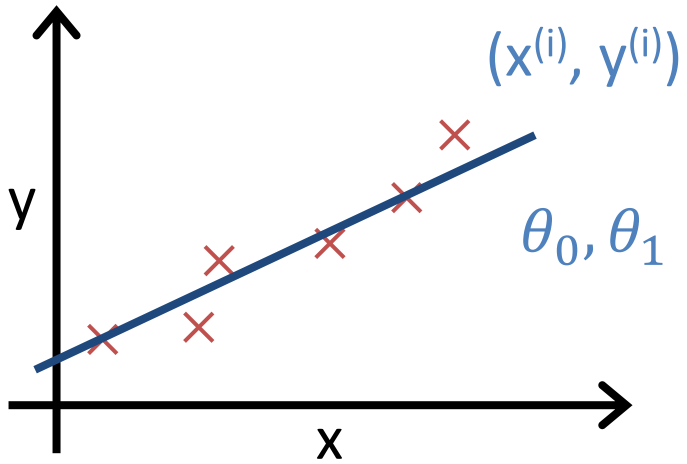
Gradient Descent

By Vipul Goyal

Linear Regression



Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ is "close" to y for our training examples $(x^{(i)}, y^{(i)})$

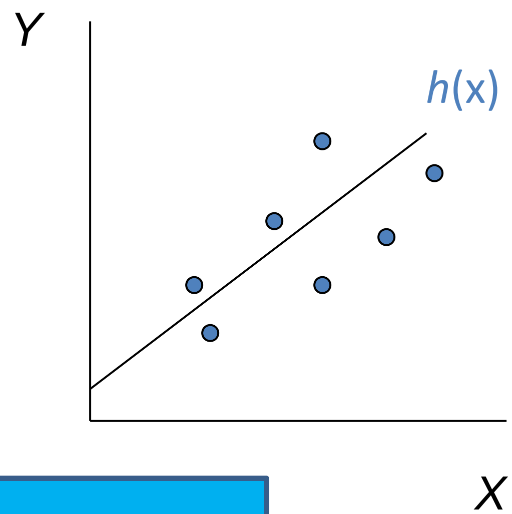
Cost Function

Hypothesis:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

Parameters:

θ_0, θ_1



Cost Function (squared error function):

$$J\,(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

Goal: minimize $J\,(\theta_0, \theta_1)$

θ_0, θ_1

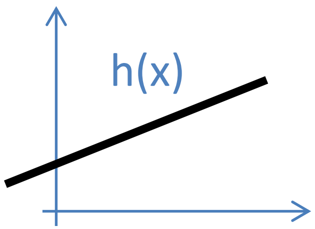
Simplified Cost Function

Hypothesis:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

Parameters:

$$\theta_0 + \theta_1$$



Cost Function:

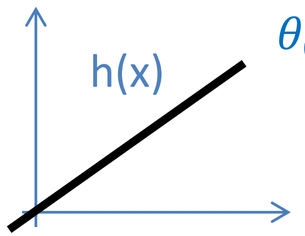
$$J\,(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J\,(\theta_0, \theta_1)$

Simplified

$$h_\theta(x) = \theta_1 x$$

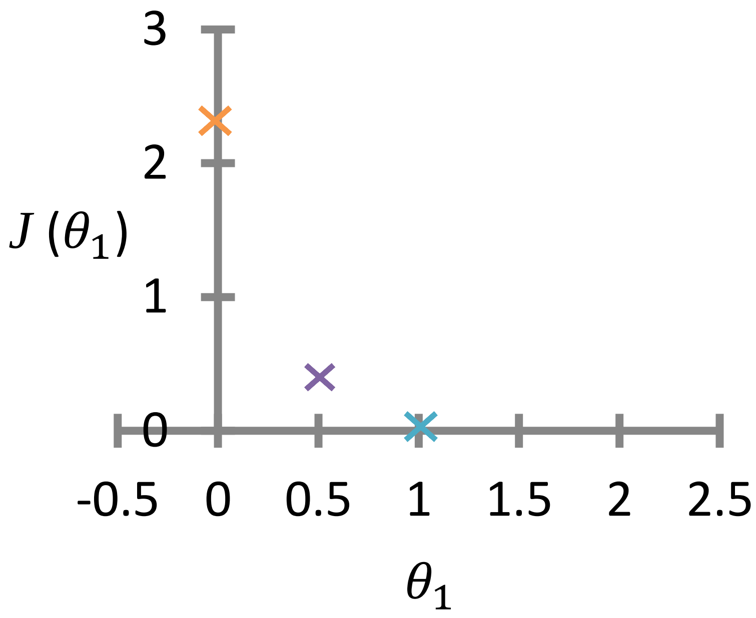
$$\theta_1$$

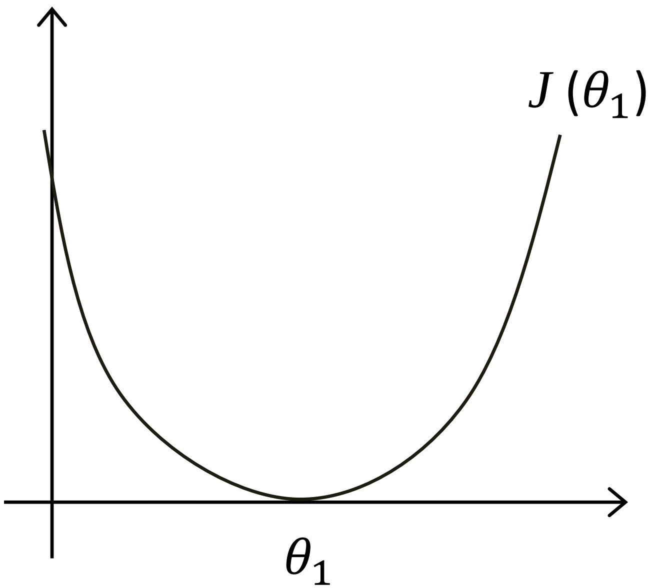


$$J\,(\theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

$\underset{\theta_1}{\text{minimize}}\ J\,(\theta_1)$

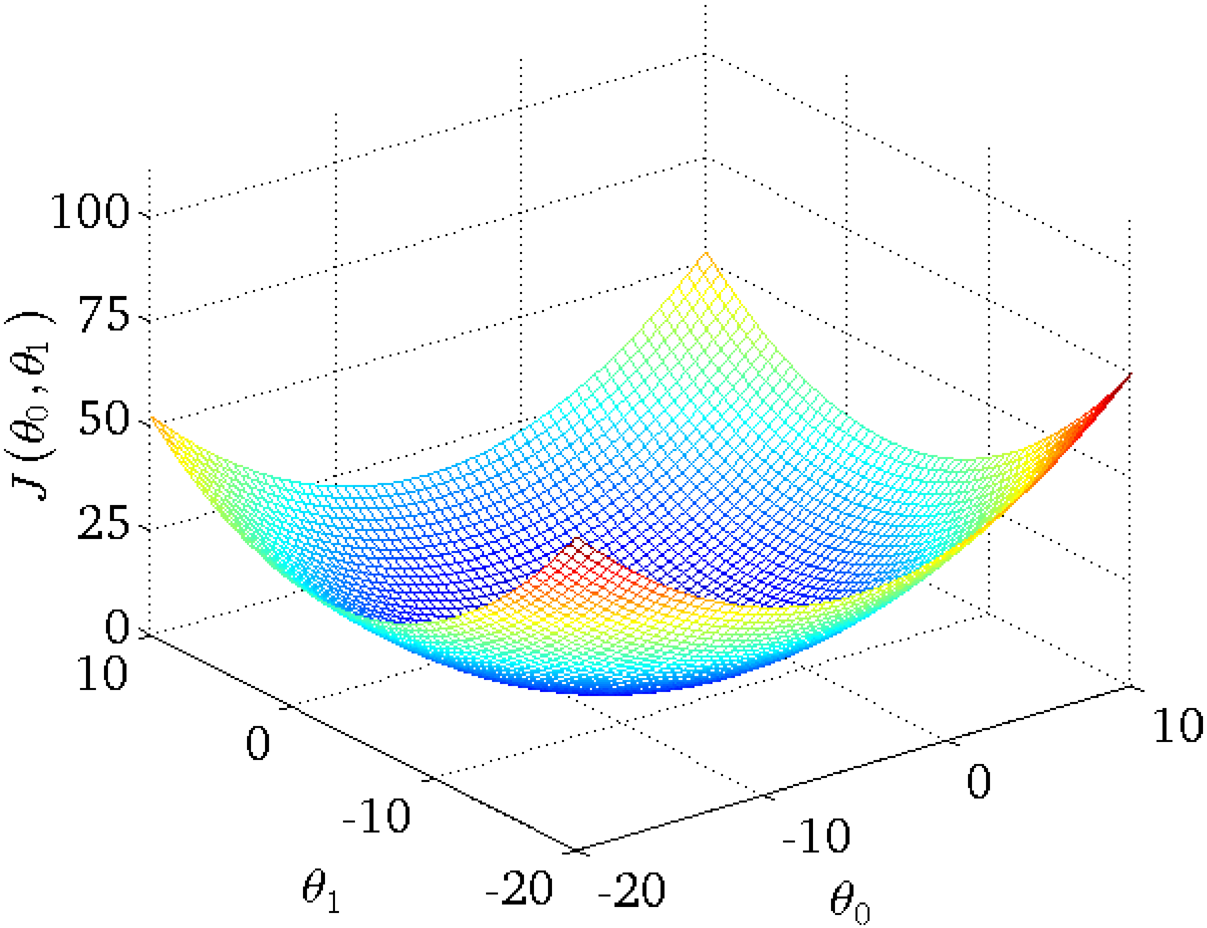
Plotting the Cost Function





This is a 1-dimensional cost function. θ_1 is the only variable here. What if θ_0 was also non-zero?

2-Dimensional Cost Function



Goal: Find θ_0 and θ_1 for which the cost function is minimized. Gradient Descent Algorithm!

Gradient Descent Idea

Have cost function $J(\theta_0, \theta_1)$

Compute $$\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$$

Idea:

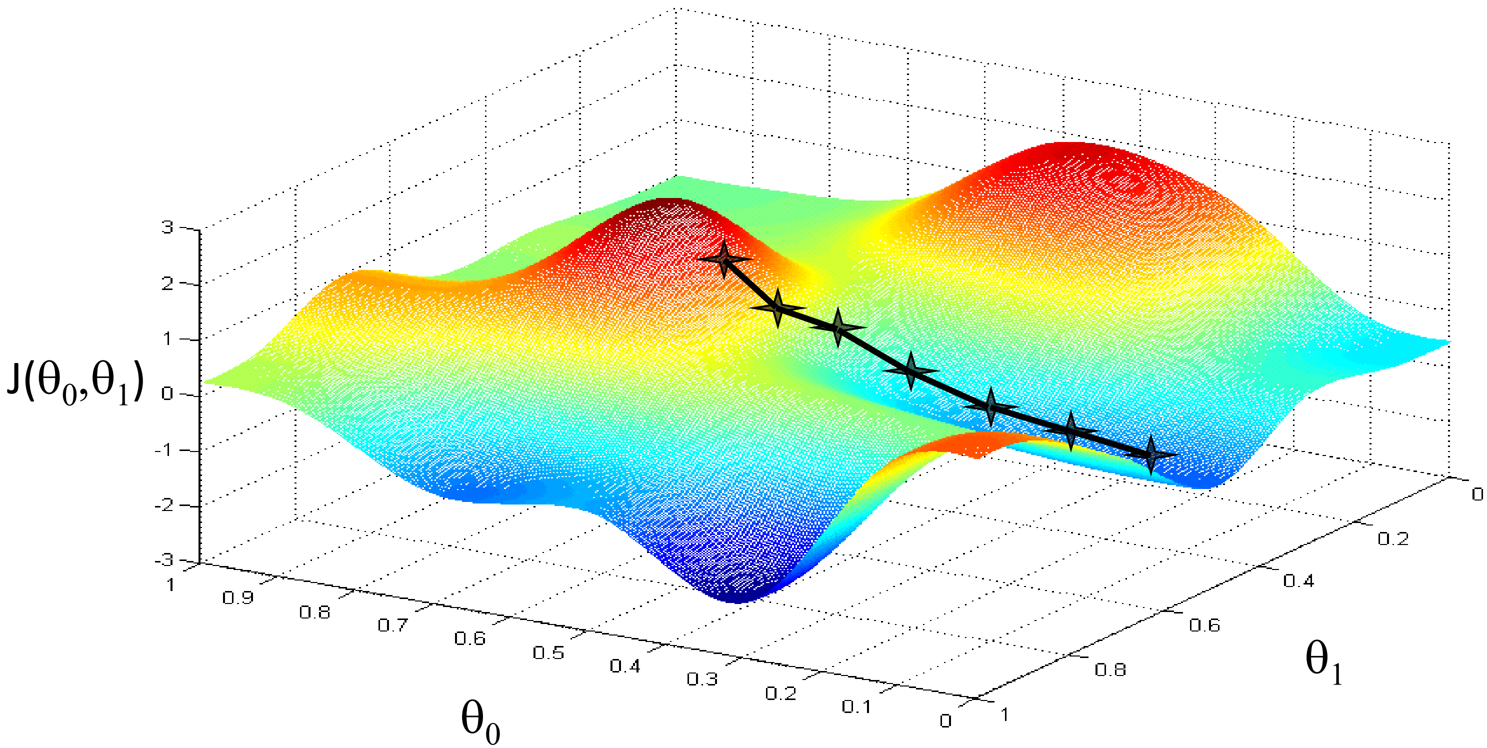
- Start with some arbitrary θ_0, θ_1
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ until we hopefully end up at a minimum

Will discuss: how to change to ensure reduction in $J(\theta_0, \theta_1)$

Gradient Descent Idea

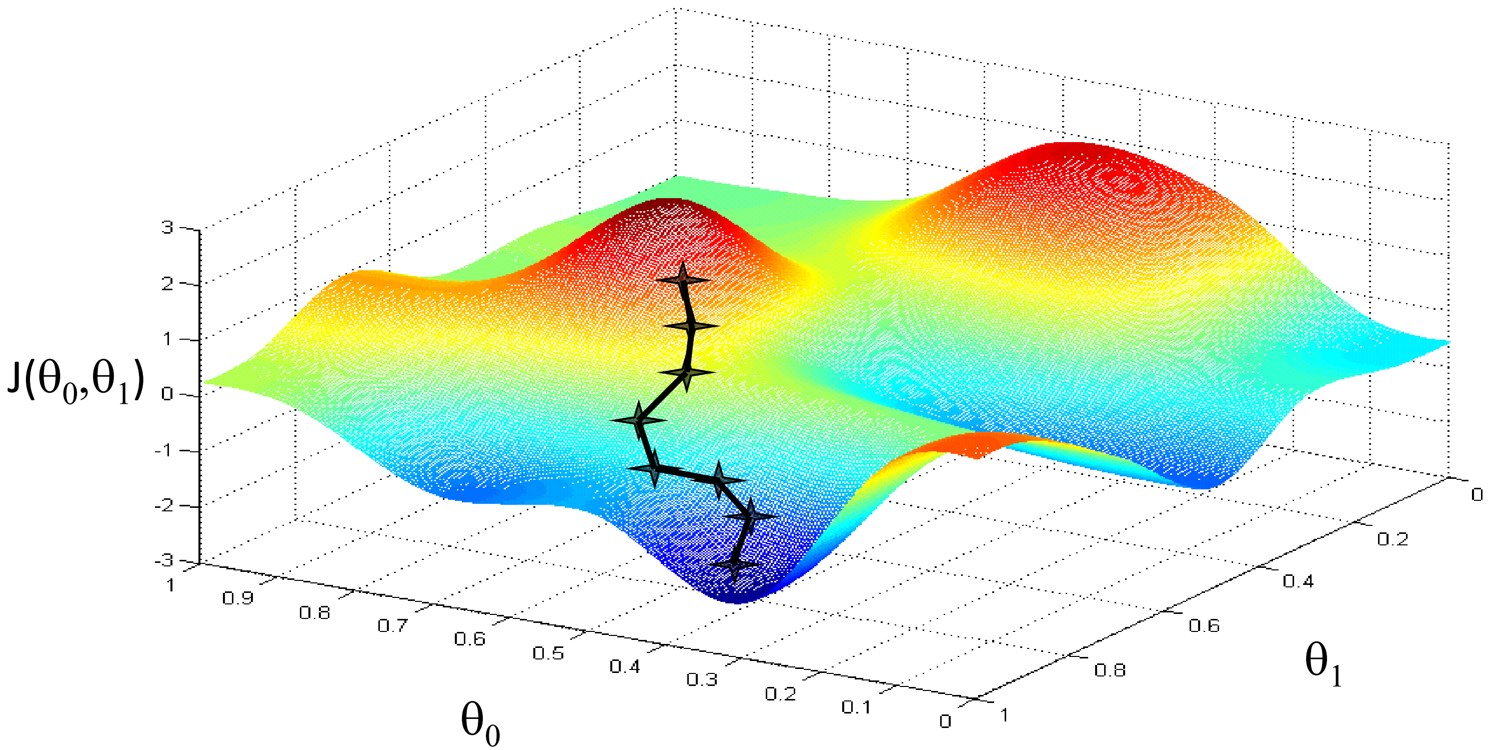
- Start with initial guesses
  - Start at random values (or some default values)
  - Keeping changing θ_0 and θ_1 a little bit to try and reduce $J(\theta_0, \theta_1)$
  - Each time you change the parameters, you select the gradient/direction which reduces $J(\theta_0, \theta_1)$ the most possible
- Repeat until you converge to a local minimum
- Has an interesting property
  - Where you start can determine which minimum you end up

Gradient Descent Execution 1

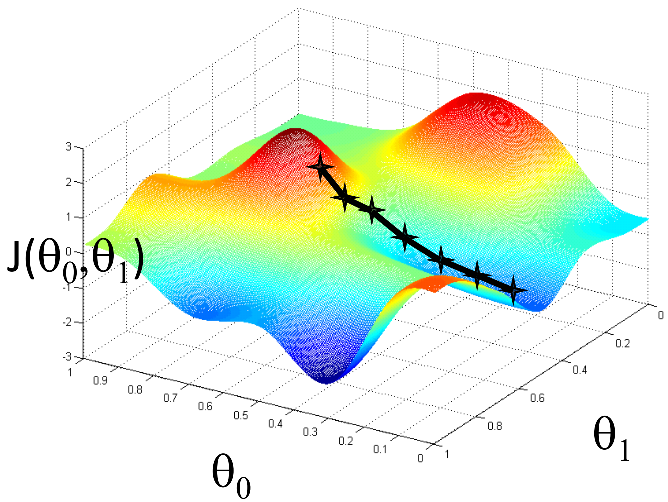


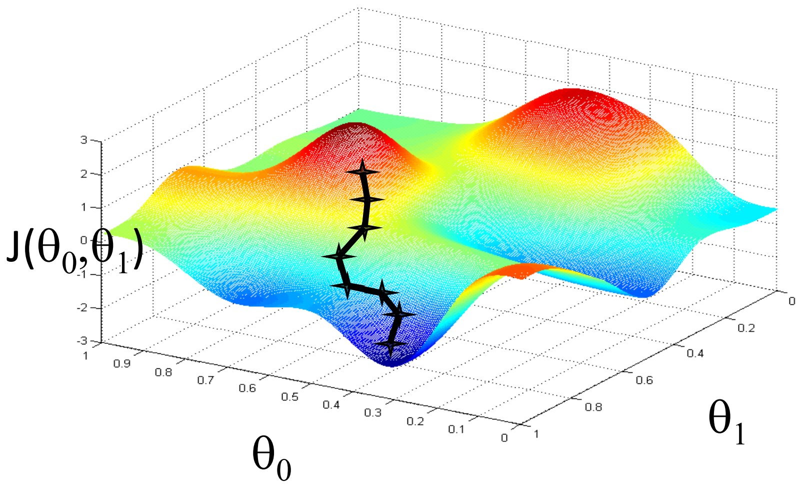
Think: A ball falling down from gravity

Gradient Descent Execution 2



Local Minima Problem





- Think: A ball falling down from gravity
- Starting point matters: different starting points can lead to different end points
- Ball can get stuck in a “local minima”. Unable to move.

Gradient Descent: More Intuition

Gradient Descent Intuition

Gradient descent algorithm

assignment

a:=b

$repeat\ until\ convergence\quad \{$

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \qquad j = 0 \text{ and } j = 1$$

$\}$

Learning rate

Correct: Simultaneous update

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$\theta_0 := \text{temp0}$$

$$\theta_1 := \text{temp1}$$

Incorrect:

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\theta_0 := \text{temp0}$$

$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$\theta_1 := \text{temp1}$$

The Derivative Term

Gradient descent algorithm

$$repeat\ until\ convergence\quad \{$$

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

$$\}$$

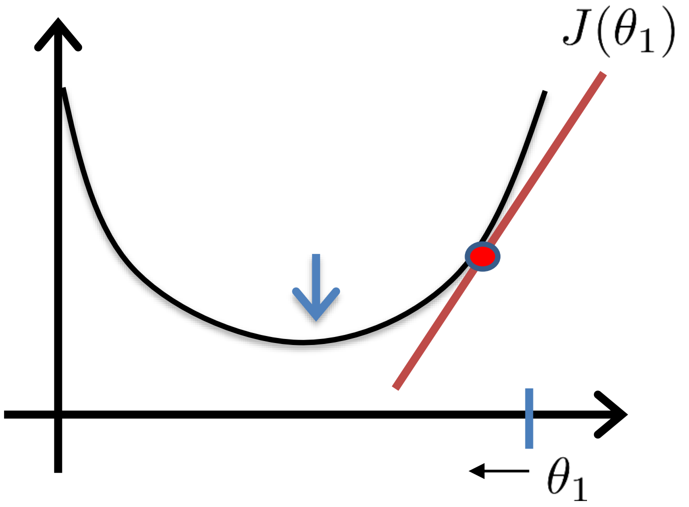
Learning rate

derivative

$$\frac{d}{d\theta_1} J(\theta_0, \theta_1) = \text{Rate of change of } J(\theta_0, \theta_1) \text{ as } \theta_1\ changes$$

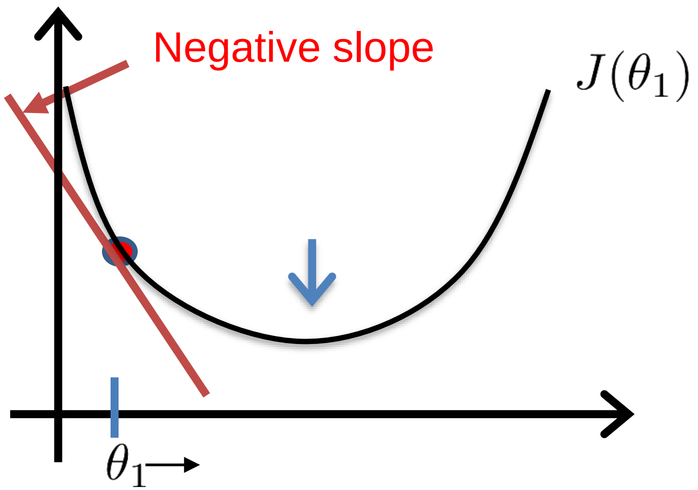
Could be positive or negative

Derivative of J



$$\theta_1 := \theta_1 - \alpha \underbrace{\boxed{\frac{d}{d\theta_1} J(\theta_1)}}_{\geq 0}$$

$$\theta_1 := \theta_1 - \alpha \cdot \ (positive\ number)$$

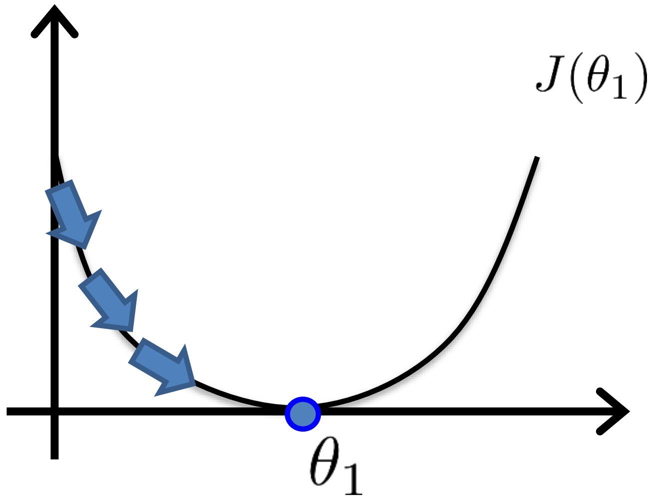


$$\theta_1 := \theta_1 - \alpha \underbrace{\boxed{\frac{d}{d\theta_1} J(\theta_1)}}_{\leq 0}$$

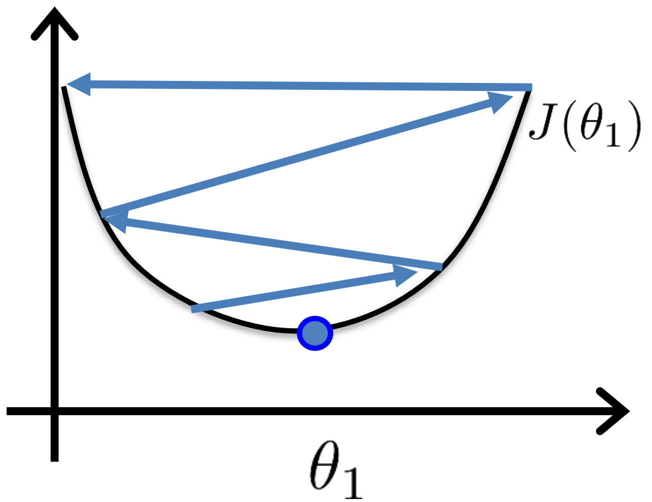
$$\theta_1 := \theta_1 - \alpha \cdot \ (negative\ number)$$

Learning Rate

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



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

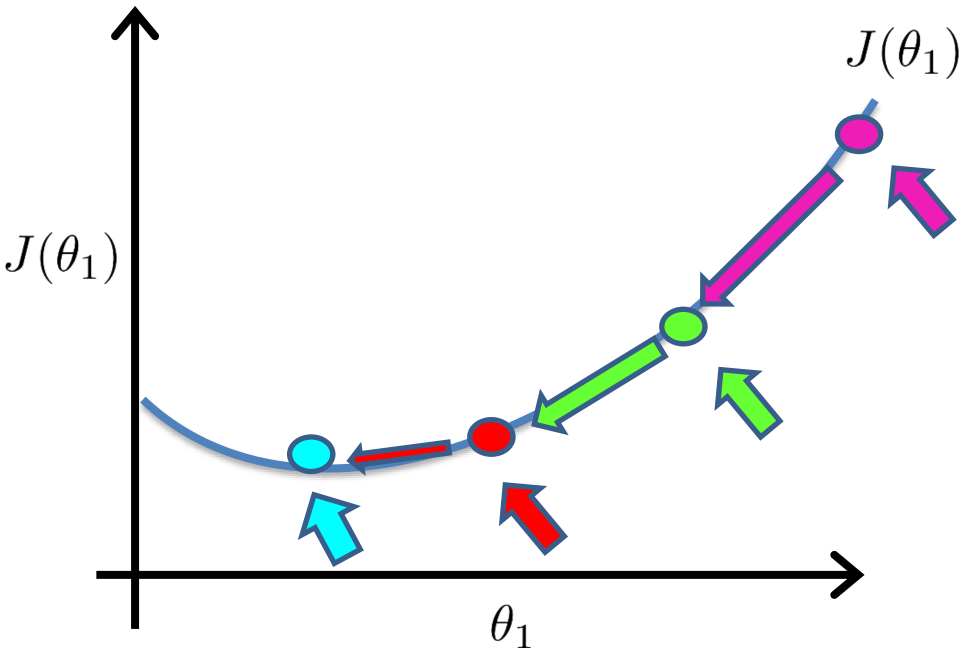


If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.

Convergence

Gradient descent can converge to a local minimum, even with the learning rate α fixed.

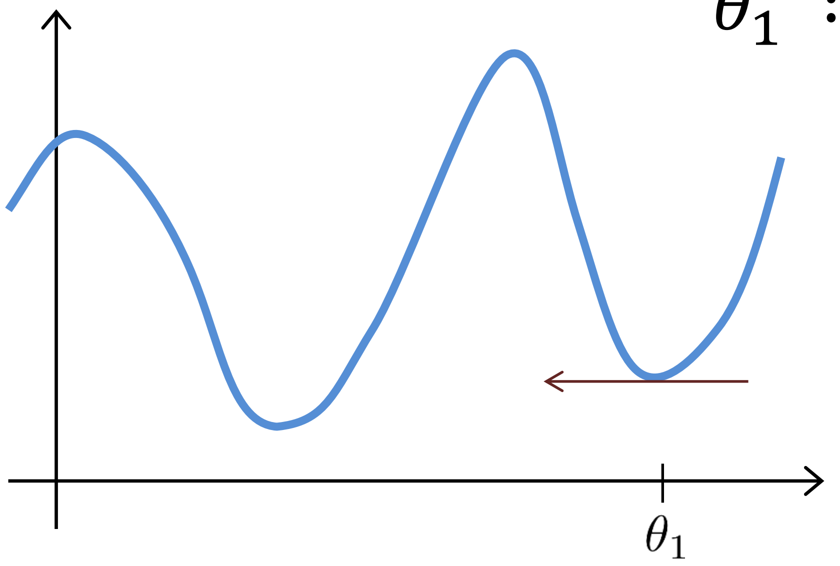
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$



As we approach a local minimum, gradient descent will automatically take smaller steps. So, no need to decrease α over time.

Derivative is 0 at Local Minima

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



Gradient Descent Details

Understanding Derivatives

$\frac{d}{d\theta} f(\theta)$ = rate of change of *$f(\theta)$* with θ

$\frac{d}{d\theta} f(\theta)$ = 2 means if θ increases by 1, f(θ) increase by 2.
Example f(θ) = 2θ

$\frac{d}{d\theta_0} f(\theta_0, \theta_1)$ = rate of change of $f(\theta_0, \theta_1)$ with θ_0 (partial derivative)

Key Formulas for Computing Derivatives

1) $\frac{d}{d\theta} c \cdot \theta = c$

2) $\frac{d}{d\theta} c \cdot \theta^n = c \cdot n \cdot \theta^{n-1}$

$$\Longrightarrow \frac{d}{d\theta} \theta^2 = 2\theta$$

3) $\frac{d}{d\theta_0} f(\theta_1) = \frac{d}{d\theta_1} f(\theta_0) = 0$

First Derivative

$$\frac{d}{d\theta_0} J\,(\theta_0,\ \theta_1)$$

$$= \frac{d}{d\theta_0} \cdot \frac{1}{2m} \sum_{i=1}^{m} (h(x^{(i)}) - y^{(i)})^2$$

$$= \frac{d}{d\theta_0} \cdot \frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 \cdot x^{(i)} - y^{(i)})^2$$

$$= \frac{d}{d\theta_0} \cdot \frac{1}{2m} \sum_{i=1}^{m} (\theta_0{}^2 + (\theta_1 \cdot x^{(i)} - y^{(i)})^2 + 2\theta_0 (\theta_1 \cdot x^{(i)} - y^{(i)}))$$

$$= \frac{1}{2m} \sum_{i=1}^{m} (2\theta_0 + 0 + 2\,(\theta_1 \cdot x^{(i)} - y^{(i)}))$$

$$= \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta (x^{(i)}) - y^{(i)})$$

Second Derivative

$$\frac{d}{d\theta_1} J(\theta_0, \theta_1)$$

$$= \frac{d}{d\theta_1} \cdot \frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 \cdot x^{(i)} - y^{(i)})^2$$

$$= \frac{d}{d\theta_1} \cdot \frac{1}{2m} \sum_{i=1}^{m} ((\theta_1)^2 (x^{(i)})^2 + (\theta_0 - y^{(i)})^2 + 2 \cdot \theta_1 \, x^{(i)} \cdot (\theta_0 - y^{(i)}))$$

$$= \frac{1}{2m} \sum_{i=1}^{m} (2\theta_1 (x^{(i)})^2 + 0 + 2x^{(i)}(\theta_0 - y^{(i)}))$$

$$= \frac{1}{m} \sum_{i=1}^{m} (\theta_1 \cdot x^{(i)} + \theta_0 - y^{(i)}) \cdot x^{(i)}$$

$$= \frac{1}{m} \sum_{i=1}^{m} (h_\theta (x^{(i)}) - y^{(i)}) \, . \, x^{(i)}$$

Linear Regression

Gradient descent algorithm

$\frac{d}{d\theta_0} \cdot J(\theta_0, \theta_1)$

repeat until convergence {

$$\theta_0 := \theta_0 - \alpha \boxed{\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})}$$

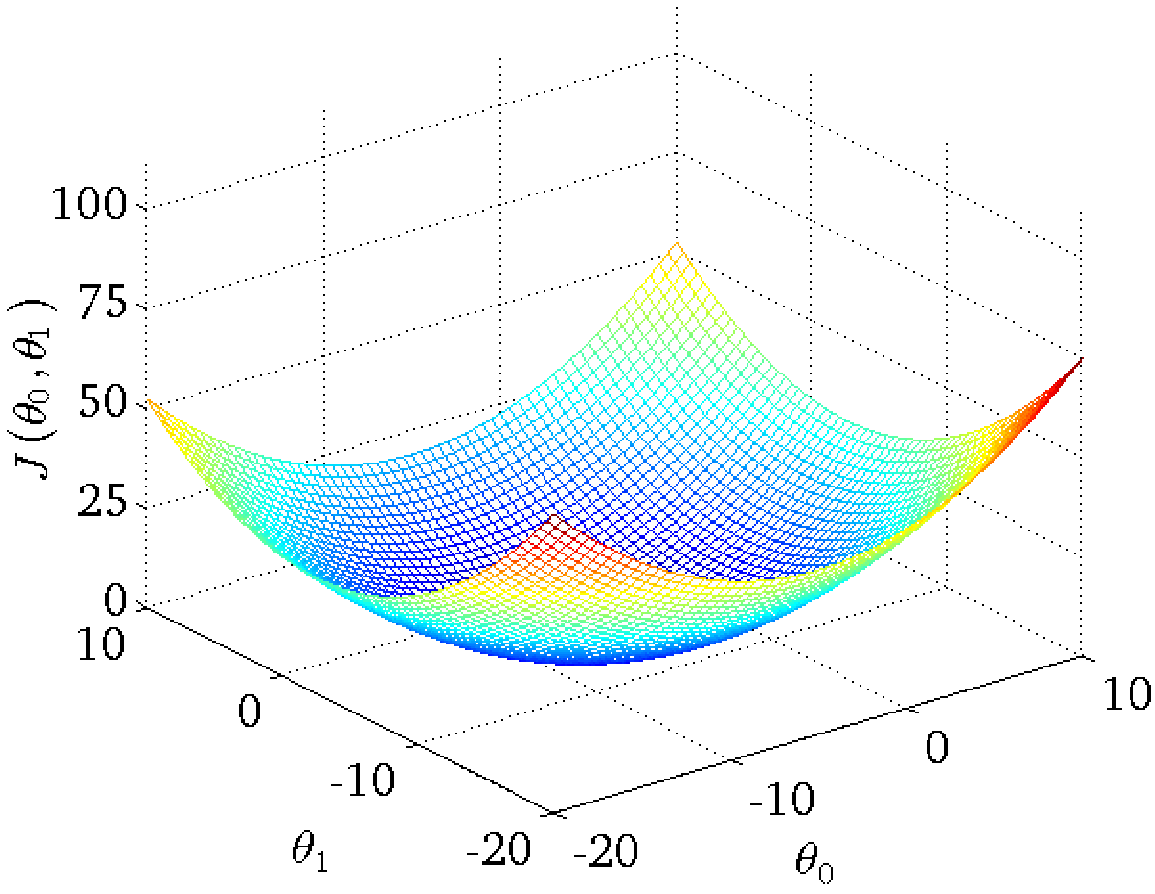
$$\theta_1 := \theta_1 - \alpha \boxed{\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)}}$$

update θ_0 and θ_1 simultaneously

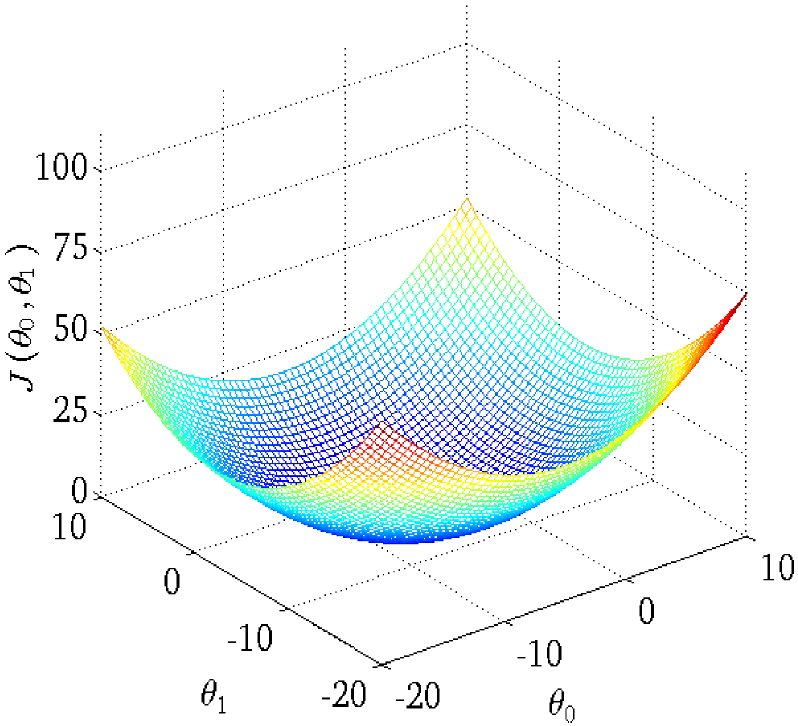
}

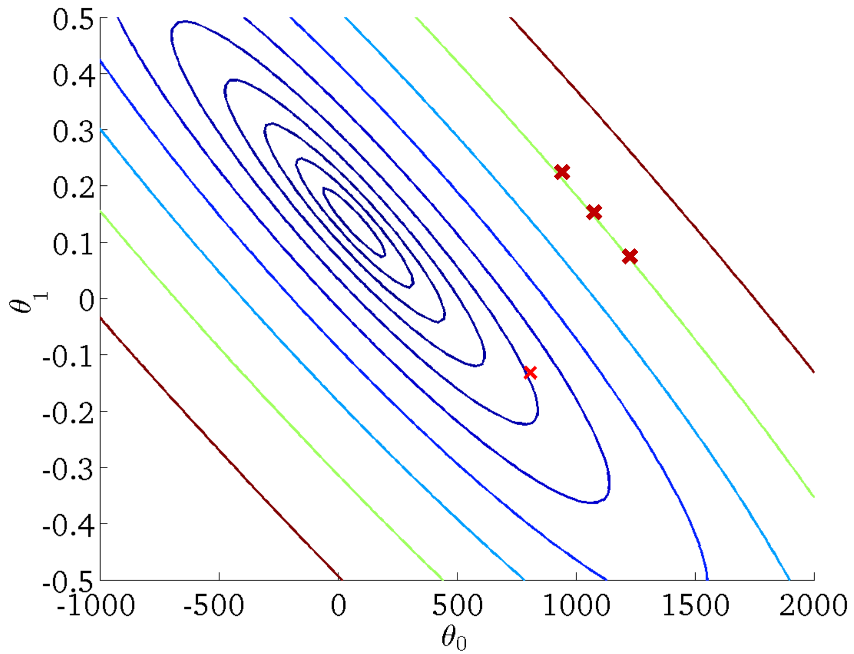
$\frac{d}{d\theta_1} \cdot J(\theta_0, \theta_1)$

3-D Plot of J



Contour Plot vs Surface Plot



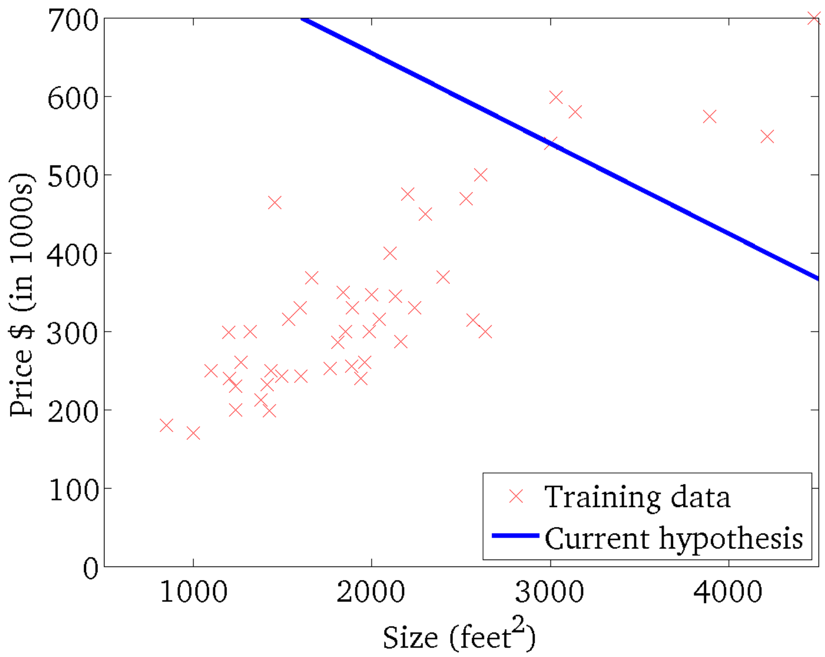


All points on a single "contour/color" have the same J value

Gradient Descent Iterations

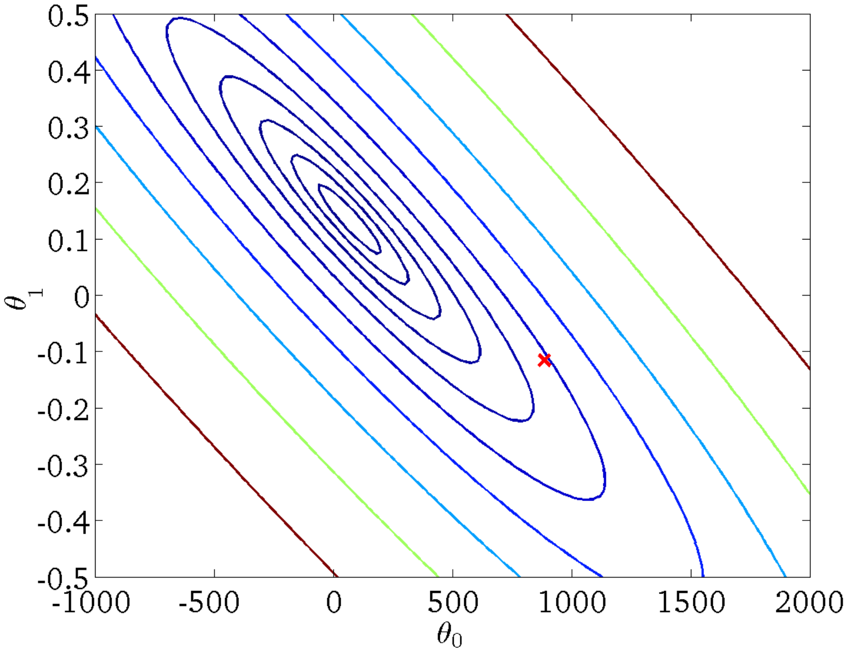
$h_\theta(x)$

(for fixed θ_0, θ_1 this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

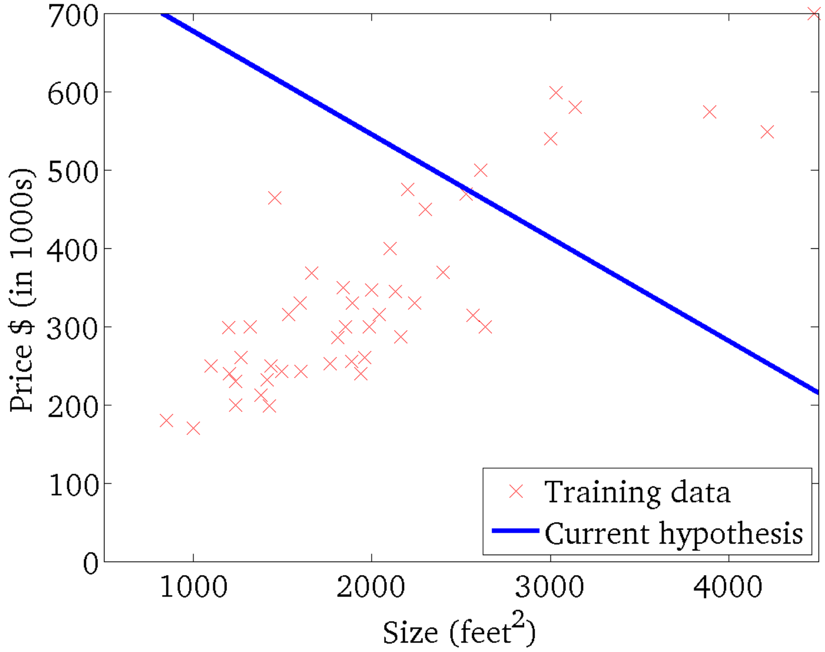


Reduce θ_0, reduce θ_1

Gradient Descent Iterations

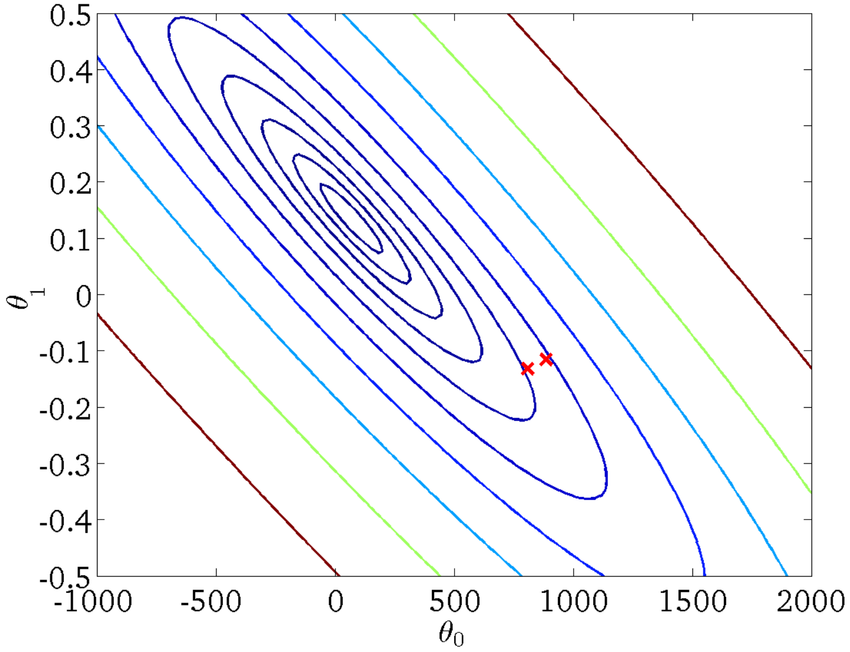
$h_\theta(x)$

(for θ_0, θ_1 this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

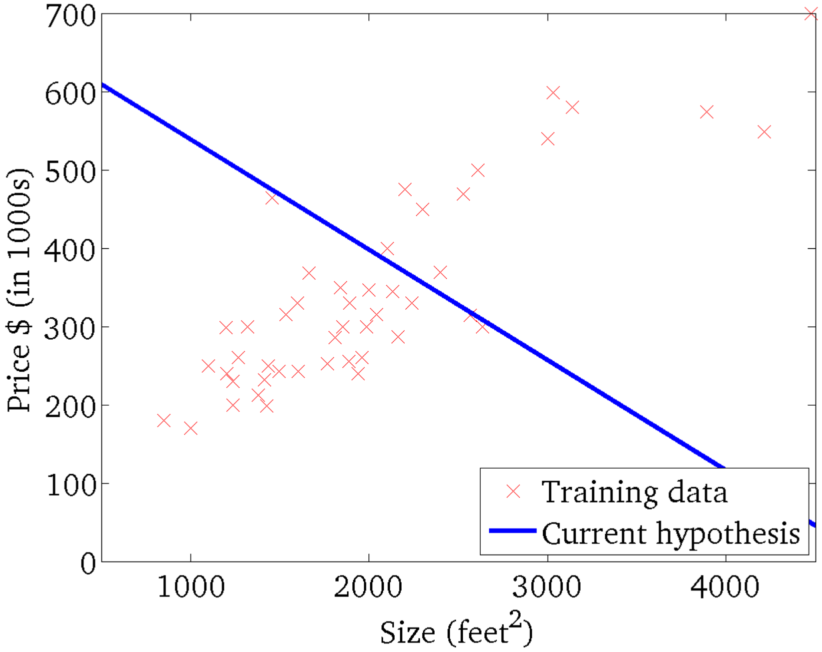


Reduce θ_0, reduce θ_1

Gradient Descent Iterations

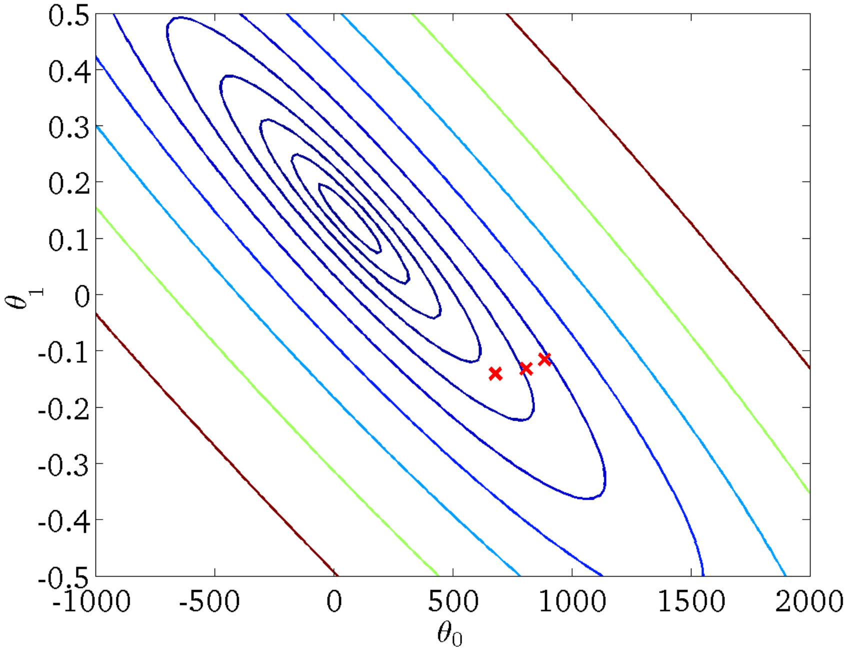
$h_\theta(x)$

(for fixed θ_0, θ_1 this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

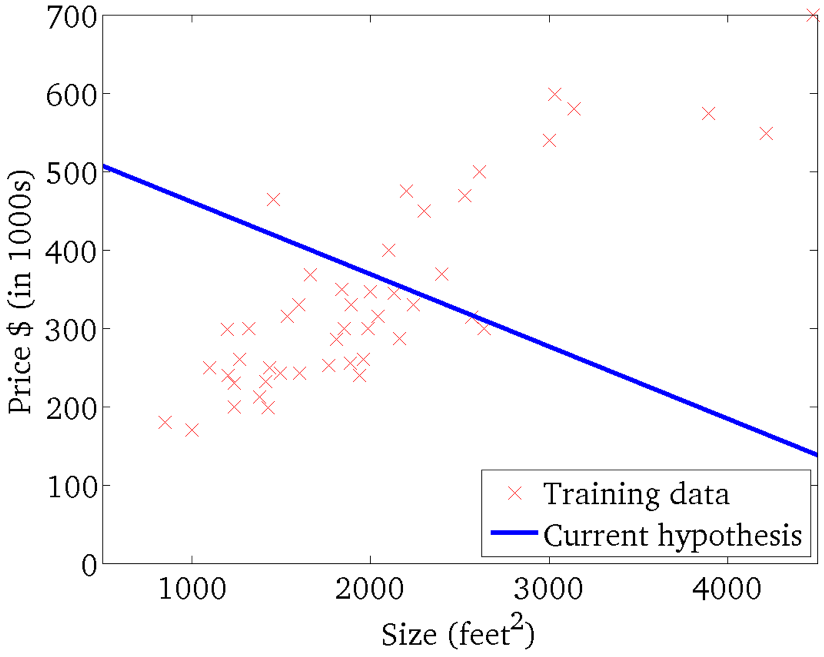


Reduce θ_0, increase θ_1

Gradient Descent Iterations

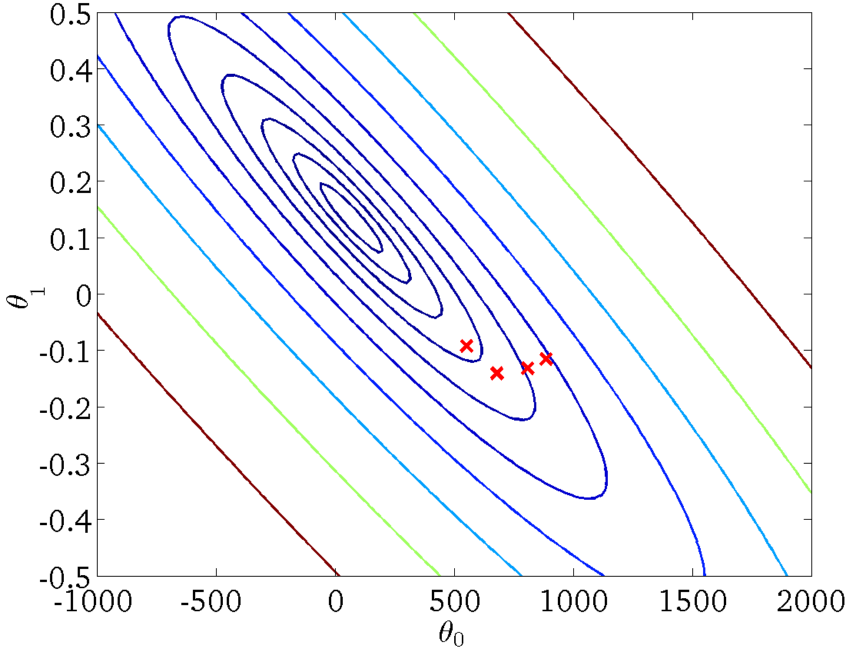
$h_\theta(x)$

(for fixed θ_0, θ_1 this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

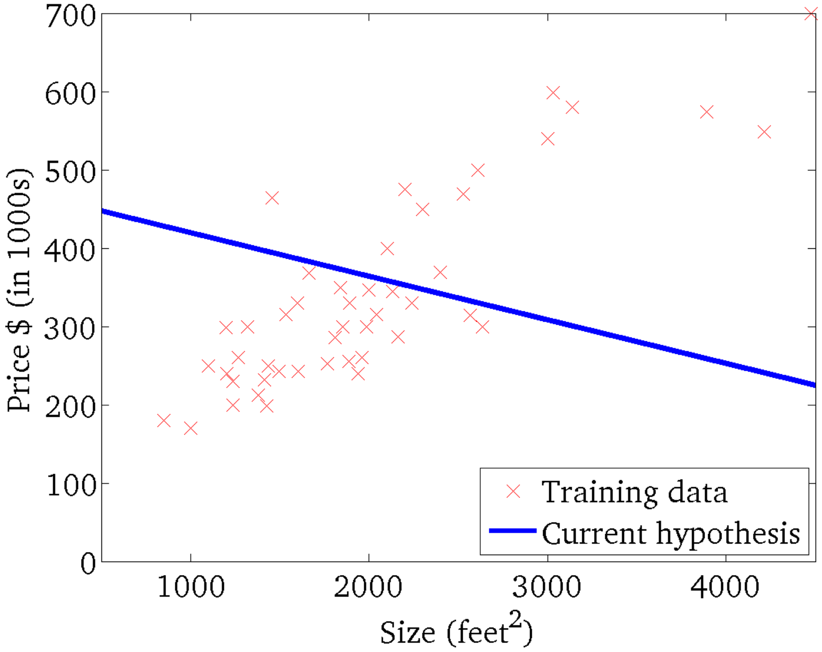


Reduce θ_0, increase θ_1

Gradient Descent Iterations

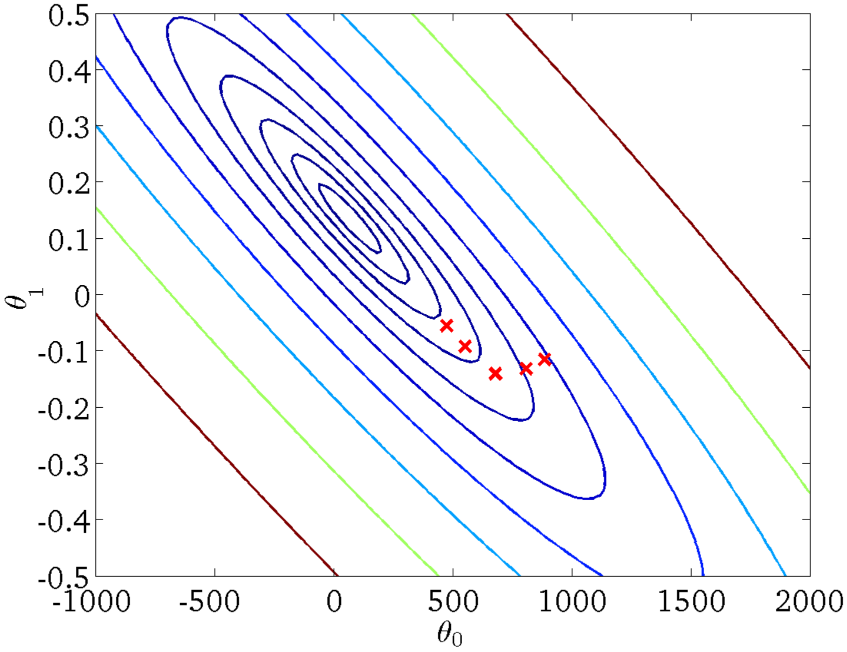
$h_\theta(x)$

(for fixed θ_0, θ_1 this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



Reduce θ_0, increase θ_1

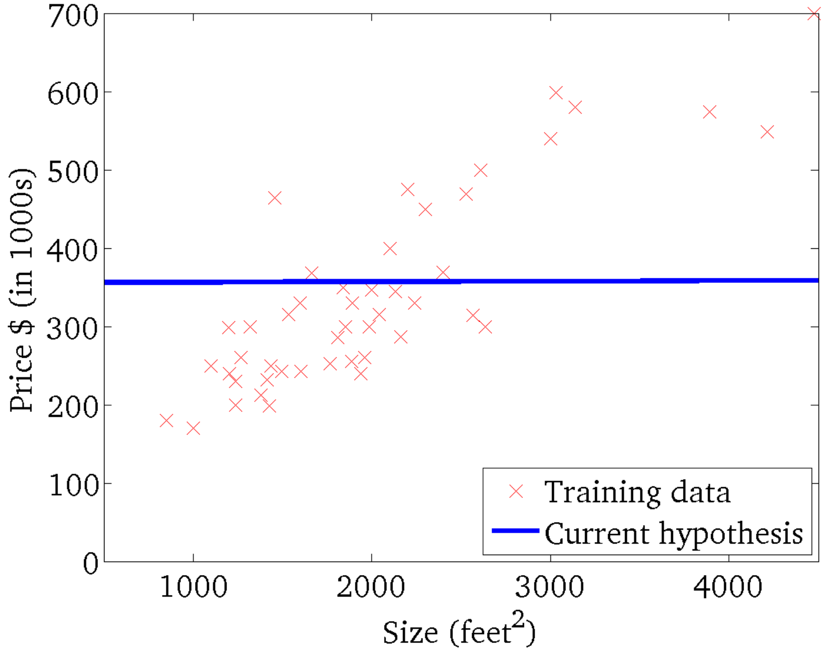
Gradient Descent Iterations

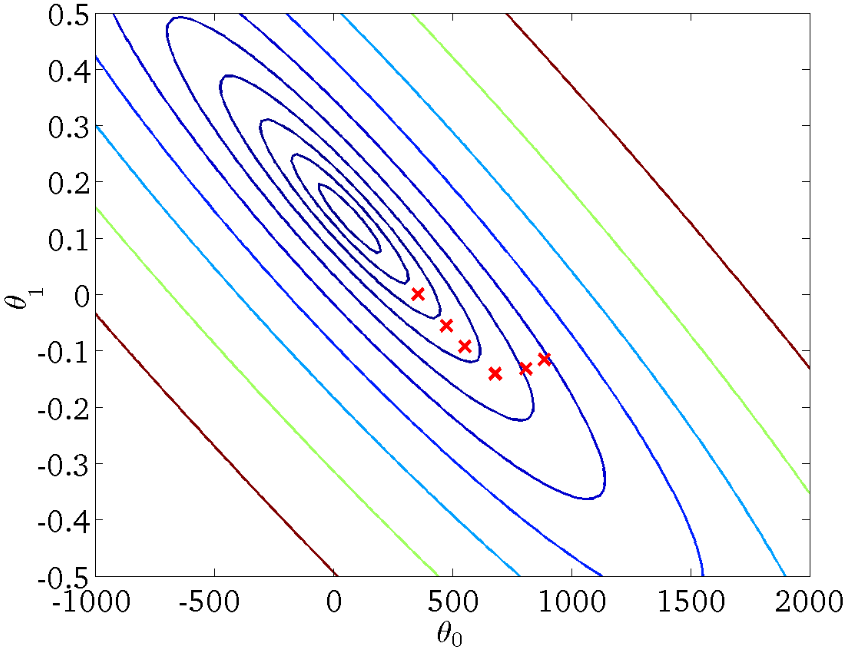
$h_\theta(x)$

(for fixed θ_0, θ_1 this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



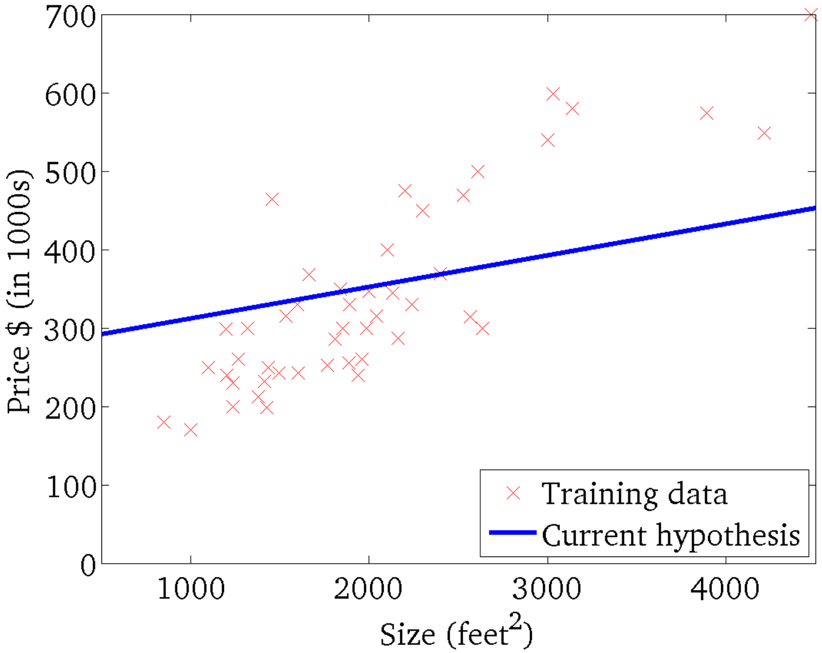


Reduce θ_0, increase θ_1

Gradient Descent Iterations

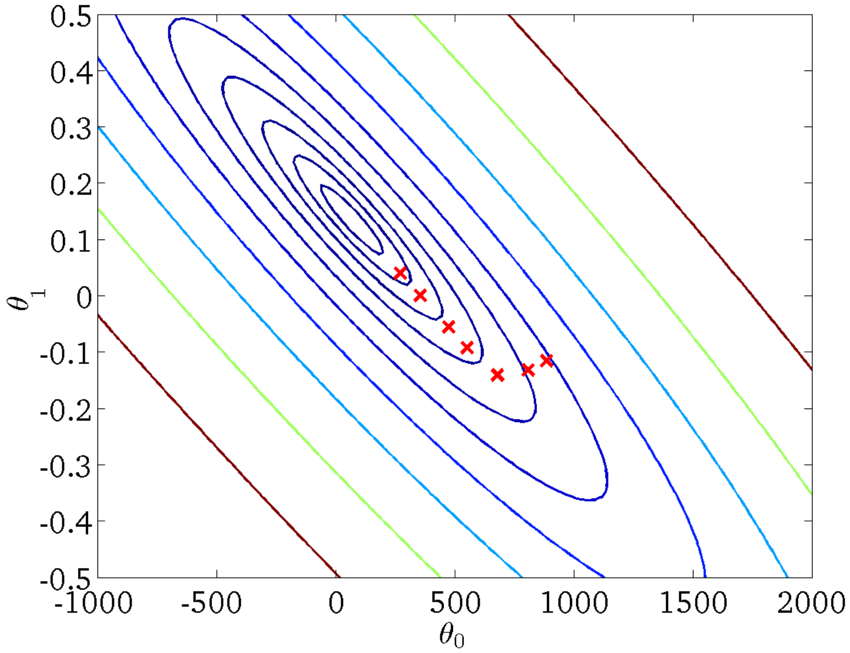
$h_\theta(x)$

(for fixed θ_0, θ_1 this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



Reduce θ_0, increase θ_1

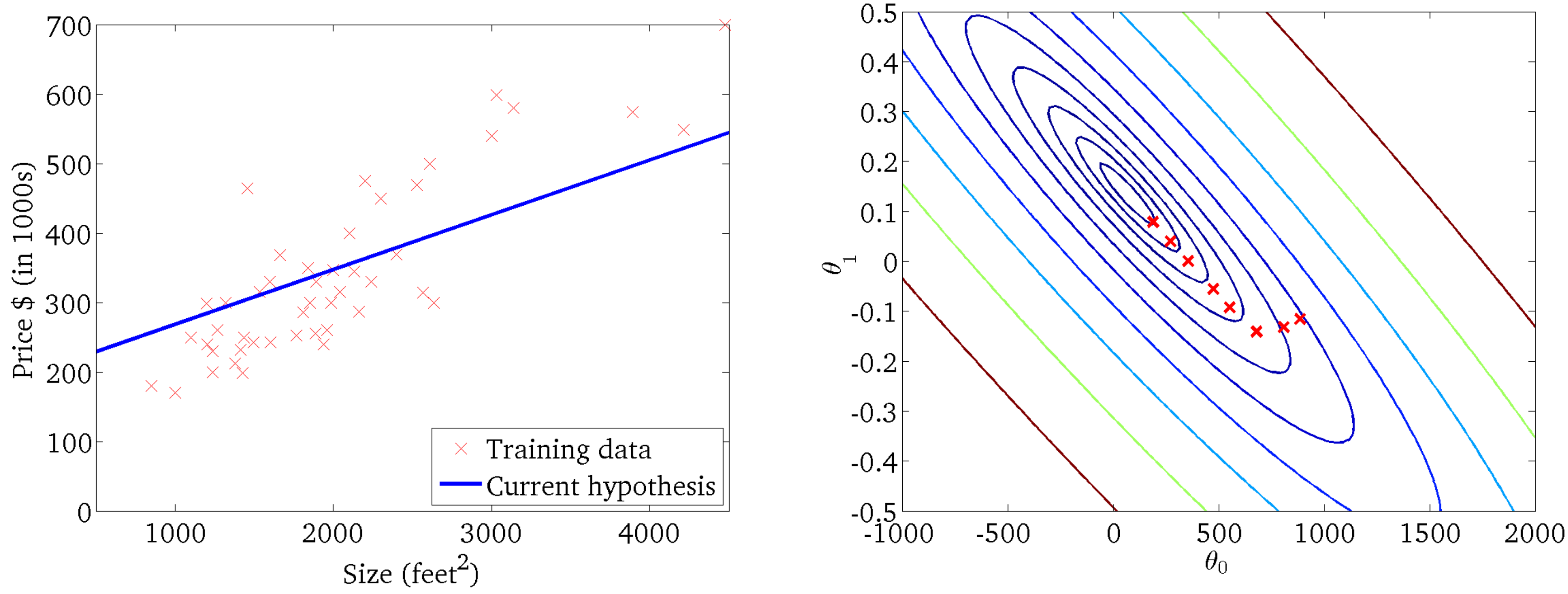
Gradient Descent Iterations

$h_\theta(x)$

(for fixed θ_0, θ_1 this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

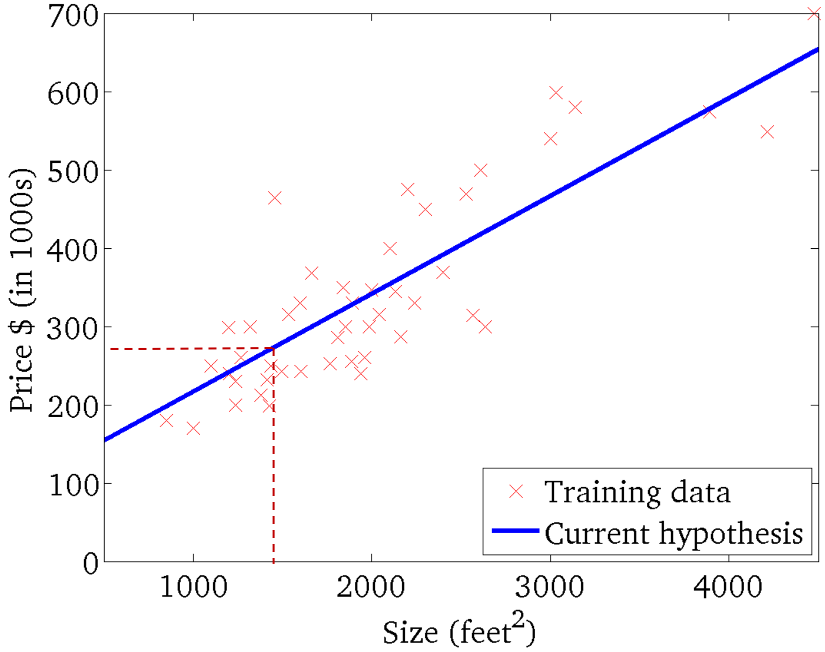


Reduce θ_0, increase θ_1

Gradient Descent Iterations

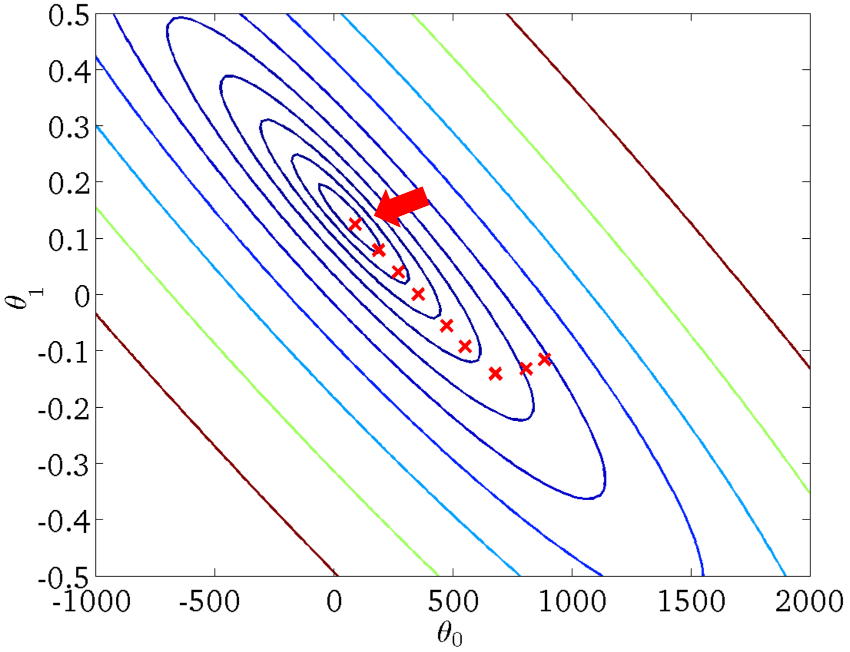
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



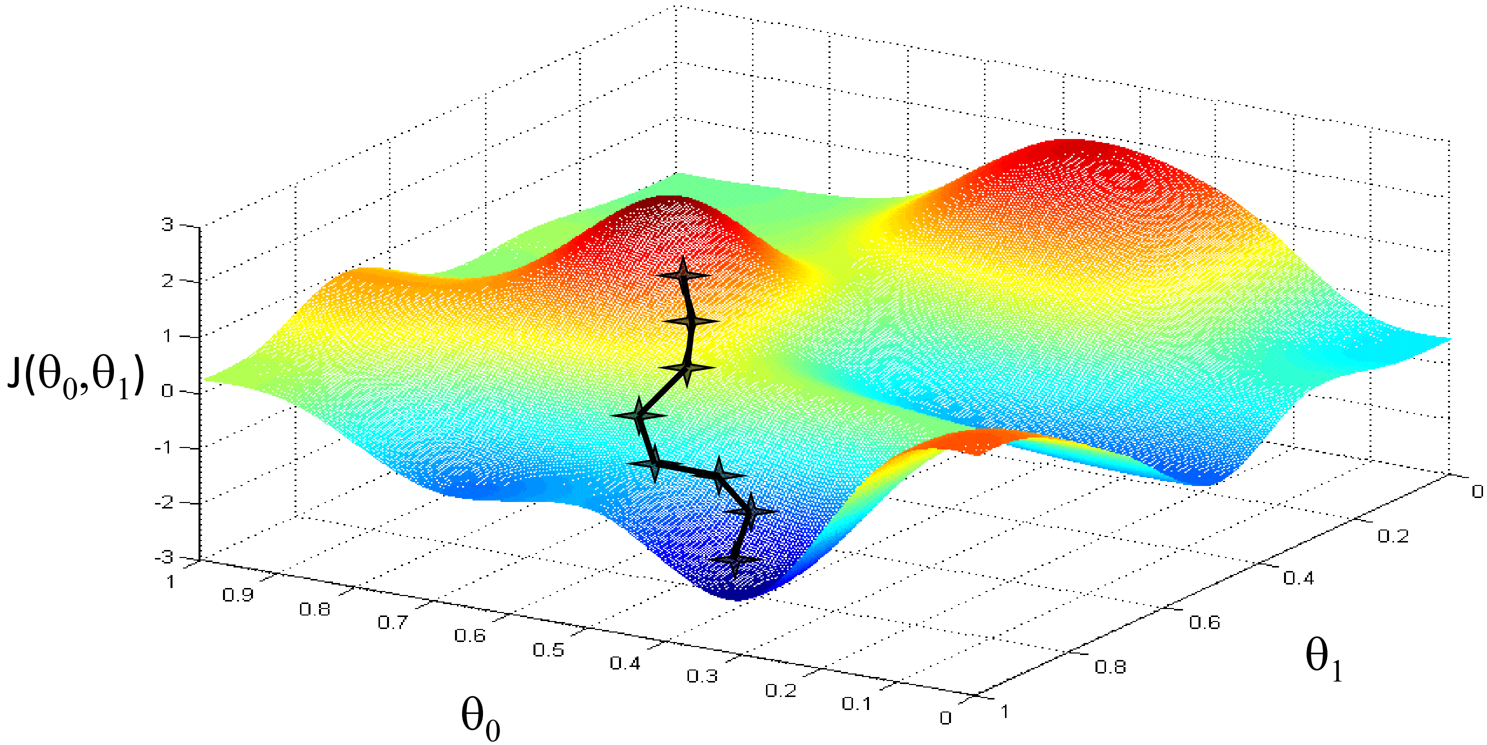
$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

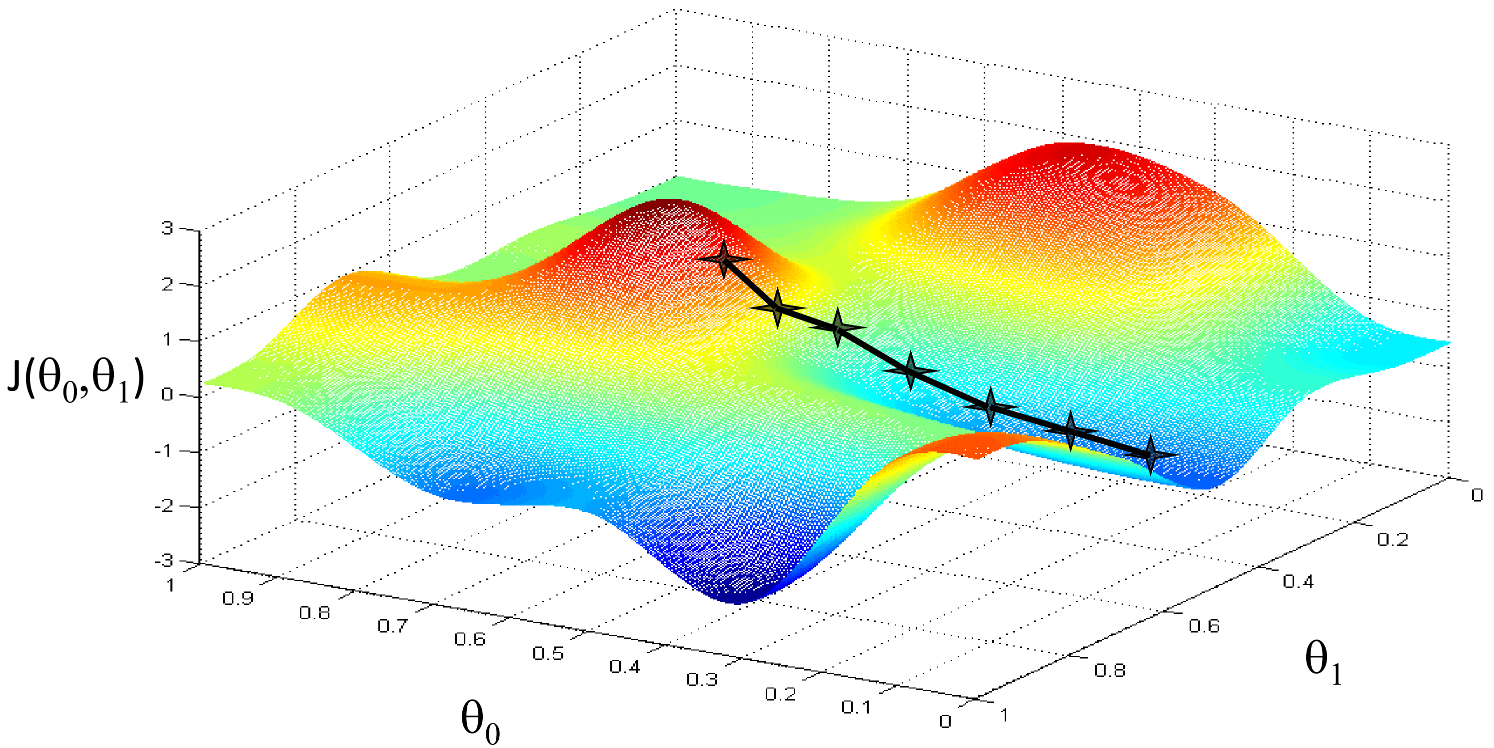


Stable

Local Minima Problem



Local Minima Problem



Starting point matters! Try with multiple starting points and take lowest cost.

Questions?