MODELOS COMPUTACIONALES: CUARTO CURSO DEL GRADO DE ING. INFORMÁTICA EN COMPUTACION

MODELOS DE REDES NEURONALES

César Hervás-Martínez
Grupo de Investigación AYRNA

Departamento de Informática y Análisis Numérico
Universidad de Córdoba
Campus de Rabanales. Edificio Einstein.
Email: chervas@uco.es

2019-2020

MODELOS DE REDES NEURONALES

1) REDES DE FUNCIONES DE BASE RADIAL

2) REDES DE HOPFIELD

REDES NEURONALES

DE BASE RADIAL

Redes de funciones de Base Radial

- Se dice que una función es de base radial (RBF) si su salida depende de la distancia del vector de entrada a un vector almacenado dado.

- La red neuronal RBF tiene una capa de entrada, una capa oculta y una capa de salida.

- En este tipo de redes RBF, la capa oculta utiliza las neuronas con funciones de activación de tipo RBF

- Las salidas de todas estas neuronas ocultas se combinan linealmente en el nodo de salida de la capa oculta

Estas redes tienen una amplia variedad de aplicaciones tales como:

Aproximación de funciones,

Predicción de series temporales,

Control y regresión

Tareas de clasificación de patrones en problemas complejos (no lineales).

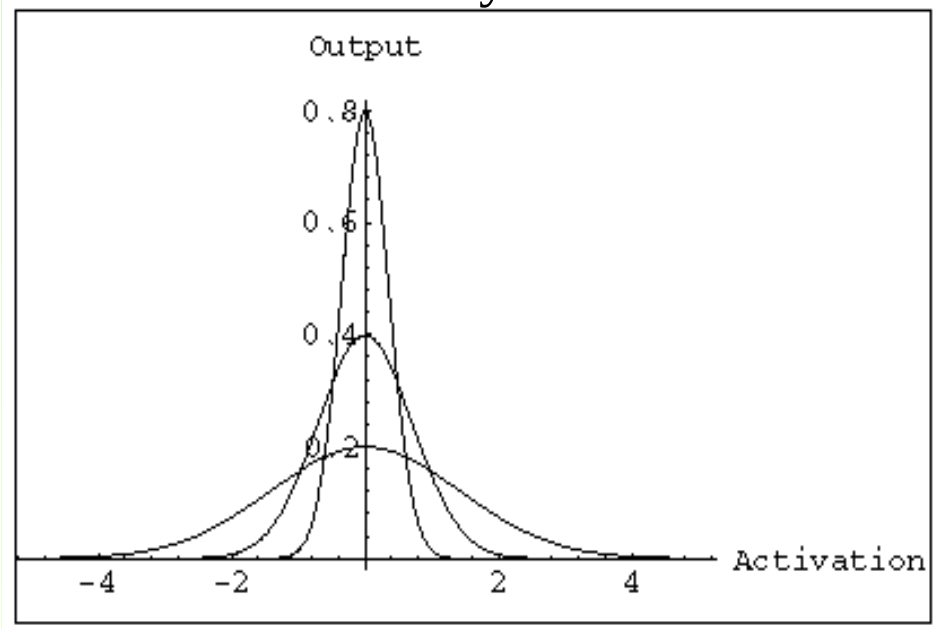
FUNCIONES DE BASE RADIAL

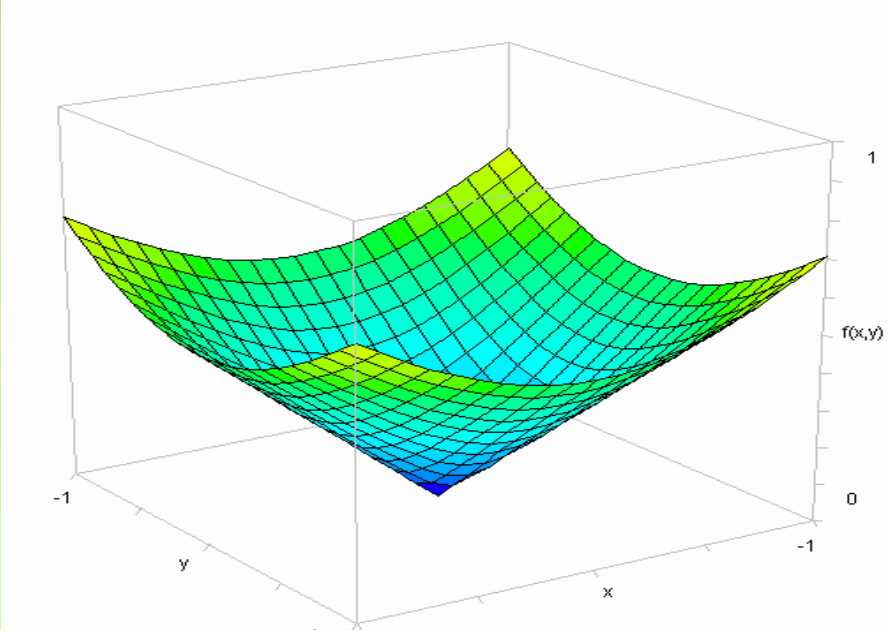
Características

- **Una sola capa oculta**
- **La función de activación de una unidad oculta esta determinada por la distancia entre el vector de entrada y un vector prototipo llamado centroide.**

$$B_j(\mathbf{x};(\mathbf{c}_j \mid r_j)) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{c}_j\|^2}{2r_j^2}\right), \text{ donde } \mathbf{c} \text{ es el centroide}$$

y r es el radio del núcleo o "kernel"





FUNCIONES DE BASE RADIAL

- La capa oculta de la red RBF tiene unas funciones, cada una con un campo de atracción dado por su centroide
- Por lo general esta función es Gaussiana B_j
- La función de la capa de salida, s, es lineal si tenemos un problema de regresión o de clasificación con funciones softmax

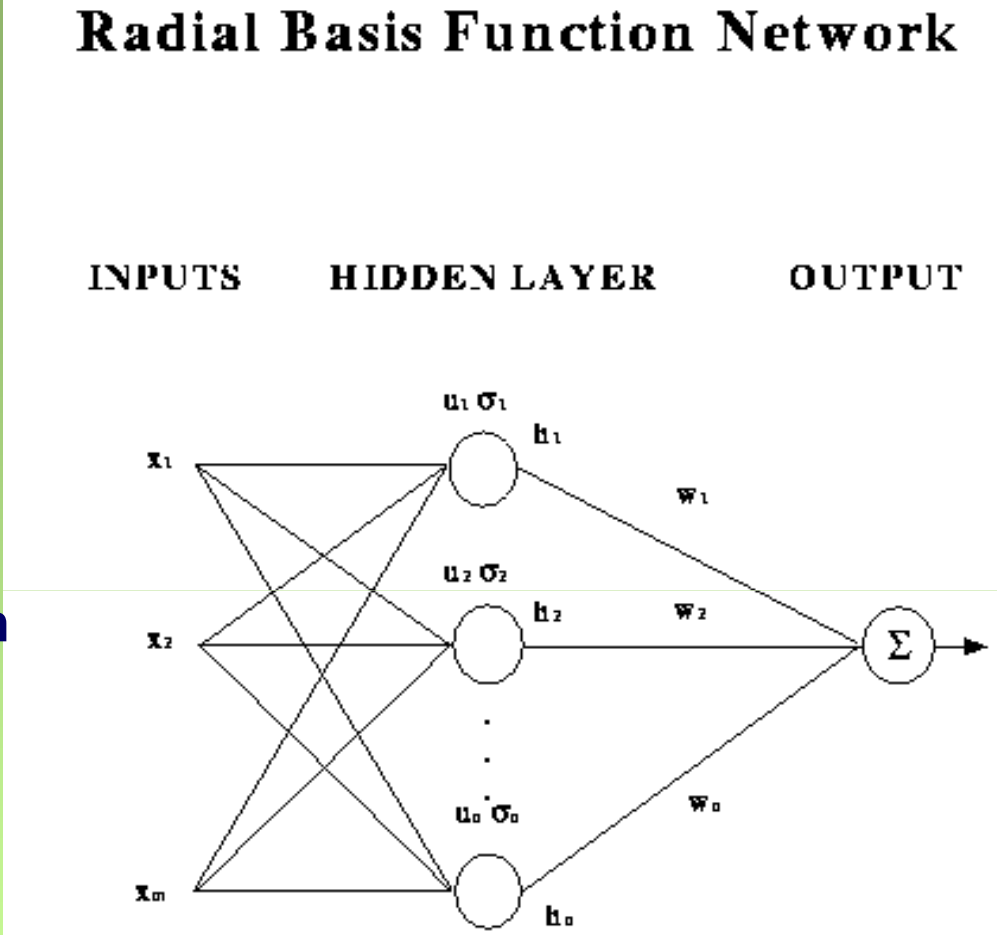
$$s(\mathbf{x}) = \sum_{j=1}^{K} w_j B_j \left(\left\| \mathbf{x} - \mathbf{c_j} \right\| \right)$$

$$B_j \left(\left\| \mathbf{x} - \mathbf{c}_j \right\| \right) = \exp - \frac{1}{2} \left(\frac{\left\| \mathbf{x} - \mathbf{c}_j \right\|}{r_j} \right)^2$$

Redes de funciones de Base Radial

- **Funciones de base radial**
  Las unidades ocultas no almacenan medias y desviaciones típicas, sino en general, centroides y radios.
  Las unidades ocultas calculan una función Gaussiana de las entradas x_1, ... x_n que constituyen el vector de entrada x

- **Aprenden los pesos w_i, los centroides μ_i , y los radios σ_i minimizando la función del error cuadrático (aprendizaje por gradiente descendente)**

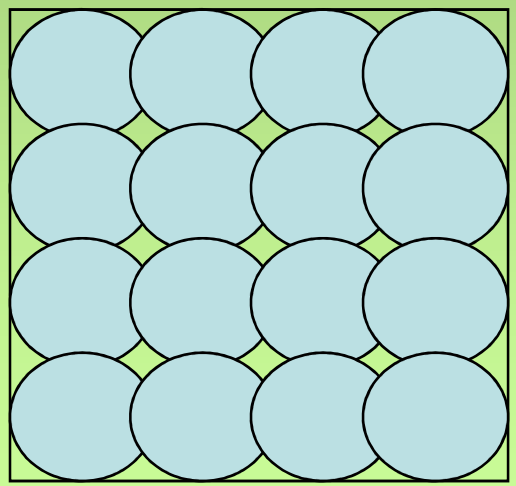


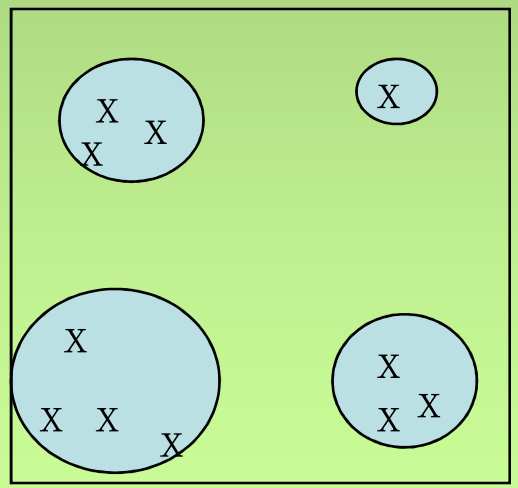
$$h_i = exp[-\frac{(\mathbf{x}-\mathbf{u}_i)^T(\mathbf{x}-\mathbf{u}_i)}{2\sigma^2}],\ y = \sum_i h_i w_i$$

REDES DE FUNCIONES DE BASE RADIAL (RBF)

En la aproximación que vamos a ver se supone tácitamente que se dan las funciones de transferencia de las redes RBF, esto es, se considera que tanto el centoide μ como el radio σ se consideran conocidos. ¿Pero como se eligen μ y σ ?

a) Recubrimiento uniforme





b) Si los patrones de entrenamiento se distribuyen de forma no homogénea entonces hay que hacer en primer lugar un análisis cluster, eligiendo los centroides, c, de las funciones de base para cada cluster y utilizando el tamaño del cluster para definir σ o r

Aprendizaje de las redes RBF mediante clustering particional

Queremos encontrar una asignación de los datos a los k clusters (cada punto a que cluster) y un conjunto de vectores que representen a cada cluster (prototipos o centroides)

Para cada x_j definimos las variables indicador r_{ij} , que valen 1 si x_j es asignado al cluster i y valen 0 en otro caso, i = 1,…, k; j = 1,…,N, siendo N el número de patrones del conjunto de entrenamiento

Problema del clustering: encontrar $\{r_{ij}\}$ y $\{\boldsymbol{\mu}_i\}$ **tal que**

$$\min J = \sum_{j=1}^{N} \sum_{i=1}^{k} r_{ij} \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2 = \sum_{x_j \in Cl_i} \sum_{i=1}^{k} \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2$$

es decir, minimizar la suma ponderada de cuadrados de las distancias de cada objeto (patrón) x_j a su prototipo o centroide $\boldsymbol{\mu}_i$

Procedimiento general

1 Seleccionar k semillas iniciales (centroides de los clusters)

2 Asignar cada objeto al cluster (centroide) más cercano

3 Actualizar los centroides

4 Repetir el proceso hasta que los centroides no cambian (convergencia)

Los distintos métodos difieren sobre todo en los pasos 1 y 3

Redes de funciones de Base Radial

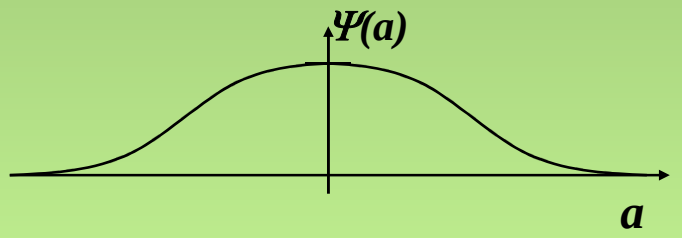
Salida de la red:

$$\text{out}_j = \sum_i w_{i,j} h_i$$

funcion de salida:
(funcion Gaussiana)

h(a)

$$h(a) = e^{-\frac{a^2}{2\sigma^2}}$$

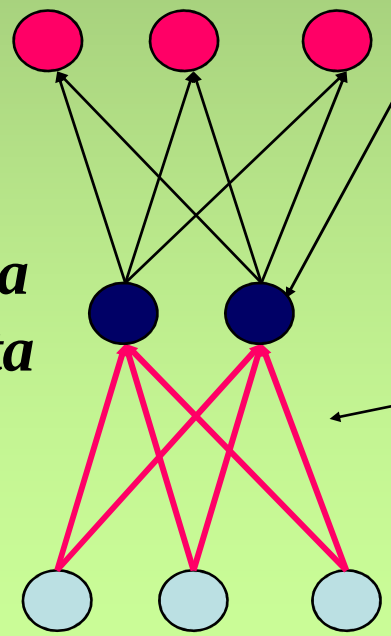


nodos de salida

nodos de la capa oculta

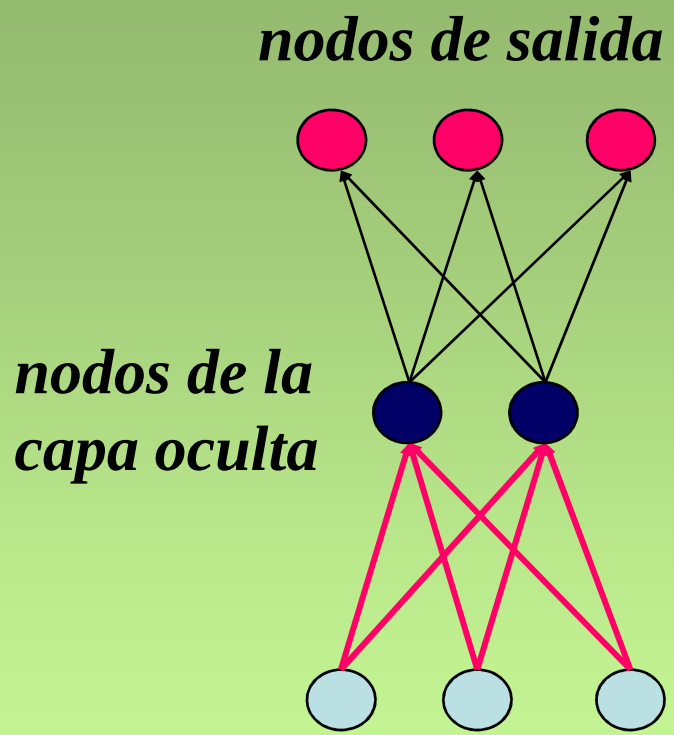
función de propagación:

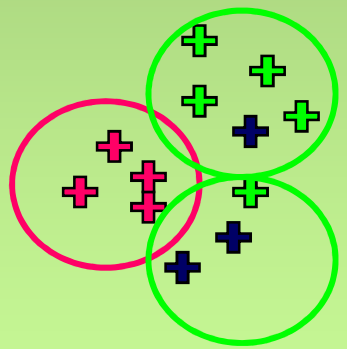
$$a_j = \sqrt{\sum_{i=1}^{n} (x_i - c_{i,j})^2}$$

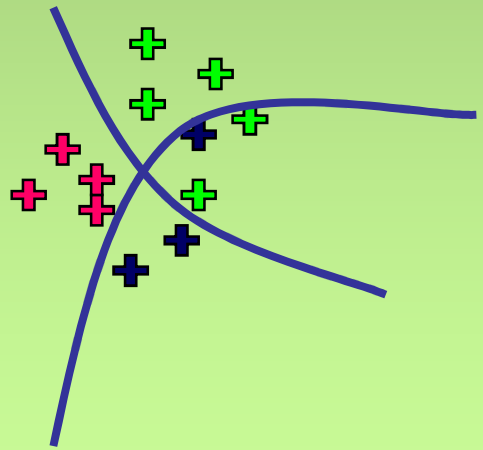


nodos de entrada

MLPs versus RBFs



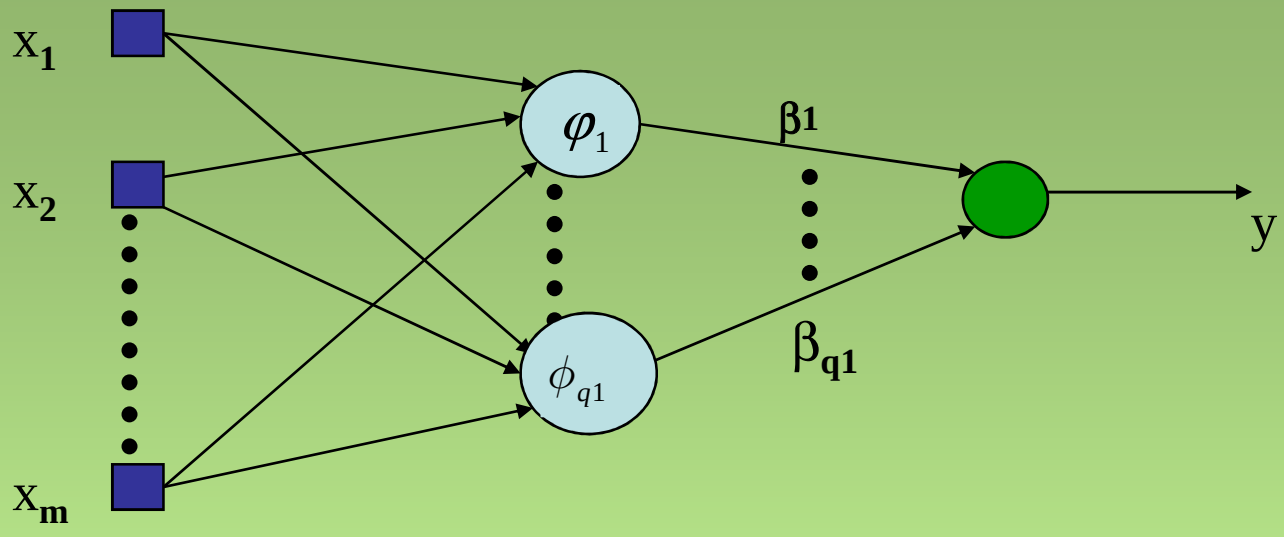




MLPs versus RBFs

- **Clasificación**
  - MLPs separa clases mediante funciones no lineales o hiperplanos
  - RBFs separa clases via hiperesferas
- **Aprendizaje**
  - MLPs usa apendizaje distribuido
  - RBFs usa aprendizaje localizado
  - RBFs entrena más rápidamente
- **Estructura**
  - MLPs tiene una o más capas ocultas
  - RBFs tiene una sola capa oculta
  - RBFs necesita, en general, más neuronas ocultas => problema de “curse of dimensionality” (maldición de la dimensionalidad)

Arquitectura de una red RBF



- **Una capa oculta con funciones de activación de tipo RBF** $\phi_1 \cdots \phi_q$
- **Una capa de salida con función de activación lineal.**

$$y = \beta_1\phi_1(\| \mathbf{x} - \mathbf{c_1} \|) + \ldots + \beta_q\phi_q(\| \mathbf{x} - \mathbf{c_q} \|)$$

$\|\mathbf{x} - \mathbf{c}\|$ distancia de $\mathbf{x}=(x_1,\ldots,x_m)$ al centroide $\mathbf{c}$ de la función

REDES DE FUNCIONES DE BASE RADIAL (RBF)

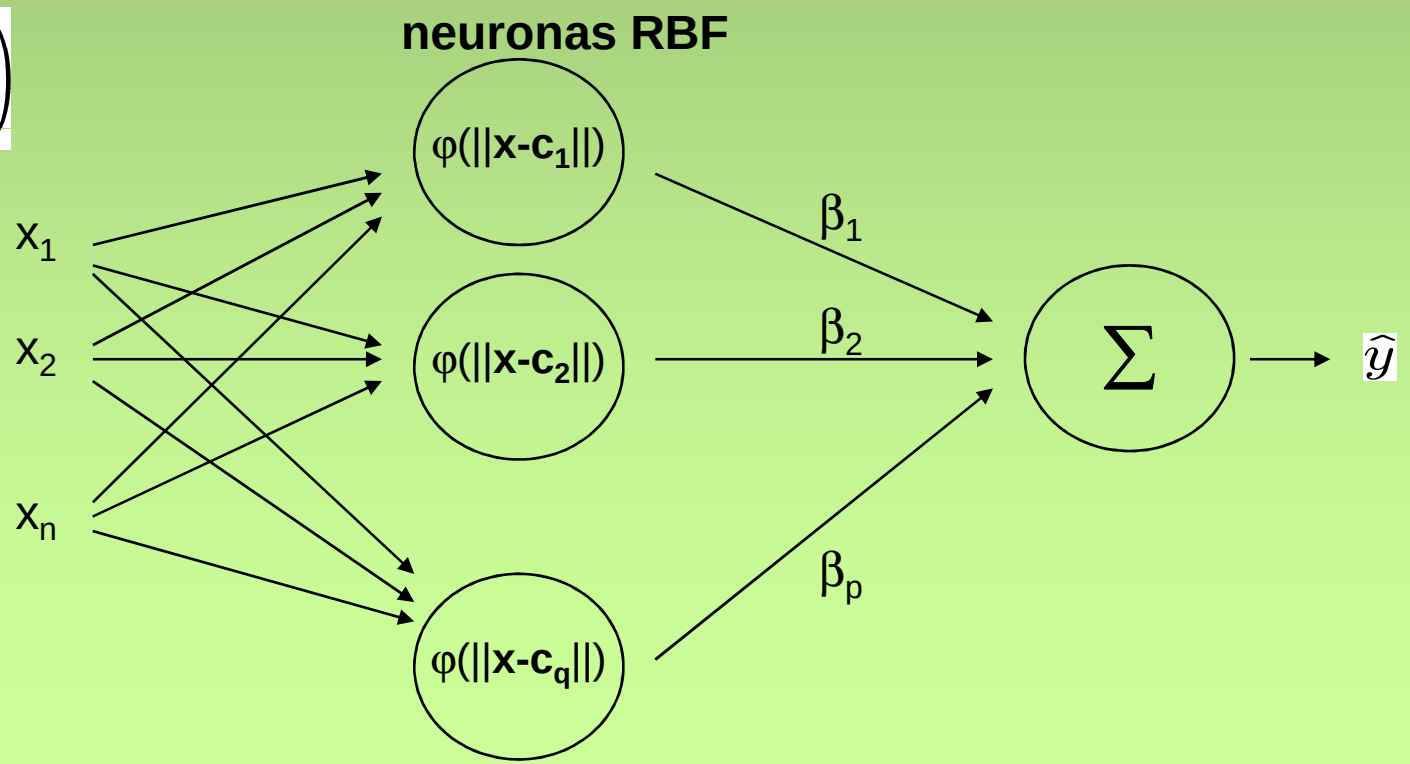
Definicion: Una funcion f: $R^n \rightarrow R$ se dice que es una red de funciones de base radial (RBF) sii

$$f(x) = \beta_1 \varphi(|| x - c_1 ||) + \beta_2 \varphi(|| x - c_2 ||) + ... + \beta_p \varphi(|| x - c_q ||).$$

Donde || x || denota la norma Euclidea del vector x.

Ejemplo función Gaussiana:

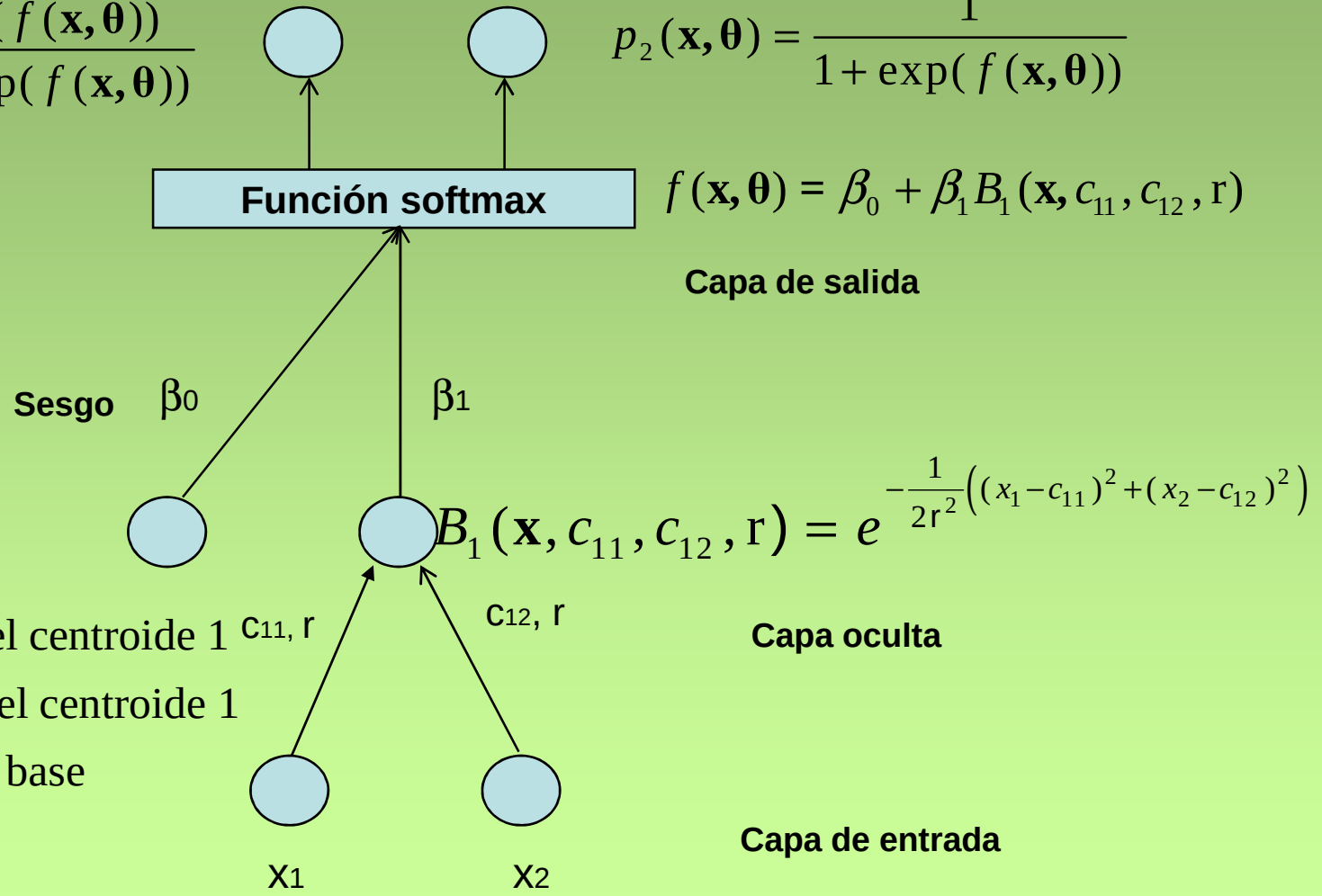
$$\varphi(r) = \exp\left(-\frac{r^2}{\sigma^2}\right)$$



Ejemplo de red neuronal de funciones de base radial: para clasificación binaria

Representación de un modelo

$$p_1(\mathbf{x},\boldsymbol{\theta}) = \frac{\exp(f(\mathbf{x},\boldsymbol{\theta}))}{1+\exp(f(\mathbf{x},\boldsymbol{\theta}))} \qquad p_2(\mathbf{x},\boldsymbol{\theta}) = \frac{1}{1+\exp(f(\mathbf{x},\boldsymbol{\theta}))}$$



c_{11} = primera coordena del centroide 1

c_{12} = segunda coordena del centroide 1

r= radio de la función de base

Arquitectura de una red RBF

Aquí necesitamos solamente aprender los pesos, β_i, de la capa oculta a la capa de salida.

Los pesos β_i se pueden determinar con la ayuda de cualquiera de los métodos iterativos estándar descritos anteriormente para las redes neuronales.

Sin embargo, ya que la función de aproximación dada abajo es lineal con respecto a β_i, se puede calcular directamente utilizando matrices asociadas a métodos lineales de mínimos cuadrados sin tener que determinar explícitamente β_i de forma iterativa.

Cabe señalar que la función f(x) es diferenciable con respecto a β_i.

$$Y = f(\mathbf{x}) = \sum_{i=1}^{N} \beta_i \varphi(\|\mathbf{x}_i - \mathbf{c}_i\|)$$

REDES DE FUNCIONES DE BASE RADIAL

Dados: **N patrones de entrenamiento (x_i, y_i) y q neuronas RBF**

Encontrar: **pesos β_1, ..., β_q con minimo error**

Solución:

Sabemos que $f(x_i) = y_i$ para i = 1, ..., N o de forma equivalente

$$\sum_{k=1}^{q} \beta_k \cdot \underbrace{\varphi\left(\left\|\mathbf{x}_i - \mathbf{c}_k\right\|\right)}_{\varphi_{ik}} = y_i, \quad \text{para } i = 1,\ldots,N$$

- β_k: **desconocido**
- φ_{ik}: **Valor conocido**
- y_i: **valor conocido**

$$\sum_{k=1}^{q} \beta_k \cdot \varphi_{ik} = y_i, \quad \text{para } i = 1,\ldots,N$$

Tenemos, por tanto, N ecuaciones lineales con q incognitas

REDES DE FUNCIONES DE BASE RADIAL (RBF)

$$\sum_{k=1}^{q} \beta_k \cdot \varphi_{ik} = y_i, \text{ para i=1,...,N}$$

Tenemos N patrones y q funciones de base radial

En forma matricial : $\mathbf{P}_{N\times q}\boldsymbol{\beta}_{q\times 1} = \mathbf{y}_{N\times 1}$, donde $\mathbf{P} \equiv (p_{ik})$

Caso N=q; entonces $\boldsymbol{\beta} = \mathbf{P}^{-1}\mathbf{y}$, si $\mathbf{P}$ es de rango completo

Caso N<q; muchas soluciones, pero entonces no tiene ninguna relevancia práctica y los pesos habría que optimizarlos mediante una heurística

Caso N>q; entonces $\boldsymbol{\beta} = \mathbf{P}^{+}\mathbf{y}$,

donde $\mathbf{P}^{+}$ es la matriz seudo inversa de Moore Penrose

En la ecuación $\mathbf{P\beta = y}$, si multiplicamos por $\mathbf{P^T}$ por la izquierda tenemos $\mathbf{P^TP\beta = P^Ty}$; y si ahora multiplicamos por $\mathbf{(P^TP)^{-1}}$ también por la izquierda tenemos $\mathbf{(P^TP)^{-1}P^TP\beta = (P^TP)^{-1}P^Ty}$, donde $\mathbf{(P^TP)^{-1}P^TP = I}$, *y* $\mathbf{(P^TP)^{-1}P^T}$ es la matriz de Moore Penrose, $\mathbf{P}^{+}$

Complejidad (simple)

$$\beta = (\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{y} = \mathbf{P}^+\mathbf{y}$$

Para el producto matricial $\mathbf{P}^T\mathbf{P}$: la complejidad es N^2q; para la inversión q^3, para el producto de matriz por vector $\mathbf{P}^T\mathbf{y}$: qN y para la multiplicación q^2, por tanto la complejidad es $O(N^2q)$

Observación: Si N, el número de patrones de entrenamiento, es grande entonces el calculo de $(\mathbf{P}^T\mathbf{P})^{-1}$ será probablemente inexacto. Si la solución es analítica, entonces utilizaremos el algoritmo de gradiente descendente a partir de esta solución. Pero para ello se necesitan funciones de base diferenciables.

REDES DE FUNCIONES DE BASE RADIAL (RBF)

Ventajas:

- **Patrones de entrenamiento adicionales implican sólo un ajuste local de los pesos.**
- **Pesos óptimos obtenidos en tiempo polinómico.**
- **Las regiones no reconocidas por la red RBF pueden ser identificadas por salidas iguales a cero.**

Inconvenientes:

- **El número de neuronas aumenta exponencialmente con la dimensión del espacio de entrada.**
- **No se puede extrapolar (ya que no hay centros fuera del entorno de los patrones de entrenamiento y las redes RBFs son locales).**

Comparación de redes RBF y MLP

	MLP
Redes con capas no lineales con activación hacia adelante	Redes con capas no lineales con activación hacia adelante
La capa oculta de la red RBF es no-lineal, la capa de salida es lineal.	Las capas oculta y de salida de una red MLP son por lo general no-lineales, a no ser que utilicemos la función softmax.
Una sola capa oculta.	Puede tener más de una capa oculta.
El modelo de neurona de la capa oculta es diferente del modelo de los nodos de la capa de salida.	Las neuronas ocultas y de salida comparten, por lo general, el mismo modelo de neurona.
La función de activación de cada neurona de la capa oculta en una RBF calcula la distancia euclidea entre el vector de entrada y el centro de esa unidad.	La función de activación de cada neurona de la capa oculta en una MLP calcula el producto interno del vector de entrada y del vector de pesos sinápticos de esa neurona.

MODELOS COMPUTACIONALES: CUARTO CURSO DEL GRADO DE ING. INFORMÁTICA EN COMPUTACION

REDES DE HOPFIELD

César Hervás-Martínez
Grupo de Investigación AYRNA

Departamento de Informática y Análisis Numérico
Universidad de Córdoba
Campus de Rabanales. Edificio Einstein.
Email: chervas@uco.es

2019-2020

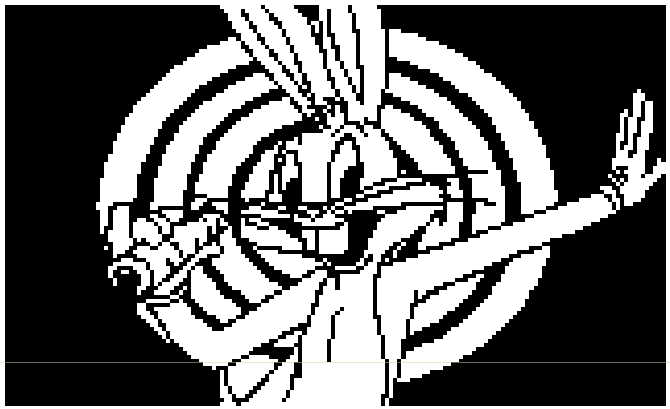
Redes de Hopfield

La importancia de las redes de Hopfield en la práctica es limitada debido a las restricciones teóricas de la estructura, pero, en algunos casos, se pueden formar modelos interesantes.

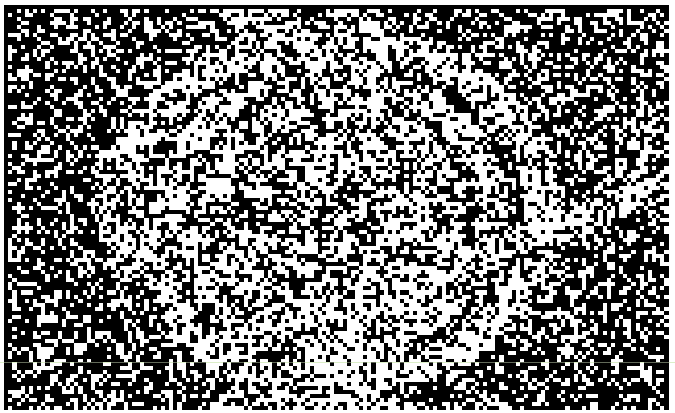
Redes de Hopfield

Por lo general los modelos de Redes de Hopfield, trabajan en tareas de lógica binaria: Por ejemplo, reconstrucción de patrones y reglas de asociación

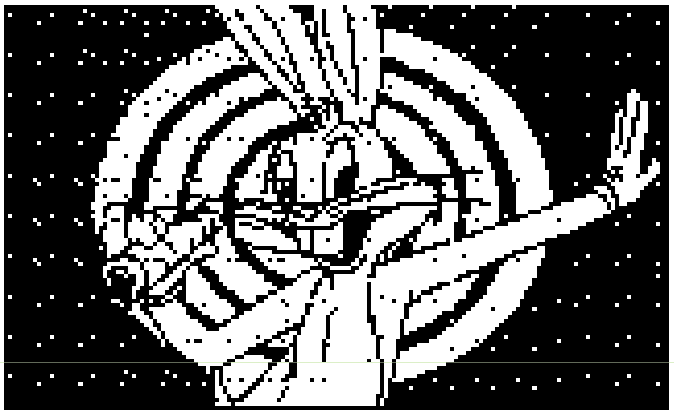
Original

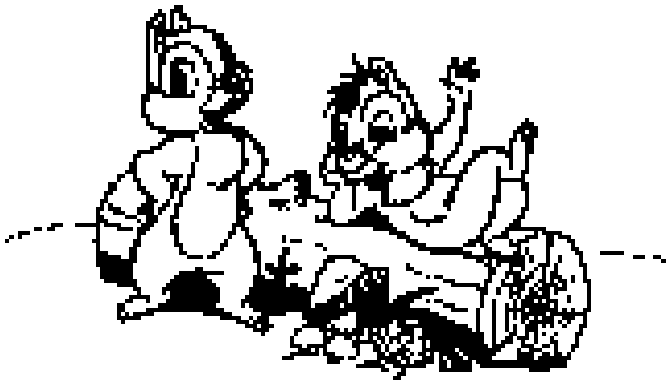


Degraded



Reconstruction





La red de Hopfield reconstruye imágenes degradadas por ruido (ver imagen superior) o por señales parciales (ver imagen inferior)

Red de Hopfield

Una red de Hopfield es un modelo de memoria asociativa.

Se basa en el aprendizaje hebbiano pero usa neuronas binarias.

Proporciona un modelo formal que puede analizarse para determinar la capacidad de almacenamiento de la red.

Está inspirado en su formulación por modelos de mecánica estadística (modelo de Ising) para materiales magnéticos.

Proporciona un método para generalizar modelos de redes deterministas para el caso estocástico.

Red de Hopfield

- Una red de Hopfield es un tipo de red monocapa recurrente cuyos valores de salida son realimentados, de nuevo, a la entrada de una manera no dirigida.
- Esta asociada a resolver problemas de memoria asociativa.
- Se trata de un aprendizaje no supervisado donde se agrupan los patrones por similaridad.
- Se compone de un conjunto de N neuronas conectadas con pesos que son simétricos y ninguna unidad está conectada a sí misma.
- No hay neuronas especiales de entrada y de salida.
- La activación de una neurona es un valor binario decidido por el signo de la suma ponderada de las conexiones de entrada a la misma.

Red de Hopfield

- Un valor umbral para cada neurona determina si está una neurona activa.
- Una neurona activa es una que activa todas las neuronas que están conectadas a ella con un peso positivo.
- La entrada se aplica simultáneamente a todas las neuronas, la cual produce una salida en cada una.
- Este proceso continúa hasta que se alcanza un estado estable.

Red de Hopfield

El problema de la memoria asociativa se resume de la siguiente manera:

Almacena un conjunto de patrones de tal forma que cuando se le presenta un nuevo patrón, la red responde produciendo uno de los patrones almacenados que más se parezca a el.

Tanto los vectores, de los patrones almacenados, como los patrones de prueba, se codifican mediante -1 (estado no activo) o 1 (estado activo) en cada una de sus componentes.

Una memoria asociativa puede pensarse como un conjunto de atractores, cada uno con su propio nicho o cuenca de atracción.

Redes de Hopfield

Al igual que todos los computadores, un cerebro es un sistema dinámico que lleva a cabo sus cálculos por el cambio de su "estado" a lo largo del tiempo.

En los modelos simples de la dinámica de los circuitos neuronales se describe que tienen propiedades dinámicas colectivas.

Estas pueden ser explotadas en el reconocimiento de patrones sensoriales.

La utilización de estas propiedades colectivas de procesamiento de la información es eficaz, en tanto en cuanto, aprovechen las propiedades espontáneas de las células nerviosas y de los circuitos para producir un cálculo robusto.

Algoritmo de activacion

Repetir

Elija cualquier unidad al azar. La unidad elegida puede estar activa o inactiva.

- **Para la unidad elegida, calcular la suma de los pesos de las conexiones sólo a los vecinos activos, si los hay.**
- **Si la suma es > 0 (se supone que el umbral es 0), entonces la unidad elegida se convierte en activa, de lo contrario, se vuelve inactiva.**
- **Si la unidad elegida no tiene vecinos activos entonces se ignora, y el estado sigue siendo igual.**
- **Hasta que la red alcanza un estado estable, esto es, un mínimo de la función de energía.**

Redes de Hopfield

La búsqueda de J. Hopfield

Mientras que el cerebro es totalmente diferente a los ordenadores actuales, gran parte de lo que hace puede ser descrito como computación.

Los seres humanos mediante la memoria asociativa, la lógica y la deducción, reconocen un olor o una posición de ajedrez, analizan el mundo a partir de objetos, y generan secuencias apropiadas de mandatos musculares del aparato locomotor que son todos descritos como cálculo.

La investigación de Hopfield se centra en el entendimiento de cómo los circuitos neuronales del cerebro producen cálculos potentes y complejos.

Redes de Hopfield

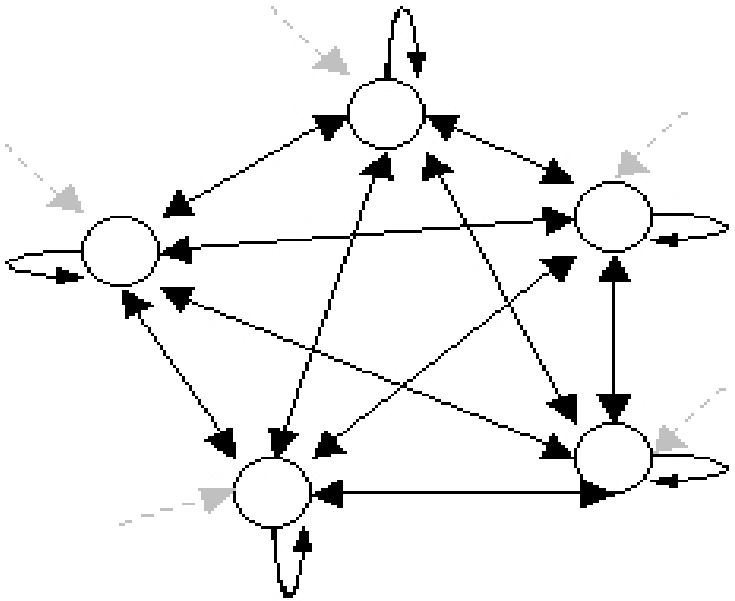
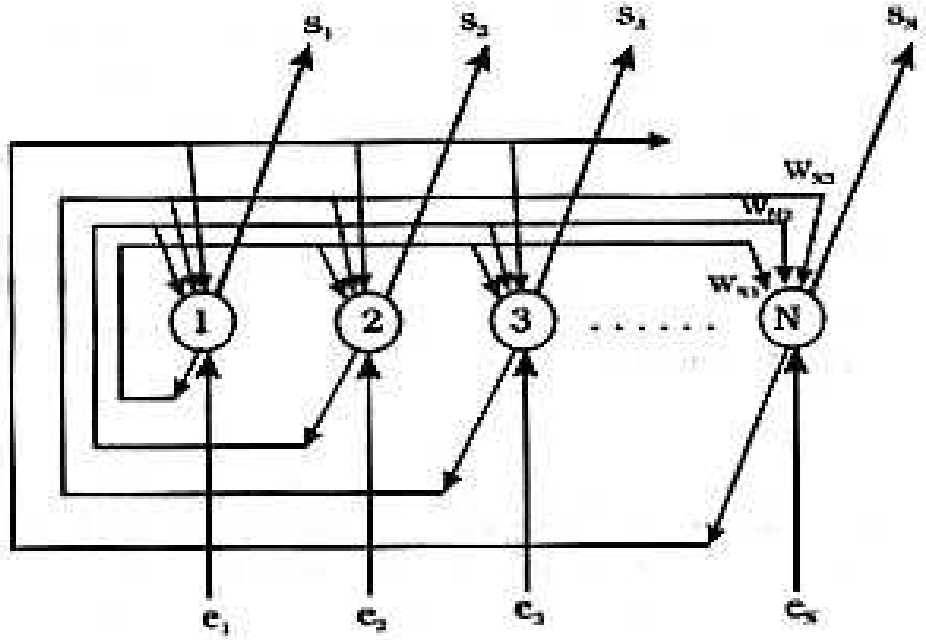


Figura 8: Arquitectura de una Red de Hopfield



Red Hopfield

Cálculo del potencial de acción

Para gran parte de la neurobiología, la información está representada por el paradigma de las “tasas de disparo", es decir, se representa la información por la tasa de generación de picos de potenciales de acción, donde el momento exacto de estos picos es poco importante.

Calculo del potencial de acción

Dado que los potenciales de acción duran sólo alrededor de una milésima de segundo, el uso del instante del potencial de acción aparece como un medio poderoso de la computación neuronal.

Hay casos, como por ejemplo en la determinación auditiva fonética binaria de la ubicación de una fuente de sonido, donde la información se codifica en el momento de los potenciales de acción.

Redes de Hopfield

- Actúan como las memorias "autoasociativas" para almacenar patrones
  - Son neuronas de tipo McCulloch-Pitts con salidas -1 o 1, y umbral Θ, en este caso 0

Red completamente conectada

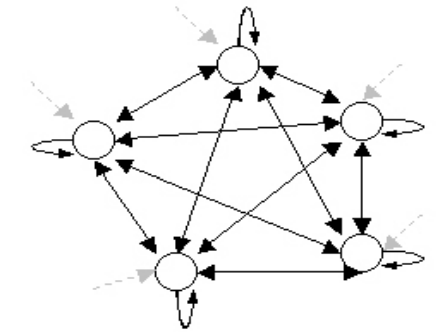


Figura 8: Arquitectura de una Red de Hopfield

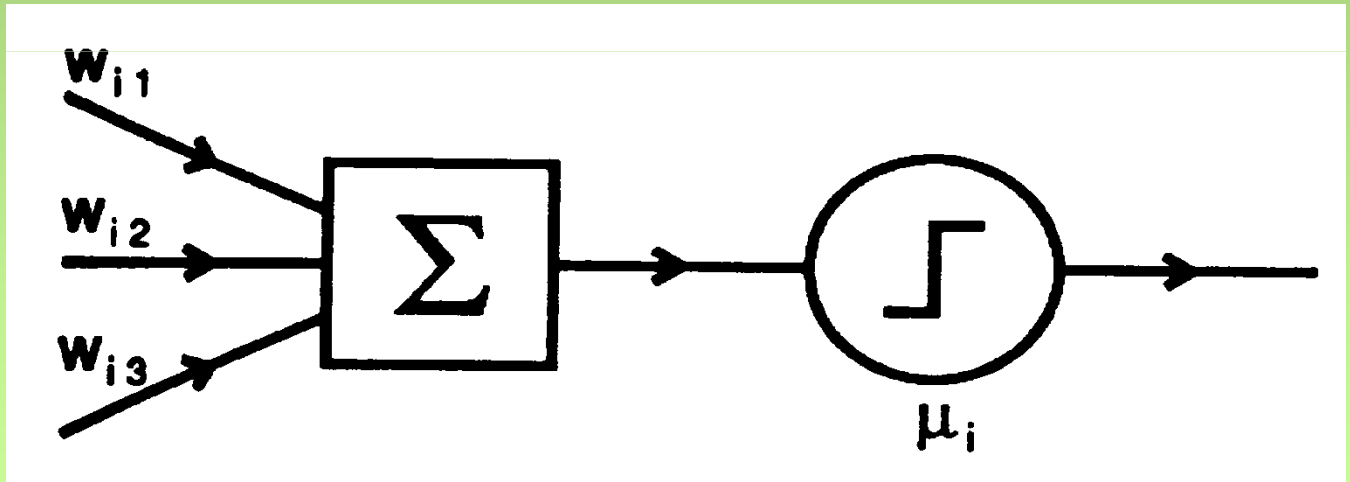
  - Todas las neuronas están conectadas entre si
    - Pesos simétricos ($w_{ij} = w_{ji}$) y $w_{ii} = 0$
    - Cambio asíncrono de las salidas
    - Sea s_i el estado de la unidad i
    - En cada etapa de tiempo se elige una unidad al azar
    - Definimos

$s_i = 1$ si $\sum_j w_{ij} s_j \geq \Theta$, para todo nodo j **activo** conectado al nodo i

$s_i = -1$ en otro caso

Neurona de McCulloch–Pitts (1943)

- **Atributos de la neurona genérica**
  - **m entradas binarias y una salida (0 o 1)**
  - **Pesos sinápticos w_{ij}**
  - **Umbral μ_i**



Redes de Hopfield

- Hopfield demostró que la actualización asíncrona en redes simétricas minimiza una función de "energía" y conduce a un estado final estable, un mínimo de la función de energía, para un estado inicial dado.

- Se define una función de energía a minimizar

$$E = -\frac{1}{2}\sum_i \sum_j w_{ij} s_i s_j + \sum_i s_i \Theta_i$$

Redes de Hopfield

Supongamos que una unidad al azar ha sido actualizada:

la función de energía E siempre decrece!

$$E(t) = -\frac{1}{2}\sum_{i,j} w_{ij} s_i s_j + \sum_i s_i \Theta_i$$

- **Si s_i es inicialmente –1 (inactiva) y $\sum_j w_{ij} s_j > \Theta_i$, esto es $-\sum_j w_{ij} s_j < -\Theta_i$ entonces s_i se convierte en +1 (activa)**
  - **Al pasar a ser s_i=1, hay un cambio en el valor inicial de E, en la forma, dado que $w_{ij} = w_{ji}$**

$$E(t+1) - E(t) = -\frac{1}{2} 2\sum_j (w_{ij} s_j - w_{ji}(-s_j)) + \Theta_i - (-\Theta_i) =$$

$$= -\frac{1}{2} 2\sum_j (w_{ij} s_j + w_{ji} s_j) + 2\Theta_i = 2(-\sum_j w_{ij} s_j + \Theta_i) < 0 \quad !!$$

- **Si s_i es inicialmente +1 y $\sum_j w_{ij} s_j < \Theta_i$ entonces s_i se convierte en -1**
  - **Al pasar a ser s_i=-1, hay un cambio en E**

$$E(t+1) - E(t) = \frac{1}{2} 2\sum_j (w_{ij} s_j + w_{ji} s_j) - 2\Theta_i = 2(\sum_j w_{ij} s_j - \Theta_i) < 0 \quad !!$$

Redes de Hopfield: Ejemplo

- Supongamos que tenemos sólo dos unidades y que s1(t)=-1 y s1(t+1)=1

$$E(t) = -\frac{1}{2} w_{12} s_1 s_2 - \frac{1}{2} w_{21} s_2 s_1 + s_1 \Theta_1 + s_2 \Theta_2 ; \text{ para } s_1 = -1, \text{ tenemos}$$

$$E(t) = +\frac{1}{2} w_{12} s_2 + \frac{1}{2} w_{21} s_2 - \Theta_1 + s_2 \Theta_2 = w_{21} s_2 - \Theta_1 + s_2 \Theta_2$$

- Si s_1 es inicialmente –1 (inactiva) y $w_{12} s_2 > \Theta_1$ entonces s_1 se convierte en +1 (activa)
  - Al pasar a ser $s_{1(t+1)}=1$, hay un cambio en el valor inicial de E, en la forma, dado que $w_{ij} = w_{ji}$

$$\text{para } s_1(t+1)=1,\ E(t+1) = -\frac{1}{2} w_{12} s_2 - \frac{1}{2} w_{21} s_2 + \Theta_1 + s_2 \Theta_2 =$$

$$= -w_{21} s_2 + \Theta_1 + s_2 \Theta_2$$

$$E(t+1) - E(t) = -w_{21} s_2 - w_{21} s_2 + 2\,\Theta_1 = 2(-w_{21} s_2 + \Theta_1) < 0 \quad !!$$

Redes de Hopfield: Ejemplo

- **Supongamos ahora que tenemos sólo dos unidades y que s1(t)=1 y s1(t+1)=-1**

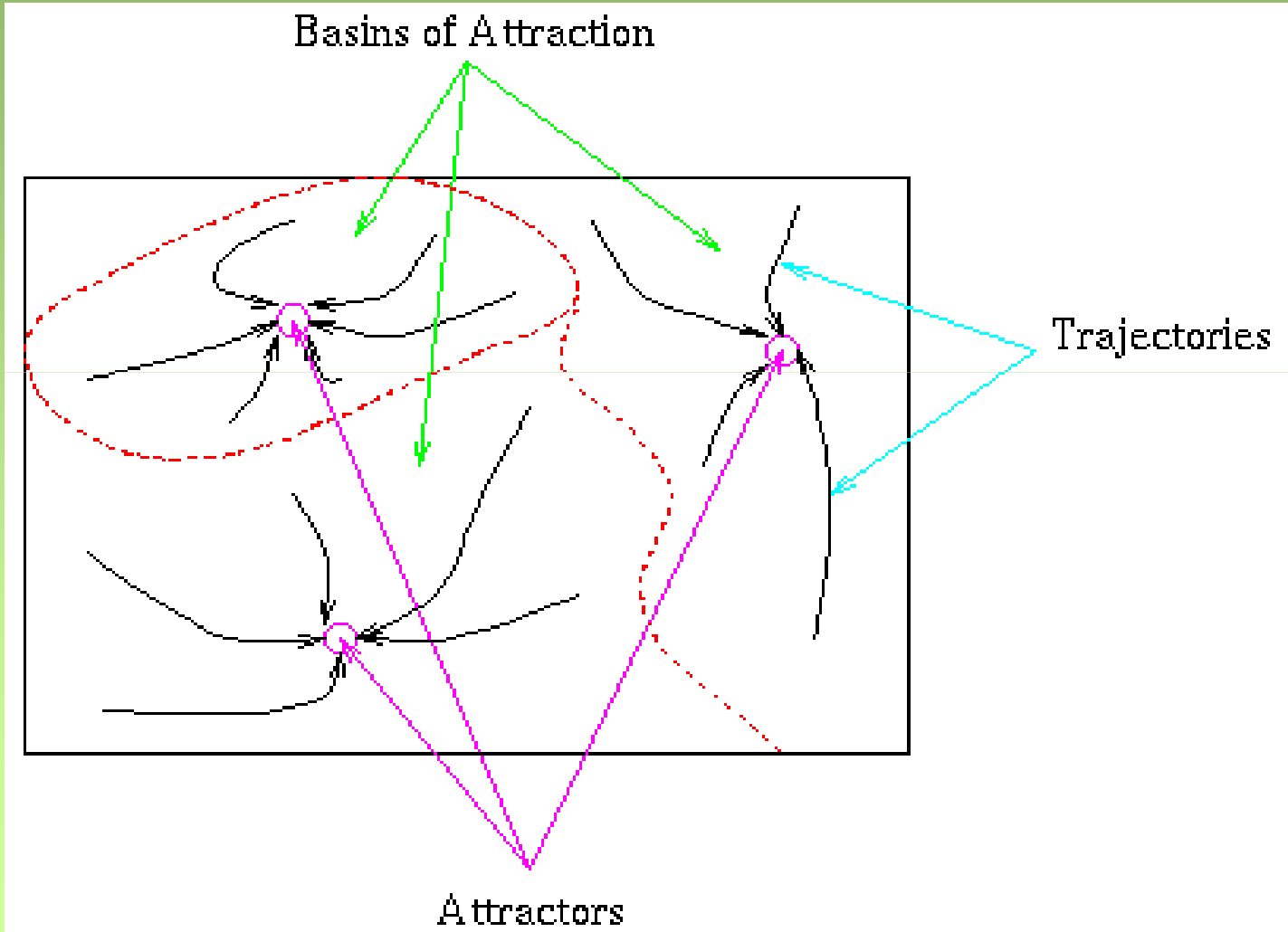
Si s_1 es inicialmente 1 (activa) y $w_{12}s_2 < \Theta_1$ entonces s_1 se convierte en -1 (inactiva), en general

$$E(t+1)-E(t)=\frac{1}{2}2\sum_j(w_{ij}s_j+w_{ji}s_j)-2\Theta_i=2(\sum_j w_{ij}s_j-\Theta_i)<0 \quad !!$$

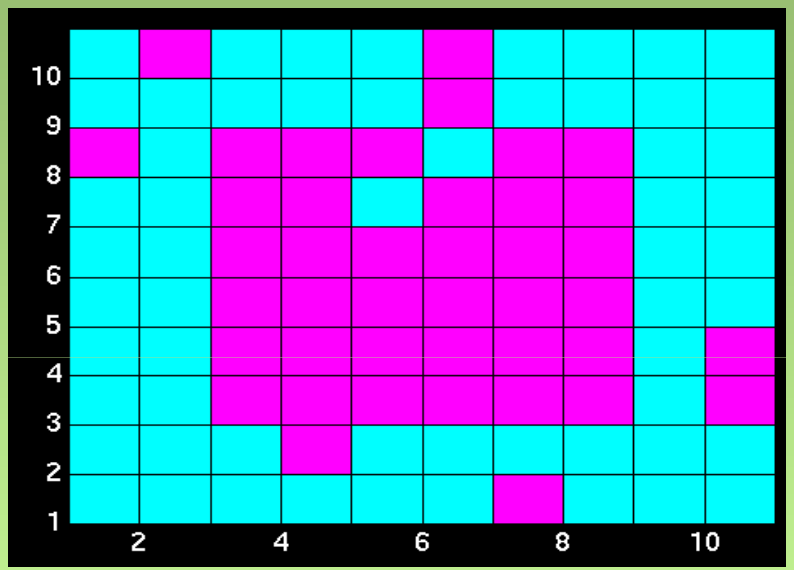
Y en particular

$$E(t+1)-E(t)=w_{21}s_2+w_{21}s_2\text{-}2\,\Theta_1=2(w_{21}s_2-\Theta_1)<0 \quad !!$$

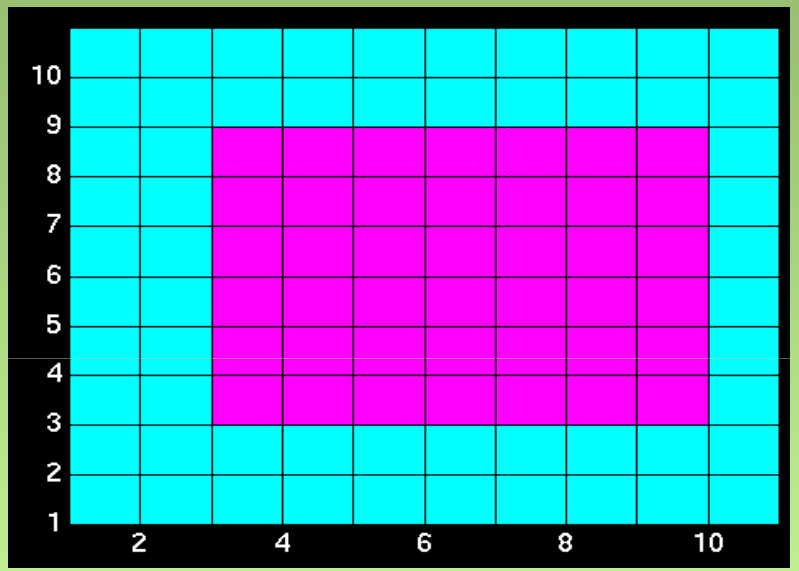
El concepto



Completar un patrón en una red de Hopfield



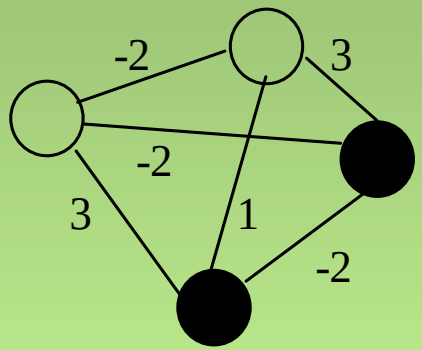
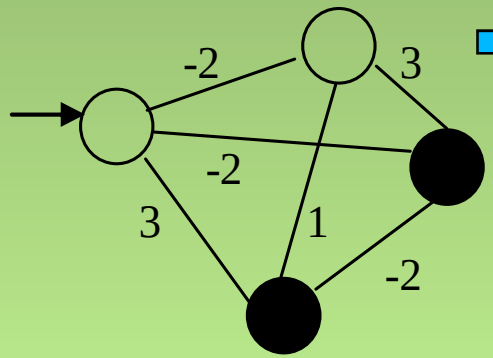
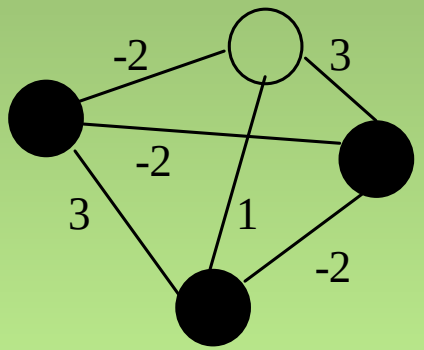
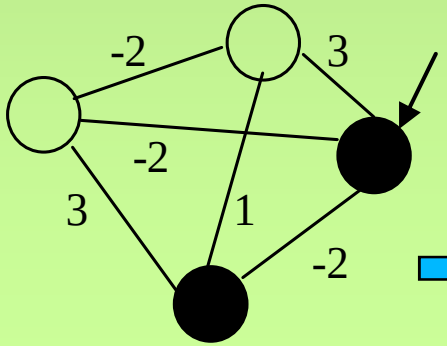
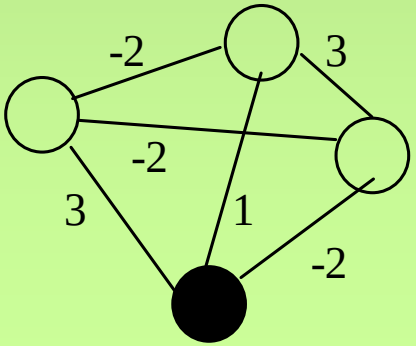
→



Mínimo local (“attractor”) de la función de energía del patrón almacenado

Ejemplo 1

Desactivada Activada

Estado actual	→ Seleccionamos una unidad a partir del estado actual	Correspondiente nuevo estado
 Aquí, se calcula la suma de pesos de las unidades activas de la unidad seleccionada		 Suma = 3 – 2 = 1 > 0; activada
		 Suma = –2 < 0; desactivada

Redes Estables: Ejemplo 2

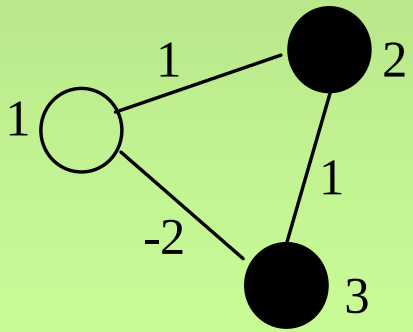
Para la unidad elegida, se calcula la suma de los pesos de las conexiones sólo a los vecinos activos, si los hay.

Si la suma es > 0, entonces la unidad elegida se convierte en activa, de lo contrario, se vuelve inactiva

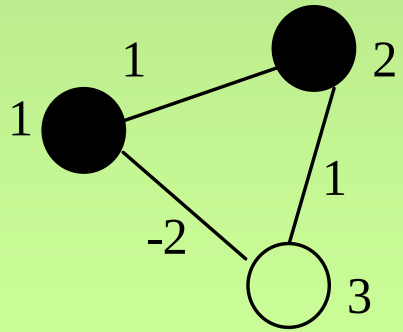
Si la unidad elegida no tiene vecinos activos entonces se ignora, y el estado sigue siendo igual.

Desactivada **Activada**

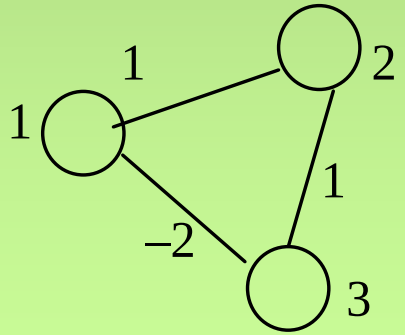
3 Estados estables



X = [0 1 1], para los nodos 1, 2 y 3



X =[1 1 0]



X =[0 0 0]

Red de Hopfield: Método de calculo de los pesos

- **Nota:** La red converge a un mínimo local el cual almacena diferentes patrones.

Almacena M vectores N-dimensionales de patrones de entrada, **a saber,** $\mathbf{x_1}, \ldots, \mathbf{x_M}$

Asi M **es el nº de patrones, y utilizamos la regla de aprendizaje de Hebb para obtener los pesos de la red, para N nodos de la red:**

$$w_{ij} = \begin{cases} \frac{1}{N}\sum_{m=1}^{M} x_{mj}\, x_{mi} & \text{para todo } j \neq i, \text{ donde } i,j=1,\ldots,N \\ 0 & \text{para } j=i \end{cases}$$

$$\mathbf{W} = \frac{1}{N}\sum_{m=1}^{M} \mathbf{x_m}\mathbf{x_m^T}, \text{ donde T es la transpuesta del vector}$$

Metodo de computación de pesos

- **Los pesos son determinados utilizando las** M **muestras de entrenamiento.**

$$\mathbf{W} = \sum_{i=1}^{M} \mathbf{x}_i \mathbf{x}_i^T - M\mathbf{I}$$

Aquí

- **W es la matriz de pesos**
- **x_i es un patrón de entrada representado por un vector de dimensión** N **con coordenadas en el conjunto {–1, 1}.**
  - N **es el número de unidades de la red, que coincide con la dimensión de los vectores de entrada. Los valores 1 y -1 representan las unidades activas e inactivas respectivamente.**
- **$(x_i)^T$ es el vector traspuesto del vector de entrada x_i ,**
- M **denota el número de patrones de entrada del conjunto de entrenamiento**
- **I es la matriz identidad de dimensión** N×N

Ejemplo 3

- **Consideremos ahora una red de Hopfield con cuatro unidades y tres vectores del conjunto de entrenamiento que han de ser aprendidos por la red. Consideremos tres vectores de entrada, a saber, x_1, x_2 y x_3 y que se definen de la siguiente manera:**

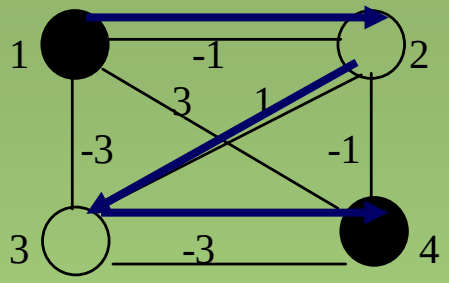
$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix}; \ \mathbf{x}_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}; \ \mathbf{x}_3 = \begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \end{bmatrix}; \text{ los pesos de la red se calculan como } \mathbf{W} = \sum_{i=1}^{M} \mathbf{x}_i \mathbf{x}_i^T - M\mathbf{I}$$

$$\mathbf{x}_1\mathbf{x}_1^T = \begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}; \ \mathbf{x}_2\mathbf{x}_2^T = \begin{bmatrix} 1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & -1 \\ 1 & 1 & -1 & 1 \end{bmatrix}; \mathbf{x}_3\mathbf{x}_3^T = \begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

$$\mathbf{W} = \begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & -1 \\ 1 & 1 & -1 & 1 \end{bmatrix} + \begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} - 3\mathbf{I} = \begin{bmatrix} 3 & -1 & -3 & 3 \\ -1 & 3 & 1 & -3 \\ -3 & 1 & 3 & -3 \\ 3 & -3 & -3 & 3 \end{bmatrix} - 3\mathbf{I}$$

Ejemplo 3

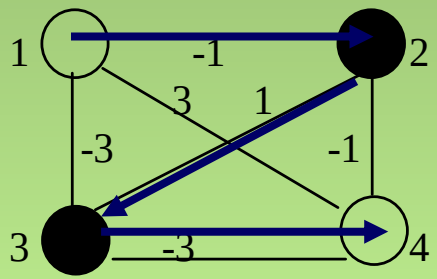
(1)



$x_1 = [1\ -1\ -1\ 1]$

Posición estable de la red

(1)



$x_3 = [-1\ 1\ 1\ -1]$

Posición estable de la red

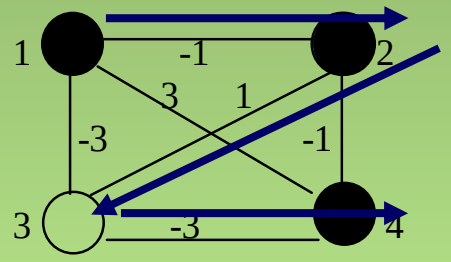
$$\mathbf{W} = \sum_{i=1}^{M} \mathbf{x}_i \mathbf{x}_i^{\mathbf{T}} - M\,\mathbf{I}$$

(1)

$$\mathbf{W} = \begin{pmatrix} 3 & -1 & -3 & 3 \\ -1 & 3 & 1 & -1 \\ -3 & 1 & 3 & -3 \\ 3 & -1 & -3 & 3 \end{pmatrix} - \begin{pmatrix} 3 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 3 \end{pmatrix} = \begin{pmatrix} 0 & -1 & -3 & 3 \\ -1 & 0 & 1 & -1 \\ -3 & 1 & 0 & -3 \\ 3 & -1 & -3 & 0 \end{pmatrix}$$

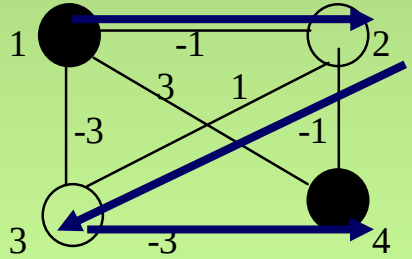
Ejemplo 3

(1)



$X_2 = [1\ 1\ -1\ 1]$

(1)



$X_1 = [1\ -1\ -1\ 1]$

x_2 se estabiliza en x_1

Ejemplo 3

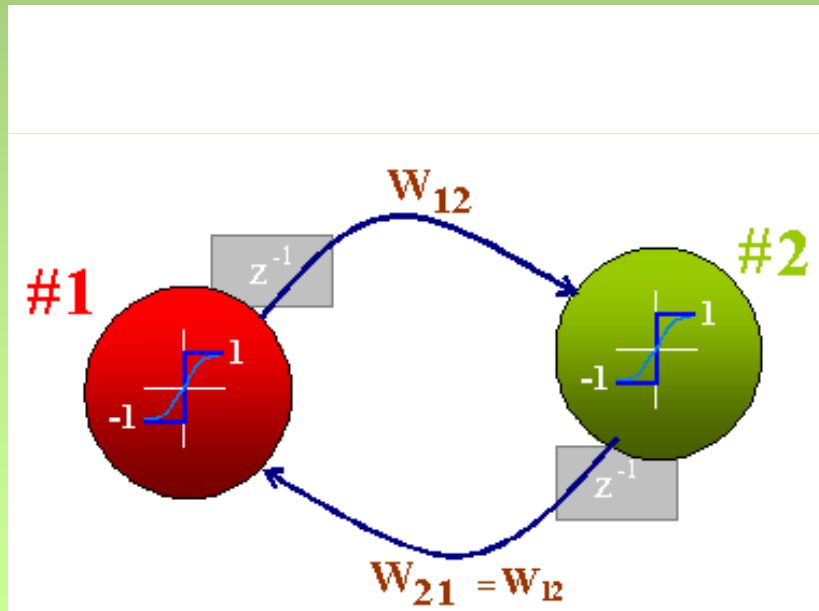
- Las redes generadas utilizando estos pesos y los vectores de entrada son estables, a excepción de $\mathbf{x}_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$
- $\mathbf{x}_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$ se estabiliza en $\mathbf{x}_1 = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix}$ (el cual está a una distancia de Hamming de 1).

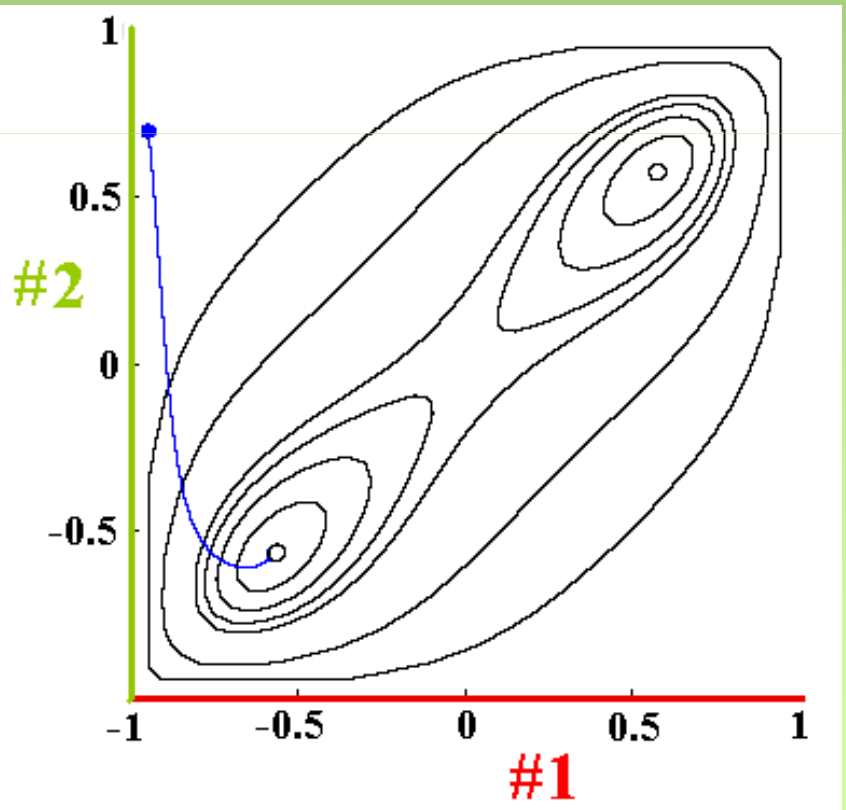
- Así, con los pesos obtenidos y los estados estables ($\mathbf{x}_1$ y $\mathbf{x}_3$), podemos estabilizar cualquier nuevo patrón (parcialmente) a uno de ellos

Ejemplo

Red de Hopfield con 2-neuronas de estados contínuos caracterizada por dos estados estables

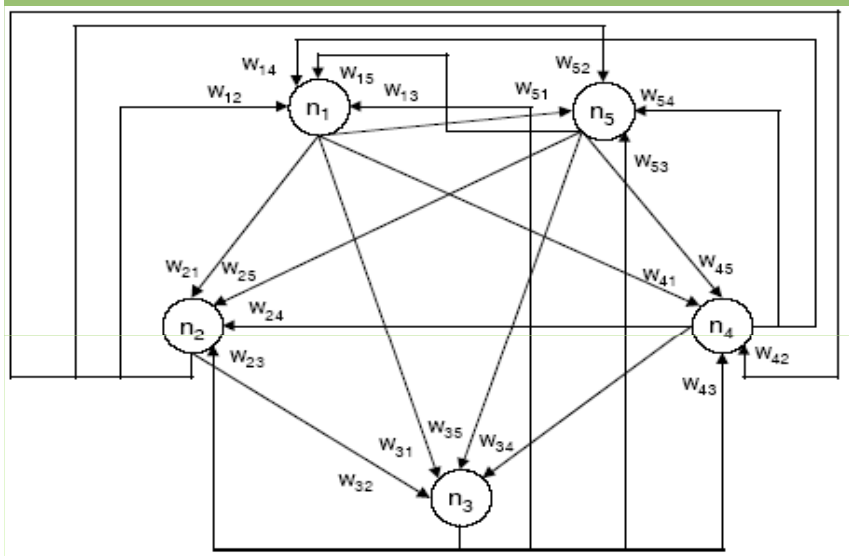
Dibujo del contorno

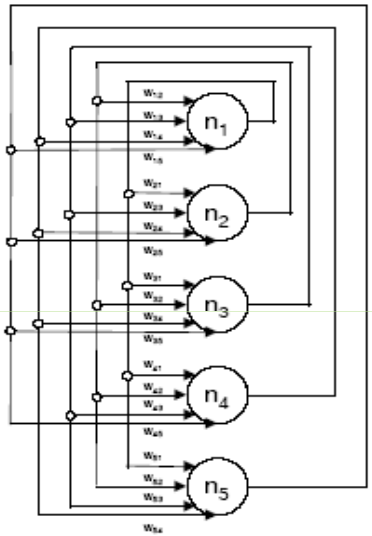




Sistema dinámico

El comportamiento de un sistema dinámico de este tipo está totalmente determinado por los pesos sinápticos

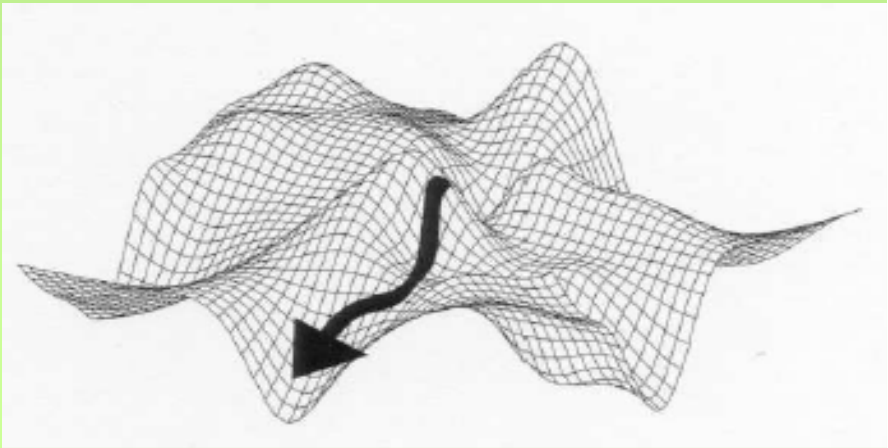




$$W_{ij} = W_{ji}$$

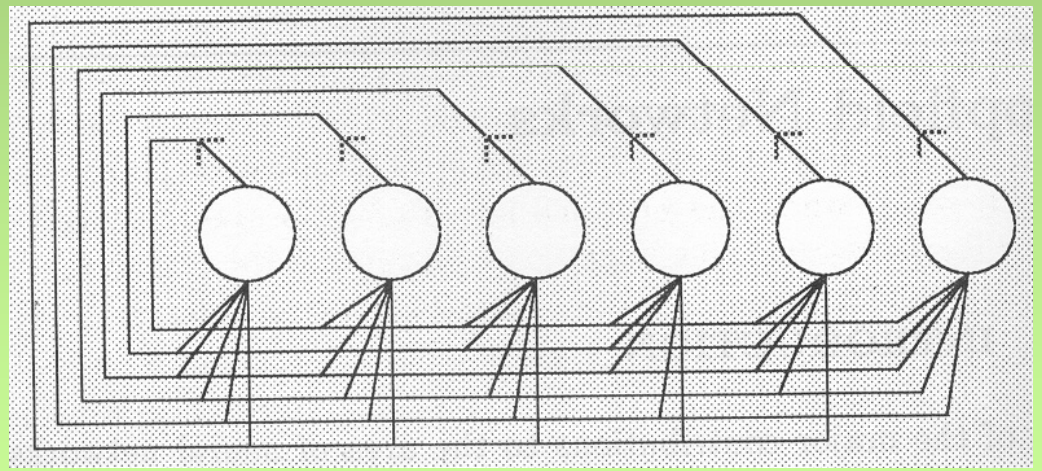
$$\begin{pmatrix} 0 & w_{12} & w_{13} & w_{14} & w_{15} \\ w_{21} & 0 & w_{23} & w_{24} & w_{25} \\ w_{31} & w_{32} & 0 & w_{34} & w_{35} \\ w_{41} & w_{42} & w_{43} & 0 & w_{45} \\ w_{51} & w_{52} & w_{53} & w_{54} & 0 \end{pmatrix}$$

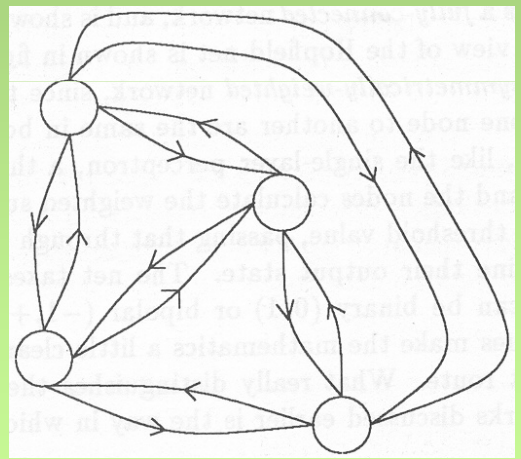
Se puede pensar como un proceso de minimización de la energía. El sistema dinámico correspondiente evoluciona hacia estados de baja energía



Sin auto conexiones

Las redes de Hopfield están completamente conectadas, son redes simétricamente ponderadas que extienden las ideas de memorias lineales asociativas mediante la adición de conexiones cíclicas





Funcionamiento de la red

Después de la "etapa de enseñanza-aprendizaje", en la que se definen los pesos, se establece el estado inicial de la red (patrón de entrada) y una regla sencilla recurrente se itera hasta que se alcance la convergencia a un estado estable (patrón de salida)

En cuanto al entrenamiento de una red Hopfield como una memoria de contenido direccionable se utiliza la regla del producto externo, xxT, para el almacenamiento de patrones.

Hay dos modos principales de funcionamiento del entrenamiento:

Actualización síncrona versus asíncrona

Hopfield Network Algorithm

1. Assign connection weights

$$w_{ij} = \begin{cases} \frac{1}{N}\sum_{s=0}^{M-1} x_i^s x_j^s & i \neq j \\ 0 & i = j, \end{cases}$$

Aprendizaje Hebbiano

where w_{ij} is the connection weight between node i and node j, and x_i^s is element i of the exemplar pattern s, and is either $+1$ or -1.
There are M patterns, from 0 to $M-1$, in total.

2. Initialise with unknown pattern

$$\mu_i(0) = x_i \qquad 0 \leq i \leq N-1$$

Patrón de prueba

where $\mu_i(t)$ is the output of node i at time t.

3. Iterate until convergence

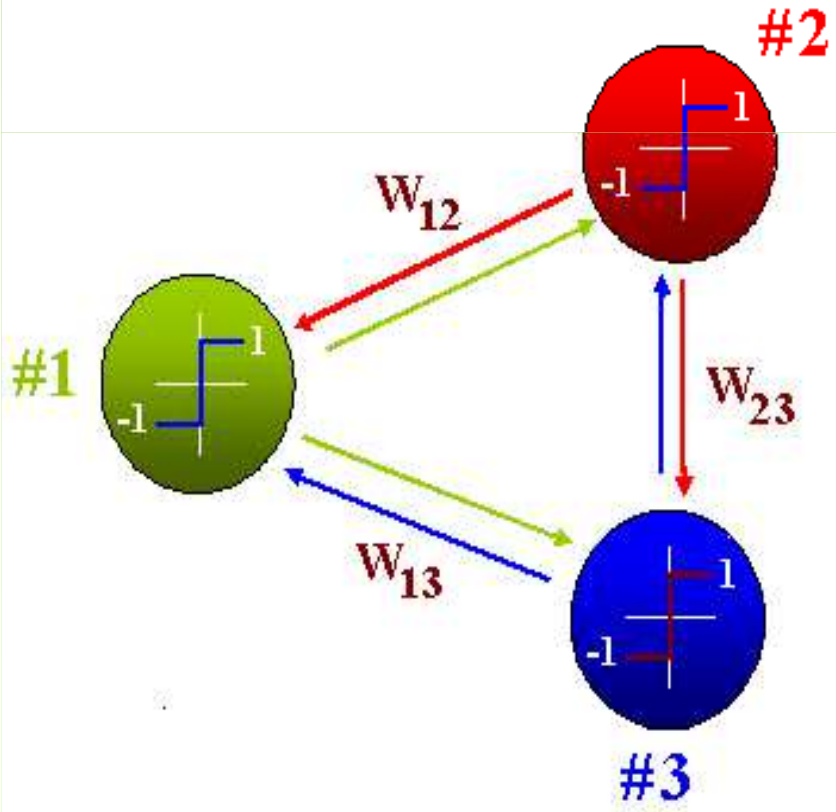
$$\mu_i(t+1) = f_h\left[\sum_{i=0}^{N-1} w_{ij}\mu_j(t)\right] \qquad 0 \leq j \leq N-1$$

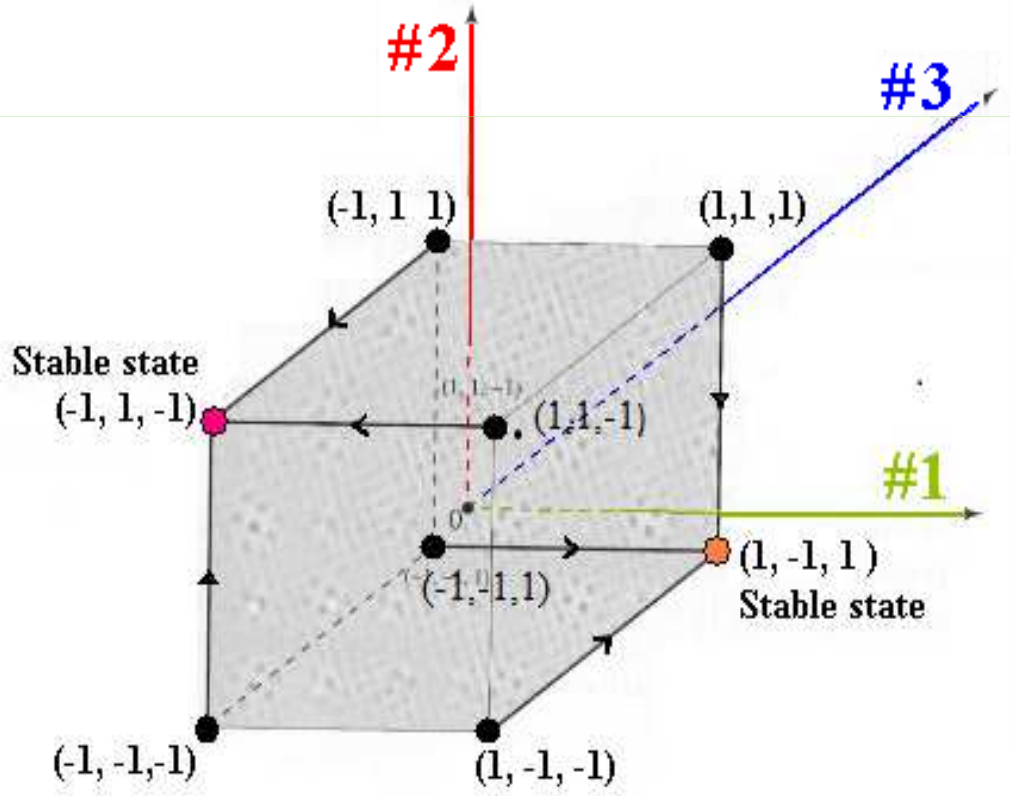
Evolución dinámica

The function f_h is the hard-limiting non-linearity, the step function,
Repeat the iteration until the outputs from the node remain unchanged.

Nuevo Ejemplo

Red de Hopfield con 3-neuronas o nodos de 8 estados caracterizada por dos estados estables





Un ejemplo sencillo

Etapa 1.

Diseñar una red con patrones (vectores) memorizados

[1, -1, 1] y [-1, 1, -1] y estables, el resto inestables

- Compute the outer products, sum, normalize, and set diagonals to 0:

$$(1, -1, 1)^T * (1, -1, 1) + (-1, 1, -1)^T * (-1, 1, -1) = \begin{bmatrix} 2 & -2 & 2 \\ -2 & 2 & -2 \\ 2 & -2 & 2 \end{bmatrix} \Rightarrow \frac{1}{3} \begin{bmatrix} 0 & -2 & 2 \\ -2 & 0 & -2 \\ 2 & -2 & 0 \end{bmatrix}$$

Force Diagonal to 0

$\frac{1}{3}$ = 1/(number of neurons)

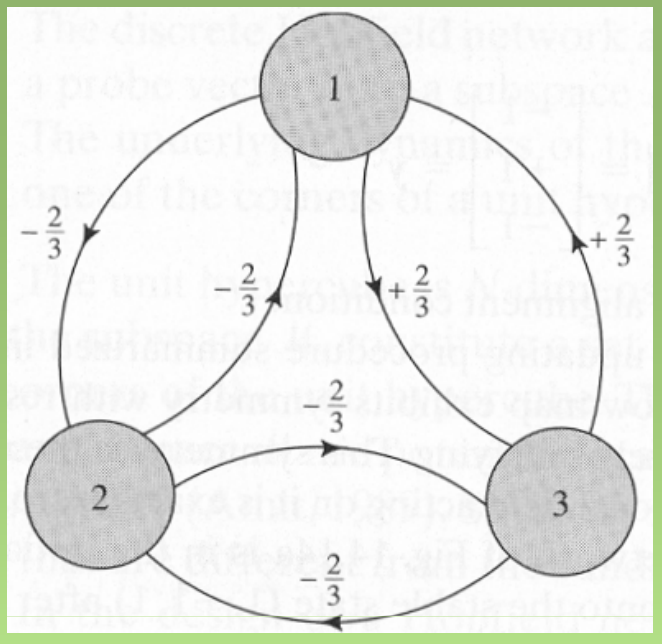
$$\mathbf{W} = \sum_{i=1}^{M} \mathbf{x}_i \mathbf{x}_i^T - M\,\mathbf{I}, o\ \mathbf{W} = \frac{1}{N} \sum_{i=1}^{M} \mathbf{x}_i \mathbf{x}_i^T - \frac{M}{N}\mathbf{I}$$

O lo que es igual

$$\mathbf{W} = \frac{1}{3}\begin{bmatrix} +1 \\ -1 \\ +1 \end{bmatrix}[+1, -1, +1] + \frac{1}{3}\begin{bmatrix} -1 \\ +1 \\ -1 \end{bmatrix}[-1, +1, -1] - \frac{2}{3}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Etapa 2. Inicialización

$$\mathbf{w} = \frac{1}{3}\begin{bmatrix} 0 & -2 & +2 \\ -2 & 0 & -2 \\ +2 & -2 & 0 \end{bmatrix}$$



Hay 8 estados diferentes que pueden ser alcanzados por la red y por lo tanto, cualquiera se puede utilizar como su estado inicial

#1: y_1
#2: y_2
#3: y_3

$$\begin{bmatrix} -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 \\ -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \\ -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \end{bmatrix}$$

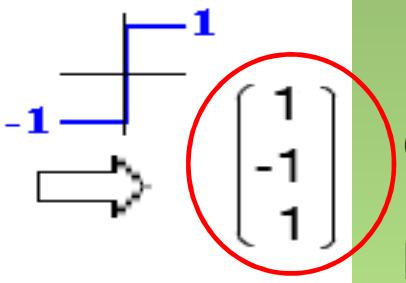
Patrones iniciales, estados estables

Etapa 3.- Iterar hasta la convergencia

- Cambio Síncrono-

3 ejemplos diferentes del flujo de la red

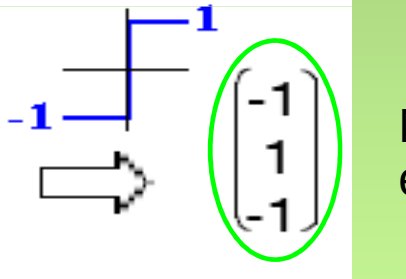
$$\frac{1}{3}\begin{pmatrix} 0 & -2 & 2 \\ -2 & 0 & -2 \\ 2 & -2 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 0 \\ -4 \\ 0 \end{pmatrix}$$



$$\begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$$

Cambia sólo el del medio, pasa a estado estable

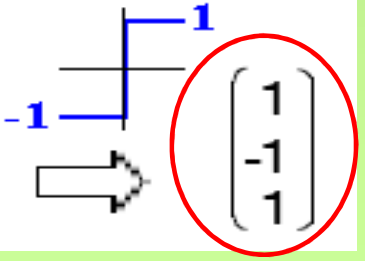
$$\frac{1}{3}\begin{pmatrix} 0 & -2 & 2 \\ -2 & 0 & -2 \\ 2 & -2 & 0 \end{pmatrix}\begin{pmatrix} -1 \\ 1 \\ -1 \end{pmatrix} = \frac{1}{3}\begin{pmatrix} -4 \\ 4 \\ -4 \end{pmatrix}$$



$$\begin{pmatrix} -1 \\ 1 \\ -1 \end{pmatrix}$$

No cambia, estado estable

$$\frac{1}{3}\begin{pmatrix} 0 & -2 & 2 \\ -2 & 0 & -2 \\ 2 & -2 & 0 \end{pmatrix}\begin{pmatrix} -1 \\ -1 \\ 1 \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 4 \\ 0 \\ 0 \end{pmatrix}$$



$$\begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$$

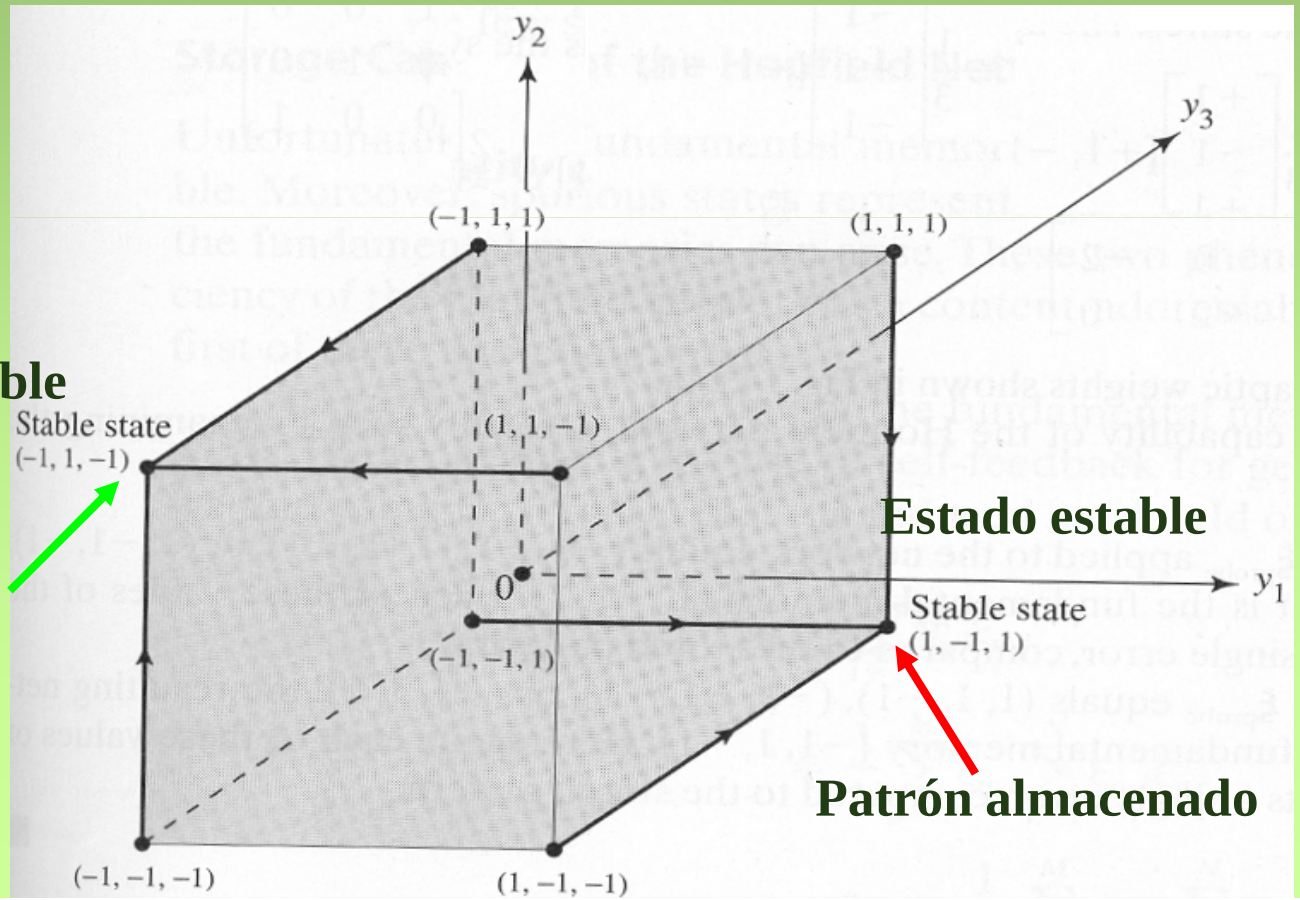
Cambia sólo el primero, pasa a estado estable

Converge inmediatamente

Etapa 3.- Iterar hasta la convergencia

- Cambio Síncrono-

Diagrama esquemático de todas las trayectorias dinámicas que corresponden a la red diseñada.



Estado estable

Estado estable

Patrón almacenado

Etapa 3.- Iterar hasta la convergencia

- Cambio Asíncrono-

Cada vez, seleccione una neurona al azar y actualice su estado con la regla anterior

Y el convenio general de que si la entrada total a la neurona es 0 su estado permanece sin cambios

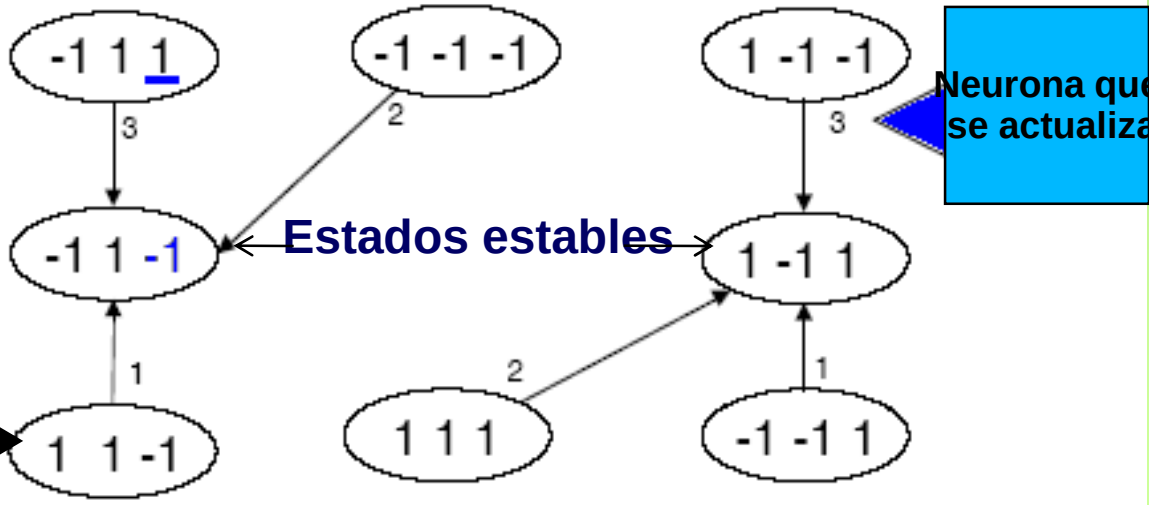
Ahora si seleccionamos la neurona primera en (1,1,-1) pasamos al estado estable (-1,1,-1)

$$\frac{1}{3}(0,-2,2)\begin{pmatrix}1\\1\\-1\end{pmatrix}=\frac{1}{3}(-4),\ \text{cambia 1 por -1}$$

possible initial states

$$\frac{1}{3}\begin{pmatrix}0&-2&2\\-2&0&-2\\2&-2&0\end{pmatrix}\qquad \begin{pmatrix}-1&-1&-1&-1&1&1&1&1\\-1&-1&1&1&-1&-1&1&1\\-1&1&-1&\underline{1}&-1&1&-1&1\end{pmatrix}$$

$$\frac{1}{3}(2\ -2\ \ 0)\begin{pmatrix}-1\\1\\\underline{1}\end{pmatrix}=\frac{1}{3}(-4)$$ thresholding → -1



MODELOS COMPUTACIONALES: CUARTO CURSO DEL GRADO DE ING. INFORMÁTICA EN COMPUTACION

Modelos de redes neuronales

GRACIAS POR SU ATENCIÓN

César Hervás-Martínez
Grupo de Investigación AYRNA

Departamento de Informática y Análisis Numérico
Universidad de Córdoba
Campus de Rabanales. Edificio Einstein.
Email: chervas@uco.es

2019-2020