2 Predicting Stock Prices

Mathematicians and economists have studied stock price predictions for many years. In this chapter, the theory of efficient markets presented will show that though no one can consistently predict an exact future stock price, it is possible, on average, to exploit inefficiencies in the commodity markets and achieve a favorable return. With this theoretical framework in hand, I describe various practical approaches and conclude on what I perceive to be a promising direction.

2.1 Efficient Market Hypothesis

The ability of capital markets to reflect and react to the data relating to a tradable security is known as the “Efficient Market Hypothesis” (EMH). Paul Samuelson first coined this term in seminal work [Samuelson 1965] and the fact that he was awarded the Nobel Prize in economics shows the importance of the EMH concept to generations of investors. Simply, the EMH states that the price of a stock is the consensus of all investors and other players in the market. If a disproportionate group believes that a security is undervalued, the buyers will outnumber the sellers, driving the price up until it has reached equilibrium. Similarly, an overvalued commodity will attract fewer buyers than sellers, so that its price will drop.

In a perfect market, one that is completely efficient, the price of a commodity reflects all information that pertains to it in any way. This includes published reports and press releases, articles in newspapers, magazines or electronic media as well as macro-economic trends, the political climate and strategic as well as tactical plans of the companies. New information and decisions would immediately lead to an adjustment of the price of the commodity.

The efficient market hypothesis is typically formulated in a weak, semi-strong and a strong form. The weak form of market efficiency assumes that security prices follow patterns with specific cycles of upward and downward trends. Analysts subscribing to the weak form of the efficient market hypothesis generally search for specific patterns in charts or the product and management structure of a company to identify under- or overvalued stocks. This includes all investment advisors who make a living researching particular companies, markets and industries. Prominent representatives of this guild are Goldman Sachs, Merrill Lynch, Salomon Smith Barney and Lehman Brothers.

Followers of the semi-strong form of the EMH assume that the prices of all securities reflect all publicly available data. This includes fundamental business data, press releases as well as rumors, which possibly spread inaccurate information about the underlying commodity. By implication, the only possible means of consistently benefiting from the stock market would be to act on non-public or internal information about a

company. This is usually information held by the directors of the relevant companies and includes plans and strategies. Much of this information can affect the stock prices if leaked to the general investment public. However, doing business on non-public information is called “insider trading” and is punishable by law.

Persons subscribing to the semi-strong form of the efficient market hypothesis build portfolios with a long-term gain in mind, based on the assumption that the stock market has traditionally outperformed risk-free investments over periods of ten years or longer. The most renowned representative of this school of thought is Warren Buffet and his Berkshire Hathaway Mutual Fund.

In contrast, the strong form of the efficient market hypothesis suggests that stock prices reflect all data relevant to the security, both publicly available and non-public information. This form is generally rejected by the investment community and expressed eloquently by Farmer and Lo. They argue that taken to its logical conclusion, no biotechnology company would attempt to develop a vaccine for the AIDS virus, because “if the market for biotechnology is efficient in the classical EMH sense, such a vaccine can never be developed – if it could, someone would have already done it! This is clearly an absurd conclusion because it ignores the challenges and gestation lags of research and development in biotechnology” [Farmer, Lo 1998].

Much work has been done by a variety of persons on the EMH to verify if price movements are indeed stochastic and unpredictable.

As early as 1963, C.W.J. Granger analyzed stock behavior based on linear models and spectral analysis and found evidence of inefficiencies [Granger 1963]. These findings were supported by the research from Niederhoffer and Osborn, by showing that professional portfolio management statistically achieved greater returns than amateur or random selections [Niederhoffer, Osborn 1966].

As Ambachtsheer showed, only the introduction of massive databases and complex algorithms permit reasonably consistent investment success [Ambachtsheer 1994]. Studies like the one by Per H. Ivarsson on inter-bank foreign exchange trading show that there continues to be extensive interest in the subject [Ivarsson 1997]. The conclusion of many authors is that inefficiencies exist and can be exploited given a coherent investment strategy.

Not surprisingly, exchanges with better infrastructure, players that are more sophisticated and better regulatory frameworks are more efficient than others. In general, the opportunities are more pronounced in smaller, less developed markets like the Bombay or Helsinki stock exchange, as opposed to the New York Stock Exchange, as Samuelson convincingly argued [Samuelson 1965].

Though it is unlikely that the debate regarding the efficiencies of markets will ever have a formal conclusion, the overwhelming evidence shows that even if stock markets are not gold mines, they do offer opportunities, given a coherent strategy. This view was succinctly expressed by Boldt and Arbit: "...trading carefully and searching for opportunities caused by a bias in conventional thinking seem to be the keys to success for professional investors in a highly competitive, but not strictly efficient, market" [Boldt, Arbit 1984].

2.2 Mathematical Modeling Techniques

Traditionally stocks were researched using fundamental analysis, a method by which the financial health of the company is evaluated and compared to those of competitors in the same industry and the market as a whole.

Warren Buffet is probably the most famous investor who used this approach successfully over decades. He has amassed a fortune both for himself and his investors of the Berkshire Hathaway Mutual Fund. At the annual meetings in Omaha, Nebraska, he makes investing sound simple with statements like his conviction to judge a company only by "its inner values" and that they "buy if we like what we see" [Heller 2000]. However, the sheer number of potential investment opportunities does not permit an in depth analysis of management personalities, business models as well as financial health. Consequently, computers were introduced

very early to analyze quantitative data from a variety of companies to help identify potential winners.

Frequently this analysis focuses on the comparison of many different ratios which investors weight relative to their importance. The most common and widely quoted ratio continues to be price-to-earnings, though all other values from the balance sheet, profit and loss statement and cash flow can be taken into account. Rüegg-Stürm and the training materials from the Financial Training Company introduce all common ratios used in the financial evaluation of companies. Both provide an intuitive tutorial on the topic [Rüegg-Stürm 1997] [The Financial Training Company 1998]

It is worth emphasizing that the ratios do not represent absolute quantities, but should only be used for comparison of companies in the same industry, region or market. Also, all ratios can only serve as one indication and should not be viewed in isolation.

$$\text{P/E multiple} = \frac{\text{Price}}{\text{Earning}} \qquad \text{Equation 2.2.1}$$

The P/E multiple or price/earnings ratio compares the closing price of the stock with the earnings of the last 12 months. A high value is often a reflection of lofty expectations of stock price and may indicate that the stock is overpriced.

$$\text{Gross Profit Margin} = \frac{\text{Gross Profit}}{\text{Sales}} \qquad \text{Equation 2.2.2}$$

The Gross Profit Margin determines a company's trading activity. It indicates a company's profit margin by showing the relationship between sales and direct production costs.

$$\text{Operating margin} = \frac{\text{Net Trading Profit}}{\text{Sales}} \qquad \text{Equation 2.2.3}$$

The Operating Margin indicates the profitability of sales, taking into account the volume of activity. Net trading profits should be before tax, interest paid and income from investments.

$$\text{Overhead Rate} = \frac{\text{Overheads}}{\text{Sales}} \qquad \text{Equation 2.2.4}$$

The Overhead Rate forms the bridge between gross profit and trading profit to sales in the previous two ratios. If there is a significant upward movement in this ratio, it may be a cause for concern.

$$\text{Return on Capital} = \frac{\text{Net Profit}}{\text{Capital Employed}} \qquad \text{Equation 2.2.5}$$

This Return on Capital Employed ratio measures the overall efficiency of the business but is only meaningful if compared within the same industry. Manufacturing, for example, tends to be more capital intensive than service industries and will

exhibit a lower return on capital ratio. "Capital Employed" should include share capital, reserves and long-term loans.

$$\text{Current Ratio} = \frac{\text{Current Assets}}{\text{Current Liabilities}} \qquad \text{Equation 2.2.6}$$

Based on the balance sheet figures, the Current Ratio comments on the working capital position of the company and is generally accepted as a measure of short-term solvency. This ratio is particularly pertinent for "dot.com" companies.

$$\text{Quick Ratio} = \frac{\text{Current Assets less Stock}}{\text{Current Liabilities}} \qquad \text{Equation 2.2.7}$$

The Quick Ratio or "Acid Test" indicates a company's ability to pay it debts quickly. Stock and "work-in-progress" are generally excluded, since they are not readily convertible to cash.

$$\text{Stock Turnover} = \frac{\text{Stock x 365 days}}{\text{Cost of Materials}} \qquad \text{Equation 2.2.8}$$

The Stock Turnover ratio indicates the stock turnover period in days. If this value increases, it could indicate excessive or obsolete stock, a negative indicator, particularly for "high-tech" companies, where the life cycle of a product is relatively short.

$$\text{Debtors Turnover} = \frac{\text{Debtors x 365 days}}{\text{Credit Sales}} \qquad \text{Equation 2.2.9}$$

The Debtors Turnover indicates the average period of credit taken by customers and is useful in determining the possible existence of bad debts.

$$\text{Gearing} = \frac{\text{Interest Bearing Debt}}{\text{Shareholder Funds}} \qquad \text{Equation 2.2.10}$$

This Gearing ratio helps measure the long-term strength of a company. High gearing indicates a high risk and susceptibility to economic fluctuations. It may also indicate that the company would have difficulties borrowing additional funds.

$$\text{Dividend Cover} = \frac{\text{Profit}}{\text{Dividend}} \qquad \text{Equation 2.2.11}$$

The Divided Cover calculates the number of times the company could have paid the dividend amount out of profit. A high dividend cover may indicate that the company is financially sound, having retained considerable amounts of profit for investment back into the company or that the dividend was very low. The latter may be sign of expansion.

$$\text{Relative Market Share} = \frac{\text{Market Share x 100}}{\text{Total Market}} \qquad \text{Equation 2.2.12}$$

The Relative Market Share measures the market share of the company as a percentage compared to its major competitors.

Calculating ratios like these does not require significant processing power. As computer capacity and processing

power increased, it became easier for analysts to visually inspect graphs of individual stock prices and values derived from them. This led to the birth of a new school of stock analysis based on charting techniques.

2.2.1 Charting Techniques

Charting techniques work with the visual representation of the stock price graph over a selected period. By enhancing the diagrams with secondary time series documenting perceived trends, the chartists identify trends or trading opportunities.

All of these charting techniques represent common analysis tools and are frequently quoted by all major investment magazines, including Capital, Börse and Wirtschaftswoche in German and Forbes, Money Magazine and Barron's in the United States. Bookstaber describes and explains these techniques in detail and shows which combination of triggers he considers particularly valuable [Bookstaber 1985].

One notable proponent of these techniques is Chrystyna Bedrij, chief investment officer at Griffin Securities in New York. She publishes a one-page document three to four times a week, called "The X list" and has gained notoriety for having produced portfolio recommendations with returns in excess of 60% since 1997. Her recommendations are naturally only available to paying customers, but frequently appear on public

web sites with a one or two week delay, including MoneyCentral on MSN.com.

Like the many ratios discussed earlier, analysts have devised numerous charting techniques worth consideration. Comparable to the Price-to-Earnings ratio in importance, the most basic enhancement used by chartists to a stock price is the trend line. It is defined by connecting local maxima or minima with a straight line. The area between the two lines is called the trend channel and provides an indication of the price tendencies.

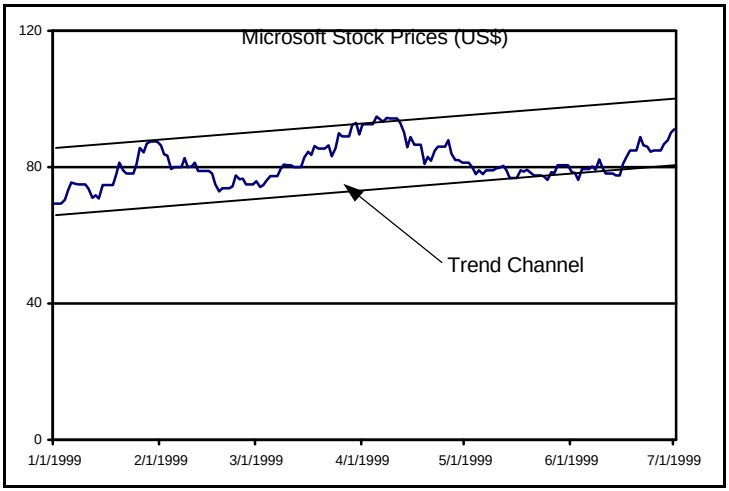

Figure 2.2.1: Trend Lines and Trend Channel

This technique is as much an art as a science since the quality of the implicit statement depends on the “correct” identification of the local maxima and minima.

Another somewhat subjective indicator is the support and resistance line. Similar to trend lines, these floors and ceilings are usually based on psychological barriers, which are frequently associated with round numbers.

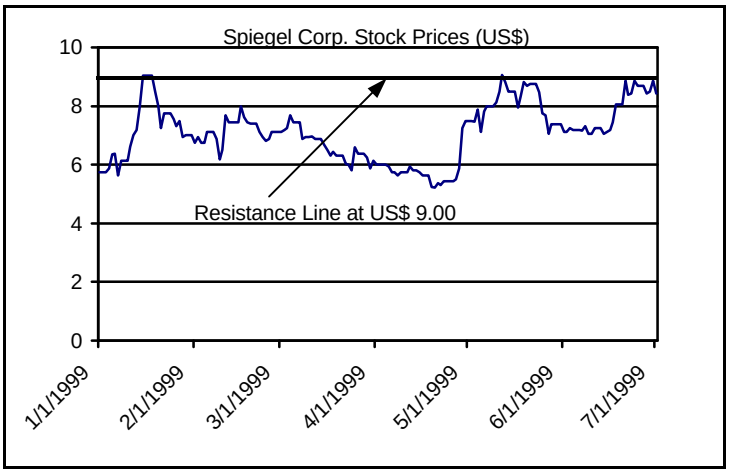

Figure 2.2.2: Resistance Line

In the example above, the Spiegel stock price has repeatedly challenged the US$ 9.00 level, but there seems to be a barrier preventing it from passing this value.

The momentum (M_t) of a price is defined as follows, where P_t is the price of the stock at time t:

$$M_t = 100 \cdot \frac{P_t}{P_{t-n}}$$ Equation 2.2.13

Using this definition, it is possible to supplement the graph of a stock with its momentum for different values of n. The momentum indicator can help identify trend reversals and is designed to show the strength of the movement. A reversal in the momentum from values smaller than 100 to bigger than 100 are interpreted as buy signals and vice versa. Interestingly, some analysts draw trend channels into the momentum lines to identify buy and sell signals.

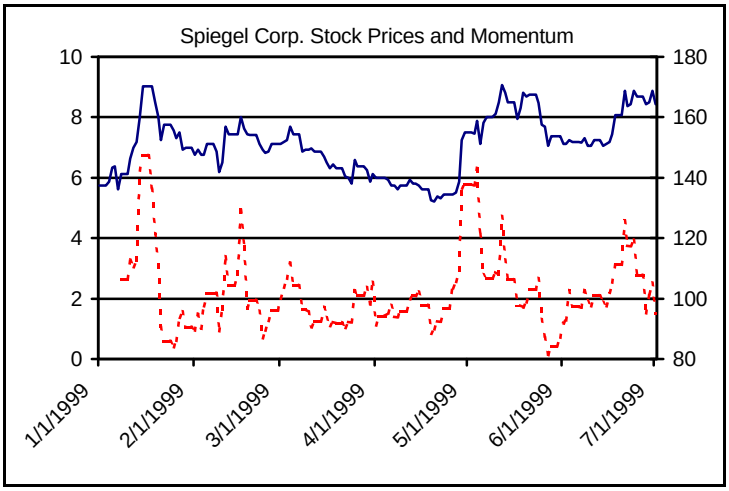

Figure 2.2.3: Momentum for n=7

Like many indicators, this technique can help identify opportunities, though some analysts claim that it primarily documents historic opportunities, instead of predicting the future. On 1/12/99 and 4/30/99 the momentum value in the diagram above climbed above 130, documenting a purchase opportunity that would have resulted in a profitable trade. It subsequently rose further and broke through the 140 level confirming the momentum.

The trend confirmation indicator (TCI_t) is the ratio of two moving averages D_n and D_m, of n and m days, with $n<m$:

$$TCI_t = \frac{D_n}{D_m} \cdot 100 \qquad \text{Equation 2.2.14}$$

A TCI value below 100 is usually interpreted as a signal forecasting a change in trends. Values above 100 confirm the current trend and provide an indication of its strength.

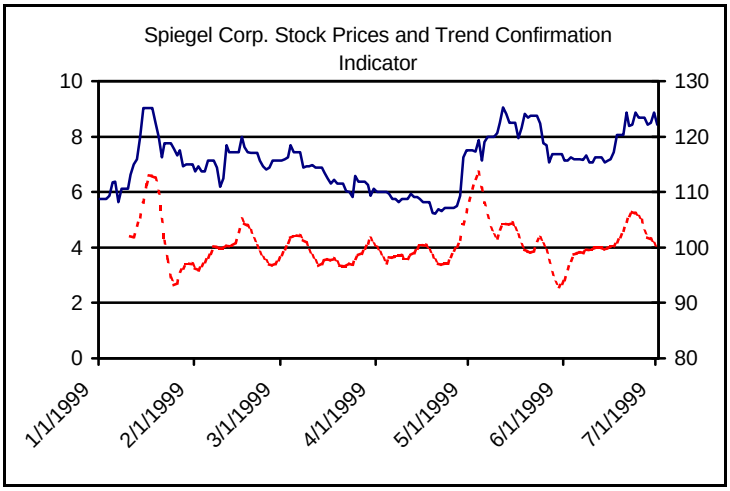

Figure 2.2.4: Trend Confirmation Indicator with n=5 and m=10

We see that this indicator can, at times, provide helpful predictions. On 4/30/99, for example, the *TCI* climbed to above 110, which would have allowed for a lucrative investment in the Spiegel stock.

Another popular indicator graphs a short and a long term moving average on the same graph. Common values for this moving average (MA) comparison are 50 and 200 days. A simple trading rule states that if the short term moving average crosses the long term moving average from below (above), the stock price shows a trend of increasing (decreasing) strength and promises to continue rising (falling).

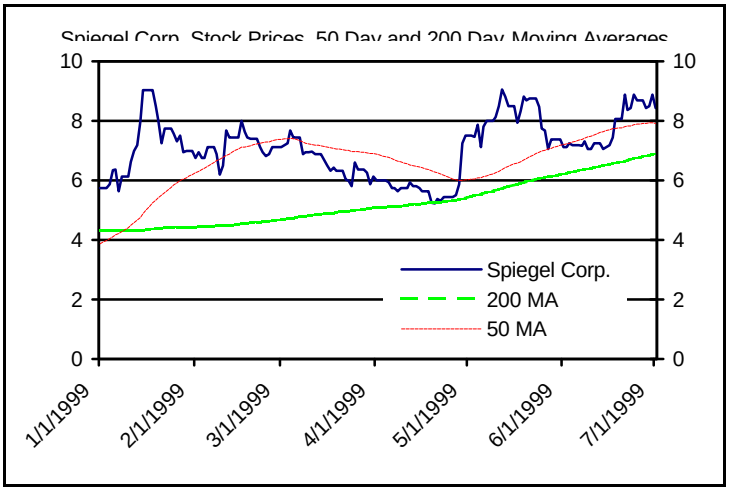

Figure 2.2.5: Comparison of the 50 and 200 Day Moving Average

In this example, this MA indicator triggered a buy signal on 1/9/99 and indeed, the stock price rose from around US$ 6.00 to close to US$ 9.00. It did not, however, identify the subsequent decrease in stock price, nor did it trigger any other buy or sell signals in the period displayed.

Trend Oscillators (TO_t) are the relationship between the current price and a moving average.

$$TO_t = \frac{P_t}{D_n} \cdot 100 \qquad \text{Equation 2.2.15}$$

This indicator is based on the theory that commodities oscillate within a defined trend channel over extended periods.

Given this indicator, it is possible to fine-tune the timing of a purchase. Values above 110 (below 90) are generally interpreted as a "buy" ("sell") signal.

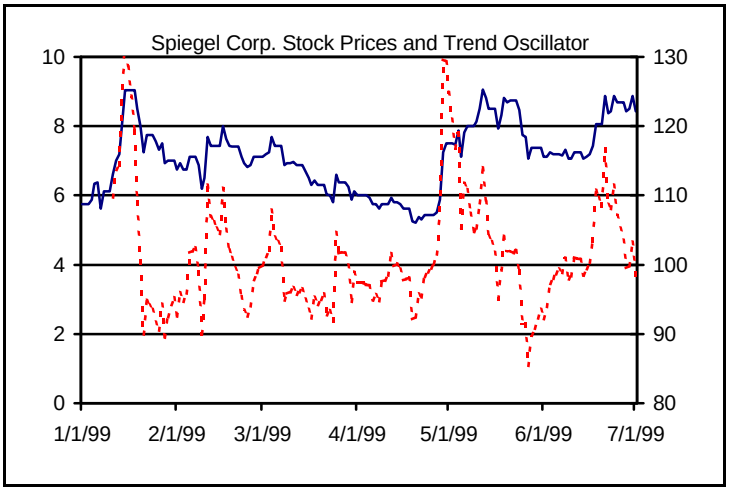

Figure 2.2.6: Trend Oscillator with a 10-Day Moving Average

Again, we see that on 1/12/99 and on 4/30/99 the trend Oscillator exceeds 110 and are followed by values up to 130. A purchase on either of these days would have been followed by a substantial increase in price within a few days.

The over-bought/over-sold indicator ($OBOS_t$) is designed to identify stocks that are currently traded at an inefficient price.

$$OBOS_t = \frac{H_n - P_t}{H_n - L_n} \cdot 100 \qquad \text{Equation 2.2.16}$$

In this equation, the values H_n and L_n are the high and low prices in the previous n days. Generally, it is assumed that a value over 90 forecasts a price reduction while a result below 10 indicates a price increase.

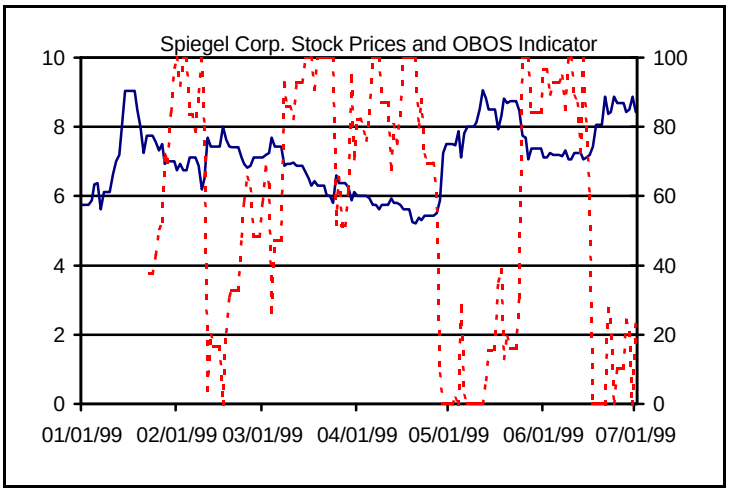

Figure 2.2.7: The Over-Bought/Over-Sold Indicator with n=20

The $OBOS_t$ indicator clearly reacts more sensitively than the previous ratios, resulting in numerous buy and sell signals. Like the previous charting techniques, it identified the 4/30/99 buying opportunity. However, there were also some "false alarms" at the beginning of February and on March 20, 1999.

These issues showed that even the chartists are not infallible and in 1986, Frankel and Froot suggested that it is necessary to take the expectations from both fundamentalists and

chartists into account, if one is to understand financial markets [Frankel, Froot, 1986]. As an example, they analyzed the value of the US dollar and devised a meta-model, based on a combination of models. They showed that the complex behavior in the years before the paper was published could be explained by the inter-play between these chartist and fundamentalist schools of thought.

After the stock market crash on October 19, 1987, non-linear dynamics and especially deterministic chaotic systems became a major topic both among the financial press and academic literature. Since the violent swings could not be explained with the usual assumptions, this approach was seen as an alternative model for the stock market.

Hsieh, for example, analyzed the S&P 500 Index between 1982 and 1990 and concluded that non-linear methods offer promising new venues in the attempt to model this data [Hsieh 1990]. His extensive tests show evidence that the stock returns tested are not independent and identically distributed.

Hutchinson also bases his work on the assumption that financial time series are fundamentally non-linear in nature [Hutchinson 1994]. He shows that though it is difficult to benefit from them, models based on radial basis functions provide better forecasts than linear predictors.

This was good news to the proponents of complex, chaotic models and a variety have been developed and tested over the past years. With their proliferation, a third approach to

stock evaluation evolved and Robinson and Zigomanis propose the extension of the work of Frankel and Froot to include the expectations of the non-linear dependence of financial data in order to optimize these models [Robinson, Zigomanis 1999].

The following sections address mathematical prediction models starting with linear auto-regressive methods as a base line. Subsequently, I focus on more advanced non-linear approaches using k-nearest neighbors, Markov Models and artificial neural networks.

2.2.2 Auto-Regressive Models

The literature usually distinguishes between two types of Linear Models: AR(p) or Auto-regressive Models of degree p have the form

$$\hat{X}_t = \sum_{i=1}^{r} j_{i,r} X_{t-i} \qquad \text{Equation 2.2.17}$$

and MA(q) or Moving Average Models of degree q are defined as follows

$$X_t = \sum_{i=1}^{r} q_i Z_{t-i} \qquad \text{Equation 2.2.18}$$

where Z_t are elements of a white noise process.

Though the two can and frequently are combined to form ARMA (p,q) models, we concentrate on AR(p) models because any MA(q) process can be represented as an AR(∞) process.

The explicit representation of an AR(p) model entails the determination of the p weights j_{ip}, which is frequently accomplished with the Durbin-Levinson algorithm. Brockwell and Davis show an elegant derivation for this method that uses a recursive scheme to circumvent the need for a large matrix inversion [Brockwell, Davis 1986].

The algorithm requires a stationary process with a constant arithmetic mean and an autocovariance function such that $g(0)>0$ and $g(h)®0$ as $h®¥$. For the algorithm, the mean squared error of the prediction n_n is defined as

$$v_n = E\left(X_{n+1} - \hat{X}_{n+1}\right)^2 \qquad \text{Equation 2.2.19}$$

Using the standard definition for the estimated autocovariance function

$$g(h) = \frac{1}{N-h}\sum_{i=1}^{N-h}\left(X_i - \overline{X}\right)\left(X_{i+h} - \overline{X}\right) \qquad \text{Equation 2.2.20}$$

it follows that

$$v_0 = \hat{g}(0) \qquad \text{Equation 2.2.21}$$

The coefficients of the AR models are calculated using the following equation.

$$j_{nn} = \frac{g(n) - \sum_{i=1}^{n-1} j_{n-1,i} g(n-i)}{v_{n-1}}$$ Equation 2.2.22

The AR(1) model follows immediately:

$$j_{1,1} = \frac{g(1)}{g(0)}$$ Equation 2.2.23

Given this recursive anchor and the following equations, it is possible to compute $j_{n,m}$ for $n=2, 3,\ldots$ and $m=1,\ldots n$ as well as n_n providing the coefficients for the AR(n), n>1, models.

$$\begin{pmatrix} j_{n,1} \\ \vdots \\ \vdots \\ j_{n,n-1} \end{pmatrix} = \begin{pmatrix} j_{n-1,1} \\ \vdots \\ \vdots \\ j_{n-1,n-1} \end{pmatrix} - j_{n,n} \begin{pmatrix} j_{n-1,n-1} \\ \vdots \\ \vdots \\ j_{n-1,1} \end{pmatrix}$$ Equation 2.2.24

$$v_n = v_{n-1}\left(1 - j_{n,n}^2\right)$$ Equation 2.2.25

AR(n) models compute a weighted mean of past values. Though very useful and easy to compute, the method does not perform well when the underlying system contains non-linear dependencies.

The results from Hsieh indicate that most financial data are non-linear in nature resulting in a natural disadvantage for these models [Hsieh 1990]. Additionally, due to their simple nature they are used widely as a baseline for comparison but consequently offer no competitive advantage.

In an effort to benefit from the extensive research done for AR models and the numerous well-documented algorithms that exist, it is possible to enhance the basic algorithm in various ways. By dividing the input space into two or more regions, defined by the Euclidean distance to their respective centers in n-dimensional space it is possible to generate different local linear models. Each approximates the function linearly in their respective input spaces. The prediction is the weighted sum of individual models, based on the distance to the input space. This generalization requires sufficient data to generate several local models and the quality of the results depends on the choice of the centers used to define the separate regions, but generally helps to reduce the model error.

Mâlâroiu, Kiviluoto and Oja proposed a different enhancement to generic time series prediction [Mâlâroiu et al 1999]. After preprocessing the target time series to zero mean and unit variance, they separate it into different independent spectral components using the FastICA package in MATLAB. Each component is then filtered to reduce the effects from supposed noise, by applying a high-pass and/or low-pass filter. The individual components are then modeled using the AR method

and combined by calculating the weighted sum of each prediction.

Hsieh developed a similar model, which decomposes exchange rate futures contracts into a (linear) predictable and a (non-linear) unpredictable component [Hsieh 1993]. He focused on US dollar contracts traded on the Chicago Mercantile Exchange for the British Pound, German Mark, Japanese Yen and Swiss Franc. Though this approach does not accurately calculate the expected prices, it is able to isolate the autoregressive volatility of the data allowing him to forecast the prediction risk.

In the Santa Fe Time Series prediction competition organized by Weigend and Gershenfeld, Sauer concentrated on the prediction of data set A, the intensity of a detuned NH3-FIR Laser [Weigend, Gershenfeld 1993, Sauer 1993]. By using delay coordinate embedding, he successfully built local-linear models to predict the output data. In the same competition, Lewis, Ray and Stevens modeled the time series A, B and C, which additionally included physiological and foreign currency exchange data. They used multivariate adaptive regression splines in another example where a linear concept is expanded in scope so that it can be applied to the non-linear domain.

2.2.3 K-Nearest Neighbors Models

The k-nearest-neighbors models (KNN) search the training data for historic points in n-dimensional space that correspond to the current configuration. The assumption is that similar configurations in the past are followed by values, which can be interpreted as predictions for the future. Usually the predictions are the weighted sum of k of these nearest neighbors based on distance, hence the name.

In its basic form, the algorithm defines a data window $\mathbf{K}_t = (x_t \ldots x_{t-n-1})$ where x_t is the value of the time series at time t. As a next step, it calculates the distance d_{t-m} between $\mathbf{K}_t$ and $\mathbf{K}_{t-m}$ for all $m<t-n$. A common metric used is the Euclidean distance, shown in the following equation, though numerous alternatives are possible.

$$d_{t-m} = \sqrt{\sum_{i=0}^{n-1} \left(x_{t-i} - x_{t-i-m}\right)^2} \qquad \text{Equation 2.2.26}$$

The resulting scalars are sorted in increasing order so that the k closest data windows $\mathbf{K}_{t1} \ldots \mathbf{K}_{tk}$, can be identified. Now simply taking the value following the historic neighbors $x_{t1+1}, \ldots x_{tk+1}$, the system has identified k predictions for the future value of the time series.

Commonly, these k nearest neighbors are averaged, using the distances $d_{t1} \ldots d_{tk}$ as a means of weighting mechanism.

$$x_{t+1} = \sum_{i=1}^{k} \frac{x_{t_{i+1}} \cdot d_{t_i}}{\sum_{j=1}^{k} d_{t_j}}$$ Equation 2.2.27

Frequently, the algorithm is extended by including values from related time series y_t, or z_t, representing the trading volume and general economic data like inflation, interest and unemployment rates, as well as the price history of major indexes or related stocks. This tends to increase the dimensionality of the data window, but does not affect the complexity of the remaining algorithm.

$$\hat{e}_t = \left(x_t..x_{t-n-1}, y_t..y_{t-m}, z_t..z_{t-l}\right)$$ Equation 2.2.28

This is a general model and performs well both for linear and non-linear time series. The results of the model can be reconstructed because the system can list the points it used for input, making the predictions very transparent.

The quality of the model deteriorates when the time series enters “uncharted territory” or domain space for which no previous examples exist. This can be avoided somewhat by normalizing the input, though this usually increases the error for the known input space.

2.2.4 Markov Models

Markov Models (MMs) assume that it is only possible to obtain certain observations of the system that describe its state, possibly incompletely. In the space-time continuum, the system „jumps“ from one state to the next. The idea is that if one observes the system long enough, one can note the progression from any state s_x to state s_y^1, s_y^2... s_y^n. With this information at hand, it is possible to calculate the probability that the state following s_x will be s_y^i for all i. If the system should end up in state s_x in the future, it is possible to determine the likelihood for each state that it will be the next one.

When we apply this method to time series analysis, we first have to define our data window w. Given a training set of N data points we are now able to define N-w=P training tuples. Each of these tuples represent a move in w-space from state $s_{t-1} = (s_{t-w}, \ldots, s_{t-1})$ to state $s_t = (s_{t-w+1}, \ldots, s_t)$. This can also be interpreted as the functional $f(s_{t-1}) \rightarrow s_t$ where $1 \leq t \leq P$. In the next step, we divide the w-space into $k \leq P$ clusters. This is done by dividing the bounded w-dimensional hypercube into $\mathbf{K}=n^w$ smaller hypercubes or by randomly selecting $\mathbf{K}$ points and attaching each tuple to its nearest representative. The second method would necessitate the definition of a metric to identify the “nearest” point, with all common variants possible.

Having now categorized the states, one would have to determine the probabilities of a transition between any two states by counting the total number of transitions from one state to another. Once the probabilities are calculated, the model is fully specified and can be used to predict the next state if a new point is presented. The prediction follows from the most probable state. Calculating the weighted mean of all follower states can extend this model. Many schemes are conceivable, with linear and exponential weights used most commonly.

The state of a financial time series is frequently identified both by the price and volume of the recent trading days. This algorithm is also frequently enhanced by including related time series as described in the KNN models.

Fraser and Dimitriadis used MMs in speech research and recognized their possibilities for time series prediction when they became aware of the Santa Fe competition [Fraser, Dimitriadis 1993]. Their contribution focused on data set D, a numerically generated series representing a chaotic process. The authors describe their use of Baum's EM algorithm, which iteratively adjusts the model parameters to maximize the likelihood of its observations and apply the resulting model to forecast the data. For every point, they are able to map the probability density making it possible to attach a confidence to the predicted value. The latter is especially appealing for stock trading predictions.

Poritz and Rabiner also came from the speech recognition field and used Markov Models to forecast the likelihood of a particular word given a certain beginning of a phrase [Poritz 1998] [Rabiner 1989]. In their work, the states described recent words in a phrase. Given numerous historic examples, the system is able to question every interpreted word and replace it with one that “makes more sense”, or mathematically expressed, where the likelihood of its placement in a particular position is higher.

2.2.5 Artificial Neural Network Models

Artificial neural networks (ANN) are based on research of the human brain. Here neurons receive input from n different electrical sources, weight the inputs through an electrical resistance and sum the results. The output of the neuron is a transformation of this sum and is fed into the next neuron. An ANN simulates this operation within a computer program.

The human brain consists of approximately 10^{10} neurons, all of which are active simultaneously and can be interconnected. The computer equivalent consists of only a few tens or hundreds of the electronic "neurons" called a unit, and normally only one is active at any one time.

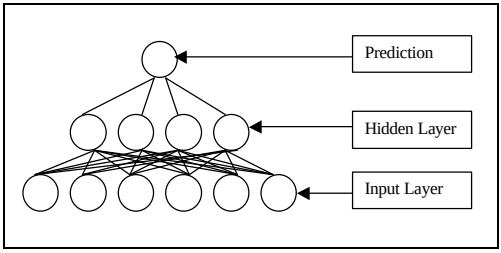

Figure 2.2.8: Artificial Neural Network

In the figure above, one can detect a strict hierarchy, with clearly identifiable layers. Each layer is fully connected to the next. Though recurrent connections are conceivable and sometimes used, we will restrict ourselves to feed-forward ANNs with this structure. The bottom layer in the diagram represents the input layer. The activation of this layer is determined by the input into the system, for example I_{t-1}, I_{t-2},..., I_{t-n}. The input data is then propagated to the next (hidden) layer as represented by the connecting line in Figure 2.2.8. Each of the four units in the middle layer sum their weighted inputs and scale the output using the transfer function $s(h)$.

$$o_j = s\left(\sum_{i=1}^{6} w_{j,i} I_i\right) \qquad \text{Equation 2.2.29}$$
$$j = 1,2,3,4$$

The I_i in this case are the values from the input units and the weights $w_{j,i}$ represent the axon or weight connection between input i and hidden unit j. It is common to use a sigmoidal function for $s(h)$, like

$$s(h) = \frac{1}{1 + e^{(-2hb)}}$$ Equation 2.2.30

or

$$s(h) = \tanh(hb)$$ Equation 2.2.31

with the alternatives being linear or sinusoidal functions for an input between -p and p. Similarly, the output unit or prediction sums and transforms the activation values from the hidden layer.

$$O_1 = s \sum_{j=1}^{4} W_{1,j} o_j$$ Equation 2.2.32

The example in the figure contains only one hidden layer and an output layer with one unit, though more output units and more layers are possible.

The output unit contains the desired prediction and depends on all the weights in the net. The ANN has to „learn“ the data to search its configuration space for a good set of weights.

This is usually done using the back propagation algorithm. In it, we define the error of the prediction of a particular pattern as a function of the weight vector to the output unit as follows, where O_i is the output of the ANN and T_i represents the target value.

$$E(\vec{w}) = \frac{1}{2}\sum_{i=1}^{i_{\max}} (O_i - T_i)^2 \qquad \text{Equation 2.2.33}$$

By now differentiating the error expression with respect to each weight w_{ij} in Equation 2.2.33 we can determine an expression that will reduce the error using the gradient descent method with a step-size ç.

$$\Delta W_{ij} = -h\frac{\P E}{\P W_{ij}} = h(O_i - T_i)s\,'O_1\sum_{k=1}^{k_{\max}} W_{ik}O_i \qquad \text{Equation 2.2.34}$$

The error is then propagated to the next layer of units where the same algorithm is then used. This update routine can be applied after the presentation of each pattern (on-line learning) or collected for all the patterns in a training set (batch mode learning). After running through all training patterns, the model is verified on the test set. Training continues as long as the test error is reduced with each iteration. This ensures that the ANN does not start modeling stochastic noise in the training set, a process called "overfitting." It is common for the error to vary dramatically in the first iterations, so that many algorithms permit the definition of a minimum number of iterations to be tested.

From this description, it is obvious that this method is computationally considerably more expensive than the previous algorithms. Also, there is no easy analytical method to determine the number of units necessary in the hidden

layers, or to make other topological decisions so that it is necessary to find the optimal configuration using trial-and-error. The high number of degrees of freedom poses some difficulties as well, since it reduces the stability during the convergence process. However, the gain, once a good configuration is found, is a truly non-linear function predictor.

This difficulty associated with models built using artificial neural networks is exemplified by an experiment conducted by White [White 1989]. He attempted to predict the quarterly IBM stock prices with a static network topology of five input and five hidden units. The single output unit was trained with data points from the second quarter of 1974 until the first quarter of 1978. The resulting artificial neural network was used to predict prices from the second quarter of 1972 until the first quarter of 1974 as well as the second quarter of 1978 until the first quarter of 1980. Though the network was able to model the training set well, the project had no control over the extent of the training on the network, so that its ability to generalize was unsatisfactory. For the two test sets, the correlation between the predicted and the real values was 0.0751 and –0.0699 respectively. This example shows that this powerful tool can not be applied blindly without tailoring the network to the specifics of the time series.

With increased complexity and consideration however, ANNs offer good opportunities. Rehkugler and Poddig show the potential of ANNs in an experiment, which included only three input values and was designed to predict the movement of the

German stock index DAX [Rehkugler, Poddig 1990]. Instead of using the prices of the index itself, the ANN was based on the nominal interest rate, a business confidence indicator and free liquidity, defined as follows:

$$L = \frac{\mathrm{M1}}{\mathrm{GNP}}$$ Equation 2.2.35

In this equation, the money supply M1 is divided by the gross national product, GNP. In order to eliminate seasonal swings, the input to the ANN included only the change in value compared to the previous year. The system was trained with this delta data from the first quarter 1965 with 6 years worth of information. The topology included different numbers of hidden layers each with different number of units. The network had five output units. One was trained to calculate a simple rise/fall predictor with the values 1 or 0 respectively. The remaining four output units defined index changes of bigger than 10%, between 0% and 10%, between 0% and –10% or smaller than –10%. During every training cycle, only one of these four units was trained with the value 1. The others were set to 0.

After training, the network was tested with 68 values. For the rise/fall indicator, values of 0.5 or greater were interpreted as a rise, while lower values were considered a falling prediction. For the four categorization output units, the algorithm assumed that the output with the highest value was the “winner” and interpreted it as the prediction.

As a simple trading strategy, the program simulated a purchase of the index as if it were a stock if the prediction was positive and it did not already own it. Similarly, it sold the index, if it owned it and the network predicted a decrease. The basis for this decision was the first rise/fall indicator. The table below summarizes the results.

Topology	Rise/Fall Output	Categorization	Return
3-5	49 correct, 19 false	43 correct, 25 false	177.28%
3-5-5	49 correct, 19 false	47 correct, 21 false	160.23%
3-9-7-5	44 correct, 24 false	44 correct, 24 false	142.29%

Table 2.2.1: First Experiment [Rehkugler, Poddig 1990]

Interestingly, the return as well as the correct prediction frequency responded negatively to increased complexity in the network. Nevertheless, the average annual returns well over 100% suggest an opportunity, although transaction costs are not considered.

Since this artificial neural network effectively produced two different outputs, the two scientists simplified the output in a second experiment to the simple rise/fall indicator. This approach ensured that the secondary categorization goal did not interfere with the desired prediction. In an effort to improve the forecasts, the output in this second experiment was interpreted more stringently: Only outputs over 0.9 and under 0.1 were considered rise or fall predictors respectively. All other values specified an undefined state and resulted in a

"hold" strategy for the hypothetical stock. The results are shown in the table below:

Topology	Rise/Fall Output	Return
3-1	32 correct, 13 false	225.16%
3-2-1	42 correct, 16 false	194.39%
3-3-1	43 correct, 14 false	237.65%
3-4-1	44 correct, 19 false	194.36%

Table 2.2.2: Second Experiment [Rehkugler, Poddig 1990]

The results indicate that the specialized approach increased returns considerably, and that the number of undecided states primarily helped reduce the number of false predictions. It is also noteworthy that the increase in network complexity reduced the undecided states and improved the number of correct predictions. Though the return did not immediately benefit from the increased quality of the predictions, one should question whether a simple portfolio management strategy is an adequate measure for this model.

In an attempt to improve the generalization ability of artificial neural networks, Utans and Moody extended the methodology in a study that was designed to predict the Standard & Poor (S&P) rating for assorted companies [Utans, Moody 1991]. These ratings define the risk associated with an investment in a particular company or market. S&P categorize the companies in 18 discrete steps. Since these ratings are updated infrequently, this ANN helped interpolate the rating between official releases.

The ANN used ten financial ratios as input data, a hidden layer and a single output unit, which categorized the risk rating for the company between 2 and 19, in the order of decreasing risk. The hidden units used sigmoidal transfer functions. In contrast, the output unit used a piece-wise linear transfer function in order to reduce the complexity and thereby increasing the training speed.

$$f(x) = \begin{cases} 1 & \text{for } x > 16 \\ \frac{x}{32} + \frac{1}{2} & \text{for } 16 \geq x \geq -16 \\ 0 & \text{for } -16 > x \end{cases} \qquad \text{Equation 2.2.36}$$

The authors tested this configuration with different numbers of hidden units, in order to determine the optimal topology.

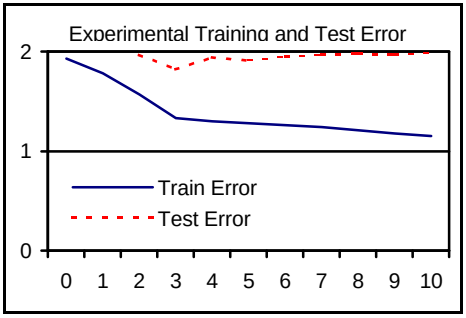

Figure 2.2.9: Utans, Moody Experimental Training and Test Error

The results show a pronounced change of trend in the training error. This point coincides with the minimum test error,

indicating that the optimal configuration should include three hidden units.

As a next step, Utans and Moody defined the importance of the i input data by defining the sensitivity of the output to the input.

$$S_i = \frac{1}{N} \sum_{p=1}^{N} \frac{\partial m}{\partial X_{pi}}$$ Equation 2.2.37

where $N = \text{number of input tuples}$

By sorting the sensitivity in decreasing order, the two authors identified the test error as more of the input data was removed.

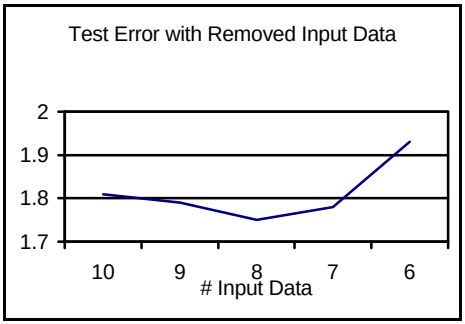

Figure 2.2.10: Utans, Moody Test Error with Removed Input Data

This approach shows that the removal of two input data actually helped improve the test error of the resulting ANN.

In a separate effort to improve the performance of the model, the authors applied the Optimal Brain Damage (OBD) method, as presented by Le Cun [Le Cun et al 1990]. This approach defined the influence of each weight in the network through the saliency function, defined in the equation below.

$$s_i = \frac{1}{2}\frac{\partial^2 E(w_i)}{\partial w_i^{\,2}} w_i^{\,2}$$ Equation 2.2.38

By resetting the weights with the lowest saliency, their effect on the output is effectively removed. The resulting network is retrained so that it can adjust to this "brain damage." The figure below shows the effect of this adjustment to the test error.

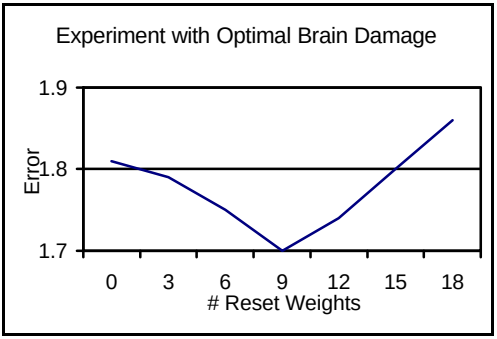

Figure 2.2.11: Utans, Moody Experiment with Optimal Brain Damage

These two comparisons both showed a promising improvement but were strangely not combined into a single model.

The results of these two ANNs were categorized by the deviation of the actual S&P classification. The table below shows that percentages for each error group.

Deviation	8 Input Units, No OBD	10 Input Units, with OBD
0	31.1 %	29.1 %
1	39.3 %	39.3 %
2	15.8 %	17.9 %
>2	13.8 %	13.7 %

Table 2.2.3: Error with Reduced Input and ODB Methods

Approximately 70 % of the test set were categorized correctly or were only off by one, resulting in reasonably accurate predictor for the S&P risk rates.

In a different attempt to optimize the network topology and improve the ability to generalize, Fahlman and LeBiere introduced cascade-correlation networks [Fahlman, LeBiere 1990]. In contrast to the standard back-propagation algorithm, in these networks the size of the hidden layer is not static.

Initially, a two-layer network is trained until the error for each vector pair in the training set falls below a defined threshold. When this occurs, a neuron is added to the hidden layer. The synaptic weights between the input layer neurons and the new

neuron are adjusted to maximize the magnitude of the correlation between its activation and the output layer error.

Deppisch, Bauer and Geisel generalize this idea in their paper on hierarchical networks [Deppisch, et al 1991]. The concept envisioned training an ANNs until it is no longer able to model the complexities of input data. In a second step, a new ANN is trained to predict the residues of the original model. Finally, the two ANNs are used together as a predictor of the time series.

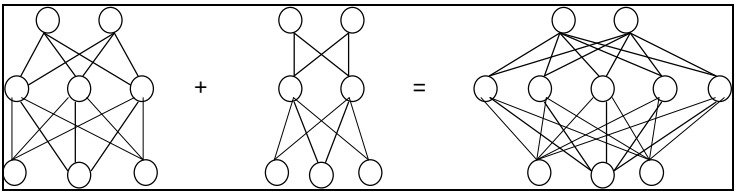

Figure 2.2.12: Hierarchical Networks

The diagram above shows the combination of two ANNs but this procedure could be extended indefinitely to continually reduce the residual model. The final output is the sum of all sub-models.

The authors tested this approach on the chaotic coupled differential Rössler equations defined in the equation below.

$$\frac{dx}{dt} = -(y+z)$$
$$\frac{dy}{dt} = x + 0.2y \qquad \text{Equation 2.2.39}$$
$$\frac{dz}{dt} = 0.2 + z(x-5.7)$$

In order to approximate the optimal learning per network, the first ANN had a topology using one input x value and two hidden units to predict the next x value. Network 1 was trained 10^3 epochs until it produced an average test error of 10^{-2} and then supplemented with a second ANN with a 1-6-1 topology until the test error reached a minimum. In Network 2, the original 1-2-1 ANN was trained until it produced an average output error of 10^{-3} and was then supplemented with the same 1-6-1 ANN. As a comparison the following diagram include the test error of the 1-2-1 without a secondary ANN and the error of an ANN where all eight hidden units are trained from the beginning.

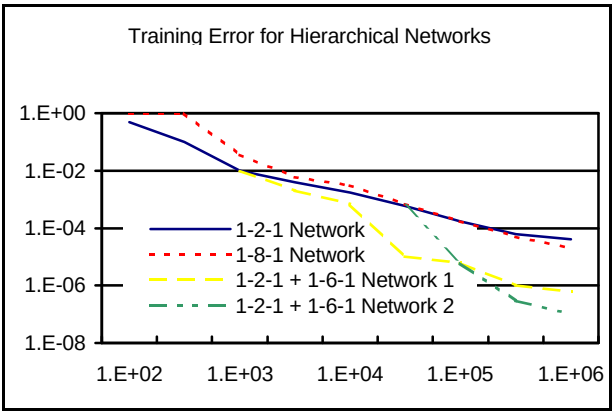

Figure 2.2.13: Training Error for Hierarchical Networks

Not surprisingly, the 1-2-1 network "learned" the data quicker than the more complex 1-8-1 network. After several thousand epochs it caught up and eventually produced a slightly lower training error. The first hierarchical network initially improved the simpler model only slightly. It only showed a significant error reduction after 10^4 epochs reducing the training error by almost two orders of magnitude. The second network immediately caught up this level and was apparently able to continue improving its model even in the next epochs. After 10^6 epochs, the training error leveled off for all models.

A possibly more significant measure of the quality of the model was the prediction quality measured by the test error. The diagram below shows a comparison of the most successful of the hierarchical models in comparison to a linear predictor.

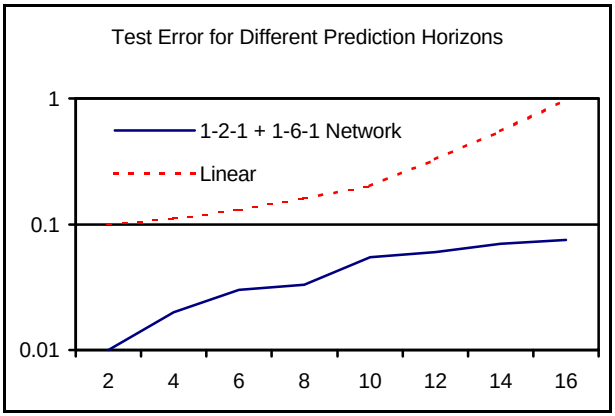

Figure 2.2.14: Test Error for Different Prediction Horizons

Though significantly larger, the hierarchical predictor is an order of magnitude better than the linear model and though unsurprisingly both errors increase with a longer prediction horizon, the ANN remains about ten times better throughout the domain shown in Figure 2.2.14.

The experiment shows that hierarchical networks are able to offer advantages in at least some cases. The difference in training and test errors point to overfitting, an issue that would need to be addressed, if this approach were to be used in a system.

Artificial Neural Networks also received a big push in popularity from two entries by Wan (data set A) and Mozer (data set C) to the Santa Fe Time Series Prediction Competition [Weigend, Gershenfeld 1993, Wan 1993, Mozer

1993]. The latter also showed that these models could be applied successfully in the financial domain, an aspect of particular relevance to this paper.

Networks based on radial basis functions have also been applied to financial time series prediction. As early as 1986, Hutchinson showed that these non-linear models performed better than linear and univariate models in a simulated stock trading comparison [Hutchinson 1986]. Building on the theoretical foundation, he developed a system that was also applicable to stock option pricing, showing the versatility of the approach.

Parkinson analyzed preprocessing techniques and their application to financial time series prediction using neural networks [Parkinson 1999]. He proposes a method whereby it should be possible to compare scaling, logarithmic transforms, smoothing, differences and ratios as well as normalization. However, the sheer number of combinations of these preprocessing techniques makes it difficult to identify which one of them optimally suits what kinds of data.

Nevertheless, artificial neural networks have been applied extensively to predict financial time series. Zimmermann enthusiastically supports their usage, because they combine the non-linearity of multi-variate calculus with the number of variables typically used in linear algebra [Zimmermann 1994]. Working with him, Braun developed several applications for the prediction of the DAX, which included manual optimization

techniques like weight and input pruning as well as the merging of hidden units and layers [Braun 1994].

2.2.6 New Approaches to Financial Market Analysis

In recent years, several new agent based approaches to the analysis and prediction of financial data have gained popularity. In 1997, Arthur, Holland, LeBaron and Tayler proposed an artificial stock market (ASM) where N simulated players, called agents, each had the option to buy, sell or keep their current position [Arthur et al 1997]. The experiment was permitted to run for 250,000 periods during which each agent was permitted to execute orders once. Each agent forecast the price of each commodity using auto-regressive models, which it adapted using genetic algorithms. The trading rules were based on a parameterized strategy, which was permitted to evolve using a genetic algorithm over the course of the experiment. This project spawned several relevant research initiatives.

One was the experiment by Joshi and Bedau, who assume that investors continually explore and develop expectation models, buy and sell assets based on the predictions of the models that perform best and confirm or discard these models based on their performance over time [Joshi, Bedau 1998]. They used the ASM to explore the volatility of prices and average wealth earned by the investors as a function of the frequency of strategy adjustments.

In their experiments, they simulated a market with a fixed number $N=25$ of agents. Time was discrete and in each interval, the agents had to decide whether to invest their portfolio in a risky stock or a risk free asset, analogous to a Treasury Bond. There was an unlimited supply of the risk free assets and it paid a constant interest rate of $r=10\%$. The risky stock, issued in S shares, paid a stochastic dividend that varied over time governed by a process that was unknown to the agents.

The agents applied forecasting rules to their knowledge of the stock's price and dividend history and performed a risk aversion calculation and decided how to invest their money at each time period. The price of the stock rose if the demand exceeded the supply and fell if the supply exceeded the demand. Each agent can submit a bid to buy or an offer to sell fractions of shares at the previous period's price.

Their results were classified in to four different classes of behavior depending on the frequency of their updates using the genetic algorithm.

Genetic Interval	Volatility of Prices	Average wealth earned	Complexity of Forecasting Rules
Never	Low	Low	Low
Every Iteration	Low	High	Very Low
10^2 < interval < 10^3	High	Low	Very High
10^3 < interval < 10^4	Moderate	High	High

Table 2.2.4: ASM Investor Types as a Function of GA Interval

Not surprisingly, the volatility of the simulated market was low, if the investors never updated their forecasting model, since each remained with the set of rules they were originally endowed with. Interestingly, the volatility of the price structure remains low when the agents update their rules at every interval and increases dramatically as the genetic interval increases to somewhere between 10^2 and 10^3. With this configuration, the complexity of the forecasting rules was also very high. At the same time, the average wealth earned by the investors was high if they adjusted their strategy on every iteration, then dropped to a low as soon as there are some changes, but peaked if the investors modify their strategy every 10^3 to 10^4 intervals.

This work was used to explain the rapid increase in volatility of the financial markets in recent years, by implying that the average trading strategies of the investment community has changed. Since it is assumed that professional investors, which represent the traditional players, do not adjust their strategy very rapidly, the authors suggest that this increase is due to privat investors, which tend to trade with a higher frequency and have presumably grown their share of the trading volume with the increasing ease of online trading via the Internet

Also based on the ASM, Kurumatani proposes a virtual stock market, Vsmart, where researchers worldwide can inquire about stock prices, and can execute purchases and sales, just like in its real world counterpart [Kurumatani et al 2000].

Unlike actual markets, no actual money will transfer ownership and all trades, trading histories and results are open to all participants. This results in complete transparency and hopefully in some insights into the dynamics of stock markets, multi-agent research and profitable trading strategies as well as human decision-making.

To this end, the authors provide a Simple Virtual Market Protocol, SVMP, which allows the academic community to automate the transaction chain via the Internet. This process allows autonomous agents to retrieve current and historic market data, which can be processed using any kind of algorithm. Using the resulting predictions, the agent can send order inquiries. The central VSmart Server matches offers and bids, thereby determining the resulting price and providing an immediate response whether the order resulted in a transaction.

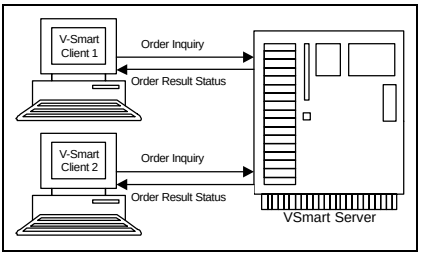

Figure 2.2.15: The VSmart Virtual Stock Market

Since all transactions are tracked and available for all participants, this open approach can bring together

heterogeneous agent types. A human interface via the Internet, even allows participants, who have not fully automated the transaction chain to interact with the server. Due to its completely transparent design, this project promises to provide interesting results, when it goes live.

In 1990, Granger hypothesized that trading volume and price movements were related [Granger 1990]. Karpoff found that stock price and trading volumes are related for bull markets, supporting this Grangers causality [Karpoff 1987]. Taking this work as a basis, Chen, Yeh and Liao extended the algorithms used in the ASM to analyze the old saying on Wall Street that "it takes volume to make price move" [Chen et al, 2000]. They developed a similar artificial agent based stock market and tested for Granger causality between these two components and showed that the relationship exists in all examples tested. This finding also emphasized the validity of the agent based simulations and reinforced this new methodology.

Ingber and Mondescu developed an interesting alternative based on an Adaptive Simulated Annealing (ASA) approach to generate buy and sell signals for S&P futures contracts [Ingber, Mondescu 2000]. The algorithm fits short-time probability distributions to observed data using a maximum likelihood technique on the Lagrangian. It was developed to fit observed data to the following function.

$$dF = f^F dt + s F^x dz(t) \qquad \text{Equation 2.2.40}$$

Using this equation, it defined the market momentum as follows:

$$\Pi^F = \frac{\frac{df}{dt} - f^F}{s^2 F^{2s}} \qquad \text{Equation 2.2.41}$$

In these equations, f^F represents the drift, s the standard deviation defining the volatility, $F(t)$ is the S&P future price and dz the standard Gaussian noise with zero mean and unit standard deviation. The parameter x was used to adjust the system to the current market conditions. The system sampled the data with different time resolutions Dt and averaged actual tick data that fell into a particular sample interval.

The ASA algorithm calculated optimal parameters for the drift f^F and the parameter x in the equations above as soon as sufficient data was available in the trading day and periodically thereafter. By then defining the null momentum

$$\Pi_0^F = -\frac{f^F}{s^2 F^{2x}} \qquad \text{Equation 2.2.42}$$

the momentum uncertainty band

$$\Delta\Pi^F = \frac{1}{sF^x\sqrt{dt}} \qquad \text{Equation 2.2.43}$$

the system was able to execute a simple trading rule for long ("buy") and short ("sell") signals.

If $P^F > MP^F_0$ then signal = buy

If $P^F < -MP^F_0$ then signal = sell

The threshold parameter M was used to limit the number of transactions by defining a momentum uncertainty band around the null momentum value.

The system was tested with different sampling intervals Dt, data windows W and threshold parameters M. The s parameter was continually updated. The results presented in the paper included US$ 35 transaction costs for each buy and sell combination and show the trading profit for two specific days, June 20 and June 22, 1999. Depending on the parameter selection for Dt, W and M, the system generated a gain of up to US$ 2285 or loss of up to US$ 1125. It does not state how much money was available for investment in total. Nevertheless, these results show how mathematical methods usually used in physics can be successfully applied to the financial trading domain.

No doubt, these are not the last new prediction strategies as scientists "keep forging ahead with always more potent algorithms, which alone would allow them to remain ahead of the pack" as Lequarré predicts. However, this race will continue, as "patterns in the price tend to disappear as agents evolve profitable strategies to exploit them" [Lequarré 1993].