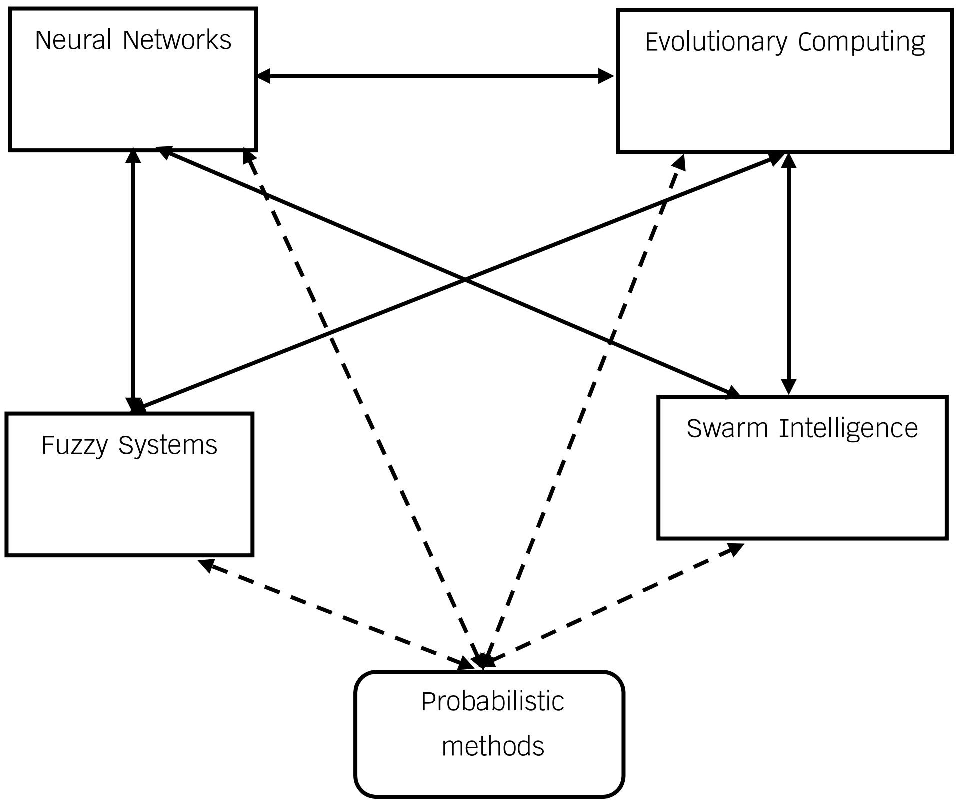
Review 1st midterm exam

Introduction to Computational Intelligence

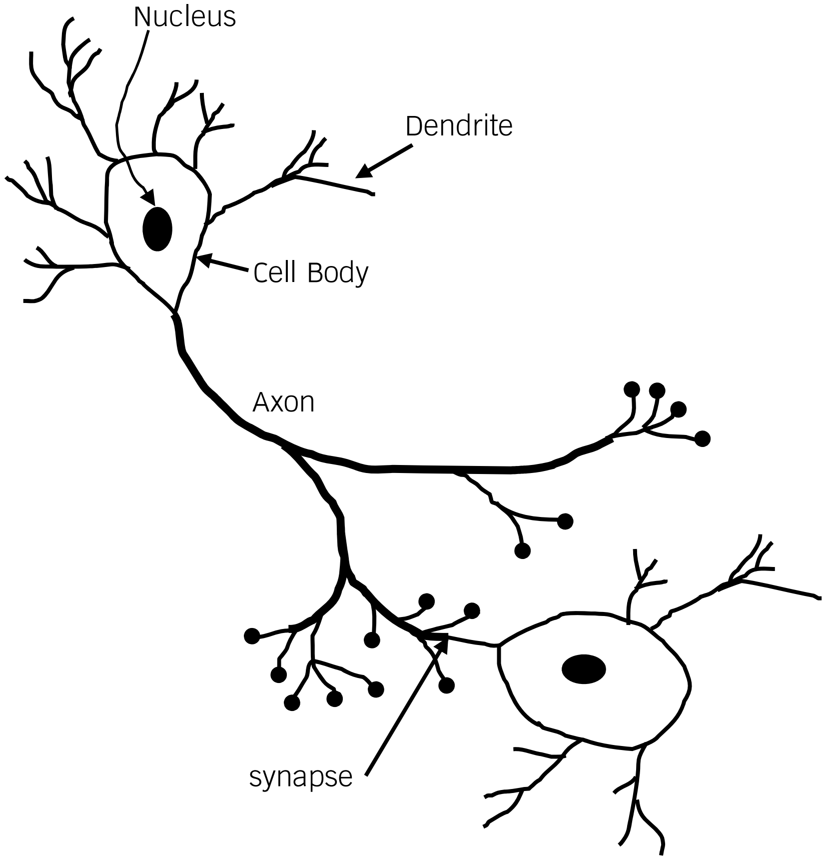
History

- Read from any books
  - History of NN
  - History of Fuzzy
  - History of Evolutionary Computation
  - History of Swarm Intelligence

Connection in CI



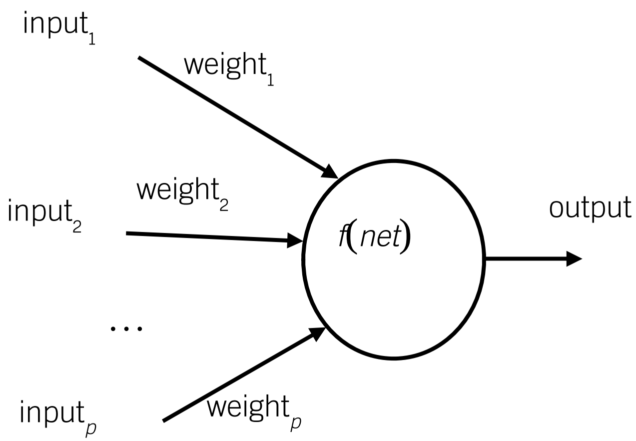
Neural Network (NN)



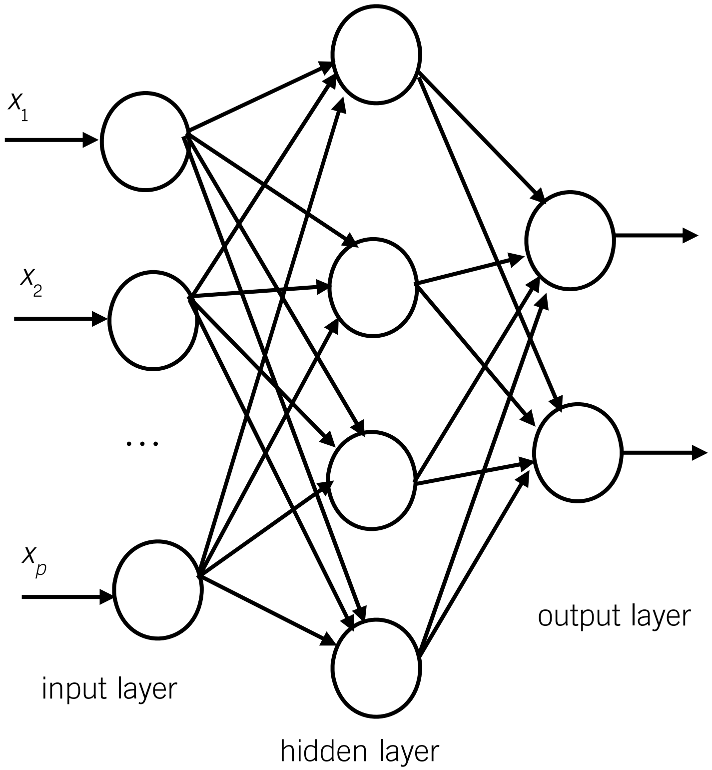
Biological Neural

NN

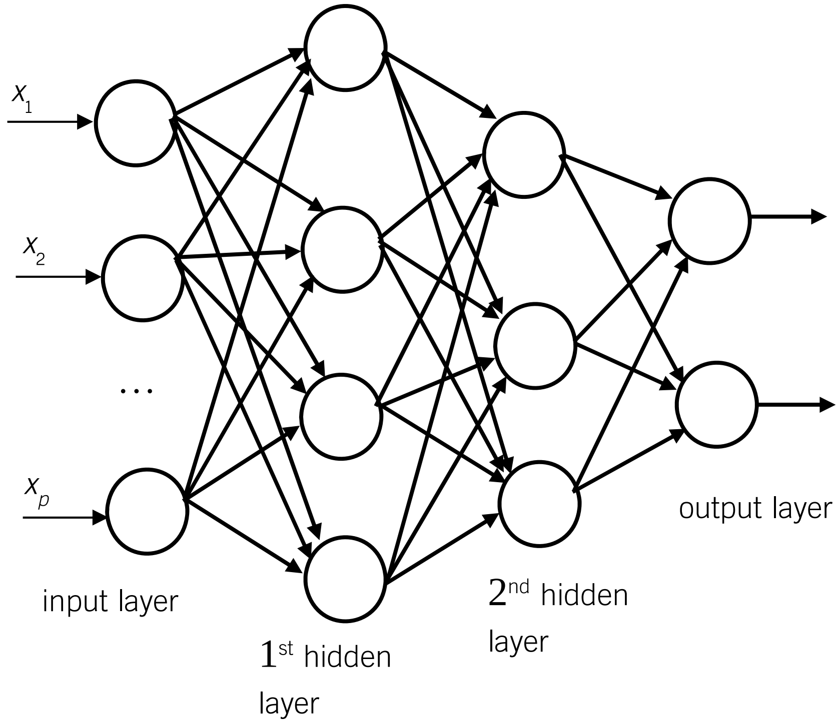
Artificial neuron



NN



p-4-2



p-4-3-2

p-m-2 :p input nodes, m hidden nodes, 2 output nodes

Number of hidden nodes or hidden layers will be shown here→ p-m1-m2-2 means that there are 2 hidden layers: one with m1 nodes and the second one is with m2 nodes

Number of output nodes normally corresponds to the number of classes

Fuzzy System

- Not only 0 or 1, there is a gray area
- Unsharp boundary
- Cope with uncertainty
- Approximate reasoning

Evolutionary Computation

- Individual→chromosome→inherit characteristic
- Population of chromosome
- Each characteristic --<gene– allel(gene value)
- Each generation→individual compete to reproduce offspring
- Best survival capability have the best chance to reproduce
- Crossover→combine part of parents to generate offspring
- Mutation→alter some of allele of the chromosome
- Survuval strength→ measured using fitness function→ objective function
- Each generation individual→ culling or survive
- Behavior→phenotype →influence evolutionary process in genetic change and evolve separately

Evolutionary Computation

- EC algo
  - Genrtic algorithm→model genetic evolutiona
  - Genetic programming→similar to GA but individual are program
  - Evolutionary programming→derived from the simulation of adaptive behavior in evolution (phenotype evolution)
  - Evolution strategies→model strategic parameters that control variation in evolution
  - Differential evolution→similar to GA except reproduction mechanism used
  - Cultural evolution → model the evolution of culture of a population and how the culture influences the genetic and phenotypic evolution of individual
  - Coevolution→ “dump” individual evolve through cooperation or in competition with one another, acquiring the necessary characteristics to survive

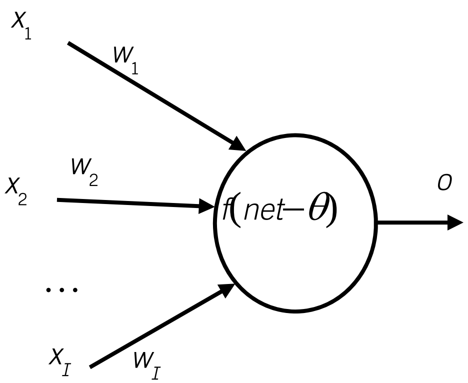
Swarm Intelligence

- Study of colonies or swarm of social organisma
- Study of social behavior of organisms (individuals) in swarm prompted the design f very efficient optimization and clustering algorithm
- Particle swarm optimization (PSO) →global optimization approach modeled on social behavior of bird flocks
  - A population based search procedure where the individuals (particles) are grouped into a swarm
  - Particle→individual→candidate solution to the optimization problem
    - Flown through the multidimensional search space→adjusting its position in search space according to its own experience and that of neighboring particles
    - Best position encounter by itself and that of its neighbor→position itself to the global minimum
    - Performance→measured according to a predefined fitness function which related to the problem being solved
- Study of ant colonies→modeling of pheromone depositing by ants in their search for the shortest paths to food sources resulted in the development of shortest path optimization algorithm

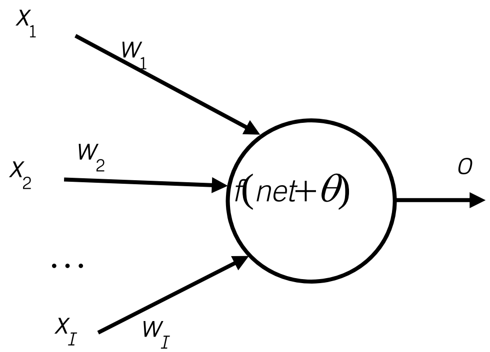
Neural Networks (NN)

- Mapping function form *I* input space to *K* output space

$$f_{NN} : \Re^{I} \rightarrow \Re^{K}$$



or

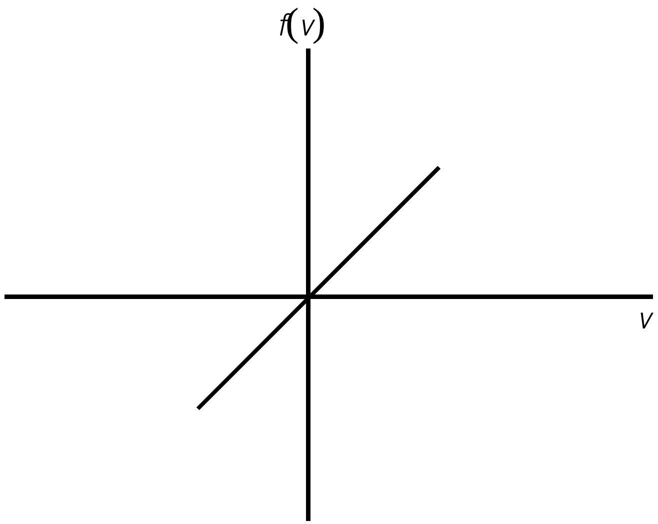


$net = \sum_{i=1}^{I} x_i w_i$ or $net = \prod_{i=1}^{I} x_i^{w_i}$ and θ called bias or threshold

NN

- Activation function → $f(-\infty)=0$ or $f(-\infty)=-1$ and $f(\infty)=1$
  - linear function

    $f(v)=\beta v$ where β is a constant

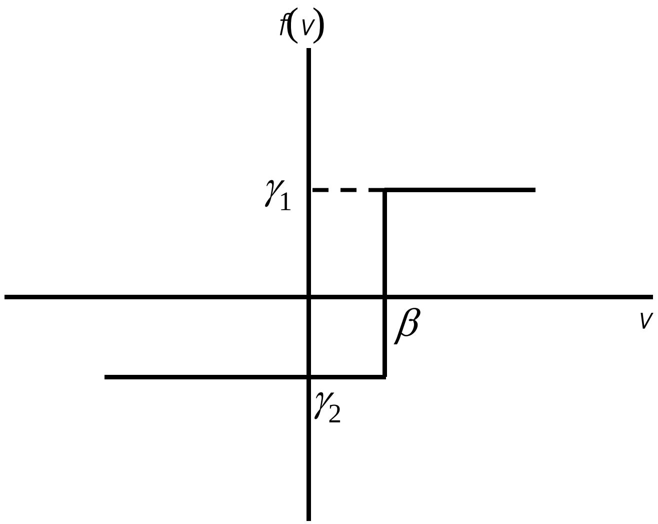


NN

- Step function or unit step function

$$f(v) = \begin{cases} \gamma_1 & \text{if } v \geq \beta \\ \gamma_2 & \text{if } v < \beta \end{cases}$$

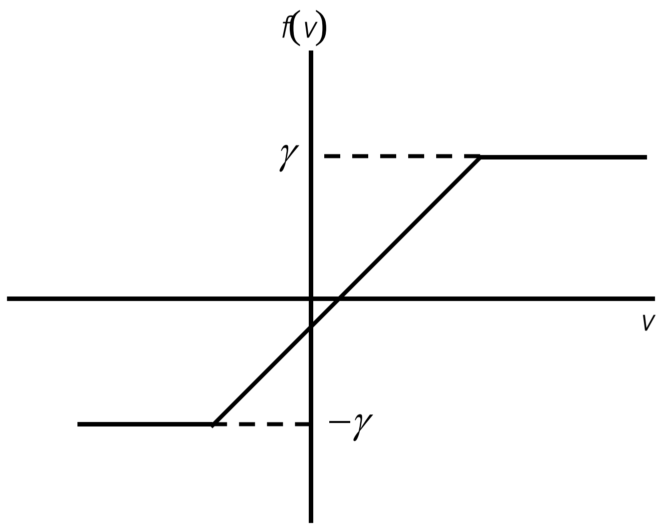
normally γ_1=1 and γ_2=0 or −1



NN

- Ramp function

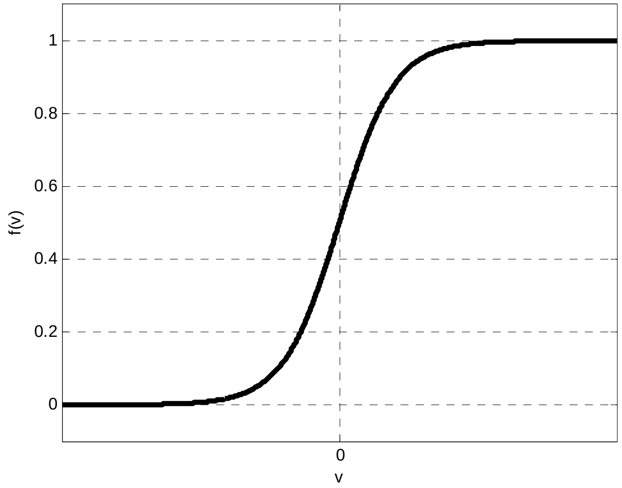
$$f(v)=\begin{cases}\gamma & \text{if } v \geq \beta \\ v & \text{if } -\beta < v < \beta \\ -\gamma & \text{if } v \leq -\beta\end{cases}$$



NN

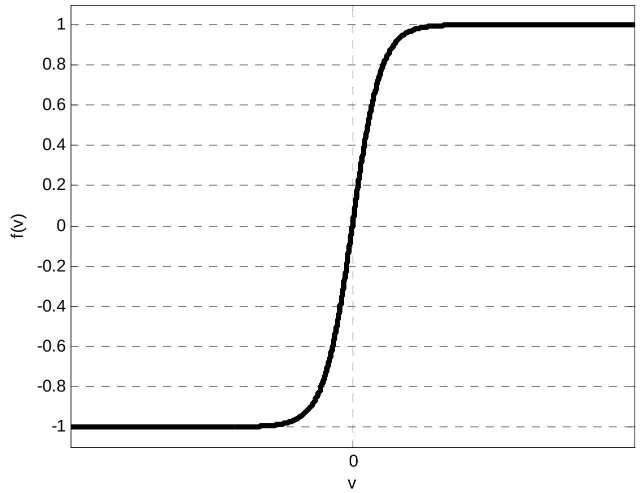
- Sigmoid function
  - Logistic function

$$f\left(v\right)=\frac{1}{1+\exp\left(-av\right)}$$



  - Hyperbolic tangent function

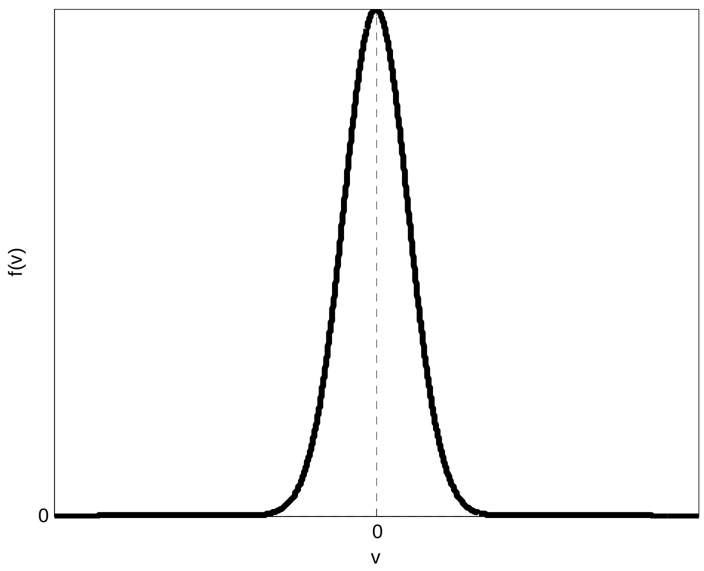
$$f\left(v\right)=\tanh\left(\frac{v}{2}\right)=\frac{1-\exp\left(-v\right)}{1+\exp\left(-v\right)}=\frac{2}{1+\exp\left(-v\right)}-1$$



NN

- Gaussian function

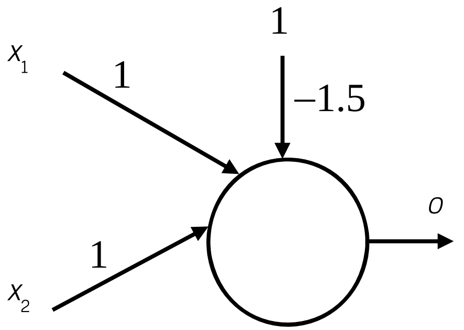
$$f(v) = \exp\left(-\frac{(v-\mu)^2}{\sigma^2}\right)$$ where $\mu \rightarrow$ mean and $\sigma \rightarrow$ standard deviation



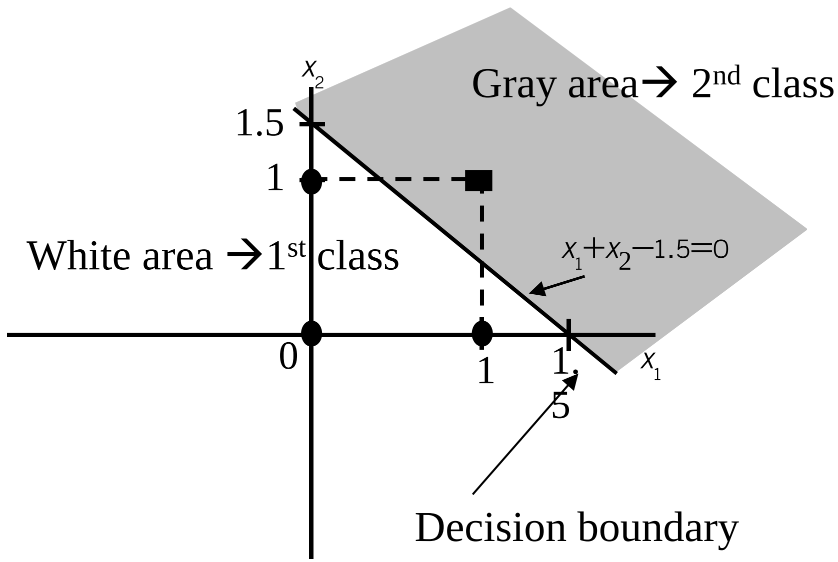
NN

- Example: AND Logic → input vector $[x_1,x_2]^t$ with output y =0 are in the same class (1st class), one with output 1 are in another class (2nd class) → use unit step function with γ_1=1, γ_2=0 and β=0as activation function

x_1	x_2	**Y (desired output)**
0	0	**0**
0	1	**0**
1	0	**0**
1	**1**	**1**



$$v = x_1 + x_2 - 1.5$$

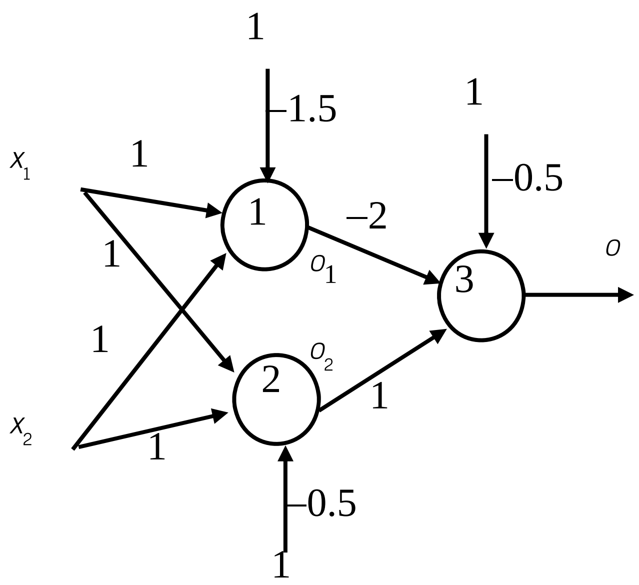


x_1	x_2	v	**o (program output)**
0	0	–1.5	**0**
0	1	–0.5	**0**
1	0	–0.5	**0**
1	**1**	**0.5**	**1**

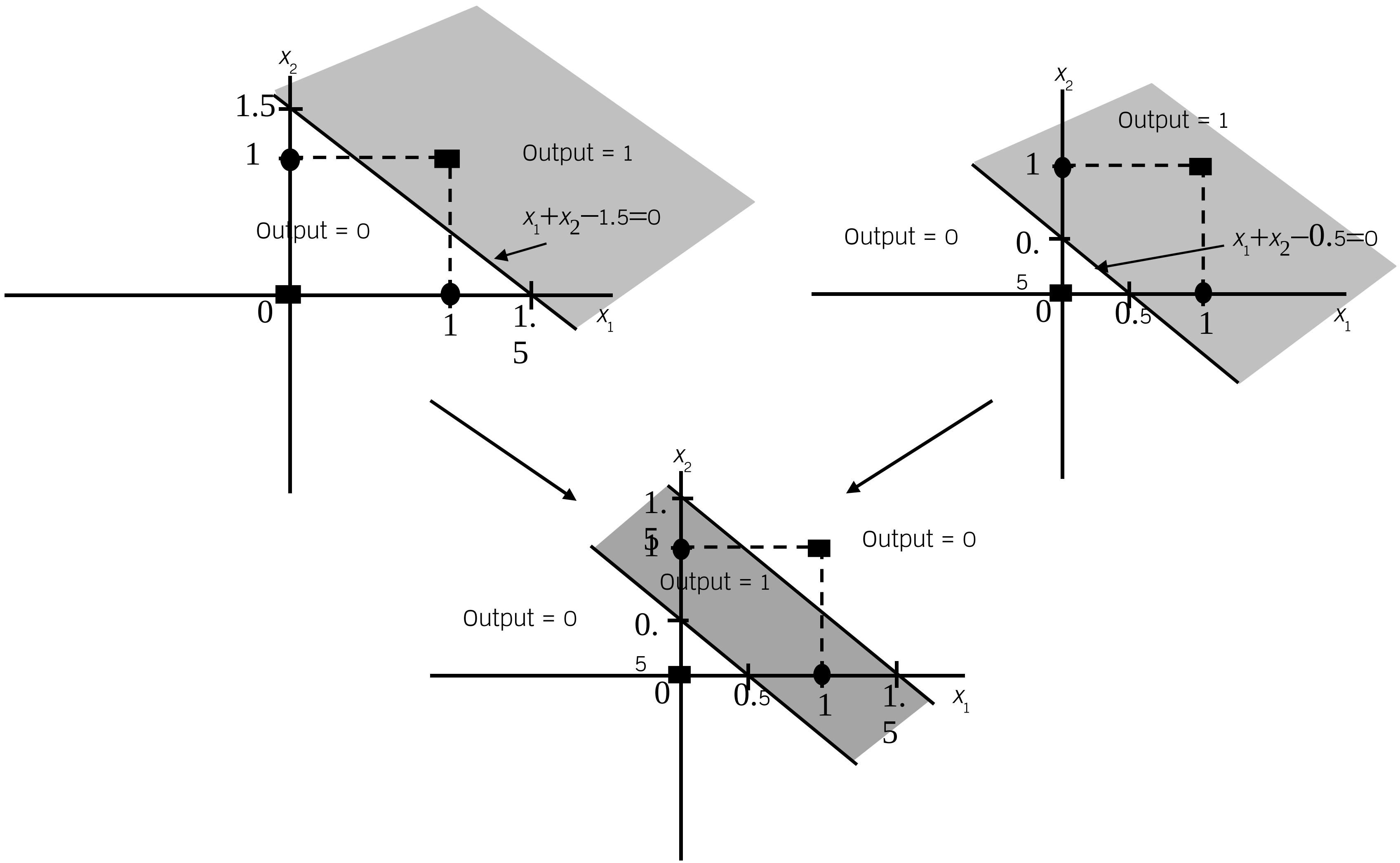
NN

- Example : XOR Logic → input vector $[x_1,x_2]^t$ with output y =0 are in the same class (1st class), one with output 1 are in another class (2nd class) → use unit step function with γ_1=1, γ_2=0 and β=0 as activation function

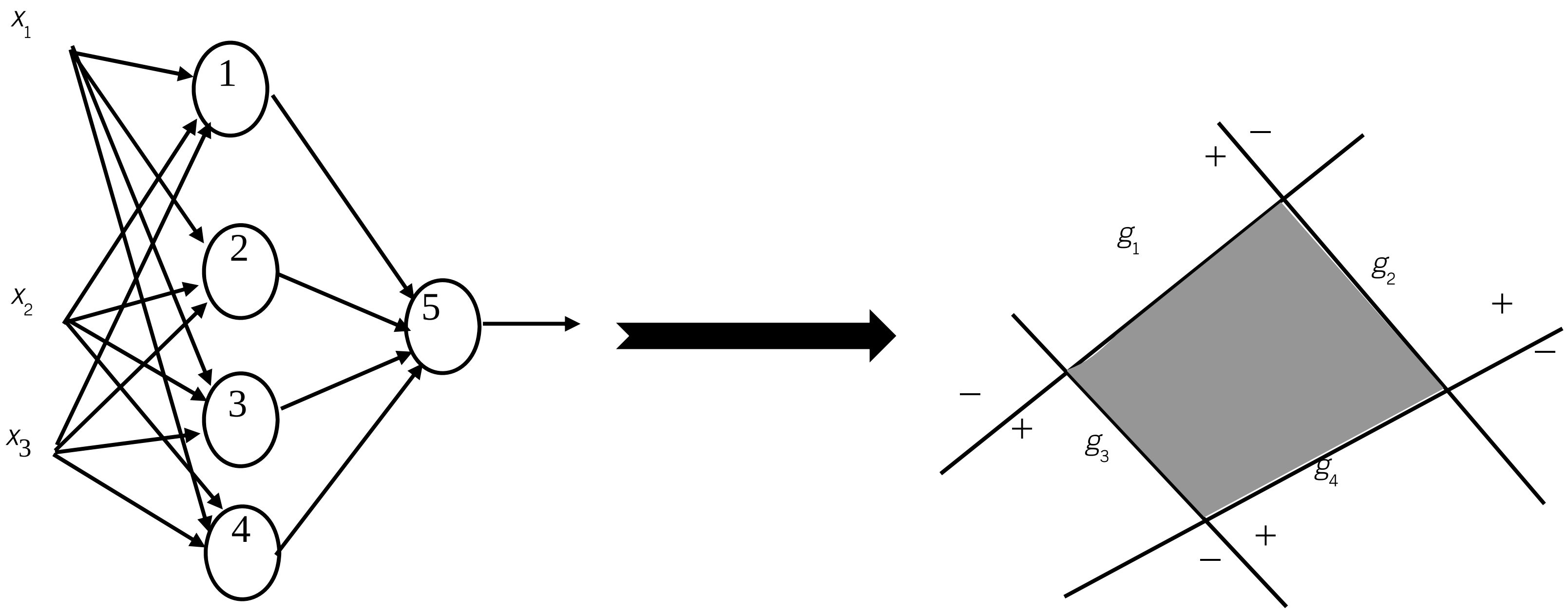
x_1	x_2	Y (desired output)
0	0	**0**
0	1	**1**
1	0	**1**
1	**1**	**0**



x_1	x_2	v_1	o_1	v_2	o_2	o (program output)
0	0	–1.5	0	–0.5	0	–0.5
0	1	–0.5	0	0.5	1	0.5
1	0	–0.5	0	0.5	1	0.5
1	**1**	0.5	1	1.5	1	–1.5

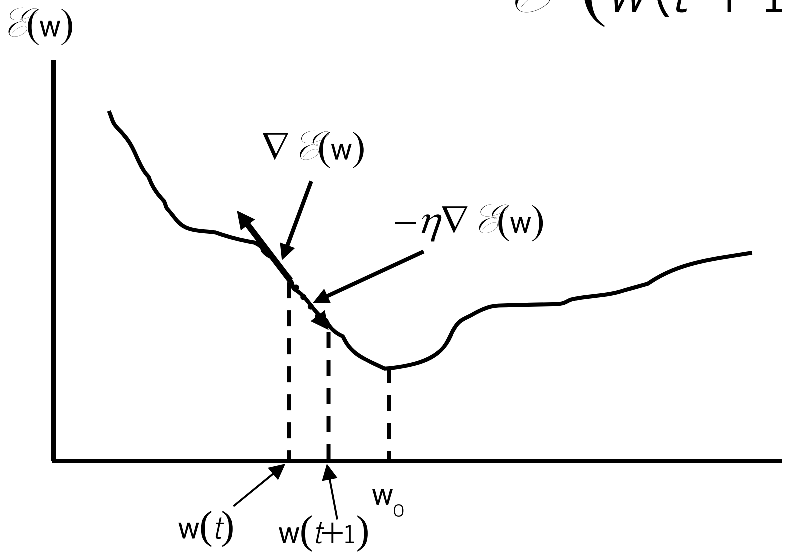


Output = 1
Output = 0
$x_1+x_2-1.5=0$
Output = 1
Output = 0
$x_1+x_2-0.5=0$
Output = 1
Output = 0
Output = 0



One hidden node create one decision boundary→ 4 hidden nodes creates 4 decision boundary→ these boundary is combined at the output node to create decision for the decision making

$$\mathscr{E}\left(w(t+1)\right) < \mathscr{E}\left(w(t)\right)$$



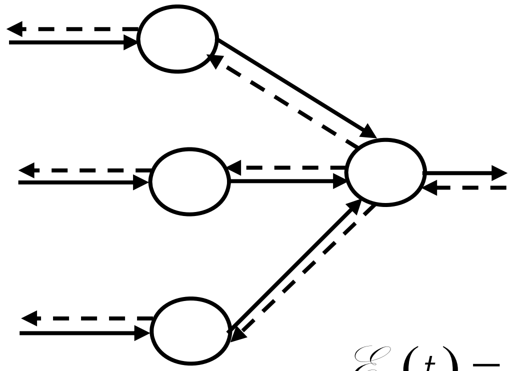
Gradient descent learning rule

Direction of gradient vector ($(\nabla\mathscr{E}(\mathbf{w}))$ is in the increasing direction→if we invert the direction we will have

$$\mathrm{w}(t+1) = \mathrm{w}(t) - \eta\nabla\mathscr{E}(w)$$

New weight that is in the decreasing direction of the error space

- if η is small, the transient response of the algorithm will be overdamped
- If η is large , the transient response of the algorithm will be underdamped
- If η is larger than the critical value, the algorithm will be unstable and may be diverged



$$\mathscr{E}(t) = \frac{1}{2}\sum_{j\in C} e_j^2(t)$$

Backpropagation

- Forward pass
- Backward pass

$$\mathscr{E}_{av} = \frac{1}{N}\sum_{n=1}^{N}\mathscr{E}(n)$$

Multilayer Perceptron

- Initialization
- Forward Computation
  - Output from neuron j in layer l

$$y_j^{(l)}(n) = \varphi_j\left(v_j\left(n\right)\right) \quad \text{where} \quad v_j^{(l)}(n) = \sum_{i=0}^{m_{l-1}} w_{ji}^{(l)}(n) y_i^{(l-1)}(n)$$

where $y_i^{(l-1)}(n)$ is the output signal of neuron i in the previous layer l–1

and $w_{ji}^{(l)}(n)$ is the synaptic weight of neuron j in layer l that is fed from neuron i in layer l–1

if $l = 1$, $y_j^{(0)}(n) = x_j\left(n\right)$ and if $l = L$, $y_j^{(0)}(n) = o_j\left(n\right)$

compute the error: $e_j(n) = d_j(n) - o_j(n)$

- Backward computation
  - Local gradients

$$\delta_j^{(l)}(n) = \begin{cases} e_j^{(L)}(n)\varphi_j{'}\left(v_j^{(L)}(n)\right) & \text{for neuron } j \text{ in output layer } L \\ \varphi_j{'}\left(v_j^{(l)}(n)\right)\sum_k \delta_k^{(l+1)}(n)\, w_{kj}^{(l+1)}(n) & \text{for neuron } j \text{ in output layer } l \end{cases}$$

where $\varphi_j{'}(\cdot)$ denotes differentiation with respect to the argument

  - Update weights

$$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \alpha\left[\Delta w_{ji}^{(l)}(n-1)\right] + \eta\delta_j^{(l)}(n)\, y_j^{(l-1)}(n)$$

$$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \alpha\left[w_{ji}^{(l)}(n) - w_{ji}^{(l)}(n-1)\right] + \eta\delta_j^{(l)}(n)\, y_j^{(l-1)}(n)$$

- Iteration: until the stopping criterion is met

- For $$y_j^{(l)}(t) = \varphi_j^{(l)}\left(v_j^{(l)}(t)\right) = \frac{1}{1+\exp\left(-v_j^{(l)}(t)\right)}$$

$$\frac{\partial y_j^{(l)}(t)}{\partial v_j^{(l)}(t)} = \varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right) = \frac{\exp\left(-v_j^{(l)}(t)\right)}{\left[1+\exp\left(-v_j^{(l)}(t)\right)\right]^2}$$

$$\varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right) = \frac{1}{1+\exp\left(-v_j^{(l)}(t)\right)}\left[1-\frac{1}{1+\exp\left(-v_j^{(l)}(t)\right)}\right]$$

$$\varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right) = y_j^{(l)}(t)\left[1-y_j^{(l)}(t)\right]$$

output layer $$\delta_j^{(L)}(t) = e_j^{(L)}(t)\left[o_j(t)\left[1-o_j(t)\right]\right]$$

hidden layer $$\delta_j^{(l)}(t) = y_j^{(l)}(t)\left[1-y_j^{(l)}(t)\right]\sum_k \delta_k^{(l+1)}(t)\, w_{kj}^{(l+1)}(t)$$

- For $$y_j^{(l)}(t) = \varphi_j^{(l)}\left(v_j^{(l)}(t)\right) = \tanh\left(\frac{v_j^{(l)}(t)}{2}\right) = \frac{2}{1+\exp\left(-v_j^{(l)}(t)\right)} - 1$$

$$\frac{\partial y_j^{(l)}(t)}{\partial v_j^{(l)}(t)} = \varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right) = \frac{2\exp\left(-v_j^{(l)}(t)\right)}{\left[1+\exp\left(-v_j^{(l)}(t)\right)\right]^2}$$

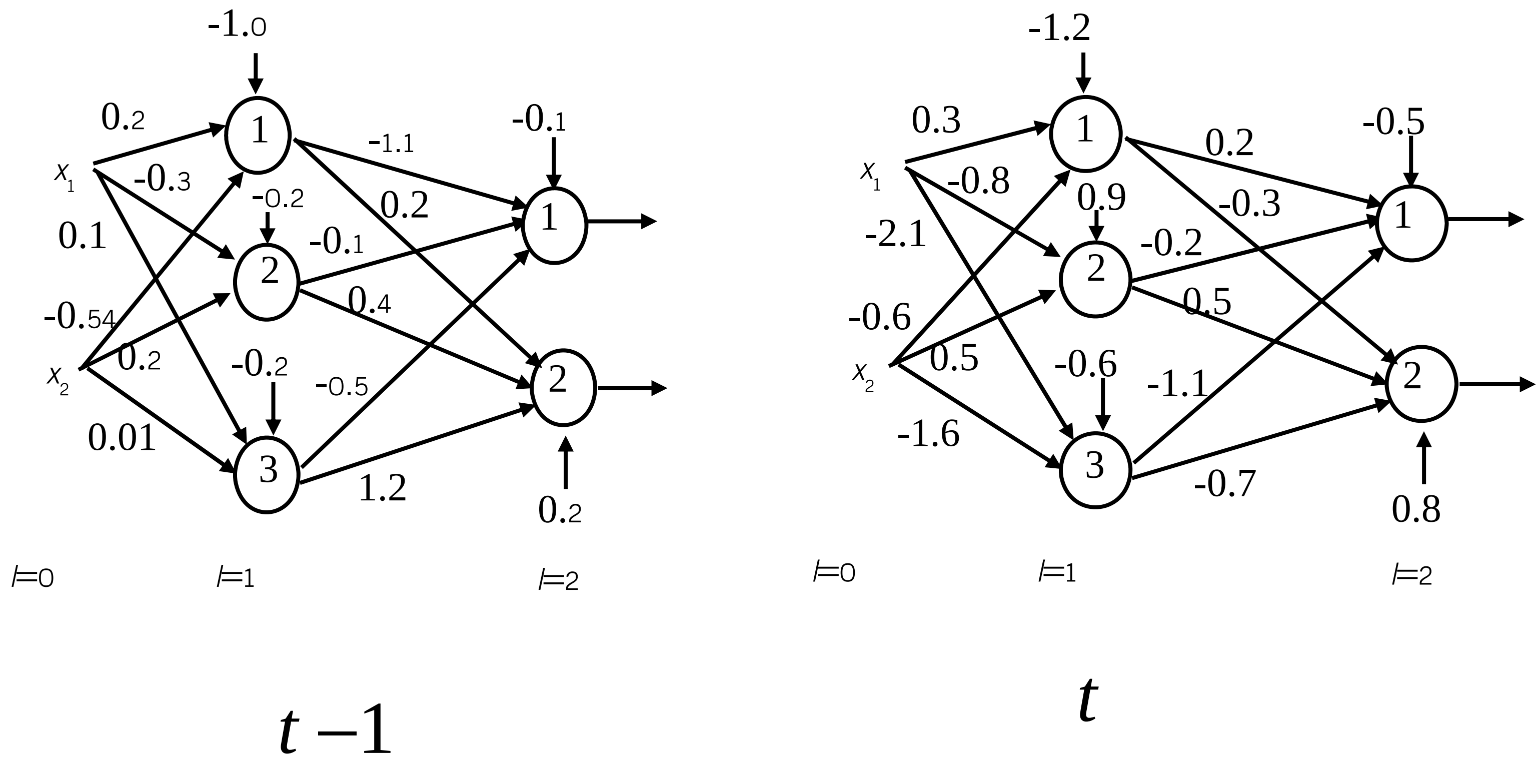
$$\varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right) = \frac{2}{1+\exp\left(-v_j^{(l)}(t)\right)}\left[1 - \frac{1}{1+\exp\left(-v_j^{(l)}(t)\right)}\right]$$

$$\varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right) = 2y_j^{(l)}(t)\left[1 - y_j^{(l)}(t)\right]$$

output layer $$\delta_j^{(L)}(t) = e_j^{(L)}(t)\left[2o_j(t)\left[1 - o_j(t)\right]\right]$$

hidden layer $$\delta_j^{(l)}(t) = 2y_j^{(l)}(t)\left[1 - y_j^{(l)}(t)\right]\sum_k \delta_k^{(l+1)}(t)\, w_{kj}^{(l+1)}(t)$$

- Example: suppose at iteration t $\mathbf{x}(t)=\begin{bmatrix}1\\1\end{bmatrix}$ and $\mathbf{d}(t)=\begin{bmatrix}0\\1\end{bmatrix}$ and η=0.2, α=0.1. Find $w_{32}^{(1)}(t+1)$



$$v_1^{(1)}(t) = \sum_{i=0}^{2} w_{1i}^{(1)} x_i = -1.2 + (0.3)(1) + (-0.6)(1) = -1.5$$

$$y_1^{(1)}(t) = \frac{1}{1 + \exp\left(-v_1^{(1)}(t)\right)} = \frac{1}{1 + \exp(1.5)} = 0.1824$$

$$v_2^{(1)}(t) = \sum_{i=0}^{2} w_{2i}^{(1)} x_i = 0.9 + (-0.8)(1) + (0.5)(1) = 0.6$$

$$y_2^{(1)}(t) = \frac{1}{1 + \exp(-0.6)} = 0.6457$$

$$v_3^{(1)}(t) = \sum_{i=0}^{2} w_{3i}^{(1)} x_i = -0.6 + (0.5)(1) + (-1.6)(1) = -1.7$$

$$y_3^{(1)}(t) = \frac{1}{1 + \exp(1.7)} = 0.1545$$

$$v_1^{(2)}(t) = \sum_{i=0}^{3} w_{1i}^{(2)} y_i^{(1)} = -0.5 + (0.2)(0.1824) + (-0.2)(0.6457) + (-1.1)(0.1545) = -0.7626$$

$$y_1^{(2)}(t) = \frac{1}{1 + \exp(0.7626)} = 0.3181 \quad \longrightarrow \quad e_1^{(2)}(t) = 0 - 0.3181 = -0.3181$$

$$v_2^{(2)}(t) = \sum_{i=0}^{3} w_{1i}^{(2)} y_i^{(1)} = 0.8 + (-0.3)(0.1824) + (0.5)(0.6457) + (-0.7)(0.1545) = 0.96$$

$$y_2^{(2)}(t) = \frac{1}{1+\exp(-0.96)} = 0.7231 \longrightarrow e_2^{(2)}(t) = 1 - 0.7231 = 0.2769$$

$$\delta_1^{(2)}(t) = e_1^{(2)}(t)\Big[o_1(t)\big[1-o_1(t)\big]\Big] = (-0.3181)\big[0.3181\big(1-0.3181\big)\big] = -0.0690$$

$$\delta_2^{(2)}(t) = e_2^{(2)}(t)\Big[o_2(t)\big[1-o_2(t)\big]\Big] = (0.2769)\big[0.7231(1-0.7231)\big] = 0.0554$$

$$\delta_3^{(1)}(t) = y_3^{(1)}(t)\Big[1-y_3^{(1)}(t)\Big]\sum_{k=1}^{2}\delta_k^{(2)}(t)w_{kj}^{(2)}(t)$$

$$\delta_3^{(1)}(t) = 0.1545\big[1-0.1545\big]\big[(-0.0690)(-1.1)+(0.0554)(-0.7)\big] = 0.0048$$

$$\Delta w_{12}^{(2)}(t) = \alpha\Delta w_{12}^{(2)}(t-1)+\eta\delta_1^{(2)}(t)\,y_2^{(1)}(t)$$

$$\Delta w_{12}^{(2)}(t) = 0.1\big(-0.2-(-0.1)\big)+0.2(-0.0690)(0.6457) = -0.0189$$

$$w_{12}^{(2)}(t+1) = w_{12}^{(2)}(t)+\Delta w_{12}^{(2)}(t) = -0.2-0.0189 = -0.2189$$

$$\Delta w_{32}^{(1)}(t) = \alpha\Delta w_{32}^{(1)}(t-1)+\eta\delta_3^{(1)}(t)\,x_2(t)$$

$$\Delta w_{32}^{(1)}(t) = 0.1(-1.6-0.01)+0.2(0.0048)(1) = -0.16$$

$$w_{32}^{(1)}(t+1) = w_{32}^{(1)}(t)+\Delta w_{32}^{(1)}(t) = -1.6-0.16 = -1.76$$

- Generalization
  - Overtrain→look up table→do not want
  - Properly fit→want
  - Factor
    - Size of training data set
    - Architecture of NN
    - Physical complexity of the problem at hand
- Performance factor
  - Data preparation
    - Missing value→discard if have a lot of train data or replace the missing value with the average of that feature or with the most frequent
    - Coding input value
    - Outlier → discard that outlier if known and have a lot of train data if not, adjust the objective function so that it can cope with the outlier→robustness

- Scaling and normalization→ if some features dominate other features→ normalize each feature with its own mean and standard deviation so that every features are in the same range

k feature of sample i → $\hat{x}_{ik} = \dfrac{x_{ik} - \overline{x}_k}{\sigma_k}$ where $\overline{x}_k = \dfrac{1}{N}\sum_{i=1}^{N} x_{ik}$ for $k = 1, 2, \ldots, p$

and $\sigma_k^2 = \dfrac{1}{N-1}\sum_{i=1}^{N}\left(x_{ik} - \overline{x}_k\right)^2$

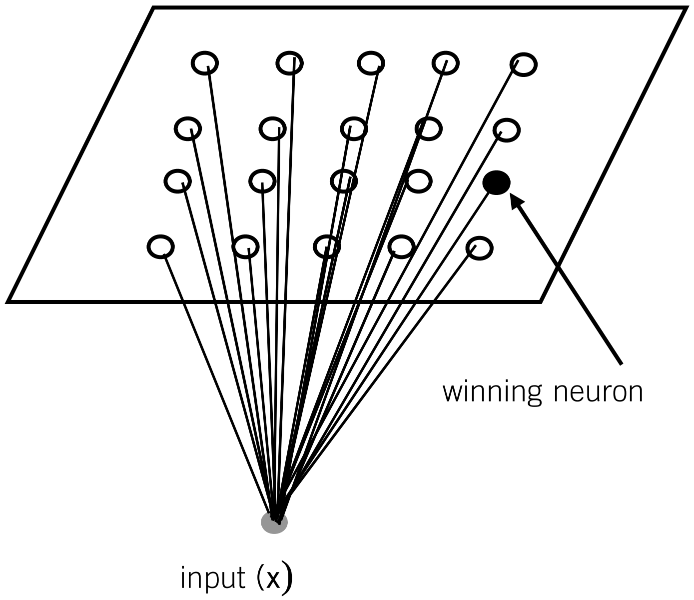
  - output value should be (0.1 or 0.9) or (-0.9 or 0.9) because if we use logistic function or hyperbolic tangent the computed valued will never goes to 0 or 1 (or -1 or 1)→ suppose there are 3 class: sample in class 1 will have the desire output be 0.9 0.1 0.1, that in class 2 will be 0.1 0.9 0.1 and that in class 3 will be 0.1 0.1 0.9→ if use tanh, it should be 0.9 -0.9 0.9 (class 1), -0.9 0.9 -0.9 (class 2) and -0.9 -0.9 0.9 (class 3)

- Or
  - Noise injection→around decision boundary
  - Training set manipulation→ selective presentation→typical pattern and confusing pattern → need priori information

- Initialize weight with

$$\left[\frac{-1}{\sqrt{fanin}}, \frac{1}{\sqrt{fanin}}\right]$$

- Learning rate and momentum rate



Self organizing feature map
self organizing map

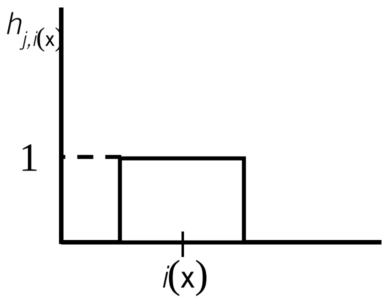
- Competitive

Input $\mathbf{x} = [x_1, x_2, \ldots, x_m]^t$ neuron *j* with $\mathbf{w}_j = [w_{j1}, w_{j2}, \ldots, w_{jm}]^t$

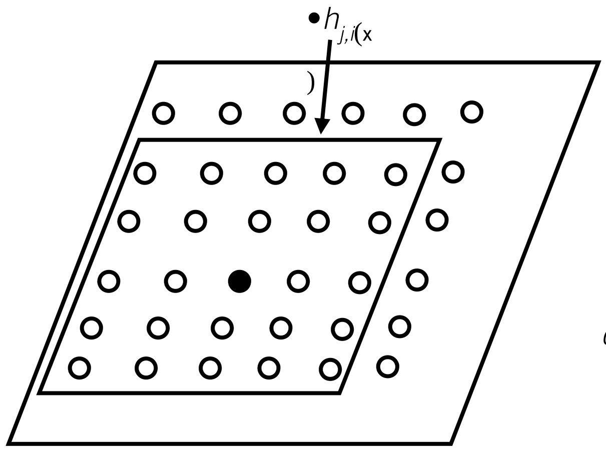
There are *l* neurons

Winning neuron $i(x) = \arg\min_j \|\mathbf{x} - \mathbf{w}_j\|$ for $j \in \mathscr{A}$

- Cooperation



1-d lattice



$$d_{j,i}^2 = \|\mathbf{r}_j - \mathbf{r}_i\|^2$$

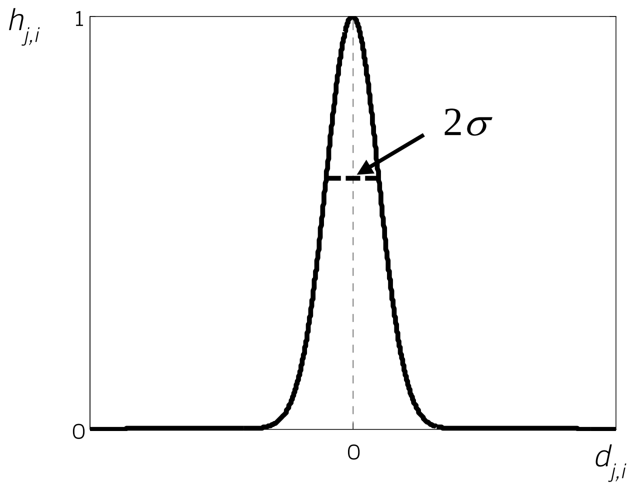
2-d lattice

  - $h_{j,i(\mathbf{x})}$: symmetric about the maximum point defined by $d_{ji}=0$ (at winning neuron)
  - Amplitude of $h_{j,i(\mathbf{x})}$ decrease monotonically with increasing lateral distance d_{ji} , decaying to 0 for $d_{ji}=\infty \rightarrow$ necessary condition for convergence

$$h_{j,i(\mathbf{x})} = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2}\right) \quad \text{for } j \in \mathscr{A}$$

$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right) \quad \text{for } n = 0,1,2,\ldots,$$

$$h_{j,i(\mathbf{x})}(n) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2(n)}\right) \quad \text{for } n = 0,1,2,\ldots,$$



Where $d_{j,i} = |j-i|$ between neuron *j* and winning neuron *i* in case of 1-d lattice and $d_{j,i}^2 = \|\mathbf{r}_j - \mathbf{r}_i\|^2$ for 2-d lattice between r_i and r_j are vector of neuron *i* and *j*

- Synaptic adaptation→since update in 1 direction→might saturate, hence, include forgetting term ($g(y_j)$)

$$\Delta \mathbf{w}_j = \eta y_j \mathbf{x} - g(y_j)\mathbf{w}_j \quad \text{for } j \in \mathcal{A}$$

$$g(y_j) = \eta y_j \quad \text{and} \quad y_j = h_{j,i(\mathbf{x})}$$

$$\Delta \mathbf{w}_j = \eta h_{j,i(\mathbf{x})}(\mathbf{x} - \mathbf{w}_j) \quad \text{for } j \in \mathcal{A}$$

If neighborhood function is rectangle function

$$\Delta \mathbf{w}_j = \begin{cases} \eta(\mathbf{x} - \mathbf{w}_j) & \text{if } j \text{ is inside neighborhood function} \\ 0 & \text{if } j \text{ is outside neighborhood function} \end{cases} \quad \text{for } j \in \mathcal{A}$$

$$\mathbf{w}_j(t+1) = \begin{cases} \mathbf{w}_j(t) + \eta(t)\left(\mathbf{x}(t) - \mathbf{w}_j(t)\right) & \text{if j is inside the neighborhood function} \\ \mathbf{w}_j(t) & \text{if j is not inside the neighborhood function} \end{cases}$$

If neighborhood function is guassian

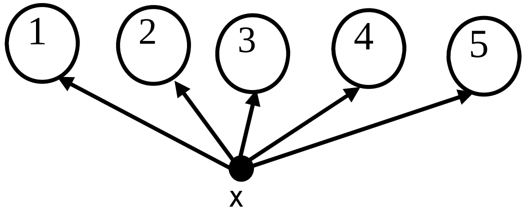
$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \eta(t) h_{j,i(\mathbf{x})}(t)\left(\mathbf{x}(t) - \mathbf{w}_j(t)\right) \quad \text{for } j \in \mathcal{A}$$

$$\eta(n) = \eta_0 \exp\left(-\frac{n}{\tau_2}\right) \quad \text{for } n = 0, 1, 2, \dots,$$

- 2 phase
  - Self-organizing or ordering phase
    - η might start from 0.1and decrease till the value is around 0.01 never goes to 0
    - Neighborhood function start from all neuron centered on the winning neuron and shrink slowly with time
  - Convergence phase
    - η maintained at a small value on the order of 0.01 do not decrease to 0
    - Neighborhood function contain only the nearest neighbor of a winning neuron eventually reduce to 1 or 0 neighbor

- Example

$$w_1 = \begin{bmatrix} 0.5 \\ 0.9 \end{bmatrix} \quad w_2 = \begin{bmatrix} -0.3 \\ -0.1 \end{bmatrix} \quad w_3 = \begin{bmatrix} -0.9 \\ 0.9 \end{bmatrix} \quad w_4 = \begin{bmatrix} 0.3 \\ -0.6 \end{bmatrix} \quad w_5 = \begin{bmatrix} 1.3 \\ -1.6 \end{bmatrix}$$



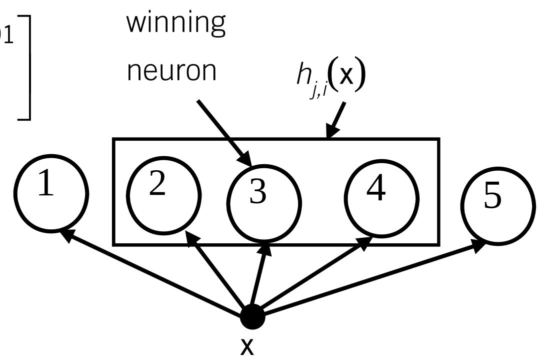
Input $x = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$ learning rate 0.1→ use $d = \sqrt{(x_1 - w_1)^2 + (x_2 - w_2)^2}$ and $h_{ji(\mathbf{x})}$ has the radius of 1

d(1) = 1.5033, d(2) =1.3038, d(3) = 0.1414, d(4) = 2.0616, d(5) = 3.4713

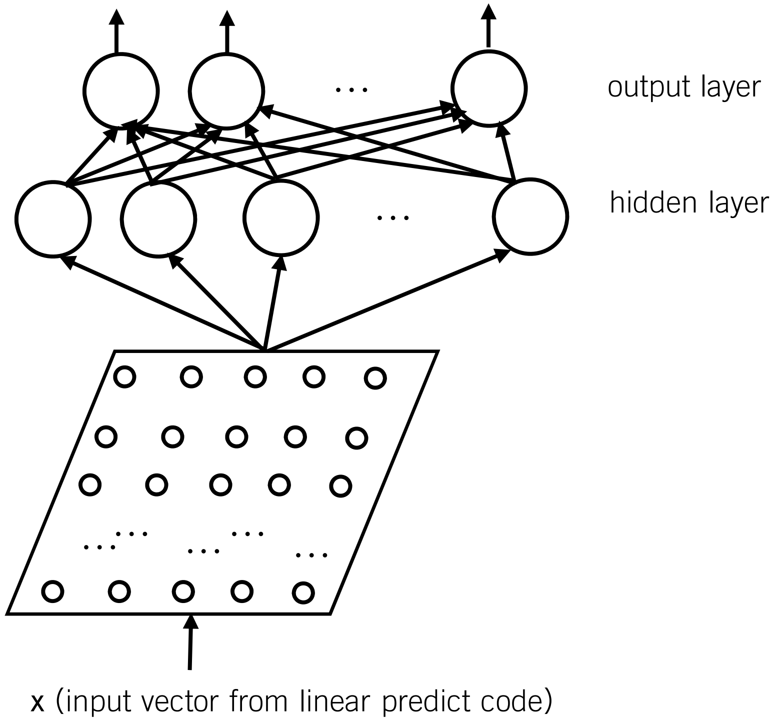
Winning neuron is neuron 3 ($i(\mathbf{x}) = 3$) , neighbor with radius of 1 will be node 2 and 4 only, hence, update weight at nodes 2, 3,and 4

$$w_2 = \begin{bmatrix} -0.3 \\ -0.1 \end{bmatrix} + 0.1\left(\begin{bmatrix} -1 \\ 1 \end{bmatrix} - \begin{bmatrix} -0.3 \\ -0.1 \end{bmatrix}\right) = \begin{bmatrix} -0.37 \\ 0.01 \end{bmatrix} \quad w_3 = \begin{bmatrix} -0.9 \\ 0.9 \end{bmatrix} + 0.1\left(\begin{bmatrix} -1 \\ 1 \end{bmatrix} - \begin{bmatrix} -0.9 \\ 0.9 \end{bmatrix}\right) = \begin{bmatrix} -0.91 \\ 0.91 \end{bmatrix}$$

$$w_4 = \begin{bmatrix} 0.3 \\ -0.6 \end{bmatrix} + 0.1\left(\begin{bmatrix} -1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0.3 \\ -0.6 \end{bmatrix}\right) = \begin{bmatrix} 0.17 \\ -0.44 \end{bmatrix} \quad w_1 = \begin{bmatrix} 0.5 \\ 0.9 \end{bmatrix} \quad w_5 = \begin{bmatrix} 1.3 \\ -1.6 \end{bmatrix}$$



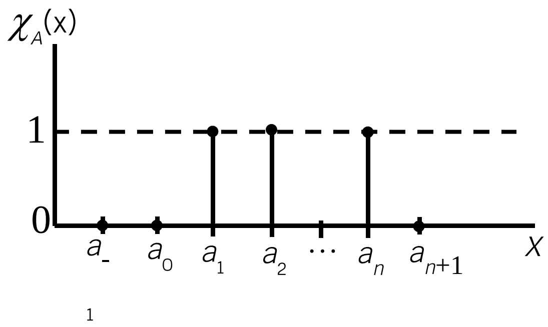
- SOFM properties
  - Approximation of the input space
  - Topological ordering
  - Density matching
  - Feature selection



Fuzzy system

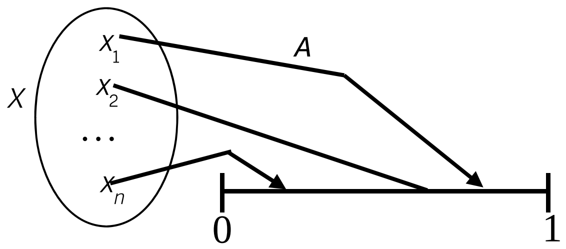
- Uncertainty and complexity
  - Driving in the unfamiliar condition
- Natural language
  - Cold: people in the north and in the south know what cold is but when it happens people in these two area will say differently→ people in the north might say that 10 degree celsius is cold but people in the south might say that 20 degree celsius is cold
- Heap paradox

- Characteristic function → crisp set

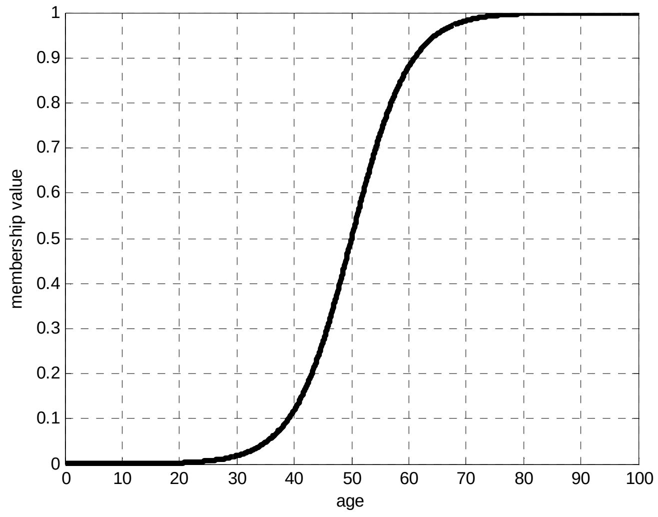


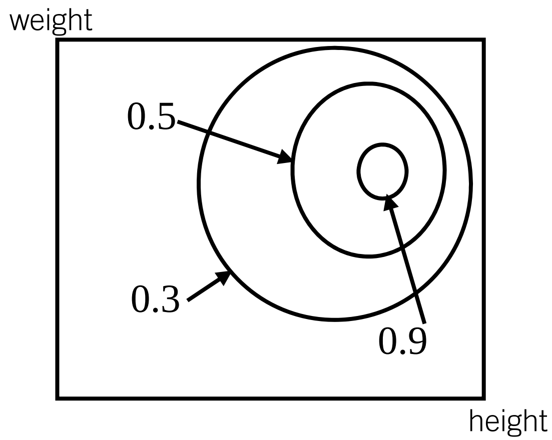
- Membership function → fuzzy set

$\boldsymbol{A}: X \to [0,1]$ หรือ $\mu_{\boldsymbol{A}}: X \to [0,1]$



Membership function of old

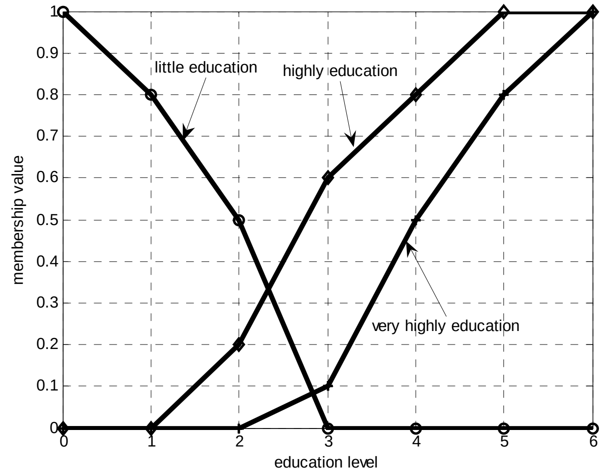




Membership function of big

Countour diagram

No.	Education level
0	No education
1	Elementary school
2	High school
3	2-year college
4	Bachelor's degree
5	Master's degree
6	Doctoral degree



name (symbol)	Membership value in fuzzy set A
Carry (x_1)	0.8
Bill (x_2)	0.3
J-H (x_3)	0.5
Wabei (x_4)	0.9

$$\longrightarrow \quad \boldsymbol{A} = \frac{0.8}{\text{Carry}} + \frac{0.3}{\text{Bill}} + \frac{0.5}{\text{J-H}} + \frac{0.9}{\text{Wabei}}$$

How to write membership function as an equation

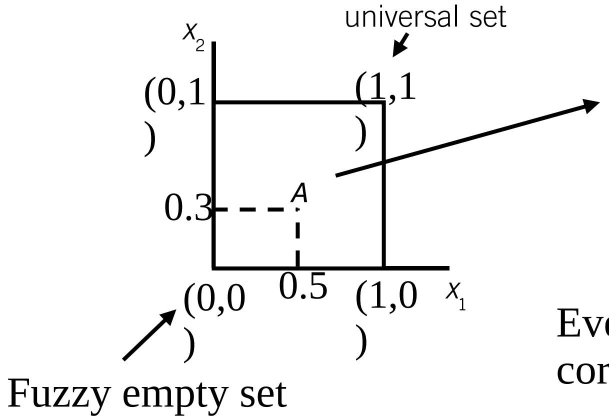
$$\boldsymbol{A} = \sum \frac{\boldsymbol{A}(x)}{x}$$

discrete

$$\boldsymbol{A} = \int_{X} \frac{\boldsymbol{A}(x)}{x}$$

continuous

Geometric representation



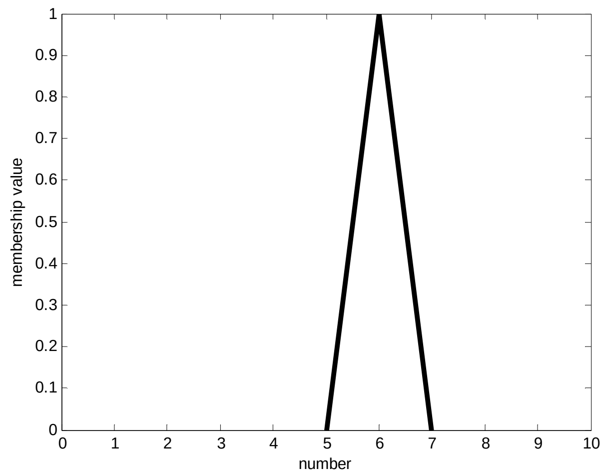
$$A = \frac{0.5}{x_1} + \frac{0.3}{x_2}$$

Every point in this area are fuzzy set→including 4 corner points

Analytic representation : function

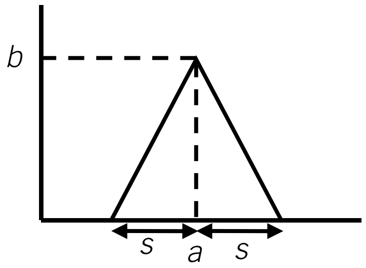
example

$$\boldsymbol{A}(x)=\begin{cases} x-5 & 5\le x\le 6 \\ 7-x & 6\le x\le 7 \\ 0 & \text{else} \end{cases} \longrightarrow$$

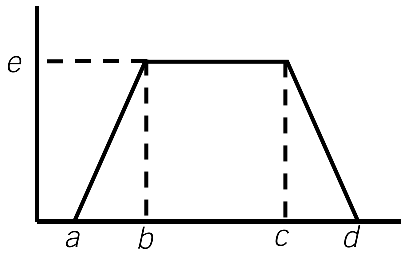


Analytic representation

Triangular shape

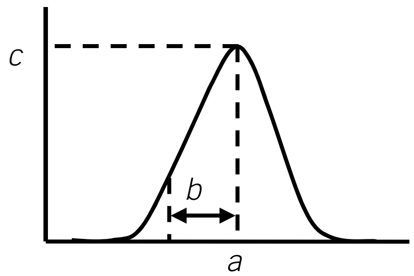


$$\boldsymbol{A}(x)=\begin{cases} b\left(1-\dfrac{|x-a|}{s}\right) & a-s\leq x\leq a+s \\ 0 & \text{else} \end{cases}$$



$$\boldsymbol{A}(x)=\begin{cases} \dfrac{(a-x)e}{a-b} & a\leq x\leq b \\ e & b\leq x\leq c \\ \dfrac{(d-x)e}{d-c} & c\leq x\leq d \\ 0 & \text{else} \end{cases}$$

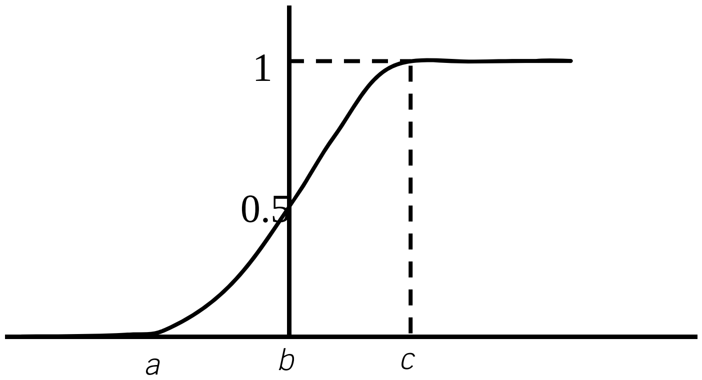
Trapezoidal shape



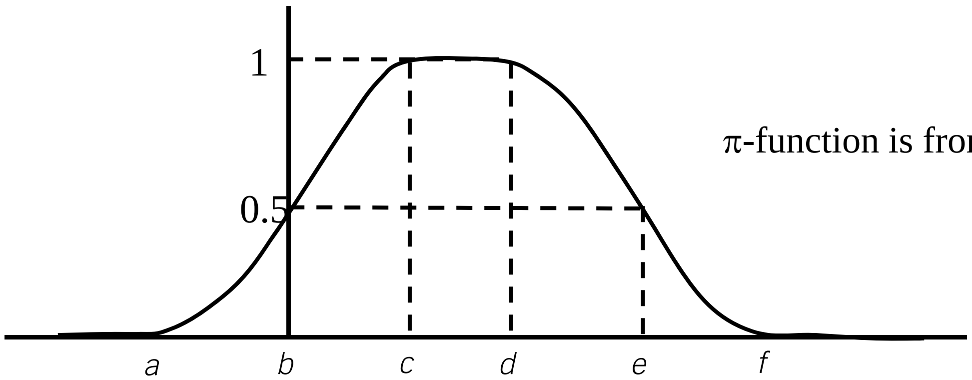
$$\boldsymbol{A}(x)=ce^{\frac{-(x-a)^2}{b}}$$

Bell-shape

S-function



$$s(x)=\begin{cases}0 & 0\le x\le a\\ \frac{1}{2}\left(\frac{x-a}{b-a}\right)^2 & a\le x\le b\\ 1-\frac{1}{2}\left(\frac{x-c}{c-b}\right)^2 & b\le x\le c\end{cases}$$

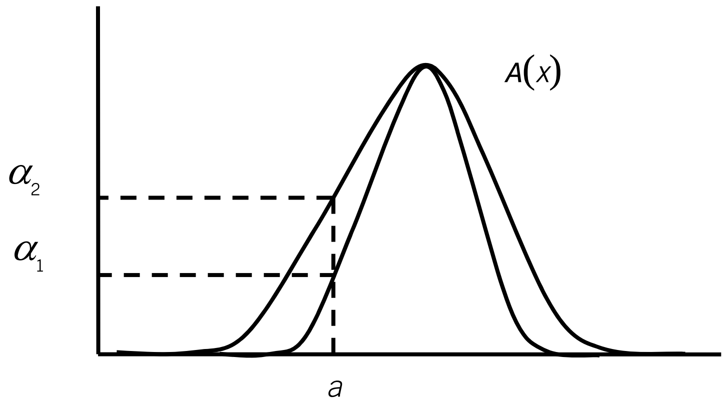


π-function is from S-function + reflected of S-function

Interval-valued fuzzy set

$$\boldsymbol{A} : X \longrightarrow \mathcal{E}([0,1])$$

$\mathcal{E}([0,1])$ denote the family of all closed interval of real number in [0,1] ($\mathcal{E}([0,1]) \subset \mathrm{P}([0,1])$)

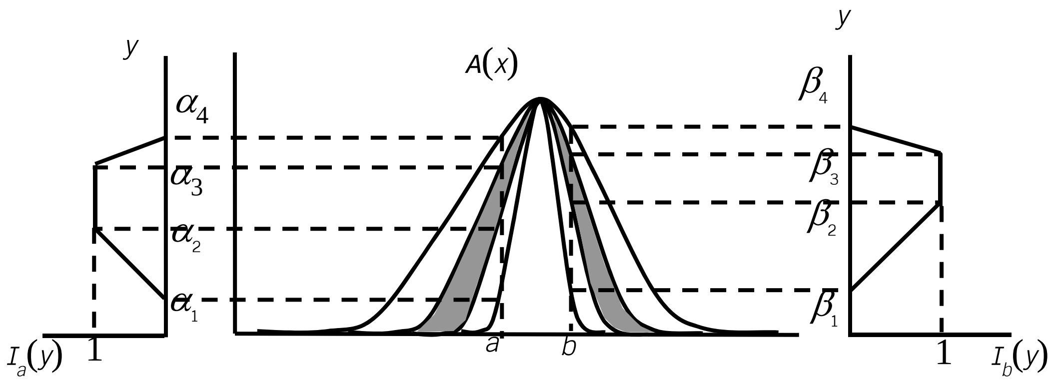


$$\boldsymbol{A}(a) = [\alpha_1, \alpha_2]$$

Type-2 fuzzy set

$$\boldsymbol{A} : X \longrightarrow \tilde{\mathcal{P}}([0,1])$$

$\tilde{\mathcal{P}}([0,1])$ denote the set of all ordinary fuzzy sets that can be defined within the universal [0,1]

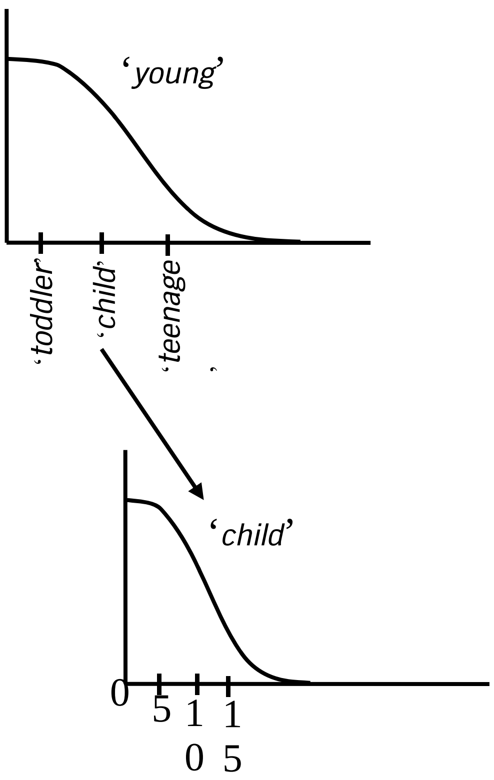


$\boldsymbol{A}(a)$ = fuzzy set (α_1, α_2, α_3, α_4) $\boldsymbol{A}(b)$ = fuzzy set (β_1, β_2, β_3, β_4)

Level 2 fuzzy set

$$\boldsymbol{A} : \tilde{\mathcal{P}}(X) \rightarrow ([0,1])$$

$\tilde{\mathcal{P}}(X)$ denote fuzzy power set of **X**

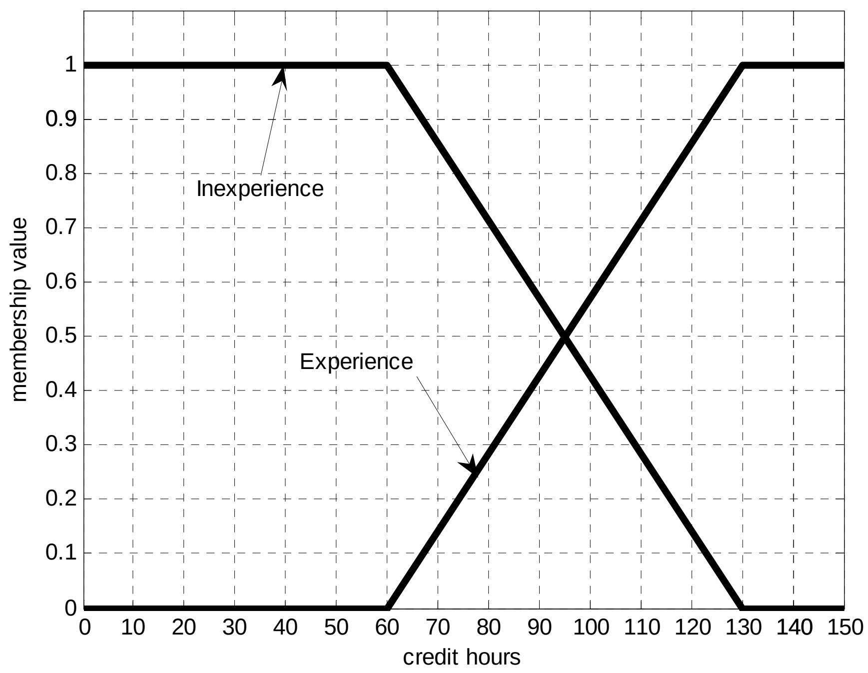


Standard fuzzy complement

$\boldsymbol{A}(x)$ express degree to which x belong to fuzzy set $\boldsymbol{A}$

$\overline{A}(x)$ express degree to which x does not belong to fuzzy set $\boldsymbol{A}$

$$\overline{A}(x) = 1 - A(x) \quad \forall x \in X$$



Standard fuzzy union

$$(\boldsymbol{A} \cup \boldsymbol{B})(x) = \max[\boldsymbol{A}(x), \boldsymbol{B}(x)]$$

patient	high blood pressure($\boldsymbol{A}$)	High fever ($\boldsymbol{B}$)	High blood pressure or high fever ($\boldsymbol{A} \cup \boldsymbol{B}$)
1	1	1	1
2	0.5	0.6	0.6
3	1	0.1	1
⋮			
n	0.1	0.7	0.7

Do not satisfy the law of excluded middle

$(A \cup \overline{A})(x)$ will not equal to universal set

Standard fuzzy intersection

$$(\boldsymbol{A} \cap \boldsymbol{B})(x) = \min[\boldsymbol{A}(x), \boldsymbol{B}(x)]$$

river	Long river (***A***)	Nevigable (***B)***	Long and nevigable (***A***∩***B***)
Amazon	1	0.8	0.8
Nile	0.9	0.7	0.7
Yang-Tsi	0.8	0.8	0.8
Danube	0.5	0.6	0.5
Rhine	0.4	0.3	0.3

Do not satisfy the law of contradiction

$\left(A \cap \overline{A}\right)(x)$ will not equal to fuzzy empty set

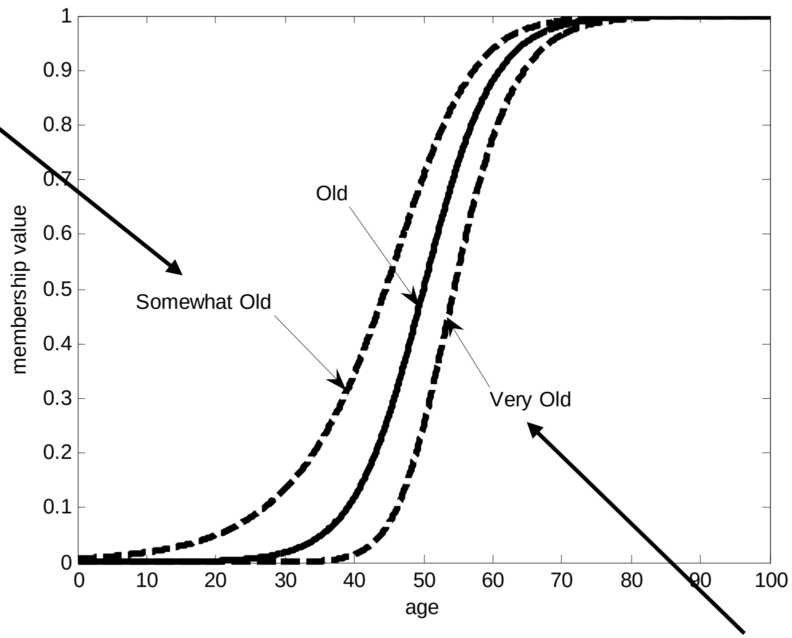
Inclusion: $\boldsymbol{A} \subseteq \boldsymbol{B}$ if $\boldsymbol{A}(x) \leq \boldsymbol{B}(x)\ \forall x \in \mathbf{X}$

Equality: $\boldsymbol{A} = \boldsymbol{B}$ if $\boldsymbol{A}(x) = \boldsymbol{B}(x)\ \forall x \in \mathbf{X}$

Unary operation: $\boldsymbol{A}^a(x) = (\boldsymbol{A}(x))^a$

if $a > 1 \rightarrow$ more specific, if $a < 1 \rightarrow$ less specific

somewhat old are from old$^{0.5}$



Very old are from old^2

Support of fuzzy set $\boldsymbol{A}$ (supp($\boldsymbol{A}$)):

$$\text{supp}(\boldsymbol{A}) = \{x \in X \mid \boldsymbol{A}(x) > 0\}$$

Height of fuzzy set $\boldsymbol{A}$ (h($\boldsymbol{A}$)) : $h(A) = \sup_x A(x)$

Largest value of membership value obtained by any element in that set that is >0

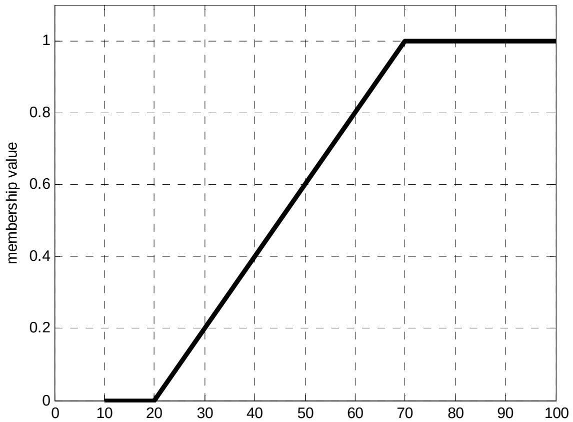
If h($\boldsymbol{A}$)=1 → normal fuzzy set
If h($\boldsymbol{A}$)<1 → supnormal fuzzy set
If h($\boldsymbol{A}$)>1 → supernormal fuzzy set

Core of fuzzy set $\boldsymbol{A}$ (core($\boldsymbol{A}$)) :

core($\boldsymbol{A}$)=$\{x \in X \mid \boldsymbol{A}(x) \geq h(\boldsymbol{A})\}$ since height is h($\boldsymbol{A}$) then we can have
core($\boldsymbol{A}$)=$\{x \in X \mid \boldsymbol{A}(x) = h(\boldsymbol{A})\}$

α**-cut of fuzzy set** $\boldsymbol{A}$ $({}^{\alpha}\boldsymbol{A})$: ${}^{\alpha}\boldsymbol{A} = \{x \in X \mid \boldsymbol{A}(x) \geq \alpha\}$

Strong α-cut of fuzzy set $\boldsymbol{A}$ $({}^{\alpha+}\boldsymbol{A})$: ${}^{\alpha+}\boldsymbol{A} = \{x \in X \mid \boldsymbol{A}(x) > \alpha\}$



If $\alpha_1 < \alpha_2$ then ${}^{\alpha 1}\boldsymbol{A} \supseteq {}^{\alpha 2}\boldsymbol{A}$
and ${}^{\alpha 1}\boldsymbol{A} \cap {}^{\alpha 2}\boldsymbol{A} = {}^{\alpha 2}\boldsymbol{A}$, ${}^{\alpha 1}\boldsymbol{A} \cup {}^{\alpha 2}\boldsymbol{A} = {}^{\alpha 1}\boldsymbol{A}$

If $\alpha_1 < \alpha_2$ then ${}^{\alpha 1+}\boldsymbol{A} \supseteq {}^{\alpha 2+}\boldsymbol{A}$
and ${}^{\alpha 1+}\boldsymbol{A} \cap {}^{\alpha 2+}\boldsymbol{A} = {}^{\alpha 2+}\boldsymbol{A}$, ${}^{\alpha 1+}\boldsymbol{A} \cup {}^{\alpha 2+}\boldsymbol{A} = {}^{\alpha 1+}\boldsymbol{A}$

${}^{0}\boldsymbol{E} = [0,100]$ or ${}^{0.2}\boldsymbol{E} = [30,100]$ or ${}^{1}\boldsymbol{E} = [70,100]$ (core($\boldsymbol{E}$))

${}^{0+}\boldsymbol{E} = (20,100]$ (supp($\boldsymbol{E}$)) or ${}^{0.2+}\boldsymbol{E} = (30,100]$ or ${}^{1+}\boldsymbol{E} = \varnothing$

Level set of fuzzy set $\boldsymbol{A}$ ($L_{\boldsymbol{A}}$ or $\wedge_{\boldsymbol{A}}$): $L_{\boldsymbol{A}} = \wedge_{\boldsymbol{A}} = \{\alpha \mid \boldsymbol{A}(x)=\alpha; \exists x \in X\}$

Special fuzzy set from α-cut (${}_{\alpha}\boldsymbol{A}$)

$$ {}_{\alpha}\boldsymbol{A}(x) = \alpha({}^{\alpha}\boldsymbol{A}(x)) \qquad \forall x \in \mathbf{X} $$

<u>Example</u>

$\boldsymbol{A} = 0.2/x_1 + 0.4/x_2 + 0.6/x_3 + 0.8/x_4 + 1/x_5$
We have $L_{\boldsymbol{A}} = \{0.2, 0.4, 0.6, 0.8, 1\}$

α-cut of $\boldsymbol{A}$ will be

${}^{0.2}\boldsymbol{A} = \{x_1, x_2, x_3, x_4, x_5\} = 1/ x_1 + 1/ x_2 + 1/ x_3 +1/ x_4 +1/ x_5$
${}^{0.4}\boldsymbol{A} = 0/ x_1 + 1/ x_2 + 1/ x_3 +1/ x_4 +1/ x_5$
${}^{0.6}\boldsymbol{A} = 0/ x_1 + 0/ x_2 + 1/ x_3 +1/ x_4 +1/ x_5$
${}^{0.8}\boldsymbol{A} = 0/ x_1 + 0/ x_2 + 0/ x_3 +1/ x_4 +1/ x_5$
${}^{1}\boldsymbol{A} = 0/ x_1 + 0/ x_2 + 0/ x_3 +0/ x_4 +1/ x_5$

Special fuzzy set will be

${}_{0.2}\boldsymbol{A} = 0.2/ x_1 + 0.2/ x_2 + 0.2/ x_3 +0.2/ x_4 +0.2/ x_5$
${}_{0.4}\boldsymbol{A} = 0/ x_1 + 0.4/ x_2 + 0.4/ x_3 +0.4/ x_4 +0.4/ x_5$
${}_{0.6}\boldsymbol{A} = 0/ x_1 + 0/ x_2 + 0.6/ x_3 +0.6/ x_4 +0.6/ x_5$
${}_{0.8}\boldsymbol{A} = 0/ x_1 + 0/ x_2 + 0/ x_3 +0.8/ x_4 +0.8/ x_5$
${}_{1}\boldsymbol{A} = 0/ x_1 + 0/ x_2 + 0/ x_3 +0/ x_4 +1/ x_5$

Decomposition theorem

From special fuzzy set in the previous example

$_{0.2}\boldsymbol{A} = 0.2/\ x_1 + 0.2/\ x_2 + 0.2/\ x_3 + 0.2/\ x_4 + 0.2/\ x_5$

$_{0.4}\boldsymbol{A} = 0/\ x_1 + 0.4/\ x_2 + 0.4/\ x_3 + 0.4/\ x_4 + 0.4/\ x_5$

$_{0.6}\boldsymbol{A} = 0/\ x_1 + 0/\ x_2 + 0.6/\ x_3 + 0.6/\ x_4 + 0.6/\ x_5$

$_{0.8}\boldsymbol{A} = 0/\ x_1 + 0/\ x_2 + 0/\ x_3 + 0.8/\ x_4 + 0.8/\ x_5$

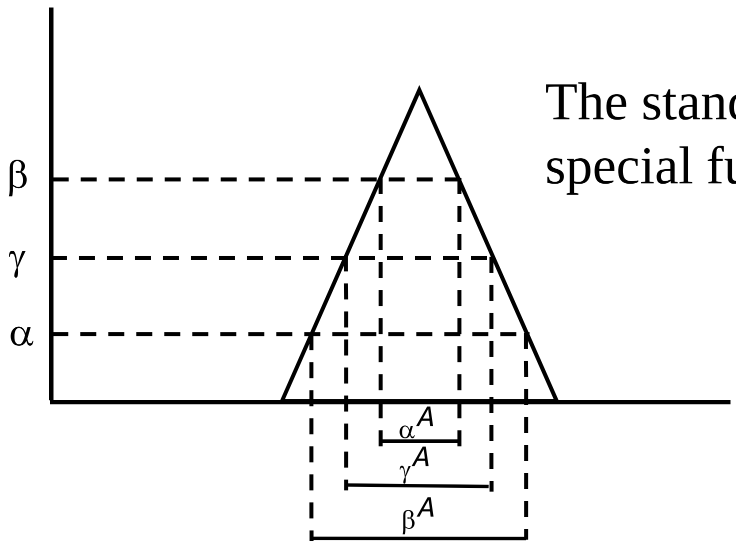
$_{1}\boldsymbol{A} = 0/\ x_1 + 0/\ x_2 + 0/\ x_3 + 0/\ x_4 + 1/\ x_5$

We can see that $({}_{0.2}\boldsymbol{A} \cup {}_{0.4}\boldsymbol{A} \cup {}_{0.6}\boldsymbol{A} \cup {}_{0.8}\boldsymbol{A} \cup {}_{1}\boldsymbol{A}) \rightarrow \boldsymbol{A}$

First theorem:

For and $\boldsymbol{A} \in \tilde{\mathcal{P}}(x)$, $A = \bigcup_{\alpha \in [0,1]} {}_{\alpha}A$

where ${}_{\alpha}\boldsymbol{A}(x) = \alpha({}^{\alpha}\boldsymbol{A}(x))$, $\forall x \in \mathbf{X}$ and $\cup$ is a standard fuzzy union



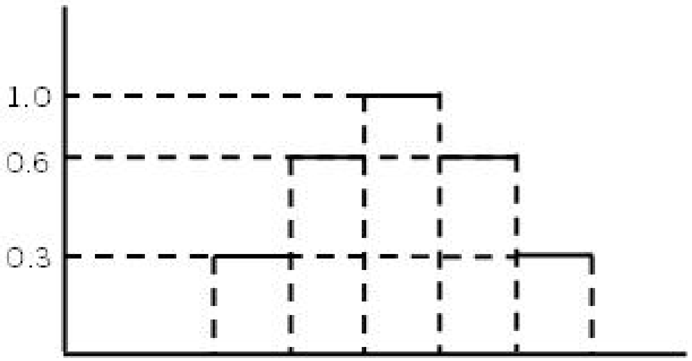
The standard fuzzy union of 3 special fuzzy sets $\rightarrow \boldsymbol{A}$

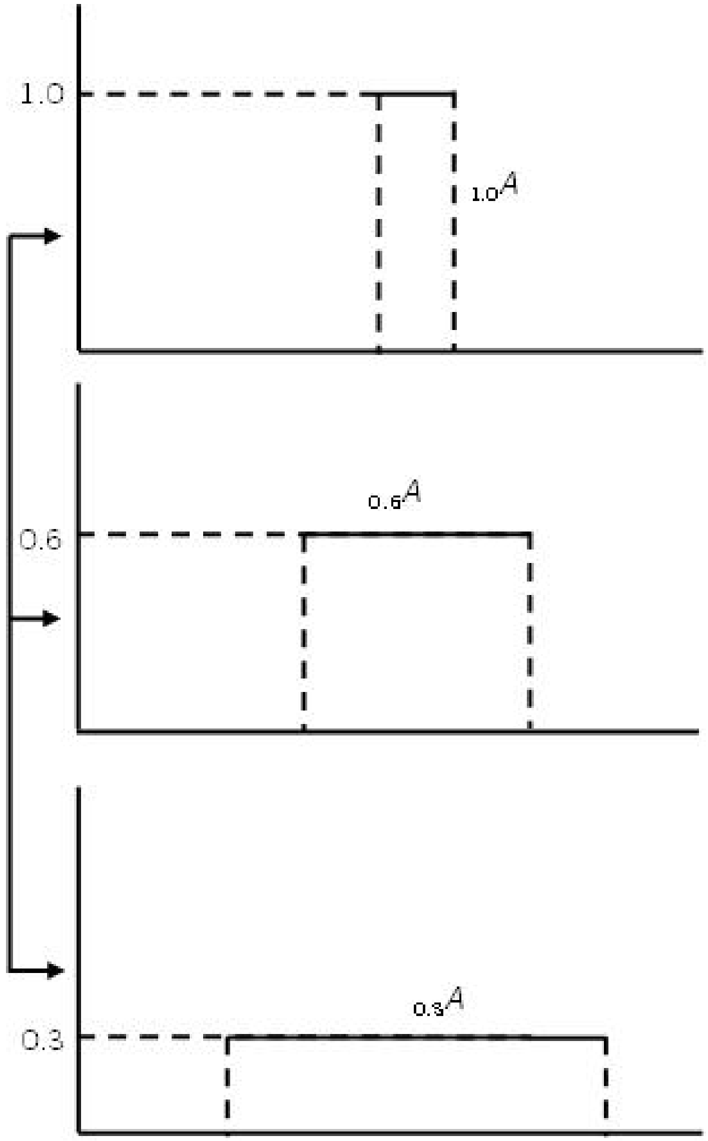
Second theorem:
For and $\boldsymbol{A} \in \tilde{\mathcal{P}}(X)$, $A = \bigcup_{\alpha \in [0,1]} {}_{\alpha+}A$

where ${}_{\alpha+}\boldsymbol{A}(x) = \alpha({}^{\alpha+}\boldsymbol{A}(x))$, $\forall x \in \mathbf{X}$ and $\cup$ is a standard fuzzy union

Third theorem:
For and $\boldsymbol{A} \in \tilde{\mathcal{P}}(X)$, $A = \bigcup_{\alpha \in \wedge_A} {}_{\alpha}A$

where ${}_{\alpha}\boldsymbol{A}(x) = \alpha({}^{\alpha}\boldsymbol{A}(x))$, $\forall x \in \mathbf{X}$ and $\cup$ is a standard fuzzy union





$\wedge_{\boldsymbol{A}} = \{0, 0.3, 0.6, 1\}$ and ${}_0\boldsymbol{A} = \varnothing$
therefore $\boldsymbol{A}$ is a union of ${}_{0.3}\boldsymbol{A}$
${}_{0.6}\boldsymbol{A}$ and ${}_1\boldsymbol{A}$

This example shows that for discrete fuzzy set only 1st and 3rd theorem will work fine

Convex fuzzy set

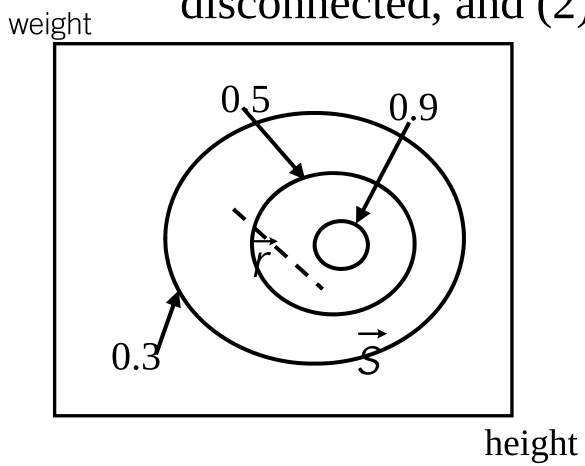
A is a convex fuzzy set if

$$A\left(\lambda\vec{r}+(1-\lambda)\vec{s}\right)\geq \min\left(A(\vec{r}),A(\vec{s})\right) \quad \text{where } 0\leq\lambda\leq 1$$

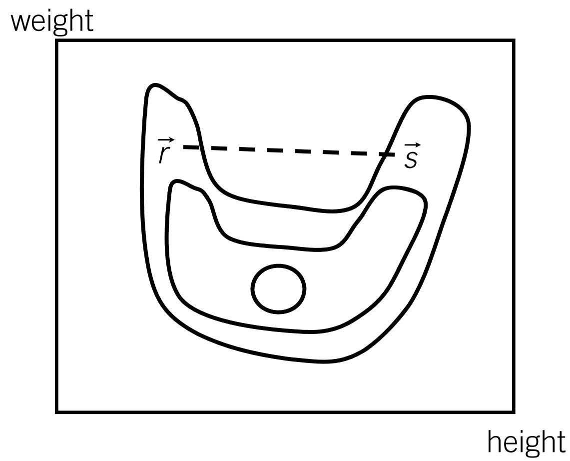
Meaning that membership values of all the points on the line connecting $\vec{r}$ and $\vec{S}$ is bigger than or equal to the minimum membership values of $\vec{r}$ and $\vec{s}$

Or

If ***A*** is a convex fuzzy set then (1) α-cut for all α are not disconnected, and (2)each α-cut is convex in crisp sense



Convex fuzzy set



All the points of the dashed line have a membership values bigger than or equal to the minimum membership value of vector *r* and vector *s*

Not convex fuzzy set