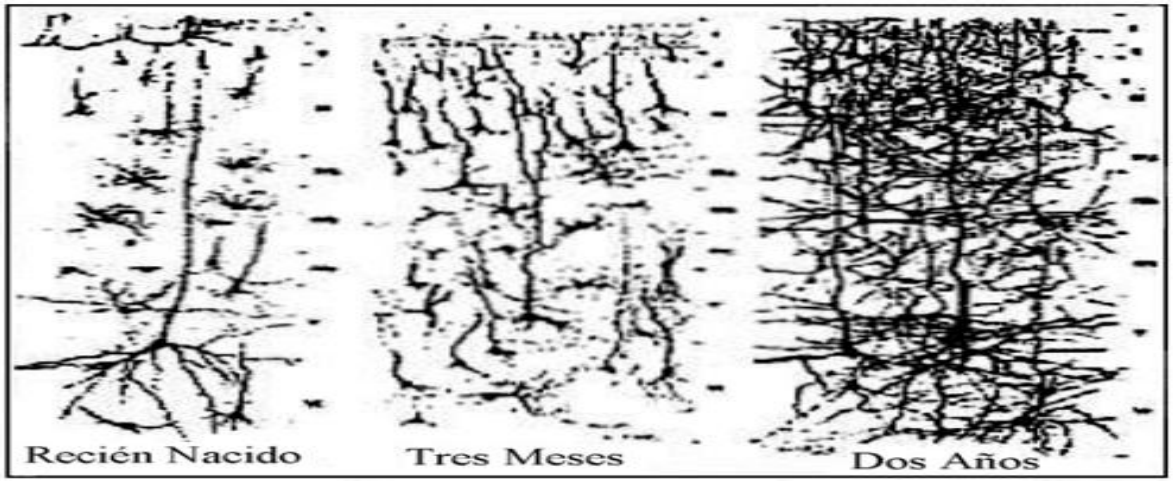
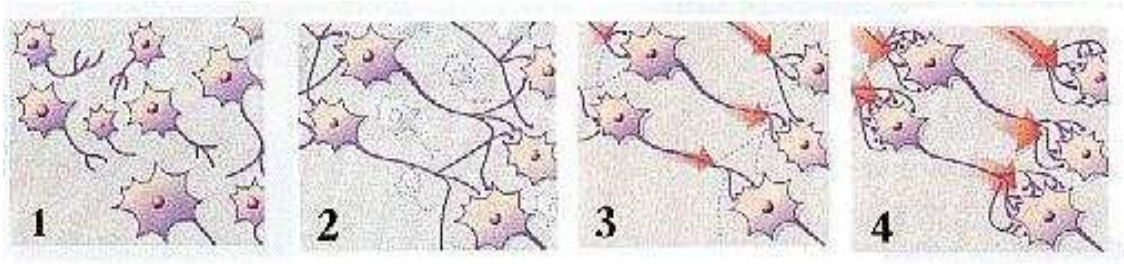
Red neuronal SOM

Jorge Luis Guevara Díaz

Mapas Autoorganizativos





G. M. Edelman premio Nobel de medicina 1972

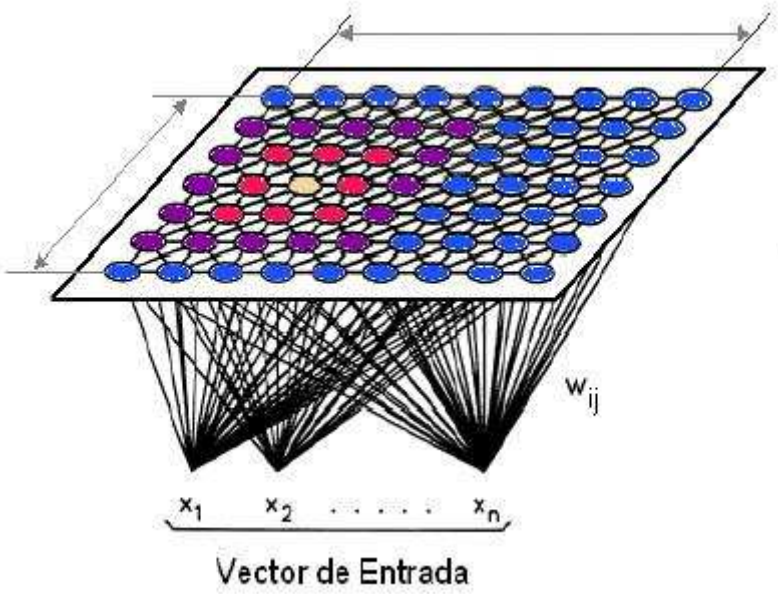
director del Instituto de Neurociencia y presidente del *Neurosciences Research Foundation*

Jorge Luis Guevara Díaz

Mapas Autoorganizativos

- Teuvo Kohonen

$$y_i = -r_i(y_i) + EntradaNeurona_i + \sum_{i=1}^{n} z_{ij} y_j$$



$$\|x - w_c\| = \min_i(\|x - w_i\|)$$

$$w_i(t+1) == \begin{cases} w_i(t) + \alpha(t)h(\mid i-g \mid)(x - w_i(t)) & \text{si } i \epsilon R \\ 0 & si \geq 0 \end{cases}$$

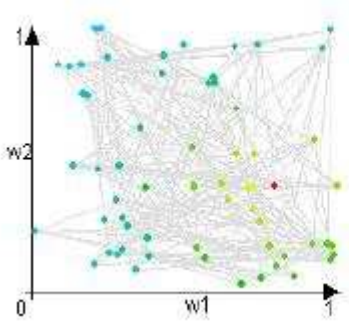
Funcion gaussiana

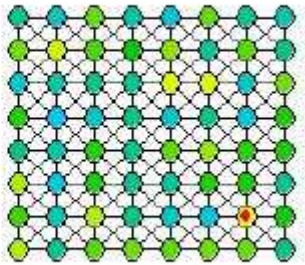
$$h(\mid i-g \mid, t) = e^{-\frac{(\mid i-g \mid)^2}{2R(t)^2}}$$

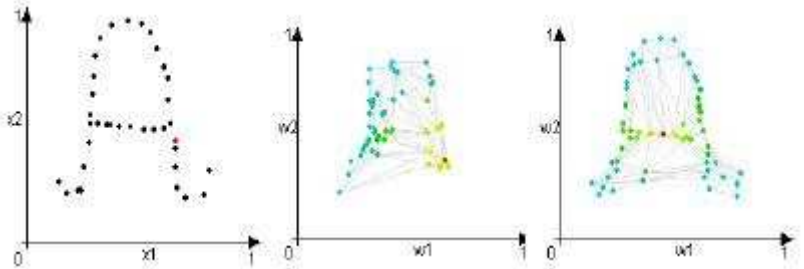
Función triangular

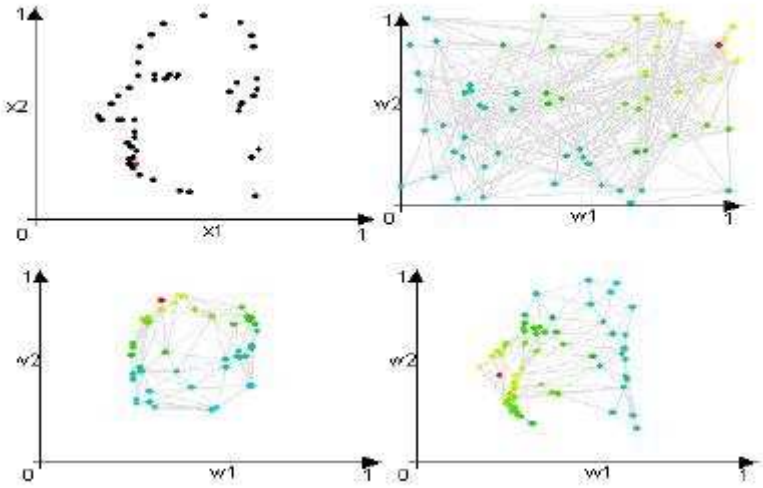
$$h(\mid i-g \mid, t) = \begin{cases} 0 & \text{si} \mid i-g \mid > t \\ \frac{R(t)-\mid i-g \mid}{R(t)} & \text{si} \mid i-g \mid \leq t \end{cases}$$

Mapas Autoorganizativos









Jorge Luis Guevara Díaz

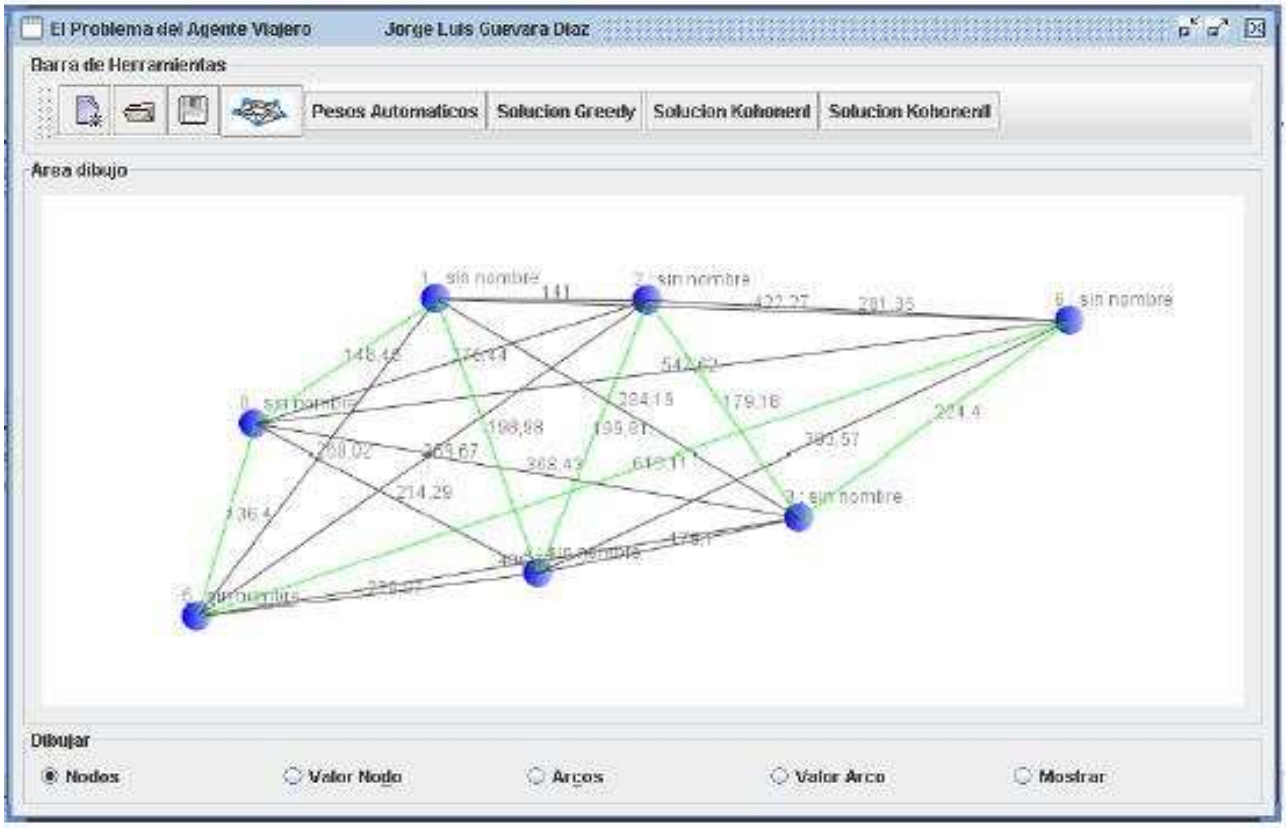
Mapas Autoorganizativos

Algoritmo 1 Aprendizaje Kohonen

Requiere: inicializar pesos $w_{i,j}$ para cada neurona en la capa de competición
  Mientras los pesos $w_{i,j}$ no varíen significativamente **Hacer**
    Para cada patrón de entrada **Hacer**
      presentar el patrón a la red
      obtener la neurona ganadora
      actualizar pesos de la neurona ganadora y de sus vecinos
    Fin Para
    actualizar tasa de aprendizaje y radio de vecindad
  Fin Mientras

Jorge Luis Guevara Díaz

Aplicación de los SOM al Problema del Agente Viajero



Jorge Luis Guevara Díaz

Aplicación de los SOM al Problema del Agente Viajero

PRUEBAS

Numero ciudades	Ruta Greedy	Ruta Kohonen G	Ruta Kohonen T
4	1160	1160	1160
5	1111.6	1111.6	1111.6
6	1285.50	1115.34	1181.98
7	1567.70	1538.50	1567.70
8	1447.71	1540.44	1659.52
9	1667.81	1688.44	1659
10	1696.86	1672.53	1647.89
11	2061.35	1876.05	1958.55
12	1427.03	1469.86	1469.86
13	2173.22	2104.41	2089.77
14	1787.16	1797.55	1773.59
15	2547.85	2370.59	2605.72

Jorge Luis Guevara Díaz

Algoritmo 2 Mapa Autoorganizativo para el Problema del Agente Viajero

Requiere: inicializar pesos $w_{i,j}$ para cada neurona en la capa de competición
Mientras los pesos $w_{i,j}$ no varíen significativamente **Hacer**
Para cada nodo u_i de G(V,E) **Hacer**
presentar la ciudad u_i a la red
obtener la neurona ganadora
Si la neurona ganadora , ha ganado antes para esta pasada **Entonces**
valor$\Leftarrow$ aleatorio
Si $valor < 0{,}5$ **Entonces**
crear neurona a la izquierda
asignar pesos a la neurona creada $w_{n,j} = (0{,}04 * w_{k-1,j} + (0{,}95)w_{k,j}) + (0{,}01\gamma_i)$
Si No
crear neurona a la derecha
asignar pesos a la neurona creada $w_{n,j} = (0{,}04 * w_{k+1,j} + (0{,}95)w_{k,j}) + (0{,}01\gamma_i)$
Fin Si
Fin Si
Si si la neurona no gano para tres iteraciones seguidas **Entonces**
eliminar la neurona
actualizar pesos de la neurona ganadora y de sus vecinos
Fin Si
Fin Para
actualizar tasa de aprendizaje y radio de vecindad
Fin Mientras

$\theta(V * n * iteraciones)$ Diaz