Torus Self-Organizing Map for genome informatics

Shinichi Horata, Toshimichi Ikemura and Tetsuyuki Yukawa
Hayama Center for Advanced Research,
The Graduate University for Advanced Studies (Sokendai),
Hayama, Kanagawa, 240-0193, Japan
horata_shinichi@soken.ac.jp, toshimichi_ikemura@soken.ac.jp and yukawa@soken.ac.jp

Abstract - *To address the problems of clarifying interspecies difference of genome sequences, Self-Organizing Map (SOM) was used as a classification method concerning species and phylotype families. This method made clusters on a map and visualized interspecies difference of genome sequence characteristics. To analyze genome sequences of a wide range of genomes with the SOM algorithm collectively, the frequency of appearance of the nucleotide word with a specific length, i.e. oligonucleotide, was introduced as the input data. This input data reflected sequence characteristics of individual genomes effectively. In order to clarify relation of clusters with species and phylotype families, we analyzed size and position of each cluster on a SOM. In this paper, we attempted to perform the torus map algorithm, which provides independence of the position on the map, and compared with the results obtained by the plane map algorithm. To investigate effects derived from map edges in detail, we performed the torus SOM algorithm for 8697 100-kb genomic sequence fragments, which derived from 206 species of bacteria, 21 species of archaea and 6 species of eukaryota. The numerical results suggested that the torus map could make clear the relation between clusters and characterize features of the phylotype families after learning.*

Key words - bioinformatics, genome informatics, oligonucleotide frequency, genome sequence

1 Introduction

Self-Organizing Map (SOM) learning algorithm makes clear the features of the input data by the competitive learning [1, 2, 3, 4, 5, 6]. This algorithm is based on unsupervised competitive learning and the neuron units are located on a two-dimensional grid space. Based on the reference data associated to each neuron unit, the high-dimensional input data space is mapped onto this grid space and clusters appear on the map after learning. Therefore, the SOM algorithms is considered as a clustering tool on a two-dimensional map space.
For the bioinformatics application, the SOM learning algorithm was shown to be one of the most powerful methods to classify genomic sequence fragments according to species with high accuracy [7, 8, 9]. The clusters of sequence fragments, generated by the SOM algorithm, were comprised by sequences from the same or closely related genomes.
For the next biological application, we characterized individual clusters in detail because the comparison between clusters is thought to provide detailed information of distinctions

among distinct genomes. It should be noted, however, that the ordinal plane map algorithm inevitably includes ambiguity concerning size and position of clusters. This is caused by the difference of the location of units, because the neighborhood sets on edges of the map are smaller than those on the center (Fig. 1).

We thus carried out the torus map algorithm instead of the plane map one, in order to avoid effects of the difference derived from neighborhood sets. In case of the torus map, the size of neighborhood sets are equivalent across the entire units on the map. Accordingly, the effects of learning spread to all units on the map, and the equivalence of the information on clusters are assured, and the information between clusters can be compared correctly on the torus map. In other words, size and position of clusters can reflect accurately the biological features, which have been established during evolution.

In this paper, we developed the torus map as a novel bioinformatics strategy and compared with the conventional SOM method, *i.e.* the plane map method. To conduct the competitive learning for genome sequences, we counted occurrence frequencies of the oligonucleotide word with a specific length. The genome sequence is the string array constituted with 4 letters, ATGC, and the above word counting method was introduced to the SOM analysis as described previously by Ref. [7, 8, 9]. To analyze relation between clusters, we focused on several hundred prokaryotic and eukaryotic genomes, which represented wide varieties of phylotype families.

2 Learning algorithm

The SOM algorithm requires two layers of processing units: the first is the layer for an input vector data, and the other is an output layer which creates a self-organizing map. The units on the output layer compete with each other for the right to be declared the winner. From the evaluation of the high-dimensional Euclidean distance between the input vector and the reference vector, the winner unit is determined. Then the training for the wining unit is carried out with the information of the neighborhood unit sets, and the sets of the similar units are gathered around each other gradually. In the SOM algorithms, the size of neighborhood around the winning unit decreases, as the training process is proceeded. This means that initially the large number of units are related, and only the small number of units are related finally. The reference vector of the wining units is adjusted according to the learning rate which decreases gradually with the training. After the learning, the SOM algorithm gathers the same or close input vectors and makes clusters on the output layer, which generate the so-called self-organizing map.

It should be noted that the difference of the topology of the output layer leads to the difference of learning. In the conventional method, the units of the output layer form a two-dimensional square- or hexagon- grid space. Inevitably, the size of neighborhood sets on edges of the map are smaller than the ones on center, (Fig. 1), and this causes dispersion of learning chances for individual units. Therefore, the size and the position of each cluster on the output layer depend on the location of units, and it becomes difficult to compare directly the properties between the clusters, although size and position of the clusters may include the information related to various properties of distinct genomes. We expected that the torus SOM learning algorithm could avoid the difference of neighborhood sets and make clear the relation between clusters, because the output layer in the torus SOM algorithm has no edges (Fig. 1). This

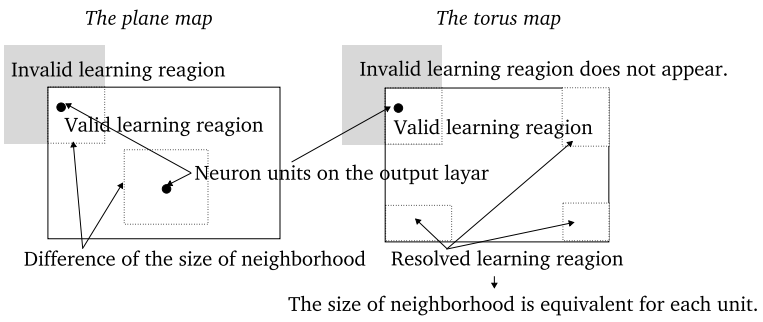

Figure 1: The schematic image of size of neighborhood set for the plane map (left hand side) and for the torus map (right hand side).

algorithm may solve the problems concerning the location of clusters.
Next, we explain actual procedures of the present biological application of the SOM learning algorithm. Whole genome sequences for a wide rage of species have been determined recently, and it has become increasingly important to characterize the massive amounts of genome sequence data currently available. The authors of Ref. [7, 8, 9] showed usefulness of the SOM application for the phylogenetic classification by analyzing wide varieties of prokaryotic and eukaryotic genome sequences. In this paper, we modified their plane map algorithm to the torus map one in order to remedy the dispersion of learning chances dependent on the map position. Using the torus SOM learning algorithm, the relation between clusters can be compared more directly. This relation may reflect the information related with the genome sequence properties of individual species, which have established during the course of evolution. For the SOM learning algorithm, we should introduce the input vectors which can reflect the genome sequence features properly, and in Ref. [7, 8, 9], the way to make the input vector on the plane map algorithm was proposed. The method was based on the frequency of the appearance of the oligonucleotide word with a specific length on a sequence. The genome sequence is a letter string which is constituted by the combination of 4 letters. The composition of the 4 letters differs among species, and the frequency of the oligonucleotide words on the sequence is an appropriate candidate of the input vector. The way to count the frequency of the oligonucleotide word was the following: first we refer to the word with a specific length on the sequence, second we update the frequency of the oligonucleotide word, next we shift the word pointer with 1 letter on the sequence and continue to count. After the word counting, the $\dim(x)$-dimensional vector for each sample is constructed, which is so-called word-counting-vector abbreviated to WCV, $\overrightarrow{x} = \{x_\mu, \mu = 1, ..., \dim(x)\}$. For example, when the sequence is given as "ATGGATAGCGTA", WCV x is computed as ,

$$\begin{aligned} x(AA) = 0, x(AT) = 2, x(AG) = 1, x(AC) = 0, \\ x(TA) = 2, x(TT) = 0, x(TG) = 1, x(TC) = 0, \\ x(GA) = 1, x(GT) = 1, x(GC) = 1, x(CA) = 0, \\ x(CA) = 0, x(CT) = 0, x(CG) = 1, x(CC) = 0, \end{aligned} \tag{1}$$

for counting with the 2 word length. To shift with 1 letter continuously, WCV is generated reflecting a series of the genome sequence. The dimension of WCV, $\dim(x)$, is given as,

$$\dim(x) = 4^n, \tag{2}$$

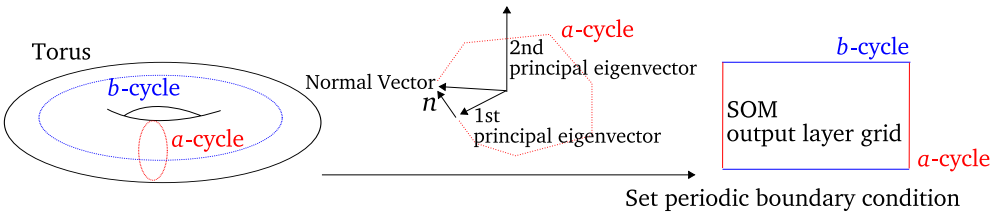

Figure 2: The method to construct the torus grid.

where n denotes the length of the counting word. For the SOM learning algorithm, WCV are normalized as $|x| = 1$ in order to avoid the difference of the length of WCV.
To lead quick convergence for learning, we provided the initial reference vector, $w_\mu(i,j)$ which is associated on each unit of the output layer at grid (i,j). The initial reference vector of the plane map, $w_\mu(i,j)$, was computed based on the principal component analysis (PCA) as described in Ref. [7, 8]. In a similar way to the plane map algorithm, we used the 1st principal eigenvector $p_\mu^{\rm 1st}$ and the 2nd principal eigenvector $p_\mu^{\rm 2nd}$ for the coordination on the torus. Since the torus map has the periodic boundary condition for each edge, we should construct the initial reference vector holding the periodicity. To build up the grid on the torus, we thought the analogy of the plane map to use $p_\mu^{\rm 1st}$ and $p_\mu^{\rm 2nd}$ as the orthogonal two-axis of the grid. Then we introduced the 1st principal component $p_\mu^{\rm 1st}$ as the a-cycle of the torus and the 2nd principal component $p_\mu^{\rm 2nd}$ as the b-cycle of the torus respectively, (Fig. 2). In order to construct the each cycle of the torus, we used the normal vector $n_\mu(i,j)$ generated by the vector product of $p_\mu^{\rm 1st}$ and $p_\mu^{\rm 2nd}$,

$$n_\mu = \epsilon_{\mu\nu\rho} p_\nu^{\rm 1st} p_\rho^{\rm 2nd}, \tag{3}$$

where $\epsilon_{\mu\nu\rho}$ denotes the 4^n-dimensional anti-symmetric tensor and μ, ν, ρ denote the suffix of the 4^n-dimensional Euclidean space. Using the normal vector n_μ, we rotated $p_\mu^{\rm 1st}$ and $p_\mu^{\rm 2nd}$ around each other as,

$$p_\mu^{\rm 1st}({\rm next}) = p_\mu^{\rm 1st}({\rm prev}) + n_\mu, \tag{4}$$

and the trajectory of the normal vector n_μ becomes the a-cycle and b-cycle of the torus grid, (Fig. 2).
Thus we can construct the periodic vector $c_\mu^{a-{\rm cycle}}(i,j)$ and $c_\mu^{b-{\rm cycle}}(i,j)$ which is constituted as the combinations of $p^{\rm 1st}$ and $p^{\rm 2nd}$ on the entire units on the output layer as,

$$\begin{aligned}
c_\mu^{a-{\rm cycle}}(i,j) &= c_\mu^{a-{\rm cycle}}(i,j)(p^{\rm 1st}, p^{\rm 2nd}), \\
c_\mu^{b-{\rm cycle}}(i,j) &= c_\mu^{b-{\rm cycle}}(i,j)(p^{\rm 1st}, p^{\rm 2nd}), \\
c_\mu^{a-{\rm cycle}}(-i,-j) &\cong c_\mu^{a-{\rm cycle}}(I-i, J-j), \\
c_\mu^{b-{\rm cycle}}(-i,-j) &\cong c_\mu^{b-{\rm cycle}}(I-i, J-j),
\end{aligned} \tag{5}$$

where I, J denote the size of the grid, $c_\mu^{a-{\rm cycle}}(i,j)$ and $c_\mu^{b-{\rm cycle}}(i,j)$ are generated by the rotation around $p^{\rm 2nd}$ and $p^{\rm 1st}$ respectively. Using the periodic vectors $c_\mu^{a-{\rm cycle}}(i,j)$ and $c_\mu^{b-{\rm cycle}}(i,j)$, the initial reference vectors are introduced as,

$$w_\mu(i,j) = \bar{x}_\mu + 5\sigma^{\rm 1st} c_\mu^{a-{\rm cycle}}(i,j) + 5\sigma^{\rm 2nd} c_\mu^{b-{\rm cycle}}(i,j), \tag{6}$$

where $\bar{x}_\mu$ denotes the average vector of the whole input WCV, $\sigma^{1\mathrm{st}}$ and $\sigma^{2\mathrm{nd}}$ are the variance along the principal eigenvector $p^{1\mathrm{st}}$ and $p^{2\mathrm{nd}}$. In order to examine the initial reference vector Eq. 6, we compared with the torus SOM result using random initial reference vectors because the initial condition of neurons may relate to the position of clusters.
We associated WCV to the unit of the input layer as the array of M-samples $\{x^m_\mu, \mu = 1, ..., 4^n, m = 1, ..., M\}$. For the unit of the output layer, we prepared the initial reference vector (Eq. 6), and updated its vector at the training process as,

$$w_\mu(i,j)_{\mathrm{next}} = w_\mu(i,j)_{\mathrm{prev}} + \alpha \sum_{\mathcal{R}} \left(x_\mu - w_\mu(i,j)_{\mathrm{prev}}\right), \tag{7}$$

where α is the learning rate, $\mathcal{R}$ denotes the region of the neighborhood set, the renewal vector $w_\mu(i,j)_{\mathrm{next}}$ also satisfy the periodic boundary condition and each neuron is updated based on batch-learning algorithm. The region $\mathcal{R}$ is the square region framed by the grid within the reach of $-\beta$ to β, because the frame of torus is built with the orthogonal grid in this case.
After the update of the reference vector, the winning unit is determined from the evaluation of the 4^n-dimensional Euclidean distance, $|w_\mu(i,j) - x_\mu|$. The parameter α and β are updated after each training sweep as,

$$\begin{aligned} \alpha &= \min(0.01, \alpha_0(1 - t/T)), \\ \beta &= \min(1, \beta_0 - t), \end{aligned} \tag{8}$$

where t denotes the number of the sweep times in T times, α_0 is the initial learning rate and β_0 is the initial size of neighborhood, respectively.

3 Numerical experiment

Using the torus map, we analyzed 206 species of bacteria, 21 species of archaea and 6 species of eukaryota, for which the whole genome sequences can be obtained from DNA Data Bank of Japan (DDBJ)[10]. In the present study, we calculated WCV from genome sequences fragments with a size of 100,000 nucleotides (100 kb), and specified the dimension of WCV as $4^4 = 256$, *i.e.* the word length is 4.
Fig. 3 shows the learning result of the plane and the torus map after 200-times training. On the numerical results of the plane map, the two clusters of the eukaryote sequences are located near the two corner and the clusters of the archaea are placed near the edges. We expected that on the plane map these clusters were forced out to corners and edges by the stress from the units associated to the bacteria. In case of the torus map, the clusters of the archaea and the eukaryota can be bounded by the units associated to the bacteria because of the basis of the algorithm. All clusters on the map can get the same chance of learning and its location is determined by the dynamical balance of each cluster. We consider that the torus map provides the way to compare between clusters properly. While, in this numerical experiments, we used WCV counting of one specific sequence length (100,000 nucleotides), the numerical results using WCV counting sequences with difference length including the whole sequence was consistent to the present WCV result.
Next, we discuss about the structure within one cluster. Because each cluster was distinguished by difference of phylogenetic families, this may also enable us to break clusters down into its constituent sub-families. In Fig. 4, we show the decomposition image of one of the

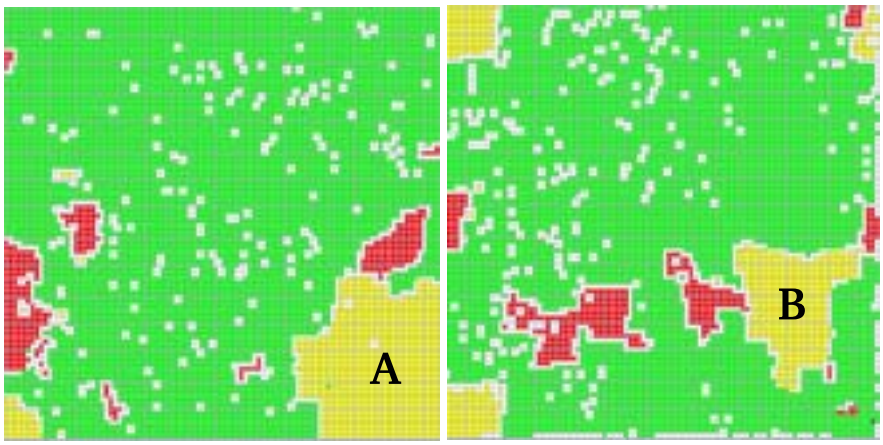

Figure 3: We show the output layer map associated with the input data. The left hand side is the result using the plane map and the other side is the result using the torus map. Each neuron is located on the square lattice plaquette. For the torus map, the space is cut open to the square grid for the visualization and the lower and upper edges are identified as are the right- and left-hand edges. The green box denotes species of the bacteria sequences, and the red and yellow box show the archaea and the eukaryota genome sequences, respectively. The symbol "A" and "B" refer to the clusters shown in Fig. 4.

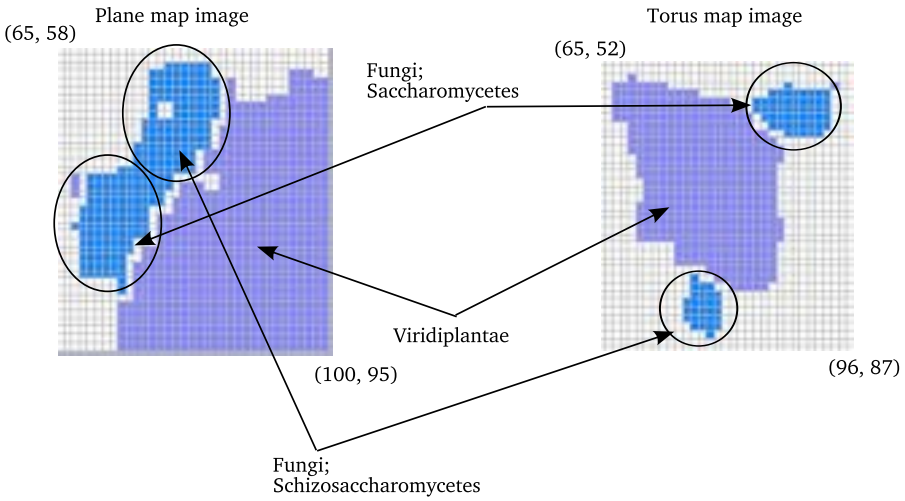

Figure 4: The focus image of the cluster in Fig. 3. The left hand side image is the cutting cluster image of the plane map; the cluster denoted "A" in Fig. 3 which is located within the region (65,58) to (100,95). The other side is the cutting image of the torus map; the cluster denoted "B" in Fig. 3 which is located within the region (65,52) to (96,87).

Figure 5: We show the output layer map associated with the all input WCVs after re-learning including sequences from all three phylotypes. The left hand side is the result using the plane map and the other side is the result using the torus map. The color legends are the same as Fig. 3.

eukaryota cluster, which includes the information about the fungi and the viridiplantae. The two different yeasts, *Saccharomycetes* and *Schizosaccharomycetes*, were clearly separated on the torus SOM but were continuously distributed on the plane SOM. This numerical result indicates that the cluster of the torus map can conserve more informations about the difference of species than the structure of the plane map. The torus map algorithm is thought to be a suitable way to clarify the relation between clusters and to study the structure of the cluster, which should reflect the genome characteristics of individual species and phylogenetic families. Similar structures were found for other clusters (our unpublished results).

As another attempt to extract detailed characteristics of distinct phylogenetic groups, we introduced a re-learning method. In this new analysis, we performed first the SOM learning only using bacterial genome sequences. Using the resulting map as the initial reference vector set, we then performed an additional SOM learning using all sequences derived from three phylotypes, (bacteria, archaea, and eukaryota). This re-learning method will become important to compare newly determined sequences with known sequences sets and to exact efficiently the features of the newly included sequence set. Fig. 5 show the clusters on the plane and the torus map, which are given by the reference vector after the re-learning. In case of the plane map, the additional input data, *i.e.* archaea and eukaryota genome sequences, is pushed out of the center, and this situation is same as the previous results in Fig. 3. On the torus map, all of clusters for archaea and eukaryota genome sequences can be engulfed by bacteria units. This numerical result indicates that the torus map algorithm can classify individual genomic sequences accurately and hold well the information concerning phylogenetic families.

Next, we measured the distance of the output reference vector between clusters, because each cluster in the torus map algorithm is located at the position determined by the dynamical balance of the difference of species. In order to calculate the distance of clusters, we picked up the center unit of clusters, which has the closest Euclidean distance for all reference vectors of the units belonging to the cluster. Since the distance between the most closely-related phylotype clusters is shortest, the distance between center units of clusters reflected the phylogenetic relations. The torus algorithm can provide a useful bioinformatics strategy to compare between phylogenetic families on the self-organizing map.

4 Conclusions

We developed a torus SOM map algorithm to clarify the genomic sequence characteristics of individual species and phylogenetic families. In order to improve the convergence for learning, we introduced the initial background reference vector on the each neuron unit as follows. By the analogy of the plane map algorithm [7, 8, 9], the initial background reference vector was constructed using the 1st and 2nd principal components (Eq. 6). Both of two algorithms, *i.e.* the plane and the torus map, provided clear classification of sequences according to species on the SOM map space. The torus SOM algorithm may be a suitable way to compare between clusters on the map, because size and position of clusters are reflected properly by the phylogenetic relations. For the future problem, it is necessary to analyze the relation and the structure of the clusters constructed for a massive amount of all genomic sequences currently available using the torus SOM algorithm. The obtained information should clarify the detailed relation of a wide range of phylogenetic families and the evolution of their genome systems. As the SOM algorithm is one of the neural network, we expect that the biological network established during the evolution might be characterized by the SOM method.

References

[1] Kohonen, T. (1982) Self-organized formation of topologically correct feature maps. *Biol. Cybern.* **43**, p. 59-69.

[2] Kohonen, T. (1984) *Self-Organization and Assoxiative Memory*. Springer Series in Information. Sciences 8.

[3] Kohonen, T. (1988) Learning vector quantization. *Neural Networks* **11**, 303,

[4] Kohonen, T. (1990) The self-organizing map. *Proc. IEEE* **78**, p. 1464-1480.

[5] Kohonen, T. (1995) *Self-Organizing Maps*. Springer Series in Information Science 30.

[6] Kohonen, T., Oja, E., Simula, O., Visa, A., and Kangas, J. (1996) Engineering applications of the self-organizing map. *Proc. IEEE* **84**, p. 1358-1384.

[7] Abe, T., Kanaya, S., Kinouchi, M., Ichiba, Y., Kozuki, T., and Ikemura, T. (2003) Informatics for unveiling hidden genome signatures. *Genome Res.* **13**, p. 693-702.

[8] Abe, T., Kozuki, T., Kosaka, Y., Fukushima, A., Nakagawa, S., Ikemura, T. (2003) Self-organizing map reveals sequence characteristics of 90 prokaryotic and eukaryotic genomes on a single map. *WSOM 2003*, p. 95-100.

[9] Kanaya, S., Kinouchi, M., Abe, T., Kudo, Y., Yamada, Y., Nishi, T., Mori, H., and Ikemura, T. (2001) Analysis of codon usage diversity of bacterial genes with a self-organizing map (SOM): characterization of horizontally transferred genes with emphasis on the E. coli O157 genome. *Gene*. **276**, p. 89-99.

[10] Fumoto, M., Miyazaki, S. and Sugawara, H. (2002) Genome Information Broker (GIB): data retrieval and comparative analysis system for completed microbial genomes and more. *Nucleic Acids Res.* **30**, p. 66-68.