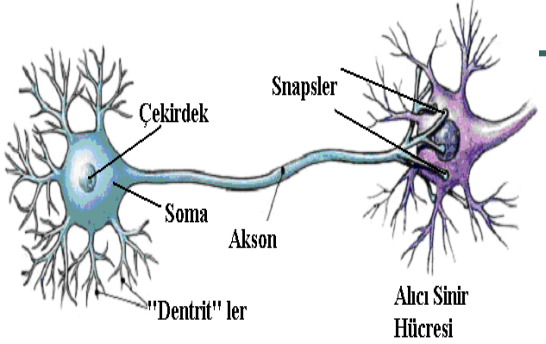
Fen Bilimleri Enstitüsü

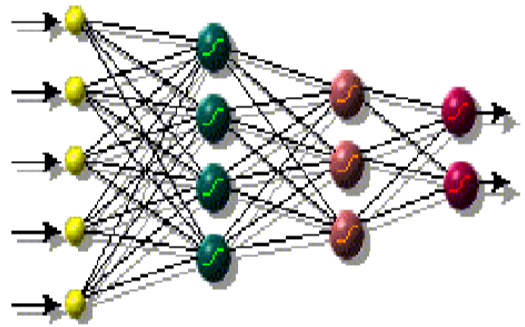
BİLGİSAYAR MÜHENDİSLİĞİ

BM508 Yapay Sinir Ağları
Ders3

Doç. Dr. Cihan KARAKUZU



Çok Katmanlı İleri Sürümlü YSA'da Geri Hesaplama (EĞİTİM/Hata Geri Yayılımı)



- *İleri sürümlü ve çok katmanlı bir YSA için eğiticili/danışmanlı öğrenme yöntemi olan GERİYE YAYILIM (Backpropagation) algoritması tanımlanacak ve türetilecektir.*
- *Algoritma çıkış katmanındaki hücre çıkışları (y_a) ile bu hücre çıkışları için istenen (y_d) çıkışlar arasındaki hata ($e = y_d - y_a$)'ya dayalı olarak işletilir.*
- ***Verilenler:*** *Eğitilecek ağ için önceden belirlenmiş bir eğitim kümemiz var…* $\left\{\left(\underset{\sim}{x}^{(n)}, \underset{\sim}{y_d}^{(n)}\right)\right\}\Big|_{n=1}^{p}$
- *Hesaplanacaklar : Önceden belirlenen ağ yapısı üzerinden, eğitime başlamadan rasgele atanmış ağ parametreleri ile eğitim kümesindeki k ncı giriş için ileri hesaplama ve hata belirleme….veee maliyet (cost) fonksiyonu (eğitim başarım ölçütü)*

j. Çıkış hücresi için hata→ $e_j^{(n)} = y_{d_j}^{(n)} - y_{a_j}^{(n)}$

j. Çıkış hücresi için ani hata→ $\frac{1}{2}\left(e_j^{(n)}\right)^2$

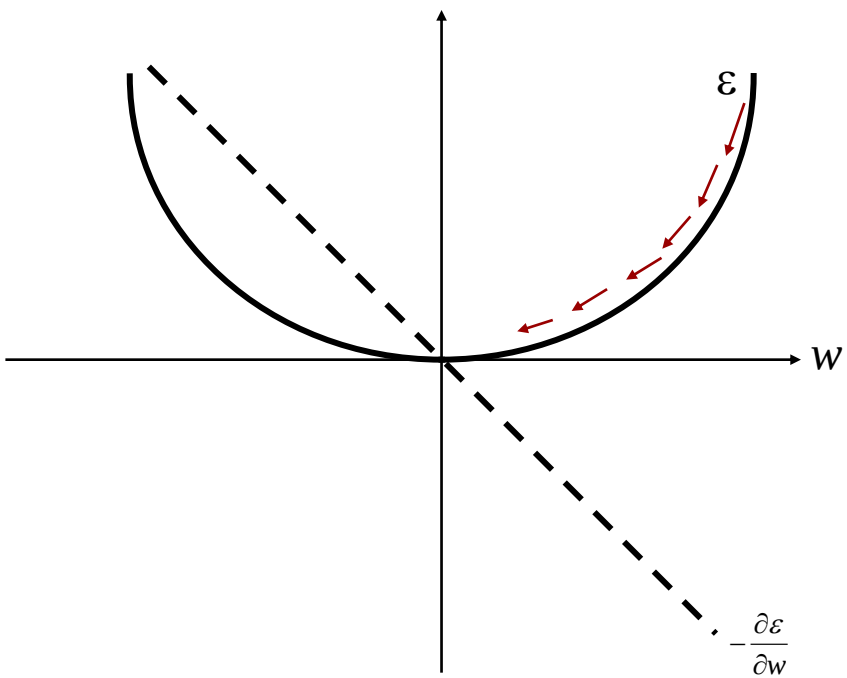
Toplam ani hata (cost f.)→ $\varepsilon^{(n)} = \frac{1}{2}\sum_{j \in C}\left(e_j^{(n)}\right)^2$

Şimdi geri hesaplama yapılacak ve parametreler bu hesaba göre güncellenecek…. NASIL?

- ***Algoritmanın tanımına göre:*** ***Herhangi bir iterasyonda, herhangi bir katmanındaki bir j. nöron ile bir önceki katmandaki bir i. nöron arasındaki bağlantı ağırlık parametresi***

$$w_{ji}(n+1) = w_{ji}(n) - \eta \frac{\partial \varepsilon(n)}{\partial w_{ji}(n)} \quad \{\text{Delta kuralıudik iniş (steepest-descent) yöntemi}\}$$

denklemi gereği güncellenir.

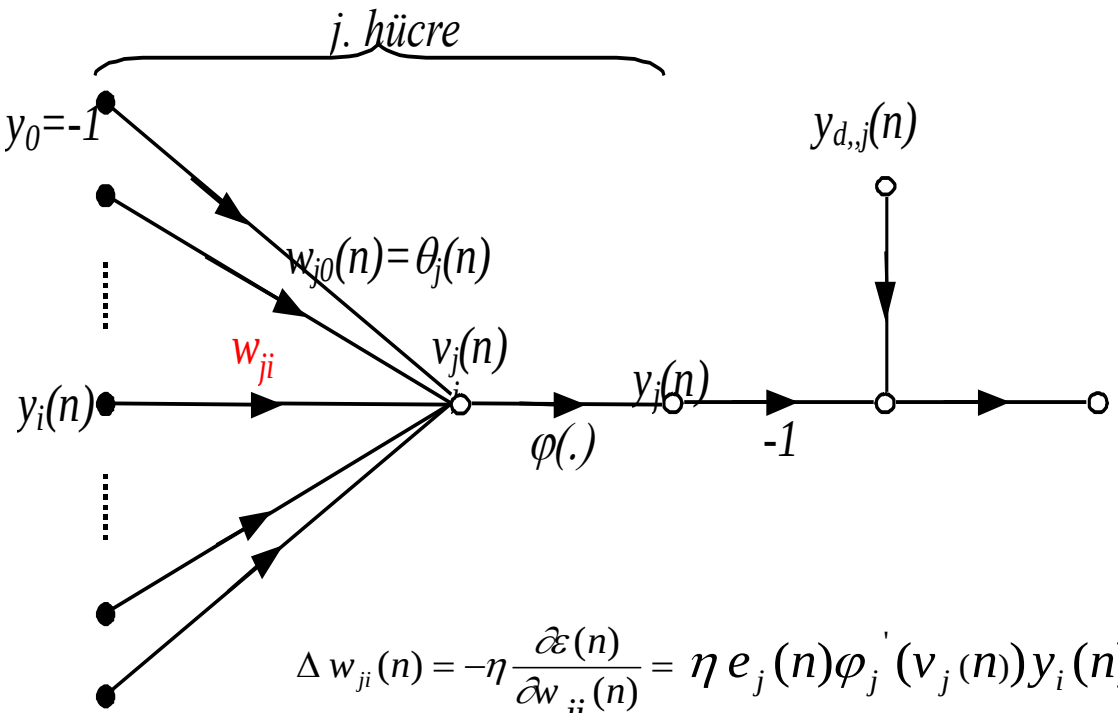


Yöntemin felsefesi

$\varepsilon = \frac{1}{2} w^2$ denklemi ile verilen tek parametreli bir değer (cost) fonksiyonunun en küçük değerini aldığı noktanın $\Delta w = -\eta \frac{\partial \varepsilon^{(k)}}{\partial w_{ji}^{(k)}}$ bağıntısı ile verilen kural ile iteratif olarak bulunabilmesine dayalıdır.

Değer fonksiyonu ve kısmi türev aynı eksenler üzerinde şekildeki gibi çizdirilsin. Şekilden de anlaşılacağı üzere her bir iterasyonda eğimin tersi yönde ilerlenirse değer fonksiyonunun minimumuna daha çok yaklaşılacaktır.

ALGORİTMANIN TÜRETİMİ: "j. hücre çıkış katmanında ise"



Geriye yayılım algoritması, $\frac{\partial \varepsilon(n)}{\partial w_{ji}(n)}$ ani gradyeni ile orantılı olan $w_{ji}(n)$ bağlantı ağırlığındaki $\Delta w_{ji}(n)$ düzeltme işlemini uygular.

Zincir kuralına göre; $\frac{\partial \varepsilon(n)}{\partial w_{ji}(n)}$ ani gradyeni şöyle ifade edilebilir :

$$\frac{\partial \varepsilon(n)}{\partial w_{ji}(n)} = \frac{\partial \varepsilon(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)}$$

$$\frac{\partial \varepsilon(n)}{\partial e_j(n)} = e_j(n) \qquad \frac{\partial e_j(n)}{\partial y_j(n)} = -1$$

$$\frac{\partial y_j(n)}{\partial v_j(n)} = \varphi_j'(v_j(n)) \qquad \frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n)$$

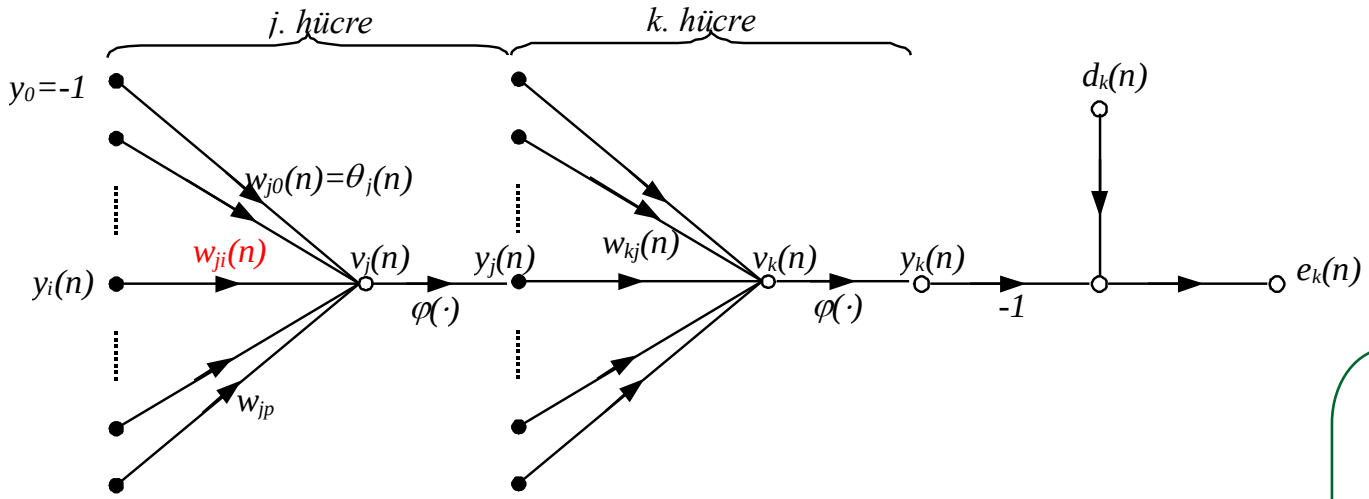
$$\frac{\partial \varepsilon(n)}{\partial w_{ji}(n)} = -e_j(n)\varphi_j'(v_j(n))y_i(n)$$

$$\Delta w_{ji}(n) = -\eta \frac{\partial \varepsilon(n)}{\partial w_{ji}(n)} = \eta\, e_j(n)\varphi_j'(v_j(n))y_i(n)$$

$$\delta_j(n) = -\frac{\partial \varepsilon(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} = e_j(n)\varphi_j'(v_j(n)) \quad \text{yerel gradyen}$$

$$\boxed{\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n)}$$

ALGORİTMANIN TÜRETİMİ: "j. hücre gizli katmanında ise"



$$\delta_j(n) = -\frac{\partial \varepsilon(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)}$$

$$= -\frac{\partial \varepsilon(n)}{\partial y_j(n)} \varphi_j'(v_j(n))$$

?

$$\varepsilon(n) = \frac{1}{2}\sum_{k\in C} e_k^2(n)$$

$$\frac{\partial \varepsilon(n)}{\partial y_j(n)} = \sum_k e_k \frac{\partial e_k(n)}{\partial y_j(n)}$$

$$\frac{\partial \varepsilon(n)}{\partial y_j(n)} = \sum_k e_k \frac{\partial e_k(n)}{\partial v_k(n)} \frac{\partial v_k(n)}{\partial y_j(n)}$$

$$e_k(n) = d_k(n) - y_k(n) = d_k(n) - \varphi_k(v_k(n))$$

$$\frac{\partial e_k(n)}{\partial v_k(n)} = -\varphi_k'(v_k(n))$$

$$v_k(n) = \sum_{j=0}^{q} w_{kj}(n) y_j(n)$$

$$\frac{\partial v_k(n)}{\partial y_j(n)} = w_{kj}(n)$$

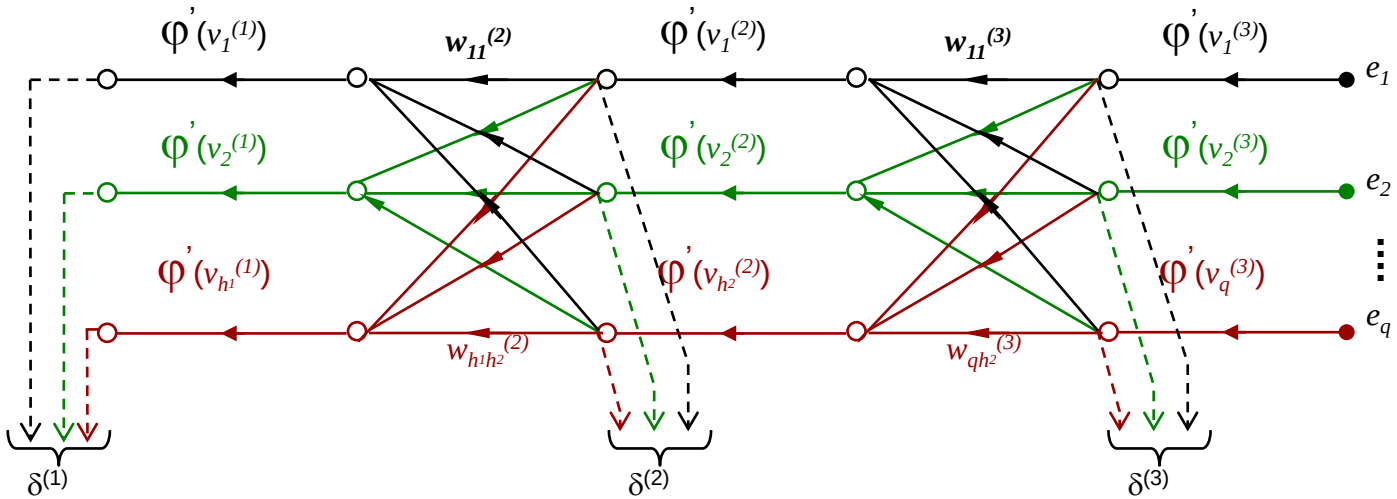
$$\frac{\partial \varepsilon(n)}{\partial y_j(n)} = -\sum_k e_k(n)\varphi_k'(v_k(n)) w_{kj}(n)$$

$$= -\sum_k \delta_k(n) w_{kj}(n)$$

$$\delta_j(n) = \varphi_j'(v_j(n)) \sum_k \delta_k(n) w_{kj}(n)$$

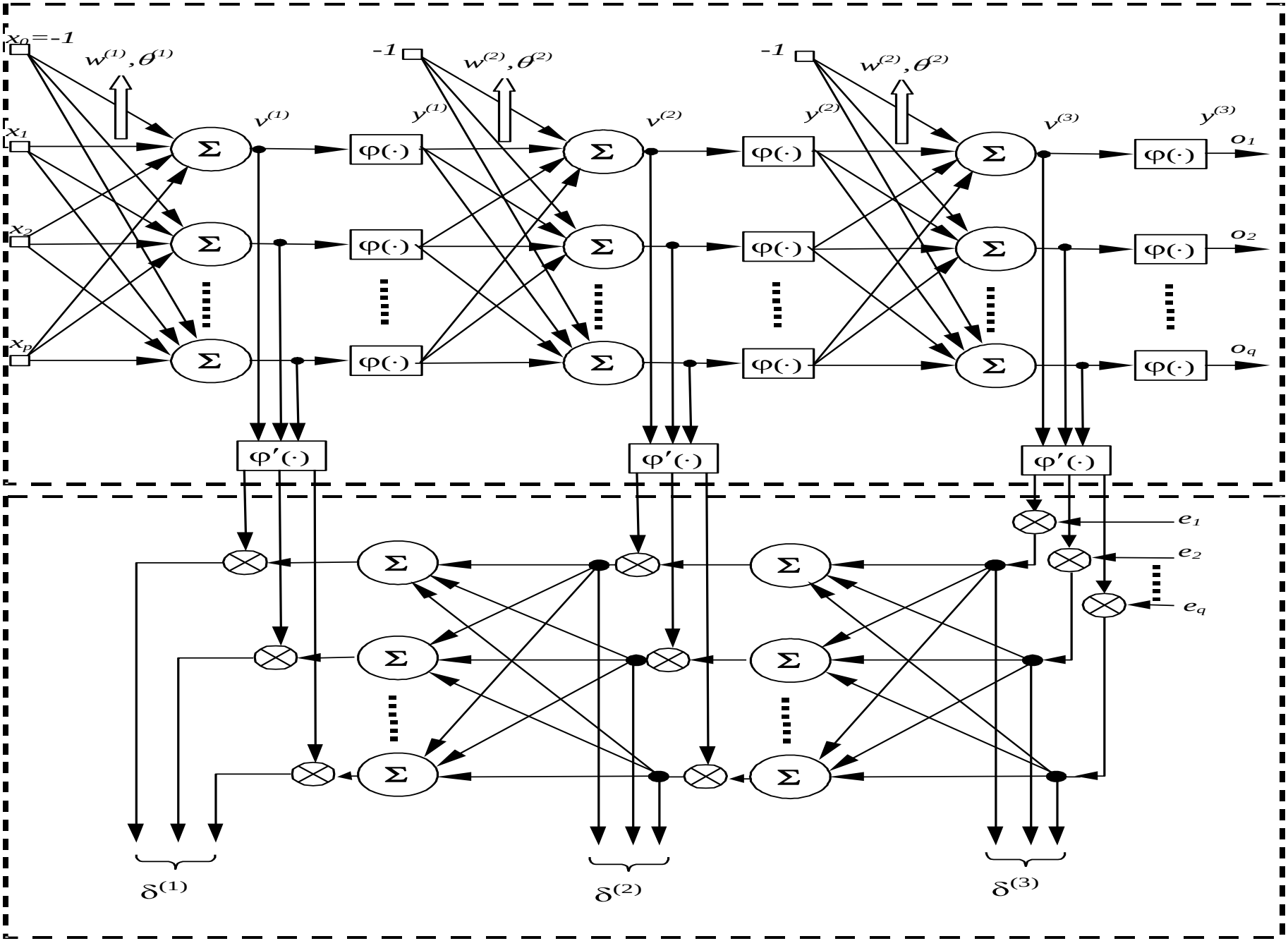
$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n)$$

3 katmanlı ileri sürümlü YSA için geri hesaplama (hata geri yayılımı/yerel gradyen hesabı) işaret akış diyagramı



Transpose network

İleri ve geri hesaplama şematik gösterimi



Aktivasyon (geçiş) işlevi türevleri

$$y = satlin(x) \Rightarrow \partial y/\partial x = tanimsiz$$
$$y = satlins(x) \Rightarrow \partial y/\partial x = tanimsiz$$

$$y = hard\,lim(x), \Rightarrow \partial y/\partial x = tanimsiz$$
$$y = hard\,lims(x) \Rightarrow \partial y/\partial x = tanimsiz$$

$$y = pureline(x) \Rightarrow \partial y/\partial x = 1$$

$$y = log\,sig(x) = \frac{1}{1+e^{-x}} \Rightarrow \partial y/\partial x = y(1-y)$$
$$y = \frac{1}{1+e^{-ax}} \Rightarrow \partial y/\partial x = ay(1-y)$$

$$y = tan\,sig(x) = \frac{2}{1+e^{-2x}} - 1 \Rightarrow \partial y/\partial x = (1-y^2)$$
$$y = a\tanh(bx) \Rightarrow \partial y/\partial x = ab(1-\tanh^2(bx))$$

$$y = gauss(x;\sigma,c) = e^{-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)^2}$$

$$\frac{\partial y}{\partial x} = -\frac{x-c}{\sigma^2}y$$

$$\frac{\partial y}{\partial \sigma} = -\frac{(x-c)^2}{\sigma^3}y$$

$$\frac{\partial y}{\partial c} = \frac{x-c}{\sigma^2}y$$