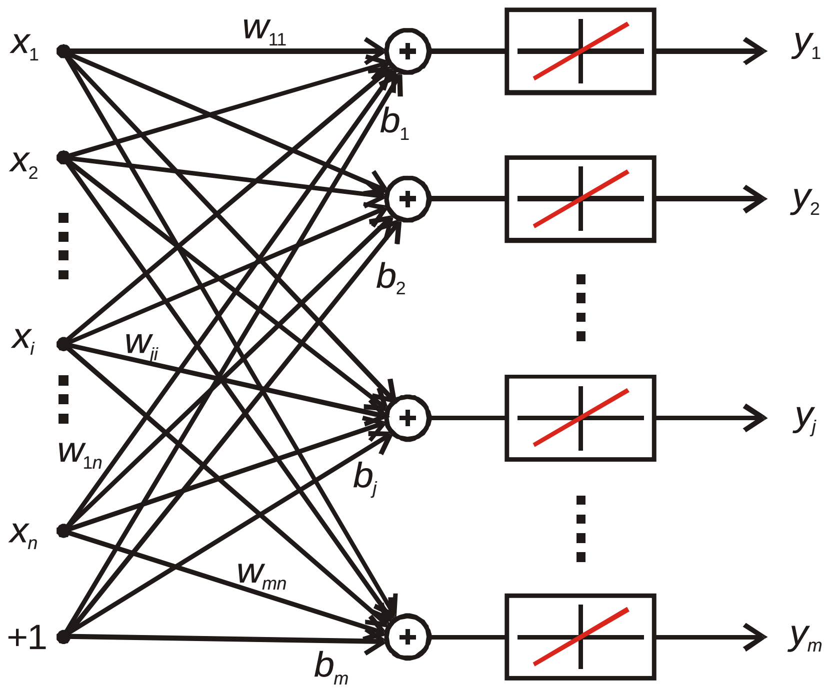
3.2. Linearní síť (Madaline, Madalina)

! je tvořena jednou vrstvou neuronů s lineární aktivační funkcí



! Pro výstup *j*-tého neuronu platí: $y_j = \lambda\left(\sum_{i=1}^{n} w_{ji}\, x_i + b_j\right)$

8 je strmost aktivační funkce (většinou se volí 8=1)

$\boldsymbol{x} = [x_1, x_2, ..., x_i, ..., x_n\,]^T$ ... vstupní vektor

$$\boldsymbol{W} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ w_{m1} & w_{m2} & \cdots & w_{mn} \end{bmatrix}$$ ... váhová matice

$\boldsymbol{b} = [b_1, b_2, ..., b_j, ..., b_m]^T$ ... prahový vektor

$\boldsymbol{y} = [y_1, y_2, ..., y_j, ..., y_m]^T$ ... výstupní vektor

Poznámky:

1. Výstup každého neuronu je určen lineární kombinací vstupů → ADAptivní LIneární NEuron (Adaline). Lineární sítě se pak označují Madaline (Many ADALINEs).

2. Jakoukoli vícevrstvou lineární síť lze nahradit ekvivalentní jednovrstvou lineární sítí.

Trénování sítě pomocí učení s učitelem

! předpokládá se, že máme k dispozici trénovací množinu, tj. množinu P dvojic [vstup x_p, požadovaný výstup u_p]

! chceme nastavit váhy a prahy sítě tak, aby výstup sítě byl pro všechny vstupní vektory z trénovací množiny správný, tzn. aby celková chyba definovaná vztahem

$$E = \sum_{p=1}^{P} \varepsilon_p = \frac{1}{2} \sum_{p=1}^{P} \| \boldsymbol{u}_p - \boldsymbol{y}_p \|^2$$

byla minimální. Přitom

$\boldsymbol{y}_p = \lambda \left(\boldsymbol{W} \cdot \boldsymbol{x}_p + \boldsymbol{b} \right)$ je skutečný výstup sítě pro vstup $\boldsymbol{x}_p$,

$\boldsymbol{u}_p$ je požadovaný výstup sítě pro vstup $\boldsymbol{x}_p$,

P je počet trénovacích dvojic,

g_p je chyba od p-té trénovací dvojice.

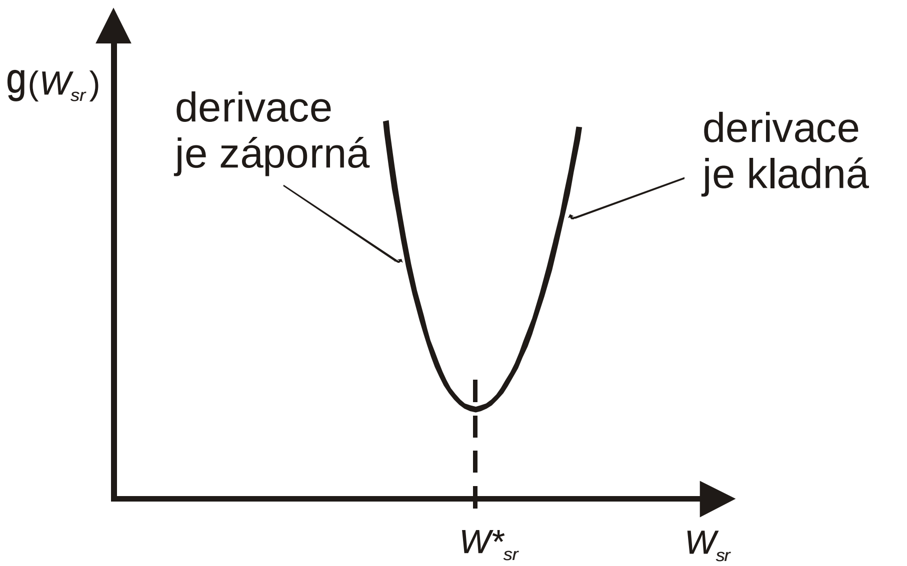
Odvození vztahů pro trénování lineární sítě

! Pro konkrétní trénovací dvojici, např. [***x***,***u***] definujeme dílčí chybu ve tvaru

$$\varepsilon = \frac{1}{2}\left\|\boldsymbol{u}-\boldsymbol{y}\right\|^2,$$

kde $\boldsymbol{y}=\lambda\left(\boldsymbol{W}\cdot\boldsymbol{x}+\boldsymbol{b}\right)$, a chceme změnit váhovou matici a prahový vektor tak, aby se chyba g zmenšila.

! Ilustrace závislosti chyby g na jedné konkrétní váze (např. w_{sr}



! Záporný gradient určuje směr největšího poklesu, kladný gradient směr největšího růstu. Tzn. že parametry sítě (tj. Váhy a prahy) je rozumné měnit zápornému gradientu chybové funkce, tj.

$$w_{sr}(k+1) = w_{sr}(k) - c\,\frac{\partial \varepsilon}{\partial w_{sr}}\ ,$$

C>0 je konstanta učení

! Protože

$$\varepsilon = \frac{1}{2}\,\|\boldsymbol{u}-\boldsymbol{y}\|^2 = \frac{1}{2}\sum_{j=1}^{m}(u_j - y_j)^2 = \frac{1}{2}\sum_{j=1}^{m}\left[u_j - \lambda\left(\sum_{i=1}^{n} w_{ji}\,x_i + b_i\right)\right]^2,$$

lze odvodit, že $\frac{\partial \varepsilon}{\partial w_{sr}} = -\lambda\,(u_s - y_s)\,x_r$. Pak tedy

$$w_{sr}(k+1) = w_{sr}(k) + c\,\lambda\,(u_s - y_s)\,x_r$$

a analogicky

$$b_s(k+1) = b_s(k) + c\,\lambda\,(u_s - y_s)\ .$$

Maticově

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) + c\,\lambda\,(\boldsymbol{u}-\boldsymbol{y})\cdot\boldsymbol{x}^{\mathrm{T}}$$

$$\boldsymbol{b}(k+1) = \boldsymbol{b}(k) + c\,\lambda\,(\boldsymbol{u}-\boldsymbol{y})$$

... Widrow - Hoffovo pravidlo pro trénování lineárních sítí (LMS pravidlo ... Least Mean Square)

Algoritmus trénování lineární sítě

Vstupní podmínky:

Je dána trénovací množina, tj. *P* dvojic [$\boldsymbol{x}_p$,$\boldsymbol{u}_p$], *p*=1,2,...,*P*.
$\boldsymbol{x}_p$ je vstupní vektor dimenze *n*
$\boldsymbol{u}_p$ je požadovaný výstupní vektor dimenze *m* pro vstupní vektor $\boldsymbol{x}_p$
Dále je dána strmost aktivační funkce 8.

1. krok - Inicializace

! Váhovou matici ***W***(1) typu *m*×*n* a prahový vektor ***b***(1) dimenze *m* inicializujeme malými náhodnými čísly.

! Zvolíme $c > 0$, $E_{\max} \$ 0$ a $TC_{\max} > 0$.

! Stanovíme $k = 1$, $p = 1$, $q = 0$ a $E(0) = 0$.

$E_{\max}$ je maximální povolená chyba, na kterou nebo pod kterou se chceme dostat
$TC_{\max}$ je maximální počet trénovacích cyklů, ve kterých chceme síť natrénovat
q je počet trénovacích cyklů
E je chyba způsobená nesprávnou činností sítě během jednoho trénovacího cyklu
c je konstanta učení

2. krok - Výpočet výstupu

$\boldsymbol{x}(k) = \boldsymbol{x}_p$

$\boldsymbol{y}(k) = 8\ (\boldsymbol{W}(k)@\boldsymbol{x}(k)+\boldsymbol{b}(k))$

$\boldsymbol{u}(k) = \boldsymbol{u}_p$

3. krok - Výpočet chyby

$E(k)=E(k\text{-}1)+½(\boldsymbol{u}(k)\text{-}\boldsymbol{y}(k))^{\mathrm{T}}\ (\boldsymbol{u}(k)\text{-}\boldsymbol{y}(k))$

4. krok - Modifikace vah a prahů

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) + c\,\lambda\left(\boldsymbol{u}(k) - \boldsymbol{y}(k)\right)\cdot\boldsymbol{x}^{\mathrm{T}}(k)$$

$$\boldsymbol{b}(k+1) = \boldsymbol{b}(k) + c\,\lambda\left(\boldsymbol{u}(k) - \boldsymbol{y}(k)\right)$$

5. krok - Konec trénovacího cyklu

Je-li $p < P$	potom	$p = p + 1$
		$k = k + 1$
		jdi na krok 2)
	jinak	$q = q + 1$
		jdi na krok 6)

6. krok - Konec trénování

$E_c(q) = E(k)$

Je-li $E_c(q)$ # E_{max}

potom KONEC (síť se podařilo natrénovat)

jinak je-li $q > TC_{max}$ potom KONEC (síť se nepodařilo natrénovat)

jinak $E(0) = 0$

$k = k + 1$

$p = 1$

jdi na krok 2)

Poznámky k trénování lineární sítě:

! *<u>Volba konstanty učení</u>*: Malá hodnota konstanty učení zpomaluje proces učení, pro velkou hodnotu hrozí nebezpečí nestability procesu učení.

! *<u>Vliv strmosti aktivační funkce</u>*: Má na proces učení stejný vliv jako konstanta učení → konstantu učení je třeba volit s ohledem na strmost aktivační funkce.

! Podobně jako u sítě s binární bipolární aktivační funkcí lze pro výpočet celkové chyby E_c využít též vztah

$$E = \frac{1}{2PM} \sum_{p=1}^{P} \|\boldsymbol{u}_p - \boldsymbol{y}_p\|^2 = \frac{1}{2PM} \sum_{p=1}^{P} \sum_{m=1}^{M} (u_{pm} - y_{pm})^2 ,$$

kde M je počet výstupů sítě, P je počet trénovacích dvojic v trénovací množině a u_{pm} resp. y_{pm} je m-tá složka vektoru $\boldsymbol{u}_p$, resp. $\boldsymbol{y}_p$. Takto definovaná chyba umožňuje lépe posoudit činnost sítí s různým počtem výstupů a různě velkou trénovací množinou.

! Pokud počet trénovacích dvojic trénovací množiny je o jednu větší, než je dimenze vstupního vektoru (tj. $P=n+1$, kde n je počet vstupů sítě), a vstupní vektory trénovací množiny jsou lineárně nezávislé, existuje jediné nastavení vah a prahů s minimální chybou a toto nastavení lze najít řešením soustavy lineárních algebraických rovnic, tzn. <u>bez učení</u>.