# Machine Intelligence 2

## 4.3 Self-Organising Maps

Prof. Dr. Klaus Obermayer

Fachgebiet Neuronale Informationsverarbeitung (NI)

SS 2017

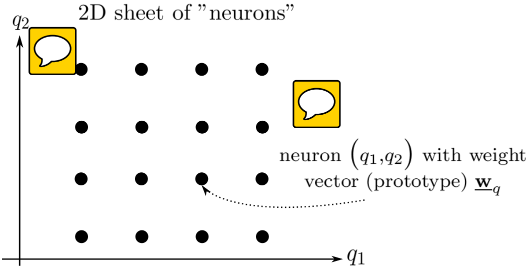
# Self-Organising Maps

# Self-organising maps (SOM)

- clustering & local embedding of vectorial data: $\underline{\mathbf{x}}^{(\alpha)}, \alpha = 1, \ldots, p$
- clustering of data based on similarity: squared Euclidean distance
- low-dimensional and **neighborhood preserving** representation for visualization

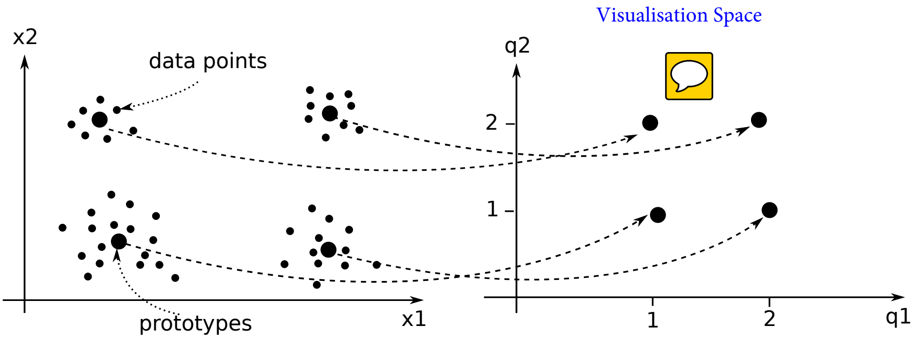
# Model class

$$m_{\underline{\mathbf{q}}}^{(\alpha)} = \begin{cases} 1, & \text{if } \underline{\mathbf{q}} = \operatorname{argmin}_{\underline{\mathbf{r}}} \left| \underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{w}}_{\underline{\mathbf{r}}} \right| \\ 0, & \text{else} \end{cases}$$




"Binary neurons" with index $\underline{\mathbf{q}} = (q_1, q_2)^T$ are assigned to prototypes $\underline{\mathbf{w}}_{\underline{\mathbf{q}}}$.

# "Topographic" maps




"map" of data space for $\begin{cases} \text{visualization} \\ \text{dimension reduction} \\ \text{preprocessing for prediction} \end{cases}$

**Algorithm 1:** On-line learning for SOM "almost kmeans clustering"

**Initialization:**

- choose no. $M$ of partitions ("neurons")
- choose annealing schedule for $\varepsilon$ and $\sigma$
- initialize prototypes, $\underline{\mathbf{w}}_{\underline{\mathbf{q}}} = 1/p \sum_\alpha \underline{\mathbf{x}}^{(\alpha)} + \underline{\eta}$, $\underline{\eta}$ small noise vector

**begin**

choose a data point $\underline{\mathbf{x}}^{(\alpha)}$

determine the closest prototype:

$$\underline{\mathbf{p}} = \operatorname*{argmin}_{\underline{\mathbf{r}}} \left| \underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{w}}_{\underline{\mathbf{r}}} \right|$$

change all prototypes according $\underline{\mathbf{w}}_q$ to:

$$\Delta \underline{\mathbf{w}}_{\underline{\mathbf{q}}} = \varepsilon \, h_{\underline{\mathbf{qp}}} \left( \underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{w}}_{\underline{\mathbf{q}}} \right) \qquad \text{for all } \underline{\mathbf{q}}$$

**end**

# Neighborhood function $h_{\underline{\mathbf{q}}\underline{\mathbf{p}}}$

$$\Delta \underline{\mathbf{w}}_{\underline{\mathbf{q}}} = \varepsilon \, h_{\underline{\mathbf{q}}\underline{\mathbf{p}}} \left( \underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{w}}_{\underline{\mathbf{q}}}^{\mathrm{old}} \right) \qquad \text{for all q}$$

- $h_{\underline{\mathbf{q}}\underline{\mathbf{p}}}$ enforces similar learning steps for neighboring neurons
- common choice:

$$h_{\underline{\mathbf{q}}\underline{\mathbf{p}}} = \exp \left\{ - \frac{(\underline{\mathbf{q}} - \underline{\mathbf{p}})^2}{2\sigma^2} \right\}$$

- $\sigma = 0$: standard on-line K-means only (update $\underline{\mathbf{q}} = \underline{\mathbf{p}}$)

# Neighborhood function $h_{\underline{\mathbf{qp}}}$

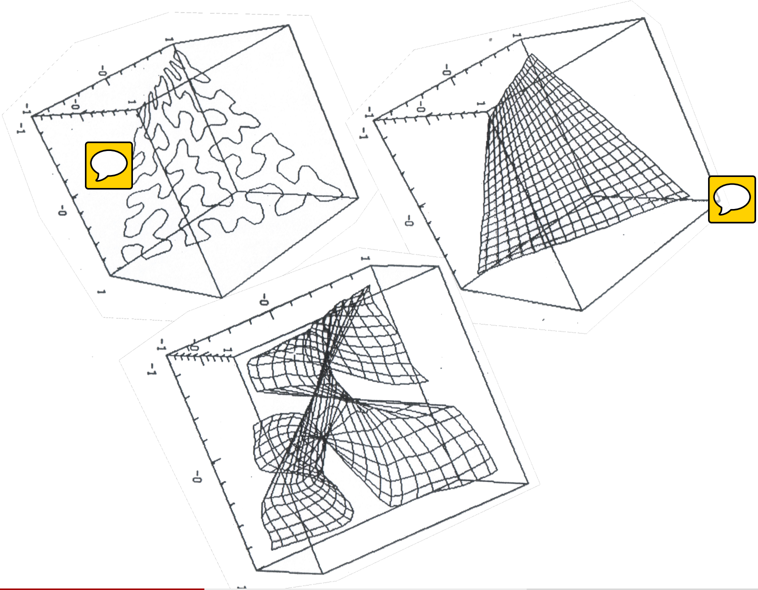
$$h_{\underline{\mathbf{qp}}} = \exp\left\{ - \frac{(\underline{\mathbf{q}} - \underline{\mathbf{p}})^2}{2\sigma^2} \right\}$$

$\sigma$ *annealing*

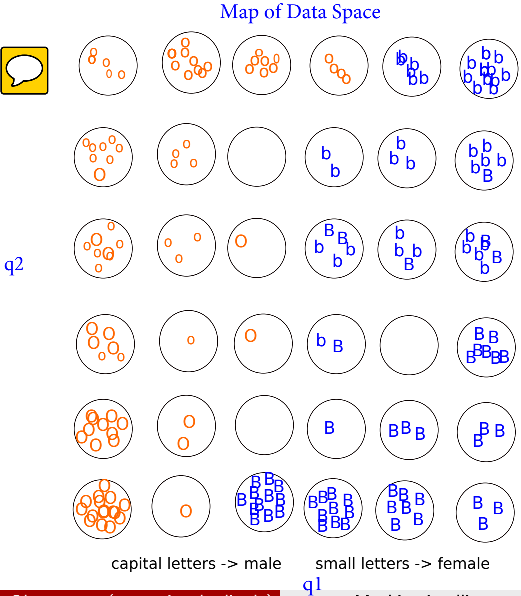
- $\rightarrow$ start with large $\sigma$ ($\rightsquigarrow$ neighborhood function convex over its support)
- $\rightarrow$ decrease linearly or exponentially (but "slow") during learning.
- $\rightarrow$ solution depends on final value of $\sigma$

- $\rightarrow$ $\sigma = 0$: minimum of the K-means clustering cost function but *"neighborhood preserving"*
- $\rightarrow$ $\sigma$ small : better representation capabilities at the expense of a non-optimal clustering cost

# 2d manifold example



# Leptograpsus example




## attributes

$X_1$: frontal lobe size (mm)

$X_2$: rear width (mm)

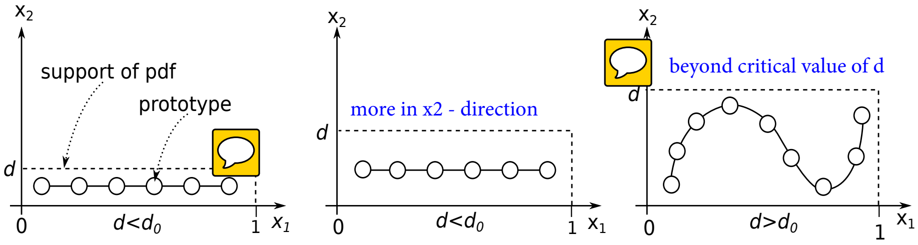
$X_3$: carapace length (mm)

$X_4$: carapace width (mm)

$X_5$: body depth (mm)

# Dimension reducing mappings

what if dimensions dont match?




one feature: x1

# Semantic maps

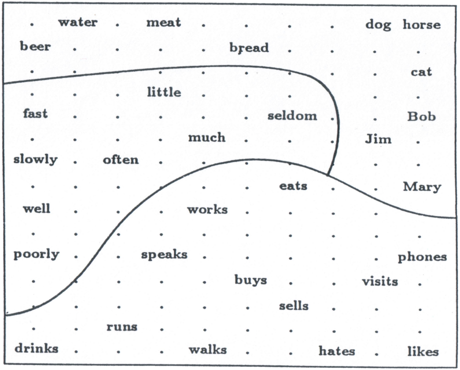
## Abstract Data: Verbal Statements

| | |
|---|---|
| Bob/Jim/Mary | 1 |
| horse/dog/cat | 2 |
| beer/water | 3 |
| meat/bread | 4 |
| runs/walks | 5 |
| works/speaks | 6 |
| visits/phones | 7 |
| buys/sells | 8 |
| likes/hates | 9 |
| drinks/eats | 10 |
| much/little | 11 |
| fast/slowly | 12 |
| often/seldom | 13 |
| well/poorly | 14 |

Sentence Patterns:

| | | |
|---|---|---|
| 1-5-12 | 1-9-2 | 2-5-14 |
| 1-5-13 | 1-9-3 | 2-9-1 |
| 1-5-14 | 1-9-4 | 2-9-2 |
| 1-6-12 | 1-10-3 | 2-9-3 |
| 1-6-13 | 1-11-4 | 2-9-4 |
| 1-6-14 | 1-10-12 | 2-10-3 |
| 1-6-15 | 1-10-13 | 2-10-12 |
| 1-7-14 | 1-10-14 | 2-10-13 |
| 1-8-12 | 1-11-12 | 2-10-14 |
| 1-8-2 | 1-11-13 | 1-11-4 |
| 1-8-3 | 1-11-14 | 1-11-12 |
| 1-8-4 | 2-5-12 | 2-11-13 |
| 1-9-1 | 2-5-13 | 2-11-14 |

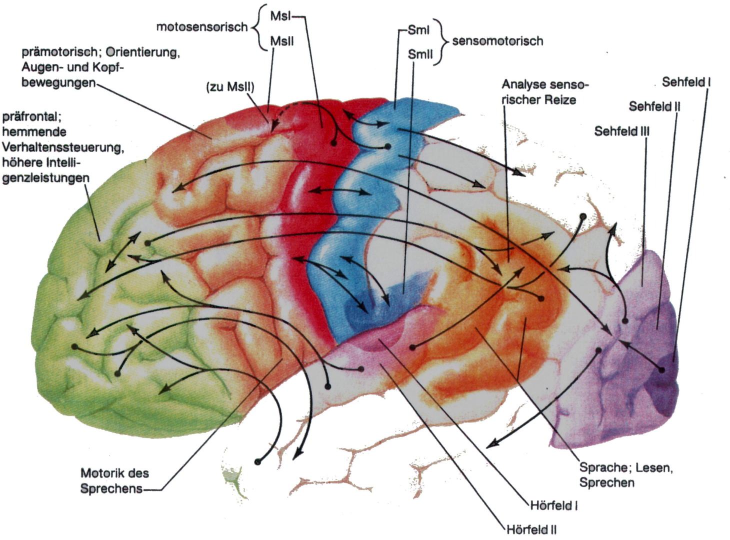
Mary likes meat
Jim speaks well
Mary likes Jim
Jim eats often
Mary buys meat
dog drinks fast
horse hates meat
Jim eats seldom
Bob buys meat
cat walks slowly
Jim eats bread
cat hates Jim
Bob sells beer
(etc.)




$$\begin{bmatrix} \vec{V}_S \\ \vec{V}_C \end{bmatrix} = \begin{bmatrix} \vec{0} \\ \vec{V}_C \end{bmatrix} + \begin{bmatrix} \vec{V}_S \\ \vec{0} \end{bmatrix}$$

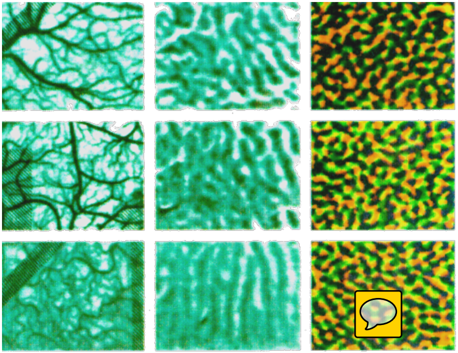
Ritter & Kohnen 1989

# Orientation & ocular dominance maps



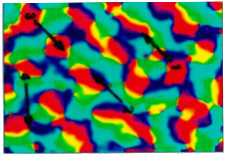
# Orientation & ocular dominance maps

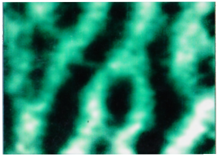
maps of feature vectors



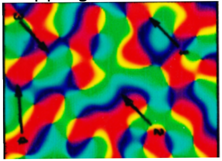
# Orientation & ocular dominance maps

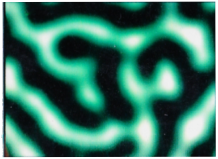
Experimental data:





mapping results:





5d feature vector

position: $x, y \in [0, d]$

orientation preference & specificity:

$r\cos(2\phi)$, $\phi \in [0, \pi[$

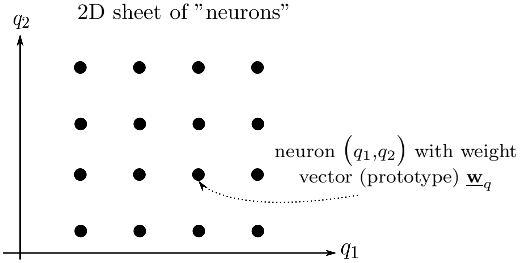
$r\sin(2\phi)$, $\phi \in [0, \pi[$

ocular dominance: $z \in \{-1, 1\}$

# Self-organizing maps for pairwise data

- distance matrix $\underline{\mathbf{D}}$ specifying the distances or 'dissimilarities' $d_{\alpha\alpha'}$ between $p$ "objects" $\alpha = 1, \ldots, p$
- set of $M$ clusters (partitions) $\underline{\mathbf{q}}$ with a geometrical structure (e.g. 1-d line or 2-d grid).
- binary assignment variables (normalized):

$$m_{\underline{\mathbf{q}}}^{(\alpha)} = \begin{cases} 1, & \text{if object } \alpha \text{ belongs to cluster } \underline{\mathbf{q}} \\ 0, & \text{else} \end{cases}$$




| | 1 | 2 | 3 | | $p$ |
|---|---|---|---|---|---|
| 1 | 0 | 1.7 | 0.99 | | 3.0 |
| 2 | 1.7 | 0 | 0.3 | $\ldots$ | 0.1 |
| 3 | 0.9 | 0.3 | 0 | | 0.2 |
| $\vdots$ | | $\vdots$ | | $\ddots$ | $\vdots$ |
| $p$ | 3.0 | 0.1 | 0.2 | $\ldots$ | 0 |

relational representation
"pairwise data"

# Cost function & model selection

$$E_{\left[\left\{m_{\underline{\mathbf{q}}}^{(\alpha)}\right\}\right]} = \frac{1}{p}\sum_{\underline{\mathbf{r}}} \frac{\sum\limits_{\alpha,\alpha'} \left(\sum\limits_{\underline{\mathbf{q}}} h_{\underline{\mathbf{r}}\underline{\mathbf{q}}} m_{\underline{\mathbf{q}}}^{(\alpha)}\right)\left(\sum\limits_{\underline{\mathbf{q}}} h_{\underline{\mathbf{r}}\underline{\mathbf{q}}} m_{\underline{\mathbf{q}}}^{(\alpha')}\right) d_{\alpha\alpha'}}{\sum\limits_{\alpha}\left(\sum\limits_{\underline{\mathbf{q}}} h_{\underline{\mathbf{r}}\underline{\mathbf{q}}} m_{\underline{\mathbf{q}}}^{(\alpha)}\right)} \overset{!}{=} \min$$

- $m_q^{(\alpha)} \rightarrow \sum\limits_{\underline{\mathbf{q}}} h_{\underline{\mathbf{r}}\underline{\mathbf{q}}} m_{\underline{\mathbf{q}}}^{(\alpha)}$
- "neighboring" clusters (w.r.t. $h_{\underline{\mathbf{r}}\underline{\mathbf{q}}}$) contribute to the total average distance
- "neighborhood preserving maps" induce lower cost

**Algorithm 2:** SOM for pairwise data (mean-field approximation)

**Initialization:**

- choose no. $M$ of partitions, initial $(\beta_0)$ and final $(\beta_f)$ noise parameters, annealing factor $\eta$, width $\sigma$ of neighborhood function $h_{\underline{\mathbf{s}}\underline{\mathbf{q}}}$, tolerance $\theta$
- initialize mean-fields $e_{\underline{\mathbf{q}}}^{(\alpha)}$ : random numbers $\in [0,1]$

$\beta \leftarrow \beta_0$

**while** $\beta < \beta_f$ **do** annealing

  **repeat** EM

    compute assignment probabilities $\left\langle m_{\underline{\mathbf{q}}}^{(\alpha)} \right\rangle_Q = \dfrac{\exp\left\{-\beta\left(e_{\underline{\mathbf{q}}}^{(\alpha)}\right)_{\text{old}}\right\}}{\sum_{\underline{\mathbf{r}}} \exp\left\{-\beta\left(e_{\underline{\mathbf{r}}}^{(\alpha)}\right)_{\text{old}}\right\}} \quad \forall \underline{\mathbf{q}}, \alpha$

    compute new mean-fields

$$\left(e_{\underline{\mathbf{q}}}^{(\alpha)}\right)_{\text{new}} = \frac{1}{p} \sum_{\underline{\mathbf{s}}} h_{\underline{\mathbf{s}}\underline{\mathbf{q}}} \left[ \frac{1}{\sum_{\gamma} \left( \sum_{\underline{\mathbf{r}}} h_{\underline{\mathbf{s}}\underline{\mathbf{r}}} \left\langle m_{\underline{\mathbf{r}}}^{(\gamma)} \right\rangle_Q \right)} \sum_{\delta} \left( \sum_{\underline{\mathbf{r}}} h_{\underline{\mathbf{s}}\underline{\mathbf{r}}} \left\langle m_{\underline{\mathbf{r}}}^{(\delta)} \right\rangle_Q \right. \right.$$

$$\left. \left. \cdot \left\{ d_{\delta\alpha} - \frac{1}{2} \frac{1}{\sum_{\gamma} \left( \sum_{\underline{\mathbf{r}}} h_{\underline{\mathbf{s}}\underline{\mathbf{r}}} \left\langle m_{\underline{\mathbf{r}}}^{(\gamma)} \right\rangle_Q \right)} \sum_{\varepsilon} \left( \sum_{\underline{\mathbf{r}}} h_{\underline{\mathbf{s}}\underline{\mathbf{r}}} \left\langle m_{\underline{\mathbf{r}}}^{(\varepsilon)} \right\rangle_Q \right) d_{\varepsilon\delta} \right\} \right] \quad \forall \underline{\mathbf{q}}, \alpha$$

  **until** $\left| \left(e_{\underline{\mathbf{q}}}^{(\alpha)}\right)_{\text{new}} - \left(e_{\underline{\mathbf{q}}}^{(\alpha)}\right)_{\text{old}} \right| < \theta \quad \forall \underline{\mathbf{q}}, \alpha$

  $\beta \leftarrow \eta\beta$

**end**

# Comments

- replacing $h_{\underline{\mathbf{sp}}}$ by $\delta_{\underline{\mathbf{sp}}}$ recovers standard pairwise clustering
- "Kohonen-approximation": replace $h_{\underline{\mathbf{sp}}}$ by $\delta_{\underline{\mathbf{sp}}}$ for the neighborhood function (only) at $\circledast$ in algorithm 2
  - $\rightarrow$ reduction of computational cost
  - $\rightarrow$ visualization properties remain
  - $\rightarrow$ original algorithm suggested by T. Kohonen is recovered for squared Euclidean distances $d_{\alpha\alpha'}$ and $\beta \rightarrow \infty$
- no need for $\sigma$ annealing

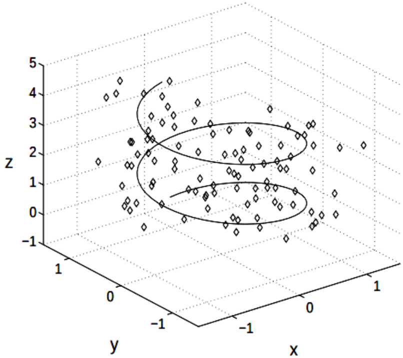
# Toy example: noisy spiral

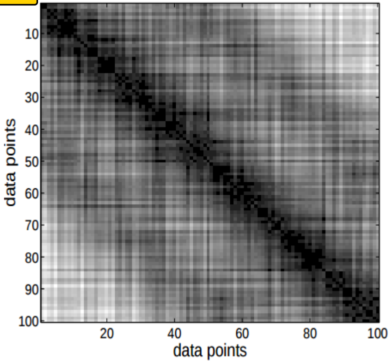
1000 data points:

$$x = \sin\theta + \eta_x$$
$$y = \cos\theta + \eta_y$$
$$z = \frac{\theta}{\pi} + \eta_z$$

$\theta \in [0, 4\pi]; \qquad \eta \backsim \mathcal{N}_{(0,0.3)}$

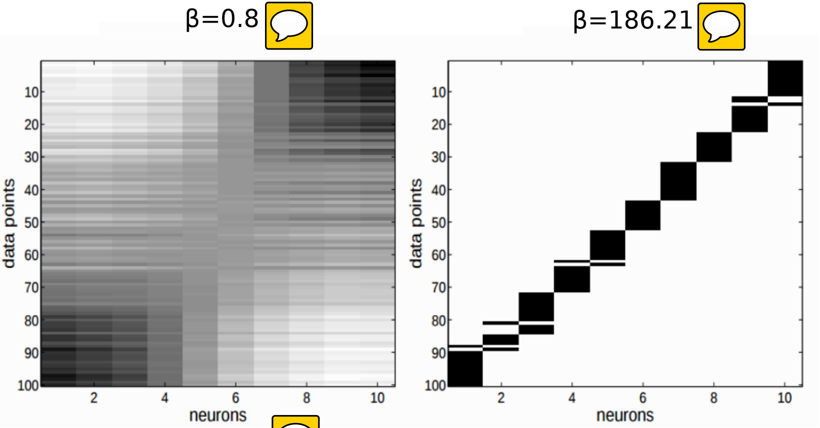
Euclidean squared distance



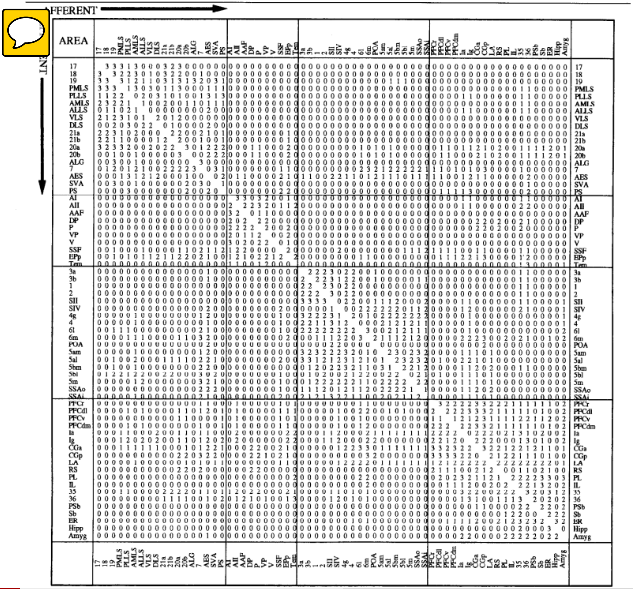



# Toy example: noisy spiral

- 10 neurons in 1D
- Gaussian neighborhood function, $\sigma = 0.5$



# Mapping cat's cerebral cortex



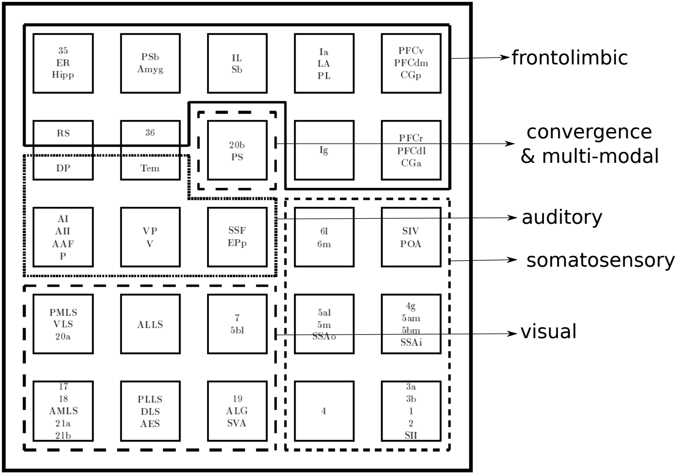

Preprocessing:

$$d_{(l,l')} \leftarrow 4 - d_{(l,l')}$$

diagonal: $d_{(l,l)} = 0$

# Mapping cat's cerebral cortex

"Kohonen"-map:
5x5 neurons,
Gaussian
neighborhood
$\delta_n = 0.4$



# Squared Euclidean distances

$$d_{\alpha\alpha'} = \frac{1}{2}\left(\underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{x}}^{(\alpha')}\right)^2$$

cost function:

$$E_{\left[\left\{m_{\underline{\mathbf{q}}}^{(\alpha)}\right\}\right]} = \tfrac{1}{p}\sum_{\underline{\mathbf{q}},\alpha}\left(\sum_{\underline{\mathbf{p}}} h_{\underline{\mathbf{qp}}} m_{\underline{\mathbf{p}}}^{(\alpha)}\right)\left(\underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{w}}_q\right)^2 = \tfrac{1}{p}\sum_{\underline{\mathbf{p}}\alpha} m_{\underline{\mathbf{p}}}^{(\alpha)} \sum_{\underline{\mathbf{q}}} h_{\underline{\mathbf{qp}}}\left(\underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{w}}_{\underline{\mathbf{q}}}\right)^2$$

$$\underline{\mathbf{w}}_{\underline{\mathbf{q}}} = \frac{\sum_{\alpha'}\left(\sum_{\underline{\mathbf{p}}} h_{\underline{\mathbf{qp}}} m_{\underline{\mathbf{p}}}^{(\alpha')}\right)\underline{\mathbf{x}}^{(\alpha')}}{\sum_{\alpha'}\left(\sum_{\underline{\mathbf{p}}} h_{\underline{\mathbf{qp}}} m_{\underline{\mathbf{p}}}^{(\alpha')}\right)}$$

$\underline{\mathbf{w}}_{\underline{\mathbf{q}}}$: center of mass of all data which belongs to cluster weighted by the neighborhood function $h_{\underline{\mathbf{qp}}}$

# On-line minimization of $E^T$

**Algorithm 3:** Online learning for SOM with Euclidean distances

Initialization of prototypes

Select learning step $\varepsilon$

**begin**

choose a data point $\underline{\mathbf{x}}^{(\alpha)}$

assign data point to the prototype with minimum assignment cost

$$\underline{\mathbf{p}} = \underset{\underline{\mathbf{r}}}{\operatorname{argmin}} \sum_{\underline{\mathbf{q}}} \underbrace{h_{\underline{\mathbf{r}}\underline{\mathbf{q}}}}_{\circledast} \left(\underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{w}}_{\underline{\mathbf{q}}}^{\text{old}}\right)^2$$

change all prototypes according to

$$\Delta \underline{\mathbf{w}}_{\underline{\mathbf{q}}} = \varepsilon h_{\underline{\mathbf{p}}\underline{\mathbf{q}}} \left(\underline{\mathbf{x}}^{(\alpha)} - \underline{\mathbf{w}}_{\underline{\mathbf{q}}}\right) \text{ for all } \underline{\mathbf{q}}$$

**end**