Applied Mathematical Sciences, Vol. 2, 2008, no. 35, 1705 - 1728

A State Space Search Algorithm and its Application to Learn the Short-Term Foreign Exchange Rates

Leong-kwan Li

Department of Applied Mathematics
The Hong Kong Polytechnic University
Hong Kong, China

S. Shao

Department of Mathematics
Cleveland State University
Cleveland, OH 44115, USA
s.shao@csuohio.edu

Teodora Zheleva

Plymouth Rock Assurance Corporation
Boston, MA 02111, USA

Abstract

We propose the use of a state space search algorithm of the discrete-time recurrent neural network to learn the short-term foreign exchange rates. By searching in the neighborhood of the target trajectory in the state space, the algorithm performs nonlinear optimization learning process to provide the best feasible solution for the nonlinear least square problem. The convergence analysis shows that the convergence of the algorithm to the desired solution is guaranteed. The stability properties of the algorithm are also discussed. The empirical results show that our method is simple and effectively in learning the short-term foreign exchange rates and is applicable to other applications.

Mathematics Subject Classification: 68T05, 68Q32, 91B26

Keywords: State space search, discrete recurrent neural networks, absolutely stable, foreign exchange rates

1 Introduction

Foreign exchange rates are one of the most important economic indices in the international monetary markets. For years, studying foreign exchange rates pose many theoretical and experimental challenges that has received much attention from trading and academic communities. The technical analysis and techniques are popular for studying the short-term foreign exchange markets because the technical indicators have been developed to chart patterns in capturing the trends of price movements. However, most conventional econometric models and traditional statistical models are not able to forecast exchange rates with significantly higher accuracy than a naive random walk model (see [18], [8]). Since neural networks are trainable analytic tools that attempt to mimic information processing patterns in the brain, they can be used effectively to automate both routine and ad hoc financial analysis tasks, such as financial and economical forecasting. Recent evident has shown that the neural network forecasting models have proven, with empirical evidence (see [3], [8], [12] and [18]), are applicable to the forecasting of exchange rates effectively. Moreover, recurrent neural networks (RNNs) have richer dynamical structures and similar to a linear time series model with moving average term. Hence RNNs can learn extremely complex temporal patterns to yield good results. Because recurrent architecture proves to be superior to the windowing technique of overlapping snapshots of data. The short-term memory retains features of the input series relevant to the forecasting task and captures the network's prior activation history, which can provide beneficial foresee ability ([12]).

In this paper, we first focus on the development of a state space search algorithm for the proposed discrete-time recurrent neural network (RNN) model. The idea is, instead of moving in the parameter space of the connection weight matrices, we search in the neighborhood of the desired trajectory in the state space for each iteration. Our approach provides the best feasible solution for the nonlinear optimization problem. Unlike the conventional gradient methods, there is no computation of the partial derivatives along the target trajectory in our method. Hence, the state space search algorithm is fast and accurate. The convergence analysis shows that it guarantees the convergence to the desired solution. The rates of asymptotic convergence are provided and the stability properties are discussed in the analysis. In our empirical examples, data consist of the daily exchange rates of six major currencies, Euro (EUR), Sterling (GBP), Swiss Franc (CHF), Australian dollar (AUD), Canadian dollar (CAD) and Japanese Yen (JPY) compared to the U.S. dollar from the beginning of May 2003 to the end of September 2005. The examples illustrate that it is possible to predict the opening exchange rates using the rates from the previous day. The relationship is obviously nonlinear which neural networks could provide flexible models. Therefore, the model is substantially appropriate for the application of learning the short-term foreign exchange rates.

The organization of the paper is as follows. In Section 2, we first develop the state space search algorithm of the discrete-time RNNs, then we present the theory behind the stability analysis, and provide the convergence analysis for the proposed algorithm. In Section 3, some examples are used to illustrate how the state space search algorithm of the discrete-time RNNs can be used to learn the short-term foreign exchange rates. The relative merits of the method is discussed in the concluding Section 4.

2 The State Space Search Algorithm of the Discrete-Time RNNs

Consider the discrete-time model of the RNN described by a nonlinear system of the form

$$y_i(t+1) = (1 - h_i a_i) y_i(t) + h_i b_i \sigma(\sum_{j=1}^{n} w_{ij} y_j(t) + \theta_i), \qquad (2.1)$$

$$i = 1, 2, 3, ...n,$$

where y_i represents the internal state of the ith neuron. $W = [w_{ij}]_{n \times n}$ is the synaptic connection weight matrix. $A =$diag$[a_1, a_2, ..., a_n]$, B=diag$[b_1, b_2, ..., b_n]$ and $H =$diag$[h_1, h_2, ..., h_n]$ are diagonal matrices with *positive diagonal entries.* σ is a neuronal activation function that is bounded, differentiable and monotonic increasing on [-1, 1]. We assume that $\sigma(z) = tanh(z)$, which is the symmetric sigmoid logistic function. $\theta = [\theta_1, \theta_2, ..., \theta_n]^T$ is the input bias or threshold vector of the system.

System (2.1) is an Euler approximation of the Leaky integrator RNN model of the system of nonlinear equations

$$\frac{dy_i}{dt} = -a_i y_i + b_i \sigma(\sum_{j=1}^{n} w_{ij} y_i + \theta_i), \qquad (2.2)$$

$$i = 1, 2, 3, ...n,$$

and h_i of (2.1) is the step size in approximating the derivative

$$\frac{dy_i}{dt} \approx \frac{y_i(t+1) - y_i(t)}{h_i}$$

for each $i = 1, 2, ..., n$. In other words, system (2.1) is a numerical discrettization of the continue-time model (2.2) ([6], [?], [7], [8]). It can be shown ([9]) that there exists at least one equilibrium point of the (2.2) and the set of solutions of (2.2) is a positive invariant and attractive set. Although the continuous-time model (2.2) and its numerical discretization of model (2.1) need not share the same dynamical behavior, system (2.1) will inherit the same dynamics of system (2.2) when $h_i \to 0$ (see [16]).

To see the connection of system (2.1) with the forecasting models, we rewrite (2.1) in matrix form,

$$y(t+1) = (I - HA)y(t) + HB\sigma(Wy(t) + \theta). \tag{2.3}$$

If we let $A = B = I_{n\times n}$ the identity matrix, $H = 0_{n\times n}$ the zero matrix, then system (2.3) becomes

$$y(t+1) = y(t), \tag{2.4}$$

which represents the last-value forecasting of $y(t+1)$. Hence, it is the usual naive model in forecasting ([8]). On the other hand, if $A = B = H = I_{n\times n}$, we have

$$y(t+1) = \sigma(Wy + \theta), \tag{2.5}$$

which is the nonlinear regression model since σ is nonlinear. These models are used to capture the behavior of the linear internal mechanism of the financial market ([8]). Therefore, we may consider system (2.3) as the generalized convex combination of two forecasting models (2.4) and (2.5) for $0 < a_i h_i < 1, i = 1, 2, ..., n$. The positive entries of the diagonal matrix H of (2.3) can also be used to represent the different economic cycles. It implies that if we choose appropriate $h_i' s$, the discrete-time RNN model (2.1) and (2.3) should be at least as good as any time series models. After Rumelhart, Hinton and Williams ([13]) introduced in 1986 the learning algorithms *'error back-propagation'* and the *'delta rule'*, it is easier to find a temporal model for short-term prediction. This adaptive property is very important for real-time modelling in stocks and foreign exchange markets. Since the discrete-time RNN model is easily implemented in digital hardware and easily simulated in computers, it presents advantages over the continuous-time model. We focus, in this section, on the discrete-time model (2.1) to develop a new robust learning algorithm – the state space search algorithm.

Before introducing the state space search algorithm, we first discuss some stability properties of the discrete-time RNN model (2.3) and provide the convergence analysis afterward. The stability and bifurcation properties of the discrete-time RNNs were analyzed by Wang and Blum [16], Li [**?**], Jin, Nikiforuk and Gupta [5]. We provide here the *absolute stability analysis* for system (2.3) in the spirit of [5].

Definition 1. A point $x^* \in [a,b]^n \subset R^n$ is defined as an *equilibrium point* of the discrete-time RNN system (2.3) if

$$x^* = (I - HA)x^* + HB\sigma(Wx^* + \theta). \tag{2.6}$$

Definition 2. The function $f(y) = (I - HA)y + HB\sigma(Wy + \theta)$ has only *asymptotically stable equilibrium point* if all the eigenvalues of the Jacobian are inside the unit circle for all the states y, a given connection weight matrix W, and the input θ.

Definition 3. If the discrete-time RNN system (2.3) has only asymptotically stable equilibrium points for a given connection weight matrix W and θ, then the system (2.3) (or 2.1) is said to be *absolutely stable.*

Remarks. It is important to notice that the asymptotic stability may depend upon the input θ, whereas the absolute stability does not depend upon the input θ. This is the fundamental difference between the asymptotic stability and the absolute stability.

Ostrowski's theorem [4]). Let $W = [w_{ij}]_{n\times n}$ be a complex matrix, $\gamma \in [0,1]$ be given, and R_i and C_i denote the delete row and delete column sums of W as following, respectively,

$$R_i = \sum_{j=1, j\neq i}^{n} |w_{ij}|,$$

$$C_i = \sum_{j=1, j\neq i}^{n} |w_{ji}|.$$

All the eigenvalues of W are then located in the union of n closed discs in the complex plane with centers w_{ii} and radii $r_i = {R_i}^{\gamma}{C_i}^{1-\gamma}, i = 1, 2, ..., n$.

Now we show in the following two theorems on absolute stability properties of the system (2.3).

Theorem 1. Let $\beta = \max\{\frac{b_i}{a_i} | i = 1, 2, ..., n\}$, then there exist at least one equilibrium point $y^* \in [-\beta, \beta]^n$ of (2.3) such that

$$y^* = A^{-1}B\sigma(Wy^* + \theta) \tag{2.7}$$

for each θ.

Proof. The result follows by applying the theorem of Jin, Nikiforuk and Gupta (Theorem 1, [5]). That is, we rewrite the equation,

$$y_i(t+1) = (1-h_ia_i)y_i(t) + h_ib_i\sigma(\sum_{j=1}^{n} w_{ij}y_j(t) + \theta_i) = f_i(y) \tag{2.8}$$

in an equivalent form and define

$$\begin{aligned} g_i(y) = \frac{b_i}{a_i}\sigma(\sum_{j=1}^{n} w_{ij}y_j + \theta_i) = y_i, \\ i = 1, 2, ..., n. \end{aligned} \tag{2.9}$$

Then the fixed points of $g(y)$ are also the fixed points of $f(y)$. For $y \in [-\beta, \beta]^n$, since $|\sigma(z)| = |\tanh(z)| \leq 1$, we have

$$\begin{aligned} y_i = g_i(y) \equiv |\frac{b_i}{a_i}||\sigma(\sum_{j=1}^{n} w_{ij}y_j + \theta_i)| \leq \frac{b_i}{a_i}, \\ i = 1, 2, ..., n. \end{aligned} \tag{2.10}$$

That implies $g(y) \in [-\beta, \beta]^n$. Since function σ is continuous, we can conclude that for any given θ and the connection weight matrix W, $g(y) : [-\beta, \beta]^n \rightarrow [-\beta, \beta]^n$ is a continuous function. By the Brouwer's fixed point theorem ([5]), $g(y)$ has a fixed point $y^* \in [-\beta, \beta]^n$. Since y^* is also a fixed point of $f(y)$, therefore, y^* is an equilibrium point of $y(t+1) = (I - HA)y(t) + HB\sigma(Wy(t) + \theta)$ with

$$y^* = A^{-1}B\sigma(Wy^* + \theta).$$

Theorem 2. If $0 < h_ia_i < 2$ for $i = 1, 2, ..., n$, and if the connection weight matrix $W = [w_{ij}]$ satisfies the inequalities

$$\sum_{j=1}^{n} w_{ij} < \delta_i = \frac{1}{h_ib_i}(1 - |1 - h_ia_i|) \tag{2.11}$$

or

$$|w_{jj}| + \frac{1}{|h_ib_i|}\sum_{i=1, i\neq j} |h_ib_iw_{ij}| < \delta_i = \frac{1}{h_ib_i}(1 - |1 - h_ia_i|) \tag{2.12}$$

for each $i = 1, 2, ..., n$, then the discrete-time RNN model (2.1) is absolutely stable.

Proof. Set $\gamma = 1$ and $\gamma = 0$ in the Ostrowski's theorem, then the results (2.11) and (2.12) can be obtained by using Gerschgorin's theorem ([4]) to apply in Ostrowski's theorem.

Remarks From Theorem 2, we notice that inequality (2.11) implies that the solution space of the connection weight matrix W forms n open convex hyper cones in n-dimensional space. Moreover, $\delta_i = \frac{1}{h_i b_i}(1 - |1 - h_i a_i|) \to \infty$ as $h_i \to 0$. Since the discrete-time model (2.1) and the continuous-time model (2.2) will share the same dynamical behavior as $h_i \to 0$, we can conclude that as $h_i \to 0$ both (2.1) and (2.2) are absolutely stable. It is known that for an absolutely stable neural network model, the system state will converge to one of the asymptotically stable equilibrium points regards of the initial state.

We now turn to develop the new learning algorithm–the state space search algorithm for the discrete-time RNN (2.1). In neural networks, learning is a process of changing the network parameters so that the system output will approach to the target trajectory. One of the classic learning algorithm - recurrent back propagation algorithm was derived by Williams and Zipser ([17]) in 1989 for the discrete-time RNNs. More than a decade, the researchers in the field have applied many numerical optimization methods and techniques, such as gradient decent, conjugate gradient and Newton's method etc. as learning algorithms, to minimizing the error functions of the least square problems. It is clear that any changes in the parameter w_{ij} of (2.1) may accumulate large derivations to the whole trajectory. Because of the necessity to compute the partial derivatives along the trajectory, learning algorithms based on gradient decent methods are complicated and time consuming for RNNs. Li and his co-workers presented in [7], [8] some new learning algorithms that takes no derivatives. Our results further develop and amplify the previous results in [8]. In addition to the experimental results, the convergence analysis is also presented to provide the theoretical justifications for the state space search algorithm.

Before we proceed to describe the state space search algorithm, we establish the following definitions. All the norms used in this paper are either L_2 or l_2.

Definition 4. An *x-convex set* is a union of line segments each with one end point at $x \in R^n$, and containing any two segments (unless they go in opposite directions) together with the entire triangle with two sides of these segments. The class of all *x-convex sets* in $Z \subset R^n$ will be denoted by $C_x(Z)$.

Definition 5. If a sequence of vectors $\{x_k\}_k$ converges to a limit point x^*, then *the asymptotic convergence rate (order of convergence)* of $\{x_k\}_k$ is defined as the supremum of the nonnegative number p satisfying

$$0 \leq \limsup_{k \to \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^p} < \infty.$$

A sequence $\{x^*\}_k$ is said to be *convergence linearly* to x^* if

$$0 \leq \lim_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = \beta < 1.$$

Remarks $C_x(Z)$ is compact in R^n if Z is closed and bounded. For the asymptotic convergence rate, larger values of the order p, imply more rapid convergence.

Definition 6. A set A^+ is called attainable of x if $s \in A^+$, there exist some h such that x is reachable through s in some neighborhood of x.

For a given trajectory $x(t) \in R^n$, we use (2.1) to approximate $x(t)$ with the error function E defined by

$$\begin{aligned} E(W, h, \theta) &= \parallel y(t, W, h, \theta) - x(t) \parallel^2 \\ &= \sum_{i=1}^{n} \sum_{t=1}^{m} [y_i(t, W, h, \theta) - x_i(t)]^2 \end{aligned} \tag{2.13}$$

for some positive integer m. Our goal is to minimize the error between the target trajectory $x(t)$ (in our case, $x(t)$ is the trajectory of the foreign exchange rates) and the system output $y(t)$. To simplify the analysis, we let $h = h_i, i = 1, 2, ..., n$. We wish to find the optimal connection weight matrix $W^* = [w_{ij}]_{n\times n}$ such that, for a fixed h and θ, W^* satisfies the nonlinear least square problem

$$E(W^*) = \min_{W^+} \sum_{i=1}^{n} \sum_{t=1}^{m} [y_i(t, W^+) - x_i(t)]^2, \tag{2.14}$$

where W^+ is a feasible solution defined by

$$W^+ = \sigma^{-1}[\frac{1}{h}(C_{t+1} - D_t) - D_t]{D_t}^T(D_t{D_t}^T)^{-1}, \tag{2.15}$$

where matrices $C_{t+1} = [y(t+1)...y(2)]$ and $D_t = [y(t)...y(1)]$.

If the network is exactly capable, that is, $E = E(W^*) = 0$, the optimal solution W^* of (2.14) can be reached using a simple optimization strategy by vary h and repeated solve W^+ by (2.15). Then we have $W^* = W^+$. If $W \neq 0$, equation (2.14) has no exact solution, which implies x is not reachable for any given h.

The state space search algorithm is developed to provide the best feasible solution of (2.14). The idea is that instead of moving in the parameter space of W, we search the class of the x-convex set $C_x(A^+)$ in the state space for each iteration, were A^+ is the set of attainable points of x in R^n. After obtaining first W^+ from (2.15), we use (2.1) to generate $y(W^+)$. In the next iteration, instead of approximating x, we use $y(W^+)$ to approximate the new trajectory x^+ defined by a convex combination

$$\begin{gathered} x^+ = \alpha_i x + (1-\alpha_i)y(W^+) \\ 0 < \alpha_i < 1, \quad \alpha_i \downarrow 0. \end{gathered} \tag{2.16}$$

In fact, we store the best solution for each α_i. Notice that x^+ may not attainable even though $y(W^+)$ is attainable. By the continuity of W, there exist some ${\alpha_1}^*$ such that

$$X_1 = {\alpha_1}^* x + (1-{\alpha_1}^*)y(W_1),$$

where X_1 is attainable in $C_x(A^+)$, $0 < {\alpha_1}^* < 1$, and

$$E(W_1) = \min_{\alpha_i}\{E(W_{\alpha_i})\} \leq E(W^+).$$

Repeated the procedures, we obtain a sequence of attainable points $\{X_k\}_k = \{y(W_{k+1})\}_k$ defined by

$$\begin{array}{lll} x, & y(W_1), & X_1 = {\alpha_1}^* x + (1-{\alpha_1}^*)y(W_1), \\ X_1 = y(W_2), & X_2 = {\alpha_2}^* x + (1-{\alpha_2}^*)y(W_2), & \\ X_2 = y(W_3), & X_3 = {\alpha_3}^* x + (1-{\alpha_3}^*)y(W_3), & \\ ... & ..., & ... \\ ... & ..., & ... \\ X_k = y(W_k), & X_{k+1} = {\alpha_{k+1}}^* x + (1-{\alpha_{k+1}}^*)y(W_{k+1}), & \\ X_{k+1} & ..., & .... \end{array} \tag{2.17}$$

To summarize the above procedures, we obtain, for each k,

$$X_{k+1} = {\alpha_{k+1}}^* x + (1-{\alpha_{k+1}}^*)y(W_{k+1}), \tag{2.18}$$

where

$$\begin{aligned} & y(W_{k+1}) = X_k, \\ & 0 < {\alpha_k}^* < 1, \quad \{{\alpha_k}^*\} \subset \{\alpha_i\}, \\ & E(W_{k+1}) \leq E(W_k), \\ & {\alpha_{k+1}}^* = {\alpha_k}^* \quad \text{if} \quad E(W_{k+1}) < E(W_k). \end{aligned} \tag{2.19}$$

The learning process stops if we obtain a solution W_N such that the error $E(W_N)$ is less than a prescribed tolerance rather than the true optimal solution W^*.

For the convergence and the rates of convergence of the state space search algorithm (2.18), we have the following Theorems. in

Theorem 3. There exists a limit point X^* such that the attainable sequence $\{X_k\}_k$ of x converges, that is, $\lim_{k\to\infty} X_k = X^*$. Moreover,
(i) if $\sum_i^\infty {\alpha_i}^* = M < \infty$, then $X^* = x + M(y(W_1) - x)$;
(ii) if $\sum_i^\infty {\alpha_i}^* = \infty$ for nonconstant sequence $\{{\alpha_k}^*\}$, then $X^* = x$.

Proof. Let A^* be the set of all attainable points of x, then the attainable sequence $\{X_k\}_k \in C_x(A^*)$, where $C_x(A^*)$ is defined by Definition 4. This implies that $\{X_k\}_k$ is bounded since A^* is compact in R^n. Since $0 < {\alpha_k}^* < 1$ for all k, we have

$$\begin{aligned} \|x - X_{k+1}\| &= \|x - {\alpha_{k+1}}^* x - (1 - {\alpha_{k+1}}^*) y(W_{k+1})\| \\ &= (1 - {\alpha_{k+1}}^*)\|x - y(W_{k+1})\| \\ &\leq \|x - X_k\| \qquad (2.20) \\ &\dots \\ &\leq \|x - X_1\|, \end{aligned}$$

the sequence $\{x - X_k\}_k$ decays monotonically and hence converges. Thus, there exists a limit point X^* such that the attainable sequence $\{X_k\}_k$ converges with

$$\lim_{k\to\infty} X_k = X^*.$$

In order to find the explicit expression of the limit point X^* of $\{X_k\}$, we have the following lemma.

Lemma A. For each positive integer k and $0 < {\alpha_k}^* < 1$, we have

$$1 - [\sum_{j=1}^k \prod_{i=j+1}^k (1 - {\alpha_i}^*){\alpha_j}^*] = \prod_{i=1}^k (1 - {\alpha_i}^*).$$

Moreover,

(a) if $\sum_i^\infty {\alpha_i}^* < \infty$, then there exists a constant M with $0 < M < 1$ such that

$$\begin{aligned} &\lim_{k\to\infty} \prod_{i=1}^k (1 - {\alpha_i}^*) = M, \\ &\lim_{k\to\infty} [\sum_{j=1}^k \prod_{i=j+1}^k (1 - {\alpha_i}^*){\alpha_j}^*] = 1 - M; \end{aligned} \qquad (2.21)$$

(b) if $\sum_i^\infty {\alpha_i}^* = \infty$, then

$$\begin{aligned} &\lim_{k\to\infty} \prod_{i=1}^k (1 - {\alpha_i}^*) = 0, \\ &\lim_{k\to\infty} [\sum_{j=1}^k \prod_{i=j+1}^k (1 - {\alpha_i}^*){\alpha_j}^*] = 1. \end{aligned} \qquad (2.22)$$

Proof of Lemma A. For all k and $0 < \alpha_k{}^* < 1$, let $\beta_k = [\sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \alpha_i{}^*)\alpha_j{}^*]$ with $\beta_1 = \alpha_1{}^*$, then

$$\beta_k = (1 - \alpha_k{}^*)\beta_{k-1} + \alpha_k{}^*,$$
$$(1 - \beta_k) = (1 - \alpha_k{}^*)(1 - \beta_{k-1}),$$

where $1 - \beta_1 = 1 - \alpha_1{}^*$. Hence

$$1 - \beta_k = \prod_{i=1}^{k} (1 - \alpha_i{}^*),$$

that is,

$$1 - [\sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \alpha_i{}^*)\alpha_j{}^*] = \prod_{i=1}^{k} (1 - \alpha_i{}^*). \tag{2.23}$$

To obtain the results in (a) and (b), we notice that $1 - \alpha_k{}^* < e^{-\alpha_k{}^*}$ for all $0 < \alpha_k{}^* < 1$, which implies

$$0 < \prod_{i=1}^{k} (1 - \alpha_i{}^*) = \exp\{-\sum_{i=1}^{k} \alpha_i{}^*\}. \tag{2.24}$$

Therefore, (a) if $\sum_i^\infty \alpha_i{}^* < \infty$, then there exists a constant M with $0 < M < 1$ such that

$$\lim_{k \to \infty} \prod_{i=1}^{k} (1 - \alpha_i{}^*) = M,$$
$$\lim_{k \to \infty} [\sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \alpha_i{}^*)\alpha_j{}^*] = 1 - M$$

by (2.24), (2.23) and (2.25). Hence, (b) if $\sum_i^\infty \alpha_i{}^* = \infty$, from (2.24) and (a), we have

$$\begin{aligned} &\lim_{k \to \infty} \prod_{i=1}^{k} (1 - \alpha_i{}^*) = 0, \\ &\lim_{k \to \infty} [\sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \alpha_i{}^*)\alpha_j{}^*] = 1. \end{aligned} \tag{2.25}$$

The proof of Lemma A is completed.

Now we use Lemma A to finish the proof of Theorem 3, that is, to find the limit point X^*. To further the analysis, we rewrite (2.18) as follows,

$$X_k =[\sum_{j=1}^{k} \prod_{i=j+1}^{k} (1-{\alpha_i}^*){\alpha_j}^*]x + [\prod_{i=1}^{k}(1-{\alpha_i}^*)]y(W_1). \tag{2.26}$$

Then we have

(i) If $\sum_k^\infty {\alpha_k}^* < \infty$, this implies $\lim_{k\to\infty} {\alpha_i}^* = 0$. Then by (2.26) and Lemma 6, we have $X^* - x = M(y(W_1) - x)$; while

(ii) if $\sum_k^\infty {\alpha_k}^* = \infty$ for non-constant sequence $\{{\alpha_k}^*\}$ with $\lim_{k\to\infty} {\alpha_k}^* = a \neq 0$, we have $X^* = x$ by (2.26) and (2.22). The proof of Theorem 3 is completed.

Remarks. For any real sequence $\{\alpha_j\}_j \subset R$, it is clear that one of the conditions (i) and (ii) must be hold. Theorem 3 shows that the state space search algorithm of the discrete-time RNN is a fast learning algorithm that provides the best feasible solution for the least square problem of (2.1). This follows from the fact that the limit point X^* of $\{X_k\}_k$ will lead us to obtain the corresponding best feasible solution W^* of (2.1). Meanwhile, the error sequence $\|E(W_k)\| \downarrow 0$ when $\lim_{k\to\infty} X_k = x$. An example of case (i) of Theorem 5 is ${\alpha_k}^* = \lambda\rho^k$ with $\rho < 1$, and $\lambda \in (0,1]$. We notice that $\lim_{k\to\infty} \prod_{i=1}^k (1-\lambda\rho^k) = \exp[-\sum_{k=1}^\infty \frac{1}{k}\frac{\lambda^k}{(1-\rho^k)}] < \exp[\frac{-\lambda}{1-\rho}] < \infty$ for any $\rho < 1$ (see [19]). In the next theorem, we discuss the asymptotic rates of convergence of $\{X_k\}_k$. Our result shows that the rates of convergence of $\{X_k\}$ depends on $\{{\alpha_k}^*\}$.

Theorem 4. If $\lim_{k\to\infty} {\alpha_k}^*$ exists, then there exists a nonnegative number $p_0 \in (0,1]$ such that the order of asymptotic convergence rate of $\{X_k\}_k$ is p_0. If $\sum_i^\infty {\alpha_i}^* = \infty$ for non-constant sequence $\{{\alpha_k}^*\}$, then the order of the asymptotic convergence rate of $\{X_k\}_k$ is linear.

Proof. Notice that

$$\begin{aligned} 0 &\leq \frac{\|X_{k+1} - X^*\|}{\|X_k - X^*\|^p} \\ &= \frac{\|{\alpha_{k+1}}^* x + (1-{\alpha_{k+1}}^*)X_k - X^*\|}{\|X_k - X^*\|^p} \\ &= \frac{\|{\alpha_{k+1}}^*(x - X^*) + (1-{\alpha_{k+1}}^*)(X_k - X^*)\|}{\|X_k - X^*\|^p}. \end{aligned} \tag{2.27}$$

By Lemma A and (2.26), we have

$$\begin{aligned} X_k - X^* =& [\sum_{j=1}^{k} \prod_{i=j+1}^{k} (1-{\alpha_i}^*){\alpha_j}^*]x + [\prod_{i=1}^{k}(1-{\alpha_i}^*)]y(W_1) - X^*, \\ &= (x - X^*) + [\prod_{i=1}^{k}(1-{\alpha_i}^*)](y(W_1)-x). \end{aligned} \tag{2.28}$$

If $\lim_{k\to\infty} {\alpha_k}^*$ exists, we substitute (2.28) into (2.27), and by Lemma A, we have

$$\begin{aligned} 0 \leq & \frac{\|X_{k+1} - X^*\|}{\|X_k - X^*\|^p} \\ =& \frac{\|{\alpha_{k+1}}^*(x - X^*) + (1-{\alpha_{k+1}}^*)(X_k - X^*)\|}{\|X_k - X^*\|^p}. \\ =& \frac{\|(x-X^*) + (1-{\alpha_{k+1}}^*)[\prod_{i=1}^{k}(1-{\alpha_i}^*)(y(W_1)-x)]\|}{\|(x-X^*) + [\prod_{i=1}^{k}(1-{\alpha_i}^*)](y(W_1)-x)\|^p}. \end{aligned} \tag{2.29}$$

Hence, there exists a $p_0 \in (0,1]$ such that

$$0 \leq \limsup_{k\to\infty} \frac{\|X_{k+1} - X^*\|}{\|X_k - X^*\|^{p_0}} < \infty. \tag{2.30}$$

If $\sum_k^\infty {\alpha_k}^* = \infty$ for non-constant sequence $\{{\alpha_k}^*\}$ with $\lim_{k\to\infty} {\alpha_k}^* = a \neq 0$, we have $X^* = x$ by Theorem 5 and (2.22), which implies

$$\begin{aligned} 0 \leq & \lim_{k\to\infty} \frac{\|X_{k+1} - x\|}{\|X_k - x\|} \\ =& \lim_{k\to\infty} \frac{\|{\alpha_{k+1}}^* x + (1-{\alpha_{k+1}}^*)X_k - x\|}{\|X_k - x\|} \\ =& \lim_{k\to\infty} \frac{(1-{\alpha_{k+1}}^*)\|(X_k - x)\|}{\|X_k - x\|} < 1. \end{aligned} \tag{2.31}$$

Therefore, the order of convergence of $\{X_k\}$ is linear at this case. We finish the proof.

Remarks. For sequence $\{{\alpha_k}^*\}_k$ associated with the SSSA, we may choose $\{{\alpha_k}^*\}_k$ such that $\{{\alpha_k}^*\}_k \uparrow 1$, which implies $\lim_{k\to\infty} {\alpha_k}^*$ exists. If the order of asymptotic rate of convergence of sequence $\{{\alpha_k}^*\}_k$ is less or equal than the convergent rate of $\{\frac{1}{k}\}_k$, where k are positive integers, then $\sum_k^\infty {\alpha_k}^* = \infty$. Theorem 4 provides the asymptotic rate of convergence of $\{X_k\}_k$ of the SSSA. The Theorem 3 and Theorem 4 also imply that the asymptotic rates of convergence of $\{X_k\}$ of the SSSA depends on $\{\alpha_k{}^*\}_k$. Furthermore, $X_k \to X^*$ as $\alpha_k{}^* \uparrow 1$.

In the next two sections, we show how the state space search algorithm can be applied to learn the short-term foreign exchange rates.

3 Examples

Data Collection and Learning Process

Notice that daily quotes of foreign exchange rates are display as time series. As opposed to the traditional trading methods by human decisions, neural networks offer a simple but automatic system in the trading off foreign exchange markets. Historical data of the foreign exchange rates are available from many web sites in the form of daily averages. We chose to retrieve the data from OANDA.com. In our study, data consist of the daily exchange rates of six major currencies, Euro (EUR), Sterling (GBP), Swiss Franc (CHF), Australian dollar (AUD), Canadian dollar (CAD) and Japanese Yen (JPY) compared to the U.S. dollar from May 2003 to September 2005. We collected 801 data for each of the currencies.

We assume that there exist short-term trends in our foreign exchange series, and we could use neural network techniques to model the short-term trend movement of the foreign exchange rates and to make predictions. Our assumption is based on the result of Yao and Tan ([18]), which they show that statistically the foreign exchange series do not support the random walk hypothesis ([8]).

We use the first 701 days observations to train and validate the RNN model. After the training, we then used the resulting neural network parameters to make the out-of-sample forecasts the last 100 observations. Out-of-sample forecast errors are measured in order to judge how good our model is in terms of its prediction abilities. The learning dynamic used for the discrete-time RNN is equation (2.1).

Before the training process, the data needs to be transformed into an appropriate form for the networks. We employ a normalization for each component series in $x(t_i)$. Since the sigmoid chosen as activation function gives an output in the interval [-1, 1], it was necessary to normalize the data into this interval to avoid working near the asymptote of the sigmoid. This normalization is given by

$$x(t_i) = [\frac{y(t_i) - \min\{y(t_i)\}}{\text{Range of} \quad y(ti)} - 0.5] \times 1.90 \qquad (3.1)$$
$$t = 1, 2, 3, ..., 801, i = 1, 2, ..., 6.$$

as in as in ([8]). The data is then denormalized using the inverse of formula (3.1).

As we knew that this normalization will generally smooth out the extreme outliers, and guarantees $x(t_i)$ to lie between -0.95 and $+0.95$, and therefore, they are inside the range of the neural activation function, $\sigma(\cdot)$, defined in (2.1). In addition, this normalization process will facilitate the computational work in the state space learning process.

Instead of using the normalized raw data to feed the system for learning, we use the moving average series $z(t)$ of order 5, 10, 20, 50, 100 and 250, respectively, obtained from $x(t)$. For example, there is approximately 250 business days in a year, we may take a moving average of 250 terms, that is, $x_i(t) = \frac{1}{250}\sum_{j=t}^{t+249} z_i(j)$. In general, the moving average is defined as:

$$x_i(t) = \frac{1}{n}\sum_{j=t}^{t-n+1} z_i(j).$$

Also, since the ranges of the ratios varies for different currencies, we normalize the ranges of the exchange rates to ±0.8. The advantage of using moving averages t model the short-term trend in foreign exchange rates can be found in ([18]).

We apply the same learning process to the trajectory of the 5, 10, 20, 50, 100, 250 days moving averages of the exchange rate sequences. For simplicity, we set the external force θ to be the zero vector. The results of learning 700 daily exchange rates by the RNN model for AUD/US, CAD/US, CHF/US, EUR/US, GBP/US and YEN/US are plotted in Figures 1-6, respectively.

In this experiment, the optimal connection weight matrix W the in-sample training process is given by the following 6 by 6 matrix that from training 701 data and using the 250 days moving average:

We use the different optimal step sizes $h_i, i = 1, 2, 3, 4, 5, 6$ range from 0.0015001 to 0.0201 for each of the currencies. The minimum least square errors of learning and out-of-sample predictions for each of the six currencies are displayed in the following table:

$Currencies$	$errors(learning)$	$errors(out-of-sample\,forecasting)$
EUR/US	0.0252	0.7252
AUD/US	$10^{(-3)}$	1.8844
CAD/US	$10^{(-3)}$	1.5079
CHF/US	$10^{(-3)}$	1.5427
YEN/US	$10^{(-2)}$	1.9828
GBP/US	$10^{(-4)}$	0.57779

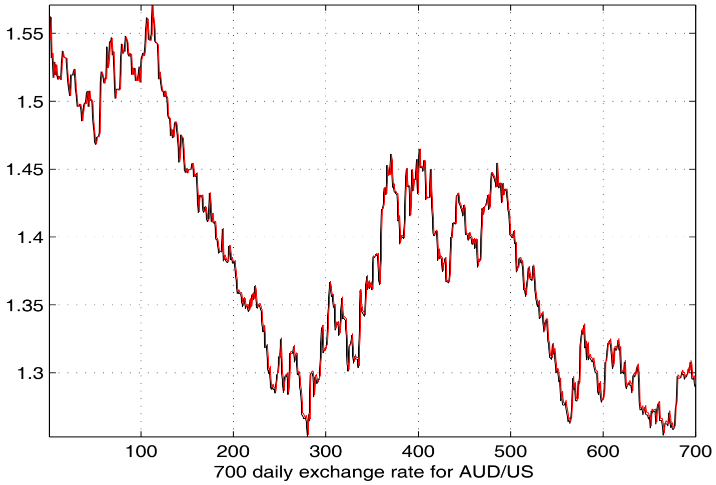

Figure 1: Learning 700 Daily Exchange Rate for AUD/US

We used 400 iterations. Software MATLAB was used to do all computations and plotted graphs.

From Figures 1-6, we notice that in-sample learning values generated by the RNN model follow very closely with the actual observations for each of the six exchange rates. While the outcomes of the RNN approximations for the six foreign exchange rates are plotted in Figures 7-13. The x-axis is scaled from 0 to 100, which in fact represents the prediction of the last hundred original observations using the RNN model. The y-axis scale varies for each currency and was automatically adjusted depending on the particular data range.

From these figures, we notice that the RNN approximations of the six foreign exchange rates are not appearing to be visually accurate comparing with the original observations. However, the errors are between 0.5779 and 1.5785, which are reasonably good.

Overall, the empirical results demonstrate that the RNN approximation was reasonably good for all of the six currencies. However, it seems that the RNN model worked the best with CHF/US and the worst with YEN/US and EURO/US. The different type of output for each currency is due to the fact that the recurrent neural network performs differently if we change even slightly either some of the parameters or the data set. The RNN models are extremely sensitive to their input. That is why, in reality we are not concerned with the exact predicted values of the short-term foreign exchange rates. Rather, we examine the pattern of the graphs. We want to know if the exchange rate

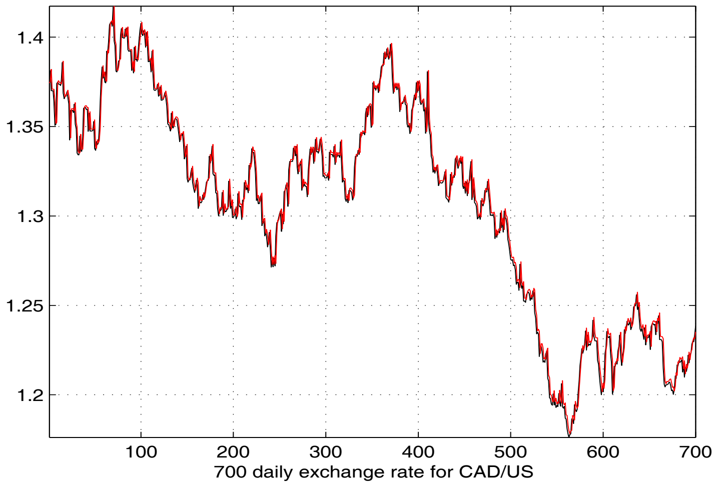

Figure 2: Learning 700 Daily Exchange Rate for CAD/US

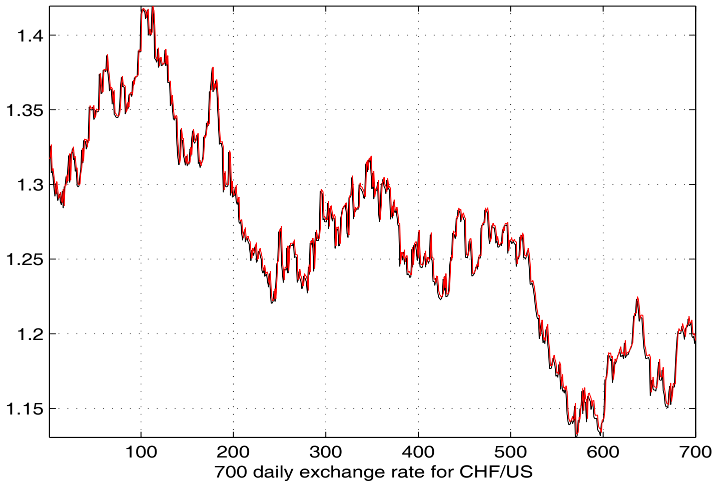

Figure 3: Learning 700 Daily Exchange Rate for CHF/US

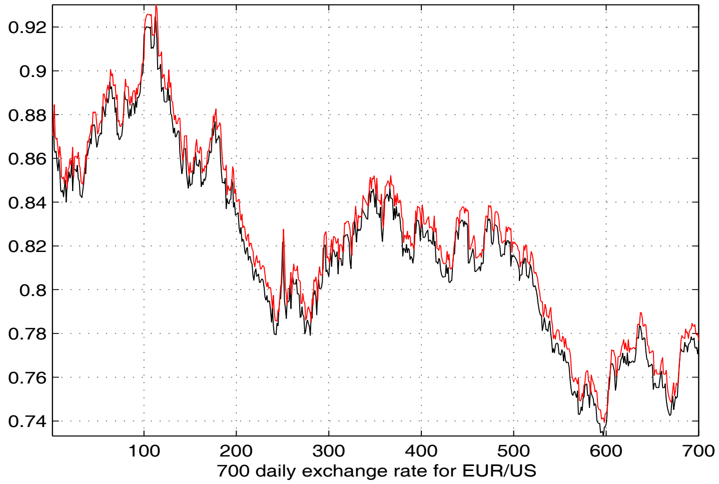

Figure 4: Learning 700 Daily Exchange Rate for EUR/US

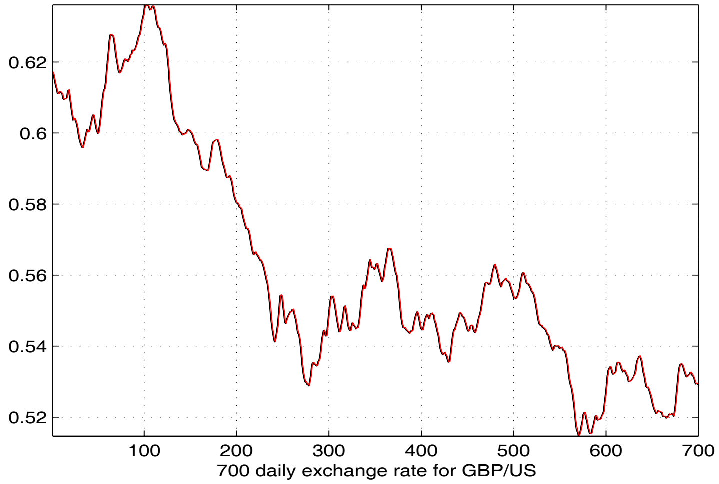

Figure 5: Learning 700 Daily Exchange Rate for GBP/US labelFigure 5

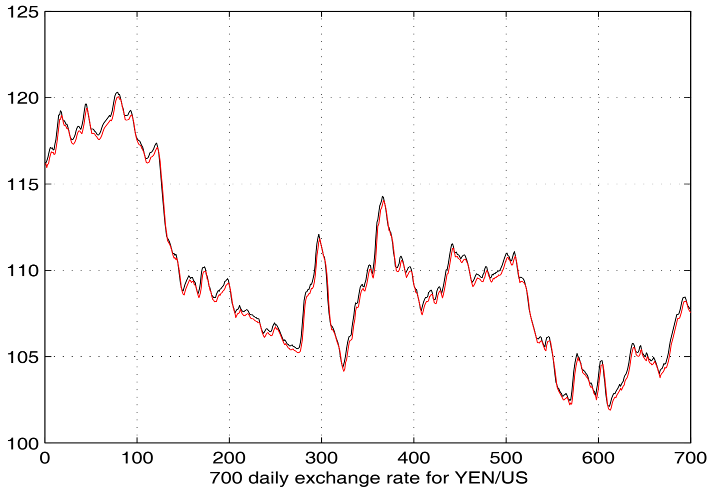

Figure 6: Learning 700 Daily Exchange Rate for YEN/US

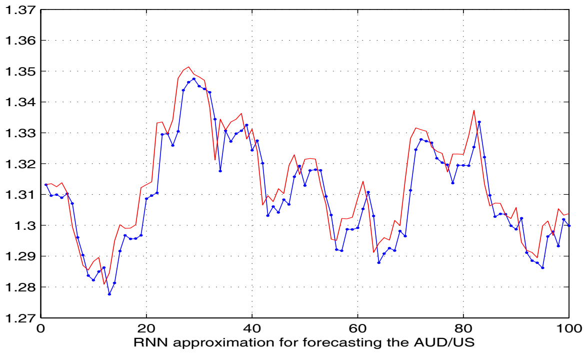

Figure 7: The RNN Approximation for Forecasting the Exchange Rate AUD/US

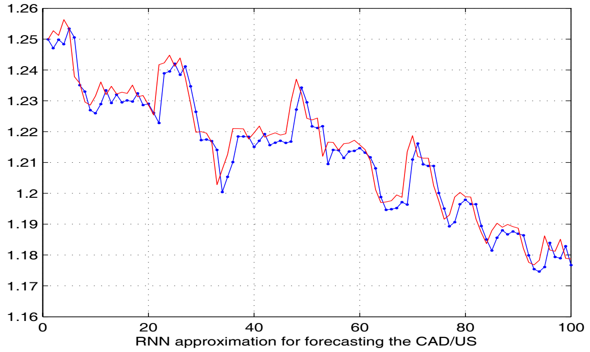

Figure 8: The RNN Approximation for Forecasting the Exchange Rate CAD/US

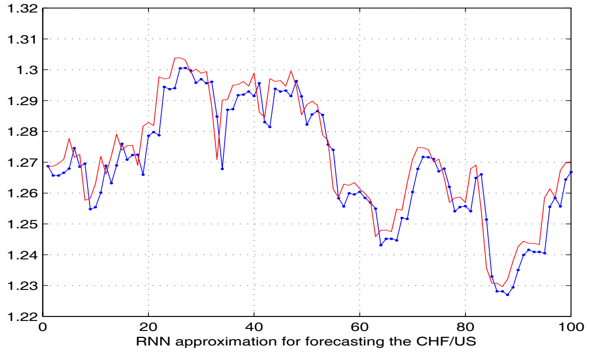

Figure 9: The RNN Approximation for Forecasting the Exchange Rate CHF/US

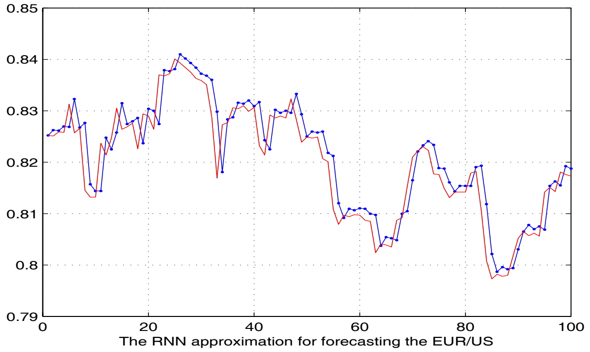

Figure 10: The RNN Approximation for Forecasting the Exchange Rate EUR/US

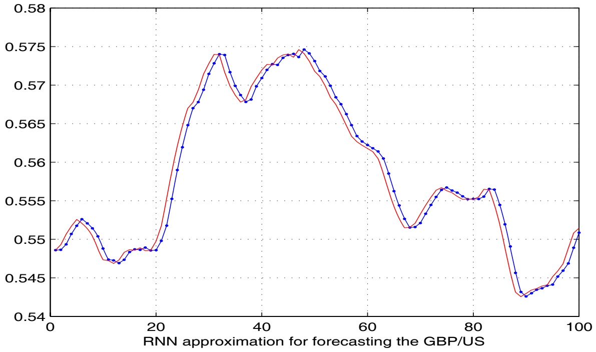

Figure 11: The RNN Approximation for Forecasting the Exchange Rate GBP/US

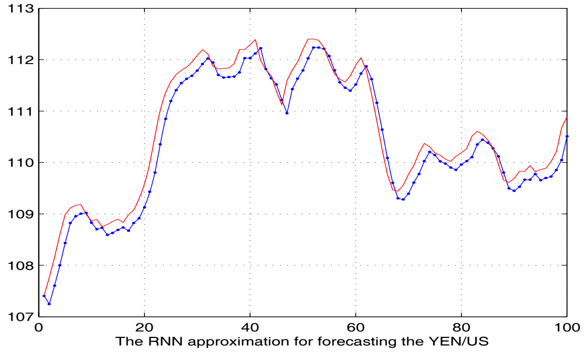

Figure 12: The RNN Approximation for Forecasting the Exchange Rate YEN/US

trend is going up or down. Our results are satisfactory as long as the RNN approximation curve tags along with the raw data curve reasonably close.

4 Conclusion and the Final Remarks

We propose the use of a state space search algorithm of the discrete-time recurrent neural network to learn the short-term foreign exchange rates. We study and discuss the stability properties and the asymptotic convergence rates of the state space search algorithm. Six major foreign currencies exchange rates (1) Euro/US, (2) Yen/US, (3) GBP/US, (4) CHF/US, (5) AUD/US and (6) CAD/US were chosen to demonstrate how effectively the method works. Our results show that RNNs are a promising tool for learning the short-term exchange rates. Several factors significantly impact the accuracy of the neural network forecasts. These factors include selection of input variables, preparing data, RNN's architecture. There is no consensus on these factors. In different cases, various decisions have their own effectiveness. There is no formal systematic model building approach ([3], [12]).

Model uncertainty comes from three main sources: model structure, parameters estimation and data. The nonlinear nature of RNNs may cause more uncertainties in model building. This learning and generalization tradeoff has been extensive, and is still an active research topic in the field. To improve generalization performance we may need to go beyond the model selection methods.

As we mentioned earlier, the network learns by example, which is the sole reason for the different approximation output for the six currencies. We decided to use the moving average of 250 and it is not surprising that it worked better with some of the currencies. The explanation for the better RNN approximations of the British Pound (GBP), comparing with the EURO (EUR), Australia Dollar (AUD), Canadian Dollar (CAD) and Swiss Franc (CHF), Japanese Yen (YEN),could be that there might be a relation between the time period and the segmentation on the results for a particular currency. However, all of the empirical results demonstrate reasonably good predictability of using RNN model for short-term foreign exchange rates.

Overall, our results confirm the reliability and potential of the RNN models as a forecasting tool. However, while the RNN models offer a promising alternative to traditional techniques, they suffer from a number of limitations. One of the major disadvantages is the inability to explain their reasoning. In addition, statistical inference techniques such as significance testing cannot always be applied, resulting in a reliance on a heuristic approach. Despite their limitations, RNNs can add value to the forecasting process. The complexity of RNN models suggests that they are capable of superior forecasts.

5 Acknowledgements

This research was supported by the Hong Kong Polytechnic University Research Projects A-PE95 and A-PA4V.

References

[1] G. E. P. Box and G. M. Jenkins, Time Series Analysis, Forecasting and Control, San-Francisco, CA: Holden-Day, 1970.

[2] R. F. Engle, Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation, *Econometrica*, 50, (1982) 987-1007.

[3] 2. W. Huang, K. K. Lai, Y. Nakamori and S. Wang, Forecasting foreign exchange rates with artificial neural networks: a review, *Int. J. Information Technology & Decision Making*, Vol. 3, No. 1, (2004), 145-165.

[4] R. A. Horn and C. A. Johnson, Matrix Analysis, Cambridge University Press, Cambridge, U.K., 1985.

[5] L. Jin, N. Nikifork, and M. M. Gupta, Absolute stability conditions for discrete-time neural networks, *IEEE Tranc. Neural Networks*, vol. 5, no. 6, (1994) 954-964.

[6] X. B. Liang and J. Wang, A recurrent neural network for nonlinear optimization with a continuously differentiable objective function and bound constraints, *IEEE Tranc. Neural Networks*, vol. 11, no. 6, (2000) 1251-1262.

[7] L. K. Li, Learning sunspot series dynamics by recurrent neural networks, Advances in Data Mining and Modeling (W. K. Ching and K. P. Ng, Eds), World Science, (2003) 107-115.

[8] L. K. Li, W. K. Pang, W. T. Yu and M. D. Troutt, Forecasting short-term exchange Rates :a recurrent neural network approach, Neural Networks in Business Forecasting (G. Peter Zhang, Eds.), Idea Group Publishing, (2004), 195-212.

[9] L. K. Li and S. Shao, A Neural Network Approach for Global Optimization with Applications to Nonlinear Least Square Problems, preprint.

[10] N. Meade, A comparison of the accuracy of short-term foreign exchange forecasting methods, *International Journal of Forecasting*, 18, (2002) 67-83.

[11] M. J. Perez-Ilzarbe, Convergence analysis of a discrete-time recurrent neural network to perform quadratic real optimization with bound constraints, *IEEE Tranc. Neural Networks*, vol. 9, no. 6, (1998) 1344-1351.

[12] P. Tenti, Forecasting foreign exchange rates using recurrent neural networks, *Appl. Artificial Intelligence*, Vol. 10, 567-581 (1996), 150-170.

[13] D. E. Rumelhalt, G. E. Hinton and Williams (Eds.), Learning internal representation by error propagation, in *Parallel Distributed Processing*, MIT Press, Cambridge, MA, (1986), 318-362.

[14] E. Simon, 'Forecasting Foreign Exchange Rates with Neural Networks', Research Project, 2002, 1-85: available at

http://liawww.epfl.ch/Teaching/StudentProjects/Old/WS2002/Simon/.

[15] H. Tuy, Convex Analysis and Global Optimization, Kluwer Academic Publisher, Dordrecht, 1998.

[16] X. Wang and E. K. Blum, Discrete-time versus continuous-time models of neural networks, *J. of Comp. Syst. Sci.*, vol. 45, 1-19, 1992.

[17] R. J. Williams and D. Zipser, A learning algorithm for continually running fully recurrent neural networks, *Neural Computation*, vol. 1, (1989), 270-280.

[18] J. Yao and C. L. Tan, A case study on using neural networks to perform technical forecasting of forex, *Neural Computation*, 34, (2000), 79-98.

[19] E. Yair, K. Zeger and A. Gersho, Competitive learning and soft competition for vector quantizer design , *IEEE Tranc. Signal Processing*, vol. 40, no. 2, (1992) 294-309.

Received: October 26, 2007