Preventing Topological Distortion in Self-Organizing Maps through Surface Normal Estimation: An Average Cross-Product Approach to Enhance Local Neuron Interactions

Lee (Jiacheng) Li
The University of Sydney, Sydney, Australia
jili5802@uni.sydney.edu.au

***Abstract*—Self-Organizing Maps (SOMs) offer a mechanism for generating low-dimensional representations of high-dimensional data while preserving the topological relationships between data points. However, errors in initialization or training can sometimes yield distorted, twisted maps, which can significantly degrade the quality of the representation and undermine the utility of the SOM in applications such as data visualization, dimension reduction, and pattern recognition. This research aims to address this issue by developing an algorithm that leverages local surface normal estimations to prevent such distortions. The method estimates the surface normal for a neuron in the SOM, a process grounded in the cross product concept of linear algebra. By incorporating these estimations into the SOM learning process, the algorithm can effectively mitigate global distortions in the SOM. The effectiveness of this algorithm is demonstrated through three examples: two clusters, a spiral, and a "C"-shaped datasets. The literature review contextualizes this work within the broader research field. Results demonstrate that the proposed algorithm enhances the performance of the SOM in data visualization, dimension reduction, and pattern recognition tasks, effectively preventing map twisting and distortions, and thereby improving the quality and utility of the resulting representations.**

***Index Terms*—Self-Organizing Map, Topological Distortion, Surface Normal**

I. Introduction

A. Background and Context

An SOM could be considered an unsupervised artificial neural network, notwithstanding training, using competitive learning rather than error-correction learning as used elsewhere by other artificial neural networks. The SOM was originally conceived by the Finnish professor Teuvo Kohonen circa 1980 and can also be referred to as a Kohonen Map or Kohonen Network [1]. The SOM or Kohonen Map is based on a computationally convenient abstraction of biological models of neural systems from circa 1970, being inspired by the way that neurons in the brain organize themselves based on inputs these neurons receive. By focusing on the essential features of the neural system within the brain, an SOM algorithm can provide a powerful tool for data analysis and pattern recognition that can be applied to a wide range of applications; for example, from image and speech recognition to natural language processing and bio-informatics [2].

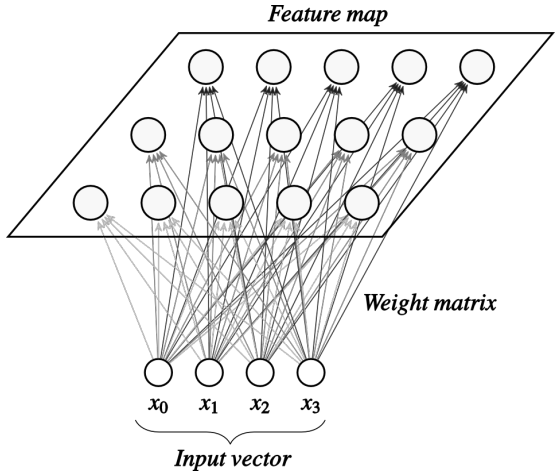


Fig. 1: Architecture of a SOM network Source: [3]

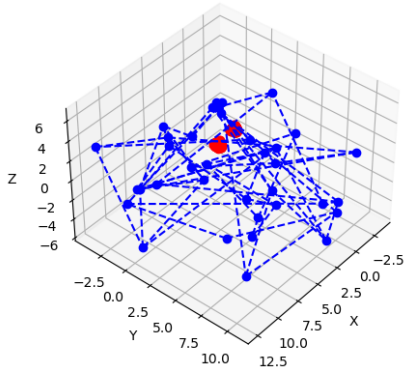
The network architecture and data processes used to model nervous systems can be divided broadly into three categories: Feedforward networks, feedback networks and competitive networks (also referred as unsupervised or self-organizing networks). The SOM belongs to the last group, which is a sheet-like artificial neural network where the neurons of which become specifically tuned to various input data patterns or classes of patterns through an unsupervised learning process. In a basic version, only one neuron or local group of neurons at a time gives an active response to the input data. The location of the response tends to become ordered as if some meaningful coordinate system or structure for different input features were being created over the network. The spatial location or coordinates of a neuron in the network corresponds to a particular domain of input data patterns; where each neuron or local neurons act as a separate decoder for the same input data [4]. Accordingly, competitive networks can be used for pattern recognition and data visualization applications, having a broad range of across numerous commercial, medical and industrial fields. The accuracy of competitive networks is crucial for the above and this Research Report gives detailed consideration towards research that will enhance such accuracy.

B. Research Question

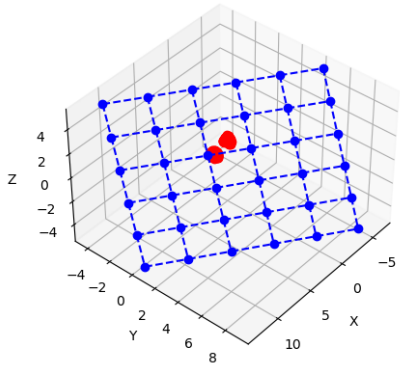
The objective of this research paper is to develop an algorithm that can prevent the occurrence of distorted or twisted maps in an SOM by utilizing local surface normal estimation. Distorted or twisted maps can occur in a SOM as a consequence of complex and nonlinear structures within the input data, errors in initialisation of neurons or learning processes. This can ultimately affect the performance and accuracy of a SOM in data visualisation, dimension reduction or pattern recognition. In the development of an algorithm that can prevent the occurrence of distorted or twisted maps in a SOM, this research can contribute to the improvement of SOM-based methods and their applications. The application of local surface normal estimations can provide a means to detect and correct distortions within the SOM leading to more accurate and reliable representations of complex data.

C. Methodology

The algorithm to be derived would be based on local surface normal estimated by average cross product to prevent map twisting as outlined in the Abstract above, whilst maintaining features such as topology and inter-relationships in concert with the original multi-dimensional data. Algorithm derivation is primarily based on the estimation of surface normal vectors for neurons during the learning process of Self-Organizing Maps (SOMs). The surface normal vector for a given neuron is calculated using mathematical concepts such as cross product and dot product, being determined by averaging the pair-wise cross product of neuron input weights. For the same group of neurons, two surface normal vectors are determined: one prior to the update and one subsequent to it. The calculation of an angle-weighted factor involves determining the angle between these two normal vectors. A minimal angle between the two vectors suggests that the local neurons are approximately oriented in the same direction, implying minimal distortion. On the contrary, a large angle indicates the presence of distortion or a twist in this update. An angle-weighted factor can thus be employed to mitigate such distortion or twisting.



(a) Random Initialization of a SOM



(b) PCA Initialization of a SOM

Fig. 2: Comparison of random initialisation(RI) and data analysis-based Initialisation

II. Literature Review

A. Initialisation of SOM

The quality of learning within a SOM is largely influenced by initial conditions, such as the initial weight of neurons, neighbourhood function, learning rate, sequence of training vectors and the number of iterations of learning steps. Numerous Initialisation methods have been variously proposed and can be divided into two groups viz random initialisation (RI) and data analysis-based Initialisation [5]. This research considers performance accuracy of the SOM in terms of the quality of learning, because of a SOM between the random initialisation (RI) and principal component analysis (PCA) methodologies.

Random Initialisation is a preferable initialisation approach due to its simplicity, notwithstanding this approach might not be an efficient methodology for producing a rapid convergence to a stable state of a SOM [6]. In a Som, a stable state is referred to as when the model's learning has converged and the positions of the neurons have settled and do not change significantly with additional training. There is various way of randomly initialising neurons, for example random component which each component of a neuron is chosen randomly from the range of values observed in a data or random selection which uses sample data, taking advantage of a samples automatically lie in the same part of the input space of data [7] [8].

PCA belongs to linear projection data-based initialisation, which is an unsupervised approach whose aim is to find the most salient features. In the context of PCA, the most salient features are represented by the principal components. These are new synthetic variables that are constructed as linear combinations of the original variables. The first principal component accounts for the largest possible variance in the data, the second principal component (which is orthogonal to the first one) accounts for the second largest variance, and so on. Therefore, PCA can be used to identify the directions in which the data varies the most, being useful for tasks like feature reduction where the goal is to reduce the dimensionality of

the data while preserving as much of the data's structure and variability as possible.

B. Competitive Learning Algorithm

The significant derivation by Kohonen (1980) was to introduce a systematic data visualization model that is composed of at least two interacting subsystems of different natures. One of these subsystems is a competitive neural network that implements the winner-take-all function, along with another subsystem that is controlled by the neural network and which modifies the local synaptic plasticity of the neurons in learning [1]. Consider first data items that are n-dimensional Euclidean vectors:

$$x(t) = [\xi_1(t), \xi_2(t), \xi_3(t), ..., \xi_n(t)]$$

Here $t|$ is the index of the data item in a given sequence. Let $i|th$ neuron be

$$m_i(t) = [\mu_1(t), \mu_2(t), \mu_3(t), ..., \mu_n(t)]$$

where $t|$ denotes the index in the sequence in which the neurons are generated. The new value of a neuron $m_i(t+1)$ is computed iteratively based on previous value $m_i(t)$ and new data item $x(t)|$ as following:

$$m_i(t+1) = m_i(t) + \alpha(t)h_{ci}(t)[x(t) - m_i(t)]$$

where $\alpha(t)$ is a scalar factor also named "learning rate" that defines the size of the correction, indicating how much the neuron will learn from this data item [1]. Additionally, a learning rate is typically monotonically decreasing with the step index $t.|$ As the training of the SOM progresses, each neuron's learning capacity gradually diminishes with an increase in the index 't', suggesting that the final SOM is reaching a state of convergence and stability. $h_{ci}(t)$ is called the neighborhood funtion which will be explained in more detail in following.

A "winning" neuron can be defined by the dissimilarity of x and m, which is generally based on Euclidean distance; "winning" neuron M_c has

$$d(x, M_c) = min\{d(x, M_i)\}$$

Here, the distance (d) can be seen to be the minimum value amongst all distances between a data item and other neurons; physically, the "wining" neuron is the closest neuron (Euclidean distance) to the data sample in a coordinate system.

The formation of ordered maps relies on an essential condition where neurons involved in learning are not affected in isolation, but rather as interconnected subsets that share a topological relationship. This means that a similar type of correction is applied to each subset. As the learning steps for the algorithm progress, these selected subsets will include neighbouring neurons, thereby encompassing a wider range of neural connections. In the present process the algorithm seeks to enforce lateral interaction directly in a general form, for arbitrary underlying network structures by defining a neighbourhood set Nc around a winner neuron c. At each learning step, all neurons within the neighbourhood set will be updated, whereas neurons outside Nc are left intact.

The Neighbourhood Function determines the rate of change of the neighbourhoods around the winner neuron, which is based on the distances between winning neuron and one of the neighbours; with the furthest distance from the wining neuron giving rise to minimal influence on one of the neighbours. A mathematical definition for neighbourhood function could be considered as:

$$h_{ij}^c = \begin{cases} \beta(t), & \text{if } (i,j) \in N_c, \\ 0, & (i,j) \notin N_c. \end{cases}$$

Where h represents the value produced by the neighbourhood function, indicting the level of influence imposed on the neighbours [9] and N_c is a set containing all neighborhood neighbors. Where equation above is one possible selection for neighbourhood function. One particular SOM will only have one neighbourhood function, where the training outcomes for SOM procedure are influenced by the neighbourhood function.

The width or radius of Nc can be time-variable; where it is preferrable that such radius is wide at the start of training and monotonically decreasing with time. A larger initial radius would allow for a broader exploration of the input space, thereby enabling the SOM to capture a wider range of input patterns and ensuring that the mapping of input data samples changes to initialise itself in a more representative manner. Reducing the radius over time encourages convergence and the formation of ordered maps. As the radius shrinks, neighbouring neurons become increasingly specialized in responding to similar input patterns, leading to a more organised and structured representation of the input space where nearby neurons exhibit greater similarity in their response characteristics.

C. Quality of learning

For a Self-Organizing Map (SOM) to serve as an effective model, it must maintain the input data's topology and neighborhood structures, while concurrently aligning well with the data. In this context, topology refers to the spatial relationships among data points, whereas neighborhoods denote clusters of data points situated within the same region of the input space. A model that fits the data will accurately encapsulate inherent relationships without incorporating noise.

The selection of quality metrics for this evaluation was made considering their relevance and widespread use. The conventional method, introduced by Kohonen, is the quantization error(QE), calculated by aggregating the distances between the nodes and corresponding data points. On the other hand, the topographic error quantifies the capability of a SOM to preserve local topological characteristics within a low-dimensional output space.

Kohonen suggested QE as the basic quality measure for evaluating self- organizing maps [4].The value for a map is calculated using

$$QE(M) = 1/n \sum_{n}^{i=1} ||\phi(x_i) - x_i||$$

where n is the number of data points in the training data and $\phi : D- > M$ is the mapping from the input space D to the SOM M [10].

Topographic error (TE) serves as a metric to evaluate the accuracy with which the map represents the structure of the input space. The computation of TE involves identifying the best-matching and second-best-matching neuron in the map for each input, followed by an assessment of their positions. If these nodes are adjacent, it signifies that the topology has been retained for this particular input. Conversely, if they are not neighboring, it is recorded as an error. The topographic error of the map is then calculated as the ratio of the total number of errors to the total number of data points [10].

This is summarized in following formula

$$TE(M) = 1/n \sum_{n}^{i=1} t(x_i)$$

$$t(x) = \begin{cases} 0, & \text{if } \mu(x) \text{ and } \mu^{'}(x) \text{ are neighbors,} \\ 1, & \text{otherwise.} \end{cases}$$

III. Research and Experiment

A. Surface Normal Estimation

Surface normal estimation is a crucial component in many 3D visualization and computer vision applications. Given a 3D point cloud, which is often obtained from devices like LIDAR scanners or 3D cameras, surface normal estimation refers to the process of determining the vector that is perpendicular (or "normal") to the surface at each point. These surface normals can provide valuable information about the underlying geometry and structure of the 3D object or scene. For example, they can be used for shading in 3D graphics, where the direction of the surface normal at a point helps determine how light will reflect off the surface and therefore how the surface will appear to the viewer. Surface normals can also be used in algorithms for object recognition, segmentation, and registration, where they can provide useful features or constraints.

In this section, two kinds of surface normal estimation techniques, optimization-based methods and averaging methods, are explained and compared; whose time complexity and accuracy being analysed and compared further along with discussions about their advantages and disadvantages.

1) Optimization-based Methods: Principle component analysis (PCA) is a classical dimensionality reduction method that attempt to find linear combinations of features in the original high dimensional data matrix to construct meaningful representation of the dataset. PCA aims to find linearly uncorrelated orthogonal principal components (PCs) to project the original data points onto those PCs. The first PC captures the largest variance in the data, in other words, it is the direction in which the data varies the most. By capturing the largest variance in the data, the first PC can provide a useful summary of the dataset that can be used for further analysis or visualization. Additionally, the subsequence PCs can capture additional patterns in the data that may not be apparent in the previous PC [11].

Many existing methods calculate the surface normal vector by solving the optimization problem

$$min_{n_i} J(p_i, Q_i, n_i)$$

where $J(p_i, Q_i, n_i)$ is a cost functional penalizing certain criteria, e.g. the distance of points to a local plane or the angle between tangential vectors and the normal vector.

PlaneSVD: A classical example of optimization-based methods, is to fit a local plane

$$S_i = n_{ix}x + n_{iy}y + n_{iz}z + d$$

to the points in Q_i^+, i.e. solve

$$min_{b_i} ||[Q_i^+ \;\; 1_{k+1}]b_i||_2$$

where 1_m indicates an $m * 1$ vector of ones and $b_i = [n_T^i, d]_T$. This optimization problem can be solved by computing the SVD $U \sum V_T$ of Q_+^i. The minimizer b_i is the vector V that corresponds to the smallest singular value in $\sum$, and hence the normal vector n_i is obtained directly [12].

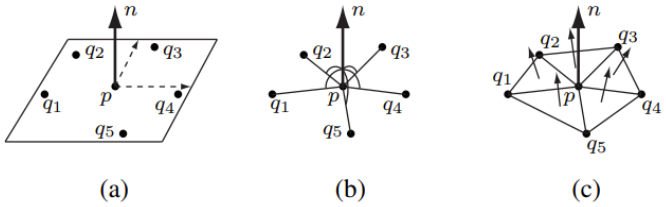


Fig. 3: Different approaches for estimating normal vectors: (a) A plane is fitted to p and its neighbors. (b) The angle between the normal vector and the tangential vectors is maximized. (c) The pair-wise cross product of vectors from a neuron to its neighbors are averaged [12]

2) Averaging methods: **Average Cross product algorithm** is used in this research (as shown in Fig 3 c) to estimate the surface normal for a particular neuron in SOM, belonging to averaging methods. Consider a neuron p as the winning neuron for a data item x_i during the learning of SOM, a list of neighbors can be determined by examine the distance between p and other neurons q_i. Then, vectors start at p and point to each q_i need to be determined by subtracting weighted vectors p from q_i:

$$v_i = q_i - p$$

All vectors v_i are preserved in the order in which they are computed. Therefore, the pairwise cross product is calculated between vectors v_i and v_j, provided the index i is smaller than the index j. Consequently, a total of $\binom{n}{2}$ cross products are calculated, where n denotes the number of vectors. The ultimate estimated cross product is subsequently derived from the average of all previous cross product [13] [12].

B. Global Distortion/twist in SOM

Self-Organizing Maps (SOMs), a form of artificial neural networks, utilize unsupervised learning methodologies to yield a reduced-dimensionality representation of high-dimensional input datasets. This portrayal, typically manifested in 2D or 3D spaces, aims to faithfully conserve the original topological and metric correlations inherent to the data. Nonetheless, certain scenarios might engender the emergence of distortions, which could manifest on either local or global scales.

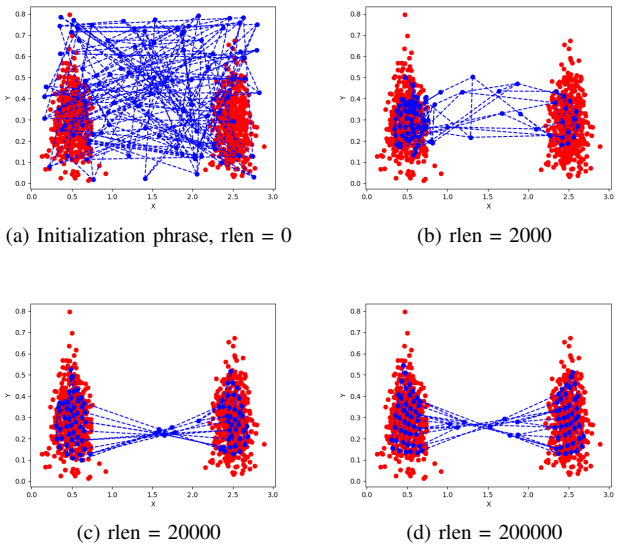
Numerous factors can catalyze global distortions within SOMs. One primary contributor is the innate trade-off accompanying the process of dimensionality reduction. The transformation of complex, high-dimensional patterns into lower-dimensional spaces can potentially introduce distortions that reflect the inherent limitations of this process. The choice of neighborhood function, responsible for guiding the learning process within SOMs, can also substantially influence the resulting data representation. A narrowly defined function can restrict participation in the learning process to a small subset of neurons, thereby fostering an environment conducive to distortion.

The physical characteristics of the map, specifically its size and shape, can dramatically impact the data representation. For instance, a map of insufficient size may not adequately encapsulate the complexities of the data, leading to distortions. Similarly, the shape of the map, such as a rectangular layout, can introduce distortions due to the corners' influence.

Training parameters, including the learning rate and radius of influence, can also induce distortions if not properly calibrated. A topological defects or distortion could be caused by a small initial radius [14]. When the initial radius is too small, it means that only a few neighboring neurons will be affected by the learning updates. Twisting occurs when neighboring neurons update their weights in a way that disrupts the original topological organization of the map. The reason why a small radius is more likely to cause twisting is because there is limited influence from neighboring neurons, and the learning updates may not be spread out evenly across the map.

Within the paradigm of a Self-Organizing Map (SOM), the principal objective revolves around preserving the topology to the input data within the resultant map space. This is accomplished by systematically updating the weights of neurons proximal to each other. As the SOM gradually learns to represent analogous input patterns within adjacent regions of the map, it is evident that the effective radius plays a pivotal role. If the radius is too small relative to neurons lattice size, it will result in limited participation of neighbors in weight updates. This, in turn, can hinder the propagation of the learning process uniformly across the map.

Let us now direct our attention towards an illustrative two-dimensional simulation. In this instance, two clusters of data input are generated randomly accordingly to gaussian-distributed with a standard deviation of 1, being separated along the x-axis by a distance of 2. A rectangular lattice sized $N = 12 * 8$ neurons are randomly generated by the algorithm



(a) Initialization phrase, rlen = 0 (b) rlen = 2000

(c) rlen = 20000 (d) rlen = 200000

Fig. 4: Example of a topological distortion (twist) in a 12 * 8 lattice. The evolution is shown for different running length (number of steps)

provided in the original "SOM_PACK" [15]. Then, the training of neurons are performed with $\alpha = 0.05$ (learning rate) and radius of 2 with "bubble" neighborhood function.

As illustrated in Fig. 4a, the neurons are initially distributed randomly, encompassing the data samples, devoid of any training applied. Upon observing Fig. 4b, one can discern that the neurons begin to adopt the topological structure of the data samples. Intriguingly, this results in the emergence of a topological distortion, specifically a twist at the central region of the diagram. As the training progresses, the twist becomes more pronounced, as evinced in Fig. 4c. The phenomenon elucidates that adjacent neurons do not necessarily share a high degree of similarity in their response characteristics. This is indicative of the unfolding topological anomaly within the self-organizing map. Further examination of Fig. 4 illustrates the evolution of the twist over time. Regrettably, a rapid decrease in the neighborhood range was observed as the lattice twisted. Consequently, even with extended simulation time and a zero neighborhood range, it is unlikely that the twist will be rectified. This highlights the challenges and constraints of topology preservation in self-organizing maps.

C. New learning algorithm for SOM based on surface normal estimation

Preventing distortions, such as twists in Self-Organizing Maps (SOMs), is crucial as the main objective of these maps is to provide a faithful representation of the input data in a reduced-dimensional space. These maps are often used for visualizing high-dimensional data, and distortions can lead to misinterpretations of the data structure, thus undermining the utility of the SOM. Distortions in a SOM might cause similar input data to be mapped far apart from each other, or dissimilar data to be mapped close to each other. This would disrupt

the topological ordering of the map, which is a key feature of SOMs. The topological ordering enables us to understand the relationships between different data points, and if this is disrupted by distortions, the interpretability of the SOM is compromised.

In this section, a new learning algorithm is presented based on surface normal estimation to prevent such twist. The algorithm can be divided into three parts, simulation of updates, comparison between surface normal estimations prior and after the updates, transformation of surface normal result into a factor. During a particular step of learning process, a winning neuron and all its neighboring neurons need to be determined first. Recall that neighboring neurons are determined by examining the euclidean distance in between them, where neurons are less r (radius) distance away from winning neuron are considered as its neighboring neuron. All neurons are then copied and stored into a collection, where a surface normal named it S_1 is estimated for all neurons in the collection by the average cross product algorithm mentioned in the previous section as follow:

$$S_1 = \frac{1}{\binom{n}{2}} \sum_{i=0}^{n} \sum_{j=i}^{n} \mathbf{v_i} \times \mathbf{v_j}$$

where v_i is a vector starting at wining neuron pointing to one of its neighboring neuron. Then usual updates are then applied to these neurons according to SOM competitive learning algorithm:

$$m_i(t+1) = m_i(t) + \alpha(t) h_{ci}(t)[x(t) - m_i(t)]$$

Now, the collection of neurons are updated according to a data sample, whose weight vectors are updated to be more likely to the data sample. Then a new surface normal S_2 is estimated with new weighted vectors:

$$S_2 = \frac{1}{\binom{n}{2}} \sum_{i=0}^{n} \sum_{j=i}^{n} \mathbf{v_i} \times \mathbf{v_j}$$

The angle in between these two surface normal are calculated by dot product

$$A * B = |A| \times |B| \cos(\theta) \Rightarrow \theta = \cos^{-1}((S_1 * S_2)/(|S_1| \times |S_1|))$$

Let θ denote the angle between two surface normals, with a value domain ranging from 0 to π . A value of θ near 0 signifies that the update of the neurons is conducted in a manner that preserves the local topological structure. Conversely, a θ value approaching π suggests that the direction of the local neurons has been altered through the update, which is indicative of the initiation or intensification of a topological distortion, colloquially known as a 'twist'. The algorithm under scrutiny leverages this quantitative parameter to modulate the extent of updates in direct proportion to the magnitude of θ , thereby providing a method to regulate the preservation or alteration of the local topology.

Then value of θ is transformed into a factor according to the following linear function

$$f(\theta) = -x/\pi + 1$$

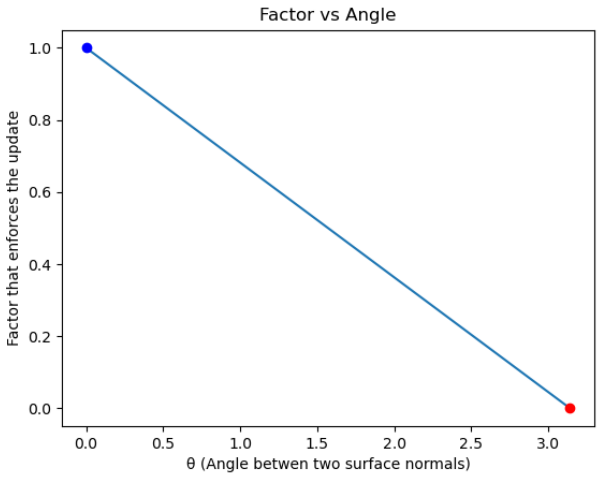


Fig. 5: Factor vs Angle in between two surface normals

In reference to fig 5, the value of scaling factor is monotonically decreasing as value of θ is increasing. The quantitative magnitude of this factor is 1 (doesn't affect an update) if the angle between surface normal vectors before and after an update is identical. Conversely, it approaches zero as θ approximates π, indicative of a direction inversion.

Then surface normal based new learning algorihtm can be formulated as follow:

$$m_i(t+1) = m_i(t) + f(\theta) \times \alpha(t) h_{ci}(t)[x(t) - m_i(t)]$$

The newly proposed learning algorithm introduces a unique strategy for mitigating topological distortions within local neuron configurations. Specifically, it is designed to discern significant directional changes in the local neuron landscape, indicative of potential twist formations. In response to such scenarios, the algorithm applies a modulating "resistant force" against the impending update, aiming to preserve the pre-existing topological orientation.

The magnitude of this corrective force is modulated by the function $f(\theta)$, where θ is the angular deviation between surface normal vectors before and after an update. The nature of the function $f(\theta)$ is such that its output value diminishes as θ increases, thereby ensuring that the corrective force is proportionately larger for greater directional deviations. This design not only discourages abrupt and severe topological changes, but also permits minor alterations, thereby striking a balance between stability and adaptability in SOM learning dynamics.

As illustrated in Figure 6, there is a range of transformation functions available to convert angular values into scalar factors. The chosen function for this study, f_1, is a linear function that monotonically decreases as the angle value increases.

Two extreme cases are represented by the constant functions f_8 and f_9, defined respectively as $f_8(\theta) = 1$ and $f_9(\theta) = 0$. The original Self-Organizing Map (SOM) learning algorithm resembles f_8 as it does not account for the angular value in the learning process, resulting in updates that are uninfluenced by the computed surface normal vectors. Conversely, f_9 halts

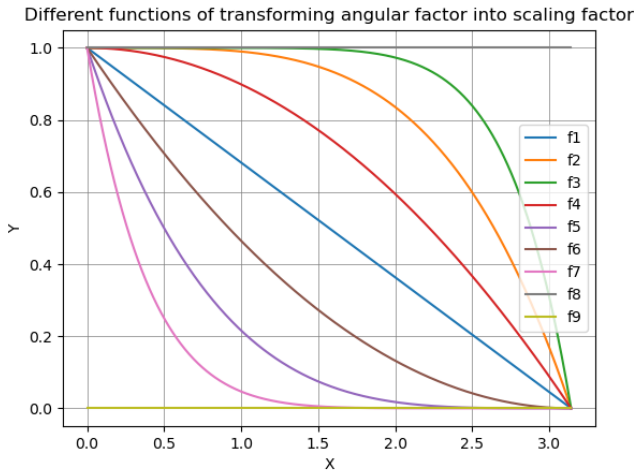


Fig. 6: Different functions of transforming angular value into scaling value

all neuron updates, effectively stopping the learning process entirely which contraries to the goals of SOM.

Neither of these approaches, f_8 or f_9, adequately serve the objective of avoiding global distortion while preserving the original characteristics of SOMs, such as maintaining topological relationships. The objective of a SOM would be thwarted if neurons learn inadequately from data samples. Therefore, both functions are not conducive for the purpose of global distortion prevention.

Functions f_{2-7} comprise polynomial functions of different degrees (2, 4, 8) sharing the same x and y intercepts (0, 1) and (π, 0). The outputs of f_1 and f_{2-7} do not differ significantly at a global scale, as both follow a similar trend. As such, this study employs the function f_1 in the new learning algorithm due to its computational simplicity and the preservation of the principal objectives of SOMs.

D. Comparison of Original and New Algorithms: learning outcome, time, Quantization Error, and Convergence Speed

This section is dedicated to elucidating the performance characteristics of the original and newly proposed algorithms, specifically in the context of three-dimensional examples. We rigorously compare the algorithms' ability to prevent the formation of topological twists, their time efficiency, quantization errors, and speed of convergence.

As shown in fig 7, the first example is conducted with two randomly generated clusters and a 6×6 hexa neuron lattice are initialized in the middle. Learning rate α is set to be 0.05 and initial radius r is set to be 2 with "bubble" neighborhood function.

As illustrated in Figures 8 and 9, both algorithms yield comparable results with respect to the structure of neurons for the initial 5000 epochs (rlen). Notably, no twist occurs as neurons extend to encapsulate more intrinsic structure of the data samples. However, an evident twist becomes observable in Figure 8c, wherein proximate neurons are mapped to distinct clusters, thereby creating a butterfly-shaped mapping.

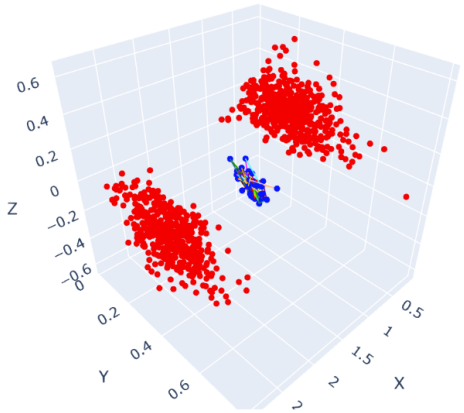
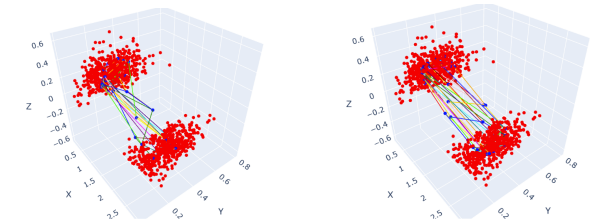
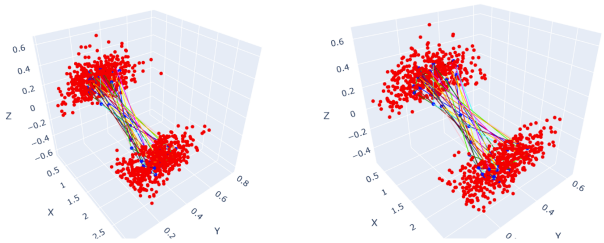


Fig. 7: Initialization of neurons and data samples



(a) rlen = 2000 (b) rlen = 5000

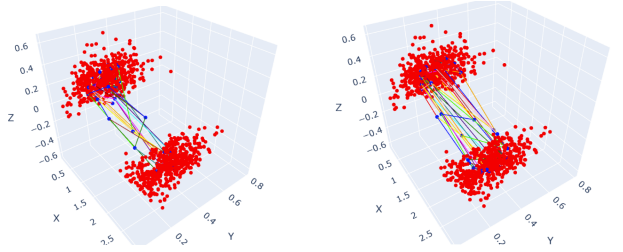


(c) rlen = 20000 (d) rlen = 50000

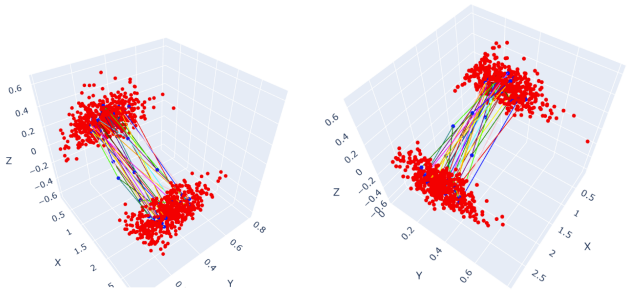
Fig. 8: Example 1 conventional Learning Algorithm Result

The conventional learning algorithm is not optimal, as the learning process results in neurons which inadequately represent the topology of data samples. Simultaneously, it generates an increased quantization error due to the resultant form. In contrast, the surface-normal based algorithm does not produce a discernible twist after an equivalent number of epochs, as depicted in Figure 9c. Here, neurons are organized in a manner that captures the general shape of the data sample, resulting in a box-like shape as the outcome of the learning process. This, therefore, suggests that the surface-normal based algorithm may be more efficient in representing the structure of the data samples in this perticular case.

Simultaneously, Figures 8d and 9d reaffirm the trends previously observed. The twist issue prevalent in the conventional learning algorithm is persistent and doesn't appear to be mitigated by an increase in epochs, thus revealing an inherent limitation in its adaptive capacity. In contrast, the surface-normal based algorithm continues to exhibit stability and consistency in its ability to avoid twisting, as evidenced in Figure 9d. No clear indication of twisting is apparent, even with the progression of the learning process. These observations



(a) rlen = 2000 (b) rlen = 5000



(c) rlen = 20000 (d) rlen = 50000

Fig. 9: Example 1 surface normal based Learning Algorithm Result

reinforce the potential advantages of the surface-normal based approach over the conventional learning algorithm, particularly in the context of epochs evolution.

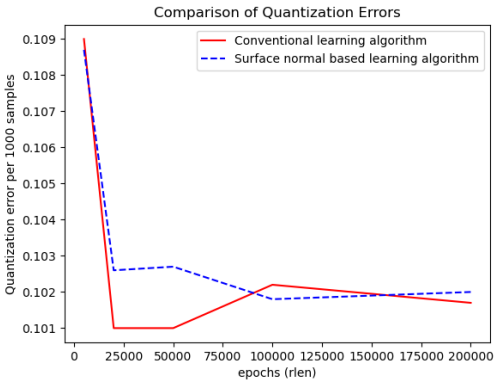
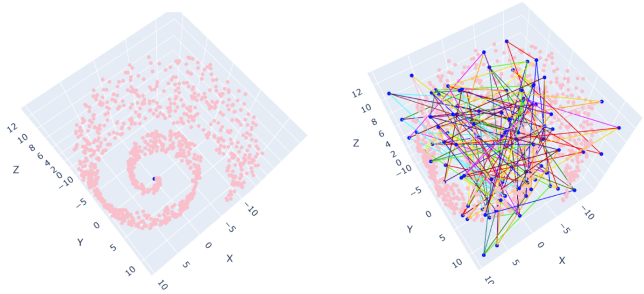


Fig. 10: Comparison of Quantization Errors

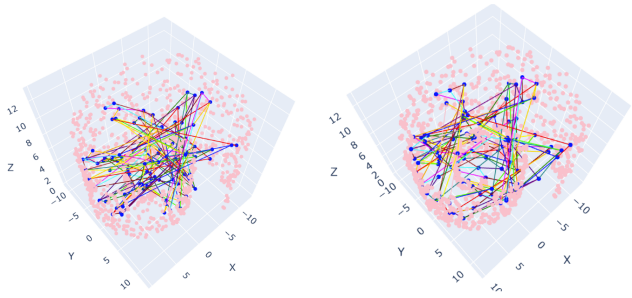
As indicated in Figure 10, there is a rapid decrease in quantization errors during the initial 25000 epochs of learning, demonstrating that the neurons enter a stable state following this period. The stability of these errors continues to be maintained from 25000 epochs to 200000 epochs, indicating a persistent stable state in the neuronal state. Furthermore, the surface normal-based learning algorithm exhibits a similar rate of convergence to the stable state as the conventional learning algorithm, as shown in Figure 10. This observation suggests that the surface-normal based algorithm is not only proficient in preventing twist, but also succeeds in maintaining a comparable learning speed to the traditional algorithm.

As shown in fig 11, the second example is conducted with

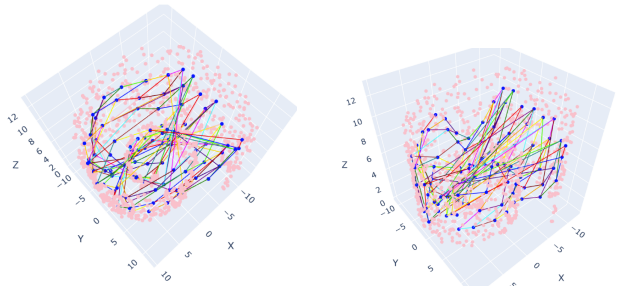


(a) Data samples (b) Data samples with neurons

Fig. 11: Example 2 Initialization of neurons and data samples



(a) rlen = 2000 (b) rlen = 5000



(c) rlen = 20000 (d) rlen = 200000

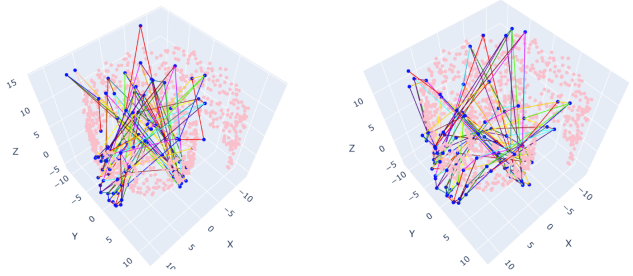
Fig. 12: Example 2 conventional Learning Algorithm Result

spiral shape data with randomly initialized hexa 12×8 neurons. Learning rate α is set to be 0.05 and initial radius r is set to be 2 with "bubble" neighborhood function.

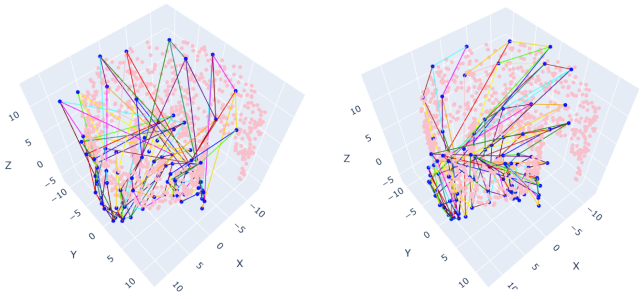
As depicted in Figures 12a and 13a, the learning outcomes from the two algorithms differ significantly. The conventional algorithm adjusts all neurons to conform to the spiral dataset, resulting in numerous distortions and twists, particularly in the central region. Conversely, the surface normal-based algorithm exhibits a slower learning pace but structures the neurons in a more organized fashion, with fewer twists observed.

In the subsequent illustrations (Figures 12 and 13), the conventional algorithm continues to generate some twists within the central region, whereas the surface normal-based algorithm yields a cone-shaped structure. This comparison further underscores the different learning strategies and outcomes associated with these two algorithms.

As illustrated in Figure 14, the quantization error exhibits a sharp initial decrease before gradually stabilizing. Notably, a



(a) rlen = 2000 (b) rlen = 5000



(c) rlen = 20000 (d) rlen = 200000

Fig. 13: Example 2 surface normal based Learning Algorithm Result

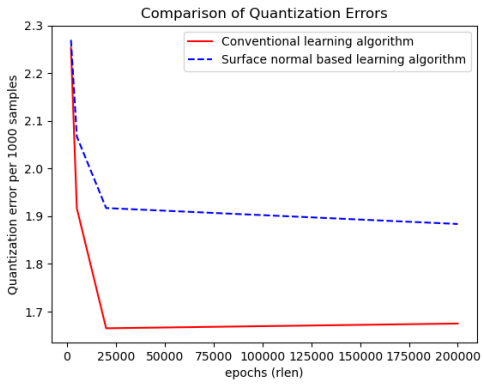
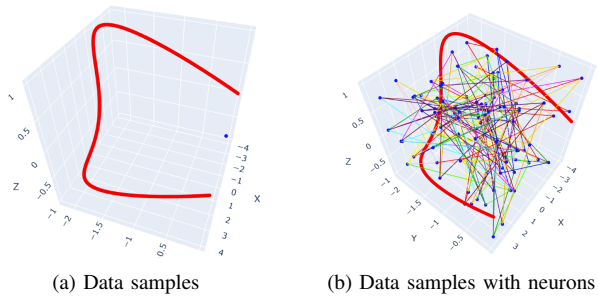


Fig. 14: Comparison of Quantization Errors

divergence between the two algorithms is observable around the 25000 epoch mark, where the traditional algorithm demonstrates a lower quantization error relative to the surface normal based algorithm.

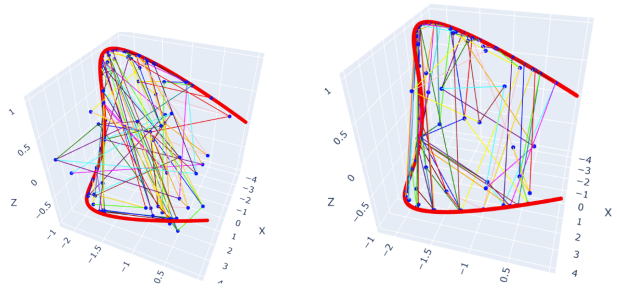
This discrepancy can be attributed to the nature of the dataset (spiral), which is non-linear and exhibits numerous twists. These complexities make it more challenging for the surface normal based algorithm to learn, resulting in a higher quantity of quantization errors. Nonetheless, both algorithms reach a state of stability around the same time, reinforcing the observations made from the preceding example.

The last example shown in fig 16 is conducted with a ”C” shape data-set and 12×8 hexa neurons are randomly generated around the data samples. Learning rate is set to be 0.05 and initial radius r is set to be 2 with ”bubble” neighborhood function.

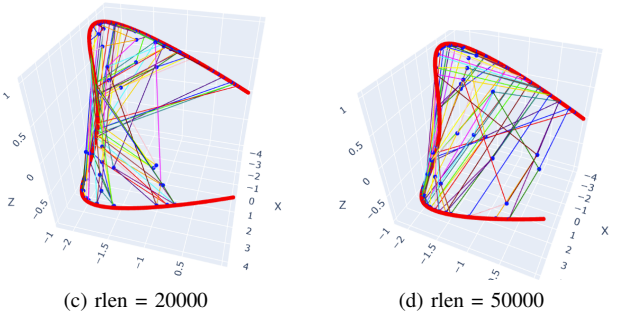


(a) Data samples (b) Data samples with neurons

Fig. 15: Example 3 Initialization of neurons and data samples



(a) rlen = 2000 (b) rlen = 5000



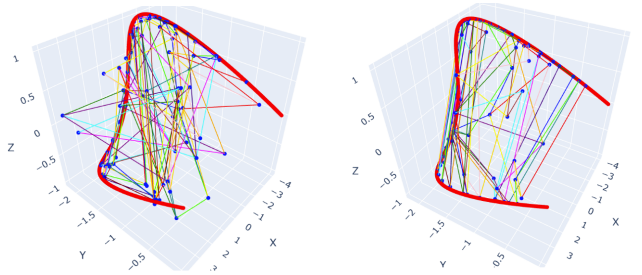
(c) rlen = 20000 (d) rlen = 50000

Fig. 16: Example 3 conventional Learning Algorithm Result

As depicted in Figure 16a and 17a, both algorithms yield similar outcomes for the initial 5000 epochs, as the neurons capture the overall ”C” shape structure of the data samples. However, divergences become evident at the 20000 epoch mark, as shown in Figure 16c and 17c.

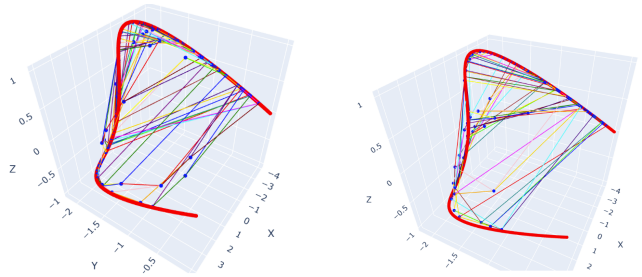
The traditional algorithm generates multiple twists concentrated around the central region, with neurons at the two ends connecting to the center. In contrast, the surface normal based algorithm results in a more elongated neuron mapping, with most neurons linking to their immediate neighbors and fewer twists occurring.

As observed earlier, both algorithms initially demonstrate a swift reduction in quantization errors within the first 5000 epochs (fig 18), subsequently stabilizing. Notably, the quantization error associated with the surface normal-based algorithm begins to escalate post the 25000 epoch mark, which is indicative of overtraining. In such a scenario, the map is more likely to accommodate the noise or outliers inherent in the data samples as opposed to identifying the overall pattern. Nevertheless, it can be inferred that both algorithms attain a



(a) rlen = 2000

(b) rlen = 5000



(c) rlen = 20000

(d) rlen = 50000

Fig. 17: Example 3 surface normal based Learning Algorithm Result

stable state at a comparable rate for this specific case.

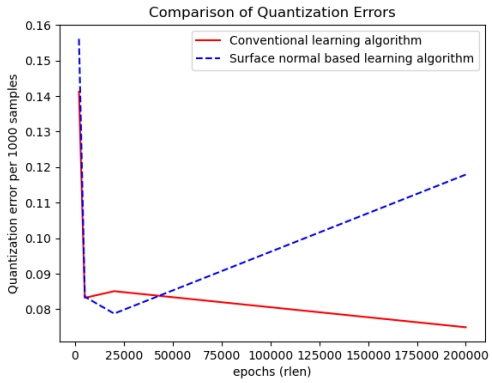


Fig. 18: Comparison of Quantization Errors

Limitation

1. A significant constraint of the surface normal (average cross product method) based algorithm is its exclusive applicability to two or three-dimensional data, stemming from the inherent definition of the cross product. This concept is only clearly defined within two or three dimensions, rendering the direct extraction of the surface normal from high-dimensional data unfeasible with this approach.

2. For non-linear datasets like spirals, which produce a substantial number of twists, the surface normal-based algorithm may generate higher quantization errors compared to conventional algorithms. This phenomenon can be attributed to the surface normal algorithm's restrictive influence on a cluster of neurons' learning capacity for a given data sample.

Improvement

To overcome the first limitation, the average cross product method can be extended to higher dimensions (beyond three) by leveraging mathematical concepts like the exterior product and Hodge star operator. By integrating these tools, we can derive an equivalent to the surface normal in higher-dimensional spaces. The synthesis of the exterior product and Hodge star operator provides a way to represent the 'surface normal' concept in dimensions greater than three.

Conclusion

In this research, we introduce a novel Self-Organizing Map (SOM) learning algorithm, known as the surface normal-based algorithm, designed to mitigate topological distortions in SOMs. This new approach factors in the orientation (surface normal) of local neurons, adjusting the learning extent of a neuron group from a data sample based on the estimated surface normal vectors pre and post-update. Essentially, the larger the angle between two surface normal vectors, the less the neurons learn from the data sample.

To illustrate the performance of the new learning algorithm, we present three different examples involving two clusters, a spiral, and a "C"-shaped dataset. All examples showcase the algorithm's capability to prevent twisting while maintaining a comparable convergence rate. Although the new algorithm learns slightly slower than the conventional one, it does so in a more deliberate and cautious manner.

In conclusion, the surface normal-based algorithm, which utilizes the averaged cross product method, consistently demonstrates stability in preventing twists and sustaining a similar convergence speed. It is an efficient approach in managing the complexity and variability of learning dynamics in SOMs.

Acknowledgment

Many appreciates to Prof Masahiro, who answered all my questions tirelessly and guided me through the research from start to finish.

References

[1] K. Teuvo, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, 1990-09.

[2] C. von der Malsburg, "Self-organization of orientation sensitive cells in the striate cortex," *Kybernetik*, vol. 14, pp. 85–100, 1973-06-01.

[3] A. Cartas, "Self-organizing maps," 2014. Accessed: 2023-05-31.

[4] K. Teuvo, "Kohonen network," *Scholarpedia*, 2007.

[5] M. E. M. Akinduko, Ayodeji A, "Initialization of self-organizing maps: Principal components versus random initialization. a case study," *n.d.*, n.d.

[6] L. B. . F. A. Mohammed Attik, "Self-organizing map initialization," *LNTCS*, vol. 3696, n.d.

[7] E. Forgy, "Cluster analysis of multivariate data : efficiency versus interpretability of classifications," 1965.

[8] M. J, "Some methods for classification and analysis of multivariate observations," *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 66, p. 281–297, 1967.

[9] S. D. W. Natita, W. Wiboonsak, "Appropriate learning rate and neighborhood function of self-organizing map (som) for specific humidity pattern classification over southern thailand," *International Journal of Modeling and Optimization 6(1)*, pp. 61–65, January 2016.

[10] B. Gregory, "Evaluating self-organizing map quality measures as convergence criteria," *DOI.org (Crossref)*, pp. 21–22, 2017.

[11] R. W. Maćkiewicz Andrzej, "Principal components analysis (pca)," *Computers Geosciences*, vol. 19, pp. 303–342, 1993-03-01.
[12] A. D. Klasing Klaas, "Comparison of surface normal estimation methods for range sensing applications," *2009 IEEE International Conference on Robotics and Automation*, pp. 3206–3211, 2009.
[13] W. D. Jin Shuangshuang, Lewis Robert R., "A comparison of algorithms for vertex normal computation," *The Visual Computer*, pp. 71–82, 2005-02-01.
[14] T. M. Heskes and B. Kappen, "Error potentials for self-organization," *IEEE International Conference on Neural Networks*, p. 1219–1223, 1993.
[15] "Neural network research centre: Som_pak and lvq_pak.." http://cis.legacy.ics.tkk.fi/hynde/lvq/. accessed Jun. 10, 2023.