8

Testing the Efficient Markets Hypothesis with Gradient Descent Algorithms

George Tsibouris and Matthew Zeidenberg,
University of Wisconsin, USA

INTRODUCTION

Accurate modelling of economic phenomena has always been one of the main concerns of economists. Most attempts to understand the economy have been either model-driven, statistical, or both. A theory based on optimising behaviour is formulated and a testable hypothesis derived from the theoretical model is checked against economic data. Statistical tests enable economists to reject incorrect hypotheses to the desired degree of confidence. Specification of financial models has traditionally been done in partial equilibrium. Both capital asset pricing models (CAPM) and arbitrage pricing theory (APT) derived models attempt to describe asset price behaviour based on optimisation in partial equilibrium. Such models have been very useful in expanding our understanding of stock price behaviour. Nevertheless, many empirical financial anomalies have remained unexplainable. It is possible that this may be due to the partial equilibrium nature of these models. Attempting to model financial markets in a general equilibrium framework still remains analytically intractable.

Stock prices, because of their highly chaotic nature, are notoriously difficult to model with standard methods such as least-squares regression. In recent years, there has been great interest within the scientific community in discerning patterns in such highly chaotic data, since such data are present not only in economic time

Neural Networks in the Capital Markets. Edited by Apostolos-Paul Refenes.

series but also in the behaviour of other dynamical systems such as weather patterns and ecological models. Models that reveal systematic trends in economic data may be generalized to other domains.

Because of their inductive nature, neural networks, like other statistical models, can bypass the step of theory formulation altogether, or can be combined with prior knowledge of the characteristics of the system. They can infer complex nonlinear relationships between variables that are given as their input and variables that are given as their output. There are four types of neural network that suggest themselves for the modelling of time-series data: feedforward backpropagation networks; recurrent backpropagation networks; cascade-correlation networks; and networks based on the temporal difference method of Sutton. We explore the use of feedforward backpropagation networks, temporal difference models, and cascade-correlation networks in the case of stock-market time-series data.

Neural network models are fitted to daily stock prices. The estimated parameters are used to obtain one-step-ahead forecasts. Learning achieved by this system is compared to traditional statistical techniques.

The next section sets out the econometric methodology that is commonly used. Then the estimation results are presented and discussed, and possible directions for further study outlined.

ECONOMETRIC METHODOLOGY

The efficient markets hypothesis (EMH) has found broad acceptance in the financial community. In its weak form, the EMH asserts that an asset price is a reflection of all the information that can be gleaned from past price behaviour. This implies that the movement of an asset's price is completely unpredictable given its history. Thus, technical analysis cannot provide a financially profitable investment strategy. The intuitive reason behind the unpredictability of asset price returns is similar to the reason why one rarely finds money lying on the ground. Nobody disputes the fact that money does indeed fall out of people's pockets but, when it does, it tends to be picked up fairly quickly. Profit opportunities in financial markets are exploited at their inception.

Traditional tests of the EMH have involved estimating a linear autoregressive model of asset returns and testing the joint null hypothesis that the coefficients of the lagged returns are equal to zero. Nevertheless, the inability to refute the null hypothesis does not imply an affirmation of the EMH. It simply may be that eventual underlying nonlinear patterns are undetectable using linear methods.

Neural networks are a rich class of functional forms that are capable of approximating arbitrary Borel measurable functions [HekrPa91]. This property allows us to exploit nonparametric estimation methods based on these neural networks. These models can be viewed as an input–output system of a particular form. A general neural network describes a mapping between a vector of inputs and a vector of outputs. The simplest model is constructed in the following form: define a $1 \times r$

vector of inputs $\mathbf{X}$ and a $1 \times p$ vector of outputs $\mathbf{Y}$. Their interaction can be described as:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta}$$

where $\boldsymbol{\beta}$ is the connection strength matrix of size $r \times p$. If we assume that one of the input neurons exhibits a value of one at all times, we then have the multivariate linear statistical model with an intercept term. In theory, each of the $\mathbf{X}$ inputs could be processed in parallel, which is one of the main analogies between this model and the brain.

The family of models described above is only capable of approximating quasi-linear models. Even with a slightly more sophisticated underlying structure (say, quadratic terms), it performs poorly. This led to the development of more complex networks by applying further analogies with neural function in the cerebral cortex. Intermediate processing layers were introduced between the input and output. These layers are called 'hidden' since it is not possible to obtain empirical observations on them. The output of this network model can be described as follows:

$$Y_k = \psi_k \left(\sum_{j=1}^{q} \alpha_j \left(X\gamma_j \right) \beta_{kj} \right) \equiv f_k(X, \theta),$$

where $\boldsymbol{\beta}' = \left(\beta_1, \ldots, \beta_p\right)$ and $\boldsymbol{\gamma}' = \left(\gamma_1, \ldots, \gamma_q\right)$ are the connection strengths, ψ_k and α_j are known functions, and $\theta' = (\boldsymbol{\beta}', \boldsymbol{\gamma}')$.

In backpropagation, a member of the family of smooth and monotonically increasing functions, for example the sigmoid function

$$\alpha_j \left(\mathbf{X}\gamma_j \right) = \frac{1}{1 + e^{-\mathbf{X}\gamma_j}}$$

is used as the activation function α_j. Without loss of generality and for computational simplicity we define the output processing function ψ_k to be the identity function.

Alternative choices of activation function are possible. One example that time-series econometricians are familiar with is letting α_j equal the sine function. We therefore get the spectral analysis model where a linear combination of sine functions at different frequencies can approximate any function arbitrarily closely.

The backpropagation method is implemented on θ using recursive m-estimation, namely:

$$\theta_{t+1} = \theta_t - \eta_t \nabla_\theta f_k(\mathbf{X}, \theta)' (\mathbf{Y} - f_k(\mathbf{X}, \theta))$$

where η_t is the learning rate, ∇_θ is the gradient and θ_0 is an arbitrary starting value.

EMPIRICAL RESULTS

For our estimation, we selected six stock price data sets from different industries in an attempt to avoid sector-specific idiosyncrasies. They are the stock prices of

Citicorp (CCI), John Deere (DE), Ford (F), General Mills (GIS), GTE, and Xerox (XRX). We use daily temporal aggregation. The observations span the period from 4 January 1988 to 31 December 1990. The total number of daily observations is 758. All our data are from the Center for Research in Security Prices (CRSP).

We divided each data set into two portions. The first portion, which, following general practice, we call the training set, is used to construct the network model and spans the period from 4 January 1988 to 29 December 1989. The second portion is used to test the validity of the model on data not used to produce the model, that is, what is commonly referred to as 'generalisation' performance. The dates for this portion span the period from 2 January 1990 to 31 December 1990.

Table 8.1 presents summary statistics of the data. It shows the division of each data set into two portions, with dates, sample means and standard deviations for the training and test sets in each of the six cases. The variables in Table 8.1 are one-day returns for holding the stock in question. The closing values are used to represent the daily price.

The average daily return of the six stocks, CCI, DE, F, GIS, GTE, and XRX for the training period is 0.0010% and for the test period is −0.0009%. The most volatile stock during the training period was CCI, with a standard deviation of 0.0202%. The most volatile stock during the test period was also CCI, with a standard deviation of 0.0267%. The least volatile stock, for both the training and test periods, was GIS, with standard deviations of 0.0127% and 0.0158%, respectively.

We implemented the backpropagation algorithm using the signs of stock returns as inputs and outputs. A similar technique was used in a previous paper by the authors using different stocks and a different sample period. For each of the six data sets CCI, DE, F, GIS, GTE, and XRX, we used a network of nine input units, five hidden units and one output unit. This represents a model of $(9 \times 5) + (5 \times 1) = 50$ weights or parameters. Backpropagation is more parsimonious in number of parameters than the equivalent linear polynomial expansion with $2^9 - 1 = 511$ parameters. As a side note, an alternative specification using signed magnitudes as

Table 8.1 Summary statistics of the data set.

Variable	Sample	Start date	End date	Mean	Std. Dev.
CCI	Train	4/1/88	29/12/89	0.0013	0.0202
	Test	2/1/90	31/12/90	−0.0026	0.0267
DE	Train	4/1/88	29/12/89	0.0013	0.0158
	Test	2/1/90	31/12/90	−0.0008	0.0190
F	Train	4/1/88	29/12/89	0.0006	0.0150
	Test	2/1/90	31/12/90	−0.0015	0.0169
GIS	Train	4/1/88	29/12/89	0.0010	0.0127
	Test	2/1/90	31/12/90	0.0014	0.0158
GTE	Train	4/1/88	29/12/89	0.0017	0.0135
	Test	2/1/90	31/12/90	−0.0004	0.0168
XRX	Train	4/1/88	29/12/89	0.0003	0.0135
	Test	2/1/90	31/12/90	−0.0015	0.0167

inputs and signs and magnitudes as two separate outputs was attempted. It did not perform very well.

As an illustrative example, to predict the value of CCI at period (day) t (corresponding to the output unit), we used the signs of the returns for the past five periods $t-1, \ldots, t-5$, as well as the return between t and $t-10$ (two weeks), $t-22$ (one month), $t-132$ (about six months), and $t-264$ (about one year), for a total of nine inputs. Positive asset returns are coded as a 1, while negative asset returns are coded as a 0. We have experimented with varying the number of hidden units and found five to be an adequate compromise. General experience has led researchers to choose a number of hidden units less than or equal to the number of inputs.

We performed backpropagation learning on the training set for 200 iterations or epochs, where an iteration represents one complete pass through the input data, altering the weights in the network. We recorded the total error sum of squares (TSS) of all the training patterns. We considered, as is general practice, the output unit to be predicting a positive sign for the subsequent return if it (a sigmoid unit) output above 0.5, and negative below 0.5. The performance is the percentage of the time that the predicted output agrees with the true output. We also calculated the RMS error, the number correctly predicted, and the average output of the network for each set. Table 8.2 presents the results of the neural network estimations on the financial data series. The TSS error and the RMS error are reported for all the asset price measures for both the training set and the test set. The average direction of asset price change, which corresponds to the average value of the output, is also presented. For the daily asset prices, the average output in the training set equals 0.4412 and in the test set equals 0.4148. This means that the neural network model predicts an upward movement approximately 44.12% of the time and a downward movement the remainder of the time (in the training set). This is broadly consistent with one of the implications of the random-walk hypothesis in which upward asset price movements should occur just about as frequently as downward movements.

The lower portion of Table 8.2 presents the actual performance statistics of the backpropagation algorithm in- and out-of-sample. The number of correct signs of the asset price returns coming from the model is compared to the actual data series. The percentage of correct signs is calculated for both the training and test runs. Standard deviations are also calculated for the percentage of correct signs under the null hypothesis of the model actually having a 0.5 probability of estimating the direction of change correctly. The probability of a type I error, namely the probability that we would get the percentage of correct signs that we report given that the model was providing absolutely no explanatory power, is also presented.

The percentage of correct responses ranges from 48.22% for the DE test set to 60.87% for the CCI test set. The average percentage of correct responses for the training sets was 60.20% and the average percentage for the test sets was 55.01%. As is usually the case in many such experiments of this type, the performance on the training set (after training) usually exceeds that on the test set. This is

Table 8.2 Empirical results for backpropagation.

Variable	Set	Sample size	TSS	RMS error	Avg. est.	No. correct	% correct	Std. Dev.	Probability of type I error
CCI	Train	505	65.6505	0.3606	0.2358	327	64.75%	6.6304	0
	Test	253	39.2403	0.3938	0.2456	154	60.87%	3.4578	0
DE	Train	505	60.5288	0.3462	0.6953	265	52.48%	1.1125	0.1335
	Test	253	47.6768	0.4341	0.7237	122	48.22%	−0.5658	0.7123
F	Train	505	67.1428	0.3646	0.3404	304	60.20%	4.5834	0
	Test	253	34.6182	0.3699	0.3356	152	60.08%	3.2063	0
GIS	Train	505	69.0926	0.3699	0.2263	307	60.79%	4.8504	0
	Test	253	41.7974	0.4065	0.2229	135	53.36%	1.0688	0.1446
GTE	Train	505	65.7384	0.3608	0.4492	313	61.98%	5.3844	0
	Test	253	35.8668	0.3765	0.4666	135	53.36%	1.0688	0.1446
XRX	Train	505	67.5382	0.3657	0.5203	308	60.99%	4.9394	0
	Test	253	44.1531	0.4178	0.4904	137	54.15%	1.3203	0.0934

because the training process tends to fit characteristics of the training set that are not generalisable (so-called 'overfitting'), that is, it fits 'noise' in the training set as well as signal. Therefore, it is not desirable to train the training set beyond a certain number of iterations, since in the early stages of learning it learns the 'first-order' characteristics of the training data, which are more likely to correspond to the true model. In subsequent iterations, the model learns 'second-order' characteristics which vary from data set to data set and would prove to be of no help in attempts to generalise.

The estimated percentages of correct responses for the training sets are all more than two standard deviations from the value maintained under the null hypothesis, with the exception of DE. The standard deviations range from 1.1125 for DE to 6.6304 for CCI. These values correspond to very low probabilities of type I errors in the training set, with the highest value for all stock measures being 13.35% for DE (all the rest are virtually zero).

In the test sets, the estimated percentages of correct responses range from −0.5658 to 3.4578 standard deviations from the null of 0.50. The probability of a type I error in the test set ranges from a high value of 71.23% for DE to 0% for CCI and F.

These results provide evidence that backpropagation models are able, using previous asset returns, to provide some predictive power for future returns. This result is evidence against the weak form of the EMH. With respect to DE, GIS, GTE and XRX, the test sets did not provide evidence of prediction abilities. This result is in accordance with the result of White [White88] who was unsuccessful in forecasting IBM daily stock prices using neural networks.

Viewed jointly, all six asset prices give a chi-square statistic of 13.21, with five degrees of freedom. This statistic is significant at the 2.5% level. This represents a reasonable confidence level that our technique is doing significantly better than chance overall.

We also ran the learning algorithm for 850 iterations, after which there was no appreciable decrease in the TSS error, signifying that the network had reached a minimum in error space. The gradient descent algorithm does not ensure a global minimum. Nevertheless, multiple runs from different randomly selected initial parameter values did converge to the same minimum. This would suggest either a global minimum or a local minimum with a large basin of attraction. Simulated annealing on the learning rate has been shown to ensure convergence to the global minimum. Nevertheless, this technique is very costly in terms of computing time. This documented the overfitting effect, since performance on the test sets was uniformly better (in one case, the same) in the case of 200 iterations of training than in the case of 850. Also, performance on the training sets was uniformly worse in the case of 200 iterations than in the case of 850, as would be expected given the hypothesis of overfitting.

We also ran the TD(0) algorithm for the same training and test sets and for the same stocks. The results are shown in Table 8.3. The percentage of correct

Table 8.3 Empirical results for TD(0).

Variable	Set	Sample size	TSS	RMSE	Avg. est.	No. correct	% correct	Std. Dev.	Probability of type I error
CCI	Train	505	63.2321	0.3539	0.3870	280	55.45%	2.4475	0.0073
	Test	253	30.3577	0.3464	0.3870	152	60.08%	3.2063	0.0007
DE	Train	505	71.2646	0.3757	0.3430	240	47.52%	−1.1125	0.5438
	Test	253	34.0338	0.3668	0.3430	131	51.78%	0.5658	0.2877
F	Train	505	62.8760	0.3529	0.4510	270	53.47%	1.5575	0.0606
	Test	253	30.8235	0.3490	0.4510	149	58.89%	2.8291	0.0023
GIS	Train	505	75.9399	0.3878	0.2400	269	53.27%	1.4685	0.0722
	Test	253	37.4637	0.3848	0.2400	137	54.15%	1.3203	0.0934
GTE	Train	505	69.3214	0.3705	0.3070	268	53.07%	1.3795	0.0853
	Test	253	34.1236	0.3673	0.3070	138	54.55%	1.4460	0.0749
XRX	Train	505	82.5044	0.4042	0.1750	275	54.46%	2.0025	0.0228
	Test	253	36.7052	0.3809	0.1750	152	60.08%	3.2063	0.0007

responses ranged from 47.52% for the DE training set to 60.08% for the CCI and XRX test sets. The average percentage of correct responses for the training sets was 52.87% and the average percentage for the test sets was 56.59%. This is a bit unusual in that performance on the test sets consistently exceeded that of the training sets. Sutton claims that the TD(0) algorithm is particularly adept at generalisation.

There were only two estimated percentages of correct responses for the training sets that are more than two standard deviations from the value maintained under the null hypothesis, namely CCI and XRX. The standard deviations ranged from −1.1125 for DE to 3.4475 for CCI.

In the test sets, the estimated percentages of correct responses range from 0.5658 to 3.2063 standard deviations from the null of 0.50. The probability of a type I error in the test set ranges from a high value of 28.77% for DE to 0.0007% for CCI and XRX.

Viewed jointly, all six asset prices gave a chi-square statistic of 16.35, with five degrees of freedom. This statistic was significant at the 1% level. This is comparable to the result achieved with backpropagation. However, this algorithm actually learned to output a constant value for each of the stocks, under both training and test sets (a different constant for each stock). Why it did this is still a puzzle to us, and a subject of further investigation.

We also attempted to use the cascade-correlation algorithm, but were unable to get it to converge for our data. We suspect that this is because the cascade-correlation algorithm does not explore as wide a volume in the parameter space as the other two algorithms, since it freezes the weights between input and hidden units.

DIRECTIONS FOR FURTHER WORK

In backpropagation learning, or, for that matter, in any algorithm that searches a parameter (or weight) space for an optimal set of parameters, the learning time increases with the number of parameters. Moreover, as Weigend *et al.* [WeHuRu91] and others have pointed out, Occam's razor would lead us to prefer models with fewer parameters. Several researchers have proposed methods for reducing the number of parameters in backpropagation learning and variants thereof. In particular, Weigend *et al* have found that using one such method, which they call weight elimination, can yield improved performance in forecasting tasks such as the one that we have attempted in this chapter. They add to the error a term that is roughly a normalised sum of the squares of the weights. Since we are minimising the error, the backpropagation algorithm will attempt to send to zero some of the weights at the expense of others, to minimise the total sum of squares of the weights. We have not yet tried this algorithm on our data sets, but it looks promising.

We plan to investigate the cascade-correlation algorithm further, possibly extending it to a recurrent version. This algorithm has proven effective in other empirical studies, and it is possible that we can adapt it for testing the EMH.

We also plan to perform a factor and/or cluster analysis of each of the backpropagation networks that we have developed, to attempt to account for its performance in terms that are theoretically understandable, instead of simply viewing the network as a 'black box'. It may instruct us as to which input data are most relevant to the network.

CONCLUSION

We estimated two models, a backpropagation model and a temporal difference model, on six stock returns. With both models, we found some evidence against the null hypothesis that the stock market is weakly efficient.

Two particularly important observations emerge from this study. Traditional tests of the efficient market hypothesis have focused on linear models of asset prices. It is only recently, with the advent of nonlinear modelling techniques, that this hypothesis has been put to a more stringent test. These new tests have provided some contradictory evidence to the efficient market hypothesis. Secondly, neural networks have been shown to be a rich class of nonlinear optimisation models. Their relatively simple and intuitive architecture has been able to model fairly complex dynamic behaviour.

ACKNOWLEDGEMENTS

We would like to thank Yoshiro Miyata for developing his excellent PlaNet neural network simulation package which we have used extensively. This chapter is a revised and updated version of a paper presented at an earlier conference.