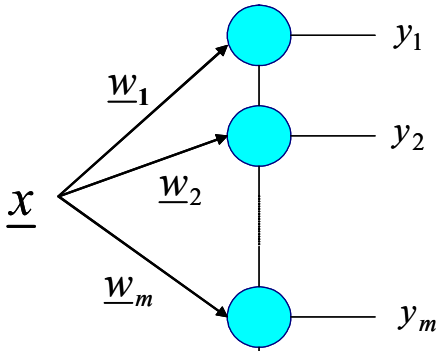
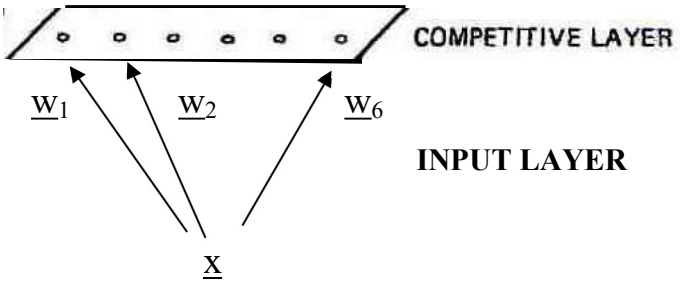
SOM - Mapas Auto-Organizáveis de Kohonen

1 – Estrutura da Rede



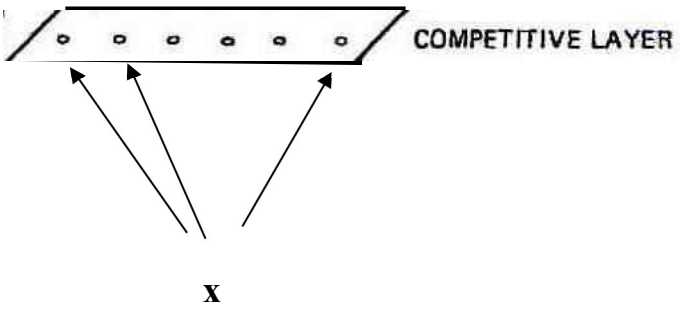
Camada de Kohonen

Unidimensional



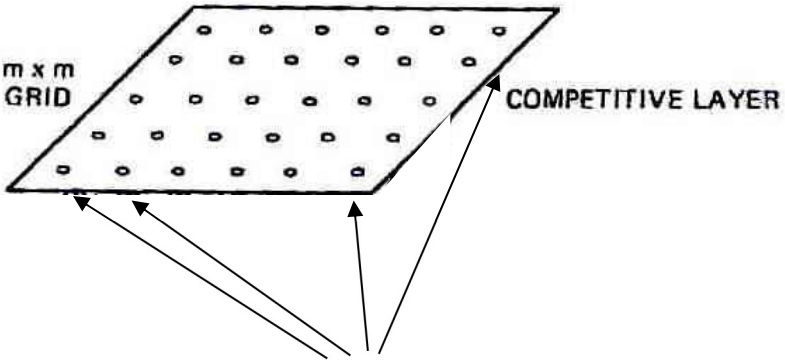
Unidimensional

P neurônios em linha

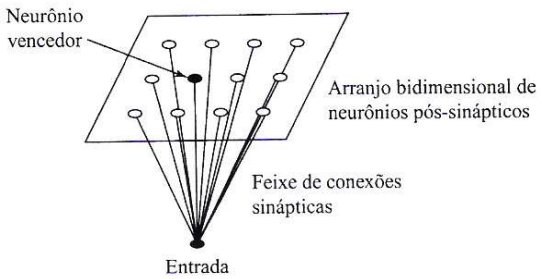


Bidimensional

Arranjo de P x Q neurônios



Bidimensional



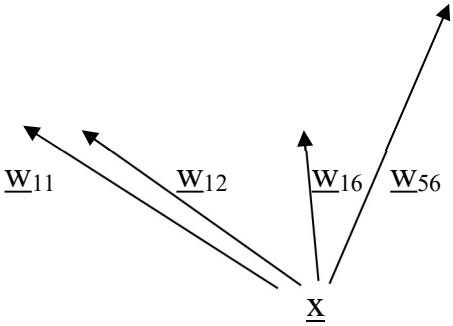
(b) Modelo de Kohonen

Fundamental: No treinamento de Kohonem existem dois espaços à considerar:

O espaço das entradas *X*

as entradas x e as sinapses w são definidas neste espaço

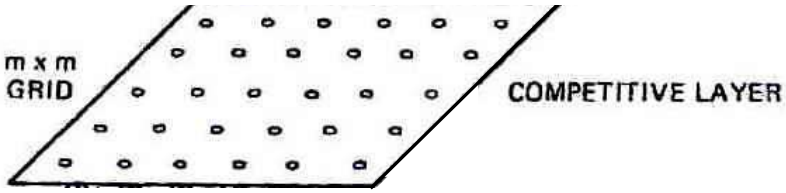
as distâncias d neste espaço serão $d = |\underline{x} - \underline{w}|$

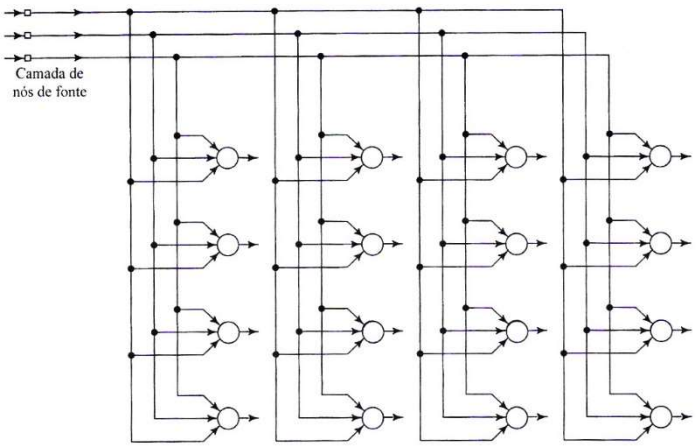


O espaço de competição *M* , o mapa auto organizável de Kohonen

As distâncias m entre os neurônios são definidas neste espaço

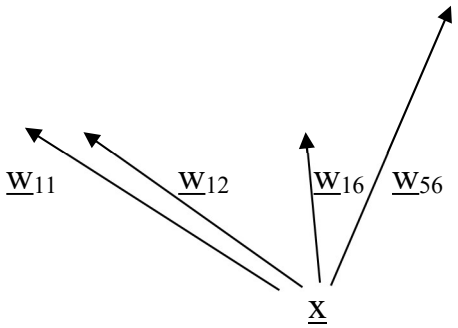
Neurônios N_i e N_i $m = |i - j|$

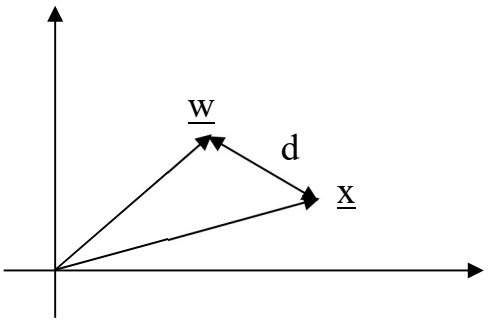




1.1 - Distâncias

Distância entre entradas e/ou sinapses no espaço de entrada





w = [2, 2]

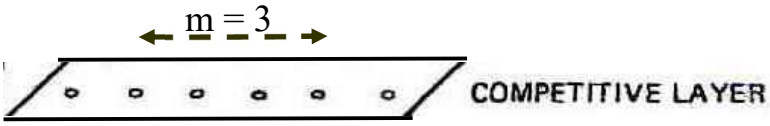
x = [3, 1]

$$d(\vec{x},\vec{w}) = \sqrt{(3-2)^2 + (1-2)^2} = \sqrt{2}$$

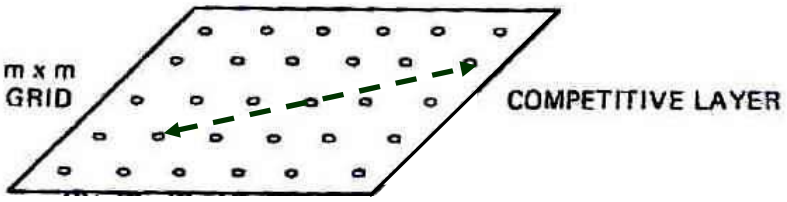
Distância entre neurônios no espaço competitivo

Mapa unidimensional

$m(N_2,N_5) = |2 - 5| = 3$

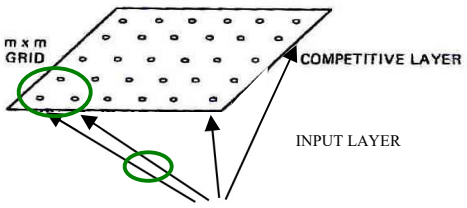


Mapa bi-dimensional



$$m(N_{22}\, N_{64}) = \sqrt{(2-6)^2 + (2-4)^2} = \sqrt{20}$$

2 – Treinamento



2.1 Treinamento de Kohonen - Objetivo

Objetivo:

Sinapses $\underline{w}_i$ e $\underline{w}_j$ próximas entre si

(proximidade medida no espaço de entrada, $d(|\underline{w}_i , \underline{w}_j) = |\underline{w}_i - \underline{w}_j|$ pequeno)

alimentam neurônios N_i e N_j próximos entre si

(proximidade medida no mapa de Kohonen, $m(N_i , N_j) = |i - j|$ pequeno)

isto é:

Similaridade nas sinapses (os padrões levantados) corresponde à

Similaridade na posição dos respectivos neurônios no mapa, e vice-versa.

2.2 - Treinamento de Kohonen - Fases

O treinamento de Kohonen passa por duas fases:

Fase de auto-organização ou ordenação

- **nesta fase inicial ocorre a ordenação topológica dos vetores sinapse**
- **é a fase mais crítica**
- **os neurônios treinam em conjunto com seus vizinhos imediatos**
- **leva usualmente cerca de 1000 passos de treinamento**
- **nesta fase manter $.01 \leq \alpha \leq .1$ para permitir uma movimentação (alteração) eficiente das sinapses.**

Fase de Convergência

- nesta fase é feita a "sintonia fina" dos neurônios do mapa, os neurônios adquirem os detalhes das estruturas das entradas

- pode ser longa, leva tipicamente 500 vezes o número de neurônios (?), P passos para mapas unidimensionais ou 500 PQ passos para mapas bidimensionais

-cada neurônio opera de nodo praticamente independente ou com uma vizinhança muito pequena. $\sigma_R^2(n) \leq 1$ **nesta fase**

2.3 - Treinamento de Kohonen – Ações

O Treinamento de Kohonen envolve três tipos de ações:

2.3.1 - Ação 1 - Competição:

É apresentada uma entrada x e os neurônios competem entre si. É declarado

Neurônio vencedor

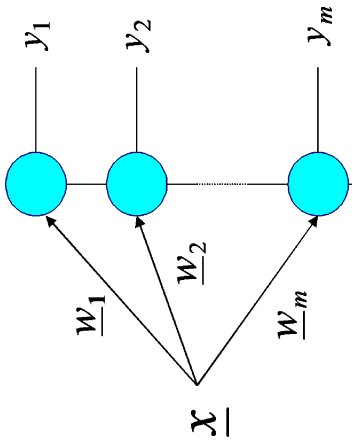
o neurônio N_i cujo vetor sinapse w_i é o mais próximo de x , isto é

$$i = \arg\underset{\forall j}{Min} \left| \vec{x} - \vec{w}_j \right|$$

Ex: Camada unidimensional com 8 neurônios

entrada (e sinapses) unidimensionais

N_i,	1	2	3	4	5	6	7	8
w_i	.2	.5	.8	.9	.1	.5	.6	.2



é apresentada a entrada x=.65.

N_7 é o vencedor porque

$$\left|\vec{x} - \vec{w}_7\right| < \left|\vec{x} - \vec{w}_j\right| \quad \forall \; j \neq 7$$

note que a competição ocorre na camada de entrada

(e não na chamada camada competitiva !)

2.3.2 - Ação 2- Cooperação

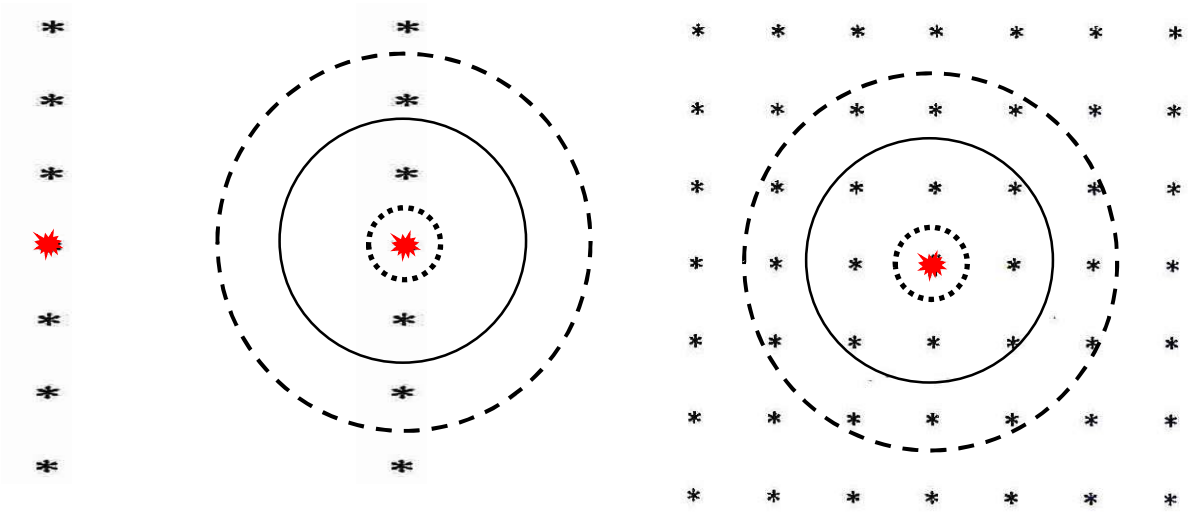
Durante a fase de auto-organização cada neurônio vencedor treina também seus vizinhos próximos. Este processo é que garante a "continuidade" do mapa, o agrupamento de entradas similares alimentando neurônios vizinhos no mapa e a transição suave entre classes no mapa.

Para a aplicação deste conceito é necessário definir uma "vizinhança" para os neurônios, e uma forma de varia-la no tempo.

Vizinhança - Distância na Camada Competitiva Uni e Bidimensional

Raio de Vizinhança R (distância entre neurônios, no mapa)

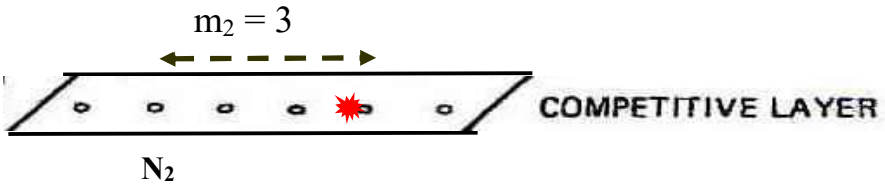
Neurônio vencedor



Função de Vizinhança $h(m_i)$

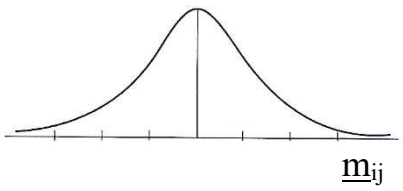
Um neurônio ativado afeta mais intensamente seus vizinhos mais próximos na camada competitiva

m_i = distância do neurônio N_i ao neurônio vencedor



Função de Vizinhança Gaussiana

$$h(m_i) = \exp(-\frac{m_i^2}{2\sigma^2})$$



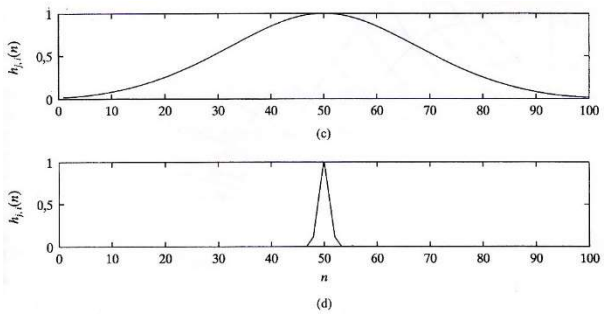
Exemplos de h(m_i) Gaussiana:

Mapa unidimensional, P = 100

Vencedor: N_{50} ✸

c) $\sigma \approx 20$

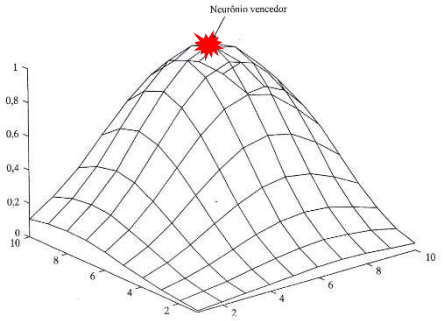
d) $\sigma \approx 1$



Mapa bidimensional, P = Q = 10

Vencedor: $N_{5,5}$ ✸

$\sigma \approx 3$



Valor inicial e taxa de decaimento da largura da vizinhança

No início do treinamento a vizinhança deve cobrir praticamente toda a camada competitiva, e a medida que o treinamento decorre deve ir diminuindo até que ao fim da fase de auto-organização deve estar reduzida a praticamente um neurônio (τ_h é arbitrado tal que $\sigma(1000) = 1$). Podemos então utilizar:

$$h_j(n, m_j) = \exp\left(-\frac{m_j^2}{2\sigma^2(n)}\right) \qquad m_j^2 = \left|N_j - N_{vencedor}\right|^2 \qquad \sigma = \sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_h}\right)$$

$$\sigma_0 \approx \frac{\text{max\_dim\_grade}}{5} = \begin{cases} 0.2\,P & \text{para mapa unidimensional} \\ 0.2\sqrt{P^2 + Q^2} & \text{para mapa bidimensional} \end{cases} \qquad \tau_h = \frac{1000}{\ln \sigma_0}$$

Formas típicas de h para n pequenos estão apresentados nos exemplos de função de vizinhança gaussiana acima.

È possível simplificar a computação recursiva utilizando aproximações pela série de Taylor de primeira ordem

$$y(n+1) \cong \left[1 + \frac{1}{y(n)} \frac{\partial y(n)}{\partial n}\right] y(n)$$

$$\sigma_0 \approx \frac{\text{max\_dim\_grade}}{5} = \begin{cases} 0.2\,P & \text{para mapa unidimensional} \\ 0.2\sqrt{P^2 + Q^2} & \text{para mapa bidimensional} \end{cases} \qquad \tau_h = \frac{1000}{\ln \sigma_0}$$

$$\sigma(0) = \sigma_0 \qquad e \qquad \sigma(n+1) \cong \left(1 - \frac{1}{\tau_h}\right) \sigma(n)$$

$$z(0) = \exp\left(-\frac{1}{2\sigma_0^2}\right) \qquad e \qquad z(n+1) \cong \left(1 - \frac{1}{\tau_h \sigma^2(n)}\right) z(n)$$

$$m_j^2 = \left| N_j - N_{vencedor} \right|^2 \qquad e \qquad h_j(n) = \left[z(n)\right]^{m_{ji}^2}$$

2.3.3 - Ação 3 – Aprendizado, alteração do valor das sinapses.

Valores iniciais das sinapses

Como não sabemos como o mapa irá se organizar é necessário que os valores iniciais sejam pequenos e randômicos, para não polarizar o processo em um mínimo local. Se o escalamento das variáveis de entrada foi feito conforme será discutido na seção de pré-processamento, o valor rms do ruído em cada dimensão será unitário. Neste caso é razoável iniciar as sinapses com valores randômicos na faixa (-1, +1).

Treinamento

Ao longo de todo o treinamento a rede aprende, isto é, tem suas sinapses modificadas em direção a uma entrada. A velocidade do treinamento necessita variar, deve ser mais rápida na fase de auto-organização, que requer maiores mudanças, e mais lenta na de convergência, para permitir que uma boa “sintonia fina”, i.e., o minimante da função objetivo, seja alcançado com precisão.

A atualização seletiva de uma sinapse é feita movendo-a na direção da entrada

$$\vec{w}_j(n+1) = \vec{w}_j(n) + \alpha\left\{\vec{x} - \vec{w}_j(n)\right\}$$

Valor inicial e taxa de decaimento do passo

Durante o treinamento o passo α deve iniciar no entorno de 0,1 decair aproximadamente 10 vezes em 1000 passos de treinamento (na fase de auto-organização do mapa, os primeiros 1000 passos aproximadamente, $.1 \geq \alpha \geq .01$, o que implica em $\tau_\alpha \sim 500$). Podemos então utilizar:

$$\alpha(n) = \alpha_0 \exp\left(-\frac{n}{\tau_\alpha}\right) \qquad \alpha_0 = .1 \qquad \tau_\alpha \cong 500$$

ou de forma recursiva usando a aproximação por série de Taylor

$$\alpha(0) = .1 \qquad e \qquad \alpha(n+1) = .998\,\alpha(n)$$

onde n é o passo de treinamento atual.

Atualização completa das sinapses

$$\vec{w}_j(n+1) = \vec{w}_j(n) + \alpha(n)\,h_j(n)\left\{\vec{x} - \vec{w}_j(n)\right\} \qquad \forall\ j$$

$$\alpha(n) = \alpha_0 \exp\left(-\frac{n}{\tau_\alpha}\right) \qquad \alpha_0 = .1 \qquad \tau_\alpha = 500$$

$$h_j(n) = \exp\left(-\frac{m_j^2}{2\sigma^2(n)}\right) \qquad m_j^2 = \left|N_j - N_{vencedor}\right|^2 \qquad \sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_h}\right)$$

$$\sigma_0 \approx \frac{\max\dim grade}{5} = \begin{cases} 0.2\,P & para\_mapa\_uni\dim ensional \\ 0.2\sqrt{P^2+Q^2} & para\_mapa\_bi\dim ensional \end{cases} \qquad \tau_h = \frac{1000}{\ln\sigma_0}$$

É possível simplificar a computação utilizando aproximações pela série de Taylor de primeira ordem. De forma aproximada

$$\vec{w}_j(n+1) = \vec{w}_j(n) + \alpha(n)\, h_j(n)\left\{ \vec{x} - \vec{w}_j(n)\right\} \qquad \forall\ j$$

$$\alpha(0) = .1 \qquad e \qquad \alpha(n+1) = .998\,\alpha(n)$$

$$\sigma_0 \approx \frac{\max\_\dim\_grade}{5} = \begin{cases} 0.2\,P & para\_mapa\_unidimensional \\ 0.2\sqrt{P^2+Q^2} & para\_mapa\_bidimensional \end{cases} \qquad \tau_h = \frac{1000}{\ln \sigma_0}$$

$$\sigma(0) = \sigma_0 \qquad e \qquad \sigma(n+1) = \left(1 - \frac{1}{\tau_h}\right)\sigma(n)$$

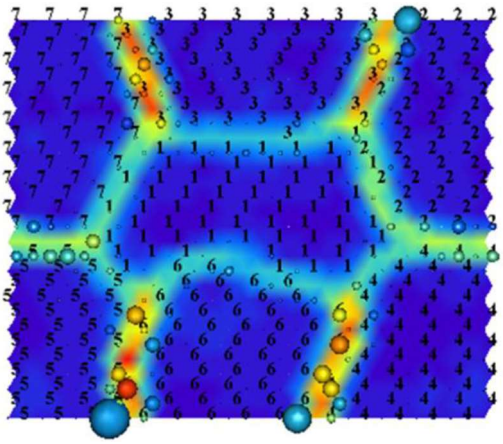
$$z(0) = \exp\left(-\frac{1}{2\sigma_0^2}\right) \qquad e \qquad z(n+1) = \left(1 + \frac{1}{\tau_h \sigma^2(n)}\right) z(n)$$

$$m_j^2 = \left| N_j - N_{vencedor} \right|^2 \qquad e \qquad h_j(n) = \left[z(n) \right]^{m_j^2}$$

Fase de Auto-organização - Término

Verificar a matriz das distâncias - U-Matrix – ítem 5.2

Se as classes já estiverem bem definidas a fase de auto-organização está concluída



2.4 - Treinamento de Kohonen - resumo

entrada x(n)

Passo (ação) 1 - verificar o neurônio N_i ganhador, $u_i > u_j \;\; \forall \;\; j \neq i$

a competição é feita no domínio de entrada (não no mapa !)

Passo(ação) 2 – verificar os vizinhos de N_i com função de vizinhança não nula

$$N_{i-R}\,,\, N_{i-R+1}\,,\, \ldots\,,\, N_{i-1}\,,\, N_i\,,\, N_{i+1}\,,\, \ldots\,,\, N_{i+R-1}\,,\, N_{i+R}$$

a similaridade é medida no mapa (não no domínio das entradas !)

Passo (ação) 3 – atualizar as sinapses dos neurônios vizinhos

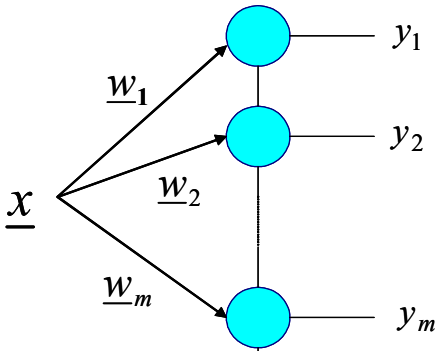
$$\vec{w}_j(n+1) = \vec{w}_j(n) + \alpha(n)\, h_j(n)\left\{\vec{x} - \vec{w}_j(n)\right\} \qquad \forall \;\; j$$

a atualização é feita no domínio da entrada

À medida que o treinamento progride as equações:

- **vão reduzindo σ**
- **vão reduzindo α**

2.5 - Exemplo de um passo de treinamento:



x unidimensional

mapa unidimensional, P = 8 neurônios

havíamos iniciado com

$\sigma_0 = 0.2\,P = 1.6 \qquad \tau_h = \dfrac{1000}{\ln \sigma_0} \cong 2000$

$\alpha_0 = 0.1 \qquad \tau_\alpha = 1000$

No passo n= 100 as sinapses tem os valores indicados abaixo

N_i, i =	1	2	3	4	5	6	7	8
w_i (100)=	.2	.5	.8	.9	.1	.5	.6	.2

e é apresentada a entrada x=.65

cálculos auxiliares:

para n=100

$$\sigma(100) = \sigma_0 \exp\left(-\frac{n}{\tau_h}\right) = 1.6 \exp\left(-\frac{100}{2000}\right) = 1.52$$

$$h_j(100) = \exp\left(-\frac{m_j^2}{2\sigma^2(n)}\right) = \left(\exp\left(-\frac{1}{2(1.52)^2}\right)\right)^{m_j^2} = (.81)^{m_j^2} = \begin{cases} 1.0 & m_j = 0 \\ .81 & m_j = 1 \\ .43 & m_j = 2 \\ .15 & m_j = 3 \\ .03 & m_j = 4 \\ \cong 0 & m_j \geq 5 \end{cases}$$

$$\alpha(100) = \alpha_0 \exp\left(-\frac{n}{\tau_\alpha}\right) = 0.1 \exp\left(-\frac{100}{1000}\right) = 0.09$$

Passo (ação) 1 - Competição no espaço de entrada: x = .65

N_i, i =	1	2	3	4	5	6	7	8
w_i (100)=	.2	.5	.8	.9	.1	.5	.6	.2

$u_i = -|x - w_i|^2$ $\quad u_7 > u_j \ \forall \ j = 1,2,3,4,5,6,8$ **N_7 vencedor**

Passo (ação) 2 – Valores de m e h para os vizinhos de N_5 no mapa

N_i	1	2	3	4	5	6	7	8
m_i	6	5	4	3	2	1	0	1
h_i	0	0	.03	.15	.43	.81	1	.81

Passo (ação) 3 - Atualização das sinapses no espaço de entrada

$$\vec{w}_j(101) = \vec{w}_j(100) + \alpha(100)\, h_j(100)\left\{ \vec{x} - \vec{w}_j(100)\right\} \qquad \forall \ j$$

N_i	1	2	3	4	5	6	7	8
w_i antigo	.2	.5	.8	.9	.1	.5	.6	.2
w_i novo	.2	.5	.8	. 896	. 121	.511	.605	.232

Note que:

- **todas as sinapses alteradas se aproximaram da entrada x = .65 e entre si.**
- **sinapses próximas do neurônio vencedor são mais afetadas**
- **sinapses (e estruturas) distantes praticamente não se alteram**

2.6 - Treinamento em Batelada

Na época n, para toda entrada $\underline{x}_k$, k = 1, 2, ..., K calcule os acréscimos a serem aplicados em cada neurônio

$$\Delta\vec{w}_j(n,k) = \alpha(n)\, h_j(n)\left\{\vec{x}_k - \vec{w}_j(n)\right\} \qquad \forall\ j$$

Aplique o acréscimo médio nas sinapses

$$\vec{w}_j(n+1) = \vec{w}_j(n) + \Delta\vec{w}_j(n) \qquad \forall\ j$$

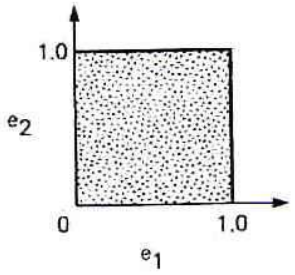
$$\Delta\vec{w}_j(n) = \alpha(n)\frac{1}{K}\sum_{k=1}^{K} h_j(n)\left\{\vec{x}_k - \vec{w}_j(n)\right\} \qquad \forall\ j$$

Idêntico, mas o acréscimo em cada neurônio é o valor médio do acréscimo calculado

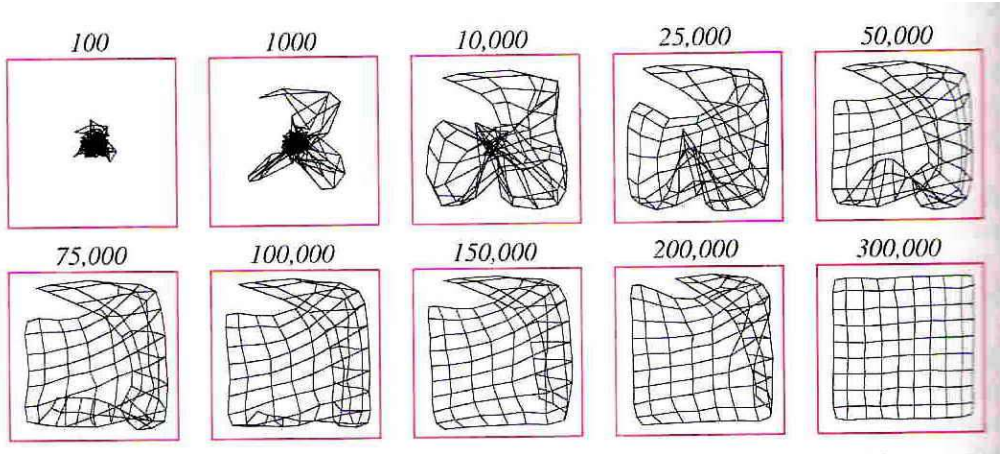
3 Exemplos / Aplicações

3.1 Evolução do mapa:

Entrada



Mapa:



3.1.1 - Problemas na

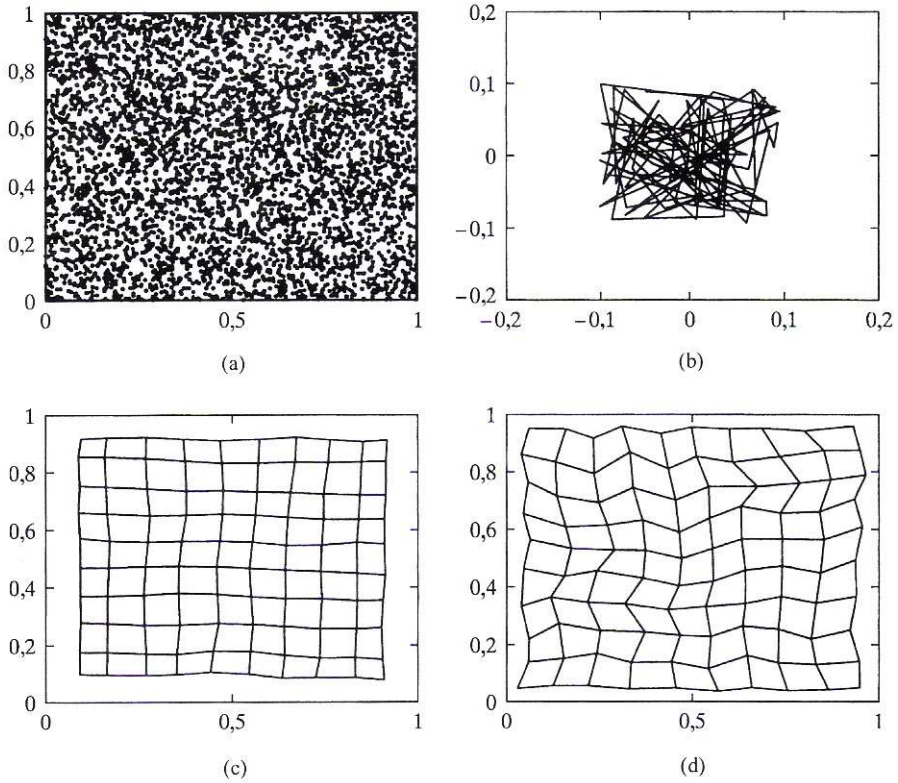
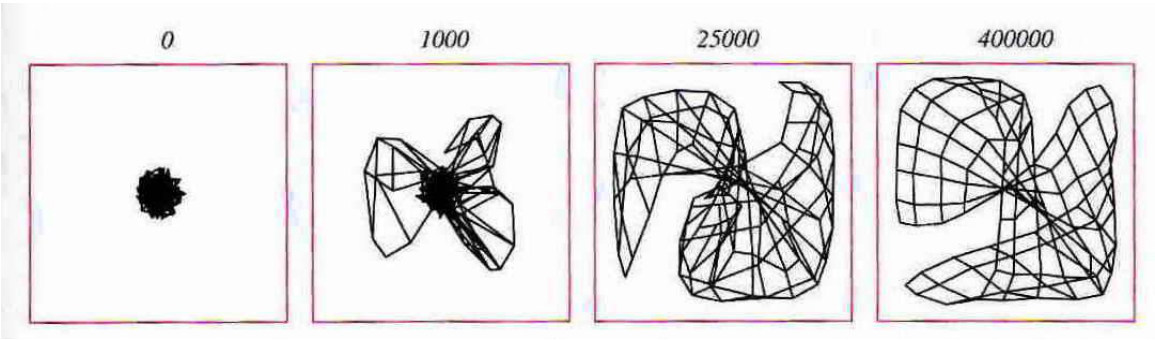
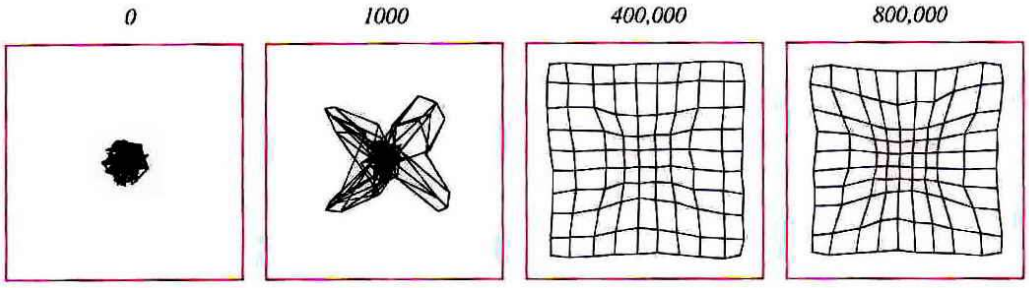


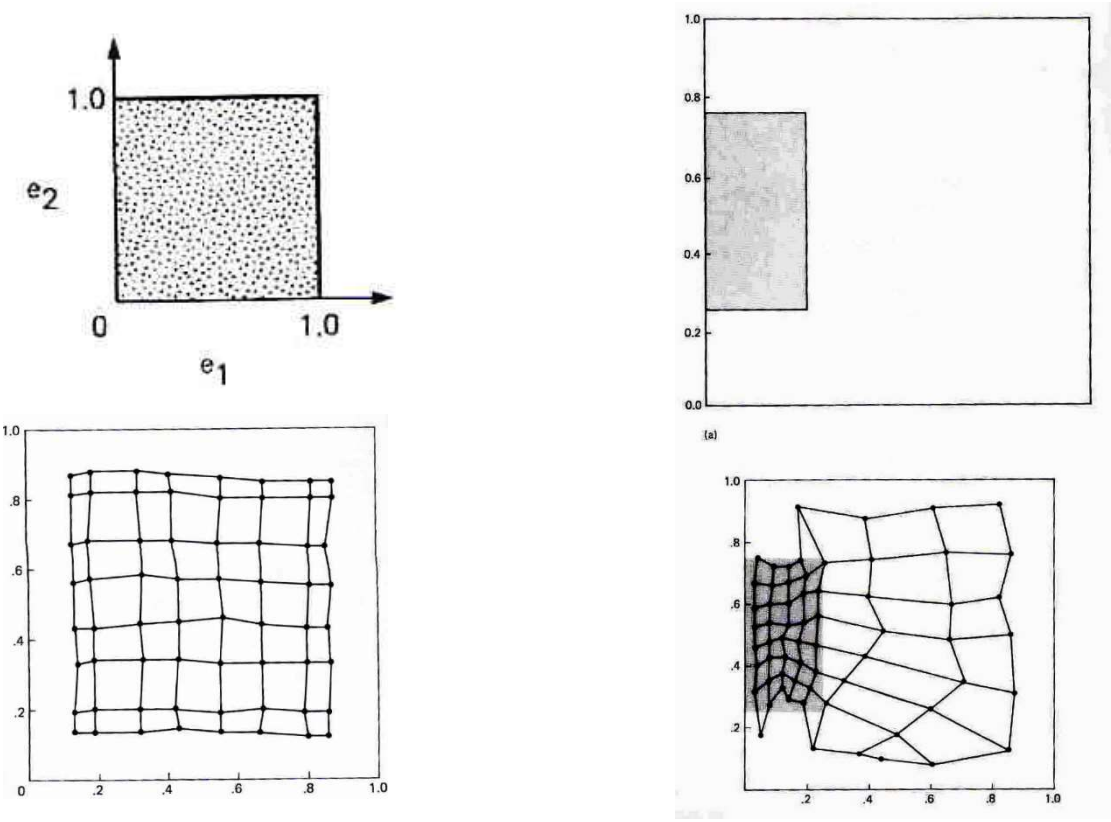
FIGURA 9.8 (a) Distribuição dos dados de entrada. (b) Condição inicial da grade bidimensional. (c) Condição da grade no final da fase de ordenação. (d) Condição da grade no final da fase de convergência



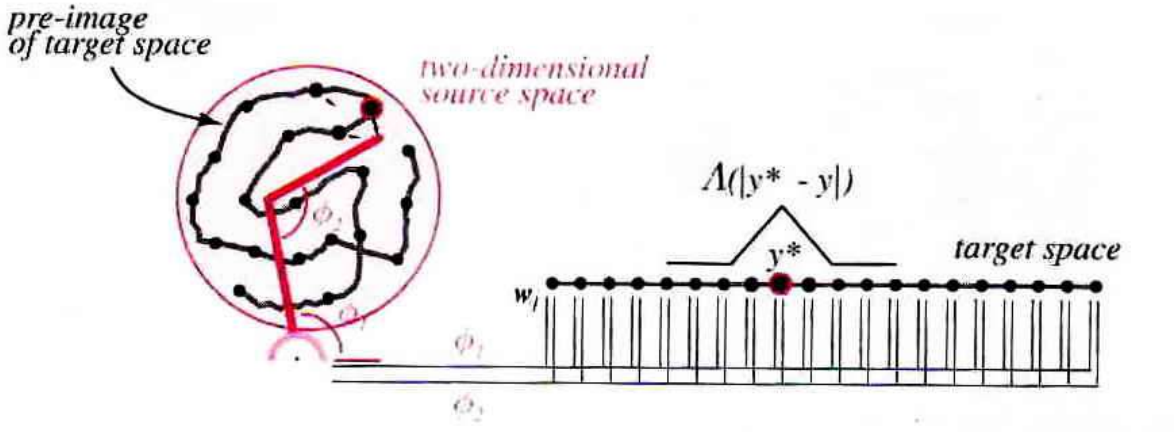
3.2 Regiões da entrada com densidades populacionais diferentes:



De novo, regiões da entrada com densidades populacionais diferentes:



3.3 Redução de dimensão de representação:



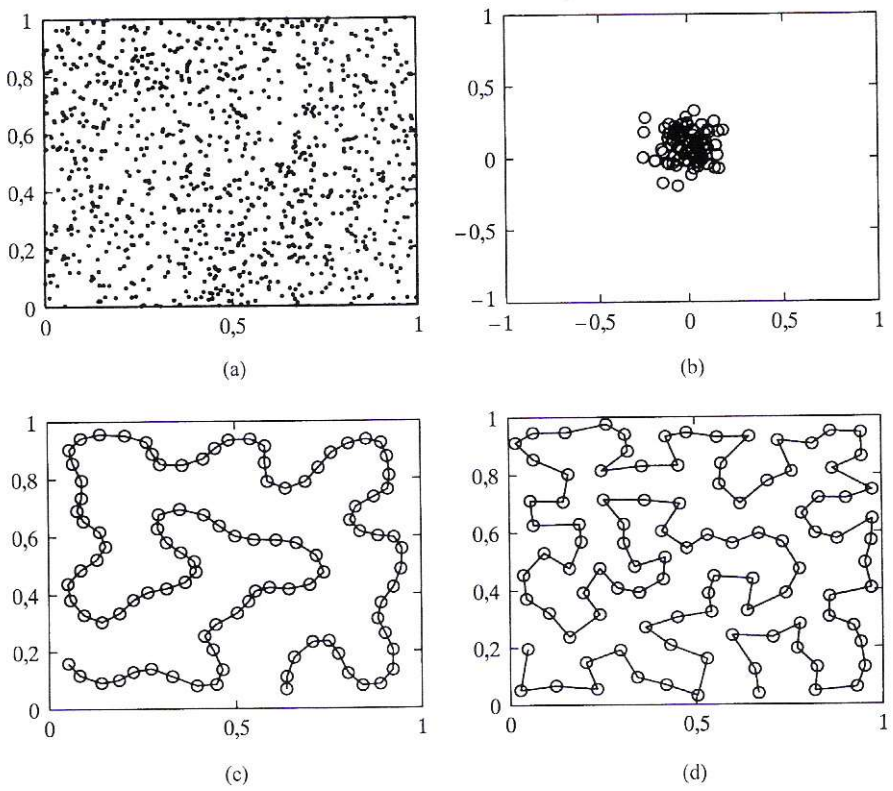


FIGURA 9.9 (a) Distribuição de dados de entrada bidimensionais. (b) Condição inicial da grade unidimensional. (c) Condição da rede no final da fase de ordenação. (d) Condição da grade no final da fase de convergência

3.4 Aplicação: agrupamento (classificação) de animais por seus atributos

Entradas

TABELA 9.3 Nomes de Animais e seus Atributos

Animal		Pombo	Galinha	Pato	Ganso	Coruja	Falcão	Águia	Raposa	Cão	Lobo	Gato	Tigre	Leão	Cavalo	Zebra	Vaca
é	pequeno	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0
	médio	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
	grande	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
tem	2 patas	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
	4 patas	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	pêlos	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	cascos	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
	crina/juba	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
	penas	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
gosta de	caçar	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0
	correr	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0
	voar	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0
	nadar	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

$\vec{x}_{falcão}$

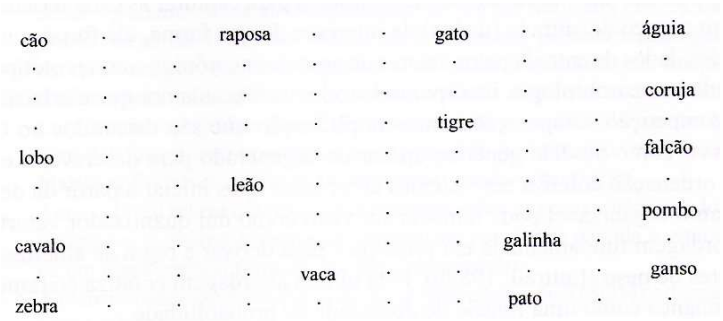
Mapa de Kohonen

mamíferos x aves

predadores x não

pequenos x grandes

...



cão	cão	raposa	raposa	raposa	gato	gato	gato	águia	águia
cão	cão	raposa	raposa	raposa	gato	gato	gato	águia	águia
lobo	lobo	lobo	raposa	gato	tigre	tigre	tigre	coruja	coruja
lobo	lobo	leão	leão	leão	tigre	tigre	tigre	falcão	falcão
lobo	lobo	leão	leão	leão	tigre	tigre	tigre	falcão	falcão
lobo	lobo	leão	leão	leão	coruja	pombo	falcão	pombo	pombo
cavalo	cavalo	leão	leão	leão	pombo	galinha	galinha	pombo	pombo
cavalo	cavalo	zebra	vaca	vaca	vaca	galinha	galinha	pombo	pombo
zebra	zebra	zebra	vaca	vaca	vaca	galinha	galinha	pato	ganso
zebra	zebra	zebra	vaca	vaca	vaca	pato	pato	pato	ganso

4 - Dimensionando o Mapa

Número N de neurônios

Heurística:

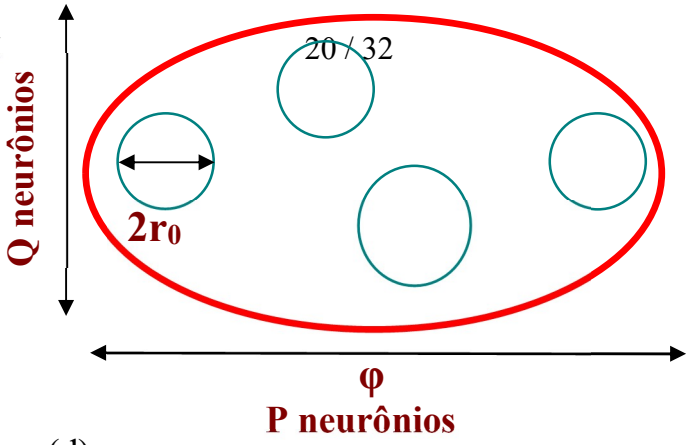
$$N = 5\sqrt{n}$$?????

n – número de instâncias (eventos)

P x Q neurônios $(P \geq Q)$

Definição da Granularidade

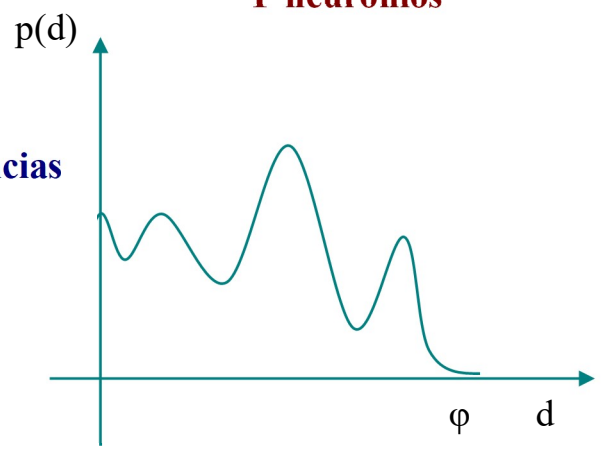
(no mapa)



No espaço de entrada:

Na direção de máximo espalhamento (1ª PCA) Do histograma p(d) das distâncias entre as entradas

$$d = \left| \vec{x}_i - \vec{x}_j \right|$$



Estimamos o diâmetro da classe única

$$\varphi = Max\, d$$

e a direção em que este diâmetro ocorre

$$\vec{d}_\varphi = \frac{\vec{x}_i - \vec{x}_i}{\left| \vec{x}_i - \vec{x}_i \right|} \qquad onde \qquad \vec{x}_i, \vec{x}_i = \arg\, Max \left| \vec{x}_i - \vec{x}_i \right|$$

(Por outro lado, se o escalamento foi feito de modo a tornar o valor rms (desvio padrão) do ruído em cada direção unitário o valor de r_0 será escolhido baseado na distribuição , conforme o percentual de elementos da classe que queremos englobar (ver escolha de r_0 para ART). O diâmetro das classes é dado por 2 r_0).

O número de classes (e de neurônios) na direção do maior diâmetro será então:

$$P > \frac{\varphi}{2r_0}$$

Para obter o número de neurônios na direção Q do mapa repetir com a direção da entrada que gera o segundo maior diâmetro

Como obter este segundo maior diâmetro φ' ? A direção $\vec{d}_{\varphi}$ do maior diâmetro φ é conhecida, da confecção do histograma. Eliminar as componentes das entradas nesta direção usando como novas entradas:

$$\vec{x}' = \vec{x} - \vec{d}_{\varphi}{}^{t} \vec{x}\, \vec{d}_{\varphi}$$

Calcular o maior diâmetro φ' para as entradas $\vec{x}'$.

Então:

$$Q > \frac{\varphi'}{2r_0}$$

uma boa alternativa é calcular a relação P/Q através das componentes PCA.

SOM e PCA são representações com dimensionalidade reduzida

PCA – Compactação linear

SOM – Compactação não linear

$$\frac{P}{Q} \approx \left[\frac{Variância 1^a\ PCA}{Variância 2^a\ PCA} \right]^{1/2}$$

5 - Determinação dos clusters

5.1 Supervisionada

Cada neurônio pertence à classe que mais o ativou

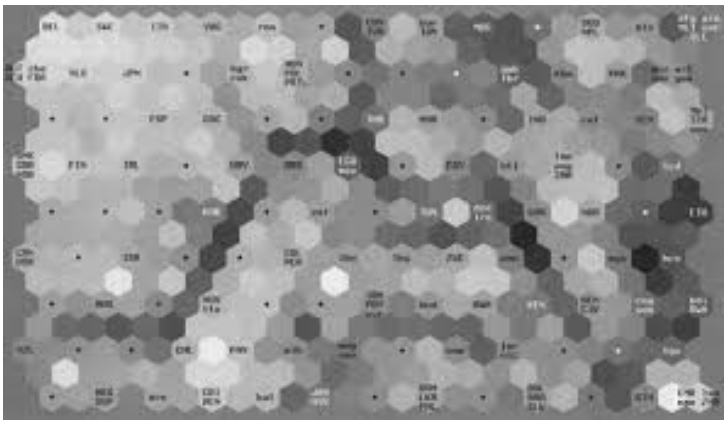
Neurônios não ativados ? **não identificados ou**
classe do maior número de vizinhos imediatos

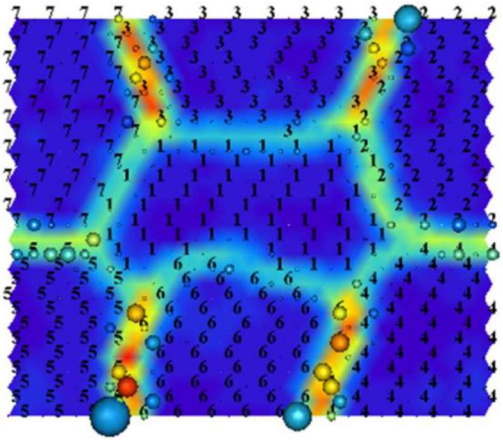
5.2 Não supervisionada

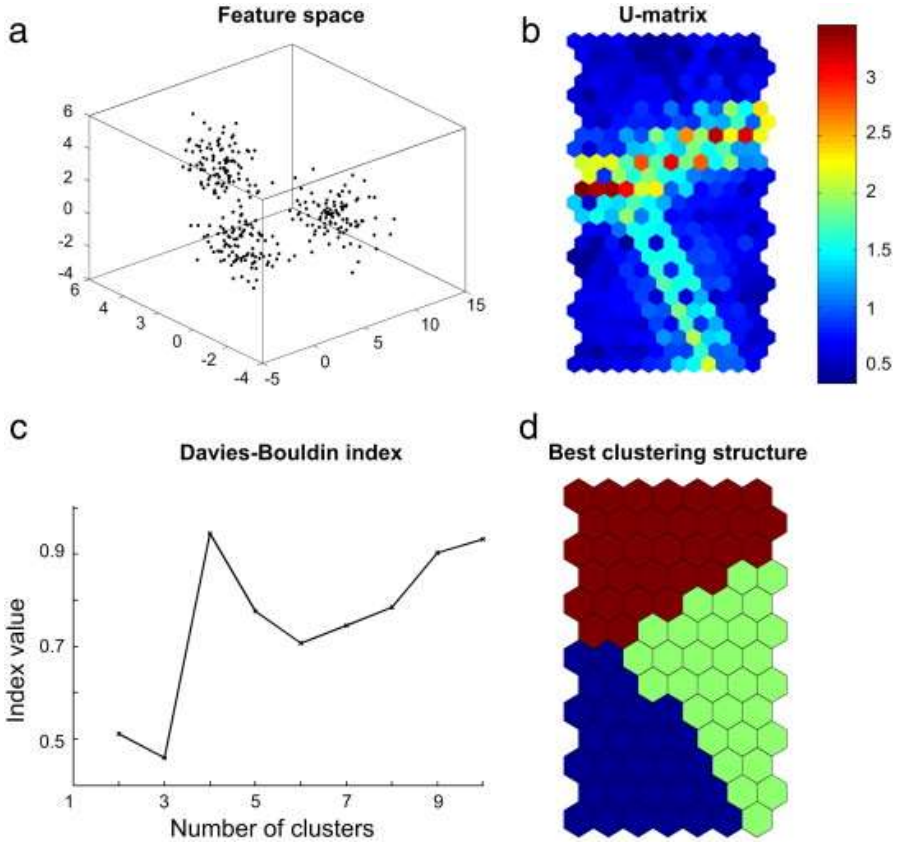
Matriz das Distâncias - U-Matrix

$$u_{ij} = \underset{\forall \vec{w}\, vizinho\, \vec{w}_{ij}}{E} \left| \vec{w}_{vizinho} - \vec{w}_{ij} \right|^2$$

picos (separações) e
vales (agrupamentos)





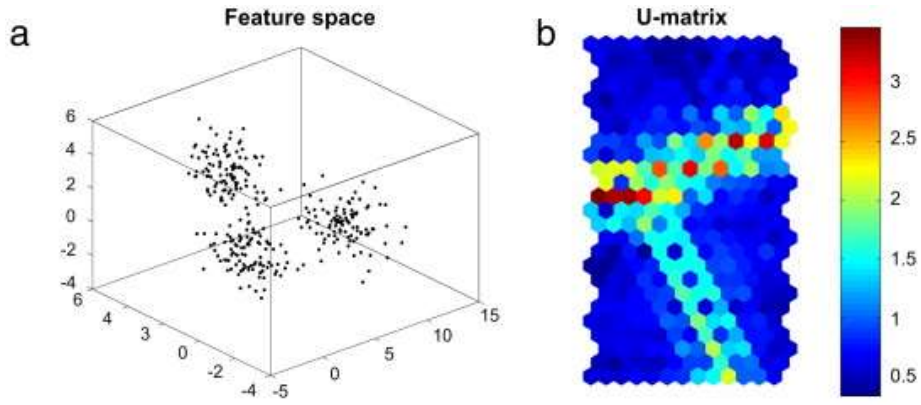


$$DB_m = \frac{1}{m}\sum_{i=1}^{m} \max_{\substack{j=1,\ldots,m \\ j\neq i}} R_{ij}$$

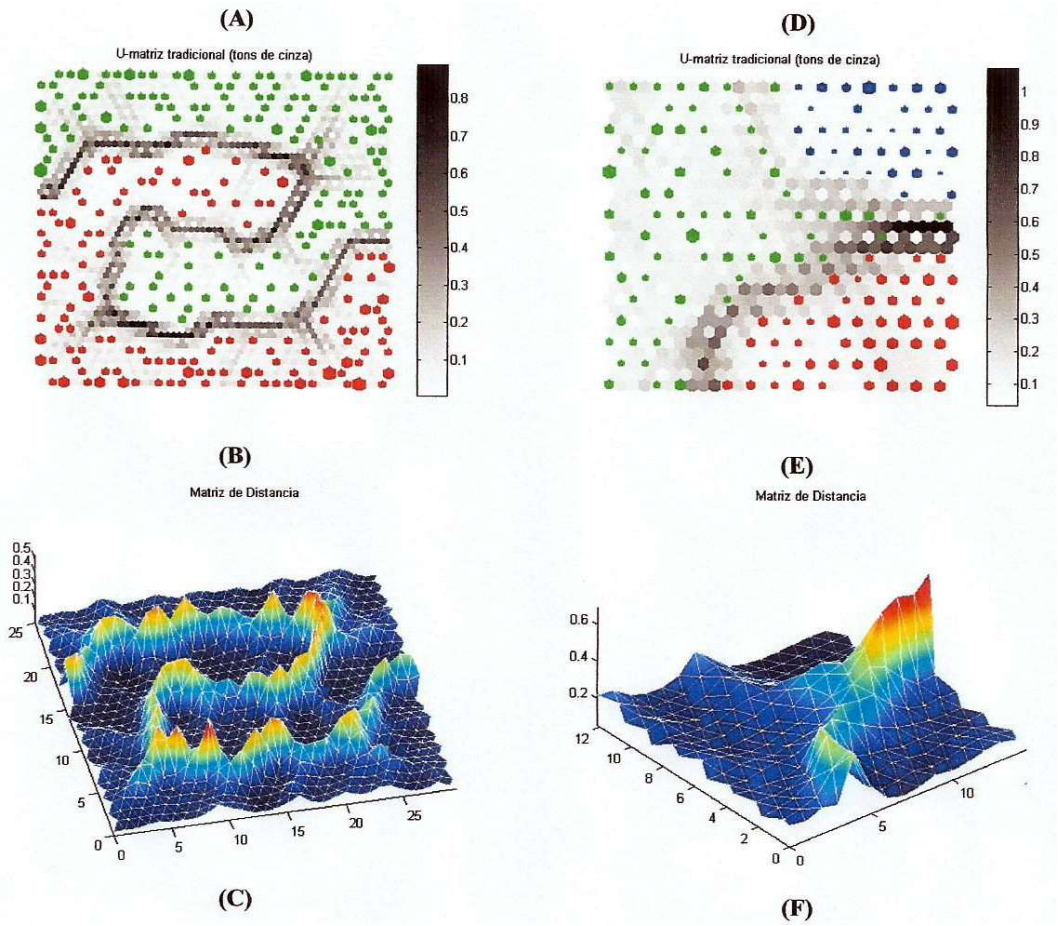
$$R_{ij} = \left(S_i + S_j\right)/ d_{ij}$$

Nomeando os clusters (supervisionada)

Quem é quem ?



Clusters não ativados ? não identificados ou considerar os vizinhos

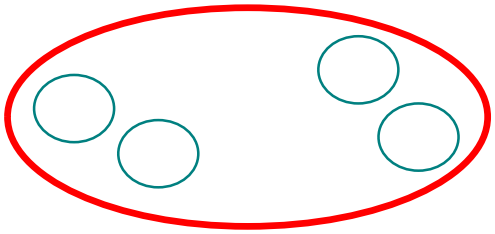


6 - Comentários:

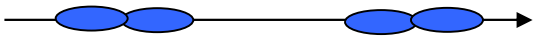
6.1 SOM vs PCA

PCA – Compactação linear

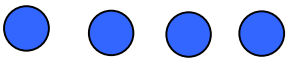
SOM – Compactação não linear, elimina espaços vazios

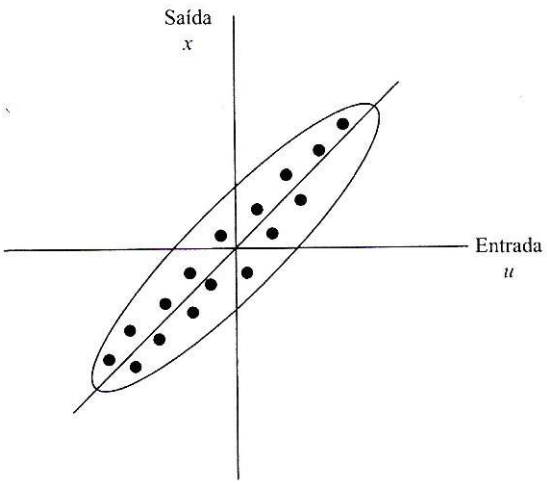


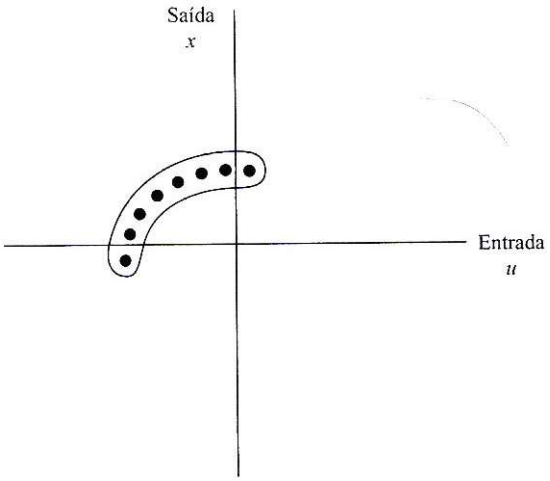
PCA



SOM







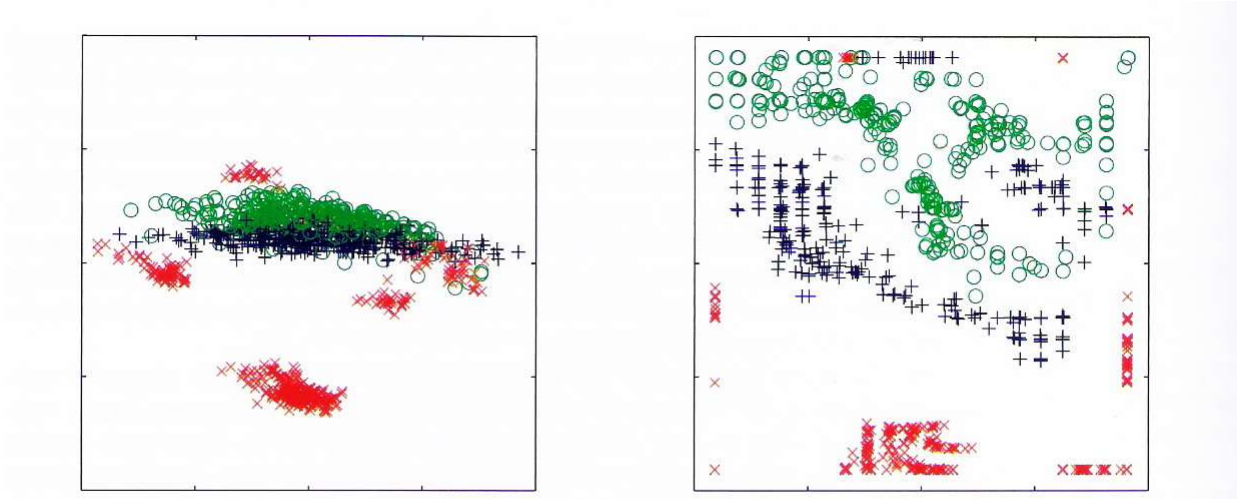


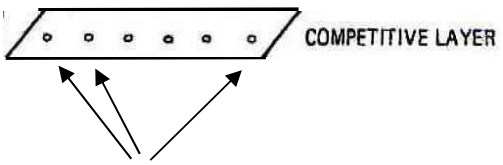
Figure 12.21 Plot of the oil flow data set visualized using PCA on the left and GTM on the right. For the GTM model, each data point is plotted at the mean of its posterior distribution in latent space. The nonlinearity of the GTM model allows the separation between the groups of data points to be seen more clearly.

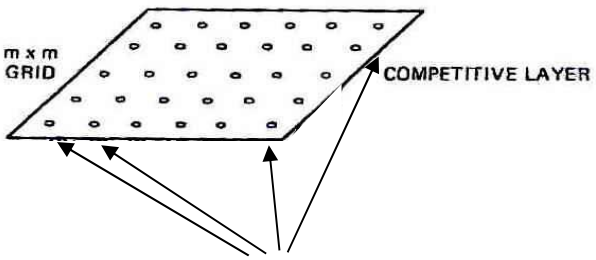
6.2 Diferença do processo anterior (Kohonen simplificada, h=0)

Kohonen simplificada: padrões similares não ficam juntos

Kohonen: Padrões similares ficam juntos,

há uma transição "suave" entre classes





6.3 – Estruturas e métricas mais usadas na camada competitiva:

6.3.1 Estrutura (arranjo)

retangular

P colunas x Q linhas

linhas **i = 1, 2, ... , Q**
colunas **j = 1, 2, ... , P**

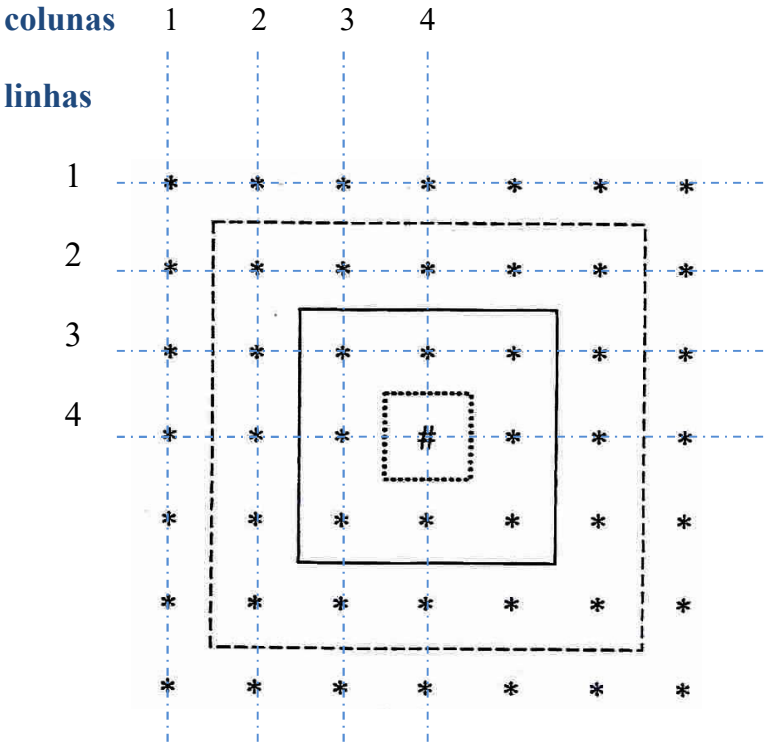
espaçamento

entre colunas = 1

entre linhas = 1

Número de neurônios **N = PQ**

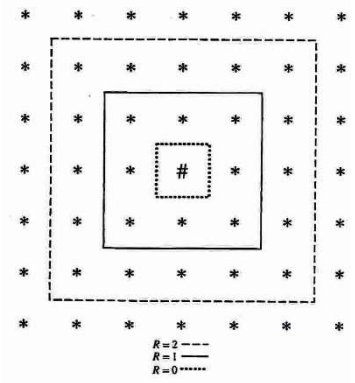
posição dos neurônios N_{ij}



Métricas

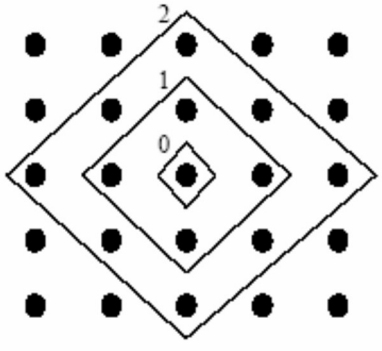
Distância "quadrados" (medida pela métrica dos quadrados)
de um neurônio $N_{i,j}$ ao neurônio N_{kl}

$$m = Max\left\{\left|i-k\right|,\left|j-l\right|\right\}$$



Métrica Manhattan

$$m = \left|i-k\right| + \left|j-l\right|$$



Distância Euclidiana

de um neurônio $N_{i,j}$ ao neurônio N_{kl}

$$d(N_{ij} - N_{kl}) = \sqrt{(i-k)^2 + (j-l)^2}$$

Distância Euclidiana entre vizinhos mais próximos:

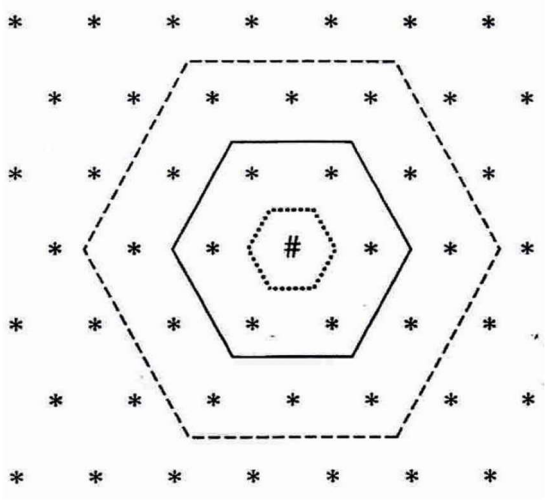
$$1 \; ou \; \sqrt{2}$$

6.3.2 Estrutura Hexagonal

2P colunas x 2Q linhas

distâncias iguais,

maiores possibilidades de vizinhanças



espaçamento entre colunas = 1/2

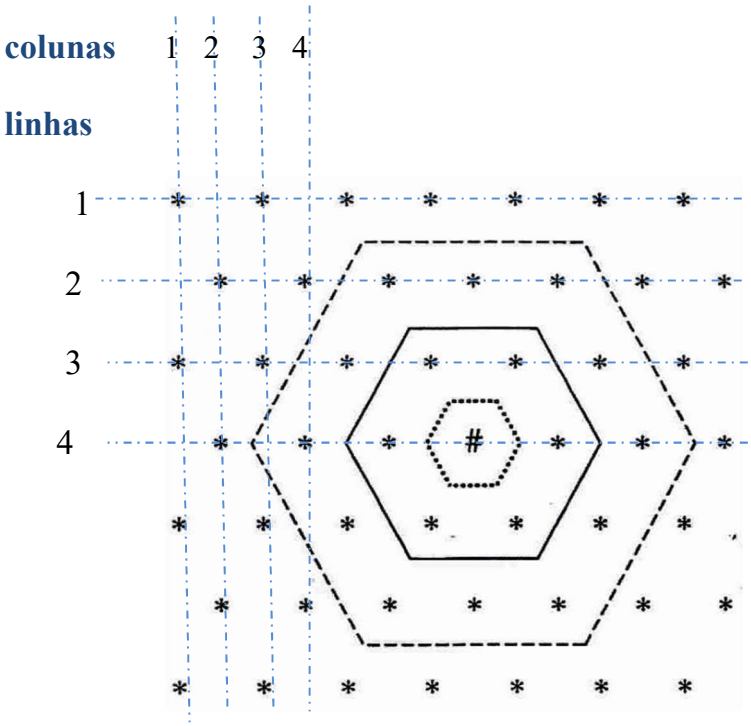
espaçamento entre linhas = $\sqrt{3}/2$

hexágonos regulares e

distância Euclidiana entre vizinhos mais próximos = 1

distância m no mapa entre vizinhos mais próximos = 1

colunas i = 1, 2, ... , 2P
linhas j = 1, 2, ... , 2Q



posição dos neurônios (i,jl)

(i-par, j-par)

ou

(i-impar, j-impar)

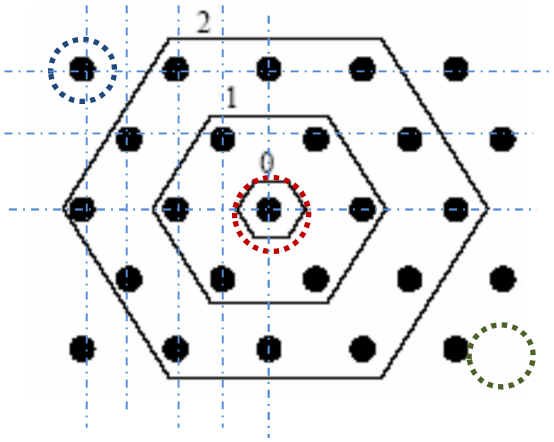
há neurônios em apenas

metade das posições (i, j) !

Neurônios intervalados

colunas impares e linhas impares
ou
colunas pares e linhas pares

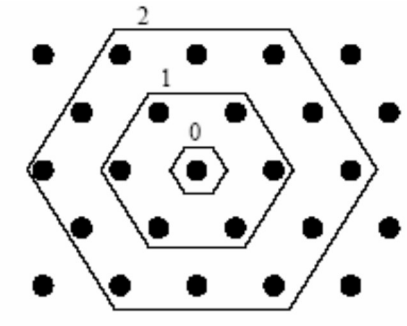
Espaçamento linhas $\sqrt{3}/2$
colunas 1/2



Exemplos: posição no canto superior esquerdo 1,1
canto inferior direito 10,5
centro 5,3

Métricas

distância "hexagonal" = distância Euclidiana



Distância entre N_{ij} e N_{kl}

linhas i, k colunas j, l

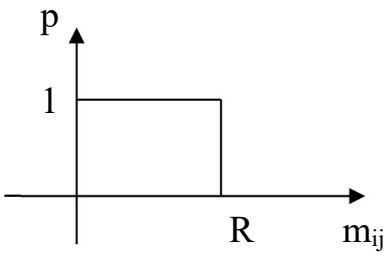
$$m(N_{ij}-N_{kl}) = d(N_{ij}-N_{kl}) = \frac{1}{2}\sqrt{3(i-k)^2+(j-l)^2}$$

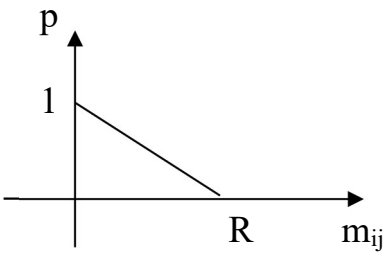
Determinação dos vizinhos à uma distância m

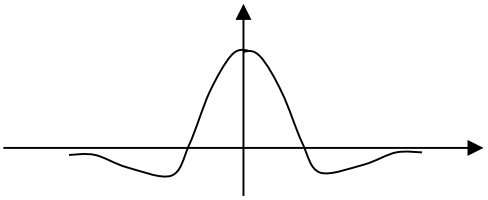
$$4m^2 = 3\Delta^2 i + \Delta^2 j \qquad onde \qquad \Delta i = |i-k| \qquad \Delta j = |j-l|$$

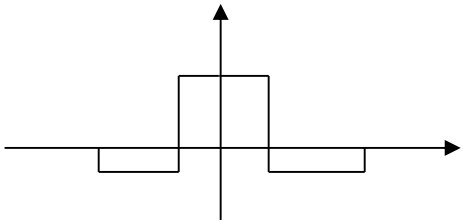
$$\Delta i,\ \Delta j \in [0,2,4,\ldots]$$

6.4 - Outras funções de vizinhança h

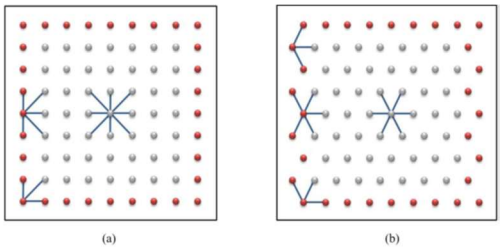


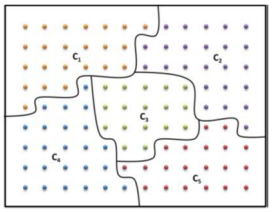






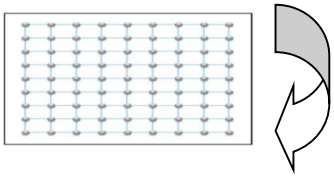
6.5 – Efeito de Borda

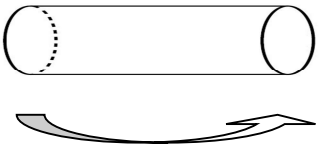


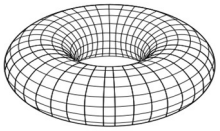


Grades neurais bidimensionais (a) retangular (b) hexagonal (c) SOM

Mapas fechados

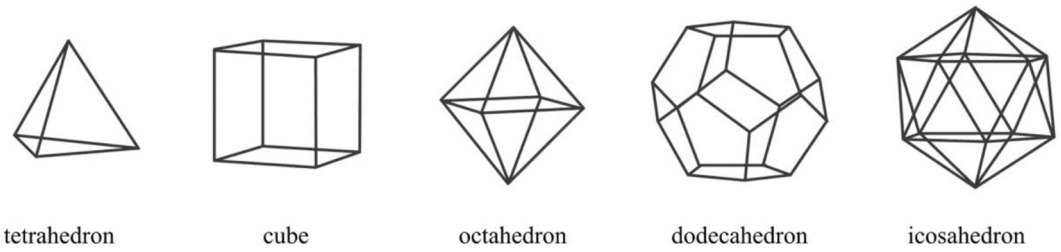




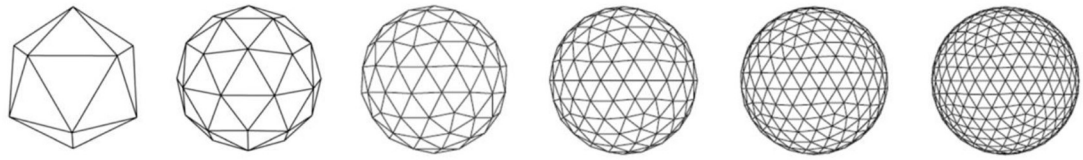


Mapas Esféricos Domo geodésico - GEOSOM

Sólidos Platônicos* e domos geodésicos

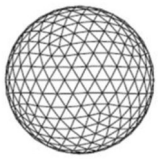


tetrahedron cube octahedron dodecahedron icosahedron

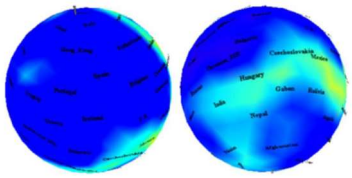


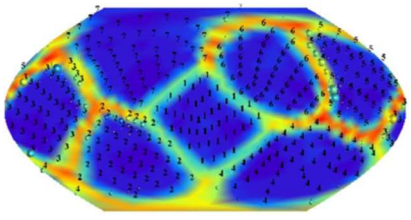
f=1 *f*=2 *f*=3 *f*=4 *f*=5 *f*=6

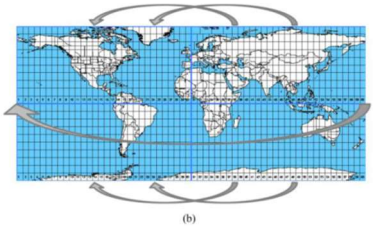
* Obs: Solido platônico é um poliedro convexo em que todas as faces são polígonos congruentes e o mesmo número de arestas encontra-se em todos os vértices. Existem cinco sólidos platônicos: tetraedro, hexaedro (cubo), octaedro, dodecaedro e icosaedro



f=6







(b)