

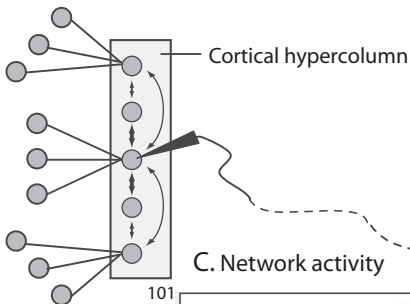
Neuroinformatics

May 4, 2017

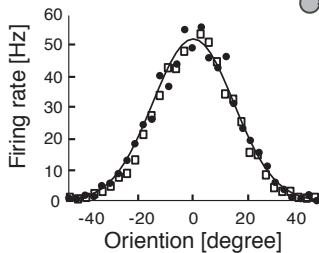
Lecture 11: Dynamical neural fields (DNF)

Motivation for SOM and DNF - Tuning Curves

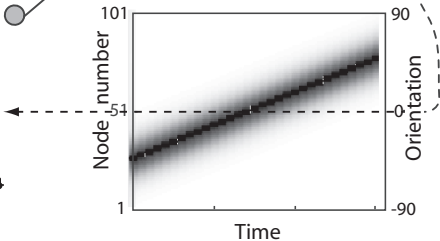
A. Model of a hypercolumn



B. Tuning curves



C. Network activity



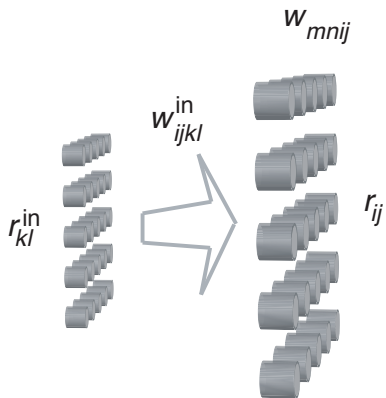
Self-organizing maps (SOMs)

- ▶ The development of SOM as a neural model is motivated by the topographical nature of cortical maps.
- ▶ Visual, tactile, and acoustic inputs are mapped in a topographical manner. Visual: retinotopy (position in visual field), orientation, spatial frequency, direction, ocular dominance, etc. Tactile: somatotopy (position on skin, thumb and SMS) Acoustic: tonotopy (frequency)
- ▶ Self-organizing maps (SOM) is based on competitive learning, where output neurons compete with each other to be activated (Kohonen, 1982)
- ▶ The output neuron that activates is called the winner-takes-all neuron
- ▶ Lateral inhibition is one way to implement competition for map formation (von der Malsburg 1973)
- ▶ In SOM, neurons are placed on a lattice, on which a meaningful coordinate system for different features is created (feature map).
- ▶ The lattice thus forms a topographic map where the spatial location on the lattice is indicative of the input features.

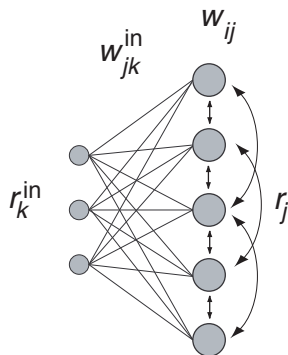
SOM - von der Malsburg 1973

Willshaw - von der Malsburg SOM

A. 2D feature space and SOM layer



B. 1D feature space and SOM layer



Network equations

Update rule of (recurrent) cortical network:

$$\tau \frac{du_i(t)}{dt} = -u_i(t) + \frac{1}{N} \sum_j w_{ij} r_j(t) + \frac{1}{M} \sum_k w_{ik}^{\text{in}} r_k^{\text{in}}(t)$$

Activation function: $r_j(t) = \frac{1}{1 + e^{\beta(u_j(t) - \alpha)}}$

Lateral weight matrix: $w_{ij} \propto r_i r_j$

$$= A_w \left(e^{-((i-j)*\Delta x)^2 / 2\sigma^2} - C \right)$$

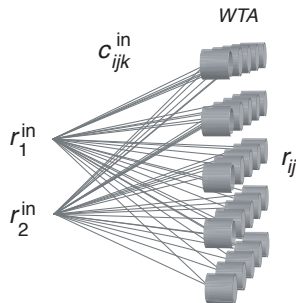
Input weight matrix: $w_{ij}^{\text{in}} \propto r_i r_j^{\text{in}}$

Kohonen - Shortcut

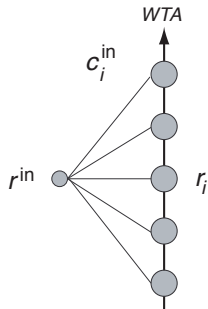
- ▶ Willshaw-von der Malsburg model: input neurons arranged in 2D lattice, output in 2D lattice. Lateral excitation/inhibition (Mexican hat) gives rise to soft competition. Normalized Hebbian learning. Biological motivation.
- ▶ Kohonen model: input of any dimension, output neurons in 1D, 2D, or 3D lattice. Relaxed winner-takes-all (neighborhood). Competitive learning rule. Computational motivation.

Kohonen SOM

A. 2-d feature space and SOM layer



B. 1-d feature space and SOM layer



Kohonen model

- ▶ cortical sheet activation, σ_r^2 width of activated area, activation fce resembles tuning curves, radial-basis networks

$$r_{ij} = \exp\left(-\sum_k (c_{ijk} - r_k^{in})^2 / 2\sigma_r^2\right)$$

- ▶ strength connection around the winning node r_{ij}^* , WTA rule - winner takes all

$$\Delta c_{ijk} = \epsilon r_{ij}^* (r_{in} - c_{ijk})$$

- ▶ ML approach (Matlab implementation):
 $w^i(q) = w^i(q-1) + \alpha(p(q) - w^i(q))$, i are lying in neighborhood
 $N(i)_d = \{j, d_{ij} < d\}$

SOM Algorithm

1. Randomly initialize weight vectors w_i
2. Randomly sample input vector x
3. Find Best Matching Unit (BMU)

$$i(x) = \arg \min_j ||x - w_j||$$

4. Update weight vectors, where $h(j, i(x))$ is neighborhood function of BMU

$$w_j = w_j + \epsilon h(j, i(x))(x - w_j)$$

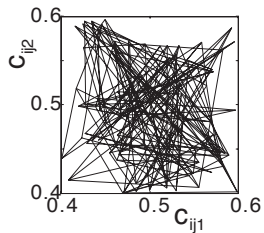
5. Repeat steps 2-4

som.m

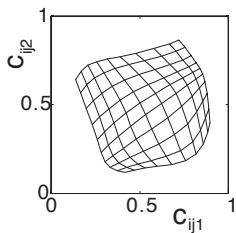
```
1 %% Two dimensional self-organizing feature map ala Kohonen
2 clear; nn=10; lambda=0.2; sig=2; sig2=1/(2*sig^2);
3 [X,Y]=meshgrid(1:nn,1:nn); ntrial=0;
4
5 % Initial centres of preferred features:
6 c1=0.5-.1*(2*rand(nn)-1);
7 c2=0.5-.1*(2*rand(nn)-1);
8
9 %% training session
10 while(true)
11     if(mod(ntrial,100)==0) % Plot grid of feature centres
12         clf; hold on; axis square; axis([0 1 0 1]);
13         plot(c1,c2,'k'); plot(c1',c2', 'k');
14         tstring=[int2str(ntrial) ' examples']; title(tstring);
15         waitforbuttonpress;
16     end
17     r_in=[rand;rand];
18     r=exp(-(c1-r_in(1)).^2-(c2-r_in(2)).^2);
19     [rmax,x_winner]=max(max(r)); [rmax,y_winner]=max(max(r'));
20     r=exp(-(X-x_winner).^2+(Y-y_winner).^2)*sig2);
21     c1=c1+lambda*r.*(r_in(1)-c1);
22     c2=c2+lambda*r.*(r_in(2)-c2);
23     ntrial=ntrial+1;
24 end
```

SOM simulation

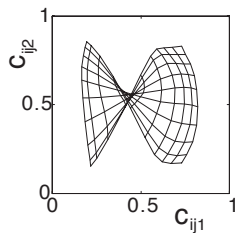
A. Initial random centres



B. After 1000 training steps

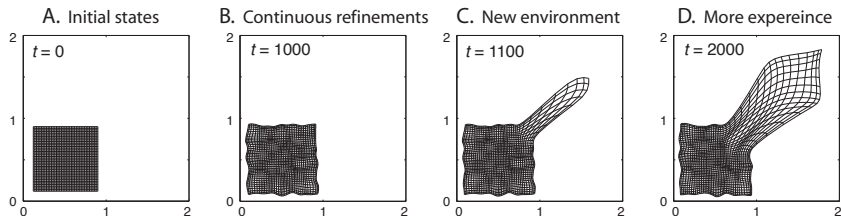


C. Topographical defect



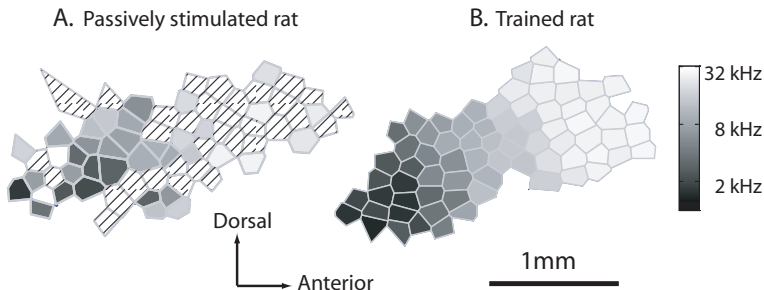
Another example

- ▶ Simulating development processes
- ▶ SOM can represent new domains, representation less fine-grained compared to initial domain
- ▶ Early in life exposed to broad feature space (learning languages)



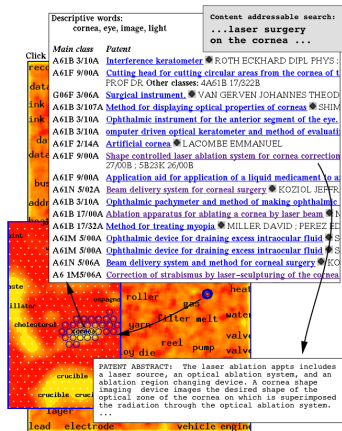
Representational plasticity - Zhou and Merzenich, PNAS 2007

- ▶ rat pups raised in noisy environment ← severely impaired tonotopicity (tones representations) in primary auditory cortex - A1
- ▶ no recovery after stimulation with sounds of different frequencies
- ▶ stimulation by discrimination task with food reward ← rats were able to recover tonotopic maps
- ▶ traditionally SOM maps are driven by data: bottom - up approach
- ▶ top-down processing explains those experimental results (reinforcement learning)



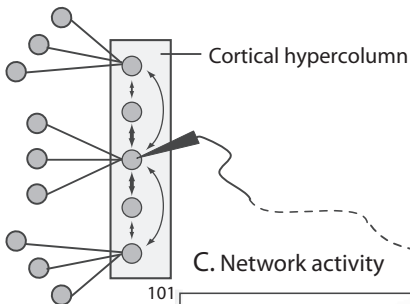
WEBSOM - Self-Organizing Maps for Internet Exploration

- ▶ find information on laser surgery on the cornea of eye, <http://websom.hut.fi>
- ▶ best matching locations marked with circles
- ▶ sparse feature vector, each row representing single document, each term relative frequency of predefined entries (e.g. 50 000 words)

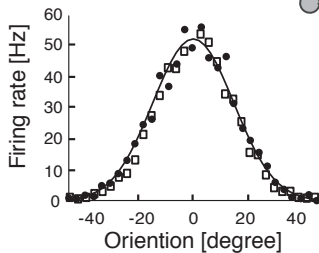


Tuning Curves

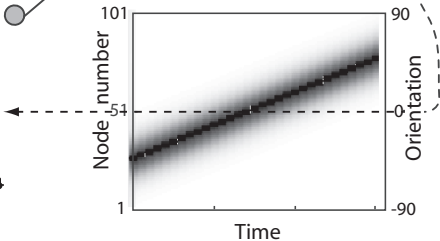
A. Model of a hypercolumn



B. Tuning curves



C. Network activity



Dynamic Neural Field Theory

Field dynamics:

$$\tau \frac{\partial \mathbf{u}(\mathbf{x}, t)}{\partial t} = -\mathbf{u}(\mathbf{x}, t) + \int_{\mathbf{y}} \mathbf{w}(\mathbf{x}, \mathbf{y}) \mathbf{r}(\mathbf{y}, t) d\mathbf{y} + I^{\text{ext}}(\mathbf{x}, t)$$

$$\mathbf{r}(\mathbf{x}, t) = g(\mathbf{u}(\mathbf{x}, t)),$$

Continuous version of equations above with discretization:

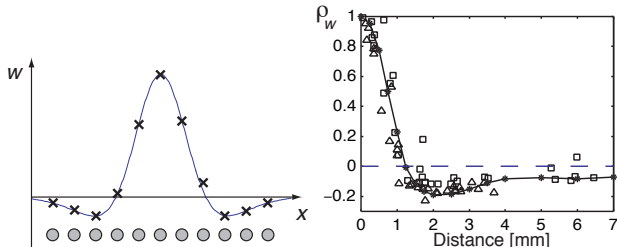
$$x \rightarrow i\Delta x \quad \text{and} \quad \int dx \rightarrow \Delta x \sum$$

Main assumption: Short-distance excitation and long-distance inhibition

Learning in cortical sheet - Lateral weight kernel

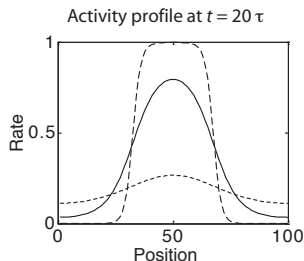
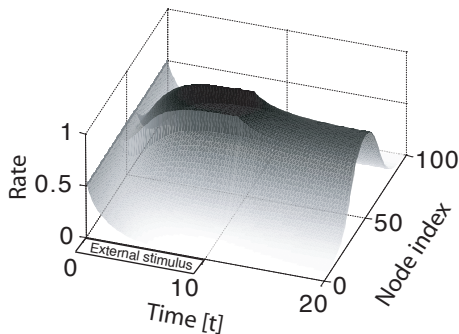
$$\begin{aligned}r(x - x^p) &= e^{(x-x^p)^2/2\sigma_r^2} \\|x - x^p| &= \min(|x - x^p|, 2\pi - |x - x^p|) \\ \mathbf{w}^E(|x - y|) &= \int_0^{2\pi} r(x - x^p)r(y - x^p)dx^p \\ \mathbf{w}^E(|x - y|) &= A_w e^{-(x-y)^2/4\sigma_r^2} \\ \mathbf{w}(|x - y|) &= A_w e^{-(x-y)^2/4\sigma_r^2} - C \\ \mathbf{w}_{ij} \propto r_i r_j &= A_w \left(e^{-((i-j)*\Delta x)^2/2\sigma^2} - C \right)\end{aligned}\quad (1)$$

Can be learned from Gaussian response curves of individual nodes



Self-sustained activity packet

- ▶ growing activity: $C \ll E$, whole map is active, undesirable
- ▶ decaying activity: $C \gg E$, decaying after removal of external input
- ▶ memory activity: stability even when external input is removed !
- ▶ simulation: string external stimulus: nodes 40-50, excitatory weights to nearby nodes, active nodes: activity packets, bubble or bump \leftarrow continuous attractor neural networks \leftarrow working memory, $A_w = 4, C = 0.5$



dnf.m

```
1 %% Dynamic Neural Field Model (1D)
2 clear; clf; hold on;
3 nn = 100; dx=2*pi/nn; sig = 2*pi/10; C=0.5;
4
5 %% Training weight matrix
6 for loc=1:nn;
7     i=(1:nn)'; dis= min(abs(i-loc),nn-abs(i-loc));
8     pat(:,loc)=exp(-(dis*dx).^2/(2*sig^2));
9 end
10 w=pat*pat'; w=w/w(1,1); w=4*(w-C);
11 %% Update with localised input
12 tall = []; rall = [];
13 I_ext=zeros(nn,1); I_ext(nn/2-floor(nn/10):nn/2+floor(nn/10))=1;
14 [t,u]=ode45('rnn_ode',[0 10],zeros(1,nn),[],nn,dx,w,I_ext);
15 r=1./(1+exp(-u)); tall=[tall;t]; rall=[rall;r];
16 %% Update without input
17 I_ext=zeros(nn,1);
18 [t,u]=ode45('rnn_ode',[10 20],u(size(u,1),:),[],nn,dx,w,I_ext);
19 r=1./(1+exp(-u)); tall=[tall;t]; rall=[rall;r];
20 %% Plotting results
21 surf(tall',1:nn,rall','linestyle','none'); view(0,90);
```

```
1 function udot=rnn_ode(t,u,flag,nn,dx,w,I_ext)
2 % odefile for recurrent network
3 tau_inv = 1.; % inverse of membrane time constant
4 r=1./(1+exp(-u));
5 sum=w*r*dx;
6 udot=tau_inv*(-u+sum+I_ext);
7 return
```

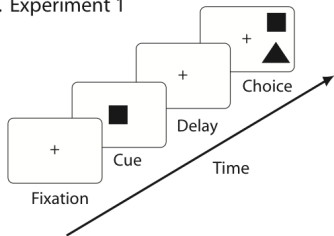
Update rule of (recurrent) cortical network:

$$\tau \frac{du_i(t)}{dt} = -u_i(t) + \frac{1}{N} \sum_j w_{ij} r_j(t) + \frac{1}{M} \sum_k w_{ik}^{\text{in}} r_k^{\text{in}}(t)$$

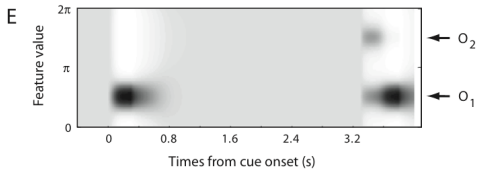
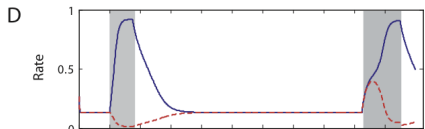
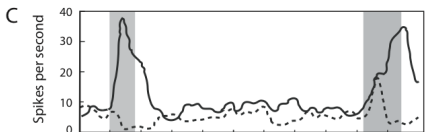
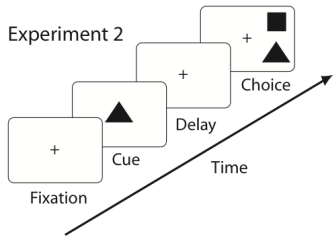
Activation function: $r_j(t) = \frac{1}{1+e^{\beta(u_j(t)-\alpha)}}$.

DNF example - Chelazzi, Nature, 1993

A. Experiment 1



B. Experiment 2



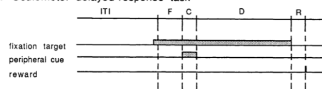
DNF example - Chelazzi, Nature, 1993, Matlab code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 1-d Continuous Attractor Neural Network with Hebbian learning
% two gaussian signal: decision network
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear; close all;
nn = 100; dx=2*pi/nn; % number of nodes and resolution in deg
%weight matrices
sig = 2*pi/20;
w_sym=hebb(nn,sig,dx);
w_inh=0.07;%use 0.04, 7,6,3; 3*(sqrt(2*pi)*sig)^2/nn;
w=500*(w_sym-w_inh);
%inputs
perc=0.01; Is=11;
Ia=(1+0.5*perc)*Is;
Ib=(1-0.5*perc)*Is;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Experiment %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
param=0;
%%% no external input
u0 = zeros(nn,1)-10;
I_ext=zeros(nn,1);
tspan=[0,40];
[t,u]=ode45('rnn_ode_u',tspan,u0,[],nn,dx,w,I_ext);
r=f1(u);
%%% external cue
u0 = u(size(t,1),:);
I_ext=zeros(nn,1);
loc1=pi/2;%+pi/16;
loc2=3*pi/2;%-pi/16;
I_ext=I_ext+in_signal_pbc(loc1,Is,sqrt(2)*sig,nn,dx);
tspan=[40 70];
[t2,u]=ode45('rnn_ode_u',tspan,u0,[],nn,dx,w,I_ext);
r=[r;f1(u)];
t=[t;t2];
%%% no external input
u0 = u(size(t2,1),:);
I_ext=zeros(nn,1);
param=0;
tspan=[70,370];
[t2,u]=ode45('rnn_ode_u',tspan,u0,[],nn,dx,w,I_ext);
r=[r;f1(u)];
t=[t;t2];
```

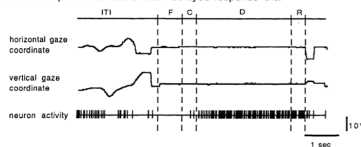
Working memory by ongoing firing - sustained DNF bubble

- ▶ F- fixation period (0.75s), C-cue period (0.5s), D - delay period (3-6 s), R - response period (0.5s) → reward

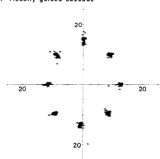
A Oculomotor delayed-response task



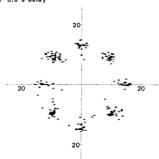
B An example of an oculomotor delayed-response trial



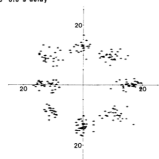
A Visually guided saccade



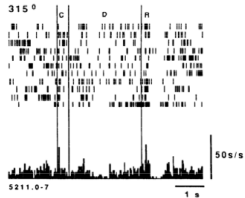
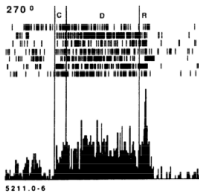
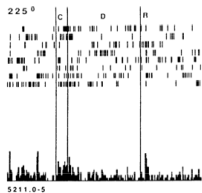
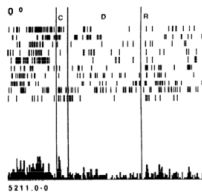
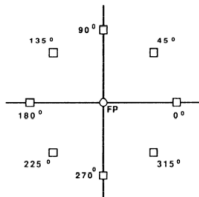
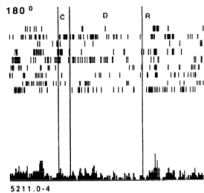
B 3.0 s delay



C 6.0 s delay



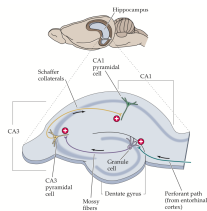
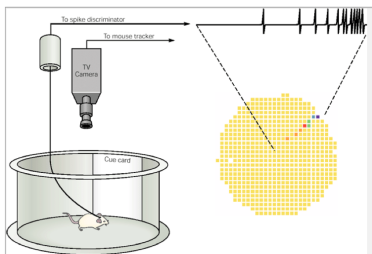
Directional delay period activity



S. Funahashi, C.J. Bruce and P.S. Goldman-Rakic, Mnemonic coding of visual space in the monkeys dorsolateral prefrontal cortex, *J Neurophysiol* 61:331349, 1989

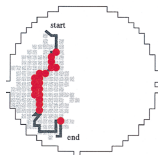
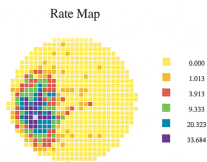
Place cells

- ▶ Place cells are neurons in the hippocampus that exhibit a high rate of firing whenever an animal is in a specific location (pyramidal cells in CA1, CA4)
- ▶ On initial exposure to a new environment, place fields become established within minutes. The place fields of cells tend to be stable over repeated exposures to the same environment.
- ▶ Remapping - In a different environment, however, a cell may have a completely different place field or no place field at all

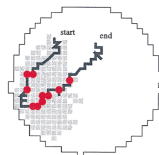


Place cells - 16 mins experiment

- ▶ colored circular region is an overhead view of a 76 cm diameter cylinder, each small square region (pixel) is about 2.5 cm squared, firing rate \rightarrow total number of spikes fired in the pixel divided by the total time spent in the pixel.
- ▶ hungry rat ran around for 16 min chasing small food pellets, the black line indicates the rat's path and the red dots the locations at which action potentials were fired, action potentials were fired all along the second path even though the rat turned and ran out of the field in the direction opposite to its entry; this is an indication that the firing is not directionally selective.
- ▶ <http://www.youtube.com/watch?v=PGHRDcPKio8>



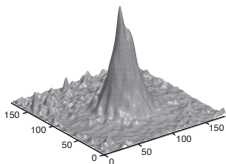
Path 1



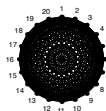
Path 2

Place cells - is there any topography ?

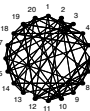
- ▶ no specific topography found with respect to neuron's maximal response to a particular place
- ▶ rearranging plot - neurons firing maximally in response to adjacent location → plot neurons adjacent to each other
- ▶ direction head cells → recurrent AAN simulation, high dimensionality - (i) before learning - equal weights for all nodes (ii) training- each node assigned (Gaussian profile) to preferred direction where fires maximally, competitive Hebb's rule (iii) strongly connected nodes adjacent to each other
- ▶ dimensionality was reduced to 1D model, networks self-organized to reflect the dimensionality of feature space



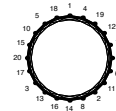
A. Fully connected



B. After learning



C. After learning (reordered)



Topographica - general simulator for cortical maps

<http://topographica.org/>

The screenshot displays the Topographica software interface, which is used for simulating cortical maps. The interface is organized into several panels:

- Loss Console:** Located at the top left, it contains a menu with options like "Quit", "Reset network", and "Reload saved network". It also shows the current command and iteration count.
- Test pattern parameters:** A central control panel with various sliders and input fields for parameters such as theta, phi, xi, and sigma. It includes a "Photograph" section for loading and displaying test images.
- Activity 3, Activity 4, Activity 2, Activity 5:** These panels show the results of simulations for different activity types. Each panel contains four sub-panels: "Eye0 Activity", "Gangliao0 Activity", "Primary InputResponse", and "Primary Activity". Below these sub-panels are control buttons: "Refresh", "Reduce", "Enlarge", and "Auto-refresh".
- Weights Array 1, Weights Array 2, Weights 1:** These panels display weight matrices. "Weights Array 1" and "Weights Array 2" show 2D grids of weights, while "Weights 1" shows a single weight matrix. Each panel has "Refresh", "Reduce", "Enlarge", and "Auto-refresh" buttons.
- Orientation 1:** This panel shows orientation selectivity maps, including a "Primary OrientationSelectivity" map and a "Primary Afferent00" map. It also includes "Refresh", "Reduce", "Enlarge", and "Auto-refresh" buttons.
- Primary LateralInhibitory:** This panel shows a map of primary lateral inhibition, with "Refresh", "Reduce", "Enlarge", and "Auto-refresh" buttons.
- Region and Name fields:** These fields allow users to specify the region and name of the current simulation.
- Skip and Situate buttons:** These buttons are used to skip or situate the current simulation.

Further Readings

- Teuvo Kohonen (1989), **Self-organization and associative memory**, Springer Verlag, 3rd edition.
- David J. Willshaw and Christoph von der Malsburg (1976), **How patterned neural connexions can be set up by self-organisation**, in **Proc Roy Soc B** 194, 431–445.
- Shun-ichi Amari (1977), **Dynamic pattern formation in lateral-inhibition type neural fields**, in **Biological Cybernetics** 27: 77–87.
- Huge R. Wilson and Jack D. Cowan (1973), **A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue**, in **Kybernetik** 13:55-80.
- Kechen Zhang (1996), **Representation of spatial orientation by the intrinsic dynamics of the head-direction cell ensemble: A theory**, in **Journal of Neuroscience** 16: 2112–2126.
- Simon M. Stringer, Thomas P. Trappenberg, Edmund T. Rolls, and Ivan E.T. de Araujo (2002), **Self-organizing continuous attractor networks and path integration I: One-dimensional models of head direction cells**, in **Network: Computation in Neural Systems** 13:217–242.
- Alexandre Pouget, Richard S. Zemel, and Peter Dayan (2000), **Information processing with population codes**, in **Nature Review Neuroscience** 1:125–132.
- Miikkulainen R., **Computational Maps in the Visual Cortex**, Springer, 2005