Soft Comput (2014) 18:497–508
DOI 10.1007/s00500-013-1070-2

METHODOLOGIES AND APPLICATION

# Financial time series prediction by a random data-time effective RBF neural network

**Hongli Niu · Jun Wang**

Published online: 5 June 2013


**Abstract** An improved neural network of time series predicting is presented in this paper. We introduce a random data-time effective radial basis function neural network in determination of the output weights, the center vectors and the widths in the hidden layer of the network. In the training modeling, we consider that the historical data on the financial market is key to the investors' decision-making for their investing positions, and the impact of historical data depends closely on the time. We develop a random data-time effective function to describe this impact strength, and a weight is given to each of the historical data, where a drift function and a random Brownian volatility function are applied to express the behavior of the time strength. Further, this neural network is applied to the prediction of financial price series of crude oil, SSE, N225 and DAX. The empirical experiments show that the proposed neural network results in better performance in financial time series forecasting and is advantageous in increasing the forecasting precision.

**Keywords** RBF neural network · Random data-time effective function · Financial time series · Prediction · Gradient descent

## 1 Introduction

Financial market is a complex evolved dynamic system with high volatilities and noises. The modelling and forecasting of financial time series, which takes an existing series of data (historical data) to predict the next value of a series known up to a specific time, is regarded as a rather challenging task in that financial time series are inherently noisy, nonstationary and deterministically chaotic (Yaser and Atiya 1996). Moreover, numerous factors, including political contexts, general economic situations, competition and even the expectations of traders, can influence the fluctuation behaviors (Niu and Wang 2013; Wang and Deng 2008) of such series. For traditional statistical methods, such as the univariate model ARIMA (autoregressive integrated moving average) and the multivariate regression model, it is difficult to capture the irregularity and nonlinearity underlying in the financial time series and it results in unsatisfactory estimations since the linear structure of the model is pre-assumed (Box et al. 1994).

Recently, more advanced nonlinear methods have been frequently applied with success. The ability of support vector machine (SVM) to solve nonlinear regression estimation problems makes SVM successful in time series forecasting (Cao and Tay 2001; Flake and Lawrence 2002; He and Wu 2011; Samsudin et al. 2010). SVM estimates the regression using a set of linear functions that are defined in a high-dimensional feature space and carries out the regression estimation by risk minimization. Fuzzy logic based modelling techniques are also appealing because of its good performance in terms of accuracy and interpretability. In particular, fuzzy systems (Babbar et al. 2013; Gacto et al. 2009; Di Martino et al. 2010; Pouzols et al. 2010) exhibit a combined description and prediction capability as a consequence of their rule-based structure. Furthermore, artificial neural network (ANN), which is an emulation of the biological system of human brain to learn and identify patterns and is composed of many interconnected neurons, has become increasingly popular in financial time series forecasting (Ao

Communicated by G. Acampora.

H. Niu · J. Wang (✉)
Institute of Financial Mathematics and Financial Engineering,
School of Science, Beijing Jiaotong University,
Beijing 100044, People's Republic of China
e-mail: wangjun@bjtu.edu.cn

2011; Azoff 1994; Bahrammirzaee 2010; Dhamija 2010; Guo et al. 2013; Kaastra and Boyd 1996; Liao and Wang 2010; Liu and Wang 2011; Nekoukar and Beheshti 2001; Oconnor and Madden 2006; Pino et al. 2008; Rojas et al. 2000; Sun et al. 2005; Virili and Freisleben 2001; Wang and Wang 2012; Yu 2009). As large-scale parallel processing nonlinear systems that depend on their own intrinsic link data, ANN can approximate any nonlinear continuous function without requiring formal specification of the model, and also has other advantages including robustness and adaptability compared to expert systems due to the large number of interconnected processing elements that can be trained to learn new patterns (Hansen 1999; Trippi and Turban 1993).

Radial basis function neural networks (RBF) (Broomhead and Low 1988), as an important branch of neural networks, have attracted considerable attention in recent time due to their ability to approximate complex nonlinear mappings directly from the input–output data with a simple topological structure, short learning time and global optimization. These advantages have enabled RBF neural networks widely-applied in financial fields (Dhamija 2010; Nekoukar and Beheshti 2001; Rojas et al. 2000; Sun et al. 2005). The training parameters in RBF neural networks merely include centers, widths and weights between the hidden layer and the output layer (Haykin 1999). There are some learning algorithms that have been proposed in the literature for training RBF networks (Grabusts 2001; Harpham and Dawson 2006; Jareanpon et al. 2004; Karayiannis 1999; Niros and Tsekouras 2012; Zheng and Billings 1999), such as orthogonal least squares algorithm, genetic algorithm, supervised and unsupervised gradient-based method, and the nearest neighbor cluster algorithm, etc. In the present paper, we train all the parameters simultaneously by applying the gradient descending algorithm.

In the real financial markets, the investing environments as well as the fluctuation behaviors of the markets are not invariant. Especially, in the current Chinese stock markets, the rapid changes of trading rules and management systems have made it difficult to reflect the markets' development using the early data. However, if only the recent data are selected, a lot of useful information (which the early data hold) will be lost. In the present paper, we suppose that the historical data can affect the volatility of the current market, specifically, the nearer the time of historical data is to the present, the stronger impact the data will have on the predicting model. Therefore, the impact of the historical data in the training set should be time-variant such that it can appropriately reflect the different behavior patterns of the markets at different time. If all the data are equivalently used to train the network, the network system may be of inconformity with the fluctuations of the real financial market. In this research, we propose a random data-time effective function, and combine it with RBF neural network, called RBFRT model or an improved RBF neural network. For this improved network model, each of historical data is given a weight depending on the time at which it occurs. The degree of impact of historical data on the market is expressed by a stochastic process (Wang 2007), where a drift function and a stochastic Brownian volatility function are employed to describe the behavior of the time strength. The Brownian motion ensures the model to have the effect of random movement while maintaining the original trend. To test the effectiveness, we apply the improved RBF neural network to the prediction of four financial time series, which are WTI crude oil price (dollar/barrel), Shanghai Stock Exchange (SSE) Composite Index, Nikkei 225 (N225) and Deutscher Aktien Index (DAX) respectively. The forecasting performance of the model is comparatively analyzed for different parameters and evaluated in various ways.

## 2 Methodology

### 2.1 Radial basis function neural network

Neural networks have been extensively tested on nonlinear dynamic systems modeling and forecasting. A radial basis function (RBF) network is a special type of neural network that uses a radial basis function as its activation function (Broomhead and Low 1988). Due to their universal approximation, more compact topology and faster learning speed, RBF networks have attracted considerable attention, and they have been widely applied in many other fields (Bors and Gabbouj 1994; Devaraj et al. 2002; Garg et al. 2008; Oyang et al. 2005). The RBF neural network is a three-layer feed-forward propagated network. The corresponding structure is $m \times h \times 1$, where $m$ is the number of inputs, $h$ is the number of neurons in the hidden layer and one output unit. Let $X_t = \{x_{1t}, x_{2t}, \ldots, x_{mt}\}$ $(t = 1, 2, \ldots N)$ denote the set of input vector of neurons, and $f(x)$ denote the output. Between the inputs and the output, there is a layer of processing units called hidden units. Each of them implements a radial basic function $\Phi$, see Fig. 1.

Primarily, time series prediction can be supposed to a modelling problem. The first step is establishing a mapping between inputs and outputs. Commonly, the mapping is nonlinear and chaotic. After such a mapping is set up, future values are predicted based on past and current observations (Rojas et al. 2000). RBF neural network achieves a mapping $f : \mathbb{R}^m \to \mathbb{R}$ as

$$f(\mathbf{x}) = w_0 + \sum_{i=1}^{h} w_i \Phi(||\mathbf{x} - \mathbf{c}_i||) \tag{1}$$

where $\|\cdot\|$ represents Euclidean norm; $w_0$ is the bias width between hidden and output layer ($w_0 = 0$ in this paper is considered); $w_i$ is the associate weights from node $i$ of hidden

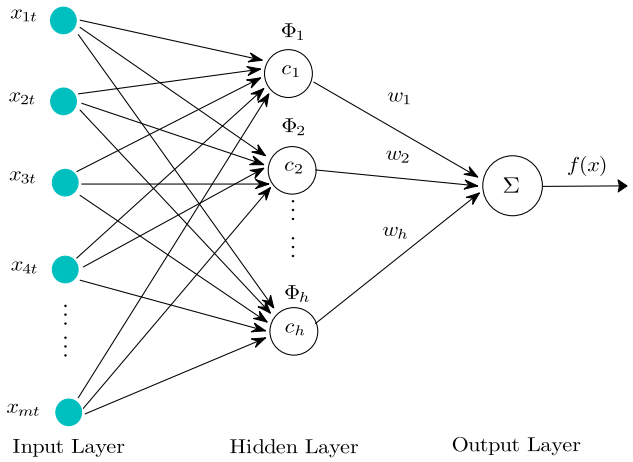


**Fig. 1** General structure of three-layer RBF neural network

layer to output layer; $\mathbf{x} = X_t$ is the input vector; $\mathbf{c}_i$ denotes the center vector of the $i$th unit in the hidden layer, and $\Phi_j$ is the nonlinear activated function. Gauss-based function is usually used for hidden-layer activated function in following expression

$$\Phi_i(||\mathbf{x} - \mathbf{c}_i||) = \exp\{-||\mathbf{x} - \mathbf{c}_i||^2/2\beta_i^2\} \tag{2}$$

where $\beta_i$ is the width of the center. From the above, the design procedure of RBF neural network includes determining the number of neurons in the hidden layer. Then, in order to obtain the desired output of RBF neural network, three parameters need to be defined for each neurons in the hidden layer, center $\mathbf{c}_i$, width $\beta_i$ and weight $w_i$.

### 2.2 Predicting algorithm with a random data-time effective function

In order to determine the parameters in RBF neural network, we employ the Gradient Descent (GD) optimization algorithm (Karayiannis 1999) which takes steps proportional to the negative of the gradient of function at the current point to minimize a given cost function, with its advantages of easily implementation and low storage requirements. Considering the single-node output, let $o_{t_n}$ denote the output value and $y_{t_n}$ be the actual value at time $t_n$, then the error of the output is $\epsilon_{t_n} = o_{t_n} - y_{t_n}$. The error of the sample $n$ is defined as

$$E(t_n) = \frac{1}{2}\mathscr{C}(t_n)(o_{t_n} - y_{t_n})^2 \tag{3}$$

where $\mathscr{C}(t_n)$ is a random data-time effective function, which is defined as

$$\mathscr{C}(t_n) = \frac{1}{\tau} \times e^{\int_{t_0}^{t_n} \mu(t)\,dt + \int_{t_0}^{t_n} \sigma(t)\,dB(t)} \tag{4}$$

where $\tau$ is the time strength coefficient, $t_0$ is the current time or the time of the newest data in the data set and $t_n$ is an arbitrary time point in the data set. $\mu(t)$ is the drift function, $\sigma(t)$ is the volatility function, and $B(t)$ is the standard Brownian motion (Harrison 1990; Meyer and Saley 2002; Wang 2007). Intuitively, the drift function is used to model deterministic trends, the volatility function is often used to model a set of unpredictable events occurring during this motion, and Brownian motion is usually thought as random motion of a particle in liquid (where the future motion of the particle at any given time is not dependent on the past). Brownian motion is a continuous-time stochastic process, and it is the limit of or continuous version of random walks. Since Brownian motion's time derivative is everywhere infinite, it is an idealised approximation to actual random physical processes, which always have a finite time scale. We begin with an explicit definition. A Brownian motion is a real-valued, continuous stochastic process $\{Y(t), t \geq 0\}$ on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$, with independent and stationary increments. In details: (a) continuity: the map $s \mapsto Y(s)$ is continuous $\mathbb{P}$ a.s.; (b) independent increments: If $s \leq t$, $Y_t - Y_s$ is independent of $\mathcal{F} = \sigma(Y_u, u \leq s)$; (c) stationary increments: If $s \leq t$, $Y_t - Y_s$ and $Y_{t-s} - Y_0$ have the same probability law. From this definition, if $\{Y(t), t \geq 0\}$ is a Brownian motion, then $Y_t - Y_0$ is a normal random variable with mean $rt$ and variance $\sigma^2 t$, where $r$ and $\sigma$ are constant real numbers. A Brownian motion is standard (we denote it by $B(t)$) if $B(0) = 0$ $\mathbb{P}$ a.s., $\mathbb{E}[B(t)] = 0$ and $\mathbb{E}[B(t)]^2 = t$. In the above random data-time effective function, the impact of the historical data on the stock market is regarded as a time variable function, the efficiency of the historical data depends on its time. Then the corresponding total error of all the data at each network repeated training set in the output layer is given as

$$E = \sum_{n=1}^{N} E(t_n) = \frac{1}{2}\sum_{n=1}^{N} \frac{1}{\tau} e^{\int_{t_0}^{t_n} \mu(t)\,dt + \int_{t_0}^{t_n} \sigma(t)\,dB(t)} (o_{t_n} - y_{t_n})^2. \tag{5}$$

The main objective of learning algorithm is to minimize the value of cost function $E$ until it reaches the pre-set minimum value $\xi$ by repeated learning. On each repetition, the output is calculated and the total error $E$ is obtained. The gradient of the cost function is given by $\Delta E = \partial E/\partial W$. Then, RBF can be optimized with adjusting the output weights, the center vector and the width value in the radial basis function by iteratively computing the partials and performing the following updates

$$\Delta w_i = -\eta_1 \frac{\partial E}{\partial w_i} = \eta_1 \epsilon_{t_n} \mathscr{C}(t_n)\Phi_i \tag{6}$$

$$\Delta \mathbf{c}_i = -\eta_2 \frac{\partial E}{\partial \mathbf{c}_i} = \eta_2 \epsilon_{t_n} w_i \mathscr{C}(t_n) \frac{\Phi_i}{{\beta_i}^2}(\mathbf{x} - \mathbf{c_i}) \tag{7}$$

$$\Delta \beta_i = -\eta_3 \frac{\partial E}{\partial \beta_i} = \eta_3 \epsilon_{t_n} w_i \mathscr{C}(t_n) \frac{\Phi_i}{{\beta_i}^3}||\mathbf{x} - \mathbf{c}_i|| \tag{8}$$

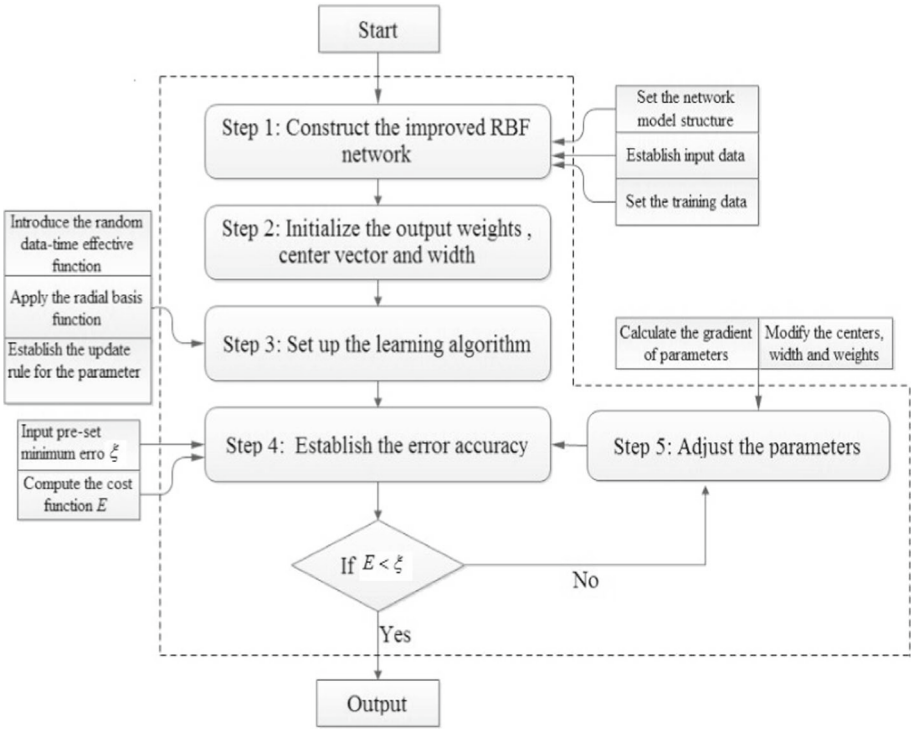


**Fig. 2** The flow chart of training algorithm for the improved RBF neural network

where $\eta_1$, $\eta_2$, $\eta_3$ are the learning rates, which are usually set between 0 and 1. Therefore the modification of the weights, the centers and the width is given by

$$w_i(l+1) = w_i(l) + \Delta w_i = w_i(l) + \eta_1 \epsilon_{t_n} \mathscr{C}(t_n) \Phi_i \tag{9}$$

$$\mathbf{c}_i(l+1) = \mathbf{c}_i(l) + \Delta \mathbf{c}_i = \mathbf{c}_i(l) + \eta_2 \epsilon_{t_n} w_i \mathscr{C}(t_n) \frac{\Phi_i}{{\beta_i}^2} (\mathbf{x} - \mathbf{c_i}) \tag{10}$$

$$\beta_i(l+1) = \beta_i(l) + \Delta \beta_i = \beta_i(l) + \eta_3 \epsilon_{t_n} w_i \mathscr{C}(t_n) \frac{\Phi_i}{{\beta_i}^3} ||\mathbf{x} - \mathbf{c}_i||. \tag{11}$$

According to the above description, the training algorithm procedure of the random data-time effective RBF neural network is briefly shown in Fig. 2.

## 3 Empirical analysis

### 3.1 Data selection and normalization

To examine the effectiveness of the improved RBF neural network, we apply it to the financial time series forecasting. The data adopted in this paper includes the WTI crude oil price, Shanghai Stock Exchange Composite Index, Nikkei 225 and Deutscher Aktien Index. The crude oil data cover the time period from 06/07/2001 up to 19/06/2012, which accounts to 2,753 data points. The SSE is from 16/02/2005 to 15/06/2012 with 1,837 data points. The data of the N225 used in this paper is from 05/06/2006 to 13/07/2012 with 1,449 data points, while that of the DAX is totally 2,301 data points from 01/07/2003 to 29/06/2012. Usually, the nontrading time periods are treated as frozen such that we adopt only the time during trading hours. Let $p(t)$ $(t = 1, 2, \ldots)$ denote the price sequences of crude oil, SSE, N225 and DAX at time $t$, then the corresponding logarithmic return is given by

$$r(t) = \ln p(t+1) - \ln p(t). \tag{12}$$

In Fig. 3, we show the plots of returns for these four price series. We can see that the prices fluctuate wildly, and this means that there is a very high level of noise in the data which brings in the difficulty in forecasting.

To reduce the impact of noise in the financial market and finally lead to a better prediction, the collected data should be properly adjusted and normalized at the beginning of the modelling. There are different normalization methods that are tested to improve the network training (Chaturvedi et al. 1996; Demuth and Beale 2002; Sola and Sevilla 1997), which include "the normalized data in the range of [0, 1]" in the following equation, which is also adopted in this work

$$p(t)' = \frac{p(t) - \min p(t)}{\max p(t) - \min p(t)} \tag{13}$$

where the minimum and maximum values are obtained on the training set during the training process. In order to obtain the true value after the forecasting, we can revert the output variables as $p(t) = p(t)'(\max p(t) - \min p(t)) + \min p(t)$. Then the data is passed to the improved RBF neural network as the nonstationary data.

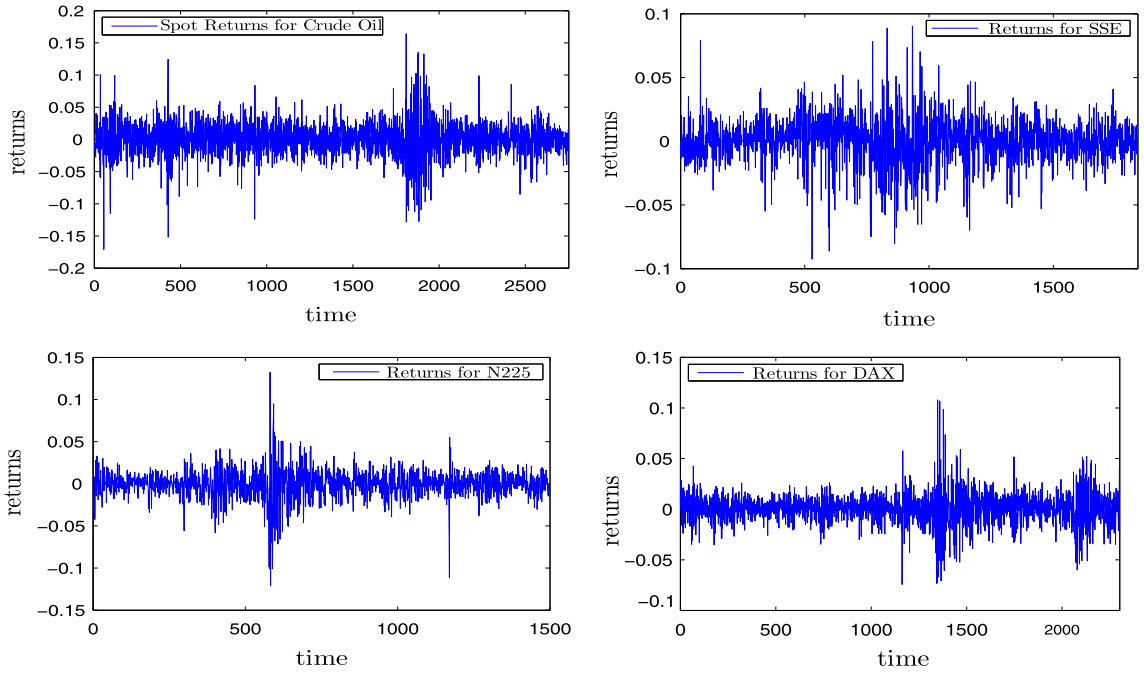


**Fig. 3** The plots of logarithmic returns for the crude oil, SSE, N225 and DAX

### 3.2 Predicting with the improved RBF neural network

Following the procedure of the three-layer RBF neural network introduced in Sect. 2.1, we initially take the number of input nodes as 4, that is, a historical lag with order 4 is considered in the analyzed data. Correspondingly, the original price data of the crude oil, SSE, N225 and DAX are first formed into 2,750, 1,834, 1,496, and 2,298 input–output data pairs respectively. Then the data sets are divided into two parts respectively to form the data training set and the data testing set. Note that the data points for these four time series are not the same, the lengths of training data and testing data are also set differently. The training set for the crude oil is from 11/07/2001 to 11/09/2008 with totally 1,800 data, while that for SSE is from 21/02/2005 to 05/01/2009 with data of 1,000. The training data for N225 are 1000 from 08/06/2006 to 07/07/2010, and those for DAX are 1500 from 04/07/2003 to 22/05/2009. The rest of the data is defined as the testing set. The number of hidden nodes in the hidden layer is pre-set as 15, then we obtain the $4 \times 15 \times 1$ neural network. The maximum training cycle is set $l = 200$, the learning rate of weight, center and width parameter is $\eta_1 = \eta_2 = \eta_3 = 0.001$, and the pre-set minimum error accuracy is 0.0001. Besides, we set that the output weights following the uniform distribution on $(-0.1, 0.1)$, the center vector following the uniform distribution on $(0, 1)$ and the widths following the uniform distribution on $(0.1, 0.3)$. For each time series, we run 10 times of the neural network with different initial points, and the average of the error rates are reported. When we apply the random data-time effective function RBF neural network to predict the daily prices of the crude oil and other three stock indexes, we assume $\mu(t)$ (the drift function) and $\sigma(t)$ (the volatility function) to be following forms

$$\mu(t) = \frac{1}{(t+a)^2}, \quad \sigma(t) = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2} \tag{14}$$

where $a$ is the predictive parameter, and we take it as the length of the time series in this paper, $\bar{x}$ is the mean of the sample data. The corresponding cost function of network training can be written by

$$E = \sum_{n=1}^{N} E(t_n) = \frac{1}{2}\sum_{n=1}^{N}\frac{1}{\tau}e^{\int_{t_0}^{t_n}\frac{1}{(t+a)^2}dt + \int_{t_0}^{t_n}\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i-\bar{x})^2}\,dB(t)} \times (o_{t_n} - y_{t_n})^2. \tag{15}$$

To exclude the significant impacts on the performance of the proposed model for the randomness of initialization of the parameters, we perform a two-sample $t$-test on the total errors in the training set of network RBFRT model for the above randomly-selected initial parameters and the fixed initial parameters respectively. We take crude oil for example, the corresponding statistical test results are presented in Table 1. Let $S_f = \{w_i, c_i, \beta_i\}, i = 1, \ldots, h$ (or $S_r = \{w_i, c_i, \beta_i\}$) represents the fixed (or random) initial parameter set, where $w_i$, $c_i$ and $\beta_i$ denotes the value of weight, center and width parameter of $i$-th neuron respectively (see Sect. 2.1). We select three different fixed initial parameter sets, that is $S_f^1 = \{0.01, 0.5001, 0.2001\}$, $S_f^2 = \{0.08, 0.8001, 0.2901\}$ and $S_f^3 = \{-0.08, 0.01, 0.1201\}$, and make the correspond-

**Table 1** Statistical test of training errors with different initialization of parameters respectively for crude oil

| $S_f^1$ vs. $S_r$ | | | $S_f^2$ vs. $S_r$ | | | $S_f^3$ vs. $S_r$ | | |
|---|---|---|---|---|---|---|---|---|
| $H$ | $t$-value | Prob. $p$ | $H$ | $t$-value | Prob. $p$ | $H$ | $t$-value | Prob. $p$ |
| 0 | −0.0925 | 0.9263 | 0 | 0.1363 | 0.8916 | 0 | −0.8297 | 0.4991 |

ing statistical test on the total errors after training for RBFRT network with each of these fixed initial parameters and random ones. It is shown in Table 1 that, for all the error pairs with two different initialization of parameters, the values of double-tail test $p$ are larger than the significance level 0.05 and the values of $H$ are 0. Thus the null hypothesis is accepted that the errors for RBFRT with random initial parameters and fixed initial parameters have no significance difference.

In the follows, we study the predicting results of the proposed RBFRT model with the pair values of $(\mu(t), \sigma(t))$. Meanwhile, the comparisons of other three pair values of $(\mu(t), 0)$, $(0, \sigma(t))$ and $(0, 0)$ are also performed. Figure 4 shows the predicting values of the crude oil for the training set and test set with parameter value $(0, 0)$ in Fig. 4a, b and with parameter value $(\mu(t), \sigma(t))$ in Fig. 4c, d respectively. From these plots, the predicting values of the improved RBF network are more close to the actual values in intuitive sense. The predicting results of training and test data for SSE, N225 and DAX with the improved RBFRT model are also correspondingly given in Fig. 5. The curves of actual data and predictive data are intuitively very approximating.

The fluctuation behaviors of time series of relative errors for the crude oil, SSE, N225 and DAX are demonstrated in Fig. 6. In these plots, the time 0 represents the farthest data to the current data, and the larger $t$ represents the data that is closer to the current data. Figure 6 manifests that the random data-time effective RBF neural network can be realized by assigning different weights to the data of different time. Time sequences of relative errors of the crude oil and SSE in Fig. 6a, b also reflect the randomness of model by the effect of the Brownian motion. From the figure, we find that the relative errors of DAX are obviously smaller than those of other three time series, and the magnitude of all errors for DAX is lower than 0.1. Moreover, most of the predicting relative errors for these four price series are between −0.05 and 0.05.

In Table 2, the predictive values and the relative errors of the crude oil in the test set for a week are given for different values $\mu(t)$ and $\sigma(t)$. It exhibits that the relative error

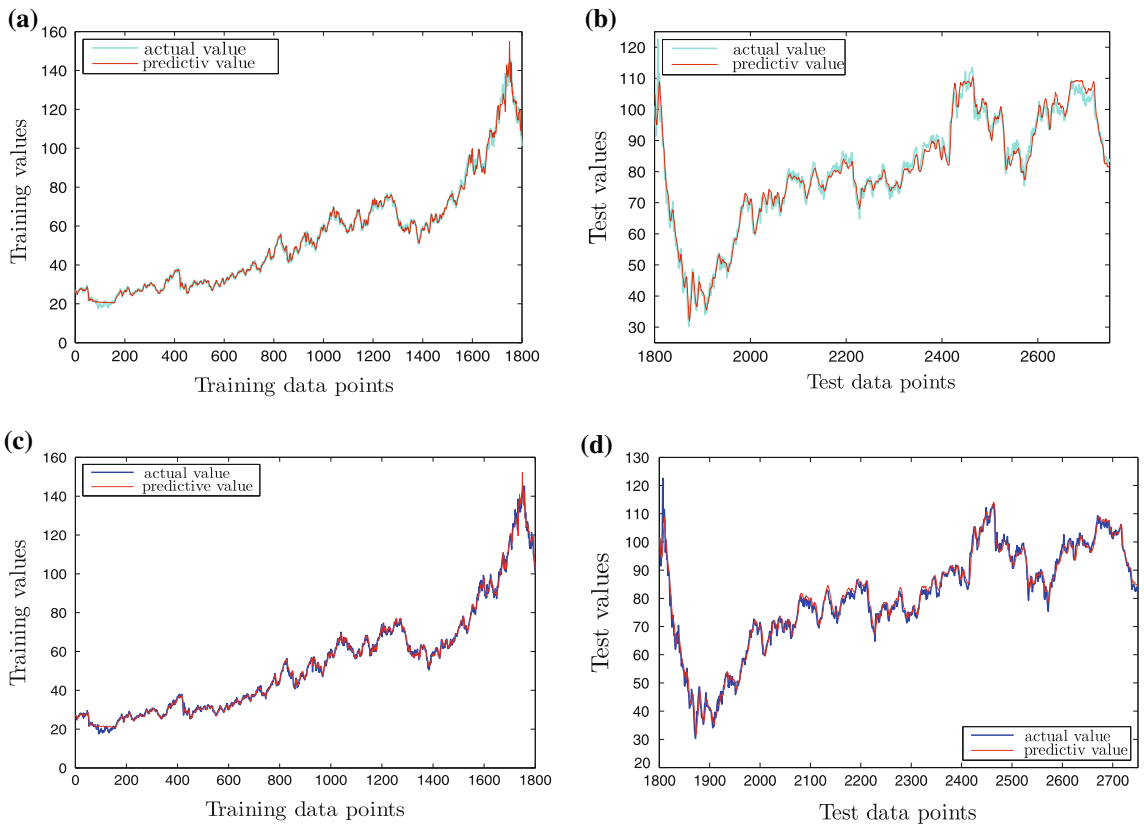


**Fig. 4** **a**, **b** The predicting results of the crude oil on the training data and the test data with $(\mu(t) = 0, \sigma(t) = 0)$. **c**, **d** The predicting results of the crude oil using training data and test data with the parameter $(\mu(t), \sigma(t))$

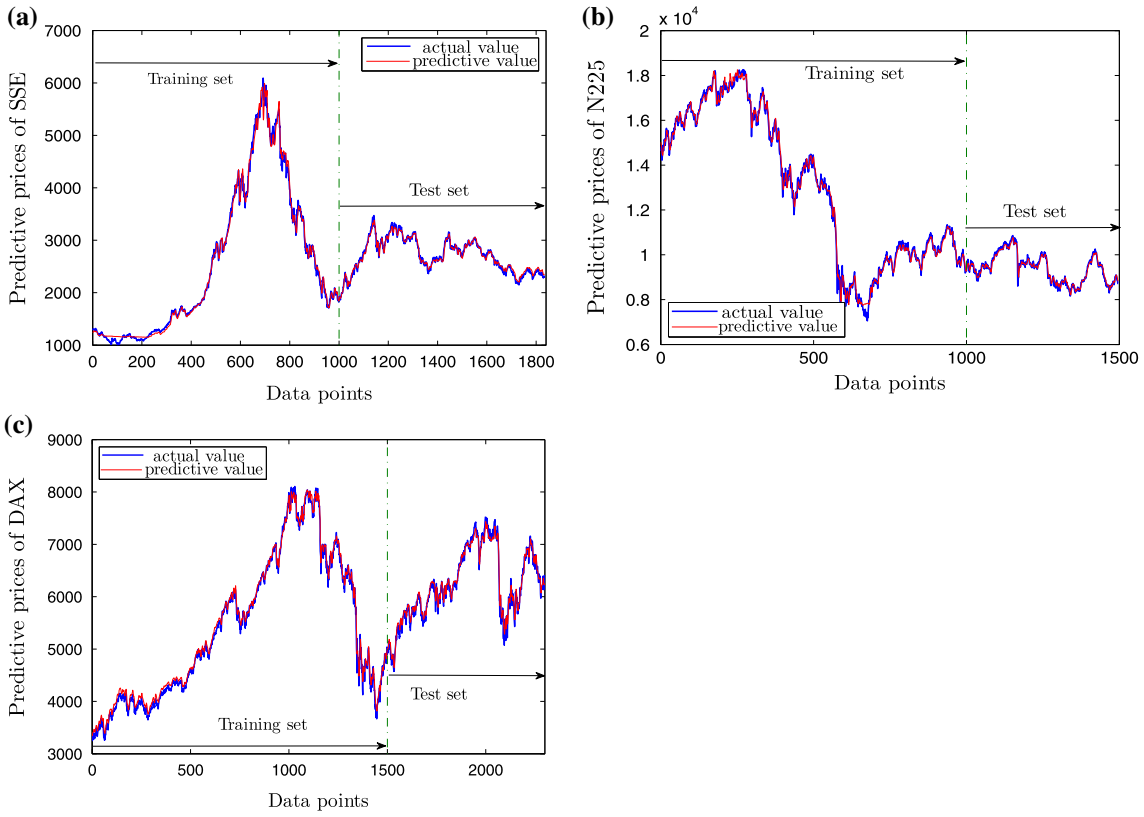


**Fig. 5** Predicting results of three actual stock indexes with the model RBFRT. **a** SSE, **b** N225, **c** DAX

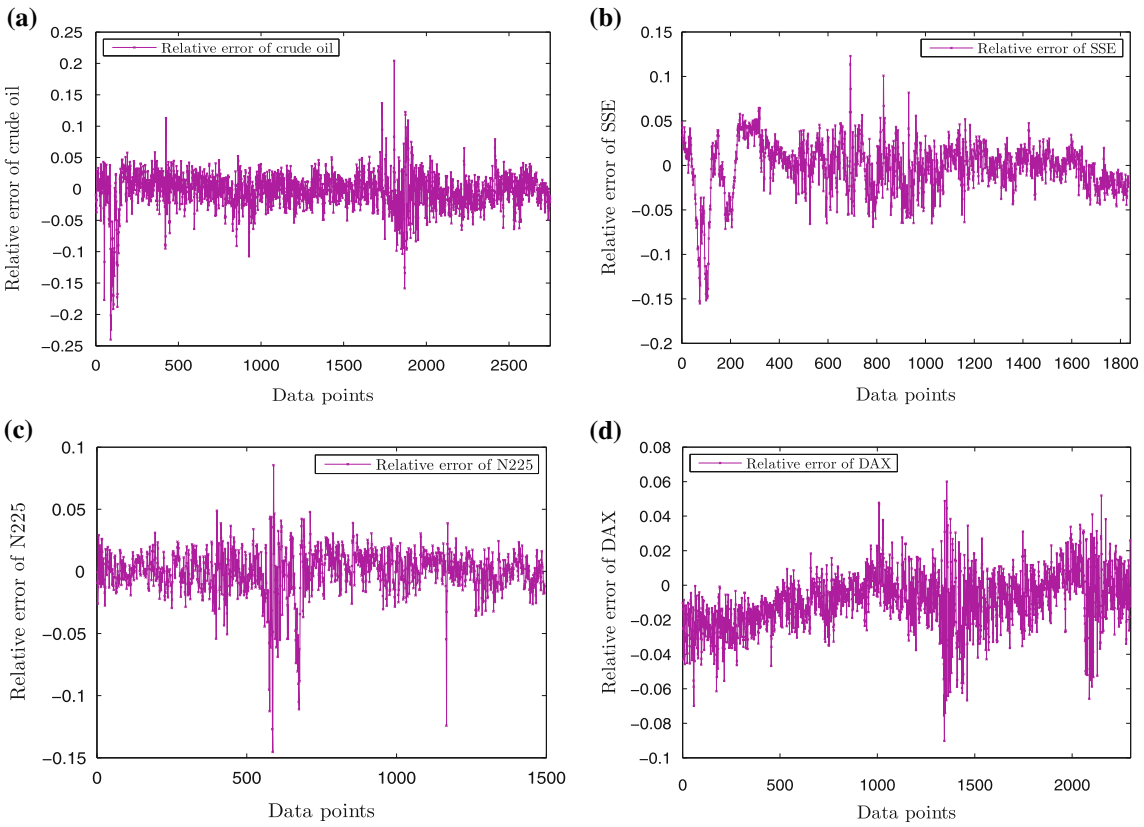


**Fig. 6** The plots of relative errors for **a** the crude oil, **b** SSE, **c** N225 and **d** DAX with the proposed RBFRT model

**Table 2** Predictive values and relative errors of the crude oil for different values of $(\mu(t), \sigma(t))$

| Time | Actual | Predictive | Error |
|---|---|---|---|
| $(\mu(t), \sigma(t))$ | | | |
| 2012/05/21 | 92.57 | 92.2287 | 0.0037 |
| 2012/05/22 | 91.44 | 91.9954 | −0.0061 |
| 2012/05/23 | 89.40 | 89.5747 | −0.0019 |
| 2012/05/24 | 90.36 | 91.0781 | −0.0079 |
| 2012/05/25 | 90.64 | 90.8144 | −0.0019 |
| $(0, \sigma(t))$ | | | |
| 2012/05/21 | 92.57 | 92.8687 | −0.0032 |
| 2012/05/22 | 91.44 | 92.4483 | −0.0110 |
| 2012/05/23 | 89.40 | 91.3108 | −0.0214 |
| 2012/05/24 | 90.36 | 91.1267 | −0.0084 |
| 2012/05/25 | 90.64 | 90.4506 | 0.0021 |
| $(\mu(t), 0)$ | | | |
| 2012/05/21 | 92.57 | 93.7608 | −0.0129 |
| 2012/05/22 | 91.44 | 90.4041 | 0.0113 |
| 2012/05/23 | 89.40 | 91.8128 | −0.0161 |
| 2012/05/24 | 90.36 | 92.1981 | −0.0313 |
| 2012/05/25 | 90.64 | 91.2418 | −0.0066 |
| (0,0) | | | |
| 2012/05/21 | 92.57 | 90.7523 | 0.0196 |
| 2012/05/22 | 91.44 | 93.3205 | −0.0205 |
| 2012/05/23 | 89.40 | 91.4946 | −0.0234 |
| 2012/05/24 | 90.36 | 89.2907 | 0.0118 |
| 2012/05/25 | 90.64 | 88.8663 | 0.0196 |

is the smallest when the pair value is $(\mu(t), \sigma(t))$ (below 1 %), and the relative error is the largest for the pair value (0, 0) (from 1 to 3 %). Take the date 2012/05/23 for instance, for the pair value $(\mu(t), \sigma(t))$, the magnitude of the relative error is 0.19 %; for the pair value $(0, \sigma(t))$, the magnitude of the relative error is 2.14 %; for the pair value $(\mu(t), 0)$, the magnitude of the relative error is 1.61 %; and for the pair value (0, 0), the magnitude of the relative error is 2.34 %. Therefore, this means that the developed drift function and volatility $(\mu(t), \sigma(t))$ in the neural network is advantageous for increasing the precision of forecasting.

Moreover, in Table 3, we give parts of testing values and relative errors of different testing dates for the crude oil, SSE, N225 and DAX with the improved RBFTR model of $(\mu(t), \sigma(t))$ respectively. Take the relative error of the crude oil for example, it is observable that the errors for the years 2008 and 2009 are larger than those in the years 2010 and 2011. The error values become smaller as the time goes on, this clearly shows the effect of the random data-time effective function. Likewise, the relative errors of SSE, N225 and DAX show the similar predicting behaviors for the testing data.

**Table 3** Comparisons of the relative errors of different testing data for the crude oil, SSE, N225 and DAX with the RBFRT model

| Time | Actual | Predictive | Error |
|---|---|---|---|
| Crude oil | | | |
| 2008/11/24 | 53.63 | 52.0758 | 0.0290 |
| 2009/11/27 | 75.95 | 77.1083 | −0.0153 |
| 2010/11/29 | 85.73 | 84.62 | 0.0129 |
| 2011/11/21 | 96.73 | 96.6777 | −0.0098 |
| SSE | | | |
| 2009/05/19 | 2,676.68 | 2,616.9795 | 0.0223 |
| 2010/05/21 | 2,583.52 | 2,552.0408 | 0.0122 |
| 2011/05/12 | 2,844.08 | 2,822.1233 | 0.0077 |
| 2012/05/21 | 2,348.30 | 2,359.1913 | −0.0047 |
| N225 | | | |
| 2010/09/13 | 9,321.82 | 9,191.6301 | 0.0139 |
| 2011/06/14 | 9,547.79 | 9,485.1204 | 0.0066 |
| 2012/06/14 | 8,568.89 | 8,613.3187 | −0.0052 |
| DAX | | | |
| 2009/06/08 | 5,004.72 | 5,077.6897 | −0.0146 |
| 2010/06/07 | 5,904.95 | 6,037.6047 | −0.0225 |
| 2011/06/08 | 7,060.23 | 7,010.8155 | 0.0069 |
| 2012/06/08 | 6,130.82 | 6,104.9193 | 0.0042 |

### 3.3 Predicting performance evaluation

To evaluate the forecasting accuracy of the proposed RBFRT model, we will compare the outputs of the model with different values of $\mu(t)$ and $\sigma(t)$ for the crude oil, SSE, N225 and DAX. First, we apply the following several error-type and trend-type performance measures to value the prediction performance. The mean absolute error MAE, the root mean square error RMSE, and the correlation coefficient $R$ are error-type measures used to estimate the forecasting accuracy. Directional symmetry (DS), correct up-trend (CP) and correct down-trend (CD) are the trend-type performance measures used to check the correct treading rate of the practical stock movement. The corresponding definitions of them are given as

$$\mathrm{MAE} = \frac{1}{l_1}\sum_{i=1}^{l_1}|y_i - o_i|,\ \mathrm{RMSE} = \sqrt{\frac{1}{l_1}\sum_{i=1}^{l_1}(y_i - o_i)^2},$$
$$R = \frac{\sum_{i=1}^{l_1}(y_i - \bar{y})(o_i - \bar{o})}{\sqrt{\sum_{i=1}^{l_1}(y_i - \bar{y})^2 \sum_{i=1}^{l_1}(o_i - \bar{o})^2}} \tag{16}$$

$$\mathrm{DS} = \frac{100}{l_1}\sum_{i=1}^{l_1} d_i,\quad d_i = \begin{cases} 1, & \text{If } (y_i - y_{i-1})(o_i - o_{i-1}) \geq 0 \\ 0, & \text{Otherwise} \end{cases} \tag{17}$$

$$\mathrm{CP}=\frac{100}{l_2}\sum_{i=1}^{l_2} d_i,\quad d_i=\begin{cases}1, & \text{If } (y_i-y_{i-1})>0 \text{ and}\\ & (y_i-y_{i-1})(o_i-o_{i-1})\geq 0\\ 0, & \text{Otherwise}\end{cases} \tag{18}$$

$$\mathrm{CD}=\frac{100}{l_3}\sum_{i=1}^{l_3} d_i,\quad d_i=\begin{cases}1, & \text{If } (y_i-y_{i-1})<0 \text{ and}\\ & (y_i-y_{i-1})(o_i-o_{i-1})\geq 0\\ 0, & \text{Otherwise}\end{cases} \tag{19}$$

where $y$ is the actual value, $o$ is the predictive value, $\bar{y}$ is the mean of the actual values, $\bar{o}$ is the mean of the predictive values, $l_1$ denotes the number of the evaluated data, $l_2$ is the number of data for $(y_i - y_{i-1}) > 0$ and $l_3$ is the number of data for $(y_i - y_{i-1}) < 0$. The smaller MAE value and RMSE value and the larger $R$ value show the less deviation of the forecasting results from the actual values. The larger the values of DS, CP, CD, the closer are the predictive values to those of the actual ones.

In Table 4, the values of error-type measures MAE, RMSE, $R$ and trend-type measures DS, CP, CD for the four time series with different $(\mu(t), \sigma(t))$ are presented. These training and testing examples illustrate the forecasting

**Table 4** Predicting performance of the crude oil with different $(\mu(t), \sigma(t))$

| Method | Crude oil | | | | | | SSE | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MAE | RMSE | $R$ | DS | CP | CD | MAE | RMSE | $R$ | DS | CP | CD |
| $(\mu(t), \sigma(t))$ | | | | | | | | | | | | |
| Training | 0.9512 | 1.4571 | 0.9986 | 67.98 | 72.12 | 62.93 | 64.2073 | 92.4743 | 0.9989 | 69.78 | 70.86 | 66.19 |
| Test | 1.5686 | 2.1248 | 0.9934 | 67.86 | 67.92 | 67.74 | 39.0553 | 48.3032 | 0.9917 | 66.15 | 68.78 | 63.17 |
| Total | 1.1645 | 1.7174 | 0.9981 | 67.92 | 70.64 | 64.35 | 52.7696 | 75.6556 | 0.9975 | 68.88 | 66.74 | 65.59 |
| $(0, \sigma(t))$ | | | | | | | | | | | | |
| Training | 0.9693 | 1.4799 | 0.9985 | 67.65 | 71.69 | 62.22 | 50.1132 | 91.2593 | 0.9984 | 67.57 | 64.95 | 63.79 |
| Test | 1.7651 | 2.3541 | 0.9919 | 67.76 | 67.71 | 67.74 | 41.9480 | 51.5778 | 0.9918 | 66.15 | 68.78 | 63.17 |
| Total | 1.2442 | 1.8297 | 0.9978 | 67.66 | 70.29 | 64.20 | 56.4018 | 69.3989 | 0.9980 | 68.03 | 69.94 | 63.49 |
| $(\mu(t), 0)$ | | | | | | | | | | | | |
| Training | 1.0104 | 1.4575 | 0.9986 | 67.59 | 70.94 | 62.46 | 64.8304 | 95.9129 | 0.9977 | 68.97 | 70.10 | 68.35 |
| Test | 1.6989 | 2.2217 | 0.9926 | 67.54 | 67.71 | 67.52 | 39.1226 | 51.0344 | 0.9891 | 65.67 | 68.33 | 62.66 |
| Total | 1.2482 | 1.7594 | 0.9980 | 67.55 | 70.22 | 64.58 | 54.7757 | 78.7423 | 0.9973 | 67.48 | 67.15 | 64.23 |
| $(0, 0)$ | | | | | | | | | | | | |
| Training | 1.3974 | 1.9810 | 0.9974 | 67.32 | 71.58 | 61.62 | 67.4750 | 97.7902 | 0.9979 | 68.78 | 68.10 | 65.23 |
| Test | 1.5676 | 2.3836 | 0.9914 | 67.76 | 67.50 | 67.74 | 41.9518 | 51.7079 | 0.9904 | 65.15 | 68.78 | 62.17 |
| Total | 1.3050 | 2.0450 | 0.9973 | 67.44 | 69.72 | 63.81 | 55.3259 | 60.9547 | 0.9964 | 68.14 | 69.52 | 63.73 |
| Method | N225 | | | | | | DAX | | | | | |
| | MAE | RMSE | $R$ | DS | CP | CD | MAE | RMSE | $R$ | DS | CP | CD |
| $(\mu(t), \sigma(t))$ | | | | | | | | | | | | |
| Training | 163.4398 | 210.4131 | 0.9980 | 63.66 | 63.65 | 63.27 | 75.3680 | 92.8449 | 0.9986 | 66.38 | 72.49 | 58.94 |
| Test | 90.6990 | 121.2587 | 0.9835 | 62.42 | 64.52 | 60.32 | 72.8157 | 95.2294 | 0.9898 | 65.25 | 69.57 | 60.26 |
| Total | 139.3226 | 185.6602 | 0.9983 | 63.38 | 63.98 | 62.28 | 74.5372 | 93.6277 | 0.9980 | 66.00 | 71.51 | 59.69 |
| $(0, \sigma(t))$ | | | | | | | | | | | | |
| Training | 171.3568 | 218.8644 | 0.9971 | 63.26 | 63.64 | 63.06 | 77.8382 | 95.9645 | 0.9985 | 66.04 | 71.76 | 58.79 |
| Test | 98.0276 | 131.2149 | 0.9817 | 62.42 | 64.52 | 60.32 | 76.0376 | 99.3182 | 0.9902 | 65.24 | 69.81 | 60.00 |
| Total | 160.8851 | 219.2122 | 0.9977 | 63.08 | 63.72 | 62.14 | 78.9675 | 97.5633 | 0.9978 | 65.78 | 71.35 | 59.36 |
| $(\mu(t), 0)$ | | | | | | | | | | | | |
| Training | 180.1834 | 230.9335 | 0.9976 | 63.36 | 64.26 | 62.65 | 78.0387 | 99.3755 | 0.9983 | 65.84 | 72.25 | 57.90 |
| Test | 109.6594 | 121.3190 | 0.9832 | 62.42 | 64.52 | 60.32 | 77.5281 | 96.0948 | 0.9908 | 65.22 | 70.53 | 60.00 |
| Total | 144.7141 | 192.0929 | 0.9979 | 63.28 | 64.64 | 61.87 | 80.4320 | 98.2487 | 0.9977 | 65.78 | 71.67 | 58.36 |
| $(0, 0)$ | | | | | | | | | | | | |
| Training | 192.0625 | 251.6922 | 0.9978 | 63.16 | 64.24 | 63.06 | 81.9770 | 102.1632 | 0.9982 | 65.58 | 72.00 | 57.61 |
| Test | 127.4936 | 163.5150 | 0.9761 | 62.42 | 64.52 | 60.32 | 81.0998 | 100.4998 | 0.9920 | 65.12 | 70.05 | 59.47 |
| Total | 162.2699 | 210.9818 | 0.9982 | 63.01 | 64.38 | 61.17 | 85.2603 | 97.4939 | 0.9977 | 65.43 | 71.10 | 58.32 |

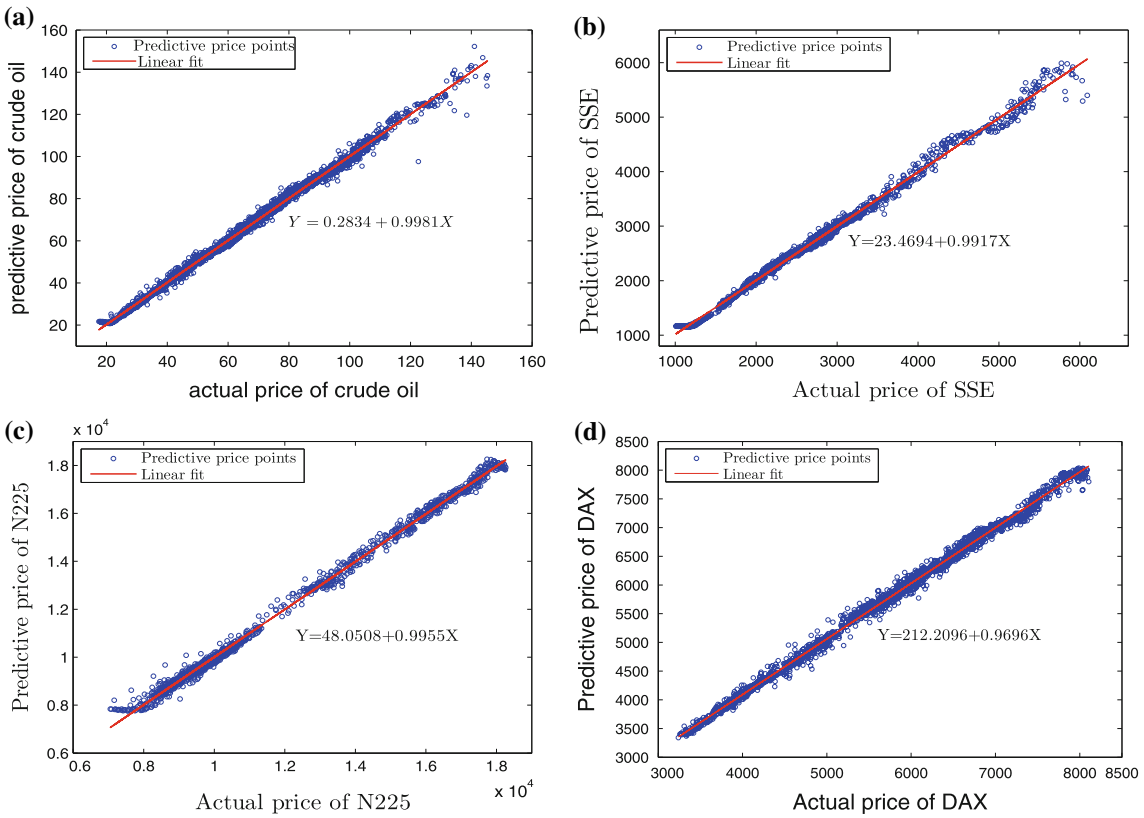


**Fig. 7** The comparison and linear regression of the actual data and the predictive value for the crude oil, SSE, N225 and DAX

accuracy and tendency with six measure-types under four prediction cases. Take the crude oil for example, we can see from the table that the proposed approach has improved the forecasting ability. Both for the training set and the test set, the measure values MAE and RMSE of the crude oil for the improved RBFRT model are smaller than those for other three cases, while the values of $R$ for it are larger than those for other three cases. In the training and test set, the tend-type measures for RBFRT models are almost all larger than those for other three different cases, that is DS = 67.98, CP = 72.12, CD = 62.93 in training period and DS = 67.86, CP = 67.92, CD = 67.74 in test period. This indicates a better predicting performance for the random data-time effective RBF neural network. Meanwhile, the performances for SSE, N225 and DAX shows the similar trends, which suggests a quite well predicting performance for the RBFRT model.

The plots of the actual and the predictive data for these four price sequences are respectively shown in Fig. 7. Through the linear regression analysis, we make a comparison of the predictive value of the improved RBF neural network with the actual value. It is known that the linear regression can be use to fit a predictive model to an observed data set of $Y$ and $X$. The linear equations of the crude oil, SSE, N225 and DAX are exhibited respectively in Fig. 7a–d. We can observe that all the slopes of the linear equations for them are drawing near to 1, which implies that the predictive values and the actual values are not deviating too much.

## 4 Extension

Since the random data-time effective function which is embedded in the gradient algorithm in the proposed improved model, is independent of the neural network itself, it can show that the improved predicting algorithm with a random data-time effective function could also be extended to many other neural networks, whose training is done by a gradient-based learning method where the learning error is propagated backwards through the network. For instance, multilayer perceptron (MLP) is such one of powerful nonlinear modelling tools, which has one or more hidden layers. The structures of MLP and RBF network are very similar. The major difference between them is the behavior of the single hidden layer (Jayawardema 1997; Memarian and Balasundram 2012). Rather than using the Gauss-based function in RBF network, the hidden units in MLP use two main sigmoidal activation functions which can be described as follows:

$$\phi(i) = \tanh(\text{net}_i), \quad \phi(i) = \frac{1}{1 + \exp\{-\text{net}_i\}} \tag{20}$$

where the former function is a hyperbolic tangent ranging from $-1$ to 1, and the latter is a logistic function similar in shape but ranges from 0 to 1. Here, $\phi(i)$ is the output of the $i$th neuron and $\text{net}_i$ is the weighted sum of the input synapses. A comprehensive discussion on MLP can be found in Popescu et al. (2009).

## 5 Conclusion

In the present paper, we introduce a random data-time effective function in the three-layer RBF neural network to modify the network's parameters, the output weights, the center vector and the widths in the hidden layer. In this random data-time effective function, we consider the timely effectiveness of $\mu(t)$ and the random volatility of $B(t)$, since we think that the data in the training set should be time-variant such that it can reflect different behavior patterns of financial market at different time. The predicting results and its effectiveness are demonstrated through applying the improved RBF neural network to the financial time series forecasting. We select four financial series, the crude oil, SSE, N225 and DAX, to test their predicting accuracy and to study the impact of random data-time effective function with different pair values of $(\mu(t), \sigma(t))$. Empirical examinations of the predicting precision for price series (by the comparison of the relative errors and the predicting measures as MAE, RMSE and $R$) show that the proposed random data-time effective function in RBF neural network has the advantage of improving the precision of forecasting, and the volatility of the financial model much approaches to the actual financial market movement. We hope that the proposed model can make some beneficial contributions to ANN research and its application in the time series forecasting.

**Acknowledgments** The authors were supported in part by National Natural Science Foundation of China Grant No. 71271026 and Grant No. 10971010.

## References

Ao SI (2011) A hybrid neural network cybernetic system for quantifying cross-market dynamics and business forecasting. Soft Comput 15:1041–1053

Azoff EM (1994) Neural network time series forecasting of financial market. Wiley, New York

Babbar N, Kumar A, Bansal A (2013) Solving fully fuzzy linear system with arbitrary triangular fuzzy numbers $(m, \alpha, \beta)$. Soft Comput 17:691–702

Bahrammirzaee A (2010) A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert system and hybrid intelligent systems. Neural Comput Appl 19:1165–1195

Bors AG, Gabbouj M (1994) Minimal topology for a radial basis function neural network for pattern classification. Digit Signal Process 44:173–188

Box GEP, Jenkins GM, Reinsel GC (1994) Time series analysis: forecasting and control, 3rd edn. Prentice Hall, New Jersey

Broomhead DS, Low D (1988) Multivariable functional interpolation and adaptive networks. Complex Syst 2:321–355

Cao LJ, Tay EH (2001) Support vector with adaptive parameters in financial time series forecasting. IEEE Trans Neural Netw 14:1506–1518

Chaturvedi DK, Satsangi PS, Kalra PK (1996) Effect of different mappings and normalization of neural network models, vol 1. Ninth national power systems conference. Indian institute of Technology, Kanpur, pp 377–386

Demuth H, Beale M (2002) Neural network toolbox for use with MATLAB. Mathworks Inc, USA

Devaraj D, Yegnanarayana B, Ramar K (2002) Radial basis function networks for fast contingency ranking. Electric Power Energy Syst 24:387–395

Dhamija AK (2010) Financial time series forecasting: comparison of neural networks and ARCH models. Int Res J Fin Econ 49:185–202

Di Martino F, Loia V, Sessa S (2010) Fuzzy transforms method in prediction data analysis. Fuzzy Sets Syst 180:146–163

Flake GW, Lawrence S (2002) Efficient SVM regression training with SMO. Mach Learn 46:271–290

Gacto M, Alcal R, Herrera F (2009) Adaptation and application of multi-objective evolutionary algorithms for rule reduction and parameter tuning of fuzzy rule-based systems. Soft Comput 13:419–436

Garg S, Patra K, Pai SK, Chakraborty D (2008) Effect of different basis functions on a radial basis function network in prediction of drill flank wear from motor current signals. Soft Comput 12:777–787

Grabusts PS (2001) A study of clustering algorithm application in RBF neural networks. Inf Technol Manage Sci 5:50–57

Guo ZQ, Wang HQ, Liu Q (2013) Financial time series forecasting using LPP and SVM optimized by PSO. Soft Comput 17:805–818

Hansen JV, McDonald JB, Nelson RD (1999) Time series prediction with genetic-algorithm designed neural networks: an empirical comparison with modern statistical models. Comput Intell 15:171–184

Harpham C, Dawson CW (2006) The effect of different basis function on a radial function network for time series prediction: a comparative study. Neurocomputing 69:2161–2170

Haykin S (1999) Neural networks: a comprehensive foundation. Prentice-Hall, Englewood Cliffs

Harrison M (1990) Brownian motion and stochastic flow systems. Krieger Publishing Company, Malabar

He Q, Wu C (2011) Membership evaluation and feature selection for fuzzy support vector machine based on fuzzy rough sets. Soft Comput 15:1105–1114

Jareanpon C, Pensuwon W, Frank RJ, Davey N (2004) An Adaptive RBF Network optimised using a genetic algorithm applied to rainfall forecasting. Int Sympos Commun Inf Technol 2004:1005–1010

Jayawardema AW, Fernando DAK, Zhou MC (1997) Comparison of Multilayer Perceptron and Radial Basis Function networks as tools for flood forecasting. Destructive water: water-caused natural disasters, their abatement and control, vol 239, pp 173–181

Kaastra I, Boyd M (1996) Designing a neural network for forecasting financial and economic time series. Neurocomputing 10:215–236

Karayiannis NB (1999) Reformulated radial basis neural networks trained by gradient descent. IEEE Trans Neural Netw 10:657–671

Liao Z, Wang J (2010) Forecasting model of global stock index by stochastic time effective neural network. Expert Syst Appl 37:834–841

Liu HF, Wang J (2011) Integrating independent component analysis and principal component analysis with neural network to predict Chinese stock market. Mathematical Problems in Engineering 2011 (Article ID 382659)

Memarian H, Balasundram SK (2012) Comparison between multi-layer perceptron and radial basis function networks for sediment load estimation in a tropical watershed. J Water Resour Protect 4:870–876

Meyer BD, Saley HM (2002) On the strategic origin of Brownian motion in finance. Int J Game Theory 31:285–319

Nekoukar V, Beheshti MTH (2001) A local linear radial basis function neural network for financial time-series forecasting. Appl Intell 33:352–356

Niros AN, Tsekouras GF (2012) A novel training algorithm for RBF neural network using a hybrid Fuzzy clustering approach. Fuzzy Sets Syst 193:62–84

Niu H, Wang J (2013) Volatility clustering and long memory of financial time series and financial price model. Digit Signal Process 23:489–498

Oconnor N, Madden MG (2006) A neural network approach to predicting stock exchange movements using external factors. Knowl Based Syst 19:371–378

Oyang YJ, Hwang SC, Ou YY, Chen CY, Chen ZW (2005) Data classification with radial basis function networks based on a novel kernel density estimation algorithm. IEEE Trans Neural Netw 16:225–236

Pino R, Parreno J, Gomez A, Priore P (2008) Forecasting next-day price of electricity in the Spanish energy market using artificial neural networks. Eng Appl Artif Intell 21:53–62

Popescu MC, Balas VE, Perescu-Popescu L, Mastorakis N (2009) Multilayer perceptron and neural networks. WSEAS Trans Circuits Syst 7:579–588

Pouzols FM, Lendasse A, Barros AB (2010) Autoregressive time series prediction by means of fuzzy inference systems using nonparametric residual variance estimation. Fuzzy Sets Syst 161:471–497

Rojas I, Pomares H, Gonzalez J, Ros A (2000) A new radial basis function networks structure: application to time series prediction. IEEE INNS ENNS IJCNN 4:449–454

Samsudin R, Shabri A, Saad P (2010) A comparison of time series forecasting using support vector machine and artificial neural network model. J Appl Sci 10:950–958

Sola J, Sevilla J (1997) Importance of input data normalization for the application of neural networks to complex industrial problems. IEEE Trans Nuclear Sci 44:1464–1468

Sun YF, Liang YC, Zhang WL (2005) Optimal partition algorithm of the RBF neural network and its application to financial time series forecasting. Neural Comput Appl 14:36–44

Trippi RR, Turban E (1993) Neural networks in finance and investing: using artificial intelligence to improve real-world performance.Probus, Chicago

Virili F, Freisleben B (2001) Neural network model selection for financial time series prediction. Comput Stat 16:451–463

Wang F, Wang J (2012) Statistical analysis and forecasting of return interval for SSE and model by lattice percolation system and neural network. Comput Ind Eng 62:198–205

Wang J (2007) Stochastic process and its application in finance. Tsinghua University Press and Beijing Jiaotong University Press, Beijing

Wang J, Deng S (2008) Fluctuations of interface statistical physics models applied to a stock market model. Nonlinear Anal Real 9:718–723

Yaser SAM, Atiya AF (1996) Introduction to financial forecasting. Appl Intell 6:205–213

Yu L, Wang SY, Lai KK (2009) A neural-network-based nonlinear metamodeling approach to financial time series forecasting. Appl Soft Comput 9:563–574

Zheng GL, Billings SA (1999) Radial basis function network configuration using mutual information and the orthogonal least squares algorithm. Neural Netw 9:1619–1637