Complexity Control Method of Chaos Dynamics in Recurrent Neural Networks

Masao Sakai, Noriyasu Homma, and Kenichi Abe

Abstract: This paper demonstrates that the largest Lyapunov exponent λ of recurrent neural networks can be controlled efficiently by a stochastic gradient method. An essential core of the proposed method is a novel stochastic approximate formulation of the Lyapunov exponent λ as a function of the network parameters such as connection weights and thresholds of neural activation functions. By a gradient method, a direct calculation to minimize a square error $(\lambda - \lambda^{obj})^2$, where λ^{obj} is a desired exponent value, needs gradients collection through time which are given by a recursive calculation from past to present values. The collection is computationally expensive and causes *unstable* control of the exponent for networks with chaotic dynamics because of chaotic instability. The stochastic formulation derived in this paper gives us an approximation of the gradients collection in a fashion without the recursive calculation. This approximation can realize not only a faster calculation of the gradient, but also *stable* control for chaotic dynamics. Due to the non-recursive calculation, without respect to the time evolutions, the running times of this approximation grow only about as N^2 compared to as N^5T that is of the direct calculation method. It is also shown by simulation studies that the approximation is a robust formulation for the network size and that proposed method can control the chaos dynamics in recurrent neural networks efficiently.

Keywords: recurrent neural networks, chaos, lyapunov exponent, stochastic analysis

I. Introduction

Recurrent neural networks, consisting of units connected with each other, have a higher degree of the parameter freedom compared with that of feedforward neural networks composed of the same number of units. Harnessing the dynamics of complicated interactions among the units, the recurrent networks are expected to become a useful model for identifying and controlling the nonlinear complex dynamical systems[1]. Most of learning algorithms for the recurrent networks are based on the algorithms for the feedforward networks. For example, Jordan has proposed a new type of the recurrent networks which can be learned by the well-known back-propagation algorithm using the supervisory signals as the feedback signals[2]. In this case, the Jordan's recurrent networks can approximate the input-output function of the target systems even if the functions are nonlinear[3]. However there is no guarantee that a dynamical complexity of the recurrent networks converges to the target complexity[4],[5]. This means an actual behavior of the target systems could be different from an estimated one which is emulated by the recurrent networks with a conventional learning method.

As pioneers, Principe and Kuo have proposed a dynamic complexity learning method which updates the weights according to a forgetting function given by the largest Lyapunov exponent for feedforward networks[6]. For recurrent networks Deco and Schürman have reported that the dynamical complexity can be learned by a stochastic "sample-by-sample" update of the weights with the forgetting function[7]. These supervisory learning methods, however, need to observe time series from target systems as training data.

On the other hand, Homma *et al.* derived a direct control method of the largest Lyapunov exponent[8]. The method minimizes a square error which can be given as a function of the network parameters and time series of neurons' state variables. Then, the changes of parameters to minimize the square error are given by their gradients which are represented in terms of the gradient collection of their state variables through time. Therefore, the method doesn't need the target time series explicitly. But it has several problems: it is computationally expensive for large-scale recurrent neural networks and the control is unstable for recurrent networks with chaotic dynamics, because a recursive calculation of the gradient collection might diverge due to the chaotic instability.

Hirasawa *et al.* proposed a combination method where a random optimization method is incorporated into a gradient method[9]. This method can control the exponent without the target time series and it also realizes chaotic dynamics, but it is still computationally expensive more than that of the simple gradient method due to the random optimization.

In this paper, we propose another method in order to reduce the computational cost and realize a "stable" control for recurrent networks with chaotic dynamics. First, we derive a novel stochastic relation between the dynamic complexity λ and parameters of the network configuration under a restriction. The new method is based on the stochastic relation that allows us to approximate the gradient collection in a fashion without time evolution. Simulation results show that this approximation is a robust formulation for the network size and that the new method can control the exponent stably for recurrent networks with chaotic dynamics.

Manuscript received: May 2, 2001, Accepted: Jan. 31, 2002.

Masao Sakai, Kenichi Abe : Dept. of Electric and Communication Eng., Graduate School of Eng., Tohoku University, Aoba 05, Aramaki, Aoba-ku Sendai, 980-8579, Japan.
({sakai, abe}@abe.ecei.tohoku.ac.jp)
Noriyasu Homma : Dept. of Radiological Tech., College of Medical Sciences, Tohoku University, Aoba 05, Aramaki, Aoba-ku Sendai, 980-8579, Japan.(homma@rad. cms.tohoku.ac.jp)

II. Complexity of recurrent networks

A subset of the Lyapunov exponents is used as a measure of the dynamic complexity. To calculate the Lyapunov exponents of a time-discrete dynamic system in an N-dimensional phase space, we monitor the long-term evolution of an infinitesimal N-sphere of initial conditions; the sphere will become an N-ellipsoid due to the locally deforming nature of the evolution. Letting t be discrete time, $t = 1, 2, \cdots$, the system is given by the following difference equations.

$$\begin{aligned} \boldsymbol{x}(t+1) &= \mathbf{f}(\boldsymbol{x}(t)), & (1)\\ \boldsymbol{x}(t) &= [x_1(t)\, x_2(t) \,\cdots\, x_N(t)]^\top, & (2)\\ \mathbf{f}(\boldsymbol{x}(t)) &= [\mathrm{f}_1(\boldsymbol{x}(t))\, \mathrm{f}_2(\boldsymbol{x}(t)) \,\cdots\, \mathrm{f}_N(\boldsymbol{x}(t))]^\top. & (3) \end{aligned}$$

Letting $\mathbf{Df}(\boldsymbol{x}(t))$ be the Jacobi matrix at $\boldsymbol{x}(t)$, the exponents λ_i are then calculated by the Gram-Schmidt method as follows[10],[11].

$$\lambda_i = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \log \|\boldsymbol{L}^i(t)\|, \quad i = 1, 2, \cdots, n, \quad (4)$$

where,

$$\boldsymbol{L}^i(t) = \begin{cases} \hat{\boldsymbol{L}}^i(t), & (i = 1),\\ \hat{\boldsymbol{L}}^i(t) - \sum_{j=1}^{i-1} \langle \hat{\boldsymbol{L}}^i(t), \delta\boldsymbol{L}^j(t)\rangle\, \delta\boldsymbol{L}^i(t), & (\text{otherwise}), \end{cases} \quad (5)$$

$$\begin{aligned} \delta\boldsymbol{L}^i(t) &= \boldsymbol{L}^i(t)/\|\boldsymbol{L}^i(t)\|, & (6)\\ \hat{\boldsymbol{L}}(t+1) &= \mathbf{Df}(\boldsymbol{x}(t))\delta\boldsymbol{L}(t), & (7)\\ \boldsymbol{L}(t) &= [\boldsymbol{L}^1(t)\, \boldsymbol{L}^2(t) \,\cdots\, \boldsymbol{L}^N(t)], & (8)\\ \boldsymbol{L}^i(t) &= \left[L_1^i(t)\, L_2^i(t) \,\cdots\, L_N^i(t)\right]^\top. & (9) \end{aligned}$$

Here $\langle \boldsymbol{A}, \boldsymbol{B}\rangle$ is an inner product of $\boldsymbol{A}$ and $\boldsymbol{B}$, and $\|\boldsymbol{L}^i(t+1)\|$ denotes the ith longest length of the ellipsoidal principal axes evolved from the past matrix, $\delta\boldsymbol{L}(t)$, where all the row vectors are normalized and orthogonal each other. Letting the initial matrix $\delta\boldsymbol{L}(0)$ be an unit matrix, the exponents λ_i can be calculated by using the Jacobi matrix $\mathbf{Df}(\boldsymbol{x}(t))$.

The complexity is defined strictly by using the complete set of the exponents. The calculation of the complete set is, however, computationally expensive. In the following, only the largest Lyapunov exponent will be concerned since it can be decided whether the systems are chaotic or not by using the largest exponent: if a system is chaotic then the largest exponent is greater than 0, otherwise the exponent is less than 0. Then the largest Lyapunov exponent $\lambda(\equiv \lambda_1)$ is defined by

$$\lambda = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \log L(t+1), \quad (10)$$

where $L(t+1)$ denotes $\|\boldsymbol{L}^1(t+1)\|$.

Fully connected recurrent networks composed of N units are considered. Activate functions of neurons are sigmoid. The outputs of neurons $x_i, i = 1, 2, \cdots, N$ are governed by the following difference equations

$$\begin{aligned} x_i(t+1) &= \frac{1}{1+\exp(-\alpha s_i(t+1))}, & (11)\\ s_i(t+1) &= \sum_{j=1}^{N} w_{ij} x_j(t) + \theta_i, & (12) \end{aligned}$$

where s_i are inputs, w_{ij} are connection strengths, α is a gain coefficient of the sigmoid functions and θ_i are biases. In this case, elements of the vector $\boldsymbol{L}^1(t+1)$ are given by[4]

$$L_i(t+1) = \alpha X_i(t+1) \sum_{j=1}^{N} w_{ij} \delta L_j(t), \quad (13)$$

where $X_i(t) \equiv x_i(t)(1 - x_i(t))$, $L_i(t+1) \equiv L_i^1(t+1)$ and $\delta L_i(t) \equiv \delta L_i^1(t)$. This implies we can calculate the largest Lyapunov exponent λ using time series of the networks, $\boldsymbol{x}(0), \boldsymbol{x}(1), \cdots, \boldsymbol{x}(T)$ and the weights w_{ij}.

III. Control of the complexity

1. Gradient method

How do we set the Lyapunov exponent of the networks to the desired value? As mentioned in Section II, the exponent λ is a function of the network parameters such as w_{ij}. That is, the question is equivalent to how we design the parameters which generate dynamics with the desired exponent. One of the methods to achieve this design is a gradient method. In the following, the gradient method is described briefly.

Letting λ^{obj} be the desired exponent, the change of parameter δw_{ij} is given by

$$\begin{aligned} \delta w_{ij} &= -\eta \frac{\partial e_\lambda}{\partial w_{ij}},\\ &= -\eta(\lambda - \lambda^{obj}) \frac{\partial \lambda}{\partial w_{ij}}, \quad (14) \end{aligned}$$

where $e_\lambda = (\lambda - \lambda^{obj})^2/2$ is a squared error and η is a positive coefficient. If the gradient $\partial\lambda/\partial w_{ij}$ is obtained, the change of parameter δw_{ij} is able to be calculated.

The control method of λ presented in [4] needs to calculate $\partial\lambda/\partial w_{ij}$ with a gradient collection through time. The number of dominant multiplications using this collection given by a recursive calculation grows about as $N^5 T$, that is $O(N^5 T)$ run time[8],[12]. Additionally, it makes the control be unstable for networks with chaotic dynamics[4].

2. Approximation method

To solve above problems, a qualitative method based on an approximate relation between the complexity and a parameter of the network configuration has been proposed[13]. The method is as follows: At first we give a restriction with respect to the network configuration which allows us to introduce a key parameter $\overline{\sigma^2}$. We initialize w_{ij} randomly, then define the biases θ_i by

$$\theta_i = -\frac{1}{2} \sum_j w_{ij}, \quad i = 1, 2, \cdots, N. \quad (15)$$

By this restriction, the inputs are given by $s_i = \sum_j s_{ij}$, where $s_{ij} \doteq w_{ij}(x_j - 1/2)$. Supposing that x_j and w_{ij} are independent of each other and that when networks dynamics are chaotic the outputs x_j are uniformly random numbers between 0 to 1 [13], and thus the probability density function is given as

$$p_0(x_j) = \begin{cases} 1, & (0 \le x_j \le 1),\\ 0, & (\text{otherwise}). \end{cases} \quad (16)$$

Then the expectation and variance of s_{ij} are given by

$$\begin{aligned} E[s_{ij}] &= E[w_{ij}] \cdot E[x_j - 1/2], \\ &= 0, \end{aligned} \quad (17)$$

$$\begin{aligned} \sigma^2(s_{ij}) &= E[w_{ij}^2] \cdot E[(x_j - 1/2)^2], \\ &= E[w_{ij}^2] \cdot \int_{-\infty}^{\infty} \left(x_j - \frac{1}{2}\right)^2 \cdot p_0(x_j)\, dx_j, \\ &= E[w_{ij}^2] \cdot \int_0^1 \left(x_j - \frac{1}{2}\right)^2 dx_j, \\ &= \frac{E[w_{ij}^2]}{12}, \end{aligned} \quad (18)$$

where $E[z]$ and $\sigma^2(z)$ denote the expectation and variance of a variable z, respectively. Here $E[w_{ij}^2]$ is equal to the average for N being large enough, i.e, $E[w_{ij}^2] = \sum_{i,j} w_{ij}^2/N^2$. Therefore by the law of large numbers, a variance $\overline{\sigma^2}$ of the inputs s_i is calculated by[13]

$$\begin{aligned} \overline{\sigma^2} &= N \cdot \sigma^2(s_{ij}), \\ &= \frac{1}{12N} \sum_{i,j} w_{ij}^2. \end{aligned} \quad (19)$$

From above arrangements, we have

$$\frac{\partial \lambda}{\partial w_{ij}} = \frac{\partial \lambda}{\partial \overline{\sigma^2}} \cdot \frac{\partial \overline{\sigma^2}}{\partial w_{ij}}. \quad (20)$$

$\partial \overline{\sigma^2}/\partial w_{ij}$ is calculated by (19) as

$$\frac{\partial \overline{\sigma^2}}{\partial w_{ij}} = \frac{1}{6N} w_{ij}. \quad (21)$$

$\partial \lambda / \partial \overline{\sigma^2}$ is calculated by a qualitative relation between λ and $\overline{\sigma^2}$ as[13]

$$\frac{\partial \lambda}{\partial \overline{\sigma^2}} \approx \begin{cases} A_1 A_2 \exp(-A_2 \overline{\sigma^2}), & (\lambda < 0), \\ B_1, & (\lambda > 0). \end{cases} \quad (22)$$

The approximate gradient $\partial \lambda / \partial w_{ij}$ is then calculated by (20) $\sim$ (22) where A_1, A_2, B_1 are positive constants defined experimentally by a parameter fitting method. The method is practical for large-scale networks since the method requires only $O(N^2)$ run time to control the exponent λ [4],[8]. However the method was based on experimental results rather than a theoretical ground, thus the control isn't always stable.

3. Stochastic analysis

In this paper we propose a new approximation method by analyzing the relation stochastically. The main point of the following analysis is to approximate the collection through time of the length $\log L(t+1)$ in (10) by a fashion without time evolution since the collection through time for chaotic systems results in computational divergence due to the chaotic instability. We try to get the average length $\overline{\log L}$ of the $\log L(t+1)$ through time. If $\overline{\log L}$ is obtained, (10) is converted to the following:

$$\lambda = \overline{\log L}. \quad (23)$$

Our strategy to get $\overline{\log L}$ is to calculate the expectation of $L_i(t+1)$ as a time-independent function of the parameter $\overline{\sigma^2}$.

At first, for simplicity, suppose that all arguments of λ such as $x_i(t)$, w_{ij} and $\delta L_j(t)$, $i, j = 1, 2, \cdots, N$, $t = 1, 2, \cdots, T$ are independent of each other. For large-scale recurrent networks with the above supposition, $\log L(t+1)$ in (10) can be calculated by (13) and the law of large numbers as

$$\begin{aligned} \log L(t+1) &= \log \sqrt{\sum_{i=1}^{N} (L_i(t+1))^2}, \\ &= \frac{1}{2} \log \left(N \cdot \frac{\sum_{i=1}^{N} (L_i(t+1))^2}{N} \right), \\ &= \frac{1}{2} \left\{ \log N + \log E\left[\{L_i(t+1)\}^2\right] \right\}, \\ &= \frac{1}{2} \left\{ \log N + \log \alpha^2 + \log E\left[\{X_i(t+1)\}^2\right] \right. \\ &\quad \left. + \log E\left[\left\{\sum_{j=1}^{N} w_{ij} \delta L_j(t)\right\}^2\right] \right\}. \end{aligned} \quad (24)$$

Letting w_{ij} be an uniformly random number whose expected value is zero, $E\left[\left\{\sum_{j=1}^{N} w_{ij} \delta L_j(t)\right\}^2\right]$ is calculated by the law of large numbers as

$$\begin{aligned} E\left[\left\{\sum_{j=1}^{N} w_{ij} \delta L_j(t)\right\}^2\right] &\approx E\left[\sum_{j=1}^{N} \{w_{ij} \delta L_j(t)\}^2\right], \\ &\approx N \cdot E\left[w_{ij}^2\right] \cdot E\left[\{\delta L_j(t)\}^2\right]. \end{aligned} \quad (25)$$

$E\left[w_{ij}^2\right]$ is calculated by (19) as

$$E\left[w_{ij}^2\right] = \frac{12}{N} \overline{\sigma^2}. \quad (26)$$

$E\left[\{\delta L_j(t)\}^2\right]$ is calculated by the law of large numbers and (6) as

$$E\left[\{\delta L_j(t)\}^2\right] = \frac{1}{N}. \quad (27)$$

Here, if $E\left[\{X_i(t+1)\}^2\right]$ is calculated as a time-independent function, λ is approximated by (23) $\sim$ (27) as

$$\lambda \approx \frac{1}{2} \log \left(12 \alpha^2 \overline{\sigma^2} E\left[\{X_i(t+1)\}^2\right] \right). \quad (28)$$

In the following, we try to calculate $E\left[\{X_i(t+1)\}^2\right]$ as a time-independent function. The output x_i of the sigmoid function in (11) can be represented by a power series as

$$\begin{aligned} x_i &= \frac{1}{1 + \exp(-\alpha s_i)}, \\ &= \sum_{n=0}^{\infty} F_1(n) \cdot (\alpha s_i)^n. \end{aligned} \quad (29)$$

$\exp(-\alpha s_i)$ is also calculated by a power series as

$$\exp(-\alpha s_i) = \sum_{k=0}^{\infty} \frac{(-1)^k}{k!} \cdot (\alpha s_i)^k. \quad (30)$$

ThenC$1 + \exp(-\alpha s_i)$ is defined as

$$1 + \exp(-\alpha s_i) = \sum_{k=0}^{\infty} G(k) \cdot (\alpha s_i)^k, \quad (31)$$

$$G(k) = \begin{cases} 2, & (k = 0), \\ \frac{(-1)^k}{k!}, & (\text{otherwise}). \end{cases} \quad (32)$$

From (29) and (31),

$$1 = \left\{\sum_{k=0}^{\infty} G(k)\cdot(\alpha s_i)^k\right\}\cdot\left\{\sum_{n=0}^{\infty} F_1(n)\cdot(\alpha s_i)^n\right\},$$
$$= \sum_{n=0}^{\infty}\sum_{k=0}^{n} G(k)\cdot F_1(n-k)\cdot(-\alpha s_i)^n. \quad (33)$$

Thus,

$$\sum_{n=0}^{\infty}\sum_{k=0}^{n} G(k)\cdot F_1(n-k) = \begin{cases} 1, & (n=0),\\ 0, & (\text{otherwise}).\end{cases} \quad (34)$$

ThereforeC$F_1(0), F_1(1), \cdots, F_1(n)$ are calculated by (32) and (34) as follows.

$$F_1(0) = \frac{1}{G(0)}, = \frac{1}{2}, \quad (35)$$

$$F_1(n) = -\frac{1}{G(0)}\sum_{k=1}^{n} G(k)\cdot F_1(n-k), = \frac{1}{2}\sum_{k=1}^{n}\frac{(-1)^{k+1}}{k!}\cdot F_1(n-k),\ (n \geq 1). \quad (36)$$

From (29), (35) and (36), x_i^m, $m = 2, 3, \cdots$, are given by

$$x_i^m = \sum_{n=0}^{\infty} F_m(n)\cdot(\alpha s_i)^n, \quad (37)$$

$$F_m(n) \equiv \sum_{k=0}^{n} F_1(k)\cdot F_{m-1}(n-k). \quad (38)$$

Thus $\{X_i(t+1)\}^2$ can be approximated on a power series representation as

$$\{X_i(t+1)\}^2 \approx \begin{cases} \sum_{n=0}^{M_1} H(n)\cdot(\alpha s_i)^n, & (|s_i| < \epsilon/\alpha),\\ 0, & (\text{otherwise}),\end{cases} \quad (39)$$

$$H(n) \equiv \{F_2(n) - 2F_3(n) + F_4(n)\}, \quad (40)$$

where M_1 is a suitable natural number and ϵ is a positive constant. The probability density $g(s_i, 0, \overline{\sigma^2})$ of the expectation of the input s_i can be given on a power series representation as follows since the probability density is defined as a normal distribution [13],[14]

$$g(s_i, 0, \overline{\sigma^2}) = \frac{1}{\sqrt{2\pi\overline{\sigma^2}}}\exp\left(-\frac{s_i^2}{2\overline{\sigma^2}}\right),$$

$$\approx \begin{cases} \sum_{n=0}^{M_2} R(n)\left(\frac{1}{\overline{\sigma^2}}\right)^{1/2}\left(\frac{s_i^2}{\overline{\sigma^2}}\right)^n, & \left(|s_i| < \beta\sqrt{\overline{\sigma^2}}\right),\\ 0, & (\text{otherwise}),\end{cases} \quad (41)$$

$$R(n) \equiv \frac{(-1)^n}{\sqrt{2\pi}\cdot 2^n\cdot n!}, \quad (42)$$

where M_2 is a suitable natural number and β is a positive constant. Supposing $M_1 = 2M_2$ to avoid much more complicated

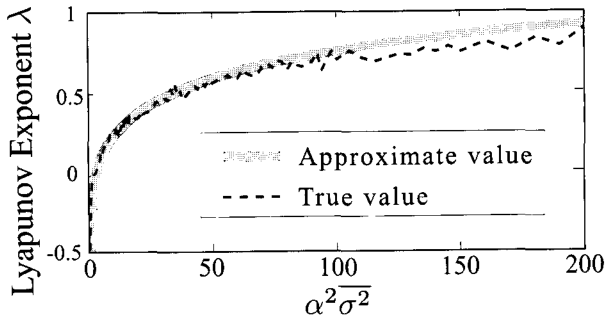


Fig. 1. The Lyapunov exponent λ as a function of the parameter $\alpha^2\overline{\sigma^2}$ ($N = 100$).

representation, $E\left[\{X_i(t+1)\}^2\right]$ can be given on a power series representation as

$$E\left[\{X_i(t+1)\}^2\right] = \int_{-\infty}^{\infty}\{X_i(t+1)\}^2\cdot g(s_i, 0, \overline{\sigma^2})\,ds_i,$$
$$\approx \sum_{\substack{0\leq n\leq M_2\\ 0\leq k\leq n}} Q(n,k)\cdot\left(\alpha^2\overline{\sigma^2}\right)^k\cdot\left(\frac{\tau^2}{\overline{\sigma^2}}\right)^{n+\frac{1}{2}}$$
$$+ \sum_{\substack{M_2+1\leq n\leq 2M_2\\ n-M_2\leq k\leq M_2}} Q(n,k)\cdot\left(\alpha^2\overline{\sigma^2}\right)^k\cdot\left(\frac{\tau^2}{\overline{\sigma^2}}\right)^{n+\frac{1}{2}}, \quad (43)$$

$$Q(n,k) \equiv \gamma\cdot\frac{2\cdot H(2k)\cdot R(n-k)}{(2n+1)}, \quad (44)$$

$$\tau \equiv \min\left(\epsilon/\alpha,\ \beta\sqrt{\overline{\sigma^2}}\right), \quad (45)$$

where γ is a positive constant. Finally from (28) and (43) we get the expectation of λ as a function of $\alpha^2\overline{\sigma^2}$.

Fig. 1 shows the approximate relation between λ and $\alpha^2\overline{\sigma^2}$. In this case, the networks are composed of 100 neurons, $\alpha = 10$ and $T = 100$. Note that our new approximation method can predict the approximate relation between λ and $\alpha^2\overline{\sigma^2}$ if $\alpha^2\overline{\sigma^2}$ is not greater than 100 roughly. Here the constants were decided experimentally such as $M_2 = 5$, $\epsilon = 2.4$, $\beta = 2$ and $\gamma = 1.2$.

Furthermore, Fig. 2 shows similar simulation results for various network sizes. Note that the approximate relation and suitable values of the constants such as M_2, ϵ, β and γ are independent of the scale of the network, if the networks are composed of large number of neurons enough to approximate a stochastic relation.

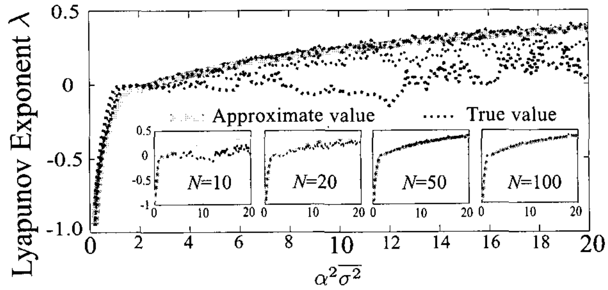


Fig. 2. The Lyapunov exponent λ as functions of the parameter $\alpha^2\overline{\sigma^2}$ for various network sizes ($N = 10, 20, 50, 100$).

4. Proposed complexity control method

From above results, $\partial\lambda/\partial\overline{\sigma^2}$ is calculated by (28) as

$$\frac{\partial\lambda}{\partial\overline{\sigma^2}} = \frac{1}{2}\left\{\frac{1}{\overline{\sigma^2}} + \frac{1}{E\left[\{X_i(t+1)\}^2\right]}\cdot\frac{\partial E\left[\{X_i(t+1)\}^2\right]}{\partial\overline{\sigma^2}}\right\}, \tag{46}$$

where $\partial E\left[\{X_i(t+1)\}^2\right]/\partial\overline{\sigma^2}$ can be calculated by (43). Thus the partial differential coefficients $\partial\lambda/\partial w_{ij}$ without the collection through time is calculated by substituting (46) and (21) into (20). Note that this also needs only $O(N^2)$ run time as same as the experimental approximation[8].

IV. Simulation results

Our control methods have been tested on a design task which requires the fully connected networks to have a desired value of the largest Lyapunov exponent. In this task, the connection weights of the networks were initialized randomly, then changed by our methods.

Fig. 3 shows the exponent λ of the networks with 20 neurons is controlled to the value of the chaotic dynamics by our methods, where $\lambda^{obj} = 0.2$ and the all other parameters, implying α, T, M_2, ϵ, β and γ were same values as the above simulation shown in Fig. 1. Dynamics of the networks after 100 iterations is shown in Fig. 4. Note that our new approximation method can control the exponent λ to the desired value and realize chaotic dynamics. On the other hand, the conventional method fails to control the exponent λ around $\lambda \approx 0$ because of the chaotic instability. Then the network dynamics become non-chaotic behavior such as a single point attractor. In this case our new approximation method requires only 0.02 seconds' cpu-time to calculate the partial differential coefficients $\partial\lambda/\partial w_{ij}$ while the conventional one requires 692 seconds' cpu-time. This implies the conventional method isn't practical for large-scale networks in comparison with our new approximation method. Fig. 5 shows similar simulation results by our new approximation method for large-scale networks. The networks were composed of 100 neurons and $\lambda^{obj} \in \{-1,\ 0,\ 0.5,\ 1\}$. Note that the exponent λ converges to desired values. In this case the cpu-time is 0.06 seconds. According to the experimental comparison, our new approximation method is practical for large-scale networks.

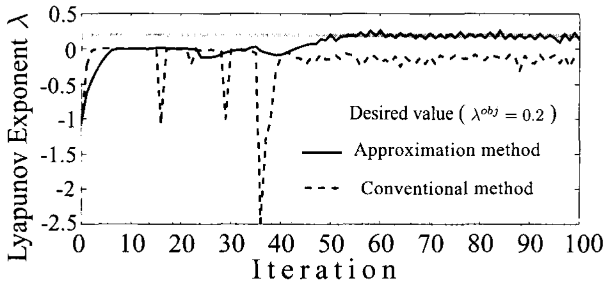


Fig. 3. The exponent λ as functions of the iteration by a conventional method and the proposed one.

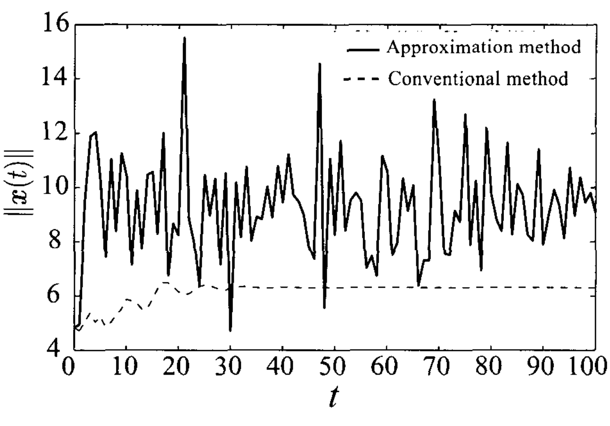


Fig. 4. Dynamics of the networks trained by a conventional method and the proposed one after 100 learning iterations.

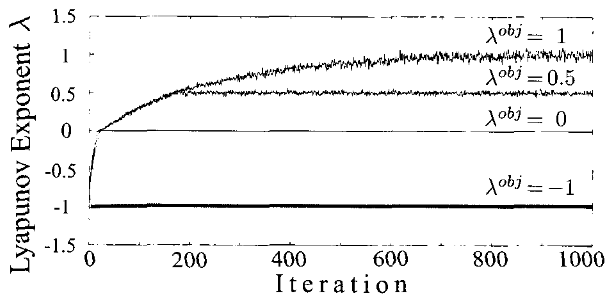


Fig. 5. The exponent λ as a function of the iteration by the proposed method for various desired values ($\lambda^{obj} = -1, 0, 0.5, 1$).

V. Conclusions

In this paper we analyzed a stochastic relation between the largest Lyapunov exponent and network parameters for large-scale fully connected recurrent networks with asymmetric connection weights w_{ij} and restricted biases $\theta_i = -\frac{1}{2}\sum_j w_{ij}$. The stochastic relation allows us to get the gradient collection through time in a fashion without time evolution.

Simulation results show that effectiveness of proposed method with respect to the computational cost and the stable control of the Lyapunov exponent of recurrent networks with chaotic dynamics.

References

[1] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Networks*, vol. 1, no. 1, pp. 4-27, 1990.

[2] M. Jordan, "Generic constraints on underspecified target trajectories," *Proc. International Joint Conference on Neural Networks(IJCNN-89)*, vol. 1, pp. 217-225, 1989.

[3] B. Irie and S. Miyake, "Capabilities of three-layered perceptrons," *Proc. of IEEE ICNN 88*, pp. I-641, 1988.

[4] N. Homma, M. Sakai, K. Abe and H. Takeda, "Complexity control methods of dynamics in recurrent neural Networks(in Japanese)," *Trans. SICE*, vol. 35, no. 1, pp. 138-143, 1999.

[5] J. C. Principe, A. Rathie and J. Kuo, "Prediction of chaotic time series with neural networks and the issue of dynamic modeling," *International Journal of Bifurcation and Chaos*, vol. 2, no. 4, pp. 989-996, 1992.

[6] J. C. Principe and J. Kuo, "Dynamic modelling chaotic time series with neural networks," In G. Tesauro et al.(Eds.), *Neural Information Processing System 7*, pp. 311-318, 1995.

[7] G. Deco and B. Schürmann, "Dynamic modelling chaotic time series," *Computational learning theory and neural learning systems*, vol. 4 of Making learning Systems Practical, MIT Press, Cambridge, Mass., chapter 9, pp. 137-153, 1997.

[8] N. Homma, M. Sakai, K. Abe and H. Takeda, "Control method of the Lyapunov exponents for recurrent neural networks," *Proc. of 14th IFAC World Congress*, Beijing, vol. K, pp. 51-56, 1999.

[9] K. Hirasawa, X. Wang, J. Murata, J. Hu and C. Jin, "Universal learning networks and its application to chaos control," *Neural Networks*, vol. 13, pp. 239-253, 2000.

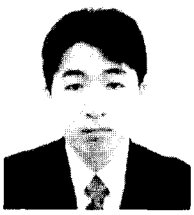
[10] A. Wolf, J. B. Swift, H. L. Swinney and J. A. Vastano, "Determining Lyapunov exponents from a time series," *Physica*, **16D**, pp. 285-315, 1989.

[11] A. Bjorck, "Solving linear least squares problems by Gram-Schmidt orthogonalization," *BIT*, vol. 7, pp. 1-21, 1967.

[12] Y. Nishikawa and S. Kitamura, *Neural networks as applies to measurement and control(in Japanese)*, Asakura Shoten Publishing Co., Ltd., Tokyo, 1995.

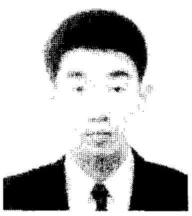
[13] N. Homma, K. Kitagawa and K. Abe, "Effect of complexity on learning ability of recurrent neural networks," *Proc. of Artificial Life and Robotics*, vol. 2, no. 2, pp. 97-101. 1998.

[14] M .Ueda, Y .Okada and Y .Yoshitani, *Probability and statistics(in Japanese)*, Dainippon Tosho Publishing Co.,Ltd., Tokyo, 1980.



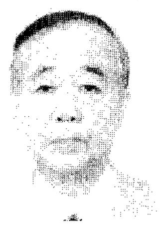
Masao Sakai

He received the B.S. degree in electronic engineering and M.S. degree in electrical engineering from Tohoku University, Japan, in 1997 and 1999, respectively. He is currently a graduate student in the doctoral course at the Tohoku University. His research interests are monitoring, analyzing and designing of dynamics of artificial neural networks. He is a member of IEEE and SICE.



Noriyasu Homma

He received the B.S., M.S., and Ph. D. degrees in electrical engineering from Tohoku University, Japan, in 1990, 1992, and 1995, respectively. He is currently an associate professor of the Tohoku University and is a visiting professor at the Intelligent Systems Research Laboratory, college of Engineering, University of Saskatchewan, Canada. His research interests include neural networks, intelligent system, complex adaptive system and chaotic dynamics. He is a member of IEEE, SICE and JSRT.



Kenichi Abe

He received the B.S., M.S., and Ph. D. degrees in electrical engineering from Tohoku University, Japan, in 1964, 1966, and 1969, respectively. He is currently a professor in the Department of Electrical Engineering, Graduate School of Engineering, Tohoku University. His research interests are learning control systems, monitoring and control of artificial heart, neural networks, fuel control of automotive engines, and analysis and design of multivariable control systems. He is a member of IEEE, SICE, IEE of Japan and JSMBE.