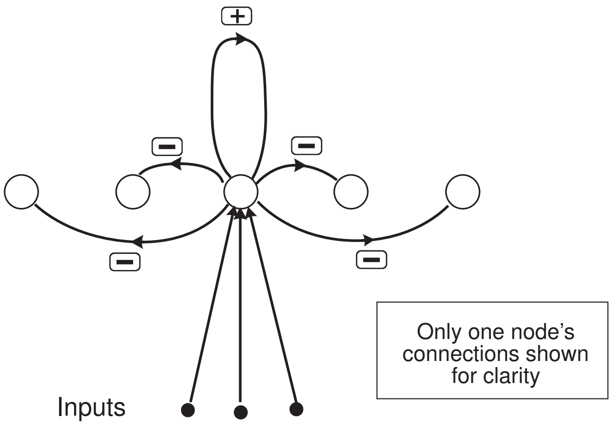
7: Competition and self-organisation: Kohonen nets

Kevin Gurney

Dept. Human Sciences, Brunel University
Uxbridge, Middx. UK

1 Competitive dynamics

Consider a layer or group of units as shown in the diagram below.



Competitive layer

Each cell receives the same set of inputs from an input layer and there are intralayer or *lateral* connections such that each node is connected to itself via an excitatory (positive) weight and inhibits all other nodes in the layer with negative weights.

Now suppose a vector $\mathbf{x}$ is presented at the input. Each unit now computes a weighted sum s of the inputs provided by this vector. That is

$$s = \sum_i w_i x_i \tag{1}$$

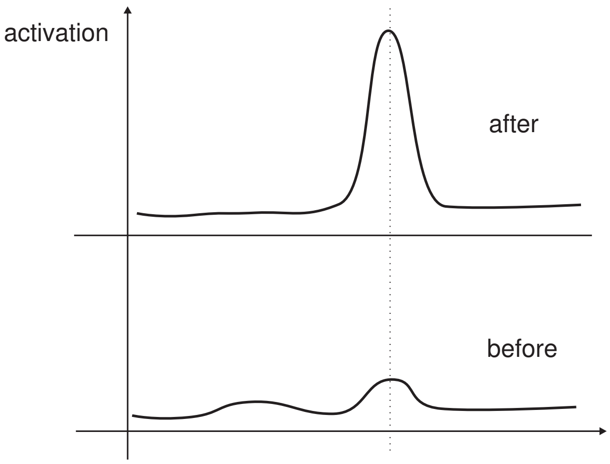
In vector notation this is, of course, just the dot product $\mathbf{w} \cdot \mathbf{x}$. This is the way of looking at things which will turn out to be most useful. Then some node k, say, will have a value of s larger than any other in the layer. It is now claimed that, if the node activation is allowed to evolve by making use of the lateral connections, then node k will develop a maximal value for a while the others get reduced. The time evolution of the node is usually governed by an equation which determines the rate of change of the activation (Lecture 1, section 'Introducing time'). This must include the input from the lateral connections as well as the 'external' input given by s. Thus if l is the weighted sum of inputs from the lateral connections

$$\frac{da}{dt} = \beta_s s + \beta_l l - \gamma a \tag{2}$$

Recall that da/dt is the rate of change of a. There will usually be a sigmoid output relation $y = \sigma(a)$

What happens is that the node with greatest excitation s from the input has its activation increased directly by this and indirectly via the self-excitatory connection. This then inhibits the neighbouring nodes, whose inhibition of k is then further reduced. This process is continued until a stability is reached. There is therefore a 'competition' for activation across the layer and the network is said to evolve via *competitive dynamics.* Under suitable conditions, the nodes whose input s was less than that on the 'winning node' k will have their activity reduced to zero. The net is then sometimes referred to as 'winner-takes-all' net, since the node with largest input 'wins' all the available activity.

If the net's activity is represented in profile along the string of nodes then an initial situation in part a) of the diagram below will evolve into the situation shown in part b).



time evolution under competitive dynamics

Competitive dynamics are obviously useful in enhancing the activation 'contrast' over a network layer and singling out the node which is responding most strongly to its input. We now examine how this process may be useful in a learning situation.

2 Competitive learning

Consider a training vector set whose vectors all have the same length, and suppose, without loss of generality, that this is one. Recall that the length $\|\mathbf{x}\|$ of a vector $\mathbf{x}$ is given by

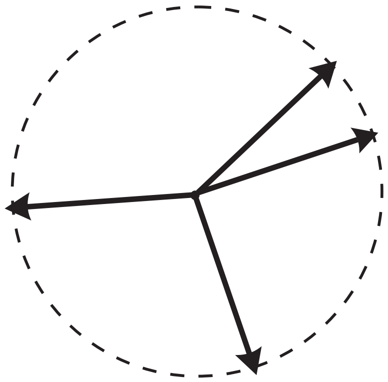
$$\|\mathbf{x}\| = \sum_i x_i^2 \tag{3}$$

A vector set for which $\|\mathbf{x}\| = 1$ for all $\mathbf{x}$ is said to be *normalised.* If the components are all positive or zero * then this is approximately equivalent to the condition

*this is consistent with the interpretation of the input as derived from the output of a previous layer

$$\sum_i x_i = 1 \tag{4}$$

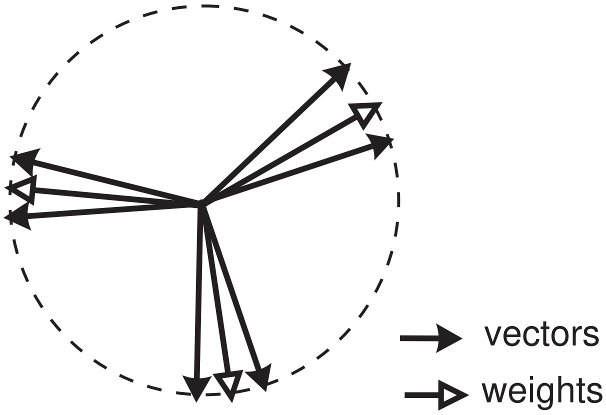
Since the vectors all have unit length, they may be represented by arrows from the origin to the surface of the unit (hyper)sphere.



vectors on unit hypersphere

Suppose now that a competitive layer has had its weight vectors normalised according to (4). Then these vectors may also be represented on the same sphere.

What is required for the net to encode the training set is that the weight vectors become aligned with any clusters present in this set and that each cluster is represented by at least one node. Then, when a vector is presented to the net there will be a node, or group of nodes, which respond maximally to the input and which respond in this way only when this vector is shown at the input.

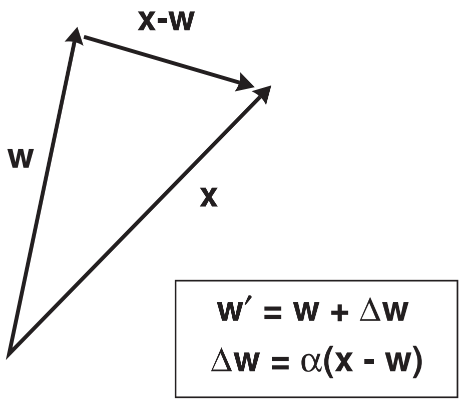


weights and vectors aligned

If the net can learn a weight vector configuration like this, without being told explicitly of the existence of clusters at the input, then it is said to undergo a process of *self-organised* or *unsupervised* learning. This is to be contrasted with nets which were trained with the delta rule or BP where a target vector or output had to be supplied.

In order to achieve this goal, the weight vectors must be rotated around the sphere so that they line up with the training set. The first thing to notice is that this may be achieved in an gradual and efficient way by moving the weight vector which is closest (in an angular sense) to the current input vector towards that vector slightly. The node k with the closest vector is that which gives the greatest input excitation s since this is just the dot product of the weight and input vectors. As shown below, the weight vector of node k may be aligned more closely with the input if a change $\Delta\mathbf{w}$ is made according to

$$\Delta\mathbf{w} = \alpha(\mathbf{x} - \mathbf{w}) \tag{5}$$



vector triangle - weights and inputs

Now it would be possible to use a supervisory computer to decide which node had the greatest excitation s but it is more satisfactory if the net can do this itself. This is where the competitive dynamics comes in to play. Suppose the net is winner- take-all so that the winning node has value 1 and all the others have value close to zero. After letting the net reach equilibrium under the lateral connection dynamics we now enforce the rule

$$\Delta\mathbf{w} = \alpha(\mathbf{x} - \mathbf{w})y \tag{6}$$

across the whole net. Then there will only be a single node (the one whose dot-product s was greatest) which has $y = 1$ and for which the weight change is the same as in (6). All other nodes will have $y = 0$ and so there weight change will also be zero. The stages in learning (for a single vector presentation) are then

1. apply vector at input to net and evaluate s for each node.
2. update the net (in practice, in discrete steps) according to (2)for a finite time or until it reaches equilibrium.
3. train all nodes according to (6)

There are a few points about the learning rule worth noting. First, if the weights are initially normalised according to (4) and the input vectors are normalised in the same way, then the normalisation of the weights is preserved under the learning rule. The change in the length of $\mathbf{w}$ is given by the sum of the changes in its components

$$\sum_i \Delta w_i = \alpha y \left(\sum_i x_i - \sum_i w_i \right) \tag{7}$$

and each of the sums in the bracket is 1 so that the right hand side is zero. The object of normalisation is a result of a subtlety that has been ignored so far in order to clarify the essentials of the situation. It was assumed the dot product s, gives an indication of the angular separation of the weight and input vectors. This is true up to a point but recall that the dot product also involves the product of vector lengths. If either the input or weight vectors are large, then s may also be large, not as a result of angular proximity (the vectors being aligned) but simply by virtue of their magnitude. We want a measure of vector alignment which does not require a separate computation of vector lengths, and the normalisation process is one way of achieving this.

Secondly, the learning rule may be expanded to the sum of two terms

$$\Delta \mathbf{w} = \alpha \mathbf{x} y - \alpha \mathbf{w} y \tag{8}$$

The first of these looks like a Hebb term while the second is a weight decay. Thus we may see competitive self-organisation as Hebb learning but with a decay term that guarantees normalisation. This latter property may be thought of in biological terms as a conservation of metabolic resources; thus, the sum of synaptic strengths may not exceed a certain value which is governed by physical characteristics of the cell to support synaptic and post-synaptic activity. There are several architectures that have used the basic principles outlined above. Rumelhart & Zipser (1986) (henceforth R & Z) give a good discussion of competitive learning and several examples. There is only space to discuss one of these here.

2.1 Letter and 'word' recognition

R & Z train using pairs of characters, each one being based on a 7 by 5 pixel grid. In the first set of experiments they used the four letter pairs *AA AB BA BB*. With just two units in a the competitive net, each unit learned to detect either *A* or *B* in a particular serial position. Thus, in some experiments, unit 1 would respond if there was an *A* in the first position while unit 2 would respond if there was a *B* in the first position. Alternatively the two units could respond to the letter in the second position. Note that these are, indeed, the two possible 'natural' pairwise groupings of these letter strings. R & Z call this net a 'letter detector'. With four units each node can learn to respond to each of the four pairs - it is a 'word detector'.

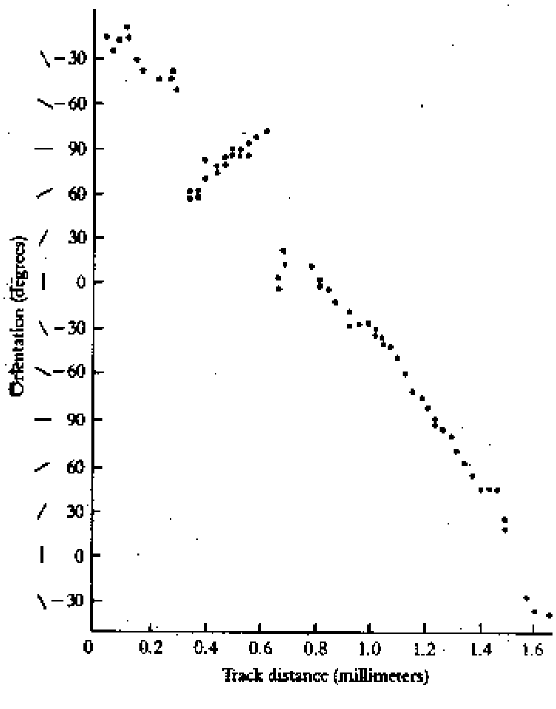
In another set of experiments, R & Z used the letter pairs *AA, AB, AC, AD, BA, BB, BC, BD*. When a net with only two units was used, one unit learned to recognise the pairs which started with *A*, while the other learned to respond to those that began with *B*. When 4 units were used each unit learned to recognise the pairs that ended in one of the four different letters *A, B, C, D*. This represents two different ways of clustering the training set. If the patterns are to be put into two groups then, clearly, it is the first letter which characterises the group. On the other hand, if there are to be four clusters, the four value feature determined by the second letter is the relevant distinction.

3 Kohonen's self-organising feature maps

3.1 Topographic maps in the visual cortex

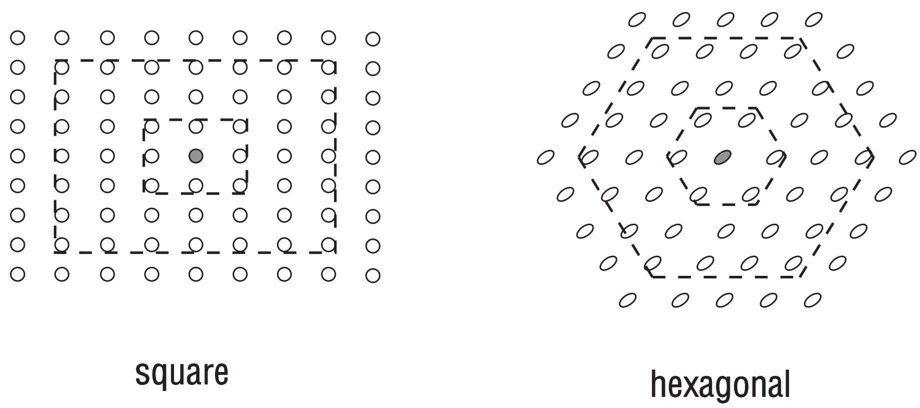
It often occurs that sensory inputs may be mapped in such a way that it makes sense to talk of one stimulus being 'close to' another according to some metric property of the stimulus. The simplest example of this occurs when the metric is just the spatial separation of localised sources. A slightly more abstract example is provided by the cells in visual area 1 of the mammalian brain, which are 'tuned' to orientation of the stimulus. That is, if a grid or grating of alternating light and dark lines is presented to the animal, the cell will respond most strongly when the lines are oriented in a particular direction and the response will fall off as the grating is rotated either way from this preferred direction. This was established in the classic work of Hubel & Weisel (1962) using microelectrode studies with cats. Two grating stimuli are now 'close together' if their orientations are similar. This defines a *metric* or measure for the stimuli.

If we were to train a competitive network on a set of gratings then each cell (unit) would learn to recognise a particular orientation. However there is an important property of the way cells are organised in biological nets which will not be captured in our scheme as described so far. That is, cells which are tuned to similar orientations tend to be physically located in proximity with one another. In visual cortex, cells with the same orientation tuning are placed vertically below each other in columns perpendicular to the surface of the cortex. If recordings are made from an electrode which is now inserted parallel to the cortex surface and gradually moved through the tissue, the optimal response from cells will be obtained at a series of orientations that vary, in general, smoothly across the cortex. There are, however occasional discontinuities as shown in the slideorienttrack



slide of data on orientation tuning

The orientation tuning over the surface forms a kind of map with similar tunings being found close to each other. These maps are called *topographic feature maps.* It is possible to train a network using methods based on activity competition in a such a way as to create such maps automatically. This was shown by C. von der Malsburg in 1973 specifically for orientation tuning, but Kohonen (1982) popularised and generalised the method and it is in connection with his name that these nets are usually known. The best exposition is given in his book (Kohonen, 1984). These nets consist of a layer of nodes each of which is connected to all the inputs and which is connected to some neighbourhood of surrounding nodes.



Kohonen neighbourhoods

3.2 The algorithm

At each vector presentation the following sequence of steps occurs

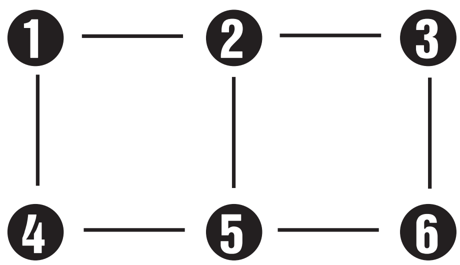
- Find the node k whose weight vector is closest to the current input vector.
- Train node k and all nodes in some neighbourhood of k.
- Decrease the learning rate slightly
- After every M cycles, decrease the size of the neighbourhood

In connection with 1), it is important to realise that Kohonen postulates that this is done with a supervisory computational engine and that his results are not based on the use of competitive dynamics to find the 'winning' node. The justification for this is that it *could* have been done by the net itself with lateral connections. The use of competitive dynamics would slow things down considerably, is very much dependent on the parameters, and does not always work, 'cleanly'. (recall the video demo). In fact the rule Kohonen uses in his examples is to look for the node which simply has the smallest value for the length of the difference vector $\mathbf{x} - \mathbf{w}$. This also appears to obviate the need for input vector normalisation (and hence for the restriction to positive components) which was a prerequisite with the inner product activation measure of proximity. However, this method cannot be the basis of any biologically plausible algorithm.

The key point in this algorithm is 2). It is through this that the topographic mapping arises. It is the use of training over a neighbourhood that ensures that nodes which are close to each other learn similar weight vectors. Decreasing the neighbourhood ensures that progressively finer features or differences are encoded and the gradual lowering of the learn rate ensures stability (otherwise the net may exhibit oscillation of weight vectors between two clusters).

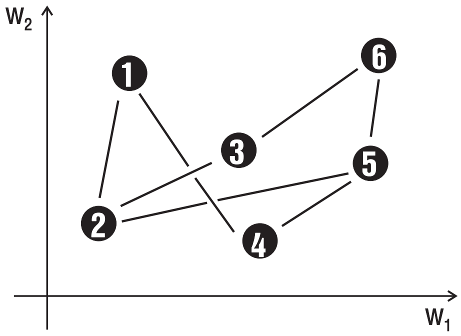
3.3 A graphic example

It is possible to illustrate the self-organisation of a Kohonen net graphically using a net where the input space has just two components. Consider a net with just 6 nodes on a rectangular grid.



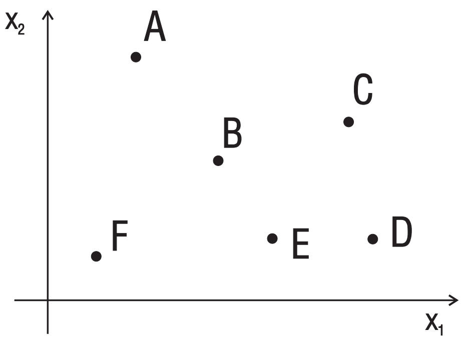
6 node net on grid

Another representation of this net is in *weight space.* Since there are only 2 weights we may draw this on the page. Initially the weights will be random, say



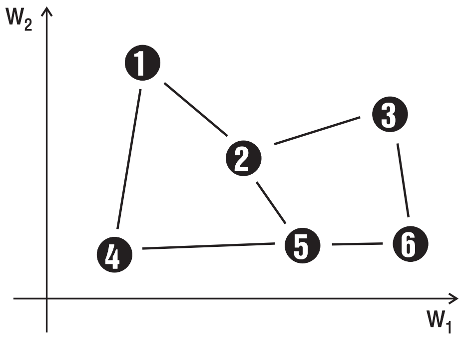
random weights

The lines are drawn to connect nodes which are physically adjacent (first diagram). Suppose now that there are 6 input vectors which may be represented in pattern space as shown below



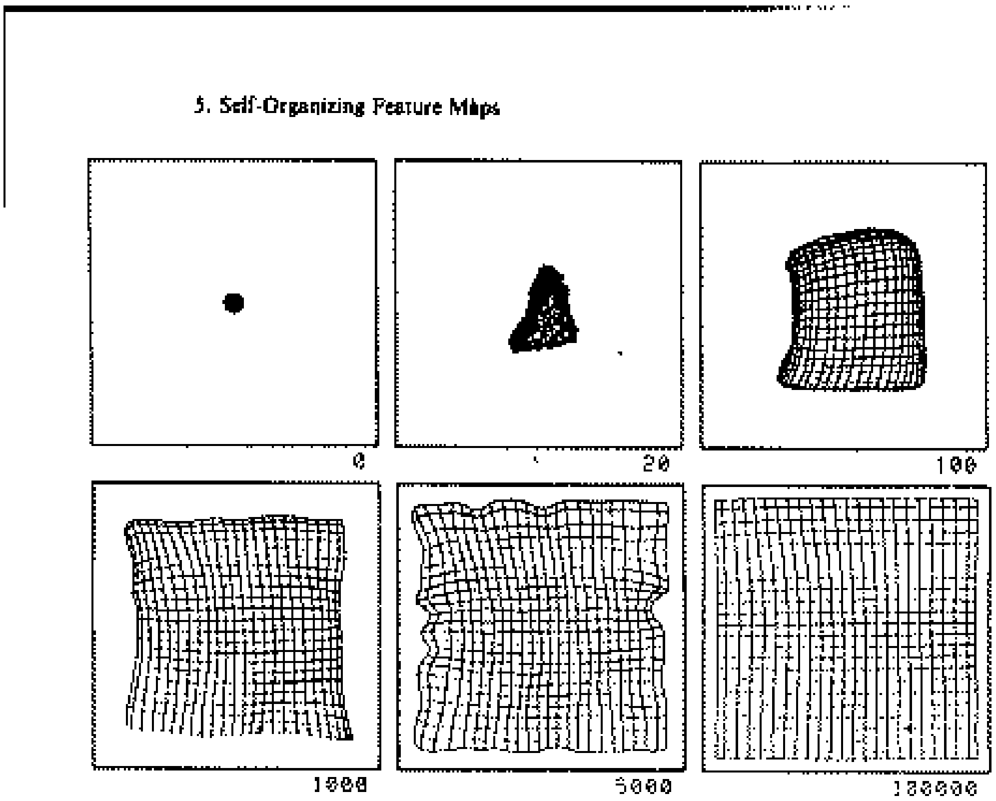
input vectors for 6 node K. net

In a well trained (ordered) net that has developed a topographic map the diagram in weight space should have the same topology as that in physical space and will reflect the properties of the training set.



Trained weight space

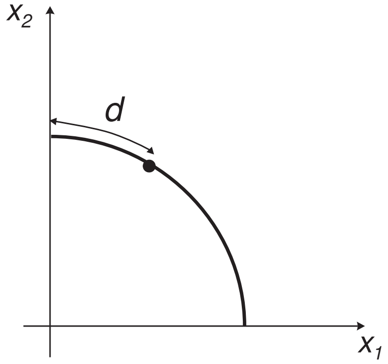
The case of 2-component vectors which are drawn randomly from the unit square and in which the initial weights are close to the centre of the unit square, is dealt with by Kohonen (1984).



slide of kohonens results for 2D square

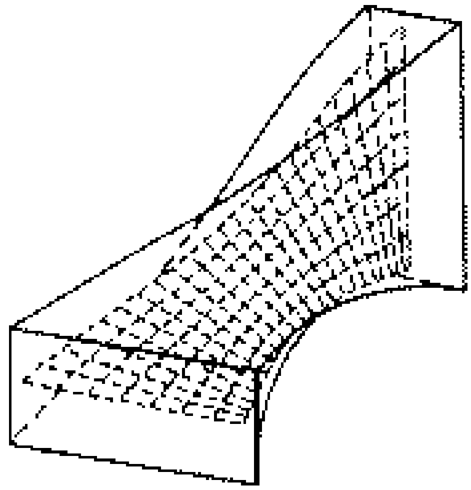
The weight diagram starts as a 'crumpled ball' at the centre and expands like a fishing net being unravelled. Kohonen deals with several other similar examples.

When the input space is more than 2-dimensional, the higher dimensions have to get 'squashed' onto the grid. This will be done in such a way as to preserve the most important variations in the input data. Thus, it is often the case that the underlying dimensionality of the input space is smaller than the number inputs. This is illustrated below for 2-D data which has an underlying dimensionality of 1



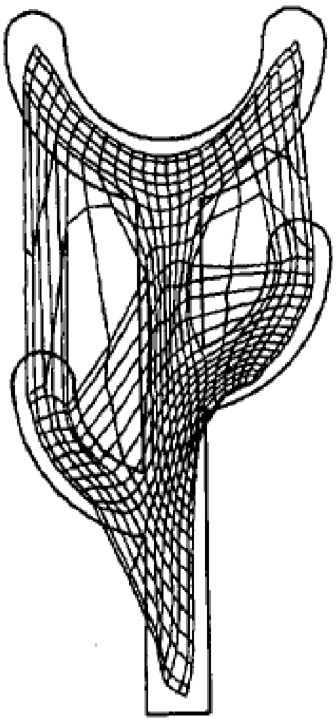
points on arc of circle

The points lie on an arc of a circle and each point may be specified by stating how far round the arc it is. A more convincing example with 3D data with an underlying dimensionalty of 2 is shown in fig 5.9 of Kohonen's book



slide of 3d projection

The topographic map will also refelect the underlying distribution of the input vectors. (Kohonen fig 5.18)



slide of 'cactus' distribution

Returning to the original example of orientation maps in the visual system some of my own recent work has focussed on training nets whose cells form a map of image velocity (Gurney and Wright, 1992)

4 Other competitive nets

Fukushima (1975) has developed a multilayered net called the 'neocognitron' which recognises characters and which is loosely based on early visual processing. The structure is quite complex and, although some of the features seem rather *ad hoc*, it is a very impressive example of modelling a large system which has many similarities with its biological counterpart.

No account of competitive learning would be complete without reference to the work of Stephen Grossberg. His Adaptive Resonance Theory (ART) has been the subject of a enormous number of papers. ART concerns the development of networks in which the number of nodes required for classification is not assigned *ab initio* but is determined by the net's sensitivity to detail within the data set (given by the so-called *vigilance* parameter). The network is embedded in a control loop which is an integral part of the entire system. It would require a complete lecture to do justice to Grossberg's networks however and it will have to suffice here to simply give a reference. This in itself is not easy - Grossberg's work is often quite hard to read and any single reference will undoubtedly be inadequate. One possible route is his 1987 paper in Cognitive Science (Grossberg, 1987).

References

Fukushima, K. (1975). Cognitron: a self-organizing multilayered neural network. *Biological Cybernetics*, 20:121 – 136.

Grossberg, S. (1987). Competitive learning: from interactive activation to adaptive resonance. *Cognitive Science*, 11:23 – 63.

Gurney, K. and Wright, M. (1992). A self-organising neural network model of image velocity encoding. *Biological Cybernetics*, 68:173 – 181.

Hubel, D. and Wiesel, T. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160:106 – 154.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59 – 69.

Kohonen, T. (1984). *Self-organization and associative memory*. Springer Verlag.

Rumelhart, D., McCllelland, J., and The PDP Research Group (1986). Feature disc overy by competitive learning. In *Parallel Distributed Processing*.

von der Malsburg, C. (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14:85 – 100.