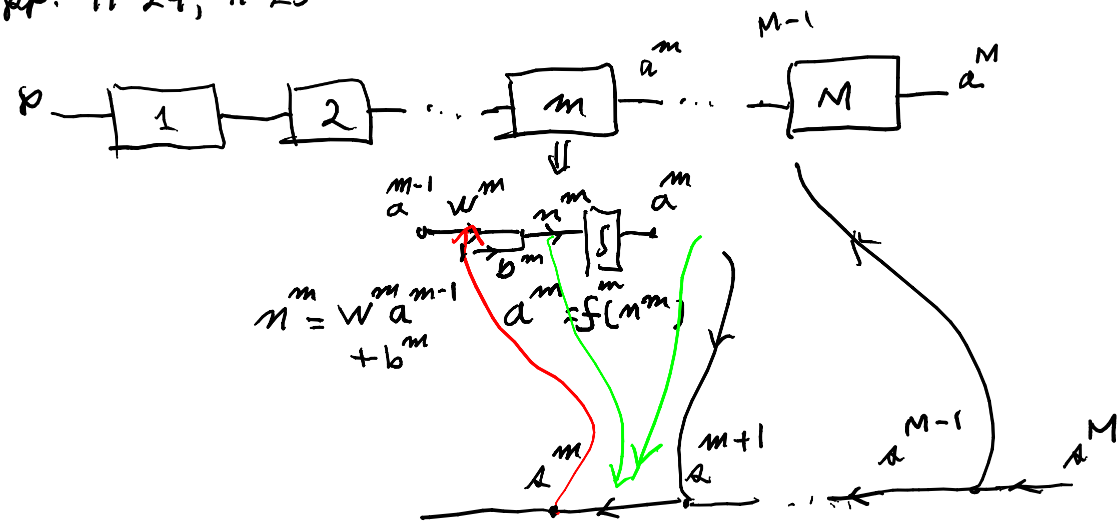
Back propagation

pp. 11-24, 11-25

EE 434
02/06/08



$n^m = W^m a^{m-1} + b^m$ $\quad a^m = f^m(n^m)$

$$W^m(k+1) = W^m(k) - \alpha\, s^m(k)\, (a^{m-1}(k))^T$$

$= W^m(k) + \Delta W^m(k+1)$

$s^m(k)$ = sensitivity vector; $(a^{m-1}(k))^T$ = output vector transpose

k = running iteration #

T = transpose

$0 < \alpha \le 1$

$$s^m = \dot{F}^m(n^m)(W^{m+1})^T s^{m+1} \qquad m = 1, \dots, M-1$$

$$s^M = -2\dot{F}^M(n^M)\cdot(t - a^M)$$

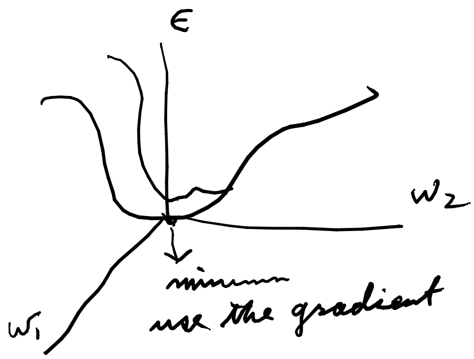
t = desired outputs
a^M = actual outputs for an input p at the first layer

$$\dot{F}^m(n^m) = \begin{bmatrix} \frac{d f_1^m(n_1^m)}{d n_1^m} & 0 & \cdots & 0 \\ 0 & \frac{d f_2^m(n_2^m)}{d n_2^m} & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & \frac{d f_{S^m}^m(n_{S^m}^m)}{d n_{S^m}^m} \end{bmatrix} \qquad f^m(n^m) = \begin{bmatrix} f_1^m(n_1^m) \\ \vdots \\ f_{S^m}^m(n_{S^m}^m) \end{bmatrix}$$

S^m = # of neurons in mth layer

$\epsilon = \| t - a^m \|$ = function of all weights (& bias) (vary) [and transfer functions] (fix)

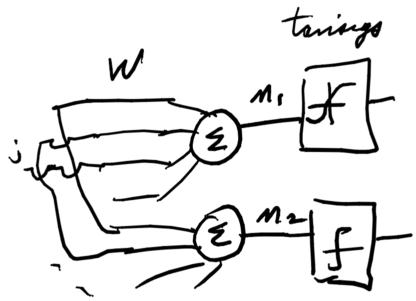
$= +\sqrt{(t-a^m)^T(t-a^m)} = +\sqrt{\text{sum of squares}}$



use the gradient

In the case of tansig:

$$f(n) = \begin{bmatrix} \text{tansig}(n_1) \\ \text{tansig}(n_2) \end{bmatrix}$$

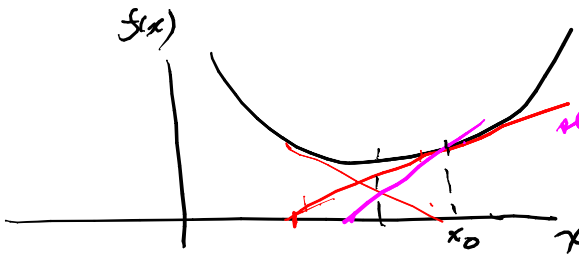


$$\dot{F}(n) = \begin{bmatrix} \frac{d\,\text{tansig}(n_1)}{dn_1} & 0 \\ 0 & \frac{d\,\text{tansig}(n_2)}{dn_2} \end{bmatrix}$$

$$\text{tansig}(n) = \tanh(n) = \frac{\sinh(n)}{\cosh(n)} = \frac{(e^{n}-e^{-n})/2}{(e^{n}+e^{-n})/2}$$

$$\frac{d\,\text{tansig}(n)}{dn} = \frac{e^{n}+e^{-n}}{e^{n}+e^{-n}} - \frac{(e^{n}-e^{-n})}{(e^{n}+e^{-n})^2}\cdot(e^{n}-e^{-n})$$

$$= 1 - \text{tansig}^2(n) = 1 - a^2 \leftarrow a = \text{output of layer}$$



slope depends on a choice, α in back prop

---

making your own neural network from scratch Matlab:

custom Networks = chapter 12 of N.Net Toolbox Manual