Stable Training Method for Echo State Networks with Output Feedbacks

Qingsong Song, Zuren Feng, *Member, IEEE*, and Mingli Lei

Abstract— **In applications of echo state network (ESN), the Wiener-Hopf solution is usually used to learn the ESN's output connection weights; however, the solution can hardly ensure the asymptotic stability of the ESNs running in a closed-loop generative mode. The reason is firstly analyzed. A sufficient condition of the asymptotic stability for the closed-loop running ESNs is proposed and proved. In addition, the output connection weight learning problem is translated into an optimization problem with a nonlinear restriction. Particle swarm optimization algorithm is explored to solve the optimization problem. The simulation experiment results show that the output weight adaptation alglrithm we proposed (we call it PSOESN) can not only result in the high-precision prediction outputs of the trained ESN, but also ensure its asymptotic stability.**

I. Introduction

SINCE being proposed in [1], Echo state networks (ESNs) have been successfully applied in many cases, such as in nonlinear time series prediction [2], in speech recognition [3], in mobile robot modeling and control [4]. Usually, an ESN consists of two main component parts: a large-scale recurrent neural network (RNN) (it is called "reservoir" in [1]) and a linear readout. The most distinctive characteristic of the ESN is that, only the weights for the connections from the reservoir neurons to the readout neurons need to be adapted for certain task. In addition, because the reservoir has rich enough dynamics, the adaptation can be treated as a simple linear regression problem. The troubling issues on RNN weight adaptation are almost entirely evaded, such as local minima, slow convergence, and even non-convergence [5].

This work was supported in part by the State Key Development Program for Basic Research of China (Grant No. 2007CB311006), and the National Natural Science Foundation of China (Grant No. 60875043).

Qingsong Song is with the System Engineering Institute, Xi'an Jiaotong University, and with the State Key Laboratory of Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an 710049, CHINA (corresponding author, phone: +86-29-82667771; fax: +86-29-82665487; e-mail: qssong@sei.xjtu.edu.cn).

Zuren Feng is with the System Engineering Institute, Xi'an Jiaotong University, and with the State Key Laboratory of Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an 710049, CHINA (e-mail: fzr9910@xjtu.edu.cn).

Mingli Lei is also with the System Engineering Institute, Xi'an Jiaotong University, and with the State Key Laboratory of Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an 710049, CHINA (e-mail: mllei768@126.com).

However, the ESNs may lose their asymptotic stability as they run in a closed-loop generative mode. In such mode, the outputs are fed back into the reservoir, a closed-loop forms (as shown in Fig.2). All we all know, the stability is prerequisite to successful neural network implementations. There are lots of literature concerning Hopfield networks', cellular neural networks' and bi-directional associative memory networks' stability. Constructing some Lyapunov functional is the most common method [6,7]. There are few literature concerning the stability of the ESNs, however. Two representative methods are the ridge regression and the noise immunization [8,9]. But, these two methods have obviously insufficiencies: it is not an easy thing to correctly set the regularization factor for the ridge regression or the noise amount for the noise immunization for specific tasks.

In order to ensure the asymptotic stability of the ESNs running in a closed-loop generative mode, we proposes another output weight adaptation algorithm, which bases on particle swarm optimization (PSO) algorithm [10-12]. We call it PSOESN. PSOESN considers the output accuracy and the stability restraint during the adaptation. The simulation experiment results show that our algorithm can not only result in high-precision prediction outputs, but also ensure the asymptotic stability of the ESNs.

II. Asymptotic stability of the autonomous ESN

We consider the ESNs made from leaky integrator neurons, since the dynamics of the leaky integrator neuron can be slowed down or sped up [9]. Fig.1 plots the system architecture of the ESN. The ESN consists of three parts: the front-part is input, the middle reservoir, and the back-end readout. And the ESN is assumed having L input neurons, M output neurons, and N reservoir neurons. Real-valued connection weights are collected in a matrix $\mathbf{W}^{in}$ for the input weights, in a matrix $\mathbf{W}$ for the reservoir connections, in a matrix $\mathbf{W}^{out}$ for the output weight matrix, and in a matrix $\mathbf{W}^{back}$ for the output feedback matrix.

The task is to train an ESN as a trajectory-generator or a pattern-generator. The training algorithm which is usually used in the literature consists of four Steps [13].

In the first Step, an untrained ESN ($\mathbf{W}^{in}$, $\mathbf{W}$, $\mathbf{W}^{back}$) is constructed, which is consistent with the assignment to the parameters (the leaking rate (a) and the gain (γ) of the leaky integrator neurons, the spectral radius ($\rho(\mathbf{W})$) and the connectivity density (D) of the reservoir, and the L, M, N).

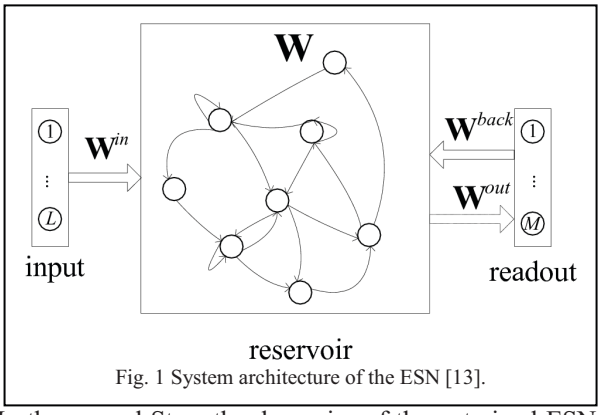

Fig. 1 System architecture of the ESN [13].

In the second Step, the dynamics of the untrained ESN is sampled. Given a training sequence $y_d(k)$, $k = 1,...,T_t$. The reservoir state is updated according to Equation 1.

$$\mathbf{X}(k+1) = (1-a\gamma)\mathbf{X}(k) + \gamma f^{PE}(\mathbf{W}^{in}u(k) + \mathbf{W}\mathbf{X}(k) + \mathbf{W}^{back}y_d(k) + \mathbf{v}(k)) \quad (1)$$

where f^{PE} is a hyperbolic tangent function applied component-wise; $\mathbf{X}(k)$, $u(k)$ and $\mathbf{v}(k)$ are the reservoir state, constant input bias and small uniform noise at time instant k, respectively. $\mathbf{X}(1) = \mathbf{0}$. From the time instant $T_0 + 1$ to the time instant T_t, the reservoir state $\mathbf{X}(k)$ and the training data point $y_d(k)$ are per time instant collected into the matrix $\mathbf{M}$ and the matrix $\mathbf{T}$, respectively. In order to eliminate the effect the initial reservoir state results in, the dynamics of the ESN is not sampled before the time instant T_0.

In the third Step, $\mathbf{W}^{out}$ is computed. As a linear readout neuron is assumed, the training problem is equivalent to a linear regression problem. The Wiener-Hopf solution is one of the most common methods to solve the regression problem.

$$y(k+1) = \mathbf{W}^{out}\mathbf{X}(k+1) \quad (2)$$

$$\mathbf{W}^{out} = ((\mathbf{M}'\mathbf{M})^{-1}(\mathbf{M}'\mathbf{T}))' \quad (3)$$

where the superscript $'$ means the transpose of a matrix; the superscript $^{-1}$ means the inverse of a matrix.

In the last Step, the resulting ESN ($\mathbf{W}^{in}$, $\mathbf{W}$, $\mathbf{W}^{back}$, $\mathbf{W}^{out}$) is ready for exploitation. Given an exploitation sequence $y(k)$, $k = 1,...,T_e$, $T_e \gg T_0$. The exploitation sequence and the training sequence originate from the same one system, but they may have different initial values ($y(1) \neq y_d(1)$). In order to excite the reservoir, while $k \leq T_0$, $y(k)$ is fed into the reservoir via the feedback connections, the reservoir state is updated according to Equation 1. While $k > T_0$, however, the resulting ESN is updated as follows.

$$\mathbf{X}(k+1) = (1-a\gamma)\mathbf{X}(k) + \gamma f^{PE}(\mathbf{W}^{in}u(k) + \mathbf{W}\mathbf{X}(k) + \mathbf{W}^{back}\hat{y}(k)) \quad (4)$$

$$\hat{y}(k+1) = \mathbf{W}^{out}\mathbf{X}(k+1) \quad (5)$$

where $\hat{y}(k)$ is the predicting output at time instant k.

We can see from Equation 4 and 5 that, the predicting output $\hat{y}(k)$ is fed back into the reservoir; a closed-loop emerges. Assuming $u(k) = 0$ holds. We can say that the ESN acts as an autonomous dynamical system and carries out a one-step predicting task. The system architecture of the autonomous ESN is shown in Fig.2.

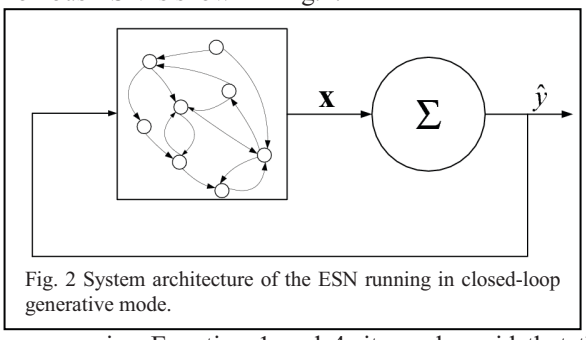

Fig. 2 System architecture of the ESN running in closed-loop generative mode.

By comparing Equation 1 and 4, it can be said that the sufficient condition ($\rho((1-a\gamma)\mathbf{I} + \gamma\mathbf{W}) < 1$) [9] for the echo state property for the untrained ESN can not ensure the asymptotic stability of the resulting ESN. The stability of the autonomous ESN difference dynamical system is determined by the matrix $\widetilde{\mathbf{W}}$. The spectral radius of the matrix $\widetilde{\mathbf{W}}$, i.e. $\rho(\widetilde{\mathbf{W}})$, is named as Effective Spectral Radius (ESR).

$$\widetilde{\mathbf{W}} = (1-a\gamma)\mathbf{I} + \gamma(\mathbf{W} + \mathbf{W}^{back}\mathbf{W}^{out}) \quad (6)$$

Theorem 1. For the autonomous ESN difference dynamical system described by Equation 4, a sufficient condition for its asymptotic stability is $\rho(\widetilde{\mathbf{W}}) < 1$.

Proof. The significance of the parameter γ is to slow down or speed up the dynamics of the ESN; it does not matter with the stability [9]. Assuming $\gamma = 1$ holds.

Let $\mathbf{X}_{k+1}$ and $\tilde{\mathbf{X}}_{k+1}$ are two possible arbitrary reservoir states at time instant $k+1$. $\mathbf{X}_{k+1}, \tilde{\mathbf{X}}_{k+1} \in \Re^N$. Since $f^{PE} = \tanh$, there is $\| f^{PE}(\mathbf{X}_{k+1}) - f^{PE}(\tilde{\mathbf{X}}_{k+1}) \| \leq \| \mathbf{X}_{k+1} - \tilde{\mathbf{X}}_{k+1} \|$.

Since all finite dimensional norms are equivalent, proving convergence in the 2-norm guarantees convergence in every other finite dimensional norm [14].

$$\begin{aligned} \| (\Delta\mathbf{X})_{k+1} \|_2 &= \| \mathbf{X}_{k+1} - \tilde{\mathbf{X}}_{k+1} \|_2 \\ &= \| ((1-a)\mathbf{X}_k + f((\mathbf{W} + \mathbf{W}^{back}\mathbf{W}^{out})\mathbf{X}_k)) \\ &\quad - ((1-a)\tilde{\mathbf{X}}_k + f((\mathbf{W} + \mathbf{W}^{back}\mathbf{W}^{out})\tilde{\mathbf{X}}_k)) \|_2 \\ &= \| ((1-a)(\mathbf{X}_k - \tilde{\mathbf{X}}_k) + (f((\mathbf{W} + \mathbf{W}^{back}\mathbf{W}^{out})\mathbf{X}_k) \\ &\quad - f((\mathbf{W} + \mathbf{W}^{back}\mathbf{W}^{out})\tilde{\mathbf{X}}_k)) \|_2 \end{aligned}$$

$$
\begin{aligned}
&\le \| ((1-a)(\mathbf{X}_k - \tilde{\mathbf{X}}_k) \|_2 \\
&\quad + \| f((\mathbf{W} + \mathbf{W}^{back}\mathbf{W}^{out})\mathbf{X}_k) \\
&\quad - f((\mathbf{W} + \mathbf{W}^{back}\mathbf{W}^{out})\tilde{\mathbf{X}}_k) \|_2 \\
&\le \| ((1-a)(\mathbf{X}_k - \tilde{\mathbf{X}}_k) \|_2 \\
&\quad + \| (\mathbf{W} + \mathbf{W}^{back}\mathbf{W}^{out})(\mathbf{X}_k - \tilde{\mathbf{X}}_k) \|_2 \\
&= \| ((1-a)\mathbf{I} + (\mathbf{W} + \mathbf{W}^{back}\mathbf{W}^{out}))(\mathbf{X}_k - \tilde{\mathbf{X}}_k) \|_2 \\
&\le \| (1-a)\mathbf{I} + (\mathbf{W} + \mathbf{W}^{back}\mathbf{W}^{out}) \|_2 \| (\Delta\mathbf{X})_k \|_2 \\
&= \bar{\sigma}(\widetilde{\mathbf{W}}) \| (\Delta\mathbf{X})_k \|_2
\end{aligned} \qquad (7)
$$

So, $\bar{\sigma}(\widetilde{\mathbf{W}}) < 1$ can ensure $\lim_{k\to\infty} \| (\Delta\mathbf{X})_k \|_2 = \mathbf{0}$, here $\bar{\sigma}(\widetilde{\mathbf{W}})$ is the largest singular value of the matrix $\widetilde{\mathbf{W}}$. And further more, there is $\underline{\sigma}(\tilde{\mathbf{w}}) \le |\lambda_i(\tilde{\mathbf{w}})| \le \bar{\sigma}(\tilde{\mathbf{w}})$ [15], here $\underline{\sigma}(\widetilde{\mathbf{W}})$ is the smallest singular value of the matrix $\widetilde{\mathbf{W}}$, $\lambda_i(\widetilde{\mathbf{W}})$ is the i^{th} eigenvalue of the matrix $\widetilde{\mathbf{W}}$, $i = 1,...,N$. So there holds $\rho(\widetilde{\mathbf{W}}) = \max_i(|\lambda_i(\widetilde{\mathbf{W}})|) \le \bar{\sigma}(\widetilde{\mathbf{W}}) < 1$. Theorem 1 has been proved.

In other word, if the loop gain of the autonomous system illustrated with Fig.2 is smaller than unit, its asymptotic stability can be ensured.

III. The Problem-Solving Process by PSO Algorithm

It can be seen from Theorem 1 that the matrix $\mathbf{W}^{out}$ has important effect not only on the predicting output precision, but also on the asymptotic stability of the resulting ESN. Equation 3 only takes into account the predicting output precision, neglects the stability restriction, however. We consider both of them. The output connection weight adaptation problem is translated into a constrained optimization problem.

$$
\begin{aligned}
&\min_{\mathbf{W}^{out}} \quad \sum_{k=T_0+1}^{T_t} (y_d(k) - \mathbf{W}^{out}\mathbf{X}(k))^2 \\
&s.t. \qquad \rho(\widetilde{\mathbf{W}}) < 1
\end{aligned} \qquad (8)
$$

We can hardly find a gradient-descent based non-stochastic algorithm to solve Equation 8. PSO is a population-based stochastic algorithm, inspired by social behavior of bird flocking. PSO algorithm can successfully optimize a wide range of continuous functions in a faster, cheaper way compared with other methods [11,16]. The detailed descriptions of PSO algorithm are given in [10-12]. We use it to solve Equation 8.

The objective function and the stability restriction in Equation 8 are synthesized into a single evaluation function $\Phi(\mathbf{W}^{out})$.

$$
\Phi(\mathbf{W}^{out}) = f(\mathbf{W}^{out}) + r_1 \cdot \theta \cdot h(\mathbf{W}^{out})^{r_2} \qquad (9)
$$

$$
f(\mathbf{W}^{out}) = \frac{1}{T_t - T_0} \sum_{k=T_0+1}^{T_t} (y_d(k) - \mathbf{W}^{out}\mathbf{X}(k))^2 \qquad (10)
$$

$$
h(\mathbf{W}^{out}) = \max[0, (\rho(\widetilde{\mathbf{W}}) - 1)] \qquad (11)
$$

where $f(\mathbf{W}^{out})$ scores the training error, i.e., mean square error (MSE); $h(\mathbf{W}^{out})$ scores the restriction violation amount; the parameters r_1, r_2 and θ determine the penalty function.

In order to insure the convergence of PSO algorithm, we use the PSO algorithm with constriction factor [12]. Each particle has two parameters: its position ($\mathbf{p}$) and its velocity ($\mathbf{V}$). The evaluation function $\Phi(\mathbf{W}^{out})$ is computed using each particle's positional coordinates as input values. Each position and each velocity are adjusted according to Equation 12 and 13 and the evaluation function is computed with the new coordinates at each time step.

$$
\begin{aligned}
\mathbf{V}(step+1) = \chi(&\mathbf{V}(step) \\
&+ c_1 \cdot rand_1 \cdot (\mathbf{p}_{best}(step) - \mathbf{p}(step)) \\
&+ c_2 \cdot rand_2 \cdot (\mathbf{g}_{best}(step) - \mathbf{p}(step)))
\end{aligned} \qquad (12)
$$

$$
\mathbf{p}(step+1) = \mathbf{p}(step) + \mathbf{V}(step+1) \qquad (13)
$$

where χ is the constriction factor, $\chi = 0.729$; $step$ is iteration counter; c_1 and c_2 are acceleration coefficients, $c_1 = c_2 = 2.05$; $rand_1$ and $rand_2$ are two uniform random numbers in [0,1]; $\mathbf{p}_{best}$ is the best position encountered by the partilce so-far, $\mathbf{g}_{best}$ represents the best position found by any member in the whole swarm population.

Once the matrixes $\mathbf{M}$ and $\mathbf{T}$ have been collected, Our PSO-based output connection weight adaptation algorithm (PSOESN) is called to seek for the best values of $\mathbf{W}^{out}$, of which the pseudocode is presented in TABLE. I.

IV. Experiments and Results

The PSOESN is tested on the multiple superimposed oscillators (MSO) problem [17,18]. The trajectories to be learned by the ESN are of the function $\sin(0.2n) + \sin(0.311n)$, $n = 1,2,...$. The related parameters are listed in TABLE. II. The non-zero entries of $\mathbf{W}$ and $\mathbf{W}^{back}$ are all uniformly distributed random variables within the range [-0.5, 0.5]. The parameters of the penalty function (r_1, r_2, θ) are specified as Equation 14, 15 and 16. As the evolvement goes on, the infeasible solutions are punished more and more severely. And if the restriction violation amounts are more, the punishment for the solutions will be severer also.

$$
r_1 = (step)^{3/2} \qquad (14)
$$

$$
r_2 = \begin{cases} 3 & 10 < h(\mathbf{w}^{out}) \\ 2 & 1 < h(\mathbf{w}^{out}) \le 10 \\ 1 & h(\mathbf{w}^{out}) \le 1 \end{cases} \qquad (15)
$$

$$
\theta = \begin{cases} 30, & \text{if } h(\mathbf{W}^{out}) > 10; \\ 10, & \text{if } 1 < h(\mathbf{W}^{out}) \le 10; \\ 1, & \text{if } h(\mathbf{W}^{out}) \le 1; \end{cases} \qquad (16)
$$

We implement the simulation experiments by using MATLAB R14 on PC (Pentium 4, CPU 2.4GHz, DDRII

1.0GB). The four Steps training algorithm given in Section 2 runs independently 20 times; during each run the third Step of the four Steps training algorithm is replaced with the PSOESN. Each run consumes about 4 minutes. The results of all the runs are averaged to evaluate the PSOESN.

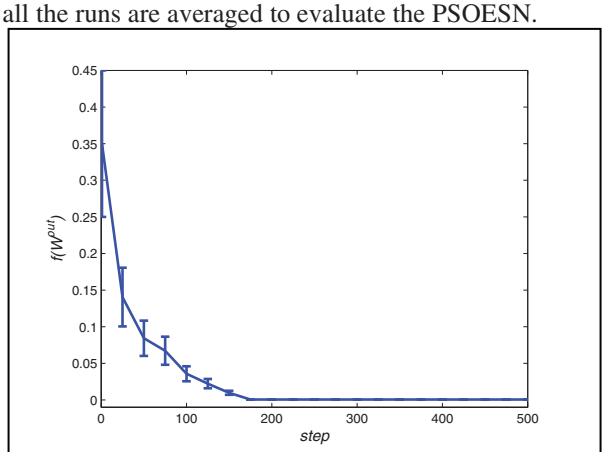

Fig. 3 Evolution of the values of the training error.

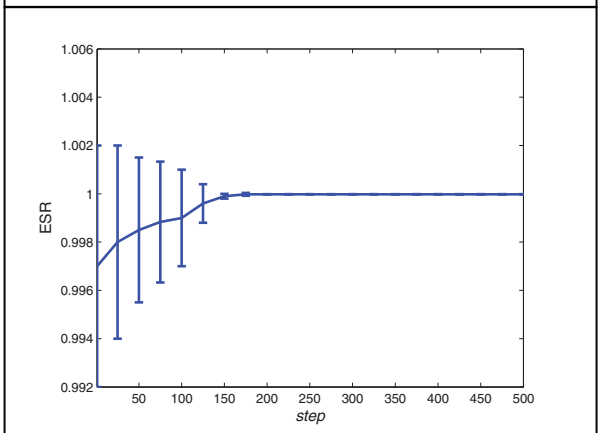

Fig. 4 Evolution of the values of ESR.

Fig.3 and Fig.4 give the evolution curves of $f(\mathbf{W}^{out})$ and $\rho(\widetilde{\mathbf{W}})$ respectively. It can be seen that the PSOESN has good robustness to different initial values of the matrix $\mathbf{W}^{out}$; and both the values of $f(\mathbf{W}^{out})$ and the values of $\rho(\widetilde{\mathbf{W}})$ have converged on the 200th step. The values of $\rho(\widetilde{\mathbf{W}})$ converge at 0.99998 finally; it satisfies **Theorem 1**, so the asymptotic stability of the resulting ESN can be insured.

During exploitation, the generalization MSE (E_{mse}) and the successful design ratio (S_{ratio}) [18] are computed.

$$E_{mse}^{i} = \left(\sum_{k=101}^{1100} (\hat{y}(k) - y(k))^2 \right) \Big/ 1000 \tag{17}$$

$$E_{mse} = \left(\sum_{i=1}^{20} E_{mse}^{i} \right) \Big/ 20 \tag{18}$$

$$S_{ratio} = \left(\sum_{i=1}^{20} \delta^{'}(E_{mse}^{i} - \theta) \right) \Big/ 20 \tag{19}$$

where $\hat{y}(k)$ and $y(k)$ are the predicting output and desired output at time instant k; E_{mse}^{i} is the MSE of the i^{th} run; if $E_{mse}^{i} < \theta$, then $\delta^{'} = 1$, and else if $E_{mse}^{i} \geq \theta$, $\delta^{'} = 0$; θ is a threshold, for convenience of comparison, let θ equal 0.0034 [18].

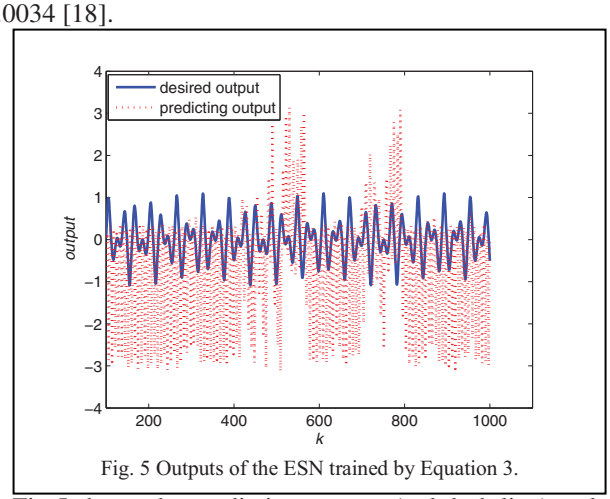

Fig. 5 Outputs of the ESN trained by Equation 3.

Fig.5 shows the predicting outputs (red dash line) and the desired outputs (blue solid line) of a resulting ESN during being exploited, which is trained by Equation 3. The ESN loses its stability (ESR>1). In fact, according to our experiments, stable ESN can not be obtained by Equation 3, the corresponding ESR values always exceed unit for all the 20 times experiments we conducted.

Fig.6 shows the predicting outputs of two resulting ESNs during being exploited, which are trained by the PSOESN. The blue dot line corresponds with the predicting output curve of a resulting ESN, of which the output weight values are the position of the best particle at the 100th step of the PSO algorithm. Whereas the red dash line corresponds with the predicting output curve of another resulting ESN, of which the output weight values are evolved until the 500th step of the PSO algorithm. The black solid line is the desired output curve. The corresponding predicting output errors are plotted in Fig.7. When the PSO algorithm evolves at the 100th step, a stable ESN is obtained: the corresponding ESR equals about 0.999. However, the output precision is not enough: the corresponding E_{mse} value is about 0.02. When the PSO algorithm stops its evolvement at the 500th step, the resulting ESN generates a perfect predicting output curve ($E_{mse} = 0.0005$, ESR = 0.99998).

The results from literature are collected in TABLE. Ⅲ. It can be said that the PSOESN outperforms *Evolino+LSTM* [17] and *DESN+RP* [18] for the same MSO problem. Though the PSOESN is slightly inferior to *DESN+MaxInfo* [18] on E_{mse}, it has obvious advantage on N and S_{ratio}.

We verify the noise-robustness of the resulting ESN trained by the PSOESN. While the ESN runs in the closed-loop generative mode, its predicting output is perturbed with Gaussian noise (the mean: 0, the variance: 0.01) from time steps 101 - 150. After that time, the noise is dispelled. Fig.8 plots the predicting output curve (the blue dash line) and the desired output curve (the black solid line).

Fig.9 plots the corresponding output errors. We can see that about 500 time steps after having been perturbed, the ESN has already converged back to the MSO attractor.

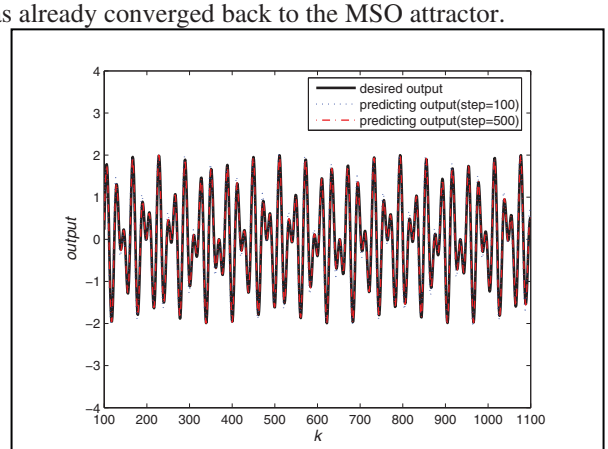

Fig. 6 Outputs of the ESN trained by the PSOESN.

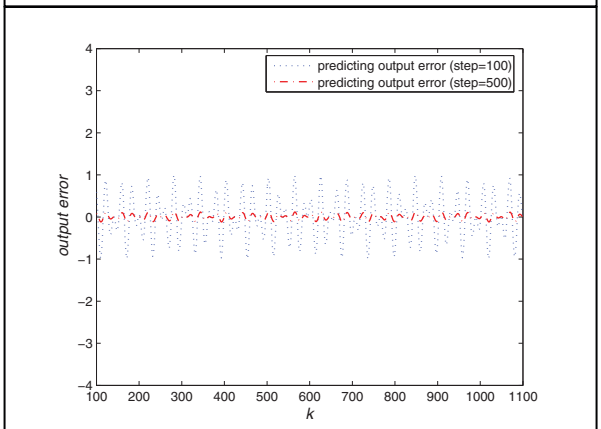

Fig. 7 The corresponding output errors of the ESN trained by the PSOESN.

V. Conclusion

We have analyzed the reason which results in the loss of the asymptotic stability of the ESNs running in a closed-loop generative mode. And then, we offer a sufficient condition for the asymptotic stability. Besides that, we consider the output connection weight adaptation problem as a kind of constrained optimization problem, and propose the PSOESN to solve the optimization problem. Our simulation experiment results show that the PSOESN can effectively meet the requirements of the output precision and of the asymptotic stability. It can be said that the PSOESN is an effective solution to the ESN output connection weight adaptation problem.

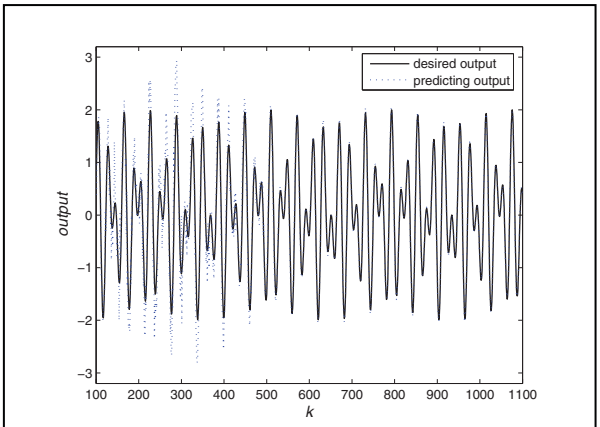

Fig. 8 Outputs of the ESN perturbed with noise. The ESN is trained by the PSOESN.

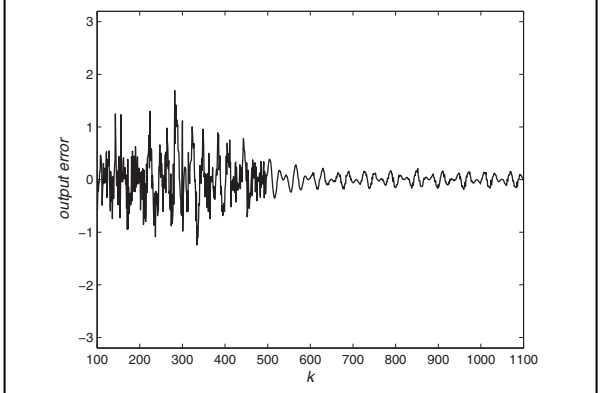

Fig. 9 The corresponding output errors of the ESN perturbed with noise.

[1] H. Jaeger, H. Haass, "Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, pp.78-80, Apr 2004.

[2] X. Lin, Z. Yang, Y. Song, "Short-term stock price prediction based on echo state networks," *Expert Systems with Applications*, vol. 36, pp.7313-7317, 2009.

[3] M. D. Skowronski, J. G. Harris, "Noise-Robust Automatic Speech Recognition Using a Predictive Echo State Network," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, pp.1724-1730, 2007.

[4] M. Salmen, P. G. Ploger, "Echo state networks used for motor control," In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, IEEE Press, 2005, pp.1953-1958.

[5] J. S. R. Jang, C. T. Sun, E. Mizutani, *Neuro-Fuzzy and Soft Computing: a Computational Approach to Learning and Machine Intelligence*. Prentice Hall, 1997.

[6] X. Lou, B. Cui, "On the global robust asymptotic stability of BAM neural networks with time-varying delays," *Neurocomputing*, vol. 70, pp.273-279, 2006.

[7] J. Xu, D. Pi, Y. Y. Cao, S. Zhong, "On stability of neural networks by a lyapunov functional-based approach," *IEEE Transactions on circuits and systems – I*, vol. 54, pp.912-924, 2007.

[8] F. Wyffels, B. Schrauwen, D. Stroobandt, "Stable output feedback in reservoir computing using ridge regression," In: *Proceedings of the 18th international conference on Artificial Neural Networks*, *Part I*, Lecture Notes in Computer Science, vol. 5163, pp.808-817, 2008.

[9] H. Jaeger, M. Lukoševičius, D. Popovici, U. Siewert, "Optimization and applications of echo state networks with leaky-integrator neurons," *Neural Networks*, vol. 20, pp.335-352, 2007.

[10] M. Clerc, "The swarm and the queen: towards a deterministic and adaptive particle swarm optimization," In: *Proc. Congress on Evolutionary Computation*, 1999, pp. 1951-1957.
[11] M. Clerc, J. Kennedy, "The particle swarm – explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on evolutionary computation*, vol. 6, pp.58-73, 2002.
[12] R.C. Eberhart, Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," In: *Proc. Congress on Evolutionary Computation*, 2000, pp. 84-88.
[13] H. Jaeger, "*Tutorial on training recurrent neural networks, covering BPTT, RTRL, EKF and the "echo state network" approach*," Tech. Rep. No. 159, Bremen: German National Research Center for Information Technology, 2002.
[14] M. Buehner, P. Young, "A tighter bound for the echo state property," *IEEE Transactions on Neural Networks*, vol. 17, pp.820-824, 2006.
[15] J. J. Steil, *Input-output stability of recurrent neural networks*. PhD thesis, University of Bielefeld, 1999.
[16] Y. Marinakis, M. Marinaki, M. Doumpos, C. Zopounidis, "Ant colony and particle swarm optimization for financial classification problems," *Expert Systems with Applications*, vol. 36, pp.10604-10611, 2009.
[17] D. Wierstra, F. J. Gomez, J. Schmidhuber, "Modeling systems with internal state using evolino," In: *Proc. of the 2005 conference on genetic and evolutionary computation (GECCO)*, 2005.
[18] Y. Xue, L. Yang, S. Haykin, "Decoupled echo state networks with lateral inhibition," *Neural Networks*, vol. 20, pp.365-376, 2007.

TABLE. I

PSEUDOCODE FOR THE PSOESN.

Initialization

Specify the particle population size (N_p), the maximal number of the generations ($step_m$), and the parameters of the penalty function (r_1 , r_2 , θ)

Generate the initial population of the particles

Evaluate each particle using Equation 9

Save the optimum solution of each particle

Save the optimum particle of the whole swarm

Main Phase

DO until the maximal number of the generations has been reached

Calculate the velocity and new position of each particle according to Equation 12 and 13

Calculate the new evaluation of each particle using Equation 9

Update the optimum solution of each particle and the optimum particle of the whole swarm

End DO

Return the best particle

TABLE. II

PARAMETER VALUES USED IN THE SIMULATION EXPERIMENTS.

L	N	M	$\rho(\mathbf{W})$	D	a	T_0	T_t	T_e	N_p	$step_m$
0	10	1	0.02	0.8	0.02	100	400	1000	30	500

TABLE. III

COMPARISONS OF SOME RESULTS.

method	N	E_{mse}	S_{ratio}
Evolino+LSTM	10	0.0034	/
DESN+RP	400	0.0021	0.228
DESN+MaxInfo	400	0.0003	0.635
PSOESN	10	0.0005	1