Neurocomputing 72 (2009) 3529–3537

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A self-organising mixture autoregressive network for FX time series modelling and prediction

He Ni [a,b], Hujun Yin [a,*]

[a] School of Electrical and Electronic Engineering, The University of Manchester, Manchester, M60 1QD, UK
[b] School of Finance, Zhejiang Gongshang University, HangZhou, P.R. China

ARTICLE INFO

Available online 11 June 2009

Keywords:
FX rates
Self-organising networks
Time series
Autoregressive models

ABSTRACT

Nowadays a great deal of effort has been made in order to gain advantages in foreign exchange (FX) rates predictions. However, most existing techniques seldom excel the simple random walk model in practical applications. This paper describes a self-organising network formed on the basis of a mixture of adaptive autoregressive models. The proposed network, termed self-organising mixture autoregressive (SOMAR) model, can be used to describe and model nonstationary, nonlinear time series by means of a number of underlying local regressive models. An autocorrelation coefficient-based measure is proposed as the similarity measure for assigning input samples to the underlying local models. Experiments on both benchmark time series and several FX rates have been conducted. The results show that the proposed method consistently outperforms other local time series modelling techniques on a range of performance measures including the mean-square-error, correct trend predication percentage, accumulated profit and model variance.

1. Introduction

Foreign exchange (FX) or forex markets have grown rapidly in recent years and have become by far the largest financial market in the world with average daily trade in the global forex and related markets were reported to be over US$ 4 trillion in April 2007.[1] FX rate forecasting has been an active and challenging area of research and has attracted a great deal of attention ever since the collapse of the Bretton–Woods system in 1973. The trend analysis in spot FX rates has been a recurrent theme among statisticians, econometricians and computer scientists. A fundamental approach is to use economic theories to underline the structural relations between exchange rates and other variables (e.g. interest rates and consumer price index) and to use statistical methods to identify the correlation between the past data and its future moves. Researchers have devoted a great deal of effort to developing various techniques in order to build a valid model. However, most econometric and time series techniques often fail to outdo the simplest random walk especially in short durations [21,12]. The reason is that most of econometric models are linear and are used under specific or strict assumptions. For instance, autoregressive (AR) and autoregressive moving average (ARMA) models assume a linear relationship between the current value of the variables and the previous values and error terms. The mean and variance of the variables are regarded as constant over time. In other words, the process is assumed to be stationary. In practice, these conditions cannot be met.

In this paper, adaptive neural networks, in particular self-organising maps (SOM), are explored in modelling FX time series in conjunction with regressive models. The approach uses SOM to divide a time series into a number of homogeneous processes and these processes are then modelled by the nodes of an enhanced temporal SOM. The proposed network is termed self-organising mixture autoregressive (SOMAR) model, which uses AR models as components in the construction of the topological mixture model. A brief review on AR and related regressive models is given first as below.

1.1. AR and ARMA processes

The statistical AR model has been a primary tool in modelling time series, including financial time series [18]. The ARMA is the extended version of the AR model. Econometricians also use the generalised autoregressive conditional heteroskedastic (GARCH) [1] to further model the volatilities or variances of a financial time series.

The notation AR(p) refers to the autoregressive model of order p. The AR(p) model is written as

$$\begin{aligned} x_t &= c + \sum_{i=1}^{p} \phi_i x_{t-i} + \varepsilon_t \\ &= c + \mathbf{x}_{t-1}^{(p)\,T}\mathbf{w} + \varepsilon_t, \end{aligned} \tag{1}$$

* Corresponding author.
E-mail addresses: nihe@mail.zjgsu.edu.cn, hujun@soft.ee.umist.ac.uk (H. Ni), h.yin@manchester.ac.uk (H. Yin).
[1] Triennial Central Bank Survey (December 2007), Bank for International Settlements.

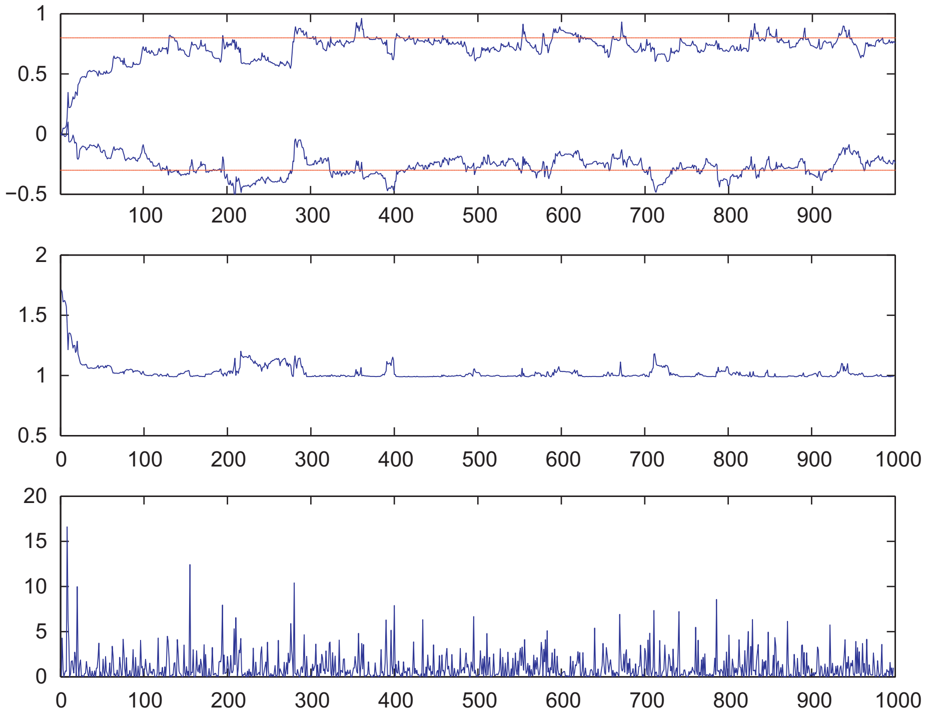


Fig. 1. Upper: estimation of AR(2) parameters; middle: sum of MSE of all inputs; lower: squared error at each time point.

where $\mathbf{w} = [\phi_1, \ldots, \phi_p]^T$ are the parameters of the model, $\mathbf{x}^{(p)}_{t-1} = [x_{t-1}, x_{t-2}, \ldots, x_{t-p}]^T$ is a concatenated input vector, c is a constant and ε_t is white noise with zero mean and variance σ^2. The process is either random walk, when x_t exhibits a unit root, or covariance-stationary if the roots of character equation are all inside unit circle.

An ARMA model with the notation ARMA(p,q) is an AR(p) model with q moving average (MA) terms—MA(q). The ARMA(p,q) model can be written as

$$x_t = c + \varepsilon_t + \sum_{i=1}^{p} \phi_i x_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}, \tag{2}$$

where $\{\theta_1, \ldots, \theta_q\}$ are the parameters of the moving average. The error terms ε_t are assumed to be independent identically distributed (i.i.d.) random variables sampled from a normal distribution with zero mean and variance σ^2. If this condition does not hold, the GARCH model in Section 1.3 provides a generalised alternative.

1.2. Adaptive recursive least-mean-square method

The parameters of an AR(p) model can be estimated by the recursive least-mean-square (LMS) method, which is a stochastic gradient descent optimisation[2]

$$\mathbf{w}(t) = \mathbf{w}(t-1) + \eta(t)\, e(t)\mathbf{x}(t), \tag{3}$$

$$e(t) = x(t+1) - \mathbf{x}(t)^T \mathbf{w}, \tag{4}$$

where $\eta(t)$ is the step size or learning rate, often a small positive constant or monotonically decreasing in value.

Fig. 1 depicts the parameter estimation process for an AR(2) process using the LMS method. The parameters have been correctly estimated, as indicated by two solid curves (estimated parameters) in the upper figure. The dashed lines indicate the generating parameters. The horizontal axis defines the time steps or iterations. The middle plot shows the overall mean-squared-error (MSE) along the estimation process; and the lower plot is the squared error for each input.

[2] For offline learning the LMS method can be used to estimate the model parameters. When the training data are not available beforehand, or in online learning, it is necessary to use recursive LMS method.

1.3. GARCH model

The GARCH model [1] explicitly considers the variance of the current residual to be a linear function of the variances of the previous residuals. It has been widely applied to modelling financial time series, including FX rates, which exhibit different volatilities from time to time. It defines periods of high oscillation followed by periods of relative calm in a time series or vice versa.

A simple GARCH(θ, ρ) model can be expressed as follows:

$$x_t = \phi_0 + \sum_{i=1}^{p} \phi_i x_{t-i} + \varepsilon_t, \tag{5}$$

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{\theta} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{\rho} \beta_j \sigma_{t-j}^2, \tag{6}$$

where ε_t is the error term with the assumption $\varepsilon_t = \sigma_t v_t$. v_t is i.i.d. with zero mean and unit variance. α_0, α_i, and β_j are model parameters to be estimated and θ, ρ are pre-determined orders.

In the literature, simple GARCH model has been extended by applying additional restrictions, such as exponential GARCH by Nelson [23], quadratic GARCH model by Sentana [27] or by changing the assumption that v_t has normal distribution, t-distribution, Cauchy distribution, etc. In this paper, only the commonly used standard GARCH model is used in the experiments.

The rest of this paper is organised as follows. Neural network approaches and SOM related time series modelling methods are discussed in Section 2. In Section 3, the proposed methodology is described. Section 4 presents applications of the proposed network for modelling and prediction of time series and FX rates. Finally, conclusions are given in Section 5.

2. Neural networks for FX modelling

The main difficulty in modelling financial time series is their nonstationarity. That is, the mean and variance of the time series are not constant but change over time. This implies that the variables switch their dynamics from time to time or have different modes in different periods. It is particularly true in FX rates due to the amount of inconstant information flow. Previous studies [8] show that the distribution of daily returns[3] is approximately symmetric and leptokurtic (i.e. heavy tailed). One possible explanation for the heavy tailed distribution is that samples are independently distributed as a normal distribution whose mean and variance change with time. Others argue that observed returns come from a mixture of normal distributions [6,19]. Further studies on volatility [36] indicate that the average returns of high frequency FX data are negligible in comparison to its volatility[4] and the kurtosis is much higher than that of the kurtosis of a normal distribution. Therefore, it is not convincing and impractical for a single parametric model to capture the dynamics of the entire time series.

Due to the recent advances in computational intelligence and increased computer power, nonparametric models have been studied and used extensively in the last few years with various successes. FX rate forecasting by adaptive neural networks provides strong evidence in terms of out-of-sample forecasting achievements. For example, Zimmermann et al. [34] have proposed a perceptron model for agents (buyers and sellers) in the FX market, and modelled the aggregate market by a neural network, in order to extract the price dynamics. Medeiros et al. [20] have tested the nonlinearity and predictability of FX rates by neural networks, and rejected the null hypothesis of linearity in most of the testing FX rates. Zhang et al. [35] used wavelet transform to decompose the time series into various resolution and used perceptron neural networks to learn to combine the forecasts from different time scales. So far, many comparative studies have shown that neural networks can outperform linear ARMA model and random walk model [9,4,13,2,15]. The most widely used techniques so far are the multilayer perceptron (MLP) [10], radial basis function (RBF) networks [10] and recurrent networks [15].

2.1. Temporal SOMs

In the early 2000s, there emerged another potential solution to tackle the problem of nonstationarity in financial time series, that is, to divide a complex problem into several smaller and simpler ones [2]. The results of the simple models are then combined to produce the final solution to the original problem. The entire input space is split into several (often disjoint) regions by means of clustering, each containing a prototype vector and a set of regression parameters. The prediction is thus made by the best-fit local models. The solution proposed in this paper is a mixture of regression models that performs the partition and regression at the same time. The proposed network is a mixture of local AR models and can converge easily to the underlying local models. It is naturally suited for characterising the dynamics of nonstationary time series such as FX rates.

The self-organising map is a neural network that is able to self-configure its neurons in order to quantise or cluster the input space into a topological structure [14]. Such a capability makes the SOM attractive in many applications that involve the use of clusters and local models—a useful pre-processing for local model approach to FX rate prediction [24]. SOM divides the input space into small regions associated by the best matching units. The area in the input space for which the reference vector is responsible is called Voronoi region. Voronoi tesselation partitions the input space into disjoint sets. Models can be created by locally fitting to specific Voronoi regions. The topological relationship of local models is maintained on the pre-determined lattice. There are various successes in financial applications. For example, Dablemont et al. [7] have applied SOM-based local models with RBF network as regressor to predict the returns of the DAX30 index. Cao [2] has proposed a SVM experts system, which relies on SOM to partition the data into several disjoint regions first. The SVM regression model is then trained on each region.

For processing time series data using the standard SOM, one can use a sliding window to group consecutive time series points into vectors as the input. We call it the vector SOM (VSOM). In VSOM temporal context is achieved by means of a temporal window of a pre-fixed length. Recently, increased interest arises in SOM with recurrent connection for temporal or sequence processing. Typical extensions include temporal Kohonen map (TKM) [5], recurrent SOM (RSOM) [15] and recursive SOM (RecSOM) [31]. TKM and RSOM introduce a concept of leaky integration (integrator $0<\alpha<1$) of the activation, which is claimed to be derived from biological phenomena. The activation $\mathbf{y}_i(t)$ of a neuron is gradually dismissed. That is, the previous active neuron is more likely to win again than other neurons. The activation is a combination of matching the current input and previous activation

$$\mathbf{y}_i(t) = (1-\alpha)\mathbf{y}_i(t-1) + \alpha(\mathbf{x}(t) - \mathbf{w}_i(t)). \tag{7}$$

RecSOM incorporates context information into weights. It has a similar aim as RSOM but in a different form of activation $\mathbf{y}_i(t)$

$$\mathbf{y}_i(t) = \exp(-\alpha\|\mathbf{x}(t) - \mathbf{w}_i^x\|^2 - \beta\|\mathbf{y}(t-1) - \mathbf{w}_i^y\|^2), \tag{8}$$

where α and β are constant coefficients. The learning is similar to the original SOM algorithm, but each neuron is represented by two weight vectors $\mathbf{w}^x$ and $\mathbf{w}^y$, for matching the input and neuron's activity, respectively. By using SOM with these recurrent or recursive connections, local models captured by the nodes become more capable in detecting temporal content in the input sequence.

Neuron gas (NG) is another SOM variant. Instead of using the distance between input vector and the reference vector and a predefined neighbourhood structure, NG uses a neighbourhood ranking to define the neighbourhood and to update reference vectors

$$\mathbf{w}_i(t) = \mathbf{w}_i(t-1) + \varepsilon \cdot e^{-k_i/\lambda}(\mathbf{x}(t) - \mathbf{w}_i(t-1)), \tag{9}$$

where k_i is the rank of neuron i (with regard to the input) and λ is a constant which determines the number of neurons that should change their weights at each step.

2.2. Support vector machine for regression

As a regressive method, support vector machines (SVMs) [29] have been used as a good alternative for MLP in time series forecasting. SVMs are established on the theory of the structural risk minimisation. Suppose one is given a set of training data $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ and y_t denotes the value of x immediately after $\mathbf{x}_t$. The prediction is achieved by estimating a linear function [30,22],

$$f(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) + b, \tag{10}$$

where $\phi(\mathbf{x})$ represents a nonlinear mapping of $\mathbf{x}$ in a high-dimensional feature space. The function f can be estimated by

[3] A simple logarithm difference transform.

[4] It most frequently refers to the standard deviation of the change in the value of a financial instrument with a specific time horizon.

minimising a regularised risk function,

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\frac{1}{n}\sum_{t=1}^{n} L_\varepsilon, \tag{11}$$

where

$$L_\varepsilon = \begin{cases} |y_t - \mathbf{w}^T\phi(\mathbf{x}_t) - b| - \varepsilon, & |y_t - \mathbf{w}^T\phi(\mathbf{x}_t) - b| \geq \varepsilon, \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

The term $\|\mathbf{w}\|^2$ is called the regularisation term. The second term of Eq. (11) is the empirical error measured by Vapnik's ε-insensitive loss function. C is a regularisation constant. The primal objective function is defined based on the above two equations by introducing slack variables ξ_t^*, ξ_t

$$\text{minimise} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{t=1}^{n}(\xi_t + \xi_t^*), \tag{13}$$

$$\text{subject to} \quad \begin{cases} y_t - \mathbf{w}^T\phi(\mathbf{x}) - b \leq \varepsilon + \xi_t, \\ \mathbf{w}^T\phi(\mathbf{x}) + b - y_t \leq \varepsilon + \xi_t^*, \\ \xi_t, \xi_t^* \geq 0. \end{cases} \tag{14}$$

By introducing Lagrange multipliers γ_t, γ_t^*, one can solve a linearly constrained quadratic problem. The decision function then has an explicit form

$$f(\mathbf{x}) = \sum_{t=1}^{n}(\gamma_t - \gamma_t^*)K(\mathbf{x}_t, \mathbf{x}) + b, \tag{15}$$

where $K(\mathbf{x}_t, \mathbf{x})$ is the so-called kernel function. The elegance of using the kernel function is that one can deal with arbitrary dimensional feature space without having to compute the non-linear mapping function explicitly. Based on the Karush–Kuhn–Tucker (KKT) conditions [16], only a small number of coefficients $(\gamma_t - \gamma_t^*)$ will be non-zero, which correspond to a set of training data points, referred to as the support vectors. Approximation errors at the points of support vectors are larger than or equal to ε.

3. Methodology

The problem of predicting future values of a stochastic process is closely related to the task of estimating the unknown parameters of a regressive model. The target process can be assumed to be generated by a number of stationary autoregressive processes. It has applications in many fields, especially in econometrics and automatic control. A number of studies recently focus on modelling such nonstationary processes using mixture models [33]. When a nonstationary process is considered as a mixture of several stationary AR processes, it can be expressed as

$$F(x_t|\mathcal{F}_{t-1}) = \sum_{i=1}^{N}\alpha_i\Phi\left(\frac{x_t - \phi_{i0} - \phi_{i1}x_{t-1} - \cdots - \phi_{ip_i}x_{t-p_i}}{\sigma_i}\right), \tag{16}$$

where $\mathcal{F}_{t-1}$ represents the information up to time $t-1$, α, ϕ, σ are parameters, N is the number of AR processes, p_i is the order of AR process i and Φ refers to certain distribution functions. This is the so-called N-component mixture autoregressive (MAR) model [33]. It has been reported to have the ability to handle cycles and conditional heteroscedasticity in time series. Its parameters are estimated via the EM algorithm.

In this paper we further simplify this mixture based on the assumption that the underlying process is a weighted average of several independent, stationary AR processes at a time. It also refers to local model approach, but possibly with some overlaps between the local models. It can be expressed as

$$F'(x_t|\mathcal{F}_{t-1}) = \sum_{i=1}^{N}\beta_{(i,\mathbf{x})}\Phi(x_t - \phi_{i0} - \phi_{i1}x_{t-1} - \cdots - \phi_{ip_i}x_{t-p_i}), \tag{17}$$

where $\beta_{(i,\mathbf{x})}$ is a positive indicator or weighting factor of the i-th local regressive model, under the restriction $\sum_{i=1}^{N}\beta_{(i,\mathbf{x})} = 1$.

3.1. Self-organising AR (SOAR) model

In 1989, Lampinen and Oja [17] proposed a self-organising map of "neural" units where each unit represents an AR model with its parameters as reference vector. The experiment conducted in [17] shows that the self-organising AR models can learn to distinguish texture images by unsupervised learning. The method in fact is a multiple AR model with the parameters of each AR model following a topological structure set by a self-organised mapping of neural units. However, the model has difficulties in converging to the correct AR models.

In the SOAR, AR models are represented by neural units, each with its weight vector defining the parameters of an AR process. The general procedure of the method is

1. For each input vector, find the best matching unit by measuring the estimation error.
2. Update the best matching unit as well as the units in its topological neighbourhood, by the Widrow–Hoff [32] rule.

In order to reduce the effect of noise in the errors, an exponential average over the recent estimation errors is used

$$u_i(t) = \beta_s e_i(t) + (1-\beta_s)u_i(t-1), \tag{18}$$

where β_s is a smoothing factor, $e_i(t)$ is the current estimation error of node i and $u_i(t-1)$ is its past averaged error. The best matching unit is the one with the smallest $u_i(t)$.

The neighbourhood is defined by a linear neighbourhood function

$$g(r) = \eta\left(1 - \frac{r}{\vartheta\left(1 - \frac{t}{t_{max}}\right) + 1}\right), \tag{19}$$

where r is the distance between the best matching unit and the unit to be updated, t_{max} is a pre-determined maximum training iteration. The adaption coefficient η can either be a scalar constant or be a slowly decreasing parameter and ϑ is an adjustable constant controlling the shrinking speed of the neighbourhood. The best matching unit and its neighbouring units update their weights according to

$$\mathbf{w}_i(t) = \mathbf{w}_i(t-1) + g(r)e(t)\mathbf{x}(t). \tag{20}$$

The model has been applied for segmenting images into texture classes, without priori knowledge about the number of classes or the class models [17]. Experimental results on 1-D artificial signal and 2-D textures are also reported in [17].

However, the performance of the SOAR model in finding underlying AR processes in mixture AR processes is poor. Due to the stochastic nature of AR processes, although the overall MSE decreases "monotonically" as shown in Fig. 1, at each input, one can always expect large oscillations even when the true model parameters are used or further manipulations, e.g. exponential average, are applied. In other words, this method has difficulties in converging to the true model parameters of the underlying AR processes.

Fig. 2 shows such an example of SOAR for the estimation of the parameters of an AR(2) process. The method fails to converge to

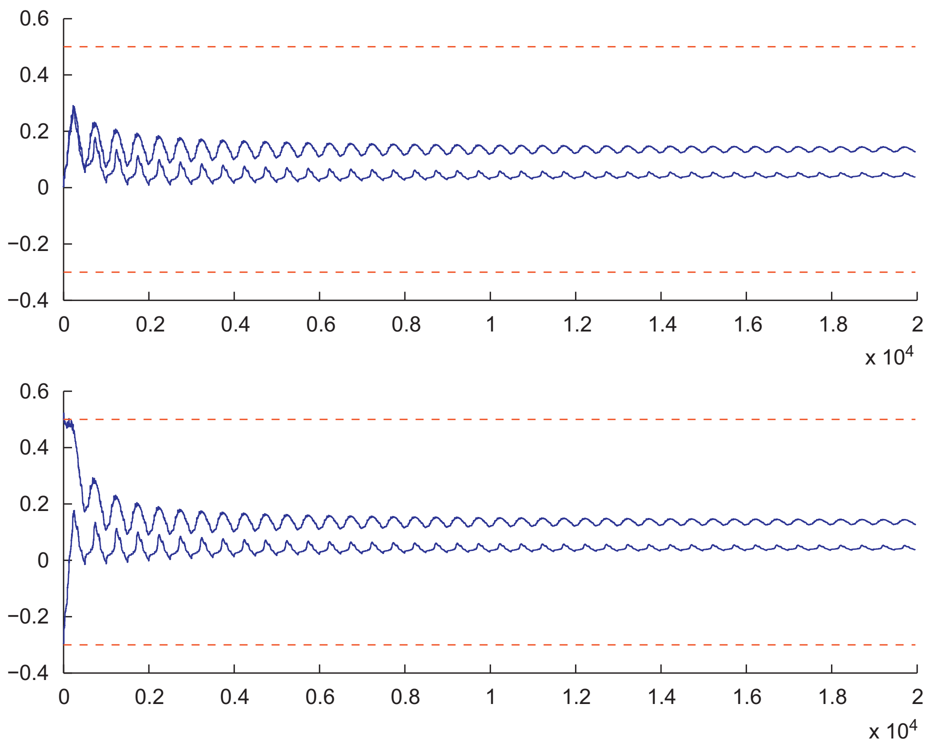


Fig. 2. Estimation of AR(2) parameters by SOAR. Upper: from random initial weights; lower: with true values as initial weights.

the true parameters with either random initial weights or the true values as initial weights.

3.2. *Self-organising mixture AR (SOMAR) network*

As a stochastic process is characterised by white noise corruption. As a sufficient condition, the modelling error should be close to white noise if the modelling is following the "correct" path. Therefore, we propose to use the autocorrelation of the error instead of the error itself to evaluate the similarity between the input vectors and the model parameters represented by the neurons weights. In order to estimate autocorrelation, we hereby use a small batch of the input, or a patch.

With a set of model parameters, the modelling error at node i is a discrete time series of length m, the patch size, with mean μ and variance σ^2

$$e_i(t) = x(t) - \mathbf{w}^T \mathbf{x}_{t-1}^{(p)}, \tag{21}$$

where p is the order of local AR model i. An estimate of the autocorrelation coefficient $R(k)$ at lag k ($k<m$) can be obtained by

$$R_i(k) = \frac{1}{m\sigma^2} \sum_{l=0}^{m-k-1} (e(t-l)-\mu)(e(t+k-l)-\mu). \tag{22}$$

Fig. 3(a) shows the autocorrelations of a set of the modelling errors from a patch of 20 input samples with the weights set as the generating parameters $[-0.2, 0.5]$. We have also randomly chosen three sets of parameters $[-0.1, 0.6]$, $[0.1, -0.1]$ and $[0.5, -0.2]$ as weights, their corresponding autocorrelations of the modelling errors are plotted in Fig. 3(b)–(d). The autocorrelation coefficients of these errors show that the errors in Fig. 3(a) are the closest to white noise compared to the others. The sum of the (absolute) autocorrelation coefficients (SAC) is 3.8832($a0$), 4.3504($a1$), 4.5224($a2$), 4.5963($a3$), respectively.

Apparently, $a1$ is closer to $a0$ than $a2$ and $a3$ are, so is the corresponding SAC value. Therefore a better way to decide the best matching unit or the AR process that generates the input is to use SAC value, instead of the modelling error itself or Eq. (18). In order to verify that SAC has only one unique global minimum corresponding to the correct regressive model, we purposely calculate the SAC value of a number of models whose parameters are within a vicinity of the pre-set parameter $[-0.2, 0.5]$. The SAC values with regard to model parameters are illustrated in a contour plot in Fig. 4.

Now the conventional SOM and SOAR have been extended to the self-organising mixture AR model, with each neuron on the lattice associated with a vector $\mathbf{w}_i$ corresponding to the parameters of a local AR model and a SAC value denoting the similarity measure.

The training of the SOMAR model consists of two steps. In the first step, a fixed number of consecutive input vectors are used to form a patch input. Then a winner for that patch input is selected if its weights have the minimum SAC, i.e.

$$v = \arg\min_i \left(\sum_{j=-m}^{m} |R_i(j)| \right), \quad i = 1, 2, \ldots, N, \tag{23}$$

where i is the index of local model and N is the total number of the local models. Then the winner and its neighbours adapt their weights

$$\mathbf{w}_i(t) = \mathbf{w}_i(t-1) + h(r,t)\,\eta(t)\,e_i(t)\,\mathbf{x}(t), \tag{24}$$

where $\eta(t)$ is a decreasing learning rate, $h(r,t)$ is the neighbourhood function and r is the distance from the best match unit v to the unit to be updated

$$h(r,t) = \kappa(t) \exp\left(-\frac{r}{\psi(t)^2} \right), \tag{25}$$

$$\kappa(t) = \frac{\kappa_0}{\kappa_1 + \kappa_2 t + \kappa_3 t^2}, \quad \psi(t) = \frac{\psi_0}{\psi_0 + \psi_1 t + \psi_2 t^2}, \tag{26}$$

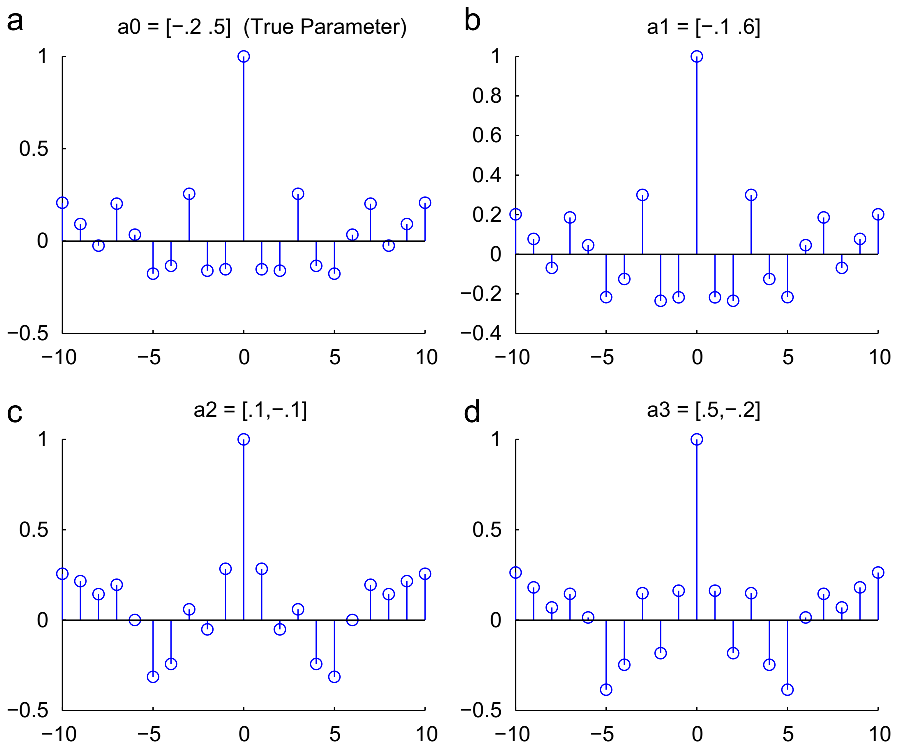


Fig. 3. Autocorrelation of a set of modelling errors.

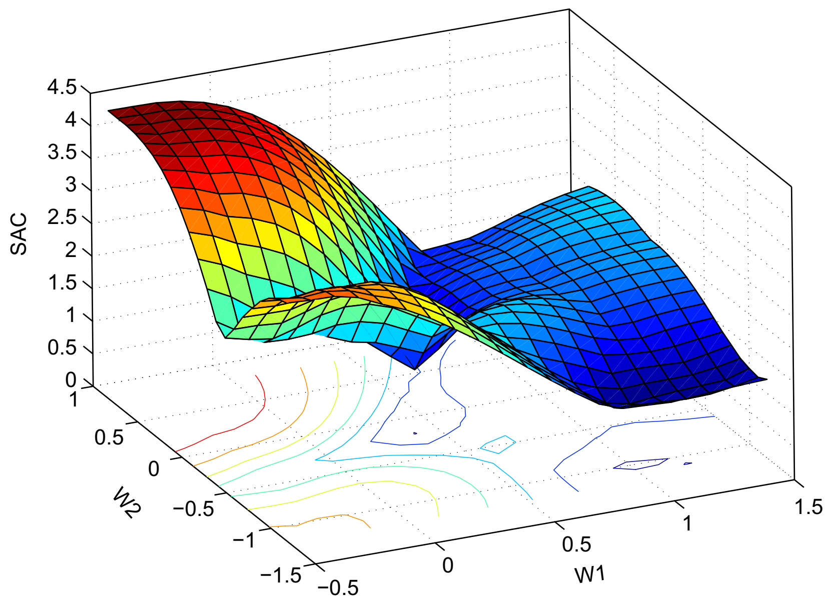


Fig. 4. A contour plot of SAC values of different regressive models with varying model parameters $\mathbf{w}_1 \in [-0.5, 1.5]$ and $\mathbf{w}_2 \in [-1.5, 1]$.

where κ_0, κ_1, κ_2, κ_3 and ψ_0, ψ_1, ψ_2, ψ_3 are constants that control the flatness or the shape of Gaussian neighbourhood function. In the initial training stage, more neighbouring local models are updated with the winning local models. Whilst fewer neighbours get updated in the later training stage.

The second step is a fine tuning stage. Since the input patches may inevitably contain input vectors generated by more than one underlying AR processes, it is necessary to filter out those inputs at later stage of the training when the parameters of local models have been roughly estimated. The errors generated by these patches would result in small differences in SAC value between the local models concerned. Therefore we skip those patches that have similar SAC values to more than one local models. An empirical threshold is set to decide whether the patch should be filtered out or should be used in the fine tuning stage. The improvement is significant compared to the experiments without fine tuning phrase. The fine tuning should start when the learnt parameters have been generally stabilised, that is, when the scale of oscillation becomes small.

Two typical results of SOMAR in learning a mixture of two AR(1) and a mixture of two AR(2) processes are illustrated in Figs. 5 and 6, respectively. The fine tuning stages are set

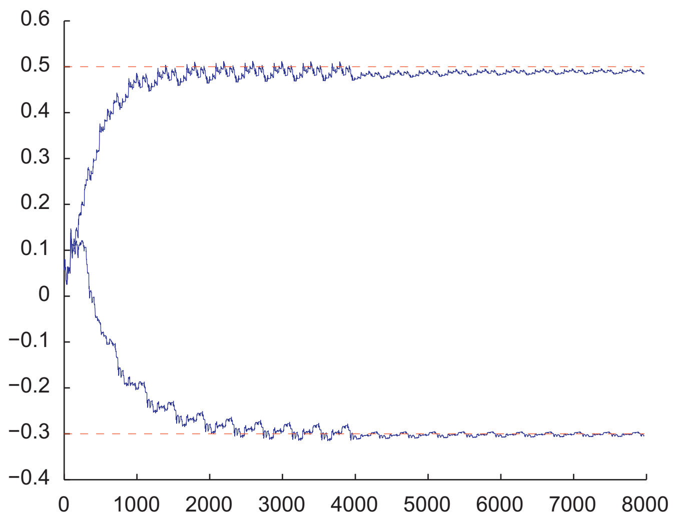


Fig. 5. Parameter estimation of two AR(1) processes $\mathbf{w}_1 = 0.5$, $\mathbf{w}_2 = -0.3$, fine tuning is set at 50% of the training time.

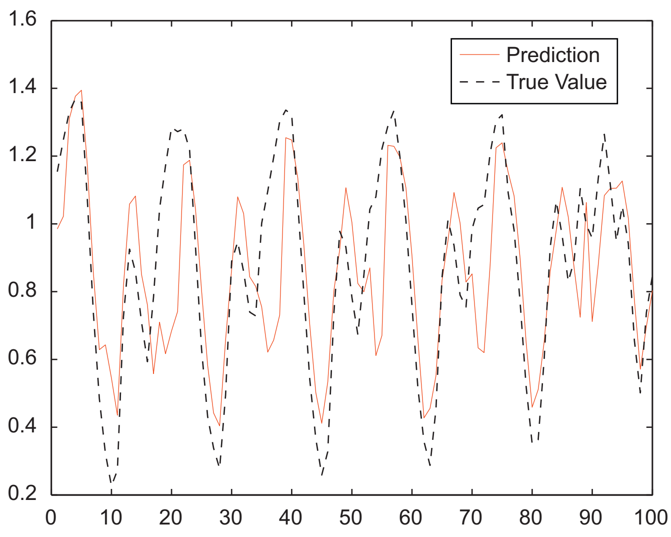


Fig. 7. Prediction of Mackey-Glass data by SOMAR: dashed line represents original data points and solid line represents the prediction.

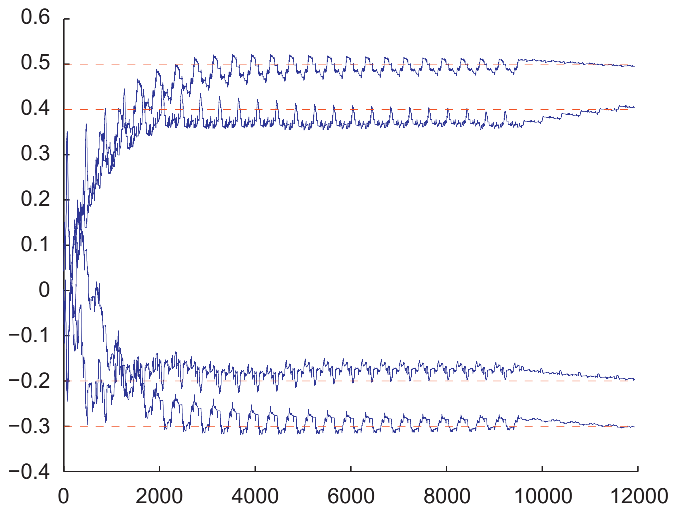


Fig. 6. Parameters estimation of two AR(2) processes, fine tuning starts from 80% of the all iterations. $\mathbf{w}_1 = [0.5, -0.2]$, $\mathbf{w}_2 = [0.4, -0.3]$.

empirically to 50% and 80% of their total training time. As can be seen the estimation errors are significantly reduced during the fine tuning.

The prediction of the SOMAR is first generated by the best underlying AR process when the empirical threshold in the fine tuning stage indicates that there is a best fitted AR model for the input. Then the overall predication model is the weighted average of a few good local models that have similar SAC values for the input patch, as indicated by Eq. (17). The mixing weights $\{\beta_{(i,\mathbf{x})}\}$ are determined inversely proportional to their SAC values: the smaller the value of SAC, the larger the weighting.

4. Experiments

In this section, experiments on both Mackey-Glass data and real-world FX rates are presented. The proposed method is used to characterise the dynamics of a nonlinear, nonstationary time series and to estimate underlying local regressive models.

4.1. Artificial data

The Mackey–Glass data were generated by a dynamic system defined by the differential equation:

$$\frac{dx}{dt} = \beta x(t) + \frac{\alpha x(t-\delta)}{1 + x(t-\delta)^{10}}, \tag{27}$$

with the parameter values set as $\delta = 17$, $\alpha = 0.2$, $\beta = -0.1$. In total 600 points were generated. The Mackey–Glass data are assumed to consist of a number of unknown AR processes. In the experiment, the series was grouped into 15-unit vector $\mathbf{x}(t) = [x(t), x(t+1), \ldots, x(t+14)]^T$ as the input vector. We prefixed the order of the AR processes to 14 in favour of the results of BIC validation in a previous study [25]. Experiments with other values have also been implemented without any significant differences. The prediction result is shown in Fig. 7.

The proposed SOMAR network has been compared with SOAR models, local models based on vector SOM, recurrent SOM, recursive SOM, neural gas, and SOM-SVM [24] in predicting the Mackey–Glass data, and the results in terms of MSE are presented in Table 1. From the table, it can be seen that SOMAR outperforms markedly the other methods.

4.2. Foreign exchange rates

The data set was retrieved from the PACIFIC exchange rate service provided by Antwiler at UBCs Sauder School of Business. It consists of 15 years' daily exchange rates (British pound against US dollar, Euro and HK dollar) excluding weekends and bank holidays when currency markets were closed. In total 3200 consecutive points were used, in which the first 3000 points were used as the training set, the next 100 points as the validation set, and the remaining 100 points as the test set. The training, validation and testing sets were windowed with the length of 15 points to form input vectors. We adopt these currency pairs as they are the most efficient according to the efficient market hypothesis [26].

For a comparison with other SOM-based methods, we conducted the following widely used tests:

Predicted return %: The correct prediction percentage of the return ($x'_t = \ln(x_{t+1}/x_t)$), which is also used as a criterion to check

Table 1
Mean squared errors of various predictors on Mackey–Glass data.

Method	MSE (10^{-2})
SOM-SVM	3.40
NeuralGas	3.30
RecSOM	3.43
RSOM	3.45
VSOM	3.55
SOAR	3.50
SOMAR	**3.12**

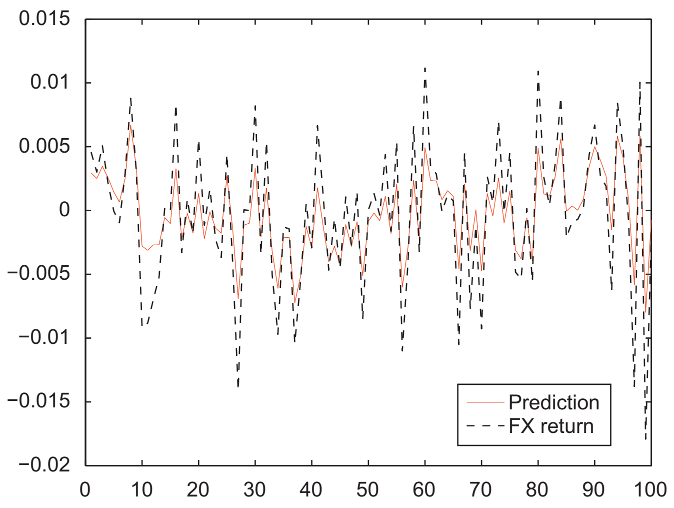


Fig. 8. Predicted returns spanned over 100 testing days. Dashed line represents FX returns, solid line represents the prediction by SOMAR.

whether the prediction is made in the right direction (or trend). In other words, we calculate how many percentages of the predicted returns have the same signs as their corresponding actual returns. Fig. 8 shows such predicted returns in the testing period of 100 days. The correct prediction percentages are shown in Table 2.

MSE of predicted rate: The MSE between the actual exchange rates and the predicted ones in the test set, shown in Table 2. Fig. 9 depicts the predicted and the actual FX rates.

Accumulated profit: The accumulated profit is the percentage gain of the accumulated profits over the testing period, 100 trading days.

Average variance of local model: The averaged variance of SOM units. The variance represents volatility and is inversely proportional to the confidence of investment—the higher the variance of a model has, the less confidence the investment decision generated by the model would be. The result is normalised to the range [0, 1].

The results of the methods under study are presented in Table 2. It can be seen that the SOMAR markedly outperforms the other methods on these performance measures. The proposed method has successfully detected the temporal content between consecutive input samples. Predicted returns are over 5% more correct than other models and the accumulated profits are also consistently higher than all other models in all these currency pairs, reaching on average of 5.23% over 100 trading days on these currency pairs. Smaller fitting errors and lower variances of the proposed SOMAR network further indicate the advantages and robustness of the method in dealing with FX time series.

This improved performance is largely due to the fact that time series segments are more accurately identified and assigned to the underlying local models by using the proposed similarity measure, i.e. the SAC value of errors, compared to using the error

Table 2
Overall predicted return %, MSE of predicated price[a] (10^{-2}), accumulated profit (P%) and average variance (Var) of various methods on three FX rates.

GBPvs	GAR	SS	NG	RecS	RS	VS	SOAR	SOMAR
US %	52.66	53.21	55.10	52.10	51.80	51.00	52.01	**58.38**
US^{-2}	4.26	3.93	4.17	4.62	4.50	4.17	4.33	**3.64**
US P%	4.98	4.97	5.02	5.03	4.96	4.95	4.93	**5.31**
US Var	n/a	0.70	0.51	0.50	0.99	0.76	0.54	**0.43**
EU %	53.00	53.03	53.75	52.91	52.36	51.23	50.32	**56.61**
EU^{-2}	4.63	4.85	4.37	5.01	4.72	4.50	4.75	**4.03**
EU P%	4.66	5.03	4.96	4.62	4.81	4.71	4.62	**5.19**
EU Var	n/a	0.99	0.58	0.77	0.64	0.86	0.63	**0.48**
HK %	53.84	54.00	53.88	54.31	52.96	53.20	53.19	**56.65**
HK^{-2}	4.52	4.65	4.54	4.75	4.87	4.57	4.62	**4.10**
HK P%	4.72	4.92	4.89	4.65	4.78	4.48	4.85	**5.20**
HK Var	n/a	0.61	0.54	0.85	0.74	0.99	0.68	**0.48**

VS, RS, RecS, NG, SS and GAR denote VSOM, RSOM, RecSOM, neural gas, SOM-SVM, and GARCH, respectively.

[a] The prices are divided by the mean exchange rate of that pair over the testing period for a fair comparison.

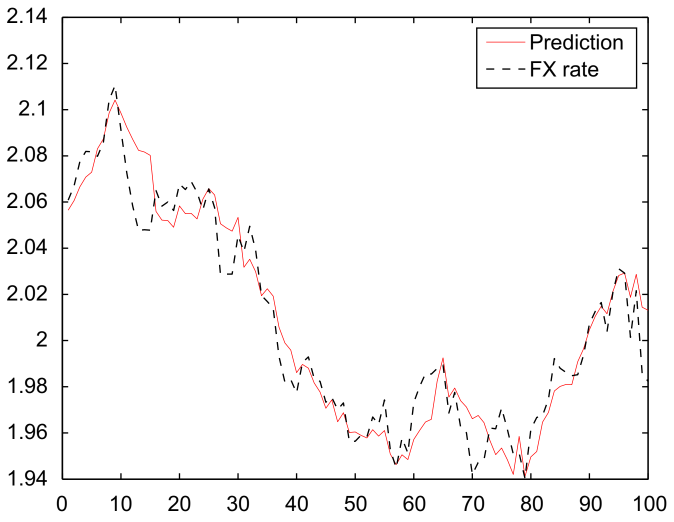


Fig. 9. Predicted rates spanned over 100 testing days. Dashed line represents FX prices, solid line represents the prediction by SOMAR.

directly. The common two-stage methods, which use 'divide-and-conquer' separately, may inevitably lose useful regressive information in the clustering process. Whilst the proposed SOMAR network conducts 'divide' and 'conquer' at the same time and thus can correctly cluster and converge to the underlying local autoregressive models. The modelling and prediction processes of the SOMAR are more flexible especially when an underlying process cannot be precisely described by one local model. At different time, a different combination of the local models is adaptively made up to match the time series. Such adaptive mixtures of local models prove effective and efficient for describing nonstationary FX time series.

5. Conclusions

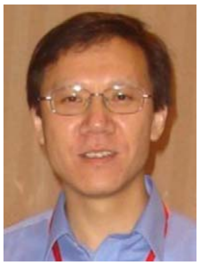
A new approach to modelling nonstationarity of financial time series has been proposed by using a self-organising mixture autoregressive network. The network consists of a number of local autoregressive models that are organised and learnt in a self-organised fashion. An autocorrelation-based similarity is proposed and adopted as the fitness measure of local models to an input segment. It makes the network learning more effective and robust compared to the error-based and Euclidean distance-based measures. The experimental results on both synthetic and real data sets show that the proposed SOMAR network can correctly detect and uncover underlying autoregressive models. They also show that the proposed method outperforms other temporal SOM methods as well as the GARCH model in modelling and prediction of nonstationary FX rate time series.

The SOMAR combines local temporal modelling and clustering in a joint estimation, which proves better than two-stage (separate clustering and regression) methods. Its topological ordering process is further making the learning error-tolerant and robust. That is, ordered local regressive models enhance the ultimate performance of the network. Whilst in two-stage methods, clustering and regression are two separate processes, which may not lead to an optimal solution.

References

[1] T. Bollerslev, Generalized autoregressive conditional heteroskedasticity, Journal of Econometrics 31 (1986) 307–327.
[2] L.J. Cao, Support vector machines experts for time series forecasting, Neurocomputing 51 (2002) 321–339.
[4] A.-S. Chen, M.T. Leung, Regression neural network for error correction in foreign exchange forecasting and trading, Computers and Operations Research 31 (2004) 1049–1068.
[5] G. Chappell, J. Taylor, The temporal Kohonen map, Neural Networks 6 (1993) 441–445.
[6] P.K. Clark, A subordinate stochastic process model with finite variance for speculative price, Econometrica 41 (1973) 135–155.
[7] S. Dablemont, G. Simon, A. Lendasse, A. Ruttiens, F. Blayo, M. Verleysen, Time series forecasting with SOM and local non-linear models—application to the DAX30 index prediction, in: Proceedings of WSOM'03, 2003, pp. 340–345.
[8] F.X. Diebold, Empirical Modeling of Exchange Rate Dynamics, Springer, New York, 1988.
[9] T.H. Hann, E. Steurer, Much ado about nothing? Exchange rate forecasting: neural networks vs. linear models using monthly and weekly data, Neurocomputing 10 (1996) 323–339.
[10] S. Haykin, Neural Networks—A Comprehensive Foundation, second ed., Prentice-Hall, Englewood Cliffs, 1998.
[12] L. Kilian, M.P. Taylor, Why is it so difficult to beat random walk forecast of exchange rates, Journal of International Economics 60 (2003) 85–107.
[13] V. Kodogiannis, A. Lolis, Forecasting financial time series using neural network and fuzzy system-based techniques, Neural Computing and Applications 11 (2002) 90–102.
[14] T. Kohonen, Self-Organizing Maps, Springer, Berlin, 1995.
[15] T. Koskela, M. Varsta, J. Heikkonen, K. Kaski, Time series prediction using recurrent SOM with local linear models, Research Reports B15, Helsinki University of Technology, Laboratory of Computational Engineering, 1997.
[16] H.W. Kuhn, A.W. Tucker, Nonlinear Programming, in: Proceedings of 2nd Berkeley Symposium on Mathematical Statistics and Probabilistics, University of California Press, Berkeley, 1951, pp. 481–492.
[17] J. Lampinen, E. Oja, Self-organizing maps for spatial and temporal AR models, in: Proceedings of 6th SCIA, Helsinki, Finland, 1989, pp. 120–127.
[18] S. Makridakis, S.C. Wheelwright, R.J. Hyndman, Forecasting: Methods and Applications, Wiley, New York, 1998.
[19] B. Mandelbrot, H. Taylor, On the distribution of stock price differences, Operations Research 15 (1969) 1057–1062.
[20] M.C. Medeiros, A. Veiga, C.E. Pedreira, Modeling exchange rates: smooth transitions, neural networks, and linear models, IEEE Transactions on Neural Networks 12 (2001) 755–764.
[21] R.A. Meese, K. Rogoff, Empirical exchange rate models of the seventies: do they fit out of sample?, Journal of International Economics 14 (1983) 3–24.
[22] K.-R. Müller, A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, V. Vapnik, Predicting time series with support vector machines, in: W. Gerstner, A. Germond, M. Hasler, J.-D. Nicoud (Eds.), Artificial Neural Networks—ICANN'97, Springer, Berlin, 1997, pp. 999–1004.
[23] D.B. Nelson, Conditional heteroskedasticity in asset returns: a new approach, Econometrica 59 (1991) 347–370.
[24] H. Ni, H. Yin, Recurrent self-organising maps and local support vector machine models for exchange rate prediction, in: Lecture Notes on Computer Science, vol. 3973, 2006, pp. 504–511.
[25] H. Ni, H. Yin, Self-organising maps and local MLP models for exchange rate prediction, in: Proceedings of the 12th CACSC, Loughborough, England, 2006.
[26] E.E. Peters, Chaos and Order in the Capital markets: A New View of Cycles, Prices, and Market Volatility, Wiley, New York, 1991.
[27] E. Sentana, Quadratic ARCH models, The Review of Economic Studies 62 (1995) 639–661.
[29] V. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, 1995.
[30] V. Vapnik, S. Golowich, A. Smola, Support vector method for function approximation, regression estimation, and signal processing, Advances in Neural Information Processing Systems, vol. 9, MIT Press, Cambridge, MA, 1997, pp. 281–287.
[31] T. Voegtlin, P.F. Dominey, Recursive self-organising maps, Neural Networks 15 (2002) 979–991.
[32] B. Widrow, P.E. Mantey, J.L. Griffiths, B.B. Goode, Adaptive antenna systems, Proceedings of IEEE 55 (12) (1967) 2143–2159.
[33] C.S. Wong, W.K. Li, On a mixture autoregressive model, Journal of the Royal Statistical Society, Series B (Statistical Methodology) 62 (2000) 95–115 (part 1).
[34] H.G. Zimmermann, R. Neuneier, R. Grothmann, Multi-agent modeling of multiple FX-markets by neural networks, IEEE Transactions on Neural Networks 12 (2001) 735–743.
[35] B.L. Zhang, R. Coggins, M.A. Jabri, D. Dersch, B. Flower, Multiresolution forecasting for future trading using wavelet decompositions, IEEE Transactions on Neural Networks 12 (2001) 765–775.
[36] B. Zhou, High-frequency data and volatility on foreign-exchange rates, Journal of Business and Economic Statistics, American Statistical Association 14 (1996) 45–52.

He Ni completed his PhD programme in August 2008 at the School of Electrical and Electronic Engineering, the University of Manchester. His research interests include financial time series forecasting, neural networks and machine learning. He obtained a BEng degree in Electronic Engineering from Xidian University and an M.Sc. degree in Automatic Control and System Engineering from the University of Sheffield in 2001 and 2003, respectively. He has started a lectureship post at Zhejiang Gongshang University since September 2008.



Hujun Yin is a Senior Lecturer (Associate Professor) at The University of Manchester, School of Electrical and Electronic Engineering. He received BEng and M.Sc. degrees from Southeast University and PhD degree from University of York in 1983, 1986 and 1996, respectively. His research interests include neural networks, self-organising systems in particular, pattern recognition, bio-/neuro-informatics, and machine learning applications. He has studied, extended and applied the self-organising map (SOM) and related topics such as unsupervised learning, principal manifolds and data visualisation extensively in the past 10 years and proposed a number of extensions including Bayesian SOM and ViSOM, a principled data visualisation method. He has published over 100 peer-reviewed articles in a range of topics from density modelling, text mining and knowledge management, gene expression analysis and peptide sequencing, novelty detection, to financial time series modelling, and recently decoding neuronal responses.

He is a senior member of the IEEE and a member of the UK EPSRC College. He is an Associate Editor of the IEEE Transactions on Neural Networks and a member of the Editorial Board of the International Journal of Neural Systems. He has served on the programme committee for more than 20 international conferences. He has been the Organising Chair, Programme Committee Chair, Steering Committee Member or Chair, Session Chair and General Chair for a number of conferences, such as International Workshop on Self-Organising Maps (WSOM), International Conference on Intelligent Data Engineering and Automated Learning (IDEAL), International Symposium on Neural Networks (ISNN), and International Joint Conference on Neural Networks (IJCNN). He has guest-edited a number of special issues on several leading international journals. He has received research funding from the EPSRC, BBSRC and DTI. He has also been a regular assessor for the EPSRC, the BBSRC, the Royal Society, the Hong Kong Research Grant Council, Netherlands Organisation for Scientific Research, and Slovakia Research and Development Council.