Backpropagation in Practice:

function [jVal, gradient] = ~~cost~~Function(theta)

↳ $\mathbb{R}^{n+1}$ → $\mathbb{R}^{n+1}$

optTheta = fminunc(@cost

Neural Network (L = 4)

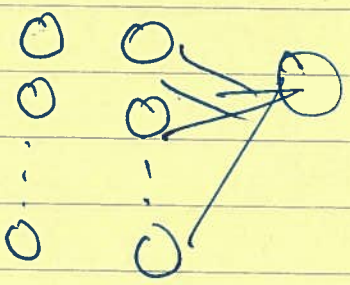
$\Theta^{(1)}, \Theta^{(2)}, \Theta^{(3)}$ → matrices (Theta1, Theta2, Thet

$D^{(1)}, D^{(2)}, D^{(3)}$ - matrices (D1, D2, D3)

"Unroll into vectors"

Example

$s_1 = 10, s_2 = 10, s_3 = 1$

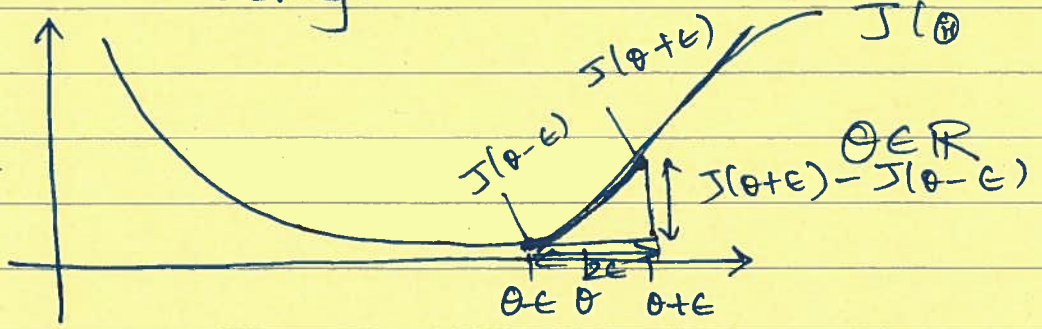


$\Theta^{(1)} \in \mathbb{R}^{10 \times 11}, \Theta^{(2)} \in \mathbb{R}^{10 \times 11}$

$\Theta^{(3)} \in \mathbb{R}^{1 \times 11}$

$D^{(1)} \in \mathbb{R}^{10 \times 11}, D^{(2)} \in \mathbb{R}^{10 \times 11}, D^{(3)} \in \mathbb{R}^{1 \times 11}$

Gradient Checking:



$$\frac{dJ(\theta)}{d\theta} \approx \frac{J(\theta+\varepsilon) - J(\theta-\varepsilon)}{2\varepsilon} \quad \Big| \quad \frac{J(\theta+\varepsilon) - J(\theta)}{\varepsilon}$$

$\varepsilon = 10^{-4}$

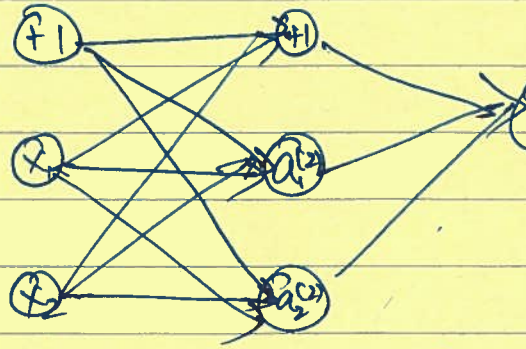
2 sided Difference. One sided differe

~~$1+\theta(1+0.01)^3 \theta - 1$~~

$$\frac{(1.01)^3 - (0.99)^3}{0.02} =$$

Random Initialization

Zero initialitation



$\Theta_{ij}^{(l)} = 0$ for all i, j, l

$a_1^{(2)} = a_2^{(2)}$ $\quad$ $\Theta_{01}^{(1)} = \Theta_{02}^{(1)}$

Also $\delta_1^{(2)} = \delta_2^{(2)}$

$$\frac{\partial}{\partial \Theta_{01}^{(1)}} J(\Theta) = \frac{\partial}{\partial \Theta_{02}^{(1)}} J(\Theta)$$

$a_1^{(2)} = a_2^{(2)}$

① $\Delta_{ij}^{(2)} := \Delta_{ij}^{(2)} + \delta_i^{(3)} * (a^{(2)})_j$

$\Delta^{(2)} := \Delta^{(2)} + \delta^{(3)} . (a^{(2)})^T$

② $\Theta_1 \quad \Theta_2$
$5 \times 3, \quad 4 \times 6$
reshape((16:39), 4, 6)

③ $J(\theta) = 2\theta^4 + 2$
$\theta = 1 \quad \epsilon = 0.01$

④ Using Grad check can help verify if ~~back prop~~ FP is bug free

⑤ comp efficiency, after we have fea. If our neural net overfits tra set, increase λ.

⑥ $J(\theta)$ ina α too large
plot $J(\theta)$ number of iteration and make sure dec

if NN using grad descent,

if you run algorithm twice with diff random initialization, gd may converge to two diff sol-n