

Exploiting Heterogeneous Units for Reservoir Computing with Simple Architecture

Gouhei Tanaka¹(✉), Ryosho Nakane¹, Toshiyuki Yamane², Daiju Nakano²,
Seiji Takeda², Shigeru Nakagawa², and Akira Hirose¹

¹ Graduate School of Engineering, The University of Tokyo, Tokyo 113-8656, Japan
gouhei@sat.t.u-tokyo.ac.jp, nakane@cryst.t.u-tokyo.ac.jp,
ahirose@ee.t.u-toyo.ac.jp

² IBM Research - Tokyo, Kawasaki, Kanagawa 212-0032, Japan
{tyamane,dnakano,seijitkd,snakagw}@jp.ibm.com

Abstract. Reservoir computing is a computational framework suited for sequential data processing, consisting of a reservoir part and a read-out part. Not only theoretical and numerical studies on reservoir computing but also its implementation with physical devices have attracted much attention. In most studies, the reservoir part is constructed with identical units. However, a variability of physical units is inevitable, particularly when implemented with nano/micro devices. Here we numerically examine the effect of variability of reservoir units on computational performance. We show that the heterogeneity in reservoir units can be beneficial in reducing the prediction error in the reservoir computing system with a simple cycle reservoir.

Keywords: Reservoir computing · Sequential data processing · Simple cycle reservoir · Heterogeneous neurons · Energy efficiency

1 Introduction

Recurrent neural networks are capable of producing high-dimensional complex dynamics due to feedback connections, which has often been utilized for information processing of sequential data [1]. The training methods for recurrent neural networks have been proposed, including the backpropagation through time algorithm, the real-time recurrent learning, and the extended Kalman filter method [2]. These algorithms try to adapt all the connection weights by minimizing the total error between the network output sequence and the desired output sequence. Since they have relatively high time complexity, their practical applications with large-scale networks are still not realized. Reservoir computing is one of the potent frameworks that can overcome the problem of the training cost in recurrent neural networks for energy efficient computing [3, 4]. The reservoir computing framework was established by combining the concepts from the echo state network (ESN) [5–7] and the liquid state machine [8].

The reservoir computing system consists of the reservoir part and the readout part. The reservoir part is used for mapping the input sequence to a high-dimensional spatiotemporal pattern. The readout part is used for adjusting the output connection weights so that the spatiotemporal pattern generated by the reservoir is appropriately mapped to the desired output sequence. Since not all the weights but only the output weights are adapted, the reservoir computing can save the learning time compared with the conventional recurrent neural networks. Moreover, the fixed reservoir can be implemented with nonlinear physical systems and devices, including optoelectronics [9], memristors [10], and wave phenomena [11, 12].

In the standard ESN [5], the reservoir is given as a randomly connected recurrent neural network. The performance of reservoir computing relies on the number of neurons and the weight matrix in the reservoir, which govern the length of the history of input sequence that can be embedded into its spatiotemporal dynamics. For constructing a good mapping from an input sequential data to an output one, the reservoir is required to satisfy the echo state property [5] which indicates the property that the influence of the input stream is gradually attenuated with time. This means that the mapping represented by the reservoir should be neither expanding nor highly contracting. Hence, the spectral radius of the weight matrix is often set to be less than and close to unity, corresponding to the edge of chaos [2]. However, the random reservoir topology is not mandatory. A deterministically designed reservoir with simple ring architecture is comparable to the standard random reservoir in their computational performance [13]. The simple cycle reservoir enables theoretical analyses of reservoir computing properties such as memory capacity. In addition, it is favorable for hardware implementation because only local connections and uniform weights are needed.

In this study, we incorporate variability into the neuron units in the simple cycle reservoir, motivated by two aspects. One is that the simple cycle reservoir with identical units seems to be too uniform to produce rich nonlinear dynamics. The unit variability is expected to diversify the dynamics of individual units. The other is that the variability of the reservoir units are inevitable when they are implemented with physical devices, particularly with nano/micro devices. We examine how heterogeneity of the reservoir units impacts on the reservoir dynamics and its computational capability. We show that the variability in the reservoir units can improve the performance of the simple cycle reservoir.

2 Methods

2.1 Model

The reservoir in the standard ESN consists of neuron units which interact with each other through weighted random connections as illustrated in Fig. 1(a). The numbers of input units, internal units, and output units are denoted by L , N , M , respectively. Then, the states of the input, internal, and output units are represented by the column vectors $\mathbf{u}(t) = (u_1(t), u_2(t), \dots, u_L(t))^T$,

$\mathbf{x}(t) = (x_1(t), x_2(t), \dots, x_N(t))^T$, and $\mathbf{y}(t) = (y_1(t), y_2(t), \dots, y_M(t))^T$, respectively. The input connectivity, the reservoir connectivity, the feedback connectivity, and the output connectivity are represented by $W^{\text{in}} \in \mathbb{R}^{L \times N}$, $W \in \mathbb{R}^{N \times N}$, an $W^{\text{fb}} \in \mathbb{R}^{M \times N}$, and $W^{\text{out}} \in \mathbb{R}^{N \times M}$, respectively.

The states of the i th internal unit ($i = 1, \dots, N$) is updated as follows:

$$\mathbf{x}_i(t+1) = f_i \left((W_i^{\text{in}})^T \cdot \mathbf{u}(t+1) + (W_i)^T \cdot \mathbf{x}(t) + (W_i^{\text{fb}})^T \cdot \mathbf{y}(t) \right), \quad (1)$$

where f_i stands for the activation function of the i th neuron in the reservoir and W_i^{in} , W_i , and W_i^{fb} are the i th row of the input, reservoir, and feedback weight matrices, respectively. The states of the j th output unit ($j = 1, \dots, M$) is given by

$$\mathbf{y}_j(t) = f^{\text{out}} \left((W_j^{\text{out}})^T \mathbf{x}(t) \right), \quad (2)$$

where f^{out} represents the activation function of the output neurons and W_j^{out} is the j th row of the output weight matrix. Here we use $f^{\text{out}}(x) = \tanh(x)$.

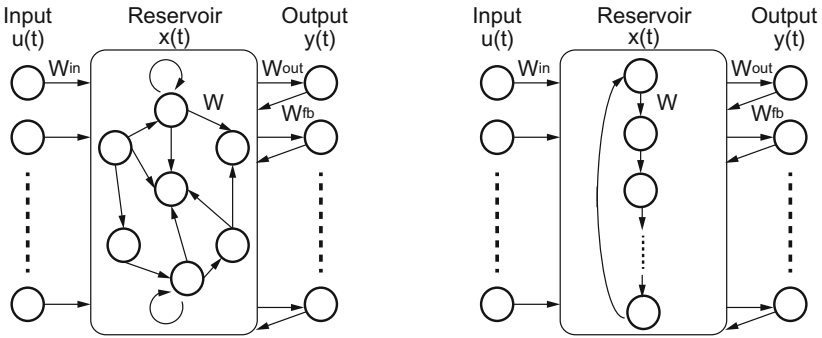


Fig. 1. Structure of the reservoir. (a) Random topology. The standard architecture in the ESN [5]. (b) Ring topology called the simple cycle reservoir [13].

When an input sequential data $\mathbf{u}(t)$ is given, the output sequence is generated by Eqs. (1)–(2). The characteristic of the reservoir computing is that the weights in the input and reservoir connections are not adapted but only the output connection weights W^{out} are determined by a learning rule. The output matrix W^{out} is obtained to minimize the error between the network output sequence $\mathbf{y}(t)$ and a desired output sequence $\mathbf{d}(t)$, given by

$$E = \langle \|\mathbf{y}(t) - \mathbf{d}(t)\|^2 \rangle, \quad (3)$$

where $\langle \cdot \rangle$ denotes an average over a time period. The minimization of E can be achieved using regression methods. Here we employ the pseudoinverse computation [2].

2.2 Reservoir Structure

We use the simple cycle reservoir as shown in Fig. 1(b), where the connectivity of the reservoir nodes has ring topology and the connection weights are uniform [13]. It is represented as the weight matrix $W = (w_{i,j})$ where $w_{i+1,i} = r$, $w_{1,N} = r$, and all the other entries are zero. For the standard reservoir, the necessary condition for the echo state property is given by $\rho(W) < 1$ where $\rho(W)$ is the spectral radius of W and the sufficient condition is given by $\bar{\sigma}(W) < 1$ where $\bar{\sigma}(W)$ is the largest singular value of W [5]. For the simple cycle reservoir [13], $\rho(W) = \bar{\sigma}(W) = r$. The simple cycle reservoir is comparable to the standard reservoir in the performance of time series predictions and its memory capacity can be theoretically derived [13].

2.3 Heterogeneous Units

In most studies on reservoir computing, the units of the reservoir have been assumed to be identical. The hyperbolic tangent function is normally used as the nonlinear activation function of the units in ESNs. In the standard reservoir, the diversity of the dynamics of the reservoir units are brought about by the random weight matrix. However, in the simple cycle reservoir with identical units, the dynamics generated by the individual units become uniform. The total system can be essentially reduced to a lower-dimensional system. This is unbenefited for producing high-dimensional spatiotemporal dynamics. Thus, we introduce the variability in the activation function of the reservoir units, which are represented as follows:

$$f_i(x) = \tanh(\beta_i x), \quad (4)$$

where the parameter β_i , corresponding to the slope of the function at the origin, controls the nonlinearity of the function. Although there are many ways to introduce variability in β_i , for simplicity we assume that β_i is randomly generated from the uniform distribution in the range $[1 - v, 1]$, where v ($0 \leq v \leq 1$) is the control parameter representing the degree of variability.

2.4 Simulation Setting

Initially, we give the internal state $\mathbf{x}(0)$ and the output weight matrix W^{out} randomly. The weights of input connections have the same absolute value p but the signs are randomly assigned. After a washout period with length T_{init} , the sample sequential data with length T_{trn} are used for training the output weights and subsequently the sequential data with length T_{test} are used for testing the generalization ability of the reservoir. The computational performance is evaluated using the normalized mean squared error (NMSE) defined as follows:

$$NMSE = \frac{\langle \|\mathbf{y}(t) - \mathbf{d}(t)\|^2 \rangle}{\langle \mathbf{d}(t)^2 \rangle}. \quad (5)$$

We use the following benchmark tasks on sequential data processing, which have been widely used to test the performance of reservoir computing.

- (1)
- Mackey-Glass equation*
- [14]:

$$\frac{dy(t)}{dt} = \frac{ay(t - \tau)}{1 + y(t - \tau)^{10}} - by(t), \quad (6)$$

where $a = 0.2$, $b = 0.1$, and $\tau = 30$. The dataset was generated by numerically solving this equation with time step $\Delta t = 1$ [15]. The task is to predict the value of $y(t + 1)$ from the past values up to time t .

- (2) *Laser dataset*: The Santa Fe Laser dataset is a crosscut through periodic to chaotic intensity pulsations of a real laser [16]. The task is the same as that in the previous one. The simple cycle reservoir has been applied to this task [13].
- (3) *NARMA 10th-order system*: The nonlinear auto-regressive moving average (NARMA) system of order 10 is described as follows:

$$y(t + 1) = 0.3y(t) + 0.05y(t) \sum_{i=0}^9 y(t - i) + 1.5u(t - 9)u(t) + 0.1, \quad (7)$$

where $u(t)$ is the input sequence which is randomly sampled from the uniform distribution in the range $[0, 0.5]$. This task is widely used in the literature of recurrent neural networks and reservoir computing [7, 13].

- (4) *NARMA 20th-order system*: The NARMA system of order 20 is described as follows [13]:

$$y(t + 1) = \tanh \left(0.3y(t) + 0.05y(t) \sum_{i=0}^{19} y(t - i) + 1.5u(t - 19)u(t) + 0.01 \right), \quad (8)$$

where $u(t)$ is generated as in the previous task. This task is more difficult compared with the NARMA 10th-order system due to the dependence of the current state on the longer history of inputs.

3 Results

In the following numerical experiments, the input and output data were scaled and shifted appropriately for each dataset. For the dataset generated by the Mackey-Glass equation, we set $N = 2$, $T_{\text{init}} = 500$, $T_{\text{trn}} = 1000$, $T_{\text{test}} = 1000$, $p = 0.87$, and $r = 1$. The result of the test performance is shown in Fig. 2(a). The plot for the variability parameter $v = 0$ corresponds to the result for the simple cycle reservoir with the identical units [13]. As the variability parameter v is increased, the NMSE is gradually decreased. Namely, the variability of the units can improve the computational performance.

For the Laser dataset, we set $N = 100$, $T_{\text{init}} = 500$, $T_{\text{trn}} = 2000$, $T_{\text{test}} = 3000$, $p = 0.87$, and $r = 0.7$. The result is shown in Fig. 2(b). As the variability increases, the prediction error decreases and reaches the bottom at around 0.7. The error slightly increases for further increase in v , but it is much lower than the case without variability.

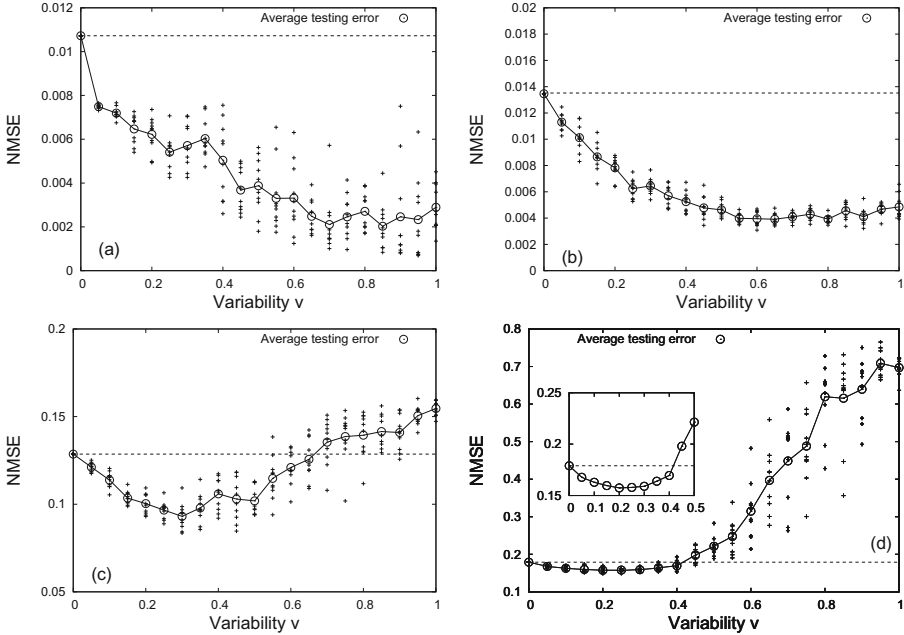


Fig. 2. The performance of the simple cycle reservoir with heterogeneous neurons. The NMSE for the test set is plotted against the variability parameter v . The crosses represent the results of 10 trials for each parameter value. The open circle indicates the average of the 10 trials. (a) Mackey-Glass equation. (b) Santa Fe Laser dataset. (c) NARMA 10th-order system. (d) NARMA 20th-order system.

For the NARMA 10th-order system, we set $N = 100$, $T_{\text{init}} = 200$, $T_{\text{trn}} = 1000$, $T_{\text{test}} = 1000$, $p = 0.87$, and $r = 0.86$. Figure 2(c) shows the result, where the variability can yield a better result if v is less than around 0.5 but for a larger value of v the result is worse than the case without variability.

For the NARMA 20th order system, we set $N = 100$, $T_{\text{init}} = 500$, $T_{\text{trn}} = 1000$, $T_{\text{test}} = 1000$, $p = 0.87$, and $r = 0.95$. The result for the NARMA 20th-order system is similar to that for the NARMA 10-th order system as shown in Fig. 2(d). Although a large value of v significantly increases the prediction error, there exists a range of v in which the variability has a positive effect (the inset).

To clarify the conditions that the performance is improved by the unit variability, we indicated the parameter regions (black) for good computational performance in Fig. 3. The performance increases with the variability v for the range of r in Figs. 3(a), (b), whereas there is a optimal range of v in Figs. 3(c), (d). There is a correlation between the values of r and v , suggesting that the effective spectral radius is determined not only by r but by β_i . It remains to explicitly give the formula for the spectral radius in the simple cycle reservoir with heterogeneous units.

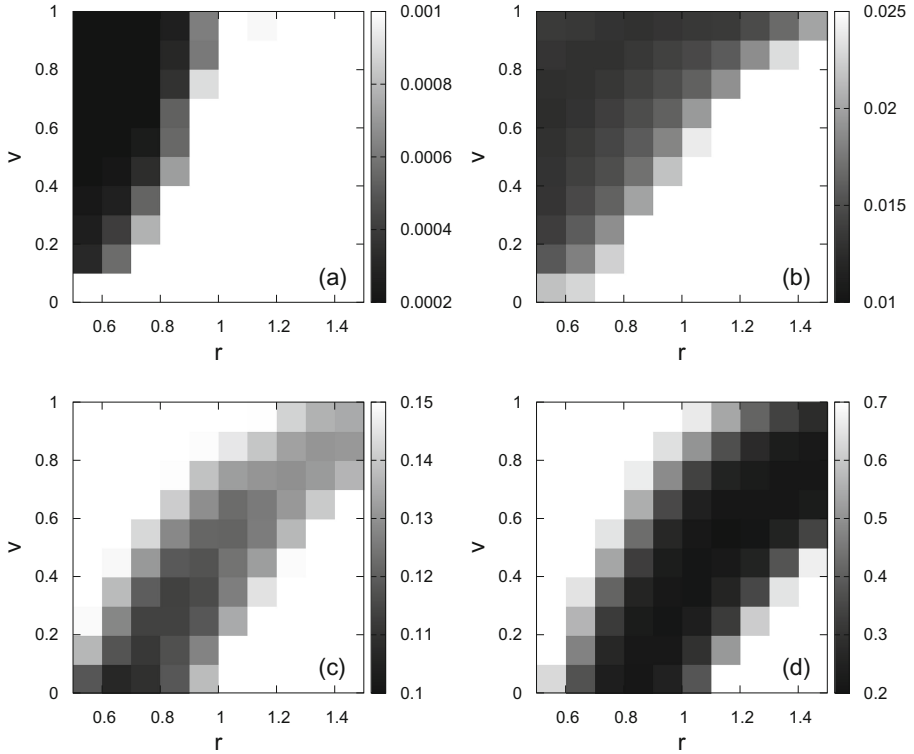


Fig. 3. The parameter region for good prediction performance on the (r, v) -plane. The color bar indicates the NMSE for the test set. (a) The Mackey-Glass equation. (b) The Santa Fe Laser dataset. (c) The NARMA 10th-order system. (d) The NARMA 20th-order system.

4 Conclusions

We have proposed to exploit heterogeneity in the reservoir units for improving the computational performance in the reservoir computing with the simple cycle architecture. We have introduced variability in the slope parameter in the hyperbolic tangent activation functions of the reservoir units. Numerical experiments have shown that both the unit variability and the connection weight govern the performance on the benchmark tasks for sequential information processing.

Our result is beneficial for hardware implementation of reservoir computing because of the simple reservoir structure and the unavoidable unit variability when implemented with nano/micro devices. For verification of the effectiveness of our method, we need further numerical experiments using other datasets. It is significant to clarify the conditions under which the unit variability works well. The mathematical mechanism of the positive role of the unit variability still remains to be investigated.

Acknowledgments. This work was partially supported by JSPS KAKENHI Grant Number 16K00326 (GT).

References

1. Haykin, S.: *Neural Networks. A Comprehensive Foundation*, 2nd edn. Prentice Hall, Englewood Cliffs (1998)
2. Jaeger, H.: Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the “echo state network” approach. GMD-Forschungszentrum Informationstechnik (2002)
3. Schrauwen, B., Verstraeten, D., Van Campenhout, J.: An overview of reservoir computing: theory, applications and implementations. In: *Proceedings of the 15th European Symposium on Artificial Neural Networks*, pp. 471–482 (2007)
4. LukošEvičIus, M., Jaeger, H.: Reservoir computing approaches to recurrent neural network training. *Comput. Sci. Rev.* **3**(3), 127–149 (2009)
5. Jaeger, H.: The “echo state” approach to analysing and training recurrent neural networks-with an erratum note. German National Research Center for Information Technology GMD Technical Report **148**, Bonn, Germany 34 (2001)
6. Jaeger, H.: Short term memory in echo state networks. GMD-Forschungszentrum Informationstechnik (2001)
7. Jaeger, H.: Adaptive nonlinear system identification with echo state networks. In: *Advances in neural information processing systems*, pp. 593–600 (2002)
8. Maass, W., Natschläger, T., Markram, H.: Real-time computing without stable states: a new framework for neural computation based on perturbations. *Neural Comput.* **14**(11), 2531–2560 (2002)
9. Paquot, Y., Duport, F., Smerieri, A., Dambre, J., Schrauwen, B., Haelterman, M., Massar, S.: Optoelectronic reservoir computing. *Scientific Reports* **2**, 287 (2012)
10. Kulkarni, M.S., Teuscher, C.: Memristor-based reservoir computing. In: *2012 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*, pp. 226–232 (2012)
11. Katayama, Y., Yamane, T., Nakano, D., Nakane, R., Tanaka, G.: Wave-based neuromorphic computing framework for brain-like energy efficiency and integration. *IEEE Trans. Nanotechnology* (Accepted)
12. Yamane, T., Katayama, Y., Nakane, R., Tanaka, G., Nakano, D.: Wave-based reservoir computing by synchronization of coupled oscillators. In: Arik, S., Huang, T., Lai, W.K., Liu, Q. (eds.) *Neural Information Processing*, pp. 198–205. Springer, Switzerland (2015)
13. Rodan, A., Tiño, P.: Minimum complexity echo state network. *IEEE Trans. Neural Netw.* **22**(1), 131–144 (2011)
14. Mackey, M.C., Glass, L.: Oscillation and chaos in physiological control systems. *Science* **197**(4300), 287–289 (1977)
15. Wyffels, F., Schrauwen, B., Verstraeten, D., Stroobandt, D.: Band-pass reservoir computing. In: *IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pp. 3204–3209. IEEE (2008)
16. Weigend, A., Gershenfeld, N.: Time series prediction: forecasting the future and understanding the past. In: *Proceedings of a NATO Advanced Research Workshop on Comparative Time Series Analysis*, held in Santa Fe, New Mexico (1994)