

4 - Dimensionando a rede

Quantas camadas de neurônios ?

Quantos neurônios por camada ?

Que tipo de neurônios ?

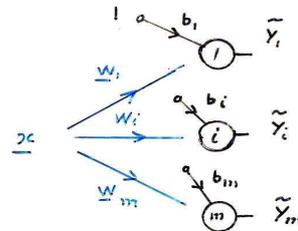
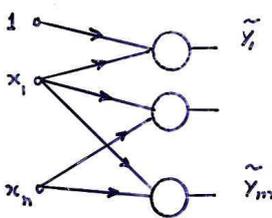
- Subdimensionada >>>** não tem capacidade de representar o mapeamento $x \gg y$

 não “aprende”, o erro permanece elevado
- Superdimensionada >>>** lenta no treino e na operação

 não generaliza bem, tendência a overtraining

4.1 Capacidade de Mapeamento das redes - Saídas contínuas

Redes de uma camada (de neurônios)



Cada saída pode ser processada (treinada e operada) independentemente das demais

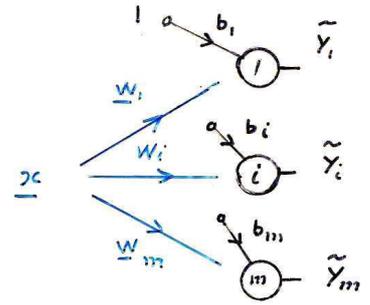
Se os neurônios são lineares, a rede é um mero **combinador linear**:

$$y_i = \sum_j x_j w_{ij} + b_i$$

e portanto tem **capacidade de mapeamento muito limitada**.

Se os neurônios são tipo $tgh(.)$, a saída é um combinador linear com saída limitada:

$$u_i = \sum_j x_j w_{ij} + b_i \quad y_i = tgh u_i$$

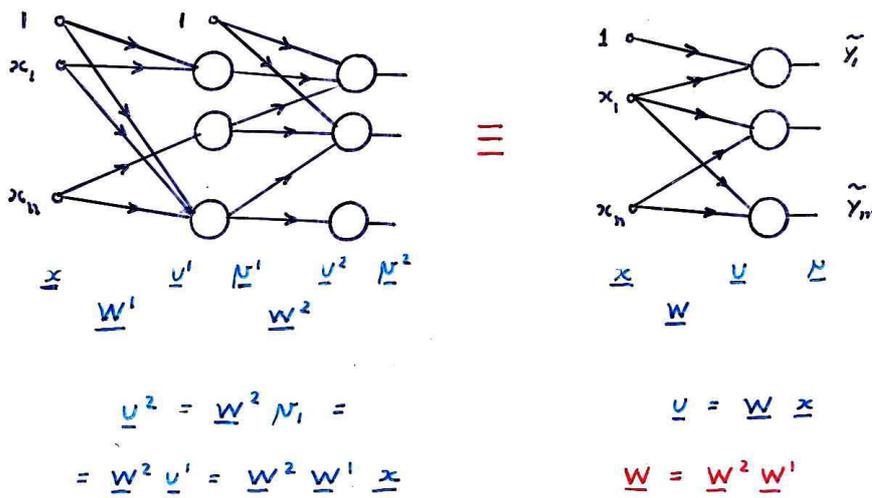


embora limitada, esta estrutura encontrará aplicação em classificadores (separadores) lineares,

conforme será visto posteriormente.

Redes com duas camadas

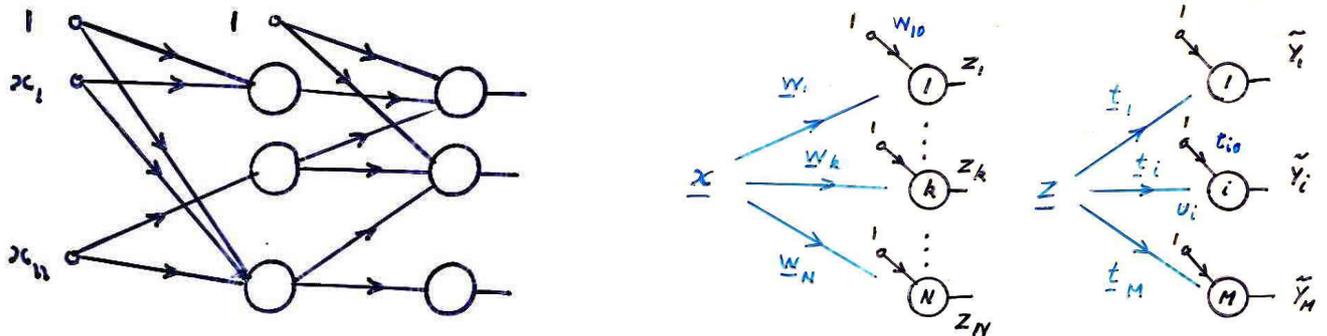
Com neurônios lineares na camada intermediária



Idêntica a uma rede com apenas uma camada !

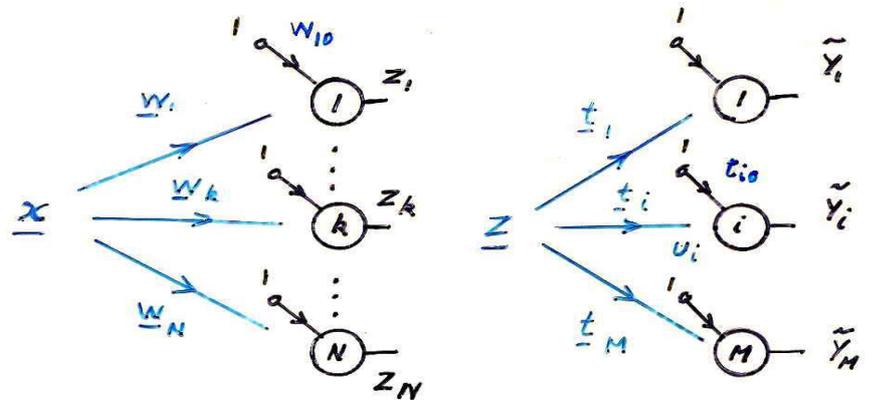
Redes com duas camadas

Com neurônios tipo tgh(.) na camada intermediária



$$\underline{X} \rightarrow \underline{Z} \rightarrow \underline{Y}$$

$$\underline{X} \rightarrow \underline{Z} \rightarrow \underline{Y}$$



$$z_k = tgh \left(\vec{x}^t \vec{w}_k + w_{k0} \right)$$

$$u_i = \vec{z}^t \vec{t} + t_{i0}$$

$$u_i = t_{i0} + \sum_k t_{ik} tgh \left(\vec{x}^t \vec{w}_k + w_{k0} \right)$$

Neurônio de saída linear:

$$y_i = t_{i0} + \sum_k t_{ik} \operatorname{tgh}(\vec{x}^t \vec{w}_k + w_{k0})$$

série de $\operatorname{tgh}(\cdot)$

Neurônio de saída tipo $\operatorname{tgh}(\cdot)$

$$y_i = \operatorname{tgh} \left[t_{i0} + \sum_k t_{ik} \operatorname{tgh}(\vec{x}^t \vec{w}_k + w_{k0}) \right]$$

série de $\operatorname{tgh}(\cdot)$ com saída limitada. Será usada em classificadores.

Teorema (sem prova):

Se uma função $f(\cdot)$ é L^2 em um domínio então uma série de $\operatorname{tgh}(\cdot)$ é um aproximador universal para a função no domínio.

Obs: Uma função $f(\cdot)$ é dita L^2 em um domínio se neste domínio $\int |f(\cdot)|^2 d.$ existe. Funções que admitem representação por série de Fourier são L^2 .

Conclusão: uma rede neural com duas camadas, a primeira com neurônios do tipo $\operatorname{tgh}(\cdot)$ e a segunda com neurônios lineares, é um aproximador universal para todas as funções de interesse prático em engenharia ¹.

Ponto fraco: O número de neurônios na camada intermediária não pode ser pré-determinado ² !

1 - Teorema de Kolmogorov (e.g. Beale, R., Jackson, T. & Jackson, T. (1990), Neural Computing: an introduction, Hilger.)

2 - Han, J., Kamber, M. & Pei, J. (2011), Data mining: concepts and techniques, Morgan Kaufmann Pub.

4.2 - Capacidade de Mapeamento das redes - Saídas lógicas { -1, 1 }

Classificadores

Será detalhada no futuro, no item classificadores.

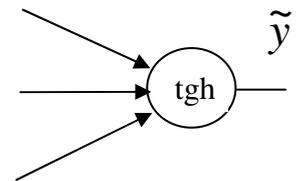
Antecipações necessárias:

Valores alvo:

$$y_i \in \{-1, +1\}$$

Neurônio de saída tipo tgh

$$\tilde{y} = tgh(u) \in (-1, +1)$$



Interpretação lógica da saída

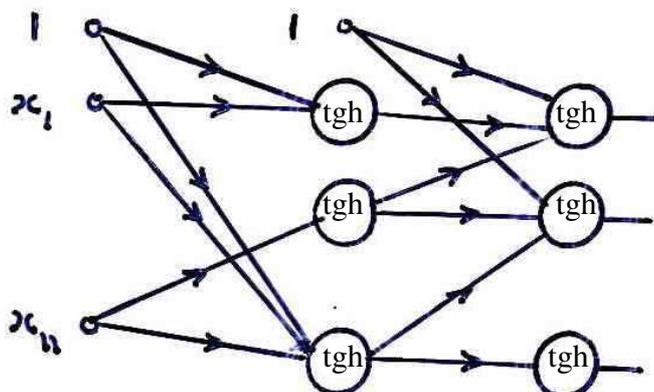
$$\tilde{y}_{\text{lógico}} = \text{sign}(\tilde{y})$$

Treinamento BP: idêntico ao caso com saída contínua,

$$\varepsilon_i = y_i - \tilde{y}_i$$

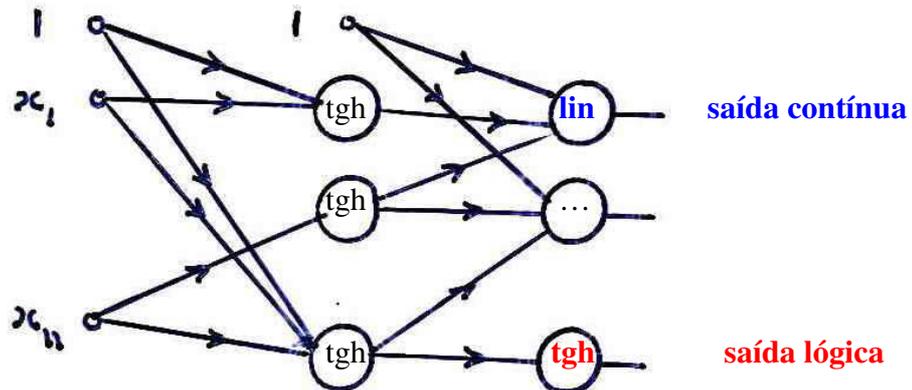
Teorema: Uma rede com duas camadas de neurônios tipo tgh(.) realiza qualquer função lógica com alvos {-1,+1} e o treinamento backpropagation minimiza o erro de classificação.

Ponto fraco: O número de neurônios na camada intermediária é difícil de ser pré-determinado !



4.2.1 - Capacidade de Mapeamento das redes - Saídas contínuas e lógicas na mesma rede

Teorema: Uma rede neural com duas camadas, (a) uma camada intermediária com neurônios tipo $\text{tgh}(\cdot)$ e (b) uma camada de saída com neurônios lineares para as saídas contínuas e neurônios tipo tgh para saídas lógicas treinada por backpropagation realiza qualquer mapeamento L^2 .



4.3 - Como dimensionar a rede ?

Usar uma ou duas camadas de neurônios

Na camada de saída:

se y contínuo, $y \in (-1, +1)$ >>> neurônio linear, tipo $y = u$

se y binário, $y \in \{-1, +1\}$ >>> neurônio não linear tipo $y = \text{tgh}(u)$

Na camada intermediária:

neurônios não lineares tipo $y = \text{tgh}(u)$

Mas como dimensionar ??

Como dimensionar a rede ?

$$\vec{x} \in R^n \Rightarrow \vec{y} \in R^m$$



complexidade do mapeamento ?

Tentativa e Erro !

Rede com uma camada:

$F_0 < F_{0\max}$ satisfeita ? **Sim: Fim**

Não: Use uma rede com duas camadas.

Rede com duas camadas

neurônios na camada intermediária, m_1

$n > m_1 > m$ (heurística)

$F_0 < F_{0\max}$ satisfeita ? **Sim: Fim (ou reduzir m_1)**

Não: aumentar m_1 .

E redes com 3 ou mais camadas ?

Em princípio (ao menos teóricamente) não são necessárias.

O dimensionamento é mais difícil.

O tempo de treinamento cresce exponencialmente
com o número de camadas,

Mas existem casos (excessões) em que levam a
estruturas mais simples (e mais rápidas).

4.4 - Operação da Rede

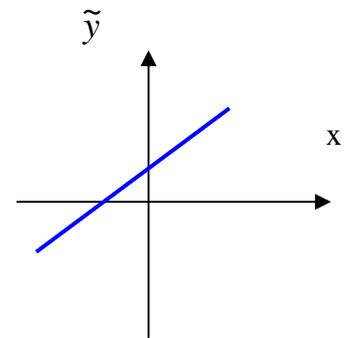
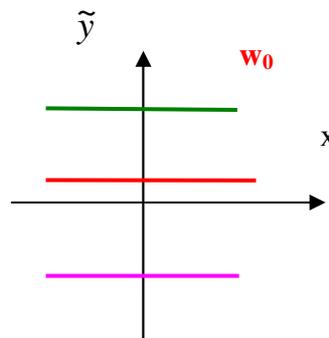
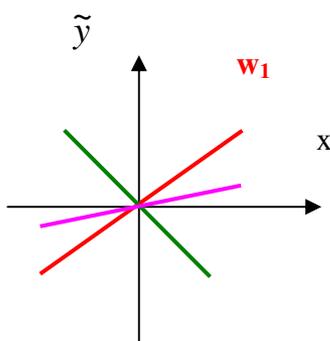
Rede de 1 camada – combinador linear

$$\underline{\tilde{y}} = \varphi(\underline{x})$$

$$\tilde{y} = \varphi(x)$$

$$\tilde{y}_j = \sum_i w_{ji} x_i + w_{j0}$$

$$\tilde{y} = w_1 x + w_0 \quad \text{reta}$$



Duas entradas $y = w_2x_2 + w_1x_1 + w_0$ plano

Rede com duas camadas

$\tilde{y} = \varphi(\underline{x})$

$\tilde{y} = \varphi(x)$

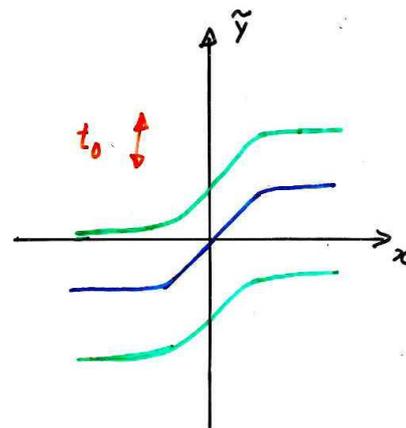
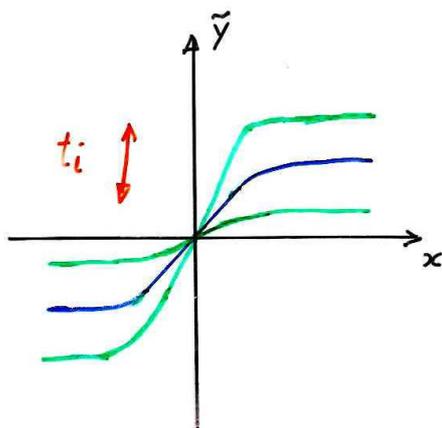
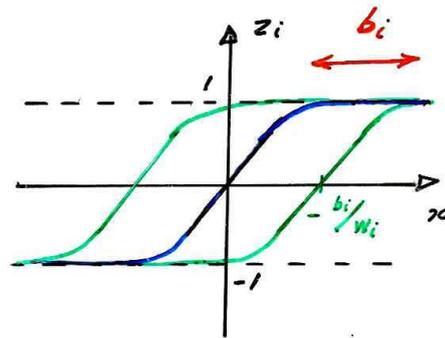
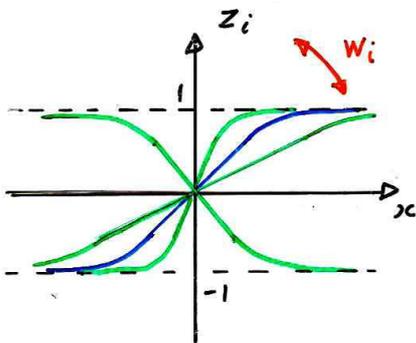
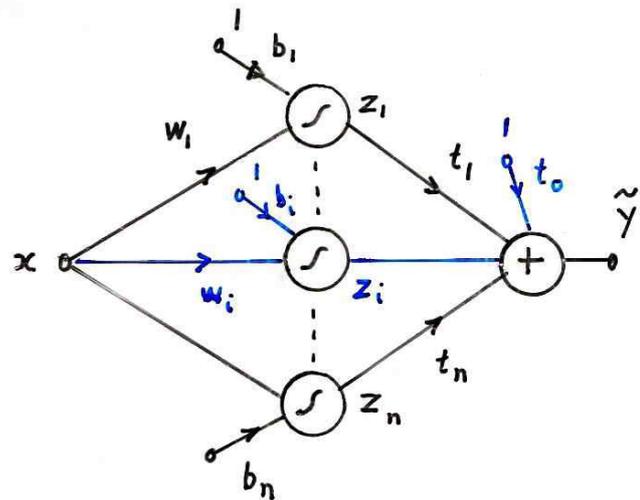
$y = \varphi(x)$

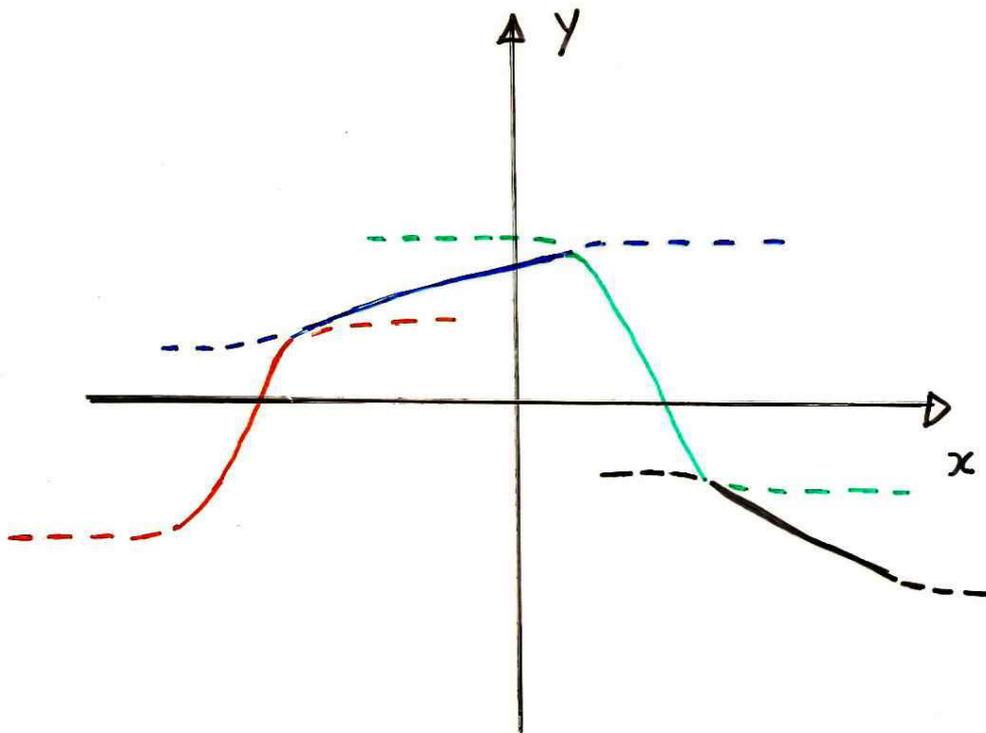
$z_i = t_0 h(w_i x + b_i)$

$\tilde{y} = t_0 + \sum t_i z_i$

$\tilde{y} = t_0 + \sum t_i t_0 h(w_i x + b_i)$

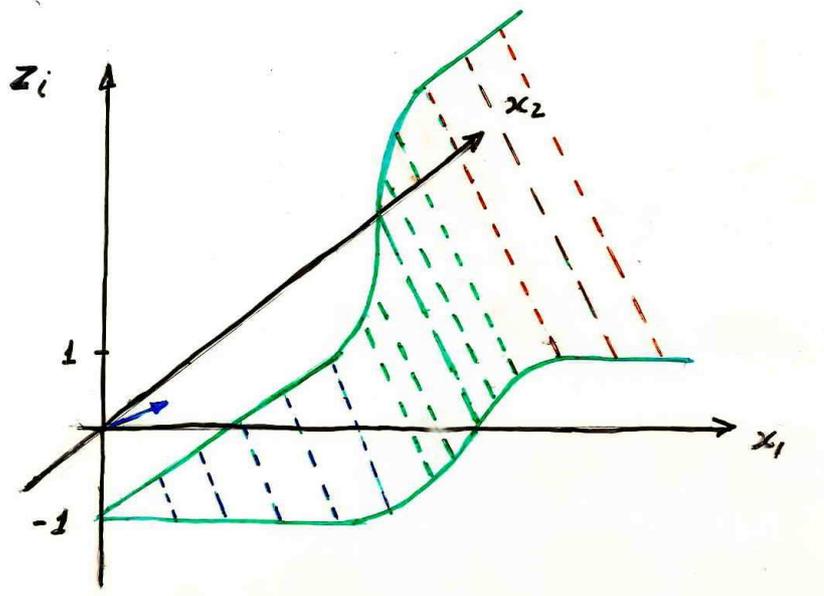
$\tilde{y} = t_0 + \sum t_i t_0 h(w_i x + b_i)$





aproximação por segmentos de retas

Duas entradas $z_i = f(x_1, x_2)$

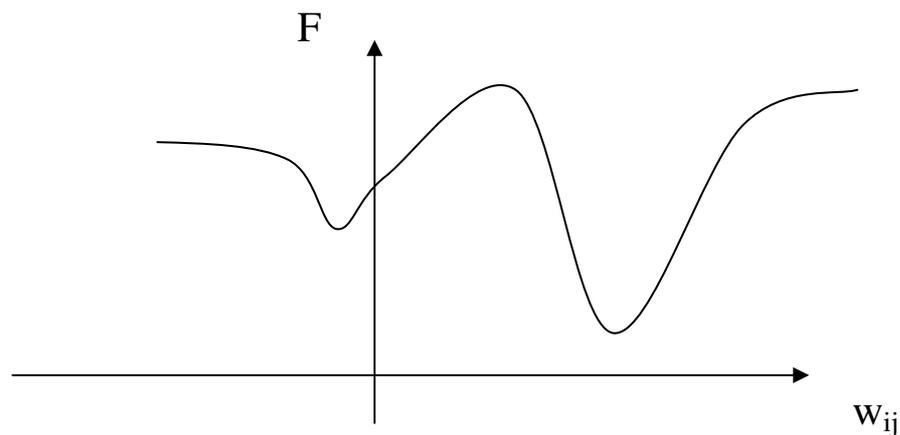


aproximação por segmentos de planos (hiperplanos)

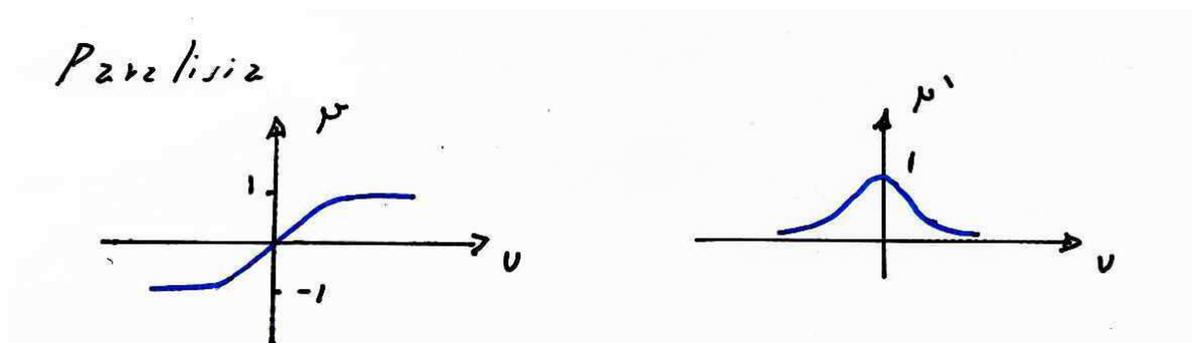
4.5 - Valores iniciais das sinapses

Duas características do processo de treinamento:

1 - Usualmente existem mínimos locais, e o processo pode ficar preso neles. Solução simples: tentar diversos pontos de partida, escolhendo valores iniciais das sinapses dentro de uma faixa grande de valores.



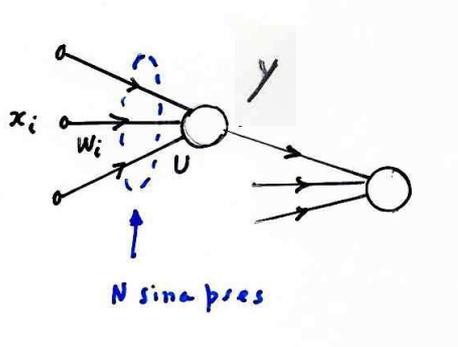
2 - O processo pode ficar paralizado. Como o erro retropropagado é multiplicado pela derivada da função de ativação no ponto de operação, valores muito elevados da excitação interna u podem levar a derivadas com valores muito pequenos e praticamente paralizar o processo. Valores grandes de u ocorrem se tivermos valores grandes das sinapses.



$$v = \operatorname{tgh} u \quad v \in (-1, +1)$$

$$\frac{dv}{du} = 1 - v^2 \quad \frac{dv}{du} \in (0, +1]$$

Os dois critérios são antagônicos: devemos escolher valores iniciais grandes para as sinapses para escapar de mínimos locais, e pequenos para não paralisar o treinamento desde o início. Que valores escolher ?



$$y = \tanh u \quad \frac{dy}{du} = 1 - y^2 \quad \frac{dy}{du} \in (0,1)$$

$$\text{para } \left| \frac{dy}{du} \right| > .01 \quad |u| < 3$$

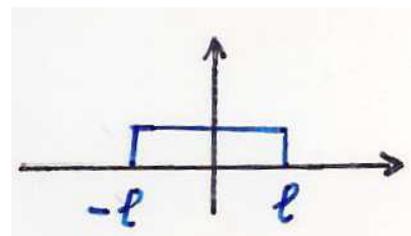
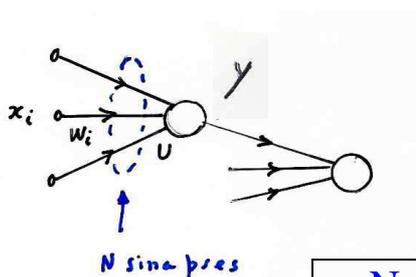
$$u = \sum_{i=1}^N w_i x_i \quad \begin{cases} \mu_u = N \mu_w \mu_x \\ \sigma_u^2 = N \sigma_w^2 \sigma_x^2 \end{cases}$$

Para 99,97 % probabilidade da rede não iniciar paralisada

$$3\sigma_u = 3\sqrt{N}\sigma_w\sigma_x < |u|_{\max} = 3$$

como $\sigma_x = 1$, $\sigma_w \leq \frac{1}{\sqrt{N}}$.

Para uma distribuição uniforme de w , $l \leq \sqrt{\frac{3}{N}}$



N	σ_w	l
4	.5	.9
10	.3	.5
100	.1	.2

Valores típicos: sorteio randômico entre $-0,2$ e $+0,2$. Se houver desconfiança de mínimos locais no entorno da origem aumentar a faixa.

4.6 Escolha de α

“a escolha de α é uma arte” - Bernard Widrow

α deve ser tal que a aproximação da função (1ª ou 2ª ordem) em que o método se baseia é válida

α muito pequeno convergência estável, processo muito lento

α muito grande processo oscila ou mesmo diverge

α típico BP .1, .05

α variável **Otimização em linha**

BP resiliente