Applied Intelligence 6, 205–213 (1996)

Introduction to Financial Forecasting

YASER S. ABU-MOSTAFA
California Institute of Technology; and NeuroDollars, Inc.
yaser@cs.caltech.edu

AMIR F. ATIYA
Department of Computer Engineering, Faculty of Engineering, Cairo University, Giza, Egypt

Abstract. This paper provides a brief introduction to forecasting in financial markets with emphasis on commodity futures and foreign exchange. We describe the basic approaches to forecasting, and discuss the noisy nature of financial data. Using neural networks as a learning paradigm, we describe different techniques for choosing the inputs, outputs, and error function. We also describe the learning from hints technique that augments the standard learning from examples method. We demonstrate the use of hints in foreign-exchange trading of the U.S. Dollar versus the British Pound, the German Mark, the Japanese Yen, and the Swiss Franc, over a period of 32 months. The paper does not assume a background in financial markets.

Keywords: financial markets, trading, forecasting, learning, hints

1. Background

A major challenge confronting speculators, investors, and businesses is to accurately forecast price movements in financial and commodity markets. In their quest to forecast the markets, they assume that future occurrences are based at least in part on present and past events and data. However, financial time series are among the 'noisiest' and most difficult signals to forecast, as we will discuss in Section 2.

This has led many economists to adopt the "efficient market hypothesis" (see Malkiel [8] and Fama [3]), which states that price changes are independent of the past, and follow a random walk. According to this hypothesis, price changes are unpredictable, and forecasting a financial market is a hopeless effort. Any change in price represents the immediate reaction to an instantaneous news event or to new and unexpected changes in supply/demand figures. If there had been any expected profit opportunity, then investors would have immediately exploited the opportunity in a way that will drive the price back to the level where it is not profitable any more.

Although there has been a lot of debate about the efficient market hypothesis, it is hard either to prove it or disprove it. Our personal experience in the markets tells us that the financial markets are 'somewhat' predictable. The existence of so many price trends in financial markets (a trend is a sustained increase or decrease in price over a substantial period of time, for example for a few months), and the undiscounted serial correlations among fundamental events and economic figures affecting the markets, are two of many evidences against the efficient market hypothesis.

Speculators trade mainly two types of financial instruments: stocks and commodity futures. In this paper, we discuss the main concepts in terms of futures, though many of the same forecasting techniques apply also to stocks. A futures contract is a binding obligation to make or take delivery of a specific quantity of a particular commodity at a certain future date. The types of commodities covered by the futures contracts are agricultural, petroleum, precious metals, foreign currencies, stock market averages, and interest rate instruments. The contractual obligation can be satisfied also by making an offsetting sale or purchase of an

equivalent futures contract prior to the delivery date. Rather than trading the actual commodity, speculators can now trade the futures contract. They do not have to worry about delivery as long as they offset their positions and get out of the market before the delivery date. The price of the futures contract varies in proportion to the price of the underlying commodity, but also other factors such as time remaining till delivery, interest rate, and expectations of future supply have some influence.

To buy or sell a futures contract, a small deposit, about 5 to 10% of the value of the contract, is needed. Such a deposit, called the *margin*, makes futures trading highly rewarding, but also very risky. In case of a 5% margin, a 1% movement in the price of the original contract represents a 20% movement relative to the initial margin investment. Taking the right position in this case would result in a 'leveraged' return of 20%.

There are two types of futures traders: hedgers and speculators. A hedger is the producer or the user of the underlying commodity. The hedger buys or sells futures contracts to protect himself against adverse price movements. For example, a corn farmer can utilize the futures market to sell his crop while it is still in the ground, by selling futures contracts. By the time his crop is harvested the price might have fallen, and this way the farmer has protected himself against such a decline. Similarly, a food processing plant can buy corn futures contracts months before it needs the corn, to protect itself against possible price increases in the future.

The speculator, on the other hand, buys and sells futures contracts for the purpose of profit. He thereby has the vital roles of accepting the risks that commodity producers and users wish to avoid, and of providing the market liquidity needed by these hedgers. The methods used by speculators to forecast and trade futures can be grouped into two main categories: fundamental analysis and technical analysis (see Kaufman [6]).

Fundamental analysis is based upon the study of supply and demand. An increase in supply or a decrease in demand tends to depress the price of a commodity. Conversely, a decrease in supply or an increase in demand will raise the price. Fundamental analysts study the news events and the economic factors that influence supply and demand. For example, a drought usually results in an increase in the prices of agricultural commodities, because of a potentially low harvest. It is therefore an occasion to buy agricultural futures. An unexpected increase in the released monthly inflation number is an opportunity for the trader to sell Treasury Bond futures, because inflation will result in higher treasury bond yields, which always moves in opposite direction to the price of the bond.

Technical analysis, on the other hand, is based on analyzing price patterns, as well as trading volume and open interest figures (the trading volume represents the number of contracts traded during the day, and the open interest is the number of open contracts, for a particular commodity). Technical analysts do not utilize any external economic data or any relevant news events. They assume that these factors get reflected in the past price pattern, and are therefore utilized in an indirect way. They believe that the recent price pattern to some extent can determine future price direction. Therefore, they study historical data to find correlations between certain patterns and subsequent market direction. Such pattern formations can be observed by looking at the price charts and performing a 'mental pattern recognition'. This approach is called chart analysis. It is largely subjective, and does not lend itself that easily to computerization.

There are, however, more quantitative approaches in technical analysis. The main one is the trend following approach, which includes the moving averages technique and the regression analysis. The moving averages technique is one of the most widely used technical indicators. It signals an uptrend if the short term average of the price signal is higher than a longer term average, otherwise it signals a downtrend. The sizes of the averaging windows are usually optimized on the data. They determine the size of the trends to be detected: whether they are long and slow trends, small trends, or even small cycles. In regression analysis, a line is fit to the most recent price data points. The slope of the line determines whether the market is in an uptrend or downtrend (see [6] and [10]).

One disadvantage of such trend-following methods is that they indicate a new trend only some time after a trend has started, thus missing the first part of it. They never anticipate, they only react. Such systems are called *reactive*. On the other hand, systems which indicate a trend or a price move before it starts are called *predictive* (some of the chart analysis systems are predictive). Predictive systems are though much harder to design, and usually result in many false alarms.

Another approach to financial forecasting is the autoregressive integrated moving average method (ARIMA) [10]. Here the time series is modeled as a linear system. The parameters of the system are estimated using the historical data. Once the parameters are

estimated, the estimated model is run and the forecast is obtained. Of course the model parameters should be updated daily to account for the new data and adapt to changes in market conditions.

Other quantitative approaches include neural networks and other learning systems, which have found extensive use in the last few years (see for example the papers by Refenes and Azema-Barac [12], Hoptroff [4], White [16], Shoenenburg [14], Hsu et al. [5], and the book by Turban and Trippi [15], the proceedings of the NNCM conference [11] as a representative sample for work done in this direction). Forecasters use a learning procedure to associate market direction with past price patterns and other technical and fundamental inputs, thus discovering the underlying relationships. In reality, however, the problem is not straightforward as it seems. Because of the huge amount of noise and price pattern variety, techniques are needed to guide the learning process, and to suitably preprocess the system inputs and outputs. These issues will be the main focus of this paper.

The rest of the paper is organized as follows. Section 2 formalizes the notion of 'very noisy data' and discusses why learning methods, such as those used in neural networks, have difficulty working with financial data. In Section 3, we describe how the inputs and outputs are selected for a neural network that forecasts the market. Although we speak in terms of neural networks, most of the techniques apply equally well to other learning models. In Section 4, we describe the hints method that helps the learning process, and report experimental results of using the method in the foreign-exchange markets.

2. Financial Data

This section provides a characterization of very noisy data that applies to the financial markets. Consider the market as a system that takes in a lot of information (fundamentals, news events, rumors, who bought what when, etc.) and produces an output $\hat{y}$ (say up/down price movement for simplicity). A model, e.g., a neural network, attempts to simulate the market (Fig. 1), but it takes an input x which is only a small subset of the information. The 'other information' cannot be modeled and plays the role of noise as far as x is concerned. The network cannot determine the target output $\hat{y}$ based on x alone, so it approximates it with its output y. It is typical that this approximation will be correct only slightly more than half the time.

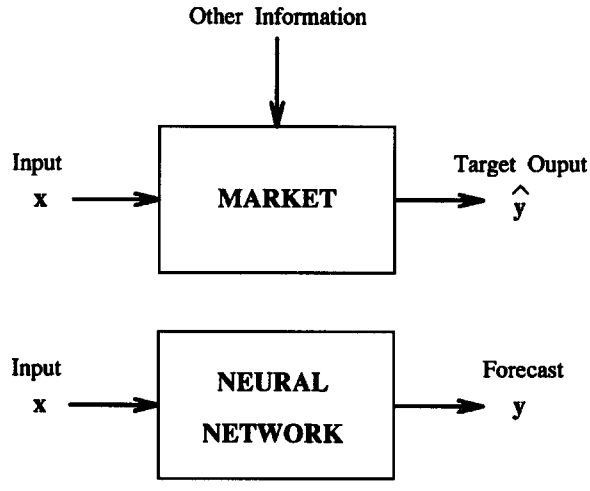

Figure 1. Illustration of the nature of noise in financial markets.

What makes us consider x 'very noisy' is that y and $\hat{y}$ agree only $\frac{1}{2}+\epsilon$ of the time (50% performance range). This is in contrast to the typical pattern recognition application, such as optical character recognition, where y and $\hat{y}$ agree $1-\epsilon$ of the time (100% performance range). It is not the poor performance *per se* that poses a problem in the 50% range, but rather the additional difficulty of learning in this range. Here is why.

During learning, we use a limited set of N examples to train (and validate) a neural network to forecast the market. In the 50% performance range, we want y to agree with $\hat{y}$ on $(\frac{1}{2}+\epsilon)N$ examples. The number of functions on N points that do that is huge. Too many random functions look like good candidates based on the limited set of examples. This is in contrast to the 100% performance range where the functions need to agree with $\hat{y}$ on $(1-\epsilon)N$ examples. The number of functions that do that is limited. Therefore, one can have much more confidence in a function that was learned in the 100% range than one learned in the 50% range. It is not uncommon to see a random trading policy making good money for a few months, but it is very unlikely that a random character recognition system will read a paragraph correctly.

Of course this problem would diminish if we used a very large set of examples, because the law of large numbers would make it less and less likely that y and $\hat{y}$ can agree $\frac{1}{2}+\epsilon$ of the time just by 'coincidence'. However, financial data has the other problem of non-stationarity. Because of the continuous evolution in the markets, old data may represent patterns of behavior that no longer hold. Thus, the relevant data for training purposes is limited to fairly recent times. Put together,

noise and non-stationarity mean that the training data will not contain enough information for the network to learn the function. More information is needed, and the hints method described in Section 4 can be the means of providing it.

3. Inputs and Outputs

In this section, we describe specific ways for choosing the inputs and outputs in financial forecasting, using neural networks as a paradigm. There are two major approaches for choosing the inputs. In the first approach the past price pattern is given as input to the neural network. By training the neural network, it learns to correlate price patterns to future price movements. In the other approach, technical and/or fundamental indicators are given to the neural network as inputs. The technical indicators are obtained by processing the raw price, volume and open interest data, for the purpose of making it easier for the neural network to perform its job. Examples of technical indicators are:

1) *The Moving Averages Indicator*: It is defined by

$$u(t) = \frac{1}{T_1}\sum_{i=t-T_1+1}^{t} x(i) - \frac{1}{T_2}\sum_{i=t-T_2+1}^{t} x(i),$$

where $x(t)$ is the price at day t, and T_1 and T_2 ($T_1 < T_2$, typically T_1 around $\frac{1}{2}T_2$) are the sizes of the averaging windows. The sign of $u(t)$ determines whether it is in an upward or a downward trend. Other averaging windows such as exponential or triangular can also be used. Since the sizes of the averaging windows determine the size and the time scale of the trends detected, different combinations of T_1 and T_2 can be used to produce several inputs to the neural network. Each input will have different and useful information, which the network can utilize.

2) *Trading Volume and Open Interest Indicators*: Volume and open interest numbers can be collected on a daily basis. They are useful signals and give some clues to future market behavior. The trading volume measures the intensity of the price move. Heavy volume often indicates the beginning of a major price move or a trend. The change in open interest represents the number of new contracts opened. At the beginning of a trend, the open interest increases considerably, reflecting new buying or short selling by traders entering the market. Close to the end of the trend, the rate of change of open interest becomes small or negative, indicating that traders are liquidating their positions and getting out of the market.

3) *Volatility*: This is a measure of the intensity of market activity. Large price swings mean high volatility, and a nearly constant uneventful market means low volatility. The following formulas are measures for the short term average of the volatility (see also [6]).

$$V_1(t) = \sqrt{\frac{1}{T}\sum_{i=t-T+1}^{t}(x(i) - \bar{x}(t))^2},$$

$$\bar{x}(t) = \frac{1}{T}\sum_{i=t-T+1}^{t} x(i),$$

$$V_2(t) = \frac{1}{T}\sum_{i=t-T+1}^{t}(H(i) - L(i)),$$

$$V_3(t) = \max_{t-T+1\le i\le t} x(i) - \min_{t-T+1\le i\le t} x(i),$$

where T is the size of the averaging window, $H(t)$ and $L(t)$ are respectively the high and the low of the prices on day t (they can be obtained on a daily basis for futures markets). Both the volatility and its rate of change are valuable inputs to the neural network. A sudden increase in volatility is an indication of an impending major move. It can signal the beginning of a trend, an end or a reversal of a trend, or possibly even a price crash.

Fundamental indicators can also be effective inputs to neural networks. Several such systems have been developed (see Kimoto et al. [7], Mendelsohn and Stein [9]). Other examples of fundamental indicators which can be used are:

1) The rate of change of inflation, the Federal Reserve Bank's federal funds rate, and the budget deficit are important indicators which affect the interest rate and hence the price for the Treasury Bonds. Increasing inflation results in buyers of Treasury Bonds demanding higher interest rates. The federal funds rate is the interest rate specified by the Federal Reserve Bank on overnight loans among banks. It directly affects short term interest rates, and hence usually the long term (or Treasury Bond's) interest rates. A higher budget deficit means a higher supply of Treasury Bonds, and hence lower prices for the Bonds.

2) The temperature, rainfall relative to the averages of the day in major production areas, and the month of the year, are important indicators for agricultural commodities. Big changes in weather can cause for example droughts, floods, delayed planting due to a late winter, thus affecting harvest, and driving prices up. The month of the year is a valuable indicator, since most agricultural commodities exhibit cyclical price patterns. Prices usually reach lows at the harvest period, and highs during the early growing season.

When applying this approach, one has to be careful that often the market focuses on different fundamental indicators at different times. For example, short term interest rate differentials can be at times the main movers of currency exchange rates, whereas at other times it could be the budget deficit. The system must therefore be adaptive, to track these structural changes in input/output relationship (see also [12]).

Another issue in the design of the neural network of almost equal importance as the choice of inputs, is the choice of output targets and error function. The neural network output should determine next day's (or next week's or next month's) price move. The targets $d(t)$ are chosen usually as one of the following:

$$d_1(t) = \left[\left(\frac{1}{M}\sum_{i=1}^{M} x(t+i)\right) - x(t)\right] \Big/ \Delta,$$

$$d_2(t) = \text{sign}\left[\left(\frac{1}{M}\sum_{i=1}^{M} x(t+i)\right) - x(t)\right],$$

where Δ is a positive normalizing factor, to get $d_1(t)$ to be in the range from -1 to 1. The term in parentheses ( ) represents the average price in the next M days. A typical choice, $M = 1$, is suitable for short-term trading. In the first approach $d_1(t)$ is 'graded', and attempts to train the neural network to forecast the degree of change. The second choice $d_2(t)$ cares more about direction, which is the goal in speculative trading.

As for the error function, many improvisations can be made to enhance performance. Other than the standard sum of square error function, one of the popular functions is

$$E = -\sum_{t=1}^{T} d_1(t)y(t).$$

For $M = 1$, this function corresponds to the total return (without the transaction cost) if the position on day t is $y(t)$.

4. The Hints Method

Learning from hints [1] is a value-added feature to the usual learning from examples method that boosts the information content in the data. The method allows us to use prior knowledge about the target function, that comes from common sense or expertise, along with the training data in the *same* learning process. Different types of hints that may be available in a given application can be used simultaneously. In this section, we give experimental evidence of the impact of hints on learning performance, and explain the method in some detail to enable the readers to try their own hints in different markets.

As far as our method is concerned, a hint is any property that the target function is known to have. For instance, consider the symmetry hint in (foreign exchange) FX markets as it applies to the U.S. Dollar versus the German Mark (Fig. 2). This simple hint asserts that if a pattern in the price history implies a certain move in the market, then this implication holds whether you are looking at the market from the U.S.

Figure 2. Illustration of the symmetry hint in FX markets.

Dollar viewpoint or the German Mark viewpoint. Formally, in terms of normalized prices, the hint translates to invariance under inversion of these prices.

Is the symmetry hint valid? The ultimate test for this is how the learning performance is affected by the introduction of the hint. The formulation of hints is an art. We use our experience, common sense, and analysis of the market to come up with a list of what we believe to be valid properties of this market. We then represent these hints in a canonical form as we will see shortly, and proceed to incorporate them in the learning process. The improvement in performance will only be as good as the hints we put in.

The canonical representation of hints is a more systematic task. The first step in representing a hint is to choose a way of generating 'examples' of the hint. For illustration, suppose that the hint asserts that the target function $\hat{y}$ is an odd function of the input. An example of this hint would have the form $\hat{y}(-x) = -\hat{y}(x)$ for a particular input x. One can generate as many examples as needed by picking different inputs.

After a hint is represented by examples, it is ready to be incorporated in the learning process along with the examples of the target function itself. Notice that an example of the function is learned by minimizing an error measure, say $(y(x) - \hat{y}(x))^2$, as a way of ultimately enforcing the condition $y(x) = \hat{y}(x)$. In the same way, an example of the oddness hint can be learned by minimizing $(y(x) + y(-x))^2$ as a way of ultimately enforcing the condition $y(-x) = -y(x)$. This involves inputting both x and $-x$ to the network and minimizing the difference between the two outputs. It is easy to show that this can be done using backpropagation [13] twice.

The generation of an example of the hint does not require knowing the value of the target function; neither $\hat{y}(x)$ nor $\hat{y}(-x)$ is needed to compute the error for the oddness hint. In fact, x and $-x$ can be artificial inputs. The fact that we do not need the value of the target function is crucial, since it was the limited resource of examples for which we know the value of the target function that got us interested in hints in the first place. On the other hand, for some hints, we can take the examples of the target function that we have, and employ the hint to duplicate these examples. For instance, an example $\hat{y}(x) = 1$ can be used to infer a second example $\hat{y}(-x) = -1$ using the oddness hint. Representing the hint by duplicating the examples of the function is an easy way to try simple hints using the same software that we use for learning from examples.

Even simple hints can result in significant improvement in the learning performance. Figure 3 shows

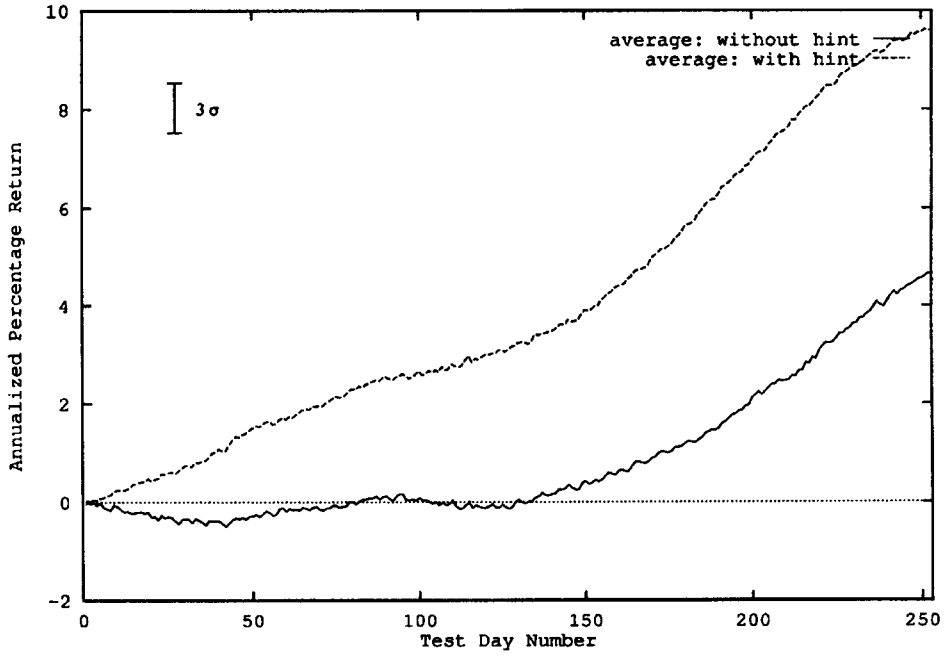

Figure 3. Learning performance with and without hint.

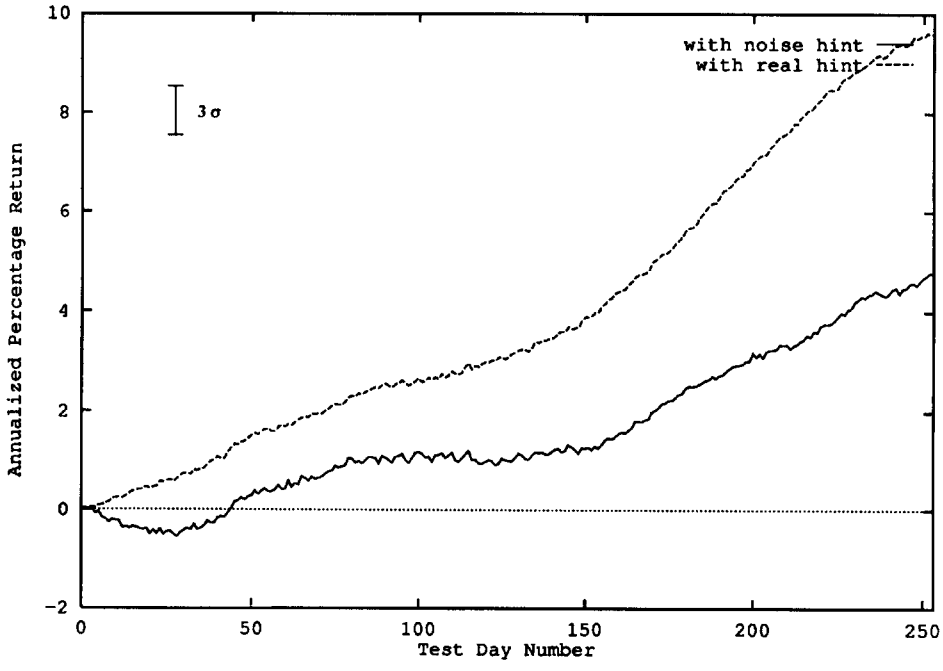

Figure 4. Performance of the real hint versus a noise hint.

the learning performance for foreign exchange (FX) trading with and without the symmetry hint (see Fig. 2), using only the closing price history. The plots are the annualized percentage returns (cumulative daily, unleveraged, transaction cost included), for a sliding one-year test window in the period from April 1988 to November 1990, averaged over the four major FX markets with more than 150 runs per currency. The error bar in the upper left corner is 3 standard deviations long (based on 253 trading days, assuming independence between different runs). The plots establish a statistically significant differential in performance due to the use of hints.

This differential holds for all four currencies; the British Pound, the German Mark, the Japanese Yen, and the Swiss Franc. In each case, only the closing prices for the preceding 21 days were used for inputs. The objective (fitness) function we chose was the total return on the training set, and we used simple filtering methods on the inputs and outputs of the networks. In each run, the training set consisted of 500 days, and the test was done on the following 253 days. All four currencies show an improved performance when the symmetry hint is used.

We ran two additional experiments to verify that the information content of the hint is the reason behind the improved performance. Since the hint plays an incidental role as a constraint on the neural network during learning, it can help the performance merely as a regularizer [2]. The additional experiments were designed to isolate the informative role from the regularizing role of the symmetry hint, we ran two experiments.

In the first experiment, we used an uninformative hint, or 'noise' hint, which provides a random target output for the same inputs used in the examples of the symmetry hint. Figure 4 contrasts the performance of the noise hint with that of the real symmetry hint, averaged over the four currencies. Notice that the performance with the noise hint is close to that without any hint (Fig. 3), which is consistent with the notion of uninformative hint. The regularization effect seems to be negligible.

In the second experiment, we used a harmful hint, or 'false' hint, in place of the symmetry hint. The hint takes the same examples used in the symmetry hint and asserts antisymmetry instead. Figure 5 contrasts the performance of the false hint with that of the real symmetry hint. As we see, the false hint had a detrimental effect on the performance. This is consistent with the hypothesis that the symmetry hint is valid, since its negation results in worse performance than no hint at all. Notice that the transaction cost is taken into

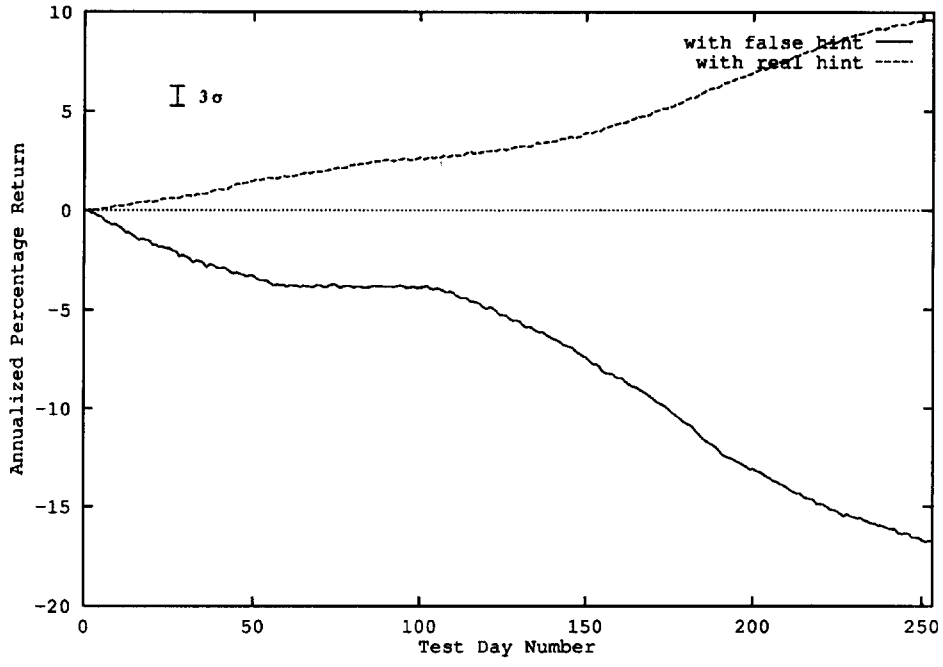

Figure 5. Performance of the real hint versus a false hint.

consideration in all of these plots, which works as a negative bias and amplifies the losses of bad trading policies.

References

1. Y. Abu-Mostafa, "Learning from hints," *Journal of Complexity*, Academic Press, vol. 10, pp. 165–178, 1994.
2. H. Akaike, "Fitting autoregressive models for prediction," *Ann. Inst. Statist. Math.*, vol. 21, pp. 243–247, 1969.
3. E. Fama, "Efficient capital markets: II," *Journal of Finance*, vol. 46, no. 5, pp. 1575–1618, 1991.
4. A. Hoptroff, "The principles and practice of time series forecasting and business modeling using neural nets," *Neural Computing and Applications*, vol. 1, no. 1, pp. 59–66, 1993.
5. W. Hsu, L. Hsu, and M. Tenorio, "Feature subset selection with application to financial prediction tasks," in *Financial Applications and Neural Networks*, edited by A. Refenes, Cambridge Press, 1993.
6. P. Kaufman, *The New Commodity Trading Systems and Methods*, John Wiley & Sons, 1987.
7. T. Kimoto, K. Asakawa, M. Yoda, and M. Takeoka, "Stock market prediction system with modular neural networks," *Proc. International Joint Conference on Neural Networks*, San Diego, CA, 1990.
8. B. Malkiel, *A Random Walk Down Wall Street*, W.W. Norton & Co.: New York, 1985.
9. L. Mendelsohn and J. Stein, "Fundamental analysis meets the neural network," *Futures*, September 1991.
10. D. Montgomery, L. Johnson, and J. Gardiner, *Forecasting and Time Series Analysis*, McGraw-Hill, Inc., 1990.
11. A. Refenes (Ed.), *Proc. of the First International Workshop on Neural Networks in Capital Markets*, London, U.K., November 1993.
12. A. Refenes and M. Azema-Barac, "Neural network applications in financial asset management," *Neural Computing and Applications*, vol. 2, pp. 13–39, 1994.
13. D. Rumelhart, G. Hinton, and R. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, MIT Press, vol. 1, pp. 318–362, 1986.
14. E. Schoenenburg, "Stock price prediction using neural networks: A project report," *Neurocomputing*, vol. 2, pp. 17–27, 1990.
15. E. Turban and R. Trippi, *Neural Network Applications in Investment and Financial Services*, Probus Publishing, 1992.
16. H. White, "Economic prediction using neural networks: The case of IBM daily returns," *Proc. IEEE Internat. Conf. on Neural Networks*, San Diego, CA, 1988, vol. 2, pp. 451–458.

Yaser S. Abu-Mostafa is Professor of Electrical Engineering and Computer Science at the California Institute of Technology, and Chairman of NeuroDollars.

He received the B.Sc. in Electrical Engineering from Cairo University in 1979, the M.S.E.E. from the Georgia Institute of Technology in 1981, and the Ph.D. in Electrical Engineering and

Computer Science from Caltech in 1983. He joined Caltech as Assistant Professor the same year.

Dr. Abu-Mostafa heads the Learning Systems Group at Caltech. His research focuses on the theory, algorithms, and applications of automated learning. He introduced the 'learning from hints' technique in 1989. He has more than 60 technical publications in the areas of learning theory, neural networks, pattern recognition, information theory, and computational complexity, including two articles in Scientific American.

Dr. Abu-Mostafa received the Clauser Prize for the most original doctoral thesis at Caltech, and numerous teaching awards including the 1996 Feynman Award. He was the founding Program Chairman of NIPS (Neural Information Processing Systems conference) in 1987, and a founding member of the IEEE Neural Networks Council. He has been a technical consultant for Citibank since 1988, and he serves on the boards of several technical journals and conferences.

Amir Atiya was born in Cairo, Egypt, on March 20, 1960. He received the B.S. degree from Cairo University, Cairo, Egypt, in 1982, and the M.S. and Ph.D. degrees in 1986 and 1991 from Caltech, Pasadena, CA, all in Electrical Engineering. From 1985 to 1990 he was a Teaching and Research Assistant at Caltech. From September 1990 to July 1991 he held a Research Associate position at Texas A&M University. From July 1991 to February 1993 he was a Senior Research Scientist at QANTXX Corporation, Houston, TX. Since then, he has been an Assistant Professor at the Department of Computer Engineering, Cairo University. He had summer visiting positions at Caltech in 1994 and 1995. His research interests are neural networks and financial forecasting. He is the recipient of the Egyptian State Award for Research in Science and Engineering in 1994. He is on the organizing committe of the Neural Networks in the Capital Markets Conference.