1997

Financial Prediction and Trading Strategies Using Neurofuzzy Approaches

K. N. Pantazopoulos

L. H. Tsoukalas

N. G. Bourbakis

M. J. Brün

Elias N. Houstis
Purdue University, enh@cs.purdue.edu

Report Number:
97-002

FINANCIAL PREDICTION AND TRADING STRATEGIES USING NEUROFUZZY APPROACHES

K. N. Pantazopoulos
L. H. Tsoukalas
N. G. Bourbakis
M. J. Brun
E. N. Houstis

CSD-TR 97-002
January 1997

Financial Prediction and Trading Strategies Using Neurofuzzy Approaches

K.N. Pantazopoulos, L.H. Tsoukalas,
N.G. Bourbakis, M.J. Brün, and E.N. Houstis

Abstract

Neurofuzzy approaches for predicting financial time series are investigated and shown to perform well in the context of various trading strategies. The horizon of prediction is typically a few days and trading strategies are examined using historical data. A methodology is presented where neural predictors are used to anticipate the general behavior of financial indices (*moving up*, *down*, or *staying constant*) in the context of stocks and options trading. The methodology is tested with actual financial data and shows considerable promise as a decision making and planning tool.

Acknowledgments: This research has been partially supported by INEL University Research Consortium. The INEL is managed by Lockheed Martin Idaho Technologies Co., for the US Department of Energy under contract No. DE-AC07-94ID3223; also by NSF grants 9123502-CDA, 92022536-CCR, and 620-92-J-0069, and ARPA grant DAAH04-94-G-0010. Finally, the gracious support of the Purdue Special Initiative Fellowship and the Purdue Research Foundation is gratefully acknowledged. The authors are also grateful to **A. Hobbs** for substantial contributions to the stock pricing model.

1. Introduction

The prediction of financial market indicators is a topic of considerable practical interest and, if successful, may involve substantial pecuniary rewards. Neural networks have been used for several years in the selection of investments because of their ability to identify patterns of behavior that are not readily observable. Much of this work has been proprietary for the obvious reason that the users want to take advantage of the insight into the market they gained through the use of neural network technology. Thus, conventional wisdom has it that if one succeeds in developing a methodology that correctly predicts the ups and downs of various market indicators she or he is unlikely to seek publication of results; conversely, if something is published it must really not "make money." We take both views as indicative of the fact that the ultimate test of value for financial prediction lies with the market. Be that as it may, financial markets, and economic systems in general, present fascinating opportunities for studying complexity and testing the limits of methodological advances and computational technologies (Anderson et al, 1988). Unlike the natural systems of physics and chemistry, financial systems involve the actual interplay of decisions and actions taken by millions of individual investors and institutions on a global scale.

Although participants in the market are constantly involved in making decisions based on their predictions or anticipations of market phenomena, there is considerable disagreement amongst experts as to what degree, if any, financial time series are predictable and how to predict them. Some researchers have identified the presence of chaos in financial indicators, implying that financial systems are characterized by non-repetitive and non-predictable fluctuations arising through the interplay of a system's participating agents and its relation to other systems (Radzicki, 1990). Others have argued for certain underlying regularities and patterns and hence for a more predictable market system (Lippit, 1990), (Peters, 1994). The "efficient market hypothesis," currently the most widely held view of market behavior, states that no investment "system" or technical strategy can yield average returns exceeding the average returns of the market as a whole. Any information that would yield extra returns spreads rapidly and is "priced out" within seconds of its introduction.

Although nearly everybody agrees on the complex and nonlinear nature of economic systems, there is skepticism as to whether new approaches to nonlinear modeling, such as neural networks, can improve economic and financial forecasts. Some researchers claim that neural networks offer no major improvement over conventional linear forecasting approaches (Farber, 1988), (LeBaron, 1994). One possible explanation may be that, while empirical studies in the natural sciences are characterized by large data sets, often numbering in the tens of thousands, data sets in economic applications usually consist of less than one thousand observations. Consequently, computational procedures designed in the former context may not be appropriate in the latter (Ramsey, 1990). In addition, there is a great variety of neural computing paradigms, involving various architectures, learning rates, etc., and hence, precise and informative comparisons may be difficult to make. Lapedes and Farber, amongst others, have offered interesting and convincing evidence that in the context of time series prediction, neural nets can very accurately model the non-linearities involved (Lapedes, 1988). In recent years, an increasing number of research in the emerging and promising field of financial engineering is incorporating neurofuzzy approaches (Refenes, 1996), (Trippi, 1996).

We use a neurofuzzy methodology to guide simulated trades in options based on the level of the Standard & Poor's 500 (S&P 500) index, and find that if we use a standard option pricing model (we use the *Black & Scholes model* found in Black, 1973), our trades would yield returns more than 150 times greater than a portfolio fully invested directly in the S&P 500 index. It would not be in line with the efficient market hypothesis, however, to expect such results from actual investment. The promise of high returns would raise the price of the options until normal return rates were again reached.

Hobbs and Bourbakis have described a neurofuzzy simulator used for stock investing (Hobbs, 1995) that identifies patterns associated with whether a stock is under-priced or over-priced. This model is expanded in Section 4 and neural predictions are used in the context of two related but different trading strategies, the first based on *stocks*, and the second based on *options*. A stock is a certificate of ownership representing one or more shares of a corporation's equity. Usually stocks are listed, i.e., they are traded in the financial markets and world's stock exchanges. An option is a financial instrument, a form of contract between two parties, which is also traded in the financial markets and organized exchanges, much like common stocks, commodities, currencies, etc. The basic difference between options and entities such as stocks, is that options are derivative products, i.e., they are based on a primary asset such as a corporation's stock. Options can also be based on aggregate metrics such as market indices. Market indices are aggregate measures (weighted averages) of stocks. The "price" of a market index fluctuates over time, representing the changes in the prices of the individual stocks. The price of the various market indices is continuously calculated and quoted by various services.

In the case of the stock trading strategy, a neural predictor is designed based on the assumption that "the price of a stock should generally change proportionately with (or against) certain market indices, e.g., the Dow Jones Industrial Average (DJIA), or the S&P 500." The neural network, after analyzing these market variables, predicts the true price for a particular stock x, and decides whether it is under- or over-priced. Although the overall market might swing up one day, and down the next, the neural network model should be right when the market stays constant; the up and down swings should offset each other on the other days. Under this assumption, a neurofuzzy model is developed, reflecting the fact that some fluctuations in market indices might be more important with respect to the price of stock x than others. It involves a neural network, where each neuron's running average error from its previous predictions can be used to calculate a reliability index for the neuron. At the last layer, the network makes a fuzzy or disproportionate weighted average of all the neuron's predictions based on a Gaussian activation function of the output neuron's average error. The network traces several market variables (*MVs*) based on a first order gradient of their running averages. The model presented here, is a fuzzy neural network with two hidden layers which analyze consecutive sets of market variables.

In the case of the option trading, we discuss strategies involving options on the S&P 500 index, i.e., options that are derived from the S&P 500, although our results can be extended for strategies based on other options in a straightforward manner. The option trading strategy is based on neural predictors of the volatility of the S&P 500 which anticipate *up*, *down* or *same* type of movements of the index. Based on the superior knowledge provided by the volatility predictions, a profitable trading strategy that involves option is made possible. The model presented here is a fuzzy recurrent neural network with one hidden layer that analyzes the stream of daily prices of the S&P 500. The S&P 500 options are among the most widely traded options and are listed in the Chicago Board of Trade. Options in general come in many different flavors and variations. In this paper we will be concerned only with basic options, *calls* and *puts*, and combinations of them, such as *straddles*. A *call* is a financial contract giving the right, but not the obligation, to the contract holder to *buy* a specified asset (the *underlying* asset) for a specified price (the *strike* price) on, or before, a specified date (the *expiration* date). A *put* is similar, only it gives the right to *sell* the underlying asset for the strike price on, or before, the expiration date. A *straddle*, is one of the many packaged options that can be created when calls and puts are combined linearly; in the case of a straddle, one *call* option and one *put* option, both on the same underlying asset, with the same expiration date, and the same strike price, are packaged together.

The rest of the paper is organized in six sections. Section 2 discusses the neurofuzzy approaches pertinent to the time series prediction problem and their practical ramifications. Section 3 provides an introduction to the prediction problem as it is encountered in financial engineering applications and a brief survey of some neural network models used in the market today. Section 4 discusses a neurofuzzy methodology for stock trading strategies and Section 5 presents an application of a neurofuzzy methodology to trading strategies of options based on predictions of volatility. Section 6 discusses the results of the neurofuzzy approach taken.

2. Neurofuzzy Approaches in Prediction

Neurofuzzy approaches represent an integration of neural networks and fuzzy logic that have capabilities beyond either of these technologies individually (see (Tsoukalas, 1997)). In applications of financial time series prediction, establishing crisp criteria for decision making is rather difficult to achieve; neurofuzzy approaches are shown to be capable of providing improved decision support in situations where crisp estimates are either meaningless or unavailable.

Although about 80% of all neural network applications utilize networks in which there are no feedbacks from the output of one layer to the inputs of the same layer or earlier layers of neurons, there are situations (e.g., when dynamic behavior is involved) where it is advantageous to use feedback. When the output of a neuron is fed back into a neuron in an earlier layer, the output of that neuron is a function of both the inputs from the previous layer at time t and its own output that existed at an earlier time, i.e., at time (t-Δt) where Δt is the time for one cycle of calculation. Whereas feed forward neural networks appear to have no memory since the output at any instant is dependent entirely on the inputs and the weights at that instant, neural networks that contain such feedback, called *recurrent neural networks*, exhibit characteristics similar to short-term memory; because the output of the network depends on both current and prior inputs. Although virtually all neural networks that contain feedback could be considered as recurrent networks, the networks considered here use backpropagation for training.

The complexity introduced by feedback connections, even for elementary systems, is readily apparent during training when it may take hours and even days of computer time for a network to converge. Yet, for some networks the increase in complexity is often compensated for because the feedback drastically reduces the number of cycles needed to train a neural network. Feedback can often be used advantageously to speed up the training of a neural network and to avoid local minima. Indeed, it is sometimes possible to train a neural network after feedback has been added whereas it may not have been previously possible to train it to the desired low level of error. However, capturing dynamic behavior in a model is the most common justification for the use of feedback, and recurrent neural networks are almost always used for dynamic signals such as the

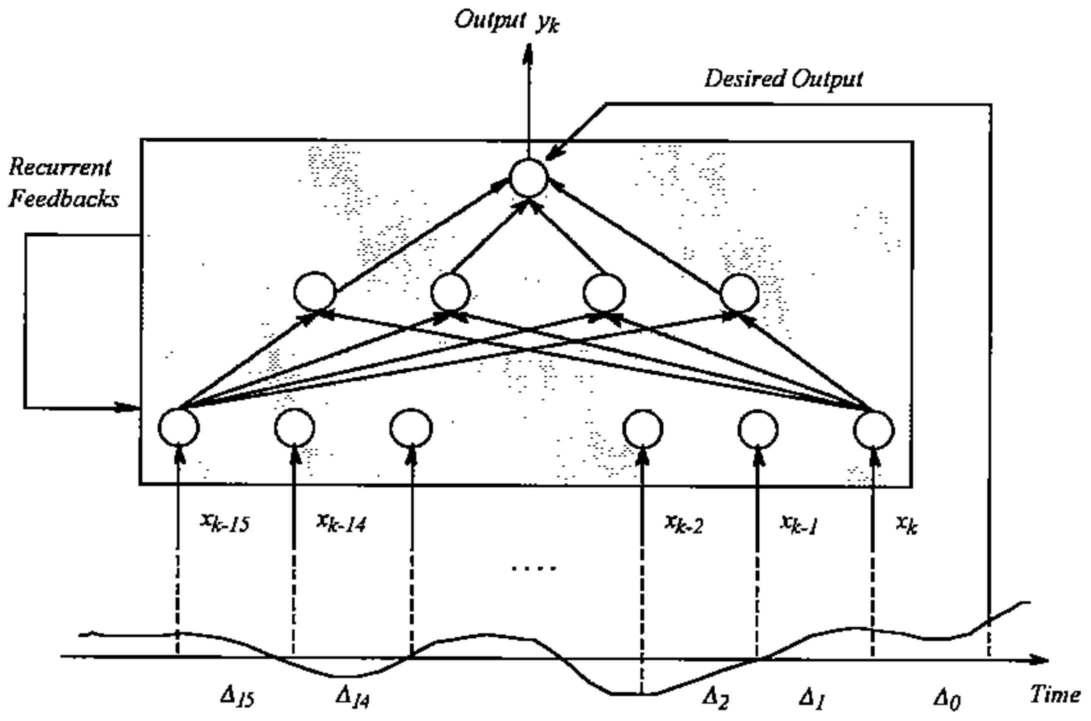


Figure 1. Neural network for time-series prediction.

financial indicators investigated in this work. Neural networks can be used to predict future values in a time series based on current and historical values. Such predictions are, in a sense, a form of inferential measurement. Because of their ability in prediction, there has been an extraordinary amount of interest in the use of neural networks to predict stock market behavior (Refenes, 1996). This popularity continues in spite of the fact that the predictions for neural networks cannot be explained or verified. Perhaps the main reasons for the continuing popularity in the field are that neural networks do not require a system model and that they are relatively insensitive to unusual data patterns.

Although backpropagation neural networks are usually used for time-series prediction, it is possible to use any neural network capable of mapping an input vector into an output vector. Typically, the input of a single time series into a neural network is made as shown in Figure 1. The fluctuating variable is sampled at an appropriate rate to avoid aliasing, and sequential samples are introduced into the input layer in a manner similar to that used in a transverse filter. At every time increment, a new sample value is introduced into the rightmost input neuron, and a sample value in the leftmost input neuron is discarded. The main difference compared to the transverse filter is that the sample preceding those in the input is introduced into the single output neuron as the desired output. In this way, the network is trained to predict the value of the time series one time increment ahead, based on the previously sampled values. The network can be trained to predict more than one time increment ahead, but the accuracy of the prediction decreases when predictions are further into the future. Since such systems are often used in real time, or with data from historic records, the amount of training data is usually very large. Even so, it is important to periodically check the training to assure that over-training does not occur. Furthermore, selection of training data is important because taking every n^{th} sample can introduce bias. This can be overcome by using unequal time intervals between input samples (one method to accomplish this is by sampling at a high rate and then using different number of time intervals between the different input samples).

Although it is possible to predict multiple outputs, it is best to predict only one value because the network minimizes the square error with respect to all neurons in the output layer. Minimizing square error with respect to a single output gives a more precise result. If multiple time predictions are needed, individual networks should be used for each prediction (Tsoukalas, 1997) in the manner shown in Figure 2.

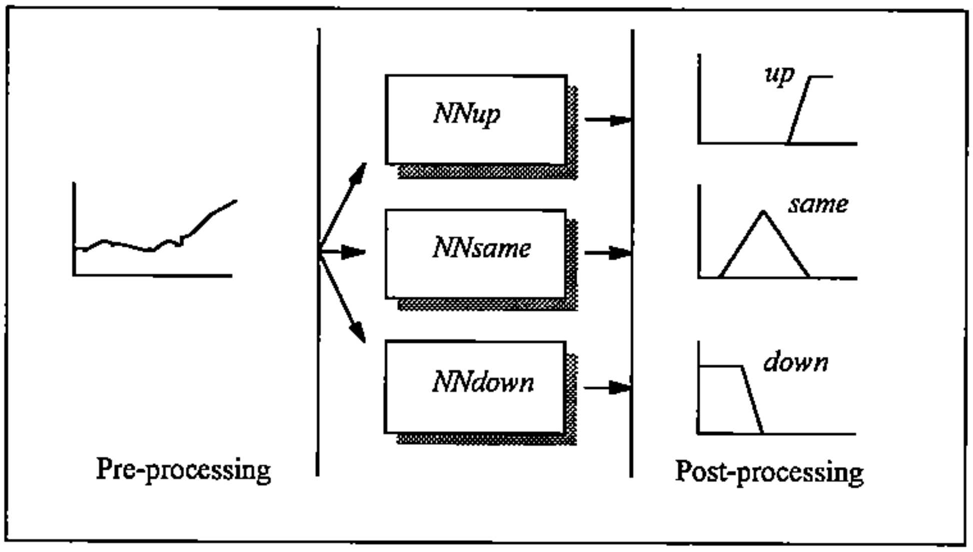


Figure 2. Different neural networks predict different features of a time series.

Generally, large scale deterministic components, such as trends and seasonal variations, should be eliminated from inputs. The reason is that the network will attempt to learn the trend and use it in the prediction. This may be appropriate if the number of input neurons is sufficient for the input data to span a complete cycle (e.g., an annual cycle). If trends are important, they can be removed and then added back in later. This allows the network to concentrate on the important details necessary for an accurate prediction.

The standard method of removing a trend is to use a least squares fit of the data to a straight line, although nonlinear fitting may be appropriate in some cases (e.g., cyclic fluctuations). An alternate method of removing trends and seasonal variations is to pass the data through a high-pass filter with a low cutoff frequency. There are alternative techniques in which a low pass filter is used to leave only the slowly varying trend which then are subtracted from the original signal, with the difference being the value sent to the neural network input layer.

One of the interesting variations of the above technique for prediction is to use differences between successive sample values as inputs to the neural network. This effectively eliminates constant trends and slowly changing trends by converting them to a constant offset. Even seasonal trends are usually removed in this way. Using differences in predicting is generally useful in stock price predictions, especially if the difference is scaled relatively to the total price of the stock, which is effectively the percent price change.

3. Prediction in Financial Engineering Applications

The vast majority of financial and economic activity involves the analysis and prediction of numerous variables. Variables that are encountered in this context are either leading or trailing economic and financial indicators of some behavior or pattern, as well as prices and indices. In the case of leading variables, the primary reason for the need of prediction and forecasting is that it can be used to take preemptive actions. More than any other context, the financial markets offer a bold example of how a reliable prediction can be capitalized upon. Consequently, it is natural for financial engineers to seek for the best possible prediction systems. Financial engineering is the means for implementing financial innovation which is an integral part of the activities of contemporary financial institutions. Finance theoreticians argue for the long standing "efficient market hypothesis (EMH)" as a basis of denouncing any technique that attempts to find useful information for the future behavior of stock prices by using historical data. However, the assumptions underlying the EMH in many cases are not realistic. Studies indicating that some form of the EMH is indeed observed in the financial markets, as well as others, that conclude to the contrary, add to the confusion. Irrespective of the theoretical foundations, technical analysis is used widely as a tool facilitating various financial activities. Among the most challenging financial activities that can make direct use of prediction systems is portfolio management and trading. Portfolio management aims primarily at maximizing profits and minimizing risks on a bundle of assets such as stocks and bonds, by selecting which assets to trade of hold.

Portfolio management systems using neural networks have been developed by a number of people and reported to achieve good results. Wilson, for example, reports on a hybrid system based on several price prediction models including technical, adaptive, and statistical models (Wilson, 1994). The first neural network layer used by Wilson's system, selects which of these models is working best for each stock followed. Then the selected model makes a recommendation on whether to buy, sell or hold a particular stock. The final layer of the system decides which stocks to buy or sell, depending on the maximum risk that the portfolio is allowed to undertake, as it is measured by the *beta* correlation coefficient of the Capital Asset Pricing Model (Black, 1972). Although the system's performance is better than S&P 500, the results show a great volatility in the value of the portfolio over the 250 weeks test period, including a dip below the S&P 500 about halfway through the tested time frame. Nonetheless, the results are quite impressive, and the concept of dynamically managing the portfolio's risk factor very promising.

Lucid Capital Network has reported on some of their preliminary tests, on neural network models and applications to the futures market (Konstenius, 1994). They used a single layer network with up to 1000 nodes, sigmoidal pre- and post-processing, and a beta learning rate of 5% to

10%. The study amplifies the importance and effectiveness of some sound pre-processing techniques; it recommends normalizing the data so as to provide the network strictly with day-to-day changes, rather than the actual prices of equities. In addition it recommends a "holiday management" processing: filling gaps in the database or the holiday intervals with the previous day's data. The concern over holiday gaps is a significant source of noise for most neural network platforms.

A feed forward analog neural network has been developed by Lowe for portfolio management (Lowe, 1994). The marketing analysis is based on the fundamental assumption that the market is not totally efficient, i.e., some stock prices are not "true." Additionally, and to some degree contrarily, this approach incorporates "Sharpe's theory" (Sharpe, 1964) as well, whose formulas are based upon the assumption that the market is always in equilibrium. Lowe's model was tested in both US and European markets. Although the test seems very short (less than a year), Lowe's portfolio has earned an impressive 15% return - even during the volatile bear market where his data was extracted from.

Ahmad and Fatmi (Ahmad, 1994) have designed a "quadric neural network (QNN)" to predict financial time series. The QNN was based on the Gabor-Kolmogorov polynomial model. The fundamental idea is that all statistical and stochastic functions must be studied within the framework of self-defined logical, statistical and physical concepts. They provide formulas that systematically lead to the development of several functions, linear and non-linear, which ultimately predict the financial time series. The model analyzes all the available data derived from a single stream of a stock's day to day prices. Much of these derived data is arrived at by varying the time frame used by the network, and dynamically updating the network's weights through a feedback function of the least mean square (LMS) algorithm. Although this approach of systematic formulas is clearly defined and the network was shown to converge for a hypothetical series, it appears that substantial testing would be required to demonstrate in practice Kolmogorov's theory and its fundamental idea of accurately predicting a series simply by analyzing its single stream of historical pricing.

4. A Neurofuzzy Methodology for Stock Trading Strategies

Stock trading is the activity of buying, selling, or sorting (selling without owning) the stock of a publicly traded corporation. In this section, a feed-forward neural network fuzzy model (*FNFM*) used for stock trading is presented where each neuron uses a number of expressions to calculate its prediction, measure its accumulated error (reliability factor) and update its parameters.

The first step in the *FNFM* is the calculation of the relative change of the market variables, so that different market variables can be compared in terms of relative growth. This transformation is described in Equation 1 (Kung, 1993):

$$d\,MV_{tv} = \log\left(\frac{MV_{tv}}{MV_{(t-A)v}}\right) \tag{1}$$

where, MV_{tv} denotes the market variable number v at time t, and $d\,MV$ the log (relative change of MV_{tv}). The *log* function is used (any base would work) so that the $d\,MV$ values will be evenly distributed as shown Figure 3.

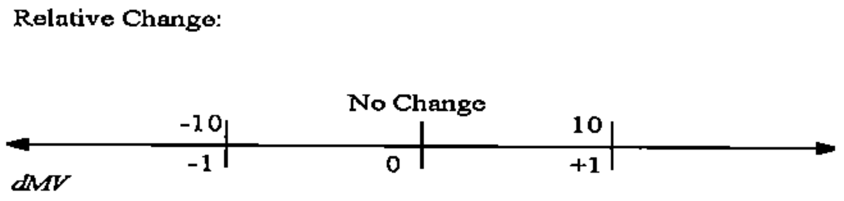


Figure 3. Distribution of the *dMV*

A 10-fold increase ($d\,MV = 1$), for example, followed by 1/10-fold increase ($d\,MV = -1$) would average to a $d\,MV = 0$, or a 1-fold increase (i.e., no change).

Calculation of the Market Variable's Ideal Weight (Sensitivity)

Each *dMV* has a certain sensitivity factor to stock x, the stock of interest. This sensitivity factor, or weight (W_{val}), indicates whether the *MV* moves with or against x, and to what degree. Equation 2 describes this relation.

$$W_{val} \cdot d\,MV_{tv} = d\,MV_{tx} \tag{2}$$

From Equation 2 we see that $W_{val} = \frac{d\,MV_{tx}}{d\,MV_{tv}}$. Thus, W_{val} indicates a particular *MV*'s sensitivity to stock x as shown in Figure 4.

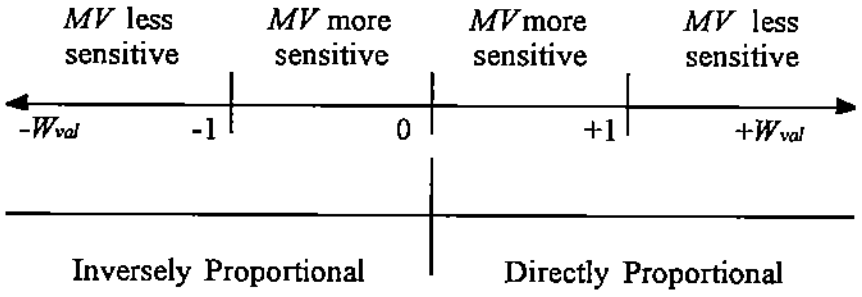


Figure 4. Sensitivity of stock x with respect to a market variable *MV*.

Learning Rate (*BETA*)

Before each neuron calculates its "prediction," its weight is revised by taking a weighted average of its previous weight and the ideal weight (i.e., the teacher variable) which should have been used in the last iteration. Equation 3 indicates the dependence of this weighted average on the learning rate *BETA*, i.e.,

$$W_{val} = BETA \cdot W_{val}{}' + (1 - BETA) \cdot W_{val}^{ideal} \tag{3}$$

where $W_{val}{}'$ is the previous (old) weight and W_{val}^{ideal} is the ideal weight for the last iteration. Since different learning rates vary with the magnitude of a *MV*'s previous performance, several *BETA* values are used independently. Thus, a single market variable can provide short term predictions as well as longer term ones. Each weight corresponds to a particular *MV* and with a particular learning rate *L*. Equation 4 shows the way *BETA* is calculated where *NumL* denotes the total number of different learning rates used for a particular market variable.

$$BETA = \frac{L + \frac{1}{2}}{NumL} \tag{4}$$

From Equation 4 we see that the smaller the learning rate *L*, the lesser the importance given to historical performance.

The Activation Function

At each iteration (each day in this model), the *FNFM* calculates (*NumA* x *NumV*) components from the overall network prediction, where *NumV* is the number of *MVs* used and *NumA* is the time frame. All these components are averaged based on each neuron's reliability index. The reliability index, *RI*, is a Gaussian activation function of the average absolute error, as shown in Figure 5, and it is described by Equation 5.

$$RI = C_1 * e^{(-error/C_2)} \qquad (5)$$

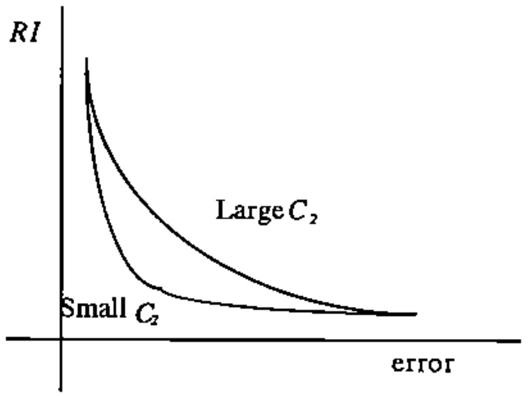


Figure 5. Gaussian activation functions.

Since C_1 is effectively normalized by the weight average, any arbitrary value, e.g. 10, will work. C_2, however, defines the steepness of the curve, and thus defines how much more important a component, e.g. with 1% average error, is than another component with a 10% average error. If a 20-fold will be chosen more important (for this example) a C_2=0.05 is used.

Market variables (the ones used here are market indices) are the primary input data that the *FNFM* analyses to gage the "true" value of stock *x*. Thirteen market variables are selected as network input. These variables are among the most common market indices which are affected and affect the price of the stock which we use in the evaluation of the performance of the *FNFM* (the common stock of IBM). Some of the indices used are shown in Table 1.

FNFM Simulator and Parameter Settings

In running the *FNFM* network simulator, first the historical price database to be used is selected; in the example described below the IBM database is selected. It contains a large table of daily prices for the IBM stock, as well as values of the indices over the past 6 years.

Table 1. Market indices used by the *FNFM*.

DJIA	Dow Jones Industrial Average (30 companies)
DJTA	Dow Jones Transportation Average (20 companies)
DJUA	Dow Jones Utility Average (15 companies)
DJ65	Dow Jones 65 Stock composite
DJFI	Dow Jones Futures Index (Commodities)
S&P 100	Standard & Poor's 100 Index
S&P EI	Standard & Poor's Electronic Instruments
S&P SC	Standard & Poor's Semiconductors
S&P BE	Standard & Poor's Computer Systems
S&P500	Standard & Poor's 500 Index

The parameters *NumV* (number of market variables) and *NumT* (size of historical sample) are set accordingly to the number of variables and number of daily prices available. The default values of *NumT*, *NumA* (time frame size), and *NumL* (number of learning rates) can also be adjusted as necessary. The parameter *NumA* defines the size of the input layer of the neural network, since, depending on the size of the time frame, an appropriate number of inputs must be used. *NumL* defines the length of the second dimension for the first layer. When the neuron updates its parameters ("learns") after each pass, it takes a weighted average of the old parameters and the new ones. The *BETA* value defines how to proportionately average the two values. While it is advantageous for some neurons to learn quickly, it is more advantageous for other nodes to learn in a slower pace. The approach implemented here is to take a combination of these neurons (introducing a third dimension to the neural network) by allowing each neuron to have its own reliability index. The parameter *NumL* defines how many of these combinations to follow for each particular neuron. The overall network has *NumV* * *NumA* * *NumL* neurons with *NumT* passes through them. These neurons all feed into the last layer, a single output neuron, that proportionately averages all of the nodes' predictions based on their proven reliability.

To evaluate the predictive capabilities of the *FNFM* model, two investment strategies are considered using historical data for the IBM stock. The first strategy invests $1 each day. The graph of the daily portfolio balance demonstrates the stability of this investment strategy (low volatility). The second strategy, starts also with $1, and every day invests this amount plus all the available cash balance of the portfolio. This second strategy is a "*dividend and capital gains reinvestment strategy*" which more accurately reflects a realistic trading practice. The overall performance of the network is evaluated based on two objectives: the rate of return, and the volatility of its day to day performance. To evaluate both, a simple and objective approach is to consider the daily cash balance of a portfolio that starts with an initial balance of $1 and subsequently trades the IBM stock

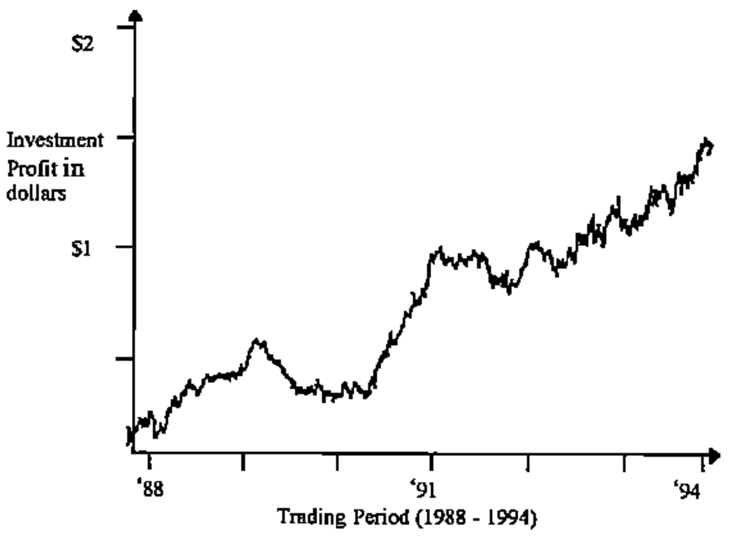


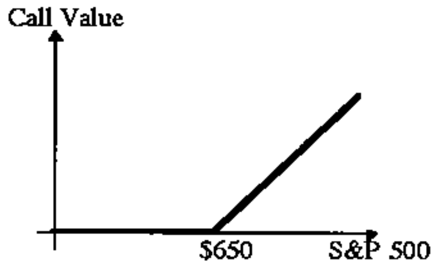
Figure 6: Using *FNFM* with IBM stock, reinvesting dividends, generated an annualized return over the test period of 20.9%.

using the *FNFM*. Figure 6 illustrates the balance of the test portfolio for IBM with the *FNFM* parameters set to *NumA*=1, *NumL* = 3, for a period of 6 years.

5. A Neurofuzzy Methodology for Prediction-Based Option Trading Strategies

As mentioned in Section 1, an *option* is a financial instrument, a form of contract between two parties, that is traded in the world's financial markets, much like common stocks or other entities (e.g. commodities, currencies, etc.). We are concerned here with basic options, *calls* and *puts* and combinations of them, such as *straddles*. The interested reader can find an in-depth discussion of options and financial derivatives in (Hull, 1997)

The *payoff* of an option is the value of the contract at the expiration date. Prior to the expiration date the option is traded in the financial markets as any other asset (e.g. stock) and its price fluctuates based on a number of economic and other parameters. For example an *S&P 500 Dec'98 650 Call* option or an *S&P 500 Dec'98 650 Put* could be bought today; its value at the expiration date (by convention the third Friday of December 1998) will depend on the actual price of the underlying asset, which in this case is the S&P 500 index. Observe that the value of the S&P 500 is not known with certainty, since, it changes as the prices of the stocks contributing to the index (500 stocks of U.S. corporations from various industrial sectors) change. Figure 7 illustrates the payoff of *call* and *put* options for a strike price of $650.



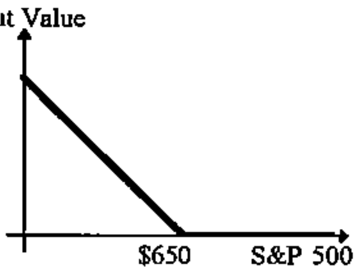


Figure 7. Payoff of a *call* (left) and a *put* option (right) for a strike price of $650.

The payoff value for the *call* is described in Equation 6

$$CV_T = \max(0, SP500 - 650), \tag{6}$$

where *SP500* stands for the S&P 500 price and CV_T is the option value at the expiration date. The payoff value for the *put* is given by Equation 7

$$PV_T = \max(650 - SP500, 0) \tag{7}$$

where PV_T is the option value at the expiration date. A *straddle*, which is one *call* and one *put* packaged together as one option, has a payoff shown in Figure 8.

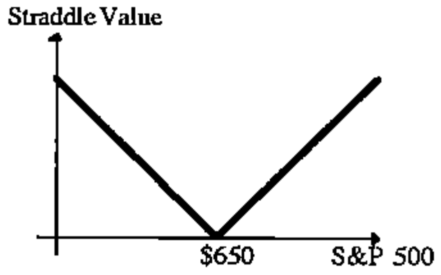


Figure 8. Payoff of a *straddle*.

The payoff of a *straddle* at expiration is given in Equation 8.

$$SV_T = \max(650 - SP500, \quad SP500 - 650) \tag{8}$$

Prior to expiration, the value of an option *V* can be computed as a function of a number of parameters. The task of pricing an option is referred to as *option pricing*. According to the widely used Black & Scholes option pricing model (Black, 1973), the value is given by

$$V = V(S, t, \sigma, q, K, r, T - t) \tag{9}$$

where: S is the price of the underlying asset, e.g. the S&P 500,

t is the current time,

q is the dividend yield,

σ is the volatility of the underlying asset's returns,

K is the strike price,

r is the risk-free rate of return, and

T is the expiration of the option measured in years

By convention, options are written with duration in multiples of 3 months (0.25 years). K, q, T, and r are usually inputs to the option trading strategies; K and T are constants known in advance, r is taken to be the T-Bill[1] rate with matching duration as the duration of the option, and q is approximated by a constant dividend yield. S is described as a stochastic process and its value is determined by the laws of supply and demand. The time to expiration is calculated as $(T - t)/252$ years[2].

The volatility σ, a measure of the variability of the time series, is taken to be constant in most practical cases; it is found either as the volatility in the more recent past of the time series, or, as the volatility implied from other option prices quoted in the financial markets. It is known that a model for future volatility that assumes the volatility is constant is not completely in agreement with the Black & Scholes modeling ("smile-effect", (Hull, 1997)). A successful trading strategy, which uses a neurofuzzy volatility predictor, can be built based on this deficiency, i.e., of using historic or implied volatility to price options. If options are priced today using either of the two estimates mentioned above (historic or implied volatility), and their price fluctuates according to the unknown future volatility behavior, a reliable prediction of future volatility can be of tremendous value; it can be used to identify over-priced or under-priced options and execute appropriate trades. The option value is an increasing function of volatility; the higher the volatility, the higher the option value and vice-versa. Consequently, a system that would provide an estimate for the

[1] *T-Bills are treasury bills, short term US government debt that is considered to be free of risk. An alternative would be to use Money Market account yields or yields on Certificates of Deposit (CODs). However, it is common practice to approximate the short term risk-free rate of return by the rate offered from T-Bills.*

[2] *In order to convert in years, we use the number of trading days in a year and not the actual days. A widely used approximation for the trading days is 252 per year (Hull, 1997).*

future volatility trend could be used to build a trading strategy, based on options, that would provide above normal returns.

Trading Strategies using Options based on Volatility Prediction

If we know that future volatility is going to be increasing (*up*), with a reliability of some degree, then we could buy options. Since the options are selling at current volatility estimates, their price will increase according to the predicted upward movement of volatility. If the prediction is realized, we stand to gain, *ceteris paribus*, the difference $V_{\sigma_R} - V_{\sigma_o}$ where σ_R denotes the predicted volatility and σ_0 the volatility currently used by the market. This will be realized if nothing else changes in this period.

Recall that the option value is a function of a number of other parameters which change over time, and in particular, the time to expiration and the underlying asset price. Intuitively, a prediction of higher volatility implies that the asset price is going to move; it does not however specify in which direction. Consequently, if, as a reaction to the prediction for higher volatility we buy call options and the asset price moves down, we might lose. On the other hand if we buy put options and the asset price goes up we might also lose. A "safer" way to react on a prediction for higher volatility would be to buy straddles, since, whichever way the price of the underlying is going to move we are going to profit assuming that the prediction of higher volatility is realized.

In the Black & Scholes option pricing framework, the risk that incurs from an option position (i.e. the risk of losing the initial investment) can be eliminated by appropriate transactions. The elimination of this risk is called *hedging* and involves getting a position in the underlying asset and the risk-free asset (T-Bills). The size of the position, i.e. the number of shares, is determined as the first order partial derivative of the value function V with respect to the underlying asset. This type of hedging strategy is referred to as "delta" hedging and prescribes that a portfolio consisting of the option and Δ shares of the underlying would not have any risk from movements in the price of the underlying asset. Δ is determined by the pricing model and it is also an increasing function of volatility. Hedging can be viewed as an insurance and as such has some cost. This cost reduces the gains realized (and the risk). In an option trading strategy that also involves selling options apart from buying them, a reliable volatility prediction could be used to reduce the necessary hedging, and thus reduce costs. Consequently, a trading strategy based on future volatility prediction can create profits both from the change in value, but also from an appropriate reduction in the size of hedging (i.e., smaller share number than prescribed by Δ).

Crisp interpretation of the prediction risk for volatility trends is very difficult. The fuzzification of these two variables, i.e., the risk (reliability) of the prediction and the prediction itself, provides a convenient to use and conceptually promising approach to fully exploiting neural network predictive capabilities in the decision making process. Figure 9 illustrates a two-dimensional fuzzy relation involving the two prediction variables: *risk* and *volatility*. The volatility trend is described as *up*, *down* or *same* and the risk level of the decision as *low*, *average* and *high*. The membership functions of the resulting fuzzy numbers are described by correspondingly subscripted μ's. The infinite crisp decision space is transformed into a coarser, more parsimonious fuzzy space, where fuzzy logic can be used to make inferences and come up with accurate recommendations to buy, sell or "do-nothing." The pyramidal shape of the fuzzy rule at the center of Figure 9, for example, is interpreted as a "do-nothing" recommendation. Furthermore, the fuzzification of the decision space reduces the complexity of the neural network.

The neurofuzzy volatility prediction system is trained to anticipate the changes in the volatility of the returns of the S&P 500 system and provides recommendations that are used to implement short term option trading strategies. The input to the prediction system, i.e., the financial time series, is the daily close value of the S&P500 index. The time series used is shown in Figure 10.

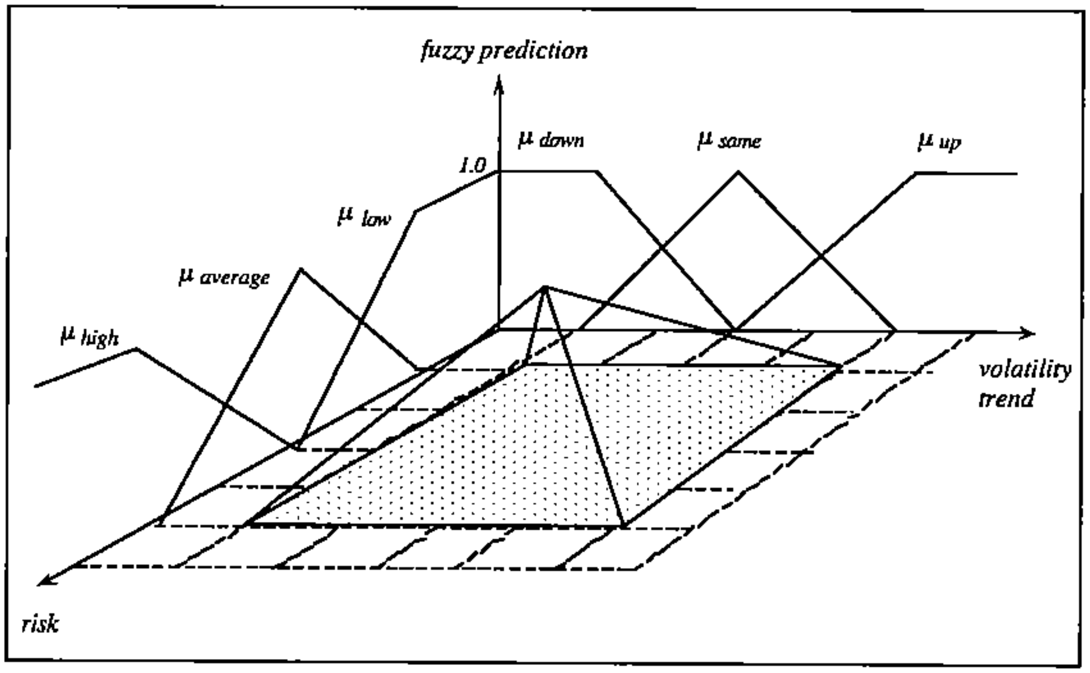


Figure 9: Fuzzy decision universe.

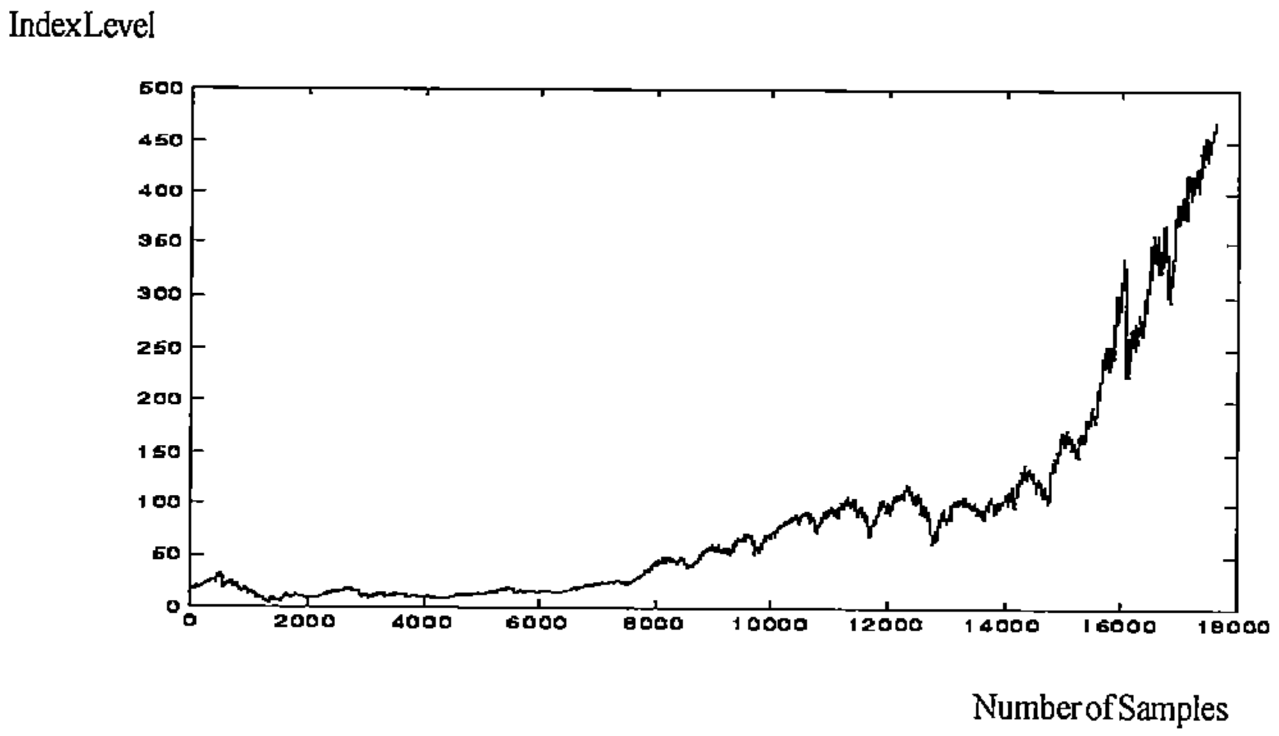


Figure 10. The Standards and Poor's, S&P 500 time series from 1928 to 1993.

Preprocessing of the Time Series Input

The S&P 500 time series that is used (17,604 daily samples) is pre-processed before used by the volatility prediction system. The pre-processing is done as follows. Let us denote by $s_t, t = 1, \ldots, 17604$ the S&P 500 daily closing prices. The logarithm of these data points is $s_t' = \log s_t$. Hence, the difference of successive data points is

$$s_t'' = s_t' - s_{t-1}' = \log\left(\frac{s_{t-1}}{s_{t-1}}\right) \tag{10}$$

where it is assumed that $s_0'' = 0$. Denoting by *NumA* the time frame, the time series is further transformed as

$$s_t''' = \left[std\left(s_t'', \cdots, s_{t-NumA}''\right)\right]^2 \tag{11}$$

where *std* denotes the standard deviation, and $t = NumA, \cdots, 17604$. The time series is scaled by

$$x_t = 10^4 \, s_t''' . \tag{12}$$

Scaling facilitates neural network processing and training. The result of the pre-processing is a modified time series, shown in Figure 11.

The time series is partitioned into three sets: the training set, the test set, and a third set which is left completely untouched. Both the test and the unseen sets are out-of-sample sets. The training set and the test set are drawn from the last 6000 elements of the time series, that is, from 2/5/1970 to 10/29/1993. The training set is comprised by the odd numbered elements and the test set by the even numbered elements.

The training set is further partitioned into three subgroups: the *up*, the *same*, and the *down*, of *NumA*-sized vectors. The *up* set contains the input vectors which precede an increase of the index volatility in excess of 16.5%. The *same* set contains input vectors which precede absolute change less than, or equal, to 16.5% and the *down* set contains input vectors which precede a decrease in volatility of more than 16.5%. The three different sets are used to train three corresponding neural networks, with one hidden recurrent sigmoidal layer and one linear output layer. The training of the three networks was done for 15000 epochs on the training data and an error threshold (sum square error) of 10^{-3}.

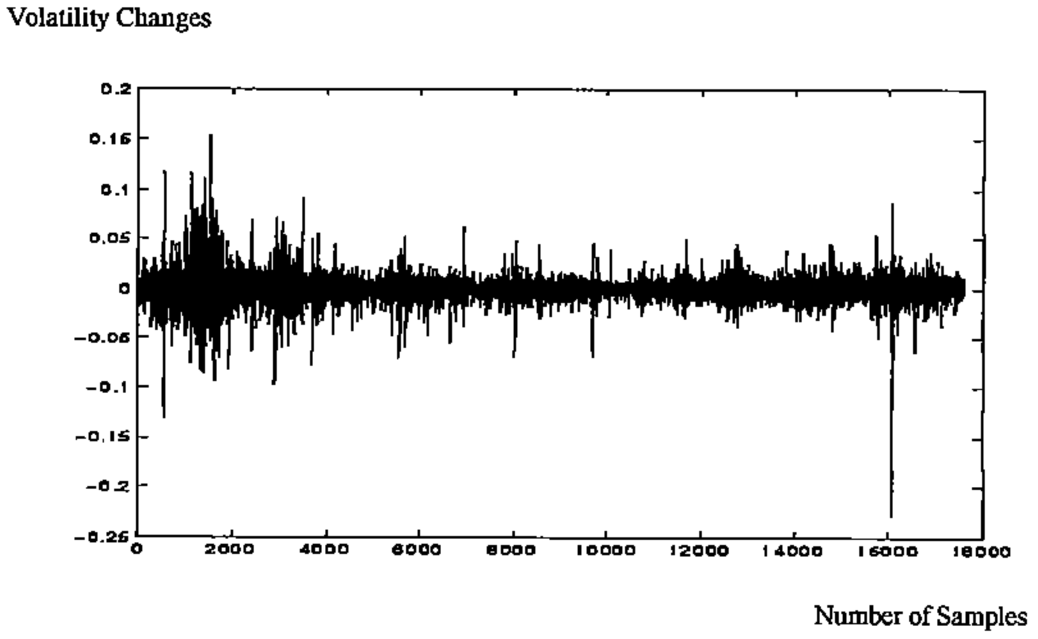


Figure 11. Modified S&P 500, time series from 1928 to 1993 (after pre-processing).

Crisp Post-Processing of the Neural Network's Output

We train three individual neural networks, called *NNup*, *NNdown* and *NNsame*, on appropriately selected training subsets, *up*, *down*, and *same*, in order to recognize patterns that precede the following movements in volatility changes: a) the *NNup* predicts a higher than 16.5% increase over the previous day, b) the *NNsame* predicts change of less than 16.5% in either direction, and c) the *NNdown* predicts a reduction of 16.5% or more. We use the relative RMS error,

$$RMS = \sqrt{\sum_k \left(\frac{x_k - \overline{x}_k}{x_k} \right)^2} \qquad (13)$$

where, k is the number of samples, x_k the true output, and $\bar{x}_k$ the predicted value, to study the performance of the system on the three sets, the train set, the test set, and the completely unseen set. We tested five different rules for combining the output of the neural networks. The output of *NNup* is denoted with u, of *NNsame* with s, and of *NNdown* with d. The rules are:

- **Rule 1**: Use the arithmetic mean of the three outputs, i.e. $(u+s+d)/3$.
- **Rule 2**: Use only u.
- **Rule 3**: Use only s.
- **Rule 4**: Use only d.
- **Rule 5**: Exclude those outputs that predict something out of the range the were trained, e.g., if $u < 16.5\%$ discard it; of those remaining, take the arithmetic average.

Table 2 summarizes the relative RMS on the three tested inputs.

Table 2. Relative root mean square error obtained by the three networks during testing.

	Training Set	Test Set	Out-of-Sample Set
Rule 1	0.3920	0.3775	0.4608
Rule 2	0.3388	0.3287	0.3350
Rule 3	1.0586	1.0397	1.4032
Rule 4	0.2788	0.2667	0.2555
Rule 5	0.2918	0.2758	0.2565

The approximation of each three neural network to the financial time series is shown Figure 12.

Fuzzy Post-Processing of the Neural Network's Output

We train three neural networks, in a similar manner as discussed above, only this time the output of the neural networks is not a crisp number but a fuzzy number. The three neural networks, *fNNup*, *fNNdown*, and *fNNsame*, provide fuzzy predictions for the change in volatility *fup*, *fdown*, *fsame*. The membership functions of the three fuzzy outputs are triangular.

Each network gives three outputs, the left (l), center (c), and right (r) which represent the membership function of the prediction. For the output of a particular network to be meaningful the condition $l < c < r$ must be satisfied. If it is not, the output is discarded. Recall that each network is trained to recognize a particular class of input patterns. We tested the following post-processing rules to evaluate the performance of the fuzzy neural network over the three sets, training, testing and unseen.

- **Rule 1**: *If* all three outputs do not satisfy the constraint of shape predict nothing. *Else*
- **Rule 2**: *If* all three satisfy the constraint then discard the one with the most distorted membership function and take the fuzzy mean of the other two. *Else*

- **Rule 3**: *If* less than three satisfy the constraint then take the fuzzy average of the those that satisfy the constraint.

In all cases that there is prediction, the prediction is the centroid of the resulting fuzzy number. In choosing the most distorted membership function to discard, in Rule 2, we use the criterion:

- **Criterion** : Compare the total support of the fuzzy number. Remove the one with biggest support. The support is computed as $(r - l)$.

Table 3. Relative root mean square error obtained by the three networks during testing

	Training Set	**Test Set**	**Out-of-Sample Set**
Criterion	0.3024	0.2864	0.2781

Evaluating the prediction capabilities using option trading strategies

In order to evaluate the predictive capabilities of the neural network we use its recommendation in an option trading strategy involving *straddle "at-the-money"* options on the S&P 500.

For brevity, and in order to avoid introducing intrinsic financial knowledge, we considering only a simple trading strategy, based on *straddles* made up of *call* and *put* options on the SP500 index. For pricing the options we are using the Black & Scholes pricing model. The formulas for the prices of the *call* and *put* options used are:

$$P(S,t) = Ke^{-r(T-t)}N(-d_2) - Se^{-q(T-t)}N(-d_1) \qquad (14)$$
$$C(S,t) = Se^{-q(T-t)}N(d_1) - Ke^{-r(T-t)}N(d_2)$$

where, $$d_1 = \frac{\log(\frac{S}{K}) + (r - q + \frac{\sigma^2}{2})(T-t)}{\sigma\sqrt{T-t}}, \quad d_2 = d_1 - \sigma\sqrt{T-t} \qquad (15)$$

In the above formulas q denotes the constant dividend yield.

For finding the current market price of the options, i.e. the price which they can be purchased, we make the assumptions listed in Table 4. The trading strategy simulation is summarized as follows:

1. Start with an initial amount (cash balance) of $1,000.
2. At the beginning of each day, mark-to-market (i.e. compute the market value) of the portfolio of options.
3. Liquidate the portfolio, if it contains holdings other than cash.
4. If the neurofuzzy volatility prediction system recommends "hold" do nothing.
5. If the volatility prediction system recommends "buy", purchase as many "at-the-money" *straddles* as the cash balance permits.
6. Go to step 2.

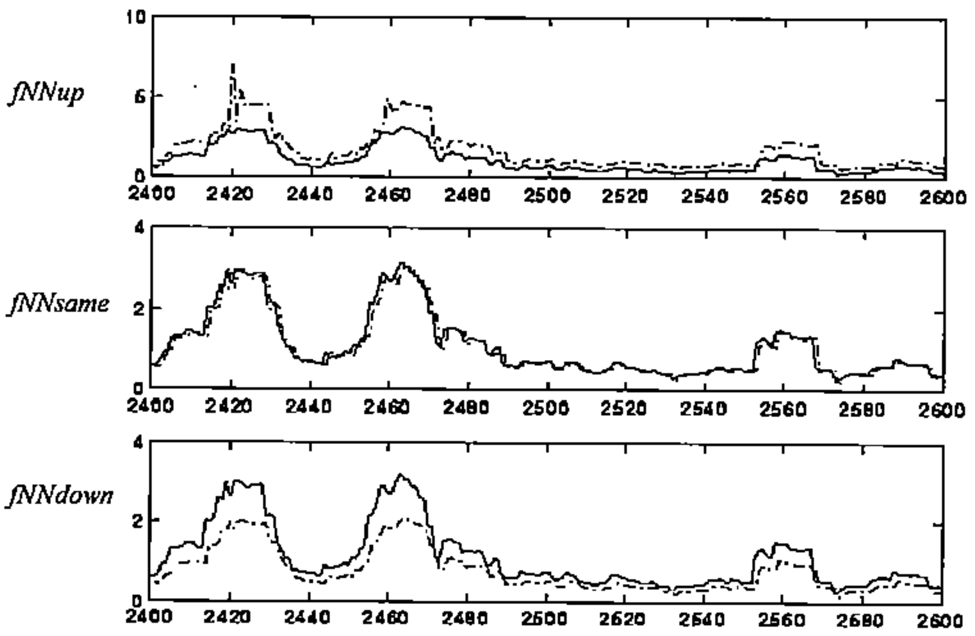


Figure 12. Comparing the modified S&P 500 time series with the neural network predictions (dashed lines denote predictions, solid lines denote actual time series data): (a) The network trained with patterns of upward movement of the index generally over-predicts the time series. (b) The network trained with patterns of "about the same" movements of the index generally stays close to the index. (c) The network trained with patterns of downwards movement generally under-predicts the actual S&P 500 time series.

We simulate this trading strategy on the out-of-sample set for approximately 10,000 days. The comparison is done between a portfolio fully invested in the S&P 500 index and the option portfolio managed using the neurofuzzy volatility prediction system. We keep track of both portfolios, and their market value at the beginning of each trading day. Returns from investing into the S&P

500 are shown in Figure 13. Returns from the option trading strategy using the volatility prediction system for the same period and with the same initial investment are shown in Figure 14.

In the case of the S&P 500 portfolio we get at the end of the period a portfolio balance of \$3,862. In the case of the option portfolio we get at the end of the period a portfolio balance of \$625,120. The resulting performance is substantially better for the option trading strategy which uses the neurofuzzy volatility prediction system. The higher volatility exhibited by the daily balances of the option portfolio is in part expected and explained by the leverage of options compared to stocks. Each option gives the right to buy or sell, depending on the type of option, 100 shares of a certain asset. Generally, a \$1 change in the price of the asset, results in a 100-fold change in the price of the option. Also, the choice to fully invest the available cash balance at all times contributes to the balance volatility. The 75% reliability of the neurofuzzy predictor leaves a 25% margin for false predictions. Note that correct prediction of direction for volatility movement says nothing about the magnitude of the movement. That means the gains from correct predictions and losses from incorrect predictions vary. A trade made based on a faulty prediction results in a loss, which in order to be covered requires twice the increase. For example, a 50% loss must be followed by a 100% profit in order to reach the same balance as the one before the faulty prediction. In general, the above trading strategy demonstrates the potential of the methodology used to predict the volatility movements. In a realistic framework it would be combined with more sophisticated management and hedging activities in order to reduce the variability of the returns. The success of the option trading strategy is primarily attributed to the prediction of the future volatility. In all cases the value of the option portfolio is more than that of the S&P 500 portfolio.

Table 4. Assumptions for the parameters used to price options.

S	The price of the index is taken to be the closing price for the day; i.e. we execute all trades simultaneously at the end of the day.
K	The strike price is taken to be the "at-the-money" value, i.e. same as the price of underlying S&P 500 index.
σ	The volatility that the market uses to price the options is the historic volatility for the last 30 days.
r	The risk-free interest rate is taken to be the T-Bill rate, approximately 5%.
q	The constant dividend yield of the index is taken to be approximately 2.5%.
(T-t)	The duration of the options used is 1 month. This in practice translates into using the option with the closest expiration date.

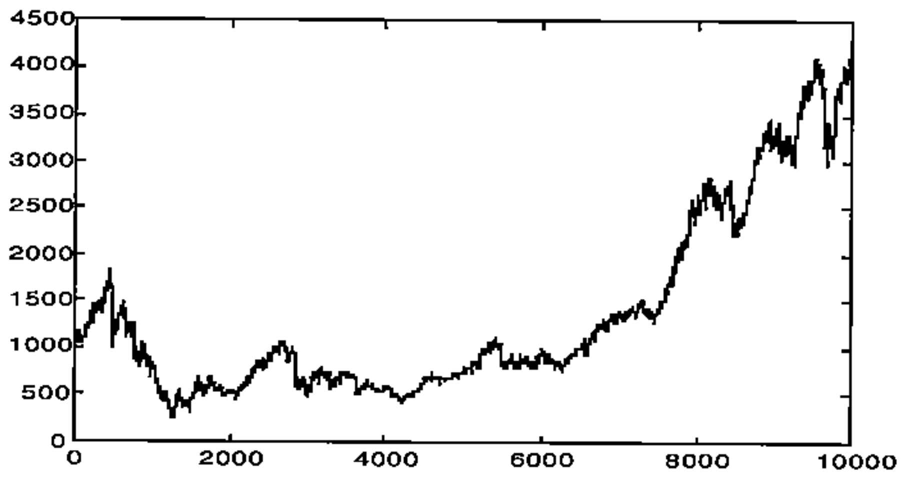


Figure 13. The returns from investing into the S&P 500 between 1928 and 1968 (the vertical axis indicates the portfolio balance in dollars).

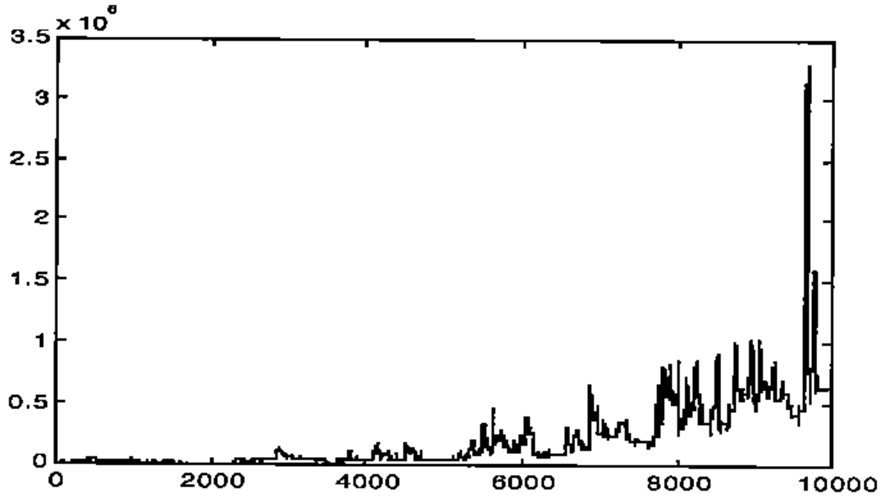


Figure 14. Returns from investing according to neurofuzzy predictions of the market for the period between 1928 to 1968 (vertical axis indicates portfolio balance in millions of dollars).

6. Conclusions

Neural networks, when properly configured, offer superb pattern matching capabilities that can be used for predicting the ups and downs of financial indicators such as the S&P 500 index. The fuzzification of such neural predictions may lead to robust and overall successful trading strategies.

We have presented in this paper two cases in which neurofuzzy prediction approaches are used successfully to predict financial variables in a way that is meaningful from an investment point of view, i.e., resulting in profitable trading of stocks and options. However, another interpretation of our results is possible. They could be a challenge to standard option pricing models. The high returns our simulations yielded would surely have led in reality to higher option prices - higher than the standard models predict.

In the present paper our simulations involved standard option pricing models and neurofuzzy approaches to strategic trading. Future work involves replacing the standard pricing models with neurofuzzy approaches to predicting prices, in order to gain insight into the impact of neurofuzzy approaches as they come to be used in actual markets.

7. References

Ahmad J., Fatmi, H., A Quadric Neural network System for Prediction of Time Series Data, *Proc. IEEE World Congress in Neural Networks*, pp. 3667-70, Orlando, FL, 1994.

Anderson, P.W., Arrow, K.J., Pines, D., Eds., *The Economy as an Evolving Complex System*, Addison-Wesley, Redwood City, California, 1988.

Black, F., Jensen, M. and Scholes M, The Capital Asset Pricing Model: Some Empirical Tests, in *Studies in Theory of Capital Markets*, Jensen M., Ed., Preager, New York, 1972.

Black, F. and M. Scholes, The Pricing of Options and Corporate Liabilities, *Journal of Political Economy*, 81, pp. 637-59, May-June 1973.

Farber, J. D. and Sidorowich, J.J., Can New Approaches to Nonlinear Modeling Improve Economic Forecasts? in *The Economy as an Evolving Complex System*, Anderson, P.W., Arrow, K.J., Pines, D., Eds., pp. 99-115, Addison-Wesley, Redwood City, California, 1988.

Hobbs, A., Bourbakis, N.G., A Neurofuzzy Arbitrage Simulator for Stock Investing, *Proceedings of the IEEE/IAFE 1995 Computational Intelligence for Financial Engineering*, New York, April 9-11, 1995.

Hull, John C., *Options, Futures, and Other Derivatives*, Third Edition, Prentice Hall, 1997.

Konstenius, J., Mirrored Markets, *Proc. IEEE World Congress in Neural Networks*, pp 3671-75, Orlando, FL, 1994.

Kung, S.Y., *Digital Neural Networks*, Prentice-Hall International, London, 1993.

Lapedes A., and Farber R., *Nonlinear Signal Processing Using Neural Networks: Prediction and System Modeling*, Los Alamos National Laboratory Technical Report, LA-UR--87-2662, June 1987.

LeBaron, B., Weigend, A.S., Evaluating Neural Network Predictors by Bootstrapping, *Proc. of the Intern. Conf. On Neural Information Processing* (ICONIP '94), pp 1207-1212, Seoul, Korea, 1994.

Lippit, V.D., The Golden 90's, *Real World Macro: A Macroeconomics Reader from Dollars & Sense*, 7th Edition, pp 73-75, Dollars & Sense, Sommerville, MA, 1990.

Lowe, D., Novel Exploitation of Neural Network Methods in Financial Markets, *Proc. IEEE World Congress in Neural Networks*, pp 3623-27, Orlando, FL, 1994.

Peters, E., *Fractal Market Analysis*, John Wiley & Sons, New York, NY, 1994

Radzicki, M.J., Institutional Dynamics, Deterministic Chaos, and Self-Organizing Systems, *Journal of Economic Issues*, Vol. XXIV, No. 1, pp. 57-102, March 1990.

Ramsey, J.B., Syers, C. L., and Rothman, P., The Statistical Properties of Dimension Calculations Using Small data Sets: Some Economic Applications, *International Economic Review*, Vol. 31. No. 4, pp. 991-1020, 1990.

Refenes, P., Abu-Mustafa, Y., Moody, J.E., Weigend, A.S., Eds., *Neural Networks in Financial Engineering*, World Scientific, Singapore, 1996.

Sharpe, W., Capital Asset Prices: A Theory of Market Equilibrium under Conditions of Risk, *Journal of Finance*, Vol. 19, pp. 452-442., 1964.

Trippi, R., Lee,K., *Artificial Intelligence in Finance & Investing*, Irwin Publishing, Chicago, 1996.

Tsoukalas, L.H., Uhrig, R.E., *Fuzzy and Neural Approaches in Engineering*, John Wiley & Sons, New York, NY, 1997.

Wilson C.L., Self-Organizing Neural Networks for Trading Common Stocks, *Proc. IEEE World Congress in Neural Networks*, pp. 3657-61, Orlando, FL, 1994.