• Economic representation of data with all their interrelationships is one of the most central problems in information sciences.

• Such an operation is obviously characteristic of the operation of the brain.

• In thinking, and in the sub-conscious information processing, there is a general tendency to compress information by forming **reduced representations of the most relevant facts**, without loss of knowledge about their interrelationships.

• The purpose of intelligent information processsing seems in general to be creation of simplified images of the observable world at various levels of abstraction.

• Cerebral cortex is is organized according according to different sensory modalities

• There also areas performing specialized tasks, e.g. speech control and analysis of sensory signals (visual, auditory, somatosensory, etc.)

• Associative areas onto which signals of different modality converge

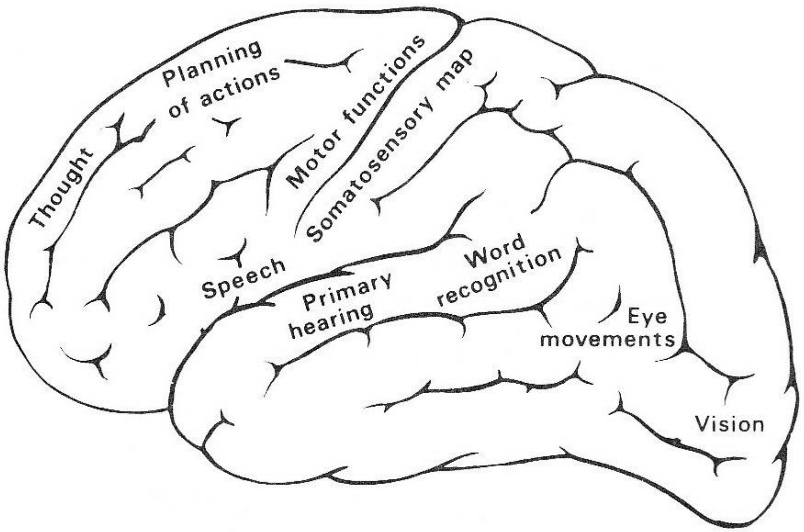


Fig. 1.1. Brain areas

• Fine structure of the areas: the visual, somatosensory, etc., response signals are abtained in the same topographical order on the cortex in which they were received at the sensory organs.

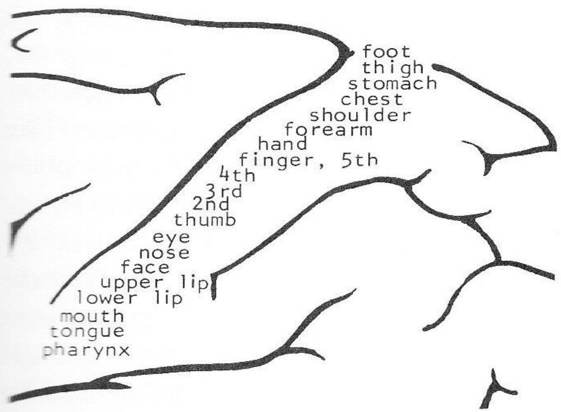


Fig. 1.2. The somatosensory map

However, the "receptive fields" could also be defined more generally; the brain contains cells which respond to a particular *feature* of the environment, e.g., to an area in the visual field, or to a range in an abstract feature space like that of acoustic frequencies (Fig. 1.3) or colours. Of course, a single cell alone may not possess such an encoding ability: the specificity of the response more probably results from computations made in a hierarchy of cells converging upon this place.

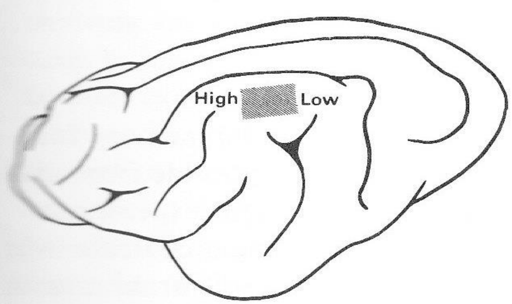
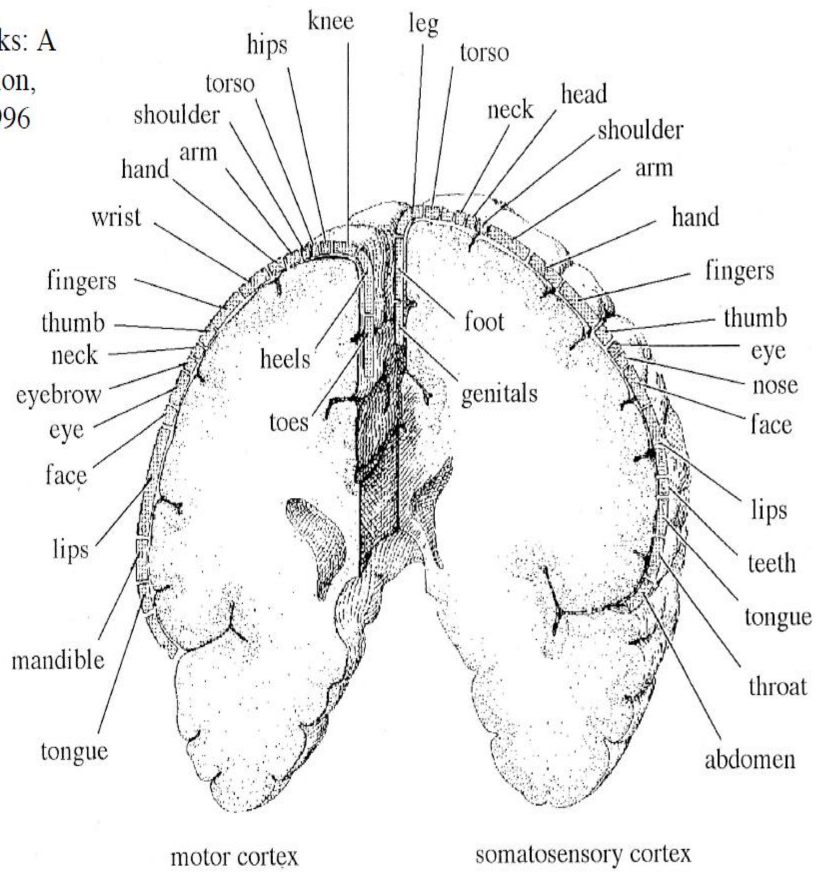


Fig. 1.3. The tonotopic map (of cat)

The Somatosensory and Motor Cortex

From: Neural Networks: A Systematic Introduction, R. Rojas, Springer, 1996



WHAT IS SELF-ORGANIZATION?

Self-organization is a process whereby pattern at the global level of a system emerges solely from interactions among the lower-level components of the system. The rules specifying the interactions among the system's components are executed using only local information, without reference to the global pattern. Examples of self-organization include a wide range of pattern formation processes in both physical and biological systems: sand grains assembling into rippled dunes, chemical reactants forming swirling spiral patterns, the patterns on sea shells, or fish swimming in coordinated schools (Figure 1). 'Pattern' is used here in a broad sense to refer not only to a particular ar-

What is Self-Organization?

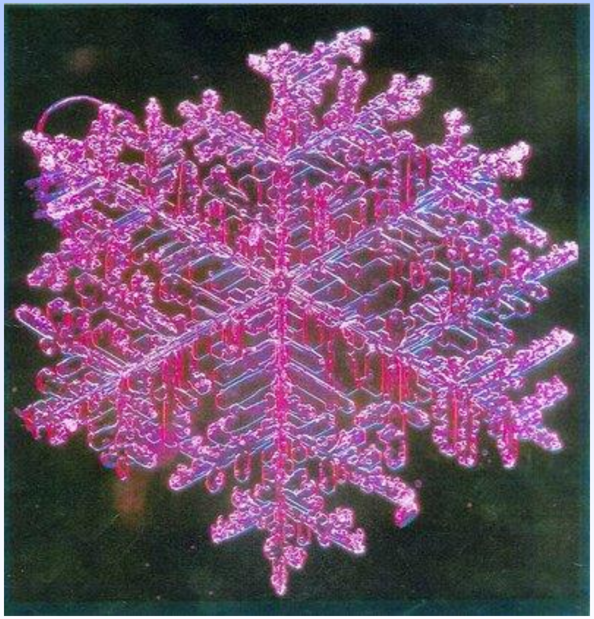
Self-organization is a process of attraction and repulsion in which the internal organization of a system, normally an open system, increases in complexity without being guided or managed by an outside source

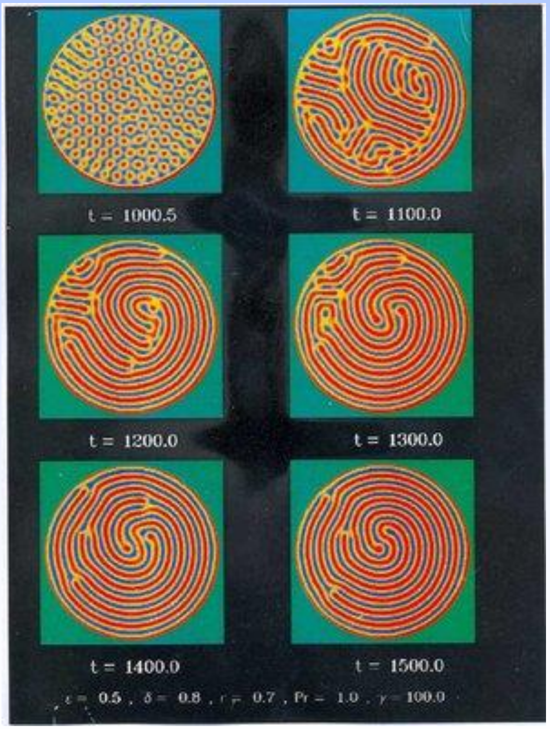
Self-organizing systems typically (but not always) display emergent properties

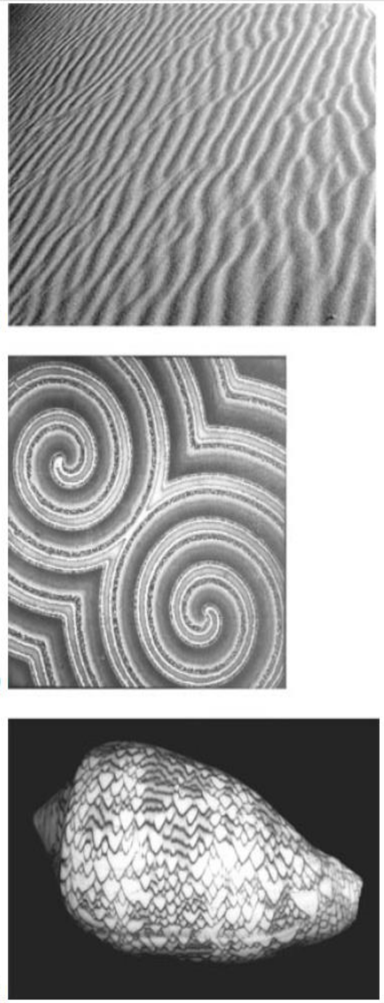
What is Self-Organization?

Self-organization is the spontaneous often seemingly purposeful formation of spatial, temporal, spatio-temporal structures or functions in systems composed of few or many components. In physics, chemistry and biology self-organization occurs in open systems driven away from thermal equilibrium.

The process of self-organization can be found in many other fields also, such as economy, sociology, medicine, technology.

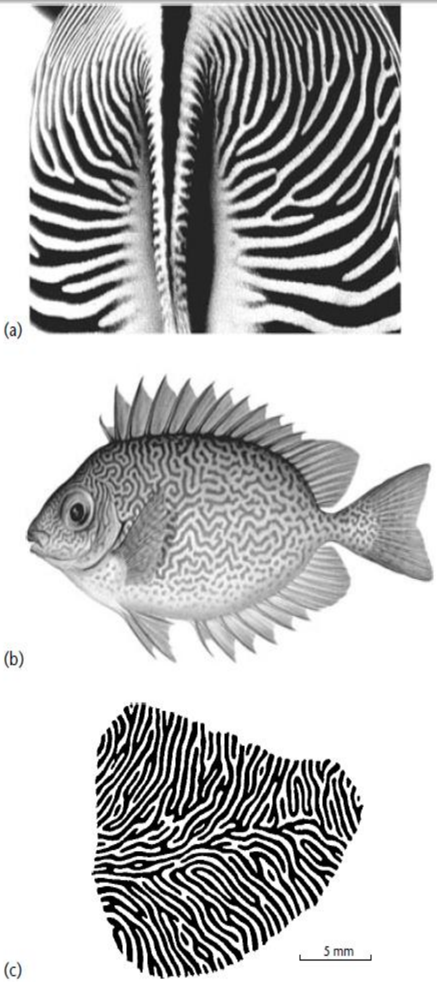






(a) (b) (c)

Figure 1. Examples of self-organized pattern formation in physical, chemical, and biological systems. (a) Sand dune stripes. (b) Belusov–Zhabotinsky chemical reaction (image courtesy of Stefan C. Müller). (c) A cone shell from Ceylon.



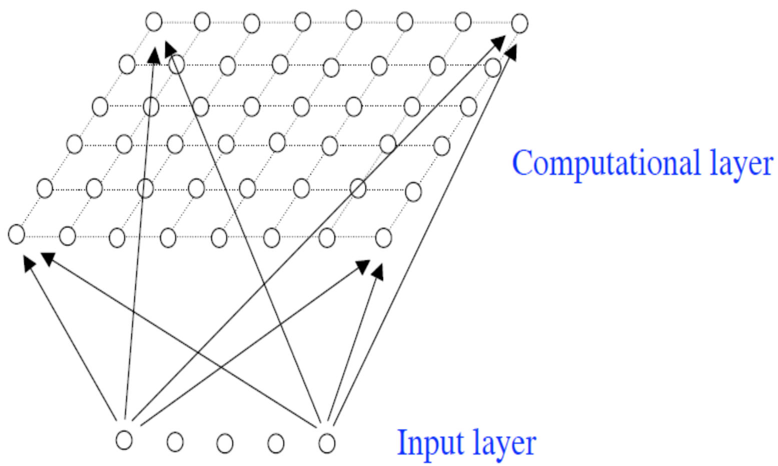
(a) (b) (c)

Figure 2. Striped and mottled patterns found in biological systems. (a) Alternating stripes on a zebra's coat (*Equus grevii*). (b) Mottled pattern of pigments on the skin of a vermiculated rabbitfish (*Siganus vermiculatus*). (c) Ocular dominance stripes in the visual cortex of a macaque monkey. Regions receiving inputs from one eye are shown in black, and regions receiving inputs from the other eye are shown in white. Adapted from: Hubel DH and Wiesel TN (1977) Functional architecture of the macaque monkey visual cortex. *Proceedings of the Royal Society, Series B* **198**: 1–59.

their diffusion rates, the initial distribution of the cells, and their thresholds for pigment production.

Kohonen Networks

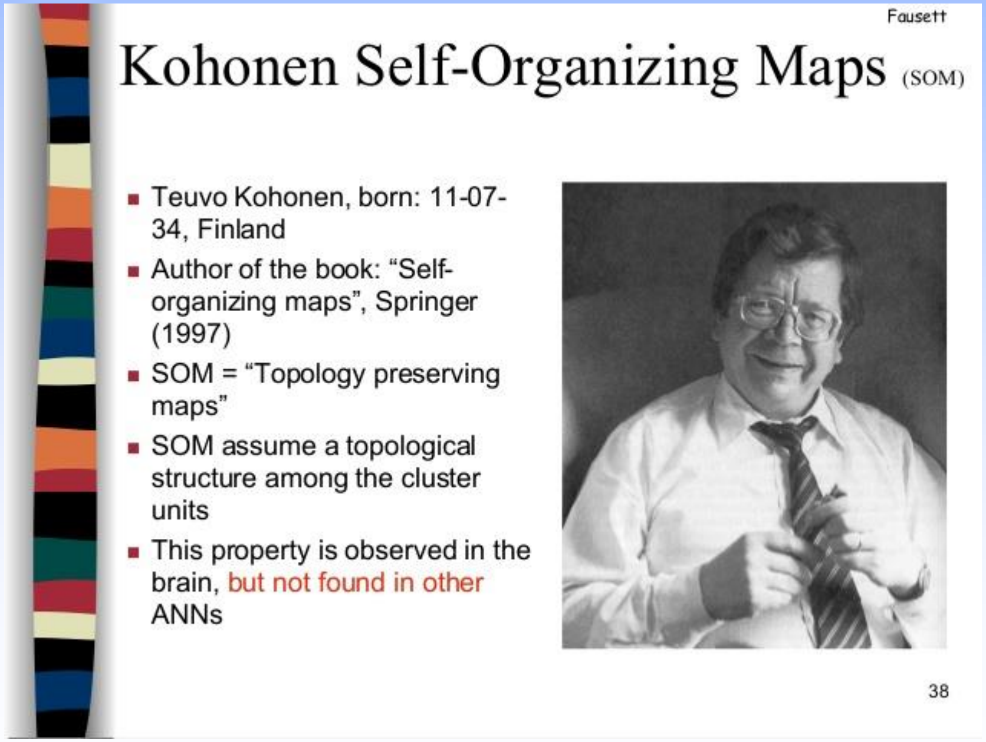
We shall concentrate on the particular kind of SOM known as a ***Kohonen Network***. This SOM has a feed-forward structure with a single computational layer arranged in rows and columns. Each neuron is fully connected to all the source nodes in the input layer:

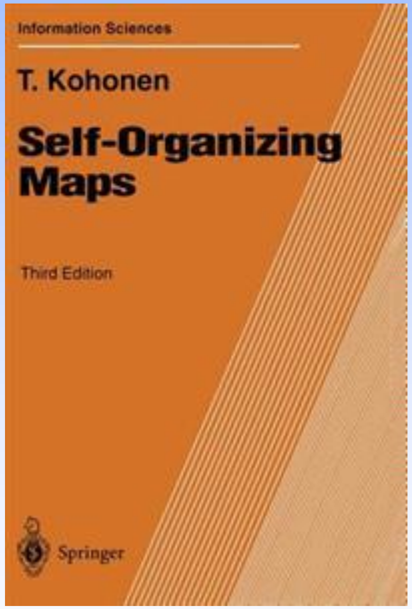


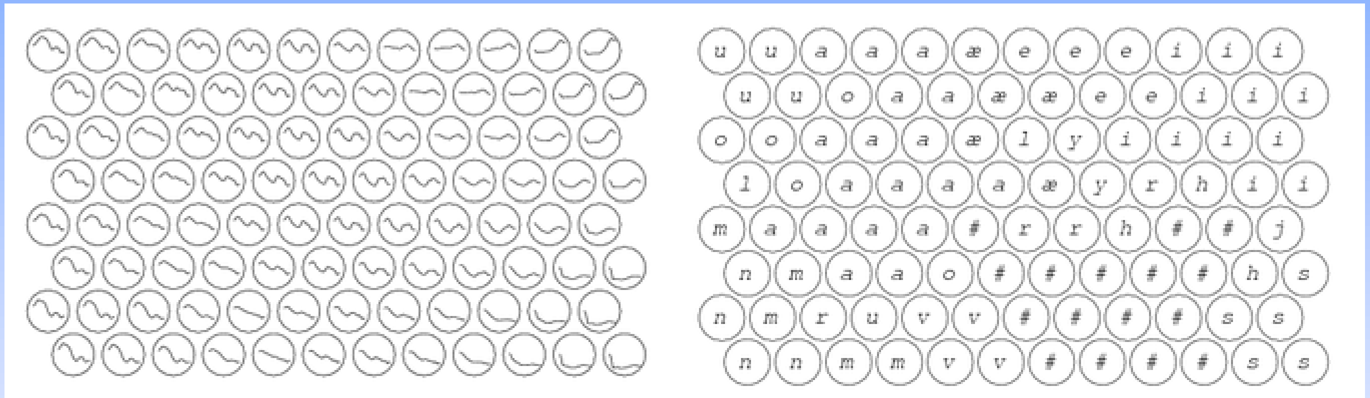
Clearly, a one dimensional map will operate in the same way but just have a single row (or a single column) in the computational layer.

- The Self-Organizing Map (SOM) is one of the most popular neural network models.
- It belongs to the category of competitive learning networks.
- The Self-Organizing Map is based on unsupervised learning, which means that no human intervention is needed during the learning and that little needs to be known about the characteristics of the input data.
- We could, for example, use the SOM for clustering data without knowing the class memberships of the input data. The SOM can be used to detect features inherent to the problem and thus has also been called SOFM, the Self-Organizing Feature Map.

- The Self-Organizing Map was developed by professor Kohonen.





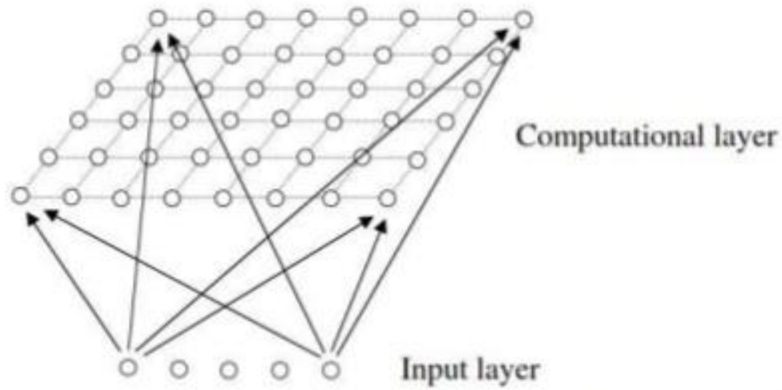

u u a a a æ e e e i i i
u u o a a æ æ e e i i i
o o a a a æ l y i i i i
l o a a a a æ y r h i i
m a a a a # r r h # # j
n m a a o # # # # # h s
n m r u v v # # # # s s
n n m m v v # # # # s s

- The SOM algorithm is based on unsupervised, competitive learning.
- It provides a topology preserving mapping from the high dimensional space to map units.
- Map units, or neurons, usually form a two-dimensional lattice and thus the mapping is a mapping from high dimensional space onto a plane.
- The property of topology preserving means that the mapping preserves the relative distance between the points. Points that are near each other in the input space are mapped to nearby map units in the SOM.
- The SOM can thus serve as a cluster analyzing tool of high-dimensional data.
- Also, the SOM has the capability to generalize. Generalization capability means that the network can recognize or characterize inputs it has never encountered before. A new input is assimilated with the map unit it is mapped to.

(http://users.ics.aalto.fi/jhollmen/dippa/node9.html)

Kohonen Networks

We shall concentrate on the particular kind of SOM known as a **Kohonen Network**. This SOM has a feed-forward structure with a single computational layer arranged in rows and columns. Each neuron is fully connected to all the source nodes in the input layer:

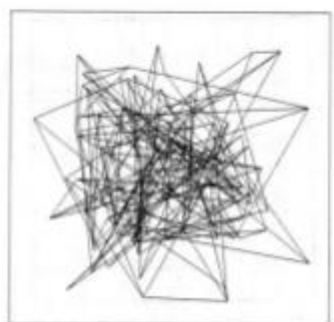


Clearly, a one dimensional map will just have a single row (or a single column) in the computational layer.

Illustration of Kohonen Learning

Inputs: coordinates (x,y) of points drawn from a square

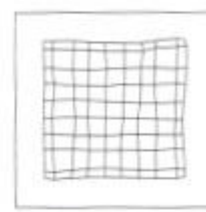
Display neuron j at position x_j, y_j where its s_j is maximum



random initial positions

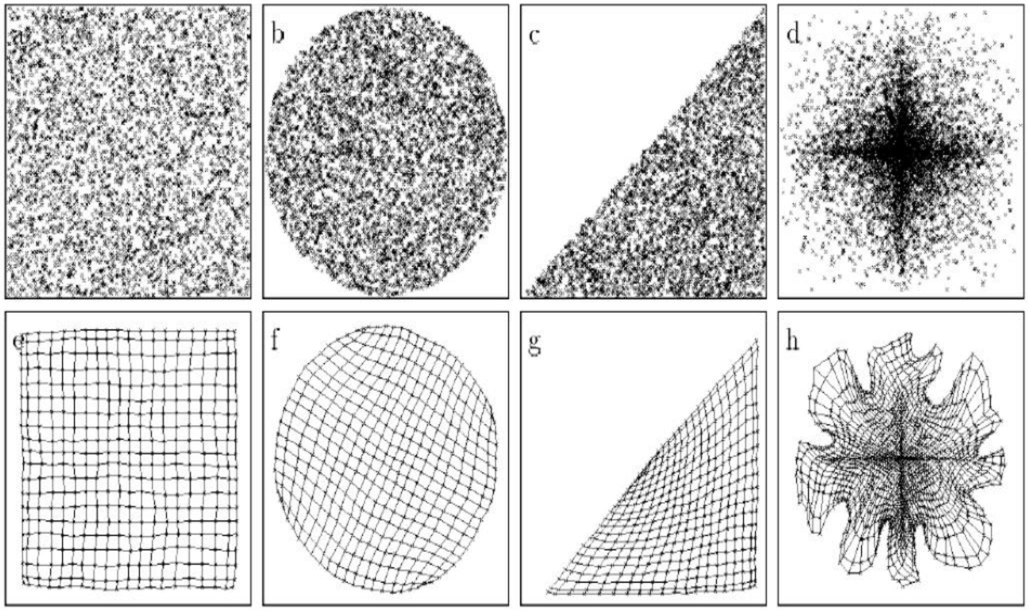
100 inputs

200 inputs



1000 inputs

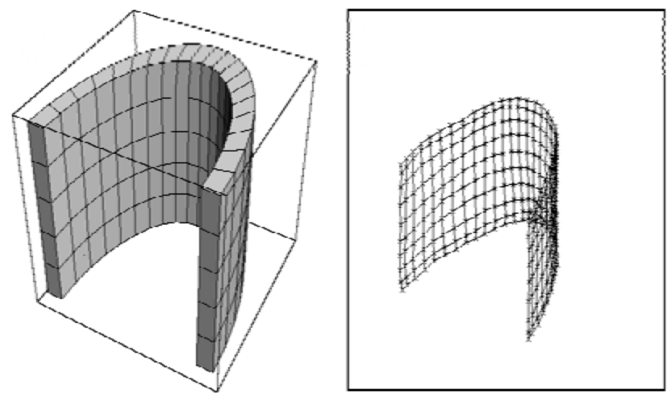
2-D examples (after convergence)



The grid mimics the initial distribution

SOM used as data analysis tool

- Mapping (projection) of a continuous distribution to a discrete set (the centroids)



- After learning: nearest neighbour rule in the input space

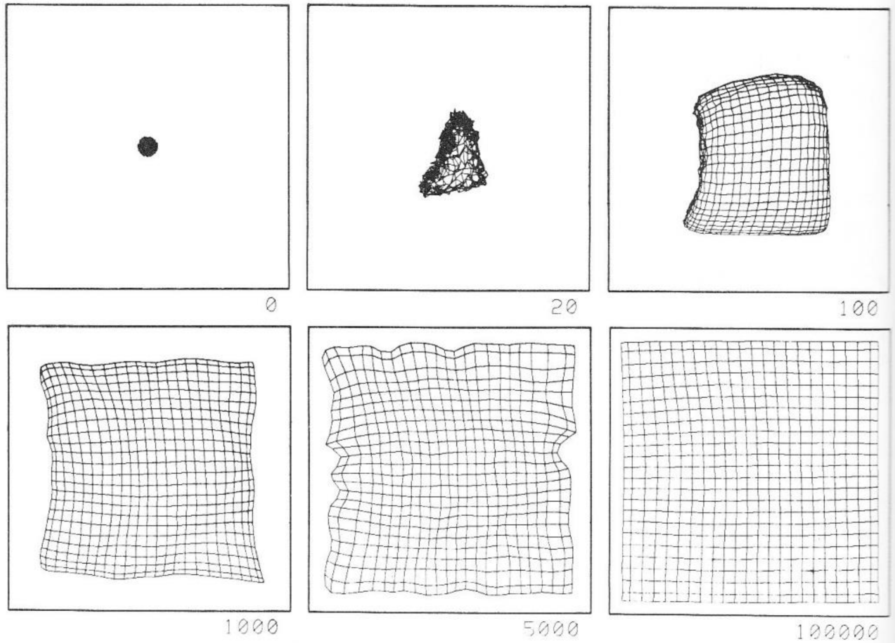


Fig. 5.16. Weight vectors during the ordering process

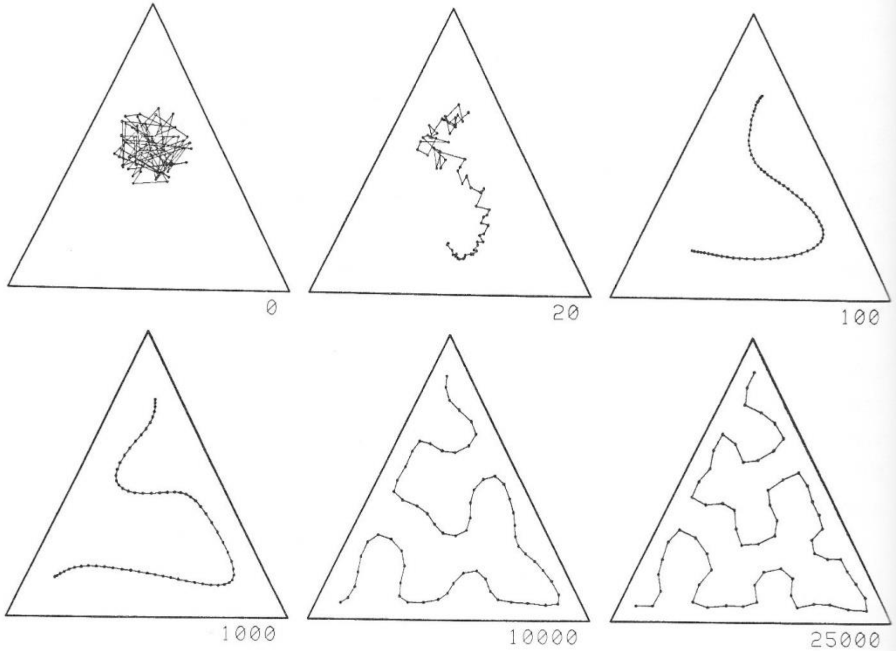
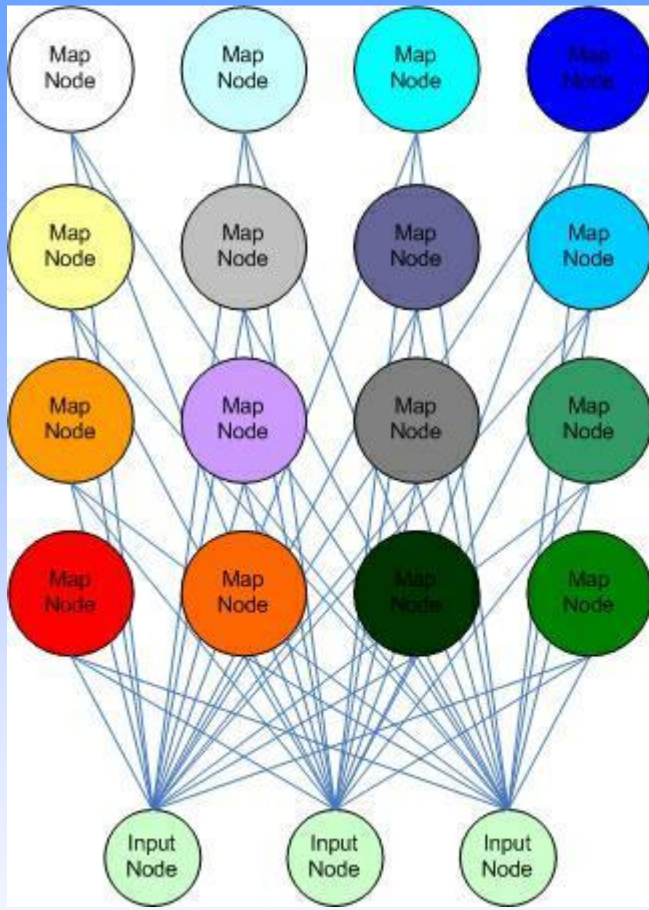
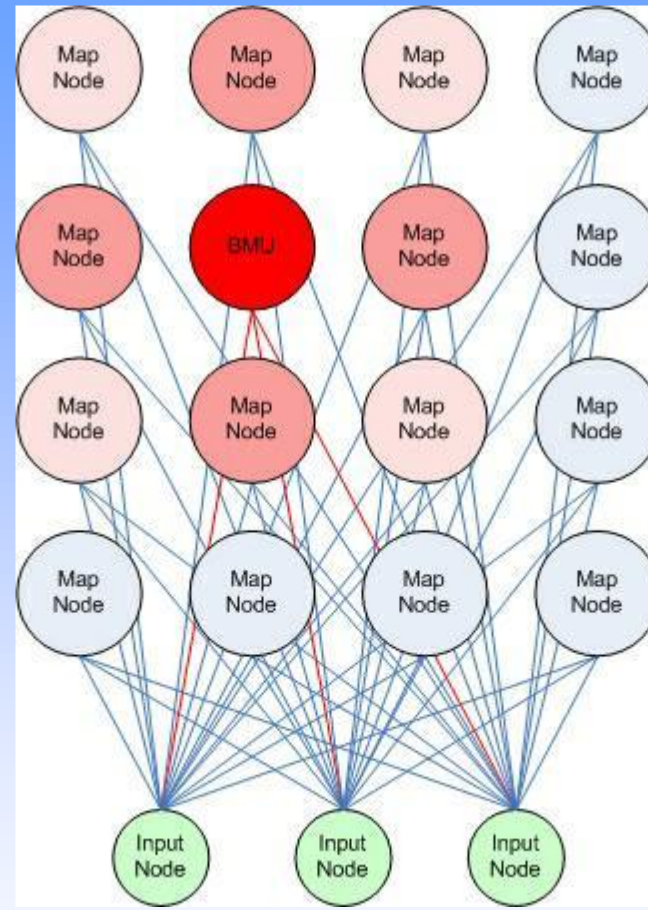


Fig. 5.17. Weight vectors during the ordering process


Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Input Node
Input Node
Input Node


Map Node
Map Node
Map Node
Map Node
Map Node
BMU
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Map Node
Input Node
Input Node
Input Node

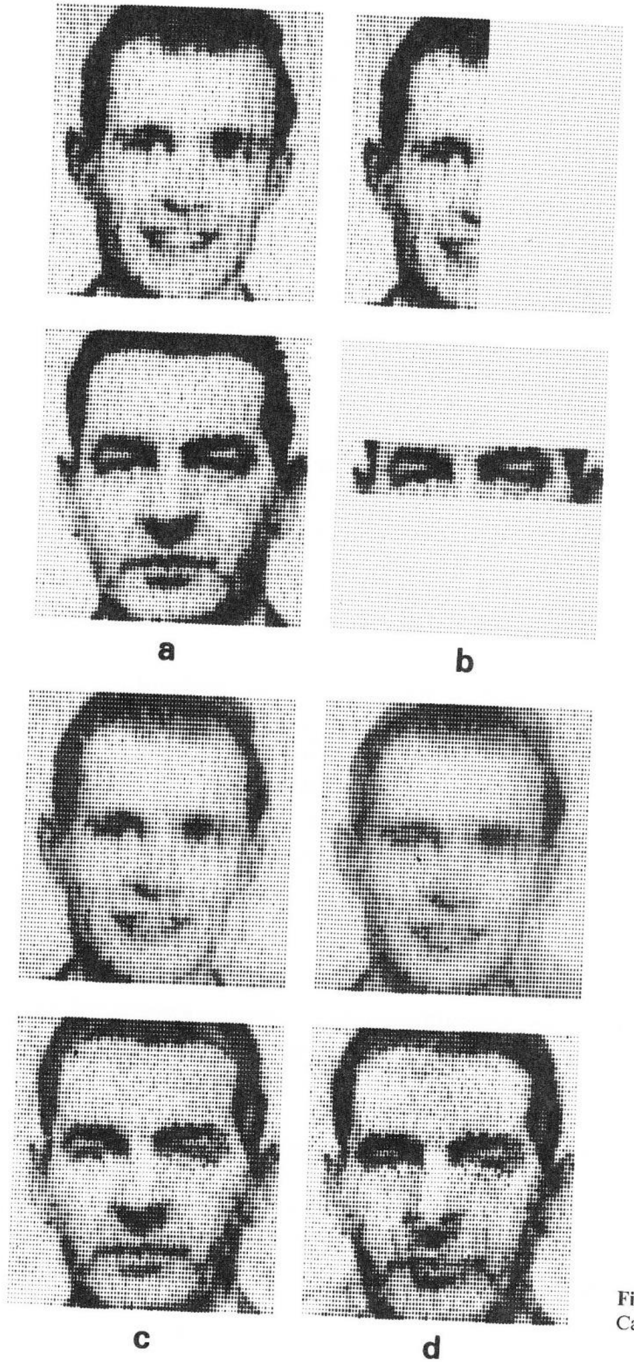
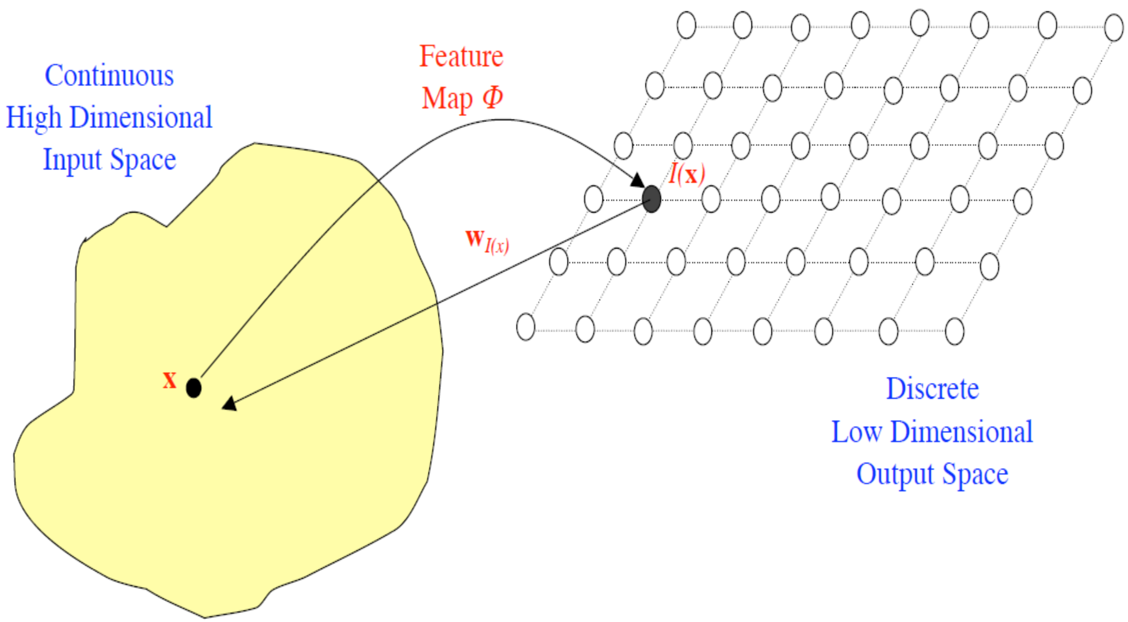


Fig. 1.4a – d.
Caption see opposite page

Organization of the Mapping

Each point $\mathbf{x}$ in the input space maps to a corresponding point $I(\mathbf{x})$ in the output space:



Each point I in the output space maps to a corresponding point $\mathbf{w}(I)$ in the input space.

Similarity Matching

$$\|x(t_k) - m_c(t_k)\| = \min_i \{\|x(t_k) - m_i(t_k)\|\} . \tag{5.6}$$

Updating

$$m_i(t_{k+1}) = m_i(t_k) + \alpha(t_k)[x(t_k) - m_i(t_k)] \quad \text{for} \quad i \in N_c ,$$
$$m_i(t_{k+1}) = m_i(t_k) \quad \text{otherwise} .$$

As pointed out many times above, the algorithm chosen here for simulations is only representative of many alternative forms. We would like to demonstrate the phenomena as simply as possible; without simplifications, the computations become intolerably heavy.

The "topological neighbourhood" $N_c = N_c(t_k)$ (which is a function of the discrete-time index) may be defined in several ways, too. Numerous simulations have shown that the best results in self-organization are obtained if the neighbourhood is selected fairly wide in the beginning and then it is let to shrink with time (Fig. 5.11). Notice that at the borders of the array, N_c is not full.

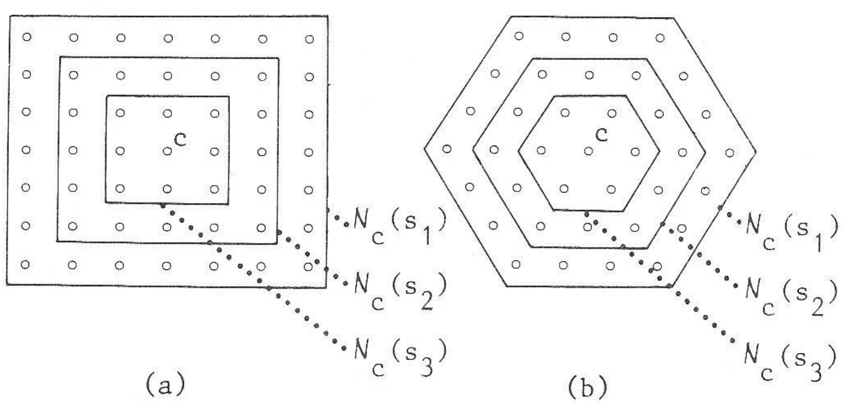
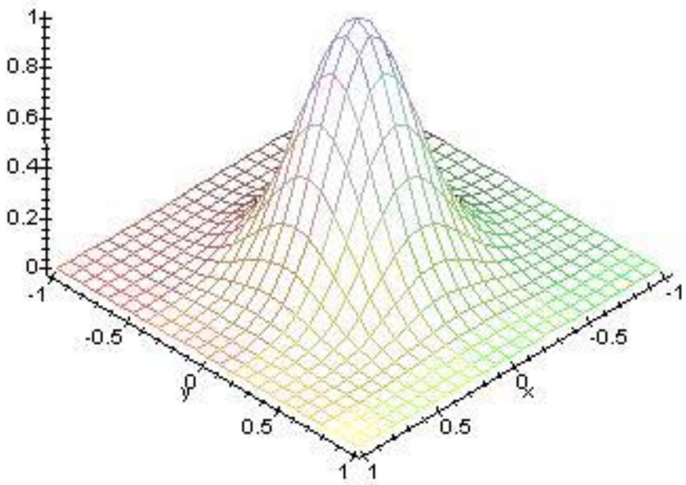


Fig. 5.11a, b. Two examples of topological neighbourhood ($s_1 < s_2 < s_3$)

Scale Neighbors

- Determining Neighbors
  - –Neighborhood size
    - Decreases over time
  - –Effect on neighbors



- Learning

$$\forall i \in N_c(t),$$
$$m_i(t+1) = m_i(t) + \alpha(t)[x(t) - m_i(t)]$$
otherwise,
$$m_i(t+1) = m_i(t)$$

The goal

- **We have to find values for the weight vectors of the links from the input layer to the nodes of the lattice, in such a way that adjacent neurons will have similar weight vectors.**
- **For an input, the output of the neural network will be the neuron whose weight vector is most similar (with respect to Euclidean distance) to that input.**
- **In this way, each (weight vector of a) neuron is the center of a cluster containing all the input examples which are mapped to that neuron.**

The learning process (1)

An informal description:

- **Given: an input pattern x**
- **Find: the neuron *i* which has closest weight vector by competition ($w_i^T x$ will be the highest).**
- **For each neuron *j* in the neighbourhood *N(i)* of the winning neuron *i*:**
  - **update the weight vector of *j*.**

The learning process (2)

- **Neurons which are not in the neighbourhood are left unchanged.**
- **The SOM algorithm:**
  - **Starts with large neighbourhood size and gradually reduces it.**
  - **Gradually reduces the learning rate η.**

The learning process (3)

- Upon repeated presentations of the training examples, the weight vectors tend to follow the distribution of the examples.

- This results in a topological ordering of the neurons, where neurons adjacent to each other tend to have similar weights.

The learning process (4)

- There are basically three essential processes:
  - competition
  - cooperation
  - weight adaption

The learning process (5)

- **Competition**:
  - **Competitive process: Find the best match of input vector *x* with weight vectors:**

$$i(x) = \arg\min_j \| x - w_j \| \qquad j = 1, 2, \ldots, \ell$$

winning neuron

total number of neurons

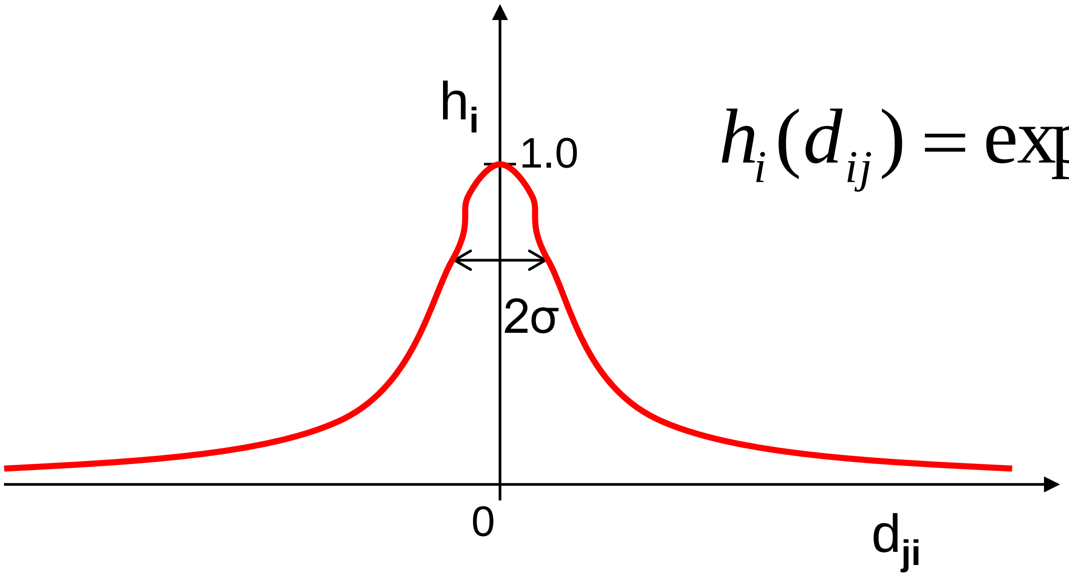
  - **The input space of patterns is mapped onto a discrete output space of neurons by a process of competition among the neurons of the network.**

The learning process (6)

- **Cooperation:**
  - **Cooperative process: The winning neuron locates the center of a topological neighbourhood of cooperating neurons.**
  - **The topological neighbourhood depends on lateral distance d_{ji} between the winner neuron *i* and neuron *j*.**

Learning Process (7) - neighbourhood function

– **Gaussian neighbourhood function**



$$h_i(d_{ij}) = \exp\left(-{d_{ij}}^2 \Big/ 2\sigma^2\right)$$

Learning process (8)

- σ **(effective width)** **measures degree to which excited neurons in the vicinity of the winning neuron participate to the learning process.**

  exponential decay update

  $$\sigma(n) = \sigma_0 \exp\left(-\, n / T_1 \right)$$

  time constant

- d_{ji}**: lateral distance**
  - **in one dimension lattice** $\| j - i \|$
  - **in two dimension lattice** $\| r_j - r_i \|$
    r_j **is the position of neuron** j **in the lattice.**

Learning process (9)

- **Applied to all neurons inside the neighbourhood of the winning neuron *i*.**

$$\Delta w_j = \underbrace{\eta y_j x}_{\text{Hebbian term}} - \underbrace{g(y_j) w_j}_{\text{forgetting term}}$$

Hebbian term **forgetting term**

scalar function of response y_j

$$g(y_j) = \eta y_j$$

$$y_j = h_{i,j(x)}$$

$$w_j(n+1) = w_j(n) + \eta(n)\, h_{ij(x)}(n) \left(\mathrm{x} - w_j(n)\right)$$

exponential decay update:

$$\eta(n) = \eta_0 \exp\left(-\,{}^{n}\!/_{T_2}\right)$$

Two phases of weight adaption

- **Self organising or ordering phase:**
  - **Topological ordering of weight vectors.**
  - **May take 1000 or more iterations of SOM algorithm.**
- **Important choice of parameter values:**
  - **$\eta(n)$:** $\eta_0 = 0.1$ $T_2 = 1000$
    $\Rightarrow$ **decrease gradually** $\eta(n) \geq 0.01$
  - **$h_{ji(x)}(n)$:** σ_0 **big enough** $T_1 = \dfrac{1000}{\log(\sigma_0)}$
  - **Initially the neighbourhood of the winning neuron includes almost all neurons in the network, then it shrinks slowly with time.**

Two phases of weight adaption

- **Convergence phase:**
  - **Fine tune feature map.**
  - **Must be at least 500 times the number of neurons in the network ⇒ thousands or tens of thousands of iterations.**
- **Choice of parameter values:**
  - **$\eta(n)$ maintained on the order of 0.01.**
  - **$h_{ji(x)}(n)$ contains only the nearest neighbours of the winning neuron. It eventually reduces to one or zero neighbouring neurons.**

A summary of SOM

- **Initialization:** **choose random small values for weight vectors such that $w_j(0)$ is different for all neurons j.**
- **Sampling:** **drawn a sample example x from the input space.**
- **Similarity matching:** **find the best matching winning neuron $i(x)$ at step n:**

$$i(x) = \arg\min_j \| x(n) - w_j \| \qquad j \in [1, 2, \dots, \ell]$$

- **Updating:** **adjust synaptic weight vectors**

$$w_j(n+1) = w_j(n) + \eta(n)\, h_{ij(x)}(n) \left(\mathrm{x} - w_j(n) \right)$$

- **Continuation:** **go to Sampling step until no noticeable changes in the feature map are observed.**

Example 1

A 2-dimensional lattice driven by a 2-dimensional distribution:

- **100 neurons arranged in a 2D lattice of 10 x 10 nodes.**

- **Input is bidimensional: $x = (x_1, x_2)$ from a uniform distribution in a region defined by:**

  $$\{ (-1 < x_1 < +1); (-1 < x_2 < +1) \}$$

- **Weights are initialised with *random* values.**

Visualisation

- Neurons are visualised as changing positions in the *weight space* (which has the same dimension of the input space) as training takes place.

Example 1: results

Section 9.6 Computer Simulations

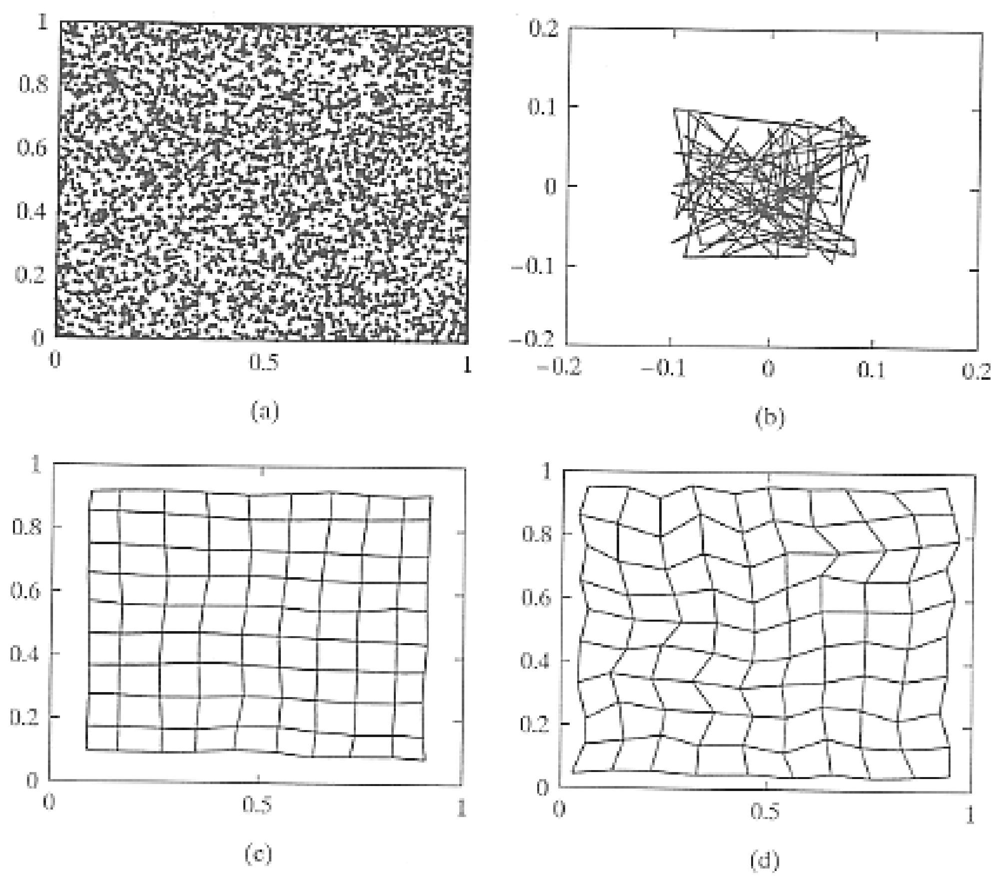


FIGURE 9.8 (a) Input data distribution. (b) Initial condition of the two-dimensional lattice. (c) Condition of the lattice at the end of the ordering phase. (d) Condition of the lattice at the end of the convergence phase.

Example 2

A one dimensional lattice driven by a two dimensional distribution:

- **100 neurons arranged in one dimensional lattice.**
- **Input space is the same as in Example 1.**
- **Weights are initialised with *random* values (again like in example 1).**
- **(Matlab programs for Examples 1, 2 available at ftp://ftp.mathworks.com/pub/books/haykin)**

Example 2: results

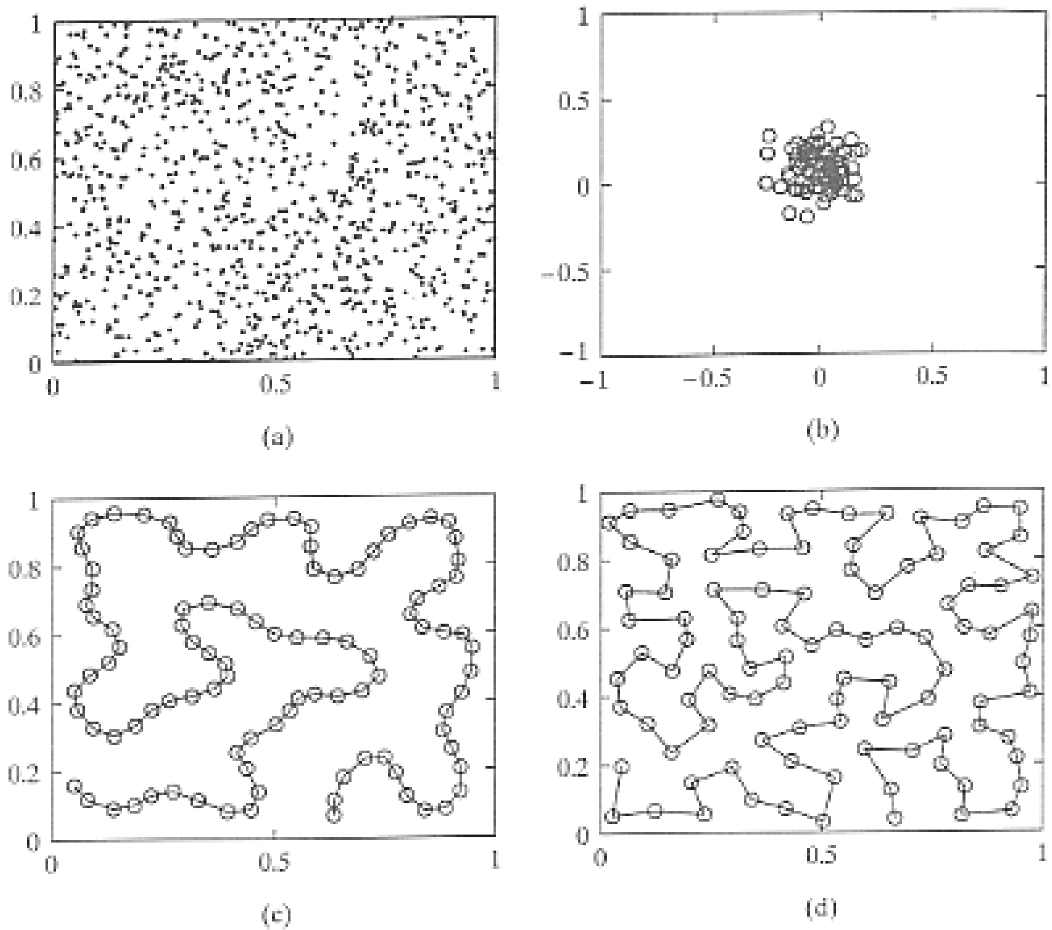


FIGURE 9.9 (a) Two-dimensional input data distribution. (b) Initial condition of the one-dimensional lattice. (c) Condition of the lattice at the end of the ordering phase. (d) Condition of the lattice at the end of the convergence phase.

Example 2: parameter evolution

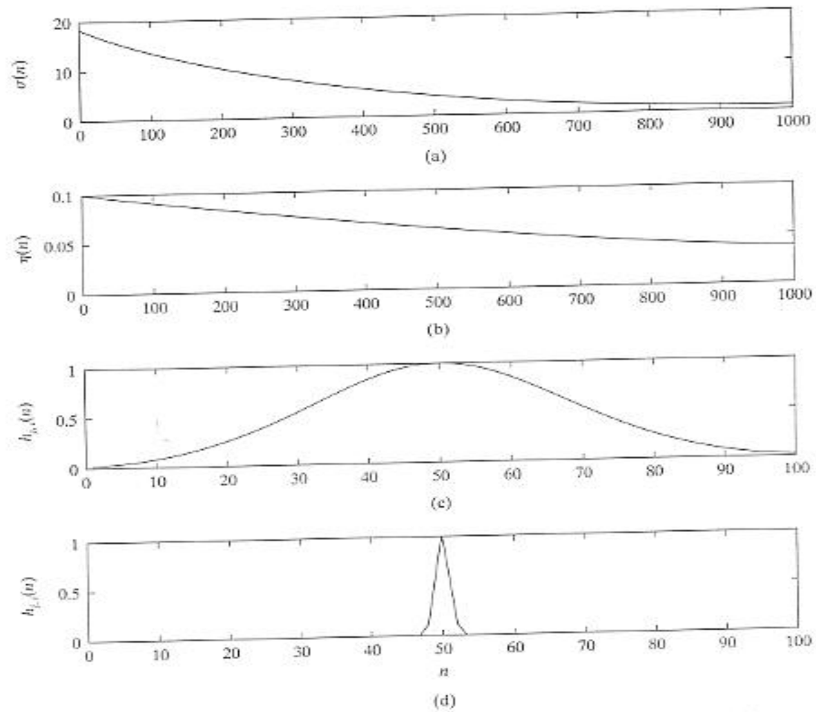


FIGURE 9.10 (a) Exponential decay of neighborhood function parameter $\sigma(n)$. (b) Exponential decay of learning-rate parameter $\eta(n)$. (c) Initial shape of the Gaussian neighborhood function. (d) Shape of the neighborhood function at the end of the ordering phase (i.e., beginning of the convergence phase).

in Fig. 9.10a, starts with an initial value $\sigma_0 = 18$ and then shrinks to about 1 in 1000 iterations during the ordering phase. During that same phase, the learning-rate parameter $\eta(n)$ starts with an initial value $\eta_0 = 0.1$ and then decreases to 0.037. Figure 9.10c shows the initial Gaussian distribution of neurons around a winning neuron located at the midpoint of the one-dimensional lattice. Figure 9.10d shows the shape of the neighborhood function at the end of the ordering phase. During the convergence phase the learning-rate parameter decreases linearly from 0.037 to 0.001 in 5000 iterations. During the same phase the neighborhood function decreases essentially to zero.

The specifications of the ordering phase and convergence phase for the computer simulations in Fig. 9.8 involving the two-dimensional lattice are similar to those used

Components of Self Organization

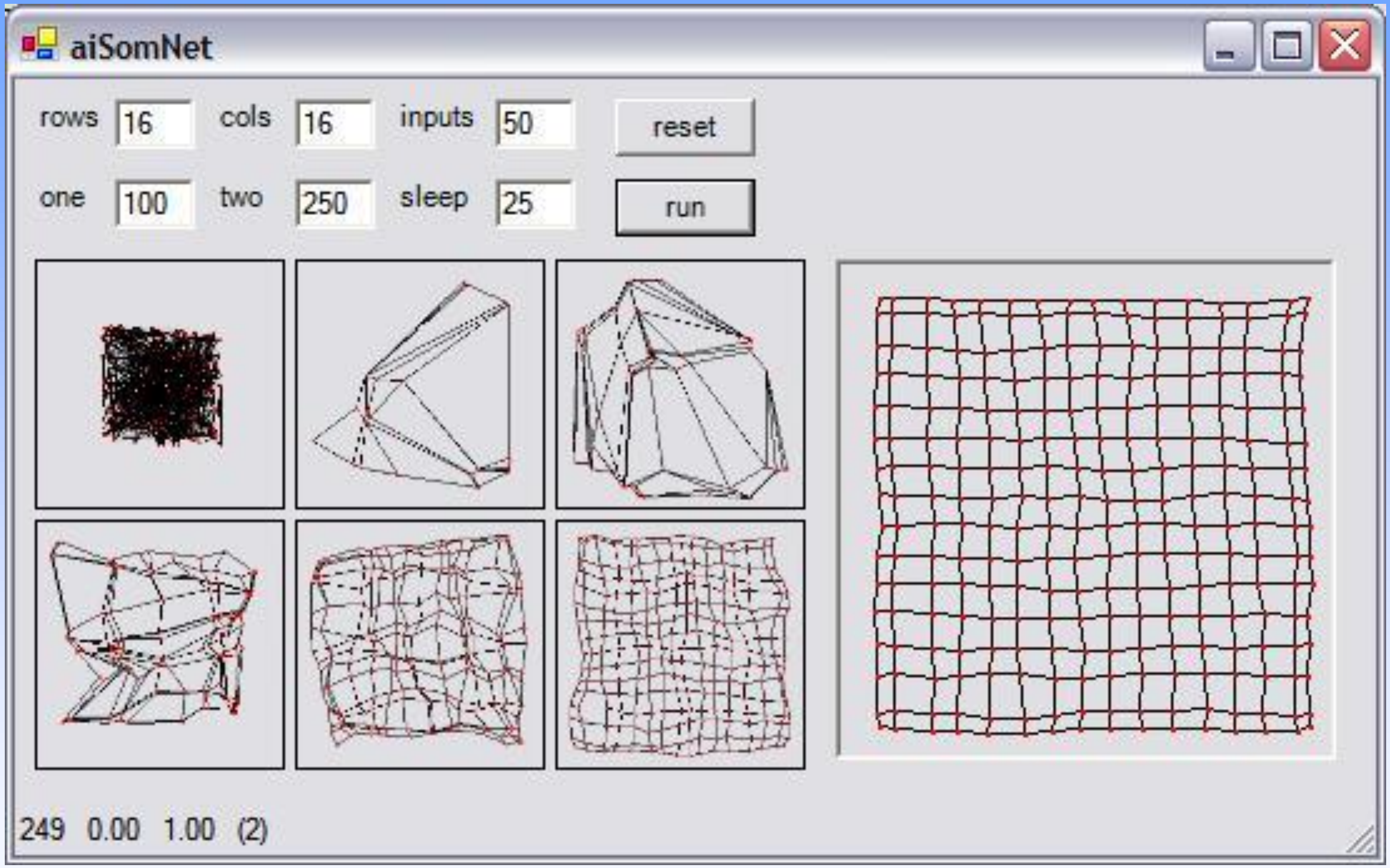
The self-organization process involves four major components:

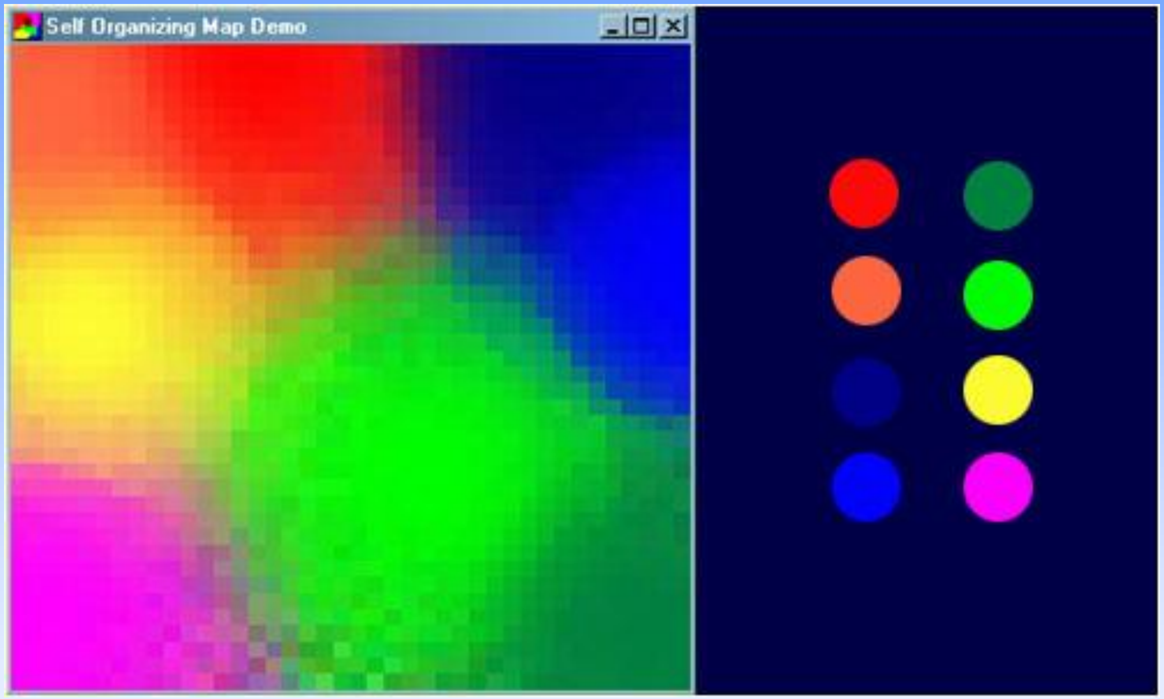
Initialization: All the connection weights are initialized with small random values.

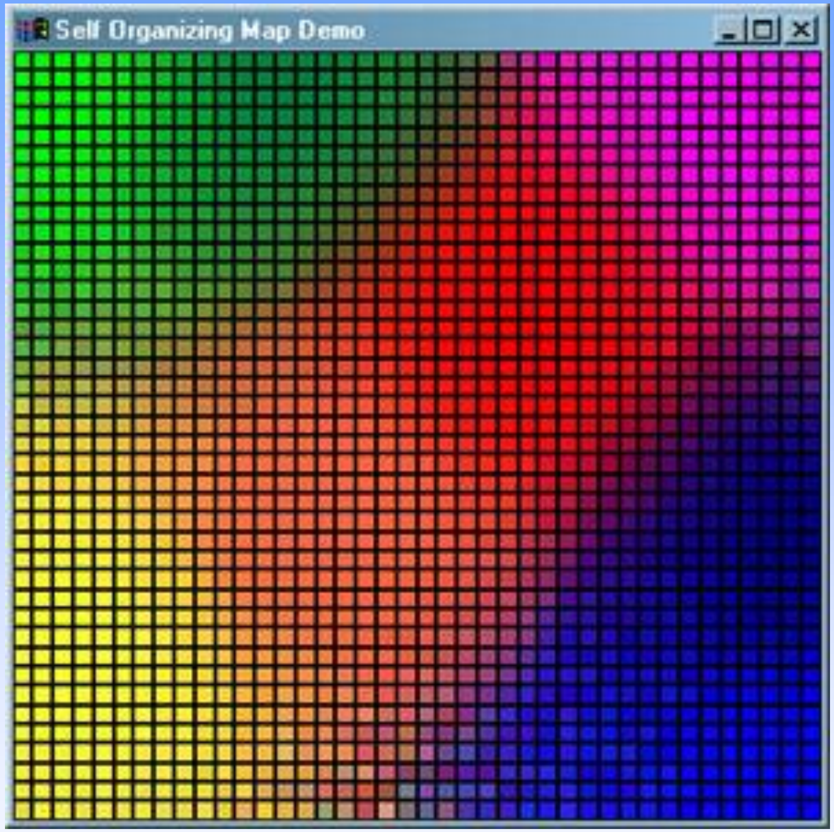
Competition: For each input pattern, the neurons compute their respective values of a *discriminant function* which provides the basis for competition. The particular neuron with the smallest value of the discriminant function is declared the winner.

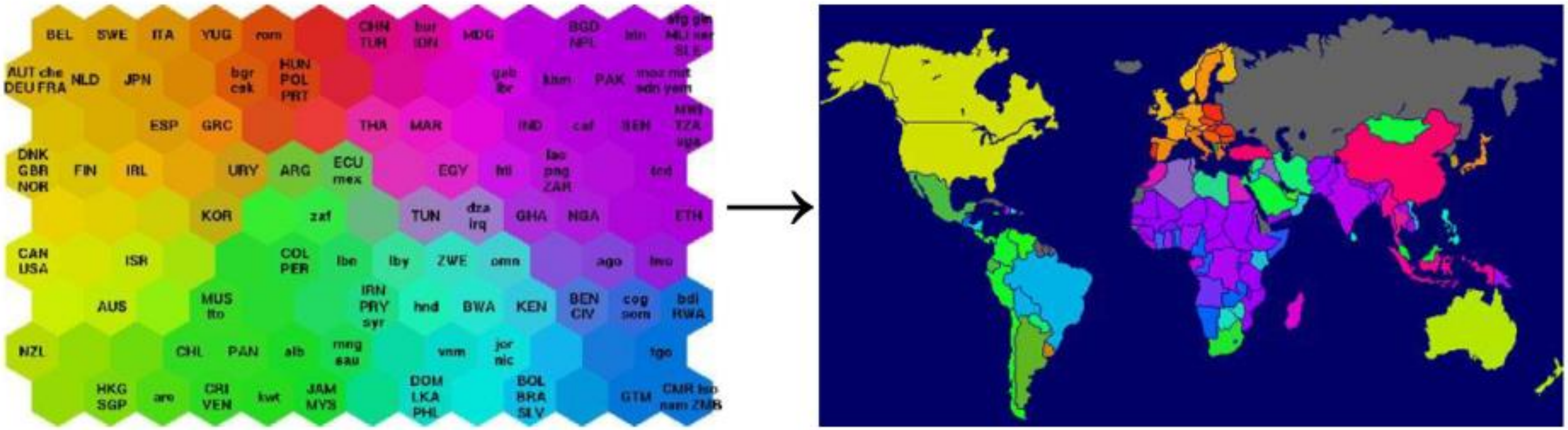
Cooperation: The winning neuron determines the spatial location of a topological neighbourhood of excited neurons, thereby providing the basis for cooperation among neighbouring neurons.

Adaptation: The excited neurons decrease their individual values of the discriminant function in relation to the input pattern through suitable adjustment of the associated connection weights, such that the response of the winning neuron to the subsequent application of a similar input pattern is enhanced.


aiSomNet
rows 16
cols 16
inputs 50
reset
one 100
two 250
sleep 25
run
249 0.00 1.00 (2)


Self Organizing Map Demo


Self Organizing Map Demo


BEL
SWE
ITA
YUG
rom
CHN TUR
bur IDN
MDG
BGD NPL
AUT che DEU FRA
NLD
JPN
bgr csk
HUN POL PRT
PAK
ESP
GRC
THA
MAR
IND
DNK GBR NOR
FIN
IRL
URY
ARG
ECU mex
EGY
lao png ZAR
KOR
zaf
TUN
dza irq
GHA
NGA
ETH
CAN USA
ISR
COL PER
lby
ZWE
omn
ago
AUS
MUS
IRN PRY syr
hnd
BWA
KEN
BEN CIV
cog som
bdi RWA
NZL
CHL
PAN
alb
mng sau
vnm
jor nic
tgo
HKG SGP
are
CRI VEN
kwt
JAM MYS
DOM LKA PHL
BOL BRA SLV
GTM
CMR
nam ZMB

Some Theory

- **Property 1. Approximation of the Input Space**. The feature map Φ, represented by the set of synaptic weight vectors $\{\mathbf{w}_j\}$ in the input space A, provides a good approximation to the input space H.
- The theoretical basis of the idea is rooted in *vector quantization theory.*

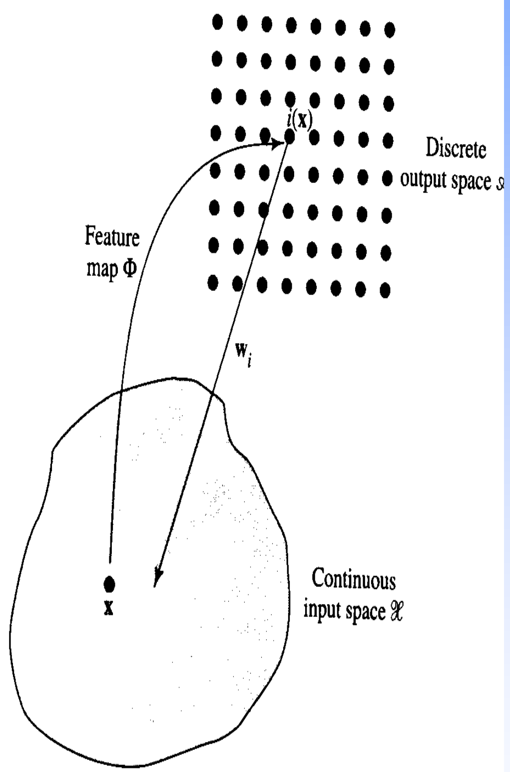


FIGURE 9.4 Illustration of the relationship between feature map Φ and weight vector $\mathbf{w}_i$ of winning neuron i.

Some Theory

- **Property 2. Topological Ordering**. The feature map Φ 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 or feature of input patterns.

Some Theory

- **Property 3. Density Matching**. The feature map Φ reflects variations in the statistics of the input distributions: regions in the input space H from which the sample vectors **x** are drawn with a high probability of occurrence are mapped onto larger domains of the output space A, and therefore with better resolution than regions in H from which sample vectors **x** are drawn with a low probability of occurrence.
- Minimum-distortion encoding, according to which the curvature terms and all higher-order terms in the distortion measure due to noise model $\pi(\boldsymbol{\nu})$ are retained. $m(\mathbf{x}) \propto f_{\mathbf{x}}^{1/3}(\mathbf{x})$
- Nearest-neighbor encoding, which emerges if the curvature terms are ignored, as in the standard form of the SOM algorithm. $m(\mathbf{x}) \propto f_{\mathbf{x}}^{2/3}(\mathbf{x})$

Some Theory

- **Property 4. Feature Selection.** Given data from an input space with an nonlinear distribution, the self-organizing map is able to select a set of best features for approximating the underlying distribution.

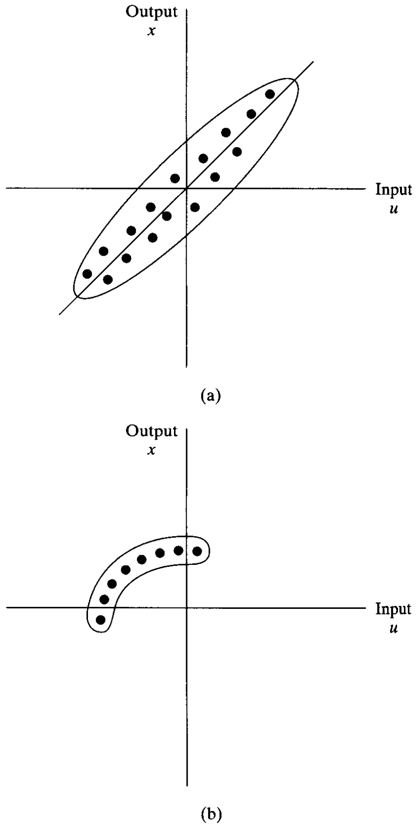
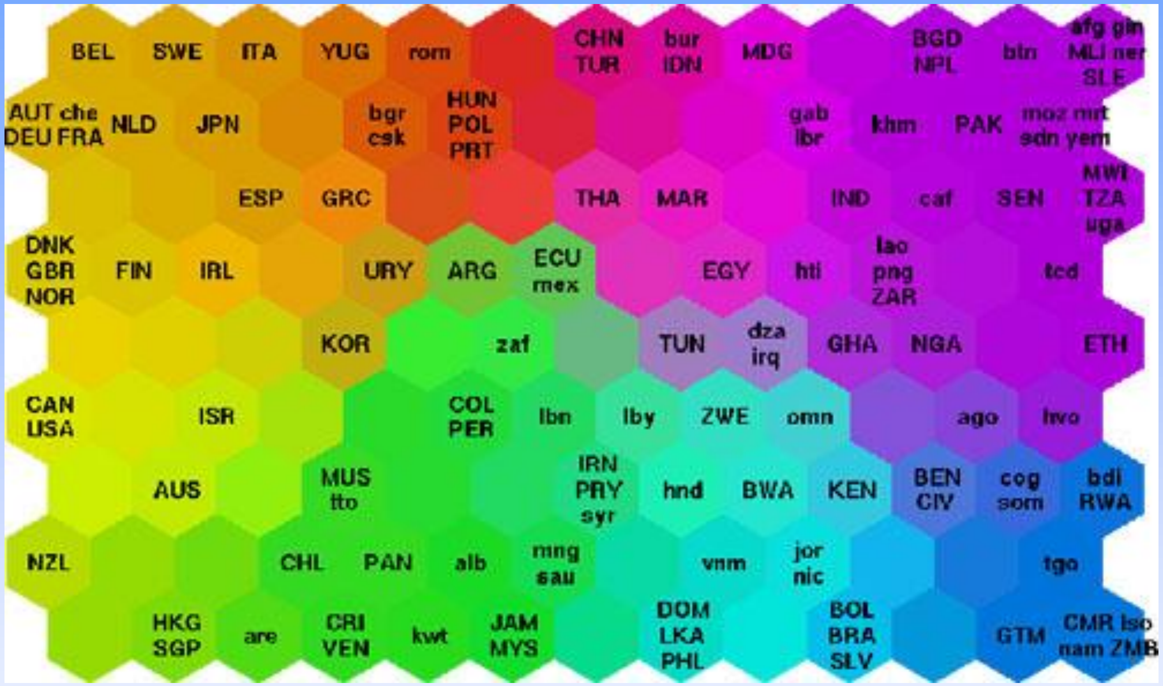


FIGURE 9.7 (a) Two-dimensional distribution produced by a linear input–output mapping. (b) Two-dimensional distribution produced by a nonlinear input–output mapping.



BEL
SWE
ITA
YUG
rom
CHN TUR
bur IDN
MDG
BGD NPL
btn
afg gin MLI ner SLE
AUT che DEU FRA
NLD
JPN
bgr csk
HUN POL PRT
gab lbr
khm
PAK
moz mrt sdn yem
ESP
GRC
THA
MAR
IND
caf
SEN
MWI TZA uga
DNK GBR NOR
FIN
IRL
URY
ARG
ECU mex
EGY
hti
lao png ZAR
tcd
KOR
zaf
TUN
dza irq
GHA
NGA
ETH
CAN USA
ISR
COL PER
lbn
lby
ZWE
omn
ago
hvo
AUS
MUS tto
IRN PRY syr
hnd
BWA
KEN
BEN CIV
cog som
bdi RWA
NZL
CHL
PAN
alb
mng sau
vnm
jor nic
tgo
HKG SGP
are
CRI VEN
kwt
JAM MYS
DOM LKA PHL
BOL BRA SLV
GTM
CMR lso nam ZMB