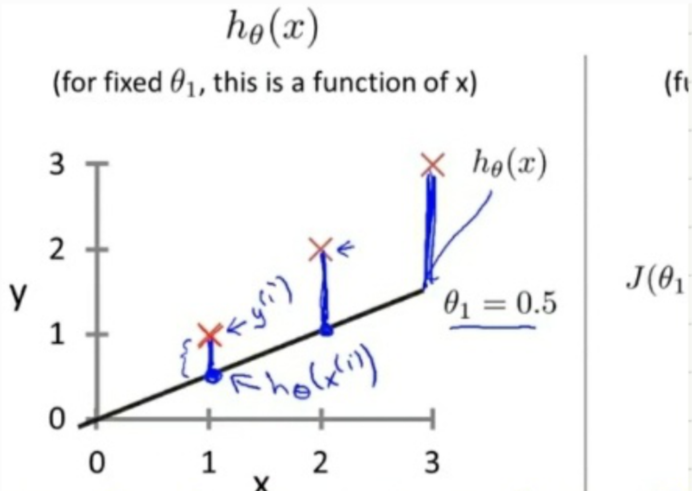
Create the best overall fit line.

- <u>Hypothesis function</u>: $h_\theta(x) = \theta_0 + \theta_1 x$ → unique feature

Our goal is to minimize a cost function, to commit the minimum number of errors.

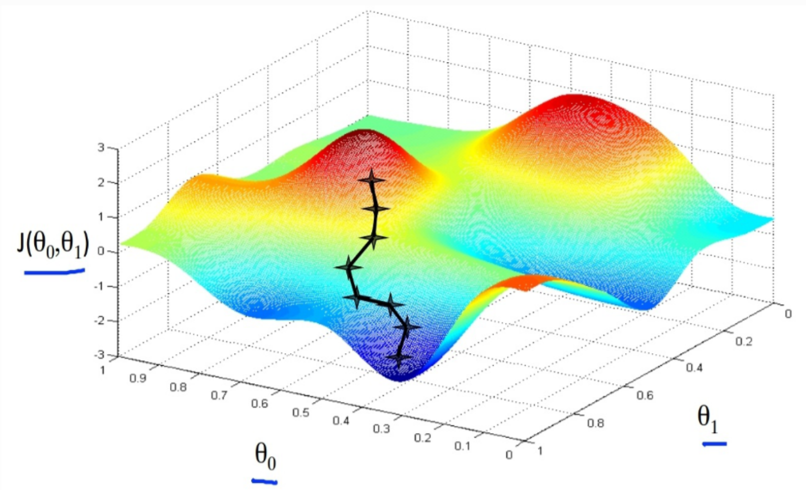
- <u>Cost function</u>: $$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

We iterate through values of θ_0 and θ_1 until we reach a minimum error point (converges).

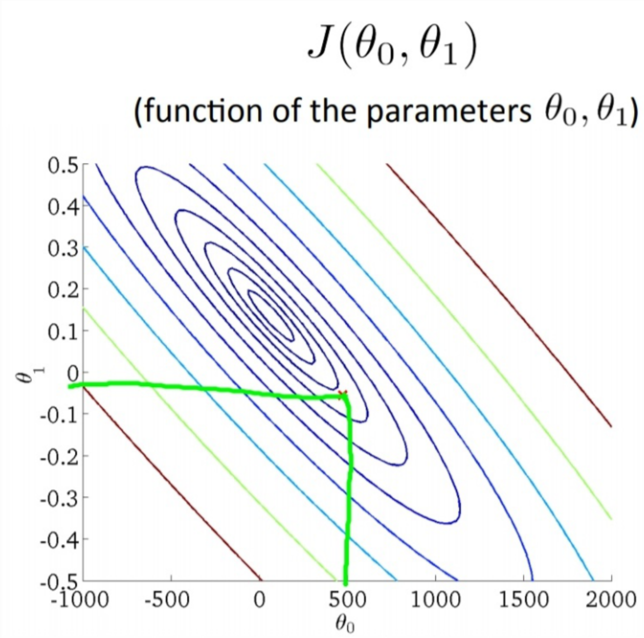


$J(0.5) = \frac{1}{2m}\left[(0.5-1)^2 + (1-2)^2 + (1.5-3)^2\right]$

$= \frac{1}{2\times 3}(3.5) = \frac{3.5}{6} \approx 0.68$

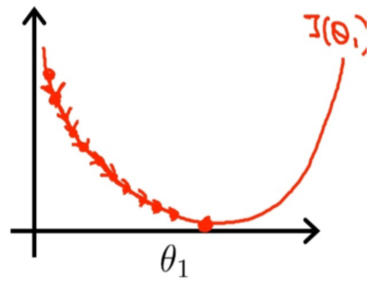


- <u>Gradient descent</u>:

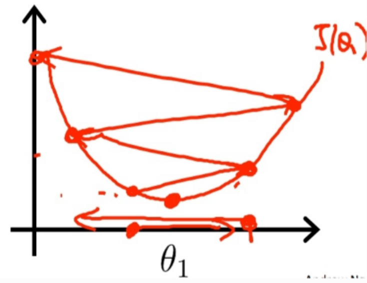


$$\boxed{\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)}$$

If α is too small, gradient descent can be slow. →



If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge. →



Gradient descent algorithm

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

(for $j = 1$ and $j = 0$)

}

"Batch" Gradient Descent

"Batch": Each step of gradient descent uses all the training examples.

WITH MULTIPLE FEATURES: