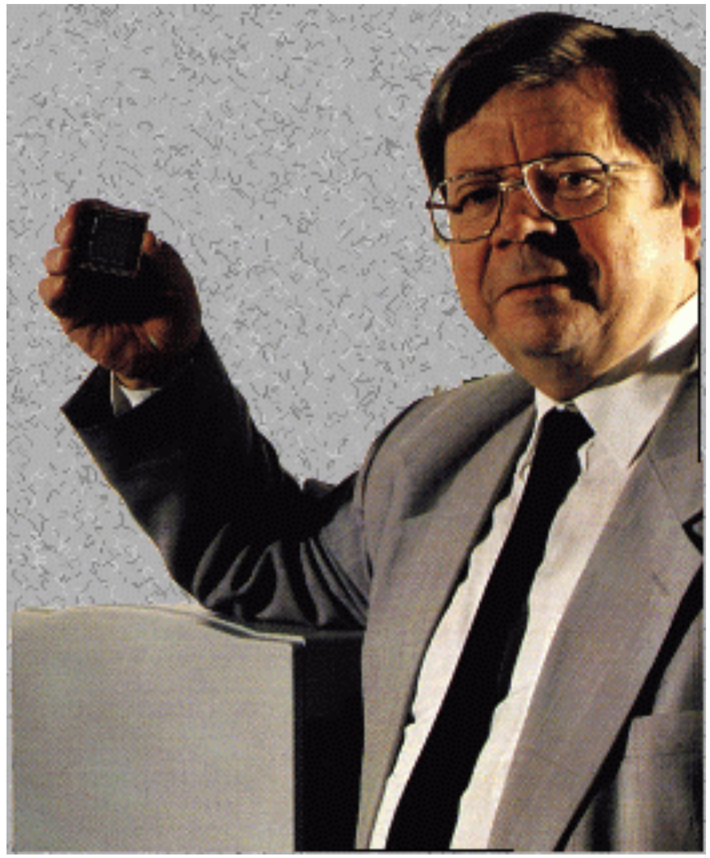
Self-Organizing Maps

"Kohonen Nets"

Feature Maps

(a form of competitive learning)

Teuovo Kohonen



Dr. Eng., Professor of the Academy of Finland;
Head, Neural Networks Research Centre,
Helsinki University of Technology, Finland

Several Different Kohonen Nets

- Linear associative memory (concept discovered at same time as that by James Anderson)
- **Self-Organizing Maps (SOMs)**
  - Similar to LVQ clustering
  - Fundamentally unsupervised
  - Supervision can be overlaid
  - Additional structure can be superimposed

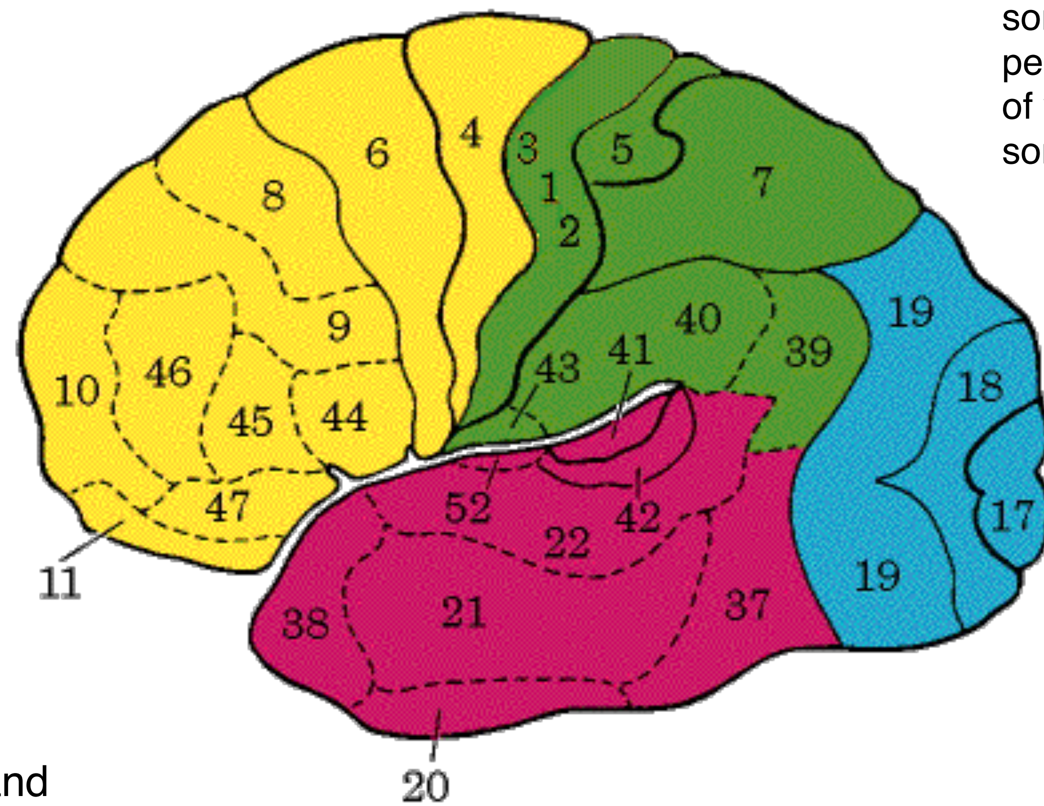
Maps in Neurobiology

- Related neural functions ***map*** onto identifiable regions of the brain
  - **retinotopic map**: vision, *superior colliculus*
  - **phonotopic map**: hearing, auditory cortex
  - **somatotopic map**: touch, somatosensory cortex

Maps in the Human Brain

Frontal Lobe
thinking, planning, and
central executive functions;
motor execution

Parietal Lobe
somatosensory
perception ntegration
of visual &
somatospatial information



Occipital Lobe
visual
perception and
processing

Temporal Lobe
language function and
auditory perception involved
in long term memory and emotion

Desired Properties of Maps

- **Approximation of input space**: generally a many-one mapping of input space into weight space
- **Topology-preserving**: points close together in input space should map to points close together in weight space.
- **Density-preserving**: regions of similar density should map to regions of proportional density.

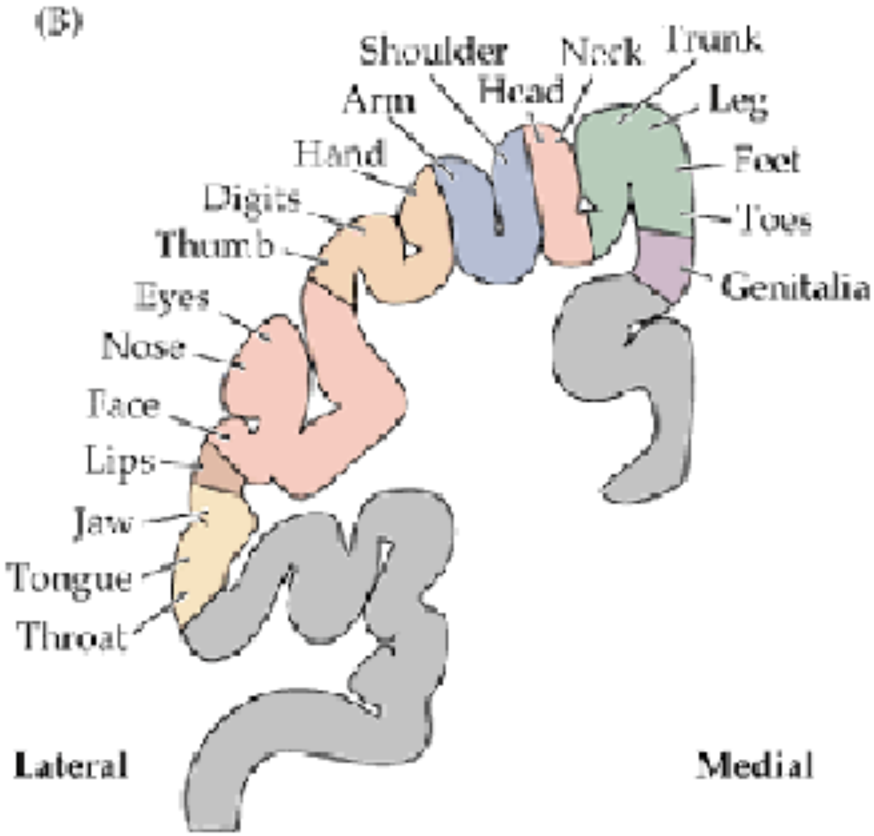
Somatotopic Map Illustration: The "Homunculus"

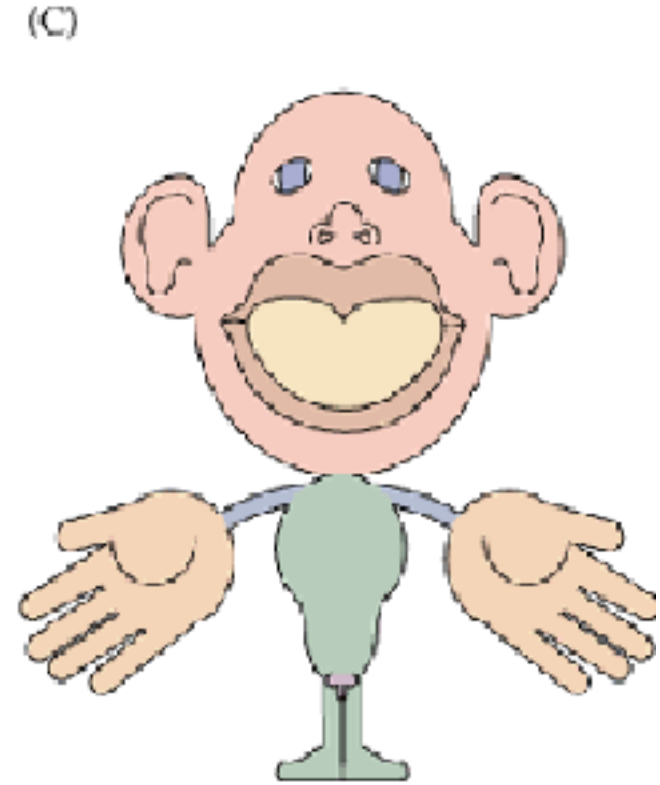
Cartoon map of the relationship between body surfaces and the regions of the brain that control them

(somewhat different from the original "little person inside" meaning).

Another Depiction of the Homunculus

Amount of **neural real-state** devoted to different sensory regions

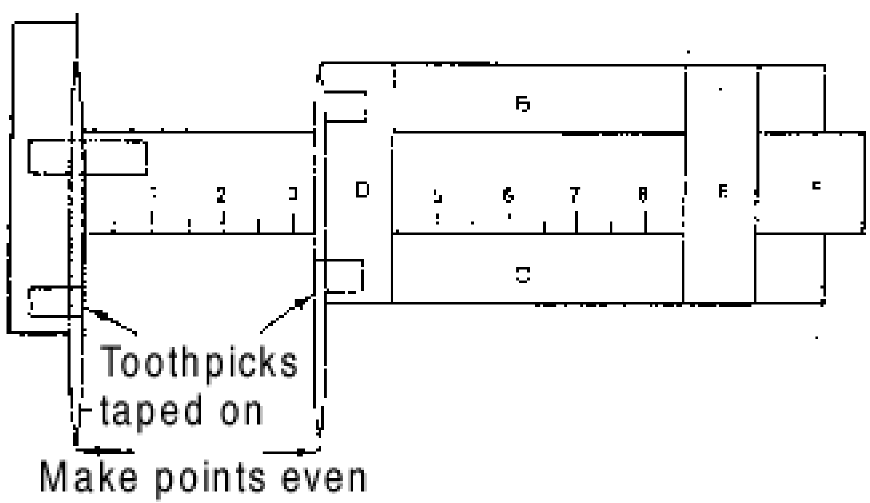




Project: Map your own homunculus:

http://www.woodrow.org/teachers/biology/institutes/1991/homunculus.html

Idea: Neural density ∝ (discriminatory threshold)$^{-1}$



sample (truncated):

Body area (left)	Two point threshold (cm)	Inverse
chin	0.55	1.82
above upper lip	0.3	3.33
scalp	5.4	0.19
forehead	1.9	0.53
nose (bridge)	0.9	1.11
eyelids	0.65	1.54
cheek	1.60	0.63
lower lip	0.3	3.3
upper lip	0.25	4.00
ear	2.40	0.42
back neck	1.35	0.74
side neck	1.85	0.54
front neck	2.55	0.39
top shoulder	4.4	0.23

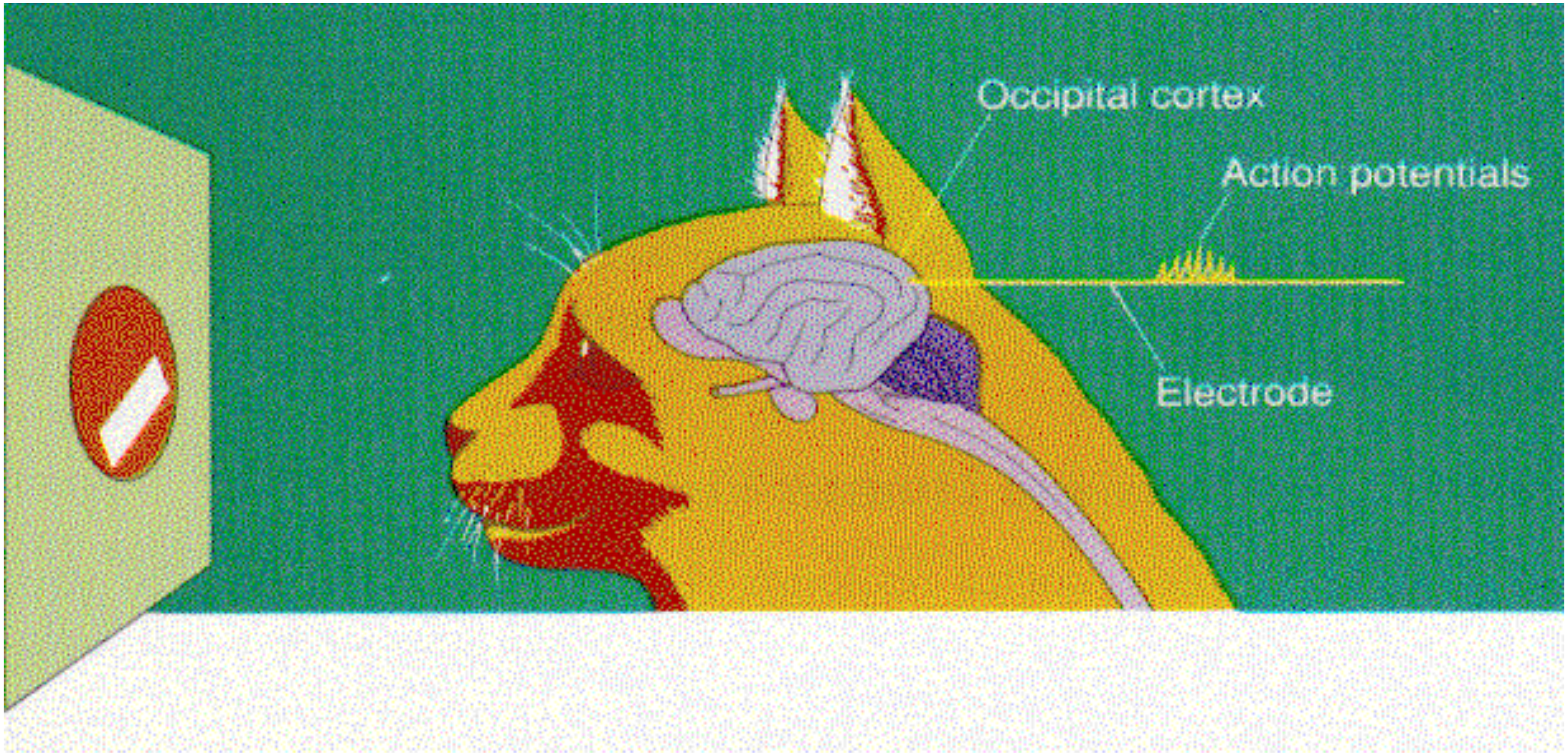
Derivation of SOM

- Consider a spatial “field” of neurons.
- Each neuron has an **on-center, off-surround** type of behavior
  - It’s response is high to an input that occurs close by, low otherwise.
- When a stimulus is presented, neurons “compete” by **mutual lateral inhibition**.
- The winner **and neurons in the neighborhood** have their weights strengthened in the direction (in weight space) of the stimulus.
- Neurons tend to learn to respond to sets of similar stimulii.

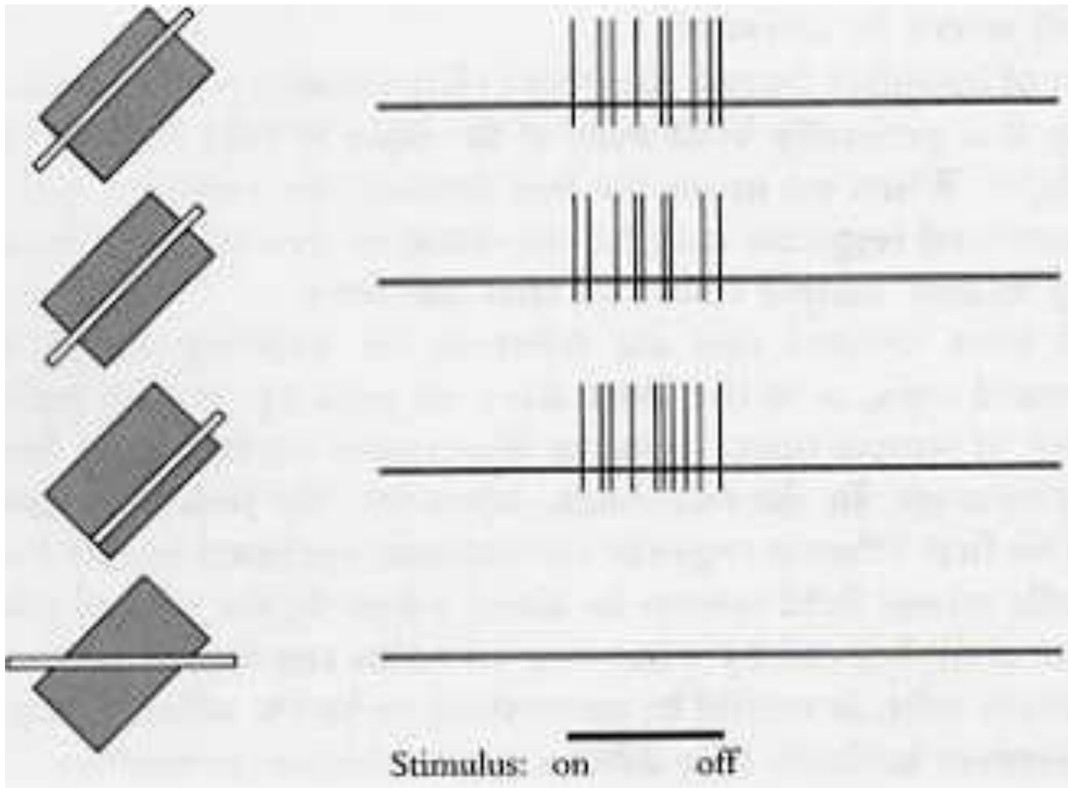
Hubel and Wiesel Findings (1958)

- **Simple** cells in the visual cortex respond to specific **angular orientations** of light bars.
- **Complex** cells respond to specific orientation and **direction of motion**.
- End-stopped cells respond to specific orientation, motion, and **length**.

Hubel and Wiesel Experiments (1958)

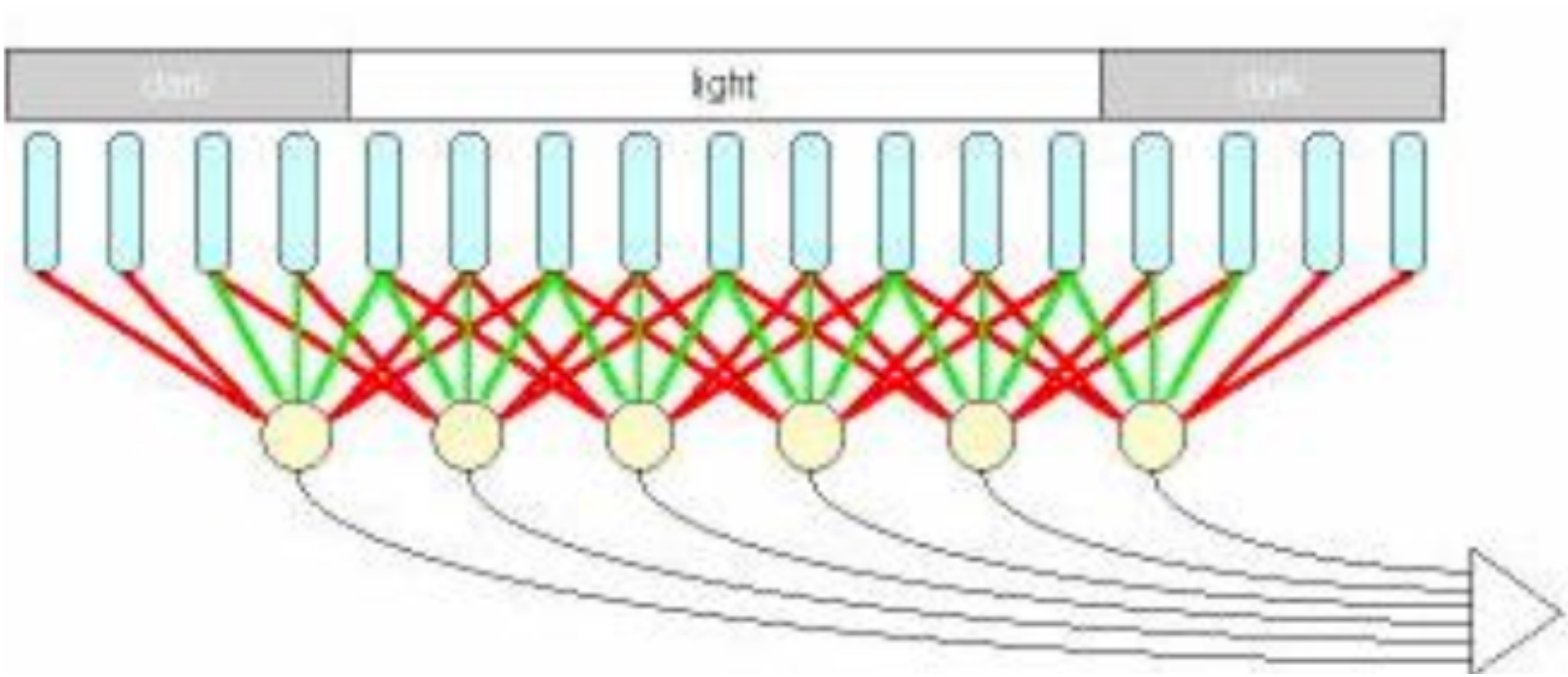


Complex Cell Response
(orientation and motion)



Hubel's book on-line:
http://neuro.med.harvard.edu/site/dh/bcontex.htm

Receptor arrays **with lateral inhibition** ≈ "competition"



The neurons have **mutually inhibitory interconnections**.

The strongest responder will suppress the others.

Receptor Network Modeling in Kohonen's Original Paper

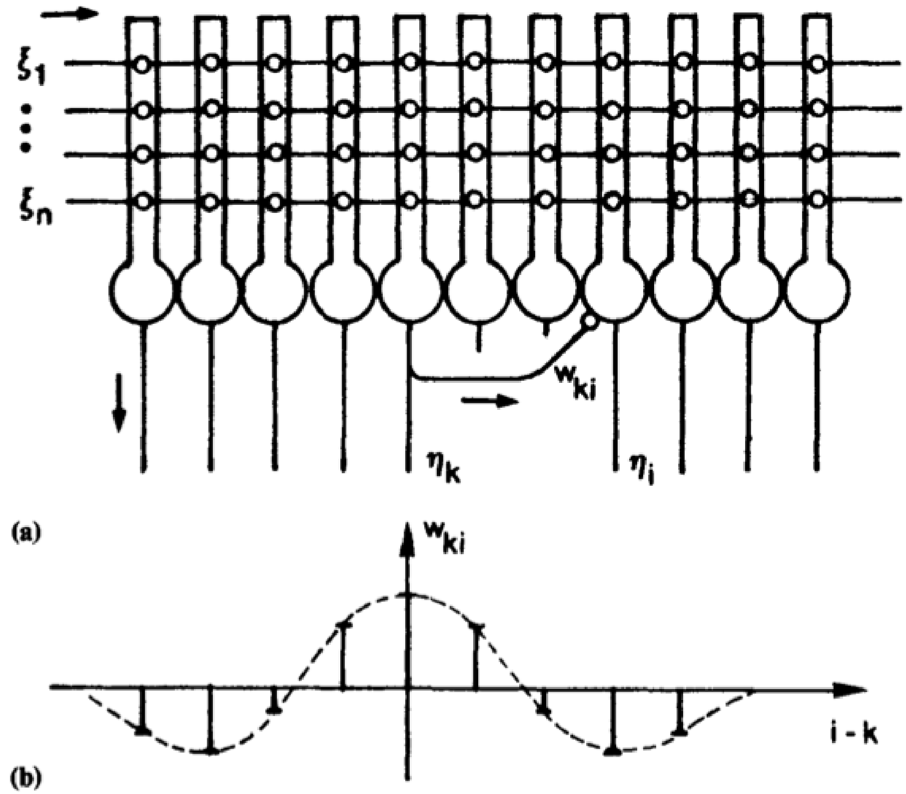
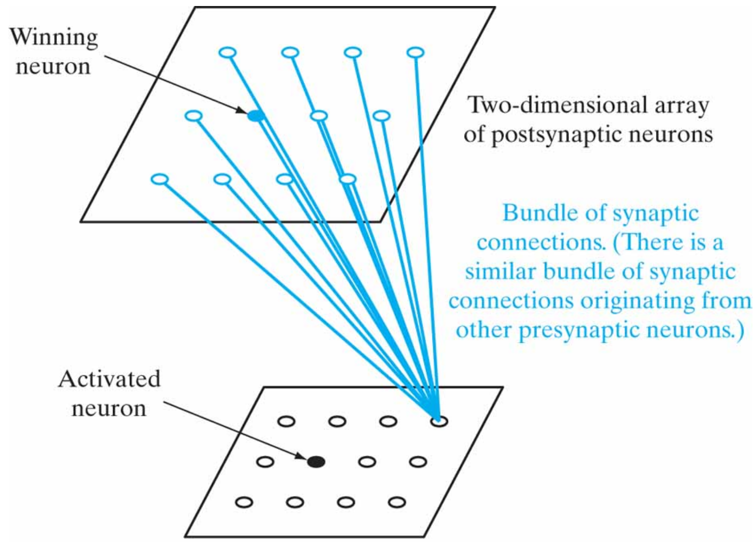


Figure 3. (a) Neural network underlying the formation of the phonotopic maps used in speech recognition. (b) The strengths of lateral interaction as a function of distance (the "Mexican hat" function).

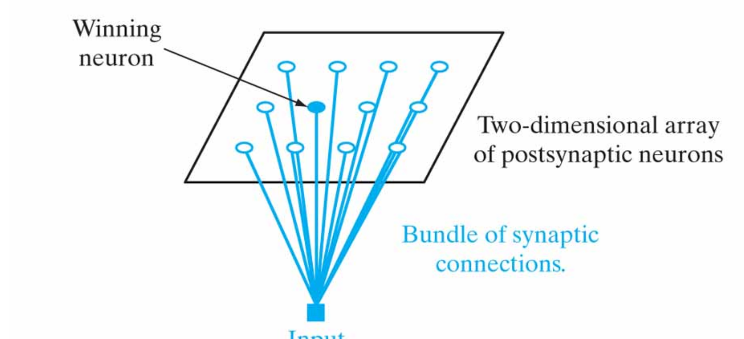
Two Map Models
(Haykin, ch. 9)

signal flow



(a) Willshaw–von der Malsburg's model

signal flow



(b) Kohonen model

Self-Organizing Map Modeling

Competitve: update weight vectors in a ***neighborhood*** of the ***winning*** neuron.

Kohonen Rule: $${}_i\mathbf{w}(q) = {}_i\mathbf{w}(q-1) + \alpha(\mathbf{p}(q) - {}_i\mathbf{w}(q-1))$$

$$i \in N_{i*}(d)$$

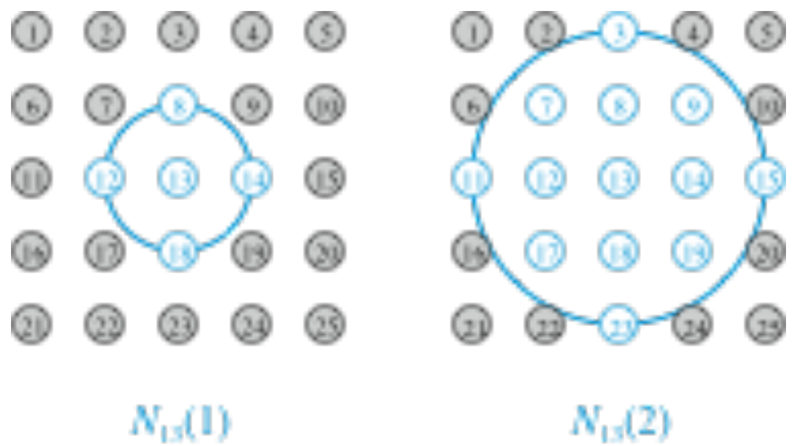
$${}_i\mathbf{w}(q) = (1-\alpha)\,{}_i\mathbf{w}(q-1) + \alpha\mathbf{p}(q)$$

$$N_i(d) = \{j, \mathrm{d}_{i,j} \le d\}$$

$$N_{13}(1) = \{8, 12, 13, 14, 18\}$$

$$N_{13}(2) = \{3, 7, 8, 9, 11, 12, 13, 14, 15, 17, 18, 19, 23\}$$

Two possible neighborhoods of neuron 13, the first narrower



Annealing Behavior

- As training progresses:
  - the learning rate is gradually reduced
  - the size of the neighborhood is gradually reduced

- Neighborhood can be based on
  - Euclidean distance, or
  - Superimposed structure

Kohonen Learning Algorithm

- Repeatedly present input vectors until resources exceeded:
  - At each vector presentation do the following:
    - Find the node *k* whose weight vector is closest to the current input vector.
    - Train node *k* and all nodes in some neighborhood of *k*.
    - Decrease the learning rate slightly.
    - After every *M* cycles, decrease the size of the neighborhood.

“Superimposed Dimensionality” of SOM

- In a general SOM, an n-dimensional grid, can be ***overlaid*** with the neurons as nodes.

- The edges connecting the neurons ***constrain the neighborhood*** *for* ***updating*** (rather than using Euclidean distance).

- Only nodes within a ***graphical diameter on the grid*** are in the neighborhood.

Uses of "Imposed Dimensionality" in Data Analysis

- Data points may have an *unknown, but often large,* **underlying dimensionality**, e.g. the underlying phenomenon or process that has several dimensions of attributes (such as color dimensions, various size or rate dimensions, etc.)

- By training a network with an ***imposed*** grid with a given dimensionality, the relationships among the data may become visualizable, especially if the imposed number of dimensions is small.

Uses of "Imposed Dimensionality" in Data Analysis

- In other words, training the map "organizes" the data according to similarity in each dimension.

Demo Applets

- /cs/cs152/kohonen/ demo1, demo2, demo3
- Legend:
  - black = input data point (random over a 2-D region; 2-dimensional data)
  - red = neuron in winner neighborhood; learns by Kohonen rule
  - blue = other neuron
- Learning rate and neighborhood both decrease with time; demo speeds up over time.

Competition Code

```
// compete finds the neuron with weights closest to the input.
// It sets winpoint to the indices of that neuron.

  public void compete(Input input)
    {
    // initialize min with an distance to an arbitrary neuron
    winpoint = 0;
    double min = neuron[winpoint].distance(input);

    // find the min distance among all neurons

    for( int point = 0; point < points; point++ )
      {
      double dist = neuron[point].distance(input);
      if( min > dist )
        {
        min = dist;             // update the min distance
        winpoint = point;
        }
      }
    }
```

Competition Code

```
// learn updates all neurons within current neighborhood
// by applying the Kohonen learning rule against the current
// input.

  public void learn(Input input, double learningRate)
    {
    for( int point = 0; point < points; point++ )
      {
      if( inWinnersNeighborhood(point) )
        {
        neuron[point].learn(input, learningRate);
        }
      }
    }
```

Optimized Computation for Fixed Datasets: Batch Mapping

Although the above iterative algorithm has been used with success in numerous applications, it has turned out that the scheme termed the *Batch Map* produces essentially similar results but an order of magnitude faster. The basic idea is that for every node j in the grid, the average $\overline{x}_j$ of all those input items $x(t)$ is formed that have m_j as the closest model. After that, new models are computed as

$$m_i = \sum_j n_j h_{ji} \overline{x}_j / \sum_j n_j h_{ji}$$

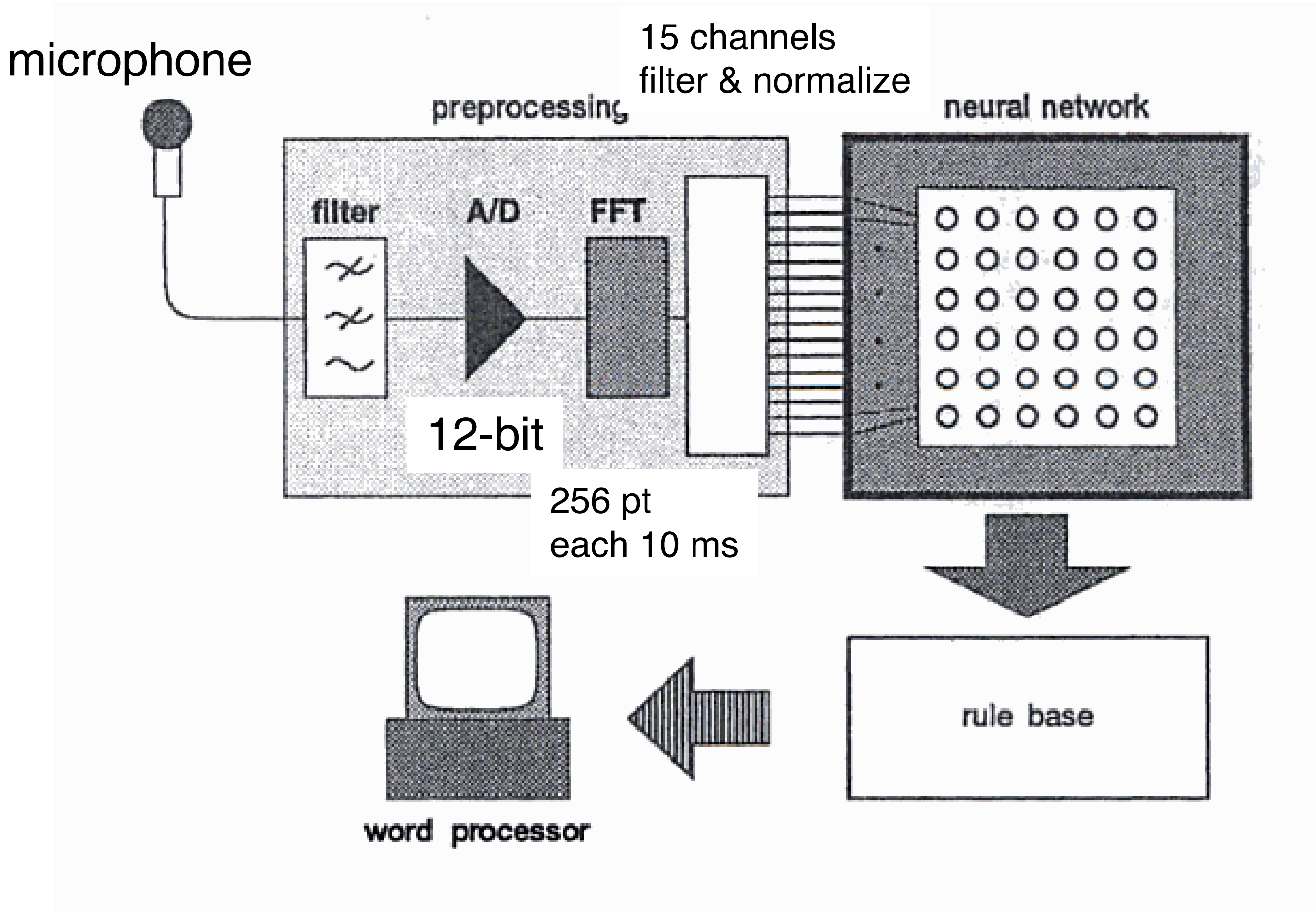
where n_j is the number of the input items mapped into the node j, and the index j runs over the nodes in the neighborhood of node i. For the updated m_i, this scheme is iterated for a few times, always using the same batch of input data items to determine the updated $\overline{x}_j$.

http://www.scholarpedia.org/article/Kohonen_network

Phonetic Typewriter: Kohonen's Original Application

- Objective: Spoken words →phonemes
- Languages: Finnish, Japanese
- Claimed ≥ 92% accuracy in 10 mins. of training.

Phonetic typewriter setup



More Details

5.7.1 Front-end Preprocessing

The front-end processing is an essential element to any neural network technique. This point cannot be over-stressed. Any neural network paradigm will perform poorly if given non-representative or inadequate training data. Neural networks do provide a novel method of abstracting feature information into a distributed encoding. They do not, however, by-pass the critical stage in any pattern recognition task of adequately defining the salient and characteristic features of the data. Kohonen's system relies on standard digital signal processing techniques to extract the phoneme spectral data from the voice input. From a microphone input the speech waveform is fed into a 5.3 kHz low pass filter driving a 12-bit A/D converter (at a sampling rate of 13.03 kHz). A 256 point Fast Fourier Transform (FFT) is computed on the digital data from the A/D at 9.83 ms intervals to capture the spectral content of the phonemes. Kohonen uses the FFT technique because it shows the clustering properties of the spectral component better than more conventional coding methods, and thus provides a more useful representation on which the classifier can train. It is also a fast, reliable and well supported technique. The output of the FFT is filtered and made logarithmic before the information is grouped into a fifteen component continuous pattern vector. The information represented in this vector is the instantaneous power in one of fifteen frequency bands ranging from 200 Hz to 5 kHz. Before being applied to the network as input the components have the signal average removed and are then normalised to a constant length. Kohonen also uses a sixteenth vector component to represent other information about the signal. He chose to use this to represent the rms value of the speech signal.

5.7.3 Post Processing

The last stage of the phonetic typewriter is the translation from the phonetic transcription to orthographic. It is here that the errors from the classification stage must be corrected. The majority of errors are caused by an effect known as coarticulation. Coarticulation is the variation in the pronunciation of a phoneme that is caused by the context of the neighbouring phonemes. To deal with this effect, Kohonen has adopted a rule based system that constructs the correct grammar from the phonetic translation. The rule base is large—typically 15000–20000 rules and deals primarily with context sensitivity of phonemes. It would be impractical to attempt to define rules to account for coarticulation without considering context. The rule base would be prohibitively large if it were to deal with all permutations and it could not cope with the contradictory cases so often found in a language. Kohonen's rule base has been developed from actual example speech data and its correct phonetic transcription. Much like the neural network stage the rules have been derived from example rather than explicitly.

The grammar rule base has been implemented efficiently using hash coding (a software technique for content addressable memory) and operates in near real time—even for a large rule base. The output of the rule base is contextually corrected phonetic strings that can produce orthographic text to drive a word processor environment.

Resulting Phonotopic Map

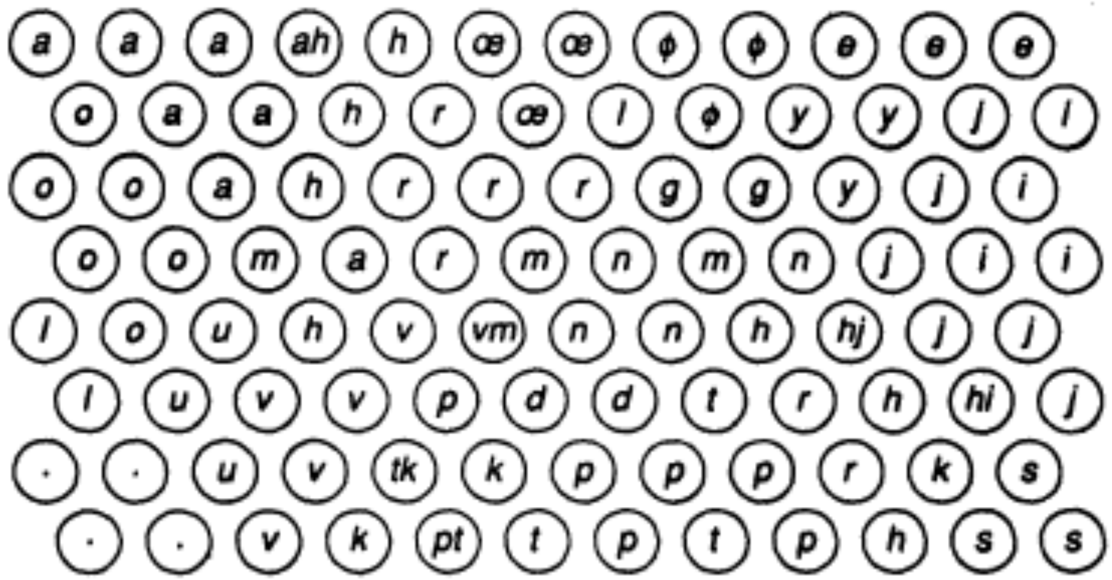
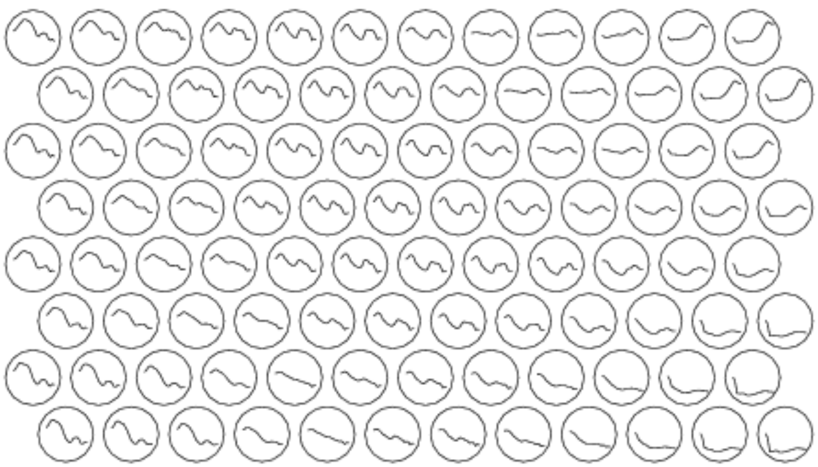


Figure 7.9 This figure is a phonotopic map with the neurons, shown as circles, and the phonemes to which they learned to respond. A double label means that the node responds to more than one phoneme. Some phonemes—such as the plosives represented by *k*, *p*, and *t*—are difficult for the network to distinguish and are most prone to misclassification by the network. *Source: Reprinted with permission from Teuvo Kohonen, "The neural phonetic typewriter."* IEEE Computer, *March 1988. ©1988 IEEE.*

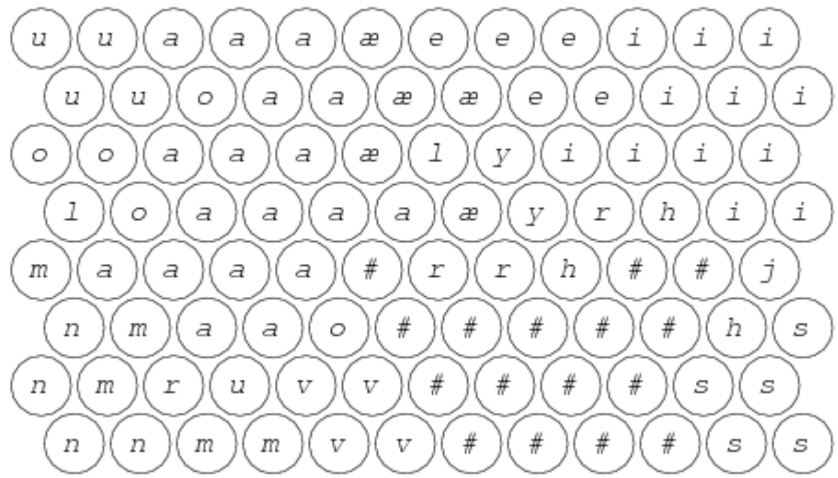
Spectra Corresponding to Quasi-Phonemes

"quasi" is because true phonemes
do not have uniform length, as these do.

The symbol # stands for the
stop consonants /k,p,t/.



Spectrum



Phoneme

http://www.scholarpedia.org/article/Kohonen_network

Resulting Phonotopic Map

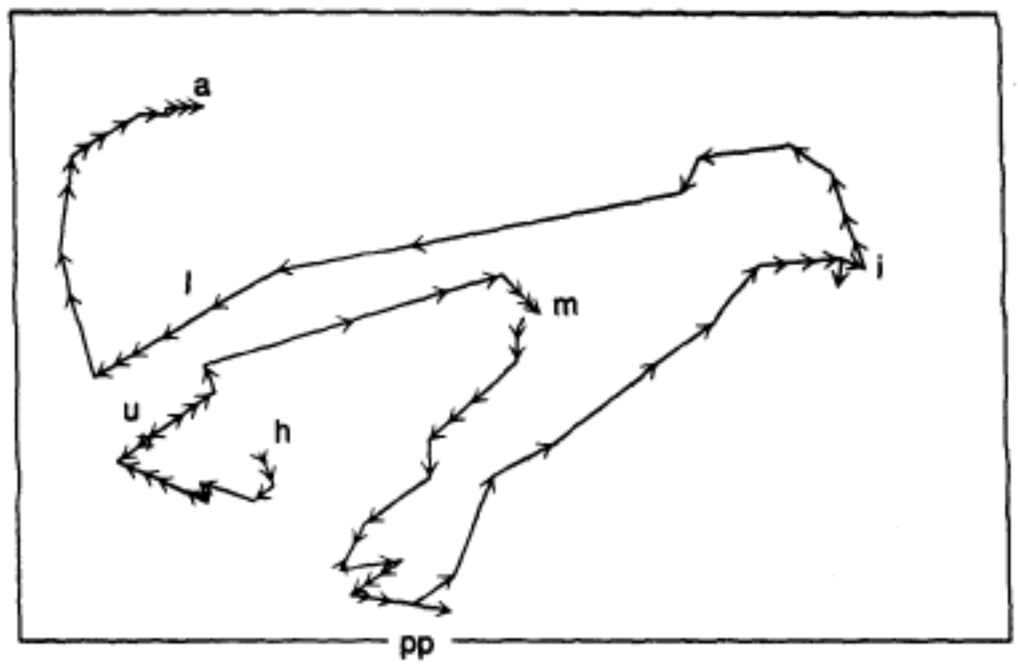


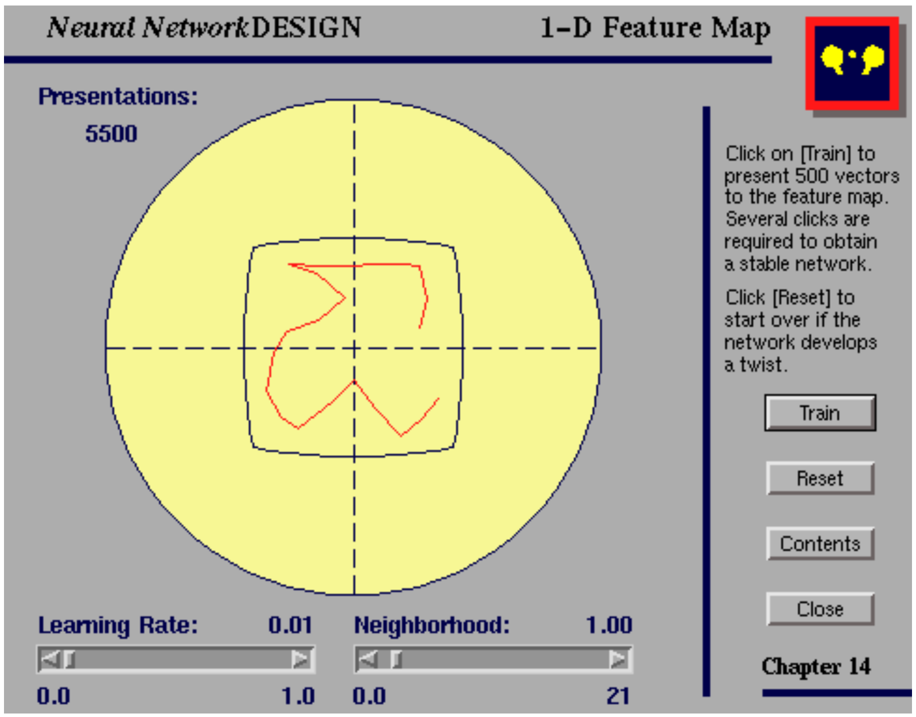
Figure 7.10 This illustration shows the sequence of responses from tl phonotopic map resulting from the spoken Finnish wo *humppila*. (Do not bother to look up the meaning of this wo in your Finnish–English dictionary: *humppila* is the name of place.) *Source: Reprinted with permission from Teuvo Kohone "The neural phonetic typewriter."* IEEE Computer, *March 198* ©*1988 IEEE.*

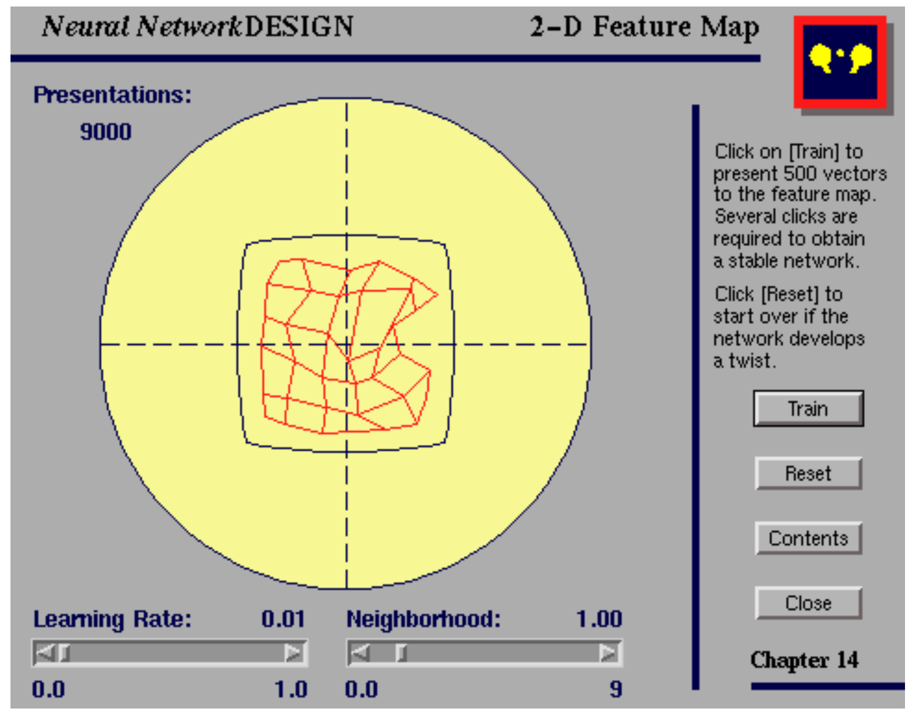
Similar matlab demos

Same input spaces, different imposed dimensions

1D

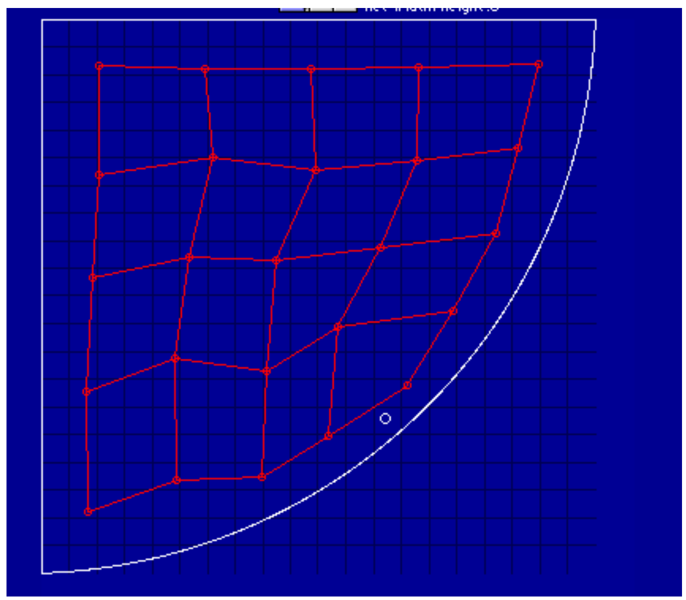
2D





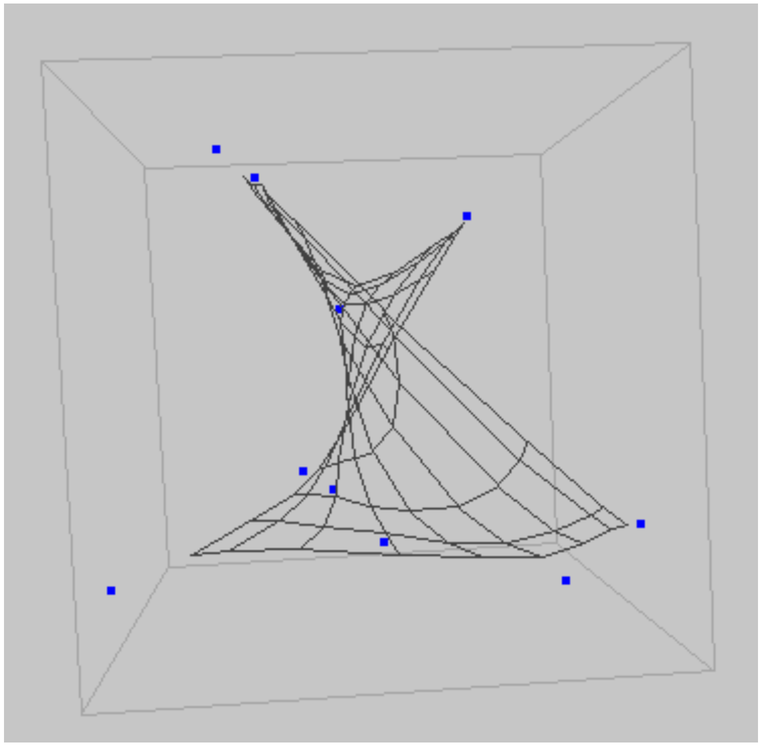
The grid adjusts to represent the data distribution

- Non-rectangular data distribution



http://www.patol.com/java/fill/index.html

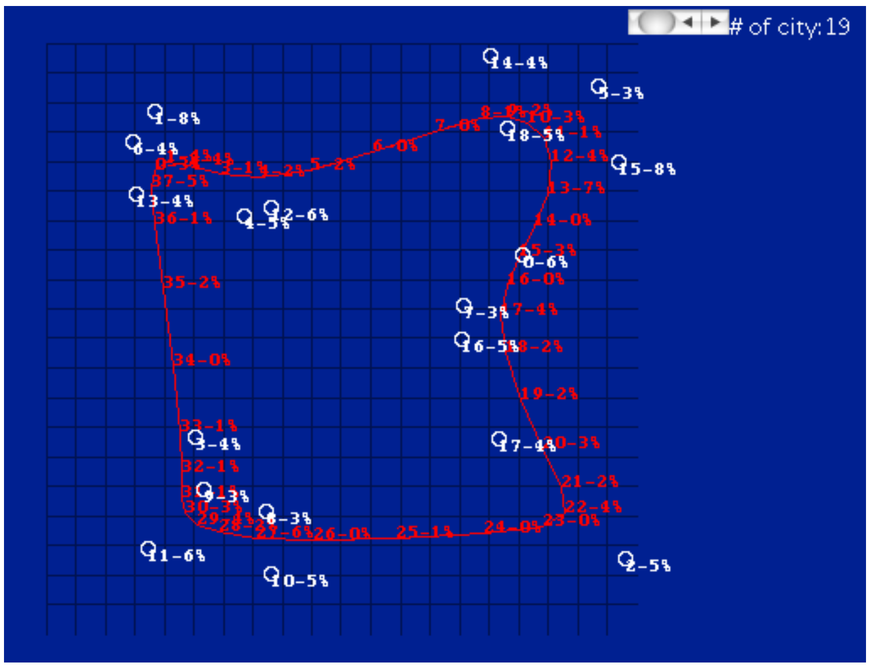
3D data to 2D Grid



http://rfhs8012.fh-regensburg.de/~saj39122/jfroehl/diplom/e-index.html

Heuristic Solutions to the TSP

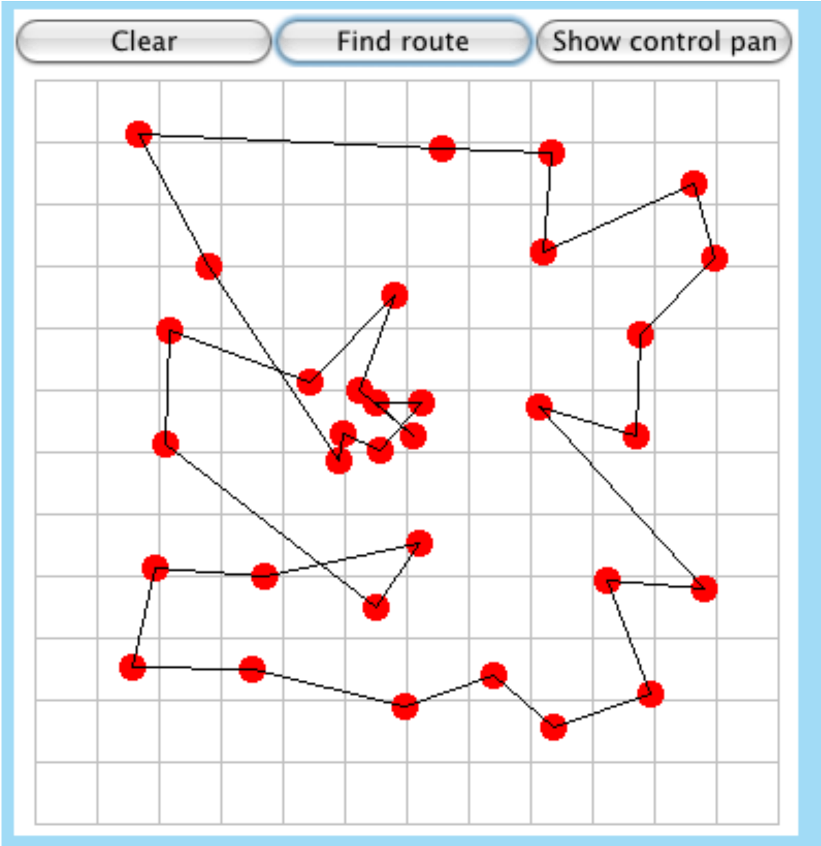
- Finding a heuristic (not necessarily optimal) solution to the Euclidean Traveling Salesperson Problem using Kohonen net
- “Elastic net” approach



red points = neurons,
red lines = overlay
circles = cities
neurons travel toward cities

http://www.patol.com/java/TSP/index.html

This approach to the TSP is neither optimal nor particularly fast, just "interesting"



http://eple.hib.no/~hib00ssl/TSPKohonen.html

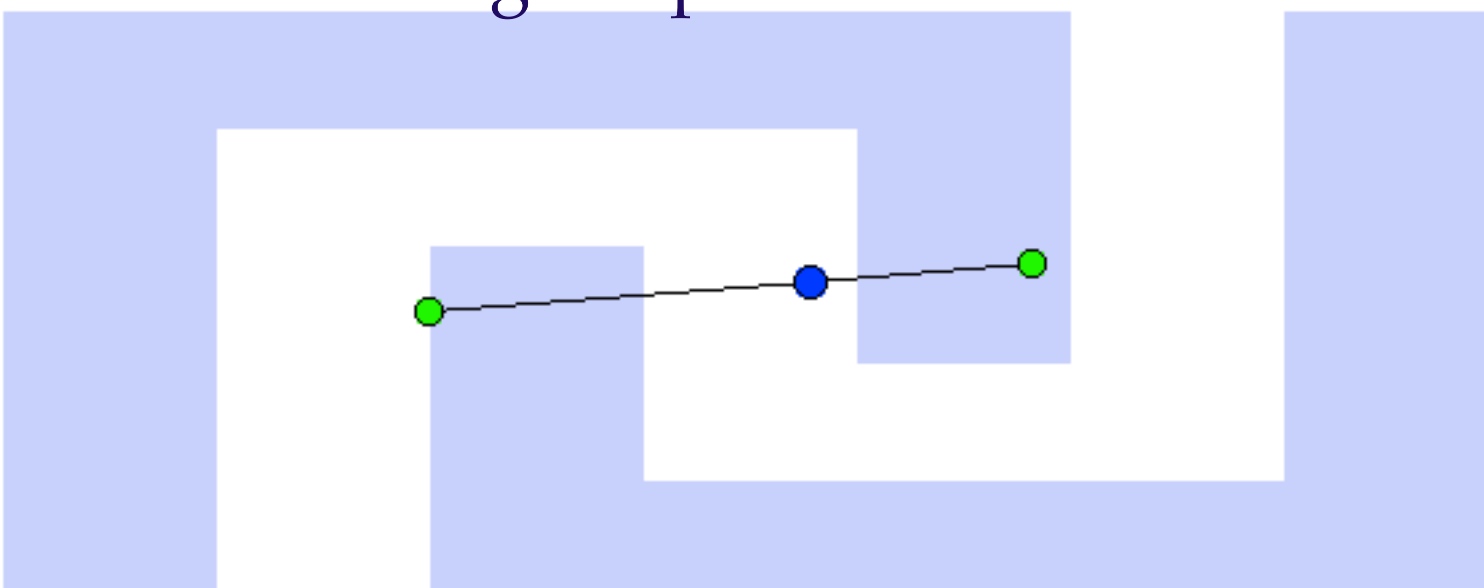
Competitive Learning Demo

http://www.sund.de/netze/applets/gng/full/GNG_0.html

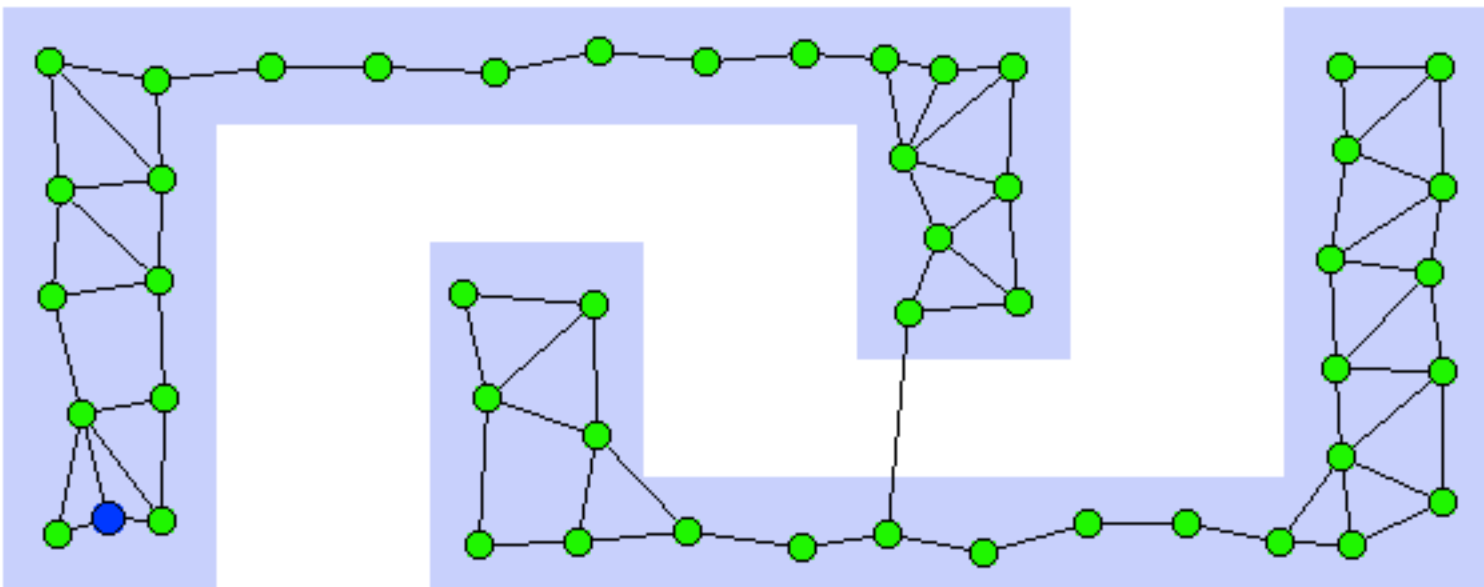
- The following competitive algorithms are available in one applet:
  - LBG (Linde, Buzo, Gray) [extension of “k-means clustering”]
  - LBG-U (Fritzke)
  - Hard Competitive Learning (standard algorithm)
  - Neural Gas (Martinetz and Schulten)
  - Competitive Hebbian Learning (Martinetz and Schulten)
  - Neural Gas with Competitive Hebbian Learning (Martinetz and Schulten)
  - Growing Neural Gas (Fritzke)
  - Growing Neural Gas with Utility (GNG-U, Fritzke)
  - Self-Organizing Map (Kohonen)
  - Growing Grid (Fritzke)
- Thirteen choices of input distribution

Growing Neural Gas

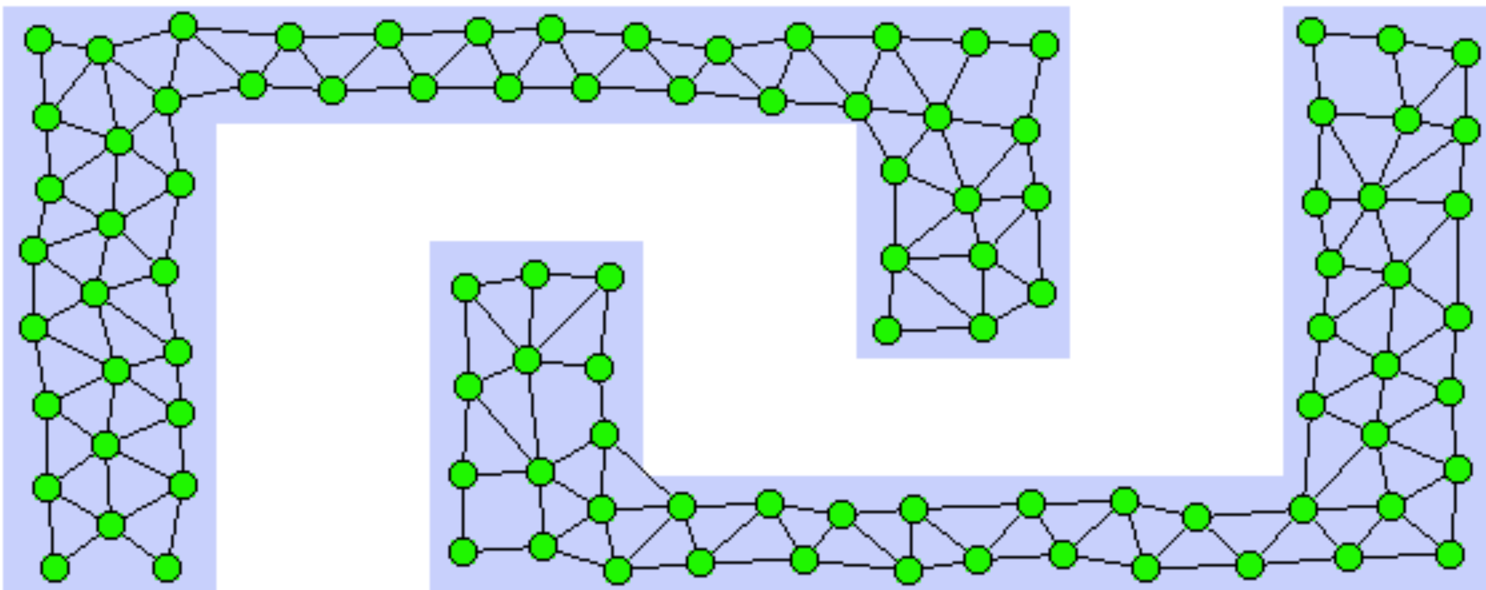
Shaded areas show data distribution.
Circles are neurons, which can be added and deleted during adaptation.



Growing Neural Gas



Growing Neural Gas



Unsupervised Applications of SOMs

- Discovering similarities in data (clustering)
- Data-mining
- KDD (Knowledge Discovery in Databases)

Color Quantization Problem (discussed earlier)

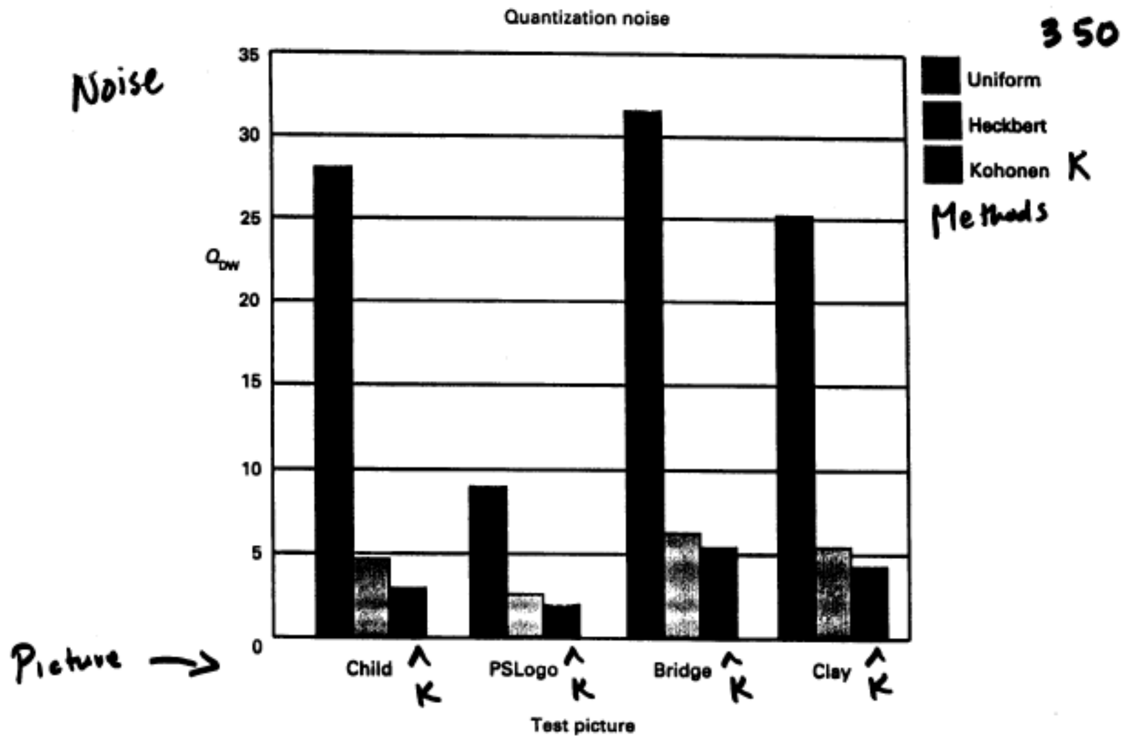
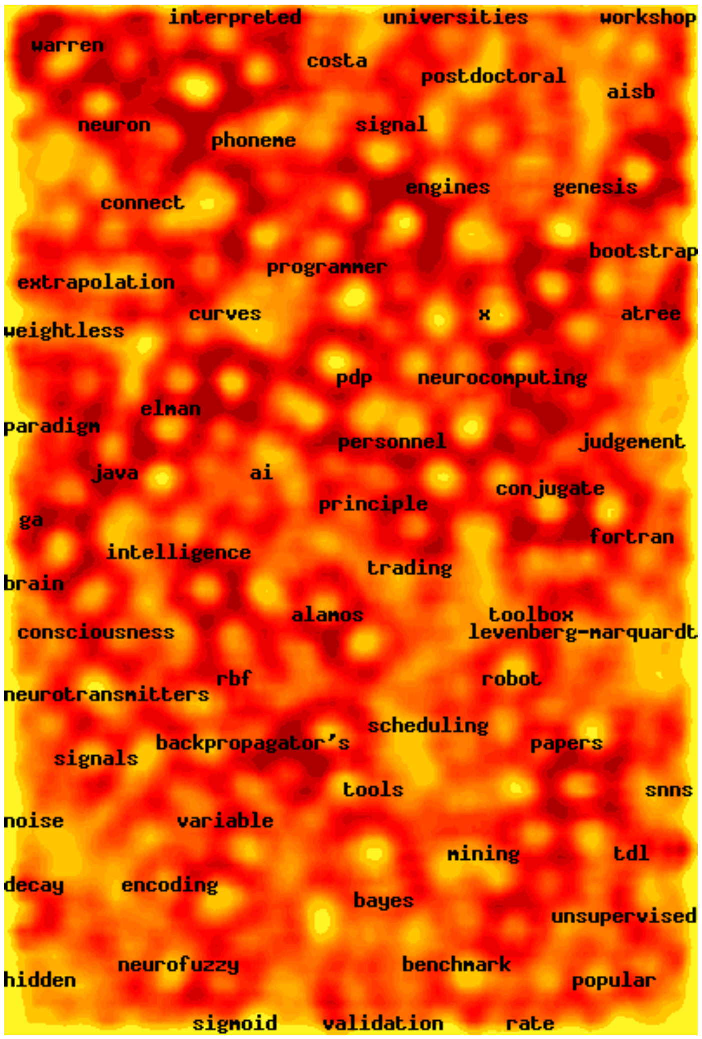


Figure 4.43 Quantization noise for different pictures and different quantization methods: uniform palette, Heckbert's algorithm and the self-organizing algorithm of Kohonen

Heckbert's median cut algorithm "Color Quantization by Dynamic Programming and Principal Analysis" by Xiaolin Wu in ACM Transactions on Graphics, Vol. 11, No. 4, October 1992, 348-372.

WEBSOM: Classifying news messages by terms occurring within ("Semantotopic map")

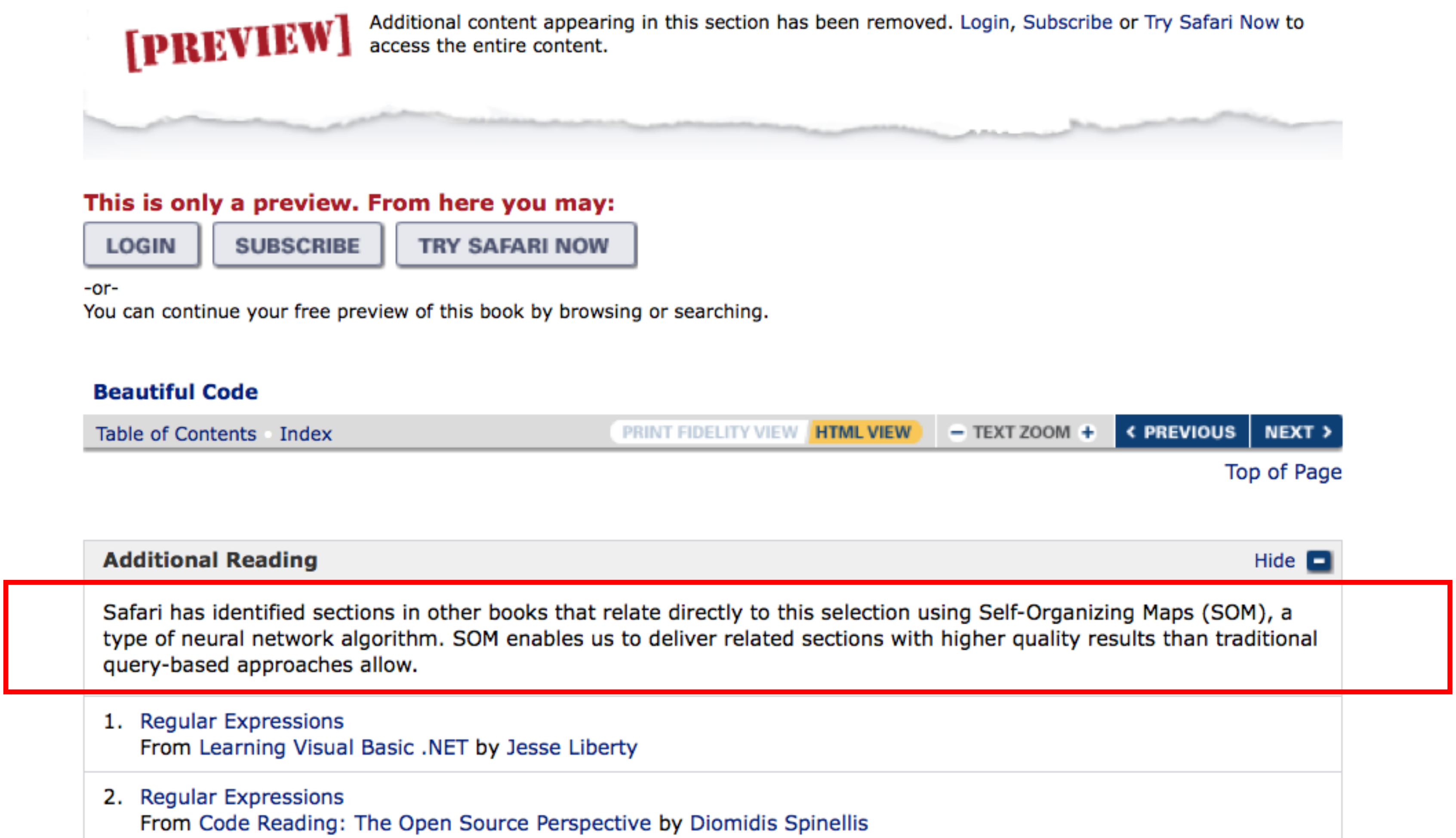
(see http://websom.hut.fi/websom/comp.ai.neural-nets-new/html/root.html)



Automatically generated labels and examples of titles in which the labels have occurred

ai - SubSymbolic AI :-O
aisb - AISB and ECCS 1997 - Call for Registrations
alamos - Graduate Research Assistantship in Los Alamos National Laboratory
atree - Atree 3.0 EK mirror sites
backpropagator's - Backpropagator's review, by Donald R. Tveter
bayes - nn and stat decisions for med: Bayes vs neorotic nets
benchmark - Benchmark data for signal processing
bootstrap - Bootstrapping vs. Bayesian
brain - Brain usage Was:Re: Function of sleep
conjugate - Moller's Scaled conjugate gradienconnect - Direct Connect Problems
consciousness - electromagnetics, brain waves and consciousness
costa - New paper available "The Application of Fuzzy Logic in
curves - Predicting Learning Curves (Ph.D. thesis, online)
decay - weight decay
elman - Elman (or Edelman?) nn, references please.
encoding - Question on Encoding Input Data
engines - IC ENGINES & NEURAL NETS
extrapolation - Generalization (Interpolation & Extrapolation)
fortran - Neural Net Fortran Code
ga - GA for learning in ANN
genesis - GENESIS 2.0
hidden - Hidden layer heuristics
intelligence - Commercial Intelligence?
interpreted - Lisp is not an interpreted language
java - NN in Java?

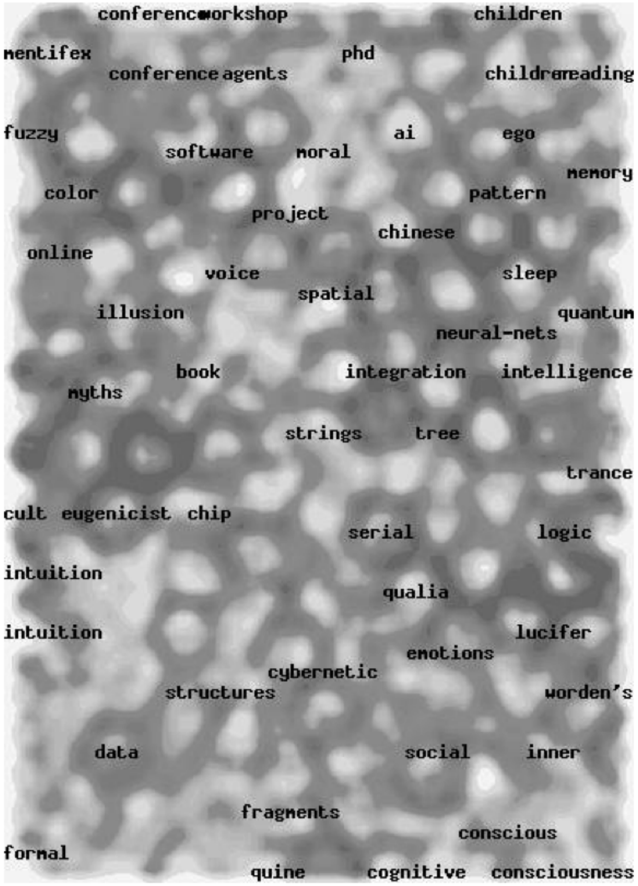
Unexpected Semantic Encounter
(O'Reilly books Safari, not Apple's Safari)



From the book “XML Topic Maps”.

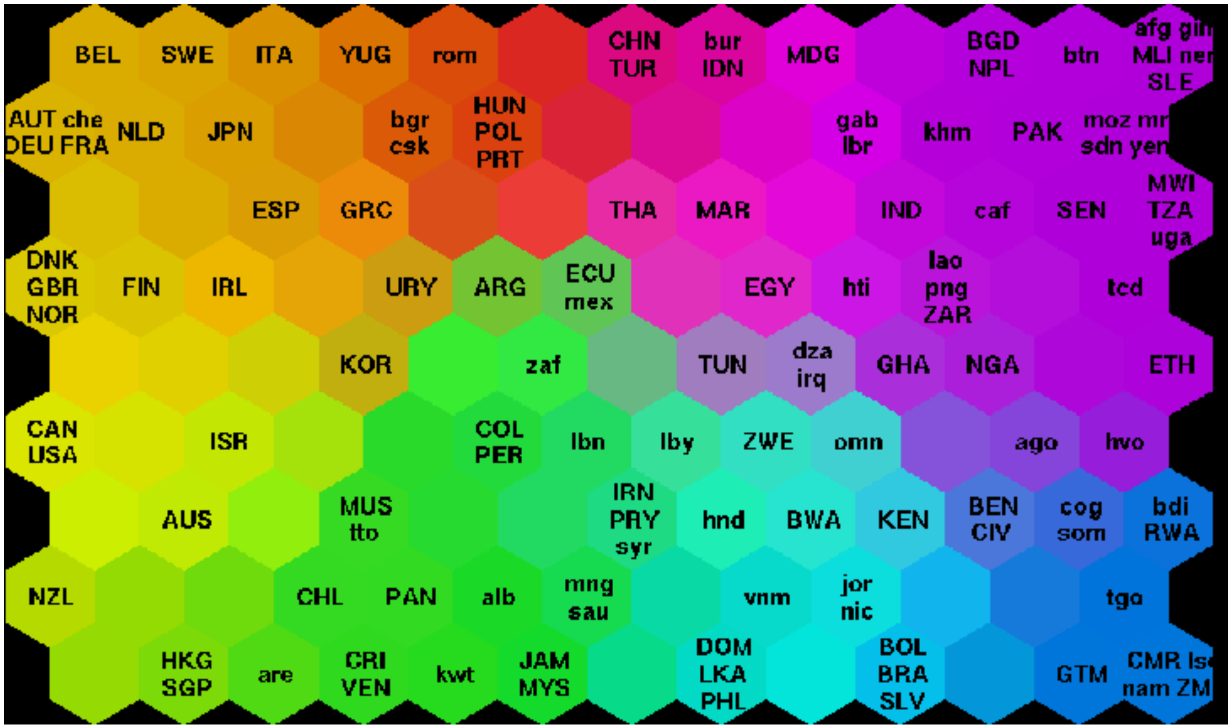
Finding structures in vast multidimensional data sets, be they measurement data, statistics, or textual documents, is difficult and time-consuming. Interesting, novel relations among the data items may be hidden in the data. The **self-organizing** map (SOM) algorithm of Kohonen [Kaski et al. 1998] can be used to aid the exploration: the structures in the data sets can be illustrated on special map displays. When applied to the mapping of documents, this algorithm automatically organizes the documents onto a two-dimensional grid so that related documents appear close to each other, as shown in Figure 11-9. This representation is suited for reflecting the structure through efficient node positioning, but it fails at displaying associations and topic maps in multiple dimensions.

Figure 11-9. Self-organizing map (courtesy of Teuvo Kohonen)



Example: Classifying World Poverty from a 39-dimension indicator

(http://www.cis.hut.fi/research/som-research/worldmap.html)



The Country Names

AFG	Afghanistan	GTM	Guatemala	NZL	New Zealand
AGO	Angola	HKG	Hong Kong	OAN	Taiwan, China
ALB	Albania	HND	Honduras	OMN	Oman
ARE	United Arab Emirates	HTI	Haiti	PAK	Pakistan
ARG	Argentina	HUN	Hungary	PAN	Panama
AUS	Australia	HVO	Burkina Faso	PER	Peru
AUT	Austria	IDN	Indonesia	PHL	Philippines
BDI	Burundi	IND	India	PNG	Papua New Guinea
BEL	Belgium	IRL	Ireland	POL	Poland
BEN	Benin	IRN	Iran, Islamic Rep.	PRT	Portugal
BGD	Bangladesh	IRQ	Iraq	PRY	Paraguay
BGR	Bulgaria	ISR	Israel	ROM	Romania
BOL	Bolivia	ITA	Italy	RWA	Rwanda
BRA	Brazil	JAM	Jamaica	SAU	Saudi Arabia
BTN	Bhutan	JOR	Jordan	SDN	Sudan
BUR	Myanmar	JPN	Japan	SEN	Senegal
BWA	Botswana	KEN	Kenya	SGP	Singapore
CAF	Central African Rep.	KHM	Cambodia	SLE	Sierra Leone
CAN	Canada	KOR	Korea, Rep.	SLV	El Salvador
CHE	Switzerland	KWT	Kuwait	SOM	Somalia
CHL	Chile	LAO	Lao PDR	SWE	Sweden
CHN	China	LBN	Lebanon	SYR	Syrian Arab Rep.
CIV	Cote d'Ivoire	LBR	Liberia	TCD	Chad
CMR	Cameroon	LBY	Libya	TGO	Togo
COG	Congo	LKA	Sri Lanka	THA	Thailand
COL	Colombia	LSO	Lesotho	TTO	Trinidad and Tobago
CRI	Costa Rica	MAR	Morocco	TUN	Tunisia
CSK	Czechoslovakia	MDG	Madagascar	TUR	Turkey
DEU	Germany	MEX	Mexico	TZA	Tanzania
DNK	Denmark	MLI	Mali	UGA	Uganda
DOM	Dominican Rep.	MNG	Mongolia	URY	Uruguay
DZA	Algeria	MOZ	Mozambique	USA	United States
ECU	Ecuador	MRT	Mauritania	VEN	Venezuela
EGY	Egypt, Arab Rep.	MUS	Mauritius	VNM	Viet Nam
ESP	Spain	MWI	Malawi	YEM	Yemen, Rep.
ETH	Ethiopia	MYS	Malaysia	YUG	Yugoslavia
FIN	Finland	NAM	Namibia	ZAF	South Africa
FRA	France	NER	Niger	ZAR	Zaire
GAB	Gabon	NGA	Nigeria	ZMB	Zambia
GBR	United Kingdom	NIC	Nicaragua	ZWE	Zimbabwe
GHA	Ghana	NLD	Netherlands		
GIN	Guinea	NOR	Norway		
GRC	Greece	NPL	Nepal		

The colors represent neurons/centers and were automatically assigned after 2-D clustering using a Kohonen map.

Map Interpretation

The same ordered display can be used for illustrating the **clustering density** in different regions of the data space. The density of the reference vectors of an organized map will reflect the density of the input samples.

In clustered areas the **reference vector**s will be close to each other, and in the empty space between the clusters they will be more sparse. Thus, the cluster structure in the data set can be brought visible by displaying the distances between reference vectors of neighboring units.

Outliers

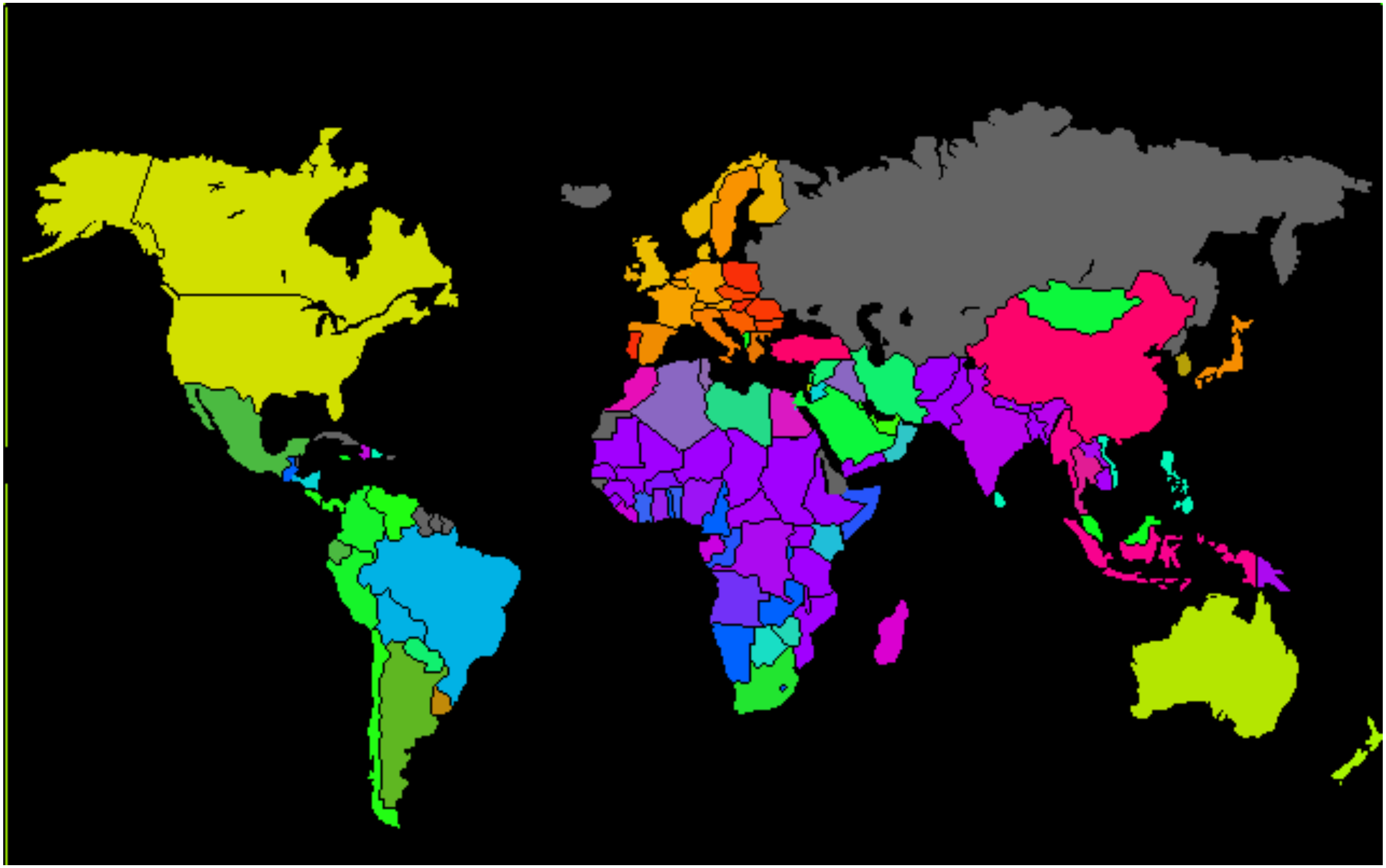
In measurement data there may exist **outliers**, data items lying very far from the main body of the data. The outliers may result, for instance, from measurement errors or typing errors made while inserting the statistics into a data base. In such cases it would be desirable that the outliers would not affect the result of the analysis. This is indeed the case for map displays generated by the SOM algorithm.

E**ach outlier affects only one map unit and its neighborhood**, while the rest of the display may still be used for inspecting the rest of the data.

Furthermore, the outliers can be easily detected based on the clustering display: the input space is, by definition, very sparsely populated near the outliers.

If desired, the outliers can then be discarded and the analysis can be continued with the rest of the data set. It is also possible that the outliers are not erroneous but that some data items really are strikingly different from the rest. In any case the map display reveals the outliers, whereby they can either be discarded or paid special attention to.

Colors (cluster representatives) transferred to a world map



Animal Similarity Experiment

Table 3.4. Animal names and their attributes

		dove	hen	duck	goose	owl	hawk	eagle	fox	dog	wolf	cat	tiger	lion	horse	zebra	cow
	small	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0
is	medium	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
	big	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
	2 legs	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
	4 legs	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
has	hair	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	hooves	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
	mane	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
	feathers	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
	hunt	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0
likes	run	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0
to	fly	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0
	swim	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Animal Similarity Experiment

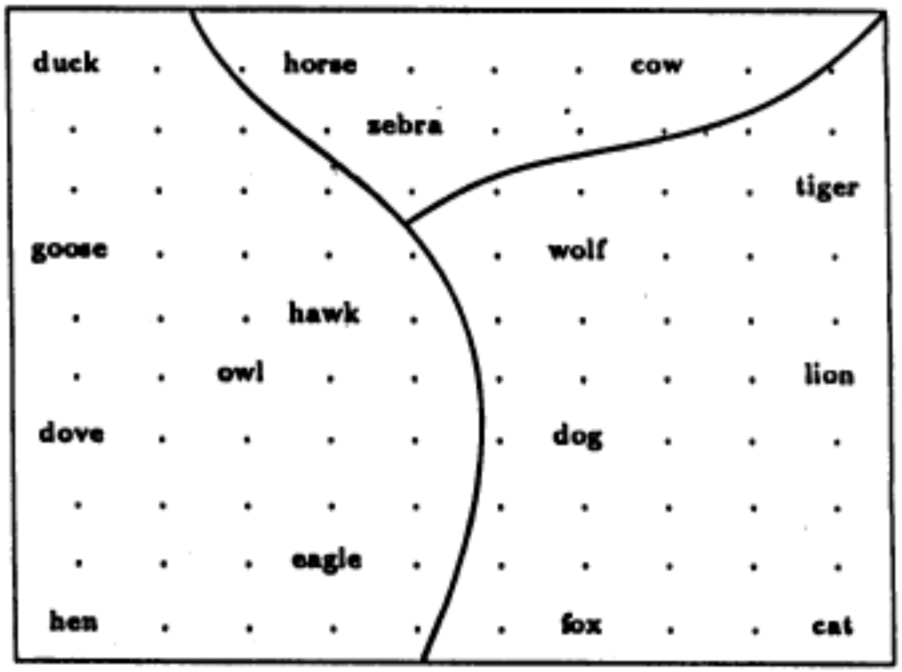


Fig. 3.22. After the network had been trained with inputs describing attribute sets from Table 3.4, the map was calibrated by the columns of Table 3.4 and labeled correspondingly. A grouping according to similarity has emerged

Semantic Map Experiment

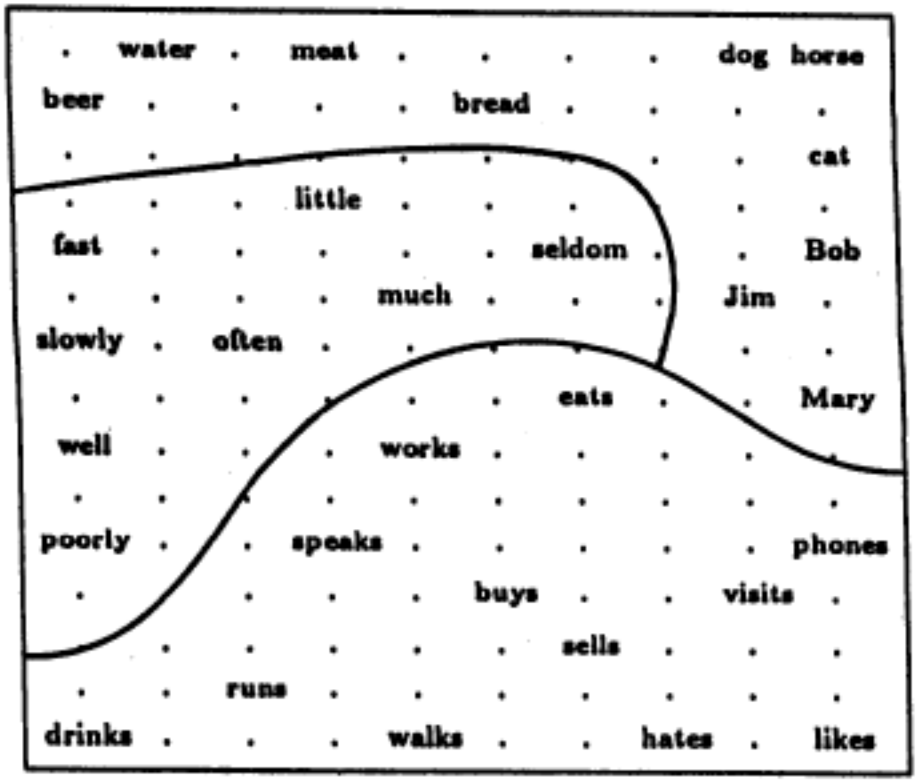


Fig. 7.11. "Semantic map" obtained on a network of 10 × 15 cells after 2000 presentations of word-context-pairs derived from 10,000 random sentences of the kind shown in Fig. 7.8. Nouns, verbs and adverbs are segregated into different domains. Within each domain a further grouping according to aspects of meaning is discernible

Tree Data Structure Experiment

- Encode a 2-dimensional tree using 5-dimensional vectors as shown:

```
ABCDE
  F
  G
  H K L M N O P Q R
  I     S     W
  J     T     X 1 2 3 4 5 6
        U     Y
        V     Z
```

all connected

connect

PATTERN	COMPONENTS				
A	1	0	0	0	0
B	2	0	0	0	0
C	3	0	0	0	0
D	4	0	0	0	0
E	5	0	0	0	0
F	3	1	0	0	0
G	3	2	0	0	0
H	3	3	0	0	0
I	3	4	0	0	0
J	3	5	0	0	0
K	3	3	1	0	0
L	3	3	2	0	0
M	3	3	3	0	0
N	3	3	4	0	0
O	3	3	5	0	0
P	3	3	6	0	0
Q	3	3	7	0	0
R	3	3	8	0	0
S	3	3	3	1	0
T	3	3	3	2	0
U	3	3	3	3	0
V	3	3	3	4	0
W	3	3	6	1	0
X	3	3	6	2	0
Y	3	3	6	3	0
Z	3	3	6	4	0
1	3	3	6	2	1
2	3	3	6	2	2
3	3	3	6	2	3
4	3	3	6	2	4
5	3	3	6	2	5
6	3	3	6	2	6

Figure 4.11 Spanning tree test data [Kohonen, 1989a].

Tree Data Structure Experiment

- Use a 2D imposed structure, with 10 x 7 neurons

- Results:

  Branches of tree tend to self-align in rows or columns of the array.

  5D $\rightarrow$ 2D mapping

Tree Data Structure Experiment

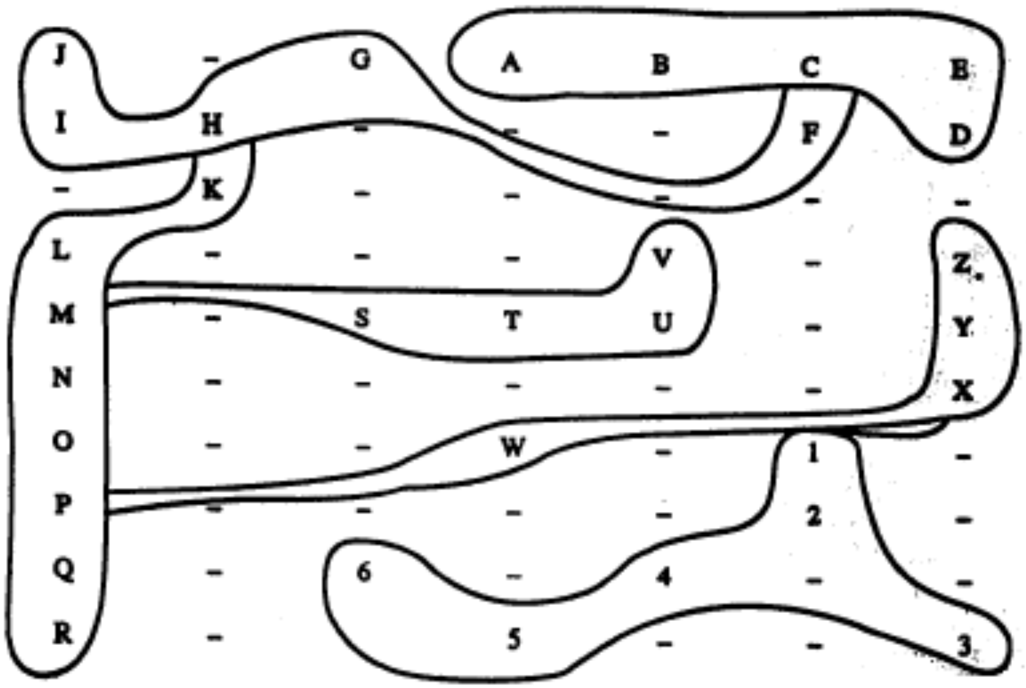


Figure 4.16 Results after another 75 iterations with $R = 0$.

Compare Original 5DTree

```
ABCDE
  F
  G
  H K L M N O P Q R
  I     S     W
  J     T     X 1 2 3 4 5 6
        U     Y
        V     Z
```

Tree Data Structure Using Hexagonal Array

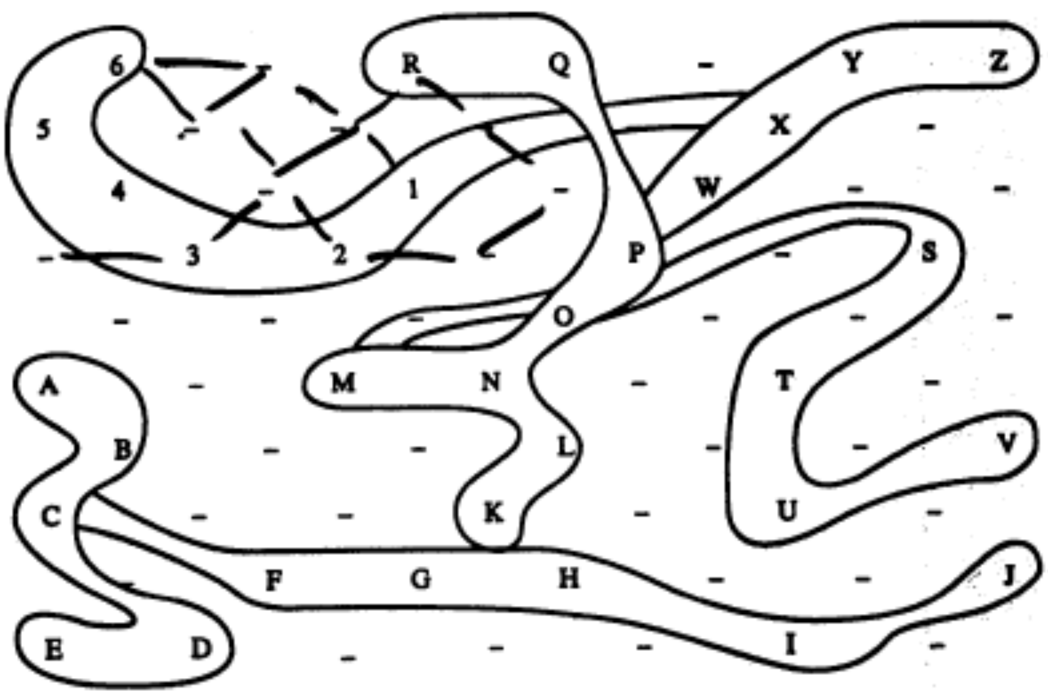


Figure 4.17 Results of spanning tree example using hexagonal array.

Compare Original 5DTree

```
ABCDE
  F
  G
  H K L M N O P Q R
  I       S     W
  J       T     X 1 2 3 4 5 6
          U     Y
          V     Z
```

Supervised Applications of Kohonen Nets

- Most applications that can be treated with MLP's can also be treated in some way by variants of Kohonen nets.

- Example: Learning a **function** f:D→R is done by treating the sample space as a set of (d, r) pairs.

Function Approximation Example

- Assume 1-dimensional domain for simplicity (not required in general).
- Represent pairs y = f(x) as (x, y).
- Learn pairs by a 2-D Kohonen net.

A Possible Aberration

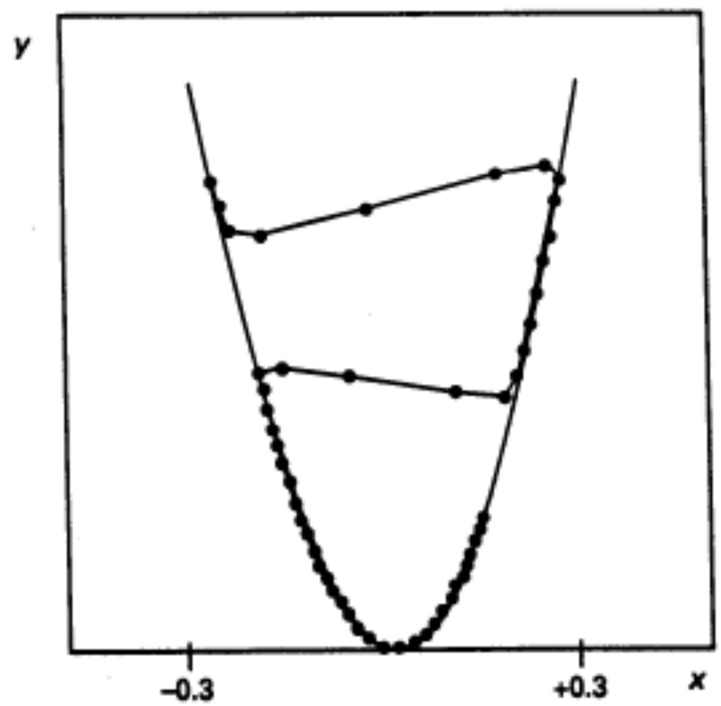


Figure 4.63 Representation of 1000 pairs of argument and function values of the function $y = 10x^2$ by two-dimensional weight vectors (dots) in a one-dimensional neural net of fifty neurons. Input argument values in $[-0.3, +0.3]$

Master-Slave Technique

- A technique known as “master-slave” can reduce the likelihood of an aberration, as shown on a previous diagram.
  - Select competition **winners** based only on x, not on the pair (x, y).
  - The weight components corresponding to x are called the **master weights**, and others are called **slave weights.**
  - Adapt both master and slave weights together.

No Aberration with Master-Slave

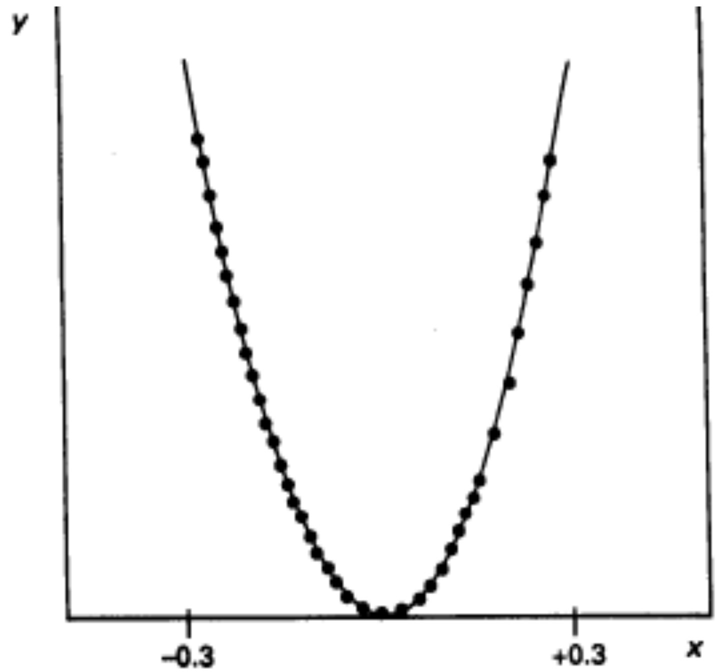
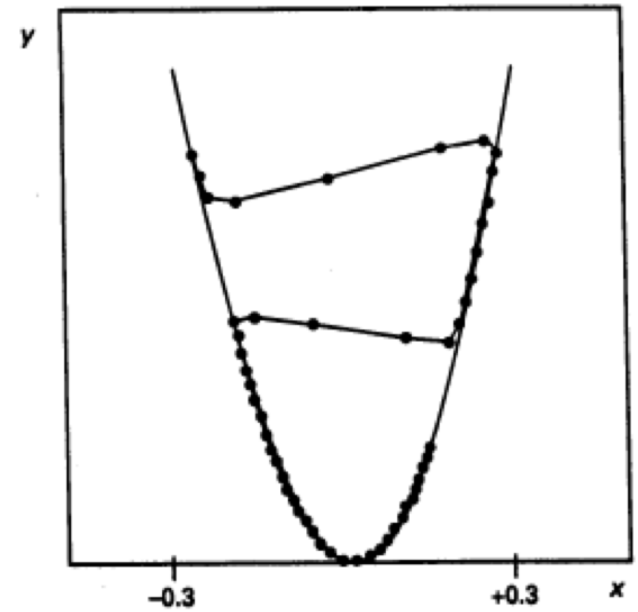


Figure 4.66 The master–slave method. The representation of 1000 pairs of argument and function values of the function $y = 10x^2$ by two-dimensional master–slave weight vectors (dots) in a one-dimensional neural net of fifty neurons. Input argument values in $[-0.3, +0.3]$



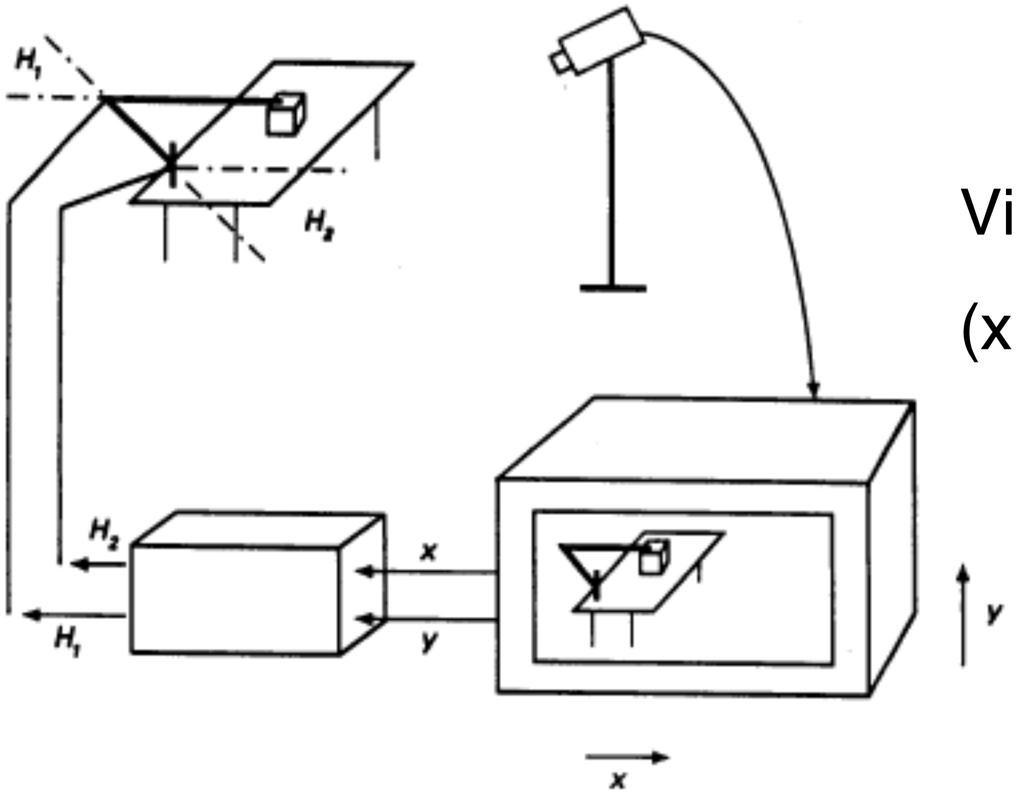
not using Master-Slave

Example: Robot Hand-Eye Coordination

- **Want a robot to coordinate its hand with what it sees on the monitor.**
- Robot arm should be able to place an object in a given position identified in **view**.
- The problem is **how to set the arm's angles** (called "inverse kinematics").

Robot Hand-Eye

Robot
Arm
(Angles)



Video View

(x, y Coordinates)

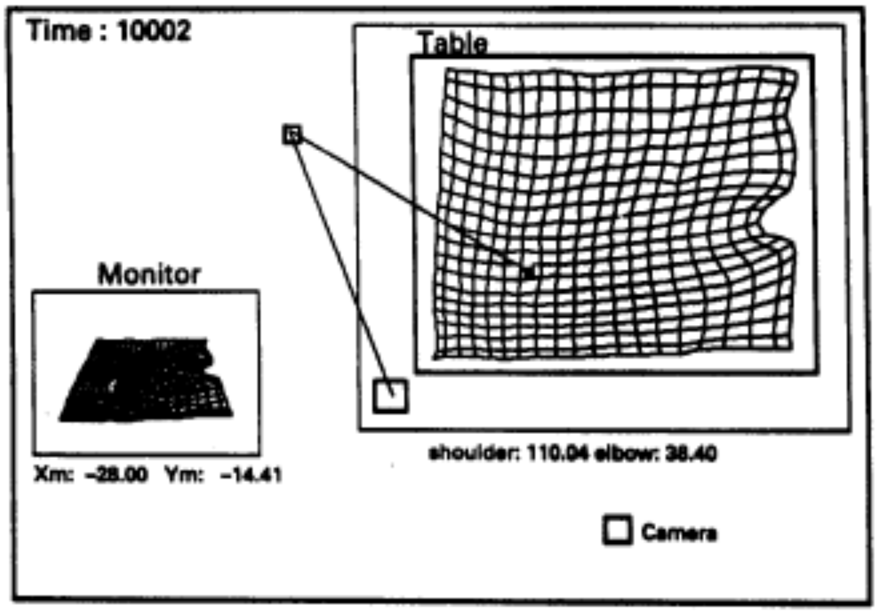
Figure 4.68 Simple outline of robot arm control

Use of Master-Slave

- The view is the master.
- The hand is the slave.
- Winner is determined w.r.t. the view.
- Both view and hand are adapted together
- Example:
  - 20x20 neuron map, 2D overlay
  - 4 weights per neuron
    (2 for master/view and 2 for slave/angles)

Video View

(x, y Coordinates)



Robot
Arm
(Angles)

Figure 4.71 Representation of table coordinates by the two-dimensional master–slave weight vectors (dots in the right-hand figure) in a two-dimensional net of 20 × 20 neurons after 10 002 training examples. Weight vectors are connected with a line if they belong to neighbouring neurons

Problems Unsuitable for Kohonen Nets

- Kohonen nets work by clustering.
- Nearby data points are expected to behave similarly, e.g. have similar outputs (topology-preserving property).
- Parity-like problems such as the XOR do not have this property. These would tend to be unstable for solution by a Kohonen net.

Other Applications Papers

The Application of SOM Network to Clustering Enterprises Based on Questionnaires*

Yan Yu[1,2], Pelian He[1], Yinghua Zhang[3], Yushan Bai[2], Zhengju Song[1], Tingting Yin[1]

ISCCSP 2008, Malta, 12-14 March 2008

Increasing Wireless Sensor Network Lifetime through the Application of SOM Neural Networks

Mario Cordina and Carl J. Debono
Department of Communications and Computer Engineering
University of Malta
Msida MSD2080, Malta
Email: mario.cordina@vodafone.com, cjdebo@eng.um.edu.mt