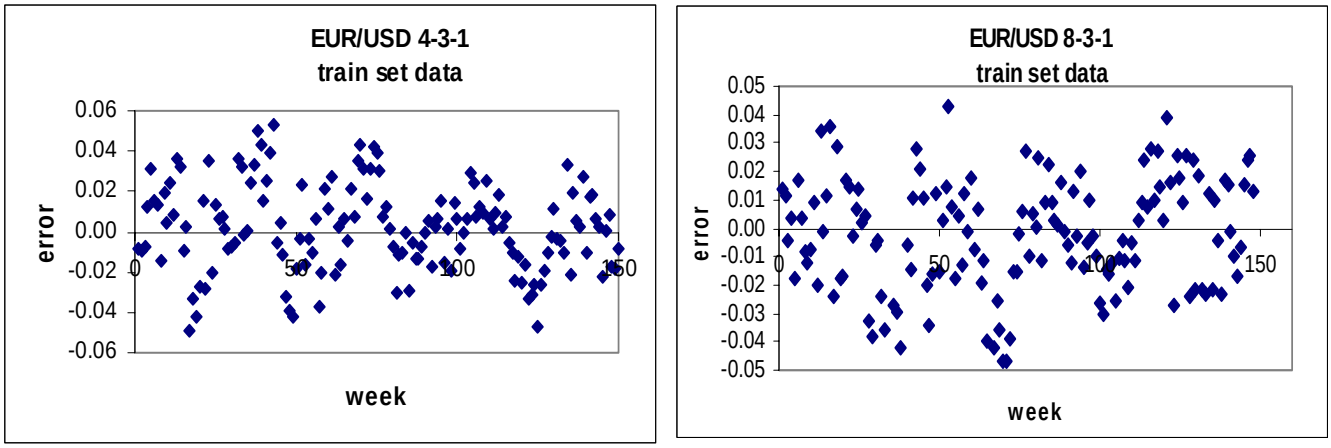


Figure 6.9. Plot of residuals for GA training (EUR/USD data)

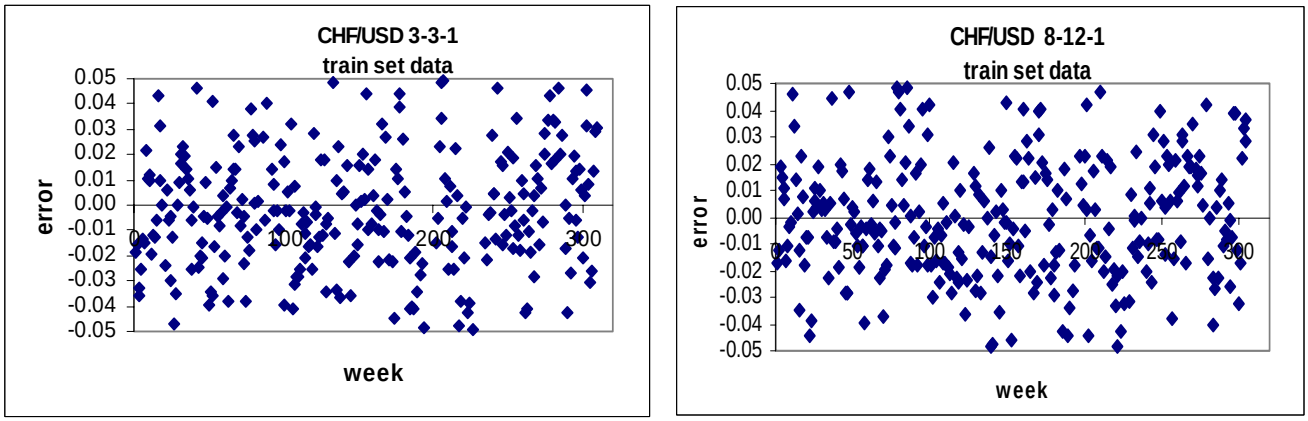


Figure 6.10. Plot of residuals for GA training (CHF/USD data)

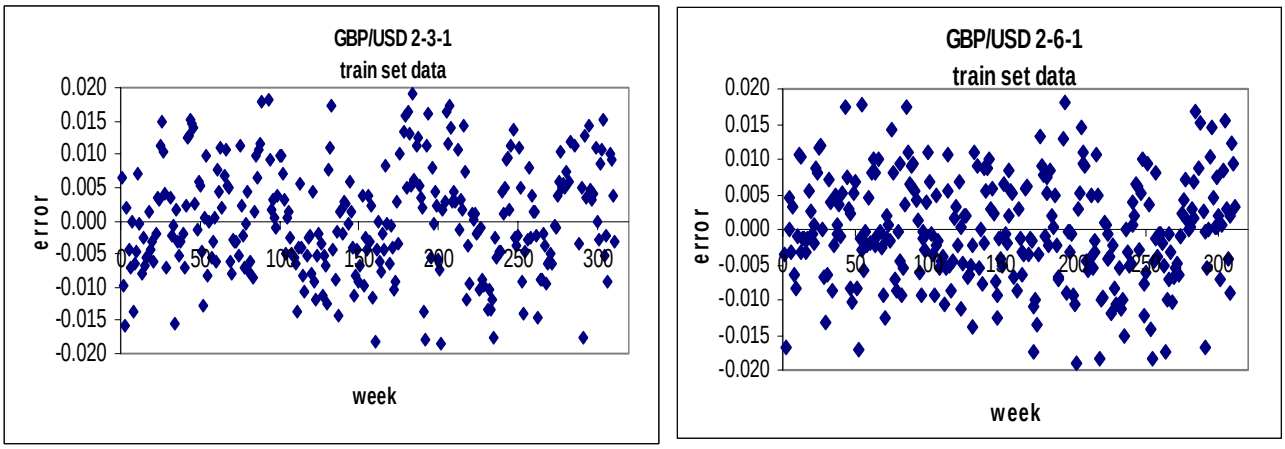


Figure 6.11 Plot of residuals for GA training (GBP/USD data)

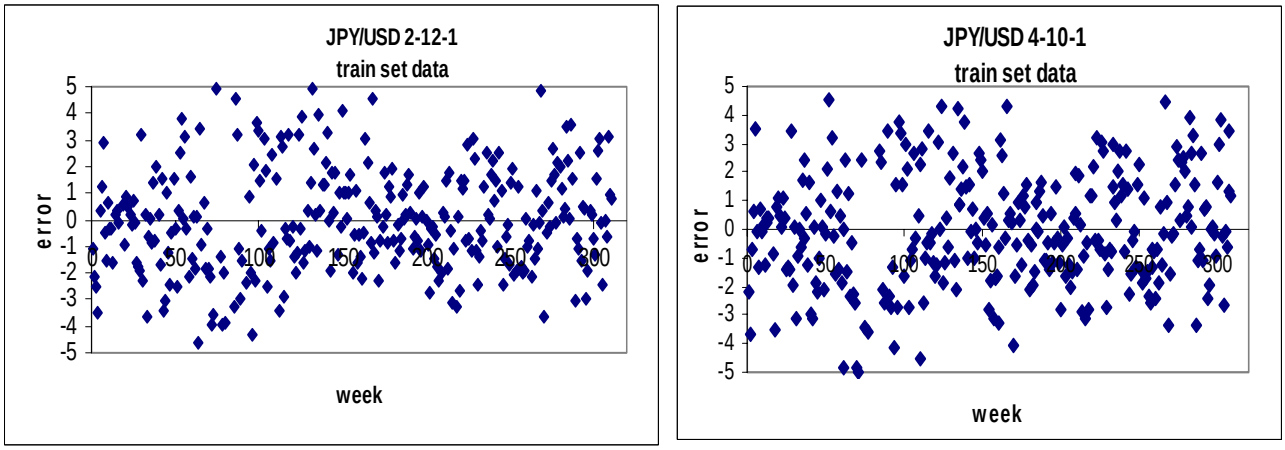


Figure 6.12 Plot of residuals for training set (JPY/USD data)

Chapter 7

Forecasting Accuracy Issues

7.1 Influence of Time Period

The influence of the time period was studied by performing six years rolling training groups for GBP/USD to be comparable with similar work done by Walczak (2001). The data is segmented into seven different data sets, each with a span of seven years (six years for training data and one year for test data) with an overlapping period of one year.

A series of NNs that use rolling training data sets for forecasting the GBP/USD exchange rates were built. Six new NNs were constructed with the identical number of input nodes and hidden nodes (two input nodes and three hidden nodes) and identical activation functions (hyperbolic tangent) using six-year training periods to forecast the following year. The data sets are given in Table 7.1, and time period seven was the default used in the previous chapters. The newest data set, i.e., training period January 1998-December 2003 and testing period January 2004-December 2004 were examined and labeled as time period eight. Results for all data segments are displayed in Table 7.2. The predictions of the weekly GBP/USD for each data segment are depicted in Figure 7.1 – Figure 7.7.

Table 7.1 Statistics for Different Time Period (GBP/USD)

Time Period	Training Period	Testing Period	Mean	Median	Max	Min	Std Dev
1	Jan 1991-Dec 1996	Jan 1997 - Dec 1997	0.6220	0.6384	0.7052	0.5010	0.0468
2	Jan 1992-Dec 1997	Jan 1998 - Dec 1998	0.6291	0.6384	0.7052	0.5010	0.0391
3	Jan 1993-Dec 1998	Jan 1999 - Dec 1999	0.6345	0.6346	0.7052	0.5863	0.0266
4	Jan 1994-Dec 1999	Jan 2000 - Dec 2000	0.6265	0.6229	0.6826	0.5863	0.0220
5	Jan 1995-Dec 2000	Jan 2001 - Dec 2001	0.6279	0.6227	0.7144	0.5863	0.0255
6	Jan 1996-Dec 2001	Jan 2002 - Dec 2002	0.6380	0.6240	0.7236	0.5863	0.0359
7	Jan 1997-Dec 2002	Jan 2003 - Dec 2003	0.6381	0.6241	0.7236	0.5657	0.0375

Table 7.2 One-Year Forecasting Performances of the GBP/USD

Time Period	*Training Period*	*Testing Period*	*MSE* *)	*MAPE* *)	*DIR* *)
1	Jan 1991 - Dec 1996	Jan 1997 - Dec 1997	6.540E-05	1.0060	50.00
2	Jan 1992 - Dec 1997	Jan 1998 - Dec 1998	5.062E-05	0.8368	50.40
3	Jan 1993 - Dec 1998	Jan 1999 - Dec 1999	3.300E-05	0.7406	52.20
4	Jan 1994 - Dec 1999	Jan 2000 - Dec 2000	8.596E-05	1.0910	54.40
5	Jan 1995 - Dec 2000	Jan 2001 - Dec 2001	6.036E-05	0.8730	45.20
6	Jan 1996 - Dec 2001	Jan 2002 - Dec 2002	7.884E-05	1.1324	51.20
7 **(Default)**	Jan 1997 - Dec 2002	Jan 2003 - Dec 2003	**5.933E-05**	**1.0754**	**64.62**
8	Jan 1998 - Dec 2003	Jan 2004 - Dec 2004	6.130E-05	1.1660	46.66

*) average from five replications

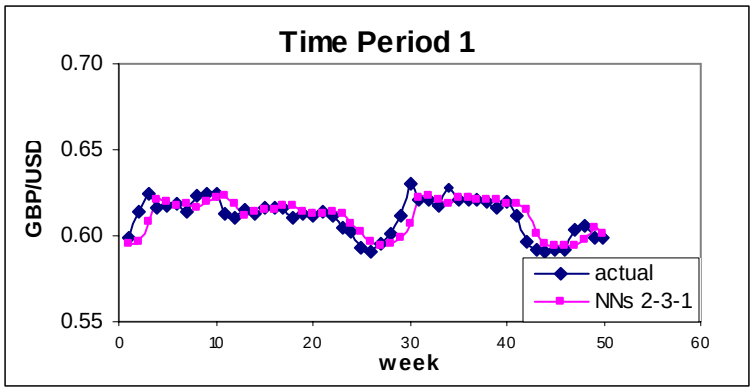


Figure 7.1 Prediction of the Weekly GBP/USD for Time Period 1

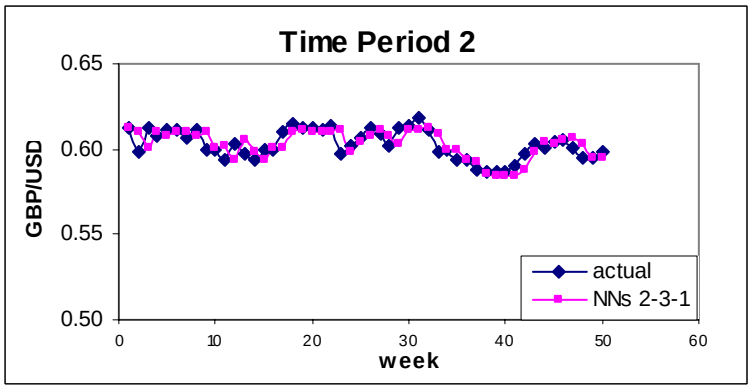


Figure 7.2 Prediction of the Weekly GBP/USD for Time Period 2

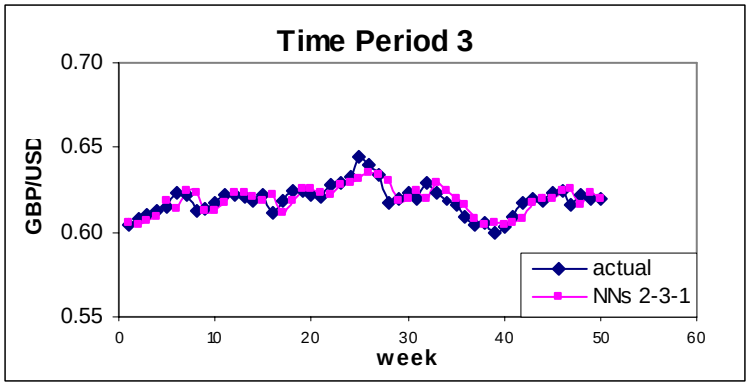


Figure 7.3 Prediction of the Weekly GBP/USD for Time Period 3

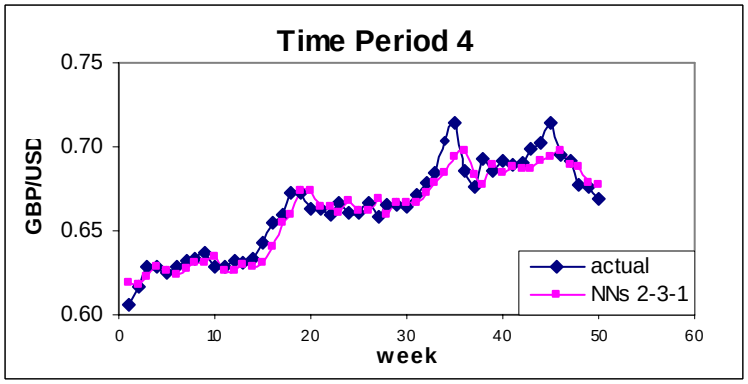


Figure 7.4 Prediction of the Weekly GBP/USD for Time Period 4

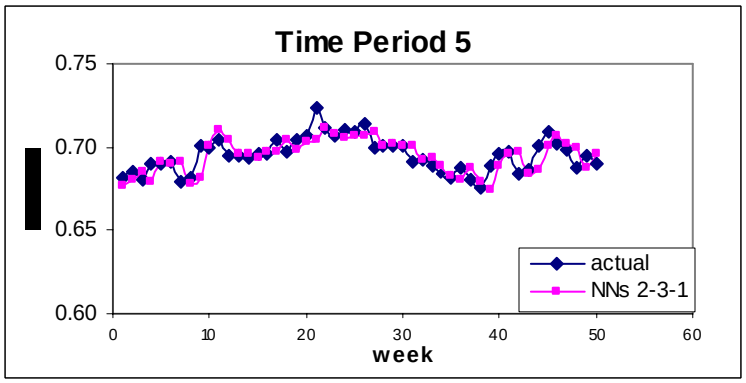


Figure 7.5 Prediction of the Weekly GBP/USD for Time Period 5

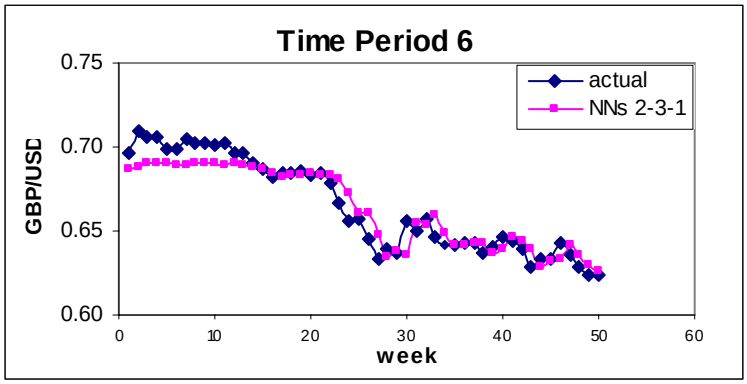


Figure 7.6 Prediction of the Weekly GBP/USD for Time Period 6

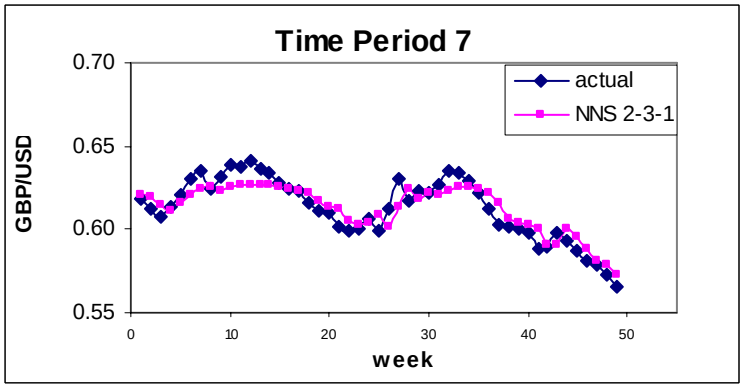


Figure 7.7 Prediction of the Weekly GBP/USD for Time Period 7

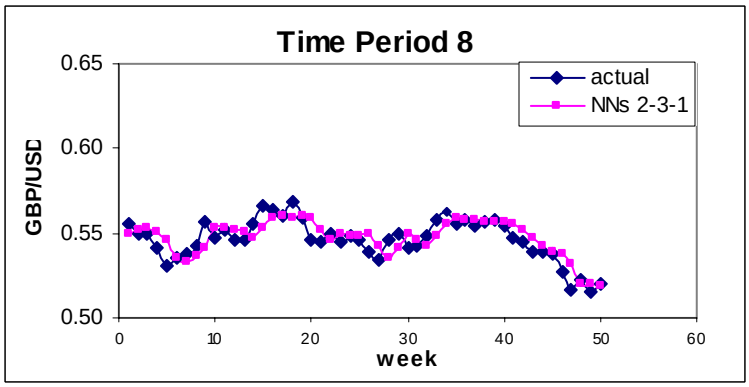


Figure 7.8 Prediction of the Weekly GBP/USD for Time Period 8

The NNs that utilize six-year rolling training for forecasting the GBP/USD exchange rate did not achieve the performances as good as the one for time period seven (the data set examined in the previous chapters), especially in terms of directional changes. It seems that there is correlation between the number of turning points in the test data set and the true directional change predictions, as depicted in Table 7.3. The two worst performances, i.e., for periods five and eight, happen to be the periods with the highest number of turning points. The best one, i.e. period seven is the one with the least number of turning points (16 out of 51). On the contrary, it seems that there is no relationship between the DIR statistics and the number of big changes in each period.

Table 7.3 Number of Turning Points

Period	DIR	Number of Turning Points	Number of Change > 0.01	Number of Change > 0.015
1	50.00	23	9	3
2	50.40	23	8	1
3	52.20	22	5	1
4	54.40	22	12	5
5	45.20	**29**	8	2
6	51.20	23	8	1
7	64.62	**16**	7	1
8	46.66	**27**	9	1

Nevertheless, the MSE and MAPE values for all periods are fairly good. The above results clearly indicate that the best topology to forecast a specific foreign exchange for a certain period is not always the best one to forecast other periods. Hence, caution should be used in determining the correct topology, because it is problem specific. This is reasonable because each data set has a different data pattern. As shown

in Figure 7.1 – 7.8, different data sets of the GBP/USD exchange rates exhibit different trends and fluctuations.

Actually, a different time period is a perfect example for an unknown problem set. Hence, one should determine the appropriate topology before applying NNs for time series forecasting. Both Genetic Algorithm for Topology Determination or Tabu Search Algorithm for Topology Determination can be employed to provide a relatively good topology. For GBP/USD for time period six for example, Tabu Search for Topology Determination suggested using number of input = 3 and number of hidden nodes = 5. A Genetic Algorithm for weight training using this topology results in the following forecasting performances: MSE=0.0000717, MAPE=1.012, DIR=55.10, which is better than that previously obtained.

It can be concluded that topology optimization is "case specific". Hence, one should make certain the proper topology for the specific set of data is determined before proceeding to forecast the foreign exchange rates using NNs.

7.2 Multi-Step Ahead Forecasting

Having studied the one-step ahead forecasting performances using NNs with various training algorithms, a similar procedure was then applied to generate multi-step ahead forecasts for GBP/USD and EUR/USD. The number of input nodes and number of hidden nodes were specified to be the same as the ones for one-step ahead forecasting, i.e., 2-3-1 for GBP/USD and 3-3-1 for EUR/USD, all with tanh activation functions. Since the training method applied was Genetic Algorithm, the training parameters, such as learning rate and momentum term, did not need to be specified. The Genetic

Algorithm was chosen because of its superiority compared to other training methods as shown in Chapter 6.

The graphical outputs for one-, two-, three-, four-, and five- step ahead forecasts for GBP/USD are presented in Figure 7.9. The one for EUR/USD is presented in Figure 7.10. The results in Figure 7.11 indicate that, given a certain topology determined for one-step ahead forecasting, the MSE increases as the number of multiple ahead forecasting increases. The same phenomena for MAPE are clearly shown. Those trends are true for both GBP/USD and EUR/USD exchange rate forecasts. Unfortunately, there is no clear pattern on the direction behavior. A summary of the multi-step ahead forecasting for GBP/USD and EUR/USD is given in Table 7.4.

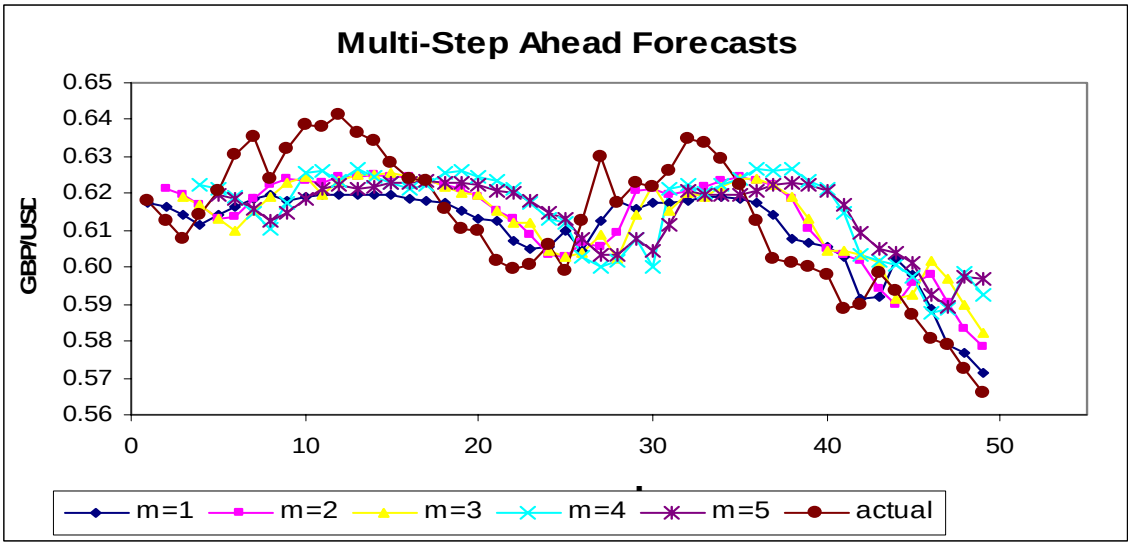


Figure 7.9 Graphical Output for GBP/USD

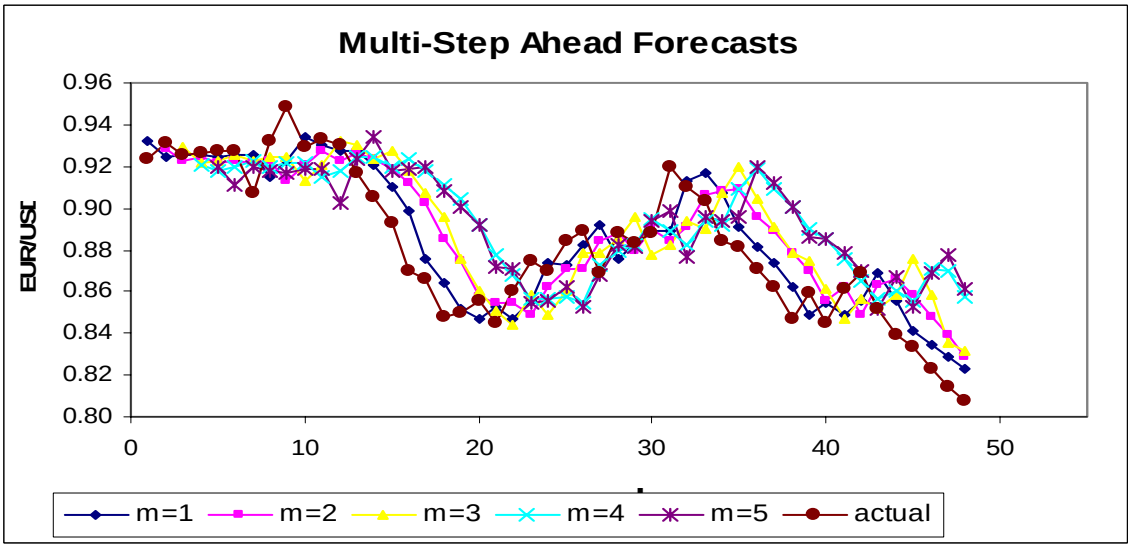


Figure 7.10 Graphical Output for EUR/USD

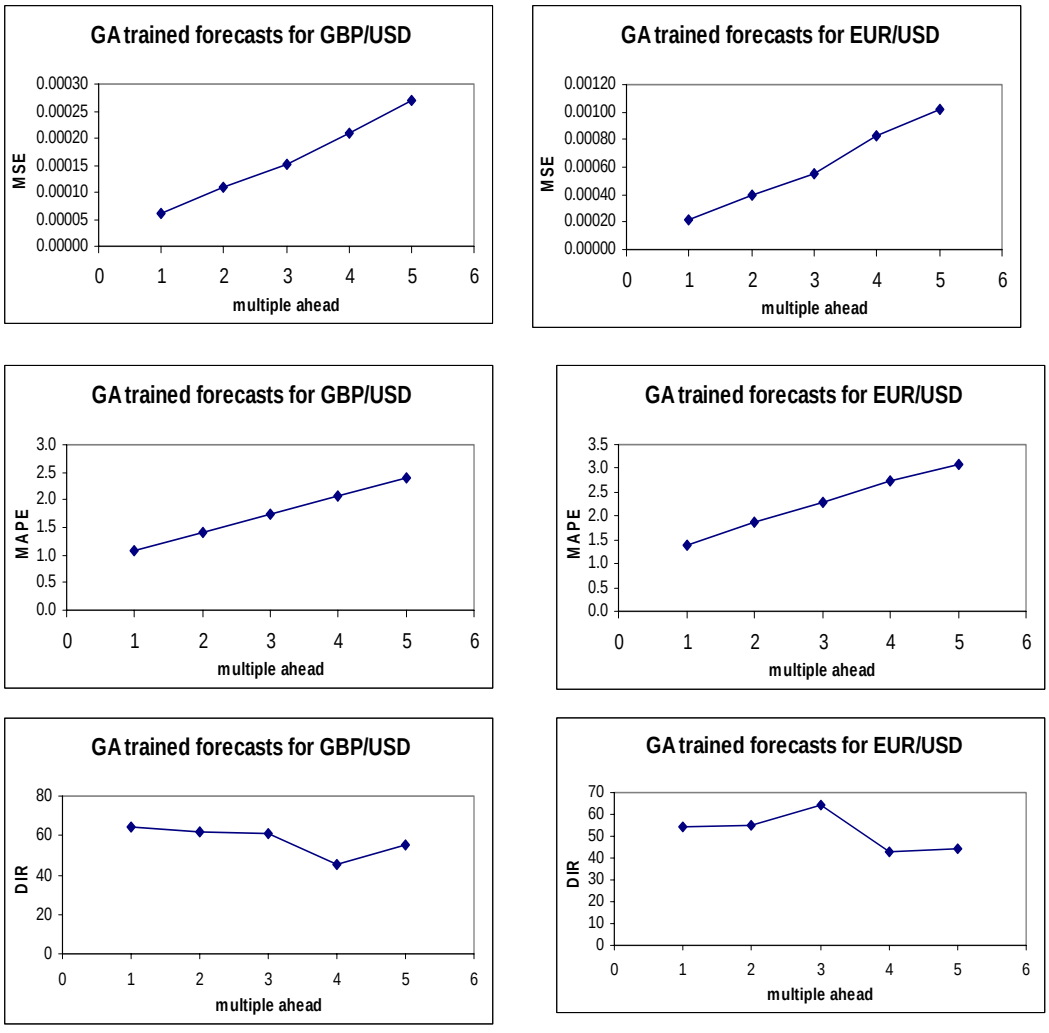


Figure 7.11 Statistical Performances Behavior for GBP/USD and EUR/USD

Table 7.4 Summary of Multi-Step Ahead Forecasting for GBP/USD and EUR/USD

GBP/USD	MSE	MAPE	DIR
mahead=2	0.0001	1.4180	61.57
mahead=3	0.0002	1.7540	61.27
mahead=4	0.0002	2.0660	49.73
mahead=5	0.0003	2.3940	55.11

EUR/USD	MSE	MAPE	DIR
mahead=2	0.0004	1.8832	55.30
mahead=3	0.0006	2.2740	64.34
mahead=4	0.0008	2.7518	43.11
mahead=5	0.0010	3.0723	44.08

It can be concluded for the two examples in Table 7.4 that one can extend the use of a chosen topology up to three-steps ahead forecasting.

Moreover, comparison study on the GA-trained neural network as discussed previously vs. Double Exponential Smoothing (Holt's method) were conducted to examine the quality of multi step ahead forecasts obtained using the first method. Five different values in the range of 0.1 – 0.5 for each of the two smoothing contents in Double Exponential Smoothing (DES), e.g. α and γ, were investigated. The one gave the smallest MSE for one step ahead forecasting ($\alpha = 0.5$ and $\gamma=0.3$) were used. As indicated in Table 7.5, GA-trained neural network provide better results compared to DES for both forex either for one-step ahead or three-step ahead forecasts. Compared to DES, GA-trained NNs improved the MSE, MAPE, and DIR for three-step ahead forecasts by 8.92 - 53.29 %, 30.40 - 31.69 %, and 15.48 - 47.07 %, respectively.

Table 7.5 Statistical Performances of Multi-Step Ahead Forecasts

		MSE	MAPE	DIR
One-Step Ahead Forecasts				
GBP/USD	NNs *)	0.00006	1.07540	64.62
	DES **)	0.00006	1.08310	42.00
EUR/USD	NNs ***)	0.00022	1.39460	57.05
	DES **)	0.00025	1.53110	53.06
Three-Step Ahead Forecasts				
GBP/USD	NNs *)	1.75400	0.00015	61.27
	DES **)	1.92570	0.00022	53.06
EUR/USD	NNs ***)	0.00055	2.27398	64.34
	DES **)	0.00119	3.26720	43.75

*) GA trained 2-3-1 **) Holt's Method (0.5, 0.3) ***) GA trained 3-3-1

7.3 Monitoring the Forecasting Process

Essentially, the NN will work best if it is retrained whenever new data is obtained. However, the time and cost to update the forecast can be huge. As a common practice, one should determine the maximum time an NN should be retrained to guarantee good forecasts. Yao et al. suggested that weekly foreign exchange estimates need to be reforecast every half-year. It is important to note that the conclusion was made based on visual interpretations (based on forecasting figures). No test was used to reach this conclusion. However, it was observed that in general the MSE, MAPE, and DIR for half-year forecasts are indeed better than those for one-year forecasts. For EUR/USD 3-3-1 three-step ahead forecasting, for example, the difference on those values was verified to be significantly different with the corresponding p values being less than 0.01.

A need to reforecast can be learned by monitoring the forecasts. One way to monitor forecasts to ensure that they are performing well is to use a tracking signal. A tracking signal is a measurement of how well the forecast is predicting actual values. In this research, a tracking signal test was applied to the testing data set for that purpose. A

tracking signal test determines when a pattern or relationship has changed; hence, there is a need to retrain NNs with newer data.

Tracking signals are expressed in the form of ratios. The numerator is a sum of forecast errors, while the denominator is the mean absolute deviation. The denominator, which measures the long-run average variability of forecast errors, was used to standardize the numerator. The tracking signal is usually computed as the running sum of the forecast errors (RSFE) divided by the mean absolute deviation (MAD), as given in the following formula:

$$\text{Tracking signal} = \frac{RSFE}{MAD} = absolute\left(\frac{\sum_i (Ai - Fi)}{MAD}\right) \quad (7.1)$$

The flowchart of a tracking signal test is given in Figure 7.12.

If the forecast is in control, the sum of the numerator has an expected value of zero. The higher the tracking signal, the greater the possibility that the pattern being monitored has changed. The common control limit for a tracking signal is absolute (4.0) (Heizer and Render, 2003). Accordingly, the NN needs to be retrained if the tracking signal is greater than absolute (4.0).

This issue was studied for EUR/USD and GBP/USD, each for one-step ahead and three-step ahead forecasting obtained using GA weight training. Table 7.6 shows the results. The tracking signal values make obvious that three-step ahead forecasting not only produced larger errors, but also had to be reforecast more often than one-step ahead forecasting. The time to reforecast is apparently different for each case.

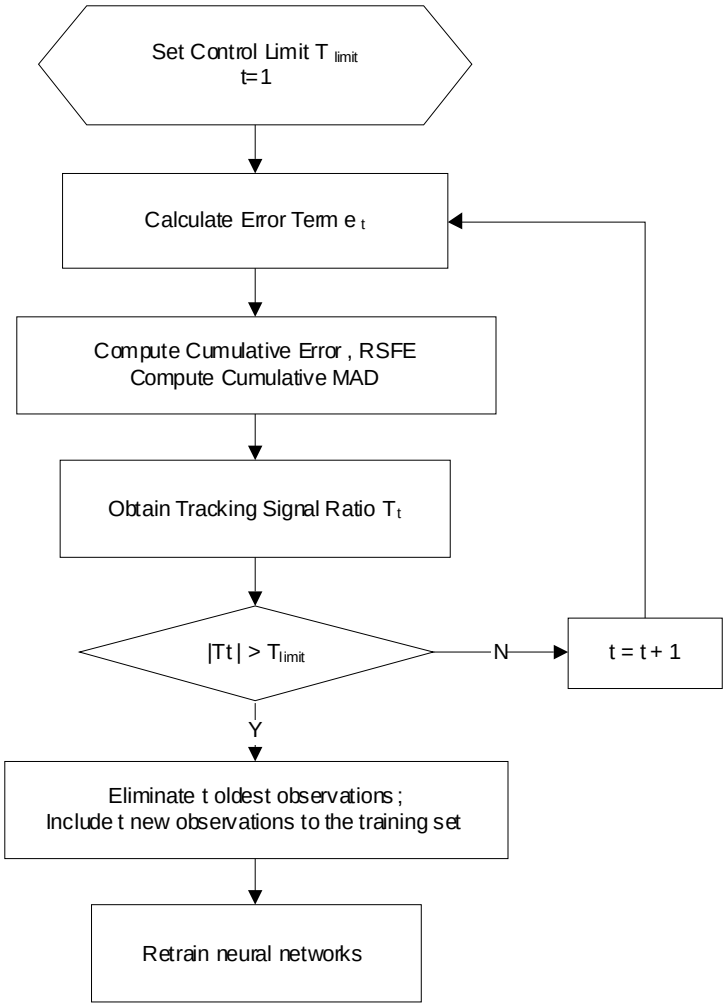


Figure 7.12 Flowchart for Tracking Signal Test

7.4 Building Prediction Intervals

Forecast error is inherent in any forecasting procedure. Therefore, stating the uncertainty associated with a forecast conveys useful information to the decision maker. Forecast accuracy can be quantitatively described by the variance of the forecast error. Further, the uncertainty associated with the forecasts can be described by prediction

Table 7.6 Tracking Signals for EUR/USD and GBP/USD

Week	Tracking Signal			
	EUR/USD 1 step	EUR/USD 3 steps	GBP/USD 1 step	GBP/USD 3 steps
1	1.00	1.00	1.00	1.00
2	0.27	-0.39	2.00	2.00
3	0.33	-1.46	3.00	1.27
4	0.38	-2.27	3.30	-0.90
5	-0.45	1.12	2.36	-2.14
6	-1.26	-0.27	0.70	-2.64
7	2.48	-2.94	-0.66	-3.27
8	-0.49	**-4.31**	-0.12	**-4.19**
9	-3.27	-5.45	-1.00	-5.35
10	-2.79	-5.70	-2.58	-6.39
11	-3.36	-4.09	-3.76	-7.24
12	-3.91	-2.11	**-5.13**	-8.05
13	-2.44	1.07	-6.10	-8.31
14	-0.54	4.19	-6.96	-7.95
15	1.43	6.22	-7.44	-7.44
16	**4.04**	8.16	-7.28	-6.14
17	4.99	9.35	-7.34	-4.51
18	6.36	10.03	-6.01	-2.94
19	6.81	10.73	-4.45	-1.45
20	6.04	9.81	-3.77	-0.35

intervals. To be able to get this information, the forecast error variance should be derived first. To show how to build a prediction method for a forecasting application, the method was applied for GBP/USD and EUR/USD, each for one-step ahead and three-step ahead forecasting.

One way to estimate the error variance is by applying the bootstrapping method. Before generating the bootstrapping residual pseudo replicates, residual analysis is a necessity for determining the empirical distribution. The residual analysis in this

research was conducted using graphical method (Q-Q plot) and quantitative method ("Goodness of Fit Test"). The Q-Q plots were obtained using SPSS® while the empirical distribution of training residuals were obtained using ARENA® input analyzer.

Having known the empirical distribution of the errors, the parameter of the bootstrapping must be determined. One important parameter in the bootstrapping method is the number of bootstrapping replications used. Generally, the higher the number of bootstrapping replications, the smaller estimated error variance. However Deng (2003) suggested using the number of bootstrapping replications of 20, which she argued is usually informative and gives a good estimate of the forecast error variance. In view of that, the bootstrapping replication for this research was set equal to 20.

To show the implementation of the method, GBP/USS and EUR/USD were used as examples for one-step ahead and three-step ahead forecasting. The preliminary examination of training residuals using Q-Q plots for both EUR/USD and GBP/USD indicates normality. The "Fits All Summary" obtained from ARENA for the residuals of both foreign exchanges forecasting indicates that the normal distribution fits the data well. The summary statistics are given in Appendix 5. They lead to the conclusion that the residuals follow normal distribution, as shown in Figure 7.13 - Figure 7.16.

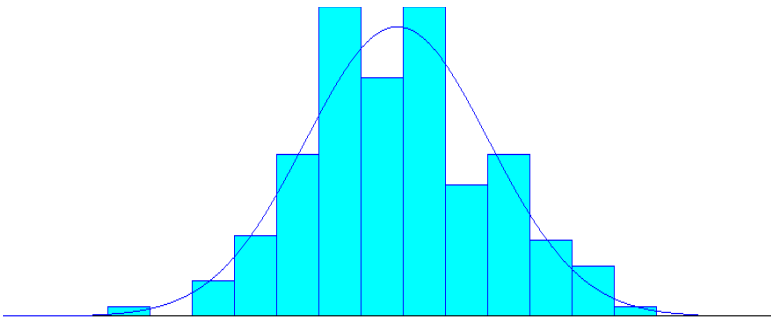


Figure 7.13 Distribution Fitting for GBP/USD One-Step Ahead Training Errors: Normal (-0.0001849, 0.00744)

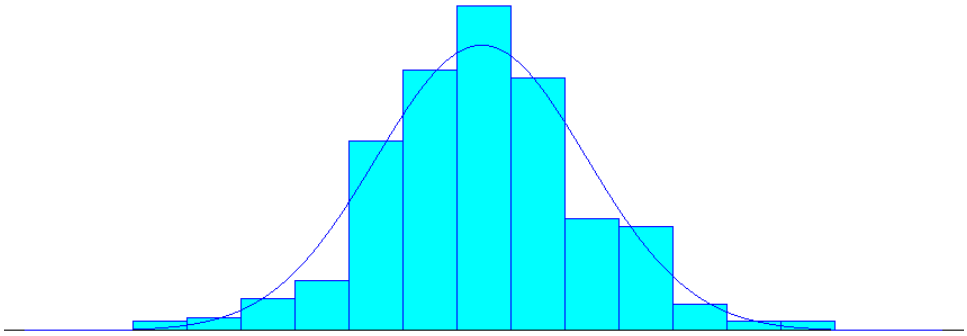


Figure 7.14 Distribution Fitting for GBP/USD Three-Step Ahead Training Errors: Normal (0.0000939, 0.0115)

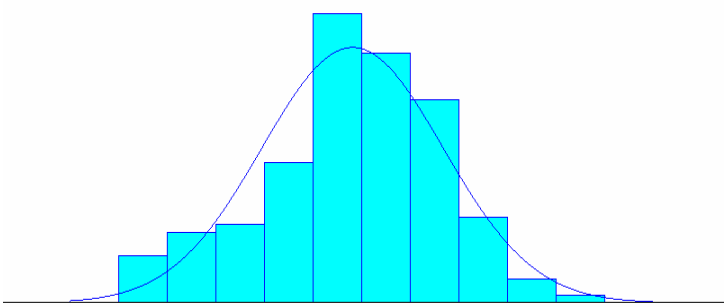


Figure 7.15 Distribution Fitting for EUR/USD One-Step Ahead Training Errors: Normal (-0.00185, 0.0186)

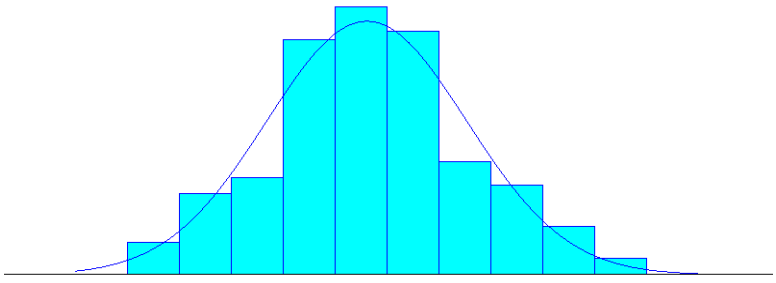


Figure 7.16 Distribution Fitting for EUR/USD Three-Step Ahead Training Errors: Normal (-0.000504, 0.0273)

The bootstrapping residual procedure for this application consists of four steps:

1. Determine the empirical distribution of the training data using bootstrap replications procedure. S-Plus® software was implemented for this purpose.
2. Generate B samples size n (n=number of training data) from the empirical distribution obtained. Again S-Plus software is used.
3. For each bootstrap sample, the network is trained and the predicted values for the testing data set are obtained.
4. Estimate the variance of the i^{th} bootstrap predicted value, where i=1,2,…. number of data in testing set.

The bootstrap replications of mean and variance for training errors for EUR/USD and GBP/USD obtained by setting the replications equal to 1,000 are graphically displayed in Figure 7.17 – Figure 7.20. For all problems, the normal probability plot of the mean for each case, i.e., Q-Q plots, indicates a desirable normality feature. The mean and standard error values of both the bootstrapped mean and the bootstrapped variance for each case are given in Table 7.7. The complete procedure to obtain the prediction interval of forecast is explained using EUR/USD three-step ahead as an example.

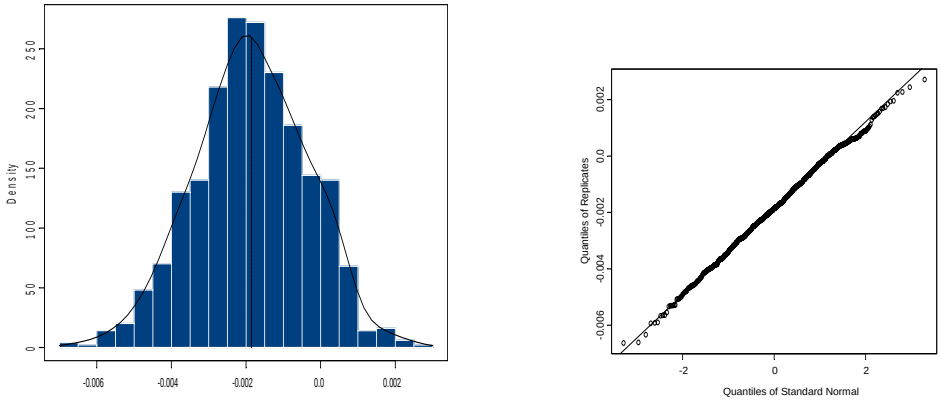


Figure 7.17a Bootstrap Replications of Mean for EUR/USD Training Errors

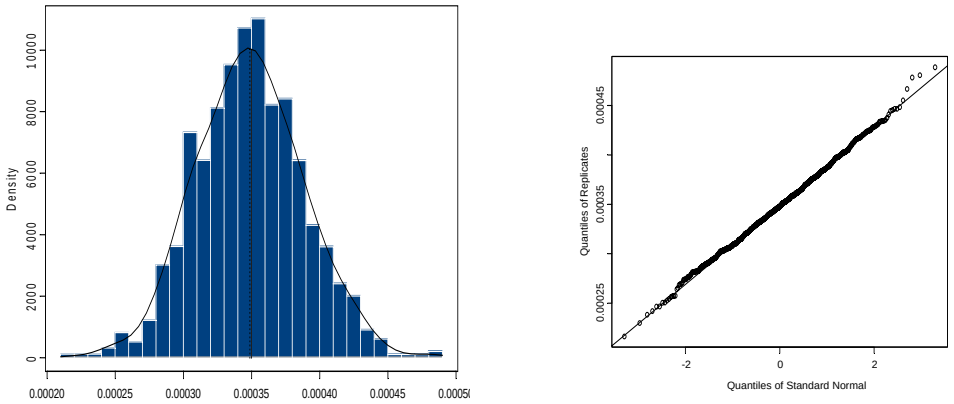


Figure 7.17b Bootstrap Replications of Variance for EUR/USD Training Errors

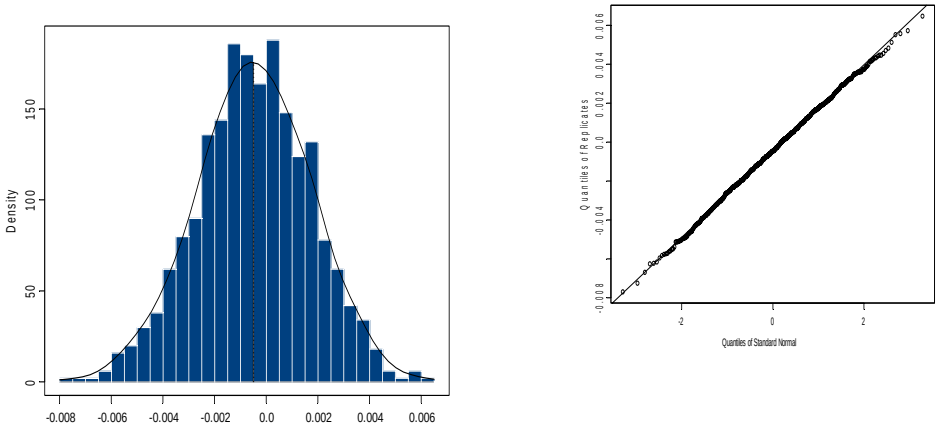


Figure 7.18a Bootstrap Replications of Mean for EUR/USD Three-Step Ahead Training Errors

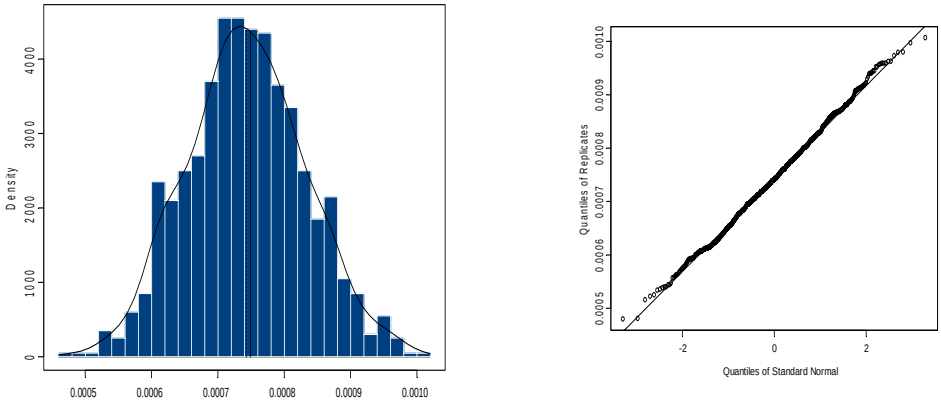


Figure 7.18b Bootstrap Replications of Variance for EUR/USD Three-Step Ahead Training Errors

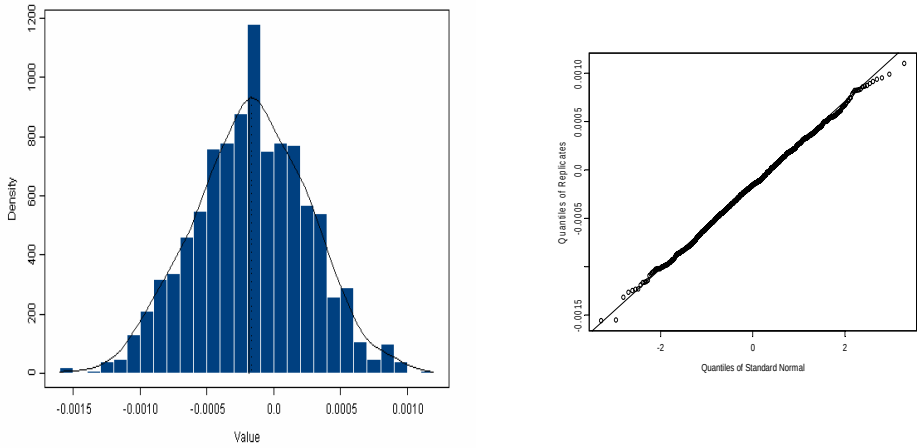


Figure 7.19a Bootstrap replications of mean for GBP/USD training errors

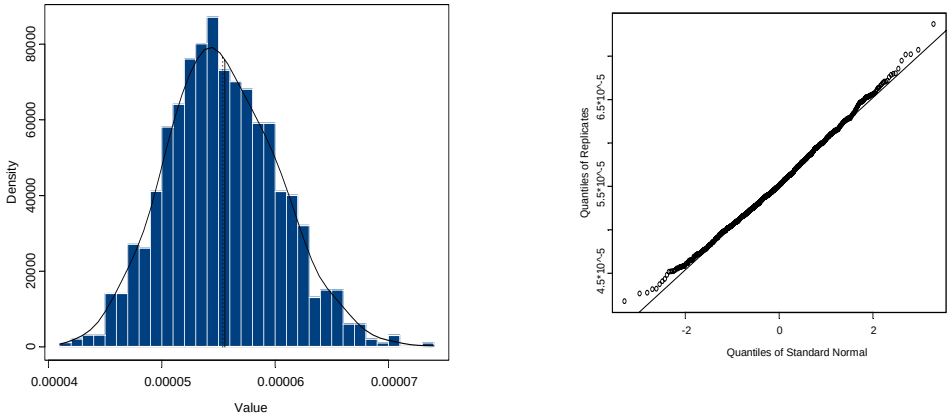


Figure 7.19b Bootstrap Replications of Variance for GBP/USD Training Errors

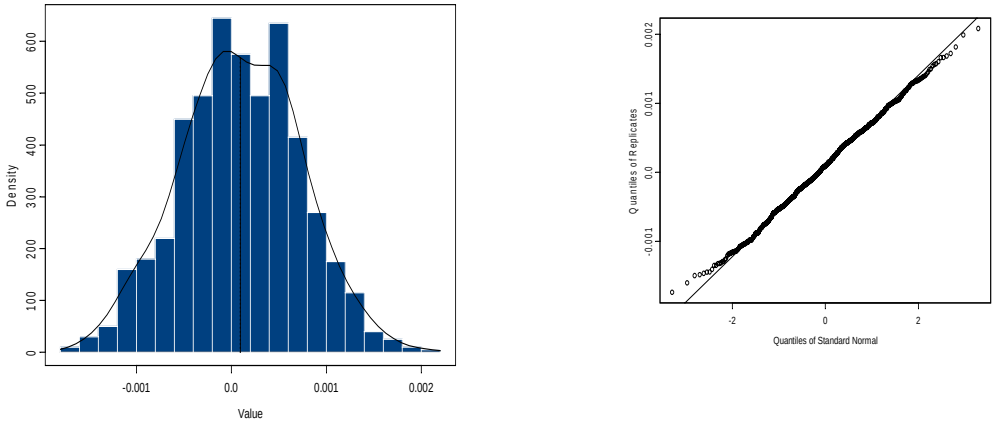


Figure 7.20a Bootstrap Replications of Mean for GBP/USD Three-Step Ahead Training Errors

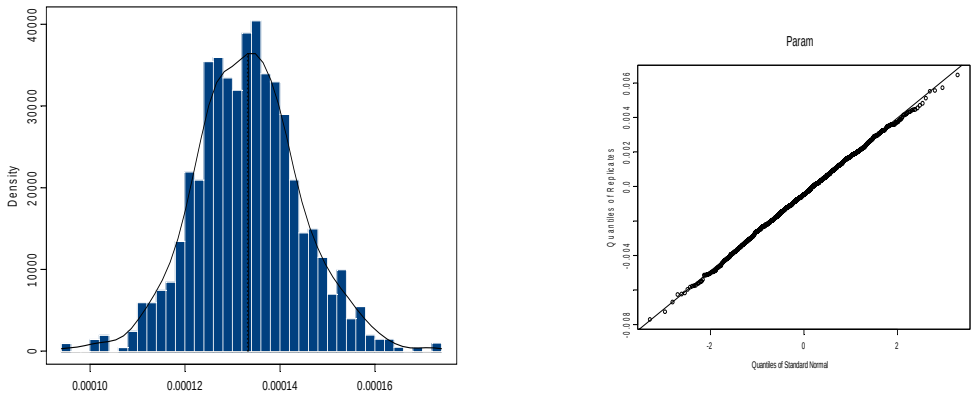


Figure 7.20b Bootstrap Replications of Variance for GBP/USD Three-Step Ahead Training Errors

Table 7.7 Bootstrap Replications for Empirical Distribution

		mean		var	
		mean	observed	mean	observed
EUR/USD	one step	-0.0018580	-0.0018530	0.0003488	0.0003499
EUR/USD	three steps	-0.0004867	-0.0005040	0.0007430	0.0007496
GBP/USD	one step	-0.0001654	-0.0001849	0.0000552	0.0000556
GBP/USD	three steps	0.0000920	0.0000939	0.0001332	0.0001334

Having known the empirical distribution for the training error, i.e., normal (mean= -0.0004867, variance= 0.000743), B bootstrap residual replicates of size n were generated. As mentioned previously, B is set equal to 20. S-Plus language was used for this purpose. By defining m as the number of bootstrap replicates, n as the number of training data, sample.mean and sample.var as mean and variance of the empirical distribution and repl.i is the destination cells for replicate # i (i=1,2…. m), the data frame name to retrieve the original data and save the result were named test.df and result.df, the code for this data set is as follows:

```
m <- 20
n <- 309
sample.mean <- 0.0004867
sample.var <- 0.000743
```

```
result.df <- test.df
for (i in 1:m) {
repl.i <- rnorm(n,mean=sample.mean,sd=sqrt(sample.var))
# names(repl.i) < paste("repl",i,sep="")
# names(repl.i)
# print(repl.i)
result.df <- data.frame(result.df, repl.i)
}
result.df
names(result.df)
Edit.data(result.df)
```

Next, the bootstrap replicates for training data were obtained. Figure 7.21 displays similarity in trends between the original training data set and two examples of bootstrap replicates. Each bootstrap replicate was then applied as the training data for EUR/USD three-step ahead to generate the predicted values of the testing data. The bootstrap replicates and the predicted values obtained are given in Table 7.7 and Table 7.8 respectively.

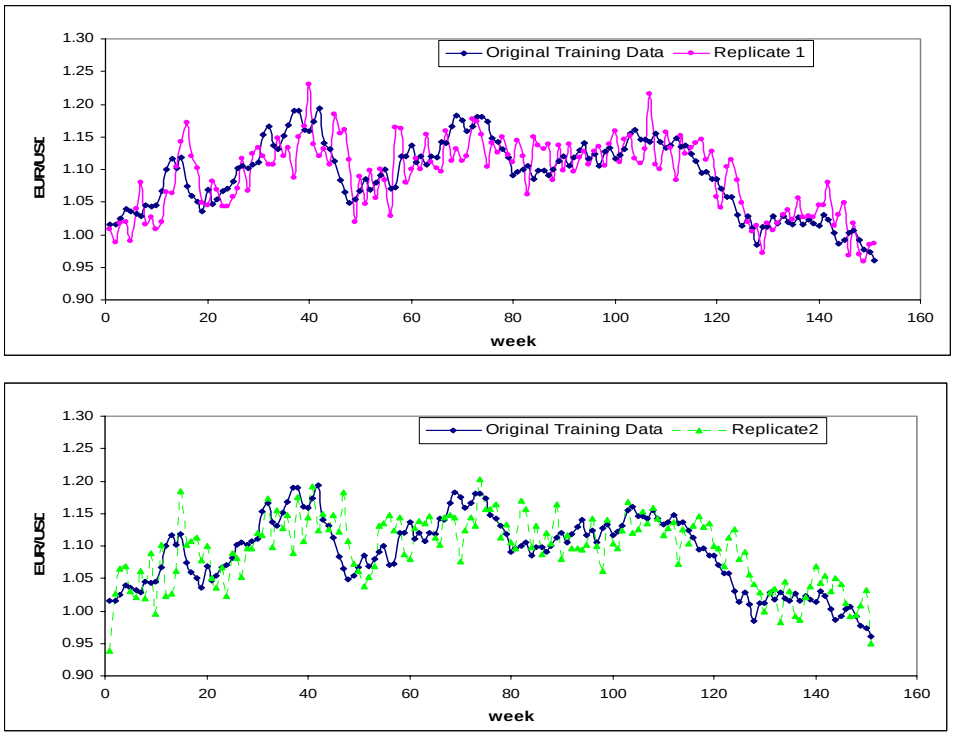
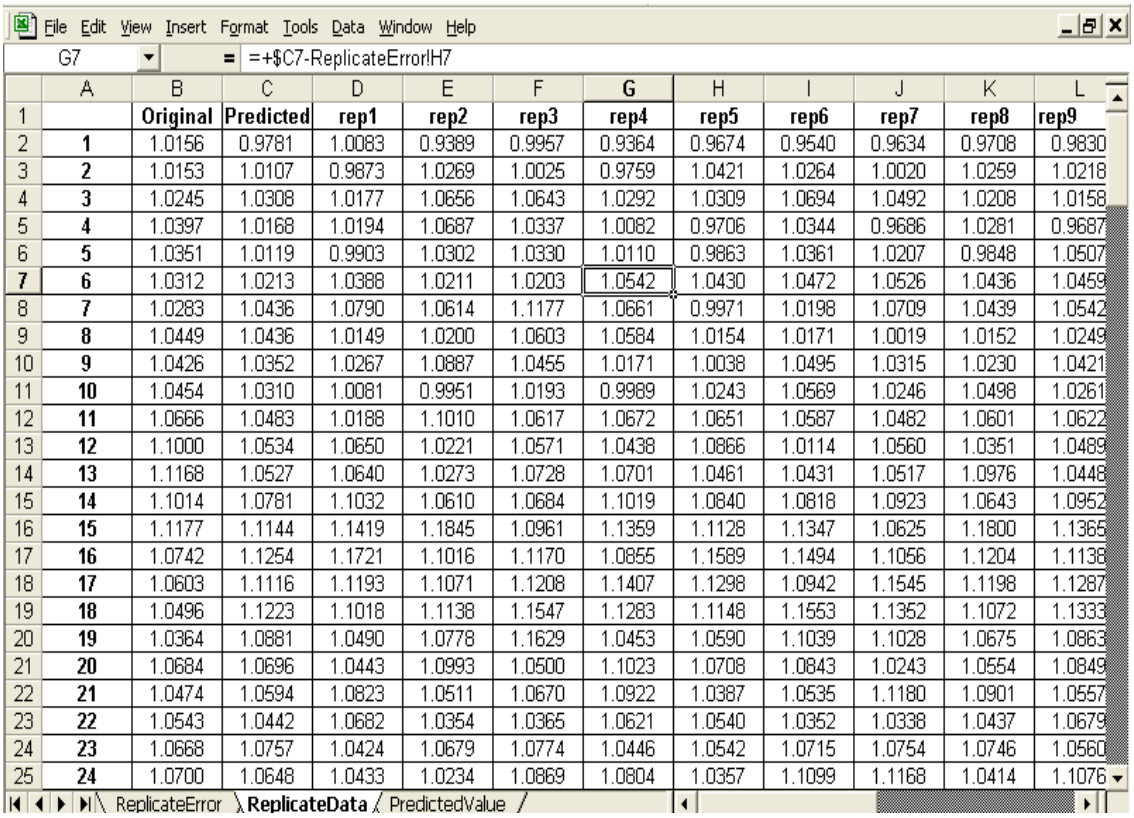


Figure 7.21 Original Training Data and Two Bootstrap Replicates

Table 7.8 Bootstrap Replicates of Training Data

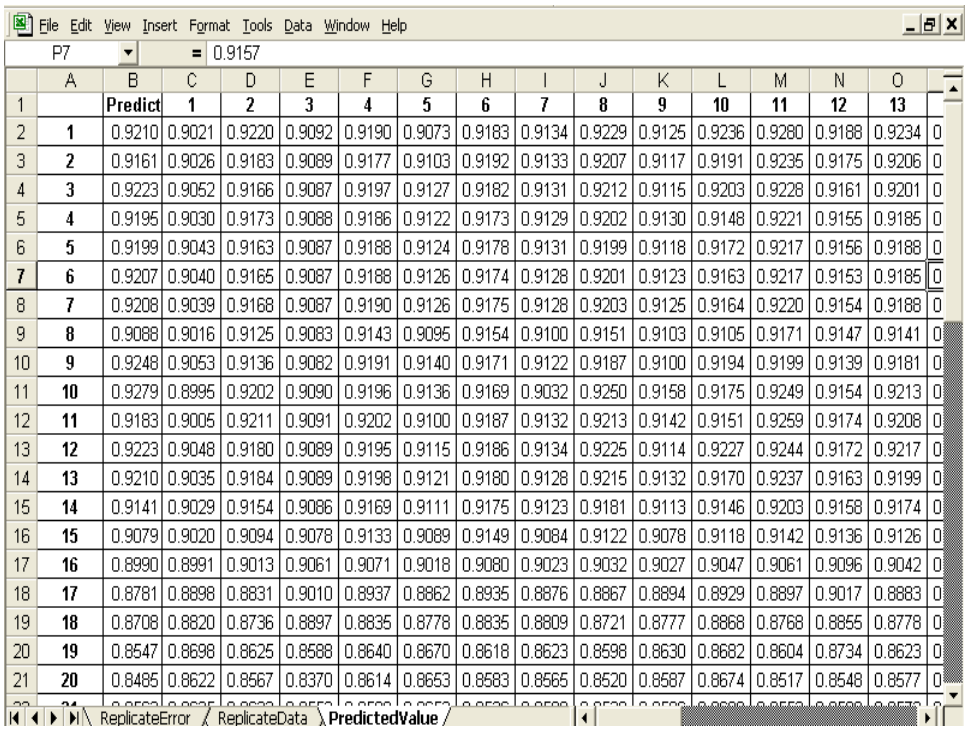


G7 = =+$C7-ReplicateError!H7

	A	B	C	D	E	F	G	H	I	J	K	L
1		Original	Predicted	rep1	rep2	rep3	rep4	rep5	rep6	rep7	rep8	rep9
2	1	1.0156	0.9781	1.0083	0.9389	0.9957	0.9364	0.9674	0.9540	0.9634	0.9708	0.9830
3	2	1.0153	1.0107	0.9873	1.0269	1.0025	0.9759	1.0421	1.0264	1.0020	1.0259	1.0218
4	3	1.0245	1.0308	1.0177	1.0656	1.0643	1.0292	1.0309	1.0694	1.0492	1.0208	1.0158
5	4	1.0397	1.0168	1.0194	1.0687	1.0337	1.0082	0.9706	1.0344	0.9686	1.0281	0.9687
6	5	1.0351	1.0119	0.9903	1.0302	1.0330	1.0110	0.9863	1.0361	1.0207	0.9848	1.0507
7	6	1.0312	1.0213	1.0388	1.0211	1.0203	1.0542	1.0430	1.0472	1.0526	1.0436	1.0459
8	7	1.0283	1.0436	1.0790	1.0614	1.1177	1.0661	0.9971	1.0198	1.0709	1.0439	1.0542
9	8	1.0449	1.0436	1.0149	1.0200	1.0603	1.0584	1.0154	1.0171	1.0019	1.0152	1.0249
10	9	1.0426	1.0352	1.0267	1.0887	1.0455	1.0171	1.0038	1.0495	1.0315	1.0230	1.0421
11	10	1.0454	1.0310	1.0081	0.9951	1.0193	0.9989	1.0243	1.0569	1.0246	1.0498	1.0261
12	11	1.0666	1.0483	1.0188	1.1010	1.0617	1.0672	1.0651	1.0587	1.0482	1.0601	1.0622
13	12	1.1000	1.0534	1.0650	1.0221	1.0571	1.0438	1.0866	1.0114	1.0560	1.0351	1.0489
14	13	1.1168	1.0527	1.0640	1.0273	1.0728	1.0701	1.0461	1.0431	1.0517	1.0976	1.0448
15	14	1.1014	1.0781	1.1032	1.0610	1.0684	1.1019	1.0840	1.0818	1.0923	1.0643	1.0952
16	15	1.1177	1.1144	1.1419	1.1845	1.0961	1.1359	1.1128	1.1347	1.0625	1.1800	1.1365
17	16	1.0742	1.1254	1.1721	1.1016	1.1170	1.0855	1.1589	1.1494	1.1056	1.1204	1.1138
18	17	1.0603	1.1116	1.1193	1.1071	1.1208	1.1407	1.1298	1.0942	1.1545	1.1198	1.1287
19	18	1.0496	1.1223	1.1018	1.1138	1.1547	1.1283	1.1148	1.1553	1.1352	1.1072	1.1333
20	19	1.0364	1.0881	1.0490	1.0778	1.1629	1.0453	1.0590	1.1039	1.1028	1.0675	1.0863
21	20	1.0684	1.0696	1.0443	1.0993	1.0500	1.1023	1.0708	1.0843	1.0243	1.0554	1.0849
22	21	1.0474	1.0594	1.0823	1.0511	1.0670	1.0922	1.0387	1.0535	1.1180	1.0901	1.0557
23	22	1.0543	1.0442	1.0682	1.0354	1.0365	1.0621	1.0540	1.0352	1.0338	1.0437	1.0679
24	23	1.0668	1.0757	1.0424	1.0679	1.0774	1.0446	1.0542	1.0715	1.0754	1.0746	1.0560
25	24	1.0700	1.0648	1.0433	1.0234	1.0869	1.0804	1.0357	1.1099	1.1168	1.0414	1.1076

ReplicateError | ReplicateData | PredictedValue

Table 7.9 Predicted Values for The Corresponding Bootstrap Replicate



P7 = 0.9157

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	
1		Predict	1	2	3	4	5	6	7	8	9	10	11	12	13	
2	1	0.9210	0.9021	0.9220	0.9092	0.9190	0.9073	0.9183	0.9134	0.9229	0.9125	0.9236	0.9280	0.9188	0.9234	0
3	2	0.9161	0.9026	0.9183	0.9089	0.9177	0.9103	0.9192	0.9133	0.9207	0.9117	0.9191	0.9235	0.9175	0.9206	0
4	3	0.9223	0.9052	0.9166	0.9087	0.9197	0.9127	0.9182	0.9131	0.9212	0.9115	0.9203	0.9228	0.9161	0.9201	0
5	4	0.9195	0.9030	0.9173	0.9088	0.9186	0.9122	0.9173	0.9129	0.9202	0.9130	0.9148	0.9221	0.9155	0.9185	0
6	5	0.9199	0.9043	0.9163	0.9087	0.9188	0.9124	0.9178	0.9131	0.9199	0.9118	0.9172	0.9217	0.9156	0.9188	0
7	6	0.9207	0.9040	0.9165	0.9087	0.9188	0.9126	0.9174	0.9128	0.9201	0.9123	0.9163	0.9217	0.9153	0.9185	0
8	7	0.9208	0.9039	0.9168	0.9087	0.9190	0.9126	0.9175	0.9128	0.9203	0.9125	0.9164	0.9220	0.9154	0.9188	0
9	8	0.9088	0.9016	0.9125	0.9083	0.9143	0.9095	0.9154	0.9100	0.9151	0.9103	0.9105	0.9171	0.9147	0.9141	0
10	9	0.9248	0.9053	0.9136	0.9082	0.9191	0.9140	0.9171	0.9122	0.9187	0.9100	0.9194	0.9199	0.9139	0.9181	0
11	10	0.9279	0.8995	0.9202	0.9090	0.9196	0.9136	0.9169	0.9032	0.9250	0.9158	0.9175	0.9249	0.9154	0.9213	0
12	11	0.9183	0.9005	0.9211	0.9091	0.9202	0.9100	0.9187	0.9132	0.9213	0.9142	0.9151	0.9259	0.9174	0.9208	0
13	12	0.9223	0.9048	0.9180	0.9089	0.9195	0.9115	0.9186	0.9134	0.9225	0.9114	0.9227	0.9244	0.9172	0.9217	0
14	13	0.9210	0.9035	0.9184	0.9089	0.9198	0.9121	0.9180	0.9128	0.9215	0.9132	0.9170	0.9237	0.9163	0.9199	0
15	14	0.9141	0.9029	0.9154	0.9086	0.9169	0.9111	0.9175	0.9123	0.9181	0.9113	0.9146	0.9203	0.9158	0.9174	0
16	15	0.9079	0.9020	0.9094	0.9078	0.9133	0.9089	0.9149	0.9084	0.9122	0.9078	0.9118	0.9142	0.9136	0.9126	0
17	16	0.8990	0.8991	0.9013	0.9061	0.9071	0.9018	0.9080	0.9023	0.9032	0.9027	0.9047	0.9061	0.9096	0.9042	0
18	17	0.8781	0.8898	0.8831	0.9010	0.8937	0.8862	0.8935	0.8876	0.8867	0.8894	0.8929	0.8897	0.9017	0.8883	0
19	18	0.8708	0.8820	0.8736	0.8897	0.8835	0.8778	0.8835	0.8809	0.8721	0.8777	0.8868	0.8768	0.8855	0.8778	0
20	19	0.8547	0.8698	0.8625	0.8588	0.8640	0.8670	0.8618	0.8623	0.8598	0.8630	0.8682	0.8604	0.8734	0.8623	0
21	20	0.8485	0.8622	0.8567	0.8370	0.8614	0.8653	0.8583	0.8565	0.8520	0.8587	0.8674	0.8517	0.8548	0.8577	0

ReplicateError | ReplicateData | PredictedValue

The forecast variance was estimated by Deng (2002) as:

$$\sigma^2 = \sigma_\varepsilon{}^2 + \tfrac{1}{B}\sum[y(x_i; S^{\wedge *b}) - \bar{y}(x_i)]^2$$

where $\bar{y}(x_i) = \sum_0^B y(x_i; \hat{S}^{*b})/(B+1)$;

$y(x_i; S^{\wedge *b})$ = predicted value of observation i obtained using bootstrap replication b as training data

while the prediction interval was estimated as:

$$\hat{y}_i \pm t_{\alpha/2,n-1}\sqrt{\frac{1}{B}\sum_0^B [y(x_i; \hat{S}^{*b}) - \bar{y}(x_i)]^2 + \sigma_\varepsilon{}^2}$$

where B = number bootstrapping replications

σ^2 = forecast variance

$\sigma_\varepsilon{}^2$ = process variance

Figure 7.22a and Figure 7.23b show the graph of the 95% and 90% prediction intervals, respectively, for the forecasts calculated from the above formulas under the assumption that the process variation is the same as the training stage, which is $\sigma_\varepsilon{}^2$ = 0.0007453 for EUR/USD three-step ahead.

The same procedure is implemented for GBP/USD one-step and three-step ahead forecasting to obtain the prediction intervals. The predicted values of the testing data for each replicate are depicted in Appendix 6. Figure 7.23a shows the graph of the 95% prediction intervals for the GBP/USD one-step ahead forecasts calculated from the previously mentioned formulas under the assumption that the process variation is the

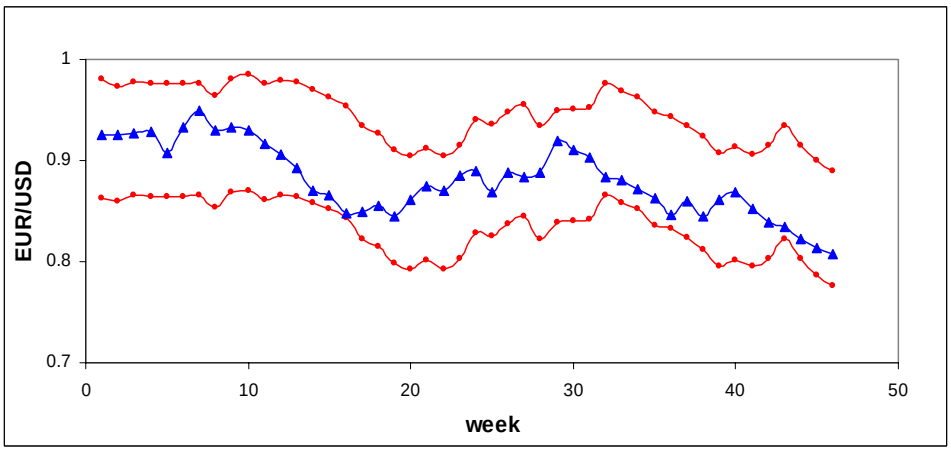


Figure 7.22a 95% Prediction Interval and The Actual Values of EUR/USD for Three-Step Ahead Forecasting

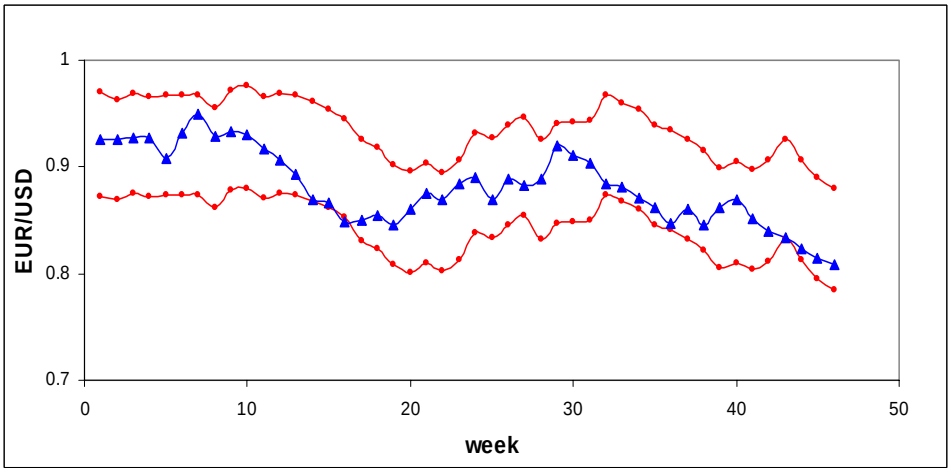


Figure 7.22b 90% Prediction Interval and The Actual Values of EUR/USD for Three-Step Ahead Forecasting

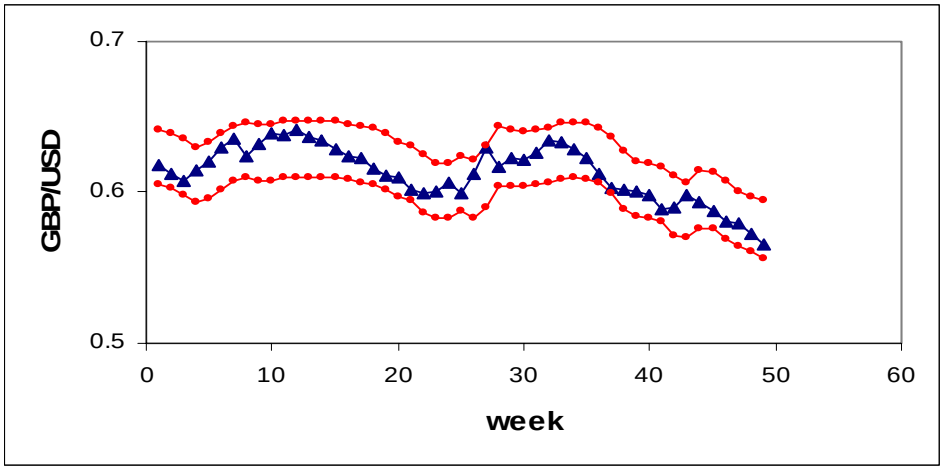


Figure 7.23a 95% Prediction Interval and The Actual Values of GBP/USD for One-Step Ahead Forecasting

same as the training stage; Figure 7.23b shows the graph of the 90% prediction intervals for the same data set.

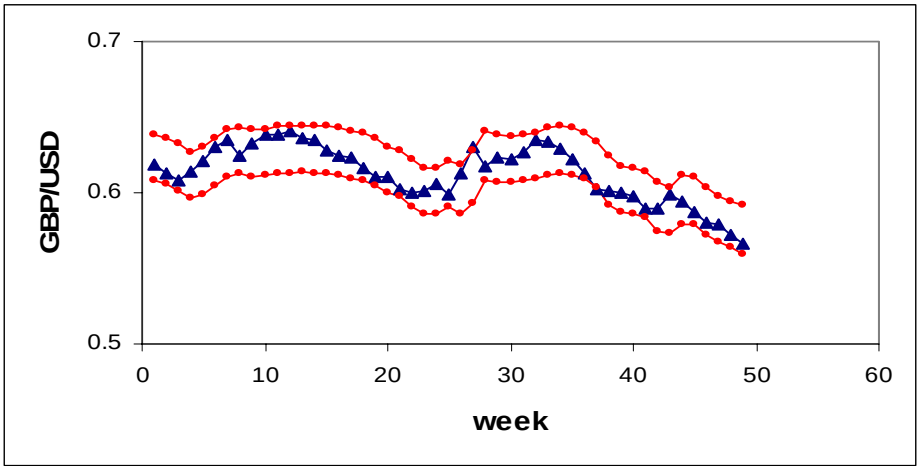


Figure 7.23b 90% Prediction Interval and The Actual Values of GBP/USD for One-Step Ahead Forecasting

Figure 7.24a and Figure 7.24 b show the graph of the 95% and the 90% prediction intervals, respectively, for the GBP/USD three-step ahead forecasts calculated from the previously mentioned formulas under the assumption that the process variation is the same as the training stage.

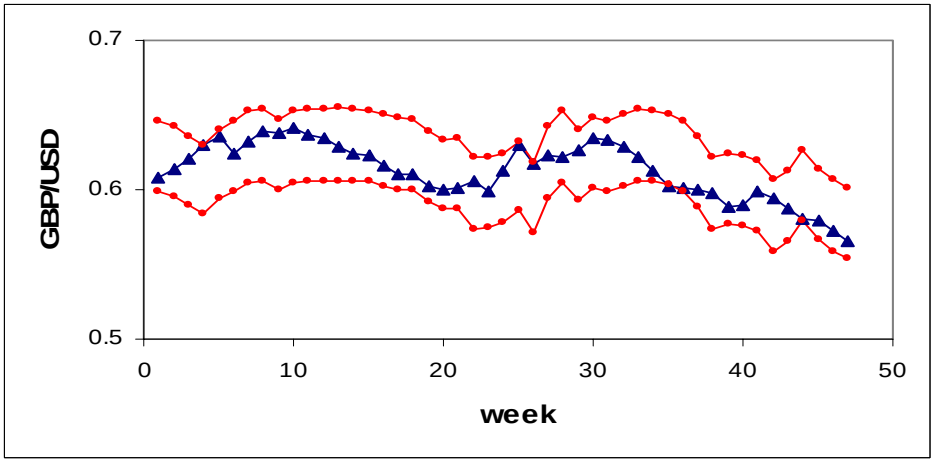


Figure 7.24a 95% Prediction Interval and The Actual Values of GBP/USD for Three-Step Ahead Forecasting

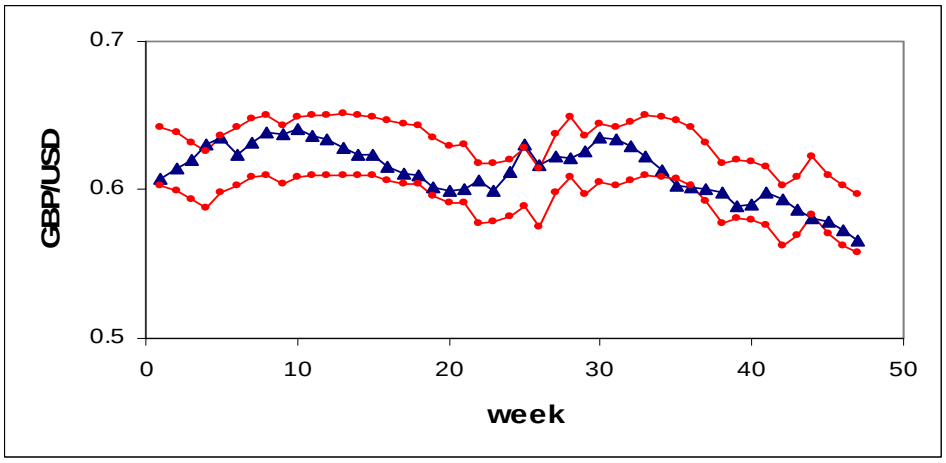


Figure 7.24b 90% Prediction Interval and The Actual Values of GBP/USD for Three-Step Ahead Forecasting

As discussed previously, the error for multiple ahead forecasting is usually greater than the one for one-step ahead forecasting. Consequently, the confidence interval for three-step ahead forecasting is wider than the one for one-step ahead forecasting. The 90% confidence interval of GBP/USD for three-step ahead forecasting is not as tight as the one for one-step ahead forecasting. The 95% confidence interval for GBP/USD for one- and three- step ahead forecasts demonstrate the same outcome.

For one-step ahead forecasting, it shows that all prediction points are within the 95% prediction interval, and only one point is out of the 90% prediction interval limits. While for the three-step ahead forecasting, there is one point outside the 95% prediction interval, and three points out of the 90% limits.

Chapter 8

Multivariate Time Series Forecasting

Most of the studies in time series forecasting have been concerned with univariate time series. It is important to note that the univariate approach can fail for several reasons, such as: (1) The market could be efficient and only driven from outside indicators; and (2) The available time series are too short for a significant technical analysis with the chosen forecasting horizon. On the other hand, multivariate time series forecasting integrates some fundamental forces that allow for forecasting financial markets under difficult circumstances (Ankembrand and Tomassini, 1996).

Multivariate time series analysis is an important statistical tool to study the behavior of time dependent data and forecast future values depending on the history of variations in the data. By studying many related variables together, a better understanding is often obtained. A multivariate time series consists of sequences of values of several contemporaneous variables changing with time.

Previous researchers (Yao and Tan, 2000) suggested that univariate time series do not provide good forecasts for the JPY/USD exchange rate, because of their poor forecasting performances. This research showed a similar problem with EUR/USD. Therefore, multivariate time series forecasting was investigated for these two foreign exchanges. It was expected that an appropriate NN multivariate time series forecasting model would perform better than a univariate time series model.

At first, the effects of economic forces, such as the net difference between the value of merchandise being imported and exported into a particular country, the flow of

funds between countries to pay for stocks and bonds purchased, the relative inflation rates, and the relative interest rates that influence exchange rates were planned to be examined. It's unfortunate that many variables that may have effect on foreign exchange can not be used in this case study. The main problem is the difference in the frequency of the data.

None was available for weekly periods, which was the case study in this research. Most of them are available quarterly, and inflation rates are available monthly, while interest rates are available either daily or monthly. Weekly interest rates, the average value of interest during one week, were adopted for this research.

8.1 Case Study 1: JPY/USD

As stated before, the only economic forces obtained to conduct a multivariate analysis were interest rates for Japan and the U.S. For predicting weekly exchange rates of JPY/USD, besides this time series itself, there are four other time series investigated: EUR/USD, Japan interest rates, U.S. interest rates, and relative interest rates of Japan and the U.S. All of those variables seemed not to have an effect on foreign exchange of JPY/USD. Correlation tests showed that among all above variables, there was no significant correlation between the two variables as given in Table 8.1. Correlations measure the relationship between two data sets that are scaled to be independent of the unit of measurement. The population correlation calculation returns the covariance of two data sets divided by the product of their standard deviations as given in Formula 8-1.

$$\rho_{x,y} = \frac{Cov(x,y)}{\sigma_x . \sigma_y} \tag{8-1}$$

These results indicate that multivariate time series forecasting may not be useful in this case. A small experiment was conducted to illustrate the problem and the results are depicted in Table 8.2. It is clear that there was no significant improvement achieved by involving other variables to predict the weekly exchange rate of JPY/USD. The phenomenon was seen by many economists. Up to date, economists do not possess reliable methods of forecasting exchange rates over short time horizons, such as days or weeks. Economic forces usually take much longer, often several years, to have an effect on exchange rates. In view of that, one should study not only the window inputs but also the lag periods to determine the appropriate inputs of the economic forces for multivariate time series forecasting.

Table 8.1 Correlation Coefficients for Weekly JPY/USD

	JPY/USD
JP/US int	-0.288
US int	-0.369
JP int	-0.329
EUR/USD	-0.169

Table 8.2 Performances of Trivariate Time Series Forecasting of Weekly JPY/USD

n1	n2	n3	h nodes	l rate	m term	MSE	MAPE	DIR
5	1	1	3	0.27	0.89	10.223	2.445	52.17
5	*0*	*0*	*3*	*0.27*	*0.89*	*5.971*	*1.659*	*53.80*
3	1	1	1	0.88	0.02	23.598	3.964	52.08
3	*0*	*0*	*1*	*0.88*	*0.02*	*3.982*	*1.471*	*54.17*
3	1	3	7	0.90	0.02	21.074	3.187	52.78
2	*0*	*0*	*12*	*0.9*	*0.02*	*57.263*	*4.533*	*61.90*

n_1= input window for JPY/USD; n_2= input window for US interest data; n_3=input window for JP interest data

It is difficult to do the multivariate analysis if one considers weekly exchange periods, hence monthly exchange rates were used to study this topic. For predicting

monthly exchange rates of JPY/USD, besides this time series itself, there are five other time series investigated: EUR/USD, Japan interest rates, US interest rates, and relative interest rates of Japan and the U.S., and U.S. inflation rates. Among those, only two variables, i.e., the U.S. interest rate and Japan's interest rate, seem to have effects on foreign exchange of JPY/USD. As it has been tested, the correlation between the two variables exist, where the correlation coefficient between JPY/USD and Japan interest rate is -0.687, and the correlation coefficient between JPY/USD and the U.S. interest rate is –0.696, as given in Table 8.3. In view of that, trivariate time series were analyzed for predicting JPY/USD, involving three variables: JPY/USD, Japan's Interest rate, and the U.S. interest rate.

Table 8.3 Correlation Coefficients for Monthly EUR/USD and JPY/USD

	EUR/USD		***JPY/USD***
EUR/US int	0.099	**JP/US int**	-0.076
US int	0.367	**US int**	**-0.696**
EUR int	**0.847**	**JP int**	**-0.687**
JPY/USD	0.256	**EUR/USD**	0.256

A trivariate time series was conducted using monthly time-delayed foreign exchange data and interest rates for the period 1997-2003. Six-year data was used as the training set, while the remaining data was used as the testing set.

Five topologies were studied. The first involved three time series with the same periods as suggested by Setyawati et al. (2003) in which two time-delayed data from each time series were fed to the NN after normalization of the data. The number of hidden nodes was set equal to two. In the second topology, three time-delayed from JPY/USD,

and one time-delayed data from each of the two other time series were fed to the NN with three hidden nodes. The third and fourth scenarios are the variants of the topology obtained using GA. In these, two time-delayed from JPY/USD, and one time-delayed data from each of the two other time series were fed to the NN with one and two hidden nodes respectively. The fifth topology was obtained using GA for Multivariate Topology Determination, where NH was set equal to 20 and NS = 10. The pseudo code of this algorithm is provided in Appendix 7. Each topology was trained for 1,000 iterations.

Figure 8.2 illustrates how to apply the sliding window technique when three time series are used as the input data as opposed to a simpler method when only one time series data is involved (Figure 8.1). Suppose there are three time series data: X, Y and Z as the input data, and one considers using a time window of three for the first time series and one for each of the remaining series. Assuming the current time is t, and the data inputted to the NN for forecasting the next period are X, X_{t-1}, X_{t-2}, Y_t, Z_t. The pseudo code of this algorithm is given in Appendix 7, and the results are depicted in Table 8.4 in which the benchmark is given in the last row.

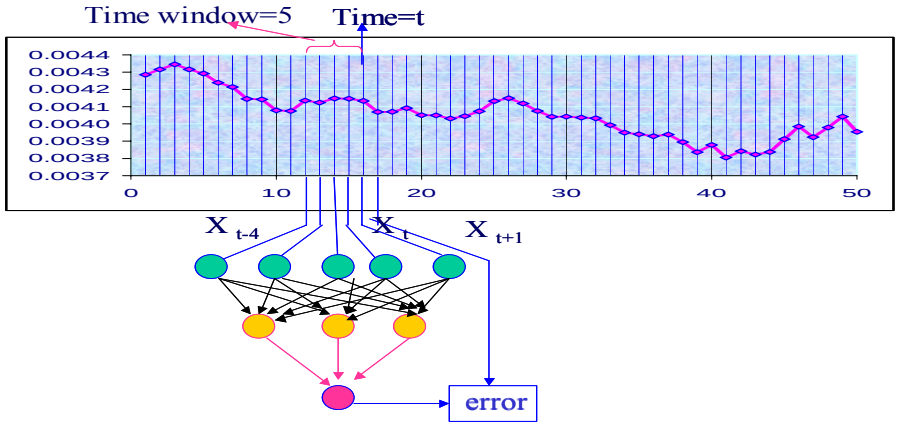


Figure 8.1 Univariate Time Series Forecasting Using Three-Layer Feed Forward Neural Network with Five Inputs and Three Hidden Nodes

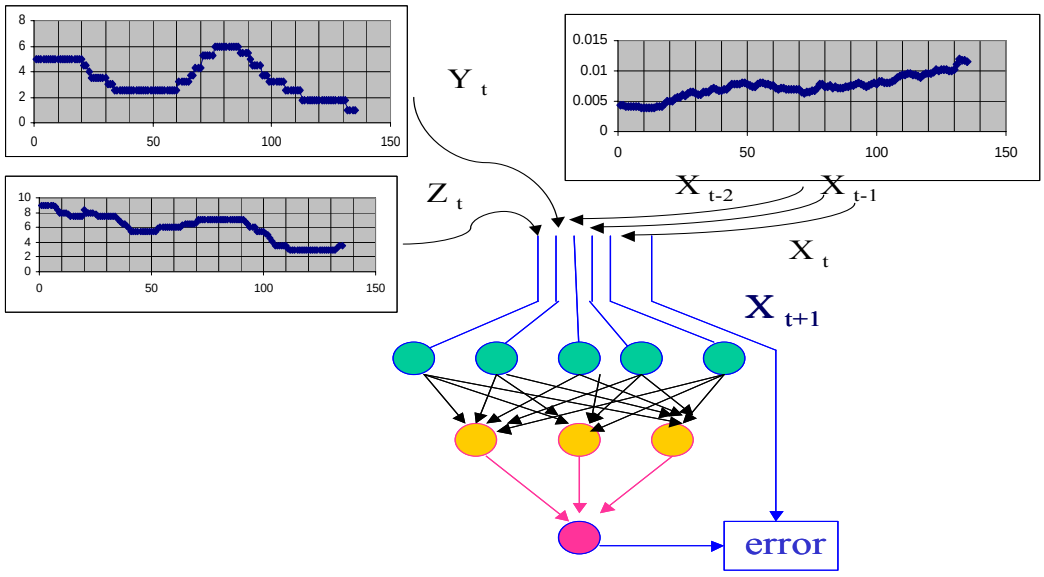


Figure 8.2 Trivariate Time Series Forecasting Using Three-Layer Feed Forward Neural Network with 5 Inputs (3-1-1) and 3 Hidden Nodes

Table 8.4 Performances of Trivariate Time Series Forecasting of Monthly JPY/USD

n1	n2	n3	h nodes	l rate	m term	MSE	MAPE	DIR
2	2	2	2	0.30	0.30	16.577	2.937	50.00
3	1	1	3	0.30	0.30	20.570	3.246	57.77
2	1	1	2	0.30	0.30	15.611	2.812	56.00
2	1	1	1	0.88	0.02	10.265	2.332	60.00
1	1	3	2	0.90	0.02	9.235	2.129	57.78
2	*2*	*2*	*2*	*0.30*	*0.30*	*N/A*	*N/A*	*54.00*

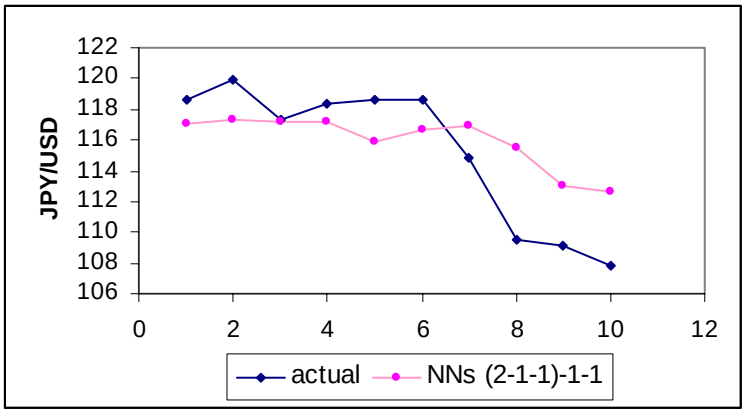


Figure 8.3 Trivariate Time Series Forecasts for Monthly JPY/USD

The best result was obtained from the (2-1-1)-1-1 NN model. The result is better than the benchmark that reported an average gradient of 54% for their model. To be able to compare trivariate time series and univariate time series for the chosen model, the performance of the corresponding monthly JPY/USD time-delayed input were looked into. The results suggested that the trivariate time series did perform better than monthly time-delayed inputs with the same number of input for the corresponding time series and the same number of hidden nodes in terms of directional changes, as illustrated in Table 8.5. It is indicated that the additional time series enables one to improve the directional change performances without significant changes on the performances of MSE and MAPE.

Table 8.5 Comparison between Trivariate and Univariate Time Series Forecasting

n1	n2	n3	h nodes	l rate	m term	MSE	MAPE	DIR	Note
2	*1*	*1*	*2*	*0.30*	*0.30*	*15.611*	*2.812*	*56.00*	*t*
2	0	0	2	0.30	0.30	14.488	2.782	50.00	u
3	*1*	*1*	*3*	*0.30*	*0.30*	*20.570*	*3.246*	*57.77*	*t*
3	0	0	3	0.30	0.30	19.914	3.135	53.34	u
2	*1*	*1*	*1*	*0.88*	*0.02*	*10.265*	*2.332*	*60.00*	*t*
2	0	0	1	0.88	0.02	9.412	2.139	54.00	u
1	*1*	*3*	*2*	*0.90*	*0.02*	*9.235*	*2.129*	*57.78*	*t*
1	0	0	2	0.90	0.02	7.520	1.947	45.45	u

t =trivariate, u= univariate

The algorithm from the previous study did not produce the same level of success as the one achieved in this experiment. Setyawati et al. (2003) argued that in spite of the improved performances they achieved, the gradients from either monthly time-delayed data or trivariate time series are not sufficiently adequate to be accepted by the practitioners. On the contrary, the (2-1-1)-1-1 model performed better than the

benchmark and reached the 60% mark with the MSE and MAPE values much lower than the benchmark model ((2-2-2)-2-1).

8.2 Case Study 2: EUR/USD

For predicting the monthly exchange rate of EUR/USD, besides this time series itself, there are five other time series investigated: the monthly exchange rate of JPY/USD, the Euro monthly interest rate, the U.S. monthly interest rate, and the relative interest rates of Euro and U.S. The only variable that may have effect on foreign exchange of EUR/USD is the Euro's interest rate. The correlation between these two variables exists with the correlation coefficient of 0.847 as shown in Table 8.3.

A trivariate time series was analyzed for predicting EUR/USD. It was predicted that the third variable, i.e. US interest rate would have no role in improving monthly forecasts of EUR/USD.

The following four unit experiments were performed in this study. The first two are the scenarios used by Setyawati (2003) to conduct multivariate time series forecasting, while the last two were GA-based topology obtained from a Genetic Algorithm for Multivariate Topology Determination. The pseudo code of this algorithm is given in Appendix 7. The four scenarios are as follows: (1) The foreign exchange value of EUR/USD for period t+1 was predicted using 2 preceding values (recorded at time points t1, and t-1 only) from the three series, i.e. x_t, x_{t-1}, y_t, y_{t-1}, z_t, z_{t-1}, (2) The time window of three was used for EUR/USD time series data, and one was used for each of the other two time series data, (3) Two inputs were obtained from EUR/USD time series data, and one input was obtained from each of the other two time series data, (4) The time

window of two for EUR/USD time series data, along with three input data from each of the other two time series data, was used to conduct the multivariate time series forecasting for EUR/USD. The results are depicted in Table 8.6.

Table 8.6 Performances of Trivariate Time Series Forecasting of Monthly EUR/USD

n1	n2	n3	h nodes	l rate	m term	MSE	MAPE	DIR
2	2	2	2	0.3	0.3	0.001	2.710	50.00
3	1	1	3	0.3	0.3	0.001	3.105	55.55
2	1	1	3	1	0.54	0.001	2.608	60.00
2	3	3	6	0.9	0.02	0.001	2.873	66.67

The results show that the multivariate time series forecasting provides good forecasts. The topology obtained using GA for Multivariate Topology Determination by setting the learning rate=0.90 and momentum term=0.2 for NH=20 and NS=10, i.e., (2-3-3)-6-1 model performed better than others in DIR and reached beyond the 60% marked.

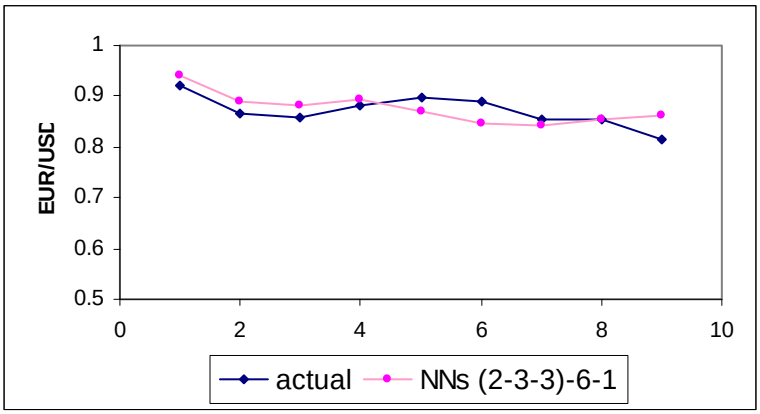


Figure 8.4 Trivariate Time Series Forecasts for Monthly EUR/USD

The results show that the multivariate time series forecasting provides good forecasts. The topology obtained using GA for Multivariate Topology Determination by

setting the learning rate=0.90 and momentum term=0.2 for NH=20 and NS=10, i.e., (2-3-3)-6-1 model performed better than others in DIR and reached beyond the 60% marked.

However, the chosen topology is not as predicted. The best topology suggested using GA employed 3 inputs from U.S. Interest time series data, despite the low correlation between this data and EUR/USD. This may be happen because of the existing correlation between U.S. interest and EUR interest. Moreover, in spite of the correlation coefficient, both U.S. interest data and EUR interest data possess positive correlations with EUR/USD foreign exchange rate data — that is, large values of EUR interest or larger values of U.S. interest are associated with large values of EUR/USD. As a result, U.S. interest time series data did improve the quality of EUR/USD exchange rate forecasts.

Chapter 9

Conclusions and Future Research Recommendations

9.1 Summary of Research

When applying an NN model to a real application, attention should be taken in every step, including the determination of the topology and the training algorithm employed. Many researchers currently use BP algorithms that converge locally as NN training. A possible solution to this local convergence dilemma is either the genetic algorithm or the tabu search algorithm. This art, especially in terms of configuring the topology and training NNs, was simplified through the use of GAs and TS in this research. Furthermore, the application of GA and TS for topology determination will eliminate the tedious trial and error procedures.

The study showed that there are at least three parameters that should be included in the topology determination model: the number of input nodes, the number of hidden nodes, and the learning rate. It was indicated that the number of input nodes is the most important issue. Besides that, there was an interaction between the number of hidden nodes and the number of input nodes. Moreover, the hyperbolic tangent activation function seems more appropriate compared to sigmoid function

The comparison performance of topology obtained using GA, TS, and benchmark led to the conclusion that neither GA nor TS guarantees better results, especially in terms of DIR. However, if there is no prior knowledge of the problem, GA searches for topology determination are favorable and provide reasonably good performance. GA is preferred over TS for foreign exchange forecasting, since GA prevails over TS in terms of MSE, MAPE, and DIR in this case study. The topology suggested by GA is indeed

better in terms of MAPE for all cases, and generally performed better in terms of MSE, but failed to show superiority in terms of DIR. However, the DIR for AUS/USD, EUR/USD, CHF/USD and GBP/USD were all above 60%.

Compared to BP and TS, GA is preferred for NN training. The GA guarantees better performance than BP in terms of MAPE and most of the time performs better in terms of MSE. It should be noted that there is no uncertainty in the relative performance of DIR. Additionally, GA is quite competitive and much easier to use.

The study on time period influences on NNs for Time Series Forecasting performances indicated that the best topology to forecast a certain foreign exchange is "data specific"; hence, the best approach for a certain period is not always the best to forecast other periods, especially in terms of directional changes. Case studies for GBP/USD suggested there is a correlation between the number of turning points in the test data set and the true directional change predictions. The two worst performances, i.e., for period five and eight, happen to be the periods with the highest number of turning points. Therefore, caution must be used in determining the correct topology, because it is problem specific.

Results on multi-step ahead forecasting for EUR/USD and GBP/USD indicated that given a certain topology chosen based on one-step ahead forecasting, the MSE and MAPE increased as the number of multi-step ahead forecasting increases. There was no clear pattern on the DIR behavior as it seemed reasonably well for up to three-step ahead forecasts. It also is apparent that the (1- α) confidence interval for three-step ahead forecasting is not as tight as the one for one-step ahead forecasting for both cases.

Multivariate Time Series forecasts for monthly JPY/USD, as well as for monthly EUR/USD, produced a higher level of success than that achieved previously (Setyawati et al., 2003). The (2-1-1)-1-1 model for JPY/USD performed better than the benchmark and reach the 60% mark with the MSE and MAPE values much lower than the benchmark model ((2-2-2)-2-1). The (2-3-3)-6-1 model for EUR/USD performed well and reached beyond the 60% mark.

9.2 Contributions of Research

The contributions of this research are:

(1) A thorough experiment about the effects of important factors in NNs for time series forecasting;

(2) A comparison of global search techniques for NN topology optimization and training using real time series data; this is the first time comparison of BP, GA, and TS used real financial time series data;

(3) An NN for foreign exchange multivariate time series forecasting was implemented; and

(4) Multi-step ahead time series forecasting was incorporated in the model.

9.3 Recommendations for Future Research

In addition to using pure time series, the inputs to NNs can also include some technical indicators, such as moving average, one of the oldest and most useful technical indicators in existence. The advantage of a moving average is that it tends to smooth out some of the irregularities that exist between market days (Yao et al., 1996). However,

one must preset the technical indicator parameters first, prior to applying for forex forecasting.

In this research, the parameters for GA and TS were preset. A more rigorous exploration of the parameters may produce better performance for both GA and TS methods.

BP is a noisy objective evaluation, due to its sensitivity. In view of this, the GA can be considered as an alternate objective value, and the replication in objective value evaluations would not be necessary.

Bibliography

1. Abid, S., F. Fnaiech, and M. Najim, 2001. A Fast Feed Forward Training Algorithm Using A Modified Form of the Standard Back Propagation Algorithm. *IEEE Transactions on Neural Networks* 12(2): 424-429.

2. Adya, M. and F. Collopy, 1998. How effective are neural networks at forecasting and predictions? A review and evaluation. *Journal of Forecasting* 17(5-6): 481-495.

3. Aiken, M. and M. Bsat, 1999. Forecasting Market Trends with Neural Networks. *Information Systems Management* 16(4): 42-48.

4. Ankenbrand, T. and M. Tomassini, 1995. Multivariate time series modeling of financial markets with artificial neural networks. *Artificial neural nets and genetic algorithms:* Proceedings of the international conference in Alies, France.

5. Ankenbrand, T. and M. Tomassini, 1996. Predicting multivariate financial time series using neural networks: the Swiss bond case. *Proceedings of the IEEE/IAFE 1996 Conference on Computational Intelligence for Financial Engineering,* March 24-26 1996, Manhattan, NY City.

6. Balkin, S.D. 2000. *A statistical implementation of feed forward neural networks for univariate time series forecasting*. Pennsylvania State University. Ph.D. dissertation.

7. Balkin, S.D. and J.K. Ord, 2000. Automatic neural network modeling for univariate time series. *International Journal of Forecasting* 16: 509-515.

8. Batiti, R. and G. Tecchiolli, 1993. *Training Neural Nets with the Reactive Tabu Search.* Dip. Mat. Univ. Trento, UTM 421, November 1993.

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

10. Cannas, B, A. Fanni, M. Marchesi, and F. Pilo. 2000. *Automated Neural Model Design for a CSTR Using a Tabu Search Algorithm.* Department of Electrical and Electronic Engineering, University of Cagliari, Cagliari, Italy.

11. Chan, M, C. Wong, and C. Lam, 2000. Financial time series forecasting by neural network using conjugate gradient learning algorithm and multiple linear regression weight initialization. *Computing in Economics and Finance* [online, cited March 24, 2002].

12. Chartterjee, A. 2000. Artificial neural network and the financial markets: A Survey. *Managerial Finance* 26(12): 32-45.

13. Chen, Chih-Liang. 1992. *Training the Multi-layer Neural Nets by Setting the Initial Weights.* Dissertation. College of Engineering, West Virginia University.

14. Chen, X, J. Racine, and N.R. Swanson, 2001. Semi Parametric ARX Neural Network Models with an Application to Forecasting Inflation. *IEEE Transactions on Neural Networks* 12(4): 674-683.

15. Cheng, Wei and B.W. McClain. 1997. Artificial neural networks make their mark as a powerful tool for investors. *Review of Business* 18(4): 4-9.

16. Davey, N., S.P. Hunt, and R.J. Frank. 2000. *Time Series Prediction and Neural Networks* [online, cited May 25, 2003]. Available from World Wide Web: (http://www. herts.ac.uk).

17. Deng, Y. 2002. *Monitoring process and assessing uncertainty for ANFIS time series forecasting.* Ph.D. Dissertation. College of Engineering and Mineral Resources, West Virginia University, Morgantown, WV.

18. Denton, J.W., 1995. How good are neural networks for causal forecasting? *Journal of Business Forecasting Methods and Systems* 14(2): 17-20.

19. Denton, J.W. and M.S. Hung, 1996. A comparison of nonlinear optimization methods for supervised learning in multiplayer feed forward neural networks. *European Journal of Operational Research* 93(2): 358 – 368.

20. De Werra, D. and A. Hertz, 1989. Tabu Search Techniques: A Tutorial and an Application to Neural Networks. *OR Spektrum* 11: 131-141.

21. Dodd, N., 1991. *Optimisation of Neural-Network Structure Using Genetic Techniques.* Application of Artificial Intelligence in Engineering VI. New York: Elsevier Applied Science.

22. Dorigo, M. and L.M. Gambardella, 1997. Ant Colonies for Traveling Salesman Problem. *Biosystems* 43: 73-81.

23. Dorigo, M. and G. Di Caro. 1999. The Ant Colony Optimization Meta Heuristic. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*. New York: Mc Graw Hill.

24. Dorigo, M., G. Di Caro, L.M. Gambardella, 1999. Ant Algorithm for Discrete Optimization. *Artificial Life* 5(2): 137-172.

25. Dorsey, R.W. and W.J. Mayer, 1994. Optimization Utilizing Genetic Algorithms. *Advances in Artificial Intelligence in Economics, Finance, and Management* 1: 69-91.

26. Dorsey, R.E., J.D. Johnson, and W.J. Mayer, 1994. A Genetic Algorithm for the Training of Feed Forward Neural Networks. *Advances in Artificial Intelligence in Economics, Finance, and Management* 1: 93-111.

27. Dougherty, M.S. and M.R. Cobbett, 1997. Short-term inter-urban traffic forecasts using neural networks. *International Journal of Forecasting* 13(1): 21-31.

28. Efron, B. and Tibshirani, R. J. 1993. *An Introduction to the Bootstrap*. New York: Chapman and Hall.

29. El Shazly, Mona R., El Shazly, Hassan E., 1999. Forecasting currency prices using genetically evolved neural network architecture. *International Review of Financial Analysis* 8(1): 67-82.

30. Fadlalla, A. and C.H. Lin. 2001. An analysis of the applications of neural networks in finance. *Interfaces* 31: 112-122.

31. Gallant, P.J. 2001. *A hybrid evolutionary algorithm to train neural networks as time series predictors.* Department of Electrical and Computer Engineering, Queen's University, Ontario, Canada. Ph.D. dissertation.

32. Glover, F. 1990. Tabu Search: A Tutorial. *Interfaces* 20: 74-94.

33. Glover, F. and M. Laguna. 1997. *Tabu Search*. Boston: Kluwer Academic Publishers.

34. Gupta, J.N.D. and R.S. Sexton, 1999. Comparing back propagation with a genetic algorithm for neural network training. *Omega* 27: 679-684.

35. Hallas, M. and G. Dorffner, 1998. A comparative study on feed forward and recurrent neural networks in time series prediction using gradient descent learning. *Proceedings of the Fourteenth European Meeting on Cybernetics and Systems Research.* Vienna, April 14-17, 1998. Vol. 2 : 644-647.

36. Hansen, J.V., J.B. McDonald, and R.D. Nelson, 1999. Time Series Prediction with Genetic-Algorithm Designed Neural Networks: An Empirical Comparison with Modern Statistical Models. *Computational Intelligence* 15(3): 171-184.

37. Harvey, C.R., K.E. Travers and M.J. Costa. 2000. Forecasting emerging market returns using neural networks. Emerging Market Quarterly. Summer 2000 4(2): 43-54.

38. Haykin, S. 1994. *Neural Networks-A Comprehensive Foundation*. New York: Macmillan College Publishing Co.

39. Hertz, A. and D. de Werra, 1990. The Tabu Search Metaheuristic: How We Used It. *Annals of Mathematics and Artificial Intelligence* 1: 111-121.

40. Hill, T., Marcus O'Connor, William Remus, 1996. Neural Network Models for Time Series Forecasting. *Management Science* 42 (7): 1082-1092.

41. Hoque, A. and A. Latif. 1993. Forecasting exchange rate for the Australian dollar vis-à-vis the US dollar using multivariate time series models. *Applied Economics* 25: 403-307.

42. Koskela, T., M. Lehtokangas, J. Saarinen, K. Kaski, 1994. Time Series Prediction with Multi-layer Perceptron, FIR, and Elman Neural Networks. *World Congress on Neural Networks, San Diego* : 491-496.

43. Kolarik, T. and G. Rudorfer, 1994. Time Series Forecasting Using Neural Networks. *Proceedings of the International Conference on APL: The Language and Its Applications*, September 11-15, 1994, Antwerp, Belgium, ACM, 1994: 86-94.

44. Koza, J.R. and J.P. Rice. 1991. Genetic generation of both the weights and architecture for a neural network. *Int. Joint Conference on Neural Network*, July 1991. Proceeding vol. 2: 397-404.

45. Krishnaswamy, C.R., E.W. Gilbert, and M.M. Pashley. 2000. Neural network applications in finance: a practical introduction. *Financial Practice and Education*: 75-84.

46. Lee, T.H. and S.C. Jung. 2000. Forecasting creditworthiness: logistic vs. artificial neural net. *The Journal of Business Forecasting.* Winter 1999/2000: 28-30.

47. Leung, M.T., A. Chen, and H. Daouk. 2000. Forecasting exchange rates using general regression neural networks. *Computer and Operations Research* 27: 1093-1110.

48. Makridakis et al. 1982. The Accuracy of Extrapolative (Time Series Methods): Results of a Forecasting Competition. *Journal of Forecasting* 1(2): 111-153.

49. Makridakis, S., S.C. Wheelwright and R.J. Hyndman. 1998. *Forecasting: Methods and Applications.* New York: John Wiley and Sons.

50. Makridakis, S. and Wheelwright. 1989. *Forecasting methods for management*. New York: Wiley.

51. Marti, L. 1992. Genetically Generated Neural Network I: Representational Effects. *International Joint Conference on Neural Networks* 4: 537-542, International Neural Network Society.

52. McMenamin, J. Stuart. 1997. A primer on neural networks for forecasting. *Journal of Business Forecasting Methods and Systems* 16(3): 17-22.

53. Meade, N. 2000. Evidence for the selection of forecasting methods. *Journal of Forecasting* 19(6): 515 – 535.

54. Medeiros, M.C., A. Veiga, and C.E. Pedreira. 2001. Modeling exchange rates: smooth transitions, neural networks, and linear models. *IEEE Transactions on Neural Networks* 12(4): 755-764.

55. Ming, W.L. 2001. Neural network with genetically evolved algorithms for stocks prediction. *Asia Pacific Journal of Operational Research* 18(1): 103-107.

56. Nasir, M.L., John, R.I., Bennett, S.C. and D.M. Russell. 2001. Selecting the neural network topology for student modeling of prediction of corporate bankruptcy. *Campus-Wide Information Systems* 18(1): 13-22.

57. Nguyen, Dat-Dao. 2000. *Forecasting macroeconomic models with artificial neural networks: An empirical investigation into the foundation for an intelligent forecasting system.* Dissertation. Concordia University, Montreal, Quebec, Canada.

58. Nicholls, D.F. and A.R. Pagan. 1985. Varying coefficient regression. In E.J. Hannan, P.R. Krishnaiah, and M.M. Rao (eds.), *Handbook of Statistics*, pages 413-449. North-Holland, Amsterdam, 1985.

59. Osmera, P. 1995. Optimization of Neural Networks by Genetic Algorithms. *Neural Network World: International Journal on Neural and Mass-Parallel Computing and Information Systems* 5(6) : 965-976.

60. Paik, H. 2000. Comments on Neural Networks. *Sociological Methods and Research* 28(4): 425-453.

61. Pham, D.T. and D. Karaboga. 2000. *Intelligent Optimisation Techniques: Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks*. Springer_Verlag, London.

62. Rooij, A.J.F. Van, L.C. Jain, and R.P. Johnson. 1996. *Neural Network Training Using Genetic Algorithms*. New Jersey: World Scientific.

63. Sarkar, M. and B. Yegnanaraya. 1997. Feedforward Neural Networks Configuration using Evolutionary Programming. *IEEE International Conference on Neural Networks* 1: 438-443.

64. Salchenberger, L.M., N. Lash and M. Cinar. 1992. Neural Networks: A New Tool for Prediction. *Decision Sciences*, Vol. 23, No. 4, July/August 1992.

65. Setyawati, B.R., R.C. Creese, M. Jaraiedi, W. Iskander, P. Klinkhachorn and J. Denton. 2002. Neural Network for Time Series Forecasting. Poster Presentation. *2002 IIE Annual Conference*, Orlando, Florida, May 2002.

66. Setyawati, B.R., R.C. Creese, M. Jaraiedi. 2003. Neural Network for Univariate and Multivariate Time Series. Refereed Article. *Proceeding of 2003 IIE Annual Conference, Portland, Oregon*, May 18-20, 2003.

67. Setyawati, B.R. 2003. *Neural Network Approach for Commodity Prices Forecasting*. Working Paper. West Virginia University, Morgantown, WV.

68. Setyawati, B.R. R.C. Creese, S. Sahirman. 2004. Genetic Algorithm for Neural Network Optimization. *Proceeding of SPIE Conference*, Philadelphia, PA, May 23-25, 2004.

69. Sexton, R.S., B. Alidaee, R.E. Dorsey, and J.D. Johnson. 1998. Global optimization for artificial neural networks: A tabu search application. *European Journal of Operational Research* 106: 570 – 584.

70. Sexton, R.S., R.E. Dorsey, and J.D. Johnson. 1999a. Beyond back propagation: Using simulated annealing for training neural networks. *Journal of End User Computing* 11(3): 3-9.

71. Sexton, R.S., R.E. Dorsey, and J.D. Johnson. 1999b. Optimization of neural networks: A comparative analysis of the genetic algorithm and simulated annealing. *European Journal of Operational Research* 114: 589 – 601.

72. Tong, H. 1990. *Nonlinear Time Series: A Dynamical System Approach*. Oxford: Oxford University Press.

73. Walczak, S. 2001. An Empirical Analysis of Data Requirements for Financial Forecasting with Neural Networks. *Journal of Management Information Systems* 17(4): 20 3-222.

74. Warner, B. and M. Misra. 1996. Understanding neural networks as statistical tools. T*he American Statistician* 50(4): 284-293.

75. White, H. 1998. Economic prediction using neural networks: the case of IBM daily stock returns. *IEEE International conference on Neural Networks*, July 24-27 1998.

76. Yao, J., Y. Li, and C.L. Tan. 1996. *Forecasting the exchange rates of CHF vs. USD using neural networks.* Dept. of Information Systems and Computer Science, National University of Singapore.

77. Yao, J., H.L Poh and T. Jasic. 1996. Foreign exchange rates forecasting with neural networks. *Proceedings of ICONIP'96 (International Conference on Neural Information Processing)*, Hong Kong, Sept. 24-27: 754-759.

78. Yao, J. and C.L. Tan. 2000. A case study on using neural networks to perform technical forecasting of forex. *Neurocomputing* 34: 79-98.

79. Yao, X. 1999. Evolving Artificial Neural Networks. *Proceedings of the IEEE* 87(9): 1423-1447.

80. Zhang et al. 1998. Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting* 14(1): 35-62

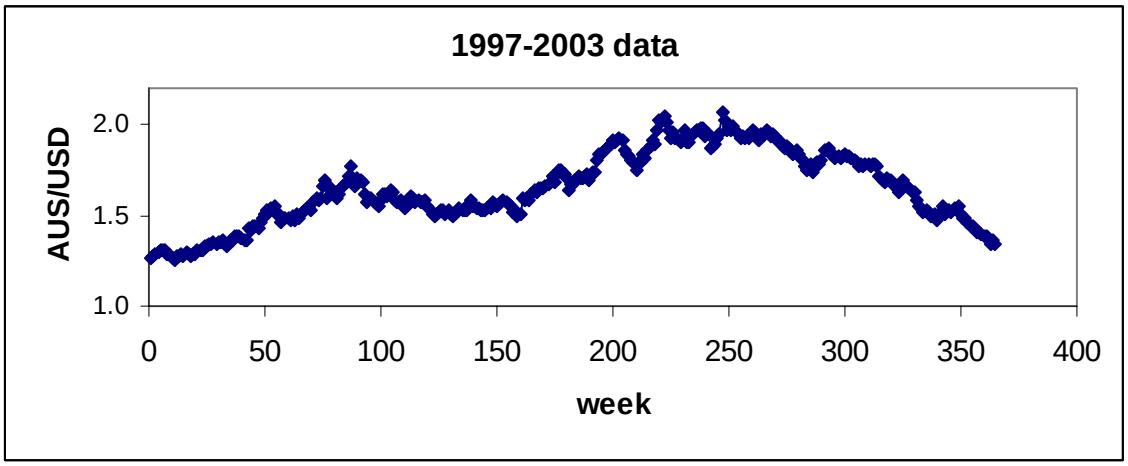
81. Zhang, G.P., B.E. Patuwo, and M.Y. Hu. 2001. A Simulation study of artificial neural networks for nonlinear time series forecasting. *Computers and Operations Research* 28: 381 - 396.

APPENDICES

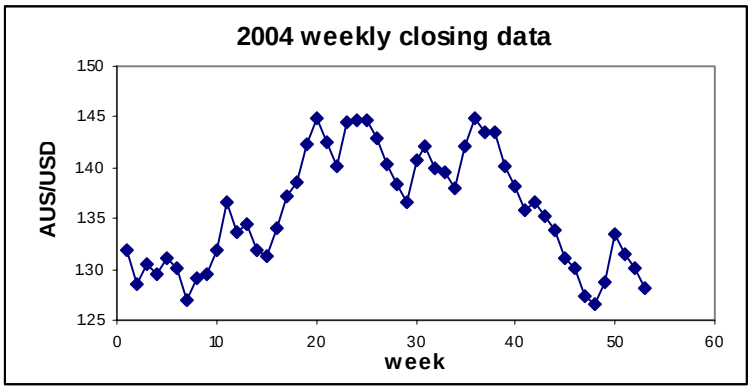
Appendix 1. Weekly Closing of Five Foreign Exchange Rates

1. AUS/USD Weekly Closing Data

Week	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003
1	1.2829	1.3201	1.4880	1.4568	1.3046	1.3364	1.2633	1.5347	1.5799	1.5258	1.7515	1.9252	1.7701
2	1.2937	1.3578	1.4830	1.4494	1.3203	1.3422	1.2791	1.5462	1.5749	1.5003	1.7977	1.9186	1.7174
3	1.2845	1.3450	1.4871	1.4230	1.3012	1.3584	1.2810	1.5081	1.5753	1.5086	1.7941	1.9413	1.6923
4	1.2881	1.3517	1.4717	1.4106	1.3117	1.3540	1.2942	1.5064	1.5877	1.5924	1.8412	1.9351	1.6878
5	1.2787	1.3290	1.4770	1.3989	1.3266	1.3240	1.3105	1.4607	1.5349	1.5821	1.8102	1.9675	1.7057
6	1.2822	1.3312	1.4760	1.3987	1.3419	1.3239	1.3119	1.4899	1.5469	1.5911	1.8628	1.9586	1.6933
7	1.2682	1.3254	1.4482	1.4006	1.3559	1.3230	1.3063	1.4834	1.5679	1.5885	1.8845	1.9334	1.6885
8	1.2748	1.3277	1.4367	1.3859	1.3563	1.3226	1.2861	1.4870	1.6103	1.6182	1.9160	1.9481	1.6740
9	1.2740	1.3262	1.4123	1.3996	1.3544	1.3132	1.2868	1.4719	1.5923	1.6424	1.8883	1.9308	1.6448
10	1.3008	1.3231	1.4062	1.4074	1.3428	1.3015	1.2677	1.4976	1.5711	1.6291	1.9671	1.9092	1.6291
11	1.2958	1.3246	1.3964	1.4034	1.3618	1.2920	1.2525	1.4787	1.5860	1.6481	2.0245	1.9021	1.6769
12	1.2932	1.3164	1.4085	1.4046	1.3720	1.2869	1.2723	1.5017	1.5755	1.6498	2.0147	1.8777	1.6920
13	1.2896	1.3054	1.4338	1.4159	1.3642	1.2788	1.2771	1.4843	1.5735	1.6485	2.0480	1.8738	1.6657
14	1.2717	1.3041	1.4105	1.3809	1.3477	1.2758	1.2882	1.5218	1.5872	1.6649	2.0159	1.8844	1.6635
15	1.2748	1.3088	1.3958	1.3822	1.3513	1.2659	1.2724	1.5262	1.5455	1.6741	1.9671	1.8733	1.6531
16	1.2933	1.3036	1.3999	1.3919	1.3670	1.2780	1.2927	1.5381	1.5342	1.6855	1.9295	1.8547	1.6262
17	1.2821	1.3246	1.4181	1.4007	1.3747	1.2639	1.2870	1.5358	1.5106	1.7123	1.9595	1.8419	1.6259
18	1.2870	1.3231	1.4115	1.3963	1.3476	1.2559	1.2775	1.5325	1.4908	1.6835	1.9271	1.8582	1.5869
19	1.2755	1.3254	1.4236	1.3841	1.3495	1.2456	1.2857	1.5719	1.5038	1.7219	1.9213	1.8366	1.5507
20	1.2773	1.3195	1.4384	1.3717	1.3982	1.2500	1.2906	1.5941	1.5156	1.7456	1.9039	1.8105	1.5377
21	1.3176	1.3201	1.4567	1.3643	1.3901	1.2626	1.3075	1.5815	1.5314	1.7482	1.9255	1.7991	1.5194
22	1.3166	1.3216	1.4821	1.3550	1.3951	1.2543	1.3124	1.5970	1.5249	1.7274	1.9659	1.7671	1.5338
23	1.3298	1.3106	1.4706	1.3620	1.3887	1.2627	1.3135	1.6655	1.5051	1.7051	1.9067	1.7508	1.5176
24	1.3241	1.3176	1.4784	1.3643	1.3727	1.2614	1.3303	1.6887	1.5306	1.6435	1.8995	1.7855	1.4982
25	1.3073	1.3290	1.4932	1.3724	1.3864	1.2641	1.3286	1.5946	1.5099	1.6761	1.9341	1.7406	1.4969
26	1.3029	1.3379	1.4948	1.3721	1.4069	1.2726	1.3408	1.6588	1.4926	1.6721	1.9606	1.7763	1.5024
27	1.3064	1.3422	1.4652	1.3718	1.3957	1.2554	1.3360	1.6170	1.5048	1.6950	1.9669	1.7952	1.4721
28	1.2978	1.3396	1.4727	1.3633	1.3690	1.2531	1.3498	1.6260	1.5140	1.7101	1.9702	1.7859	1.5120
29	1.2845	1.3433	1.4744	1.3519	1.3592	1.2643	1.3451	1.5955	1.5432	1.7065	1.9780	1.8000	1.5479
30	1.2886	1.3400	1.4541	1.3532	1.3576	1.2694	1.3548	1.6208	1.5328	1.7018	1.9745	1.8616	1.5054
31	1.2833	1.3432	1.4701	1.3486	1.3468	1.2933	1.3510	1.6441	1.5307	1.7140	1.9311	1.8552	1.5295
32	1.2759	1.3566	1.4748	1.3477	1.3487	1.2829	1.3609	1.6651	1.5337	1.7253	1.9476	1.8673	1.5310
33	1.2862	1.3870	1.4763	1.3540	1.3587	1.2753	1.3454	1.6827	1.5650	1.6935	1.8672	1.8374	1.5174
34	1.2727	1.3878	1.5016	1.3432	1.3446	1.2651	1.3356	1.7176	1.5810	1.7424	1.8834	1.8390	1.5355
35	1.2731	1.3938	1.5402	1.3482	1.3333	1.2646	1.3599	1.7718	1.5457	1.7345	1.8967	1.8194	1.5400
36	1.2709	1.3888	1.5385	1.3423	1.3298	1.2528	1.3622	1.7073	1.5340	1.7990	1.9228	1.8288	1.5465
37	1.2555	1.3780	1.5350	1.3433	1.3167	1.2605	1.3849	1.6656	1.5425	1.8341	1.9430	1.8164	1.5064
38	1.2531	1.3662	1.5280	1.3526	1.3415	1.2613	1.3899	1.7006	1.5326	1.8326	2.0629	1.8286	1.4860
39	1.2546	1.3837	1.5432	1.3509	1.3231	1.2642	1.3826	1.6910	1.5290	1.8459	2.0207	1.8371	1.4816
40	1.2599	1.3958	1.5199	1.3552	1.3132	1.2671	1.3729	1.6826	1.5279	1.8699	1.9742	1.8303	1.4677
41	1.2544	1.3931	1.5095	1.3599	1.3174	1.2658	1.3612	1.6207	1.5444	1.8862	1.9932	1.8231	1.4441
42	1.2523	1.3888	1.4947	1.3671	1.3333	1.2558	1.3637	1.5769	1.5374	1.8937	1.9675	1.8175	1.4456
43	1.2723	1.3918	1.5015	1.3461	1.3253	1.2627	1.4278	1.5989	1.5686	1.9097	1.9922	1.8025	1.4269
44	1.2794	1.4379	1.4837	1.3341	1.3190	1.2666	1.4205	1.5980	1.5661	1.9045	1.9679	1.7875	1.4122
45	1.2731	1.4266	1.5181	1.3252	1.3531	1.2694	1.4418	1.5759	1.5542	1.9149	1.9443	1.7669	1.4106
46	1.2735	1.4409	1.5105	1.3201	1.3495	1.2655	1.4351	1.5631	1.5646	1.9258	1.9205	1.7767	1.3931
47	1.2626	1.4609	1.5102	1.3168	1.3532	1.2307	1.4323	1.5525	1.5750	1.9163	1.9336	1.7731	1.3830
48	1.2742	1.4504	1.4946	1.3009	1.3560	1.2276	1.4627	1.5795	1.5780	1.8606	1.9208	1.7845	1.3810
49	1.2837	1.4447	1.4892	1.2909	1.3548	1.2553	1.4867	1.6181	1.5708	1.8360	1.9403	1.7843	1.3583
50	1.2980	1.4483	1.4718	1.2896	1.3487	1.2618	1.5088	1.6122	1.5564	1.8306	1.9286	1.7709	1.3464
51	1.3030	1.4459	1.4734	1.2865	1.3514	1.2584	1.5292	1.6104	1.5556	1.8019	1.9694	1.7795	1.3582
52	1.3112	1.4462	1.4727	1.2899	1.3450	1.2569	1.5227	1.6359	1.5221	1.7975	1.9556	1.7829	1.3464
53		1.4511						1.6318					

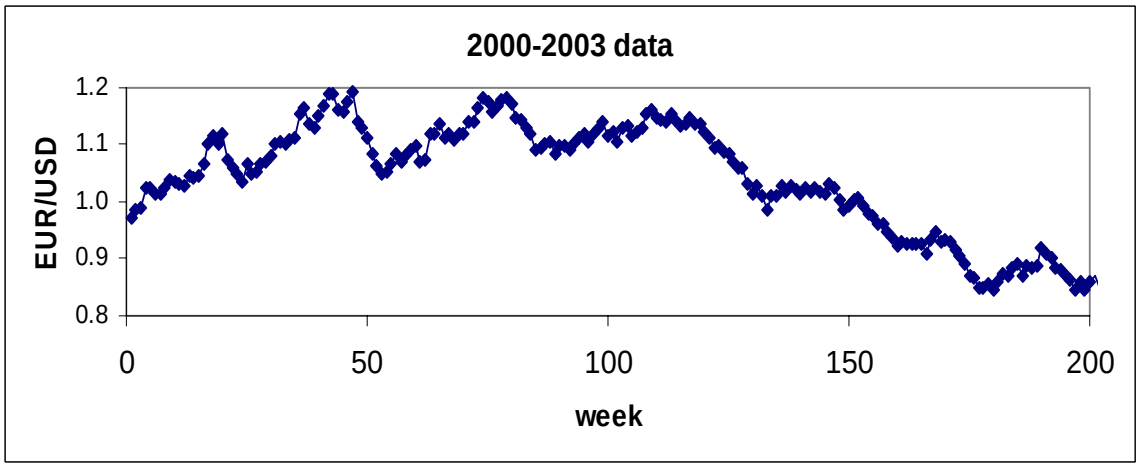


week	AUS/USD	week	AUS/USD
1	1.3197	27	1.4034
2	1.2857	28	1.3836
3	1.3051	29	1.3670
4	1.2951	30	1.4082
5	1.3114	31	1.4213
6	1.3021	32	1.3988
7	1.2694	33	1.3949
8	1.2909	34	1.3808
9	1.2958	35	1.4207
10	1.3181	36	1.4488
11	1.3652	37	1.4352
12	1.3375	38	1.4341
13	1.3453	39	1.4017
14	1.3186	40	1.3823
15	1.3131	41	1.3592
16	1.3409	42	1.3655
17	1.3717	43	1.3523
18	1.3862	44	1.3389
19	1.4238	45	1.3116
20	1.4492	46	1.3006
21	1.4259	47	1.2727
22	1.4008	48	1.2656
23	1.4442	49	1.2868
24	1.4464	50	1.3342
25	1.4478	51	1.3147
26	1.4301	52	1.3008
		53	1.2821

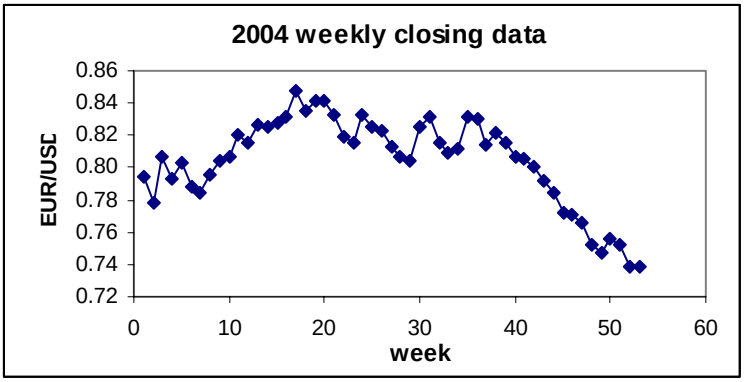


2. EUR/USD Weekly Closing Data

Week	2000	2001	2002	2003
1	0.9715	1.0487	1.1173	0.9598
2	0.9870	1.0542	1.1226	0.9488
3	0.9902	1.0674	1.1308	0.9379
4	1.0245	1.0855	1.1551	0.9237
5	1.0243	1.0687	1.1607	0.9311
6	1.0156	1.0792	1.1462	0.9259
7	1.0153	1.0913	1.1453	0.9260
8	1.0245	1.0992	1.1420	0.9272
9	1.0397	1.0705	1.1555	0.9278
10	1.0351	1.0733	1.1428	0.9077
11	1.0312	1.1202	1.1332	0.9324
12	1.0283	1.1204	1.1374	0.9489
13	1.0449	1.1375	1.1480	0.9292
14	1.0426	1.1109	1.1356	0.9334
15	1.0454	1.1205	1.1372	0.9301
16	1.0666	1.1072	1.1244	0.9169
17	1.1000	1.1204	1.1133	0.9058
18	1.1168	1.1189	1.0952	0.8928
19	1.1014	1.1415	1.0971	0.8696
20	1.1177	1.1397	1.0861	0.8663
21	1.0742	1.1658	1.0845	0.8483
22	1.0603	1.1831	1.0708	0.8500
23	1.0496	1.1753	1.0585	0.8551
24	1.0364	1.1583	1.0585	0.8453
25	1.0684	1.1667	1.0302	0.8607
26	1.0474	1.1800	1.0144	0.8748
27	1.0543	1.1812	1.0282	0.8700
28	1.0668	1.1728	1.0096	0.8845
29	1.0700	1.1475	0.9845	0.8895
30	1.0814	1.1429	1.0121	0.8686
31	1.1019	1.1310	1.0118	0.8884
32	1.1057	1.1183	1.0293	0.8833
33	1.1027	1.0911	1.0166	0.8883
34	1.1081	1.0964	1.0281	0.9198
35	1.1118	1.1000	1.0194	0.9102
36	1.1540	1.1058	1.0156	0.9031
37	1.1664	1.0859	1.0263	0.8843
38	1.1371	1.0983	1.0163	0.8813
39	1.1310	1.0990	1.0234	0.8709
40	1.1513	1.0907	1.0179	0.8621
41	1.1672	1.1007	1.0146	0.8465
42	1.1897	1.1125	1.0299	0.8598
43	1.1893	1.1210	1.0238	0.8450
44	1.1603	1.1059	1.0028	0.8614
45	1.1596	1.1191	0.9863	0.8691
46	1.1741	1.1298	0.9918	0.8517
47	1.1929	1.1401	1.0034	0.8393
48	1.1405	1.1165	1.0068	0.8337
49	1.1309	1.1232	0.9921	0.8226
50	1.1133	1.1057	0.9778	0.8141
51	1.0833	1.1281	0.9738	0.8077
52	1.0646	1.1332	0.9605	0.8046

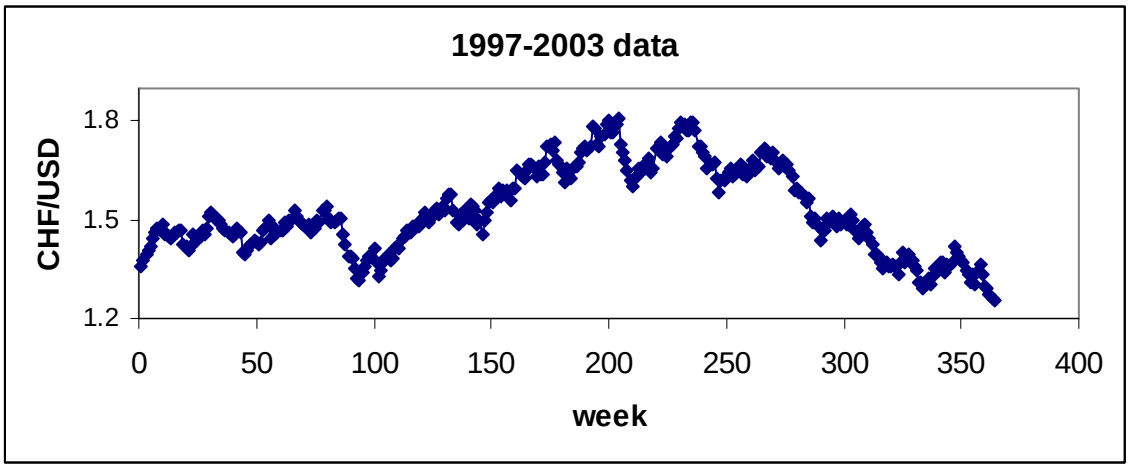


week	EUR/USD	week	EUR/USD
1	0.7942	**27**	0.8124
2	0.7781	**28**	0.8073
3	0.8067	**29**	0.8040
4	0.7930	**30**	0.8249
5	0.8029	**31**	0.8311
6	0.7880	**32**	0.8148
7	0.7842	**33**	0.8093
8	0.7958	**34**	0.8115
9	0.8038	**35**	0.8316
10	0.8064	**36**	0.8297
11	0.8202	**37**	0.8143
12	0.8151	**38**	0.8214
13	0.8270	**39**	0.8157
14	0.8257	**40**	0.8064
15	0.8272	**41**	0.8052
16	0.8314	**42**	0.8010
17	0.8473	**43**	0.7912
18	0.8348	**44**	0.7844
19	0.8412	**45**	0.7723
20	0.8419	**46**	0.7712
21	0.8326	**47**	0.7662
22	0.8190	**48**	0.7526
23	0.8157	**49**	0.7467
24	0.8323	**50**	0.7563
25	0.8252	**51**	0.7523
26	0.8233	**52**	0.7388
		53	0.7388

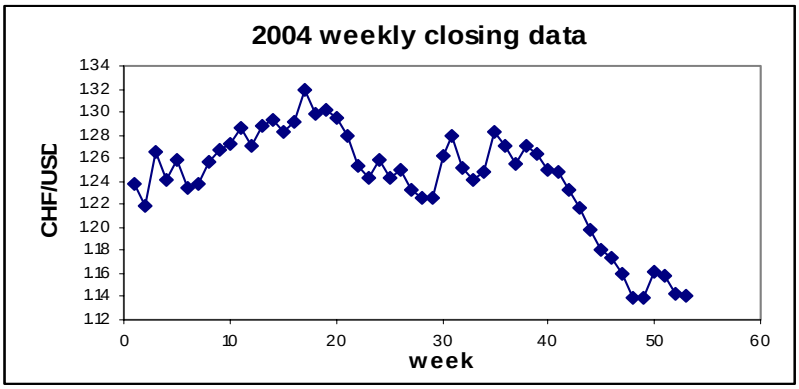


3. **CHF/USD Weekly Closing Data**

Week	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003
1	1.2755	1.3764	1.4934	1.4708	1.3110	1.1600	1.3595	1.4663	1.3970	1.5615	1.6028	1.6506	1.3970
2	1.2801	1.4080	1.4965	1.4800	1.2904	1.1606	1.3761	1.4763	1.3783	1.5940	1.6236	1.6599	1.3863
3	1.2610	1.4220	1.4589	1.4645	1.2711	1.1945	1.3958	1.4981	1.3830	1.5957	1.6338	1.6629	1.3678
4	1.2595	1.4075	1.4860	1.4615	1.2735	1.2071	1.4078	1.4437	1.4153	1.6505	1.6576	1.7022	1.3532
5	1.2514	1.4280	1.5266	1.4571	1.2915	1.2175	1.4214	1.4825	1.4179	1.6467	1.6429	1.7178	1.3680
6	1.2435	1.3896	1.5410	1.4776	1.2845	1.2087	1.4450	1.4542	1.4159	1.6300	1.6558	1.6900	1.3585
7	1.2675	1.4670	1.5064	1.4506	1.2591	1.1879	1.4618	1.4646	1.4463	1.6281	1.6761	1.6980	1.3613
8	1.2895	1.4936	1.5240	1.4285	1.2478	1.1725	1.4712	1.4700	1.4487	1.6474	1.6883	1.6892	1.3615
9	1.3316	1.4841	1.5459	1.4400	1.2145	1.2053	1.4744	1.4662	1.4689	1.6711	1.6471	1.7057	1.3557
10	1.3641	1.5170	1.5230	1.4175	1.1801	1.2061	1.4843	1.4948	1.4613	1.6667	1.6549	1.6809	1.3338
11	1.3875	1.5064	1.5080	1.4355	1.1500	1.1893	1.4559	1.4773	1.4628	1.6629	1.7158	1.6588	1.3691
12	1.4090	1.5246	1.5086	1.4161	1.1755	1.1940	1.4572	1.4974	1.4829	1.6331	1.7162	1.6621	1.3998
13	1.4450	1.4969	1.4911	1.4266	1.1325	1.1893	1.4501	1.4957	1.4815	1.6625	1.7365	1.6838	1.3718
14	1.4190	1.4899	1.4766	1.4430	1.1360	1.1970	1.4407	1.5289	1.4829	1.6380	1.7004	1.6629	1.3871
15	1.4281	1.5085	1.4761	1.4534	1.1521	1.2260	1.4647	1.5134	1.4990	1.6404	1.7054	1.6680	1.3930
16	1.4705	1.5450	1.4295	1.4380	1.1361	1.2230	1.4595	1.5010	1.5073	1.6772	1.6925	1.6505	1.3760
17	1.4779	1.5330	1.4346	1.4100	1.1455	1.2361	1.4691	1.4898	1.5256	1.7244	1.7231	1.6294	1.3611
18	1.4733	1.5004	1.4270	1.4140	1.1321	1.2425	1.4689	1.4881	1.4911	1.7325	1.7298	1.5915	1.3478
19	1.4574	1.5284	1.4505	1.4260	1.2050	1.2475	1.4222	1.4799	1.5060	1.7086	1.7514	1.5948	1.3116
20	1.4679	1.4880	1.4726	1.4050	1.2005	1.2547	1.4193	1.4882	1.5168	1.7337	1.7482	1.5811	1.3104
21	1.4550	1.4884	1.4180	1.4024	1.1375	1.2670	1.4051	1.4641	1.5267	1.6831	1.7796	1.5826	1.2925
22	1.4805	1.4574	1.4599	1.4165	1.1690	1.2523	1.4168	1.4819	1.5369	1.6676	1.7979	1.5691	1.3023
23	1.5175	1.4504	1.4495	1.4060	1.1584	1.2668	1.4531	1.4759	1.5186	1.6433	1.7924	1.5552	1.3233
24	1.5326	1.4269	1.5004	1.3590	1.1624	1.2533	1.4474	1.4963	1.5408	1.6169	1.7711	1.5623	1.3047
25	1.5464	1.4159	1.5129	1.3295	1.1470	1.2630	1.4400	1.4932	1.5309	1.6568	1.7728	1.5123	1.3288
26	1.5580	1.3834	1.5095	1.3355	1.1515	1.2500	1.4489	1.5266	1.5678	1.6315	1.7967	1.4914	1.3511
27	1.5755	1.3569	1.5304	1.3201	1.1575	1.2655	1.4704	1.5304	1.5775	1.6283	1.7939	1.5063	1.3478
28	1.5489	1.3560	1.5156	1.3130	1.1635	1.2583	1.4581	1.5387	1.5748	1.6535	1.7734	1.4829	1.3724
29	1.5130	1.3085	1.5186	1.3494	1.1560	1.2170	1.4719	1.5047	1.5291	1.6636	1.7260	1.4385	1.3674
30	1.5160	1.3284	1.5225	1.3439	1.1462	1.2114	1.5123	1.4940	1.4929	1.6742	1.7247	1.4676	1.3428
31	1.5079	1.3186	1.5006	1.3341	1.1502	1.2035	1.5242	1.4907	1.4860	1.7056	1.7053	1.4696	1.3666
32	1.5135	1.3134	1.5264	1.3089	1.1915	1.2040	1.5069	1.4977	1.5158	1.7180	1.6916	1.5032	1.3578
33	1.5415	1.3240	1.4774	1.2925	1.2205	1.2115	1.5057	1.5072	1.4984	1.7232	1.6583	1.4912	1.3720
34	1.5320	1.2766	1.4720	1.3230	1.2182	1.2003	1.4980	1.5016	1.5307	1.7088	1.6670	1.5127	1.4178
35	1.5280	1.2596	1.4296	1.3090	1.2010	1.2005	1.4885	1.4529	1.5069	1.7228	1.6687	1.5007	1.4002
36	1.5235	1.2516	1.3949	1.2820	1.2165	1.2159	1.4748	1.4253	1.5448	1.7857	1.6770	1.4816	1.3887
37	1.4760	1.2880	1.4005	1.2714	1.2080	1.2405	1.4658	1.3879	1.5435	1.7808	1.6232	1.5026	1.3764
38	1.4710	1.3051	1.4366	1.2835	1.1337	1.2395	1.4595	1.3885	1.5301	1.7301	1.5849	1.4878	1.3708
39	1.4584	1.2975	1.4244	1.2868	1.1559	1.2557	1.4529	1.3806	1.4866	1.7256	1.6187	1.5013	1.3438
40	1.4670	1.2349	1.4055	1.2809	1.1484	1.2563	1.4497	1.3545	1.4999	1.7511	1.6196	1.4887	1.3343
41	1.4831	1.3141	1.4266	1.2635	1.1572	1.2545	1.4564	1.3213	1.4585	1.7603	1.6326	1.4858	1.3112
42	1.4764	1.3190	1.4751	1.2474	1.1417	1.2711	1.4760	1.3145	1.4975	1.7889	1.6416	1.5138	1.3350
43	1.4885	1.3655	1.4810	1.2605	1.1331	1.2597	1.4626	1.3414	1.5250	1.8054	1.6539	1.5008	1.3065
44	1.4605	1.3770	1.4930	1.2710	1.1381	1.2697	1.4008	1.3554	1.5518	1.7668	1.6305	1.4669	1.3372
45	1.4509	1.4305	1.4970	1.2859	1.1357	1.2655	1.3965	1.3738	1.5576	1.7651	1.6398	1.4428	1.3638
46	1.4520	1.4125	1.5066	1.3185	1.1345	1.2780	1.4066	1.3907	1.5541	1.7907	1.6550	1.4538	1.3330
47	1.4115	1.4325	1.4976	1.3226	1.1422	1.2694	1.4167	1.3911	1.5802	1.8091	1.6682	1.4772	1.2979
48	1.4355	1.4386	1.4901	1.3304	1.1736	1.3031	1.4256	1.4125	1.5930	1.7273	1.6404	1.4858	1.2910
49	1.3889	1.4334	1.4564	1.3375	1.1678	1.3142	1.4398	1.3692	1.5743	1.7065	1.6592	1.4603	1.2757
50	1.3985	1.4065	1.4545	1.3309	1.1627	1.3166	1.4321	1.3293	1.5883	1.6799	1.6306	1.4418	1.2631
51	1.3620	1.4115	1.4380	1.3335	1.1585	1.3329	1.4276	1.3481	1.5855	1.6482	1.6535	1.4246	1.2562
52	1.3505	1.4480	1.4875	1.3095	1.1531	1.3495	1.4342	1.3754	1.5923	1.6204	1.6785	1.3953	
53		1.4656						1.3735					

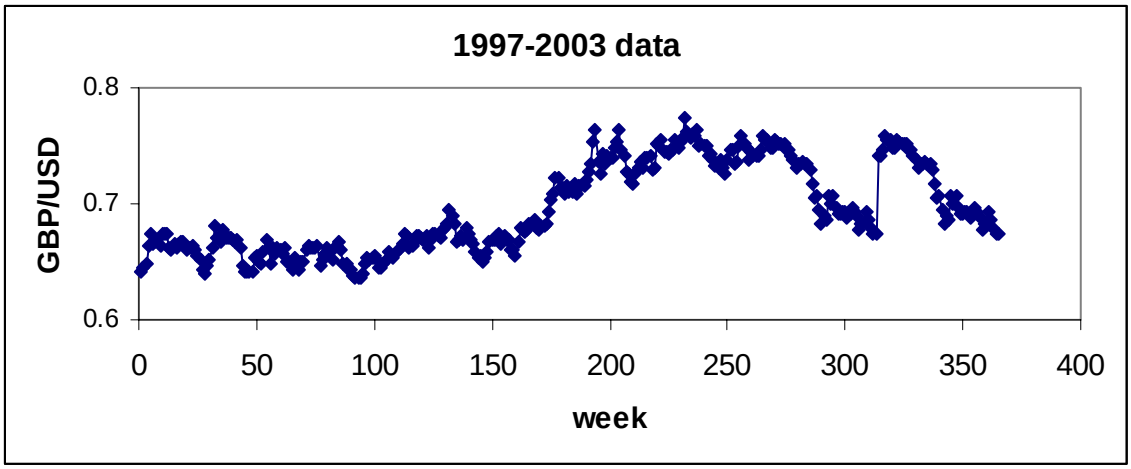


week	CHF/USD	week	CHF/USD
1	1.2385	**27**	1.2332
2	1.2187	**28**	1.2259
3	1.2659	**29**	1.2262
4	1.2420	**30**	1.2629
5	1.2590	**31**	1.2788
6	1.2342	**32**	1.2512
7	1.2372	**33**	1.2407
8	1.2572	**34**	1.2490
9	1.2676	**35**	1.2832
10	1.2727	**36**	1.2712
11	1.2858	**37**	1.2558
12	1.2709	**38**	1.2704
13	1.2876	**39**	1.2634
14	1.2931	**40**	1.2495
15	1.2829	**41**	1.2487
16	1.2911	**42**	1.2327
17	1.3198	**43**	1.2168
18	1.2981	**44**	1.1986
19	1.3019	**45**	1.1799
20	1.2958	**46**	1.1736
21	1.2791	**47**	1.1591
22	1.2536	**48**	1.1393
23	1.2435	**49**	1.1391
24	1.2594	**50**	1.1610
25	1.2438	**51**	1.1574
26	1.2501	**52**	1.1427
		53	1.1412

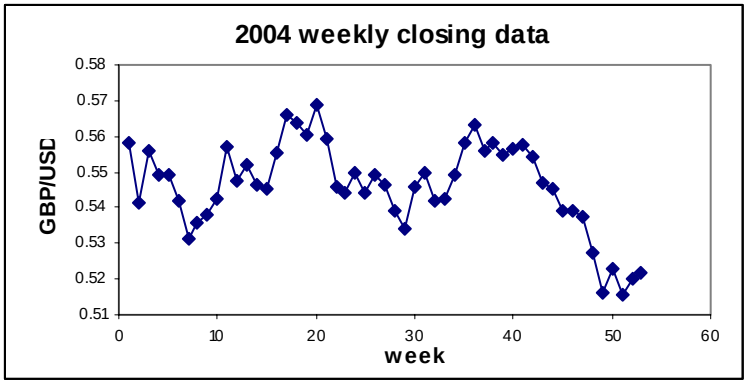


4. GBP/USD Weekly Closing Data

Week	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003
1	0.5172	0.5416	0.6489	0.6709	0.6445	0.6441	0.5915	0.6091	0.6094	0.6102	0.6673	0.6919	0.6233
2	0.5242	0.5571	0.6534	0.6705	0.6384	0.6455	0.5946	0.6187	0.6057	0.6112	0.6765	0.6921	0.6220
3	0.5149	0.5627	0.6525	0.6687	0.6299	0.6626	0.5986	0.6120	0.6041	0.6054	0.6810	0.6959	0.6181
4	0.5107	0.5539	0.6723	0.6666	0.6295	0.6652	0.6143	0.5990	0.6076	0.6170	0.6857	0.7093	0.6126
5	0.5060	0.5585	0.6909	0.6759	0.6389	0.6583	0.6241	0.6126	0.6108	0.6288	0.6802	0.7060	0.6079
6	0.5031	0.5438	0.7052	0.6826	0.6397	0.6527	0.6161	0.6076	0.6129	0.6280	0.6904	0.7054	0.6141
7	0.5075	0.5653	0.6883	0.6759	0.6343	0.6455	0.6169	0.6117	0.6150	0.6255	0.6894	0.6983	0.6206
8	0.5144	0.5719	0.7027	0.6725	0.6296	0.6490	0.6190	0.6108	0.6238	0.6283	0.6910	0.6990	0.6303
9	0.5270	0.5692	0.6916	0.6714	0.6164	0.6547	0.6138	0.6070	0.6220	0.6325	0.6790	0.7048	0.6354
10	0.5342	0.5819	0.6976	0.6662	0.6323	0.6559	0.6236	0.6119	0.6122	0.6328	0.6814	0.7028	0.6241
11	0.5461	0.5840	0.6714	0.6698	0.6319	0.6556	0.6240	0.6000	0.6134	0.6364	0.7009	0.7022	0.6320
12	0.5590	0.5860	0.6707	0.6675	0.6293	0.6523	0.6243	0.5999	0.6172	0.6285	0.6995	0.7010	0.6387
13	0.5719	0.5754	0.6607	0.6787	0.6172	0.6552	0.6129	0.5935	0.6224	0.6281	0.7050	0.7022	0.6379
14	0.5648	0.5736	0.6544	0.6773	0.6243	0.6541	0.6105	0.6031	0.6223	0.6317	0.6952	0.6968	0.6411
15	0.5626	0.5664	0.6551	0.6784	0.6242	0.6612	0.6156	0.5978	0.6204	0.6304	0.6945	0.6959	0.6364
16	0.5807	0.5730	0.6347	0.6721	0.6209	0.6592	0.6127	0.5936	0.6182	0.6330	0.6932	0.6908	0.6345
17	0.5931	0.5645	0.6374	0.6607	0.6213	0.6623	0.6158	0.5997	0.6216	0.6424	0.6959	0.6874	0.6283
18	0.5895	0.5602	0.6358	0.6693	0.6254	0.6649	0.6164	0.5995	0.6118	0.6542	0.6962	0.6818	0.6241
19	0.5814	0.5595	0.6500	0.6671	0.6366	0.6568	0.6168	0.6106	0.6183	0.6591	0.7050	0.6846	0.6232
20	0.5826	0.5496	0.6487	0.6625	0.6355	0.6601	0.6106	0.6146	0.6242	0.6731	0.6976	0.6849	0.6157
21	0.5777	0.5511	0.6406	0.6625	0.6232	0.6616	0.6124	0.6125	0.6241	0.6724	0.7039	0.6858	0.6106
22	0.5886	0.5464	0.6590	0.6640	0.6301	0.6453	0.6114	0.6120	0.6219	0.6631	0.7062	0.6837	0.6099
23	0.5979	0.5447	0.6555	0.6629	0.6272	0.6487	0.6140	0.6114	0.6213	0.6630	0.7236	0.6846	0.6018
24	0.6094	0.5400	0.6682	0.6536	0.6221	0.6510	0.6112	0.6132	0.6280	0.6593	0.7113	0.6788	0.5996
25	0.6131	0.5378	0.6773	0.6443	0.6232	0.6501	0.6044	0.5974	0.6290	0.6663	0.7070	0.6672	0.6008
26	0.6179	0.5284	0.6636	0.6498	0.6287	0.6437	0.6018	0.6021	0.6332	0.6609	0.7101	0.6559	0.6061
27	0.6188	0.5238	0.6752	0.6468	0.6270	0.6435	0.5931	0.6069	0.6443	0.6604	0.7088	0.6575	0.5993
28	0.6062	0.5206	0.6762	0.6417	0.6270	0.6439	0.5902	0.6125	0.6399	0.6670	0.7143	0.6451	0.6126
29	0.5912	0.5130	0.6680	0.6530	0.6274	0.6468	0.5958	0.6085	0.6333	0.6589	0.6998	0.6336	0.6301
30	0.5924	0.5263	0.6734	0.6485	0.6248	0.6431	0.6009	0.6022	0.6173	0.6655	0.7012	0.6396	0.6173
31	0.5900	0.5208	0.6680	0.6485	0.6221	0.6485	0.6116	0.6122	0.6192	0.6652	0.7006	0.6364	0.6230
32	0.5896	0.5184	0.6854	0.6470	0.6339	0.6454	0.6308	0.6137	0.6229	0.6647	0.7003	0.6563	0.6218
33	0.6019	0.5218	0.6618	0.6456	0.6480	0.6463	0.6215	0.6178	0.6195	0.6713	0.6913	0.6505	0.6264
34	0.5958	0.5133	0.6660	0.6513	0.6484	0.6439	0.6207	0.6108	0.6294	0.6784	0.6924	0.6572	0.6347
35	0.5949	0.5045	0.6551	0.6462	0.6436	0.6405	0.6169	0.5983	0.6235	0.6849	0.6887	0.6461	0.6339
36	0.5893	0.5010	0.6456	0.6447	0.6458	0.6405	0.6279	0.5979	0.6186	0.7041	0.6835	0.6411	0.6292
37	0.5772	0.5197	0.6517	0.6313	0.6445	0.6433	0.6211	0.5937	0.6163	0.7143	0.6810	0.6417	0.6224
38	0.5780	0.5745	0.6649	0.6321	0.6329	0.6429	0.6207	0.5935	0.6088	0.6860	0.6873	0.6426	0.6128
39	0.5755	0.5833	0.6642	0.6341	0.6329	0.6398	0.6212	0.5874	0.6043	0.6762	0.6803	0.6433	0.6025
40	0.5750	0.5794	0.6511	0.6305	0.6325	0.6396	0.6195	0.5868	0.6054	0.6924	0.6753	0.6373	0.6012
41	0.5816	0.5910	0.6612	0.6282	0.6357	0.6349	0.6168	0.5863	0.5993	0.6853	0.6886	0.6407	0.6003
42	0.5797	0.6041	0.6787	0.6150	0.6346	0.6288	0.6194	0.5869	0.6030	0.6919	0.6957	0.6467	0.5979
43	0.5845	0.6200	0.6725	0.6163	0.6329	0.6227	0.6117	0.5897	0.6086	0.6889	0.6969	0.6438	0.5890
44	0.5716	0.6396	0.6727	0.6209	0.6327	0.6111	0.5961	0.5975	0.6172	0.6904	0.6840	0.6394	0.5897
45	0.5655	0.6506	0.6745	0.6254	0.6346	0.6070	0.5915	0.6030	0.6196	0.6988	0.6861	0.6283	0.5985
46	0.5653	0.6454	0.6794	0.6380	0.6446	0.6006	0.5906	0.6012	0.6187	0.7026	0.7004	0.6332	0.5937
47	0.5563	0.6577	0.6757	0.6397	0.6413	0.5951	0.5917	0.6040	0.6230	0.7144	0.7094	0.6338	0.5872
48	0.5666	0.6621	0.6709	0.6408	0.6512	0.5944	0.5920	0.6057	0.6244	0.6957	0.7018	0.6428	0.5807
49	0.5525	0.6423	0.6689	0.6410	0.6522	0.6090	0.6033	0.6012	0.6163	0.6919	0.6987	0.6355	0.5790
50	0.5504	0.6419	0.6707	0.6398	0.6496	0.6022	0.6060	0.5947	0.6222	0.6772	0.6872	0.6290	0.5725
51	0.5376	0.6388	0.6651	0.6475	0.6481	0.5998	0.5992	0.5951	0.6191	0.6763	0.6950	0.6242	0.5657
52	0.5326	0.6564	0.6766	0.6388	0.6439	0.5912	0.5989	0.5979	0.6191	0.6688	0.6906	0.6241	
53		0.6605						0.6014					

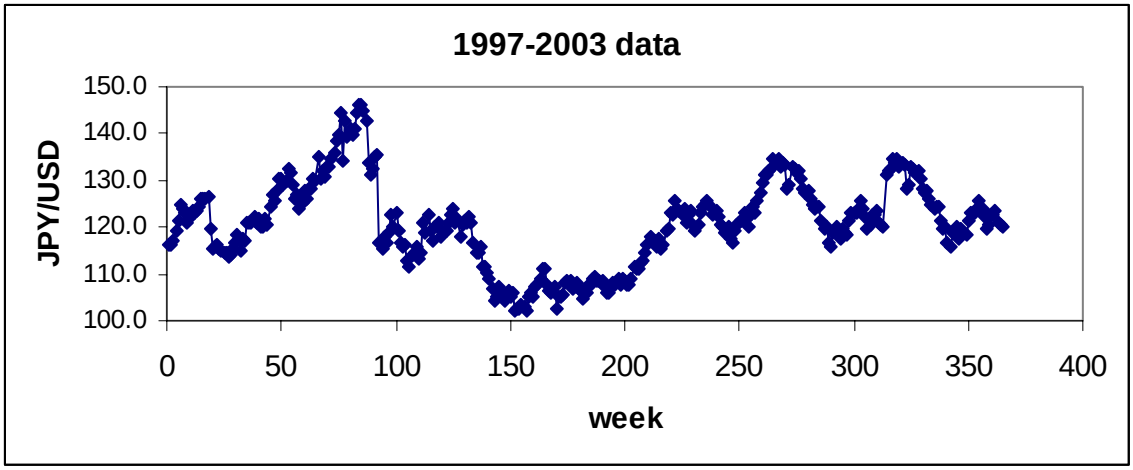


week	GBP/USD	week	GBP/USD
1	0.5583	27	0.5464
2	0.5413	28	0.5389
3	0.5557	29	0.5343
4	0.5491	30	0.5457
5	0.5491	31	0.5496
6	0.5417	32	0.5417
7	0.5310	33	0.5425
8	0.5357	34	0.5490
9	0.5382	35	0.5579
10	0.5426	36	0.5631
11	0.5572	37	0.5561
12	0.5477	38	0.5579
13	0.5522	39	0.5546
14	0.5465	40	0.5563
15	0.5455	41	0.5574
16	0.5555	42	0.5541
17	0.5657	43	0.5472
18	0.5637	44	0.5451
19	0.5604	45	0.5391
20	0.5685	46	0.5392
21	0.5591	47	0.5375
22	0.5456	48	0.5273
23	0.5444	49	0.5163
24	0.5499	50	0.5227
25	0.5444	51	0.5157
26	0.5489	52	0.5199
		53	0.5219

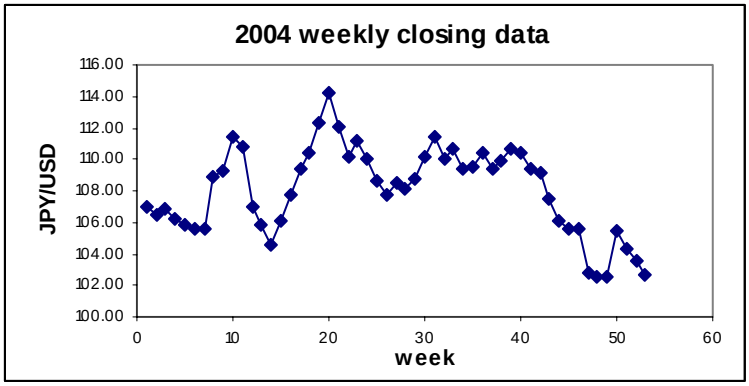


5. JPY/USD Weekly Closing Data

Week	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003
1	135.150	124.850	125.299	111.868	101.318	105.110	116.320	132.440	111.530	105.190	116.200	131.010	131.010
2	134.140	127.145	126.076	111.247	98.781	105.120	116.090	131.590	114.140	105.790	117.750	132.140	132.140
3	132.640	127.403	124.830	111.213	99.338	105.410	117.090	129.160	114.640	104.920	117.210	132.630	132.630
4	132.250	123.404	124.764	109.430	99.298	106.560	119.020	126.020	115.950	107.120	117.130	134.460	134.460
5	131.450	125.593	124.368	109.093	99.687	106.640	121.270	126.970	113.140	107.550	115.600	133.750	133.750
6	127.800	125.146	120.807	107.682	98.858	107.110	124.580	124.110	114.340	108.950	117.560	134.790	134.790
7	130.450	127.896	119.129	104.156	97.482	105.310	124.060	125.370	120.970	110.920	115.460	132.790	132.790
8	132.190	128.701	118.008	104.670	97.069	104.720	123.280	127.720	118.600	110.970	116.120	133.880	133.880
9	134.510	129.197	117.679	105.626	94.364	105.810	120.820	126.090	122.520	107.780	119.020	133.340	133.340
10	136.190	131.747	117.737	105.068	91.179	105.940	121.930	128.230	119.050	106.220	119.650	128.210	128.210
11	137.150	133.556	115.977	106.140	89.163	105.870	123.480	128.150	117.010	106.060	122.900	129.080	129.080
12	137.140	134.005	116.299	104.883	89.091	106.820	122.880	130.330	120.210	107.240	122.570	132.890	132.890
13	140.610	133.101	114.222	103.751	86.787	106.920	123.620	130.060	120.160	102.650	125.590	132.610	132.610
14	136.860	133.247	113.202	105.072	84.333	107.670	124.170	134.940	121.030	105.310	124.010	131.750	131.750
15	136.250	132.849	112.593	103.590	83.340	108.610	126.020	130.140	117.890	105.080	123.510	132.100	132.100
16	138.350	134.007	110.622	103.632	82.865	107.100	125.950	131.990	119.260	105.410	122.490	130.470	130.470
17	138.150	134.545	111.209	102.031	83.978	105.550	126.040	130.960	119.380	108.030	123.730	128.170	128.170
18	138.510	132.695	110.322	102.695	84.047	104.850	126.540	132.990	120.820	108.520	121.050	127.200	127.200
19	138.760	133.356	110.862	104.894	86.973	105.510	119.760	132.790	122.700	108.700	122.460	127.870	127.870
20	138.500	130.199	110.322	103.966	86.746	106.540	115.510	134.430	123.900	106.880	123.630	126.000	126.000
21	138.410	129.397	107.232	104.248	82.799	107.870	115.630	135.730	120.850	107.120	120.720	124.720	124.720
22	138.390	127.699	107.723	105.315	84.925	108.070	116.430	138.510	121.970	108.120	119.130	124.130	124.130
23	140.300	126.751	106.070	103.736	84.402	109.160	115.320	139.690	118.160	106.750	120.620	124.460	124.460
24	140.550	126.648	109.237	102.731	84.642	108.850	114.870	144.390	120.510	106.240	122.930	124.360	124.360
25	138.940	126.897	106.095	100.471	84.462	109.040	114.620	133.980	121.430	104.610	124.200	121.240	121.240
26	137.960	125.797	108.725	98.507	84.725	109.470	114.720	142.930	121.130	106.120	124.710	119.850	119.850
27	138.350	124.705	109.622	98.164	86.694	110.850	113.850	139.350	122.350	107.790	125.810	120.180	120.180
28	136.800	125.400	108.148	98.078	87.854	110.700	113.920	141.040	121.010	107.860	124.740	116.690	116.690
29	136.450	124.602	106.756	98.556	88.568	108.260	115.390	139.620	116.500	109.080	122.800	115.720	115.720
30	137.800	127.606	104.995	100.015	88.170	108.310	116.770	140.920	114.630	109.590	123.260	118.780	118.780
31	136.740	127.347	104.410	100.412	91.252	107.000	118.400	144.600	114.620	108.510	123.480	119.040	119.040
32	136.790	127.493	102.231	100.261	93.761	108.150	114.910	146.350	115.630	108.620	122.040	120.150	120.150
33	137.240	126.252	104.972	98.517	96.871	108.000	117.720	146.310	111.420	108.410	120.370	117.550	117.550
34	137.100	126.048	103.915	100.426	96.613	108.530	117.010	144.910	111.680	107.120	120.110	119.520	119.520
35	136.850	123.254	105.023	99.158	97.460	108.690	120.740	142.670	110.180	105.780	118.780	118.750	118.750
36	136.150	123.146	106.273	99.151	99.673	109.350	120.930	133.950	109.000	106.000	120.040	118.500	118.500
37	133.900	124.193	104.435	98.707	103.486	110.420	121.050	131.090	106.970	107.320	117.440	121.530	121.530
38	134.490	124.651	105.959	97.620	98.471	109.780	122.040	132.640	104.140	107.910	116.770	123.170	123.170
39	133.340	120.388	106.017	99.025	99.027	110.840	120.740	135.170	105.090	107.870	119.220	122.870	122.870
40	129.950	119.362	105.531	100.387	100.772	111.640	121.910	135.560	107.330	108.770	120.340	123.060	123.060
41	129.750	121.861	107.034	98.378	100.830	111.690	119.930	116.810	105.410	107.630	120.920	124.030	124.030
42	129.760	119.750	108.193	97.347	100.185	112.520	120.200	115.410	105.760	109.080	121.170	125.510	125.510
43	131.400	121.996	108.407	97.325	101.632	113.340	121.960	117.900	104.240	108.700	123.040	124.120	124.120
44	130.290	123.261	108.126	97.577	103.798	113.230	120.340	116.590	106.320	107.510	121.660	122.150	122.150
45	130.310	123.146	106.071	97.945	100.949	111.620	124.180	118.720	105.310	107.850	120.230	119.700	119.700
46	130.110	123.873	108.287	98.437	102.015	111.260	126.910	122.700	106.120	108.810	122.680	120.380	120.380
47	129.500	124.189	108.715	98.759	101.477	111.420	125.820	120.220	102.170	111.370	124.430	122.850	122.850
48	130.050	124.385	108.376	100.344	101.170	113.820	127.620	123.130	102.570	111.180	123.220	122.720	122.720
49	128.050	124.918	109.177	100.188	100.933	112.880	130.160	119.280	102.380	111.300	125.630	123.670	123.670
50	128.810	123.812	110.091	100.230	102.178	113.790	130.460	116.770	103.330	112.580	127.360	120.730	120.730
51	127.550	123.226	110.894	100.129	102.582	114.160	129.000	115.690	102.960	112.770	129.470	120.520	120.520
52	125.500	123.890	111.635	99.560	103.268	116.060	129.880	116.190	102.140	114.340	131.300	119.920	119.920
53		124.739						112.950					



week	JPY/USD	week	JPY/USD
1	106.94	27	108.56
2	106.51	28	108.19
3	106.82	29	108.78
4	106.18	30	110.10
5	105.84	31	111.40
6	105.55	32	110.04
7	105.59	33	110.67
8	108.83	34	109.34
9	109.25	35	109.50
10	111.43	36	110.46
11	110.75	37	109.35
12	106.99	38	109.94
13	105.88	39	110.71
14	104.51	40	110.42
15	106.08	41	109.44
16	107.70	42	109.10
17	109.39	43	107.45
18	110.37	44	106.04
19	112.29	45	105.59
20	114.26	46	105.53
21	112.07	47	102.81
22	110.17	48	102.53
23	111.23	49	102.50
24	110.08	50	105.52
25	108.64	51	104.27
26	107.80	52	103.61
		53	102.71



Appendix 2. Experiment I Results

1. AUS/USD Forecasting

a. Analysis of Variance for MSE

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	0.038018	6	0.006336	**4.510715**	0.000329	2.163929
H. Nodes	0.011363	9	0.001263	0.898786	0.528268	1.947349
Interaction	0.065961	54	0.001221	0.869548	0.717438	1.429385
Within	0.196664	140	0.001405			
Total	0.312006	209				

b. Analysis of Variance for MAPE

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	1013.332	6	168.8887	**9.131233**	2.72E-08	2.171308
H. Nodes	195.5425	8	24.44282	1.321539	0.238671	2.012655
Interaction	789.1833	48	16.44132	0.888925	0.673284	1.45825
Within	2330.461	126	18.49572			
Total	4328.519	188				

c. Analysis of Variance for Percentage of True Directional Changes

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	5712.117	6	952.0194	**10.78606**	7.62E-10	2.163929
H. Nodes	1556.513	9	172.9459	**1.959418**	0.048437	1.947349
Interaction	8814.057	54	163.2233	**1.849264**	0.002199	1.429385
Within	12356.95	140	88.26391			
Total	28439.63	209				

2. EUR/USD Forecasting

a. Analysis of Variance for MSE

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	0.000899	6	0.00015	**6.051318**	1.18E-05	2.163929
H. Nodes	9.41E-05	9	1.05E-05	0.421909	0.921658	1.947349
Interaction	0.001218	54	2.26E-05	0.910507	0.646628	1.429385
Within	0.003468	140	2.48E-05			
Total	0.00568	209				

b. Analysis of Variance for MAPE

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	559.737	6	93.28949	**11.4103**	2.3E-10	2.163929
H. Nodes	39.76754	9	4.418616	0.540444	0.842951	1.947349
Interaction	428.7783	54	7.940338	0.971188	0.53779	1.429385
Within	1144.626	140	8.175902			
Total	2172.909	209				

c. Analysis of Variance for Percentage of True Directional Changes

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	357.4503	6	59.57505	**2.21917**	0.044671	2.163929
H. Nodes	428.3081	9	47.58978	1.772719	0.078544	1.947349
Interaction	2103.524	54	38.95414	**1.451042**	0.043241	1.429385
Within	3758.389	140	26.84564			
Total	6647.672	209				

3. CHF/USD Forecasting

a. Analysis of Variance for MSE

ANOVA						5%	10%
SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*	*F crit*
Input Nodes	0.00193	6	0.000322	**5.252956**	6.57E-05	2.163929	1.816638
H. Nodes	0.000339	9	3.77E-05	0.615436	0.782382	1.947349	1.676653
Interaction	0.003679	54	6.81E-05	1.112295	0.306573	1.429385	1.320619
Within	0.008574	140	6.12E-05				
Total	0.014522	209					

b. Analysis of Variance for MAPE

ANOVA						5%	10%
SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*	*F crit*
Input Nodes	267.3359	6	44.55598	**10.29503**	1.98E-09	2.163929	1.816638
H. Nodes	27.46637	9	3.051818	0.705148	0.703387	1.947349	1.676653
Interaction	326.3288	54	6.043126	***1.396314***	0.06214	1.429385	1.320619
Within	605.9079	140	4.327913				
Total	1227.039	209					

c. Analysis of Variance for Percentage of True Directional Changes

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	260.0613	6	43.34356	1.363211	0.233615	2.16392948
Hidden N.	490.8704	9	54.54115	1.71539	0.09078	1.947348949
Interaction	2650.51	54	49.08351	**1.543741**	0.022683	1.429384611
Within	4451.326	140	31.79518			
Total	7852.767	209				

4. GBP/USD Forecasting

a. Analysis of Variance for MSE

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	0.00011	6	2E-05	**5.5363**	4E-05	2.1639
H. Nodes	2.7E-05	9	3E-06	0.9366	0.4956	1.9473
Interaction	0.00014	54	3E-06	0.771	0.8618	1.4294
Within	0.00046	140	3E-06			
Total	0.00073	209				

b. Analysis of Variance for MAPE

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	373	6	62.09	**3.966**	0.0011	2.164
H. Nodes	109	9	12.14	0.775	0.6394	1.947
Interaction	832	54	15.4	0.984	0.5149	1.429
Within	2192	140	15.65			
Total	3505	209				

c. Analysis of Variance for Percentage of True Directional Changes

ANOVA

SV	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Input Nodes	6194	6	1032	**22.04**	4E-18	2.164
H. Nodes	959.7	9	106.6	**2.277**	0.0206	1.947
Interaction	7884	54	146	**3.118**	4E-08	1.429
Within	6556	140	46.83			
Total	21594	209				

1. Experiment II for AUS/USD and the corresponding effects and percent contribution

Runs	MSE	MAPE	Dir
1	0.001	1.96	67.34
2	0.001	1.99	69.38
3	0.008	5.15	65.30
4	0.001	1.82	67.34
5	0.001	2.07	65.30
6	0.001	1.98	59.18
7	0.001	2.40	65.30
8	0.001	1.84	57.14
9	0.001	2.07	67.34
10	0.001	2.16	53.06
11	0.001	1.79	63.26
12	0.001	1.87	61.22
13	0.023	8.78	42.85
14	0.001	1.98	61.22
15	0.001	1.81	65.30
16	0.001	2.16	63.26

	Effect			**% Contribution**		
Activation Function 1	0.00	-1.28	-1.27	10.94	13.02	0.97
Activation Function 2	0.00	-0.52	2.81	2.93	2.14	4.73
Learning rate	0.00	0.48	-4.60	2.93	1.82	12.7
Momentum term	0.00	0.42	-1.02	6.93	7.00	7.23
Statistical Performance	MSE	MAPE	Dir	MSE	MAPE	Dir

2. Experiment II for EUR/USD and the corresponding effects and percent contribution

Runs	MSE	MAPE	Dir
1	0.0010	3.16	53.48
2	0.0010	3.76	46.51
3	0.0004	1.71	44.18
4	0.0010	3.57	55.81
5	0.0010	3.89	53.48
6	0.0004	1.79	46.51
7	0.0002	1.50	51.16
8	0.0003	1.57	48.83
9	0.0010	3.22	58.13
10	0.0004	1.86	41.86
11	0.0003	1.58	48.83
12	0.0006	2.42	48.83
13	0.0030	5.78	58.13
14	0.0020	4.90	51.16
15	0.0005	2.01	41.86
16	0.0010	3.31	39.53

	Effect			% Contribution		
Activation Function 1	0.00	0.04	-1.27	0.39	0.03	0.97
Activation Function 2	-0.01	-1.34	2.81	23.98	**28.55**	4.72
Learning rate	0.00	0.47	-4.60	7.75	3.64	12.7
Momentum term	0.00	-0.15	-1.02	3.09	1.47	7.235
Statistical Performance	MSE	MAPE	Dir	MSE	MAPE	Dir

3. Experiment II for CHF/USD and the corresponding effects and percent contribution

Runs	MSE	MAPE	Dir
1	0.0006	1.43	52.08
2	0.0005	1.37	52.08
3	0.0005	1.28	50.00
4	0.0008	1.67	56.25
5	0.001	2.3	60.41
6	0.0005	1.44	43.75
7	0.0005	1.28	52.08
8	0.0005	1.43	45.83
9	0.0005	1.34	50.00
10	0.0006	1.52	43.75
11	0.0005	1.33	47.91
12	0.0005	1.42	47.92
13	0.002	2.84	58.33
14	0.0005	1.45	43.75
15	0.001	2.09	50.00
16	0.0005	1.44	50.00

	Effect			% Contribution		
Activation Function 1	0.00	-0.27	-4.69	13.17	10.17	**23.6**
Activation Function 2	0.00	-0.22	-0.52	5.33	6.74	0.29
Learning rate	0.00	0.25	-1.04	7.40	10.98	3.785
Momentum term	0.00	0.12	1.04	2.02	2.44	1.455
Statistical Performance	MSE	MAPE	Dir	MSE	MAPE	Dir

4. Experiment II for GBP/USD and the corresponding effects and percent contribution

Runs	MSE	MAPE	Dir
1	0.000	1.48	59.18
2	5.631	1.02	65.30
3	5.408	0.99	61.22
4	7.616	1.20	65.30
5	7.450	1.11	57.14
6	5.381	1.01	65.30
7	6.439	1.07	61.22
8	5.552	1.16	65.30
9	5.670	1.00	61.22
10	5.549	1.00	59.18
11	7.114	1.12	61.22
12	7.008	1.16	65.30
13	0.000	1.36	61.22
14	0.001	2.99	61.22
15	0.000	1.33	59.18
16	7.578	1.16	61.22

	Effect			**% Contribution**		
Activation Function 1	1.53	0.02	3.32	7.14	2.75	**41.52**
Activation Function 2	2.13	-0.22	1.28	13.84	5.66	6.14
Learning rate	-1.39	0.27	-1.02	5.87	8.27	4.18
Momentum term	1.57	-0.31	-1.02	7.51	10.67	4.18
Statistical Performance	MSE	MAPE	Dir	MSE	MAPE	Dir

Appendix 4. Pseudo Codes

1. Pseudo Code of Modified Tabu Search for NNs Learning

```
%Input necessary data
n=number of input nodes
p=number of hidden nodes
mahead=m multiple ahead forecast
data set being forecasted
maxitt=maximum number of iteration without improvement
MAXGEN=number of neighborhoods searched
MAXLOC=number of local search
TL= tabu length
TC= tabu criterion

%Determine the number of variables in the model
NVAR = n*p+2*p+1

%Neighborhood Search Looping
NH=1; %NH is counter for number of neighborhood searched

while NH <= MAXGEN

 % Initialize population
 X=randn(NVAR,1)

 % Evaluate initial population
 NN= OBJTABU(X)

 % Track best individual
 NNBEST=NN;
 XBEST=X';

%Record Tabu List
 TABUNN(1)=NN;
 TABUX(1,:)=X';

Numitt=1; %Counter for number of iterations that give the same results

%Local Search Looping
NS=1; %Counter for number of local search

 while NS <= MAXLOC
   X=X0+randn(NVAR,1)/100.*X0;
   NN=ObjTABU(X);
         if NN < NNBEST
```

```
        NNBEST=NN;
        XBEST=X';
        Move TABUNN one step forward
        Move TABUX
        Numitt=1
           else if NN~= TABUNN*(1±TC/100))  and  X ~= TABUX*(1±TC/100))
                Move TABUNN one step forward
                Move TABUX one step forward
                Numitt=Numitt+1
             end;
        end;
   NS=NS+1; %Increment Counter for number of local search
   end;

if Numitt > maxitt, STOP


NH = NH+1;%Increment Counter for number of neigborhood

% Update display and record current best individual
end;

%display the chosen weight matrices
v=reshape(XCHOSEN(1:1:n*p),n,p)
bv=reshape(XCHOSEN((n*p+1):1:(n*p+p)),1,p)
w=reshape(XCHOSEN((n*p+p+1):1:(n*p+2*p)),p,1)
bw=XCHOSEN(n*p+2*p+1)
```

2. Pseudo Code of Genetic Algorithm for NNs Learning

```
%Input necessary data
n=number of input nodes
p=number of hidden nodes
mahead=m multiple ahead forecast
Nind=number of individuals in each generation
data set being forecasted
maxitt=maximum number of iteration without improvement
MAXGEN=maximum number of generations
XOVR=crossover rate
MUTR=mutation rate

%Determine the number of variables in the model
Nvar=n*p+2*p+1;

%Initialize Population
Chrom=randn(Nind, Nvar);

%Evaluate initial population
ObjVal=objku(Chrom);

% Reset counters
Best = NaN*ones(MAXGEN,1);       % best in current population

gen=1;

% Track best individual
[Best(gen),xi] = min(ObjVal);

% Matrix for storing best individuals
IndAll = [];
IndAll = [IndAll; Chrom(xi,:)];

%Generational loop
numitt=0;

while gen < MAXGEN

%Assign fitness values to entire population
FitnV=ranking(ObjVal);

%Determine the best individual in this generation
new=Chrom(xi,:);

%Select individuals for breeding using roulette wheel selection
```

```
%Apply Crossover (linear crossover) and Mutation

%Apply elitsm scheme
SelCh=[SelCh; new];

% Evaluate offspring, call objective function
ObjVSel = objku(SelCh);

%Insert best offspring in population replacing worst parents
[Chrom ObjVal]=reins(Chrom,SelCh ,ObjVal,ObjVSel);

%Update record current best individual and record all best invididual for each
%generation;
[Best(gen+1),xi] = min(ObjVal)
IndAll = [IndAll; Chrom(xi,:)];

%Calculate the number of iteration without improvement
if Best(gen+1)== Best(gen)
   numitt=numitt+1;
else
   numitt=0;
end;

%Stop if the number of iteration without improvement = maxitt
if numitt > maxitt
   genstop=gen
   gen=(MAXGEN-1);

end;

%Increment counter
gen=gen+1;

end;

%Translate selected individual  into appropriate weight matrices
v=reshape(chosen(1:1:n*p),n,p);
bv=reshape(chosen((n*p+1):1:(n*p+p)),1,p);
w=reshape(chosen((n*p+p+1):1:(n*p+2*p)),p,1);
bw=chosen(n*p+2*p+1);
```

3. Pseudo Code of Modified Tabu Search for NNs Topology Determination

```
%Input necessary variables
MAXGEN= number of neighborhoods search
MAXLOC= number of local search
TL= tabu length
TC= tabu criterion
The selected data

%Initialize Necessary Matrices
TABUNN=Tabu list of the objective function
TABUT=Tabu list of the topology

%Determine number of variables for Topology Determination
NVAR = 6

%Neighborhood Search looping
NH=1; %Counter for Number of Neighborhood

while NH <= MAXGEN

 % Initialize population T0 using random number

 % Evaluate initial population using BP training
 NN= OBJTOPSPLIT(T0);

 % Track best individual
NNBEST=NN;
TBEST=T0';
TABUNN(1)=NN;
TABUT(1,:)=T0';

NS=1; %Counter for number of local search
X0=T0(2:end);

%Local Search Looping
 while NS <= MAXLOC %local search
    X(1)=X0(1)+randint(1,1,[-2 2]);
    X(2)=randint(1,1,[0 1]);
    X(3)=randint(1,1,[0 1]);
    X(4)=X0(4)+randn*0.1;
    X(5)=X0(5)+randn*0.1;

    T=[T0(1:1);X]
    NN=ObjTOPSPLIT(T)
```

```
        if NN < NNBEST
        NNBEST=NN;
        TBEST=T';
        Move TABUNN one step forward
        Move TABUX
        Numitt=1;
           else if repmat(NN,[TL,1]) ~= TABUNN  ;
              if repmat(T',[TL,1]) ~= TABUT ;
                 Move TABUNN one step forward
                 Move TABUX one step forward
                 Numitt=Numitt+1;
              end;
           end;
        end;

   NS=NS+1 ;%Increment Counter for number of local search

end;

if Numitt > Maxitt , STOP


NH = NH+1%Increment Counter for number of neighborhood

% Update display and record current best individual

end;

%Display the chosen individual
```

4. Pseudo Code of Genetic Algorithm for NNs Topology Determination

```
%Input necessary data
mahead=m multiple ahead forecast
Nind=number of individuals in each generation
data set being forecasted
maxitt=maximum number of iteration without improvement
MAXGEN=maximum number of generations
XOVR=crossover rate
MUTR=mutation rate

%Determine the variables in the model and its precision
Nvar=number of variables  (example: 6)
PRECI1=precision of variables 1
PRECI2=precision of variables 2
PRECI3=precision of variables 3
PRECI4=precision of variables 4
PRECI5=precision of variables 5
PRECI6=precision of variables 6

%Calculate the number of bits (length) of each individual
LIND=PRECI1+PRECI2+PRECI3+ PRECI4+PRECI5+PRECI6

% Build field descriptor
FieldD=Range of values of each variable

% Initialize population using gray coding
%Chrom=chromosomes in one generation

%Evaluate initial population
ObjV = GATOPObj(Chrom)

% Track best individual
[Best(gen),xi] = min(ObjVal);

% Matrix for storing best individuals
IndAll = [];
IndAll = [IndAll; Chrom(xi,:)];

%Generational loop
numitt=0;

while gen < MAXGEN

%Assign fitness values to entire population
FitnV=ranking(ObjVal);
```

```
% Select individuals for breeding using roulette wheel selection method
SelCh = select('rws', Chrom)


% Recombine individuals (crossover) using double point crossover
SelCh = recombin('xovdp',SelCh,xovr)


% Apply mutation
SelCh = mut(SelCh);


% Evaluate offspring, call objective function
ObjVSel = GATOPObj(SelCh);


% Reinsert offspring into population
[Chrom ObjV]=reins(Chrom,SelCh,ObjV,ObjVSel);

%Update record current best individual and record all best individual for each generation
[Best(gen+1),xi] = min(ObjV);
IndAll = [IndAll; Chrom(xi,:)];

%Increment counter
gen=gen+1;

end;

%Determine the best topology
```

Appendix 5. Summary Statistics for Residual Distribution Fittings

1. EUR/USD One Step Ahead Forecasting

Fit All Summary

Function	Sq Error
Normal	0.00647
Weibull	0.00673
Beta	0.00873
Erlang	0.01900
Gamma	0.01990
Triangular	0.02040
Lognormal	0.03010
Uniform	0.07650
Exponential	0.11000

Distribution Summary

Distribution:	Normal
Expression:	NORM(-0.00185,0.0186)
Square Error:	0.006466
Chi Square Test	
Number of intervals	6
Degrees of freedom	3
Test Statistic	4.92
Corresponding p-value	0.193
Kolmogorov-Smirnov Test	
Test Statistic	0.0579
Corresponding p-value	> 0.15

2. EUR/USD Three Step Ahead Forecasting

Fit All Summary

Function	Sq Error
Normal	0.00348
Weibull	0.00449
Beta	0.00459
Gamma	0.00892
Erlang	0.00911
Triangular	0.01350
Lognormal	0.01570
Uniform	0.06780
Exponential	0.10100

Distribution Summary

Distribution:	Normal
Expression:	NORM(-0.000504, 0.0273)
Square Error:	0.003476
Chi Square Test	
Number of intervals	6
Degrees of freedom	3
Test Statistic	4.58
Corresponding p-value	0.217
Kolmogorov-Smirnov Test	
Test Statistic	0.0582
Corresponding p-value	> 0.15

3. GBP/USD One Step Ahead Forecasting

Fit All Summary

Function	Sq Error
Beta	0.00524
Normal	0.00556
Weibull	0.00749
Erlang	0.00777
Gamma	0.00821
Lognormal	0.01230
Triangular	0.03250
Uniform	0.08480
Exponential	0.11100

Distribution Summary

Distribution:	Normal
Expression:	NORM(-0.000185, 0.00744)
Square Error:	0.005558
Chi Square Test	
Number of intervals	9
Degrees of freedom	6
Test Statistic	15.5
Corresponding p-value	0.0182
Kolmogorov-Smirnov Test	
Test Statistic	0.0550
Corresponding p-value	> 0.15

4. GBP/USD Three Step Ahead Forecasting

Fit All Summary

Function	Sq Error
Normal	0.00249
Weibull	0.00313
Beta	0.00381
Gamma	0.00775
Erlang	0.00780
Lognormal	0.01260
Triangular	0.01670
Uniform	0.06160
Exponential	0.08780

Distribution Summary

Distribution:	Normal
Expression:	NORM(0.0000939, 0.0115)
Square Error:	0.002494
Chi Square Test	
Number of intervals	9
Degrees of freedom	6
Test Statistic	10.70
Corresponding p-value	0.0988
Kolmogorov-Smirnov Test	
Test Statistic	0.0348
Corresponding p-value	> 0.15

Appendix 6. The Predicted Values of The Replicates

3. GBP/USD One Step Ahead Forecasting

	Predict	1	2	3	4	5	6	7	8	9	10
1	0.6233	0.6223	0.6238	0.6235	0.6232	0.6238	0.6217	0.6206	0.6230	0.6235	0.6202
2	0.6212	0.6206	0.6225	0.6216	0.6211	0.6205	0.6203	0.6188	0.6206	0.6218	0.6190
3	0.6170	0.6174	0.6194	0.6179	0.6170	0.6151	0.6176	0.6158	0.6163	0.6182	0.6166
4	0.6119	0.6135	0.6150	0.6132	0.6124	0.6098	0.6141	0.6125	0.6114	0.6135	0.6134
5	0.6146	0.6153	0.6138	0.6148	0.6155	0.6168	0.6165	0.6157	0.6158	0.6139	0.6143
6	0.6204	0.6198	0.6193	0.6202	0.6208	0.6229	0.6202	0.6193	0.6213	0.6196	0.6180
7	0.6258	0.6243	0.6242	0.6253	0.6252	0.6293	0.6233	0.6232	0.6267	0.6247	0.6214
8	0.6281	0.6267	0.6279	0.6285	0.6276	0.6316	0.6245	0.6252	0.6288	0.6281	0.6235
9	0.6261	0.6248	0.6277	0.6265	0.6251	0.6253	0.6232	0.6221	0.6251	0.6273	0.6224
10	0.6267	0.6252	0.6257	0.6266	0.6262	0.6301	0.6238	0.6239	0.6274	0.6261	0.6222
11	0.6287	0.6275	0.6287	0.6293	0.6279	0.6327	0.6249	0.6260	0.6296	0.6286	0.6241
12	0.6288	0.6279	0.6299	0.6300	0.6287	0.6324	0.6250	0.6262	0.6296	0.6298	0.6247
13	0.6291	0.6283	0.6300	0.6304	0.6287	0.6334	0.6252	0.6269	0.6302	0.6298	0.6249
14	0.6288	0.6279	0.6302	0.6300	0.6287	0.6318	0.6250	0.6260	0.6293	0.6300	0.6247
15	0.6284	0.6272	0.6292	0.6292	0.6282	0.6311	0.6247	0.6253	0.6287	0.6291	0.6240
16	0.6271	0.6257	0.6281	0.6275	0.6266	0.6280	0.6238	0.6235	0.6267	0.6279	0.6230
17	0.6250	0.6238	0.6259	0.6253	0.6247	0.6254	0.6226	0.6217	0.6246	0.6256	0.6214
18	0.6239	0.6228	0.6243	0.6241	0.6238	0.6247	0.6220	0.6211	0.6237	0.6241	0.6206
19	0.6203	0.6200	0.6225	0.6209	0.6198	0.6182	0.6197	0.6178	0.6192	0.6214	0.6186
20	0.6149	0.6158	0.6176	0.6160	0.6151	0.6129	0.6163	0.6145	0.6143	0.6163	0.6153
21	0.6126	0.6140	0.6143	0.6135	0.6135	0.6121	0.6148	0.6136	0.6127	0.6133	0.6136
22	0.6061	0.6089	0.6111	0.6081	0.6062	0.6026	0.6093	0.6078	0.6048	0.6089	0.6099
23	0.6010	0.6045	0.6039	0.6022	0.6021	0.5997	0.6050	0.6051	0.6007	0.6026	0.6055
24	0.6011	0.6044	0.6025	0.6019	0.6024	0.6010	0.6052	0.6057	0.6014	0.6017	0.6052
25	0.6059	0.6084	0.6055	0.6064	0.6072	0.6077	0.6099	0.6099	0.6072	0.6055	0.6084
26	0.6024	0.6058	0.6072	0.6044	0.6029	0.5995	0.6061	0.6053	0.6013	0.6052	0.6072
27	0.6105	0.6118	0.6067	0.6105	0.6114	0.6154	0.6138	0.6139	0.6130	0.6082	0.6109
28	0.6243	0.6227	0.6206	0.6231	0.6235	0.6293	0.6225	0.6225	0.6260	0.6218	0.6200
29	0.6227	0.6219	0.6253	0.6232	0.6213	0.6197	0.6211	0.6190	0.6211	0.6242	0.6203
30	0.6224	0.6214	0.6214	0.6222	0.6225	0.6247	0.6213	0.6205	0.6230	0.6217	0.6193
31	0.6231	0.6222	0.6236	0.6233	0.6231	0.6237	0.6216	0.6205	0.6229	0.6234	0.6201
32	0.6247	0.6234	0.6240	0.6246	0.6246	0.6271	0.6226	0.6221	0.6251	0.6242	0.6209
33	0.6276	0.6261	0.6268	0.6276	0.6269	0.6313	0.6243	0.6248	0.6284	0.6271	0.6230
34	0.6282	0.6269	0.6287	0.6288	0.6279	0.6309	0.6245	0.6251	0.6285	0.6288	0.6238
35	0.6272	0.6259	0.6280	0.6277	0.6269	0.6285	0.6239	0.6237	0.6270	0.6280	0.6230
36	0.6246	0.6235	0.6259	0.6249	0.6240	0.6241	0.6223	0.6211	0.6238	0.6254	0.6212
37	0.6184	0.6186	0.6216	0.6194	0.6176	0.6153	0.6184	0.6162	0.6171	0.6201	0.6176
38	0.6078	0.6103	0.6134	0.6098	0.6074	0.6035	0.6106	0.6086	0.6061	0.6109	0.6112
39	0.6026	0.6059	0.6051	0.6038	0.6038	0.6017	0.6066	0.6065	0.6026	0.6039	0.6067
40	0.6014	0.6048	0.6037	0.6025	0.6026	0.6006	0.6054	0.6056	0.6013	0.6026	0.6057
41	0.5990	0.6027	0.6019	0.6002	0.6001	0.5977	0.6030	0.6034	0.5985	0.6006	0.6039
42	0.5910	0.5946	0.5958	0.5927	0.5911	0.5888	0.5936	0.5938	0.5889	0.5943	0.5973
43	0.5886	0.5915	0.5887	0.5880	0.5894	0.5882	0.5907	0.5932	0.5878	0.5886	0.5926
44	0.5956	0.5988	0.5930	0.5957	0.5962	0.5982	0.5995	0.6024	0.5967	0.5939	0.5989
45	0.5949	0.5987	0.5985	0.5961	0.5957	0.5930	0.5984	0.5991	0.5937	0.5971	0.6006
46	0.5882	0.5913	0.5913	0.5889	0.5887	0.5868	0.5903	0.5910	0.5864	0.5904	0.5937
47	0.5821	0.5832	0.5834	0.5821	0.5827	0.5821	0.5832	0.5814	0.5810	0.5831	0.5853
48	0.5794	0.5795	0.5794	0.5788	0.5804	0.5801	0.5803	0.5779	0.5798	0.5791	0.5801
49	0.5757	0.5740	0.5767	0.5768	0.5770	0.5788	0.5791	0.5680	0.5781	0.5760	0.5751

11	12	13	14	15	16	17	18	19	20
0.6204	0.6218	0.6232	0.6216	0.6238	0.6239	0.6187	0.6222	0.6224	0.6228
0.6197	0.6201	0.6212	0.6206	0.6220	0.6238	0.6171	0.6202	0.6200	0.6206
0.6177	0.6169	0.6175	0.6182	0.6181	0.6226	0.6143	0.6166	0.6157	0.6164
0.6146	0.6131	0.6133	0.6146	0.6129	0.6193	0.6113	0.6128	0.6113	0.6115
0.6130	0.6184	0.6173	0.6154	0.6140	0.6100	0.6139	0.6158	0.6157	0.6140
0.6169	0.6215	0.6217	0.6192	0.6201	0.6163	0.6174	0.6202	0.6207	0.6199
0.6202	0.6239	0.6257	0.6220	0.6255	0.6199	0.6211	0.6242	0.6263	0.6256
0.6229	0.6246	0.6276	0.6239	0.6285	0.6250	0.6230	0.6268	0.6289	0.6272
0.6233	0.6221	0.6242	0.6234	0.6272	0.6271	0.6203	0.6243	0.6246	0.6253
0.6214	0.6242	0.6264	0.6228	0.6267	0.6224	0.6218	0.6252	0.6272	0.6264
0.6234	0.6248	0.6282	0.6242	0.6292	0.6252	0.6238	0.6272	0.6300	0.6273
0.6243	0.6248	0.6285	0.6246	0.6298	0.6269	0.6240	0.6284	0.6301	0.6267
0.6243	0.6250	0.6289	0.6247	0.6301	0.6266	0.6246	0.6284	0.6309	0.6267
0.6244	0.6246	0.6282	0.6246	0.6299	0.6273	0.6239	0.6285	0.6298	0.6264
0.6239	0.6245	0.6277	0.6244	0.6291	0.6267	0.6232	0.6276	0.6289	0.6268
0.6233	0.6233	0.6260	0.6238	0.6279	0.6267	0.6215	0.6258	0.6265	0.6261
0.6219	0.6224	0.6243	0.6227	0.6257	0.6256	0.6198	0.6237	0.6240	0.6245
0.6207	0.6222	0.6238	0.6219	0.6244	0.6241	0.6192	0.6228	0.6232	0.6235
0.6198	0.6187	0.6198	0.6203	0.6215	0.6247	0.6162	0.6191	0.6185	0.6197
0.6164	0.6154	0.6158	0.6168	0.6160	0.6213	0.6131	0.6151	0.6139	0.6143
0.6138	0.6151	0.6148	0.6148	0.6130	0.6163	0.6121	0.6137	0.6127	0.6122
0.6121	0.6056	0.6069	0.6099	0.6075	0.6183	0.6072	0.6082	0.6054	0.6064
0.6061	0.6030	0.6035	0.6045	0.6011	0.6068	0.6044	0.6043	0.6022	0.6022
0.6046	0.6048	0.6042	0.6042	0.6006	0.6017	0.6049	0.6045	0.6030	0.6023
0.6068	0.6116	0.6096	0.6084	0.6049	0.6008	0.6086	0.6089	0.6081	0.6063
0.6092	0.6021	0.6037	0.6064	0.6034	0.6144	0.6048	0.6055	0.6024	0.6034
0.6071	0.6173	0.6141	0.6109	0.6083	0.5942	0.6121	0.6130	0.6134	0.6102
0.6173	0.6236	0.6242	0.6199	0.6232	0.6086	0.6204	0.6219	0.6256	0.6241
0.6219	0.6192	0.6206	0.6217	0.6243	0.6265	0.6174	0.6207	0.6203	0.6221
0.6185	0.6223	0.6231	0.6204	0.6222	0.6191	0.6185	0.6217	0.6224	0.6219
0.6203	0.6218	0.6231	0.6215	0.6236	0.6237	0.6186	0.6221	0.6223	0.6227
0.6203	0.6232	0.6248	0.6219	0.6247	0.6221	0.6201	0.6236	0.6247	0.6244
0.6221	0.6245	0.6271	0.6233	0.6277	0.6233	0.6226	0.6259	0.6284	0.6270
0.6236	0.6244	0.6275	0.6242	0.6288	0.6264	0.6229	0.6273	0.6286	0.6269
0.6233	0.6236	0.6263	0.6238	0.6280	0.6266	0.6217	0.6260	0.6268	0.6263
0.6220	0.6217	0.6236	0.6226	0.6255	0.6260	0.6193	0.6230	0.6232	0.6241
0.6193	0.6167	0.6177	0.6193	0.6200	0.6248	0.6148	0.6174	0.6164	0.6178
0.6139	0.6062	0.6078	0.6115	0.6096	0.6209	0.6080	0.6095	0.6064	0.6078
0.6070	0.6054	0.6054	0.6061	0.6026	0.6072	0.6057	0.6057	0.6039	0.6035
0.6059	0.6041	0.6041	0.6048	0.6013	0.6053	0.6049	0.6047	0.6028	0.6025
0.6045	0.6004	0.6012	0.6023	0.5991	0.6049	0.6029	0.6026	0.6003	0.6005
0.5998	0.5863	0.5902	0.5939	0.5918	0.6044	0.5948	0.5956	0.5909	0.5936
0.5905	0.5876	0.5887	0.5887	0.5880	0.5870	0.5936	0.5918	0.5904	0.5910
0.5945	0.6013	0.5982	0.5961	0.5937	0.5875	0.6014	0.5995	0.5994	0.5974
0.6018	0.5935	0.5959	0.5978	0.5952	0.6033	0.5992	0.5990	0.5958	0.5971
0.5948	0.5843	0.5874	0.5903	0.5887	0.5960	0.5921	0.5924	0.5884	0.5909
0.5844	0.5806	0.5803	0.5846	0.5825	0.5840	0.5836	0.5845	0.5811	0.5839
0.5774	0.5803	0.5784	0.5807	0.5795	0.5780	0.5800	0.5794	0.5789	0.5800
0.5753	0.5845	0.5770	0.5831	0.5765	0.5708	0.5714	0.5742	0.5730	0.5743

4. GBP/USD Three Step Ahead Forecasting

	Predict	1	2	3	4	5	6	7	8	9	10
1	0.6228	0.6216	0.6183	0.6145	0.6186	0.6214	0.6217	0.6226	0.6204	0.6211	0.6219
2	0.6190	0.6202	0.6170	0.6137	0.6173	0.6195	0.6198	0.6203	0.6182	0.6193	0.6200
3	0.6126	0.6172	0.6144	0.6119	0.6148	0.6158	0.6159	0.6158	0.6146	0.6160	0.6168
4	0.6070	0.6131	0.6111	0.6095	0.6116	0.6114	0.6110	0.6104	0.6106	0.6118	0.6124
5	0.6172	0.6140	0.6114	0.6101	0.6119	0.6137	0.6115	0.6135	0.6137	0.6123	0.6118
6	0.6227	0.6189	0.6154	0.6128	0.6157	0.6188	0.6174	0.6197	0.6184	0.6178	0.6185
7	0.6284	0.6228	0.6194	0.6150	0.6194	0.6237	0.6232	0.6257	0.6237	0.6233	0.6250
8	0.6298	0.6248	0.6226	0.6164	0.6224	0.6263	0.6272	0.6287	0.6264	0.6266	0.6278
9	0.6235	0.6238	0.6213	0.6160	0.6216	0.6241	0.6262	0.6256	0.6228	0.6237	0.6225
10	0.6289	0.6236	0.6206	0.6156	0.6205	0.6247	0.6248	0.6268	0.6247	0.6246	0.6262
11	0.6304	0.6252	0.6234	0.6166	0.6231	0.6270	0.6281	0.6297	0.6276	0.6277	0.6291
12	0.6301	0.6257	0.6244	0.6170	0.6242	0.6275	0.6294	0.6301	0.6279	0.6280	0.6278
13	0.6306	0.6257	0.6248	0.6169	0.6244	0.6278	0.6296	0.6307	0.6287	0.6288	0.6294
14	0.6297	0.6258	0.6246	0.6171	0.6245	0.6274	0.6296	0.6300	0.6276	0.6278	0.6266
15	0.6293	0.6254	0.6235	0.6168	0.6234	0.6267	0.6285	0.6291	0.6267	0.6270	0.6268
16	0.6266	0.6245	0.6221	0.6164	0.6222	0.6251	0.6269	0.6270	0.6244	0.6249	0.6244
17	0.6242	0.6230	0.6200	0.6154	0.6202	0.6230	0.6241	0.6245	0.6220	0.6228	0.6231
18	0.6238	0.6221	0.6188	0.6148	0.6190	0.6220	0.6224	0.6234	0.6210	0.6217	0.6225
19	0.6158	0.6197	0.6167	0.6134	0.6171	0.6187	0.6194	0.6192	0.6171	0.6187	0.6194
20	0.6103	0.6156	0.6130	0.6110	0.6135	0.6140	0.6138	0.6136	0.6129	0.6143	0.6150
21	0.6107	0.6132	0.6111	0.6096	0.6116	0.6120	0.6109	0.6112	0.6115	0.6117	0.6119
22	0.5977	0.6085	0.6078	0.6069	0.6084	0.6064	0.6063	0.6043	0.6057	0.6078	0.6085
23	0.5982	0.6024	0.6036	0.6035	0.6041	0.6018	0.6008	0.5988	0.6023	0.6018	0.6009
24	0.6013	0.6018	0.6031	0.6032	0.6036	0.6019	0.6003	0.5988	0.6027	0.6009	0.5992
25	0.6090	0.6059	0.6057	0.6056	0.6063	0.6062	0.6036	0.6041	0.6069	0.6043	0.6022
26	0.5952	0.6048	0.6052	0.6048	0.6059	0.6031	0.6029	0.6005	0.6030	0.6044	0.6046
27	0.6182	0.6088	0.6073	0.6070	0.6079	0.6101	0.6061	0.6092	0.6112	0.6067	0.6038
28	0.6287	0.6206	0.6170	0.6136	0.6170	0.6221	0.6201	0.6239	0.6226	0.6207	0.6225
29	0.6168	0.6215	0.6187	0.6146	0.6191	0.6208	0.6226	0.6216	0.6190	0.6206	0.6204
30	0.6243	0.6205	0.6169	0.6137	0.6172	0.6205	0.6197	0.6218	0.6200	0.6198	0.6208
31	0.6227	0.6215	0.6182	0.6144	0.6184	0.6212	0.6215	0.6225	0.6203	0.6209	0.6217
32	0.6263	0.6224	0.6190	0.6149	0.6191	0.6227	0.6226	0.6244	0.6222	0.6223	0.6235
33	0.6297	0.6243	0.6216	0.6160	0.6214	0.6257	0.6261	0.6280	0.6258	0.6258	0.6276
34	0.6292	0.6252	0.6231	0.6167	0.6230	0.6264	0.6280	0.6288	0.6263	0.6267	0.6268
35	0.6272	0.6245	0.6221	0.6164	0.6222	0.6253	0.6269	0.6273	0.6246	0.6252	0.6249
36	0.6226	0.6227	0.6198	0.6153	0.6200	0.6226	0.6239	0.6239	0.6214	0.6223	0.6224
37	0.6119	0.6185	0.6156	0.6127	0.6161	0.6171	0.6179	0.6171	0.6154	0.6174	0.6181
38	0.5975	0.6104	0.6091	0.6078	0.6098	0.6079	0.6082	0.6061	0.6068	0.6095	0.6106
39	0.6006	0.6039	0.6046	0.6044	0.6051	0.6033	0.6020	0.6005	0.6037	0.6030	0.6021
40	0.5997	0.6025	0.6036	0.6036	0.6042	0.6021	0.6009	0.5991	0.6027	0.6018	0.6006
41	0.5962	0.6004	0.6022	0.6023	0.6027	0.6000	0.5991	0.5967	0.6006	0.6000	0.5990
42	0.5830	0.5933	0.5966	0.5970	0.5970	0.5922	0.5931	0.5895	0.5930	0.5939	0.5946
43	0.5894	0.5892	0.5922	0.5931	0.5924	0.5904	0.5897	0.5866	0.5912	0.5886	0.5874
44	0.6030	0.5945	0.5968	0.5979	0.5975	0.5975	0.5941	0.5932	0.5986	0.5940	0.5897
45	0.5903	0.5965	0.5993	0.5997	0.5998	0.5961	0.5959	0.5926	0.5969	0.5966	0.5962
46	0.5827	0.5900	0.5935	0.5941	0.5937	0.5894	0.5904	0.5869	0.5904	0.5900	0.5906
47	0.5774	0.5851	0.5861	0.5867	0.5857	0.5831	0.5852	0.5838	0.5835	0.5829	0.5853

11	12	13	14	15	16	17	18	19	20
0.6223	0.6200	0.6179	0.6205	0.6216	0.6206	0.6241	0.6206	0.6199	0.6220
0.6202	0.6187	0.6171	0.6185	0.6191	0.6191	0.6203	0.6188	0.6185	0.6201
0.6159	0.6159	0.6153	0.6148	0.6147	0.6161	0.6135	0.6152	0.6155	0.6165
0.6107	0.6120	0.6125	0.6105	0.6100	0.6122	0.6084	0.6109	0.6114	0.6120
0.6128	0.6123	0.6130	0.6132	0.6120	0.6125	0.6100	0.6131	0.6122	0.6155
0.6189	0.6169	0.6162	0.6182	0.6182	0.6172	0.6151	0.6181	0.6170	0.6200
0.6249	0.6211	0.6185	0.6231	0.6252	0.6218	0.6252	0.6231	0.6213	0.6244
0.6281	0.6237	0.6197	0.6255	0.6294	0.6251	0.6263	0.6258	0.6238	0.6269
0.6259	0.6229	0.6190	0.6231	0.6250	0.6240	0.6240	0.6232	0.6225	0.6244
0.6261	0.6221	0.6190	0.6240	0.6268	0.6231	0.6285	0.6241	0.6223	0.6253
0.6291	0.6243	0.6199	0.6262	0.6308	0.6259	0.6245	0.6267	0.6244	0.6277
0.6297	0.6249	0.6201	0.6265	0.6314	0.6268	0.6220	0.6270	0.6248	0.6282
0.6302	0.6251	0.6202	0.6269	0.6324	0.6271	0.6221	0.6276	0.6251	0.6287
0.6297	0.6250	0.6201	0.6264	0.6312	0.6269	0.6216	0.6269	0.6248	0.6281
0.6288	0.6244	0.6199	0.6258	0.6299	0.6260	0.6227	0.6261	0.6242	0.6273
0.6270	0.6234	0.6194	0.6242	0.6270	0.6246	0.6243	0.6243	0.6231	0.6255
0.6244	0.6216	0.6187	0.6221	0.6239	0.6224	0.6281	0.6222	0.6214	0.6235
0.6230	0.6205	0.6182	0.6211	0.6225	0.6211	0.6255	0.6212	0.6204	0.6226
0.6194	0.6184	0.6168	0.6176	0.6179	0.6188	0.6200	0.6179	0.6181	0.6192
0.6138	0.6143	0.6142	0.6131	0.6127	0.6145	0.6111	0.6135	0.6139	0.6147
0.6112	0.6119	0.6127	0.6113	0.6104	0.6121	0.6089	0.6115	0.6115	0.6131
0.6050	0.6083	0.6088	0.6054	0.6053	0.6084	0.6038	0.6059	0.6071	0.6053
0.5997	0.6029	0.6038	0.6011	0.6005	0.6034	0.6002	0.6012	0.6017	0.6006
0.5996	0.6022	0.6032	0.6013	0.6000	0.6029	0.6004	0.6012	0.6012	0.6013
0.6041	0.6053	0.6067	0.6057	0.6037	0.6059	0.6038	0.6055	0.6048	0.6075
0.6013	0.6051	0.6058	0.6022	0.6023	0.6054	0.6011	0.6026	0.6038	0.6014
0.6082	0.6074	0.6084	0.6099	0.6078	0.6078	0.6063	0.6092	0.6076	0.6127
0.6228	0.6188	0.6169	0.6219	0.6233	0.6192	0.6172	0.6216	0.6193	0.6233
0.6222	0.6207	0.6177	0.6197	0.6203	0.6212	0.6269	0.6200	0.6202	0.6210
0.6210	0.6185	0.6172	0.6198	0.6205	0.6189	0.6183	0.6198	0.6187	0.6215
0.6222	0.6199	0.6179	0.6204	0.6215	0.6204	0.6236	0.6205	0.6198	0.6219
0.6237	0.6206	0.6183	0.6220	0.6236	0.6212	0.6247	0.6220	0.6207	0.6234
0.6273	0.6230	0.6194	0.6249	0.6284	0.6241	0.6287	0.6251	0.6231	0.6263
0.6284	0.6241	0.6198	0.6255	0.6295	0.6256	0.6236	0.6258	0.6240	0.6270
0.6271	0.6234	0.6195	0.6244	0.6273	0.6247	0.6246	0.6245	0.6232	0.6257
0.6240	0.6215	0.6185	0.6216	0.6231	0.6223	0.6279	0.6218	0.6212	0.6230
0.6176	0.6174	0.6160	0.6159	0.6160	0.6176	0.6172	0.6164	0.6169	0.6174
0.6069	0.6100	0.6101	0.6068	0.6067	0.6100	0.6050	0.6074	0.6089	0.6068
0.6012	0.6040	0.6052	0.6026	0.6017	0.6046	0.6015	0.6027	0.6030	0.6027
0.6000	0.6029	0.6039	0.6015	0.6006	0.6035	0.6005	0.6015	0.6018	0.6013
0.5978	0.6012	0.6018	0.5993	0.5989	0.6018	0.5987	0.5993	0.5999	0.5981
0.5903	0.5955	0.5939	0.5914	0.5938	0.5956	0.5913	0.5915	0.5933	0.5886
0.5887	0.5903	0.5876	0.5888	0.5891	0.5911	0.5905	0.5891	0.5893	0.5863
0.5945	0.5949	0.5949	0.5961	0.5940	0.5962	0.5962	0.5957	0.5945	0.5957
0.5939	0.5981	0.5977	0.5953	0.5960	0.5986	0.5952	0.5954	0.5964	0.5927
0.5880	0.5921	0.5895	0.5886	0.5910	0.5922	0.5892	0.5887	0.5902	0.5865
0.5839	0.5857	0.5817	0.5836	0.5857	0.5845	0.5840	0.5827	0.5850	0.5836

3. EUR/USD Three Step Ahead Forecasting

	Predict	1	2	3	4	5	6	7	8	9	10
1	0.9210	0.9021	0.9220	0.9092	0.9190	0.9073	0.9183	0.9134	0.9229	0.9125	0.9236
2	0.9161	0.9026	0.9183	0.9089	0.9177	0.9103	0.9192	0.9133	0.9207	0.9117	0.9191
3	0.9223	0.9052	0.9166	0.9087	0.9197	0.9127	0.9182	0.9131	0.9212	0.9115	0.9203
4	0.9195	0.9030	0.9173	0.9088	0.9186	0.9122	0.9173	0.9129	0.9202	0.9130	0.9148
5	0.9199	0.9043	0.9163	0.9087	0.9188	0.9124	0.9178	0.9131	0.9199	0.9118	0.9172
6	0.9207	0.9040	0.9165	0.9087	0.9188	0.9126	0.9174	0.9128	0.9201	0.9123	0.9163
7	0.9208	0.9039	0.9168	0.9087	0.9190	0.9126	0.9175	0.9128	0.9203	0.9125	0.9164
8	0.9088	0.9016	0.9125	0.9083	0.9143	0.9095	0.9154	0.9100	0.9151	0.9103	0.9105
9	0.9248	0.9053	0.9136	0.9082	0.9191	0.9140	0.9171	0.9122	0.9187	0.9100	0.9194
10	0.9279	0.8995	0.9202	0.9090	0.9196	0.9136	0.9169	0.9032	0.9250	0.9158	0.9175
11	0.9183	0.9005	0.9211	0.9091	0.9202	0.9100	0.9187	0.9132	0.9213	0.9142	0.9151
12	0.9223	0.9048	0.9180	0.9089	0.9195	0.9115	0.9186	0.9134	0.9225	0.9114	0.9227
13	0.9210	0.9035	0.9184	0.9089	0.9198	0.9121	0.9180	0.9128	0.9215	0.9132	0.9170
14	0.9141	0.9029	0.9154	0.9086	0.9169	0.9111	0.9175	0.9123	0.9181	0.9113	0.9146
15	0.9079	0.9020	0.9094	0.9078	0.9133	0.9089	0.9149	0.9084	0.9122	0.9078	0.9118
16	0.8990	0.8991	0.9013	0.9061	0.9071	0.9018	0.9080	0.9023	0.9032	0.9027	0.9047
17	0.8781	0.8898	0.8831	0.9010	0.8937	0.8862	0.8935	0.8876	0.8867	0.8894	0.8929
18	0.8708	0.8820	0.8736	0.8897	0.8835	0.8778	0.8835	0.8809	0.8721	0.8777	0.8868
19	0.8547	0.8698	0.8625	0.8588	0.8640	0.8670	0.8618	0.8623	0.8598	0.8630	0.8682
20	0.8485	0.8622	0.8567	0.8370	0.8614	0.8653	0.8583	0.8565	0.8520	0.8587	0.8674
21	0.8563	0.8625	0.8633	0.8552	0.8599	0.8653	0.8536	0.8590	0.8538	0.8586	0.8609
22	0.8486	0.8600	0.8604	0.8528	0.8561	0.8630	0.8490	0.8533	0.8516	0.8547	0.8564
23	0.8591	0.8648	0.8637	0.8482	0.8634	0.8674	0.8596	0.8612	0.8553	0.8619	0.8677
24	0.8846	0.8741	0.8782	0.8752	0.8688	0.8732	0.8669	0.8783	0.8684	0.8731	0.8699
25	0.8804	0.8811	0.8800	0.8723	0.8736	0.8754	0.8729	0.8773	0.8738	0.8780	0.8751
26	0.8922	0.8876	0.8866	0.8893	0.8853	0.8836	0.8866	0.8900	0.8802	0.8856	0.8879
27	0.9003	0.8904	0.8936	0.8943	0.8881	0.8880	0.8894	0.8945	0.8891	0.8932	0.8882
28	0.8784	0.8880	0.8828	0.8973	0.8848	0.8812	0.8843	0.8846	0.8826	0.8860	0.8848
29	0.8936	0.8901	0.8883	0.8959	0.8926	0.8878	0.8939	0.8950	0.8828	0.8882	0.8948
30	0.8952	0.8891	0.8915	0.8929	0.8854	0.8854	0.8860	0.8910	0.8878	0.8917	0.8849
31	0.8965	0.8937	0.8928	0.9004	0.8945	0.8905	0.8953	0.8971	0.8894	0.8932	0.8946
32	0.9210	0.8971	0.9066	0.9050	0.9039	0.9058	0.9069	0.9063	0.9061	0.9058	0.9048
33	0.9137	0.8965	0.9108	0.9075	0.9070	0.9069	0.9077	0.9090	0.9123	0.9110	0.9012
34	0.9071	0.9015	0.9068	0.9072	0.9112	0.9067	0.9120	0.9076	0.9088	0.9063	0.9088
35	0.8920	0.8966	0.8961	0.9050	0.9024	0.8964	0.9028	0.8976	0.8982	0.8993	0.8997
36	0.8880	0.8927	0.8888	0.9007	0.8967	0.8898	0.8971	0.8937	0.8876	0.8913	0.8968
37	0.8792	0.8873	0.8816	0.8947	0.8844	0.8806	0.8843	0.8847	0.8795	0.8837	0.8856
38	0.8680	0.8801	0.8720	0.8842	0.8768	0.8744	0.8763	0.8760	0.8696	0.8744	0.8803
39	0.8518	0.8672	0.8593	0.8540	0.8629	0.8660	0.8605	0.8604	0.8570	0.8608	0.8677
40	0.8575	0.8663	0.8629	0.8427	0.8648	0.8679	0.8624	0.8617	0.8558	0.8629	0.8705
41	0.8507	0.8608	0.8630	0.8591	0.8557	0.8630	0.8480	0.8551	0.8532	0.8552	0.8544
42	0.8589	0.8659	0.8636	0.8438	0.8648	0.8680	0.8621	0.8621	0.8559	0.8631	0.8704
43	0.8784	0.8722	0.8754	0.8735	0.8664	0.8710	0.8632	0.8741	0.8656	0.8699	0.8667
44	0.8592	0.8714	0.8672	0.8586	0.8634	0.8675	0.8607	0.8626	0.8621	0.8648	0.8661
45	0.8429	0.8575	0.8506	0.8337	0.8574	0.8627	0.8532	0.8523	0.8489	0.8542	0.8630
46	0.8324	0.8458	0.8478	0.8334	0.8512	0.8591	0.8394	0.8401	0.8404	0.8475	0.8513

11	12	13	14	15	16	17	18	19	20
0.9280	0.9188	0.9234	0.9175	0.9251	0.9205	0.8797	0.9229	0.9184	0.9159
0.9235	0.9175	0.9206	0.9157	0.9215	0.9174	0.8967	0.9204	0.9153	0.9138
0.9228	0.9161	0.9201	0.9156	0.9205	0.9157	0.9067	0.9187	0.9172	0.9131
0.9221	0.9155	0.9185	0.9158	0.9201	0.9152	0.9054	0.9177	0.9176	0.9114
0.9217	0.9156	0.9188	0.9155	0.9196	0.9151	0.9068	0.9179	0.9168	0.9121
0.9217	0.9153	0.9185	0.9157	0.9197	0.9149	0.9073	0.9175	0.9175	0.9117
0.9220	0.9154	0.9188	0.9159	0.9200	0.9151	0.9067	0.9177	0.9177	0.9118
0.9171	0.9147	0.9141	0.9124	0.9160	0.9118	0.9073	0.9153	0.9122	0.9097
0.9199	0.9139	0.9181	0.9143	0.9177	0.9122	0.9128	0.9161	0.9169	0.9117
0.9249	0.9154	0.9213	0.9177	0.9235	0.9170	0.9050	0.9189	0.9200	0.9096
0.9259	0.9174	0.9208	0.9169	0.9232	0.9179	0.8912	0.9208	0.9189	0.9130
0.9244	0.9172	0.9217	0.9158	0.9220	0.9173	0.9013	0.9203	0.9169	0.9143
0.9237	0.9163	0.9199	0.9165	0.9214	0.9164	0.9029	0.9190	0.9184	0.9125
0.9203	0.9158	0.9174	0.9144	0.9186	0.9144	0.9053	0.9176	0.9146	0.9116
0.9142	0.9136	0.9126	0.9106	0.9133	0.9103	0.9095	0.9139	0.9097	0.9090
0.9061	0.9096	0.9042	0.9050	0.9056	0.9038	0.9066	0.9072	0.9047	0.9041
0.8897	0.9017	0.8883	0.8893	0.8900	0.8918	0.8932	0.8945	0.8923	0.8940
0.8768	0.8855	0.8778	0.8778	0.8749	0.8813	0.8792	0.8830	0.8856	0.8853
0.8604	0.8734	0.8623	0.8600	0.8601	0.8681	0.8623	0.8645	0.8645	0.8695
0.8517	0.8548	0.8577	0.8539	0.8520	0.8606	0.8616	0.8587	0.8612	0.8643
0.8552	0.8590	0.8573	0.8584	0.8534	0.8610	0.8590	0.8549	0.8513	0.8591
0.8510	0.8591	0.8551	0.8527	0.8511	0.8589	0.8544	0.8535	0.8461	0.8567
0.8571	0.8574	0.8599	0.8619	0.8547	0.8624	0.8636	0.8583	0.8593	0.8640
0.8718	0.8758	0.8690	0.8782	0.8691	0.8741	0.8676	0.8653	0.8645	0.8693
0.8774	0.8857	0.8735	0.8800	0.8761	0.8795	0.8717	0.8733	0.8724	0.8771
0.8855	0.8896	0.8836	0.8917	0.8831	0.8859	0.8821	0.8843	0.8897	0.8876
0.8922	0.8960	0.8885	0.8968	0.8923	0.8916	0.8867	0.8882	0.8899	0.8893
0.8859	0.8971	0.8820	0.8859	0.8857	0.8869	0.8845	0.8859	0.8881	0.8881
0.8891	0.8918	0.8888	0.8934	0.8859	0.8884	0.8881	0.8908	0.8964	0.8928
0.8905	0.8960	0.8858	0.8930	0.8910	0.8899	0.8850	0.8859	0.8861	0.8872
0.8943	0.8988	0.8918	0.8979	0.8929	0.8935	0.8916	0.8938	0.8982	0.8948
0.9068	0.9040	0.9058	0.9108	0.9071	0.9026	0.9034	0.9036	0.9083	0.9010
0.9122	0.9099	0.9082	0.9092	0.9131	0.9069	0.9077	0.9076	0.9076	0.9018
0.9113	0.9116	0.9093	0.9092	0.9104	0.9076	0.9091	0.9108	0.9091	0.9069
0.9012	0.9075	0.8988	0.9007	0.9011	0.8999	0.9028	0.9028	0.9012	0.9007
0.8927	0.8999	0.8918	0.8941	0.8914	0.8934	0.8935	0.8960	0.8966	0.8959
0.8838	0.8937	0.8811	0.8857	0.8827	0.8858	0.8820	0.8847	0.8882	0.8874
0.8735	0.8840	0.8730	0.8747	0.8718	0.8782	0.8737	0.8768	0.8806	0.8808
0.8569	0.8683	0.8605	0.8567	0.8571	0.8659	0.8613	0.8630	0.8637	0.8680
0.8577	0.8581	0.8611	0.8619	0.8554	0.8636	0.8651	0.8609	0.8644	0.8669
0.8530	0.8634	0.8559	0.8538	0.8526	0.8600	0.8530	0.8538	0.8453	0.8557
0.8579	0.8574	0.8611	0.8627	0.8553	0.8632	0.8652	0.8603	0.8635	0.8663
0.8689	0.8742	0.8662	0.8734	0.8662	0.8718	0.8647	0.8628	0.8612	0.8666
0.8635	0.8764	0.8631	0.8632	0.8625	0.8693	0.8617	0.8637	0.8608	0.8683
0.8452	0.8529	0.8544	0.8465	0.8488	0.8575	0.8570	0.8563	0.8560	0.8612
0.8331	0.8382	0.8476	0.8399	0.8425	0.8461	0.8492	0.8463	0.8361	0.8487

Appendix 7. Pseudo Code of Multivariate Neural Network

1. Pseudo Code of Trivariate Neural Network for Time Series Forecasting

```
% Define input windows for each time series under consideration
n1=str2num(get(handles.n1,'string'));
n2=str2num(get(handles.n2,'string'));
n3=str2num(get(handles.n3,'string'));

%Calculate the number of input nodes
n=n1+n2+n3;

%Input number of hidden nodes
p=str2num(get(handles.hnodes,'string'));

%Input the values of learning rate, momentum term, and number of iterations
alpha=learning rate
mu=momentum term

%Input the training data using switch argument
DATA1=the first time series data
DATA2= the second time series data
DATA3=the third time series data

%Transform input data to data in the range of  [ 0 1]
INPUTDATA1=transformed first time series data
INPUTDATA2=transformed second time series data
INPUTDATA3=transformed third time series data

%Initialize weight using random initialization
w=randn(p,1)*0.5;
v=randn(n,p)*0.5;
bv=randn(1,p)*0.5;
bw=randn(1,1)*0.5;

%BP training main loop

% Set the iteration
iter=1;

%Determine the number of pairs for each iteration
nmax=max(n1,n2,n3)
[aa bb]=size of INPUTDATA1
nn=aa-nmax
```

```
While iter < maxit
%Set pair number=1

While training pair (tp) < nn

%input the data
X(tp,:)=[INPUTDATA1((tp+nmax-1):-1:(nmax-n1+tp));INPUTDATA2((tp+nmax-1):-
        1:(nmax-n2+tp)); INPUTDATA3((tp+nmax-1):-1:(nmax-n3+tp)) ]';

%input the target data
TAR=INPUTDATA1(nmax+mahead+tp-1);

%For each training pair, do BP training

%Increase the number of training pair by 1
tp=tp+1;

%End of training for all pairs

%Increase the number of  iteration by 1
iter=iter+1;

%End of BP training

%Retransform the actual values and the predicted values to the original range
```

2. Pseudo Code of Genetic Algorithm for Multivariate Topology Determination

```
% Input the necessary data
MAXGEN=maximum number of generation
NIND=number of individuals in each generation
DATA1=the first time series (data being forecasted)
DATA2=the second time series
DATA3=the third time series
XOVR=crossover rate
MUTR=mutation rate

%Determine the variables in the model and its precision
NVAR = number of variables (example:  4, input windows for each time series and h
nodes, learning rate and momentum are preset)

PRECI1=precision of variable 1
PRECI2=precision of variable 2
PRECI3=precision of variable 3
PRECI4=precision of variable 4

%Calculate the number of bits (length) of each individual
LIND=PRECI1+PRECI2+PRECI3+PRECI4

% Build field descriptor
FieldD=Range of values of each variable

% Initialize population using gray coding
%Chrom=chromosomes in one generation
Chrom = crtbp(NIND, LIND);

% Evaluate initial population
ObjV = OBJTOPMUL(Chrom)

% Generational loop

% Track best individual
[Best(gen+1),xi] = min(ObjV);

% Matrix for storing best individuals
IndAll = [];
IndAll = [IndAll; Chrom(xi,:)];

%Generational loop
while gen < MAXGEN,
```

```
% Assign fitness values to entire population
FitnV = ranking(ObjV)

% Select individuals for breeding using roulette wheel selection method
SelCh = select('rws', Chrom)

% Recombine individuals (crossover) using double point crossover
SelCh = recombin('xovdp',SelCh,xovr);

% Apply mutation
SelCh = mut(SelCh);

% Evaluate offspring, call objective function
ObjVSel = objTopMUL(bs2rv(SelCh,FieldD));

% Reinsert offspring into population
[Chrom ObjV]=reins(Chrom,SelCh,ObjV,ObjVSel);

% Update record current best individual and record all best individual for each generation
 Best(gen) = min(ObjV)
IndAll=[IndAll; Chrom(xi,:)]

% Increment counter
gen = gen+1

end;

%Determine the chosen topology
```

3. The Objective Function for Genetic Algorithm for Multivariate NNs Topology Determination

```
function ObjVal = OBJTOPMUL(Chrom);

% Compute population parameters
[Nind,Nvar] = size(Chrom);

%Determine the data series
DATA1=The first time series data (the data being forecasted)
DATA2=The second time series data
DATA3=The third time series data

%Evaluation of The Objective Values for each individual
for irun=1:Nind

%Translate individual to the corresponding topology
n1=Chrom(irun,1)
n2=Chrom(irun,2)
n3=Chrom(irun,3)
p=Chrom(irun,4)

%Pre set other parameters
lrate=0.9, mterm=0.02, activation function is tanh (=1)

%Calculate the number of input nodes
n=n1+n2+n3;

%Initialize weight matrices
v=randn(n,p);
bv=randn(1,p);
w=randn(p,1);
bw=randn(1,1);

%determine the number of training pair
nmax=max(n1,n2,n3)
[aa bb]=size(DATA1)
tp=aa-nmax

for epochs=1:MAXGEN
tp=1;

for tp=1:nn

%input the data
```

```
X(tp,:)=[DATA1((tp+nmax-1):-1:(nmax-n1+tp));DATA2((tp+nmax-1):-1:(nmax-
n2+tp)); DATA3((tp+nmax-1):-1:(nmax-n3+tp)) ]';

%input the target data
TAR=DATA1(nmax+mahead+tp-1);

%Do the BP training for each pair



% end for one pair of training set

%Increment counter
tp=tp+1;

end;

%Increment counter
epochs=epochs+1;

end;

%Determine the MSE for each individual
ObjVal(irun)=mse

end;

%Obtain the MSE values for each individual
```