(category: genetic algorithms)
Neural Network Model of Associative Memory: To Visualize Solutions in Weight Space

Akira Imada
Nara Institute of Science and Technology
1-11-13-604 Abeno, Hannan, Osaka 545-0021, Japan
E-mail: ai@kk.iij4u.or.jp, Phone: +81-66629-2405, Fax: +81-6627-3052

Abstract

We apply some variants of evolutionary computations to the Hopfield model of associative memory. In the model, a number of patterns can be stored in the network as attractors if synaptic weights are determined appropriately. One of our goals of this study is to learn the number and distribution of these solutions in weight space, which is still an open problem. To address this issue, we test a method to visualize solutions in high-dimensional space in this paper.

1 INTRODUCTION

In studies using evolutionary algorithms, visualization of high-dimensional space provides various aspects of insight into the search space explored. We can imagine, for instance, convergence/divergence behaviors of a population, topology of a fitness landscape, what does a walk from a random point to the global optimum look like, and so on. The problem of mapping a number of points in multi-dimensional space to points in 2D space with the distances among the original points remaining as much as possible is one of those techniques. Shine et al. [1] and Collins [2] argued such a technique together with other possible alternatives. Collins call this technique *"Sammon Mapping"* after Sammon [3] who proposed this technique originally (Shine et al. call this *"Distance Map"*). Since the technique is an optimization problem, we can employ a genetic algorithm (GA) to solve this problem. Here we employ this technique in somewhat of a different way.

We apply some variants of evolutionary computations to the fully-connected neural network model of associative memory. In the model, a number of patterns can be stored in the network as attractors if synaptic weights are determined appropriately. Although some of the solutions of weights have been found heuristically, the number and distribution of the whole solutions are still unknown issue. As a preliminary stage toward addressing this issue, we apply the Sammon Mapping to visualize our wight space.

Since neither Collins nor Shine gave us any description such as how large dimensionality can be explored, or how many points can be mapped properly, we start by visualizing two known shapes in the space of high dimensionality. Then we apply the technique to our weight space of the neural network model of associative memory.

The remainder of this paper is organized as follows. Since our aim is to visualize solutions of fully-connected neural network model of associative memory, the model is first described in Section 2. Section 3 gives some descriptions of how a GA mapped the points in high-dimensional space to 2D points with distance relation of the points remaining unchanged. After presenting experimental results and discussion in Section 4, we present concluding remarks.

2 ASSOCIATIVE MEMORY

Associative memory is a dynamical system which has a number of stable states with a domain of attraction around them [4]. If the system starts at any state in the domain, it will converge to the stable state. Hopfield [5] proposed a fully connected neural network model of associative memory in which information is stored by being distributed among neurons, and we can retrieve the information from dynamically relaxed neurons' states. In the model, some of the appropriate configurations of synaptic weights give the network a function of associative memory.

The Hopfield model consists of N neurons and N^2

synapses. Each neuron state is either active (+1) or quiescent (−1). When an arbitrary N-bit *bipolar pattern*, a sequence of +1 and −1, is given to the network as an initial state, the dynamical behavior of neurons' states afterwards are characterized by the strengths of the N^2 synapses. The synaptic strengths are called *weights*, and the weight from neuron j to neuron i is denoted as w_{ij} in this paper. Provided the synaptic weights are determined appropriately, network can store some number of patterns as attractors. Hopfield employed the so-called Hebbian rule [6] to prescribe the weights. That is, to store p bipolar patterns ξ^ν:

$$\xi^\nu = (\xi_1^\nu, \cdots, \xi_N^\nu), \quad \nu = 1, \cdots, p,$$

the weight values are determined as:

$$w_{ij} = \frac{1}{N} \sum_{\nu=1}^{p} \xi_i^\nu \xi_j^\nu \quad (i \neq j), \quad w_{ii} = 0.$$

An instantaneous state of a neuron is updated asynchronously (one neuron at a time) as:

$$s_i(t+1) = sgn\left(\sum_{j \neq i}^{N} w_{ij} s_j(t) \right), \tag{1}$$

where $s_i(t)$ is a state of the i-th neuron at time t. If an initial state converges to one of the stored patterns ξ^ν as an equilibrium state, then the pattern is said to be *recalled*. Furthermore, if an initial state chosen from the stored patterns remains unchanged from the beginning, then the pattern is said to be stored as a *fixed point*.

In analyzing the Hopfield model, there have been basically two different approaches: one is to explore *pattern space* searching for attractors under a specific weight configuration, and the other is to explore *weight space* searching for an appropriate weight configuration that stores a given set of patterns. To be more specific, the former is an analysis of the Hamiltonian energy as a function of all the possible configurations of bipolar pattern to be given to the network, where synaptic weights are pre-specified using a learning algorithm, usually the Hebb's rule, so that the network stores a set of p given patterns. In this context, the model for $p = 1$ corresponds to the Mattis model of spin-glass [7], in which the Hamiltonian energy has two minima, while the model for infinitely large p corresponds to the Sherrington-Kirkpatrick model [8], in which the synaptic weights become Gaussian random variables. Analyses of the former type have been made in between these two extreme cases (see Amit [9]). The latter analysis, on the other hand, was addressed by Gardner [10]. She discussed the optimal weight configurations for a *fixed* number of given patterns in terms of the volume of the solutions in weight space, suggesting that the volume shrinks to vanish when p approaches to $2N$. In short, the former approach searches for the optimal pattern configurations which minimize the Hamiltonian energy *in pattern space with the weights being fixed*, while the latter searches for the weight configurations *in weight space* that optimally store a set of given *fixed patterns*.

In this paper, our concern is on weight space where some points give a network a capability to store a fixed set of patterns, and we call these points solutions.

3 SAMMON MAPPING

As Collins [2] wrote, the dimension reduction has been an important technique for visualization of the space of high dimensionality. The Sammon Mapping [3] is one of these techniques. This enables us to map a set of $\mathcal{N}$ points in n-dimensional space to 2-D location data so that the distance information is preserved as much as possible, or as Shine [1] wrote *"so that the n-dimensional distances are approximated by 2 dimensional distances with a minimal error."* This problem is an optimization problem.

Shine et al. [1] and Collins [2] proposed a method to solve this problem by a Genetic Algorithm, as follows. First, the distance matrix whose entries are Euclidean distances between all possible pairs of $\mathcal{N}$ points in the n-dimensional space is calculated. Then tentative $\mathcal{N}$ points in 2-dimensional space are determined representing the original $\mathcal{N}$-points in the n-dimensional space. The distance matrix of these $\mathcal{N}$ 2D points is also calculated, which then will be subtracted from the original n-dimensional distance matrix, yielding an error matrix. A GA is used to minimize this error matrix.

For the sake of simplicity, we assume here the dimension reduction from 2401D space to 2D space. Given $\mathcal{N}$ points in 2401D space

$$X^1, \cdots, X^{\mathcal{N}},$$

where each point X^k is expressed by 2401 coordinates as

$$X^k = (z_1^k, \cdots, z_{2401}^k).$$

Then the square distance between m-th point and n-th point is calculated as

$$R^{mn} = \sum_{i=1}^{2401} (z_i^m - z_i^n)^2$$

The values for all the possible combination of m and n construct a distance matrix. Since the matrix is symmetric with zero diagonal elements, we use the lower triangle elements ($m < n$).

Then we generate $\mathcal{P}$ sets of the 2-dimensional $\mathcal{N}$ points at random, i.e.,

$$\begin{array}{l} \Psi^1(1), \cdots, \Psi^{\mathcal{N}}(1) \\ \Psi^1(2), \cdots, \Psi^{\mathcal{N}}(2) \\ \cdots \\ \Psi^1(\mathcal{P}), \cdots, \Psi^{\mathcal{N}}(\mathcal{P}) \end{array}$$

where the k-th point of the i-th set is represented as:

$$\Psi^k(i) = (\zeta_1^k, \zeta_2^k).$$

Thus the i-th set of these $\mathcal{N}$ points has its distance matrix whose elements are

$$r^{mn}(i) = \sum_{i=1}^{2} (\zeta_i^m - \zeta_i^n)^2$$

The objective function of the i-th sets $f(i)$ can be defined as

$$f(i) = \sum_{m<n} r^{mn}(i) - R^{mn}$$

Starting with a random configuration of $\mathcal{N}$ points in 2-dimensional space, the GA corrects these points one generation to the next applying crossover and mutation[1] to 2D coordinates. The correction is repeated until the error converges to an acceptable minimum.

4 RESULTS AND DISCUSSION

Towards the goal of visualizing solutions in weight space, we apply the dimension-reduction technique to two toy examples, as test functions, in which distribution of the point is known. One is a set of points on a hyper-line, and the other is a set of points in the two separate regions. Our experiments of the fully-connected neural network model of associative memory are carried out on networks with 49 neurons, which implies the weight space is $49^2 = 2401$ dimensional space. So the dimensionality is set to 2401 in this paper.

4.1 Hyper-line

The first test is a visualization of a hyper-line.

[1] We employ uniform crossover [11] and BGA mutation [12] here.

In mapping some points in high dimensionality to points in 2D space, there exists some constraints in general. It is clear, for example, that the four vertices of a tetrahedron in 3D space can never been exactly mapped to four points in 2D space. On the other hand, there is no such constraint in the case of points on a hyper-line. In that sence, hyper-line is a good benchmark for the algorithm.

First, we pick up 120 points that are distributed with equal interval on a diagonal line of the 2401-dimensional hyper-cube. To be more specific, the points are:

$$(x_1^k, x_2^k, \cdots, x_{2401}^k), \quad k = 0, 1, \cdots, 119$$

where

$$x_1^k = x_2^k = \cdots = x_{2401}^k = -1 + k \cdot (2/119)$$

Then they are mapped to 120 points on 2-dimensional space so that the distance relation among the 120 points on the 2401-dimensional space is kept as much as possible. A result is shown in Figure 1 (top). We can see a straight line in the 2-dimensional space. The task to search for an appropriate configuration of 2D points is quite easy in this case. As evolution proceeds, the objective function that took the value 7,560,777 at the start asymptotically approaches the small value around 0.1. The evolution is shown in Figure 1 (bottom).

4.2 Two Hyper-cubes

Next, we proceed to an example in which we can imagine the shape of the region in high-dimensional space. We sampled 60 points randomly from the 2401 dimensional region whose ordinates are all between 0.5 and 1.0 as well as the other 60 points from the region whose ordinates are between -0.5 and -1.0. Namely, points are sampled either from two separate hyper-cubes of the same size. In Figure 2 (top), a result of dimension reduction of these 120 points is shown, together with a point that corresponds to the origin. The evolution of the objective function is shown in Figure 2 (bottom). The value starts with 14,567,428, and eventually approaches 106,125. Though the final value of the objective function is not so small, the ratio of the final value is 0.7% of the initial value. We can clearly see the two separate regions in the 2-dimensional space.

4.3 Hyper-sphere

As stated earlier, multiple configurations of weights give a network a function of associative memory. The

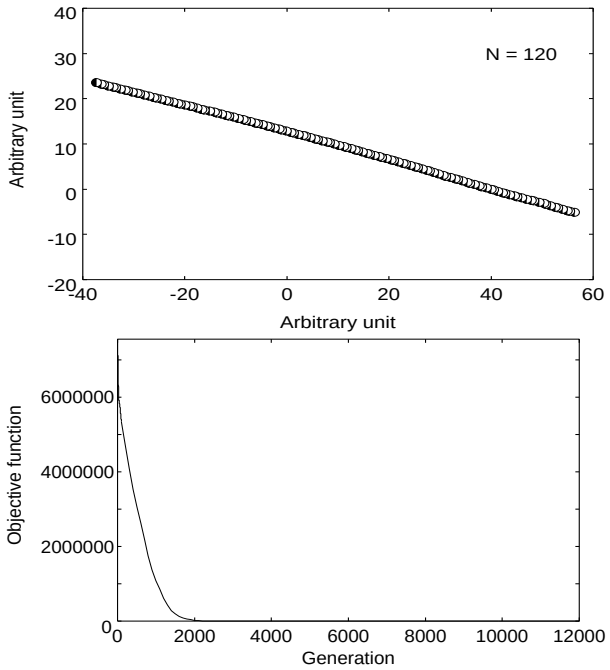

Figure 1: Points mapped to 2D space from 120 points on a diagonal line of the 2401-D space (top), and the time evolution of objective function (bottom).

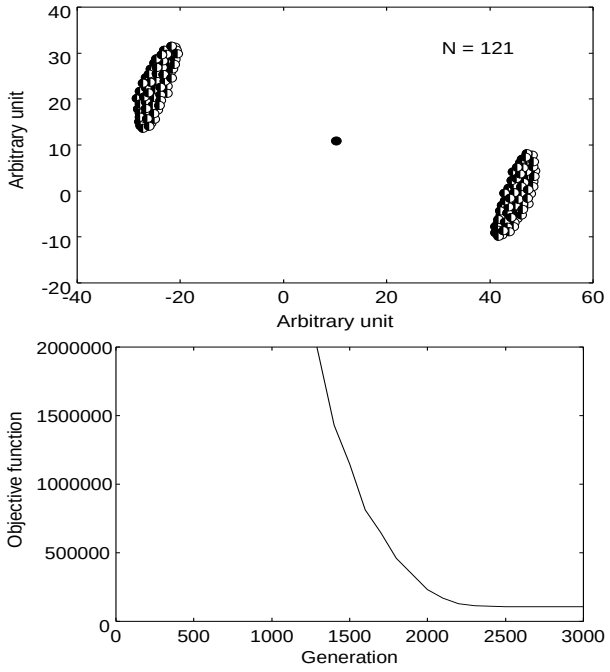

Figure 2: Two regions of the 2401-D space mapped to the 2-D space (top), and the time evolution of objective function (bottom). Filled-in circle • indicates the origin.

number of these configurations is known to be dependent on p, the number of patterns to be stored. Storing just one pattern gives a maximum number of solutions of weights, while as p approaches the number of neurons, all the solutions vanishes [10]. However, the number and distribution as a function of p is still unknown. So far, we have applied some variants of evolutionary algorithms to search for these solutions (see e.g., [13]– [18]). Here, we study the solutions found by the Breeder Genetic Algorithm (BGA) among others, since only this algorithm has been able to search for solutions for a wide range of p (see [19]). Our experiments were carried out on networks with 49 neurons and the BGA found solutions for up to $p = 90$. The solutions that the BGA found are also expected to be different from run to run, as Mühlenbein et al. [12] wrote: *"the BGA mutation scheme is able to optimize many multi-modal functions."* As a preliminary stage of the goal of learning the number and distribution as a function of p, we sampled 30 such solutions for $p = 1$. It is important to note here that since each weights, w_{ij}, can take an arbitrary real value, there are infinite number of equivalent configurations which differ only by scaling factor. In other words, for any scaling factor κ, κw_{ij} works exactly in the same way as w_{ij} in updating neuron states (see equation (1)). So, we normalized the solutions obtained such that they locate on the hyper-sphere of radius 1. We suspect that these normalized solutions for $p = 1$ are distributed uniformly on the surface of the hyper-sphere. We show the results of the 2D points mapped from the 2401-dimensional solution space for the number of solutions $\mathcal{N} = 9, 10$, and 30 in Figure 3 (a), (b), and (c), respectibly. As can be seen in Figure 3 (a), nine 2D points corresponding to the solutions are almost uniformly distributed on the circle whose center corresponds to the origin of 2401-dimensional space, while the distribution of these 2D points are disturbed more or less for $\mathcal{N}$ more than 9.

In Figure 4, we show the time evolution of each objective function for $n = 9$ and $n = 30$. The value for $\mathcal{N} = 9$ starts with 1,758 while the value for $\mathcal{N} = 30$ starts with 37,115 and ended up 8 and 129, respectively. The difference of these values is due to the degree of constraint of the dimension reduction problem.

We also tested the similar experiment with the dimensionality 256 instead of 2401, but we found that the results were almost the same (not depicted here), in that the limit in which points of high dimensionality are properly mapped to 2D space was around $\mathcal{N} = 9$.

Next, we apply the technique to solutions also obtained

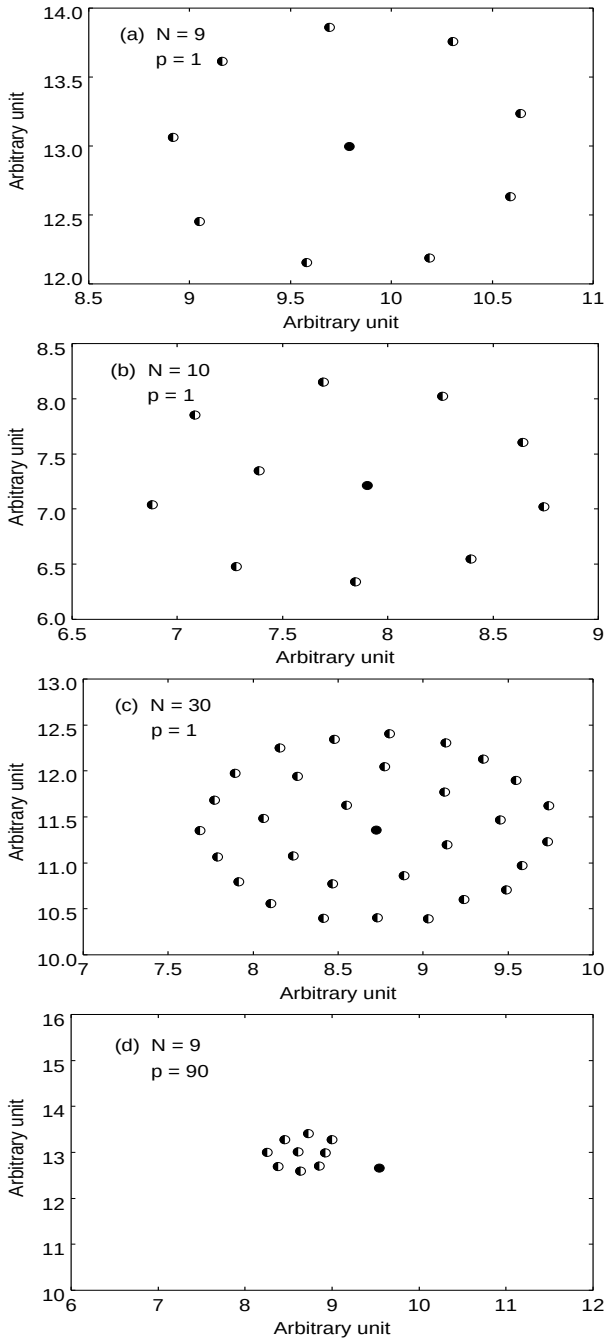

Figure 3: 2D points mapped from solutions in the 2401-dimensional weight space. Filled-in circle ● indicates the origin. The number of mapped points is (a) 9, (b) 10, and (c) 30. (d) Nine 2D solutions that store 90 patterns.

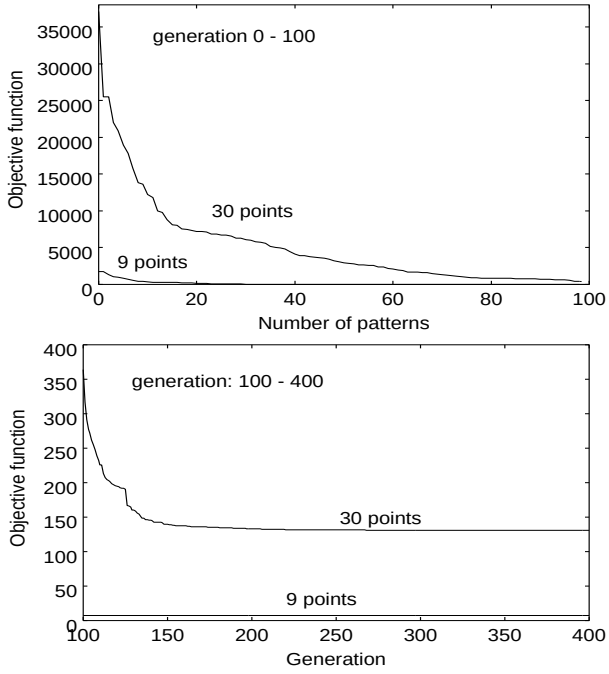

Figure 4: Time evolution of objective function.

by the BGA runs for $p = 90$. This number of patterns to be stored is almost the upper bound of the storage capacity for a network with 49 neurons, and the solutions are expected to be localized into small region of weight space [10]. A result is shown in Figure 3 (d). Though resolution is not so good, we can anyhow imagine the localized solutions.

4.4 Summary

Though we had expected to obtain the limitation of the dimension-reduction technique as to the dimensionality and number of points, we found that results are highly dependent on the initial distribution of the points to be mapped to 2D space. In Table 1, we summarized the values of objective function before a GA run and those obtained when the GA terminated.

The criterion of when we terminate the algorithm is also dependent on how we use the result. Nevertheless the mapping of 120 points from two regions of the 2401D space terminated with large value (10^5), we can imagine how the original points are distributed. On the other hand, the mapping of 30 points from hypersphere, the algorithm terminated with smaller value (10^2) but we cannot imagine the surface of the original sphere any more.

Table 1: Comparison of the values of objective function at the start and when terminated.

source	# of points	objective function	
		start	end
hyper-line	120	7,560,777	0.1
two-regions	121	14,567,428	106,125
hyper-sphere	10	1,758	8
hyper-sphere	11	2,682	10
hyper-sphere	31	37,115	129

5 CONCLUSIONS

We have described a technique to map a high-dimensional search space to 2D points remaining the distance information of the source points being as much as possible. What we are interested in is to visualize weight configurations in weight space that give a network a function of associative memory. Since the topic has not been the subject of extensive research, as Shine et al. [1] pointed out, we preliminary applied the technique to two somewhat of trivial examples to know the limitation of the technique. We have observed that 120 points on a 2401-dimensional hyper line are mapped to a straight line on 2D space, and 120 points that are distributed randomly on two separate hyper-cubes in 2401-dimensional space are mapped to two separate regions of 2D points.

Then we apply the technique to our weight space in which we search for the solutions that give a network a function of associative memory. The dimension of the weight space is 2401 since we use neural networks with 49 neurons. If p, the number of patterns to be stored to the network as an associative memory, is only one, then the solutions are expected to be uniformly distributed in the space. On the other hand, if $p = 90$ that is almost the upper bound of the storage capacity, then the solutions are expected to be highly localized. We normalized these solutions such that they locate on the hyper-sphere of radius one. Then we used the technique to visualize these solutions in 2D space, and found that we can obtain the expected results unless the number of solutions exceeds nine. This limit is not so good, but the result suggests that the solutions are distributed uniformly at random for $p = 1$ and very localized for $p = 90$.

ACKNOWLEDGMENTS

We thank Trevor D. Collins for answering us several questions about Sammon Mapping via e-mail.

References

[1] W. B. Shine, and C. F. Eick (1997) *Visualizing the Evolution of Genetic Algorithm Search Processes.* Proceedings of the IEEE International Conference on Evolutionary Computation, pp.367–372.

[2] T. D. Collins (1997) *Using Software Visualization Technology to Help Evolutionary Algorithm Users Validate their Solutions.* Proceedings of the 7th International Conference on Genetic Algorithms, pp.307–314.

[3] J. W. Sammon (1969) *A Nonlinear Mapping for Data Structrue Analysis.* IEEE Transactions on Computers. C-18(5), pp.401–408.

[4] J. Komlós, and R. Paturi (1988) *Convergence Results in an Associative Memory Model.* Neural Networks, 1, pp.239–250.

[5] J. J. Hopfield (1982) *Neural Networks and Physical Systems with Emergent Collective Computational Abilities.* Proceedings of the National Academy of Sciences, USA, 79, pp.2554-2558.

[6] D. O. Hebb (1949) *The Organization of Behavior. Wiley.*

[7] D. C. Mattis (1976) *Solvable Spin Systems with Random Interactions.* Physics Letters, 56A, pp.421–422.

[8] D. Sherrington, and S.Kirkpatrick (1975) *Solvable Model of a Spin Glass.* Physical Review Letters, 35, pp.1792–1796.

[9] D. J. Amit (1989) *Modeling Brain Function: The World of Attractor Neural Networks.* Cambridge University Press.

[10] E. Gardner (1988) *The Phase Space of Interactions in Neural Network Models.* Journal of Physics, 21A, pp.257–270.

[11] G. Syswerda (1989) *Uniform Crossover in Genetic Algorithms.* Proceedings of the 3rd International Conference on Genetic Algorithms, pp.2–9.

[12] H. Mühlenbein, and D. Schlierkamp-Voosen (1996) *Predictive Models for the Breeder Genetic Algorithm I. Continuous Parameter Optimization.* Evolutionary Computation, 1, pp.25–49.

[13] A. Imada, and K. Araki (1995) *Genetic Algorithm Enlarges the Capacity of Associative Memory.* Proceedings of the 6th International Conference on Genetic Algorithms, pp.413–420.

[14] A. Imada, and K. Araki (1996) *Lamarckian Evolution of Associative Memory.* Proceedings of IEEE International Conference on Evolutionary Computation, pp.676–680.

[15] A. Imada, and K. Araki (1997) *The Baldwin Effect on the Evolution of Associative Memory.* Proceedings of the 3rd International Conference on Artificial Neural Networks and Genetic Algorithms, pp.354–358.

[16] A. Imada, and K. Araki (1997), *Searching Real-Valued Synaptic Weights of Hopfield's Associative Memory using Evolutionary Programming.* Proceedings of the 6th Annual Conference on Evolutionary Programming, Springer-Verlag, Lecture Notes in Computer Science, No.1213, pp.13–22.

[17] A. Imada, and K. Araki (1997), *Application of an Evolution Strategy to the Hopfield Model of Associative Memory.* Proceedings of the IEEE International Conference on Evolutionary Computation, pp.679–683.

[18] A. Imada, and K. Araki (1997), *Random Perturbations to Hebbian Synapses of Associative Memory using a Genetic Algorithm.* Proceedings of International Work-Conference on Artificial and Natural Neural Networks, Springer-Verlag, Lecture Notes in Computer Science No.1240, pp.398–407.

[19] A. Imada, and K. Araki (1997) *Evolution of Hopfield Model of Associative Memory by the Breeder Genetic Algorithm.* Proceedings of the 7th International Conference on Genetic Algorithms, pp.784–791.