DE GRUYTER OPEN

JAISCR, 2018, Vol. 8, No. 3, pp. 237 – 249
10.1515/jaiscr-2018-0015

COMPLEX-VALUED ASSOCIATIVE MEMORIES WITH PROJECTION AND ITERATIVE LEARNING RULES

Teijiro Isokawa[1], Hiroki Yamamoto[1], Haruhiko Nishimura[2],
Takayuki Yumoto[1], Naotake Kamiura[1], Nobuyuki Matsui[1]

[1]*Graduate School of Engineering, University of Hyogo, 2167 Shosha, Himeji, Hyogo, 671-2280 Japan*

[2]*Graduate School of Applied Informatics, University of Hyogo, 7-1-28 Minatojima-Minami-cho, Chuo-ku, Kobe, Hyogo, 650-0047 Japan*

Submitted: 16th October 2017; accepted: 14th October 2017

Abstract

In this paper, we investigate the stability of patterns embedded as the associative memory distributed on the complex-valued Hopfield neural network, in which the neuron states are encoded by the phase values on a unit circle of complex plane. As learning schemes for embedding patterns onto the network, projection rule and iterative learning rule are formally expanded to the complex-valued case. The retrieval of patterns embedded by iterative learning rule is demonstrated and the stability for embedded patterns is quantitatively investigated.

Keywords: complex-valued neural networks, associative memory, projection

1 Introduction

Complex-valued neural networks (CVNNs) are neural networks of which neuronal parameters, such as input, output, and connection weights, are encoded by complex values. Various types of CVNNs have been extensively investigated [1, 2, 3]. One of the advantages in CVNNs is that they can treat two-dimensional signal as a single entity, e.g. amplitude and phase of signals and two-dimensional coordinates. Hence CVNNs can be naturally applied to engineering problems, such as land-mine detection based on radar wave [4] and equalization for communication channels [5].

Complex-valued multistate neural network is a type of CVNNs where the state of a neuron is represented by a distinct point on a unit circle, i.e. a phase value, in a complex plane [6, 7, 8, 9, 10]. This network can be used as an associative memory because some discrete values such as pixel values in an image can be mapped to the phase values and updates for neuron's state can be easily conducted. Several types of multistate networks have been proposed and analyzed for both theoretical and experimental approaches.

It is important to develop learning schemes for embedding patterns onto associative memory. There are several schemes for multistate networks [10, 11, 12, 13]. The Hebbian rule is a basic and straightforward scheme for storing patterns, but it keeps a major limitation for the properties of patterns to be stored; patterns must be orthogonal to each other. For practical uses of associative memory, orthogonality is hardly satisfied among all patterns. Projection rule is an improved scheme from the Hebbian rule by projecting non-orthogonal patterns to orthogonal ones [12, 14, 15]. This learning scheme requires to calculate a pseudo-inverse of matrix and its computational cost becomes rather higher according to the size of each pattern and the number of stored patterns.

Thus an iterative learning rule for complex-valued and quaternionic multistate networks has been proposed and analyzed in [10]. It is an implementation of Projection rule and was firstly proposed for the real-valued Hopfield networks in [16]. By this rule, the connection weights in the network are gradually modified by iterative presentations of stored patterns, instead of the calculation of inverse matrix. The modification of connection weights works so as to enlarge the basin of attractor for the presented pattern in the energy landscape spanned by the connection weights, enabling to the presented pattern being more deeply embedded to the network. Thus this rule could have a flexibility concerning the basins of attractors by controlling the presentation frequencies of stored patterns. For the complex-valued multistate networks, there is only a theoretical analysis in [10] proving that embedding patterns can be successfully conducted by this scheme, but none of experimental results by using real or artificially generated patterns have been found.

This paper explores the stabilities of patterns embedded on the multistate neural networks by the iterative learning rule through comparisons with Projection rule (a short version of this paper was presented in [17]). Embedding and retrieving patterns are demonstrated by using gray-scaled images, and the stabilities of embedded patterns with these rules are evaluated by using the randomly generated patterns with several resolution of a neuron state.

This paper is organized as follows. Section 2 describes the fundamentals of complex-valued multistate network and its learning rules. Experimental results for the stability of embedded patterns are shown in Section 3. This paper concludes in Section 4.

2 Preliminaries

2.1 Complex-valued Multistate Neural Network

We first describe the complex-valued multistate neuron model used in this paper that is also used in [7, 9, 10]. In this model, the state and threshold of a neuron and the connection weights between the neurons are represented by complex values. The output of a neuron is also a complex value, but it is restricted to one of the distinct points on the unit circle in a complex plane. Therefore, the output of this model can only be represented by a phase value. Figure 1 shows an example of output points in a complex plane where the number of phase quantizing delimiter K is 6.

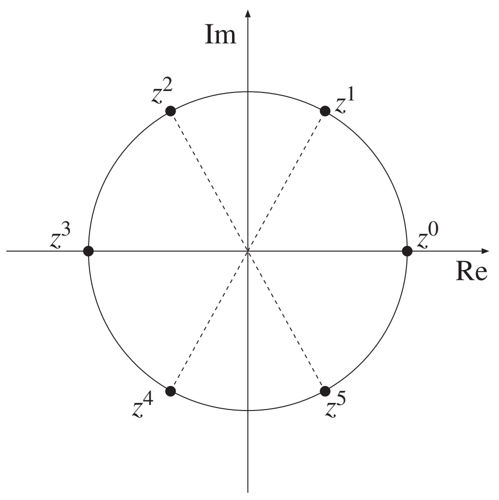


Figure 1. An example of output points in the complex-valued multistate neuron model

We consider a Hopfield neural network with N complex-valued multistate neurons. The action potential $h_p(t)$ of a neuron p at a discrete time t is given by

$$h_p(t) = \sum_{q}^{N} w_{pq} u_q(t), \tag{1}$$

where $u_q(t)$ is the state of the neuron q at a time t, and w_{pq} is the connection weight from neuron q to neuron p. The state of neuron p at $(t+1)$ is determined by

$$u_p(t+1) = csign(h_p(t) \cdot z^{1/2}), \tag{2}$$

where $z^{1/2} = e^{i\varphi_0/2}$ is a fixed threshold value, and $\varphi_0 = 2\pi/K$ defines a quantized unit. The function $csign(\cdot)$ is an activation function of a multistate neuron defined as

$$csign(u) = \begin{cases} z^0 & 0 \le \arg(u) < \varphi_0 \\ z^1 & \varphi_0 \le \arg(u) < 2\varphi_0 \\ \vdots & \\ z^{K-1} & (K-1)\varphi_0 \le \arg(u) < K\varphi_0 \end{cases} . \tag{3}$$

The state of a neuron has K quantized levels, called K-stage phase quantizer.

The procedure for updating neuron's state is illustrated in Figure 2 where $K=6$ is adopted. In this procedure, (1) the action potential $h_p(t)$ is firstly calculated, (2) this action potential is rotated by $z^{1/2}$, and (3) the updated state of a neuron is determined by $csign(\cdot)$ function.

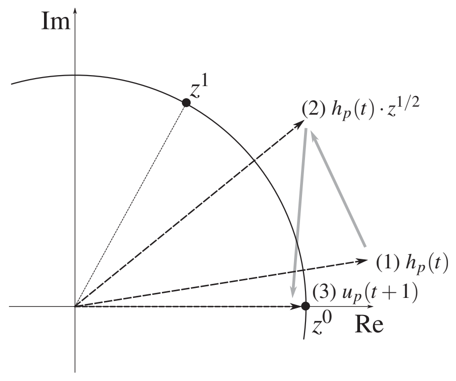


Figure 2. Procedures for calculation of neuron's state

The stability of the network using this model was proved by showing that the energy function of the network monotonically decreased under the condition $|\Delta\varphi| < \varphi_0/2$, where $\Delta\varphi$ is a phase difference between the state at time $(t+1)$ and the action potential at time t for the neuron undergoing its update. The energy function E is given by

$$E = -\frac{1}{2}\sum_p\sum_q w_{pq}u_p^*u_q, \qquad (4)$$

where the connection matrix $W=\{w_{pq}\}$ is a Hermitian matrix ($w_{pq}=w_{qp}^*$) with the condition of $w_{rr}\geq 0$ $(1\leq r\leq N)$. A proof for this stability is shown in Section A of Appendix.

2.2 Condition for Embedding Patterns

In order to introduce the learning schemes into the multistate network, the stability condition for the neurons' states is important. Let $\xi^\mu=\{\xi_p^\mu\}$ $(p=1,\cdots,N;\mu=1,\cdots,n_p)$ be a vector of the μ-th memory pattern, where n_p denotes the number of patterns to be embedded. ξ_p^μ takes one of K integer values ($\xi_p^\mu\in\{0,\cdots,K-1\}$). Therefore, storing a pattern to the network, each ξ_p^μ is mapped onto ε_p^μ, which is a point on the unit circle in a complex plane by

$$\varepsilon_p^\mu = z^{\xi_p^\mu} = e^{i\xi_p^\mu\varphi_0}. \qquad (5)$$

ε_p^μ represents a stable network configuration if

$$u_p(t+1) = u_p(t) = \varepsilon_p^\mu \qquad (6)$$

is satisfied for every neuron p $(p=1,\cdots,N)$. This condition leads to the phase relation

$$\arg(\varepsilon_p^\mu) \leq \arg(h_p(t)\cdot z^{1/2}) < \arg(\varepsilon_p^\mu)+\varphi_0. \qquad (7)$$

According to Eq.(2), this relation can be expressed as

$$|\arg(h_p(t))-\arg(\varepsilon_p^\mu)| < \frac{\varphi_0}{2}. \qquad (8)$$

Also, in order to obtain greater stability of the desired memory patterns, a threshold parameter κ is introduced so that the left-hand side of Eq.(8) should be smaller than κ, that is,

$$|\arg(h_p(t))-\arg(\varepsilon_p^\mu)| < \kappa < \frac{\varphi_0}{2}. \qquad (9)$$

For an appropriate κ, the trainable $\{w_{pq}\}$ in h_p should satisfy this condition.

2.3 Projection Rule

A straightforward way to embed patterns onto the associative memory networks is the use of Hebbian rule. The Hebbian rule is defined as

$$w_{pq} = \frac{1}{N}\sum_{\mu=1}^{n_p}\varepsilon_p^\mu\varepsilon_q^{\mu*}, \qquad (10)$$

where n_p is the number of patterns to be embedded. However, there is a major limitation in the Hebbian rule, i.e., it works only when the patterns $\varepsilon^\mu=\{\varepsilon_1^\mu,\cdots,\varepsilon_N^\mu\}$ satisfy the condition

$$\varepsilon^\mu\cdot\varepsilon^{\nu*}=0 \qquad (11)$$

for all combination of μ and ν where $1\leq\mu,\nu\leq n_p$ and $\mu\neq\nu$. This means that the patterns to be embedded in the network must be orthogonal to each other, though the patterns provided in most cases are non-orthogonal.

Projection rule [14, 15, 12] is a learning scheme that can embed non-orthogonal patterns in a network. The key idea of the Projection rule is that non-orthogonal patterns are first projected onto orthogonal ones, and then the Hebbian rule is applied to the projected patterns [10]. Projection is conducted by introducing the matrix $\{Q_{\mu\nu}\}$, defined as:

$$Q_{\mu\nu} = \frac{1}{N}\sum_p\varepsilon_p^{\mu*}\varepsilon_p^\nu. \qquad (12)$$

The weight matrix of the network, $\tilde{w}$, is calculated by

$$\tilde{w}_{pq} = \frac{1}{N}\sum_{\mu,\nu} \varepsilon_p^{\mu} \left(Q^{-1}\right)_{\mu\nu} \varepsilon_q^{\nu *}, \qquad (13)$$

where Q^{-1} is the pseudo inverse matrix of Q. Patterns embedded by this scheme become stable points in the network, as in the case of the Hebbian rule. This can be checked by calculating the action potential of a neuron $\tilde{h}_p$ by applying an embedded pattern ε^{σ} as the input to the network:

$$\begin{aligned} \tilde{h}_p &= \sum_{q=1}^{N} \tilde{w}_{pq}\varepsilon_q^{\sigma} \\ &= \frac{1}{N}\sum_{\mu,\nu} \varepsilon_p^{\mu} \left(Q^{-1}\right)_{\mu\nu} \sum_q \varepsilon_q^{\nu *}\varepsilon_q^{\sigma} \\ &= \sum_{\mu,\nu} \varepsilon_p^{\mu} \left(Q^{-1}\right)_{\mu\nu} Q_{\nu\sigma} \\ &= \sum_{\mu} \varepsilon_p^{\mu} \left(Q^{-1}Q\right)_{\mu\sigma} \\ &= \sum_{\mu} \varepsilon_p^{\mu}\delta_{\mu,\sigma} \\ &= \varepsilon_p^{\sigma}, \end{aligned} \qquad (14)$$

where $\delta_{\mu,\sigma}$ denotes the Kronecker delta function.

2.4 Iterative Learning Rule

Iterative learning rule for a complex-valued multistate neural network has been proposed and theoretically analyzed in [10]. This learning rule is a complex-valued extension of the iterative learning for real-valued one in [16]. The connection weight w_{pq} is updated by using the desired memory pattern as

$$w_{pq}^{new} = w_{pq}^{old} + \delta w_{pq}, \qquad (15)$$

$$\delta w_{pq} = \frac{1}{N}\varepsilon_p^{\mu}\varepsilon_q^{\mu *}, \qquad (16)$$

where $\varepsilon_p^{\mu *}$ is the complex conjugate of ε_p^{μ}, i.e., $\varepsilon_p^{\mu *} = e^{-i\xi_p^{\mu}\varphi_0}$. After the update, the action potential of neuron p becomes

$$\begin{aligned} h_p^{new} &= \sum_{q=1}^{N} w_{pq}^{new} u_q \\ &= \sum_q \left(w_{pq}^{old} + \delta w_{pq}\right) u_q \\ &= h_p^{old} + \sum_q \frac{1}{N}\varepsilon_p^{\mu}\varepsilon_q^{\mu *} u_q. \end{aligned} \qquad (17)$$

Consider the effect of updating the connection weight on the memory pattern state ε_p^{μ}. This can be confirmed by setting $u_q = \varepsilon_q^{\mu}$ in Eq.(17) as

$$\begin{aligned} h_p^{new} &= h_p^{old} + \sum_q \frac{1}{N}\varepsilon_p^{\mu}\varepsilon_q^{\mu *}\varepsilon_q^{\mu} \\ &= h_p^{old} + \varepsilon_p^{\mu}, \end{aligned} \qquad (18)$$

since $\sum_q \varepsilon_q^{\mu *}\varepsilon_q^{\mu} = N$ holds. These vectors in a complex plane are shown in Figure 3. The relation of these vectors indicates that the direction of h_p^{new} is turning towards ε_p^{μ} by the iterative applications of Eq.(15). Hence, the memory pattern will be sufficiently stable when the condition described in Eq.(9) is satisfied. The achievement of stability by the iterative applications of Eq.(15) is explicitly shown in Section B of Appendix. Since the modification of the connection weights by Eq.(18) is conducted for a given memory pattern, this rule could have a flexibility for making the basin of attractor for a memory pattern by controlling the presentation frequencies of the memory patterns.

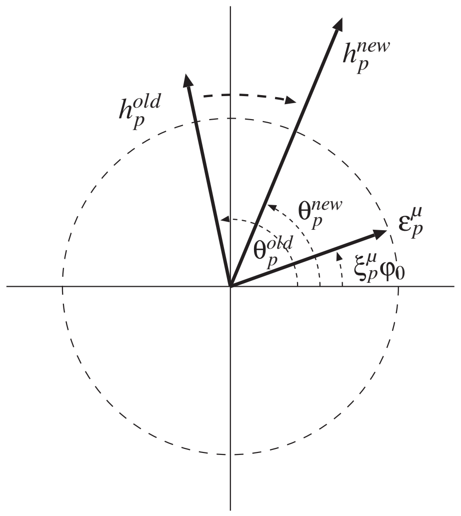


Figure 3. Behavior of vectors in a complex plane due to the application of iterative learning

3 Experimental Results

3.1 Image Retrieval from Noisy Input

We first investigate the ability for retrieving the stored patterns from the noisy inputs, in order to ensure that associative memory with the iterative learning rule actually works. Figure 4 shows three stored patterns for this experiment, which are used in [18]. The size of these images are adjusted to

$90 \times 90 = 8100$ pixels and each pixel value is represented by 5 bits (32 levels).

The number of neurons in the associative memory network is the same as the size of the images, i.e., $N = 8100$. The resolution factor for the neuron state is also set to the same as the resolution of the pixel value, $K = 32$. In learning stage, each of the input images are used for modifying the connection weights, in order of the images in Figs. 4(a), 4(b), and 4(c). The number of iterations is set to 10.

After learning, the test input images are used to the initial configuration of the network, and then the updates of the neuron states are conducted until the configuration of the network converges, i.e. until the states of all neurons in the network do not change by using the Eqs.(1) and (2).

For the test input images to the associative memory, one of these stored patterns with corruption is adopted. The corruption is conducted so that, for each of pixels in the target image, the pixel value is replaced to random value with a certain probability (r_n). Examples of test input images with various r_ns are shown in Figure 5. Higher value of r_n results in an image with larger region being corrupted.

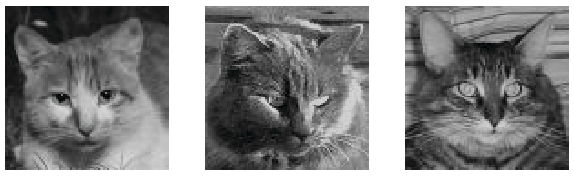

(a) (b) (c)

Figure 4. Three stored images for the image retrieval experiment

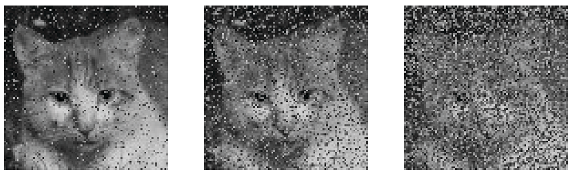

(a) $r_n = 0.1$ (b) $r_n = 0.3$ (c) $r_n = 0.6$

Figure 5. Input images with various noise probabilities r_n

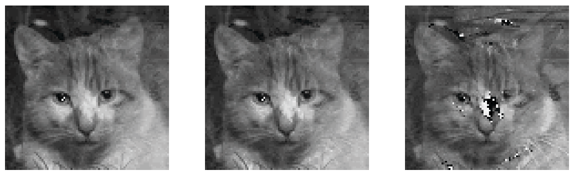
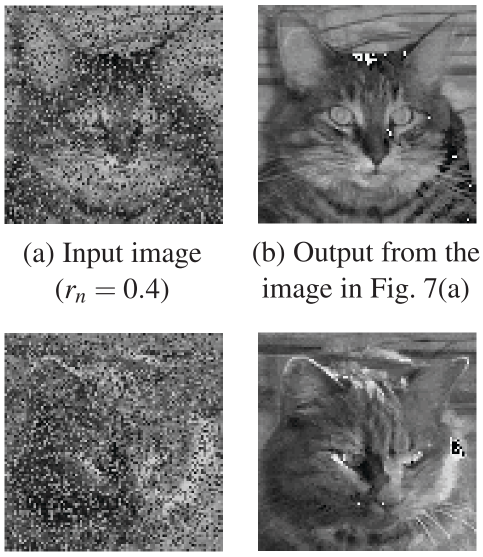

(a) $r_n = 0.1$ (b) $r_n = 0.3$ (c) $r_n = 0.6$

Figure 6. Output images from the network when the images in Figure 5 are used for input

Figure 6 shows the output configurations of the network for the inputs of Figures 4(a), 4(b), and 4(c), respectively. This result shows that the network can retrieve the correct image for the input image with lower r_n, thus iterative rule could certainly embed each of the stored patterns with a certain attractors around it. By using other input images that are originated from the images in Figures. 4(b) and 4(c), similar results can be obtained as shown in Figure 7.


(a) Input image ($r_n = 0.4$) (b) Output from the image in Fig. 7(a)

(c) Input image ($r_n = 0.5$) (d) Output from the image in Fig. 7(c)

Figure 7. Other example of input and output images

3.2 Stability Analysis for Embedded Patterns

We next explore the stabilities of patterns embedded by two learning rules. This is conducted by evaluating retrieval performances for the network from one of stored patterns as its initial state. In this experiment, the embedded patterns are composed of randomly generated values. The size of the pattern (the number of neurons in the network) is set to 100, and the resolution factor K is set to $K = 4, 6, 8, 16, 32, 64$. The number of the stored patterns, denoted by M, varies such that $M = 1, \cdots, 30$.

The stability of the patterns are checked by the following procedure. First, for given K and M, the stored patterns are prepared and are embedded to the network. In the iterative learning, presenting the stored patterns to the network with 10 iterations is conducted for modifying the connection weights of the network. After the learning, each of the stored patterns is set to the network as its initial configuration, then the updates of the network are conducted for all the neurons in the network. If the configuration of the network does not change, the input pattern (one of the stored patterns) can be stable; otherwise the input pattern cannot be embedded. An embedding process with given parameters is regarded as successful if all the stored patterns are stable.

Figure 8 shows the M dependencies of the retrieval success rates with several Ks, in which 100 different sets of patterns are used for each M. For the fixed size of the network, it becomes difficult to embed the larger number of patterns, as in the case of real-valued Hopfield networks. Also, the number of patterns to be embedded correctly depends on the resolution K for the neuron state. A neuron with larger value of K should contain much more information for the embedded patterns, and it is equivalent to larger value of M in the network. Thus, increasing M and/or K in the network with small number of neurons will lead to degradation of embedded patterns in the network. For comparison of performance, the M dependence of the retrieval success rates obtained by projection rule is shown in Figure 9. Similar tendencies to the results obtained by iterative learning rule are shown though overall capacities of patterns in the network are much higher.

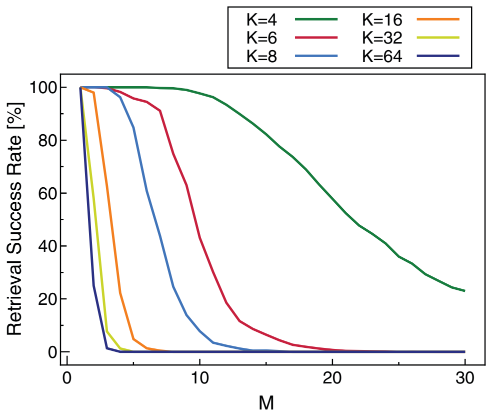


Figure 8. M dependencies of the retrieval success rates by iterative learning

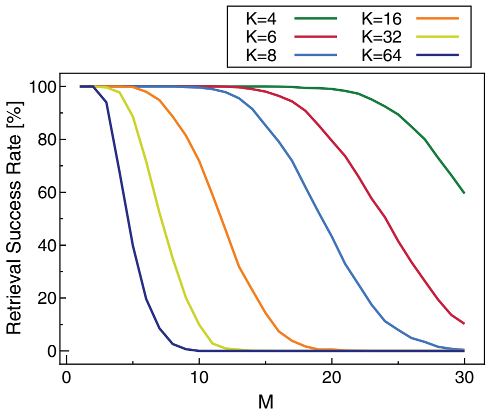
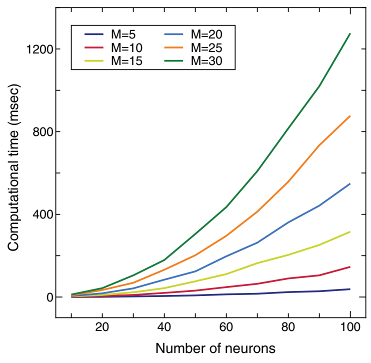
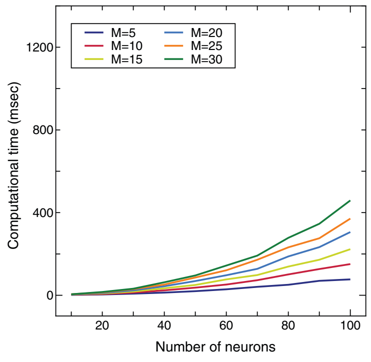


Figure 9. M dependencies of the retrieval success rates by Projection rule

The computational costs for the projection rule and iterative learning rule are discussed. Most of the computation on the projection rule is devoted to the creation of the weight matrix (Eq.(13)), thus the number of neurons (N) becomes a major parameter determining the computational time. Figure 10(a) shows computational time by the projection rule with respect to the number of neurons, in which evaluations are conducted on a PC (Core i7 860 2.80GHz, Memory 8GB). The curves in this figure have similar tendencies and are approximated as aN^2 where a is a parameter depending on the number of memory patterns M. In the iterative learning, computational cost is affected by the numbers of iterations, neurons, and patterns. The computational time, with various Ms and 10 iterations, with respect to the number of neurons is shown in Figure 10(b). Computational time can also be approximated as aN^2, though overall computational costs for the iterative learning are relatively low.



(a) Projection rule



(b) Iterative learning rule

Figure 10. Computational time for (a) projection rule and (b) iterative learning rule with respect to the number of embedded patterns

3.3 Stability of Embedded Patterns by Iterative Learning Rule

In the previous Section, we show some comparisons of iterative learning rule and projection rule from the viewpoint of retrieval performances by setting one of stored patterns as an initial state of the network. Performance differences arise due to the stability in the embedding processes by these learning schemes. In this Section, we investigate the stabilities on the patterns embedded by the iterative learning in more detail, through an analysis on the evolutionary process of weight connections in learning.

We have already discussed the stability of the memory patterns in Section 2.2 by using the phase relation (Eqs.(7) and (8)). Thus, it is useful to adopt the phase difference between the action potential of a neuron and a memory state for the corresponding neuron, in order to evaluate the stability of an embedded pattern. A phase difference d_p^μ for the p-th neuron in the μ-th pattern at the t-th iteration in learning is defined as

$$d_p^\mu(t) = |\arg(h_p(t)) - \arg(\varepsilon_p^\mu)|. \qquad (19)$$

Thus the basic stability condition is $d_p^\mu < \varphi_0/2$. The average of $d_p^\mu(t)$ for all neurons is denoted as $\overline{d^\mu(t)}$.

We introduce a procedure for embedding the memory patterns by iterative learning. In the first iteration in embedding patterns, each of the memory patterns is prepared and the connection weights are updated by this pattern according to

$$w_{pq}^{new} = w_{pq}^{old} + \alpha \cdot \delta w_{pq}, \qquad (20)$$

where α is a real-valued parameter. When $\alpha = 1.0$, this updating scheme corresponds to the original updating scheme (Eq.(15)). Then, d_p^μ is calculated for each of the memory patterns and for each of the neurons. At each subsequent iteration in learning, only the patterns with the condition of $d_p^\mu \geq \varphi_0/2$ for any p are prepared and used for updating the connection weights.

The following conditions are used for the experiments. The numbers of neurons and memory patterns are $N = 100$ and $M = 15$, respectively, and the resolution factor K is set to 4 (thus $\varphi_0/2 = 0.7854$). Each element of the memory patterns is randomly generated. Iterative learning with $\alpha = 0.1$ in Eq.(20) and projection rule are used for embedding the memory patterns, and the performances of these schemes are evaluated. As in the experiment in the previous Section, all patterns are embedded to the network and then each of the patterns is set to the initial state of the network. If the configuration of the network do not change from the initial configuration, this pattern is stably embedded. Retrieval is regarded as successful if all the patterns are stably embedded.

First we show the evolutions of the averaged phase differences by iterative learning. Table 1 shows an example of averaged phases for the first six iterations of learning. In this table there are the averaged phase differences for only four patterns ($\mu = 2, 4, 5, 14$) that are involved in the iterations of learning. At the first iteration of learning there are two neurons that do not satisfy the condition $d_p^\mu < \varphi_0/2$ for the 14-th pattern. Thus in the second iteration, only the 14-th pattern is used for updating the connection weights. As a result, the averaged phase difference for this pattern actually decreased but the phase differences for the other patterns slightly increased. At the third iteration, the second pattern is involved in learning and then the phase difference for this pattern decreased, by the reason that some phase differences with respect to the second pattern exceed to the threshold $\varphi_0/2$. Similar process is conducted at the each subsequent iteration in learning. The averaged differences gradually decreases according to the iterations, though sometimes they slightly increase.

As described above, strengthening the embed for a particular pattern often results in weakening the other patterns, though embedding all patterns is still possible. Figure 11 shows an example of changes of retrieval success rate with respect to the iteration in learning. The success rate increases according to the iterations of learning and sometimes fluctuates by strengthening and weakening the stored patterns by learning. At the 66-th iteration, all the patterns could be retrieved thus embedded. The averaged phase differences for the network with 66-th iteration in learning are shown in Figure 12. The results for the network in which the patterns are embedded by projection rule are also shown in this figure. The phase differences by projection rule are much smaller than those by iterative learning rule, thus the patterns are better embedded by projection rule. This result would reflect the possible number of embedded patterns, shown in Figs. 8 and 9.

The parameters for embedding patterns, such as the number of memory patterns M and the resolution factor K, impact the process of embedding, i.e., the evolution of the retrieval success rate. Figure 13 shows the retrieval success rates with respect to the learning iterations in the case of $M = 20$ and $K = 4$ and in the case of $M = 15$ and $K = 6$. Increase of K or M makes difficult to embed patterns in the iterative learning rule, as suggested in Figure 8.

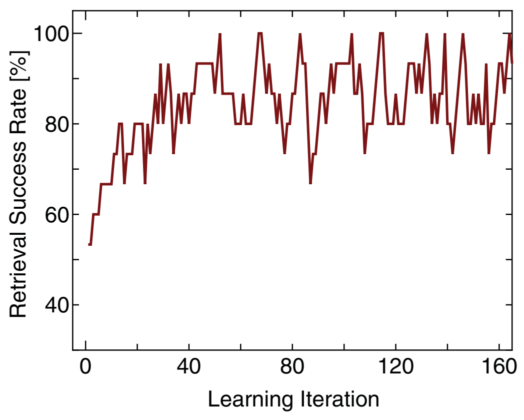


Figure 11. Evolution of retrieval success rate by iterative learning rule ($N = 100, M = 15, K = 4$)

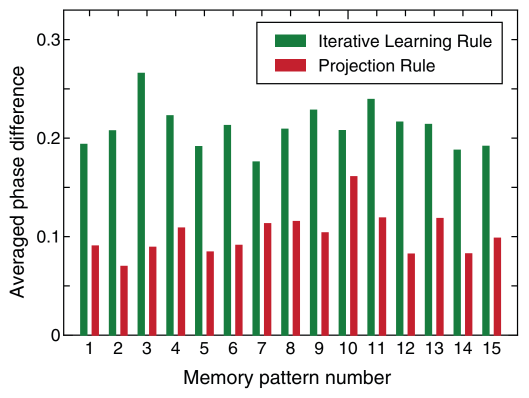


Figure 12. Averaged phase differences for 15 patterns by iterative learning rule with 66-th iteration and projection rule

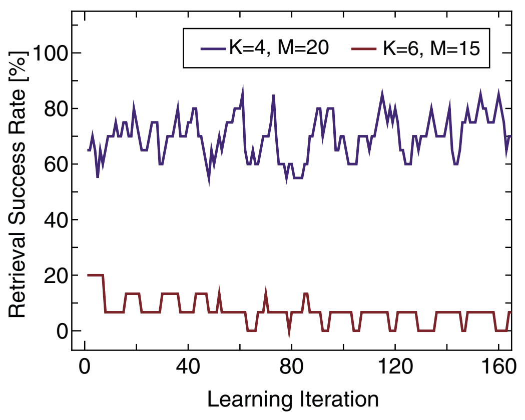


Figure 13. Evolutions of retrieval success rates by iterative learning rule ($N = 100, M = 20, K = 4$) and ($N = 100, M = 15, K = 6$)

Table 1. An example of evolutions for the averaged phase differences according to the iterations by iterative learning rule. At the first iteration of learning, all the patterns are involved in learning. At the second, third, fourth, and fifth iteration of learning, the 14-th, the second, the fourth, and the fifth patterns, respectively, are involved in learning.

μ	$\overline{d^\mu(1)}$	$\overline{d^\mu(2)}$	$\overline{d^\mu(3)}$	$\overline{d^\mu(4)}$	$\overline{d^\mu(5)}$	$\overline{d^\mu(6)}$
2	0.23302	0.23318	0.21511	0.22016	0.22528	0.20918
4	0.23391	0.23553	0.24084	0.22185	0.20570	0.21051
5	0.21062	0.21377	0.21710	0.21755	0.21805	0.22197
14	0.22466	0.20580	0.20582	0.20862	0.21174	0.21180

4 Conclusion

In this paper, the stabilities of embedded patterns are investigated in the complex-valued associative memory. The associative memory is based on complex-valued multistate Hopfield neural network, and iterative learning rule and projection rule are adopted for embedding patterns. The experimental results show that the patterns can be certainly embedded in the network with their attractors, and the capability for embedding depends on the number of patterns and the resolution for representing neuron states. The stability of embedded patterns have also been explored through measuring the phase differences on presenting the memory patterns to the trained networks.

Though the computational cost for iterative learning rule is much lower than that of projection rule, the capability of iterative learning rule is not high as projection rule, as indicated in the experimental results. One of our future researches directs to developing improvement schemes for the learning capability by iterative learning rule. This could be realized by incorporating a more sophisticated control of updating the connection weights in the network. Analyzing other types of learning schemes, such as the scheme in [11], will be necessary. The analysis of this learning scheme for higher dimensional associative memory is also a challenging problem, such as for quaternionic multistate Hopfield neural networks [19, 20, 21] in which two or three kinds of phase values are available for representing neuron's state.

Acknowledgment

This study was financially supported by the Japan Society for the Promotion of Science (Grant-in-Aids for Young Scientists (B) 24700227 and Scientific Research (C) 16K00248).

References

[1] A. Hirose, editor, Complex-Valued Neural Networks: Theories and Application, volume 5 of Innovative Intelligence, World Scientific Publishing, Singapore, 2003.

[2] T. Nitta, editor, Complex-Valued Neural Networks: Utilizing High-Dimensional Parameters, Information Science Reference, Hershey, New York, 2009.

[3] A. Hirose, editor, Complex-Valued Neural Networks: Advances and Applications, The IEEE Press Series on Computational Intelligence, Wiley-IEEE Press, 2013.

[4] Y. Nakano and A. Hirose, Improvement of Plastic Landmine Visualization Performance by Use of Ring-CSOM and Frequency-Domain Local Correlation, IEICE Transactions, 92-C(1), pp.102–108, 2009.

[5] Rajoo Pandey, Complex-Valued Neural Networks for Blind Equalization of Time-Varying Channels, International Journal of Signal Processing, 1(1), pp.1–8, 2004.

[6] A. J. Noest, Associative Memory in Sparse Neural Networks, Europhysics Letters, 6(6), pp.469–474, 1988.

[7] N. N. Aizenberg and I. N. Aizenberg, CNN Based on Multi-Valued Neuron as a Model of Associative Memory for Gray-Scale Images, Proceedings of the 2nd IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA-92), pp.36–41, 1992.

[8] I. N. Aizenberg, N. N. Aizenberg, and J. Vandewalle, Multi-Valued and Universal Binary Neurons – Theory, Learning and Applications –, Kluwer Academic Publishers, Boston/Dordrecht/London, 2000.

[9] S. Jankowski, A. Lozowski, and J. M. Zurada, Complex-Valued Multistate Neural Associative Memory, IEEE Transactions on Neural Networks, 7(6), pp.1491–1496, 1996.

[10] T. Isokawa, H. Nishimura, and N. Matsui, An Iterative Learning Scheme for Multistate Complex-Valued and Quaternionic Hopfield Neural Networks, Proceedings of International Joint Conference on Neural Networks (IJCNN2009), pp.1365–1371, 2009.

[11] M. K. Müezzinoğlu, C. Güzeliş, and J. M. Zurada, A New Design Method for the Complex-Valued Multistate Hopfield Associative Memory, IEEE Transactions on Neural Networks, 14(4), pp.891–899, 2003.

[12] D.-L. Lee, Improvements of complex-valued Hopfield associative memory by using generalized projection rules, IEEE Transaction on Neural Networks, 17(5), pp.1341–1347, 2006.

[13] M. Kobayashi, Pseudo-relaxation learning algorithm for complex-valued associative memory, International Journal of Neural Systems, 18(2), pp.147–156, 2008.

[14] T. Kohonen, Self-Organization and Associative Memory, Springer, Berlin, Heidelberg, 1984.

[15] L. Personnaz, I. Guyon, and G. Dreyfus, Collective Computational Properties of Neural Networks: New Learning Mechanisms, Physical Review A, 34, pp.4217–4228, 1986.

[16] S. Diederich and M. Opper, Learning of Correlated Patterns in Spin-Glass Networks by Local Learning Rules, Physical Review Letters, 58, pp.949–952, 1987.

[17] H. Yamamoto, T. Isokawa, H. Nishimura, N. Kamiura, and N.Matsui, Pattern Stability on Complex-Valued Associative Memory by Local Iterative Learning Scheme, Proceedings of 6th International Conference on Soft Computing and Intelligent Systems & 13th International Symposium on Advanced Intelligent Systems (SCIS-ISIS 2012), pp.39–42, 2012.

[18] F. Flueret and D. Geman, Stationary Features and Cat Detection, Journal of Machine Learning Research, 9, pp.2549–2578, 2008.

[19] T. Isokawa, H. Nishimura, A. Saitoh, N. Kamiura, and N. Matsui, On the Scheme of Quaternionic Multistate Hopfield Neural Network, Proceedings of Joint 4th International Conference on Soft Computing and Intelligent Systems and 9th International Symposium on advanced Intelligent Systems (SCIS & ISIS 2008), pp.809–813, 2008.

[20] T. Isokawa, H. Nishimura, and N. Matsui, Commutative Quaternion and Multistate Hopfield Neural Networks, Proceedings of IEEE World Congress on Computational Intelligence (WCCI2010), pp.1281–1286, 2010.

[21] T. Minemoto, T. Isokawa, H. Nishimura, and N. Matsui, Quaternionic multistate Hopfield neural network with extended projection rule, Artificial Life and Robotics, 21(1), pp.106–111, 2016.

A Proof for the Stability of the Network

This Section describes a proof for the stability of the network, i.e., the network state with an initial configuration always converges to one of the local minima. This can be shown by monotonically decrease of the energy defined in Eq.(4) under the following conditions: $W = \{w_{pq}\}$ being a Hermitian matrix ($w_{pq} = w^*_{pq}$), self-connections of the network take real and non-negative values ($w_{rr} \geq 0$), and a phase difference in neuron state's update is not large such as $|\Delta\varphi| < \varphi_0$.

The first stage of the proof is to explicitly write the contribution of a neuron to the whole of energy in the network with N neurons. Let $E_r(t)$ be a contribution of neuron r at the time t to the energy. This can be written as

$$\begin{aligned}
E_r(t) &= -\frac{1}{2}\left(\sum_{p=1}^{N} w_{pr}u^*_p(t)\right)u_r(t) \\
&\quad -\frac{1}{2}\left(\sum_{p=1}^{N} w_{rq}u_q(t)\right)u^*_r(t) \\
&\quad +\frac{1}{2}w_{rr}u^*_r(t)u_r(t) \\
&= -\frac{1}{2}\left(\sum_{p=1}^{N} w^*_{rp}u^*_p(t)\right)u_r(t) \\
&\quad -\frac{1}{2}\left(\sum_{p=1}^{N} w_{rp}u_q(t)\right)u^*_r(t) \\
&\quad +\frac{1}{2}w_{rr} \\
&= -\frac{1}{2}h^*_r(t)u_r(t) - \frac{1}{2}h_r(t)u^*_r(t) + \frac{1}{2}w_{rr} \\
&= -Re\left(u^*_r(t)h_r(t)\right) + \frac{1}{2}w_{rr}. \qquad (21)
\end{aligned}$$

Let us assume that this neuron r updates its state. The contribution $E_r(t+1)$ is then described as

$$E_r(t+1) = -Re\left(u^*_r(t+1)h_r(t+1)\right) + \frac{1}{2}w_{rr}. \qquad (22)$$

The action potential of the neuron r, $h_r(t+1)$, can be represented by its variables at the time t and time $(t+1)$.

$$\begin{aligned} h_r(t+1) &= \sum_{q=1}^{N} w_{rq}u_q(t+1) \\ &= \sum_{q=1}^{N} w_{rq}u_q(t+1) - w_{rr}u_r(t) \\ &\quad + w_{rr}u_r(t+1) \\ &= h_p(t) - w_{rr}\left(u_r(t) - u_r(t+1)\right). \end{aligned} \tag{23}$$

Using this variable, Eq.(22) can be rewritten as

$$\begin{aligned} E_r(t+1) &= -Re\left(u_r^*(t+1)h_r(t)\right. \\ &\quad \left.- w_{rr}\left(u_r(t) - u_r(t+1)\right)\right) + \frac{1}{2}w_{rr} \\ &= -Re\left(u_r^*(t+1)h_r(t)\right) \\ &\quad + w_{rr}Re\left(u_r^*(t+1)\left(u_r(t) - u_r(t+1)\right)\right) \\ &\quad + \frac{1}{2}w_{rr} \\ &= -Re\left(u_r^*(t+1)h_r(t)\right) \\ &\quad + w_{rr}Re\left(u_r^*(t+1)u_r(t) - 1\right) + \frac{1}{2}w_{rr}. \end{aligned} \tag{24}$$

The difference of the contribution between the time t and $(t+1)$, ΔE, is calculated from Eqs.(21) and (22).

$$\begin{aligned} \Delta E &= E_r(t+1) - E_r(t) \\ &= -\left\{Re\left(u_r^*(t+1)h_r(t)\right) - Re\left(u_r^*(t)h_r(t)\right)\right\} \\ &\quad + w_{rr}\left\{Re\left(u_r^*(t+1)u_r(t)\right) - 1\right\} \\ &= -(X_1 - X_2) + w_{rr}X_3, \end{aligned} \tag{25}$$

where

$$\begin{aligned} X_1 &= Re\left(u_r^*(t+1)h_r(t)\right), \\ X_2 &= Re\left(u_r^*(t)h_r(t)\right), \\ X_3 &= Re\left(u_r^*(t+1)u_r(t)\right) - 1. \end{aligned}$$

Proving the monotonically decrease of E corresponds to showing $\Delta E \leq 0$ with respect to the change of network state. Thus it is necessary to investigate each of X_1, X_2, and X_3. For this purpose the following two relations are required. One is the relation of the outputs between the time t and $(t+1)$:

$$u_r(t+1) = u_r(t)e^{ia\varphi_0}, \tag{26}$$

where a is an integer. The other relation refers to the action potential $h_r(t)$ and the output $u_r(t)$:

$$\begin{aligned} h_r(t) &= |h_r(t)|e^{i\Delta\varphi}u_r(t+1) \\ &= |h_r(t)|e^{i(a\varphi_0+\Delta\varphi)}u_r(t). \end{aligned} \tag{27}$$

Investigation the term X_3 is firstly conducted. the term $u_r^*(t+1)u_r(t)$ is decomposed by using the abovementioned relations, as

$$\begin{aligned} u_r^*(t+1)u_r(t) &= \left(u_r(t)e^{ia\varphi_0}\right)^* u_r(t) \\ &= u_r^*(t)e^{-ia\varphi_0}u_r(t) \\ &= e^{-ia\varphi_0}. \end{aligned}$$

Thus $X_3 = \cos(-a\varphi_0) - 1$ is obtained. This term does not take a positive value due to $-1 \leq \cos(-a\varphi_0) \leq 1$. Consequently $w_{rr}X_3 \leq 0$ is satisfied under the condition of w_{rr} being a non-negative value.

Similarly the terms X_1 and X_2 are investigated as

$$\begin{aligned} X_1 &= Re\left(u_r^*(t)e^{-ia\varphi_0}|h_r(t)|e^{i(a\varphi_0+\Delta\varphi)}u_r(t)\right) \\ &= |h_r(t)|Re(e^{i\Delta\varphi}) \\ &= |h_r(t)|\cos(\Delta\varphi), \end{aligned}$$

and

$$\begin{aligned} X_2 &= Re\left(u_r^*(t)|h_r(t)|e^{i(a\varphi_0+\Delta\varphi)}u_r(t)\right) \\ &= |h_r(t)|\cos(a\varphi_0 + \Delta\varphi), \end{aligned}$$

respectively. Thus under the condition $|\Delta\varphi| < \varphi_0$,

$$\begin{aligned} X_1 - X_2 &= |h_r(t)|\left(\cos(\Delta\varphi) - \cos(a\varphi_0 + \Delta\varphi)\right) \\ &\geq 0, \end{aligned} \tag{28}$$

is obtained. Finally $\Delta E \leq 0$ is obtained from these relations.

B Proof for the stability of local iterative learning scheme

This Section describe a proof concerning the stability of iterative learning rule, i.e., this rule can actually embed each of the stored patterns as its stable configuration in the network. Let θ_p^{new} and θ_p^{old} be the angles of h_p^{new} and h_p^{old} in a complex plane, respectively. Equation (18) can be expressed as

$$|h_p^{new}|e^{i\theta_p^{new}} = |h_p^{old}|e^{i\theta_p^{old}} + |\varepsilon_p^{\mu}|e^{i\xi_p^{\mu}\varphi_0}.$$

Multiplication this equation with $e^{-i\xi_p^{\mu}\varphi_0}$ gives

$$|h_p^{new}|e^{i(\theta_p^{new}-\xi_p^{\mu}\varphi_0)} = |h_p^{old}|e^{i(\theta_p^{old}-\xi_p^{\mu}\varphi_0)} + |\varepsilon_p^{\mu}|.$$

This can be written as two equations for real and imaginary components:

$$\begin{aligned} &|h_p^{new}|\cos\left(\theta_p^{new} - \xi_p^{\mu}\varphi_0\right) \\ &= |h_p^{old}|\cos\left(\theta_p^{old} - \xi_p^{\mu}\varphi_0\right) + |\varepsilon_p^{\mu}| \end{aligned} \tag{29}$$

and

$$|h_p^{new}|\sin\left(\theta_p^{new} - \xi_p^{\mu}\varphi_0\right) = |h_p^{old}|\sin\left(\theta_p^{old} - \xi_p^{\mu}\varphi_0\right). \tag{30}$$

To represent the value of the update frequency, h_p^{L+1} and h_p^L for h_p^{new} and h_p^{old}, respectively, are substituted where L is the number of updates in iterative learning. Then, Eq.(29) can be expressed as

$$\begin{aligned} & |h_p^{L+1}|\cos(\theta_p^{L+1} - \xi_p^\mu\varphi_0) \\ = \ & |h_p^L|\cos(\theta_p^L - \xi_p^\mu\varphi_0) + |\varepsilon_p^\mu| \\ = \ & |h_p^{L-1}|\cos(\theta_p^{L-1} - \xi_p^\mu\varphi_0) + 2|\varepsilon_p^\mu| \\ & \vdots \\ = \ & |h_p^0|\cos(\theta_p^0 - \xi_p^\mu\varphi_0) + (L+1)|\varepsilon_p^\mu|. \end{aligned}$$

Hence, the following relation is obtained:

$$\cos(\theta_p^{L+1} - \xi_p^\mu\varphi_0) = \frac{|h_p^0|\cos(\theta_p^0 - \xi_p^\mu\varphi_0) + (L+1)|\varepsilon_p^\mu|}{|h_p^{L+1}|}. \tag{31}$$

In a similar way, using Eq.(30), the following relation is obtained:

$$|h_p^{L+1}|\sin(\theta_p^{L+1} - \xi_p^\mu\varphi_0) = |h_p^0|\sin(\theta_p^0 - \xi_p^\mu\varphi_0) \tag{32}$$

From these equations,

$$\begin{aligned} & \cos(\theta_p^{L+1} - \xi_p^\mu\varphi_0) \\ = \ & \frac{|h_p^0|\Theta + (L+1)|\varepsilon_p^\mu|}{\sqrt{|h_p^0|^2 + 2(L+1)|h_p^0||\varepsilon_p^\mu|\Theta + (L+1)^2|\varepsilon_p^\mu|^2}}. \end{aligned} \tag{33}$$

where $\Theta = \cos(\theta_p^0 - \xi_p^\mu\varphi_0)$.

According to the iterative update in the connection weights, Eq.(33) becomes

$$\cos(\theta_p^{L+1} - \xi_p^\mu\varphi_0) \simeq \frac{(L+1)|\varepsilon_p^\mu|}{\sqrt{(L+1)^2|\varepsilon_p^\mu|^2}} = 1, \tag{34}$$

for large values of L. This shows that the direction of h_p^{L+1} approaches that of ε_p^μ, that is,

$$|\arg(h_p^{L+1}) - \xi_p^\mu\varphi_0| < |\arg(h_p^0) - \xi_p^\mu\varphi_0| \tag{35}$$

for large value of L.

It can be seen that the direction of h_p would move towards that of ε_p^μ by even one update. From the Eqs.(29) and (30), the following relations hold:

$$\cos(\theta_p^{new} - \xi_p^\mu\varphi_0) = \frac{|h_p^{new}|^2 - |h_p^{old}|^2 + |\varepsilon_p^\mu|^2}{2|h_p^{new}||\varepsilon_p^\mu|}$$

and

$$\cos(\theta_p^{old} - \xi_p^\mu\varphi_0) = \frac{|h_p^{new}|^2 - |h_p^{old}|^2 - |\varepsilon_p^\mu|^2}{2|h_p^{old}||\varepsilon_p^\mu|}.$$

We see that the difference between the above two equations is always greater than 0, i.e.,

$$\begin{aligned} & \cos(\theta_p^{new} - \xi_p^\mu\varphi_0) - \cos(\theta_p^{old} - \xi_p^\mu\varphi_0) \\ = \ & \frac{\left(|h_p^{new}| + |h_p^{old}|\right)\left\{\left(|h_p^{new}| - |h_p^{old}|\right)^2 + |\varepsilon_p^\mu|^2\right\}}{2|h_p^{new}||h_p^{old}||\varepsilon_p^\mu|} \\ > \ & 0. \end{aligned} \tag{36}$$

Consequently, we obtain

$$|\arg(h_p^{new}) - \xi_p^\mu\varphi_0| < |\arg(h_p^{old}) - \xi_p^\mu\varphi_0|. \tag{37}$$

This indicates that even one update can make the direction of h_p move towards that of ε_p^μ. Thus the desired memory patterns can be embedded by the proposed learning scheme expressed in Eq.(15).

In the case of all the patterns being orthogonal to one another, this rule becomes equivalent to the Hebbian learning rule. The connection weights are defined by the Hebbian rule in Eq.(10). The action potential at $u_p = \varepsilon_p^\mu$ becomes

$$\begin{aligned} h_p &= \sum_{q=1}^{N} w_{pq}u_q \\ &= \sum_{q=1}^{N} \frac{1}{N}\left(\sum_{\nu=1}^{n_p} \varepsilon_p^\nu\varepsilon_q^{\nu*}\right)\varepsilon_q^\mu \\ &= \sum_\nu \varepsilon_p^\nu \cdot \frac{1}{N}\sum_q \varepsilon_q^{\nu*}\varepsilon_q^\mu \\ &= \sum_\nu \varepsilon_p^\nu \delta_{\nu,\mu} \\ &= \varepsilon_p^\mu. \end{aligned}$$

In this case, the relation

$$|\arg(h_p) - \xi_p^\mu\varphi_0| = 0 < \kappa \tag{38}$$

always holds.

Teijiro Isokawa received his B.E. degree (Electronic Engineering), M.E. degree (Electronic Engineering), and D.E. degree (Doctor of Engineering) in 1996, 1999, and 2004, respectively, from Himeji Institute of Technology, Japan. He is currently an associate professor in the Graduate School of Engineering, University of Hyogo, Japan. His research interests include nanocomputing, molecular robotics, hypercomplex-valued neural networks, and cognitive models.

Hiroki Yamamoto received his B.E. degree and M.E. degree in 2012 and 2014, respectively, from University of Hyogo, Japan. His research interests include associative memories based on complex and quaternionic neural networks.

Haruhiko Nishimura graduated from the Department of Physics, Shizuoka University in 1980, and completed the doctoral program at Kobe University and received the PhD degree in 1985. He is currently a professor in the Graduate School of Applied Informatics, University of Hyogo. His research field is intelligent systems science by several architectures such as neural networks and complex systems. He is also presently engaged in research on biomedical, healthcare, and high confidence sciences. He is a member of the IEEE, IEICE, IPSJ, ISCIE, JNNS, and others, and was awarded ISCIE paper prize in 2001 and JSKE paper prize in 2010.

Takayuki Yumoto received the BE, Master of Informatics, and PhD degrees in Informatics from Kyoto University, in 2002, 2004, and 2007, respectively. He is an assistant professor at University of Hyogo since 2007. His research interests include Web search and mining. He is a member of the ACM, the IEEE Computer Society, the Information Processing Society of Japan (IPSJ), the Institute of Electronics, Information and Communication Engineers (IEICE), and the Database Society of Japan (DBSJ).

Naotake Kamiura received the B.E. degree (Electronic Engineering) in 1990, the M.E. degree (Electronic Engineering) in 1992 and the D.E. degree (Doctor of Engineering) in 1995 from Himeji Institute of Technology, Japan. He is currently a professor in the Department of Electronics and Computer Science, Graduate School of Engineering, University of Hyogo. His research interests include the application of soft computing to medical engineering. He is members of the Institute of Electronics, Information and Communication Engineers, the the Japanese Society of Medical Imaging Technology and the IEEE.

Nobuyuki Matsui received his B.S. degree in physics from the Faculty of Science, Kyoto University, Japan, in 1975, and M.E. and Dr. Eng. degrees in nuclear engineering from Kyoto University in 1977 and 1980, respectively. He is currently a professor emeritus at University of Hyogo. Dr. Matsui is a member of INNS, IEICE, ISCIE, SICE and the Physical Society of Japan.