Received 6 October 2018
Revised 21 March 2019
Accepted 3 April 2019

Utility of a Shuffled Differential Evolution algorithm in designing of a Pi-Sigma Neural Network based predictor model

Rajashree Dash, Rasmita Rautray and Rasmita Dash
Department of Computer Science and Engineering,
ITER, Siksha O Anusandhan (Deemed to be University), Bhubaneswar, India

Abstract

Since the last few decades, Artificial Neural Networks have been the center of attraction of a large number of researchers for solving diversified problem domains. Due to its distinguishing features such as generalization ability, robustness and strong ability to tackle nonlinear problems, it appears to be more popular in financial time series modeling and prediction. In this paper, a Pi-Sigma Neural Network is designed for foretelling the future currency exchange rates in different prediction horizon. The unrevealed parameters of the network are interpreted by a hybrid learning algorithm termed as Shuffled Differential Evolution (SDE). The main motivation of this study is to integrate the partitioning and random shuffling scheme of Shuffled Frog Leaping algorithm with evolutionary steps of a Differential Evolution technique to obtain an optimal solution with an accelerated convergence rate. The efficiency of the proposed predictor model is actualized by predicting the exchange rate price of a US dollar against Swiss France (CHF) and Japanese Yen (JPY) accumulated within the same period of time.

Keywords Neural Network, Evolutionary learning techniques, Differential Evolution, Shuffled Differential Evolution

Paper type Original Article

1. Introduction

Currency Exchange rate is a conversion factor that represents the currency quotation of one nation with respect to another one. It is one of the leading factors that determine the relative level of a country's economic stability in the global monetary market. Any valuable investment and trading decisions that are taken by the government or companies go through the analysis of these dynamic exchange rate values. So accurate forecasting of foreign exchange rates has a great importance on forecasting the behavior of an economy. The complicated nonlinear structure and rapid variations of these time series data make it hard to

Publishers note: The publisher wishes to inform readers that the article "Predictive modelling and analytics for diabetes using a machine learning approach" was originally published by the previous publisher of *Applied Computing and Informatics* and the pagination of this article has been subsequently changed. There has been no change to the content of the article. This change was necessary for the journal to transition from the previous publisher to the new one. The publisher sincerely apologises for any inconvenience caused. To access and cite this article, please use Shashikant, R., Chetankumar, P. (2020), "Predictive modelling and analytics for diabetes using a machine learning approach", *Applied Computing and Informatics*, Vol. ahead-of-print No. ahead-of-print. https://10.1016/j.aci.2018.12.004. The original publication date for this paper was 22/06/2019.

Applied Computing and Informatics
Vol. 19 No. 1/2, 2023
pp. 22-40
Emerald Publishing Limited
e-ISSN: 2210-8327
p-ISSN: 2634-1964
DOI 10.1016/j.aci.2019.04.001

forecast its future behavior. Still, the financial benefits fascinated by accurate prediction of currency exchange rates have encouraged researchers for developing newer and advanced tools and models.

Over the years, a quite significant exploration of time series models, computational intelligence techniques and hybrid techniques are accomplished in the domain of exchange rate prediction. Artificial Neural Network (ANN), being the mainstream of computational intelligent techniques has gained more popularity in a large variety of modeling and forecasting problems. The wider usage of ANN is due to its distinguishing features such as generalization ability, robustness and strong ability to tackle nonlinear problems. In several studies, ANN has clearly outperformed the time series models for daily, weekly or monthly exchange rate predictions [1–3]. Previously proposed time series models are based on the assumption of correlated and linear nature of input data, whereas real-life data are more dynamic and nonlinear. So they may fail to come up with accurate predictions. Ye et al., in [4] have proposed a dynamic backpropagation neural network model and an Auto Regressive Moving Average (ARMA) model to forecast the RMB exchange rate. The results reveal the better performance of ANN over ARMA for both the trend and deviation prediction. Pedram and Ebrahimi, in [5] have compared the performance of an ANN with an Auto Regressive Integrated Moving Average (ARIMA) model for a short-term forecast of daily USD to Rial exchange rate. The results have clearly shown that the error statistics observed from ANN is nearly half of the ARIMA model. In [6], Adebiyi et al. did a similar comparison and got better accuracy with ANN than the ARIMA model. The comparative study carried out by Kamruzzan & Sarker in [2] also yields better performance of ANN over ARIMA for prediction of six currencies against the Australian dollar. In last few decades, Radial Basis Function Neural Network (RBF) [7], Functional Link Artificial Neural Network (FLANN) [8,9], Multi Layer Perception Network (MLP) [10], Pi-Sigma Neural Network (PSNN) [11] are the different types of ANN, that have been successfully tested for forecasting currency exchange rates. Still, identifying the unknown weights of an ANN is appearing as a challenging issue for designing an efficient predictor model. The traditional backpropagation algorithm with a gradient descent method is the commonly used learning technique for ANNs. But it suffers from the issues of imprecise learning rate, local minimal and slow rate of convergence [12–14]. To avoid the common drawbacks of back propagation algorithm and to increase the optimality of network performance, several researchers have proposed meta-heuristic methods in the training phase of the neural networks. To increase forecasting speed and accuracy, researchers have also tried to combine and optimize different algorithms and build hybrid models.

In this study, the author has designed a PSNN based predictor model with a Shuffled Differential Evolution (SDE) based network learning algorithm, to foretell the successive currency exchange rate prices. The evolutionary steps of a Differential Evolution (DE) technique is integrated with the partitioning and random shuffling scheme of Shuffled Frog Leaping Algorithm (SFLA) in proposed SDE learning algorithm. Shuffled complex evolution framework originally proposed by Duan et al. in [15,16] is a global-searching algorithm in which the population is partitioned into different complexes and each complex evolves for a specific number of iterations independently and then all are forced to shuffle. Mariani et al. in [17] have proposed a Modified Shuffled Complex Evolution (MSCE) optimizer by combining the DE with the original shuffled complex evolution for solving unconstrained optimization problems. The authors have used the DE/rand/1 mutation strategy in the evolution of subpopulations. They have suggested a dynamic adaptation for the mutation scale of DE with a linear decrease of values from 0.7 to 0.3 and a random selection of crossover rate between 0.2 and 0.8 during the optimization cycle, to improve the balance between the exploration and exploitation in DE. Testing over six benchmark functions the proposed algorithm is found to give better result compared to original shuffled complex evolution algorithm. Again Reddy, & Vaisakh, in [18,19] have proposed a combination of the shuffled

frog leaping algorithm and differential evolution for solving economic dispatch problem. Instead of using DE/rand/1 mutation strategy they have used DE/best/2 mutation strategy in the evolution of subpopulations. A fixed crossover rate and scaling factor are chosen through simulation for solving the problem. Further Naeini et al., in [20] have modified the SDE algorithm using three mutation rates in three attempts. Initially, a larger mutation rate is applied to explore the search space with larger jump rates. Then in the second attempt, the mutation rate is reduced to a quarter of first value to enhance the exploitation capability. If a better offspring is not possible with the two mutation rates then in next attempt mutation rate is set to half of the first attempt. Lastly, if none of these attempts produce a better offspring, a new point is selected randomly. In contrast to the above three approaches, in this paper, the authors have used the DE/Current to best/1 Mutation strategy in the evolution process of the subpopulation. They have also provided an adaptation scheme for mutation scale and crossover rate to keep a balance between exploration and exploitation. In the early stage of iteration, the search space is needed to be explored more whereas in the convergence end it requires less exploration. So the two controlling parameters of DE such as mutation scale and crossover are linearly decreased from a suitable higher to lower value through the iterative steps. The hybrid learning algorithm is suggested to achieve an optimal trained network with a faster convergence speed and improved predictive ability. The main motivation of the study is to explore the utility of PSNN trained using SDE for predicting currency exchange rates. A comparative assessment of the network with other learning techniques such as DE, SFL and Particle Swarm Optimization (PSO) algorithm is actualized by predicting the exchange rate price of USD/CHF and USD/JPY in different prediction horizons.

The remaining part of the paper is organized as follows. Section 2 focuses on a survey of neural networks and suggested meta-heuristic learning techniques for the neural network in the domain of currency exchange rate prediction. Section 3 describes the details of PSNN-SDE based prediction framework. Simulation study depicting the comparative assessment of different predictor models is analyzed in Section 4, followed by the conclusion in Section 5.

2. Literature survey

In the former few decades, Artificial Neural Network (ANN) has opened up a new dimension in developing and modeling efficient prediction models for financial time series data. In [7] an ensemble forecasting model is presented by combining a multistage RBFN with a Conditional Generalized Variance (CGV) minimization method for extracting relevant ensemble members of the model. The ensemble forecasting model has shown better level and direction measurements than four ensemble forecasting models and an RBF model. Majhi et al., in [8] have presented a two-stage cascaded functional link neural network, which has shown better monthly prediction results compared to the LMS and FLANN model. Further compared to individual LMS and FLANN model, Jena et al., in [9] have shown acceptably improved prediction results of a KGANN in monthly exchange rate prediction. Chen et al., in [21] have shown better performance of a Generalized Regression Network (GRNN) compared to the random walk, multivariate transfer function and MLP in predicting monthly exchange prices. In [22] Ni and Yin suggested a hybrid model by blending a Recurrent Self-Organizing Map (RSOM) with support vector regression and genetic algorithm for foreign exchange rate prediction. A hybrid Differential EMD based SVR Model has left behind traditional MS-GARCH and Markov Switching Regression (MSR) in predicting nonlinear currency exchange rates [23]. Rehman et al., in [24] suggested an ANN with Cartesian Genetic Programming, by adopting best input features with network pattern for prediction of foreign exchange rates

Identifying the unknown weights is one of the important issues in designing an efficient ANN based predictor model. To increase the optimality of network performance, several

researchers have proposed meta-heuristic methods such as Artificial Fish Swarm algorithm [25], SFL [11,13,26,27], PSO [28], DE [29], Differential Harmony Search (DHS) [12,30] and so on in training phase of the neural networks. Eusuff and Lansey, in [31] have proposed the SFL algorithm induced by the activity of a group of frogs exploring for their food. Originally the algorithm is applied for pipe network optimization problem and later for discrete optimization problem in [32]. SFLA has successfully applied in several optimization problems such as optimal generation expansion planning problem [33], channel equalization problem [26], multi-depots vehicle routing problems [34,35], flop shop scheduling problem [36] and so on. Similarly, differential evolution based on recombination, mutation, and selection as its key operations has also appeared as one of the popular stochastic function optimizers. Though quick convergence and smaller parameter space size are the benefits of DE, still it may suffer from the problem of stagnation. Inspired by the arithmetic operators used in the evolutionary step of DE and the partitioning and randomly shuffling scheme used in SFLA, in this study, a novel Shuffled Differential Evolution (SDE) algorithm is proposed as a learning algorithm for PSNN. The SDE algorithm is designed to further balance the exploitation and exploration abilities of traditional DE algorithm. With the hybridized approach the population diversity will be increased during the evolution process by providing a better information sharing approach between different individuals [17–20,37].

3. Proposed hybrid predictor model using Pi Sigma Neural Network with Shuffled Differential Evolution based learning algorithm (PSNN-SDE)

This section outlines the detailed architecture of PSNN; SDE based learning approach and the detailed steps of PSNN-SDE for future currency exchange rate prediction.

3.1 Pi-Sigma Neural Network

The structure of Pi-Sigma Neural Network (PSNN) is equivalent to a feedforward network having 3 tiers: one specifying input layer with one node corresponding to each input variable. The next level containing summation units represent the hidden layer and the last layer with the product unit represents the output layer. PSNN has successfully applied in several other domains [38–41]. Figure 1 depicts the structure of a PSNN. A fully connected PSNN represents a weighted connection of lower level neurons with all upper-level neurons. Summation unit with an activation function is used in the output layer to capture the nonlinearity exist between input and output space. Achieving an accurate result with a smaller number of processing units is one of the key features of PSNN. It also leads to reduced learning time of the network. The number of processing neurons in the hidden layer specifies the order of PSNN.

Any d dimensional input pattern $IP = [ip_1, ip_2 \ldots ip_d]^T$ when passed through p number of summation units, produces a weighted output with a bias as follows:

$$O(h_c) = B_c + \sum_{r=1}^{d} w_{rc} ip_r \quad for\ c\ in\ [1, p] \tag{1}$$

where h_c represents the summation neuron of the hidden layer; w_{rc} are the weights between input and hidden nodes; ip_r represents the input node. An output of the product unit z, with fixed weight value 1 considered between hidden and output layer, is worked out as follows:

$$z = \prod_{c=1}^{p} O(h_c) = \prod_{c=1}^{p} \left(B_c + \sum_{r=1}^{d} w_{rc} ip_r \right) \quad for\ c\ in\ [1, p] \tag{2}$$

Then the final output y is obtained by using the hyperbolic tangent (tanh ( )) as the activation function S ( ) as follows:

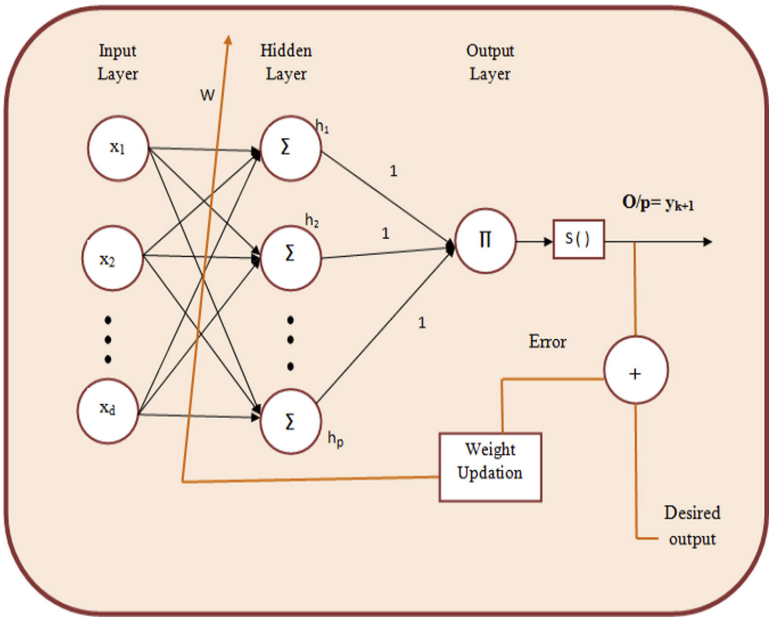

Figure 1.
Detailed Structure of Pi-Sigma Neural Network (PSNN).

$$y = S(z) = S\left(\prod_{c=1}^{p}\left(B_c + \sum_{r=1}^{d} w_{rc} ip_r\right)\right) \tag{3}$$

3.2 Learning algorithms for PSNN

Estimating the unseen weights by minimizing prediction error is one of the challenging issues in the design of PSNN. Hence parameter estimation of PSNN can be cast as an optimization problem with a suitable error metric as an objective function. In this application, minimizing Root Mean Squared Error (RMSE) is set as an objective function, which is defined as follows:

$$E(x) = RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(y_k - \hat{y}_k)^2} \tag{4}$$

where

x is the parameter set including the unknown weights w_{rc} used in the Eq. (3).

y_k = actual price on the k^{th} day

$\hat{y}_k$ = predicted price on the k^{th} day

N = number of data samples.

Inspired by the promising output of neural network with several meta-heuristic learning algorithms, in this paper the author has analyzed the utility of a Shuffled Differential Evolution (SDE) based learning algorithm in designing of PSNN. The desire is to integrate the partitioning and random shuffling scheme of Shuffled frog leaping algorithm with evolutionary steps of a Differential Evolution technique to obtain an optimal solution with an accelerated convergence rate.

3.2.1 Differential Evolution. Differential Evolution (DE) is an evolutionary, stochastic, population-based optimization algorithm introduced by Storn and Price in 1996. Figure 2 depicts the flow diagram of DE. In DE an optimal solution is explored from a randomly generated starting population by means of three evolutionary operations such as mutation, crossover and selection. For each generation, the individuals of the current population become target vectors and their fitness function value is calculated using Eq. (4). Then the mutation operation produces a mutant vector mi for each individual target vector xi using the following formula:

$$m_{ij} = x_{r_1 j} + F \times \left(x_{r_2 j} - x_{r_3 j}\right) \quad where\, r_1 \neq r_2 \neq r_3 \neq i \tag{5}$$

where r1, r2, r3 are random and mutually exclusive integers.

The mutant vector along with the target vector is further passed through the crossover operation to produce a trial vector as follows:

$$t_{ij} = m_{ij}\ if\ rand\,[0,1] \leqslant cr\ or\ j = j_{rand} \quad else \quad x_{ij} \tag{6}$$

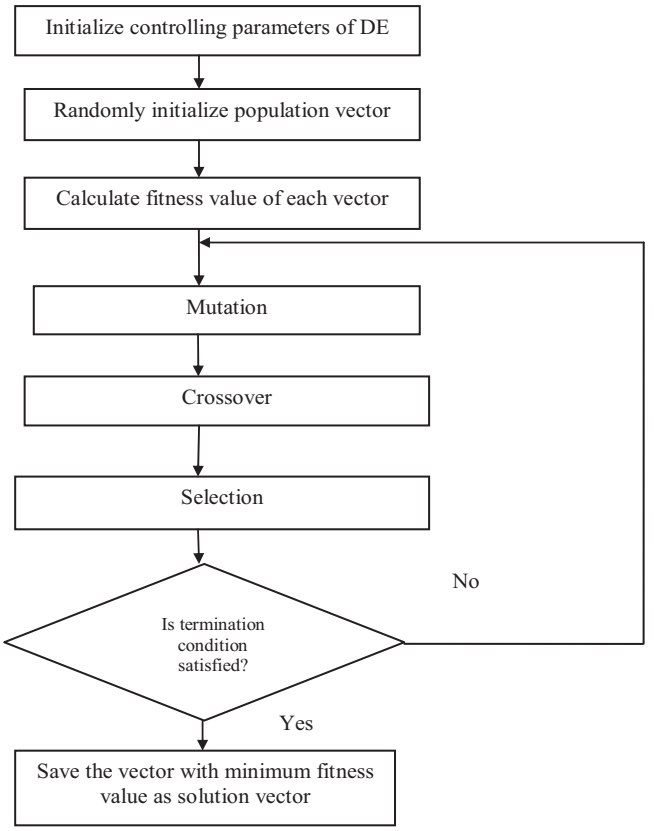

Figure 2.
Detailed steps of Differential Evolution Algorithm.

The trial vector replaces the target vector if its fitness value is better than the target vector. So it can be summarized that mutation enlarges the search space, Crossover recapitulates previously successful individuals and selection encourages the survival of the fittest. The mutation, crossover and selection operations are repeated until some termination condition is reached.

3.2.2 Shuffled Frog Leaping. Shuffled Frog Leaping algorithm (SFLA) is a nature based optimization method that emulates the memetic evolution of a team of frogs, exploring locations with more amount of available food. In SFLA, a team of frogs represents a list of possible solutions to the optimization problem. The team is further divided into a number of parallel communities specified as memeplexes. Then the frog with best position $x_{b,}$ the frog with worst position x_w within each memeplex and the frog having global best position x_g with respect to the entire population of frogs are identified. The worst frog's location in a memeplex is updated based on the location of a local best or a global best frog or randomly to a position so that the frogs can forward towards an optimal position. The new location for the worst frog is obtained by exchanging information in memeplex according to the following equation:

$$\begin{aligned} &x_w(new) = x_w + s_i = x_w + r1 \times (x_b - x_w) \\ &where \quad s_{\min} \leqslant s_i \leqslant s_{\max} \end{aligned} \tag{7}$$

s_i is the frog leaping step size of i^{th} frog, r1 is a random number within a range [1,1]. If the fitness value of x_w(new) is better than the current one, x_w will be replaced by the new one. Otherwise, the new position for the worst frog is obtained by exchanging information between memeplexes according to the following equation:

$$x_w(new) = x_w + s_i = x_w + r2 \times (x_g - x_w) \tag{8}$$

where r2 is a random number within a range [1,1]. If the fitness value of x_w(new) is better than the current one, x_w will be replaced by the new one. Otherwise, if the resulting leap does not produce any improvement for the worst frog, then the new position for the worst frog is generated randomly within a specified range. After a specified number of memetic evolutionary steps, a new population of frogs is produced by combining the memeplexes through a shuffling process that helps to promote a global information interchange between the frogs. Creation of memeplexes, local search within memeplex and shuffling of memeplexes are continued until some termination condition is reached. Figure 3 depicts the flow diagram of SFL.

3.2.3 Shuffled Differential Evolution. SDE is a hybridized meta-heuristic approach developed by the fusion of evolutionary operations of DE with the shuffled complex evolution of SFLA. Though smaller parameter space size and quick convergence are the eminent features of DE, still it may occasionally face the problem of premature convergence and stagnation. Similarly, the insufficient learning mechanism of SFLA may lead to non-comprehensive solution domain exploration. Hence blending of the two popular approaches as a single one is suggested here to surpass the intrinsic limitations of both and emphasizing in the meanwhile on their betterment [17–20]. In SDE, the initial population vector is divided into subpopulations using the sorting and division process as used in SFL. Then the evolutionary steps of DE such as mutation, crossover and selection are applied on each subpopulation to generate a new set of vectors. In each iterative step, the individual of the current subpopulation is treated as a target vector. Through the mutation process, a mutant vector is generated for each target vector. Then by applying crossover operation to mutant and target vector, a new trial vector is generated. A trial vector having a better fitness value replaces the target vector in the next iterative step. After applying the mutation, crossover and selection steps to each subpopulation for pre-specified iterative steps, then using the shuffling technique of SFL the vectors of all subpopulations are merged to generate a new set

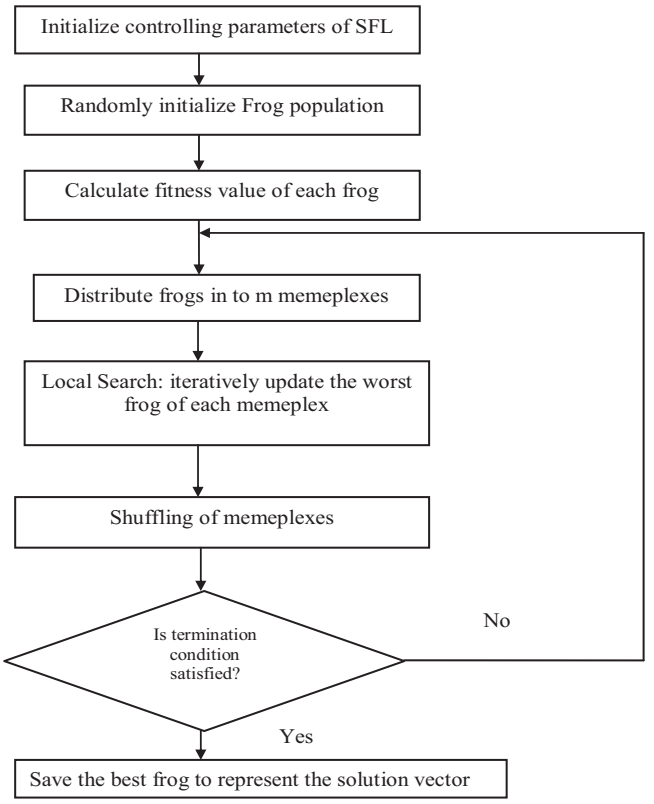

Figure 3.
Detailed steps of Shuffled Frog Leaping Algorithm.

of population. Unlike original SFL, with this new approach, all the vectors (termed as frogs in SFL) take part in the population evolution and the local search process of SFL is enhanced by the evolutionary steps of DE. A suitable proportion between global and local exploration is also attained by adapting the control parameters of SDE within a range.

The detailed steps of Shuffled Differential Evolution based learning algorithm for PSNN are illustrated as follows:

Step 1: Initialization of controlling parameters of SDE:

- The parameters of SDE such as Number of Iterations (NI), population size (NP), subpopulations (s), iterative steps for each subpopulation (si), upper and lower bound of scaling factor (F_U, F_L), upper and lower bound of crossover rate (CR_U, CR_L) are set in the preliminary step.

Step 2: Encoding of PSNN weights to vector position of SDE:

- According to the population size, the position of each individual in the population vector is generated randomly, representing possible weight vectors of PSNN in the following format:

$F_r = [w_0, w_{1,1}, w_{2,1}, \ldots, w_{d,1}, \ldots, w_{1,p}, w_{2,p}, \ldots, w_{d,p}]$

where w_0 is the bias value, $w_{r,c}$ is the weight corresponding to the r^{th} input node joined with c^{th} hidden layer node.

Step 3: Fitness calculation of each vector:

- Measure the objective function value of each individual vector using Eq. (4).

Step 4: Subpopulation formation:

- The vectors are rearranged according to their decreasing order of fitness values.
- Then the sorted vectors are partitioned into s number of subpopulations. Each subset contains n vectors such that NP = s × n. The partition is carried out, such that a vector having maximum fitness value will put in the first subset, accordingly the next vector in the second subset and so on.
- Then the vector with the best position within each subpopulation is represented as t_b.

Step 5: Updation of subpopulations:

- The new location for the worst frog is obtained by exchanging information in subpopulations by applying mutation, crossover and selection operations.
- Mutation: With a scaling factor F, a mutant vector v_i is generated from the target vector t_i as follows:

$$v_{ij} = t_{ij} + F \times (t_{b,j} - t_{ij}) + F \times \left(t_{r_1 j} - t_{r_2 j}\right) \quad where \;\; r_1 \neq r_2 \neq i \tag{9}$$

- Crossover: A trial vector u_i is obtained by conjoining the mutant vector with the target vector through the following crossover equation:

$$u_{ij} = v_{ij} \; if \; rand\,[0,1] \leqslant cr \; or \; j = j_{rand} \quad else \quad t_{ij} \tag{10}$$

Selection: Out of trial and target vector, the one showing less fitness value is selected to the next generation as follows:

$$if \; f(u_i) < f(t_i) \quad t_i = u_i \tag{11}$$

- Repeat the mutation, crossover and selection operations for each vector of each subpopulation according to a specific number of iterative steps (si).

Step 6: Shuffling Process:

- The vectors of all subpopulations are again shuffled and arranged to accomplish the loop of evolution.

Step 7: Adaptation of controlling parameters:

- The parameters F and CR are adapted as follows:

$$F(t) = F_U - (F_U - F_L) \times t/NI \quad CR(t) = CR_U - (CR_U - CR_L) \times t/NI \tag{12}$$

Repeat steps 1 through 7 until the number of iterations NI is attained.

3.3 Detailed steps of PSNN-SDE for currency exchange rate prediction

This section provides details of PSNN-SDE to predict future currency exchange rate direction. The framework of the proposed PSNN-SDE model is shown in Figure 4.

Step 1: Collection of the data set

Initially, real exchange rate prices are collected in due time.

Step 2: Data preprocessing

Through data preprocessing the raw time series data is transformed into an acceptable input-output form for a machine learning technique. The preprocessing steps included in the study are listed below:

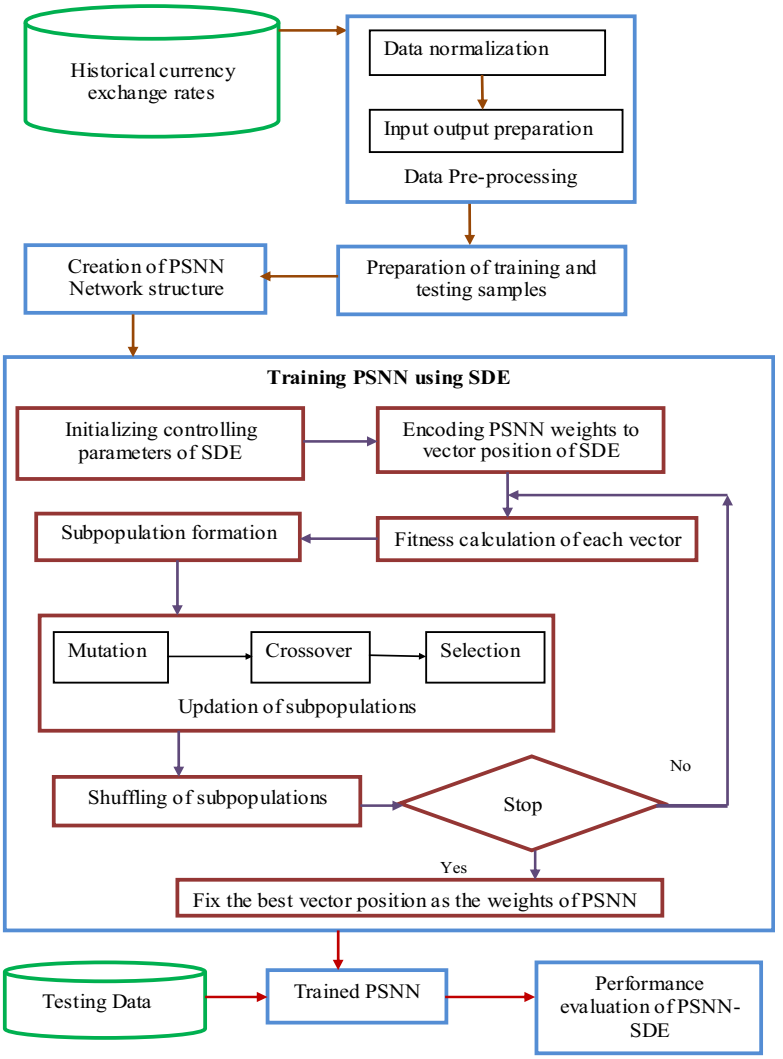

Figure 4.
Detailed architecture of PSNN-SDE.

a) Data normalization

Originally collected exchange rate prices are continuous in nature. So they are scaled in the range to [0, 1] using the min max normalization as follows:

$$nv = \frac{v - v_{\min}}{v_{\max} - v_{\min}} \tag{13}$$

Where

nv = normalized value.

v = value to be normalized

v_{min} = minimum value of the series to be normalized

v_{max} = maximum value of the series to be normalized

b) Input output preparation

Applying a sliding window, with a window size *w* and prediction horizon *hr*, the normalized exchange rate prices related to consecutive w trading days along with their simple moving average are arranged to create the input row vector, whereas the normalized exchange rate prices related to $(w + hr)^{th}$ day is set as its corresponding output pattern. In each step, the sliding window is shifted one position by dropping the data in beginning and adding a new in the end.

Step 4: Preparation of training and testing samples

After arranging the input-output vectors through data preprocessing, two third of these are utilized as in-sample data and the rest are kept as out-sample data. The in-sample data are used for training and validation of the predictor model. Finally, the model performance is tested over out-sample data.

Step 5: Network structure creation of PSNN

PSNN is a simple feedforward network structure with three layers. The number of neurons in the first layer of PSNN is fixed to $(w + 1)$ to represent the *w* normalized exchange rate values corresponding to each trading day and moving average of that. The output layer contains one node to produce the predicted currency exchange rate price. In this study, the number of hidden layer nodes is set to a value one more than half of the sum of the number of input and output layer nodes. Then a weight vector having a size equal to one more than the product of the input layer and hidden layer nodes is initialized randomly within a given range, which is further estimated by a learning algorithm.

Step 6: Training Using SDE

During the training phase, the network is adjusted over in-sample data for identifying the suitable weights of PSNN. The position of each individual vector of the population is randomly initialized within $[-1, 1]$ according to the population size, that specifies the weight of the network. Then the fitness function value of each individual is obtained by applying the weights specified in each individual in the Eq. (3) with a nonlinear tanh ( ) function. By repeatedly applying the basic operations of SDE based learning algorithm such as division of initial population to subpopulations, updating individuals of subpopulations through mutation, crossover and selection and then shuffling of subpopulations, the weight vectors specified in the population are updated iteratively. Finally, the best vector position is fixed as the weights of PSNN and the network is used for testing.

Step 7: Performance Evaluation

In the testing process, the performance of PSNN is accessed by calibrating the divergence between actual and predicted exchange rate prices through three statistical error metrics such as RMSE, Mean Absolute Percentage Error (MAPE), Mean square error (MSE) along with Theil's U statistic and correlation coefficient. The MAPE and MSE are defined as follows:

$$MAPE = \frac{1}{N}\sum_{k=1}^{N}\left|\frac{y_k - \widehat{y}_k}{y_k}\right| \times 100 \tag{14}$$

$$MSE = \frac{1}{N}\sum_{k=1}^{N}(y_k - \widehat{y}_k)^2 \tag{15}$$

where

y_k = actual closing price on k^{th} day

$\widehat{y}_k$ = predicted closing price on k^{th} day

N = number of data samples.

A ratio of the accuracy of the proposed model to that of a naïve forecast model is calculated using Theil's U statistic as follows:

$$Theil's\ U = \frac{\sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(\frac{y_k - \widehat{y_k}}{y_k}\right)^2}}{\sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(\frac{y_k - y_{k+1}}{y_k}\right)^2}} \tag{16}$$

Theil's U statistic is a relative accuracy measure that compares the forecasted results with the results of forecasting of a naïve forecast model [1,11,21]. Here the numerator describes the degree of deviation between the actual prices and the predicted prices corresponding to the proposed model, whereas the denominator describes the degree of fluctuation corresponding to a Naïve forecast model. Naïve forecast model is a one step ahead trivial predictor model in which the predicted value is assumed to be the same as the last observation. The Theil's U value equal to 1 implies the equal performance of the model with that of the naïve forecast model, which in turn suggests that it is impossible to predict the prices due to its random nature. A value greater than 1 indicates the worse performance of the model compared to a naïve forecast model, whereas a value less than 1 expresses better attainment of the model compared to a naïve forecast model. A better predictor model always suggests Theil's U value closer towards 0.

4. Experimental result analysis

For validating the predictability of the proposed model, the daily exchange rate prices of US Dollar (USD) in opposition to Swiss Franc (CHF) and Japanese Yen (JPY) are gathered in due time of 1/1/2014 to 28/10/2016 and put forward for data preprocessing. The number of samples collected for each set is 738. Initial simulations are carried out for predicting the exchange rate prices for next day ahead with a window size of 5. Further, it is extended for other prediction horizons. The generated 733 input-output patterns through the sliding window process are further partitioned into in-sample and out-sample for training, validation and testing of the model. The number of patterns belongs to the in-sample set is 586, that is

generated from daily exchange rates gathered during period 1/1/2014 to 6/4/2016 and number patterns in the out-sample set is 147 generated from exchange rates gathered during period 7/4/2016 to 28/10/2016. A 5 fold cross-validation is used for measuring the generalization ability of the network. The in-sample data set is divided into 5 groups. Out of these 5 groups, 4 groups are used for training and the other is used for validation. The entire process is carried out for 10 independent runs. After input-output preparation, the structure of PSNN is created with 6 neurons in the first layer, 4 summation units in the middle layer and 1 product unit in the output layer. Then a weight vector having a row size of 25 is initialized randomly, which is further estimated through the training process of the network. A comparative assessment of the PSNN is observed with respect to different meta-heuristic based training process such as the proposed SDE, DE, SFL and PSO. In all the learning algorithms, minimization of RMSE is set as the cost function. The values of control parameters of different meta-heuristic learning algorithms are set through simulations for both the dataset. As the performance of any evolutionary algorithm depends on its selected controlling parameters and it is completely application oriented, so in this study, these values are set according to the suggested values during simulations. The set of parameter settings of different meta-heuristic algorithms considered for the experiment are listed in Table 1. Prediction ability of a model can be judged as a good one through its lower forecasting errors obtained over out-sample data. Tables 2 and 3 report the prediction output of PSNN with different learning algorithms.

The observations corresponding to USD/CHF dataset clearly depicts the least RMSE, MAPE, Theil's U, MSE values and a higher correlation coefficient of PSNN-SDE model compared to other models. Whereas the RMSE, Theil'sU, MSE and correlation coefficient values corresponding to PSNN-SDE model and MAPE value corresponding to PSNN-DE model seem better for USD/JPY dataset. In both the data sets, the Theil's U value less than

SDE	DE	SFL	PSO
Population size: NP = 30 **Memeplex size:** m = 5 **Iterative steps for memeplex:** im = 10 F = iteratively decreases between 0.5 and 0.2 Cr = iteratively decreases between 0.9 and 0.5 **Number of iterations:** NI = 100	**Population size:** NP = 30 **Crossover rate:** Cr = 0.9 **Mutation scale:** F = iteratively decreases between 0.5 and 0.2 **Number of iterations:** NI = 100	**Population size:** NP = 30 **Memeplex size:** m = 5 **Iterative steps for memeplex:** im = 10 **Number of iterations:** NI = 100	**Population size:** NP = 30 **Cognitive and social acceleration constants:** C1 = 1.9 C2 = 1.9 **Inertia Weight:** ϕ = iteratively decreases between 0.8 and 0.4 **Number of iterations:** NI = 100

Table 1.
Control parameters of different evolutionary algorithms.

Models	RMSE	MAPE	Theil's U	MSE	Correlation Coefficient
PSNN-SDE	**0.0247**	**0.3738**	**0.1191**	**0.0006**	**0.9006**
PSNN-DE	0.0425	0.6407	0.2132	0.0018	0.7016
PSNN-SFL	0.0394	0.6142	0.1908	0.0016	0.8008
PSNN-PSO	0.0560	0.8849	0.2730	0.0031	0.5993

Bold values in the table represents better results obtained by the model in comparision to other models.

Table 2.
Prediction output of PSNN with different learning algorithms over out-sample data of USD/CHF data set.

1 clearly indicates the better performance of the predictor models in comparison to the naïve forecast model. The better convergence of PSNN-SDE compared to PSNN-DE, PSNN-SFL and PSNN-PSO is also shown for two data sets in Figures 5 and 6. Figures 7 and 8 outline the one day ahead predicted exchange rate prices obtained by the proposed PSNN-SDE model with the actual prices, both for USD/CHF and USD/JPY data set respectively.

The statistical significance of the proposed model is supported by a Kolmogorov-Smirnov Predictive Accuracy (KSPA) test [42]. Table 4 gives the p and h values of the KSPA test result performed over the absolute value of forecast errors with a 5% significance level. An h value 1 and p value less than 0.05 obtained from KSPA test statistically indicates that the outcome of compared predictor models is different. Most of the cases of KSPA test corresponding to Table 4 provide sufficient indication to conclude that the proposed PSNN-SDE model reports a better forecasting accuracy compared to PSNN-DE, PSNN-SFL and PSNN-PSO models from a statistical perspective.

Finally, analyzing the RMSE error specified in Table 5 with different prediction horizons such as 2, 3, 5, 7, 10 and 15, it is also clear that, the PSNN-SDE model provides less error statistics with other prediction horizons compared to PSNN-DE, PSNN-SFL, and PSNN-PSO models.

Models	RMSE	MAPE	Theil's U	MSE	Correlation Coefficient
PSNN-SDE	**0.0321**	0.6745	**0.2032**	**0.0009**	**0.9602**
PSNN-DE	0.0338	**0.6670**	0.2068	0.0011	0.9560
PSNN-SFL	0.0370	0.7740	0.2204	0.0014	0.9496
PSNN-PSO	0.0361	0.7479	0.2156	0.0013	0.9550

Bold values in the table represents better results obtained by the model in comparision to other models.

Table 3.
Prediction output of PSNN with different learning algorithms over out-sample data of USD/JPY data set.

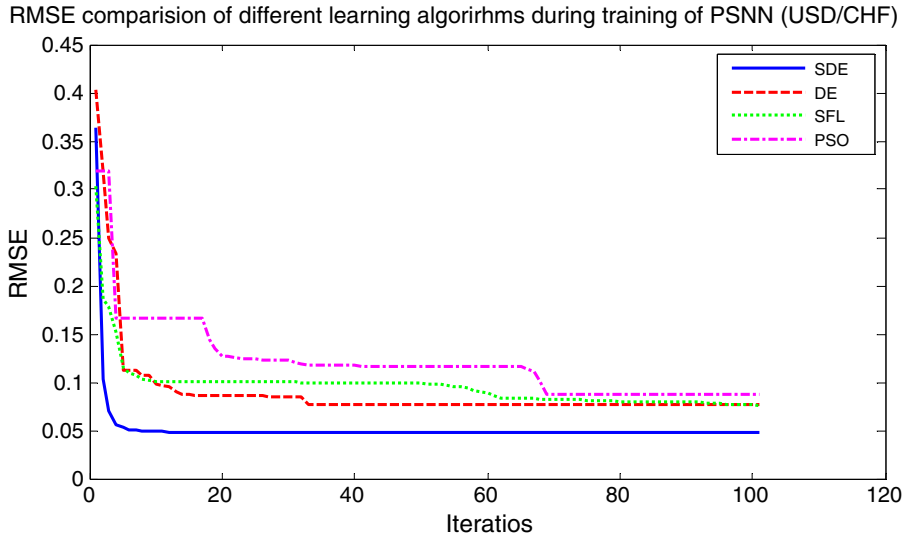

Figure 5.
RMSE comparison of PSNN with SDE, DE, SFL, and PSO based training algorithm for USD/CHF data set.

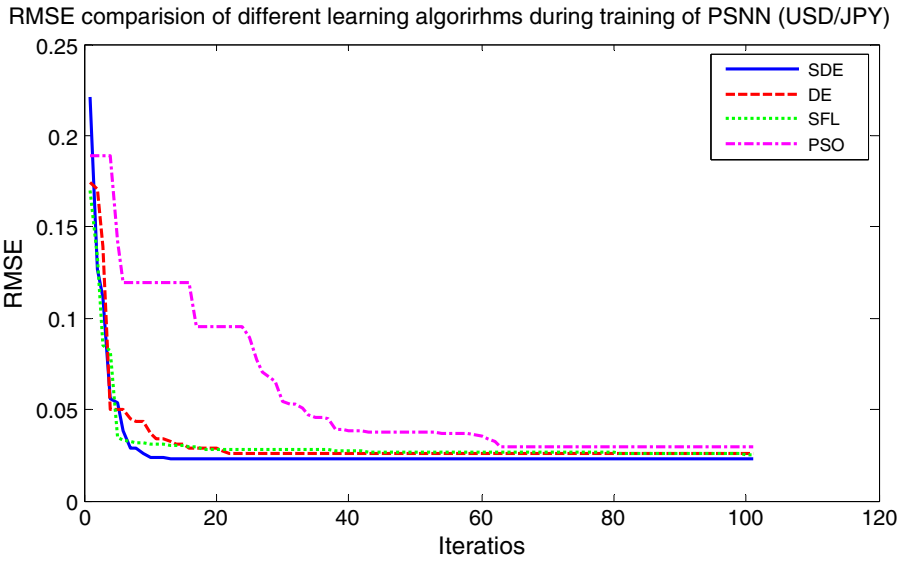

Figure 6.
RMSE comparison of PSNN with SDE, DE, SFL, and PSO based training algorithm for USD/JPY data set.

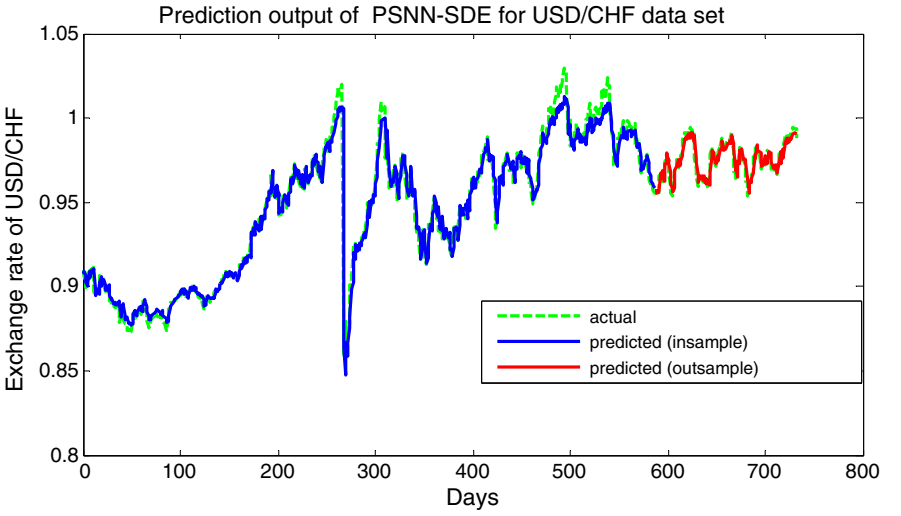

Figure 7.
Output of PSNN-SDE for USD/CHF data set with prediction horizon 1.

5. Conclusion

In this study, the authors have explored the effectiveness of a hybrid learning technique termed as Shuffled Differential Evolution (SDE) in estimating the unrevealed parameters of a Pi-Sigma Network in application to prediction of future currency exchange rates based on past observations. Integration of the evolutionary steps of DE with the partitioning and random shuffling scheme of SFL in the hybrid SDE technique helps in improving the overall performance of PSNN network. A simulation for the comprehensive evaluation of the

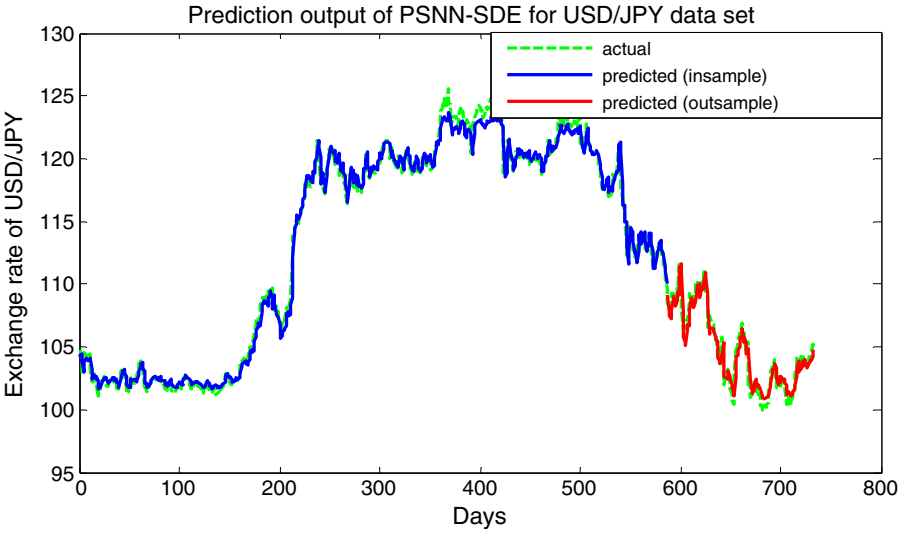

Figure 8.
Output of PSNN-SDE for USD/JPY data set with prediction horizon 1.

Data set	PSNN-SDE/PSNN-DE	PSNN-SDE/PSNN-SFL	PSNN-SDE/PSNN-PSO
USD/CHF	0.0001 (1)	0.0000 (1)	0.0002 (1)
USD/JPY	0.0369 (1)	0.0294 (1)	0.0297 (1)

Table 4.
P and h values of Kolmogorov-Smirnov test.

Data set	Model	2	3	5	7	10	15
USD/CHF	PSNN-SDE	**0.0312**	**0.0485**	**0.0594**	**0.0698**	**0.0896**	**0.1189**
	PSNN-DE	0.0426	0.0566	0.0625	0.0716	0.1131	0.1194
	PSNN-SFL	0.0392	0.0604	0.0638	0.0937	0.0971	0.1249
	PSNN-PSO	0.0388	0.0504	0.0789	0.0867	0.0983	0.1206
USD/JPY	PSNN-SDE	**0.0545**	**0.0657**	**0.0748**	**0.0826**	**0.1085**	**0.1211**
	PSNN-DE	0.0582	0.0669	0.0753	0.0843	0.1097	0.1293
	PSNN-SFL	0.0561	0.0663	0.0767	0.0873	0.1102	0.1396
	PSNN-PSO	0.0564	0.0671	0.0828	0.0977	0.1108	0.1375

Bold values in the table represents better results obtained by the model in comparision to other models.

Table 5.
Prediction output with different prediction horizons.

PSNN-SDE model is presented over two currencies such as USD/CHF and USD/JPY. The evaluation includes several performance measures such as RMSE, MAPE, Theil's U statistic, and KSPA test. The model performance is also observed with different prediction horizons. From the decisive observations of prediction errors and result of KSPA test, it is clear that the PSNN-SDE model produces enhanced forecasting accuracy than other predictor models included in the examination. Future research will involve exploring SDE for both structure and parameter optimization of PSNN. Determining the best choice of input features of the network model to improve prediction accuracy using evolutionary techniques will also be suggested.

Abbreviations:

ANN	Artificial Neural Network
ARMA	Auto Regressive Moving Average
ARIMA	Auto Regressive Integrated Moving Average
CGV	Conditional Generalized Variance
DHS	Differential Harmony Search
FLANN	Functional Link Artificial Neural Network
GRNN	Generalized Regression Network
KSPA	Kolmogorov-Smirnov Predictive Accuracy
MSR	Markov Switching Regression
MAPE	Mean Absolute Percentage Error
MLP	Multi-Layer Perception Network
NI	Number of Iterations
PSNN	Pi-Sigma Neural Network
PSO	Particle Swarm Optimization
RBF	Radial Basis Function
RNN	Recurrent Neural Network
RSOM	Recurrent Self-Organizing Map
RMSE	Root Mean Squared Error
SDE	Shuffled Differential Evolution
SFLA	Shuffled Frog Leaping Algorithm

References

[1] T.H. Hann, E. Steurer, Much ado about nothing? exchange rate forecasting: neural networks vs. linear models using monthly and weekly data, Neurocomputing 10 (4) (1996) 323–339.

[2] J. Kamruzzaman, R. Sarker, Comparing ANN based models with ARIMA for prediction of forex rates, Asor. Bull. 22 (2) (2003) 2–11.

[3] M. Nilashi, O. Ibrahim, N. Janahmadi, L. Ebrahimi, Comparative study of artificial neural network and ARIMA models in predicting exchange rate, Res. J. Appl. Sci., Eng. Technol. 4 (21) (2012) 4397–4403.

[4] Z. Ye, X. Ren, Y. Shan, Comparing RMB exchange rate forecasting accuracy based on dynamic BP neural network model and the ARMA Model, J. Stock Forex Tradi. 5 (1) (2015) http://dx.doi.org/10.4172/2168-9458.

[5] M. Pedram, M. Ebrahimi, Exchange rate model approximation, forecast and sensitivity analysis by neural networks, case of Iran, Bus. Econ. Res. 4 (2) (2014) 49–62.

[6] A.A. Adebiyi, A.O. Adewumi, C.K. Ayo, Comparison of ARIMA and artificial neural networks models for stock price prediction, J. Appl. Math. 1–7 (2014).

[7] L. Yu, K.K. Lai, S. Wang, Multistage RBF neural network ensemble learning for exchange rates forecasting, Neurocomputing 71 (16) (2008) 3295–3302.

[8] R. Majhi, G. Panda, G. Sahoo, Efficient prediction of exchange rates with low complexity artificial neural network models, Expert Syst. Appl. 36 (1) (2009) 181–189.

[9] P.R. Jena, R. Majhi, B. Majhi, Development and performance evaluation of a novel knowledge guided artificial neural network (KGANN) model for exchange rate prediction, J. King Saud Univ.-Comp. Inf. Sci. 27 (4) (2015) 450–457.

[10] S. Galeshchuk, Neural networks performance in exchange rate prediction, Neurocomputing 172 (2016) 446–452.

[11] R. Dash, Performance analysis of a higher order neural network with an improved shuffled frog leaping algorithm for currency exchange rate prediction, Appl. Soft Comput. 67 (2018) 215–231.

[12] R. Dash, P.K. Dash, R. Bisoi, A self adaptive differential harmony search based optimized extreme learning machine for financial time series prediction, Swarm Evol. Comput. 19 (2014) 25–42.

[13] X. Cheng, X. Zhang, L. Zhao, A. Deng, Y. Bao, Y. Liu, Y. Jiang, The application of shuffled frog leaping algorithm to wavelet neural networks for acoustic emission source location, C. R. Méc. 342 (4) (2014) 229–233.

[14] T. Yang, A.A. Asanjan, M. Faridzad, N. Hayatbini, X. Gao, S. Sorooshian, An enhanced artificial neural network with a shuffled complex evolutionary global optimization with principal component analysis, Inf. Sci. 418 (2017) 302–316.

[15] Q. Duan, S. Sorooshian, V. Gupta, Effective and efficient global optimization for conceptual rainfall-runoff models, Water Resour. Res. 28 (4) (1992) 1015–1031.

[16] Q.Y. Duan, V.K. Gupta, S. Sorooshian, Shuffled complex evolution approach for effective and efficient global minimization, J. Optim. Theory Appl. 76 (3) (1993) 501–521.

[17] V.C. Mariani, L.G.J. Luvizotto, F.A. Guerra, L. Santos Coelho, A hybrid shuffled complex evolution approach based on differential evolution for unconstrained optimization, Appl. Math. Comput. 217 (12) (2011) 5822–5829.

[18] A.S. Reddy, K. Vaisakh, Shuffled differential evolution for economic dispatch with valve point loading effects, Int. J. Electr. Power Energy Syst. 46 (2013) 342–352.

[19] A.S. Reddy, K. Vaisakh, Shuffled differential evolution for large scale economic dispatch, Electr. Power Syst. Res. 96 (2013) 237–245.

[20] M.R. Naeini, T. Yang, M. Sadegh, A. AghaKouchak, K.L. Hsu, S. Sorooshian, Q. Duan, X. Lei, Shuffled Complex-Self Adaptive Hybrid EvoLution (SC-SAHEL) optimization framework, Environ. Modell. Software 104 (2018) 215–235.

[21] A.S. Chen, M.T. Leung, Regression neural network for error correction in foreign exchange forecasting and trading, Comput. Oper. Res. 31 (7) (2004) 1049–1068.

[22] H. Ni, H. Yin, Exchange rate prediction using hybrid neural networks and trading indicators, Neurocomputing 72 (13) (2009) 2815–2823.

[23] B. Premanode, C. Toumazou, Improving prediction of exchange rates using differential EMD, Expert Syst. Appl. 40 (1) (2013) 377–384.

[24] M. Rehman, G.M. Khan, S.A. Mahmud, Foreign currency exchange rates prediction using CGP and recurrent neural network, IERI Procedia 10 (2014) 239–244.

[25] W. Shen, X. Guo, C. Wu, D. Wu, Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm, Knowl.-Based Syst. 24 (3) (2011) 378–385.

[26] S. Panda, A. Sarangi, S.P. Panigrahi, A new training strategy for neural network using shuffled frog-leaping algorithm and application to channel equalization, AEU-Int. J. Electr. Commun. 68 (11) (2014) 1031–1036.

[27] R. Dash, An improved shuffled frog leaping algorithm based evolutionary framework for currency exchange rate prediction, Physica A 486 (2017) 782–796.

[28] R. Dash, P.K. Dash, Prediction of financial time series data using hybrid evolutionary Legendre neural network: evolutionary LENN, Int. J. Appl. Evol. Comput. 7(1) a) (2016) 16–32.

[29] P. Mohapatra, A. Raj, T.K. Patra, Indian stock market prediction using differential evolutionary neural network model, Int. J. Electr. Commun. Comp. Technol. 2 (4) (2012) 159–166.

[30] R. Dash, P.K. Dash, Efficient stock price prediction using a self evolving recurrent neuro-fuzzy inference system optimized through a modified differential harmony search technique, Expert Syst. Appl. 52 (2016) 75–90.

[31] M.M. Eusuff, K.E. Lansey, Optimization of water distribution network design using the shuffled frog leaping algorithm, J. Water Resour. Plann. Manage. 129 (3) (2003) 210–225.

[32] M.M. Eusuff, K.E. Lansey, F. Pasha, Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization, Eng. Optim. 38 (2) (2006) 129–154.

[33] M. Jadidoleslam, E. Bijami, N. Amiri, A. Ebrahimi, J. Askari, Application of shuffled frog leaping algorithm to long term generation expansion planning, Int. J. Comp. Electr. Eng. 4 (2) (2012) 115.

[34] J. Luo, M.R. Chen, Improved shuffled frog leaping algorithm and its multi-phase model for multi-depot vehicle routing problem, Expert Syst. Appl. 41 (5) (2014) 2535–2545.

[35] J. Luo, X. Li, M.R. Chen, H. Liu, A novel hybrid shuffled frog leaping algorithm for vehicle routing problem with time windows, Inf. Sci. 316 (2015) 266–292.

[36] D. Lei, X. Guo, A shuffled frog-leaping algorithm for hybrid flow shop scheduling with two agents, Expert Syst. Appl. 42 (23) (2015) 9333–9339.

[37] W.L. Xiang, N. Zhu, S.F. Ma, X.L. Meng, M.Q. An, A dynamic shuffled differential evolution algorithm for data clustering, Neurocomputing 158 (2015) 144–154.

[38] A.J. Hussain, P. Liatsis, Recurrent pi-sigma networks for DPCM image coding, Neurocomputing 55 (1) (2003) 363–382.

[39] A.J. Hussain, A. Knowles, P.J. Lisboa, W. El-Deredy, Financial time series prediction using polynomial pipelined neural networks, Expert Syst. Appl. 35 (3) (2008) 1186–1199.

[40] J. Nayak, B. Naik, H.S. Behera, A novel chemical reaction optimization based higher order neural network (CRO-HONN) for nonlinear classification, Ain Shams Eng. J. 6 (3) (2015) 1069–1091.

[41] J. Nayak, B. Naik, H.S. Behera, A novel nature inspired firefly algorithm with higher order neural network: performance analysis, Eng. Sci. Technol. Int. J. 19 (1) (2016) 197–211.

[42] H. Hassani, E.S. Silva, A Kolmogorov-Smirnov based test for comparing the predictive accuracy of two sets of forecasts, Econometrics 3 (3) (2015) 590–609.

Corresponding author
Rajashree Dash can be contacted at: rajashreedash@soa.ac.in