

UNIVERSITY OF
ILLINOIS LIBRARY
AT URBANA-CHAMPAIGN
BOOKSTACKS

330
B385

STX

1992:128 COPY 2

Forecasting Exchange Rates Using Feedforward and Recurrent Neural Networks

The Library of UIC
JUN 8 1992
University of Illinois
of Urbana-Champaign

Chung-Ming Kuan
Department of Economics
University of Illinois

Tung Liu
Department of Economics
Ball State University

BEBR

FACULTY WORKING PAPER NO. 92-0128

College of Commerce and Business Administration


University of Illinois at Urbana-Champaign

May 1992

Forecasting Exchange Rates Using Feedforward and Recurrent Neural Networks

Chung-Ming Kuan
Department of Economics

Tung Liu
Department of Economics



Digitized by the Internet Archive
in 2012 with funding from
University of Illinois Urbana-Champaign

FORECASTING EXCHANGE RATES USING
FEEDFORWARD AND RECURRENT
NEURAL NETWORKS

Chung-Ming Kuan

Department of Economics
University of Illinois at Urbana-Champaign

and

Tung Liu

Department of Economics
Ball State University

May 15, 1992

† We would like to thank Roger Koenker, Bill Maloney and Paul Newbold for useful discussions. We are most grateful to Richard Baillie for providing us the data set and Hal White for permitting us to access his NEUTON program, which helped us to improve our program significantly. C.-M. Kuan also thanks the Research Board of the University of Illinois for partial research support.

Abstract

In this paper we investigate the forecasting ability of feedforward and recurrent networks based on empirical foreign exchange rate data. A two-step procedure is proposed to construct suitable networks, in which networks are selected based on the predictive stochastic complexity (PSC) criterion. We find that PSC is a sensible criterion in selecting networks and that neural networks perform reasonably well in terms of out-of-sample MSE and sign predictions. In particular, the networks selected based on PSC have rather satisfactory sign prediction results and compare favorably with the ARMA models selected based on the SIC criterion.

1 Introduction

Neural network is a general class of nonlinear models which has been successfully applied in many different fields. Numerous empirical and computational applications can be found in the Proceedings of the International Joint Conference on Neural Networks and Conference of Neural Information Processing Systems. In spite of its success in various fields, there are only a few applications of neural networks in economics. Neural networks are novel in econometric applications in the following two respects. First, the class of multi-layer neural networks can well approximate a large class of functions (Hornik, Stinchcombe, and White (1989) and Cybenko (1989)), whereas most of commonly used nonlinear time series models do not have this property. Second, the approximation capability of neural networks requires only that the number of parameters grow linearly (Barron (1991)). This is in contrast to polynomial, spline, and trigonometric expansions which require the number of parameters to grow exponentially to achieve the same approximation rate. Thus, if the behavior of economic variables exhibits nonlinearity, a suitably constructed neural network can serve as a useful tool to capture such regularity.

In this paper we investigate possible nonlinear patterns in foreign exchange data using *feedforward* and *recurrent* networks. It has been widely accepted that foreign exchange rates are I(1) (integrated of order one) processes and that changes of exchange rates are uncorrelated over time. Hence, exchange rates are not predictable in general. For a comprehensive review in these issues we refer to Baillie and McMahon (1989). Since the empirical studies supporting these conclusions rely mainly on *linear* time series techniques, it is not unreasonable to conjecture that the linear unpredictability of exchange rates may be due to limitations of linear models. Hsieh (1989) finds that changes of exchange rates may be nonlinearly dependent, even though they are linearly uncorrelated. Some researchers also give evidence in favor of nonlinear forecasts, e.g., Taylor (1980,1982), Engel and Hamilton (1990), Engel (1991), and Chinn (1991). On the other hand, Diebold and Nason (1990) find that nonlinearities of exchange rates, if any, cannot be exploited to improve forecasting. Therefore, we focus on whether neural networks can provide superior out-of-sample forecasts.

In our application, a two-step procedure for network construction is proposed. In the first step, we compute the so-called “predictive stochastic complexity” (Rissanen (1987))

using a computationally efficient, recursive estimation method, from which we can select suitable networks. In the second step, the recursive estimates are “smoothed” to improve statistical efficiency. Our procedure differs from previous applications of feedforward network in economics, e.g., White (1988) and Kuan and White (1990), in that networks are selected objectively. Also, the application of recurrent network appears to be new in economics; as a recurrent network can be viewed as a model with dynamic latent variables, its performance relative to feedforward network should also be of interest to researchers. Our results show that predictive stochastic complexity is a sensible criterion in selecting networks and that neural networks perform reasonably well in terms of out-of-sample MSE and sign predictions. In particular, the networks obtained from the proposed procedure yield rather satisfactory sign prediction results, especially for the Japanese Yen, Deutsche Mark, and Swiss Franc series, and compares favorably with ARMA models.

This paper proceeds as follows. We review various network architectures and estimation methods in section 2. The network construction procedures are described in section 3. Empirical results are analyzed in section 4. Section 5 concludes the paper. We summarize a “recurrent Newton” algorithm in the Appendix.

2 Feedforward and Recurrent Networks

In this section we briefly describe feedforward and recurrent networks and associated estimation methods. For more details see Kuan and White (1991a).

2.1 Network Architectures

A typical single-output, *feedforward* neural network consists of an input layer with n input units, a hidden layer with q hidden units, and an output layer with an output unit. Let x be an n -vector of input variables. The input variables first simultaneously activate q hidden units through some function Ψ , and the hidden unit activations $h_i, i = 1, \dots, q$, then activate output units through some function Φ to produce the network output o . Symbolically, we have

$$h_i = \Psi(\gamma_{i0} + \sum_{j=1}^n \gamma_{ij}x_j), \quad i = 1, \dots, q,$$

$$o = \Phi\left(\beta_0 + \sum_{i=1}^q \beta_i h_i\right). \quad (1)$$

More compactly, we can write

$$\begin{aligned} o &= \Phi\left(\beta_0 + \sum_{i=1}^q \beta_i \Psi\left(\gamma_{i0} + \sum_{j=1}^n \gamma_{ij} x_j\right)\right) \\ &=: f(x, \theta), \end{aligned} \quad (2)$$

where θ is the vector of parameters containing all β 's and γ 's. This is a flexible nonlinear functional form in that the activation functions Ψ and Φ can be chosen quite arbitrarily, except that Ψ is generally required to be a bounded function. Hornik, Stinchcombe, and White (1989) and Cybenko (1989) show that the function f in (2) can approximate a large class of functions arbitrarily well (in a suitable metric), provided that the number of hidden units, q , is sufficiently large. This property is very similar to that of nonparametric methods. Barron (1991) also shows that, to achieve the same approximation rate, a feedforward network uses only linearly many parameters $O(qn)$, whereas traditional polynomial, spline, and trigonometric expansions use exponentially many parameters $O(q^n)$. These two properties make feedforward networks an attractive econometric tool in (non-parametric) applications.

However, the inputs x included in a feedforward network may not be sufficient to characterize the behavior of targets in some applications. In view of this deficiency, various networks allowing feedbacks have been proposed. In particular, we consider the following *recurrent* network due to Elman (1988):

$$\begin{aligned} h_{i,t} &= \Psi\left(\gamma_{i0} + \sum_{j=1}^n \gamma_{ij} x_{j,t} + \sum_{\ell=1}^q \delta_{i\ell} h_{\ell,t-1}\right), \quad i = 1, \dots, q, \\ o_t &= \Phi\left(\beta_0 + \sum_{i=1}^q \beta_i h_{i,t}\right). \end{aligned} \quad (3)$$

Here, the hidden-unit activations h_i feed back to the input layer with delay and serve to “memorize” the past information, cf. (1). Note that we have added the time index t to (3) to indicate the feedback (time delay) effect. It is straightforward to see that by recursive substitution,

$$o_t = \Phi\left(\beta_0 + \sum_{i=1}^q \beta_i \Psi\left(\gamma_{i0} + \sum_{j=1}^n \gamma_{ij} x_{j,t} + \sum_{\ell=1}^q \delta_{i\ell} h_{\ell,t-1}\right)\right)$$

$$\begin{aligned}
&= \Phi\left(\beta_0 + \sum_{i=1}^q \beta_i \Psi\left(\gamma_{i0} + \sum_{j=1}^n x_{j,t} \gamma_{ij} + \right.\right. \\
&\quad \left.\left. \sum_{\ell=1}^q \delta_{i\ell} \Psi\left(\gamma_{\ell 0} + \sum_{k=1}^n \gamma_{\ell k} x_{k,t-1} + \sum_{m=1}^q \delta_{\ell m} h_{m,t-2}\right)\right)\right) \\
&\quad \vdots \\
&=: g(x^t, \tilde{\theta}),
\end{aligned} \tag{4}$$

where $x^t = (x_t, x_{t-1}, \dots, x_1)$ and $\tilde{\theta}$ is the vector of parameters containing all β 's, γ 's, and δ 's. In contrast with (2), the network output o_t is a function of x_t and its entire history. We thus expect that a recurrent network may capture more dynamic characteristics of y_t than does a feedforward network.

2.2 Estimation Methods

Given a target variable y and a feedforward network (2), we want to find suitable parameters θ^* minimizing

$$E|y - f(x, \theta)|^2 = E|y - E(y|x)|^2 + E|E(y|x) - f(x, \theta)|^2. \tag{5}$$

This is equivalent to minimizing $E|E(y|x) - f(x, \theta)|^2$. That is, we want to use feedforward network to approximate the unknown conditional mean function. Since $E(y|x)$ is the best L_2 -predictor of y given x , the network output $f(x, \theta^*)$ should match the target variables fairly closely, at least in the L_2 sense. In view of (5), the unknown parameters can be estimated using the method of Nonlinear Least Squares (NLS). Alternatively, recursive estimation methods may be used. Although recursive estimation is important for adaptive learning and on-line signal processing, it is well known that recursive algorithms do not utilize the data efficiently in finite samples. However, recursive estimation can provide useful starting values for the NLS estimator and facilitate network selection (see discussions in Section 3). Specifically, we consider the following *stochastic Newton* algorithm:

$$\begin{aligned}
\hat{\theta}_{t+1} &= \hat{\theta}_t + \eta_t \hat{G}_t^{-1} \nabla f(x_t, \hat{\theta}_t) [y_t - f(x_t, \hat{\theta}_t)], \\
\hat{G}_{t+1} &= \hat{G}_t + \eta_t [\nabla f(x_t, \hat{\theta}_t) \nabla f(x_t, \hat{\theta}_t)' - \hat{G}_t],
\end{aligned} \tag{6}$$

where $\nabla f(x, \theta)$ is the (column) gradient vector of f with respect to θ and $\{\eta_t\}$ is a sequence of learning rates of order $1/t$. Kuan and White (1991a) show that the estimates of the

algorithm (6) are root- T consistent and asymptotically equivalent to the NLS estimator under very general conditions. In practice, an algebraically equivalent form of (6) can be employed to avoid matrix inversion in the algorithm; see Kuan and White (1991a) for details.

Similarly, the parameters of interest of a recurrent network are $\tilde{\theta}^*$ that minimize

$$E |y_t - g(x^t, \tilde{\theta})|^2,$$

and $g(x^t, \tilde{\theta}^*)$ can be viewed as an approximation of $E(y_t|x^t)$. However, estimation of a recurrent network is not straightforward. In view of (4), the network output o depends on $\tilde{\theta}$ directly and indirectly through the presence of lagged hidden-unit activations. Hence g is a very complex function of $\tilde{\theta}$. In particular, in calculating the derivatives of g with respect to $\tilde{\theta}$, parameter dependence of feedbacks $h_{i,t-1}$ must be taken into account. Owing to this “state dependent” structure, the method of NLS becomes infeasible and the algorithm (6) is invalid. In our applications, a “recurrent Newton” algorithm analogous to (6) is adopted. Kuan and Liu (1992) show that this algorithm is strongly consistent, provided that recurrent connections δ ’s are constrained suitably, and is computationally more efficient than the “recurrent back-propagation” algorithm proposed in Kuan, Hornik, and White (1991). To avoid introducing excessive notations here, the details of this algorithm are deferred to the Appendix. The working papers cited above are available upon request from the first author.

3 Network Construction

In this paper, we choose the activation functions Ψ as the logistic function and Φ as the identity function. These choices are quite standard in neural network literature. We adopt the following two-step procedure to estimate networks.

1. Perform recursive estimation using the stochastic Newton algorithm (6) or its recurrent counterpart described in the Appendix.
 - We generate 10 sets of parameters and choose the one with the lowest mean squared error (MSE) as the initial values for recursive algorithms.
 - We let the algorithm run through the data set 10 times; the final estimates from each pass of the data are used as the initial values of the next pass.

2. Perform NLS estimation using FORTRAN subroutine MINPACK.

- For feedforward network, the final recursive estimates from the last pass of the data are used as initial values of the NLS estimator for θ .
- For recurrent network, we fix the recurrent connection δ 's at the final recursive estimates and use the final estimates as initial values of the NLS estimator for forward connections β 's and γ 's.

From our experience, performing recursive estimation more than 5 times yields quite stable results. In “smoothing” the estimates for recurrent network, the parameters δ 's are fixed to avoid constraint minimization. (Recall that δ 's must be constrained suitably to ensure proper convergence behavior.) Hence, the second step for recurrent network is analogous to building a partially hard-wired recurrent network (Kuan and Hornik (1991)).

The more difficult problem is to determine network complexity. A very simple network may not be able to approximate the unknown conditional mean function well; an excessively complex network may over fit the data. There is, however, no definite conclusion regarding the determination of network complexity. One possible criterion is the Schwarz (1978) Information Criterion (SIC). Rissanen (1983,1984) shows that this criterion can be applied to a more general setting than linear models; in particular, the SIC is asymptotically equivalent to *stochastic complexity* of a model (Rissanen (1987)). When the SIC is applied to determine the order of an ARMA model, it is also known that the SIC is dimensionally consistent (Hannan (1980)). Note, however, that selecting networks based on SIC is computationally demanding because NLS is required for estimating *every* possible network.

An alternative criterion to regularize network complexity is the “Predictive Stochastic Complexity” (PSC) criterion due to Rissanen (1986a,b); see also Rissanen (1987). Given a function $h(x, \theta)$, where θ is a k -dimensional parameter vector, and a sample of T observations, PSC is computed from *honest* prediction errors as

$$\sum_{t=k+1}^T (y_t - h(x_t, \hat{\theta}_{t-1}))^2 / (T - k), \quad (7)$$

where $\hat{\theta}_{t-1}$ is the parameter estimate obtained using the data up to time $t - 1$. The prediction error $y_t - h(x_t, \hat{\theta}_{t-1})$ is “honest” in the sense that no information at time t

or beyond is used to calculate $\hat{\theta}_{t-1}$. A particular model is selected if it has the smallest PSC within a class of models. If two models have the same PSC, the simpler one is selected. Clearly, the PSC criterion is based on forward validation, which is particularly important in forecasting. Rissanen also shows that for encoding a sequence of numbers, the PSC criterion can determine the code with the shortest code length asymptotically. For a thorough discussion of the notion of stochastic complexity we refer to Rissanen (1989). This criterion has also been applied to determine the order of ARMA models, e.g., Gerencsér (1990), Hemerly and Davis (1989), and Hannan, McDougall, and Poskitt (1989). Obviously, calculation of PSC is also computationally demanding if NLS is required to estimate $\hat{\theta}_t$ at each t . Following the idea of Gerencsér and Rissanen (1991), we can compute $\hat{\theta}_t$ using the recursive estimation method, which is more tractable computationally. In our two-step procedure, PSC can be computed easily in the first step; specifically, PSC is computed from the last pass of the data in recursive estimation.

4 Empirical Results

In this paper five exchange rates, including Canadian Dollar (CD), Deutsche Mark (DM), Japanese Yen (JY), Pound Sterling (PS), and Swiss Franc (SF), are investigated. The data are daily opening bid prices of NY Foreign Exchange Market from March 1, 1980 to January 28, 1985, consisting of 1245 observations. All series except PS are US dollars per unit of foreign currency. This data set has also been used in Baillie and Bollerslev (1989).

Let $S_{i,t}$ denote the i -th exchange rate at time t , and $y_{i,t} = \log S_{i,t} - \log S_{i,t-1}$, $i =$ CD, DM, JY, PS, SF. By applying various unit-root tests of Phillips (1987), Phillips and Perron (1988), and Perron (1988), Baillie and Bollerslev (1989) find that $\log S_{i,t}$ are unit root processes without drift and that changes of log exchange rates behave like a martingale difference sequence. In addition, we estimate 36 ARMA models from ARMA(0,0) to ARMA(5,5) on $y_{i,t}$ and evaluate the resulting SIC values. These SIC values, which are summarized in Table 1, indicate that ARMA(0,0) is the best model for all five series. As the SIC is dimensionally consistent, this result agrees with the finding of Baillie and Bollerslev.

[Table 1 About Here]

To construct neural networks, we follow the two step procedure described in Section 3 and take $y_{i,t}$ as target variables. We use 1194 observations for in-sample estimation and reserve the last 50 observations for out-of-sample forecasting. In the first step, 36 feedforward and recurrent networks (with 1–6 lagged targets as inputs and 1–6 hidden units) are estimated using the Newton algorithms. We shall write the network with L lags and H hidden units as the network (L, H) . For each series, five networks with best PSC are selected. In the second step, the parameter estimates of the selected networks are “smoothed” using NLS. Table 2 contains the PSC values of all feedforward and recurrent networks. To save space, we do not report in-sample MSE here. It is not surprising to note that, in general, in-sample MSE from NLS estimation are much better than those from recursive estimation.

[Table 2 About Here]

It is typical to evaluate forecasting performance based on out-of-sample MSE. Another important criterion is to compare out-of-sample sign predictions of different models. Sign prediction provides forecasts of the direction of future changes, hence gives important information in financial forecasting. In an extreme case, a model could have small out-of-sample MSE but predict all the signs incorrectly, hence is virtually useless. We summarize out-of-sample MSE and percentage of correct sign predictions of the selected networks in Table 3. As a comparison, out-of-sample forecasting results from five ARMA models, including ARMA(0,0) which is the best model based on the SIC, are also included. It can be seen that the PSC criterion selects a wide variety of networks for each series. Note, however, that the PSC criterion tends to select more complex networks; most of the selected networks contain 3–6 hidden units. From Table 3 we also observe the following.

1. Out-of-sample MSE:

- (a) The selected feedforward and recurrent networks do not dominate each other, and a better network (i.e., a network with smaller PSC) need not have better out-of-sample MSE.
- (b) For the DM and JY, four out of five selected feedforward networks perform better than all ARMA models; for the SF, three out of five selected feedforward networks perform better than all ARMA models.

- (c) For the JY and SF, four out of five selected recurrent networks perform better than all ARMA models.
- (d) For the CD, ARMA models perform better than network models.
- (e) The best feedforward network performs better than ARMA(0,0) for all five series, and the best recurrent network performs better than ARMA(0,0) for the DM, JY, and SF.

2. Out-of-sample sign predictions:

- (a) Correct sign predictions of ARMA models fluctuate around 50%, except for the PS, which are close to 60%.
- (b) For the DM, JY, and SF, the selected feedforward networks perform better than ARMA models and usually have more than 60% correct sign predictions. In particular, for the JY, all five selected feedforward networks have correct sign predictions more than 60%, and two of them have 66% correct; for the SF, four best feedforward networks have correct sign predictions more than 60%.
- (c) For the DM, JY, PS, and SF, the selected recurrent networks perform better than ARMA models; and for the PS and SF, three out of five selected recurrent networks have more than 60% correct sign predictions.
- (d) For all series except the CD, almost all the selected networks have more than 50% correct sign predictions.

[Table 3 About Here]

The results above suggest that the PSC criterion is a quite sensible criterion. The best network selected based on PSC has good out-of-sample performance and compares favorably with the best ARMA model selected based on the SIC. As far as out-of-sample MSE being concerned, the selected networks seem to perform well for the DM, JY, and SF, but their performance does *not* dominate ARMA models significantly. In terms of out-of-sample sign predictions, while ARMA models usually perform no better than tossing a coin (i.e., 50% chance being correct), network models have rather satisfactory predicting ability. This is especially true for the DM, JY, and SF series. It also appears that feedforward networks have more stable sign prediction results than recurrent networks. It is somewhat

surprising to us that recurrent networks do not perform as good as feedforward networks. One possible interpretation is that the feedback structure in recurrent networks cannot be very effective if there is very little correlation across target variables.

To obtain a complete picture of the performance of feedforward and recurrent networks, we “smoothed” all other networks not selected by the PSC criterion. By inspecting the resulting SIC values, we find that the SIC criterion almost always selects the simplest network (1,1). This is true for both feedforward and recurrent networks. (We do not give a detailed table of the SIC values here.) Note that the SIC penalizes a model in terms of the number of parameters. Thus, the SIC of the network (2,2) has the same complexity penalty as the SIC of the network (6,1), but clearly, the nonlinear structures of these two networks are very different. The out-of-sample MSE and sign predictions results of all 36 networks are collected in Tables 4 and 5. We compare these results with four ARMA models used in Table 3 and give a summary in Table 6. As ARMA(1,0) and ARMA(0,1) perform very similarly, comparison with ARMA(0,1) is not included in Table 6.

Tables 4 and 5 show that feedforward and recurrent networks perform similarly; in particular, networks models have quite satisfactory sign prediction results for the DM, JY, PS, and SF series. We also observe that a more complex network need not predict better than a simpler network and that no network with certain number of hidden units can systematically beat other networks with different number of hidden units. In view of Table 6, we can see that for three exchange rates (DM, JY, and SF), both feedforward and recurrent networks usually perform better than ARMA models in terms of out-of-sample MSE. Again, the out-of-sample MSE of network models do not significantly dominate those of ARMA models. For all five exchange rates, network models have much better sign prediction results than ARMA models. This is compatible with previous prediction results based on selected networks. We have also estimated networks without the bias term β_0 to see whether sign predictions can be improved. However, the estimation results turn out to be unstable; many parameter estimates tend to be extremely large or small with huge variances.

[Tables 4–6 About Here]

5 Conclusions

In this paper we have carefully estimated feedforward and recurrent networks to forecast changes of log exchange rates. We find that PSC is a sensible criterion in selecting networks. Based on this criterion, it is possible to construct a network with better out-of-sample MSE and/or sign prediction than ARMA models. Therefore, the proposed two-step procedure may be used as a standard network construction procedure in other applications. As far as out-of-sample MSE being concerned, we share the same conclusion with Diebold and Nason (1990) that nonlinearities of exchange rates, if any, may not be exploited to improve point prediction. On the other hand, if we are not so ambitious about point forecasts and confine ourselves to sign predictions, our results also suggest that network models perform quite well for this purpose. In particular, it usually performs better than ARMA models and coin tossing. Finally, different exchange rates have different behavior and characteristics. In our application, network models do not predict well for the CD but perform quite well for the DM, JY, and SF series. It also appears that feedforward networks perform slightly better than recurrent networks and have more stable prediction results.

Appendix

A Summary of Recurrent Newton Algorithm:

We write a recurrent network as

$$\begin{aligned}
 o_t &= \Phi\left(\beta_0 + \sum_{i=1}^q \beta_i \Psi\left(\gamma_{i0} + \sum_{j=1}^n \gamma_{ij} x_{j,t} + \sum_{\ell=1}^q \delta_{i\ell} h_{\ell,t-1}\right)\right) \\
 &=: \Phi\left(\beta_0 + \sum_{i=1}^q \beta_i \psi_i(x_t, h_{t-1}, \theta)\right) \\
 &=: \phi(x_t, h_{t-1}, \theta),
 \end{aligned}$$

where θ includes all β 's, γ 's, and δ 's. Note that h_{t-1} is also a function of θ . The recurrent Newton algorithm contains the following updating equations:

$$\begin{aligned}
 \hat{e}_t &= y_t - \phi(x_t, \hat{h}_{t-1}, \hat{\theta}_t), \\
 \nabla \hat{e}_t &= -\phi_\theta(x_t, \hat{h}_{t-1}, \hat{\theta}_t)' - \hat{\Delta}_t \phi_h(x_t, \hat{h}_{t-1}, \hat{\theta}_t)', \\
 \hat{\theta}_{t+1} &= \hat{\theta}_t - \eta_t \hat{G}_t^{-1} \nabla \hat{e}_t \hat{e}_t, \\
 \hat{G}_{t+1} &= \hat{G}_t + \eta_t (\nabla \hat{e}_t \nabla \hat{e}_t' - \hat{G}_t),
 \end{aligned}$$

where the i -th hidden unit is updated according to

$$\begin{aligned}
 \hat{h}_{i,t} &= \psi_i(x_t, \hat{h}_{t-1}, \hat{\theta}_t) \\
 &= (\hat{\gamma}_{i0,t} + \sum_{j=1}^n \hat{\gamma}_{ij,t} x_{j,t} + \sum_{\ell=1}^q \hat{\delta}_{i\ell,t} \hat{h}_{\ell,t-1}), \quad i = 1, \dots, q,
 \end{aligned}$$

the j -th column of $\hat{\Delta}_{t+1}$ is updated according to

$$\hat{\Delta}_{j,t+1} = \psi_{j,\theta}(x_t, \hat{h}_{t-1}, \hat{\theta}_t)' + \hat{\Delta}_t \psi_{j,h}(x_t, \hat{h}_{t-1}, \hat{\theta}_t)', \quad j = 1, \dots, q,$$

and the initial values $\hat{\theta}_0$, \hat{h}_0 , and $\hat{\Delta}_0$ are chosen arbitrarily. Here, ϕ_θ and ϕ_h are (row) vectors of the first order derivatives of ϕ with respect to θ and h , respectively, and $\psi_{i,\theta}$ and $\psi_{i,h}$ are (row) vectors of the first order derivatives of the i -th hidden unit ψ_i with respect to θ and h , respectively.

This algorithm differs from the recurrent back-propagation algorithm of Kuan, Hornik, and White (1991) in that a Newton direction \hat{G}_t^{-1} is added in the updating equation of $\hat{\theta}_t$. Note that in this algorithm the derivatives of prediction error e with respect to θ contains

two parts because e depends on θ directly and indirectly through the presence of lagged hidden-unit activations, i.e.,

$$\nabla \hat{e}_t = -\phi_\theta(x_t, \hat{h}_{t-1}, \hat{\theta}_t)' - \frac{dh_{t-1}}{d\theta} \phi_h(x_t, \hat{h}_{t-1}, \hat{\theta}_t)'.$$

Hence, the updating equation for $\hat{\Delta}_t$ allows us to update the $dh_{t-1}/d\theta$ term recursively. Clearly, a recurrent network not depend on h_{t-1} is a feedforward network. In this case, ϕ_h term is zero, and there is no need to consider $\hat{\Delta}_t$ term. The recurrent Newton algorithm simply reduces to the standard Newton algorithm (6). For more details see Kuan and Liu (1992).

References

- Baillie, R. T., and T. Bollerslev (1989). Common Stochastic Trends in a System of Exchange Rates, *Journal of Finance*, **44**, 167–181.
- Baillie, R. T., and P. C. McMahon (1989). *The Foreign Exchange Market: Theory and Econometric Evidence*, New York: Cambridge University Press.
- Barron, A. R. (1991). Universal approximation bounds for superpositions of a sigmoidal function, Technical Report No. 58, Department of Statistics, University of Illinois, Urbana-Champaign.
- Chinn, M. D. (1991). Some linear and nonlinear thoughts on exchange rates, *Journal of International Money and Finance*, **10**, 214–230.
- Cybenko, G. (1989). Approximations by superpositions of a sigmoidal function, *Mathematics of Control, Signals and Systems*, **2**, 303–314.
- Diebold, F. X., and J. A. Nason (1990). Nonparametric exchange rate prediction?, *Journal of International Economics*, **28**, 315–332.
- Elman, J. L. (1988). Finding structure in time, CRL Report 8801, Center for Research in Language, University of California, San Diego.
- Engel, C. (1991). Can the Markov switching model forecast exchange rates?, Working Paper, University of Washington.
- Engel, C., and J. Hamilton (1990). Long Swings in the exchange rates: Are they in the data and do markets know it?, *American Economic Review*, **80**, 689–713.
- Gerencsér, L. (1990). On Rissanen’s predictive stochastic complexity for stationary ARMA processes, Technical Report, Department of Electrical Engineering, McGill University.
- Gerencsér, L., and J. Rissanen (1991). Asymptotics of predictive stochastic complexity, *Proceedings of the 1990 IMA Workshop: New Directions in Time Series Analysis*, Springer-Verlag.

- Hannan, E. J. (1980). The estimation of the order of an ARMA process, *Annals of Statistics*, **8**, 1071–1081.
- Hannan, E. J., McDougall, A. J., and D. S. Poskitt (1989). Recursive estimation of autoregressions, *Journal of the Royal Statistical Society, B*, **51**, 217–233.
- Hemerly, E. M. and M. Davis (1989). Strong consistency of the PLS criterion for order determination of autoregressive processes, *Annals of Statistics*, **17**, 941–946.
- Hornik, K., Stinchcombe, M., and H. White (1989). Multi-layer feedforward networks are universal approximators, *Neural Networks*, **2**, 359–366.
- Kuan, C.-M., and K. Hornik (1991). Learning in a partially hard-wired recurrent network, *Neural Network World*, **1**, 39–45.
- Kuan, C.-M., Hornik, K., and H. White (1990). Some convergence results for learning in recurrent neural networks, Discussion Paper 90-42, Department of Economics, University of California, San Diego.
- Kuan, C.-M., and T. Liu (1992). A Recurrent Newton Algorithm and Its Convergence Property, in preparation.
- Kuan, C.-M., and H. White (1990). Predicting appliance ownership using logit, neural network, and regression tree models, BEBR Working Paper 90-1647, College of Commerce, University of Illinois, Urbana-Champaign.
- Kuan, C.-M., and H. White (1991a). Artificial neural networks: An econometric perspective, *Econometric Reviews*, forthcoming.
- Kuan, C.-M., and H. White (1991b). Strong convergence of recursive m-estimators for models with dynamic latent variables, Discussion Paper 91-05R, Department of Economics, University of California, San Diego.
- Perron, P. (1988). Trends and random walks in macroeconomic time series: Further evidence from a new approach, *Journal of Economic Dynamic and Control*, **12**, 297–332.

- Phillips, P. C. B. (1987). Time series regression with a unit root, *Econometrica*, **55**, 277–301.
- Phillips, P. C. B., and P. Perron (1988). Testing for a unit root in time series regression, *Biometrika*, **75**, 335–346.
- Rissanen, J. (1983). A universal prior for integers and estimation by minimum description length, *Annals of Statistics*, **11**, 416–431.
- Rissanen, J. (1984). Universal coding, information, prediction, and estimation, *IEEE Transactions on Information Theory*, **IT-30**, 629–636.
- Rissanen, J. (1986a). A predictive least-squares principle, *IMA Journal of Mathematical Control & Information*, **3**, 211–222.
- Rissanen, J. (1986b). Stochastic complexity and modeling, *Annals of Statistics*, **14**, 1080–1100.
- Rissanen, J. (1987). Stochastic complexity (with discussions), *Journal of the Royal Statistical Society*, B, **49**, 223–239 and 252–265.
- Rissanen, J. (1989). *Stochastic Complexity in Statistical Inquiry*, Singapore: World Science Publishing Co.
- Schwarz, G. (1978). Estimating the dimension of a model, *Annals of Statistics*, **6**, 461–464.
- Taylor, S. J. (1980). Conjectured models for trends in financial prices, tests and forecasts, *Journal of the Royal Statistical Society*, A, **143**, 338–362.
- Taylor, S. J. (1982). Tests of the random walk hypothesis against a price trend hypothesis, *Journal of Financial and Quantitative Analysis*, **17**, 37–61.
- White, H. (1988). Economic prediction using neural networks: The case of IBM Stock Prices, *Proceedings of the IEEE Second International Conference on Neural Networks*, II, pp. 451–458, San Diego: SOS Printing.

Table 1. The SIC Values of ARMA Models for Changes of log Exchange Rates.

ARMA Models	SIC Values				
	CD	DM	JY	PS	SF
(0,0)	-11.9439	-9.8901	-9.9605	-10.0422	-9.7273
(0,1)	-11.9392	-9.8850	-9.9588	-10.0372	-9.7222
(0,2)	-11.9333	-9.8810	-9.9529	-10.0316	-9.7201
(0,3)	-11.9276	-9.8762	-9.9476	-10.0273	-9.7142
(0,4)	-11.9228	-9.8704	-9.9461	-10.0233	-9.7084
(0,5)	-11.9182	-9.8648	-9.9421	-10.0174	-9.7039
(1,0)	-11.9389	-9.8843	-9.9596	-10.0385	-9.7214
(1,1)	-11.9335	-9.8786	-9.9537	-10.0333	-9.7160
(1,2)	-11.9338	-9.8759	-9.9478	-10.0278	-9.7134
(1,3)	-11.9316	-9.8697	-9.9446	-10.0249	-9.7076
(1,4)	-11.9294	-9.8637	-9.9412	-10.0191	-9.7019
(1,5)	-11.9122	-9.8580	-9.9372	-10.0139	-9.6971
(2,0)	-11.9340	-9.8792	-9.9531	-10.0327	-9.7184
(2,1)	-11.9317	-9.8742	-9.9472	-10.0268	-9.7126
(2,2)	-11.9236	-9.8695	-9.9417	-10.0222	-9.7074
(2,3)	-11.9275	-9.8635	-9.9410	-10.0183	-9.7017
(2,4)	-11.9227	-9.8573	-9.9415	-10.0127	-9.6953
(2,5)	-11.9089	-9.8517	-9.9368	-10.0070	-9.6916
(3,0)	-11.9293	-9.8745	-9.9474	-10.0284	-9.7122
(3,1)	-11.9242	-9.8688	-9.9419	-10.0230	-9.7063
(3,2)	-11.9197	-9.8632	-9.9361	-10.0169	-9.7005
(3,3)	-11.9147	-9.8588	-9.9374	-10.0115	-9.6969
(3,4)	-11.9167	-9.8526	-9.9315	-10.0099	-9.6901
(3,5)	-11.9104	-9.8453	-9.9303	-10.0043	-9.6930
(4,0)	-11.9266	-9.8678	-9.9445	-10.0242	-9.7057
(4,1)	-11.9261	-9.8619	-9.9413	-10.0183	-9.6999
(4,2)	-11.9161	-9.8563	-9.9359	-10.0130	-9.6940
(4,3)	-11.9107	-9.8501	-9.9306	-10.0094	-9.6889
(4,4)	-11.9053	-9.8454	-9.9253	-10.0033	-9.6825
(4,5)	-11.9119	-9.8402	-9.9199	-9.9974	-9.6801
(5,0)	-11.9224	-9.8616	-9.9423	-10.017	-9.7002
(5,1)	-11.9205	-9.8556	-9.9363	-10.011	-9.6942
(5,2)	-11.9157	-9.8501	-9.9337	-10.006	-9.6897
(5,3)	-11.9057	-9.8439	-9.9293	-10.003	-9.6830
(5,4)	-11.9048	-9.8380	-9.9244	-9.9974	-9.6771
(5,5)	-11.8976	-9.8328	-9.9167	-9.9929	-9.6710

Table 2. The PSC Values of Feedforward and Recurrent Networks.

Network Models	PSC Values									
	Feedforward Networks					Recurrent Networks				
	CD	DM	JY	PS	SF	CD	DM	JY	PS	SF
(1,1)	.6906	.5580	.5143	.4872	.6428	.6936	.5580	.5172	.4973	.6430
(2,1)	.6923	.5577	.5028	.4869	.6426	.6907	.5574	.5144	.4868	.6415
(3,1)	.6869	.5590	.5149	.4863	.6423	.6913	.5566	.5159	.4863	.6411
(4,1)	.6905	.5571	.5173	.4873	.6407	.6889	.5586	.5161	.4871	.6408
(5,1)	.6914	.5530	.5159	.4877	.6395	.6901	.5531	.5170	.4866	.6425
(6,1)	1.200	.5569	.5146	.4850	.6400	.6868	.5579	.5174	.4844	.6427
(1,2)	.6827	.5577	.5143	.4873	.6398	.6931	.5567	.5174	.4861	.6426
(2,2)	.6906	.5580	.5028	.4865	.6425	.6882	.5589	.5146	.4863	.6408
(3,2)	.6918	.5563	.5131	.4864	.6418	.6904	.5533	.5148	.4870	.6403
(4,2)	.6852	.5560	.5126	.4843	.6410	.6880	.5575	.5143	.4843	.6406
(5,2)	.6875	.5544	.5152	.4827	.6417	.6826	.5579	.5166	.4849	.6421
(6,2)	.6893	.5577	.5137	.4842	.6360	.6913	.5570	.5120	.4847	.6415
(1,3)	.6817	.5567	.5125	.4906	.6409	.6905	.5578	.5052	.4894	.6433
(2,3)	.7125	.5595	.5084	.4862	.6396	.7003	.5571	.5509	.4849	.6430
(3,3)	.6904	.5571	.5143	.4848	.6399	.7888	.5581	.5167	.4948	.6373
(4,3)	.6885	.5520	.5123	.4834	.6402	.6859	.5556	.5144	.4853	.6417
(5,3)	.6908	.6847	.5120	.4821	.6414	.6889	.5552	.5205	.4819	.6411
(6,3)	.6855	.5579	.5136	.4864	.6428	.6796	.5544	.5143	.4828	.6380
(1,4)	.7005	.5555	.4941	.4884	.6452	.8418	.5578	.6200	.5046	.6712
(2,4)	.7144	.5564	.5095	.4992	.6349	.7323	.5539	.4962	.4886	.6419
(3,4)	.6807	.5550	.5144	.4853	.6391	.6885	.5572	.5130	.4838	.6405
(4,4)	.6834	.5537	.5153	.4820	.6421	.6846	.5561	.5112	.4805	.6396
(5,4)	.6875	.5545	.5061	.4838	.6423	.6839	.5520	.5107	.4851	.6400
(6,4)	.6909	.5520	.5051	.4805	.6348	.6845	.5485	.5141	.4805	.6412
(1,5)	.6824	.5617	.4885	.4867	.6419	.7093	.5625	.5094	.5022	.6457
(2,5)	.6853	.5544	.4960	.4856	.6431	.6880	.5532	.5432	.4908	.6419
(3,5)	.7247	.5520	.5108	.4835	.6390	.6846	.5566	.5028	.4840	.7249
(4,5)	.6874	.5769	.5050	.4836	.6628	.1075	.5544	.5134	.4810	.6417
(5,5)	.6863	.5559	.5057	.4836	.6329	.6850	.5524	.5112	.4724	.6320
(6,5)	.6830	.5550	.5118	.4821	.6401	.6698	.5457	.5091	.4786	.6412
(1,6)	.6868	.5701	.4875	.4868	.6385	.7016	.5575	.5819	.4875	.6950
(2,6)	.7125	.5593	.5091	.4822	.6389	.7058	.5505	.5313	.4819	.6572
(3,6)	.6862	.5543	.4934	.4896	.6422	.6738	.5548	.5079	.4819	.6328
(4,6)	.7251	.5519	.5139	.4918	.6378	.6739	.5544	.4899	.4861	.6397
(5,6)	.6815	.5528	.5055	.4773	.6399	.6800	.5514	.5089	.4820	.6403
(6,6)	.6810	.5580	.5055	.4759	.6252	.6890	.5532	.5084	.4795	.6362

Notes: Network model (L, H) is the network with L lagged targets as inputs and H hidden units. The other tables follow this convention. The PSC values are the numbers in the table $\times 10^{-1}$, except for the CD, which are $\times 10^{-2}$.

Table 3. Out-of-Sample MSE and Sign Predictions of the Selected Networks.

Exchange Rates	Feedforward Networks			Recurrent Networks			ARMA		
	Selected	MSE	Sign	Selected	MSE	Sign	Models	MSE	Sign
CD	(3,4)	.1896	.52	(6,5)	.2350	.42	(0,0)	.1906	N/A
	(6,6)	.2423	.44	(3,6)	.1862	.50	(1,0)	.1889	.46
	(5,6)	.2730	.36	(4,6)	.1924	.54	(0,1)	.1888	.46
	(1,3)	.1884	.54	(6,3)	.2353	.42	(1,1)	.1884	.46
	(1,5)	.1875	.58	(5,6)	.2196	.38	(2,2)	.1895	.44
DM	(4,6)	.1795	.68	(6,5)	.2041	.56	(0,0)	.2098	N/A
	(6,4)	.1962	.56	(6,4)	.2416	.54	(1,0)	.2098	.48
	(4,3)	.1830	.58	(2,6)	.1984	.64	(0,1)	.2098	.48
	(3,5)	.1899	.60	(5,6)	.1873	.62	(1,1)	.2096	.46
	(5,6)	.2017	.66	(5,4)	.2107	.52	(2,2)	.2033	.54
JY	(1,6)	.1160	.66	(4,6)	.1168	.58	(0,0)	.1225	N/A
	(1,5)	.1158	.62	(2,4)	.1133	.60	(1,0)	.1199	.52
	(3,6)	.1205	.60	(3,5)	.1220	.58	(0,1)	.1201	.50
	(1,4)	.1162	.62	(1,3)	.1166	.60	(1,1)	.1198	.52
	(2,5)	.1127	.66	(3,6)	.1122	.58	(2,2)	.1202	.50
PS	(6,6)	.3880	.52	(5,5)	.4034	.62	(0,0)	.3884	N/A
	(5,6)	.4204	.50	(6,5)	.4295	.54	(1,0)	.3893	.60
	(6,4)	.3866	.46	(6,6)	.4004	.62	(0,1)	.3896	.60
	(4,4)	.3929	.62	(4,4)	.3895	.66	(1,1)	.3915	.56
	(5,3)	.3902	.64	(6,4)	.3838	.48	(2,2)	.3909	.58
SF	(6,6)	.2129	.62	(5,5)	.1796	.62	(0,0)	.2157	N/A
	(5,5)	.1929	.64	(3,6)	.1965	.64	(1,0)	.2162	.56
	(6,4)	.1897	.64	(6,6)	.2490	.50	(0,1)	.2162	.54
	(2,4)	.1954	.62	(3,3)	.1950	.66	(1,1)	.2158	.58
	(6,2)	.2146	.50	(6,3)	.1958	.54	(2,2)	.2124	.52

Notes: For each exchange rate, the selected networks are ordered from the best to the 5-th best, according to the PSC values in Table 2. “MSE” stands for out-of-sample MSE; “Sign” stands for the percentage of correct sign predictions of corresponding models. MSE are the numbers in the table $\times 10^{-4}$, except for the *CD*, which are $\times 10^{-5}$.

Table 4. Out-of-Sample MSE of Feedforward and Recurrent Networks.

Network	Out-of-Sample MSE									
	Feedforward Networks					Recurrent Networks				
Models	CD	DM	JY	PS	SF	CD	DM	JY	PS	SF
(1,1)	.1861	.1998	.1184	.3681	.2063	.1881	.1998	.1217	.3704	.2098
(2,1)	.1883	.1982	.1192	.3653	.1995	.1874	.1982	.1192	.3750	.1992
(3,1)	.1879	.1947	.1217	.3785	.1972	.1887	.1960	.1196	.3735	.2003
(4,1)	.1830	.1937	.1181	.3721	.2005	.1931	.1882	.1162	.3751	.2005
(5,1)	.1839	.1830	.1191	.3697	.2032	.1879	.1845	.1179	.3888	.2008
(6,1)	.1886	.2025	.1243	.3902	.1992	.2028	.2001	.1240	.3555	.1946
(1,2)	.1902	.1948	.1202	.3681	.1974	.1881	.1878	.1159	.3496	.2058
(2,2)	.1860	.1982	.1168	.3748	.1960	.1833	.1929	.1161	.3767	.2006
(3,2)	.1687	.2015	.1238	.3913	.1966	.1973	.1921	.1154	.3800	.1989
(4,2)	.1872	.1887	.1171	.4042	.1957	.1896	.1896	.1142	.3862	.1993
(5,2)	.2080	.2124	.1147	.3812	.2033	.1819	.1995	.1167	.3857	.1952
(6,2)	.2128	.1984	.1174	.3619	.2146	.1878	.1893	.1148	.3809	.2041
(1,3)	.1884	.1930	.1206	.3580	.1966	.1873	.1936	.1166	.3543	.2057
(2,3)	.1808	.2044	.1187	.3875	.1963	.1894	.1974	.1163	.3856	.1933
(3,3)	.1884	.1934	.1185	.3842	.2003	.1878	.1925	.1113	.3791	.1950
(4,3)	.1971	.1830	.1136	.3612	.2012	.1878	.1797	.1159	.3949	.1977
(5,3)	.1908	.1985	.1183	.3902	.2081	.1890	.1817	.1156	.3927	.2127
(6,3)	.2086	.2069	.1287	.3651	.2087	.2353	.1870	.1192	.3750	.1958
(1,4)	.1885	.1935	.1162	.3610	.2044	.1897	.1915	.1146	.3750	.1998
(2,4)	.1873	.1911	.1188	.3936	.1954	.1871	.1892	.1133	.3996	.1913
(3,4)	.1896	.1807	.1146	.3686	.1910	.1912	.1822	.1101	.3758	.2053
(4,4)	.1913	.1872	.1089	.3929	.1990	.1840	.1981	.1066	.3895	.2026
(5,4)	.2072	.1842	.1090	.4094	.2038	.2009	.2107	.1022	.3834	.2002
(6,4)	.2130	.1962	.1080	.3866	.1897	.2138	.2416	.1070	.3838	.1903
(1,5)	.1875	.1933	.1158	.3672	.1985	.1901	.1877	.1152	.3638	.2010
(2,5)	.1819	.1923	.1127	.4045	.1984	.1829	.1967	.1138	.3714	.2021
(3,5)	.1836	.1899	.1149	.3861	.1908	.2010	.1935	.1220	.3888	.1932
(4,5)	.1878	.1796	.1202	.4033	.2039	.1953	.1979	.1276	.3722	.2264
(5,5)	.2035	.1926	.1258	.3954	.1928	.1758	.2050	.1139	.4034	.1796
(6,5)	.1998	.2155	.1123	.4002	.1934	.2350	.2041	.1151	.4295	.2293
(1,6)	.1907	.1933	.1160	.3699	.2011	.1882	.1963	.1166	.3731	.1995
(2,6)	.1895	.1953	.1150	.3788	.1920	.1747	.1984	.1122	.4195	.2094
(3,6)	.1826	.1853	.1205	.3854	.1845	.1862	.1933	.1122	.3842	.1965
(4,6)	.2121	.1795	.1069	.4051	.2000	.1924	.1969	.1168	.3915	.1956
(5,6)	.2730	.2017	.1061	.4204	.1784	.2196	.1873	.1361	.3885	.1954
(6,6)	.2423	.1999	.1344	.3880	.2129	.2236	.1995	.1116	.4004	.2490

Notes: MSE are the numbers in the table $\times 10^{-4}$, except for the *CD*, which are $\times 10^{-5}$.

Table 5. Out-of-Sample Sign Predictions of Feedforward and Recurrent Networks.

Network	Percentages of Correct Sign Predictions									
	Feedforward Networks					Recurrent Networks				
	CD	DM	JY	PS	SF	CD	DM	JY	PS	SF
(1,1)	.56	.64	.60	.72	.66	.56	.62	.60	.72	.64
(2,1)	.56	.62	.64	.72	.60	.56	.62	.64	.70	.66
(3,1)	.56	.54	.60	.72	.56	.56	.62	.62	.54	.62
(4,1)	.54	.58	.60	.58	.64	.50	.60	.60	.72	.64
(5,1)	.56	.60	.52	.64	.68	.56	.58	.52	.46	.54
(6,1)	.56	.62	.46	.72	.54	.30	.52	.50	.50	.62
(1,2)	.50	.62	.50	.72	.66	.56	.62	.66	.70	.66
(2,2)	.56	.62	.62	.58	.66	.52	.54	.56	.60	.66
(3,2)	.58	.58	.50	.42	.60	.46	.56	.62	.66	.58
(4,2)	.56	.60	.60	.60	.68	.48	.60	.58	.58	.54
(5,2)	.26	.52	.50	.64	.54	.54	.58	.58	.68	.58
(6,2)	.44	.60	.60	.58	.50	.56	.60	.52	.64	.60
(1,3)	.54	.60	.62	.64	.66	.56	.66	.60	.60	.54
(2,3)	.52	.54	.62	.66	.72	.54	.56	.60	.60	.56
(3,3)	.54	.56	.62	.56	.62	.52	.60	.64	.68	.66
(4,3)	.42	.58	.68	.60	.58	.58	.64	.60	.66	.66
(5,3)	.52	.58	.60	.64	.56	.56	.62	.62	.60	.56
(6,3)	.56	.54	.54	.60	.60	.42	.62	.56	.40	.54
(1,4)	.54	.60	.62	.58	.60	.50	.64	.66	.72	.68
(2,4)	.56	.54	.50	.62	.62	.56	.60	.60	.68	.62
(3,4)	.52	.64	.62	.60	.66	.52	.60	.62	.70	.46
(4,4)	.50	.64	.62	.62	.60	.56	.58	.64	.66	.54
(5,4)	.46	.64	.56	.50	.56	.48	.52	.64	.64	.60
(6,4)	.44	.56	.66	.46	.64	.44	.54	.60	.48	.62
(1,5)	.58	.58	.62	.68	.66	.54	.68	.66	.64	.58
(2,5)	.56	.64	.66	.62	.64	.52	.58	.58	.68	.62
(3,5)	.52	.60	.64	.62	.64	.48	.62	.58	.68	.64
(4,5)	.56	.60	.60	.52	.56	.54	.58	.52	.68	.62
(5,5)	.52	.64	.44	.48	.64	.60	.54	.56	.62	.62
(6,5)	.52	.58	.58	.46	.54	.42	.56	.54	.54	.60
(1,6)	.52	.58	.66	.62	.62	.56	.66	.58	.72	.62
(2,6)	.52	.60	.58	.62	.64	.56	.64	.64	.52	.60
(3,6)	.52	.58	.60	.52	.68	.50	.64	.58	.60	.64
(4,6)	.38	.68	.70	.60	.58	.54	.58	.58	.66	.60
(5,6)	.36	.66	.70	.50	.62	.38	.62	.50	.54	.64
(6,6)	.44	.50	.50	.52	.62	.40	.58	.56	.62	.50

Table 6. Out-of-Sample Forecasting Comparison: Networks vs. ARMA Models.

Out-of-Sample	Target ARMA Models	Number of Better Networks									
		Feedforward Network					Recurrent Network				
		CD	DM	JY	PS	SF	CD	DM	JY	PS	SF
MSE	(0,0)	22	34	31	23	36	23	34	33	24	33
	(1,0)	19	34	26	23	36	18	34	31	27	33
	(1,1)	17	34	26	26	36	17	34	31	29	33
	(2,2)	20	32	28	25	34	20	32	31	28	32
Sign	Coin	28	36	34	32	36	26	36	36	33	35
	(1,0)	29	36	29	22	32	30	36	34	27	29
	(1,1)	29	36	29	27	28	30	36	34	28	27
	(2,2)	32	34	34	26	35	31	34	36	28	34

Notes: Each numbers in the table is the number of networks (out of 36 estimated networks) that predict better than or the same as corresponding target models. “Coin” stands for 50% chance of getting correct sign prediction.

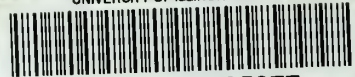
BECKMAN
LABORATORY INC.



JUN 95

To-Please[®] N. MANCHESTER,
INDIANA 46962

UNIVERSITY OF ILLINOIS-URBANA



3 0112 060295877