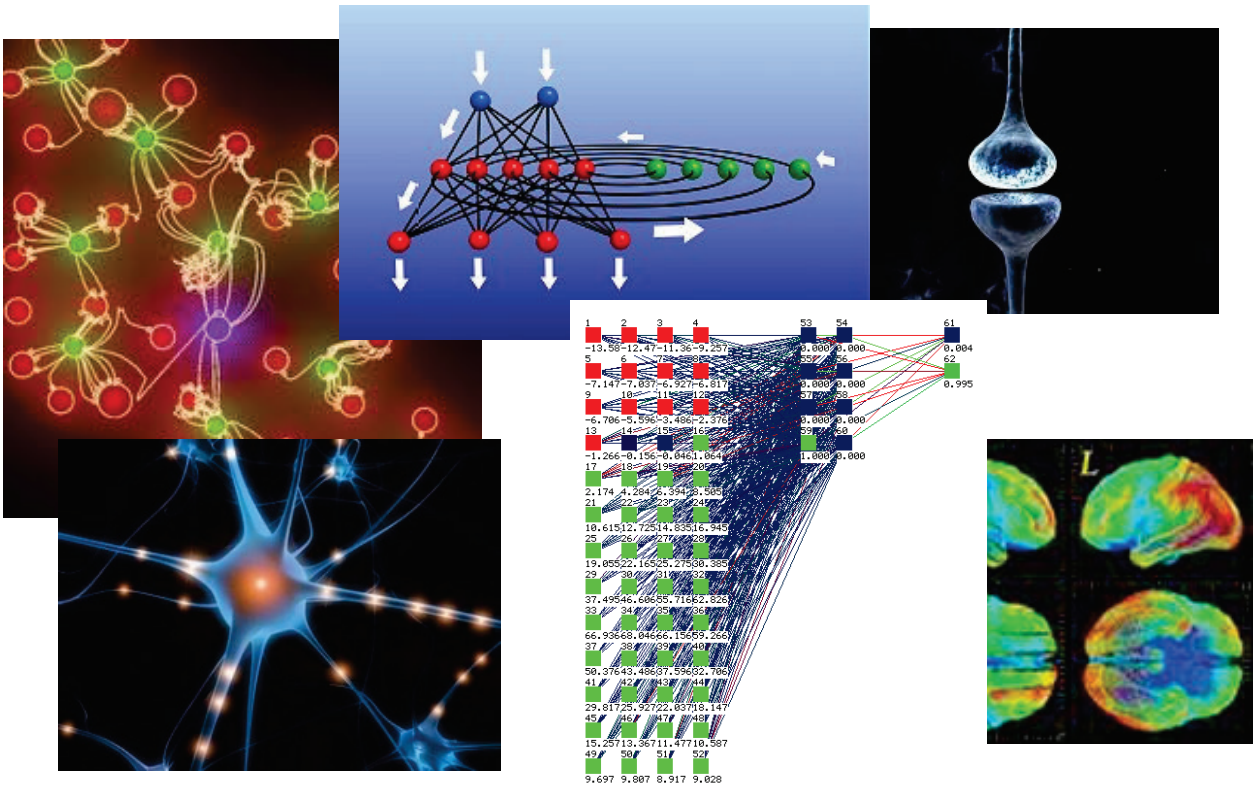
Reti neurali

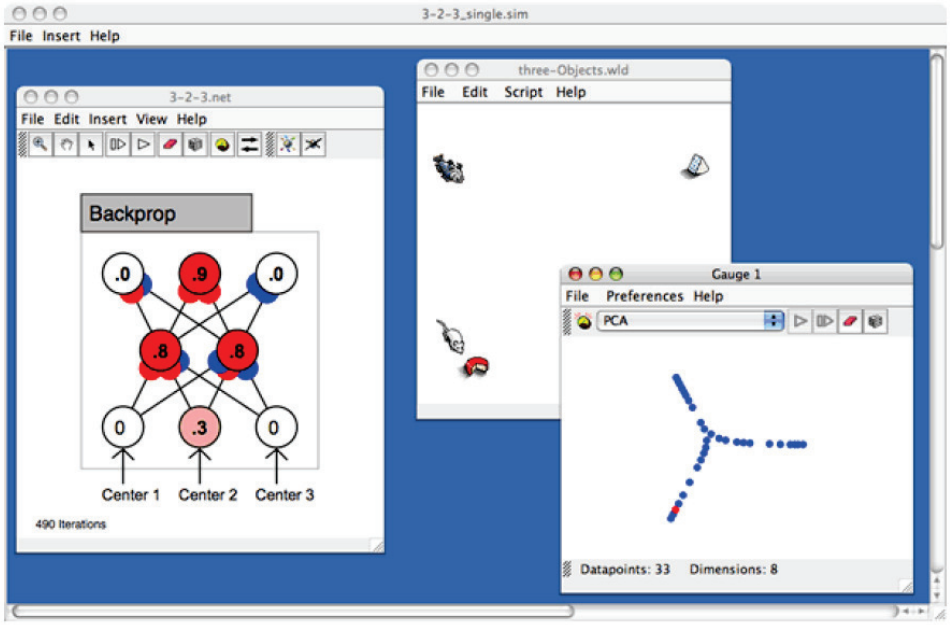


Ambienti

- **Diverse (contemporanee) esigenze:**
  - **Ambiente visuale e di facile utilizzo**
    siamo interessati alla dinamica, più che a dettagli implementativi
  - **Possibilità di sviluppo di architetture di rete arbitrarie**
    reti di Hopfield, competitive, feed-forward, ricorrenti, ecc. ecc.
  - **Possibilità di comprendere la dinamica rete-ambiente**
    presenza di «mondi virtuali»
  - **Visualizzazioni «furbe» del comportamento della rete**
    è bene utilizzare spazi astratti, capaci di comprimere l'informazione multidimensionale (od almeno la parte rilevante) in poche dimensioni
  - **Ambiente free**
    possibilmente utilizzato da un'ampia comunità

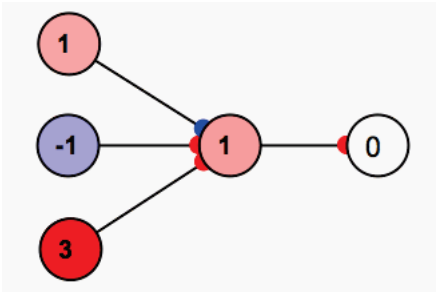
Simbrain

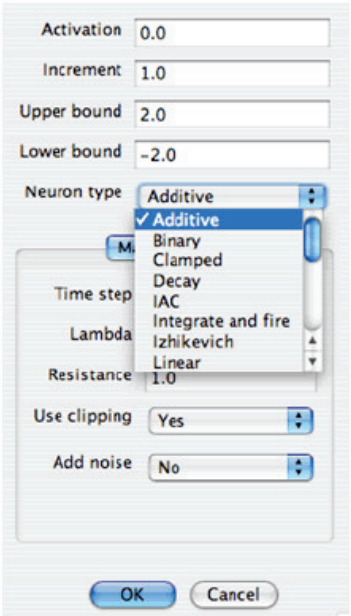
- **Workspace**
  - **Networks**
    reti di Hopfield, competitive, feed-forward, ricorrenti, ecc. ecc.
  - **Gauges**
    tools di visualizzazione
  - **Virtual worlds**
    presenza di «mondi virtuali»



Reti

- **Reti**
  - **Neuroni**
  - **Sinapsi**
- **Proprietà comuni**
  - **Attivazione**
    - Valore attuale del neurone
  - **Incremento**
    - Passo dell'incremento manuale (disattivo durante le iterazioni)
  - **Limiti all'attivazione**
    - Limite superiore
    - Limite inferiore
  - **Clipping**
    - Attiva o meno i limiti all'attivazione



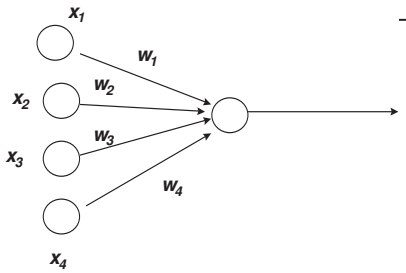


Reti

- Reti
  - Neuroni
  - Sinapsi
- Proprietà comuni
  - **Attivazione**
    - Valore attuale del neurone
  - **Incremento**
    - Passo dell'incremento manuale (disattivo durante le iterazioni)
  - **Limiti all'attivazione**
    - Limite superiore
    - Limite inferiore
  - **Clipping**
    - Attiva o meno i limiti all'attivazione
- Tipi di neurone
  - Additive
  - Binary
  - Clamped
  - Decay
  - IAC
  - Integrate and fire (s)
  - Izhikevich (s)
  - Linear
  - LMS
  - Logistic
  - Naka-Rushton
  - Point
  - Random
  - Sigmoidal
  - Sinusoidal
  - Stochastic
  - Three valued
  - Trace

Reti

$$I_i = W_i = w_{i1}x_1 + w_{i2}x_2 + w_{i3}x_3 + w_{i4}x_4$$



- Tipi di neurone
  - Additive
  - Binary
  - Clamped
  - Decay
  - IAC
  - Integrate and fire (s)
  - Izhikevich (s)
  - Linear
  - LMS
  - Logistic
  - Naka-Rushton
  - Point
  - Random
  - Sigmoidal
  - Sinusoidal
  - Stochastic
  - Three valued
  - Trace

- Neurone binario
  - Soglia (Threshold - θ)
  - Bias (b)

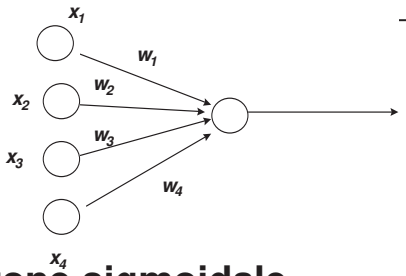
$$a = \begin{cases} u & W + b > \theta \\ l & \text{otherwise} \end{cases}$$

- Neurone lineare
  - Pendenza (m)
  - Bias (b)

$$a = m(W + b)$$

Reti

$$I_i = W_i = w_{i1}x_1 + w_{i2}x_2 + w_{i3}x_3 + w_{i4}x_4$$



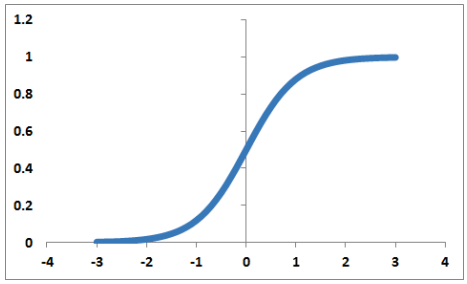
- Neurone sigmoidale
  - Tanh
  - Arctan

$$a = \frac{u-l}{\pi}\arctan\left(\frac{\pi m(W+b)}{u-l}\right) + \frac{u+l}{2}$$

  - Logistic

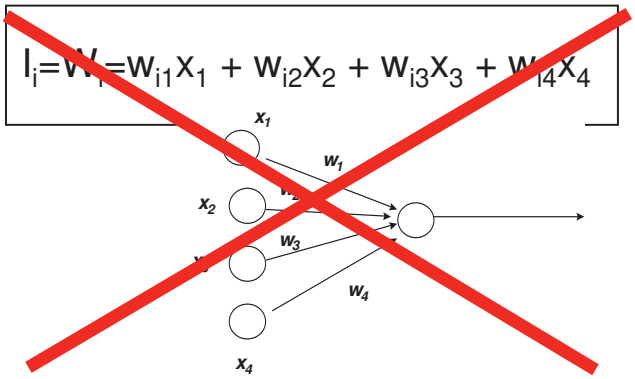
$$a = \frac{u-l}{2}\mathrm{sgm}\left(\frac{4m(W+b)}{u-l}\right) + l$$

sgm(t)=1/(1-e^{-t})



- Tipi di neurone
  - Additive
  - Binary
  - Clamped
  - Decay
  - IAC
  - Integrate and fire (s)
  - Izhikevich (s)
  - Linear
  - LMS
  - Logistic
  - Naka-Rushton
  - Point
  - Random
  - Sigmoidal
  - Sinusoidal
  - Stochastic
  - Three valued
  - Trace

Reti



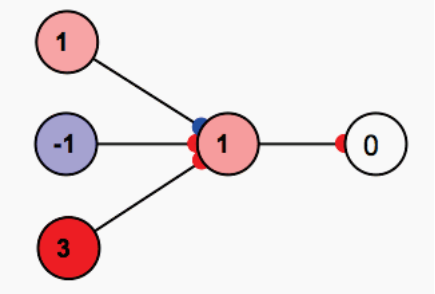
- Neurone «clamped»
  - Non cambia il proprio valore (che però può essere inizializzato dall'utente)

- Tipi di neurone
  - Additive
  - Binary
  - Clamped
  - Decay
  - IAC
  - Integrate and fire (s)
  - Izhikevich (s)
  - Linear
  - LMS
  - Logistic
  - Naka-Rushton
  - Point
  - Random
  - Sigmoidal
  - Sinusoidal
  - Stochastic
  - Three valued
  - Trace

- Neurone «random»
  - Funziona autonomamente
  - Prende il proprio valore dal generatore stocastico del computer

Reti

- Reti
  - Neuroni
  - Sinapsi
- Proprietà comuni
  - Strength
    - Valore del peso
  - Incremento
    - Passo dell'incremento manuale (disattivo durante le iterazioni)
  - Limiti all'attivazione (può dipendere dal tipo di sinapsi)
    - Limite superiore
    - Limite inferiore
  - Clipping
    - Attiva o meno i limiti all'attivazione



- Delay
  - Numero di passi necessario perché il segnale giunga al neurone bersaglio

Reti

- Reti
  - Neuroni
  - Sinapsi
- Tipi di sinapsi
  - Clamped
  - Hebbian
  - Hebbian Threshold
  - Oja
  - Random
  - Short Term Plasticity
  - Signal
  - S. Normalization
- Peso Hebbiano con soglia
  - Learning rate ε
  - Soglia (Threshold - θ)

$$\Delta w = \epsilon a_s a_t (a_t - \theta)$$

  - Se la soglia può variare (BCM rule)

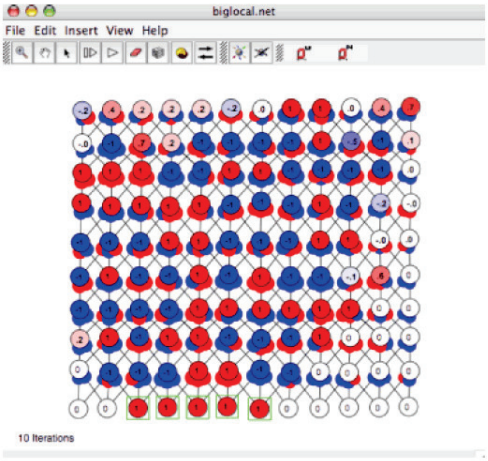
$$\Delta\theta = \epsilon_\theta (a_t{}^2 - \theta)$$

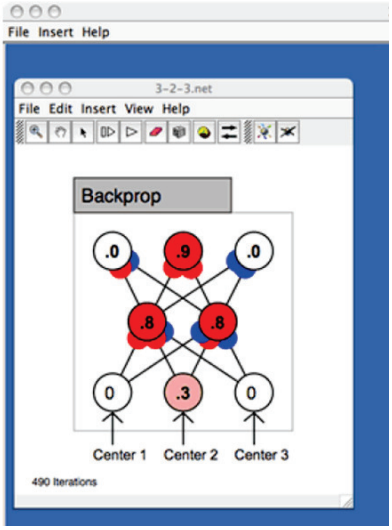
- Peso Hebbiano
  - Implementa l'apprendimento Hebbiano
  - Parametro: learning rate ε

$$\Delta w = \epsilon a_s a_t$$

Reti

- Reti
  - Struttura generale

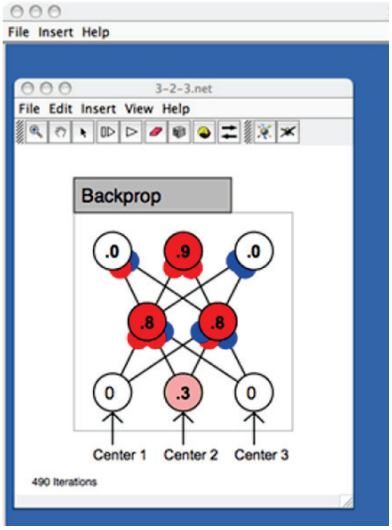




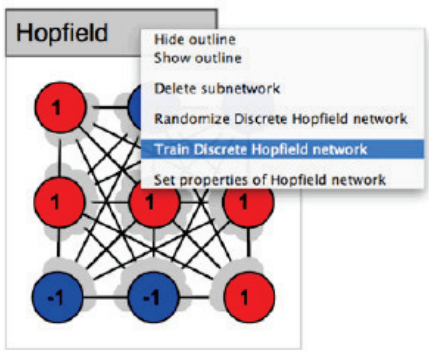
- Tipi di struttura
  - reti
    - Unioni di neuroni variamente connessi
  - Sottoreti
    - Unioni omogenee sottoposte ad aggiornamenti particolari (ordine di aggiornamento, variabili non locali, …)
  - Gruppi (prossimamente…)
    - Creazione di gruppi logici, controllata dall'utente

Reti

- Reti
  - Struttura generale



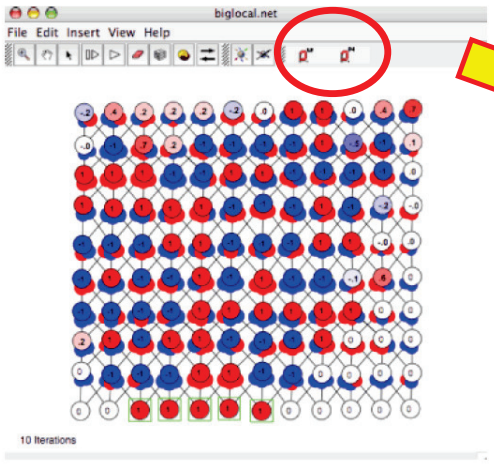
- Sottoreti
  - Competitiva
  - Hopfield
  - Winner take all (WTA)
  - K-Winner take all (KWTA)
  - Self organising map (SOM)



- Sottoreti
- Unioni omogenee sottoposte ad aggiornamenti particolari (ordine di aggiornamento, variabili non locali,…)

Reti

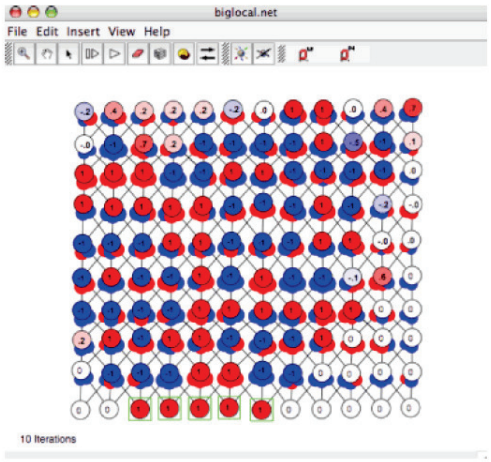
- Reti
  - Struttura generale



- Clamping
  - Possibilità di bloccare i valori dei neuroni (durante l'apprendimento Hebbiano) o delle sinapsi (funzionamento «normale»)

Reti

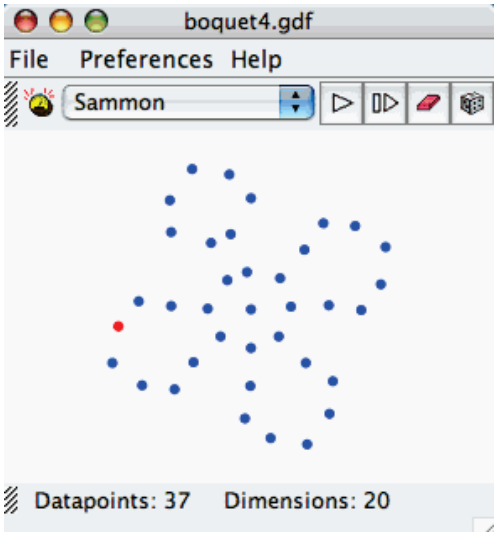
- Reti
  - Struttura generale



- Aggiornamento
  - Bufferizzato (default)
    - Tutti i neuroni sono «sincronizzati»
  - Priority-Based Update
    - Asincrono, con ordine crescente con l'«ID» del neurone
  - Custom
    - Non accessibile da GUI
  - «Sottorete»
    - Se del caso, adeguato al tipo di sottorete
- Reti «continue»
  - Una parte rilevante dell'ambiente è dedicata a reti continue, immerse nel tempo, con neuroni «spike»
  - Non si può fare tutto …

Plots

- **One of the main methods for studying the activity of the network component in Simbrain is by using a plot component**
  - **the plot components are general and can be used to show the values of any attribute in Simbrain**



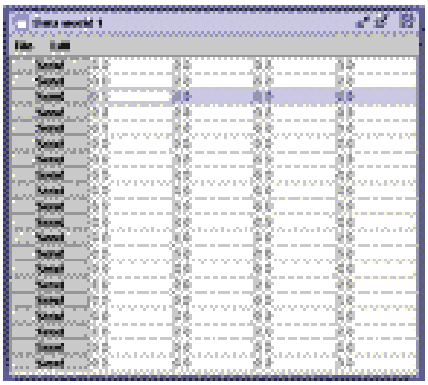
- **Plots**
  - **Bar Chart**
  - **Pie Chart**
  - **Projection**
  - **Scatter Plot**
  - **Time Series**
- **Plots are "data consumer"**
  - **and therefore need some data producers (e.g. neurons)**

Plots

- **Projection Methods**
  - **Coordinate Projection**
  - This is perhaps the simplest possible projection technique. If one has a list of datapoints with 40 components each, coordinate projection to two-dimensions simply ignores all but two of these components, which are then used to display the data in two-space
  - **Principal Component Analysis (PCA)**
  - PCA builds on coordinate projection by making use of the "principal axes" of the dataset. PCA selects the two principal axes along which the dataset is the most spread out and projects the data onto these two axes
  - **Sammon map**
  - The Sammon map is an iterative technique for making interpoint distances in the low-dimensional projection as close as possible to the interpoint distances in the high-dimensional object. By minimizing an error function between the high and low dimensional sets of interpoint distances, the Sammon map does its best to preserve these distances in the projection
    - Two points close together in the high-dimensional space should appear close together in the projection, while two points far apart in the high dimensional space should appear far apart in the projection.

Mondi virtuali

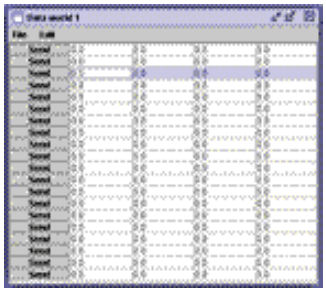
- **Mondi virtuali**
  - «Dataworld»
  - «Odorworld»



- **E’ un vero e proprio «mondo» (molto semplificato…) cui è possibile collegare una o più reti**
- **E’ un «punto di accesso» al sistema**
- **Agisce anche come produttore interno di dati**

Dataworld - Collegamenti

- **Dataworld è essenzialmente una tabella, in cui è possibile:**
  - **immettere valori**
  - **creare dati casuali (uniformemente fra una soglia minima ed un massima)**
  - **Importare ed esportare dati in formato csv**



- **Le colonne in dataworld possono essere consumatori oppure produttori di dati**
  - **Quando le colonne producono dati, durante ogni aggiornamento dello spazio di lavoro (workspace update) i valori delle colonne nella riga corrente sono spediti al consumatore (es. un neurone)**
  - **Quando le colonne consumano dati, durante ogni aggiornamento dello spazio di lavoro (workspace update) i valori provenienti da un produttore (es. un neurone) sono scritti nella riga corrente, in corrispondenza delle dette colonne**