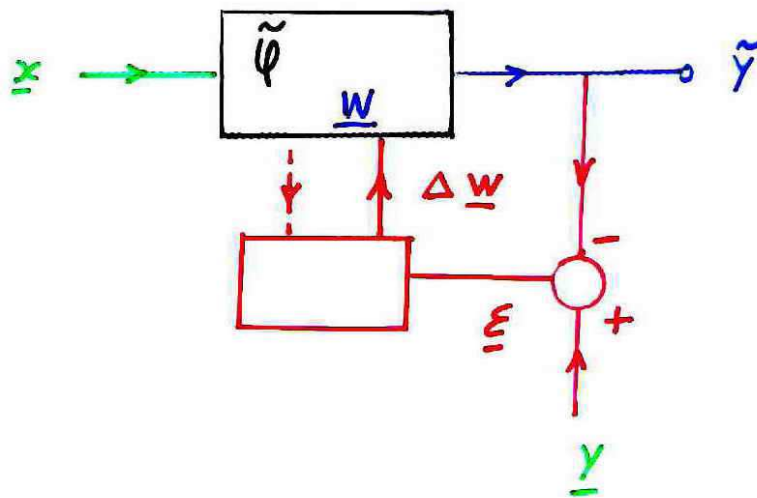
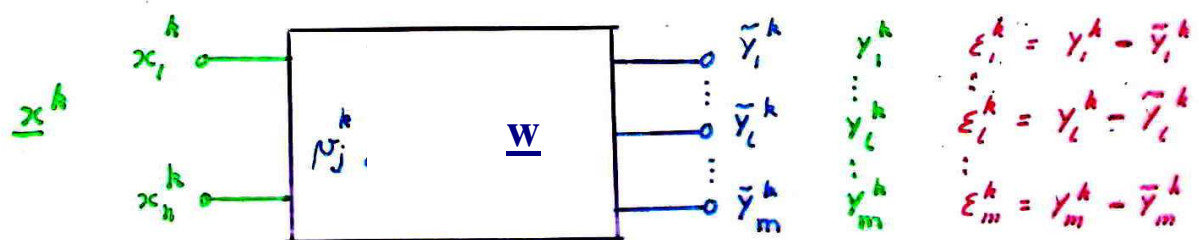


## Treinamento Backpropagation de uma Rede Neural Feedforward



Erro na saída  $\longrightarrow$  Função objetivo à minimizar

A saída pode ser multidimensional:



**Erro quadrático** na saída para o k-ésimo par entrada-saída k

$$(\mathcal{E}^k)^2 = \left| \underline{y}^k - \underline{\tilde{y}}^k \right|^2 = \sum_{l=1}^L (y_l^k - \tilde{y}_l^k)^2 = \sum_{l=1}^L (\mathcal{E}_l^k)^2$$

$$\text{onde } \mathcal{E}_1^k = y_1^k - \tilde{y}_1^k \quad e \quad \tilde{y}_1^k = \tilde{y}_1^k(\underline{w})$$

Erro quadrático médio para todos os P pares entrada-saída

$$(\underline{x}^k, \underline{y}^k) \quad k = 1, \dots, P$$

Erro médio quadrático:

$$F(\underline{w}) = E_k (\varepsilon^k)^2 = \frac{1}{P} \sum_{k=1}^P (\varepsilon^k)^2$$

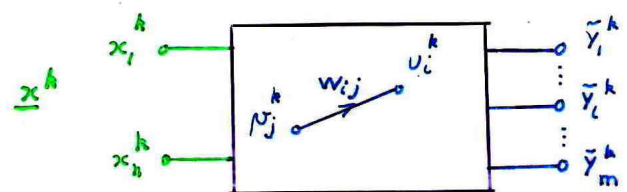
**Função objetivo à minimizar**

**Acréscimo a aplicar em cada sinapse:**

$$F_0 = E(\varepsilon^{2p}) = F_0(\underline{w}) \quad \Delta w_{ij} = -\alpha \frac{\partial F_0(w_{ij})}{\partial w_{ij}}$$

**Como calcular**

$$\frac{\partial F_0(w_{ij})}{\partial w_{ij}} \quad ???$$



$$1 - F_0(w_{ij}) = E_k \left[ (\epsilon^k)^2 \right] = \frac{1}{P} \sum_{k=1}^P (\epsilon^k)^2$$

$$\frac{\partial}{\partial w_i} E_k (\epsilon^k)^2 = \frac{\partial}{\partial w_i} \frac{1}{P} \sum_{k=1}^P (\epsilon^k)^2 = \frac{1}{P} \sum_{k=1}^P \frac{\partial}{\partial w_i} (\epsilon^k)^2 = E_k \frac{\partial}{\partial w_i} (\epsilon^k)^2$$

$$\nabla_k E_k (\epsilon^k)^2 = E_k \nabla_k (\epsilon^k)^2$$

### Propriedade importante:

o gradiente do valor esperado do erro quadrático é igual  
ao valor esperado do gradiente do erro quadrático

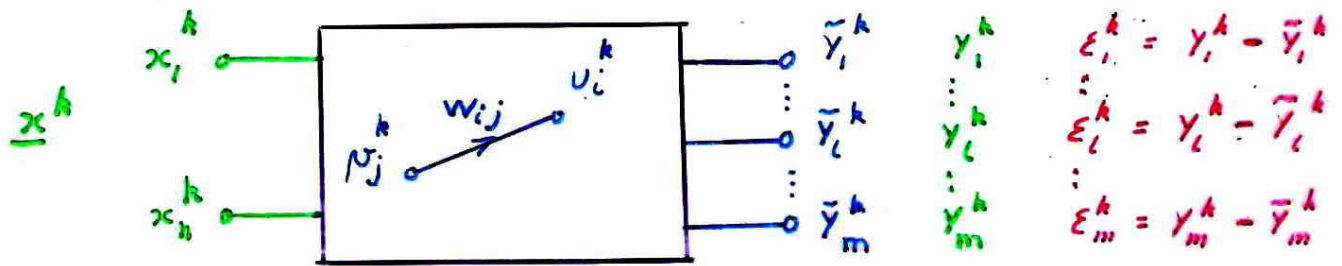
cada par entrada-saída é tratado isoladamente.

Consequência:

Calcularemos separadamente para cada par  $\frac{\partial}{\partial w_i} (\epsilon^k)^2 \quad \forall k$

e tomaremos a média  $\frac{\partial}{\partial w_i} E_k (\epsilon^k)^2 = E_k \frac{\partial}{\partial w_i} (\epsilon^k)^2$

## 2 – Cálculo de $\frac{\partial(\epsilon^k)^2}{\partial w_{ij}}$ para cada par entrada – saída k



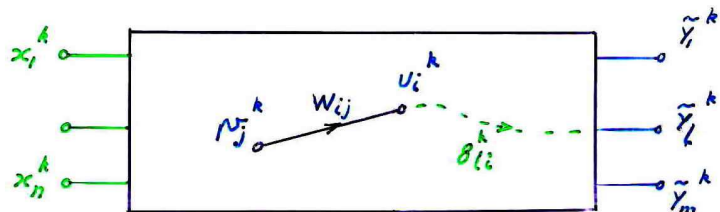
$$(\epsilon)^2 = \sum_{l=1}^m (\epsilon_l)^2 = \sum_{l=1}^m (y_l - \tilde{y}_l)^2$$

$$\frac{\partial(\epsilon)^2}{\partial w_{ij}} \quad ???$$

$$\frac{\partial(\epsilon)^2}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}} \sum_{l=1}^m \epsilon_l^2 = \sum_{l=1}^m \frac{\partial \epsilon_l^2}{\partial w_{ij}} = 2 \sum_{l=1}^m \epsilon_l \frac{\partial \epsilon_l}{\partial w_{ij}} =$$

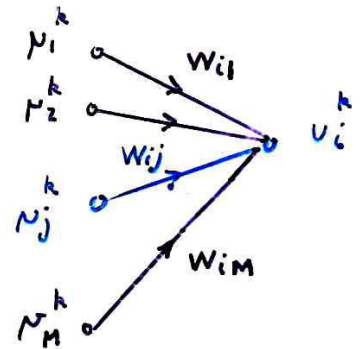
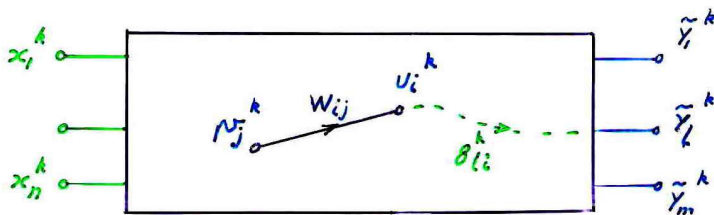
$$= 2 \sum_{l=1}^m \epsilon_l \frac{\partial (y_l - \tilde{y}_l)}{\partial w_{ij}} = -2 \sum_{l=1}^m \epsilon_l \frac{\partial \tilde{y}_l}{\partial w_{ij}} = -2 \sum_{l=1}^m \epsilon_l \frac{\partial \tilde{y}_l}{\partial u_i} \frac{\partial u_i}{\partial w_{ij}} =$$

$$\frac{\partial \tilde{y}_l}{\partial u_i} = g_{li}$$



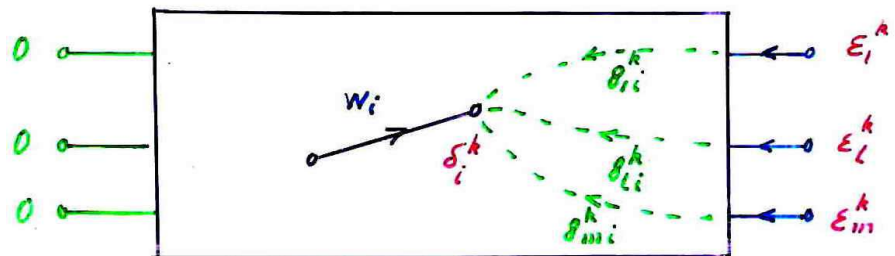
$$\frac{\partial(\epsilon)^2}{\partial w_{ij}} = -2 \sum_{l=1}^m \epsilon_l \frac{\partial \tilde{y}_l}{\partial u_i} \frac{\partial u_i}{\partial w_{ij}} = -2 \sum_{l=1}^m \epsilon_l g_{li} v_j$$

$$\frac{\partial \tilde{y}_l}{\partial u_i} = g_{li} \quad u_i = \sum_{m=1}^M w_{mj} v_j \quad \frac{\partial u_i}{\partial w_{ij}} = v_j$$



$$\frac{\partial(\epsilon)^2}{\partial w_{ij}} = -2 \sum_{l=1}^m \epsilon_l g_{li} v_j = -2 v_j \sum_{l=1}^m \epsilon_l g_{li} = -2 v_j \delta_i$$

$$\delta_i = \sum_{l=1}^m \epsilon_l \frac{\partial \tilde{y}_l}{\partial u_i} = \sum_{l=1}^m \epsilon_l g_{li}$$



$\delta_i$  é o erro retropropagado da saída até a extremidade da sinapse

3 - como:

$$\Delta w_{ij} = -\alpha \frac{\partial F_0(w_{ij})}{\partial w_{ij}}$$

$$\frac{\partial F_0(w_{ij})}{\partial w_{ij}} = \frac{1}{P} \sum_{p=1}^P \frac{\partial}{\partial w_{ij}} (\epsilon^{2^p})$$

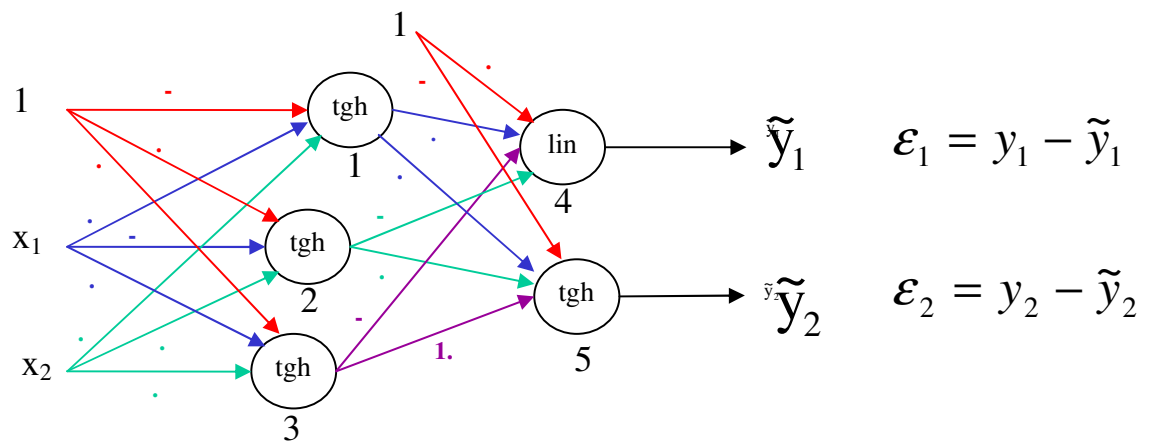
$$\frac{\partial (\epsilon)^2}{\partial w_{ij}} = -2 v_j \delta_i \quad \text{para um par entrada -saída}$$

**Para os P pares entrada saída**

$$\Delta w_{ij} = 2\alpha \frac{1}{P} \sum_{p=1}^P v_j \delta_i \Big|_p$$

## Error Backpropagation -Princípio do Algoritmo:

### 1 - Rede Original:



Propagar o sinal na rede original e conservar o valor do sinal  $v_j$  na entrada de cada sinapse  $w_{ij}$ . Calcular o erro em cada saída.

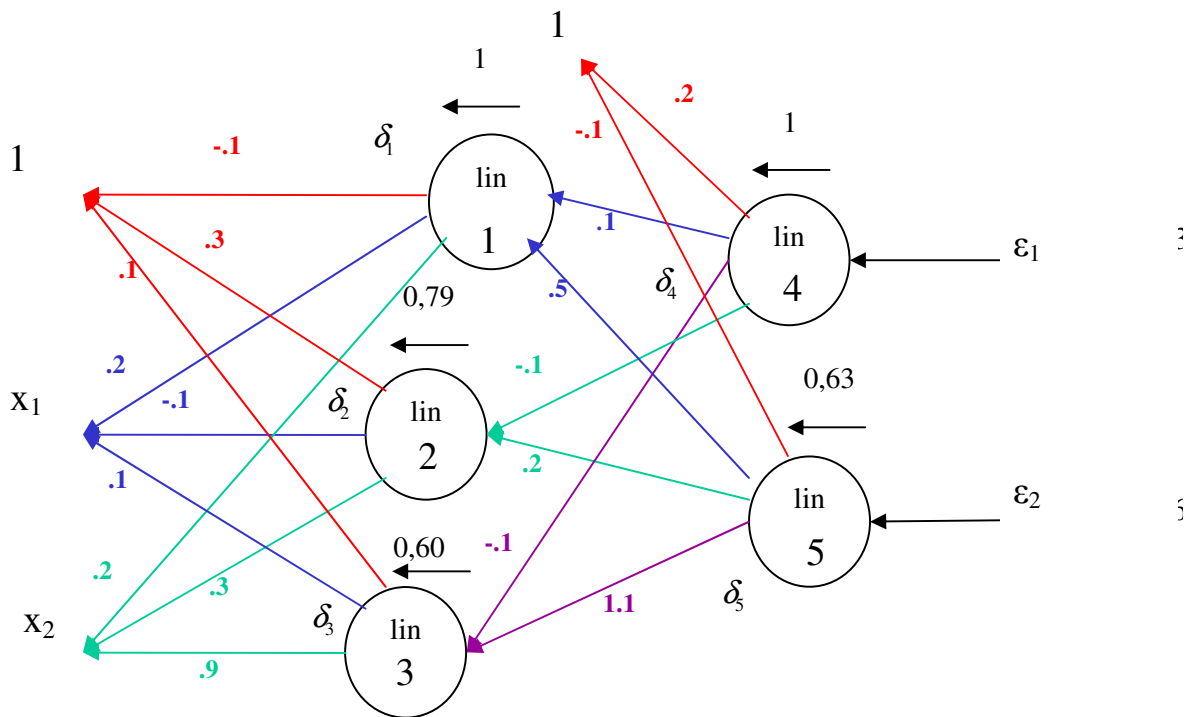
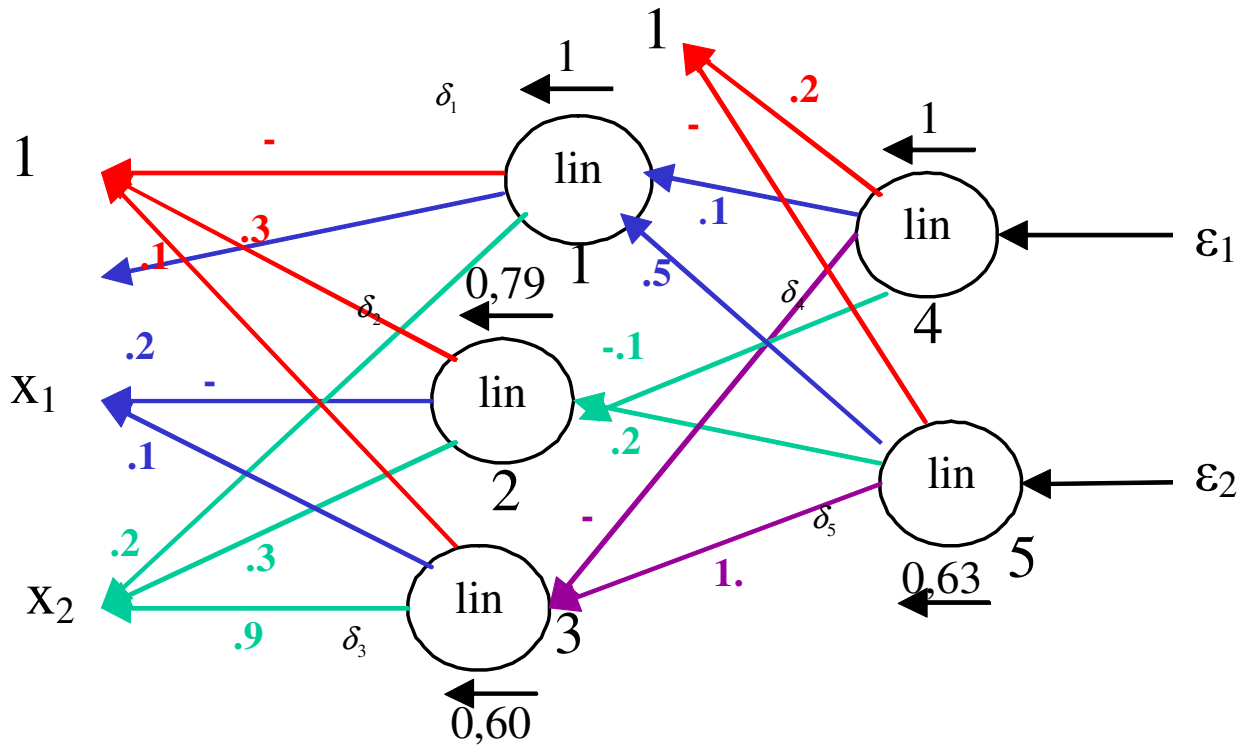
### 2 – Criar uma “Rede Associada” a partir da Rede Original

mesma arquitetura, mas

#### 2.1 - “Linearizar” (os neurônios) da rede original

Rede Original	→	Rede Associada
Neurônio não linear $v_0 = \text{tgh}(u_0)$	→	Neurônio linear $v = (1-v_0^2) u$
Neurônio linear $v_0 = u_0$	→	Neurônio linear $v = u$

**2.2- Inverter todos os sentidos de transmissão (dos neurônios e sinapses)**





**4 – O acréscimo para cada sinapse  $w_{ij}$  para o par  $p$  entrada-saída em operação é dado por:**

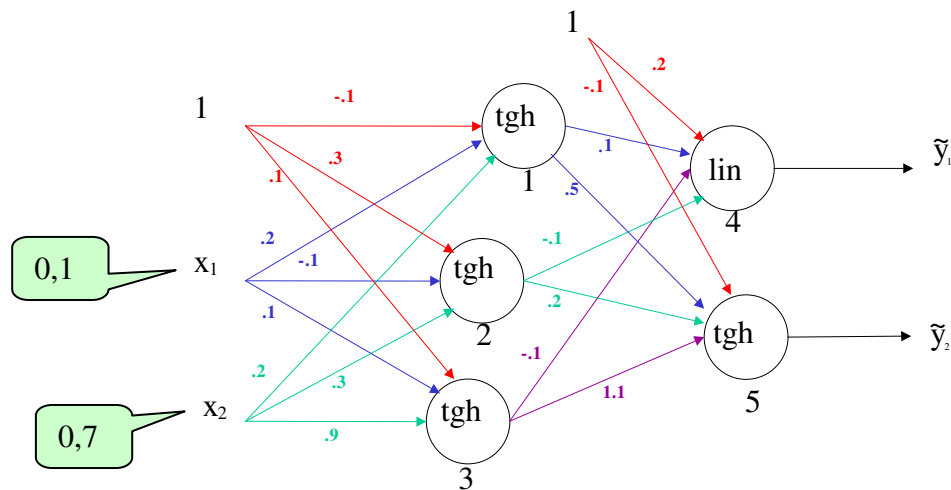
$$\Delta_p w_{ij} = 2 \alpha v_j \delta_i$$

**5 – O acréscimo a ser aplicado na sinapse  $w_{ij}$  é o valor esperado dos acréscimos calculados para todos os pares entrada-saída.**

$$\Delta w_{ij} = E_p(\Delta_p w_{ij}) = 2 \alpha E_p(v_j \delta_i | p)$$

### Processo em Batelada – Batch

**Exemplo anterior (regra delta):**

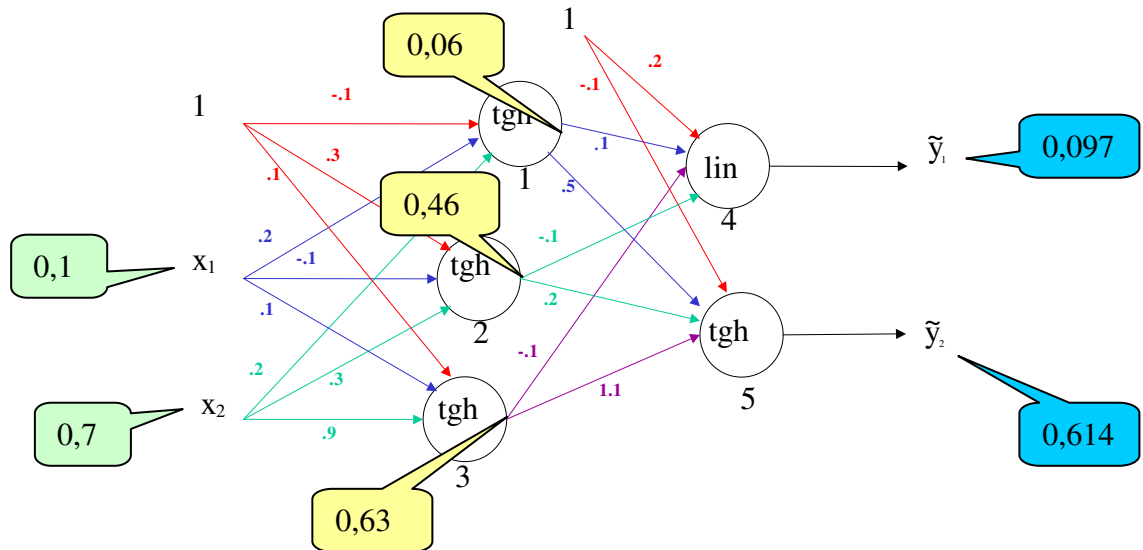


$$\underline{\mathbf{x}} = \begin{bmatrix} 0,1 \\ 0,7 \end{bmatrix}$$

$$\tilde{\mathbf{y}} = ?$$

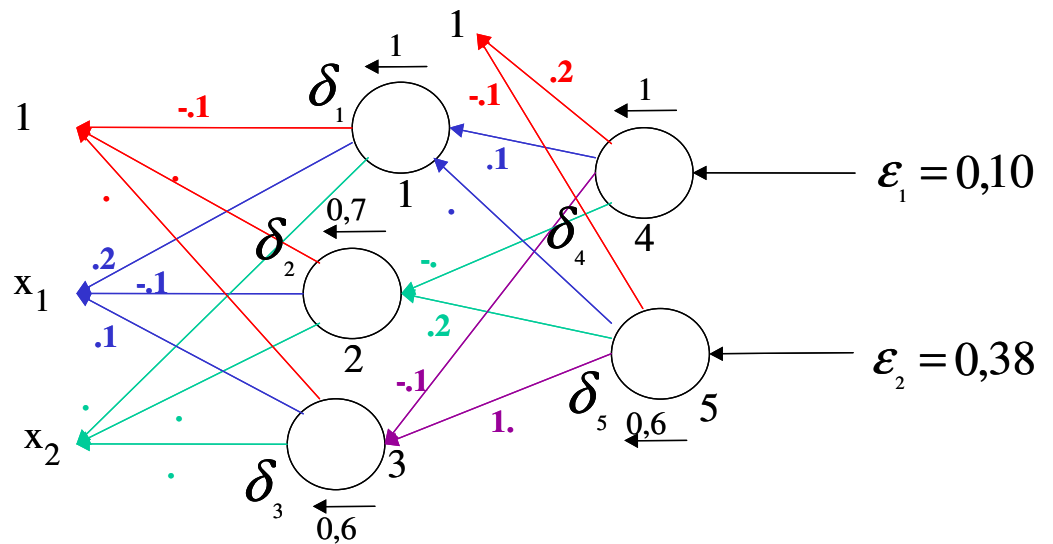
$$\mathbf{y} = \begin{bmatrix} 0,2 \\ 0,1 \end{bmatrix}$$

### Signal Feedforward



$$\mathbf{x} = \begin{bmatrix} 0,1 \\ 0,7 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} 0,2 \\ 1 \end{bmatrix} \quad \tilde{\mathbf{y}} = \begin{bmatrix} 0,097 \\ 0,614 \end{bmatrix} \quad \boldsymbol{\varepsilon} = \begin{bmatrix} 0,103 \\ 0,386 \end{bmatrix}$$

### Rede associada e retropropagação do erro:

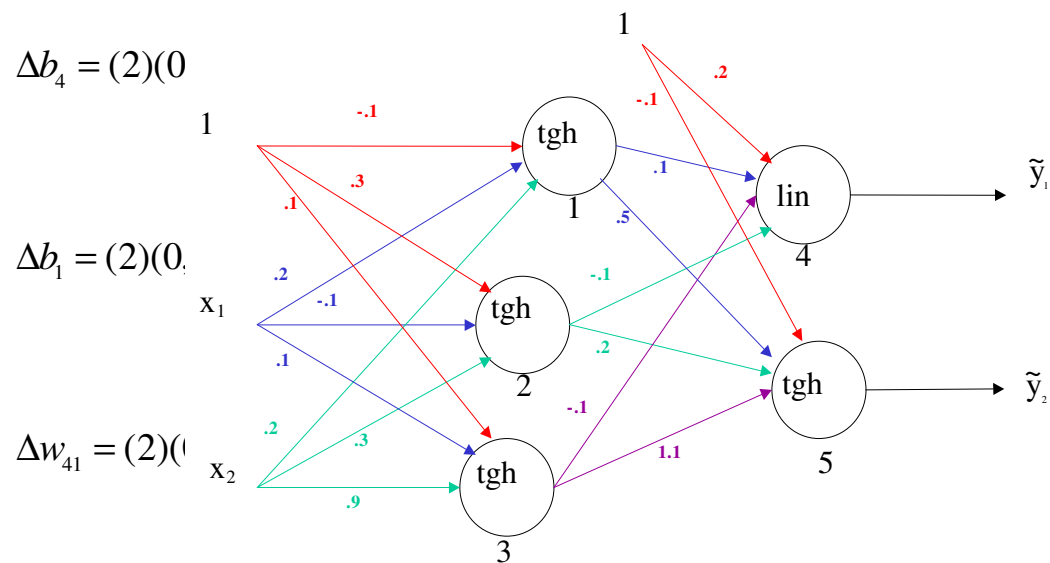


$$\begin{aligned} \delta_5 &= (0,386)(0,63) = 0,24 & \delta_4 &= (0,103)(1) = 0,103 \\ \delta_3 &= [(0,103)(-0,1) + (0,24)(1,1)](0,60) = 0,152 \\ \delta_2 &= [(0,103)(-0,1) + (0,24)(0,2)](0,79) = 0,030 \\ \delta_1 &= [(0,103)(0,1) + (0,24)(0,5)](1) = 0,130 \end{aligned}$$

## Treinamento Regra Delta

Atualização dos valores das sinapses:

$$\Delta w_{ij} = 2\alpha v_j \delta_i$$



$$\Delta w_{3b} = (2)(0,1)(0,7)(0,152) = 0,021$$

$$w_{3b} \rightarrow 0,9 + 0,021 = 0,921$$

## Algoritmos BP

para cálculo do acréscimo na sinapse  $w_{ij}$  devido ao par  $p$ ,  $\Delta_p w_{ij}$

para dois casos específicos:

## Redes com uma única camada

Definição das variáveis:

N entradas:  $x_1, x_2, \dots, x_N$  e  $x_0$  (bias) = 1

$w_{ij}$  - sinapse conectando a entrada j  
ao neurônio i

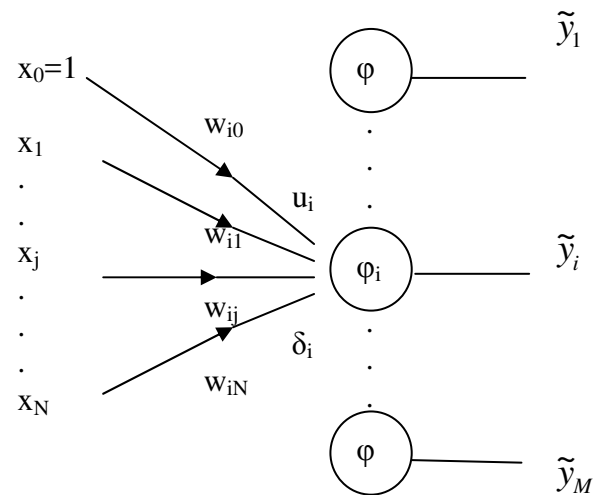
M neurônios com função de ativação

$\varphi(\cdot)$  linear ou tgh

$u_i$  - excitação interna do neurônio i

$\tilde{y}_i$  - saída do neurônio i

### Rede Original



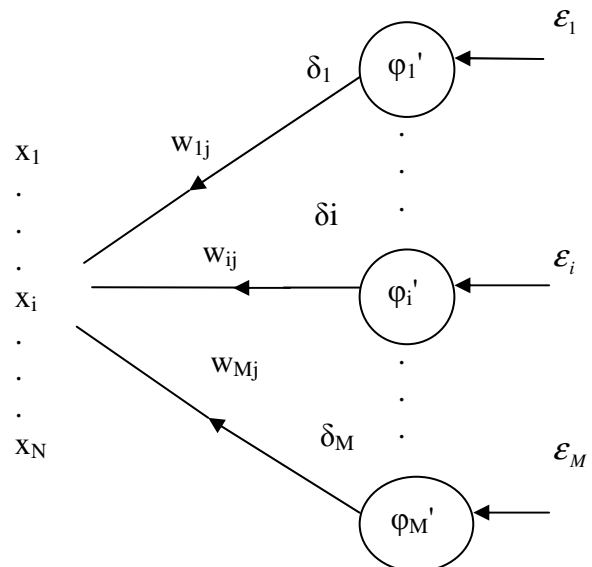
Definição das variáveis:

$\epsilon_i$  - erro na saída do neurônio i

$\delta_i$  - erro retropropagado até  
a entrada do neurônio i

$\varphi_i'$  - valor numérico da derivada de  
 $\varphi_i$  no ponto de operação

### Rede Associada



**Algoritmo BP acréscimo  $\Delta w_{ij}^p$  - Rede 1 camada**

Para o par entrada-saída p ( $\underline{x}$ ,  $y$ )

Signal feedforward: Propague o sinal para a frente, calcule a saída e o erro em cada saída:

$$u_i = \sum_{j=0}^N w_{ij} x_j$$

$$\tilde{y}_i = \varphi_i(u_i) = \begin{cases} u_i & \text{neurônio linear} \\ \text{tgh}(u_i) & \text{neurônio tgh} \end{cases}$$

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

Calcule o erro retropropagado até a entrada de cada neurônio:

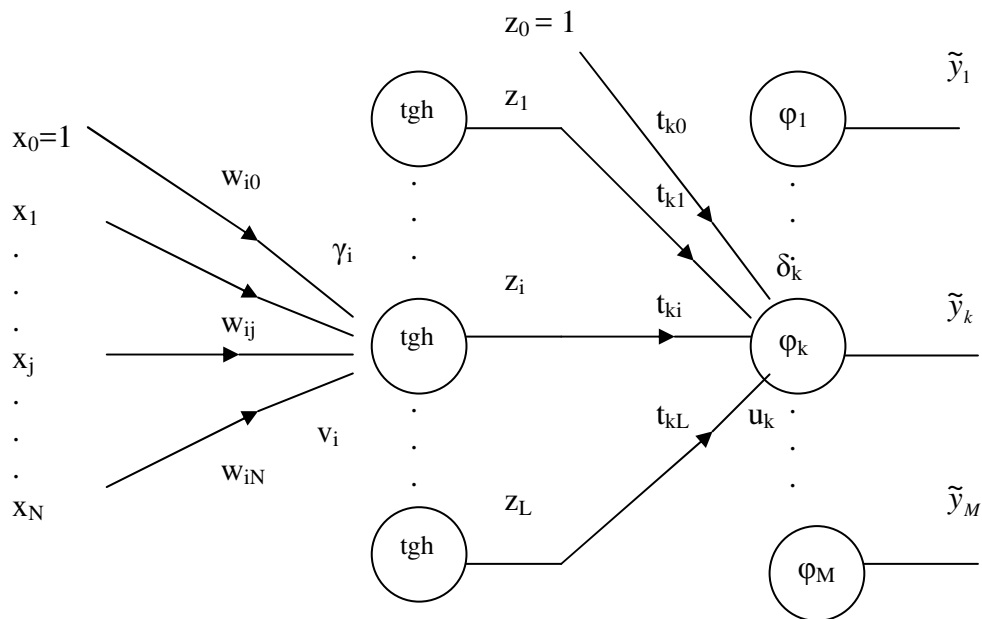
$$\delta_i = \begin{cases} \varepsilon_i & \text{neurônio linear} \\ (1 - \tilde{y}_i^2) \varepsilon_i & \text{neurônio tgh} \end{cases}$$

Calcule o acréscimo em cada sinapse devido ao par p:

$$\Delta w_{ij}^p = 2\alpha x_j \delta_i$$

*fim do algoritmo*

## Redes com duas camadas - Rede Original



Definição das variáveis:

$N$  entradas:  $x_1, x_2, \dots, x_N$  e  $x_0$  (bias) = 1

$L$  neurônios na primeira camada com função de ativação tgh

$M$  neurônios na camada de saída com função de ativação  $\phi(\cdot)$  linear ou tgh

$w_{ij}$  - sinapse conectando a entrada  $j$  ao neurônio  $i$  da camada intermediária

$t_{ki}$  - sinapse conectando a saída  $z_i$  do neurônio  $i$  da camada intermediária com a entrada do neurônio  $k$  da camada de saída.

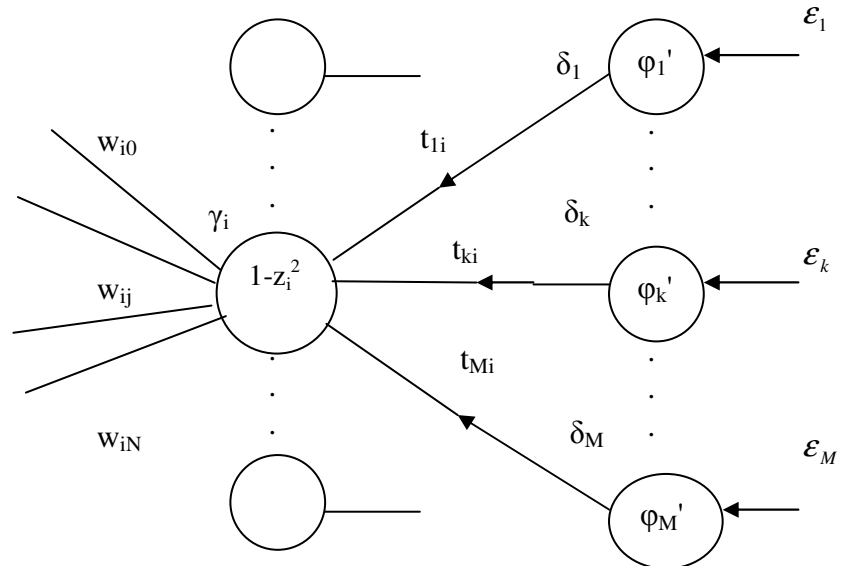
$v_i$  - excitação interna do neurônio  $i$  da primeira camada

$z_i$  - saída do neurônio  $i$  da camada intermediária.  $z_0$  (bias) = 1

$u_k$  - excitação interna do neurônio  $k$  da camada intermediária

$\tilde{y}_k$  - saída do neurônio  $k$  da camada de saída

## Rede Associada



Definição das variáveis:

$\epsilon_k$  - erro na saída do neurônio da camada de saída

$\delta_k$  - erro retropropagado até a entrada do neurônio  $k$  da camada de saída

$\phi_i'$  - valor numérico da derivada de  $\phi_i$  no ponto de operação

## Algoritmo BP acréscimos $\Delta w_{ij}^p$ e $\Delta t_{ki}^p$ - Rede com 2 camadas

Para o par entrada-saída p ( $\underline{x}$ ,  $\underline{y}$ )

Signal feedforward: Propague o sinal para a frente, calcule a saída e o erro em cada saída:

Para todos os neurônios da primeira camada  $i = 1, \dots, L$

$$v_i = \sum_{j=0}^N w_{ij} x_j \quad x_0 = 1$$

$$z_i = \text{tgh}(v_i) \quad z_0 = 1$$

Para todos os neurônios da segunda camada  $k = 1, \dots, M$

$$u_k = \sum_{i=0}^L t_{ki} z_i \quad z_0 = 1$$

$$\tilde{y}_k = \varphi_k(u_k) = \begin{cases} u_k & \text{neurônio linear} \\ \text{tgh}(u_k) & \text{neurônio tgh} \end{cases}$$

$$\varepsilon_k = y_k - \tilde{y}_k$$

Retropropagação do erro

Para todo neurônio da segunda camada,  $k = 1, \dots, M$ :

$$\delta_k = \begin{cases} \varepsilon_k & \text{neurônio linear} \\ (1 - \tilde{y}_k^2) \varepsilon_k & \text{neurônio tgh} \end{cases}$$

Para todo neurônio da primeira camada,  $i = 1, \dots, L$

$$\gamma_i = (1 - z_i^2) \sum_{k=1}^M t_{ki} \delta_k$$

Acréscimo nas sinapses da primeira camada devido ao par p:

$$\Delta w_{ij}^p = 2 \alpha x_j \gamma_i \quad \forall j = 0, 1, \dots, N \quad e \quad i = 1, 2, \dots, L$$

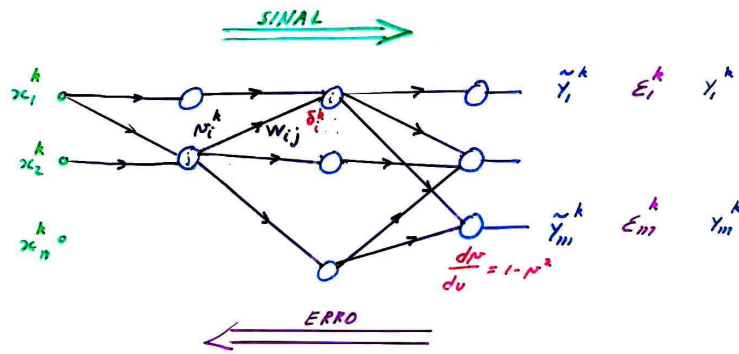
Acréscimo nas sinapses da segunda camada devido ao par p:

$$\Delta t_{ki}^p = 2 \alpha z_i \delta_k \quad \forall i = 0, 1, \dots, L \quad e \quad k = 1, 2, \dots, M$$

*fim do algoritmo*



## Error Backpropagation - Resumo:



$$\frac{\partial \epsilon_i^k}{\partial w_{ij}} = -2 p_j^k \delta_i^k$$

$$\nabla F_0 = \left[ \frac{\partial F_0}{\partial w_{ij}} \right] = -2 E(p_j \delta_i)$$

$$\Delta w_{ij} = 2 \alpha E(p_j \delta_i)$$