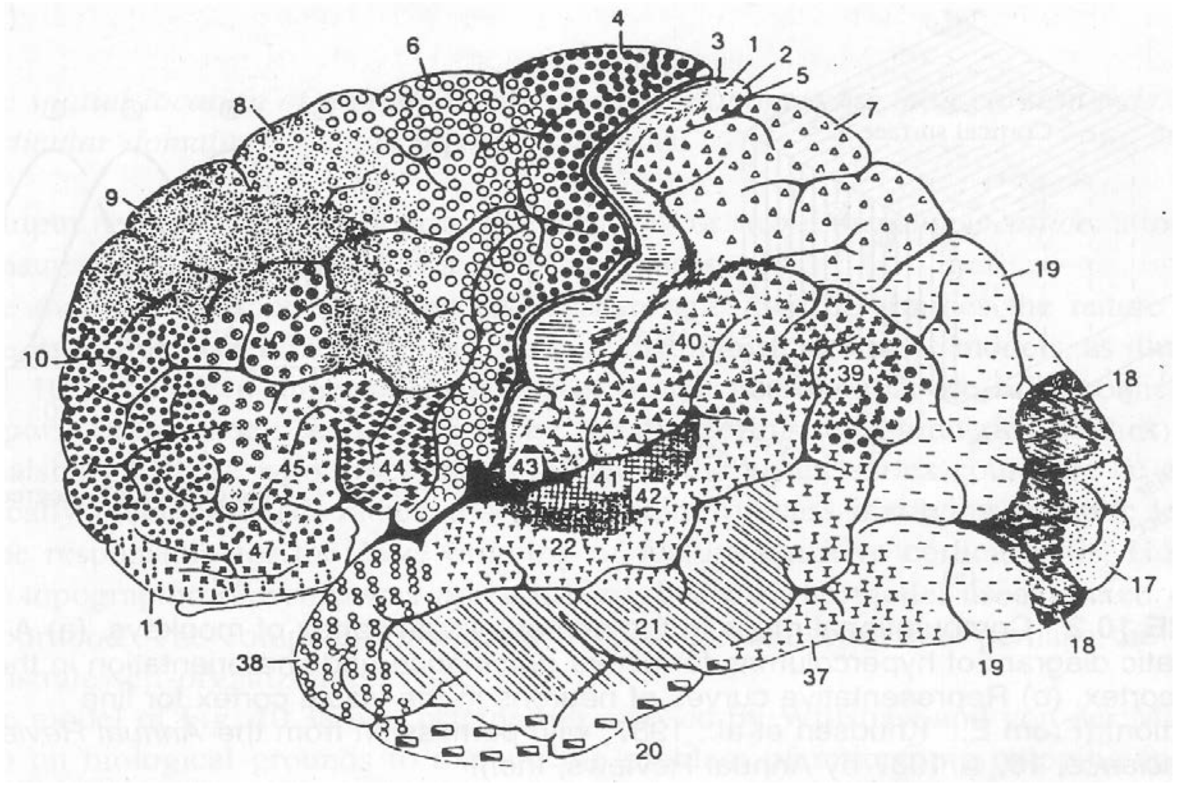
5. Self-Organizing Map Neural Network

The development of the self-organizing map is inspired by the research in neurobiology. Consider the map of a human brain:



The map of a human brain

The brain diagram presents the map of the cerebral cortex. The different areas of the cortex are identified by the thickness of their layer and the types of neurons within them. Some of the most important specific areas are as follows:

(1). Motor cortex:	area 4, area 6 and area 8
(2). Visual cortex:	areas 1,2 and 3.
(3). Auditory cortex:	areas 41 and 42.

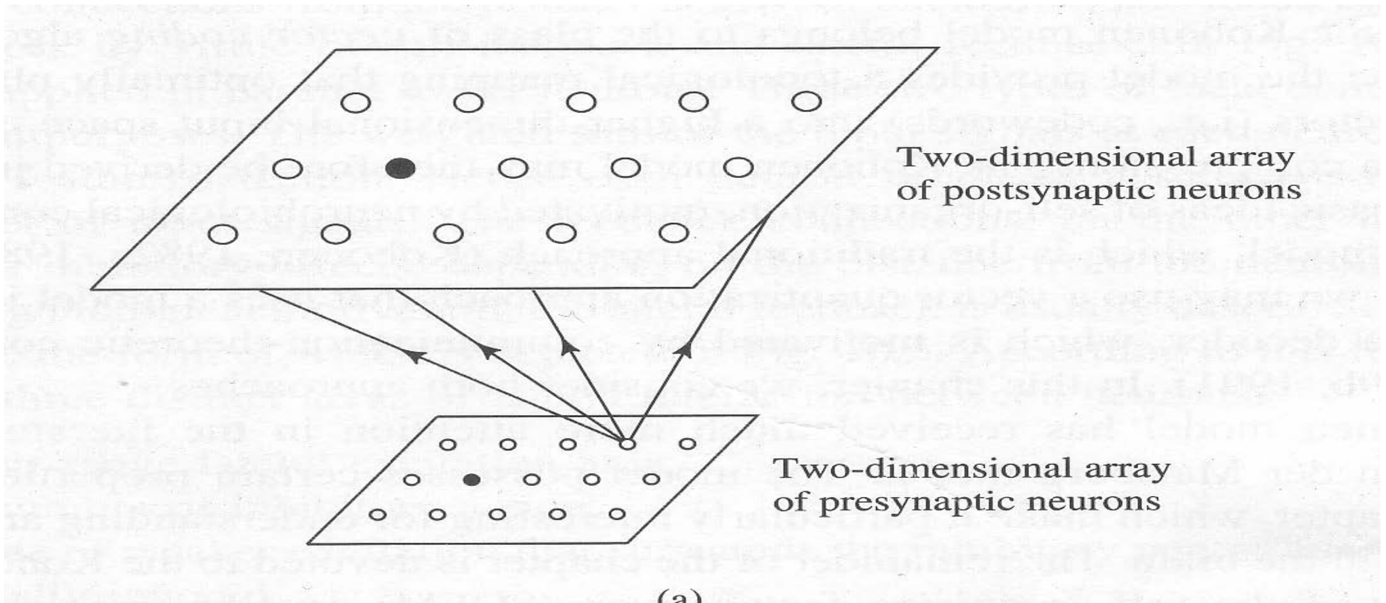
The discovery in the brain map shows clearly that different sensory inputs (motor, visual and auditory, etc) are mapped onto corresponding areas of the cerebral cortex in an orderly fashion. These cortical maps are not entirely genetically predetermined. Rather, they are sketched in during the early development of the nervous system.

The essential point of the discovery in the neurobiology lies in the principle of topographic map formation. This principle can be stated as follows:

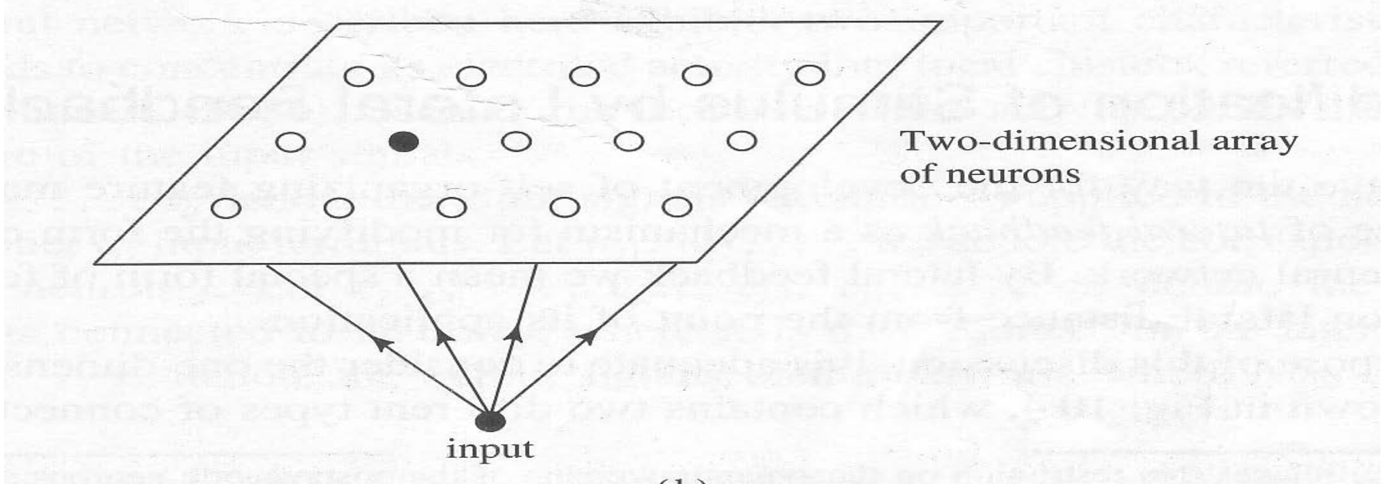
<u>The spatial location of an output neuron in the topographic map corresponds to a particular domain or feature of the input data.</u>

The output neurons are usually arranged in a one-dimensional or two-dimensional lattice, a topology ensures that each neuron has a set of neighbors.

The manner in which the input patterns are specified determines the nature of the mapping model. In particular, we may identify two basic models, both were inspired by the discovery that the model of a visual cortex could not be entirely genetically predetermined, a self-organizing process involving synaptic learning may be responsible for local ordering of feature sensitive cortical cells.



(a)



(b)

Two models of the self-organizing mapping

The first type of mapping model has two separate two-dimensional lattice, with one projecting onto the other. The first lattice represents pre-synaptic (input) neurons, and the other lattice represents post-synaptic (output) neurons. This model tries to explain details of neurobiology.

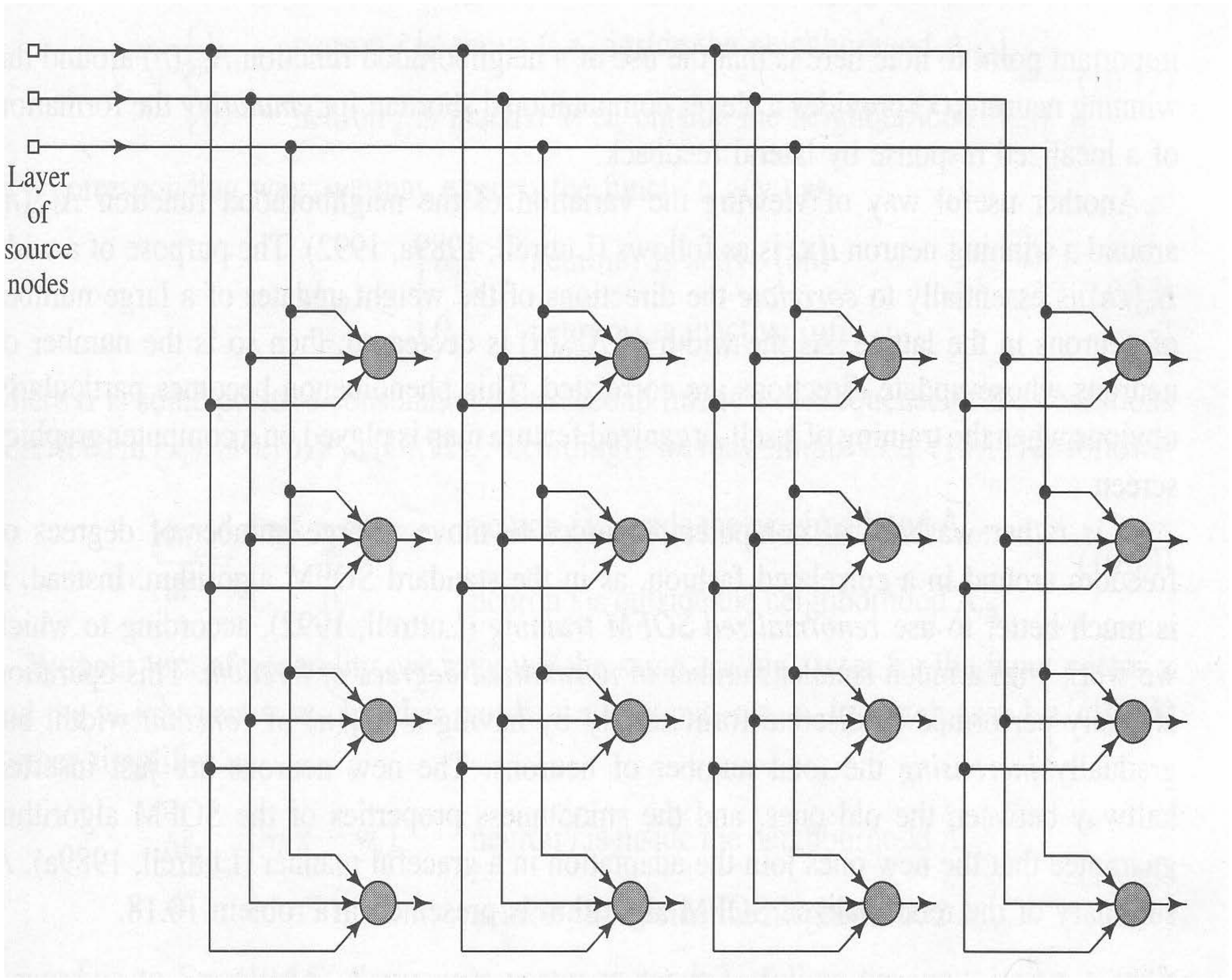
The second type of mapping model has only one two-dimensional lattice. Instead of explaining the details, this type of model tries to capture the essential features of computational maps in the brain and yet remain computational tractable. This model is developed by Kohonen, and is often called Kohonen self-organizing map.

SOM has received considerable attention since it was proposed in 1982. Now it has become a popular and power tool for feature extraction in pattern recognition. This is due to the property of SOM that SOM approximates the input space.

Self-Organizing Map (SOM)

The principle of Kohonen SOM is to transform an incoming signal pattern of arbitrary dimension into a one-dimensional or two-dimensional discrete map, and to perform this transform adaptively in a topological ordered fashion. The essential ingredients of the SOM neural network are as follows:

(1) A one- or two-dimensional lattice of neurons that computes simple discriminant functions of inputs received from an input of arbitrary dimension.
(2) A mechanism that compares these functions and selects the neurons with the largest discriminant function value.
(3) An adaptive process that enables the activated neurons to increase their discriminant values.
(4) An interactive network that activates the selected neuron and its neighbors.



Two-dimensional lattice of neurons

The algorithm responsible for the formation of the SOM proceeds first by initializing the synaptic weights in the network through assigning them small random values. This process is called initialization. Once the network has been properly initialized, there are three essential processes involved in the formation (training) of the self-organizing map:

(1) Competition. For each input pattern, the neurons in the network compute their respective values of a discriminant function. This discriminant function provides the basis for competition among the neurons. The neuron with the largest discriminant function is the winner of the competition.
(2) Cooperation. The winning neuron determines the spatial location of a topological neighborhood of excited neurons.
(3) Weights adaptation. This last mechanism enables the excited neuron to increase their individual values of discriminant function through suitable adjustments to the weights.

Competitive process

Assume an input pattern selected randomly from the input space denoted by:

$$\mathbf{x} = [x_1, x_2, \cdots, x_n]^T$$

The weight vector of each neuron in the network has the same dimension as the input space. Let the weight vector of neuron j be denoted by:

$$\mathbf{w}_j = [w_{j1}, w_{j2}, \cdots, w_{jn}]$$

To find the best match of the input vector **x** with the synaptic weight vector $\mathbf{w}_j$ we simply compare the inner products $\mathbf{w}_j\mathbf{x}$ for j=1,2,…,m, where m is the number of neurons, and select the largest. In practice, we often find it is convenient to normalize the weight vector to constant Euclidean norm. In such a case, the maximum inner product is equivalent to minimum Euclidean distance between vectors. If we use i(**x**) to identify the neuron that best matches the input vector **x**, we may then determine i(**x**) by using:

$$i(\mathbf{x}) = \arg\min \left\| \mathbf{x} - \mathbf{w}_j \right\|$$

Where ||*|| denotes the Euclidean norm of the argument vector. i(**x**) is the subject of attention, and the corresponding neuron i is called winning neuron for the input vector **x**.

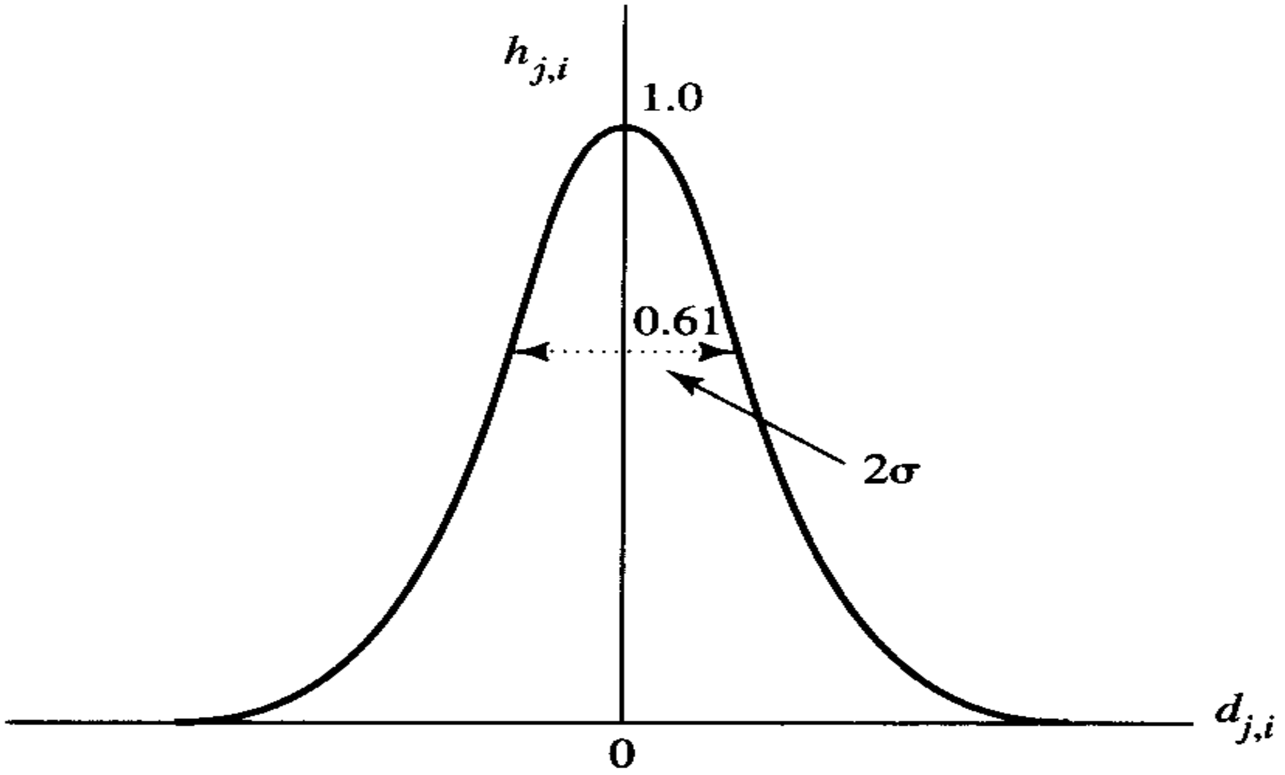
Cooperation process

The winning neuron locates the center of a topological neighborhood of cooperating neurons. The key question is: how we define a topological neighborhood that is neurobiological correct? Research in neurobiology reveals that when a neuron is excited, its immediate neighborhood tends to be excited more than those that are far away from it. This observation leads us to make the topological neighborhood around the winning neuron *i* decay smoothly with the distance. Let h_{ji} denote the topological neighbor-hood centered on winning neuron *i*, neuron j is one of the set of excited (cooperating) neurons. The distance between winning neuron i and excited neuron j is denoted by d_{ji}. Then we may assume that the neighborhood h_{ji} is an unimodal function of the lateral distance d_{ji} such that it satisfies two distinct requirements:

(1) The topological neighborhood h_{ji} is symmetric about the maximum point defined by d_{ji}. In other words, it attains its maximum value at the winning neuron.
(2) The amplitude of the topological neighborhood h_j decreases monotonically with increasing lateral distance d_{ji}, decaying to zero when $d_{ji} \rightarrow \infty$.

A typical choice of h_{ji} that satisfies these requirements is the Gaussian function:

$$h_{ji} = \exp\left(-\frac{d_{ji}^2}{2\sigma^2}\right)$$



Parameter σ is the width of the Gaussian function, it determines the degree to which excited neurons in the vicinity of the winning neuron participate in the learning process.

For cooperation among neighboring neurons to hold, it is necessary that the topological neighborhood h_{ji} be dependent on distance between winning neuron *i* and excited neuron *j* in the output space rather than the distance measure in the original measurement space.

In the case of a one-dimensional space, d_{ji} is an integer defined as:

$$d_{ji} = \|j - i\|$$

In the case of a two-dimensional space, d_{ji} is defined as:

$$d_{ji} = \|\mathbf{r}_j - \mathbf{r}_i\|$$

Where $\mathbf{r}_i$ and $\mathbf{r}_j$ denotes the discrete position of winning neuron i and excited neuron j measured in the output space respectively.

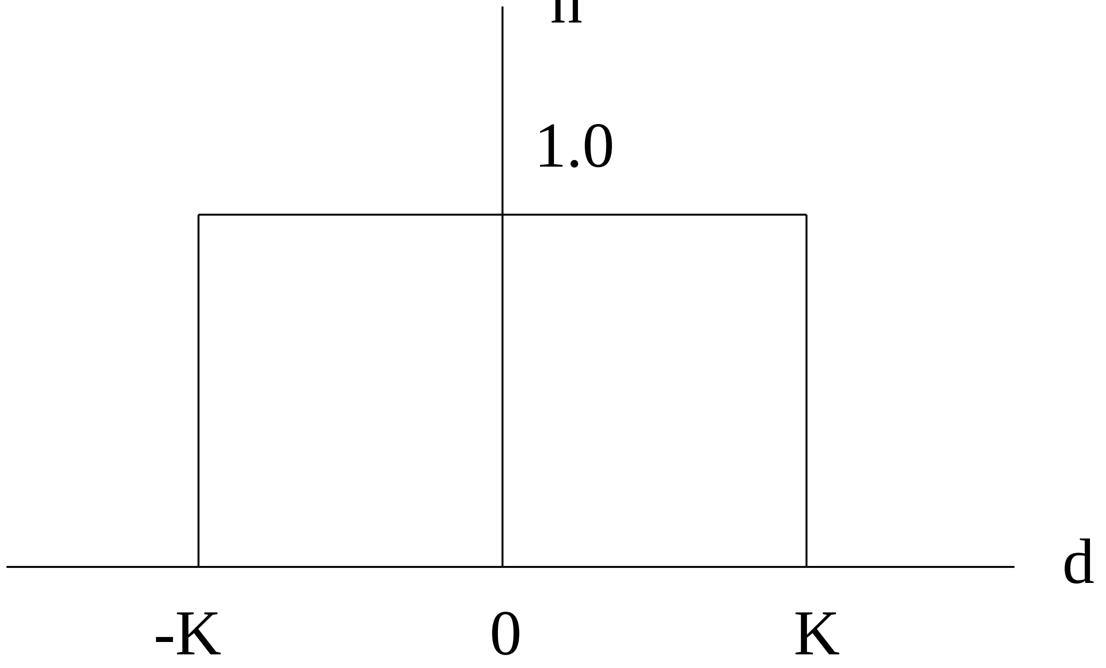
Another unique feature of SOM is that the size of the topological neighborhood shrinks with the learning process. This can be done by making the width parameter σ of the topological neighborhood function h_{ji} decrease with the learning process. A typical choice for the dependence of σ on the learning process is the exponential decay described by:

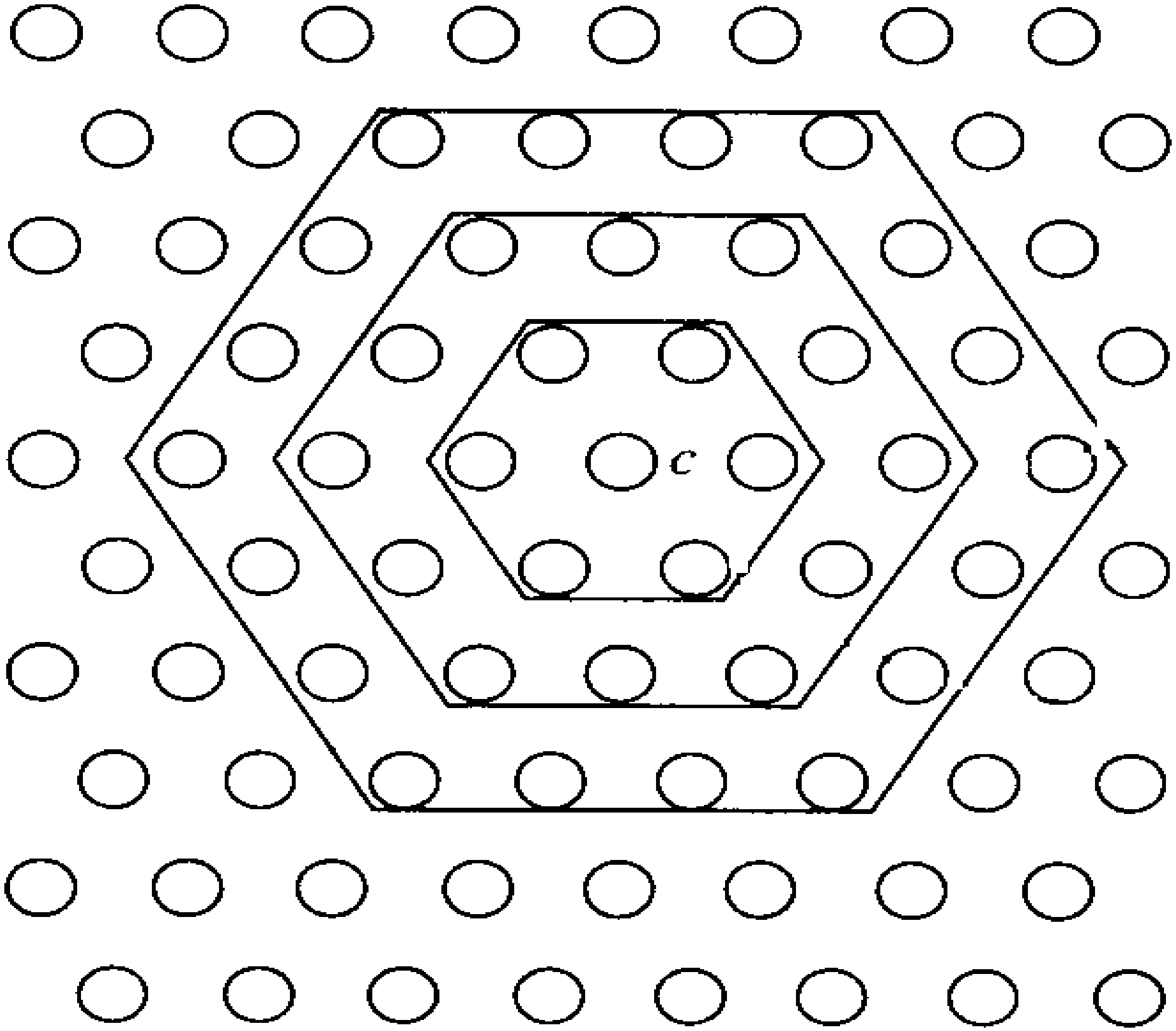
$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right)$$

Where σ_0 is the value at the initiation of the SOM algorithm, and τ_1 is a time constant, n is the number of iterations. Thus, as n increase, the width $\sigma(n)$ decreases at an exponential rate, and the topological neighborhood shrinks in a corresponding manner as described below:

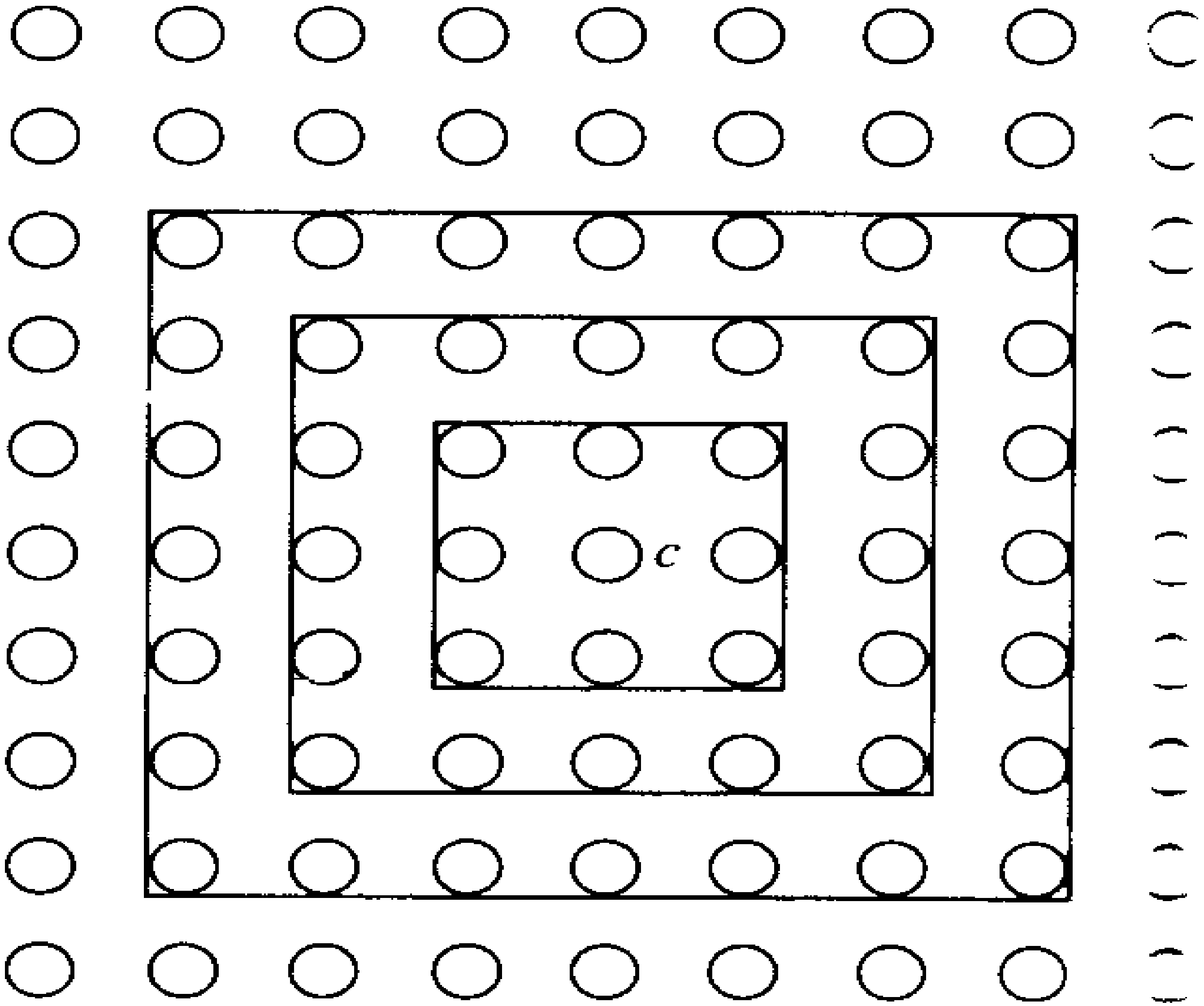
$$h_{ji}(n) = \exp\left(-\frac{d_{ji}^2}{2\sigma^2(n)}\right)$$

Besides the Gaussian neighborhood function, hexagon and square neighborhood are also used in SOM. Neurons inside the defined neighborhood are excited. In such cases, the h_{ji} is in the following form:





Hexagon



Square

Adaptive process

In the weight adaptive process, the weight vector $\mathbf{w}_j$ of neuron j in the network is required to change in relation to the input vector **x**. The question is how to make the change. In the SOM algorithm, the change to the weight vector of neuron j, which is inside the topological neighborhood of winning neuron i, is as follow:

$$\Delta \mathbf{w}_j(n) = \eta(n) h_{ji(x)} [\mathbf{x} - \mathbf{w}_j(n)]$$

Where η (n) is the learning rate at iteration n. After change, the weight vector becomes:

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta(n) h_{ji(x)} [\mathbf{x} - \mathbf{w}_j(n)]$$

The above equation has the effect of moving the weight vector of winning neuron toward the input vector **x**. Upon repeated presentation of the training data, the weight vectors tend to follow the distribution of the input vectors.

The learning-rate parameter $\eta(n)$ should be time-varying. It should start at an initial value and then decrease gradually with the increase of iteration number *n* as shown below:

$$\eta(n) = \eta_0 \exp\left(-\frac{n}{\tau_2} \right)$$

Where η_0 is the initial value, and τ_2 is another time constant.

Two phases of the Adaptive process

Starting from an initial state of complete disorder, the SOM algorithm gradually leads to an organized representation of activation patterns drawn from the input space.

The adaptation process of weight vectors in the network can be decomposed into two phases: an ordering or self-organizing phase followed by a convergence phase as described below:

(1) Self-organizing

It is during this first phase of the adaptive process that topological ordering of the weight vectors takes place. The ordering phase may take as many as 1000 iterations, or even more, of the SOM algorithm. In this phase, learning rate parameter and neighbor-hood functions must be carefully selected.

The learning rate parameter $\eta(n)$ should begin with a value close to 0.1; and the value decreases gradually, but remain above 0.01. These desirable values are satisfied by the setting $\eta_0=0.1$ and $\tau_1=1000$. Thus, the varying learning rate formula becomes:

$$\eta(n) = 0.1\exp\left(-\frac{n}{1000}\right)$$

The neighborhood function $h_{ji}(n)$ should initially include almost all neurons in the network centered on the winning neuron *i*, and then shrink slow with iterations. Specifically, during the ordering phase that may occupy 1000 iterations or more, $h_{ji}(n)$ is permitted to reduce to a small value of only a couple of neighboring neurons around the winning neurons, or even to the winning neuron itself. We may set the size of σ_0 to the radius of the two-dimensional lattice, and set the time constant τ_1 as follow:

$$\tau_1 = \frac{1000}{\ln \sigma_0}$$

(2) Convergence phase

This second phase of the adaptive process is needed to fine tune the map and therefore provide an accurate statistical quantification of the input space. As a general rule, the number of iterations of the convergence phase must be at least 500 times the number of neurons in the network. Thus, the convergence phase may have to go on for thousands and possibly tens of thousands of iterations.

For good statistical accuracy, the learning rate parameter $\eta(n)$ should be maintained during the convergence phase at a small value on the order of 0.01, but should not be zero.

The neighborhood function $h_{ji}(n)$ should contain only the nearest neighbors of a winning neuron, which may eventually reduce to one or zero neighboring neurons.

Summary of the SOM algorithm

There are four basic steps involved in the algorithm, namely, initialization, sampling, similarity matching and updating. The three steps after initialization are repeated until the map function is completed. The algorithm is summarized as follows.

(1) Initialization. Choose random values for the initial weight vectors $\mathbf{w}_j(0)$. The only restriction on the initialization is that the initial values must be different from each other. And it may be desirable to keep the magnitude of the weights small.

(2) Sampling. Draw a sample **x** from the input distribution with a certain probability. Usually, the **x** is drawn from the given training samples set.

(3) Similarity matching. Find the winning neuron i(**x**) at the time *n* using the minimum-distance Euclidean criterion:

$$i(\mathbf{x}) = \arg\min \left\| \mathbf{x} - \mathbf{w}_j(n) \right\|$$

where j=1,2,…,N.

(4) Updating. Adjust the synaptic weight vectors of all neurons using the updating formula:

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta(n) h_{ji(x)} [\mathbf{x} - \mathbf{w}_j(n)]$$

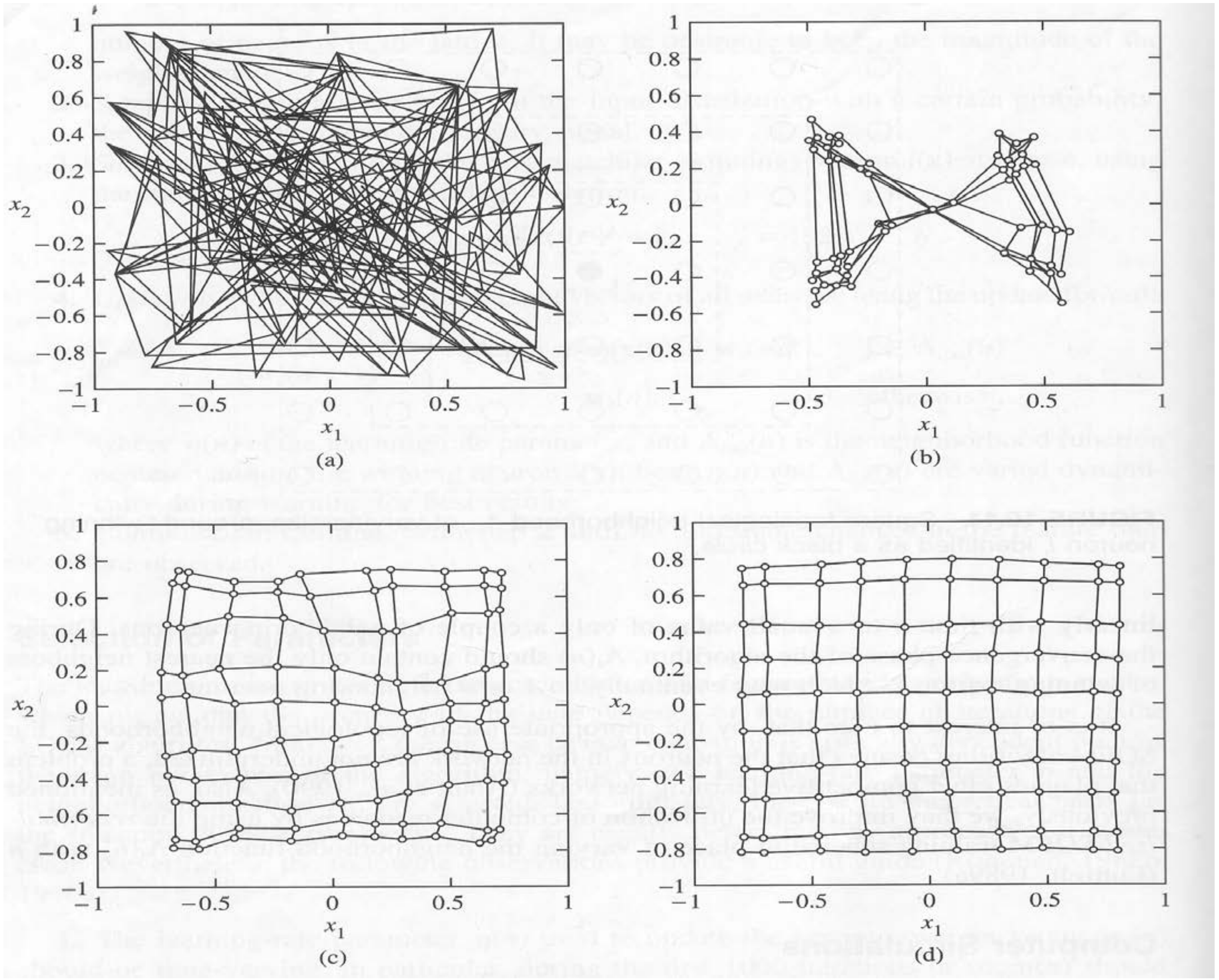
(5) Continuation. Repeat steps 2-4 until no noticeable changes in the map are observed.

Example of SOM

Consider a computer simulation of a SOM arranged in a two-dimensional lattice with 10 rows and 10 columns. The network is trained with a two-dimensional input vector **x**, whose elements x_1 and x_2 are uniformly distributed in the region:

$$-1 < x_1 < +1$$

$$-1 < x_2 < +1$$



Example of SOM training

Example 1

Given 4 training samples:

$$\mathbf{x}_1 = [1,1,0,0]^T \qquad \mathbf{x}_2 = [0,0,0,1]^T$$

$$\mathbf{x}_3 = [1,0,0,0]^T \qquad \mathbf{x}_4 = [0,0,1,1]^T$$

We wish to find 2 clusters of the training samples. Suppose the learning rate is

$$\eta(0) = 0.6$$
$$\eta(n+1) = 0.5\eta(n)$$

The neighborhood function is set so that only the winning neuron is updated with its weights at each step.

Step 1: initialization. Initialize two weight vectors:

$$\mathbf{w}_1(0) = [0.2,0.6,0.5,0.9]^T \qquad \mathbf{w}_2(0) = [0.8,0.4,0.7,0.3]^T$$

Step 2: randomly select a sample, say $\mathbf{x}_1$, we have:

$$d_1 = \|\mathbf{w}_1(0) - \mathbf{x}_1\|^2 = (0.2-1)^2 + (0.6-1)^2 + (0.5-0)^2 + (0.9-0)^2 = 1.86$$

$$d_2 = \|\mathbf{w}_2(0) - \mathbf{x}_1\|^2 = 0.98$$

So the winning neuron is neuron 2, i($\mathbf{x}$)=2; update the weights

$$\mathbf{w}_2(1) = \mathbf{w}_2(0) + 0.6[\mathbf{x}_1 - \mathbf{w}_2(0)] = [0.92, 0.76, 0.28, 0.12]^T$$

$$\mathbf{w}_1(1) = \mathbf{w}_1(0) = [0.2, 0.6, 0.5, 0.9]^T$$

Step 3: randomly select another sample, say x_2, we have:

$$d_1 = \|\mathbf{w}_1(1) - \mathbf{x}_2\|^2 = 0.66$$

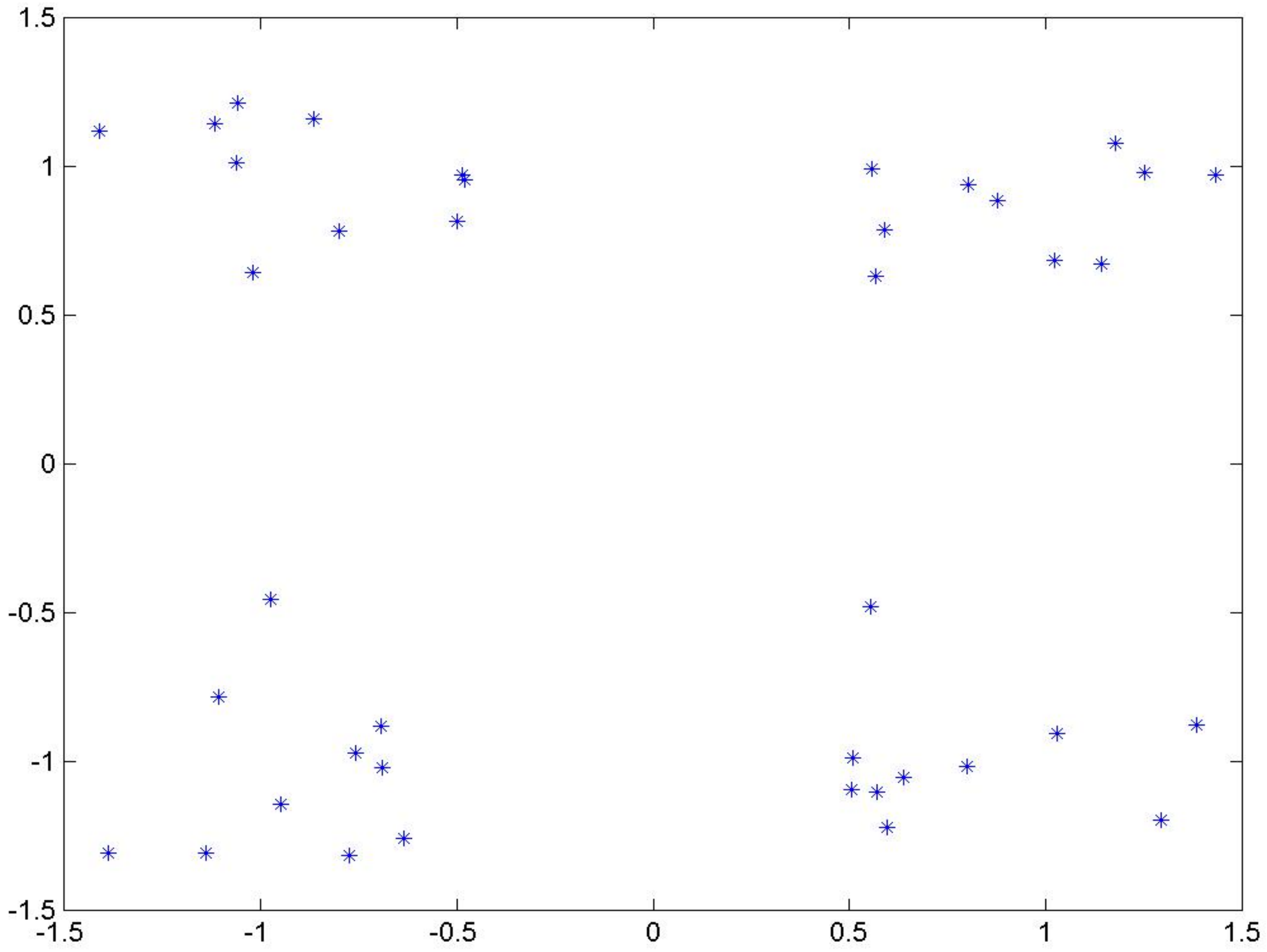
$$d_2 = \|\mathbf{w}_2(1) - \mathbf{x}_2\|^2 = 2.28$$

So the first neuron wins, i($\mathbf{x}$)=1; update the weights:

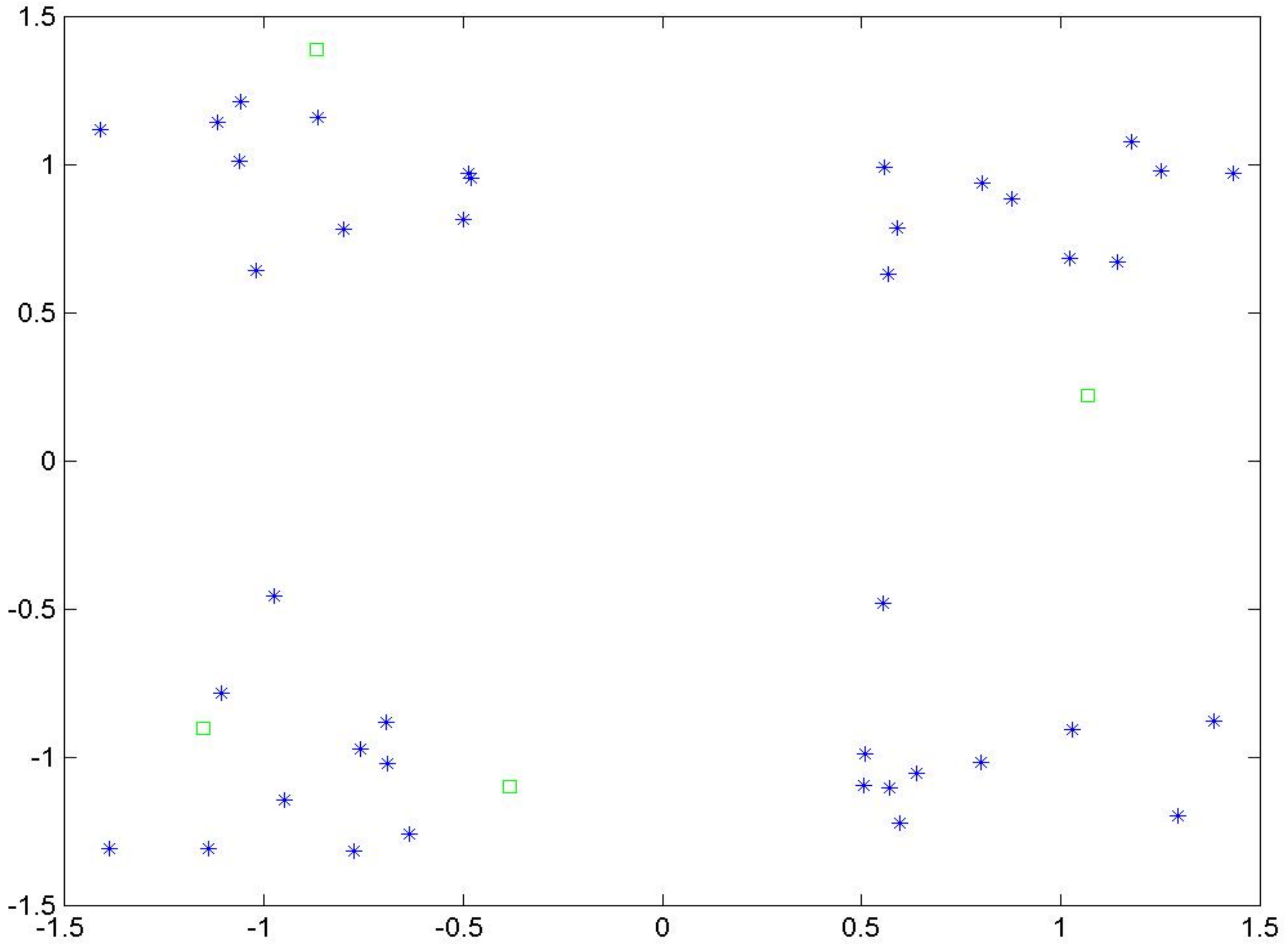
$$\mathbf{w}_1(2) = \mathbf{w}_1(1) + 0.3[\mathbf{x}_2 - \mathbf{w}_1(1)] = [0.14, 0.42, 0.35, 0.93]^T$$

$$\mathbf{w}_2(2) = \mathbf{w}_2(1) = [0.92, 0.76, 0.28, 0.12]^T$$

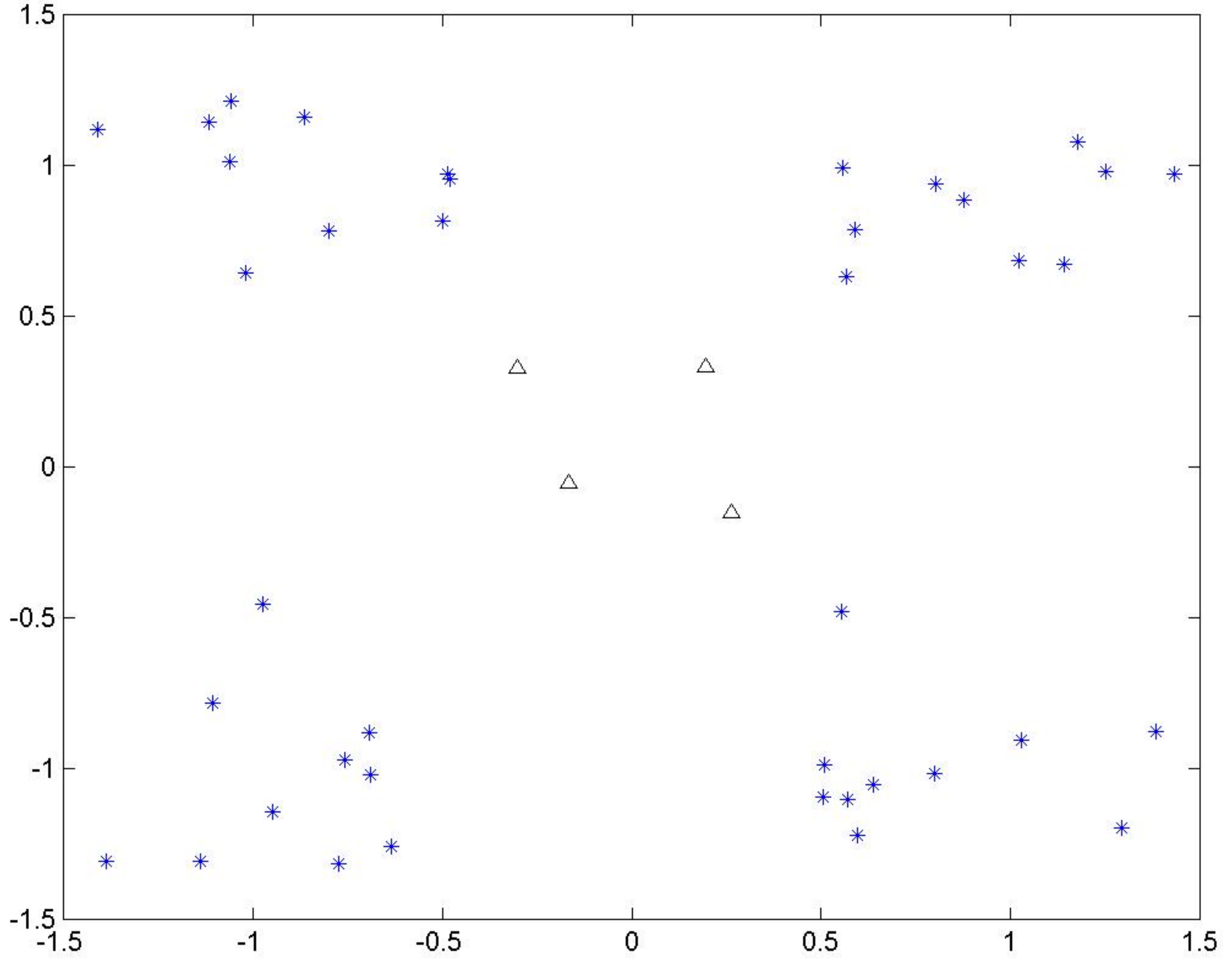
Example 2



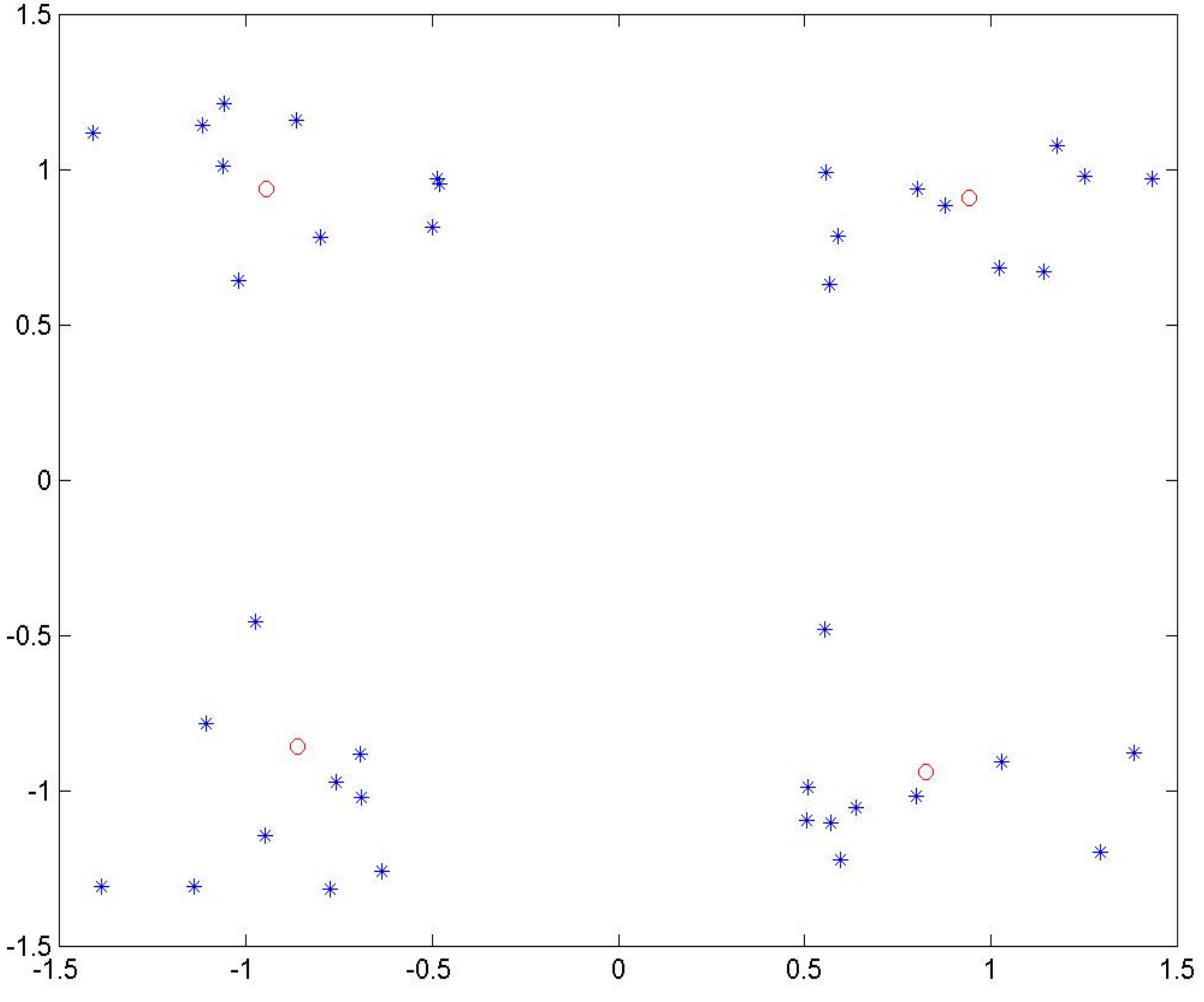
Initial weight vectors:



After the self-organizing phase (denoted by Δ)



After the convergence phase (denoted by “o”)



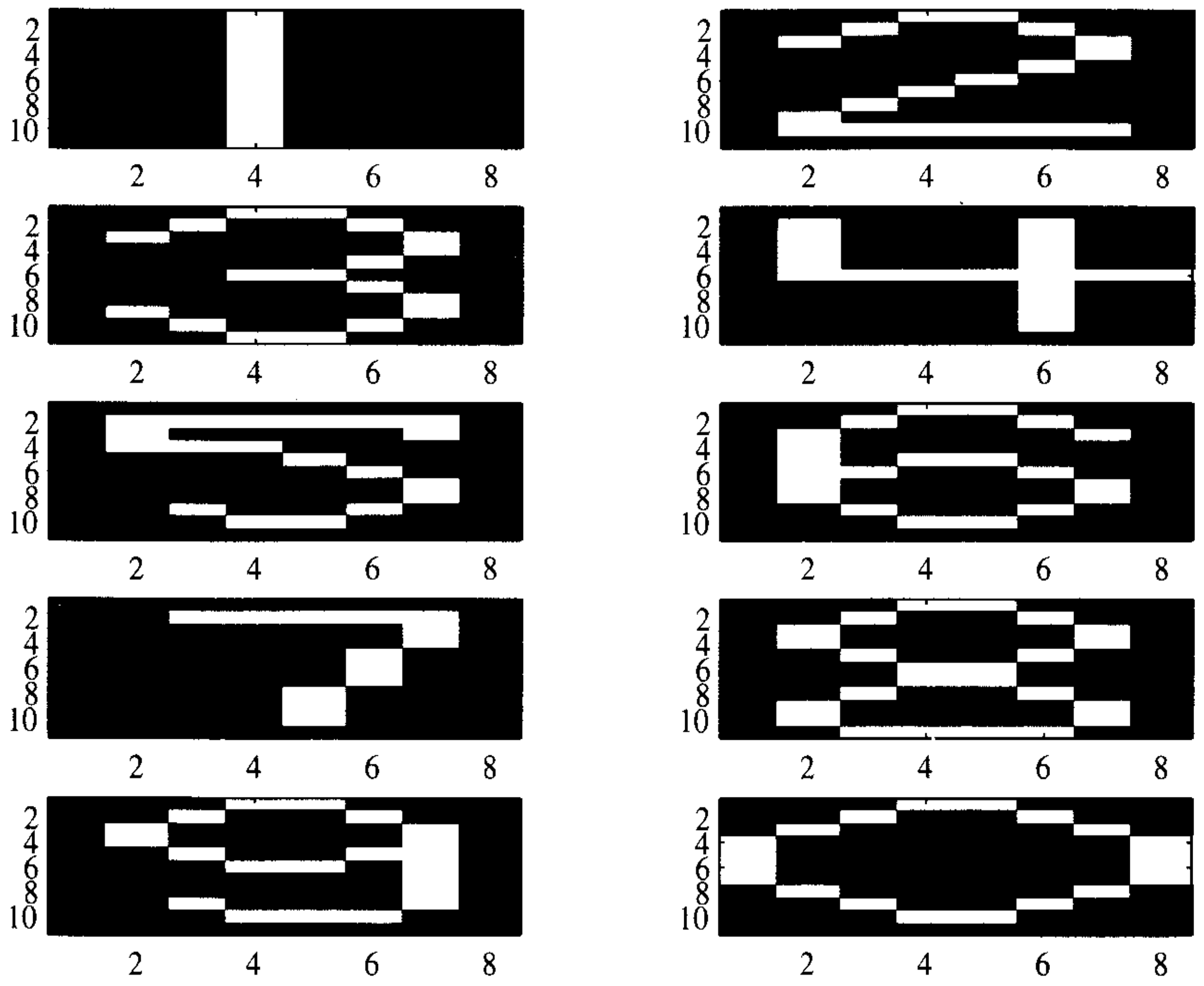
Properties of the Feature Map

Once the SOM algorithm is converged, the feature map produced by the algorithm displays important statistical characteristics of the input space.

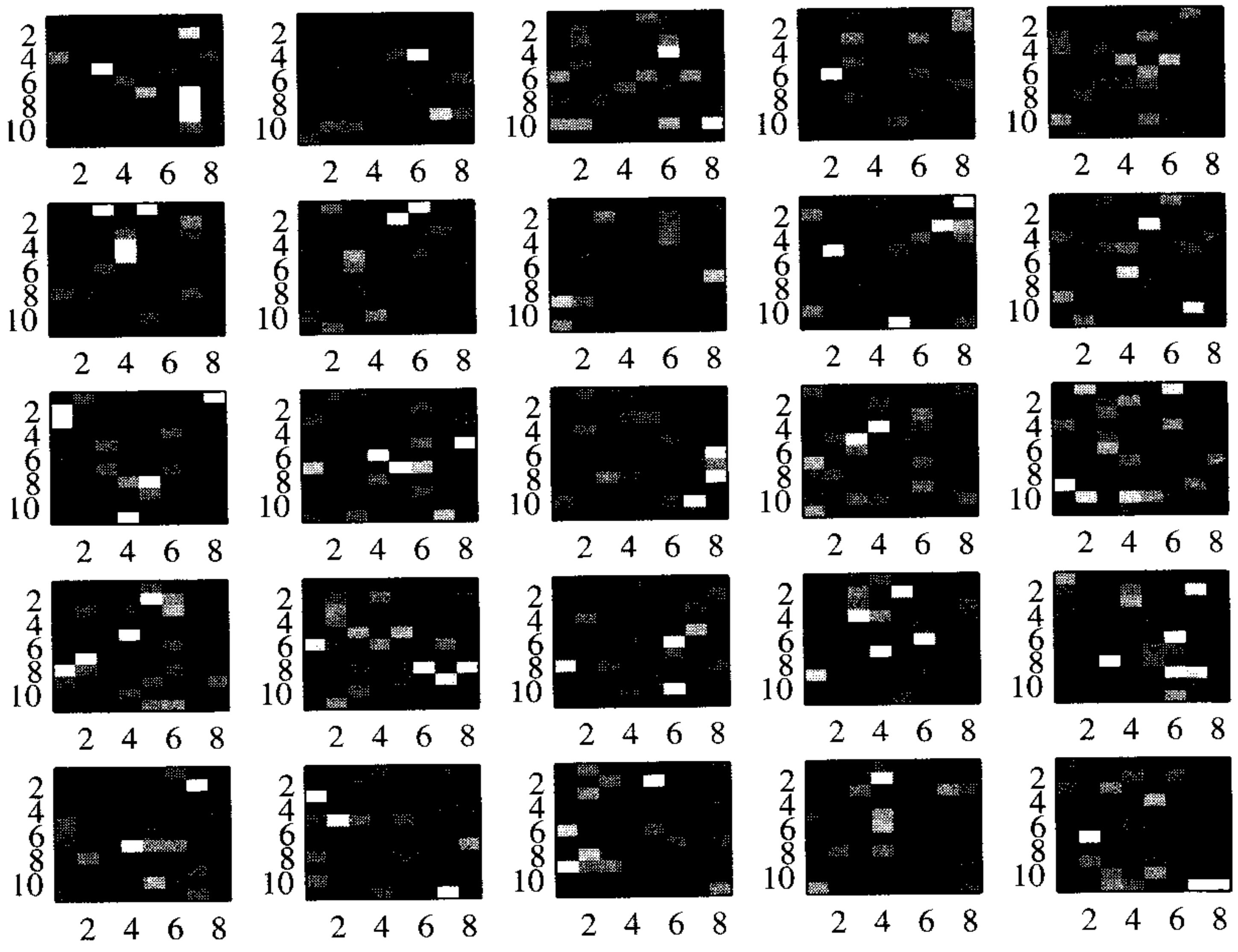
(1) Approximation of the input space. The feature map represented by a set of weight vectors in the output space provide a good approximation to the input space. The basic aim is to store a large set of input vectors by finding a smaller set of prototypes.

(2) Topological ordering. The feature map computed by the SOM algorithm is topologically ordered in the sense that the spatial location of a neuron in the lattice corresponds to a particular domain of the input patterns. This property is a direct consequence of the updated equation that forces weight vector $\mathbf{w}_i$ of the winning neuron to move toward $\mathbf{x}$.

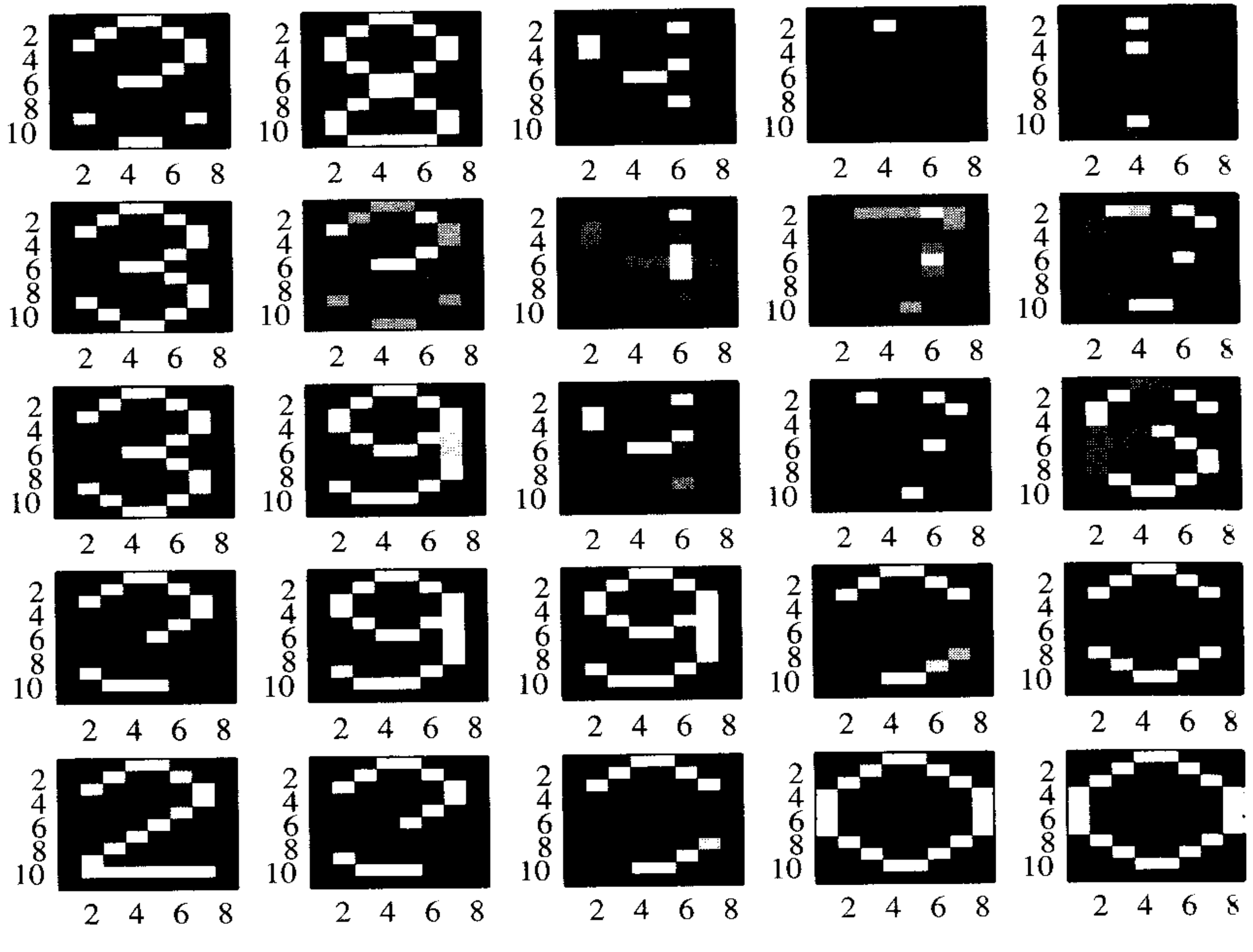
Consider 10 digits, represented by 11×8 dot matrices, we build a 5 ×5 2-D SOM map.



Initial weights are generated randomly and are shown below:



The resulted SOM, with weights viewed as digits, is shown below. Obviously, weights of adjacent neurons are similar.

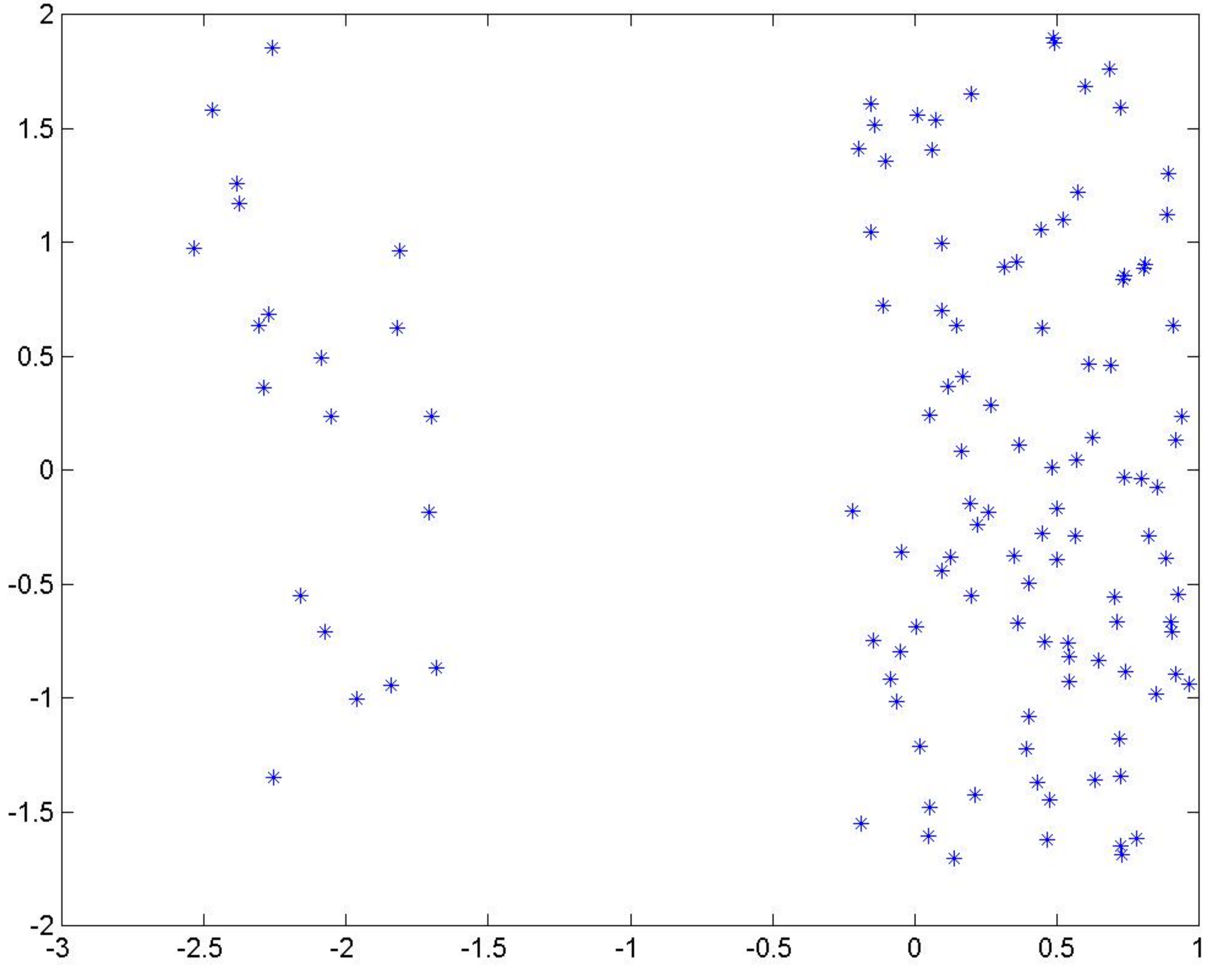


(3) Density Matching. The map reflects variations in the statistics of the input distribution: regions in the input space from which sample are drown with a high probability of occurrence are mapped onto larger domains of the output space, and therefore with better resolution than regions in the input space from which samples are drawn with a low probability of occurrence.

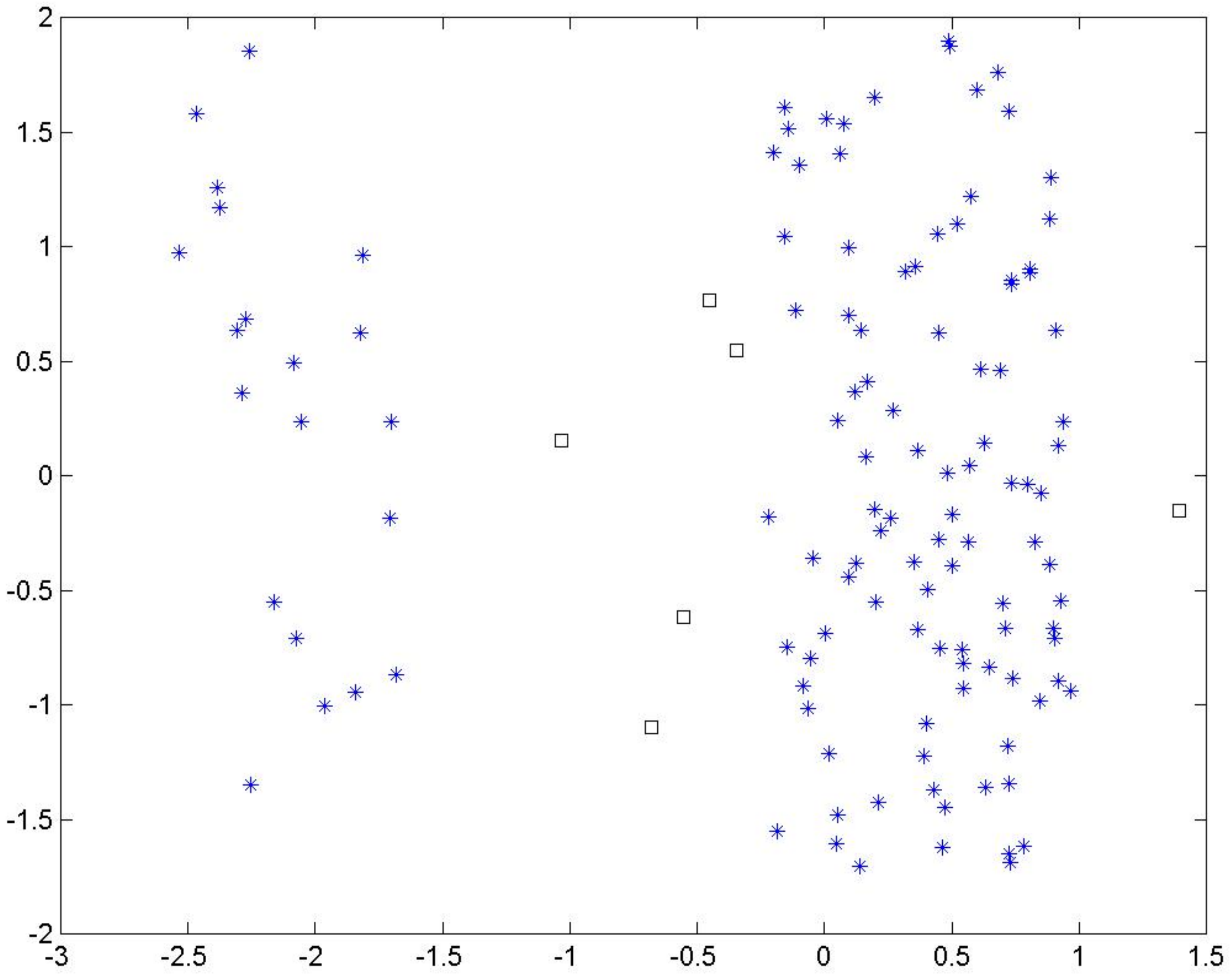
Consider the example shown below, where 20 samples (1/6) are from cluster 1, and 100 samples (5/6) are from cluster 2.

After training, 1 of the 6 neurons corresponds to cluster 1, and 5 of the 6 neurons correspond to cluster 2.

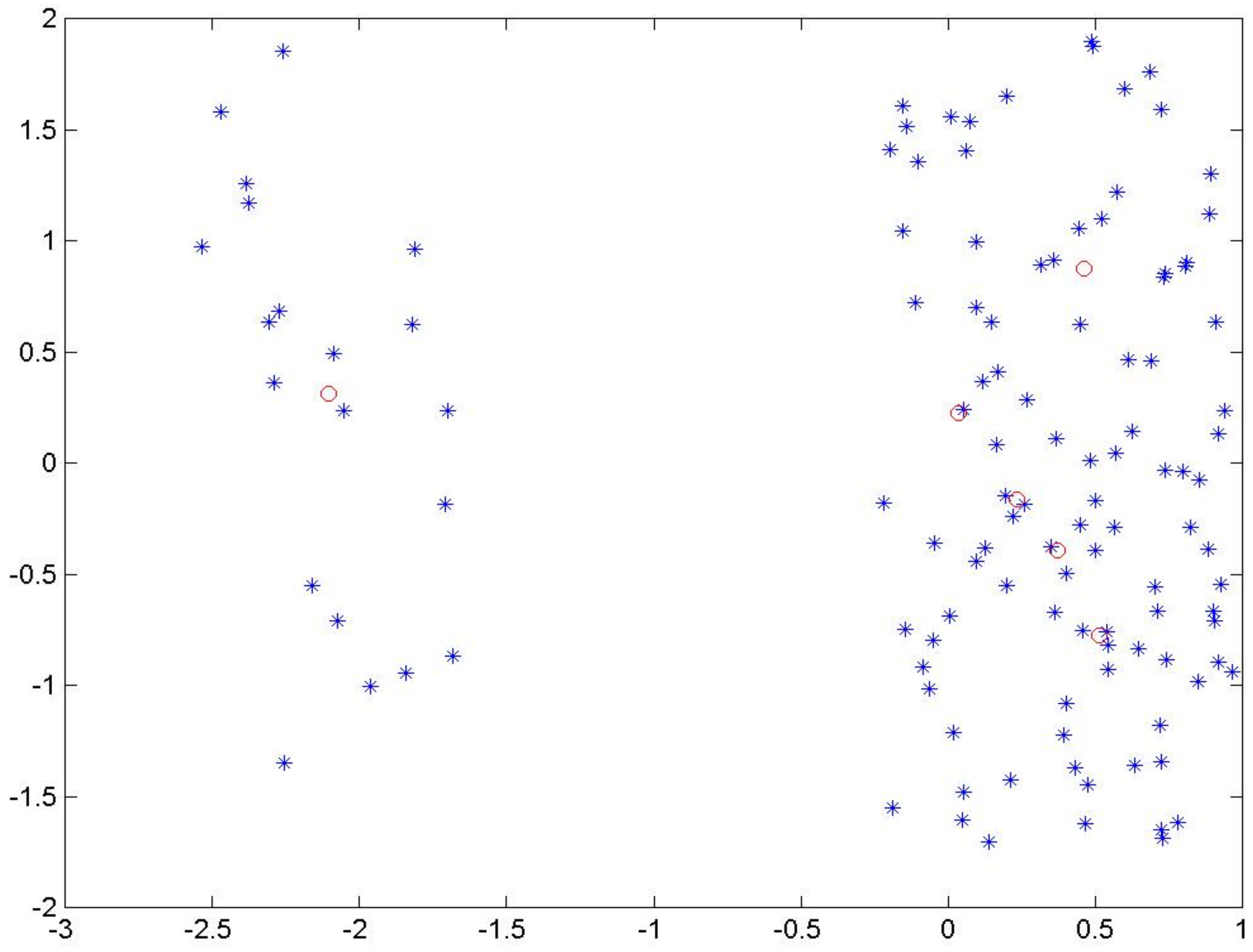
Samples:



Initial weight vectors (denoted by □)

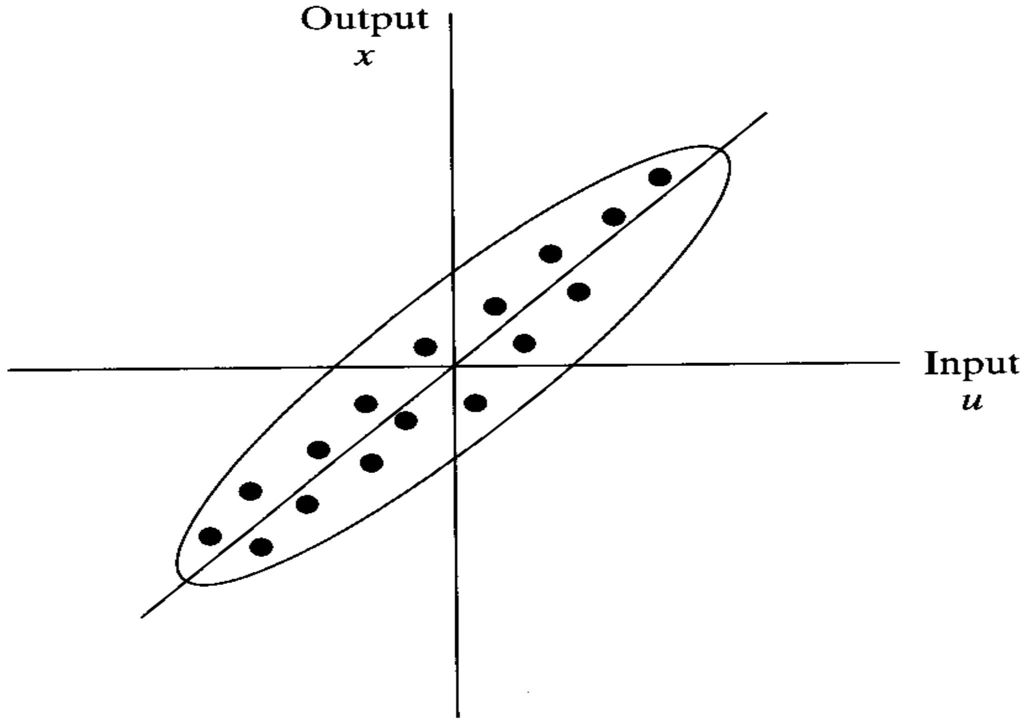


The weight vectors after convergence phase (denoted by o)

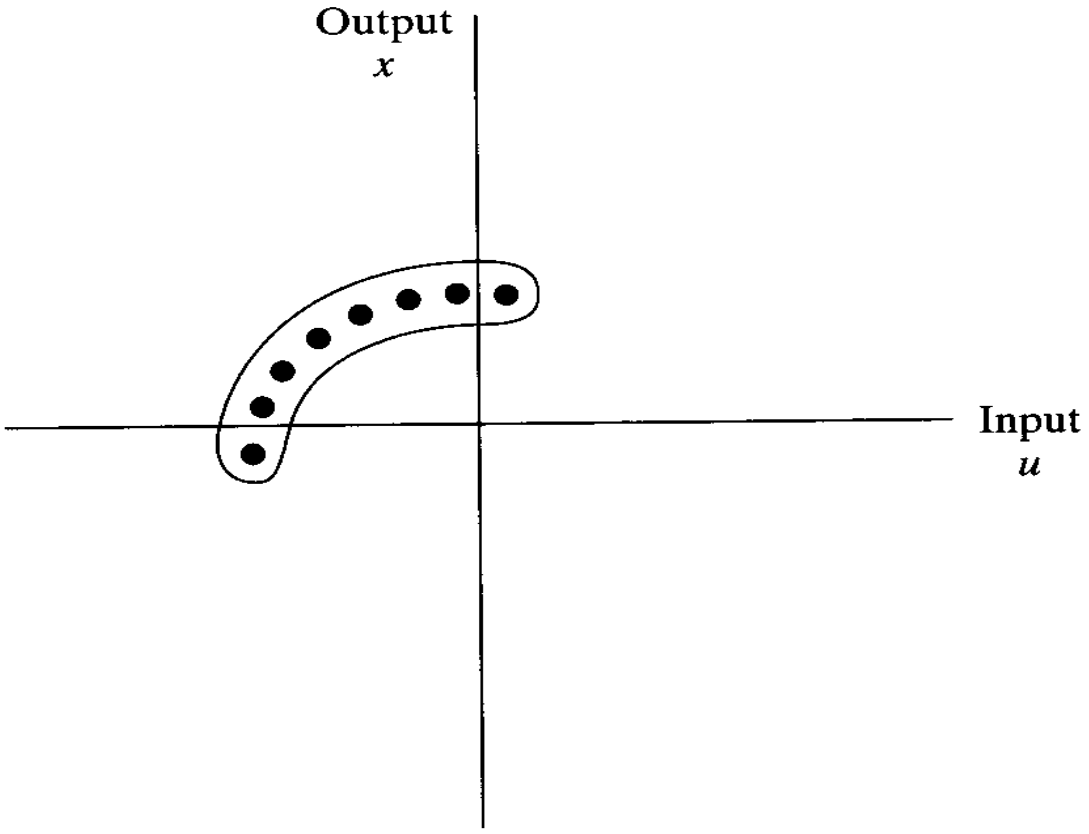


(4) Given data from an input space with nonlinear distribution, the self-organizing map is able to select a subset of best features for approximating the underlying distributions.

For a linear mapping between u and x as shown below, we can use a straight line to describe the mapping.



For a nonlinear mapping shown below, it is impossible to use a straight line (linear equation) to describe the data. But the use of a self-organizing map built on a one-dimensional lattice of neurons is able to overcome this approximation problem.


Output
x
Input
u