Introduction to Neural Networks
U. Minn. Psy 5038

Self-organizing Adaptive Maps

Initialization

In[151]:=
```
Off[SetDelayed::write]
Off[General::spell1]
```

Cortical maps

Work in monkey, and human brain, shows that the cortex is characterized by numerous distinct areas. It has been estimated that there are more than 30 visual areas alone in the macaque cortex. The earlier areas typically show a spatial topographic representation of visual space--nearby regions of visual space map to nearby regions of cortex. The so-called retinotopic map of the primary visual area (V1) is the clearest example of this (cf. Engel et al., 1994). Other visual areas of the brain also show geometrical organization (Wandell et al. 2005). We earlier saw that more abstract features, such as orientation, show spatial organization where similar orientations map to nearby spatial locations. Other areas of the brain show spatial organization of "non-spatial features". For example, the auditory cortex has tonotopic maps in which the spatial order of cell responses corresponds to pitch or acoustic frequency (Talavage et al. (2004) describe neuroimaging results in humans). The somatosensory cortex also shows a spatial organization (the "cortical homunculus").

In regions of the cortex with no obvious maps, it is quite possible that other kinds of maps wait to be discovered. Tanaka and colleagues (Tanaka, 1996; 2003) have shown that region TE of the monkey inferotemporal cortex has columns with cells that have similar visual shape preferences. Along the surface of the cortex, receptive field properties may correspond to other kinds of variation, such as rotation in depth of a face, over limited extents (on the order of 1 mm or so).

The widespread use of spatial organization in cortex suggests the possibility of a general constraint underlying the development of neural receptive field organization. We know more about primary visual cortex than any other area, so let's take a closer look at what it does.

Quantitative modeling of the retinotopic map to V1

We've learned that primary cortex is spatially organized so that nearby image points map to nearby cortical points. Can we say more about the metrical structure of this mapping? As one moves from an image point above the foveal/fixation point (i.e. starting at a point a fixed distance along the vertical meridian) along an arc (say counter-clockwise), the corresponding point on V1 moves up in a roughly straight line from the lower bank (towards the lingual gyrus) of calcarine the to the midline and then up on the upper bank (towards the cuneus). In other words, retinal rings map (approximately) to vertical cortical lines. If one moves from the fovea along a "spoke" to the periphery, the corresponding point on V1 moves from near the pole (most posterior point) of the occipital cortex toward interior and anterior region of V1. In other words, retinal spokes map (approximately) to horizontal lines. The change from image coordinates to cortical coordinates has been modeled as a log polar or complex log map (Schwarz, 1977). For a demo, see smallRetinaCortexMap.nb or this demo. These topographic properties are used to distinguish the boundaries between visual areas such as V1 and V2.

Let's treat the cortex as a 2D sheet. The topographical map says how to map retinal positions to cortical positions: i.e. take 2D inputs to 2D outputs. But we know that cortex represents more than positional features. Cells show selectivity for the degree of ocularity, orientation, motion,...This suggests that a functional role for the spatial organization of cortex is to map N-D inputs (in feature space) to 2-D outputs (topographic cortical space), where N>2.

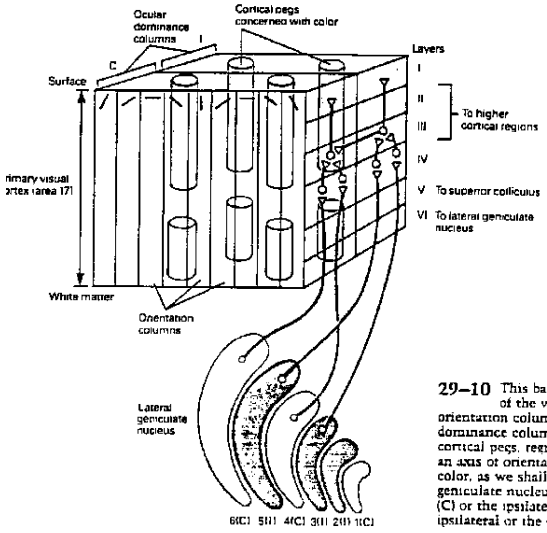
Dimension reduction framework for understanding cortical maps

Primary visual cortex does not simply have the job of representing nearby retinal points at nearby cortical locations. Much physiological research has shown that V1 brings together information from the two eyes, along similar orientations, as well as location. Together with anatomical studies, it is now commonly accepted that in many species, including humans, neurons with similar orientation preferences and various degrees of relative input from the two eyes are organized into "hypercolumns" (See Figure below, and earlier Lecture). (A major puzzle, however, is the observation that not all species have ocular dominance columns, and the function of such columns is not understood (Horton and Adams, 2005).) Hypercolumns preserve spatial contiguity and smoothness of the placement of neurons selective for features of the input.

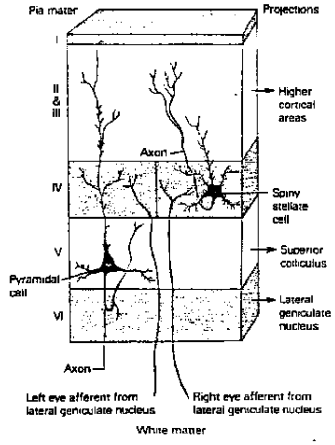
This observation suggests that a general principle may account for the organization and development of cortical maps: Neighboring points in feature or parameter space (e.g. orientation, ocular dominance, as well as retinal position) should map to nearby points on the 2D cortical sheet. (See: Durbin & Mitchison, 1990)

The underlying assumption is that most operations performed in the cortex are local, thus the related input for these computations should be physically near the computing units. For example, one task of vision is to go beyond the mere detection of contour segments, but to link contours that are likely to belong together to form a global object outline. Thus it would make sense to have the cells that signal similar orientations to be near. Visual information from the two eyes is close in the world, but separated by a great distance anatomically in the left and right eyes. This information needs to be brought physically together to process the two images binocularly, for example, to compute stereoscopic depth. Further, operations that occur frequently, that need to combine many sources of information, and that need to be done quickly could be done more efficiently if the brain could avoid having too many long connections.

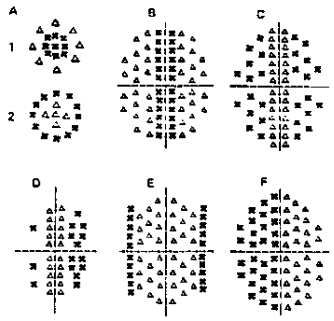
But there seems to be a problem: How to map a high dimensional feature vector to a 2-dimensional surface?



29–10 This basic cortical module (hypercolumn) in area 17 of the visual cortex contains a complete set of orientation columns representing 360° and a set of ocular dominance columns. Each hypercolumn also contains several cortical pegs, regions of cortex in which the cells do not have an axis of orientation. The cells in the pegs are concerned with color, as we shall see in Chapter 30. Each layer of the lateral geniculate nucleus receives input from either the contralateral (C) or the ipsilateral (I) eye and projects in turn to the ipsilateral or the contralateral ocular dominance columns.



28–14 The afferent and efferent connections of the primary visual cortex are made in specific layers of cortex.



29–7 Comparison of the receptive fields of neurons in the retina and in the lateral geniculate nucleus with those of simple cortical cells in area 17. **A.** Cells of the retina and lateral geniculate fall into two classes: on-center (1) and off-center (2). **B–F.** Neurons of the primary visual cortex also fall into two major classes: simple and complex. Each of these classes, moreover, has several subclasses. This is illustrated here for simple cells. Despite this variety, however, all simple cells are characterized by three features: (1) specific retinal position, (2) their discrete excitatory (x) and inhibitory (Δ) zones, and (3) specific axis of orientation. For simplicity, only receptive fields with a vertical axis of orientation from 12 to 6 o'clock are shown in this figure; each has a rectilinear configuration. In fact, each region of the retina is represented in area 17, not only for this but for all axes of orientation—vertical, horizontal, and various obliques. (Adapted from Hubel and Wiesel, 1962.)

Minimum wiring length constraint

■ Nematode

A number of people have sought a simple organizational principle that would predict the spatial layout of neurons. One such principle is that the layout of nervous system components minimizes total connection cost. Christopher Cherniak, a philosopher at the University of Maryland calculated the total wiring length for the ~40,000,000 (11!) possible layouts of the 11 hypothetically "moveable" ganglia (connecting 302 neurons) in the nematode worm *C. elegans*. Remarkably, he reported that the layout the worm actually has is indeed the one with the shortest total connection length (Cherniak, 1991, 1995, 2004). Similar arguments have been made by Cherniak and others for the layout of the multiple areas of cortex. (But see Young, 1994). The problem of minimizing connection lengths is also encountered in VLSI component layout in the design of computer chips.

```
11 !
```

```
39 916 800
```

Why is the brain in the head? (See Cherniak)

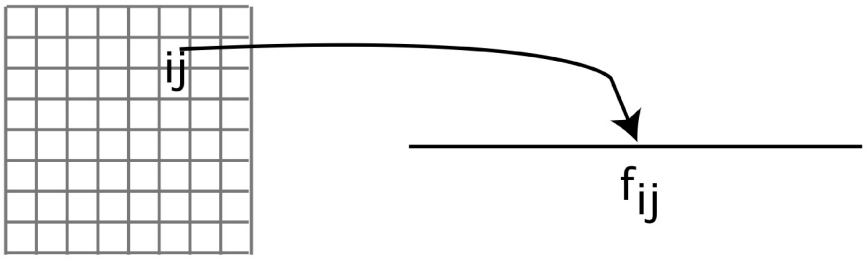
Can you think of exceptions to a minimum wiring constraint?

■ Minimum wiring length & dimensionality reduction in cortical maps

One interesting biological application of the idea of reducing the cost of orderings was published by Durbin and Mitchison in Nature (1990).

Let's look at a simple and small version of the problem that Durbin and Mitchison addressed, that of mapping a higher dimensional parameter space to one of lower dimension. Suppose we have a 2D feature space that we wish to map to a "1D cortex". Points in the NxN 2D feature space can be represented by indices (i,j) or with an appropriate mapping by an index number, f_{ij} (that ranges from 1 to N^2) assigned to the (i,j)th coordinate. Then f_{ij} specifies the position in the 1D representation.

In standard raster ordering for images (e.g. the signal sent to your TV), matrix rows are laid out one after the next in one long vector. This is exactly what we've done earlier when we take an image in matrix format and use **Flatten[]** to convert it to a vector. While nearby horizontal pixels are still close, nearby vertical pixels in the image now become far apart in the vector representation. f(i,j+1) - f(i,j) = 1, but f(i+1,j) - f(i,j) = N. A question of mathematical interest is whether there are other possible orderings that give lower costs, for example in the sense of minimizing the sum of the distances.



There are several ways of assigning costs for various orderings. Mitchison and Durbin analysed the following connection cost:

$$C(f) = \sum \Delta_{ij}$$

$$\Delta_{ij} = \left| f_{i,j+1} - f_{i,j} \right|^q + \left| f_{i+1,j} - f_{i,j} \right|^q$$

If q = 1, then the standard "raster" ordering, the index,

$$f_{ij} = (i-1)N + j$$

What are the inverse functions that map f_{ij} ->{i,j}?

---

gives a cost of:

$$C(f) = N^3 - N$$

It is not computationally feasible to find the minimum cost for dimensional reduction mappings of higher dimension, for example from (x,y,r,theta)->(x',y') as one would like to do for the formation of retinotopic and orientation maps in V1. The alternative is to see whether some biologically plausible rules could act to accomplish an efficient mapping of the higher dimensional feature space onto the 2D cortex.

Two biologically plausible rules are: 1) there are competitive winner-take-all interactions selective for distinct inputs 2) the units also strengthen their responses to those stimuli that their neighbors respond to. The first rule divides up the input domain, and the second rule imposes a continuity constraint on the formation of a map. Durbin and Mitchison developed an algorithm which applied these rules and showed that the kind of 2D maps which developed looked very much like the visual cortical maps, revealed from photo-sensitive dye studies (e.g. T'so et al., 1990).

Let's look at the general problem of how to get nearby neurons to be selective for "nearby" features. For that, we'll step back in time to the classic work of Teuvo Kohonen.

Kohonen's algorithm for topology-preserving mappings

■ Theory

The Finnish scientist, Teuvo Kohonen, was the first to develop topology-preserving adaptive maps for neural networks (Kohonen, T. , 1984). Let's look at the basic structure of a simple adaptive map.We consider a simple feedforward network, and the question is how should the weights adapt to the input patterns so that nearby features map to nearby neurons.

Kohonen boiled the essential features of self-organizing topology-preserving maps down to two basic processes:

1. Find the neuron that shows the most activity among a set of neurons in response to a specific randomly sampled input. We assume that maximum activity occurs for patterns which match the receptive field (i.e. as with a cross-correlator).

2. Define a set of neighbors around this maximum, and make these neighbors more likely to respond to that input in the future by making their weights more like the input. Typically the neighborhood N_c starts off large, and is gradually reduced over time.

Let x be an n-dimensional vector representing a feature sample. Let m_i be a n-dimensional vector representing the weights of the i^{th} unit. Let $x(t_k)$ and $m_c(t_k)$ be the vector and weight values at time t_k for neuron c. We will follow an example by Kohonen and use the following rules:

1. Similarity matching. Find unit c such that:

$$\left|x(t_k) - m_c(t_k)\right| = \min_i \left|x(t_k) - m_i(t_k)\right|$$

The idea is to find the neuron that responds best to the input pattern x. In this example, the one that responds best is the one whose weights are the best match to the input pattern itself.

Exercise

What is the relationship between: 1) the "distance" between an input vector and a neuron's weights and 2) the dot product between the input and the neuron's weights?

2. Updating. Update the weights for unit c, and all the units within c's neighborhood:

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

The idea is to adjust the weight vector m in a direction that brings it closer to the input pattern x. $\alpha(t_k)$ controls the learning rate. In our simulations below, we'll ramp it down linearly.

Exercise

What is the steady-state solution for this learning rule?

Demonstration of Kohonen's algorithm for mapping 2D features to a 1D line

Let's simplify the problem. Imagine a 1D visual cortex, i.e. the neurons are arranged in a straight line (rather than a 2D sheet). But the input images are 2D. We want to map 2D to 1D. In general, the input could represent abstract 2D features specified by continous valued inputs (e.g. 2 units whose values represent x and y positions), and these get mapped to a discrete set of output units whose position is correlated with the feature values. The biologically realistic retina to cortex problem is set up differently--the inputs themselves are arranged in a 2D spatial array and get mapped to a 2D cortical sheet. However, as discussed above, when one considers other features such as orientation, we have a problem in which a higher dimensional feature space (e.g. a 3D space representing 2D position and oriention) is mapped to a lower dimensional space (a 2D space representing location of neurons on the cortical sheet).

■ Define functions

In[153]:=
```
(*ramp[] is used below to define both the α() term,
    and the rate of change of the neighborhood size *)

ramp[x_,yint_,end_,plat_] :=
    If[(x>=end)||(-2*x*yint/end+yint<plat),plat,
    -2*x*yint/end+yint];


(*rv randomly samples the 2D "feature space"*)
rv := {RandomReal[],RandomReal[]};

(*These are alternative samplings to try
rvdiscrete := 1/8 + (Floor[(4 rv)]/4);
rvline := {xx=RandomReal[],xx};*)
```

■ Neighborhood function

The neighborhood function determines the neighbors, and thus the topology of the connections between the neurons. In our example, the neigborhood is 1-D and is defined along a line.

neigh[] is a neighborhood function that produces a list of indices for the neighbors of unit **c**. We will not use a toroidal geometry here. Instead, **neigh[]** generates shorter lists of indices near the borders, so the **min_** and **max_** of the range need to be specified. This neighborhood function only defines neighbors along a line, i.e. in one dimension. You could elaborate this algorithm to find maps from 2D to 2D, allowing neighbors to be nearby regions of 2D space.

In[158]:=
```
neigh[c_,numneigh_,min_,max_]:=Module[{i,nn,temp},
            temp = numneigh/2;
            nn={};
            For[i=c-temp,(i<=c+temp)&&(i<=max),i++,
                If[i>=min,nn=Join[nn,{i}]];
                ];
            Return[nn];
            ];
```

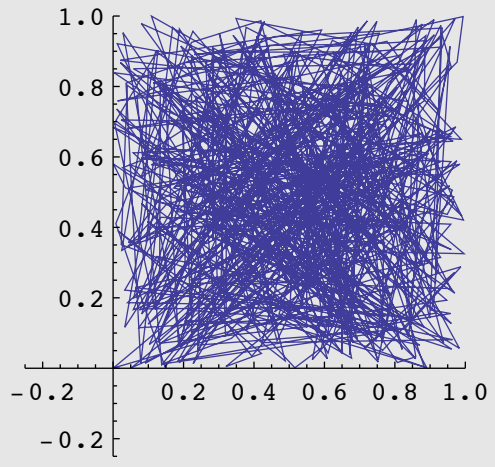
■ Initializing the simulation parameters

n is the number of nodes in the 1D line. **mu** is the matrix with the weights that will get updated according to the above update rule. **niter** is the number of iterations.

numneigh0 is the intial neighborhood size. If this is too small, the topography map can get tangled. We will start off with a neighborhood size that is 60-80% of the total size, **n**. Execute the cell below, first with n = 10, so you can see what is being represented. Then set n=600 for the simulation below.

In[219]:=
```
n = 600;
mu = Table[rv, {j, 1, n}];
g1 = ListPlot[mu, PlotRange → {{-0.25, 1}, {-0.25, 1}},
   AspectRatio → 1, Joined → True, ImageSize → Small];
niter = 4000; numneigh0 = Floor[0.6 * n / 2]; eta[t_] := ramp[t, 0.9, niter, 0.1];
numneigh[t_] := 2 Floor[ramp[t, numneigh0, niter, 2]];
g1
```

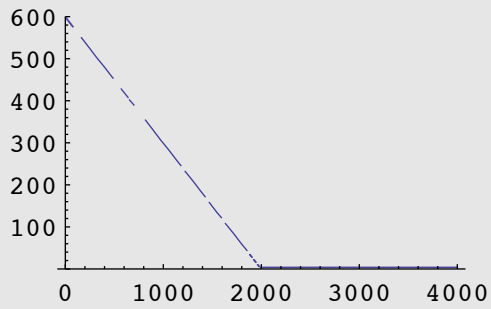
Out[223]=



Let's take a look at how the neighborhood size and α =**eta**, decrease with the number of iterations, **t** :

In[203]:=
```
Plot[eta[t], {t, 1, niter}];
Plot[numneigh[t], {t, 1, niter}, PlotRange → {0, 2 numneigh0},
 ImageSize → Small]
```

Out[203]=



■ The algorithm

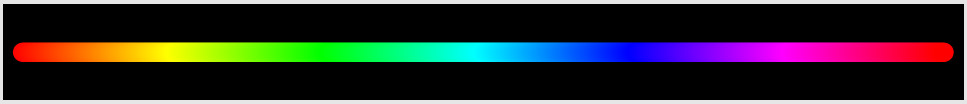
We will make a series of plots, showing the first ten iterations, and after that sampling every 50. The plots will show how the matrix **mu** (which evolves the topography of a 1D line, because of the way we defined the **neigh[]** function) gradually fits itself to the geometry of the 2D input space.

We'll use hue to colorcode the positions of the units along the line. So unit 1 (and others near it) on the left are reddish orange, ones near the middle (neuron #200) are greenish blue, and

on the right (near neuron #400), they are magenta-reddish.

In[228]:=
```
gcolor2 = Graphics[Table[{Hue[i/n], PointSize[0.02`], Point[{i/n, n/2}]},
    {i, 1, n}]];
Show[gcolor2, Background → RGBColor[0, 0, 0], AspectRatio → 1/10]
```

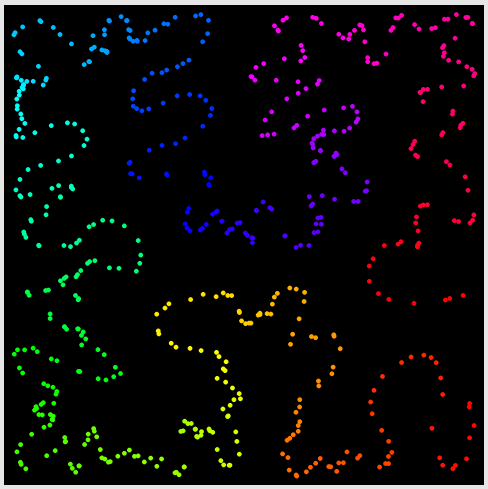
Out[229]=



In[230]:=
```
gcolor = Graphics[Table[{Hue[i / n], PointSize[0.01], Point[mu[[i]]]},
    {i, 1, n}]];
```

In[231]:=
```
Dynamic[Show[gcolor, Background -> RGBColor[0, 0, 0], AspectRatio -> 1 ,
  ImageSize → Small]]
```

Out[231]=



In[232]:=
```
For[t=1,t<=niter,t++,
    If[(Mod[t,50]==1) || t<=10,
        gcolor = Graphics[Table[{Hue[i/n],PointSize[0.01],
        Point[mu[[i]]]},{i,1,n}]];
    ];

 (*Pick a uniformly distributed "feature" sample from a 2D array*)

    x=rv;

 (*Do the similarity matching. mini is the unit whose weights best match th

    diffs = Map[Norm,Transpose[Transpose[mu]-x]];
    minarg = Min[diffs];
    mini = Part[Position[diffs,minarg],1,1];

 (*Make a list, j,  of the neighbors for this index,at this t^th iteration *

    j=neigh[mini,numneigh[t],1,n];

 (*Update the weights in the neighborhood of i to move them
    towards feature x, by eta proportion of the difference*)

    For[s=1,s<=Length[j],s++,
        mu[[ j[[s]] ]] = mu[[ j[[s]] ]] +
        eta[t] (x-mu[[j[[s]] ]])
    ];
];
```

Let's summarize what we have done. There are n (= 600) "neurons", each with 2 input weights, represented by matrix **mu**. We imagine representing the weights of these 400 neurons by a location in weight space. To the extent that a neuron's weights define a template for feature matching, nearby points in weight-space should correspond to nearby points in input or "feature space". So you can also think of the two dimensions of our plot as representing two dimensions in feature space. Neural selectivity divides up and covers feature space. The weight vectors are represented in the same coordinate system as the input vectors in order to show which neuron each weight vector belongs.

A point for each neuron is represented by a different hue in the graph above--neurons with similar hues are neighbors along a line, i.e. next to each other in our 1D "cortex". We randomly sampled a location in the 2-dimensional input space defined by the unit square. Thus, initially for example, "reddish" points (that are close on the 1D cortex) were scattered all over in weight space. Nearby neighbors could be activated by quite different stimulus features. Not good.

We then surveyed the **n** neurons to see which one had input weights that were closest to the sampled location. Note that we didn't have to calculate the response of the unit--which if linear would be the cross-correlation of the input with its weights. Then we adjusted the 2 weights of that neuron to move them closer to the sampled input point. Further, we adjusted the weights of all of the neighbors of that neuron (i.e. those with similar hues) to be closer too. We reduced the size of the neighborhood as the number of iterations increased. So a unit's weights are less affected by distant neurons as time goes on. The end result is that nearby points in feature space tend to activate nearby neurons that are arrayed in a 1-D line. For other interesting examples, and for a discussion of the relationship of Kohonen maps to space-filling curves, see Kohonen (1984).

Although we motivated Kohonen topology-preserving networks with the problem of feature mapping in cortex, there is a large range of applications that extend outside the problem we've considered (e.g. regression, Cherkassky & Lari-Najafi, 1991).

Exercises

Compute the connection cost for the Kohonen adaptive map in the above example. Compare it to a raster scheme.

Try sampling from rvline,and watch how the algorithm learns the topology of the 1D input space.

Try playing with the intial neighborhood size, numneigh0.What happens if it starts off small,(e.g.let the number of neighbors be fixed at 2 throughout the similation).

Define a 2D feature input space which is not rectangular. For example, rvcould sample from a triangular or circular region within the unit square.

References

Adams, D. L., Sincich, L. C., & Horton, J. C. (2007). Complete pattern of ocular dominance columns in human primary visual cortex. J Neurosci, 27(39), 10391-10403.

Bonhoeffer, T., & Grinvald, A. (1991). Iso-orientation domains in cat visual cortex are arranged in pinwheel-like patterns. Nature, 353(6343), 429-431.

Bonhoeffer, T., & Grinvald, A. (1993). The layout of iso-orientation domains in area 18 of cat visual cortex: optical imaging reveals a pinwheel-like organization. J Neurosci, 13(10), 4157-4180.

Cherkassky, V., & Lari-Najafi, H. (1991). Constrained Topological Mapping for Non-Parametric Regression Analysis. Neural Networks, 4(1), 27-40.

Cherniak, 1991, Component placement optimization in the brain, UMIACS-TR-91-98 or CS-TR-2711)

Cherniak, C. (1995). Neural component placement. TINS, 18(12), 522-527.

Cherniak, C., Mokhtarzada, Z., Rodriguez-Esteban, R., & Changizi, K. (2004). Global optimization of cerebral cortex layout. Proc Natl Acad Sci U S A, 101(4), 1081-1086.

Durbin, R., & Mitchison, G. (1990). A dimension reduction framework for understanding cortical maps. Nature, 343, 644-647.

Engel, S. A., Rumelhart, D. E., Wandell, B. A., Lee, A. T., Glover, G. H., Chichilnisky, E.-J., & Shadlen, M. N. (1994). fMRI of human visual cortex. Nature, 369, 525.

Grinvald, A., Bonhoeffer, T., Malonek, D., Shoham, D., Bartfeld, E., Arieli, A., Hildesheim, R., & Ratzlaff, E. (1991). Optical Imaging of Architecture and Function in the Living Brain. In L. R. Squire, N. M. Weinberg, G. Lynch, & J. L.

McGaugh (Eds.), Memory: Organization & Locus of Change, (pp. 49-85): Oxford University Press.

Horton, J. C., & Adams, D. L. (2005). The cortical column: a structure without a function. Philos Trans R Soc Lond B Biol Sci, 360(1456), 837-862.

Kohonen, T. (1984). Self-Organization and Associative Memory . Berlin, New York: Springer-Verlag.

Maldonado, P. E., Godecke, I., Gray, C. M., & Bonhoeffer, T. (1997). Orientation selectivity in pinwheel centers in cat striate cortex. Science, 276(5318), 1551-1555.

Sajda P, F Han (2003) Perceptual salience as novelty detection in cortical pinwheel space. Neural Engineering, Conference Proceedings. ieeexplore.ieee.org

Schwartz, E. L. (1977). Spatial mapping in the primate sensory projection: analytic structure and relevance to perception. Biol Cybern, 25(4), 181-194.

Shmuel, A., & Grinvald, A. (2000). Coexistence of linear zones and pinwheels within orientation maps in cat visual cortex. Proc Natl Acad Sci U S A, 97(10), 5568-5573.

Swindale, N. V., Shoham, D., Grinvald, A., Bonhoeffer, T., & Hubener, M. (2000). Visual cortex maps are optimized for uniform coverage [see comments]. Nat Neurosci, 3(8), 822-826.

Swindale, N. V. (2000). How many maps are there in visual cortex? Cereb Cortex, 10(7), 633-643.

Talavage, T. M., Sereno, M. I., Melcher, J. R., Ledden, P. J., Rosen, B. R., & Dale, A. M. (2004). Tonotopic organization in human auditory cortex revealed by progressions of frequency sensitivity. J Neurophysiol, 91(3), 1282-1296.

Tanaka, K. (1996). Inferotemporal cortex and object vision. Annual Review of Neuroscience, 19, 109-139.

Tanaka, K. (2003). Columns for complex visual object features in the inferotemporal cortex: clustering of cells with similar but slightly different stimulus selectivities. *Cereb Cortex, 13*(1), 90-99.

Ts'o, D. Y., Frostig, R. D., Lieke, E. E., & Grinvald, A. (1990, 27 July 1990). Functional Organization of Primate Visual Cortex Revealed by High Resolution Optical Imaging. Science, 249, 417-420.

Young M.P. "Objective analysis of the topological organization of the primate cortical visual system" Nature 358: 152-155, 1992.

Wandell, B. A., Brewer, A. A., & Dougherty, R. F. (2005). Visual field map clusters in human cortex. Philos Trans R Soc Lond B Biol Sci, 360(1456), 693-707.