Operations Research Forum (2023) 4:10
https://doi.org/10.1007/s43069-023-00196-6

BRIEF REPORT

Optimizing Structure and Internal Unit Weights of Echo State Network for an Efficient LMS-Based Online Training

Javad Saadat[1] · **Mohsen Farshad**[1] · **Hussein Eliasi**[1]

Received: 17 July 2022 / Accepted: 26 November 2022 / Published online: 15 February 2023

Abstract

Echo state network (ESN) is a special type of recurrent neural networks (RNN) wherein a dynamic reservoir is used in the hidden layer, the weight of internal units of ESN is kept fix during training process, and output weights are the only trainable weights. Therefore, network training in an offline mode can be changed into a linear regression equation which is simply solved, although it is required to use online training of ESN in some applied problems. The least mean square (LMS) algorithm can provide an easy and constant method for online training of ESN; however, the huge eigenvalue spreads of the correlation matrix of internal network states reduce the speed of the algorithm convergence. In this study, harmony search algorithm (HSA) is used to optimally produce the structure and weight of internal network units. It is possible to significantly reduce the eigenvalue spreads of the correlation matrix of network states by means of this algorithm. Thereafter, the LMS algorithm is used for the online training of ESN built with the help of HSA. Already-obtained simulation results show that the eigenvalue spreads of the correlation matrix are reduced millions of times, and the LMS algorithm increases the online training speed of the network several times with an acceptable precision of training.

Keywords Echo state network · Least mean square algorithm · Harmony search algorithm · Prediction and chaotic time series

This article is part of the Topical Collection on *Advances and Applications in Supply Chain Management*

✉ Mohsen Farshad
mfarshad@birjand.ac.ir

Javad Saadat
saadatjavad90@birjand.ac.ir

Hussein Eliasi
h_eliasi@birjand.ac.ir

[1] Department of Electrical and Computer Engineering, University of Birjand, Birjand, Iran

1 Introduction

An echo state network (ESN) is a new type of recurrent neural network (RNN) that uses a large dynamic reservoir in the hidden layer. The weights of the reservoir connections are randomly generated and remain fixed during the training process. Therefore, it is possible to use more units in the reservoir to produce more dynamics. This feature of ESN has attracted the attention of many researchers. The important research done so far on ESN includes detailed scrutiny of intrinsic plasticity in ESN [1, 2], an augmented ESN for nonlinear adaptive filtering of complex noncircular signals [3], using leaky integrator neurons in ESN [4, 5], and echo state Gaussian process [6]. Some other scientists studied various ESN schemes [7–12] and several researchers attended to echo state property (ESP) [13–16].

The simple use as well as the ability to model dynamic systems make ESN an appropriate alternative for modeling unknown dynamic systems. ESN has been successfully used in solving many problems such as chaotic time-series prediction [17–19], speech recognition [20], grammatical structure learning [21], communication channel equalization [22], dynamical pattern recognition [20, 23], power system monitoring [24–27], and energy consumption forecasting [28, 29]. However, in some practical problems where the parameters change during operation or the network needs output feedback, it is necessary to train the network online [30].

The least mean square (LMS) is one of the simple algorithms for online training of RNN. However, the convergence of this algorithm is very slow or may not converge due to the eigenvalues spread of the autocorrelation matrix of network states [22]. To tackle this issue, several structures were used to construct ESN [31, 32], but none of these structures have a significant effect on the eigenvalues spread of the autocorrelation matrix. Therefore, other algorithms such as recursive least squares (RLS) [33], particle swarm optimization (PSO) [34, 35], and harmony search algorithm (HSA) [36] have been used for the purpose of ESN online training, although these algorithms require complex calculations and much time is spent for online training. It has been tried to provide the possibility of using the LMS algorithm for online training of the network by building an appropriate reservoir [22].

A mixed method for the optimization of the reservoir and then online training of ESN will be introduced in this study. In this approach, HSA will be used to optimize an ESN so that the eigenvalues spread of the autocorrelation matrix is reduced, then the LMS algorithm will be used to train the optimized ESN online. In the following, the obtained results will be compared with PSO, HS, and RLS algorithms.

ESN, its conventional structure as well as its offline training will be explained in the section "Echo State Network". LMS and HSA will be briefly introduced in sections "Least Mean Square (LMS) Algorithm" and "Harmony Search Algorithm (HSA)," respectively. In the section "Main Results", a mixed method for the construction and online training of ESN will be proposed. With the help of some examples, the efficiency of the proposed method will be illustrated in the section "Simulation". Finally, the results obtained from simulations will be referred to, and some applied suggestions will be made for the construction and online training of ESN.

2 Echo State Network

ESN is a discrete-time network, which contains three layers, namely input, hidden, and output. These layers consist of K, N, and L units, respectively. The size of input units at time n is represented by $u(n) = \left(u_1(n), u_2(n)..., u_K(n)\right)$, the size of reservoir states is represented by $x(n) = (x_1(n), x_2(n), \dots, x_N(n))$, and the size of output units is represented by $y(n) = (y_1(n), y_2(n), \dots, y_L(n)$. The weight of the connections entering into the reservoir in an N×K matrix is $W^{in} = (w^{in}_{ij})$, the weight of the internal connections of the reservoir in an N×N matrix is $W = (w_{ij})$, the weight of connections returning from output to the reservoir in an N×L matrix is $W^{back} = (w^{back}_{ij})$, and the weight of connections entering into the output in an L×(K+N+L) matrix is $W^{out} = (w^{out}_{ij})$. These weights are all gathered together. In ESN, the size of units and weight of network connections are real values and input can directly enter into the output [37–39]. Figure 1 shows the structure of the echo state neural network.

Reservoir states in ESN are calculated using Eq. (1):

$$x(n+1) = f\begin{pmatrix} W^{in}.u(n+1) + W.x(n) \\ +W^{back}.y(n) \end{pmatrix} \tag{1}$$

In the above equation, $\mathrm{f} = \left(\mathrm{f}_1, \mathrm{f}_2, \dots, \mathrm{f}_\mathrm{N}\right)$ is the performance function of reservoir units and is usually selected as a sigmoidal function (or tanh), and network outputs are obtained through the following relation:

$$\mathrm{y}(\mathrm{n}+1) = \mathrm{f}^{\mathrm{out}}\left(\mathrm{W}^{\mathrm{out}}.\emptyset(\mathrm{n}+1)\right) \tag{2}$$

In the above equation, $\mathrm{f}^{\mathrm{out}} = (\mathrm{f}^{\mathrm{out}}_1, \mathrm{f}^{\mathrm{out}}_2, \dots, \mathrm{f}^{\mathrm{out}}_\mathrm{L})$ represents the performance function of output units and is selected as a linear or sigmoidal function and $\emptyset(\mathrm{n}+1) = \left[\mathrm{u}(\mathrm{n}+1)\ \ \mathrm{x}(\mathrm{n}+1)\ \ \mathrm{y}(\mathrm{n})\right]$ is obtained from the combination of input, reservoir, and output vectors of the previous moment [36].

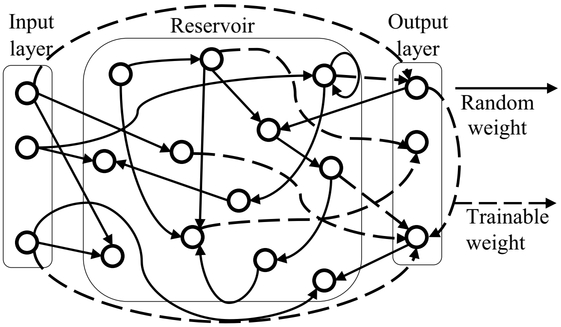


Fig. 1 Block figure of echo state network

2.1 Echo State Property

A main condition for the correct performance of ESN (as the name is clear) is that reservoir states be a function (echo) of the history of the inputs operated on the network. This condition is called echo state property. Although the investigation of necessary and sufficient conditions to make certain about the satisfaction of echo state property is a difficult task, the network can satisfy echo state property on each input and all states of $x \in [-1, 1]$ only if the spectral radius of the reservoir (the highest eigenvalue of matrix W is called reservoir spectral radius) is smaller than one [12, 36].

2.2 ESN Construction

Production of a suitable dynamic reservoir is necessary to build an echo state neural network. Although there is no precise method for determining network units, the number of network units could be selected in a range from one-tenth to half of the training data size. In addition to the size of the reservoir, the level of scattering in connections also affects the precision and convergence speed of the ESN. In many practical applications, the level of scattering could be considered to be 10% [9]. Once the number of reservoir units and the scatter level are set, it is time to propose the reservoir weight matrix so that the network maintains its echo state characteristic. The following steps provide a systematic method to construct an echo state neural network [40]:

1. Produce W^{in}, W^{back}, and W0 matrices with optional spread and at random.
2. Divide the W0 matrix by the largest spectral radius of itself to get matrix W1 with the spectral radius of one, $W_1 = \frac{W_0}{|\lambda_{max}|}$.
3. Multiply an appropriate value of α within the range of $0 < a < 1$ to get the matrix W with an optional value of the spectral radius, i.e., $W = \alpha.W_1$ [36, 38].

2.3 Offline ESN Training

The purpose of neural network training is to determine the weight of network connections so that a minimum deviation from the desired output could be achieved. There are two general methods for the training of a neural network: supervised and unsupervised learning. Generally, for the training of echo state neural networks, the supervised learning method is used in both offline and online modes. In this study, a step-by-step method is proposed for offline training. Although, an online training method is also suggested in the following sections:

First step: produce W^{in}, W^{back}, and W matrices randomly.

Second step: sample network dynamics with the operation of training data on the network, this step includes the following:

1. First determine the initial states of the reservoir arbitrarily.
2. Operate the network by training data and bringing it up-to-date through (1).
3. Put d (0) = 0 at the beginning time of $n = 0$ where d(0) has not been defined.
4. Put the network states in a row matrix of M states.
5. Put the reverse of the performance function of output units (f^{-1}(d(n)) in matrix T, similar to the previous stage.

   To eliminate the effect of the initial states of the reservoir, we should omit the first sample from M and T matrices and store x(n) and (f^{-1}(d(n)) vectors in M and T matrices simultaneously.

Third step: calculate the weight matrix of output connections W^{out}, multiply the pseudo-inverse matrix of M by matrix T, and get transpose of them to obtain W^{out}.

$$W^{out} = \left(M^{-1} \ . \ T\right)^{T} \tag{3}$$

Fourth step: now, the obtained echo state neural network (W^{out}, W, W^{in}, and W^{back}) is ready to be used for solving new problems [2, 35, 36].

3 Least Mean Square (LMS) Algorithm

LMS is an adaptive algorithm, which uses an error gradient vector for the estimation of weight vectors. This algorithm has been made up of a repetitive structure for the movement of weight vectors towards the symmetry of the error gradient vector and finally minimizes the mean squared error. Compared to the RLS algorithm, the calculations of this algorithm are very simple. This algorithm does not need the correlation matrix and matrix inversion [41, 42]. Assume to calculate the network output via Eq. (4).

For the symmetric gradient vector error

$$\mathrm{y(n)} = \mathrm{W^{out}}\emptyset\mathrm{(n)} \tag{4}$$

In the above relation, the output function has been assumed a linear one, and the current matrix of $\emptyset(\mathrm{n}) = \left[\,\mathrm{u(n+1)} \ \ \mathrm{x(n+1)} \ \ \mathrm{y(n)}\,\right]$ has been considered as the algorithm input. Then, the weight matrix of W^{out} is estimated with the help of this algorithm and based on reducing the maximum gradient in order to minimize the mean squared error. The equation of weight estimation is expressed as the following simplified Eq. (5).

$$\mathrm{W^{out}(n+1)} = \mathrm{W^{out}(n)} + \mu.\,\emptyset\mathrm{(n)}.\,\mathrm{e(n)} \tag{5}$$

In the above relation, e(n) represents the error of real network output from the desired degree, and μ is the step size of the algorithm and determines its convergence. We need to select $0 < \mu < \frac{1}{\lambda_{\max}}$ to make certain about the convergence algorithm. If a large value is selected for the step size of the algorithm, its convergence will be endangered and the small step size will reduce the speed of algorithm convergence. In this study, $\mu = \frac{1}{2.\lambda_{max}}$ is considered to keep a balance between speed and algorithm precision. The convergence speed of the algorithm depends on the eigenvalue spreads of the correlation matrix corresponding to reservoir states which will be minimized using the harmony search algorithm in the next section.

The goal of the harmony search algorithm is to minimize the S gain. In these cases, the largest spectral radius of the autocorrelation matrix of the network states is minimized, and therefore, a larger μ can be selected for the LMS algorithm. Also, in this method, eigenvalues close to zero of the autocorrelation matrix of the network states are removed. As a result, the training speed and convergence of this algorithm improved together.

4 Harmony Search Algorithm (HSA)

HSA has been designed based on composers' performance for making better music. As composers search different nets to achieve beautiful music, engineers search appropriate weight vectors for the optimization of objective functions [43, 44].

Harmony memory (HM), harmony memory size (HMS), harmony memory considering rate (HMCR), and pitch adjusting rate (PAR) constitute the parameters of this algorithm for optimization. To use the algorithm, we should store the number of HMSs of the feasible vector in HM and then produce a new harmony by means of PAR and HMCR. If the new harmony outweighs the worst harmony in harmony memory, it can replace it. This process should be repeated until an acceptable vector for the optimization of the objective function is achieved [45, 46].

The harmony search algorithm is operationalized using the following steps:

Step 1: First select the algorithm parameters and optimization problem. Then determine PAR, HMCR, HMS, a suitable objective function, and a small neighborhood radius of bw, and an allowed range of D for the vector values in this step.
Step 2: Then store as much as the capacity of harmony memory of the produced vector along with the values of the objective function for each vector.
Step 3: Based on Eq. (6), a new vector (harmony) is produced; each component of the new vector is selected with the probability of HMCR out of the components of existing vectors in harmony memory and with the probability of 1-HMCR out of the allowed range.

$$\begin{array}{ll} \mathrm{x'}_{\mathrm{i}}^{\mathrm{j}} \in \left\{\mathrm{x}_{\mathrm{i}}^{1}, \mathrm{x}_{\mathrm{i}}^{2}, \ldots, \mathrm{x}_{\mathrm{i}}^{\mathrm{HMS}}\right\} & \mathrm{HMCR} \\ \mathrm{x'}_{\mathrm{i}}^{\mathrm{j}} \in \mathrm{D} & 1-\mathrm{HMCR} \end{array} \tag{6}$$

Then the selected components from harmony memory will be regulated with the probability of PAR, and the components selected from the allowed range will remain unchanged. Equation (7) shows how vector components have been regulated.

$$\mathrm{x'}_{\mathrm{i}}^{\mathrm{j}} = \mathrm{x'}_{\mathrm{i}}^{\mathrm{j}} + \mathrm{bw} * \mathrm{rand} \quad \mathrm{PAR} \tag{7}$$

Step 4: In this case, the new objective function of the vector (harmony) outweighs the worst existing vector in harmony memory; the new vector and value of the related objective function are replaced in harmony memory.
Step 5: Steps 3 and 4 will be repeated until an acceptable answer is reached [12].

Due to the efficiency of HSA in problems with extensive search space, we have used this algorithm for the construction of a proper dynamic reservoir with a small s rate. Then we train the weight of network output connections by means of the LMS algorithm in an online state.

5 Main Results

In this paper, a mixed method for ESN training and construction is proposed, which consists of two substantial steps. The former is the optimization of the structure and weight of internal connections while the latter is online training of the weight of output connections. In the first step, HSA for the optimization of the structure and weight of internal connections of ESN is used to provide the conditions for the convergence of online training of ESN by means of the LMS algorithm. In this way, we should have the lowest possible value for eigenvalue spreads of the state correlation matrix. This step is aimed at optimizing the structure and weight of internal connections of the network; therefore, the weight of output connections remains unchanged and is determined by means of the LMS algorithm after network optimization. To use the HS algorithm, we should introduce the evaluation function at first. Here, s rate is introduced as the evaluation function. In other words, the purpose of the algorithm is to reduce eigenvalue spreads of the state autocorrelation matrix.

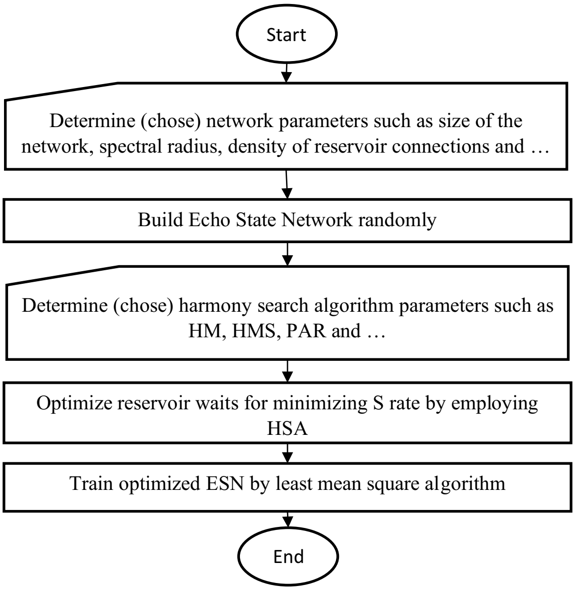
$$\mathrm{R} = \mathrm{E}\left[\emptyset(\mathrm{n})\emptyset^{\mathrm{T}}(\mathrm{n})\right] \tag{8}$$

$$\mathrm{f}(\mathrm{x}) = \mathrm{s} = \left|\frac{\lambda_{\mathrm{max}}(\mathrm{R})}{\lambda_{\mathrm{min}}(\mathrm{R})}\right| \tag{9}$$

In the above relations, the expression of (n) = [u(n) x(n) y(n – 1)] holds true, and λ_{max}(R) and λ_{min}(R) are representative of the maximum and minimum values of the correlation matrix corresponding to R, respectively. In order to use time in a better way, it is suggested that we construct an optimized echo state neural network for use within an extensive range of applied issues. Thereafter, we can use this network for each specific application with a slight modification. The relevant modifications can also be made by means of HSA with fewer repetitions.

In the second step, the network constructed by HSA is trained in an online state by means of the LMS algorithm. For the training of ESN, the step size of the algorithm (μ) is selected. The step size is set as $\mu = \frac{1}{2.\lambda_{max}}$ to make a balance between algorithm speed and precision. Then, training data are operated on the optimized network, reservoir states are calculated through (1), and the weight of output connections of the network are regulated through Eq. (5). Flowchart 1 shows the proposed method briefly. Using the LMS, the algorithm not only increases the speed of training and needs a very smaller memory, but also it increases the network stability via proper step sizes.

Flowchart 1 Brief description of the proposed method.


Start
Determine (chose) network parameters such as size of the network, spectral radius, density of reservoir connections and …
Build Echo State Network randomly
Determine (chose) harmony search algorithm parameters such as HM, HMS, PAR and …
Optimize reservoir waits for minimizing S rate by employing HSA
Train optimized ESN by least mean square algorithm
End

6 Simulation

One of the most common applications of ESNs is the prediction of chaotic time series. In this paper, we have used this network for the prediction of the Rossler, Mackey–Glass, and Lorenz time series. To this end, we should determine the number of reservoir units and their spectral radius. In this paper, 500 units in the reservoir network and a spectral radius of 0.90 have been used. The size of training data sets in all simulations has been considered 4000 where the first 500 samples are ignored to nullify the effect of the initial states of the reservoir. The next 3000 ones are used for training and the remaining 500 ones are used for performance evaluation. All the simulations have been made in the Matlab environment version of 2013 by means of a personal computer with INTEL CORETM i7-3470 CPU to make certain about the correctness of obtained results.

6.1 Mackey–Glass Time Series

Mackey–Glass time series is obtained through the following differential equation:

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

In the above equation, x(t) represents the value of the Mackey–Glass time series at time t; in order for the time series to be chaotic, we have selected equation parameters as $\delta = 17$, $\beta = -0.1$, and $\alpha = 0.2$. This time, the series is constructed by means of the second-order Runge–Kutta method with a step size of 0.01 [41, 47].

6.2 Lorenz Time Series

The following three-variable differential (11) is used for the construction of the Lorenz time series.

$$\begin{cases} \frac{dx}{dt} = -a\, x(t) + a\, y(t) \\ \frac{dy}{dt} = b\, x(t) - y(t) - x(t)z(t) \\ \frac{dz}{dt} = x(t)y(t) - c\, z(t) \end{cases} \tag{11}$$

To ensure that the obtained series is chaotic, we should select parameters as $a = 10$, $b = 28$, and $c = \frac{8}{3}$, and we produce data sets using the fourth-order Runge–Kutta method with a step size of 0.02 and initial states of x (0) = 12, y (0) = 2, and z(0) = 9.

6.3 Rossler Time Series

The following differential (12) is used for the construction of the Rossler time series.

$$\begin{cases} \frac{dx}{dt} = -z(t) - y(t) \\ \frac{dy}{dt} = x(t) + d\, y(t) \\ \frac{dz}{dt} = e + z(t)(x(t) - f) \end{cases} \tag{12}$$

To produce data sets out of the above chaotic time series, we should consider $d = 0.15$, $e = 0.2$, and $f = 10$ and use the fourth-order Runge–Kutta method with a step size of 0.01.

After the production of training data sets, we use the proposed optimized ESN for the construction and training of the network. In the first step, weights of the network reservoir should be produced in a way that correlation matrix eigenvalues of the reservoir have the least possible spreads. Algorithm parameters have also been selected as $HMS = 10$, $HMCR = 0.85$, $PAR = 0.2$, $bw = 0.4$, $NI = 100000$, and $X \in \{-0.4, -0.2, 0, 0.2, 0.4\}$. In this step, three training data sets are operated

Table 1 Eigenvalue spreads different structures

Structure	S rate
ESN	2.0762×10^{15}
Scale-free	6.3238×10^{13}
Small world	2.5567×10^{13}
Spatial growth	1.4262×10^{14}
LMS-ESN	8.4639×10^{8}

on the input network simultaneously, and the algorithm is repeated NI times. Table 1 shows eigenvalue spreads of the correlation matrix of network internal states constructed with compromise structures, of the network constructed with the conventional random method, and of the network constructed with HSA. It clearly shows the performance of HSA in the optimization construction of ESN.

In Table 1, short expressions of LMS-ESN have been used for the proposed method. As it is seen, the S rate has been reduced 106 times through HAS compared to the available methods. The algorithm is also repeated $\frac{NI}{10}$ other times for each time series. The obtained results are briefly illustrated in Table 2.

The obtained results show that training online conditions of ESN have been provided using the LMS algorithm; therefore, in the second step, the LMS algorithm is used for online training of ESN. In this paper, mean squared error for training (MSEtrain) and for testing (MSEtest) and spent time during network training (t) are used as performance criteria of each training method.

$$\mathrm{MSE} = \frac{1}{T}\sum\nolimits_{n=1}^{T} \|D(n) - Y(n)\|^2 \tag{13}$$

Tables 3, 4, and 5 show the results obtained from Mackey–Glass, Lorenz, and Rossler time series, respectively.

As it is seen in the above tables, the speed of the proposed method for online training of echo state network is higher than other methods, and the precision of this method is also higher than that of PSO and HS. Furthermore, this proposed method guarantees network stability with selecting a proper step size and uses a very shorter memory compared to existing methods.

Table 2 Eigenvalue spreads of the network state matrix

Time series	S RATE
Mackey glass	6.3564×10^{8}
Rossler	1.3491×10^{8}
Lorenz	8.7259×10^{8}

Table 3 Prediction results of Mackey glass time series

Algorithm	MSE_{TRAIN}	MSE_{TEST}	Time(s)
RLS	$5.23\ e^{-11}$	$4.71\ e^{-10}$	1722
PSO	$9.43\ e^{-6}$	$5.81\ e^{-6}$	981
HS	$2.67\ e^{-7}$	$9.28\ e^{-7}$	853
LMS	$4.26\ e^{-10}$	$8.63\ e^{-10}$	113

Table 4 Prediction results of Lorenz time series

Algorithm	MSE_{TRAIN}	MSE_{TEST}	Time(s)
RLS	$2.38\ e^{-10}$	$7.45\ e^{-10}$	1912
PSO	$6.84\ e^{-6}$	$9.12\ e^{-6}$	1025
HS	$6.73\ e^{-7}$	$1.78\ e^{-6}$	960
LMS	$8.35\ e^{-10}$	$5.83\ e^{-9}$	152

Table 5 Prediction results of Rossler time series

Algorithm	MSE_{TRAIN}	MSE_{TEST}	Time(s)
RLS	$7.42\ e^{-11}$	$2.63\ e^{-10}$	1786
PSO	$5.71\ e^{-7}$	$9.45\ e^{-7}$	934
HS	$7.53\ e^{-7}$	$4.62\ e^{-6}$	863
LMS	$2.75\ e^{-10}$	$8.54\ e^{-10}$	105

7 Conclusion

In this paper, a mixed method using LMS and HS algorithm has been introduced to tackle the issue and pitfalls of online training of ESN. In this proposed method, the structure and weight of internal connections of echo state neural network are produced so that we have the lowest possible value for eigenvalue spreads of the internal state correlation matrix. Then, the network is trained in an online state by means of an LMS algorithm. Simulation results clearly indicate that spent time during network training through the proposed method is much lower than that of other methods while the precision of outputs is acceptable. In addition, the proposed method is advantageous over the other two methods because it has simple calculations, guarantees network stability, and uses less memory. It is also possible to construct an echo state neural network by means of HSA and apply it within an extensive range of problems. Although the training time has been improved in this method, the process of optimizing reservoir connections is a time-consuming and complex process. In the future, other methods of improving reservoir performance can be studied and investigated (Appendix: Table 6).

Appendix

Table 6 Index and parameters description

Index or parameter	Description
K, N, L	Number of input, reservoir, and output units of ESN respectively
W, W^{in}, W^{back} *and* W^{out}	Matrix of internal, input, feedback, and output weights respectively
λ_{max}	The largest spectral radius of the internal weight matrix
α	Optional spectral radius
μ	The step size of the LMS algorithm
HM, HMS, HMCR, PAR	Harmony Memory, Harmony Memory Size, Harmony Memory Considering Rate, and Pitch Adjusting Rate respectively
$s = \left\| \frac{\lambda_{\max}(R)}{\lambda_{\min}(R)} \right\|$	eigenvalue spreads of the state autocorrelation matrix
$\lambda_{\max}(R)$ and $\lambda_{\min}(R)$	maximum and minimum values of the state autocorrelation matrix, respectively

Author Contribution All authors contributed to the study conception and design. Material preparation, data collection, and analysis were performed by [Javad Saadat], [Mohsen Farshad], and [Hussein Eliasi]. The first draft of the manuscript was written by [Javad Saadat], and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Data Availability In this article, all the simulations are done in MATLAB software. If needed, the relevant files will be sent via email. Saadatjavad90@birjand.ac.ir, Saadatjavad90@gmail.com, Saadatjavad90@yahoo.com.

Declarations

Consent for Publication Authors should make sure to also seek consent from individuals to publish their data prior to submitting their paper to a journal. This is in particular applicable to case studies.

Competing Interests This article is extracted from Javad Saadat Ph.D thesis, Professor Mohsen Farshad and Professor Hossein Eliasi are the supervisors of this thesis.

References

1. Schrauwen B, Wardermann M, Verstraeten D, Steil JJ, Stroobandt D (2008) Improving reservoirs using intrinsic plasticity. Neurocomputing 71(7–9):1159–1171
2. Steil JJ (2007) Online reservoir adaptation by intrinsic plasticity for backpropagation–decorrelation and echo state learning. Neural Netw 20(3):353–364
3. Xia Y, Jelfs B, Van Hulle MM, Príncipe JC, Mandic DP (2010) An augmented echo state network for nonlinear adaptive filtering of complex noncircular signals. IEEE Trans Neural Netw 22(1):74–83
4. Jaeger H, Lukoševičius M, Popovici D, Siewert U (2007) Optimization and applications of echo state networks with leaky-integrator neurons. Neural Netw 20(3):335–352
5. Lun SX, Hu HF, Yao XS (2019) The modified sufficient conditions for echo state property and parameter optimization of leaky integrator echo state network. Appl Soft Comput 77:750–760
6. Chatzis SP, Demiris Y (2011) Echo state Gaussian process. IEEE Trans Neural Netw 22(9):1435–1445
7. Deng Z, Zhang Y (2007) Collective behavior of a small-world recurrent neural system with scale-free distribution. IEEE Trans Neural Netw 18(5):1364–1375
8. Gallicchio C, Micheli A, Pedrelli L (2018) Design of deep echo state networks. Neural Netw 108:33–47

9. Xue Y, Yang L, Haykin S (2007) Decoupled echo state networks with lateral inhibition. Neural Netw 20(3):365–376
10. Ozturk MC, Xu D, Principe JC (2007) Analysis and design of echo state networks. Neural Comput 19(1):111–138
11. Rodan A, Tino P (2010) Minimum complexity echo state network. IEEE Trans Neural Netw 22(1):131–144
12. Liu J, Sun T, Luo Y, Yang S, Cao Y, Zhai J (2020) An echo state network architecture based on quantum logic gate and its optimization. Neurocomputing 371:100–107
13. Yildiz IB, Jaeger H, Kiebel SJ (2012) Re-visiting the echo state property. Neural Netw 35:1–9
14. Manjunath G, Jaeger H (2013) Echo state property linked to an input: exploring a fundamental characteristic of recurrent neural networks. Neural Comput 25(3):671–696
15. Gallicchio C, Micheli A (2017) Echo state property of deep reservoir computing networks. Cogn Comput 9(3):337–350
16. Li X, Bi F, Yang X, Bi X (2022) An echo state network with improved topology for time series prediction. IEEE Sens J 22(6):5869–5878
17. Shi Z, Han M (2007) Support vector echo-state machine for chaotic time-series prediction. IEEE Trans Neural Netw 18(2):359–372
18. Wang Z, Yao X, Huang Z, Liu L (2021) Deep echo state network with multiple adaptive reservoirs for time series prediction. IEEE Transact Cognitive Develop Syst 13(3):693–704
19. Chen HC, Wei DQ (2021) Chaotic time series prediction using echo state network based on selective opposition grey wolf optimizer. Nonlinear Dyn 104(4):3925–3935
20. Skowronski MD, Harris JG (2007) Automatic speech recognition using a predictive echo state network classifier. Neural Netw 20(3):414–423
21. Tong MH, Bickett AD, Christiansen EM, Cottrell GW (2007) Learning grammatical structure with echo state networks. Neural Netw 20(3):424–432
22. Jaeger H, Haas H (2004) Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. Science 304(5667):78–80
23. Ozturk MC, Principe JC (2007) An associative memory readout for ESNs with applications to dynamical pattern recognition. Neural Netw 20(3):377–390
24. Wang H, Lei Z, Liu Y, Peng J, Liu J (2019) Echo state network based ensemble approach for wind power forecasting. Energy Convers Manage 201:112188
25. Chitsazan MA, Fadali MS, Trzynadlowski M (2019) Wind speed and wind direction forecasting using echo state network with nonlinear functions. Renew Energy 131:879–889
26. Hu H, Wang L, Tao R (2021) Wind speed forecasting based on variational mode decomposition and improved echo state network. Renew Energy 164:729–751
27. Stefenon SF, Seman LO, Neto NFS, Meyer LH, Nied A, Yow KC (2022) Echo state network applied for classification of medium voltage insulators. Int J Electr Power Energy Syst 134:107336
28. Hu H, Wang L, Lv SX (2020) Forecasting energy consumption and wind power generation using deep echo state network. Renew Energy 154:598–613
29. Hu H, Wang L, Peng L, Zeng YR (2020) Effective energy consumption forecasting using enhanced bagged echo state network. Energy 193:116778
30. Lukoševičius M, Jaeger H (2009) Reservoir computing approaches to recurrent neural network training. Comput Sci Rev 3(3):127–149
31. Liebald B (2004) Exploration of effects of different network topologies on the ESN signal crosscorrelation matrix spectrum. Bachelor of Science (B. Sc.) thesis, International University Bremen, Spring
32. Kawai Y, Park J, Asada M (2019) A small-world topology enhances the echo state property and signal propagation in reservoir computing. Neural Netw 112:15–23
33. Song Q, Zhao X, Feng Z, Song B (2011) Recursive least squares algorithm with adaptive forgetting factor based on echo state network. In 2011 9th World Congress on Intelligent Control and Automation (pp. 295–298). IEEE
34. Xue Y, Zhang Q, Neri F (2021) Self-adaptive particle swarm optimization-based echo state network for time series prediction. Int J Neural Syst 31(12):2150057
35. Chouikhi N, Ammar B, Rokbani N, Alimi AM (2017) PSO-based analysis of echo state network parameters for time series forecasting. Appl Soft Comput 55:211–225
36. Saadat J, Moallem P, Koofigar H (2017) Training echo state neural network using harmony search algorithm. Int J Artif Intell 15(1):163–179
37. Lukoševicius M, Jaeger H (2007) Overview of reservoir recipes (No. 11). Technical Report

38. Jaeger H (2001) The “echo state” approach to analyzing and training recurrent neural networks-with an erratum note. Bonn, Germany: German National Research Center for Information Technology GMD Technical Report 148(34):13
39. Jaeger H (2002) Short term memory in echo state networks. gmd-report 152. In GMD-German National Research Institute for Computer Science (2002). http://www.faculty.jacobs-university.de/hjaeger/pubs/STMEchoStatesTechRep.pdf
40. Jaeger H, Maass W, Principe J (2007) Special issue on echo state networks and liquid state machines
41. Kennel MB, Brown R, Abarbanel HD (1992) Determining embedding dimension for phase-space reconstruction using a geometrical construction. Phys Rev A 45(6):3403
42. Widrow B, McCool J, Larimore MG, Johnson CR (1977) Stationary and nonstationary learning characteristics of the LMS adaptive filter. In Aspects of signal processing (pp. 355–393). Springer, Dordrecht
43. Lee KS, Geem ZW (2005) A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Comput Methods Appl Mech Eng 194(36–38):3902–3933
44. Heinzelman WB, Chandrakasan AP, Balakrishnan H (2002) An application-specific protocol architecture for wireless microsensor networks. IEEE Trans Wireless Commun 1(4):660–670
45. Geem ZW (Ed.) (2009) Music-inspired harmony search algorithm: theory and applications (Vol. 191). Springer
46. Lee KS, Geem ZW (2004) A new structural optimization method based on the harmony search algorithm. Comput Struct 82(9–10):781–798
47. Fraser AM, Swinney HL (1986) Independent coordinates for strange attractors from mutual information. Phys Rev A 33(2):1134