Linear Regression

Machine Learning

(Largely based on slides from Andrew Ng)

Prof. Sandra Avila
Institute of Computing (IC/Unicamp)

MC886, August 14, 2019

House Price Prediction

$ 70 000

https://www.youtube.com/watch?v=IpGxLWOIZy4

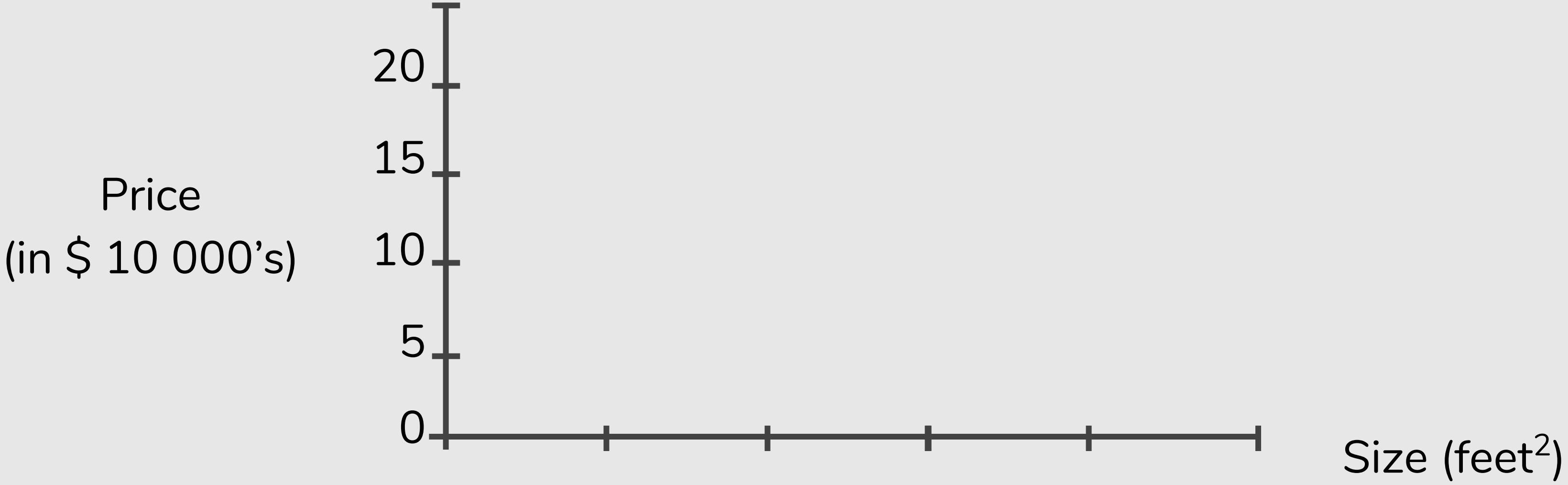
House Price Prediction

$ 160 000

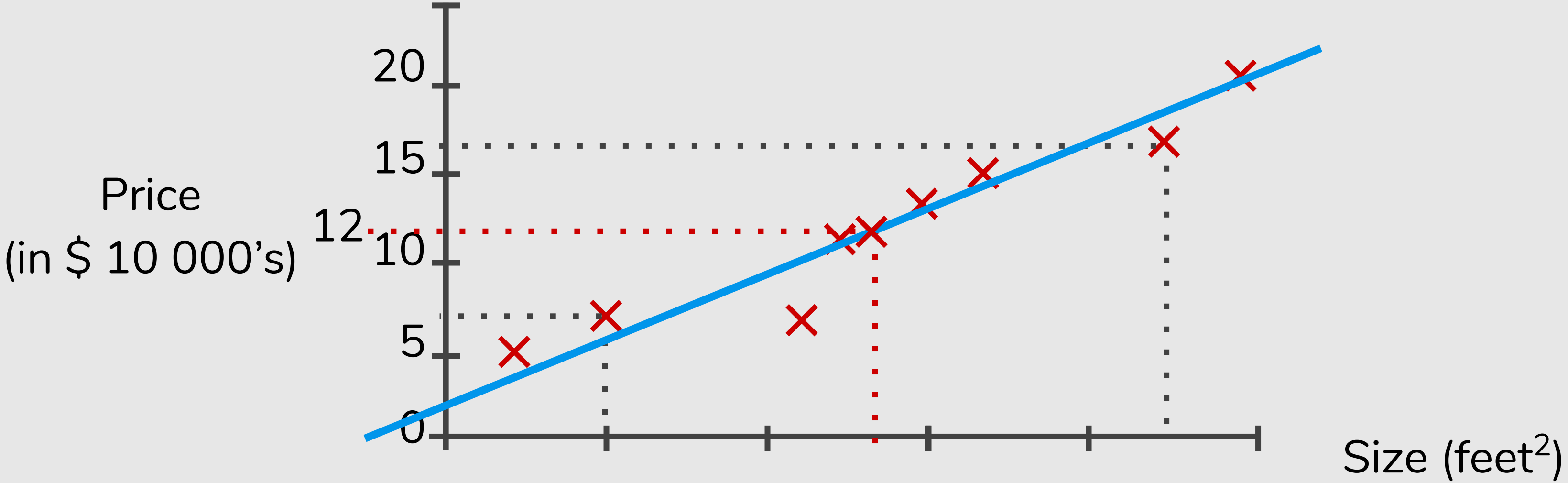
House Price Prediction

???

House Price Prediction



Linear Regression



Today’s Agenda

- **Linear Regression with One Variable**
    - **Model Representation**
    - **Cost Function**
    - **Gradient Descent**
- **Linear Regression with Multiple Variables**
    - **Gradient Descent for Multiple Variables**
    - Feature Scaling
    - Learning Rate
    - Features and Polynomial Regression
    - Normal Equation

Model Representation

kaggle | Search kaggle | Competitions | Datasets | Kernels | Discussion | Jobs | ••• | Sign In

House Sales in King County, USA

Predict house price using regression

108

harlfoxem • last updated a year ago

Overview | Kernels | Discussion | Activity | Download (778 KB) | New Kernel

Tags: finance, home, small, featured

Kernels

Feature Ranking w Random... run 2 days ago	55 votes
Step by Step House Price Pre... run 7 months ago	41 votes
House_Price_Prediction_Part_1 run a year ago	29 votes

Discussion

Variable explanation 6 days ago	15 replies
King County Geoclustering... 9 days ago	7 replies
RF, RFE, linear models|特征... 10 days ago	3 replies

Top Contributors

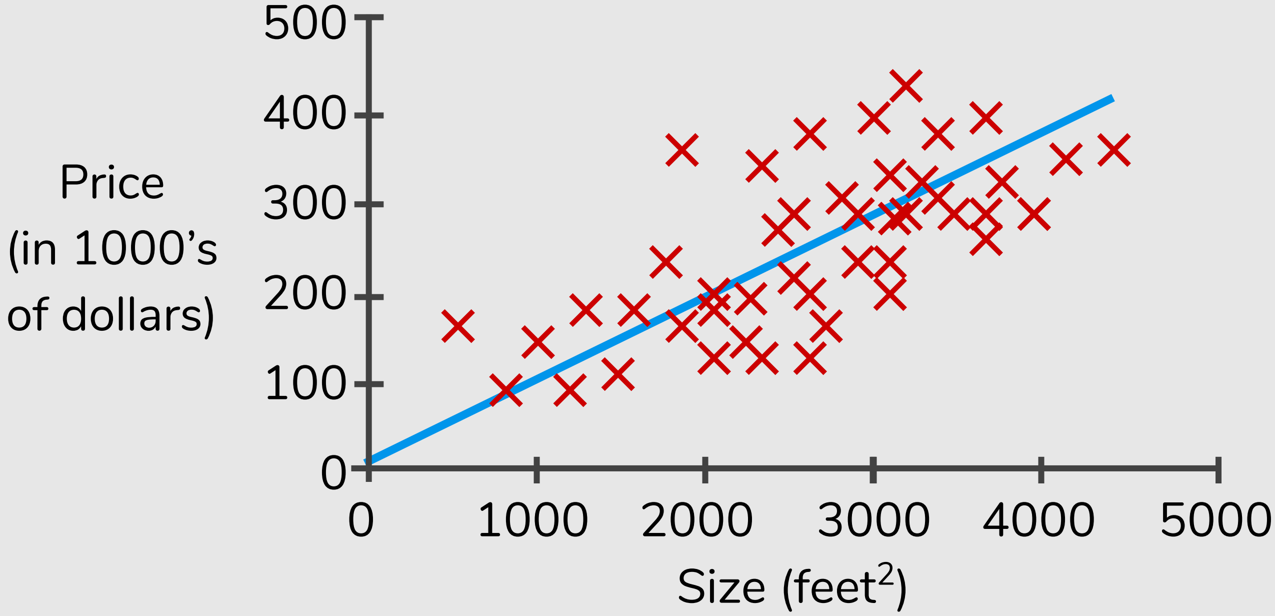
harlfoxem	1st
Anisotropic	2nd
ArmanUygur	3rd

Recent Activity

Jois Leonida Lobo	Ran version 10 of kernel **HousePricePrediction_SimpleLinearRegression**	13 hours ago
Anisotropic	Ran version 41 of kernel **Feature Ranking w RandomForest, RFE, linear models**	2 days ago
DavidTan	Commented on dataset discussion **Variable explanation**	6 days ago
Harsh Tyagi	Ran version 3 of kernel **King County's Housing Market, various techniques.**	8 days ago

https://www.kaggle.com/harlfoxem/housesalesprediction

Housing Prices



Supervised Learning

Given the "right answer" for each example in the data.

Regression Problem

Predict real-valued output

Training set of housing prices

Size in feet^2 (x)	**Price ($) in 1000's ($y$)**
2104	460
1416	232
1534	315
852	178
...	...

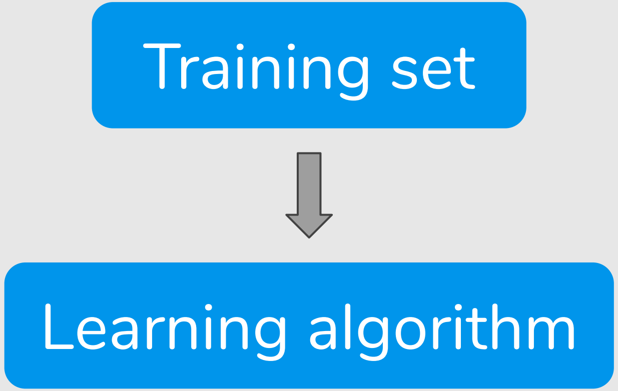
Notation:

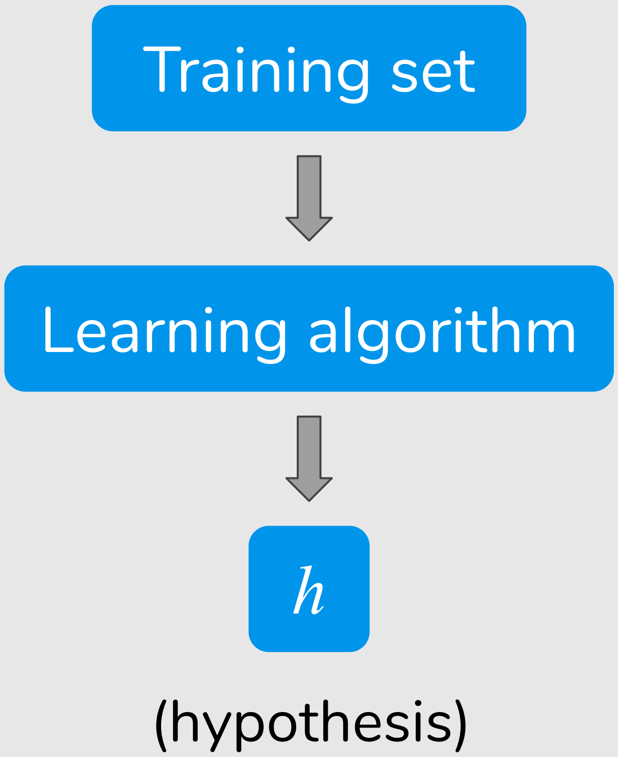
m = Number of training examples

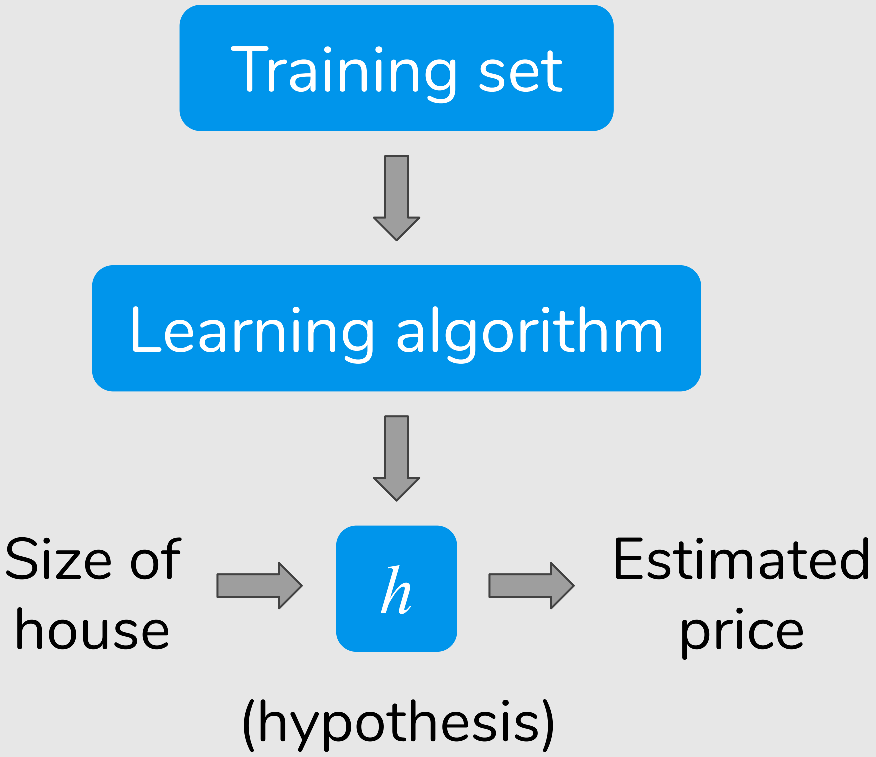
x's = "input" variable / features

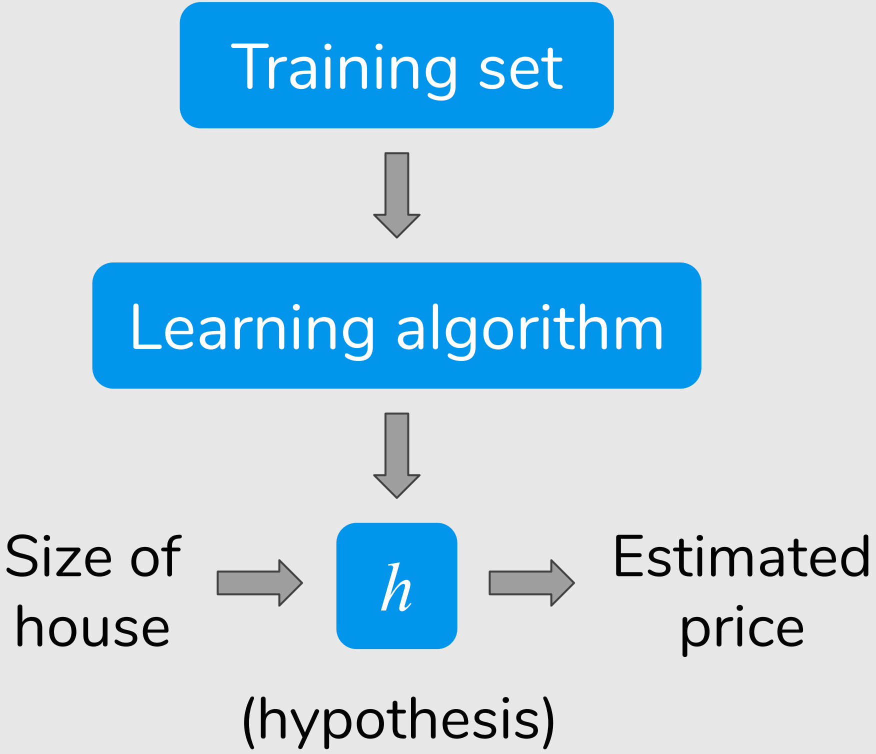
y's = "output" variable / "target" variable

Training set


Training set
Learning algorithm

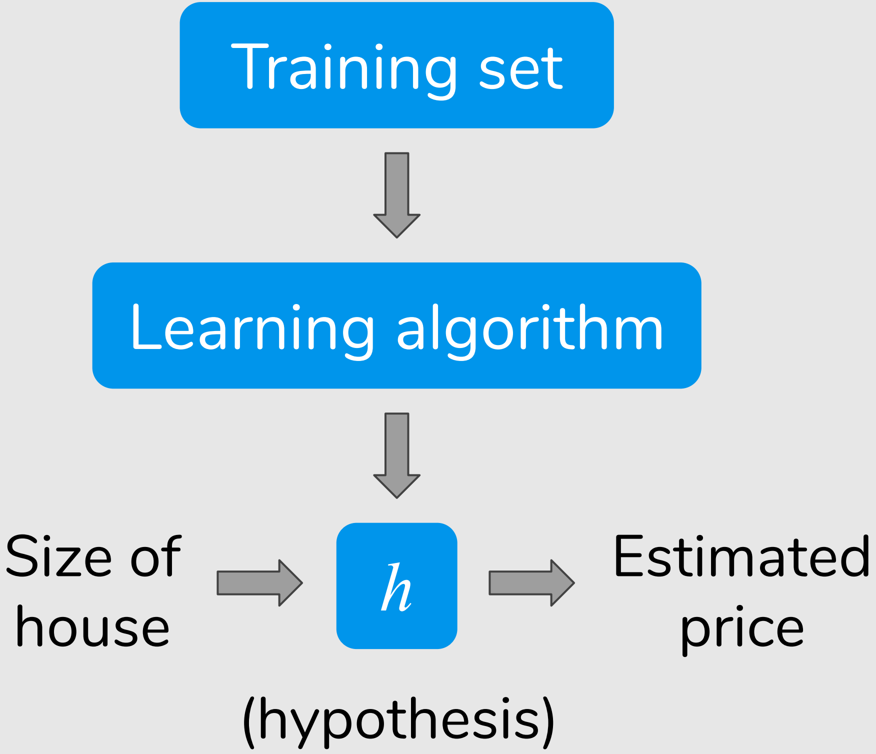

Training set
Learning algorithm
h
(hypothesis)


Training set
Learning algorithm
Size of house
h
Estimated price
(hypothesis)



h maps x's to y's

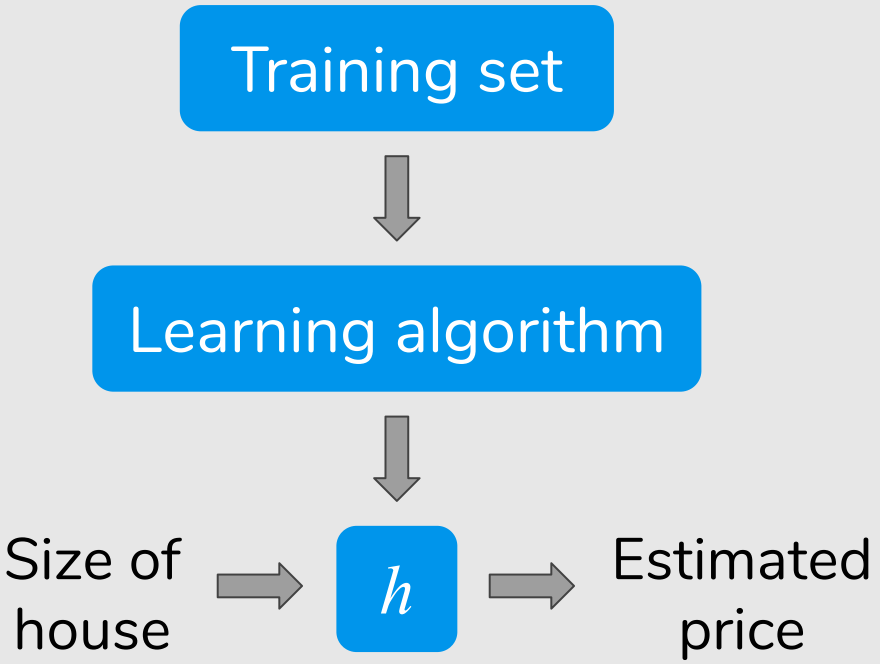
How do we represent h ?



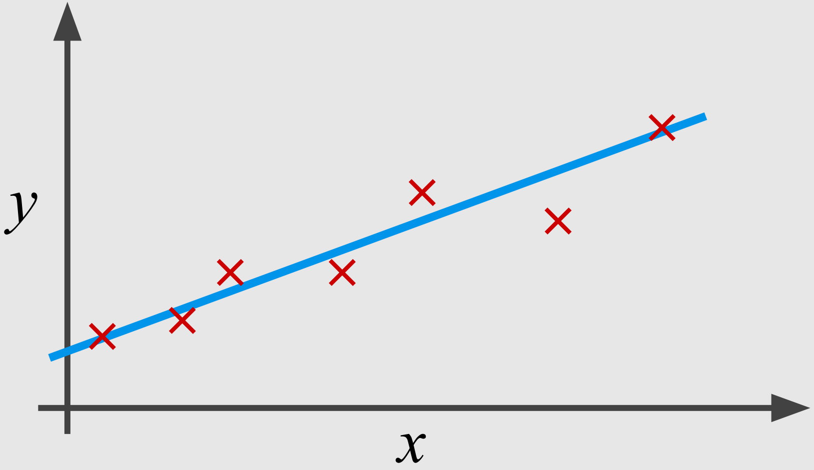
h maps x's to y's

How do we represent h ?

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

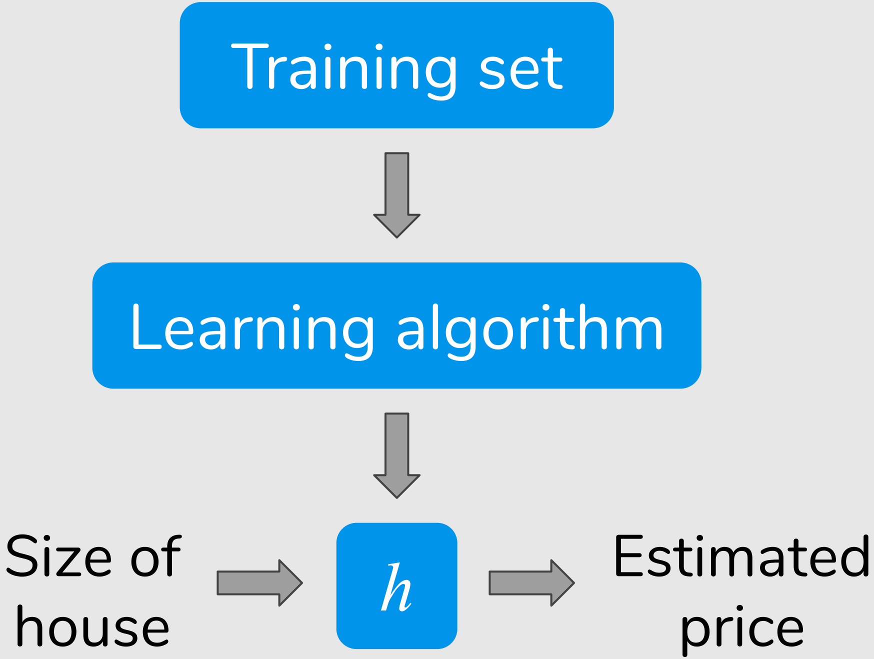


(hypothesis)



h maps x's to y's

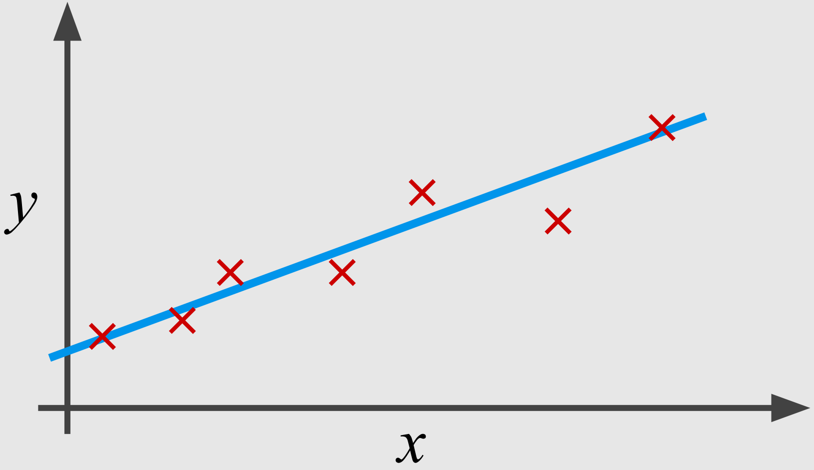
How do we represent h ?



(hypothesis)

h maps x's to y's

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



Linear regression with one variable.

Univariate linear regression.

Cost Function

Training Set

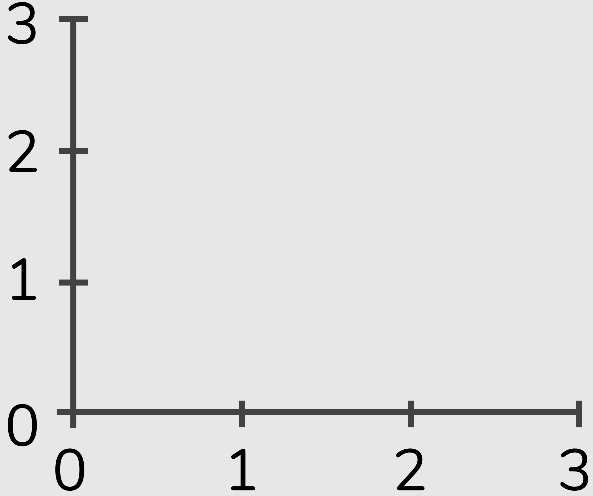
Size in feet^2 (x)	Price (\$) in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	…

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

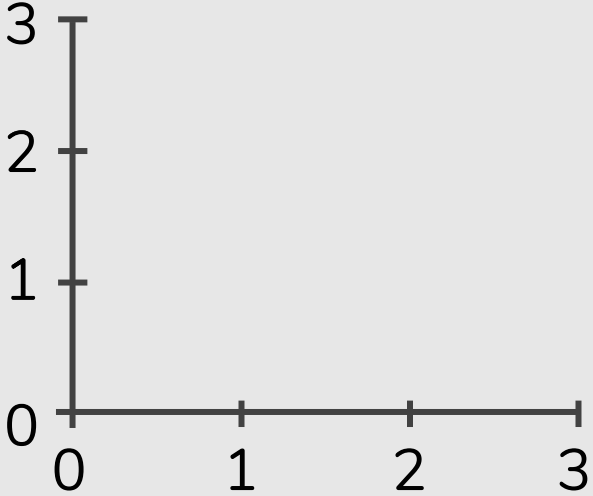
θi's: Parameters

How to choose θi's ?

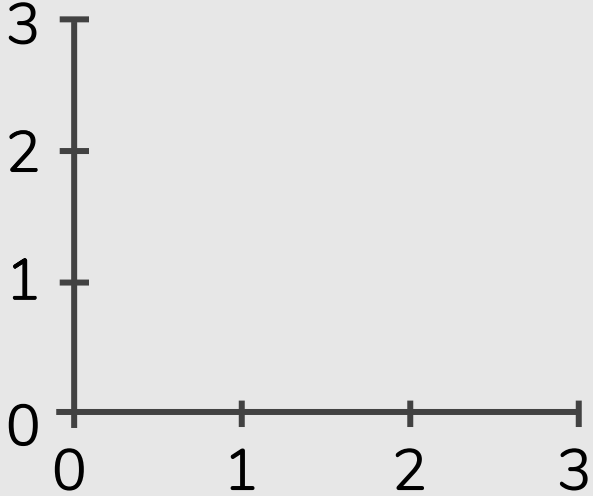
$$h_\theta(x) = \theta_0 + \theta_1 x$$



$\theta_0 = 1.5$
$\theta_1 = 0$

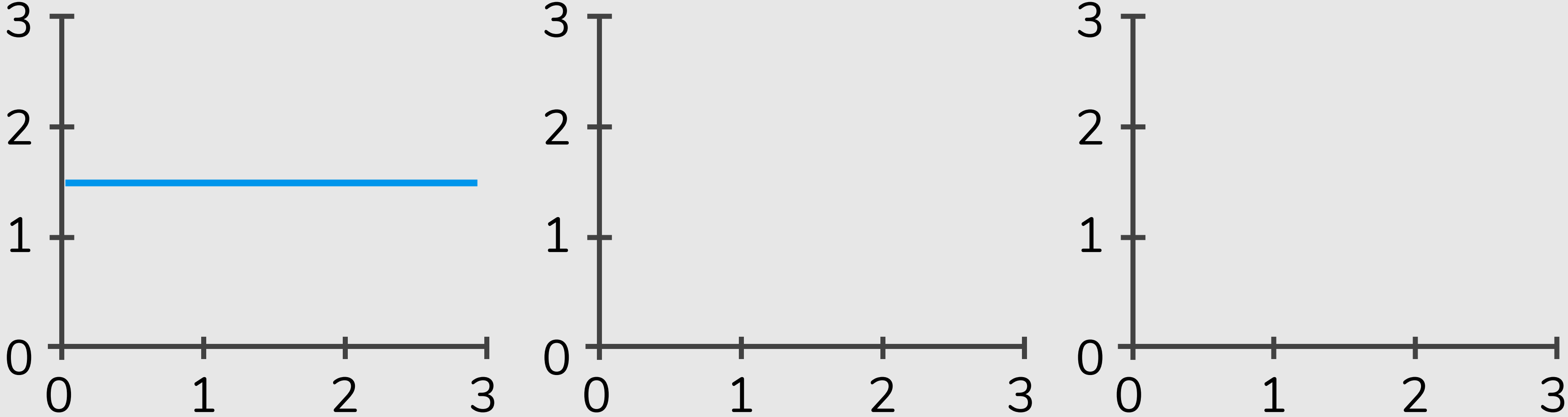


$\theta_0 = 0$
$\theta_1 = 0.5$



$\theta_0 = 1$
$\theta_1 = 0.5$

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

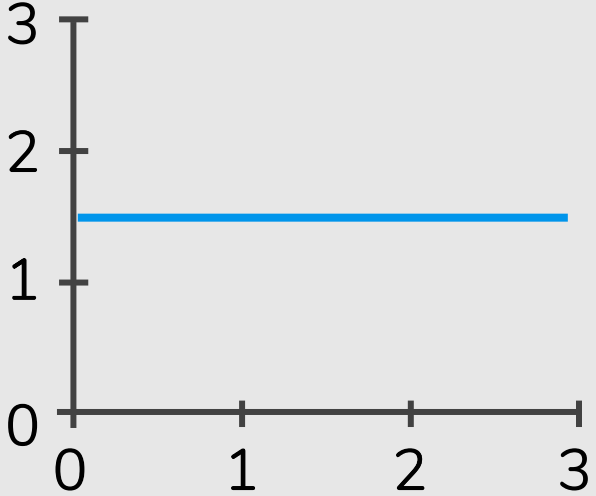


$\theta_0 = 1.5$
$\theta_1 = 0$

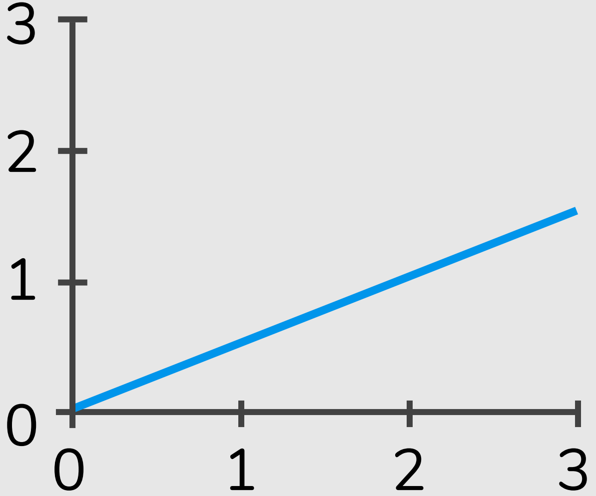
$\theta_0 = 0$
$\theta_1 = 0.5$

$\theta_0 = 1$
$\theta_1 = 0.5$

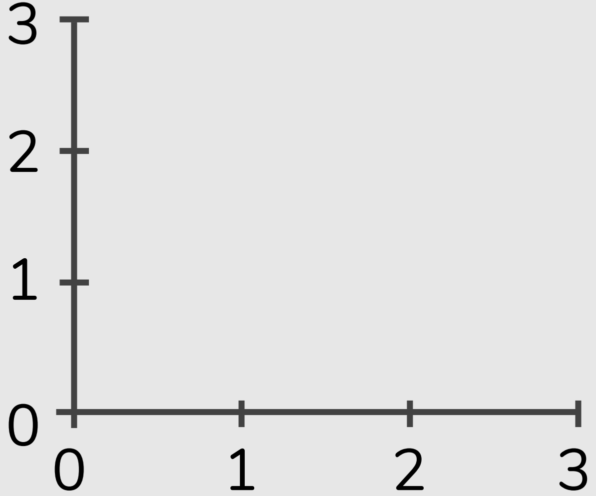
$$h_\theta(x) = \theta_0 + \theta_1 x$$



$\theta_0 = 1.5$
$\theta_1 = 0$

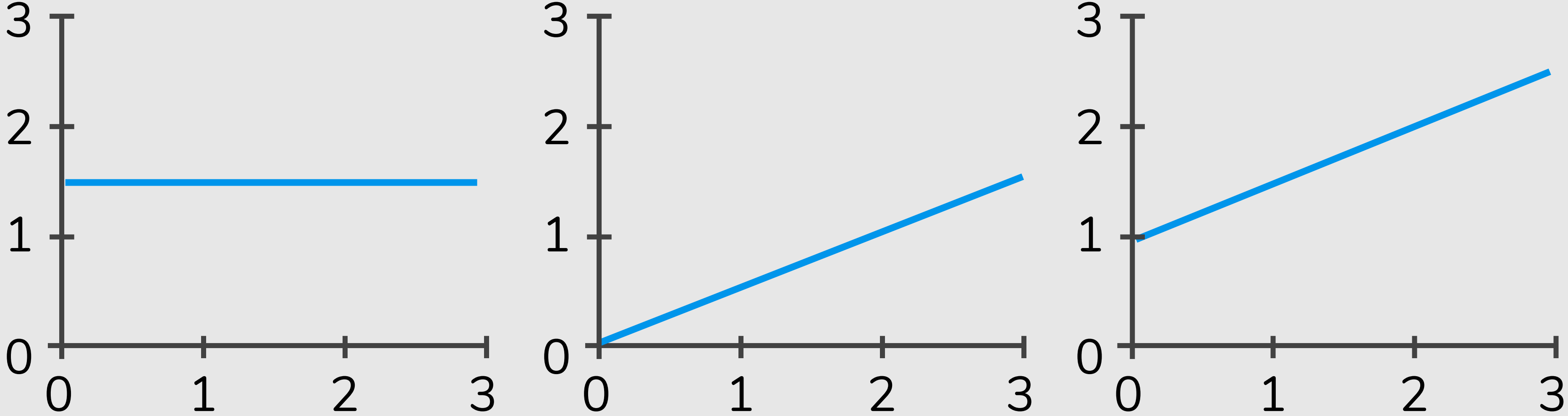


$\theta_0 = 0$
$\theta_1 = 0.5$



$\theta_0 = 1$
$\theta_1 = 0.5$

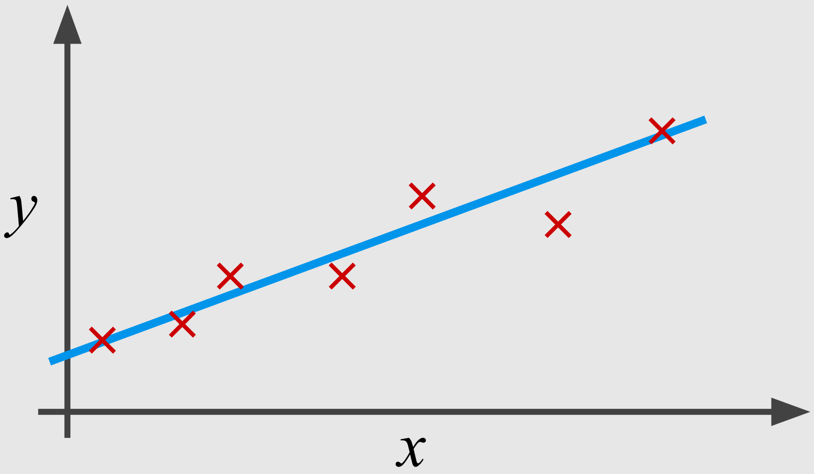
$$h_\theta(x) = \theta_0 + \theta_1 x$$



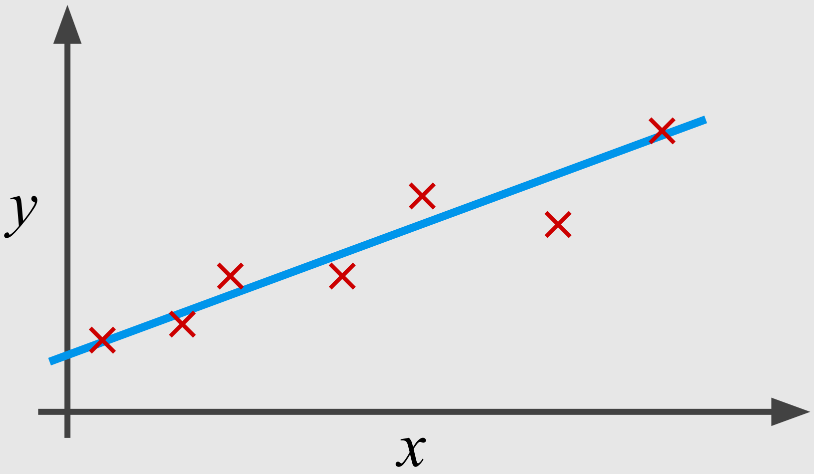
$\theta_0 = 1.5$
$\theta_1 = 0$

$\theta_0 = 0$
$\theta_1 = 0.5$

$\theta_0 = 1$
$\theta_1 = 0.5$

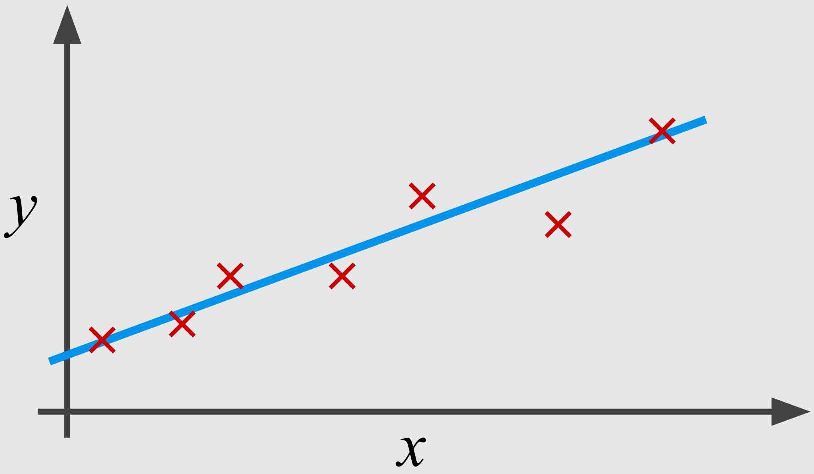


Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ close to y for our training examples (x,y)



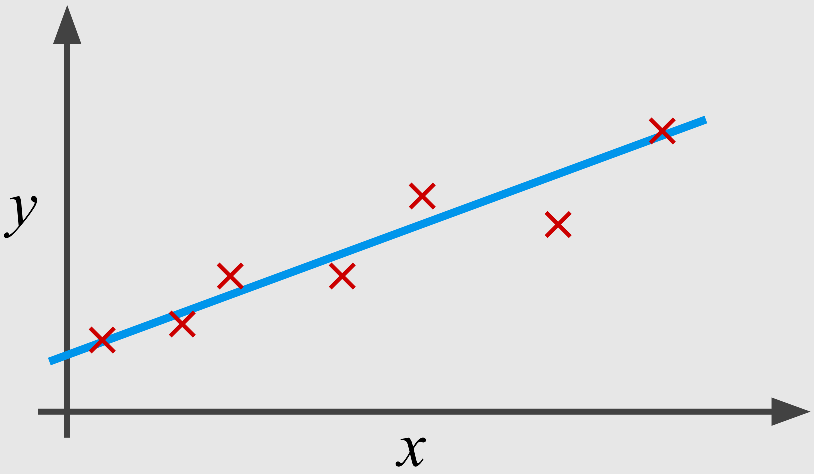
$$\underset{\theta_0, \theta_1}{\text{minimize}}$$

Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ close to y for our training examples (x,y)



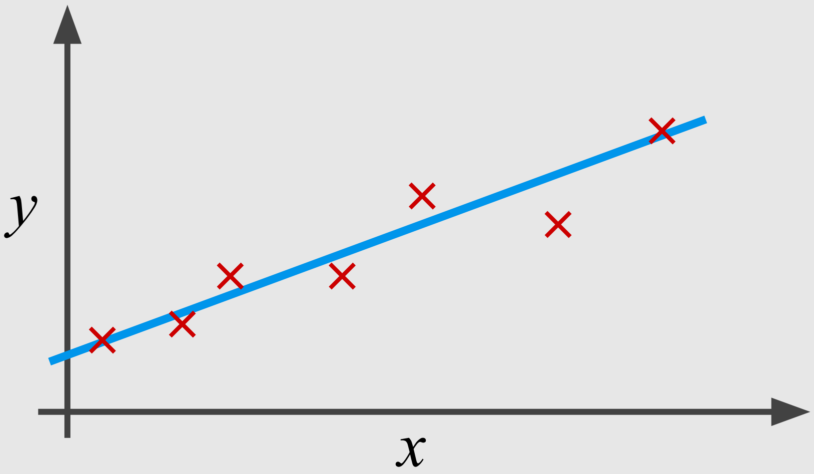
$$\underset{\theta_0,\theta_1}{\text{minimize}} \quad (h_\theta(x\ ) - y\ )^2$$

Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ close to y for our training examples (x,y)



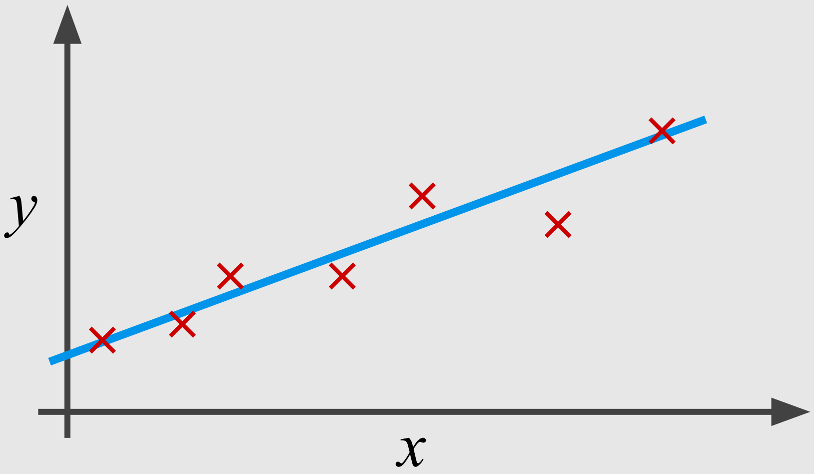
$$\underset{\theta_0,\theta_1}{\text{minimize}} \quad \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ close to y for our training examples (x,y)



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

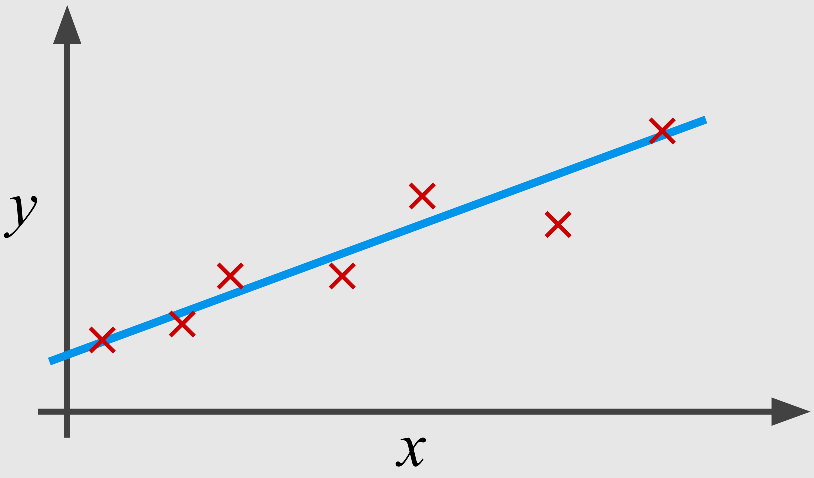
Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ close to y for our training examples (x,y)



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

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

Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ close to y for our training examples (x,y)



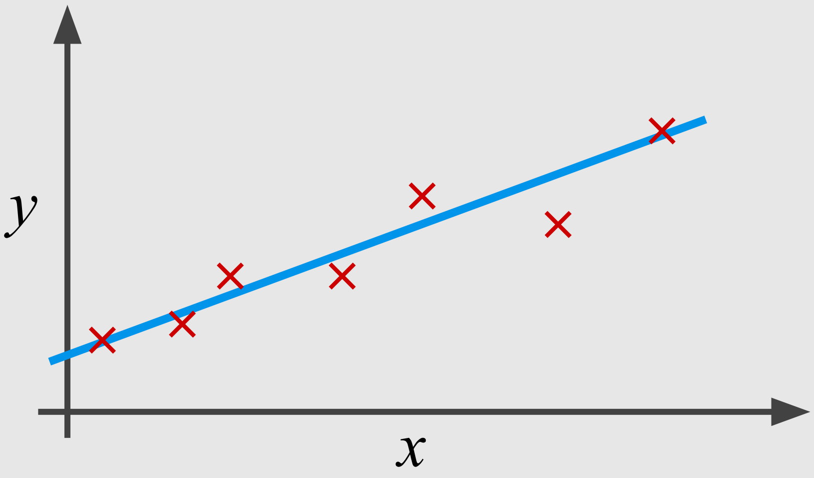
$$\underset{\theta_0,\theta_1}{\text{minimize}} \; \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

$$\Downarrow$$

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

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

Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ close to y for our training examples (x,y)



Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ close to y for our training examples (x,y)

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

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

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

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

Cost function
(Squared error function)

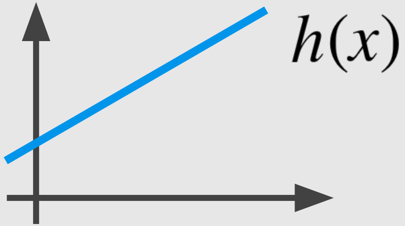
Cost Function Intuition I

Hypothesis:

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

Parameters:

$$\theta_0, \theta_1$$



Cost Function:

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

Goal:

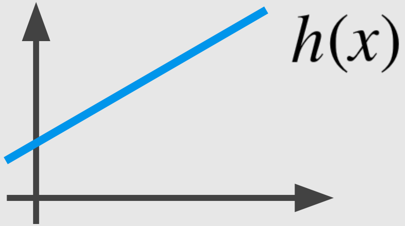
$$\underset{\theta_0, \theta_1}{\text{minimize}}\, J(\theta_0, \theta_1)$$

Hypothesis:

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

Parameters:

$$\theta_0, \theta_1$$



Cost Function:

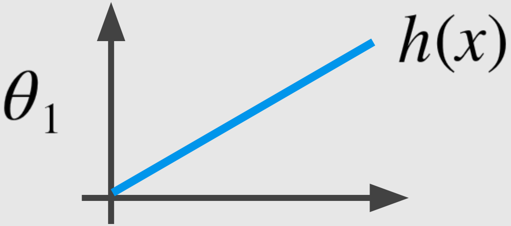
$$J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Goal:

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

<u>Simplified</u>

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



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

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

$h_\theta(x)$

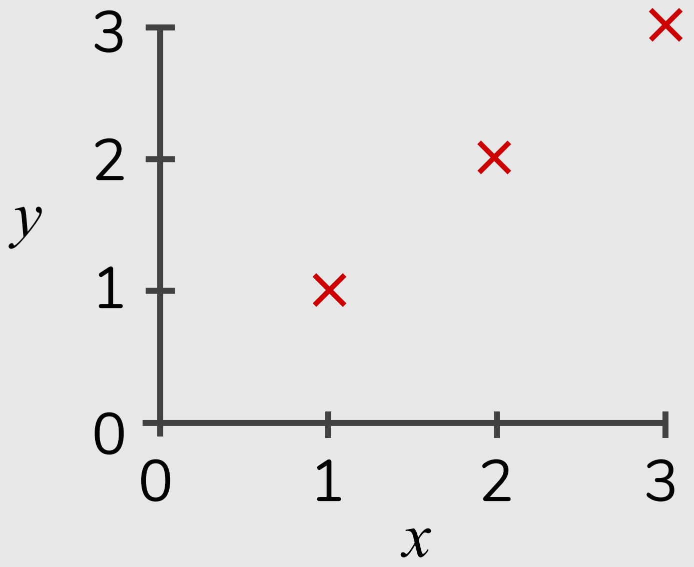
(for fixed θ_1, this is a function of x)

$J(\theta_1)$

(function of the parameters θ_1)

$h_\theta(x)$

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

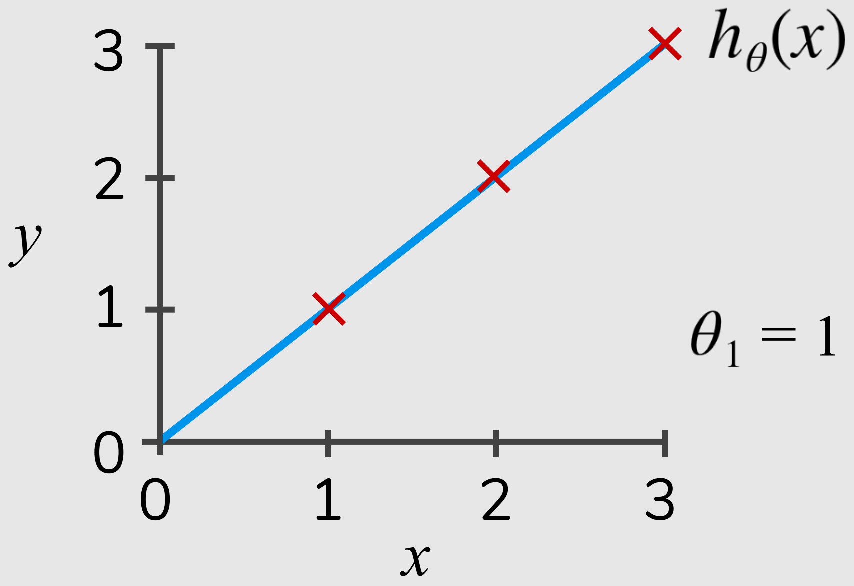


$J(\theta_1)$

(function of the parameters θ_1)

$h_\theta(x)$

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



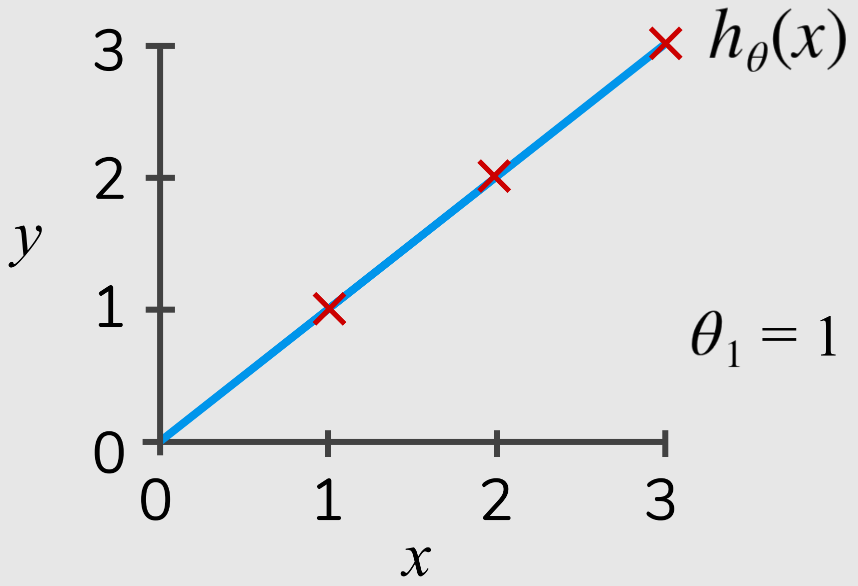
$J(\theta_1) = J(1) = ?$

$J(\theta_1)$

(function of the parameters θ_1)

$h_\theta(x)$

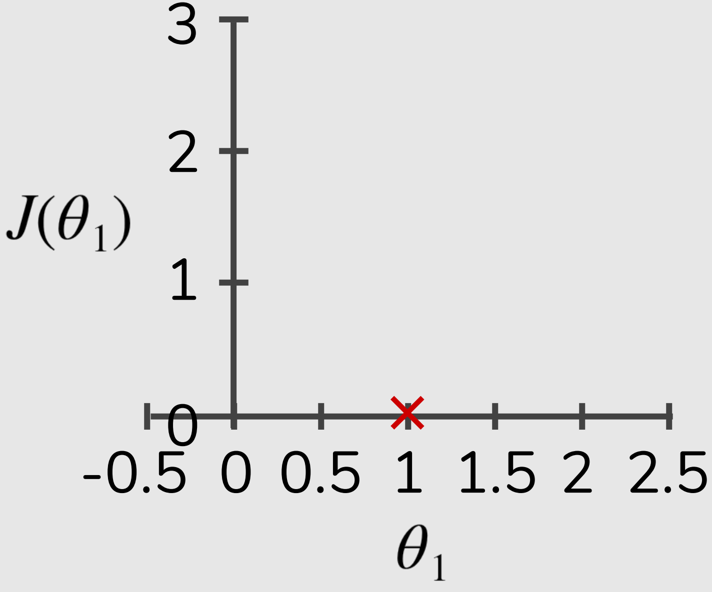
(for fixed θ_1, this is a function of x)



$J(\theta_1) = J(1) = 0$

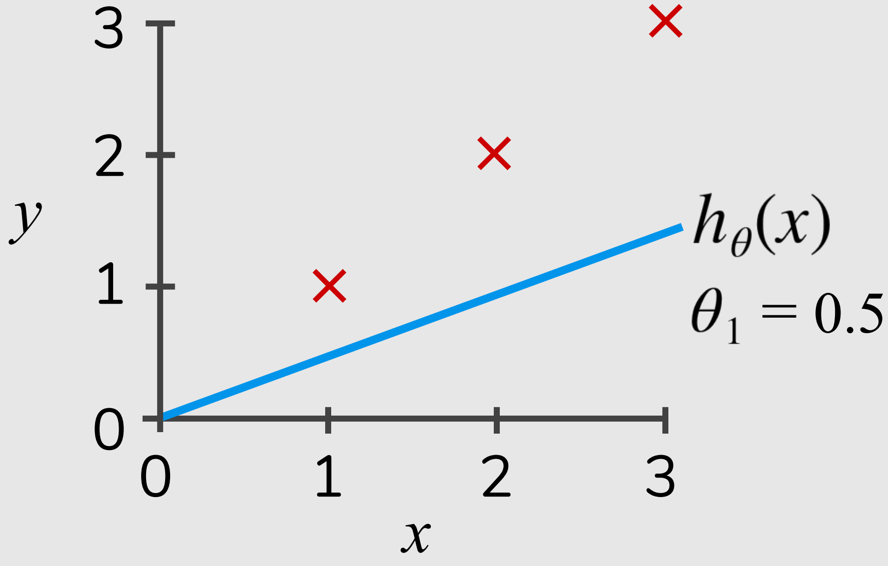
$J(\theta_1)$

(function of the parameters θ_1)



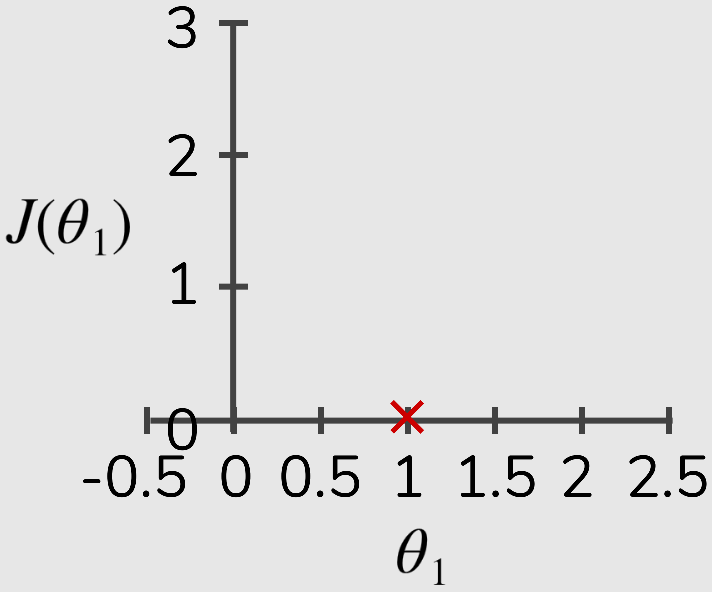
$h_\theta(x)$

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



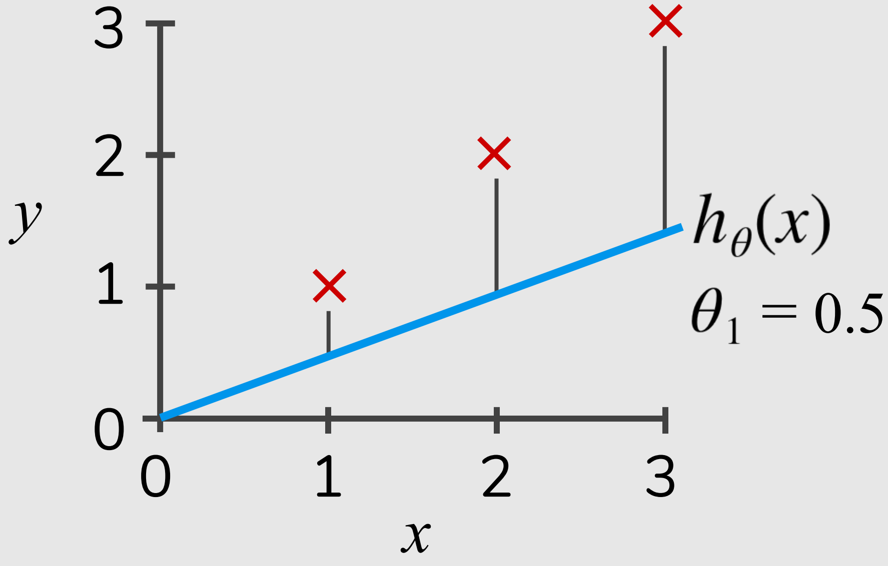
$J(\theta_1)$

(function of the parameters θ_1)



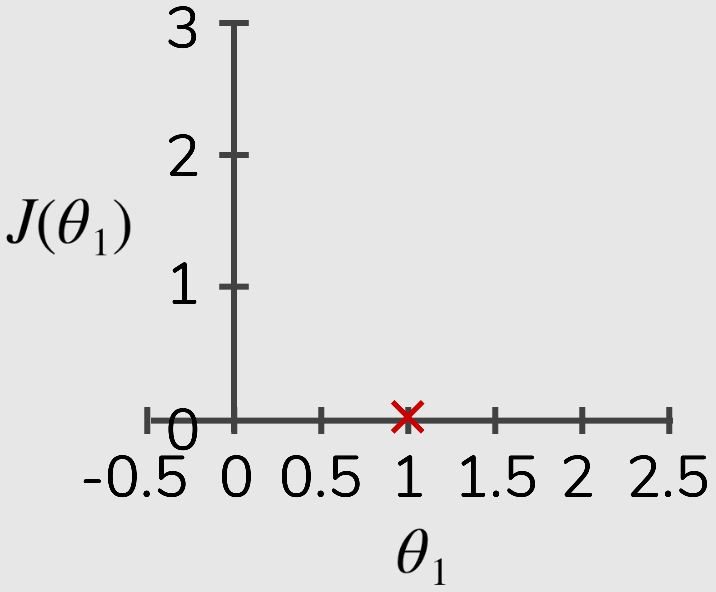
$h_\theta(x)$

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



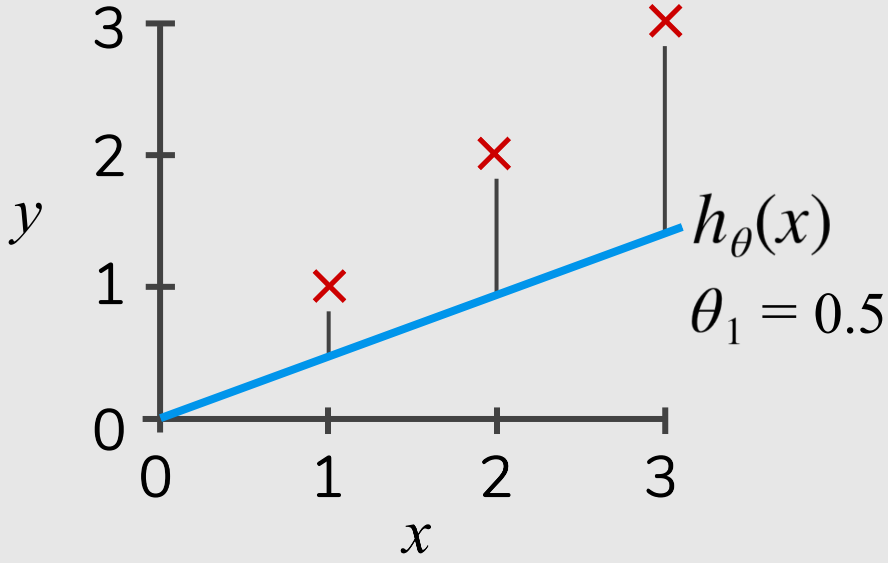
$J(\theta_1)$

(function of the parameters θ_1)



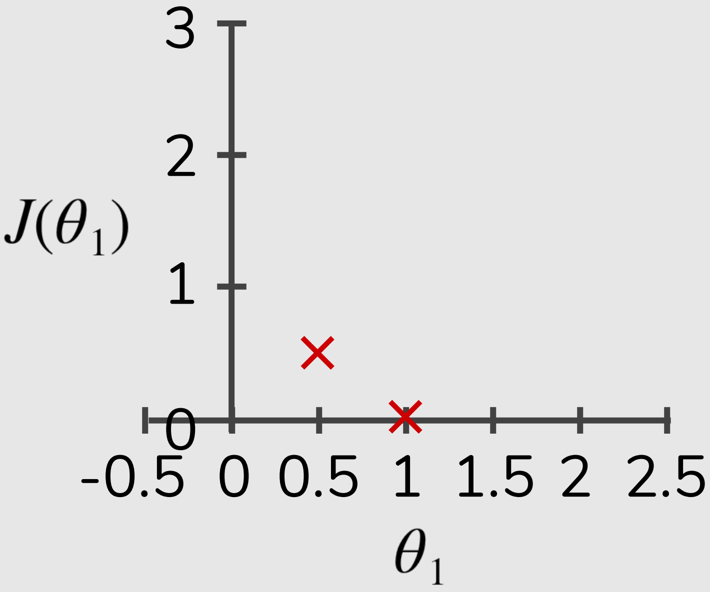
$h_\theta(x)$

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



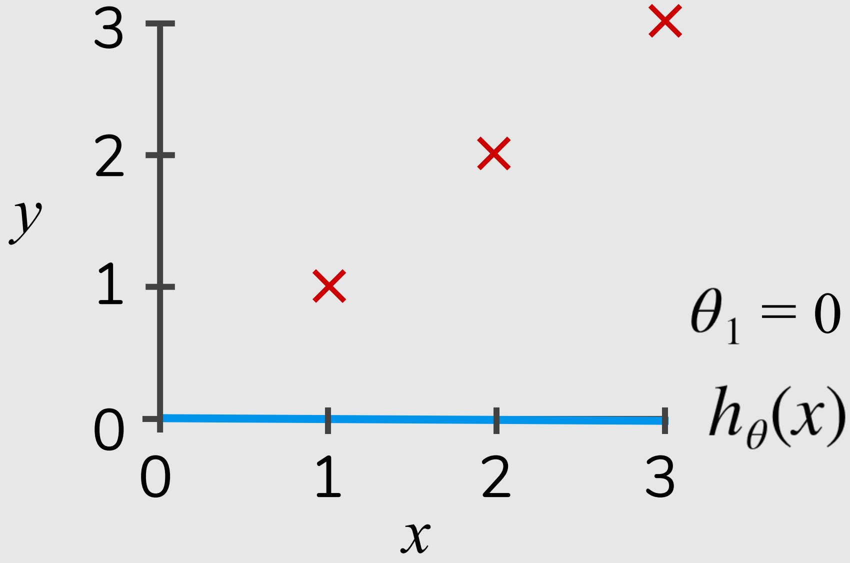
$J(\theta_1)$

(function of the parameters θ_1)



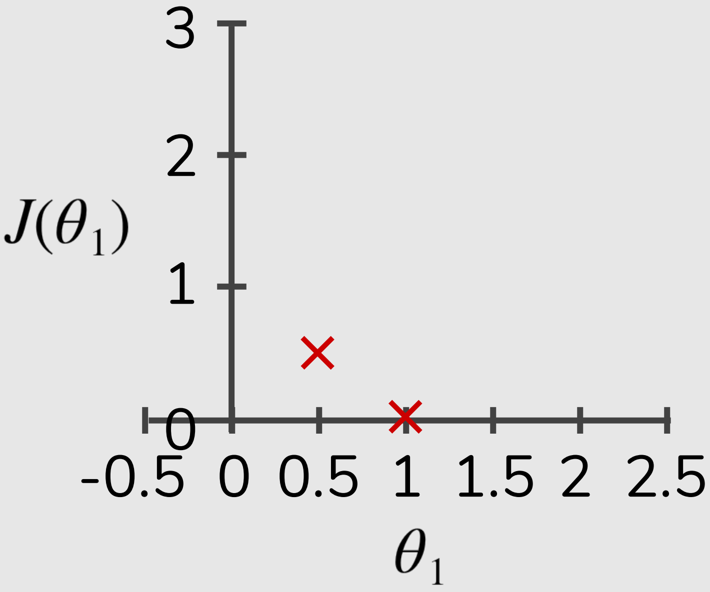
$h_\theta(x)$

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



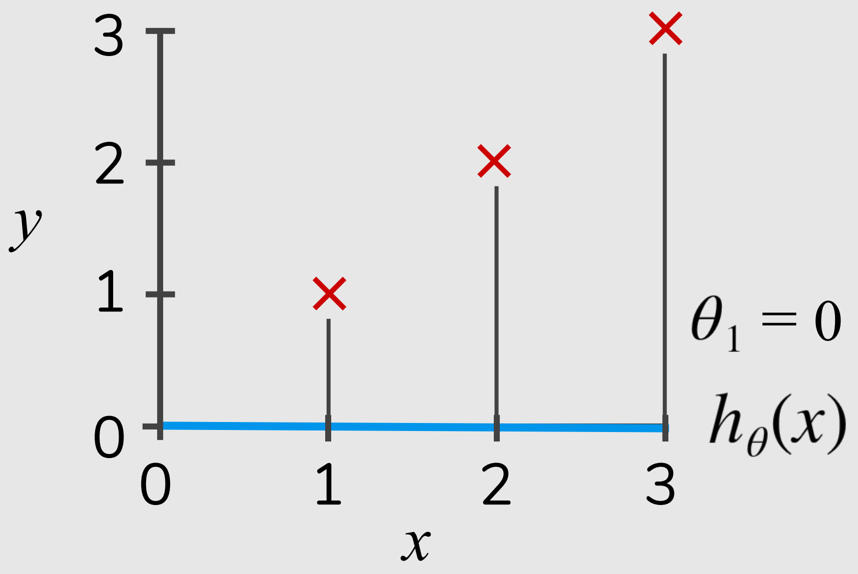
$J(\theta_1)$

(function of the parameters θ_1)



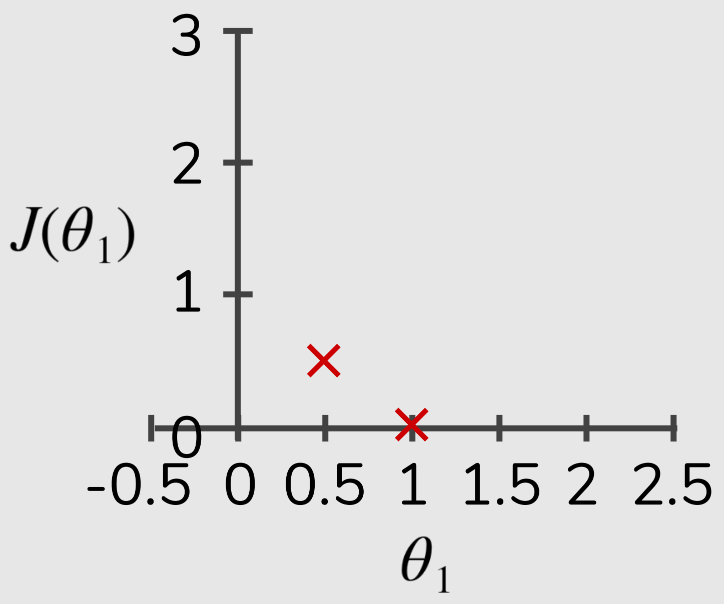
$h_\theta(x)$

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



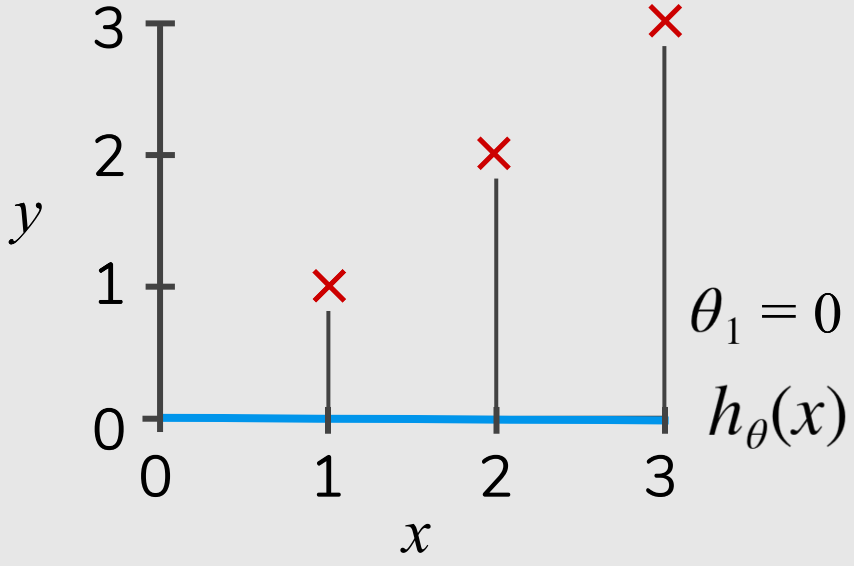
$J(\theta_1)$

(function of the parameters θ_1)



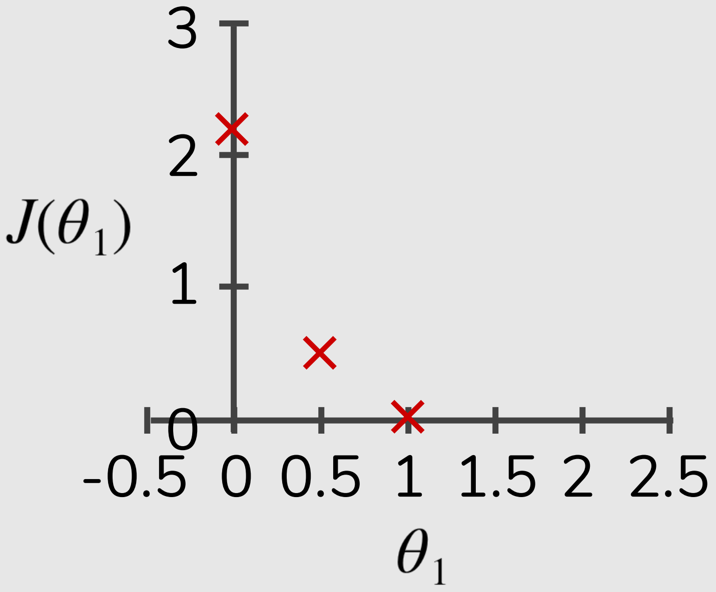
$h_\theta(x)$

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



$J(\theta_1)$

(function of the parameters θ_1)

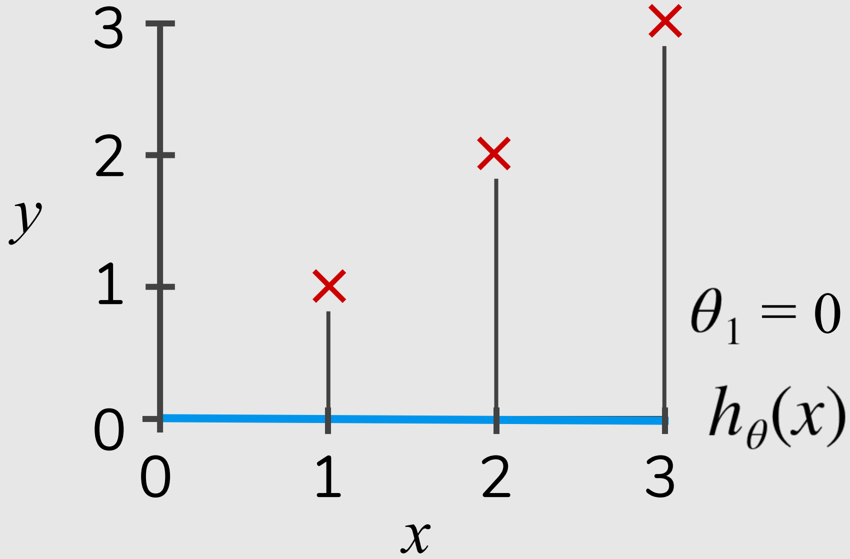


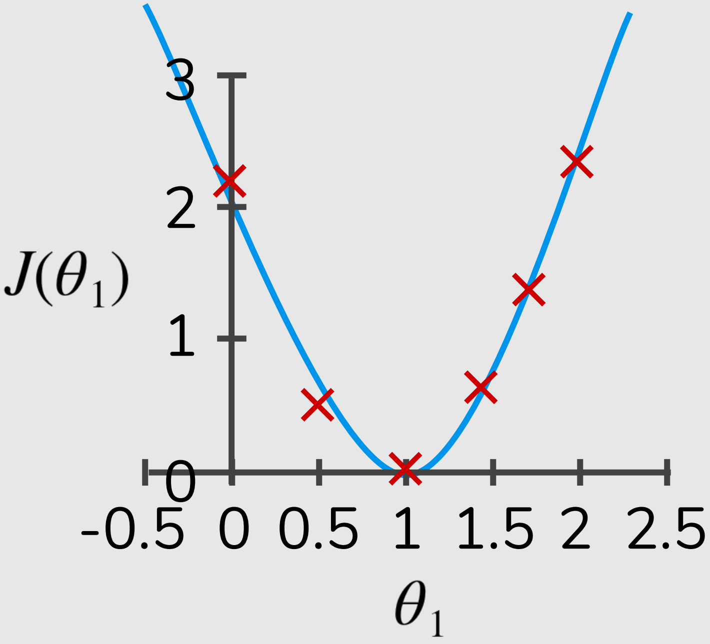
$h_\theta(x)$

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

$J(\theta_1)$

(function of the parameters θ_1)





Cost Function Intuition II

$h_\theta(x)$

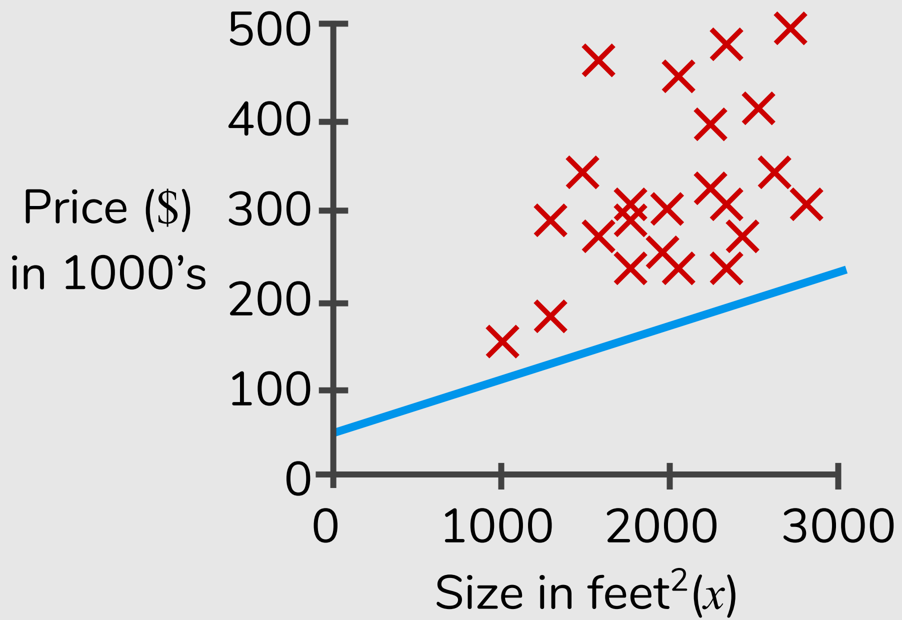
(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

$h_\theta(x)$

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



$$h_\theta(x) = 50 + 0.06x$$

$\theta_0 = 50$

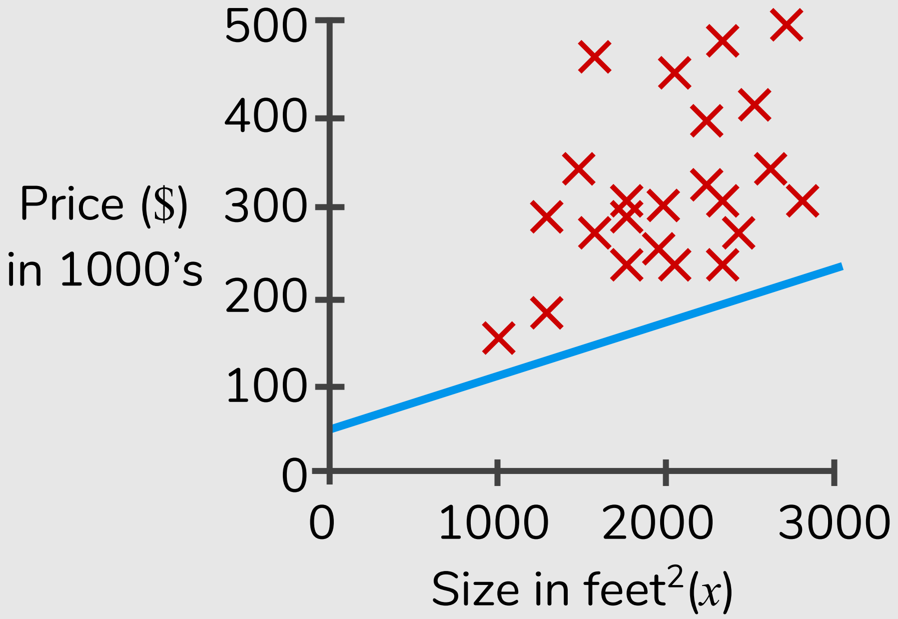
$\theta_1 = 0.06$

$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

$h_\theta(x)$

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



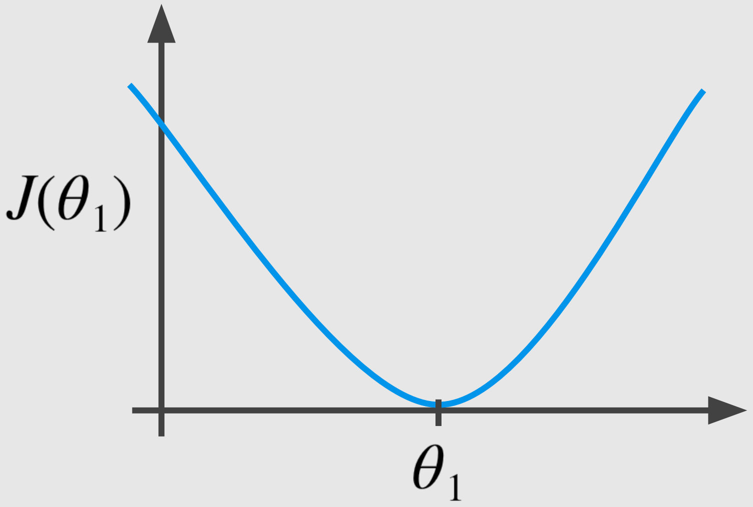
$$h_\theta(x) = 50 + 0.06x$$

$\theta_0 = 50$

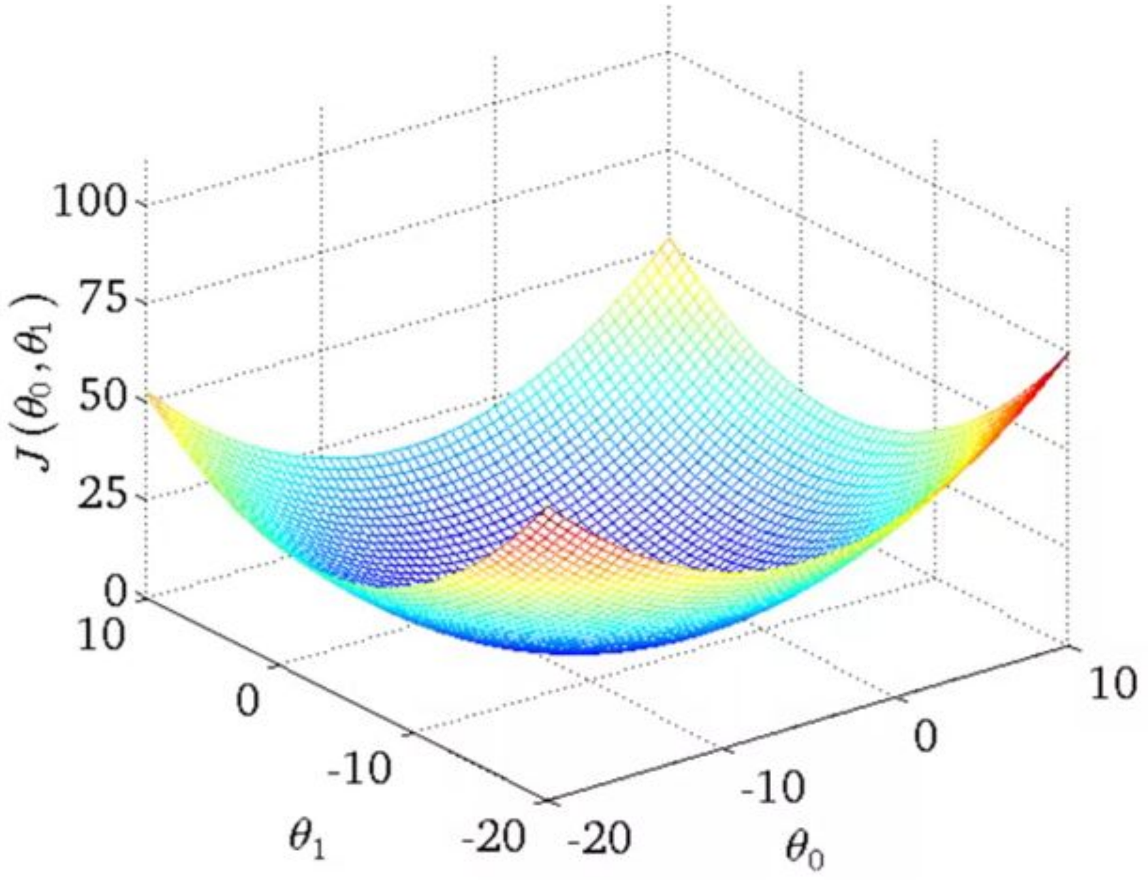
$\theta_1 = 0.06$

$J(\theta_0, \theta_1)$

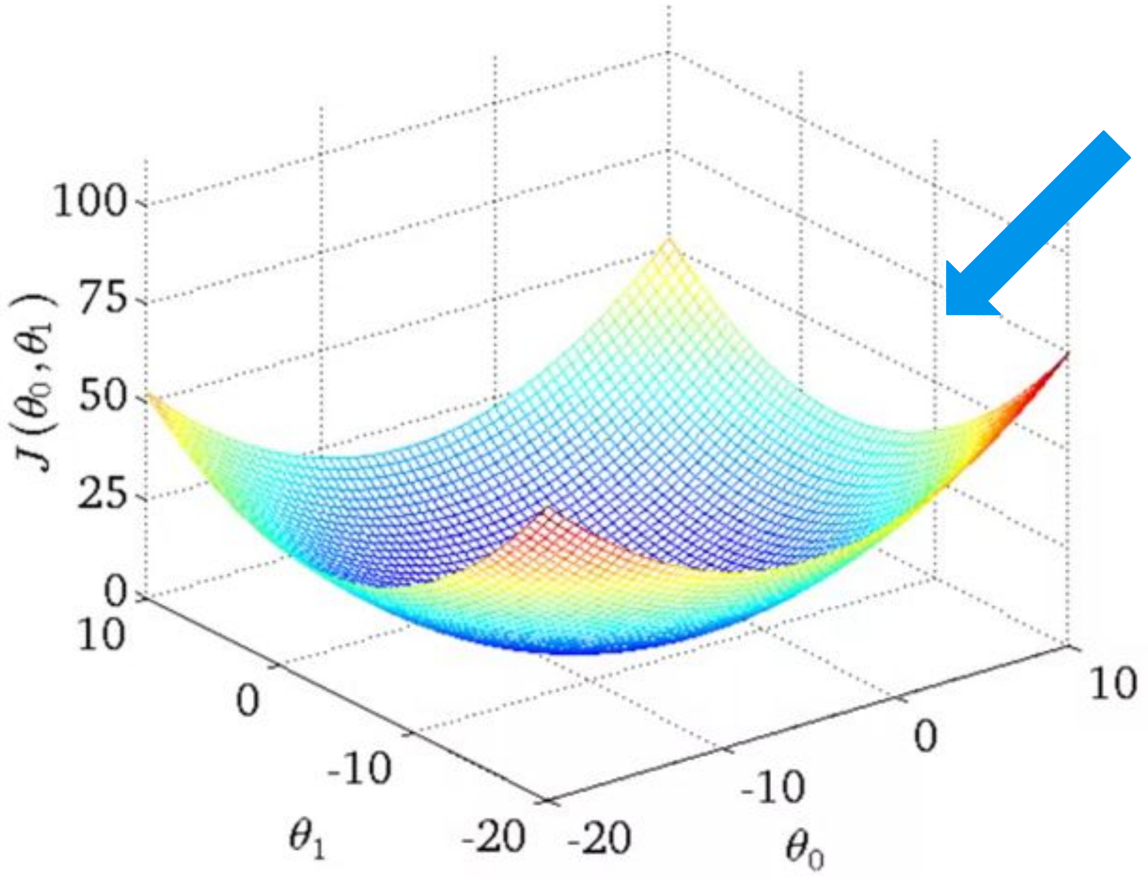
(function of the parameters θ_0, θ_1)



θ_0 and θ_1?



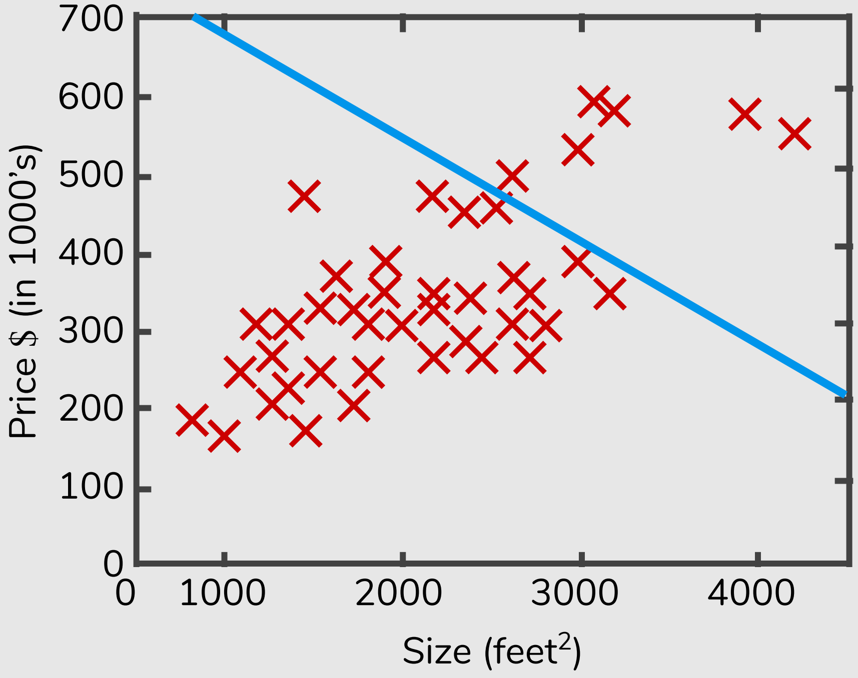
100
75
50
25
0
$J(\theta_0, \theta_1)$
10
0
-10
-20
θ_1
-20
-10
0
10
θ_0



Convex Function

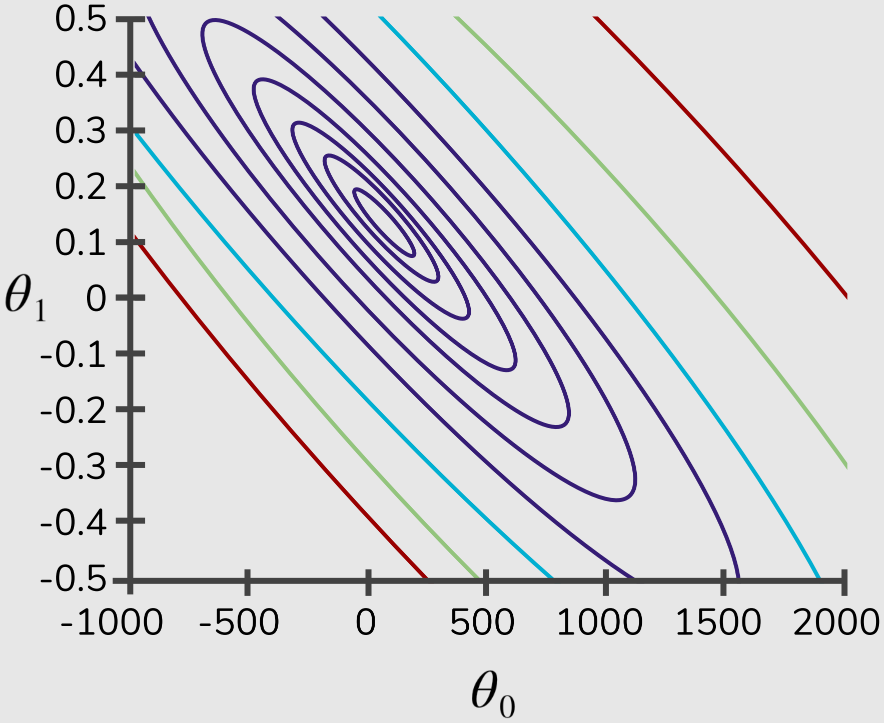
$h_\theta(x)$

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



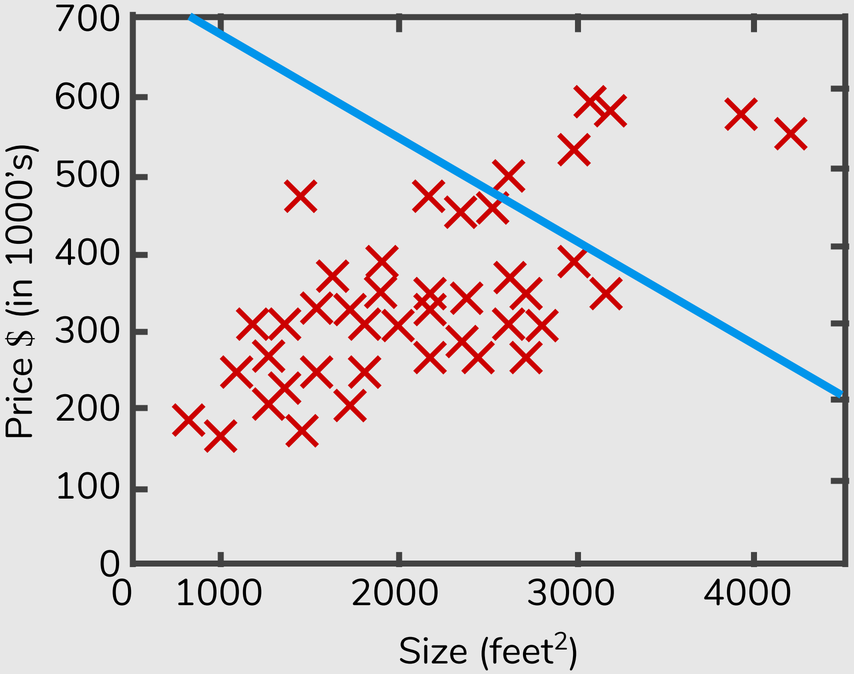
$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



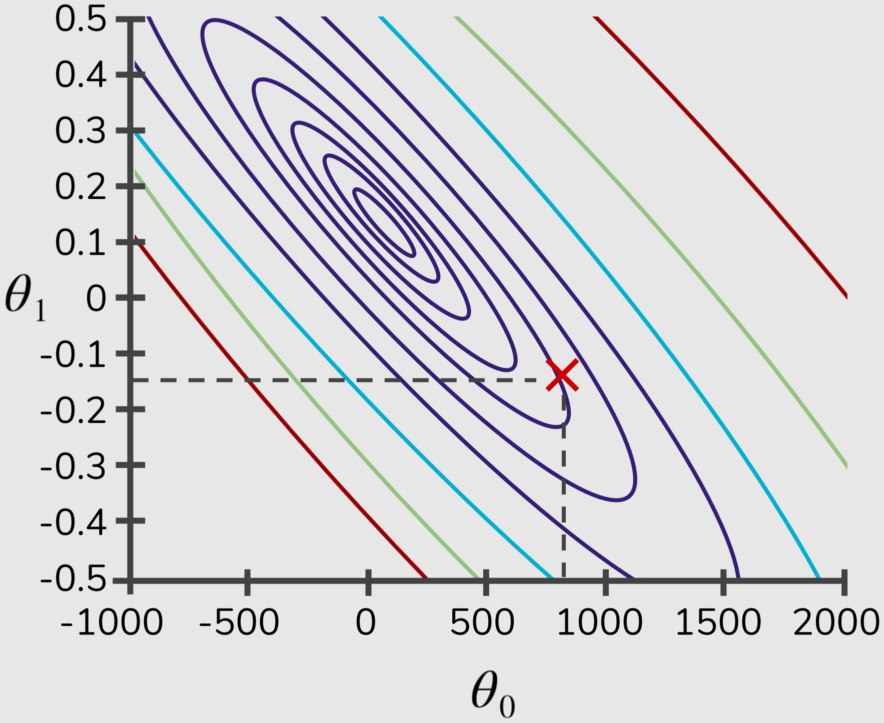
$h_\theta(x)$

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



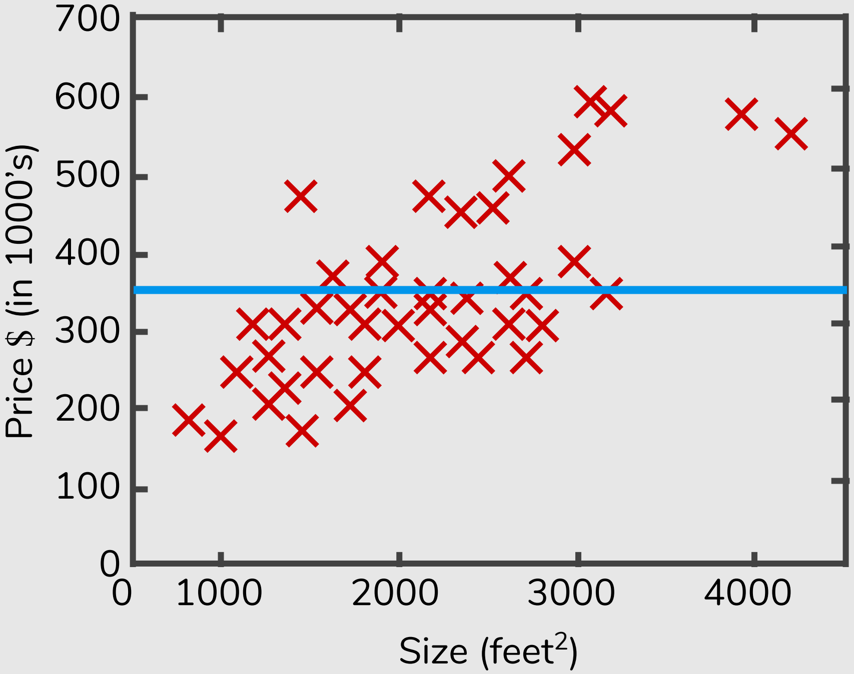
$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



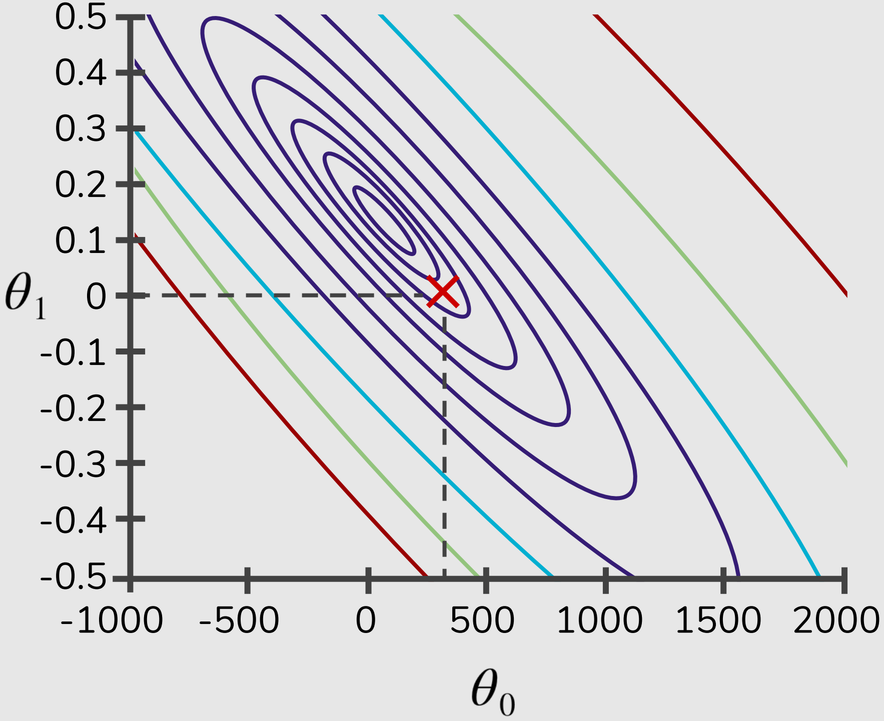
$h_\theta(x)$

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



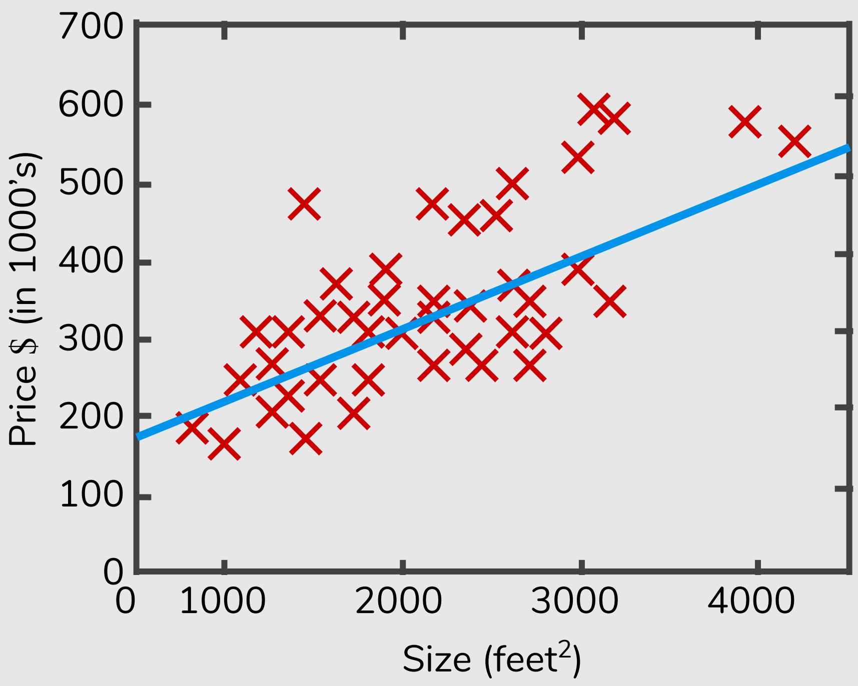
$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



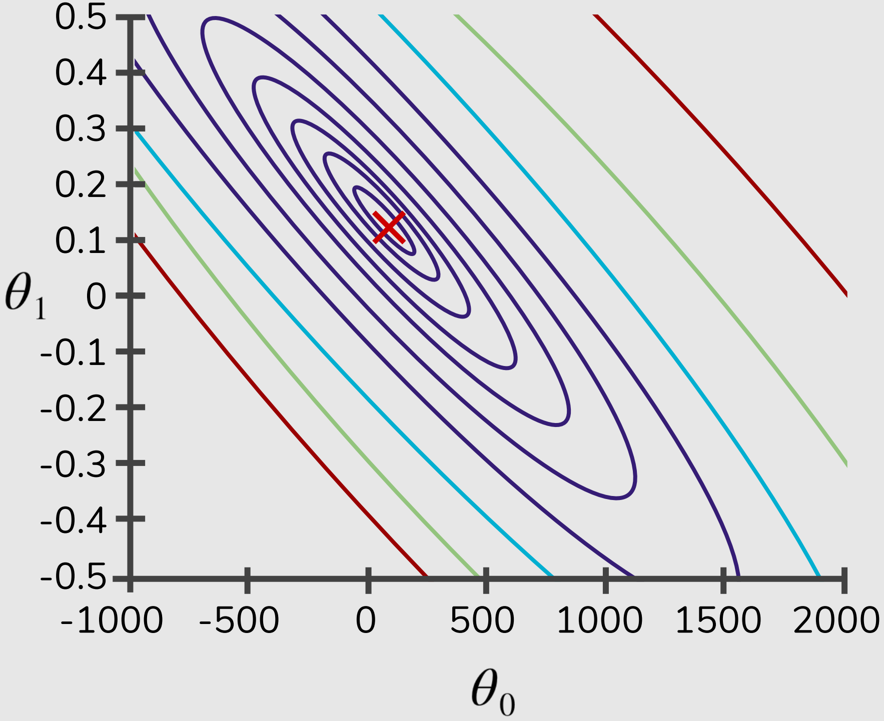
$h_\theta(x)$

(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)



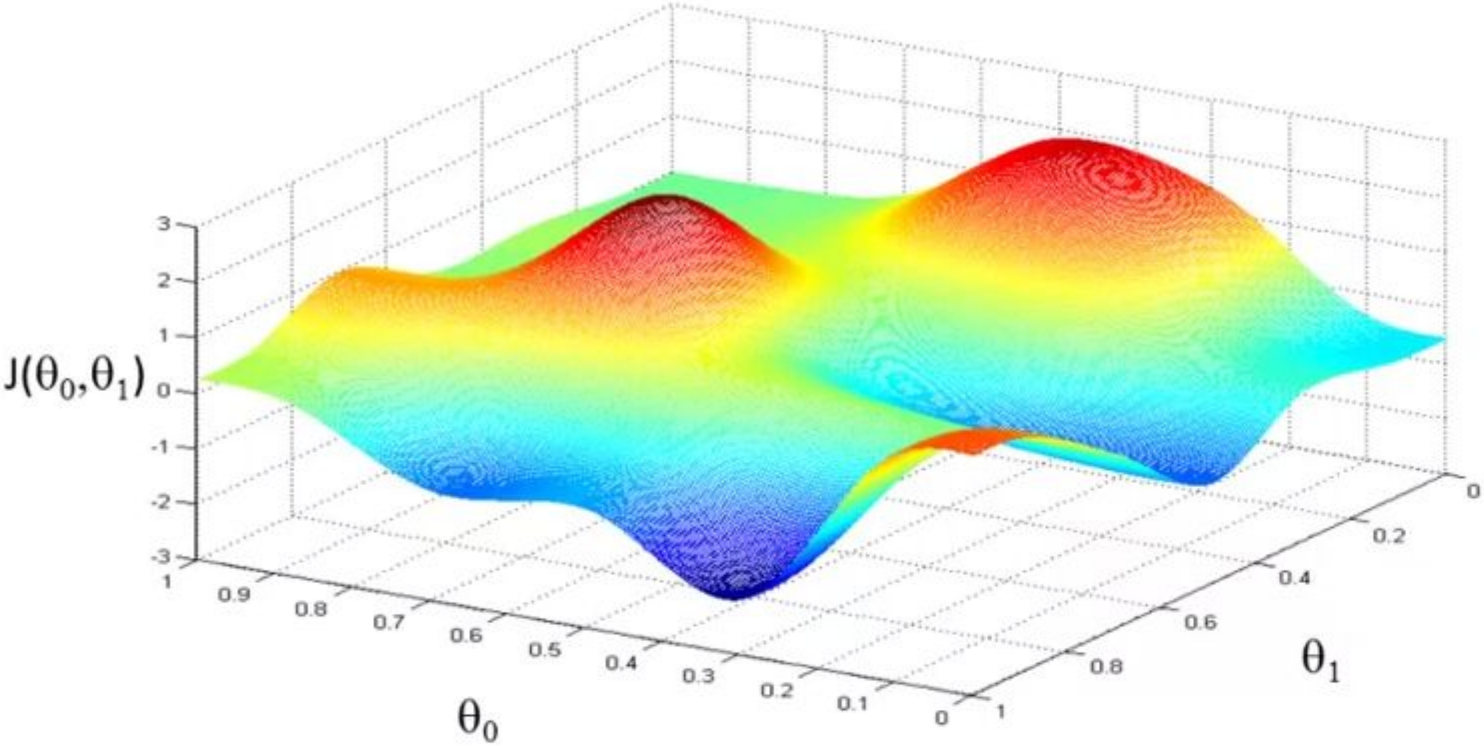
Gradient Descent

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

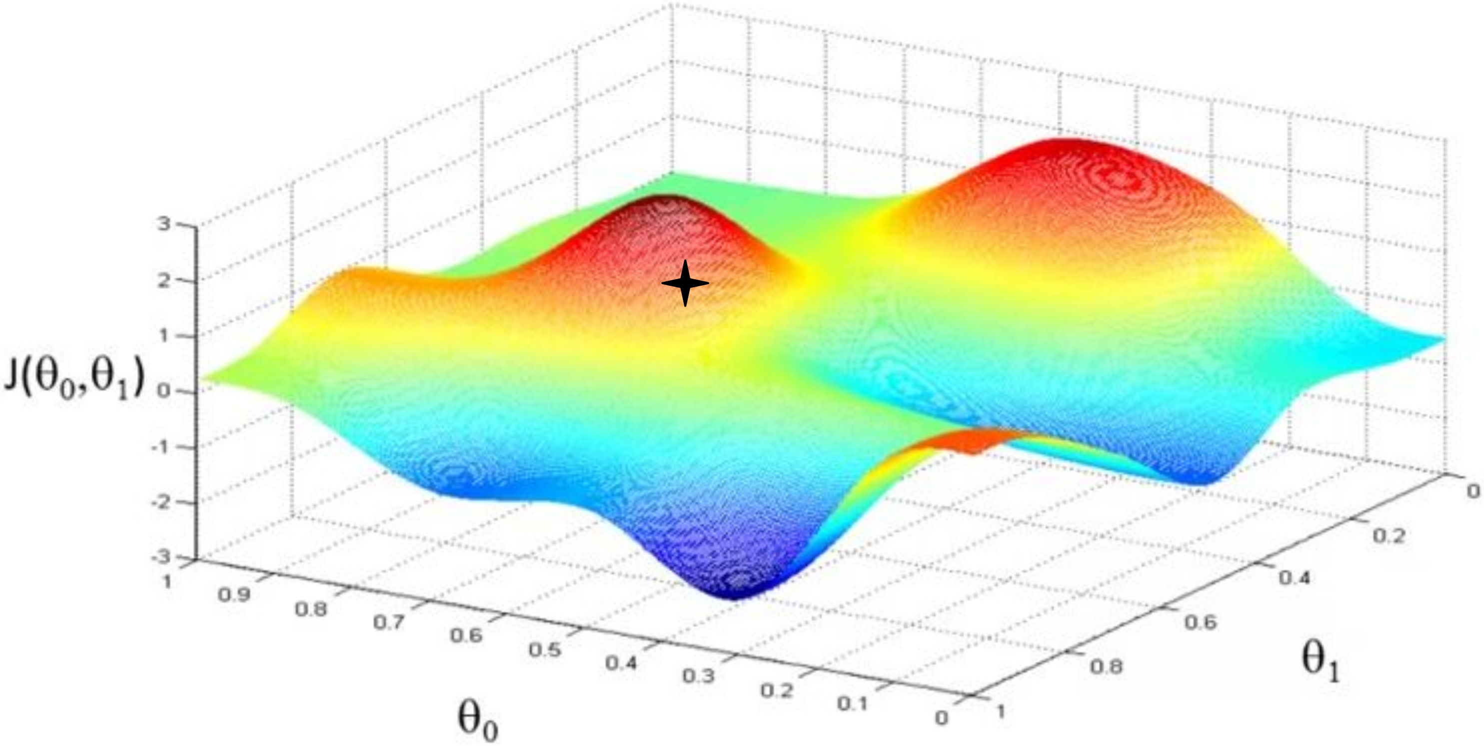
Want $\underset{\theta_0,\theta_1}{\text{minimize}}\, J(\theta_0,\theta_1)$

Outline:

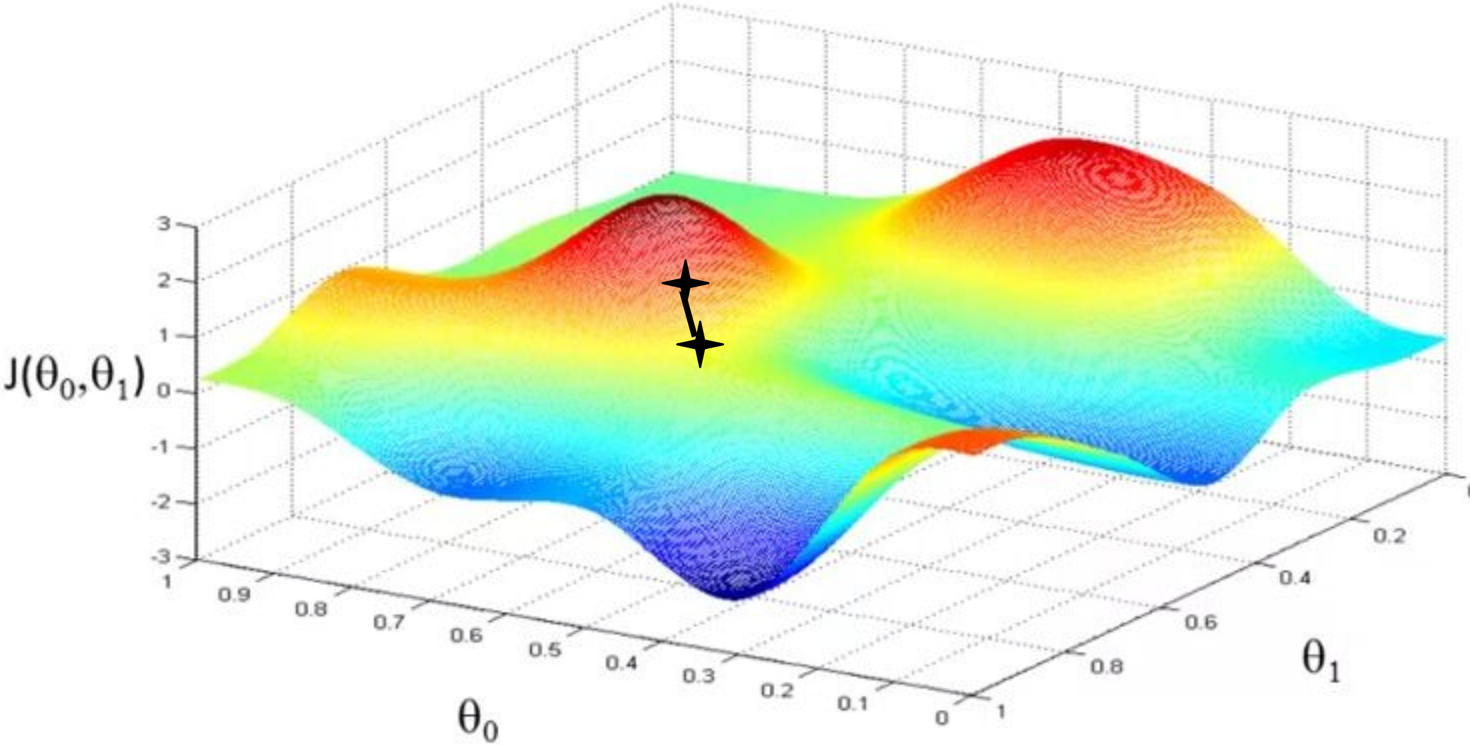
- Start with some θ_0, θ_1
- Keep changing θ_0, θ_1 to reduce $J(\theta_0,\theta_1)$ until we hopefully end up at a minimum



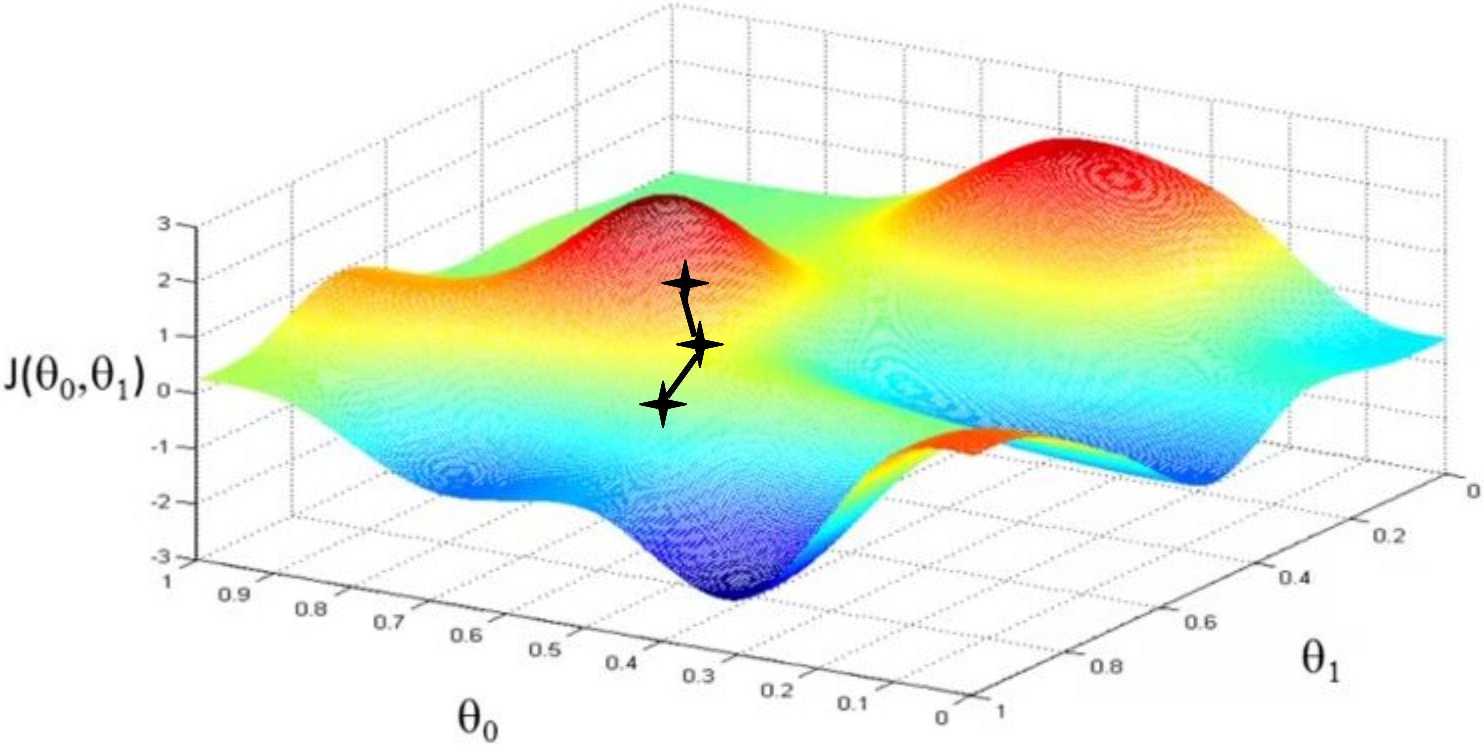
$J(\theta_0,\theta_1)$
3
2
1
0
-1
-2
-3
θ_0
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_1
0
0.2
0.4
0.6
0.8
1



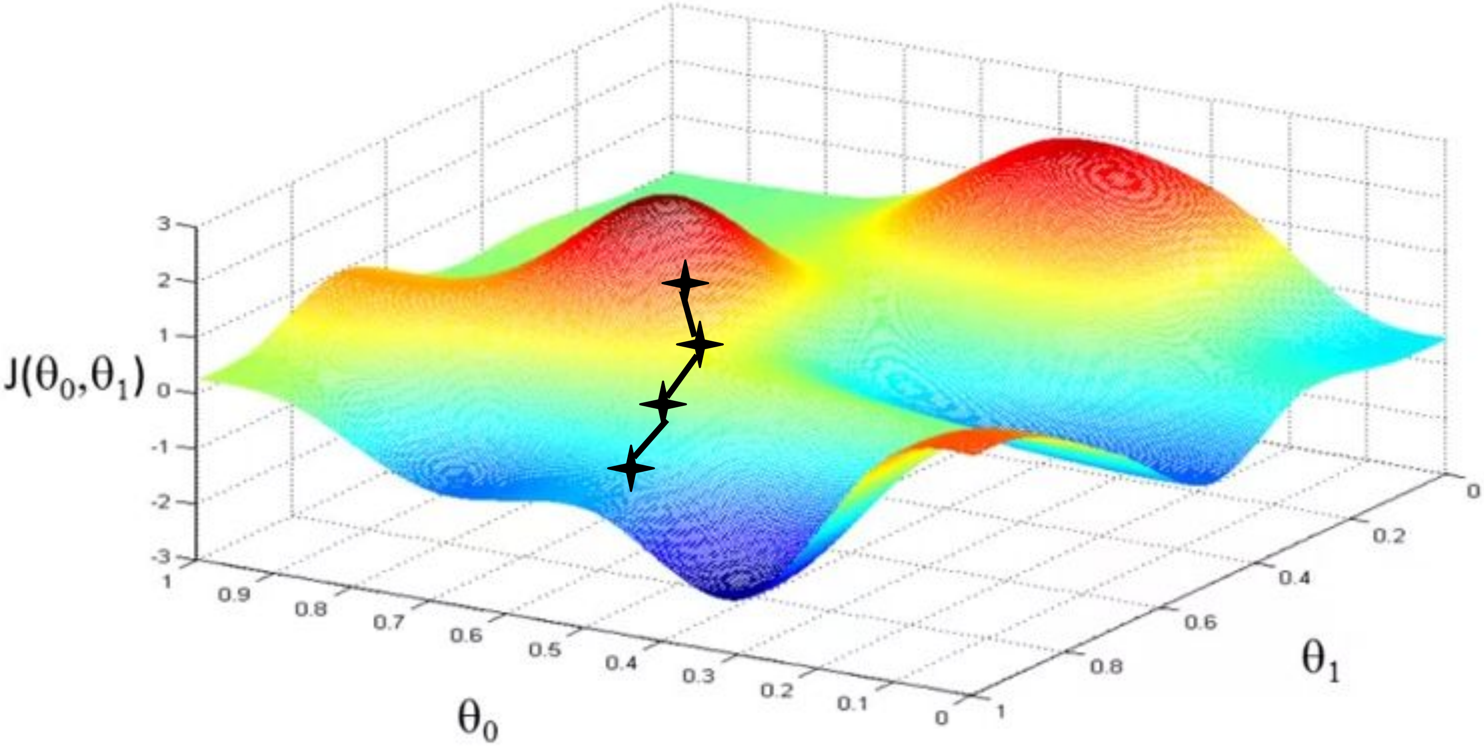
3
2
1
$J(\theta_0,\theta_1)$
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_0
0
0.2
0.4
0.6
0.8
1
θ_1



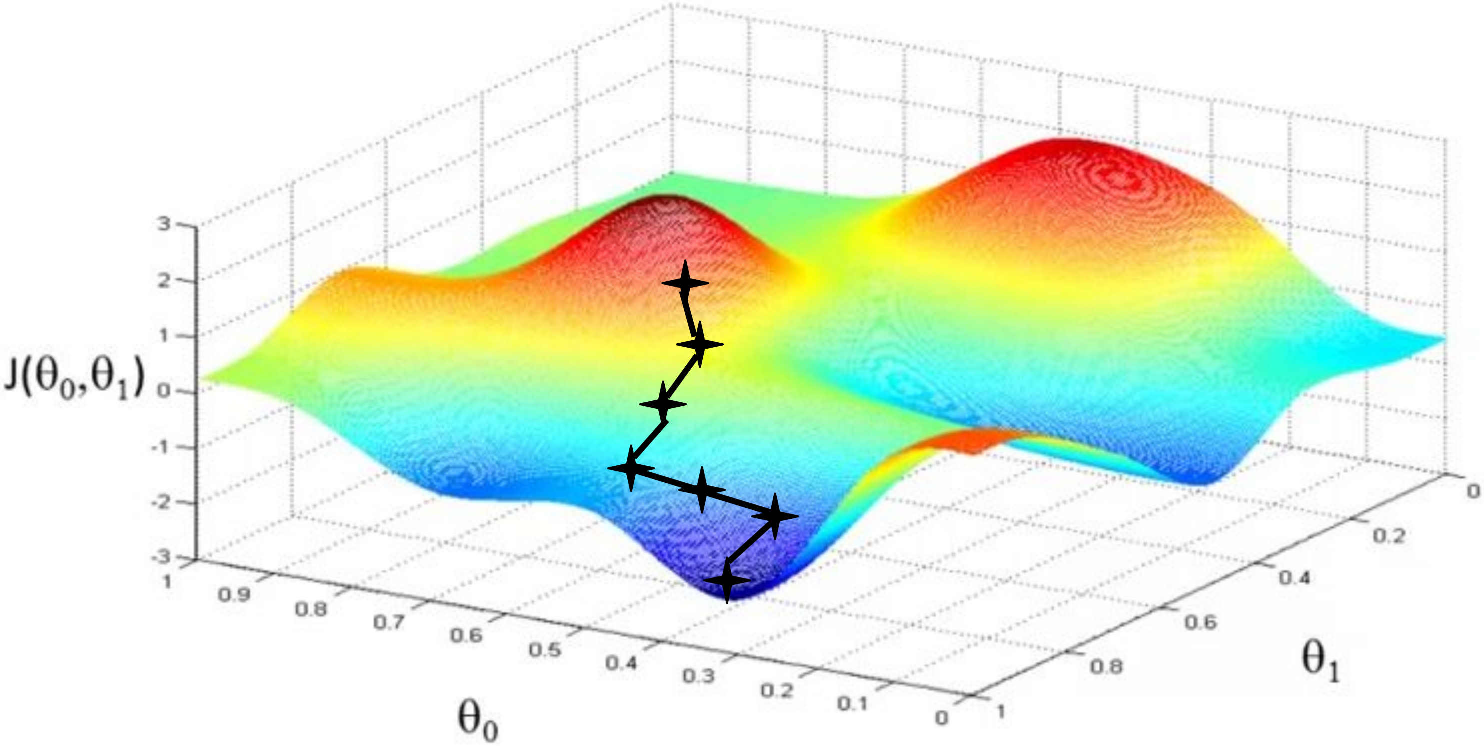
$J(\theta_0,\theta_1)$
3
2
1
0
-1
-2
-3
θ_0
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_1
0
0.2
0.4
0.6
0.8
1



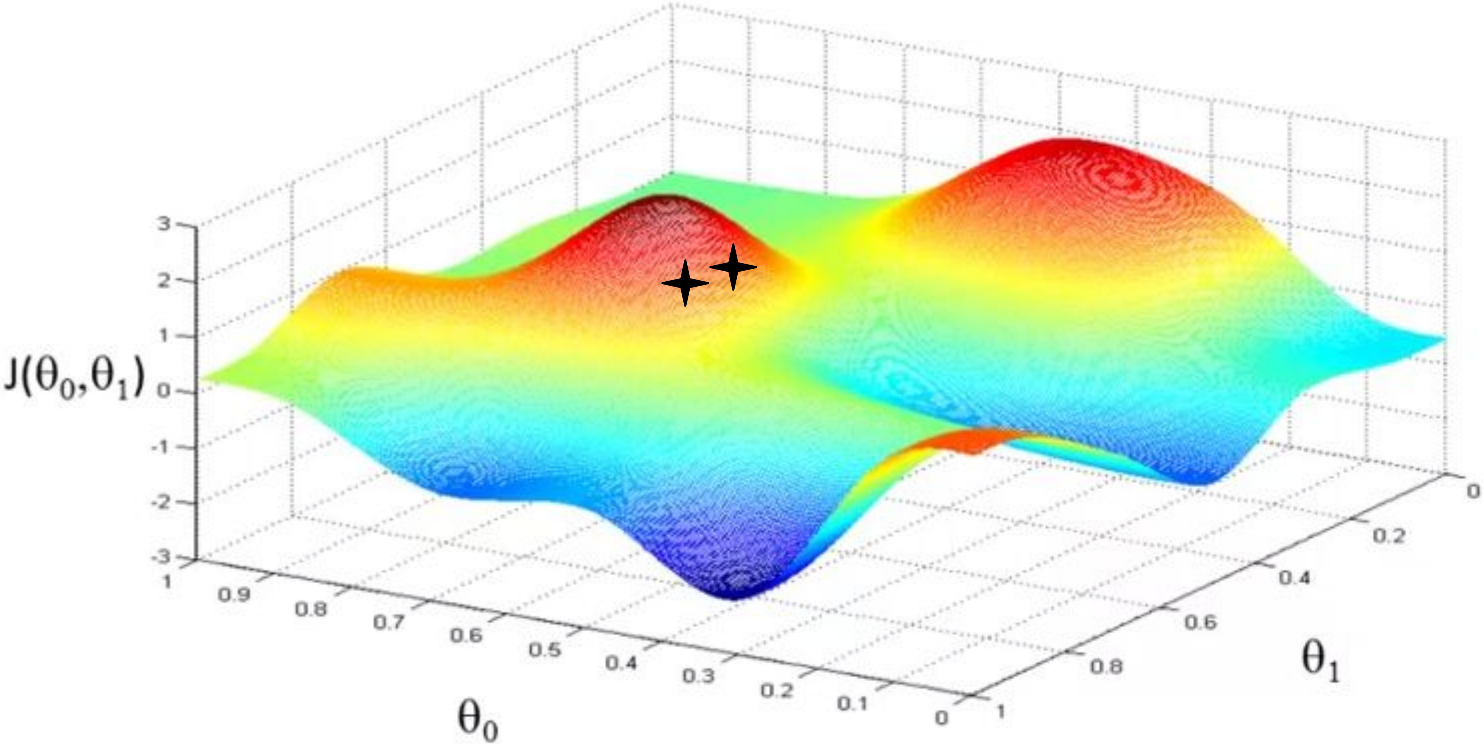
3
2
1
0
-1
-2
-3
J(θ0,θ1)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ0
0
0.2
0.4
0.6
0.8
1
θ1



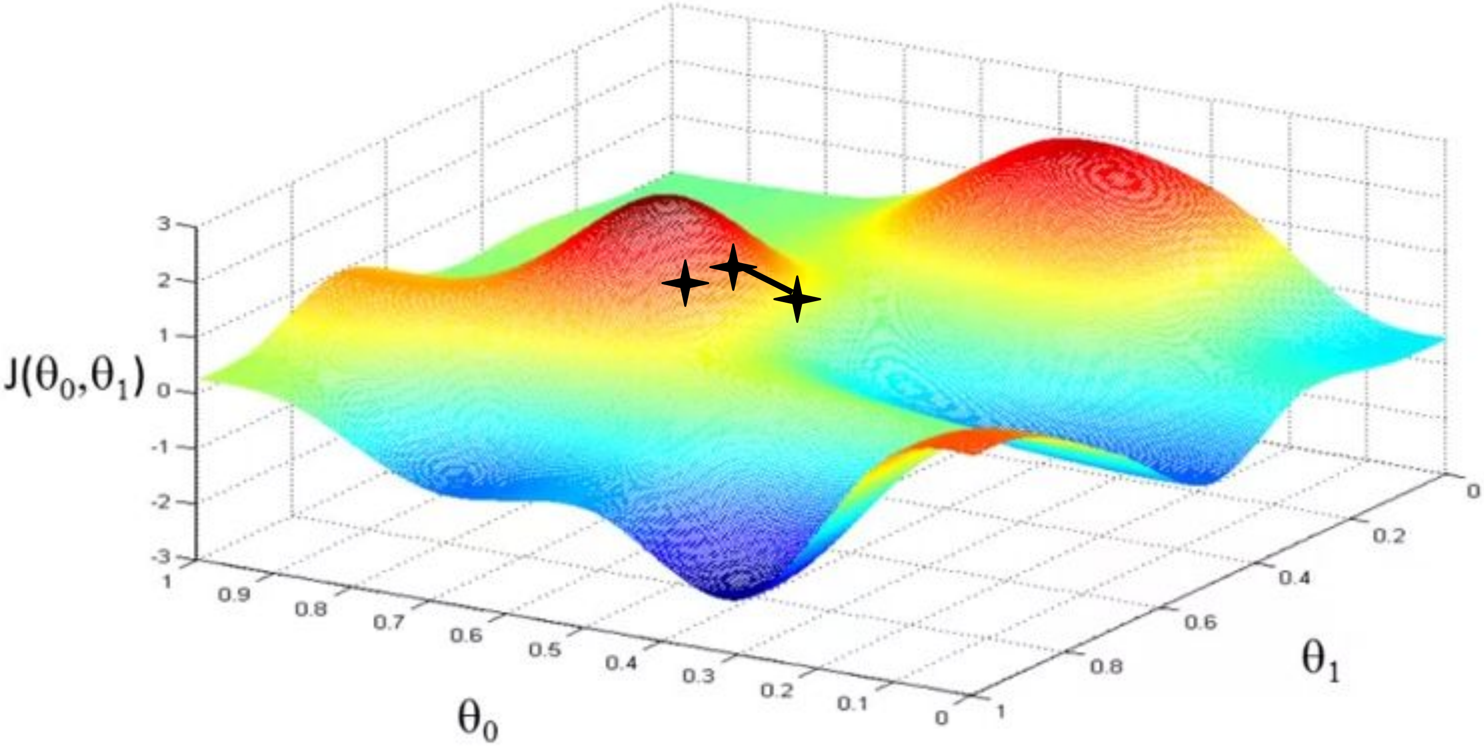
3
2
1
J(θ0,θ1)
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ0
0
0.2
0.4
0.6
0.8
1
θ1



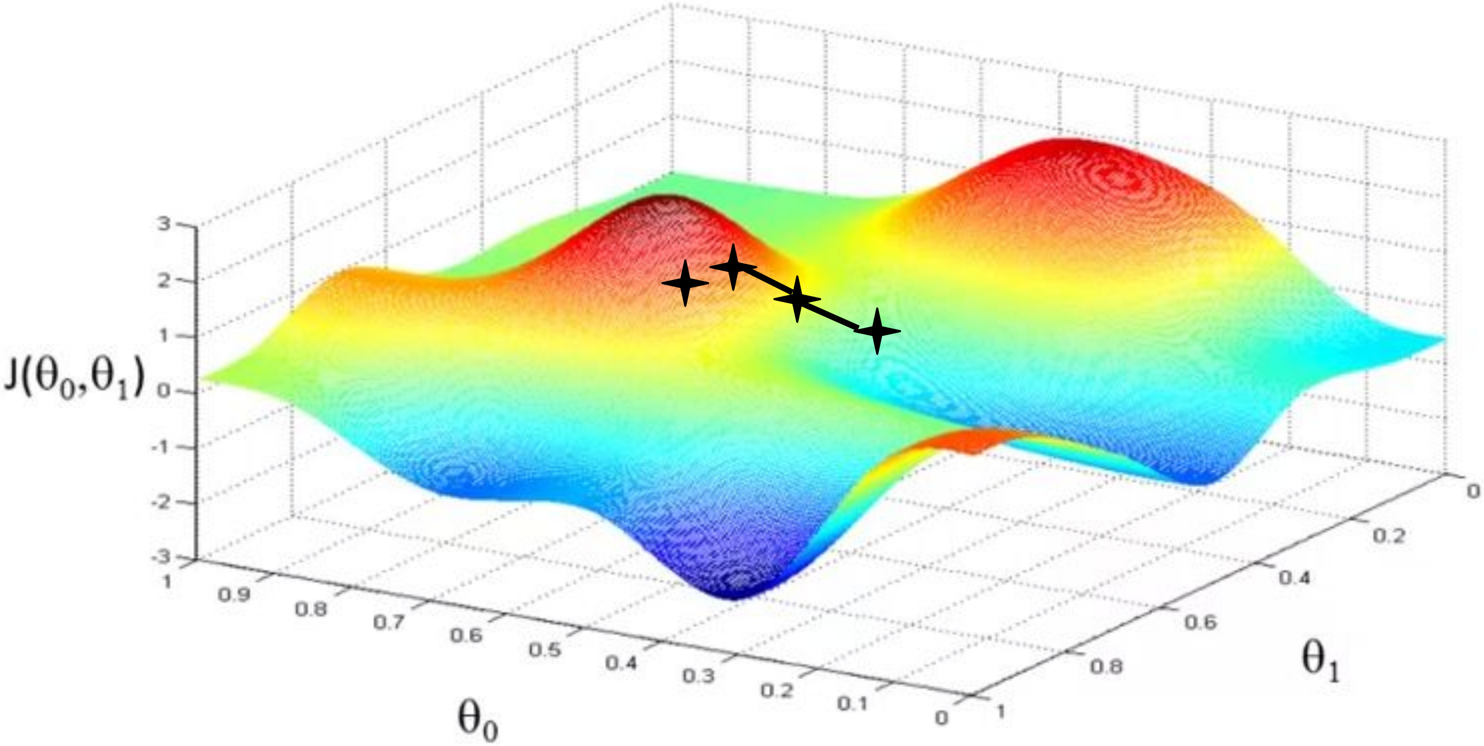
J(θ0,θ1)
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ0
0
0.2
0.4
0.6
0.8
1
θ1



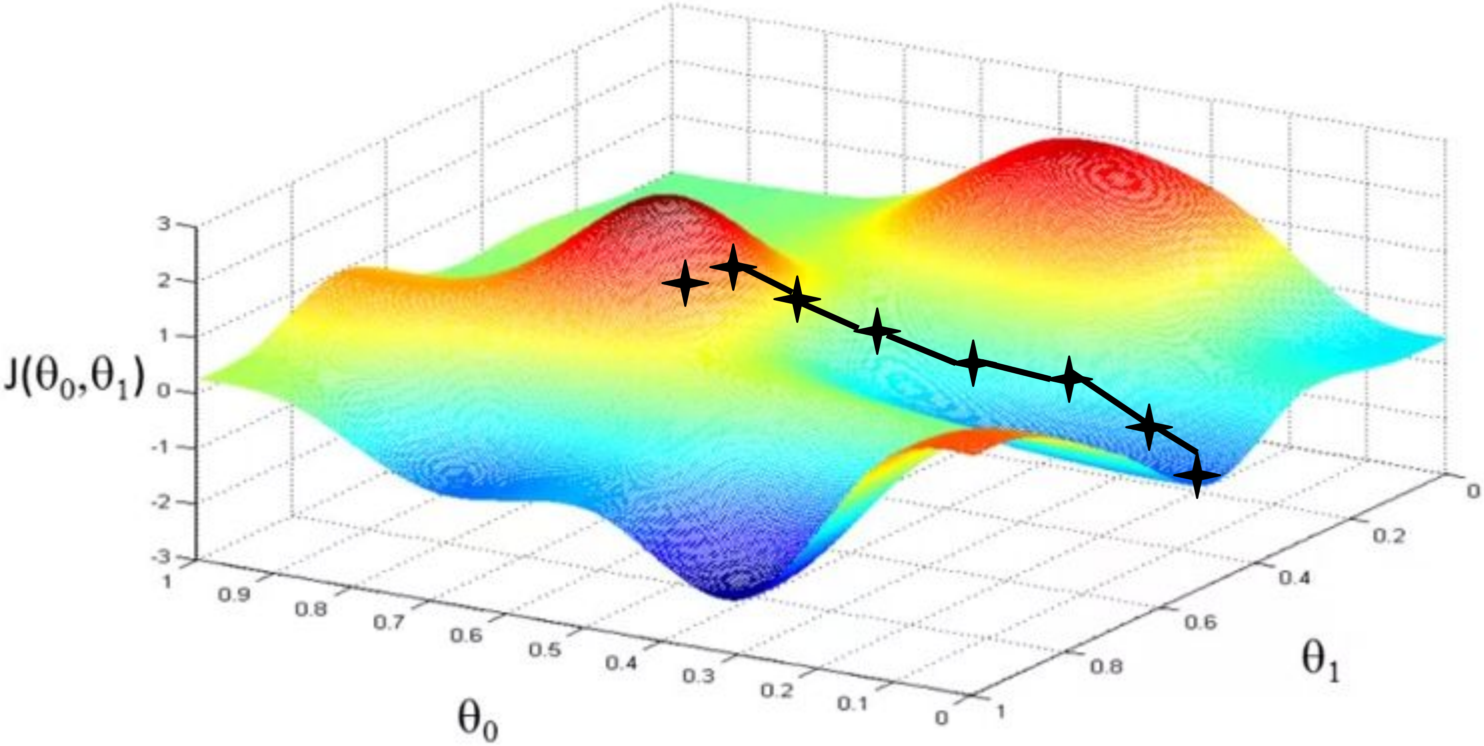
$J(\theta_0,\theta_1)$
3
2
1
0
-1
-2
-3
θ_0
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_1
0
0.2
0.4
0.6
0.8
1



$J(\theta_0,\theta_1)$
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_0
0
0.2
0.4
0.6
0.8
1
θ_1



$J(\theta_0,\theta_1)$
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_0
0
0.2
0.4
0.6
0.8
1
θ_1



$J(\theta_0,\theta_1)$
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_0
0
0.2
0.4
0.6
0.8
1
θ_1

Gradient Descent algorithm

repeat until convergence {

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

(simultaneously update

$j = 0$ and $j = 1$)

}

Gradient Descent algorithm

repeat until convergence {

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

(simultaneously update $j = 0$ and $j = 1$)

}

Learning rate

Derivative term

Gradient Descent algorithm

repeat until convergence {

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

}

Correct: Simultaneous update

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

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

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

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

Gradient Descent algorithm

repeat until convergence {

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

}

Correct: Simultaneous update

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

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

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

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

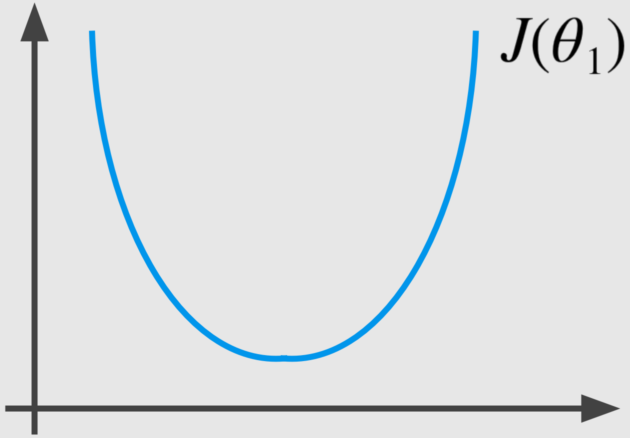
Incorrect

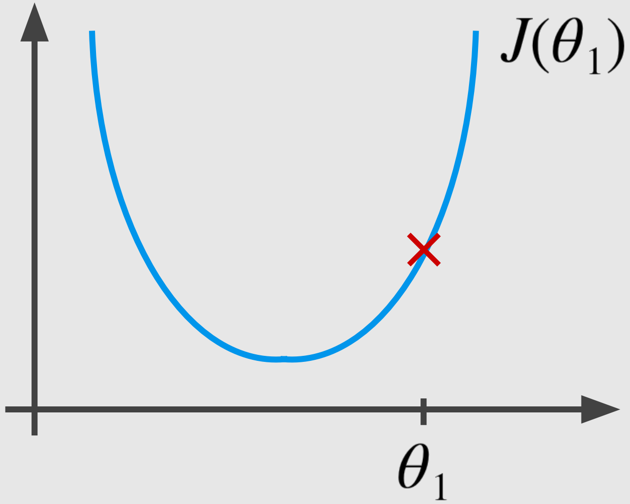
$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

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

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

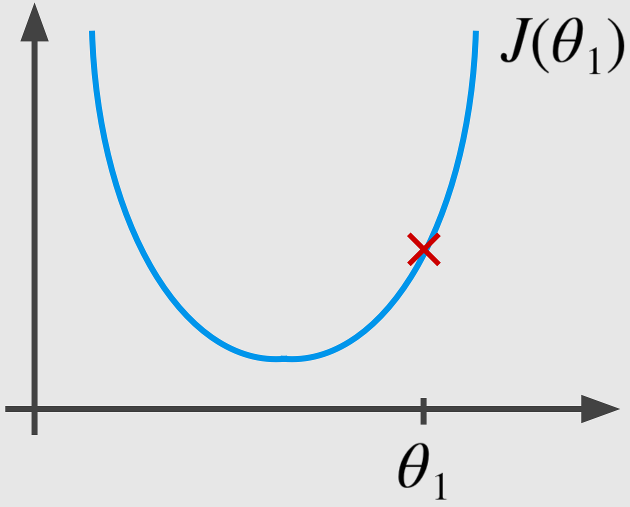
$$\theta_1 := \text{temp1}$$


$J(\theta_1)$



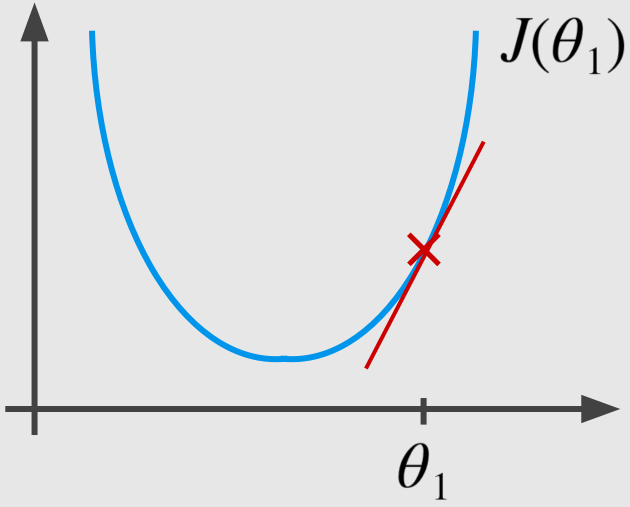
$\theta_1 \in \mathbb{R}$

$$\theta_1 \in \mathbb{R}$$



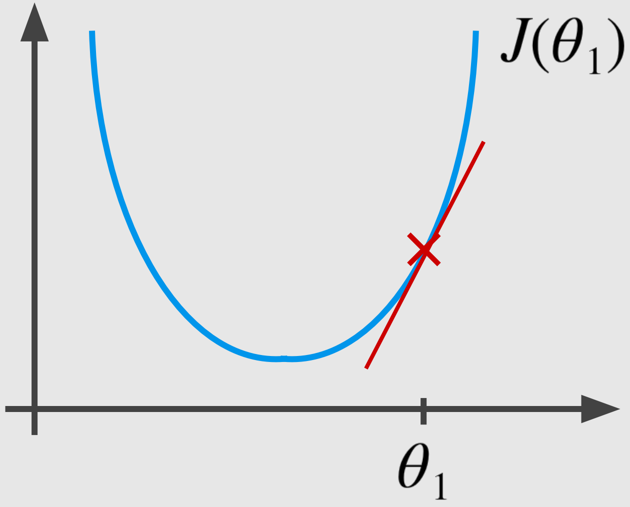
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

$$\theta_1 \in \mathbb{R}$$



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

$\theta_1 \in \mathbb{R}$

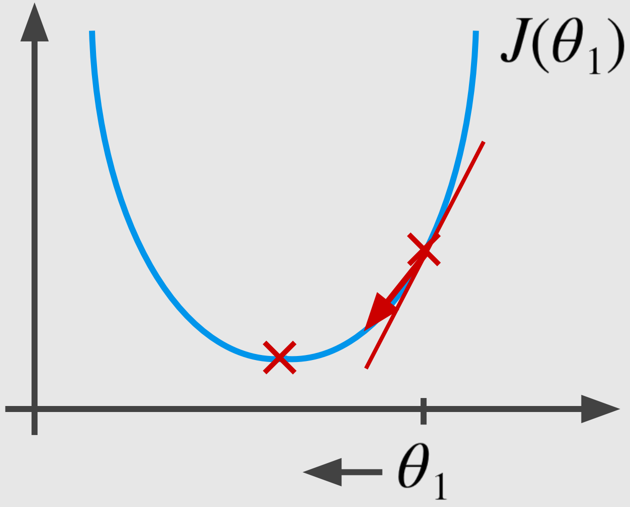


≥ 0

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

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

$$\theta_1 \in \mathbb{R}$$

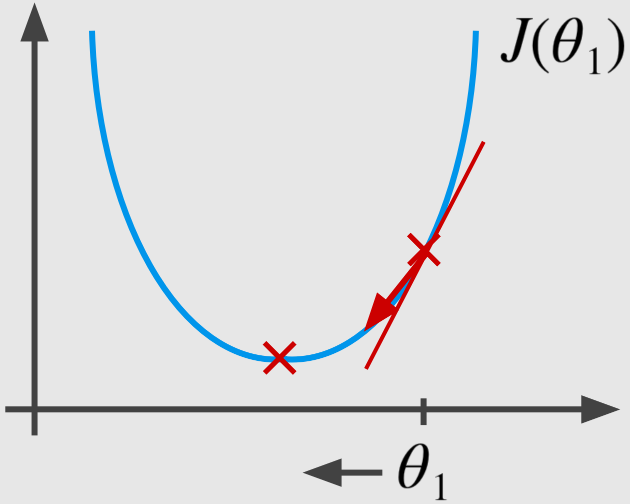


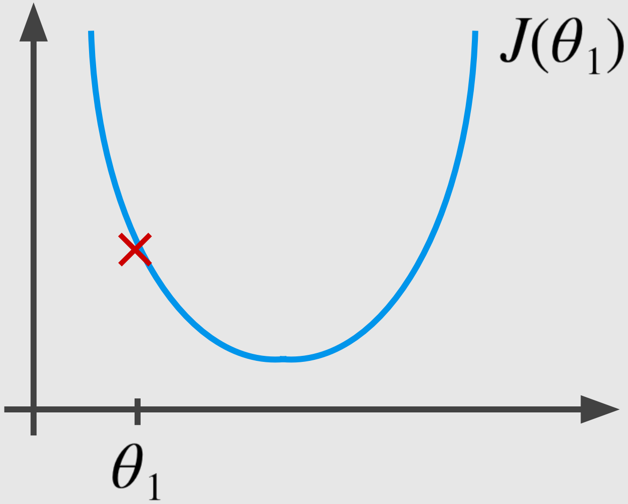
≥ 0

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

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

$$\theta_1 \in \mathbb{R}$$



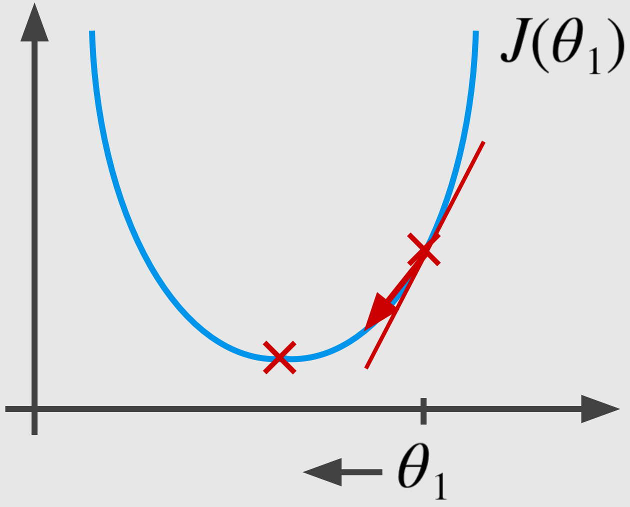


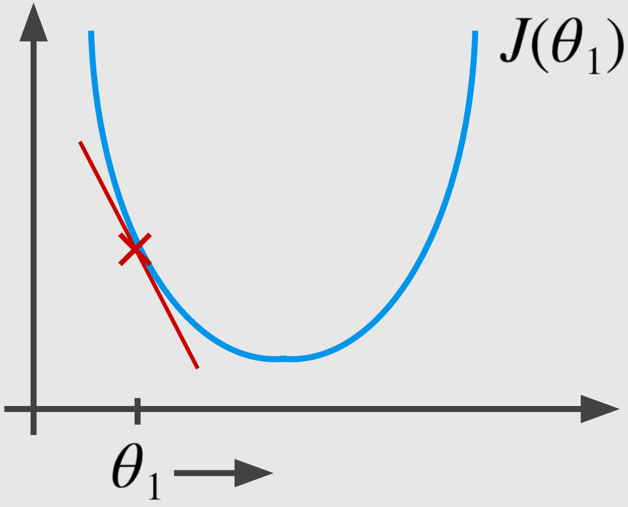
≥ 0

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

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

$$\theta_1 \in \mathbb{R}$$





≥ 0

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

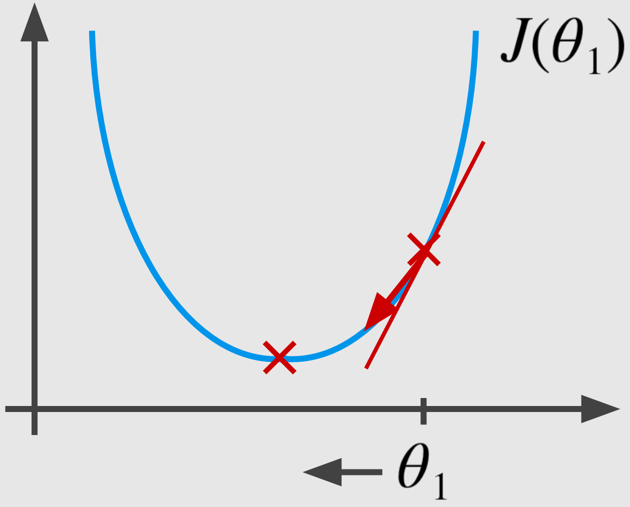
$$\theta_1 := \theta_1 - \alpha \cdot \text{(positive number)}$$

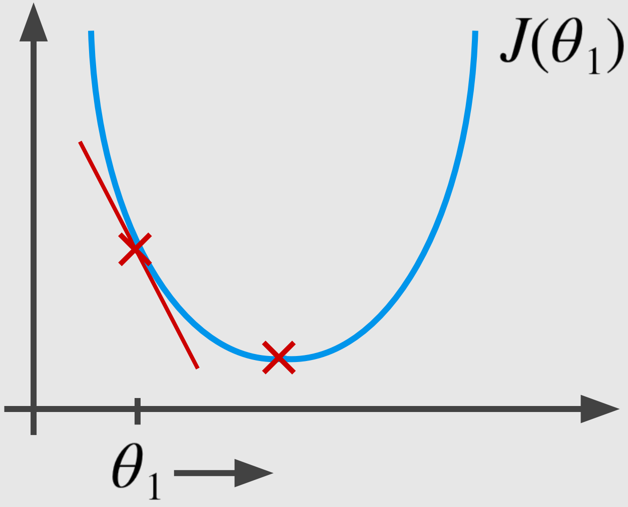
≤ 0

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

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

$$\theta_1 \in \mathbb{R}$$





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

≥ 0

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

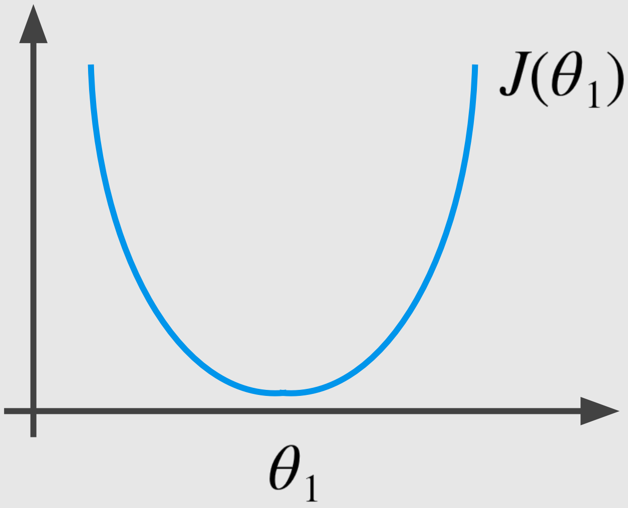
$$\theta_1 := \theta_1 - \alpha \boxed{\frac{d}{d\theta_1} J(\theta_1)}$$

≤ 0

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

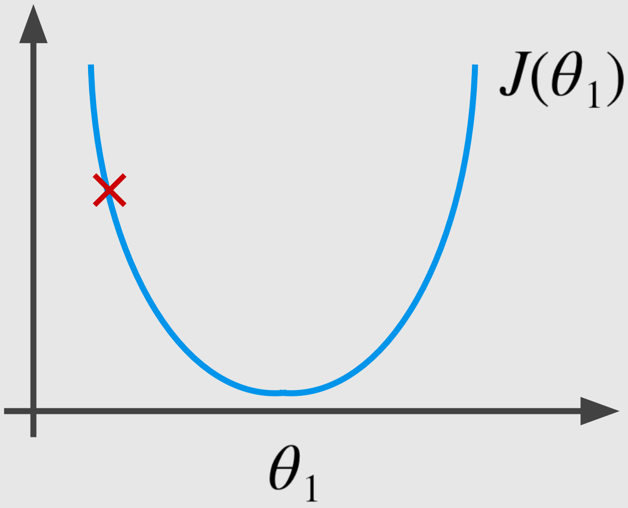
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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



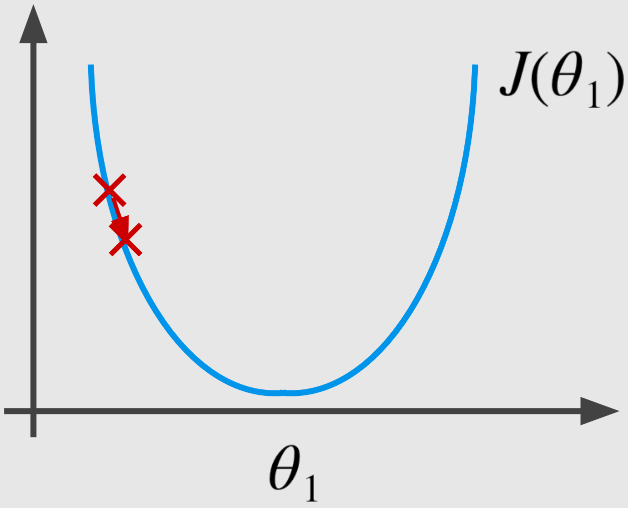
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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



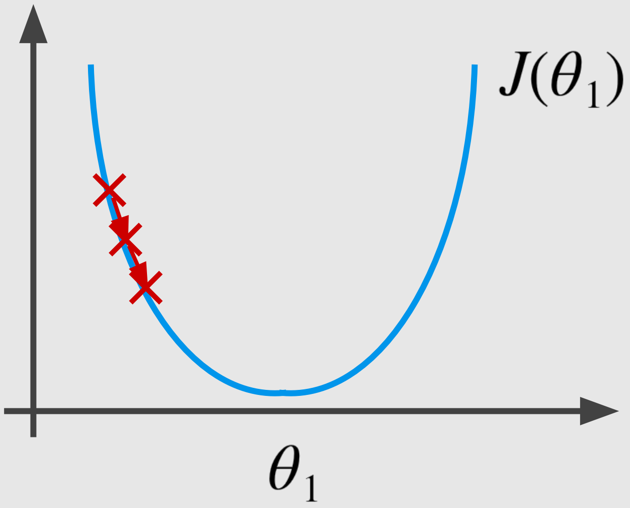
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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



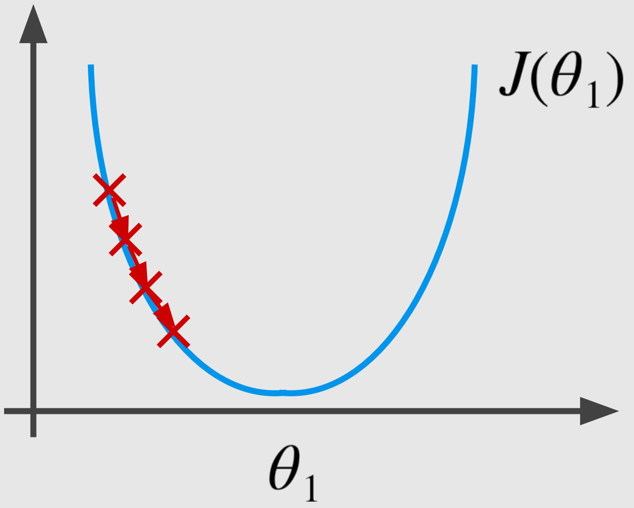
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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

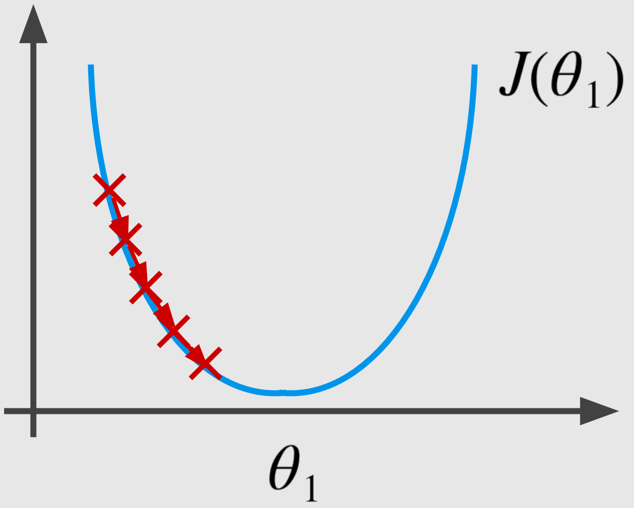


$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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



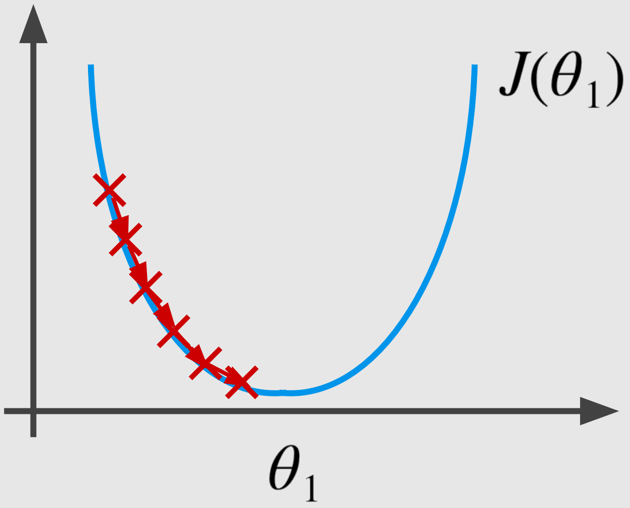
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$



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

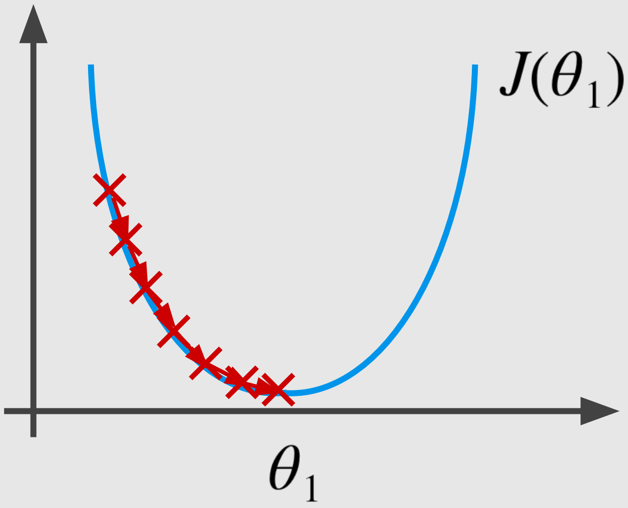
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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



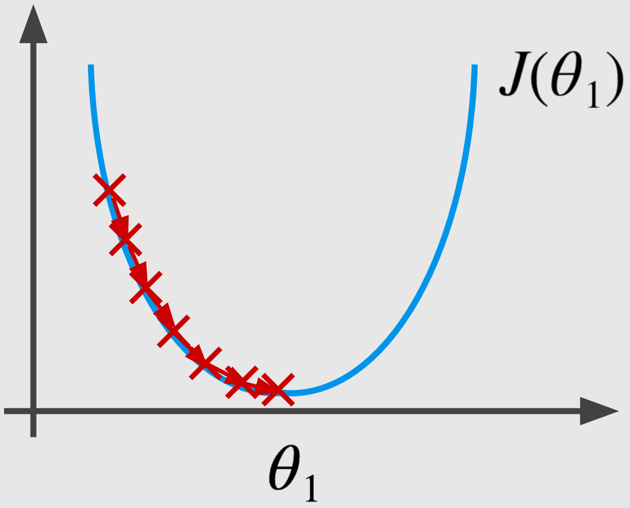
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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

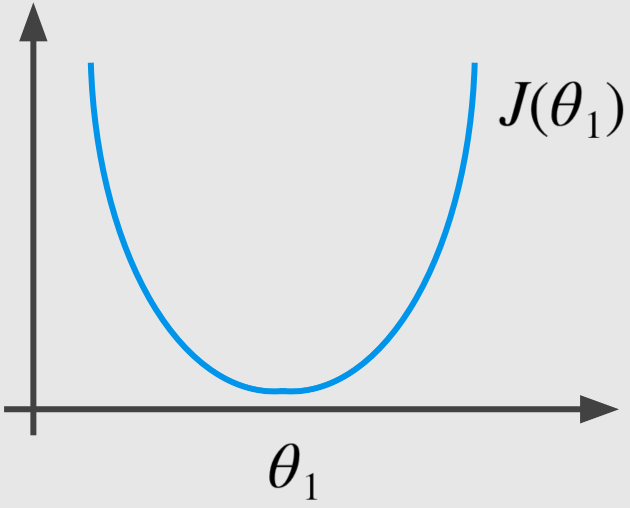


$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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

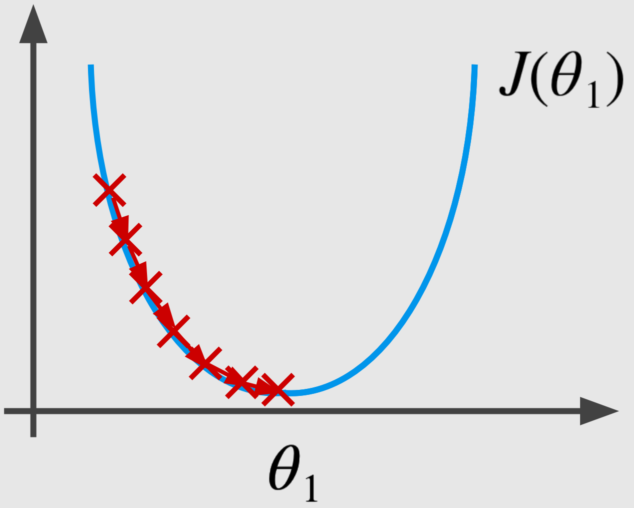


If α is too large, gradient descent can be ...

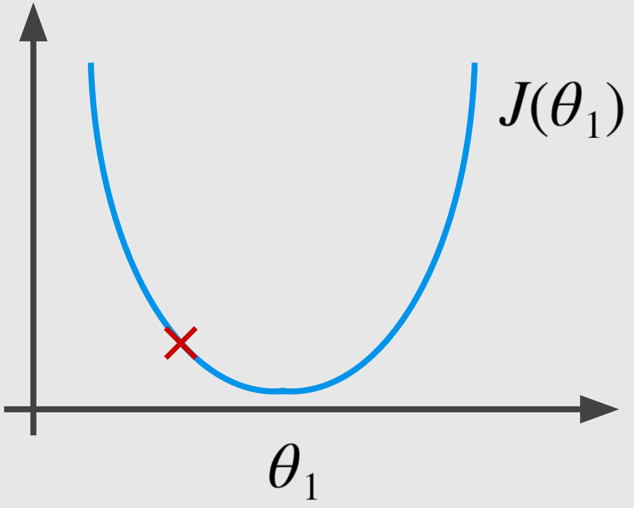


$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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

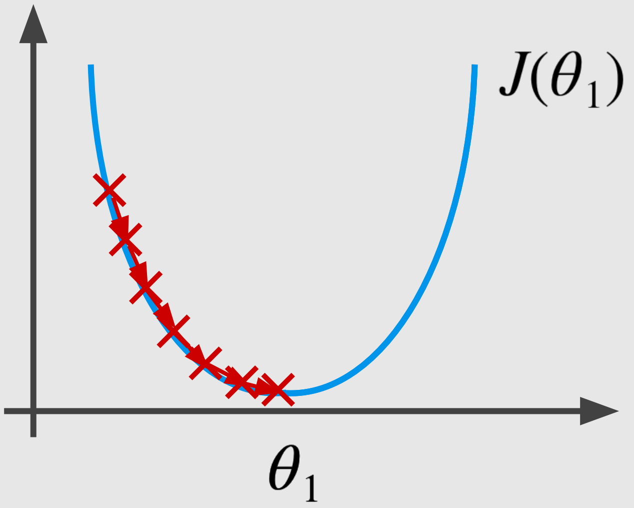


If α is too large, gradient descent can be ...

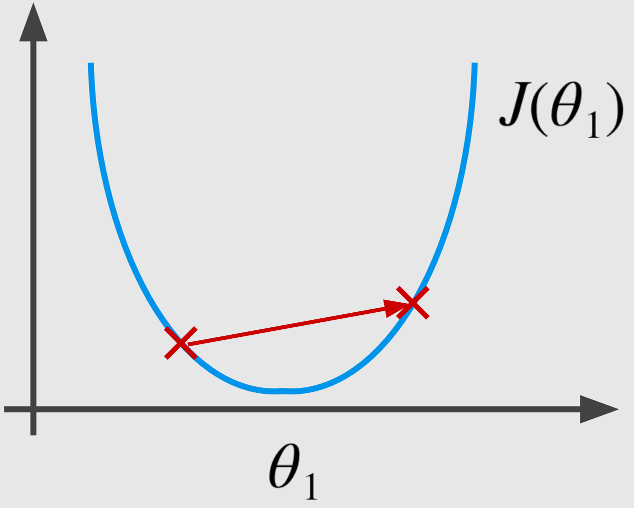


$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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

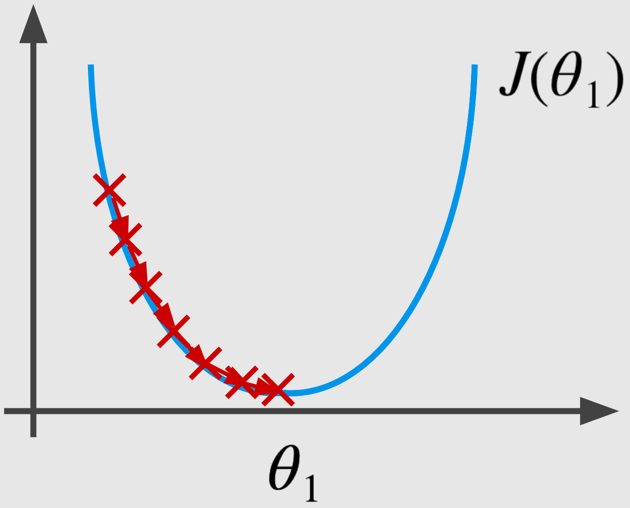


If α is too large, gradient descent can be overshoot the minimum. It may fail to converge, or even diverge.

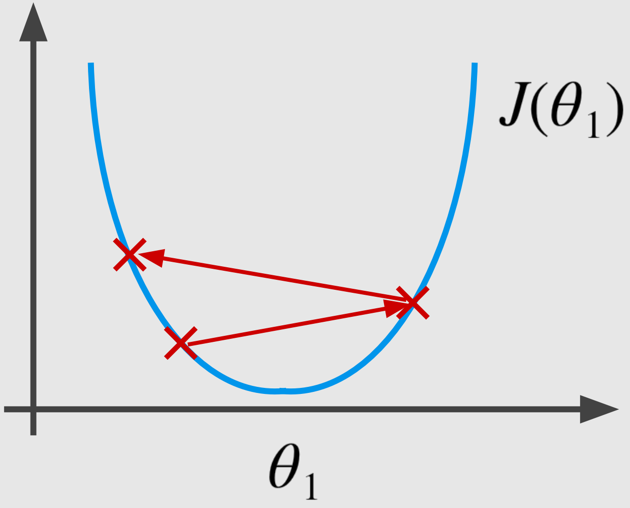


$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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

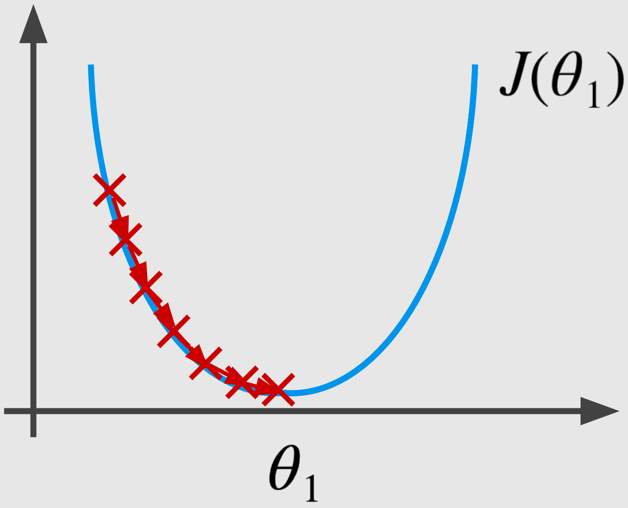


If α is too large, gradient descent can be overshoot the minimum. It may fail to converge, or even diverge.

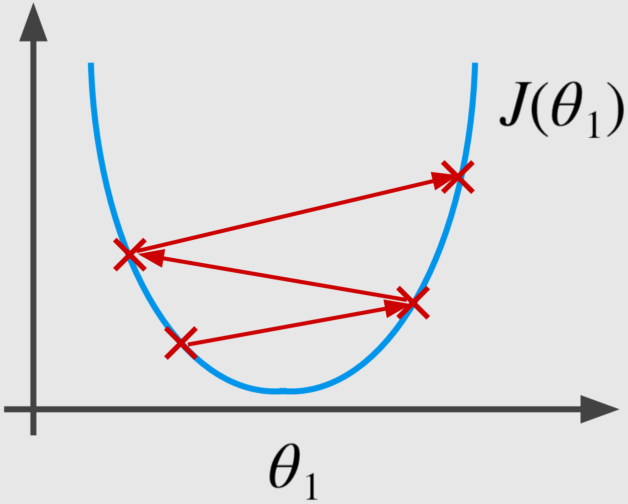


$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

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

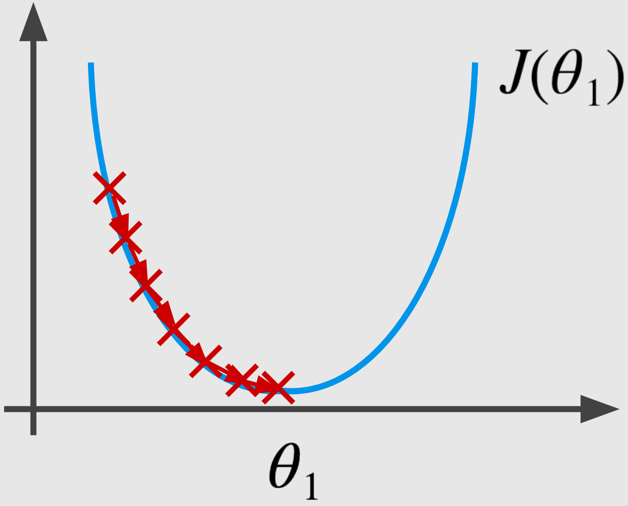


If α is too large, gradient descent can be overshoot the minimum. It may fail to converge, or even diverge.

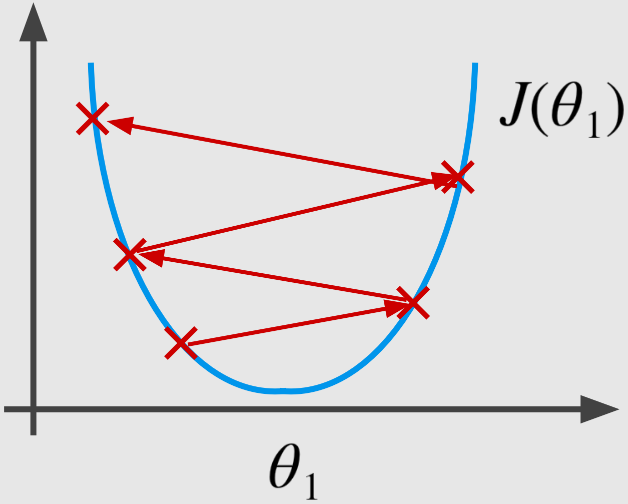


$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

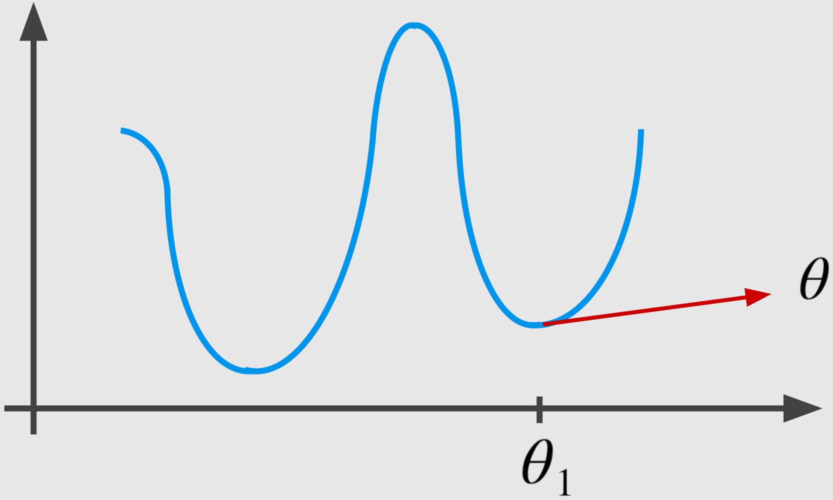
If α is too small, gradient descent can be slow.



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

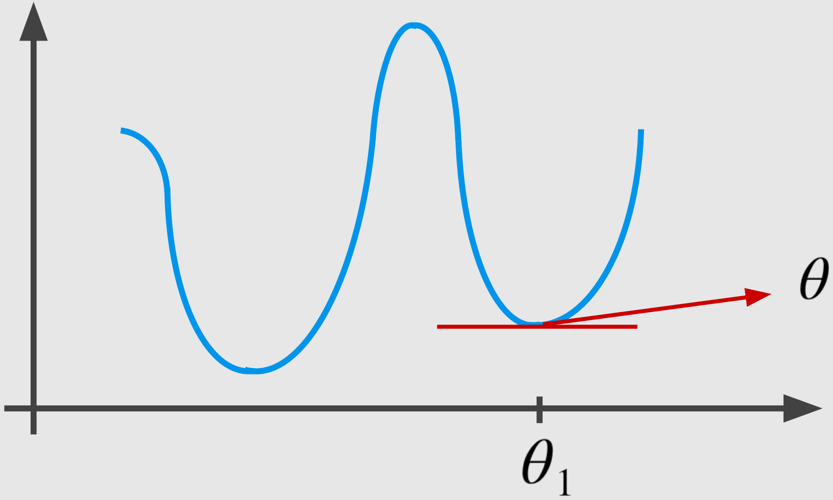


What will one step of gradient descent $\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$ do?



θ_1 at local optima

What will one step of gradient descent $\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$ do?



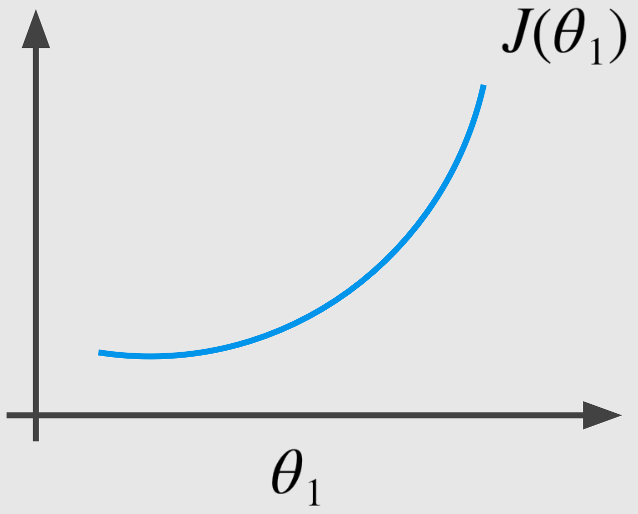
θ_1 at local optima

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

$$= 0$$

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

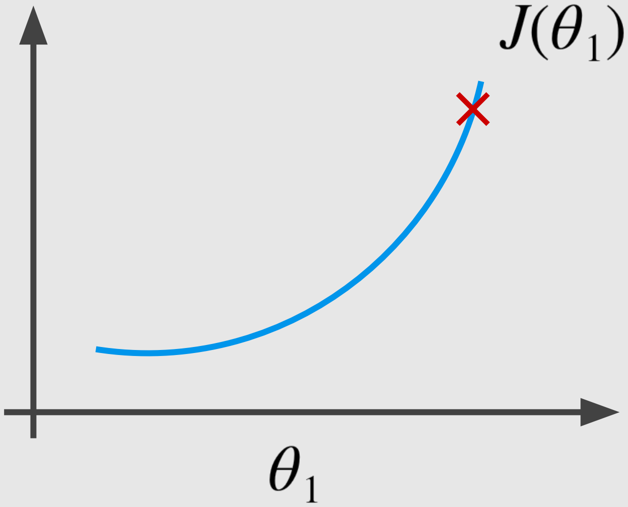
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$



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

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

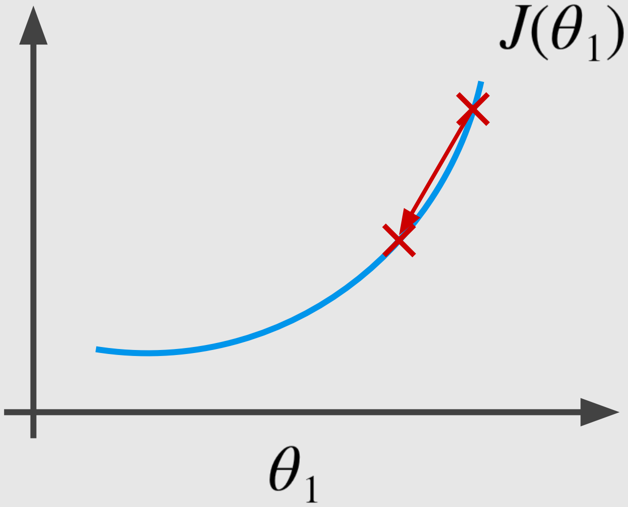
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$



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

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

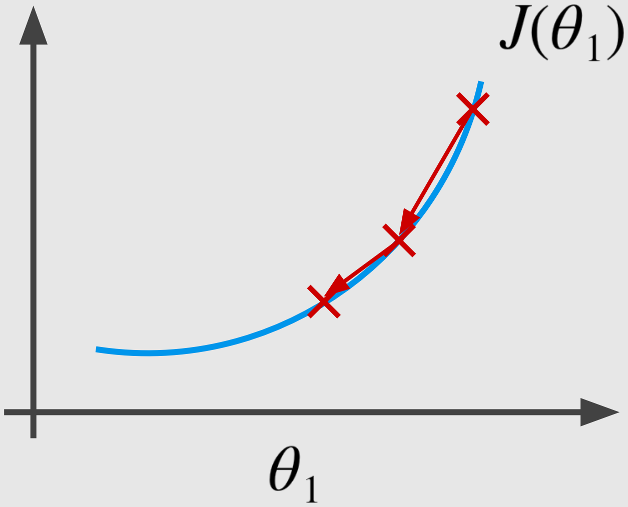
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$



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

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

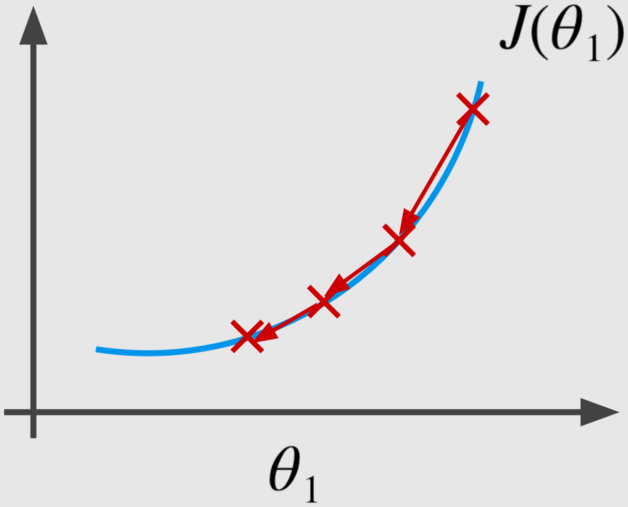
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$



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

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

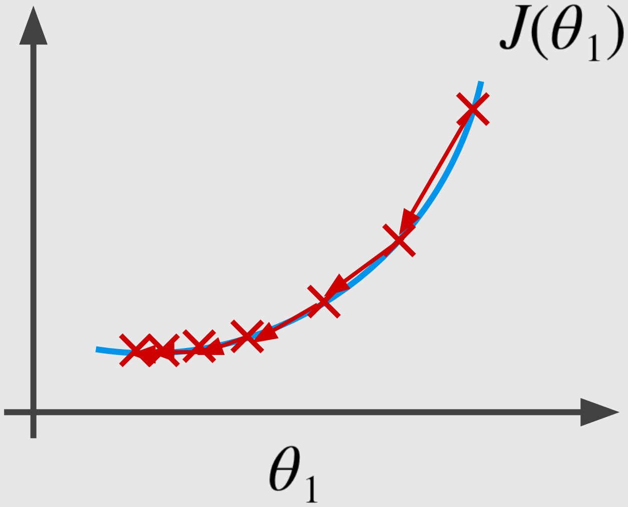
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$



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

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

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



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

Gradient Descent algorithm

repeat until convergence {

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

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

}

Linear Regression Model

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

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

Gradient Descent algorithm

repeat until convergence {

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

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

}

Linear Regression Model

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

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

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

$$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

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

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

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

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

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

$$j = 0: \ \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$$

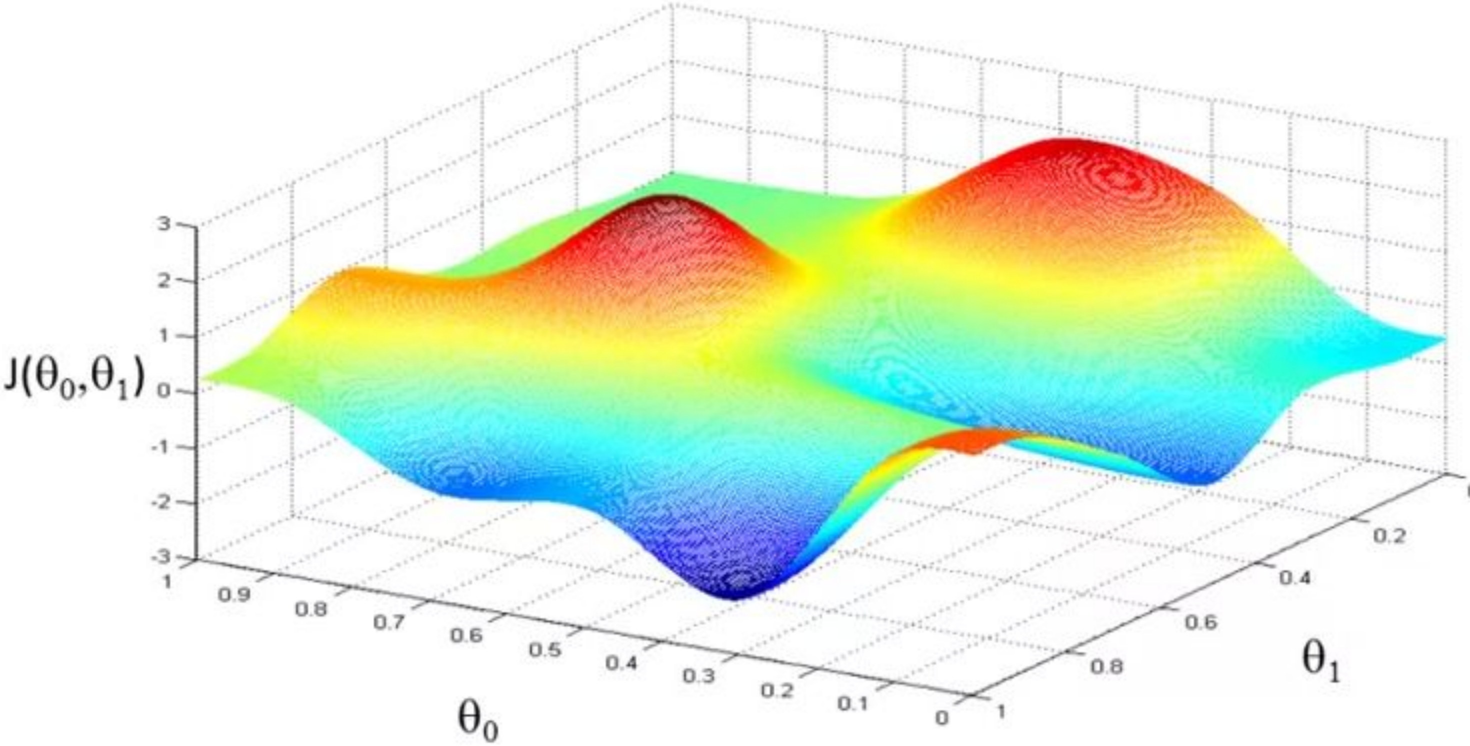
$$j = 1: \ \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})\, x^{(i)}$$

Gradient Descent algorithm

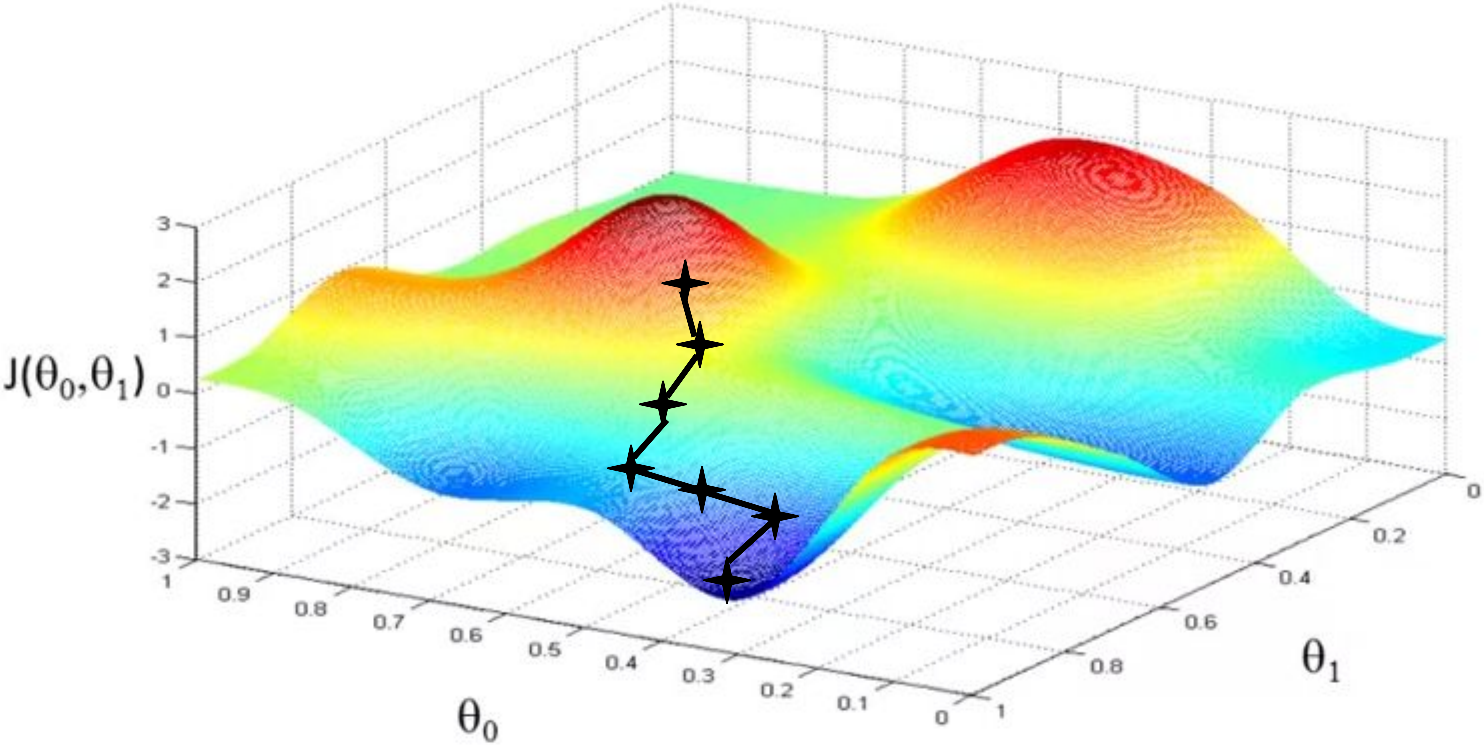
repeat until convergence {

$$\left.\begin{aligned} \theta_0 &:= \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \\ \theta_1 &:= \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})\, x^{(i)} \end{aligned}\right\} \text{update } \theta_0 \text{ and } \theta_1 \text{ simultaneously}$$

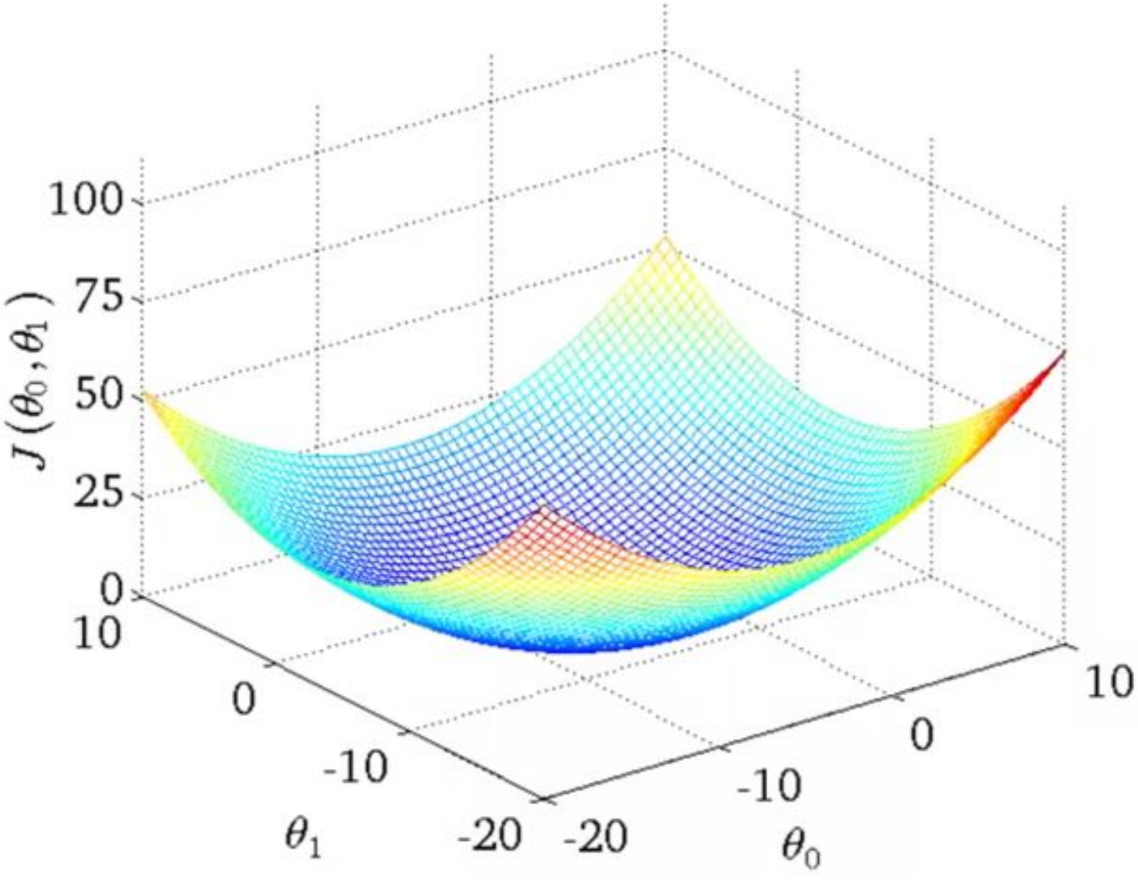
}



3
2
1
0
-1
-2
-3
$J(\theta_0,\theta_1)$
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_0
0
0.2
0.4
0.6
0.8
1
θ_1



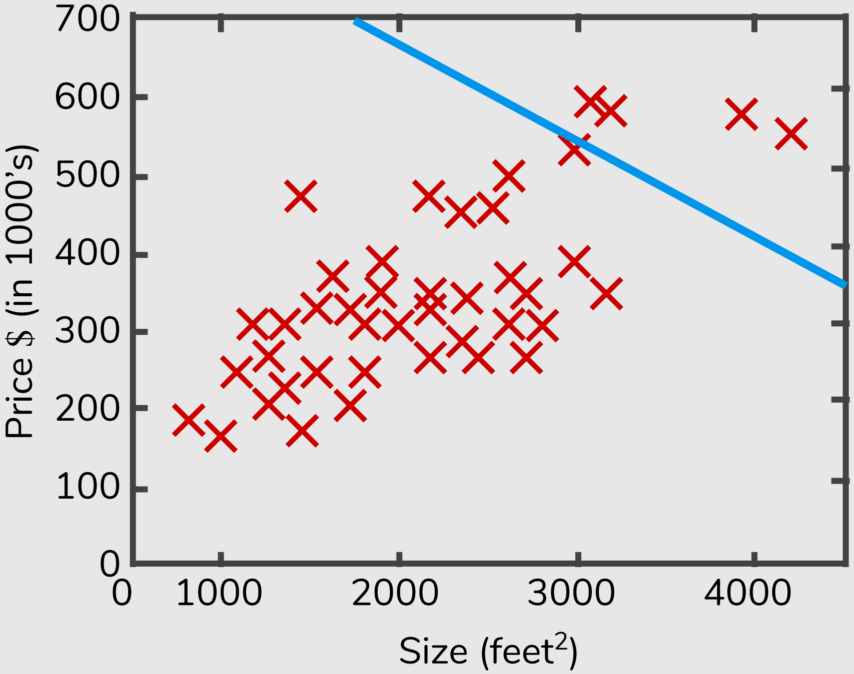
$J(\theta_0,\theta_1)$
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_0
0
0.2
0.4
0.6
0.8
1
θ_1



$J(\theta_0, \theta_1)$
100
75
50
25
0
10
0
-10
-20
θ_1
-20
-10
0
10
θ_0

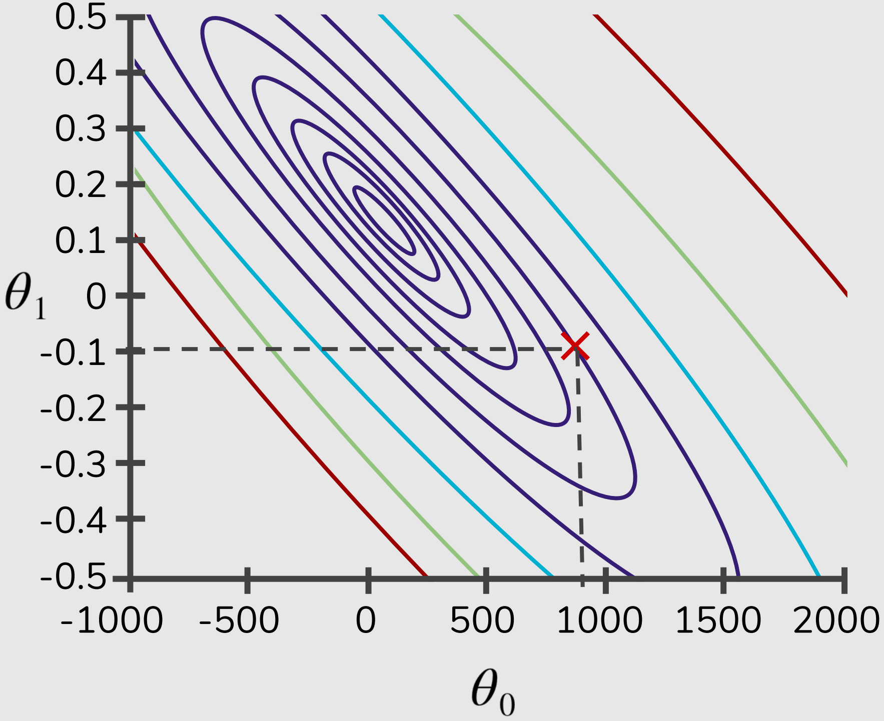
$h_\theta(x)$

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



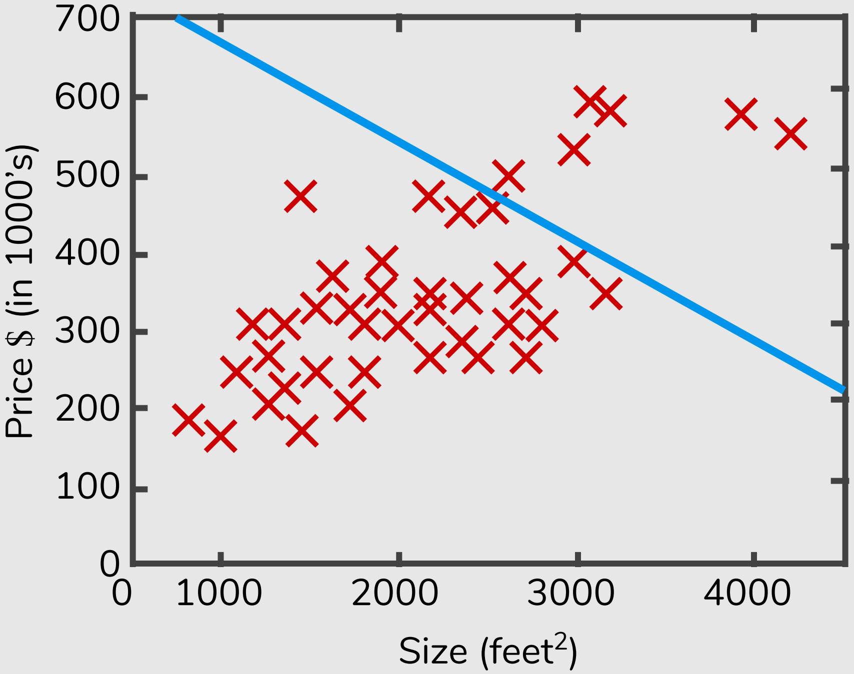
$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



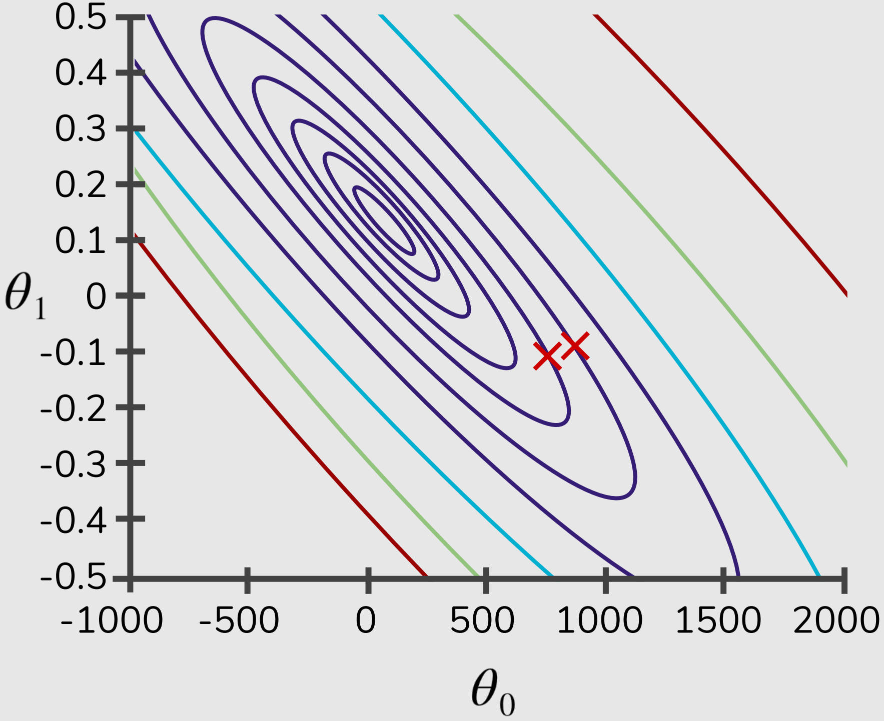
$h_\theta(x)$

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



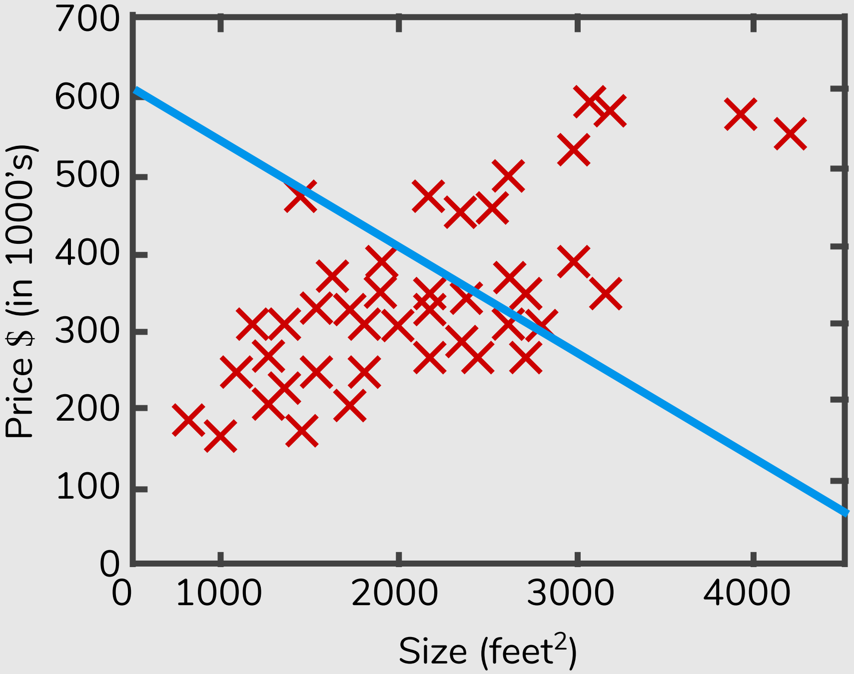
$J(\theta_0,\theta_1)$

(function of the parameters θ_0, θ_1)



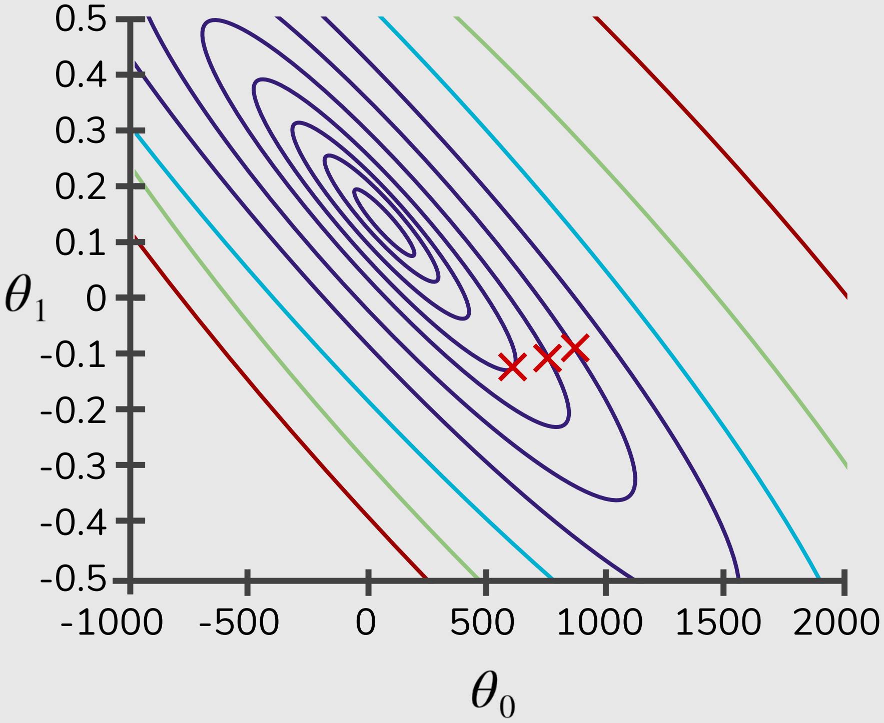
$h_\theta(x)$

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



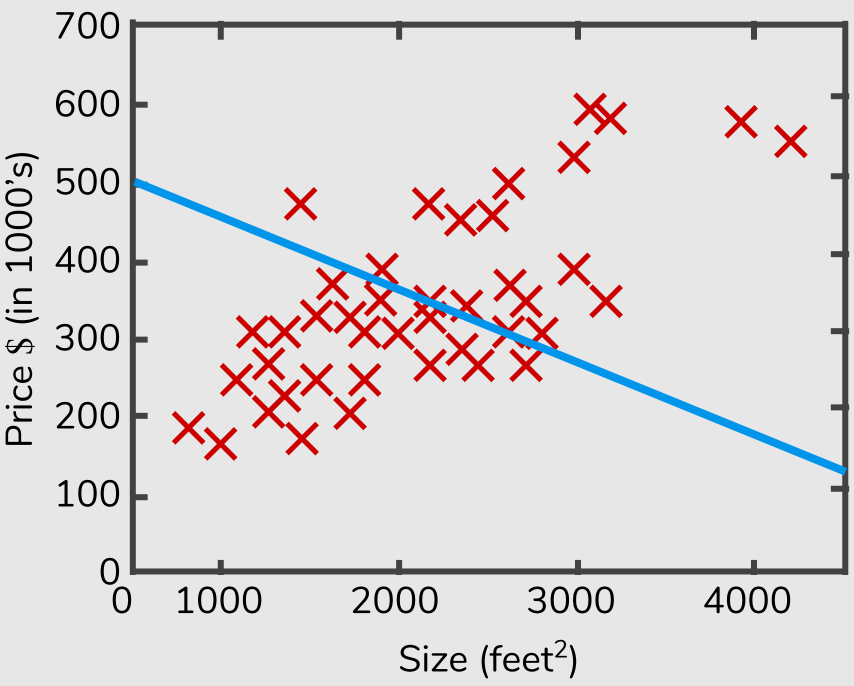
$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



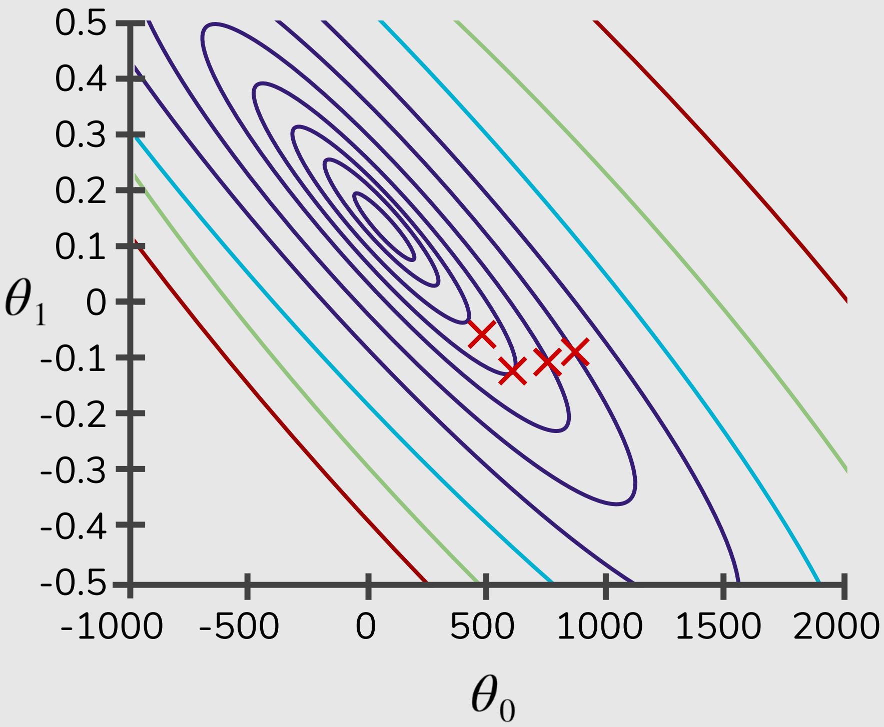
$h_\theta(x)$

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



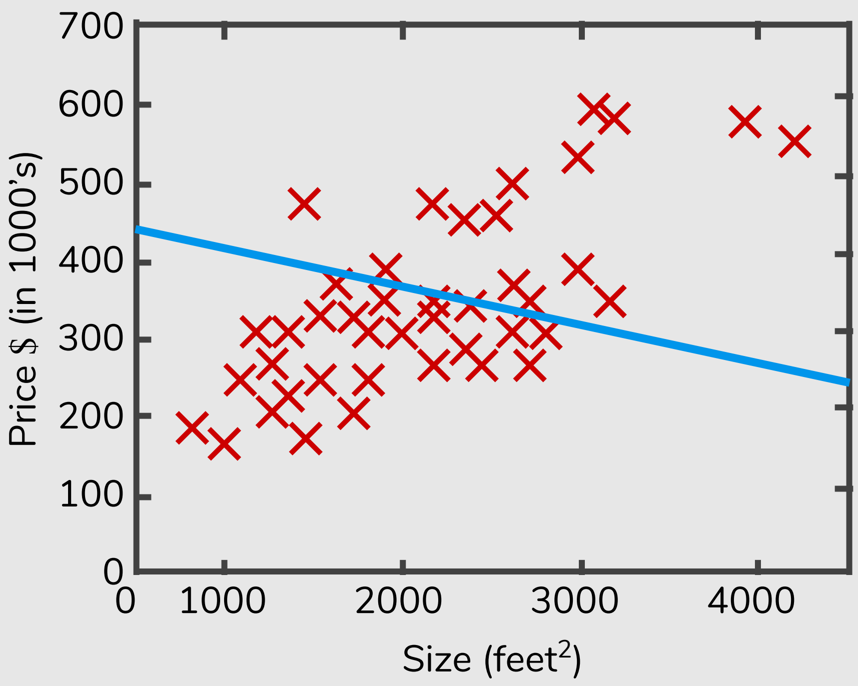
$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



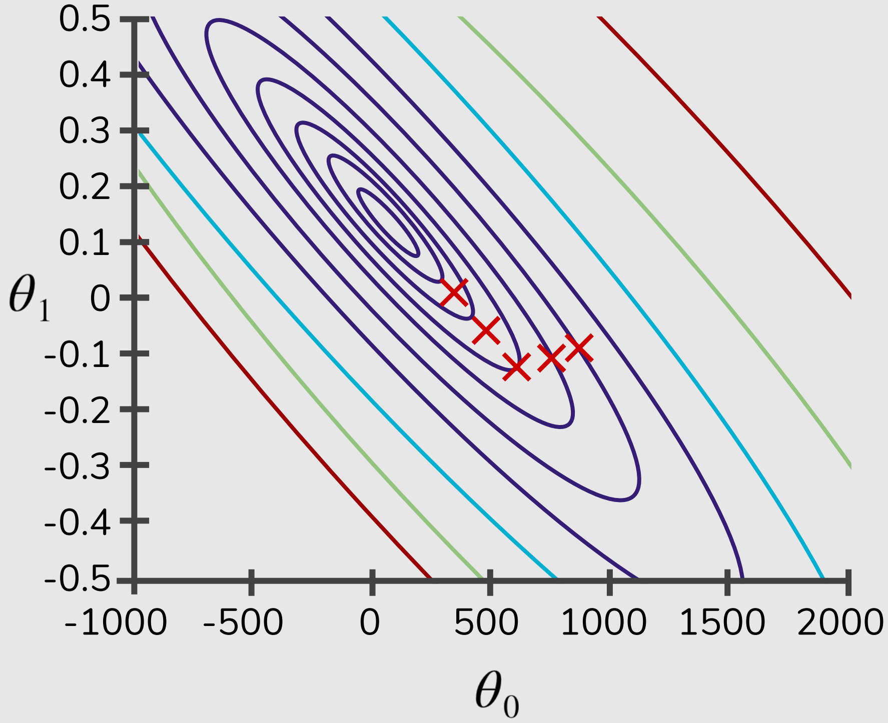
$h_\theta(x)$

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



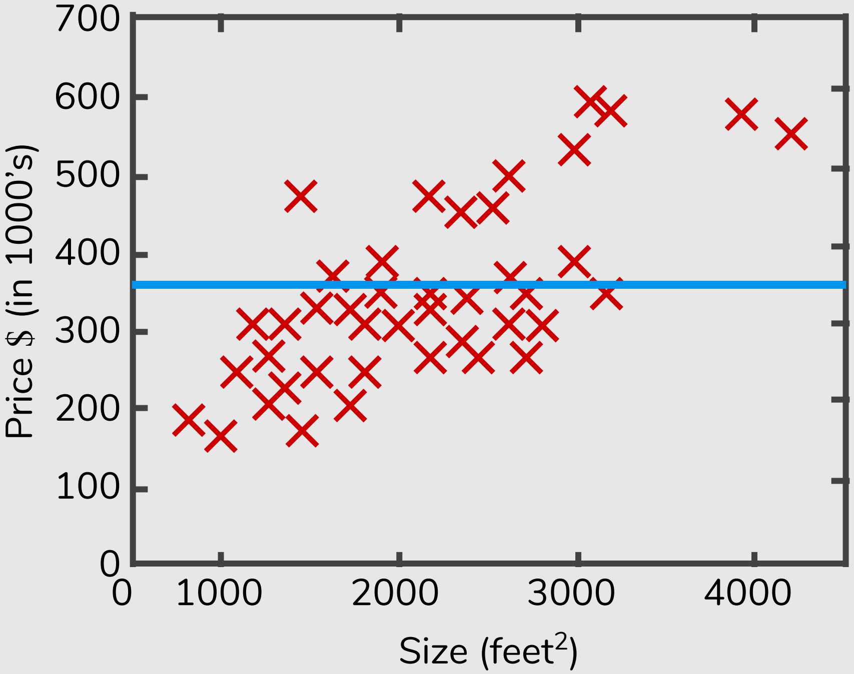
$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



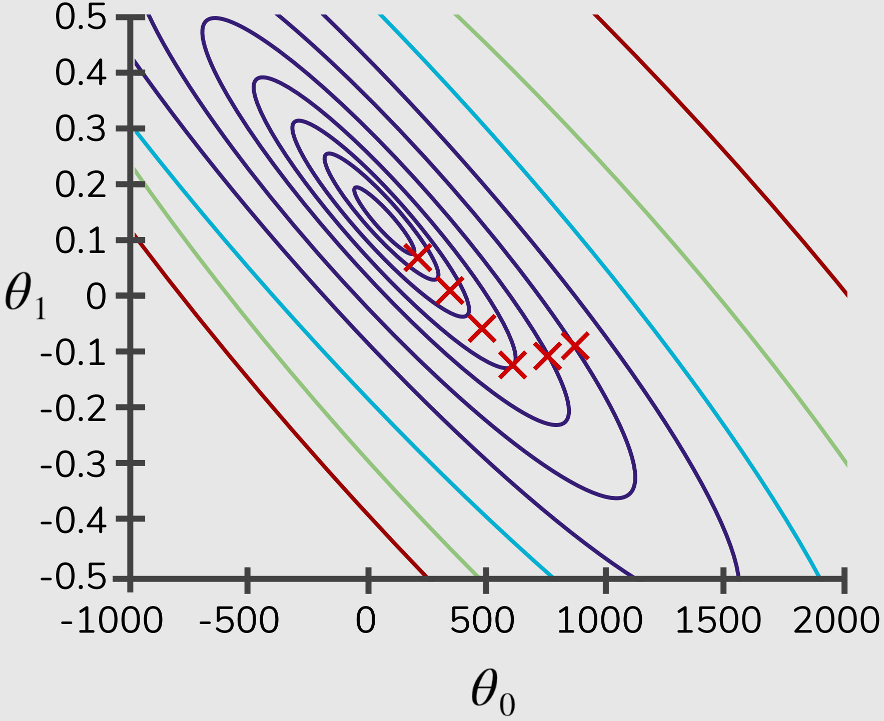
$h_\theta(x)$

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



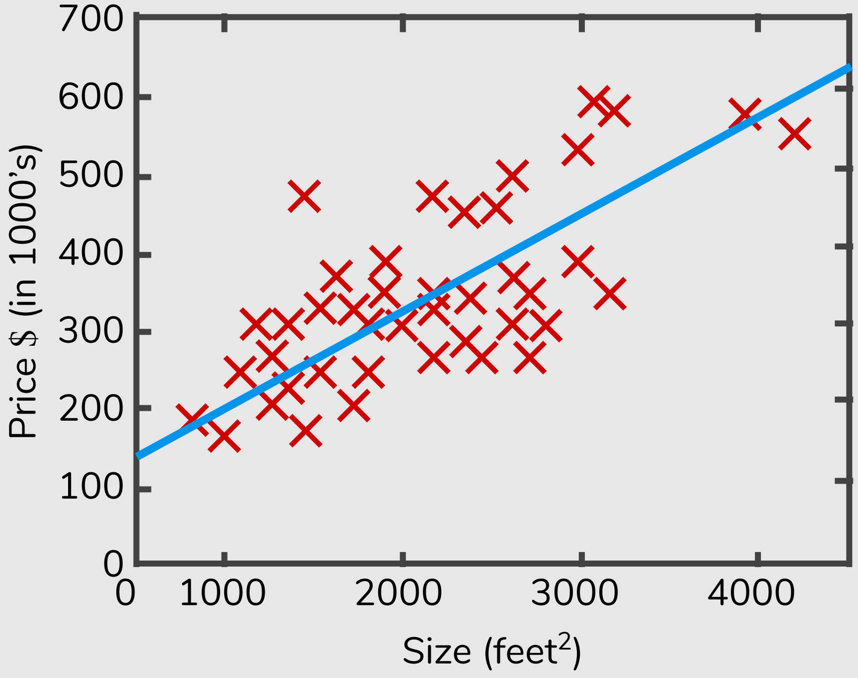
$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



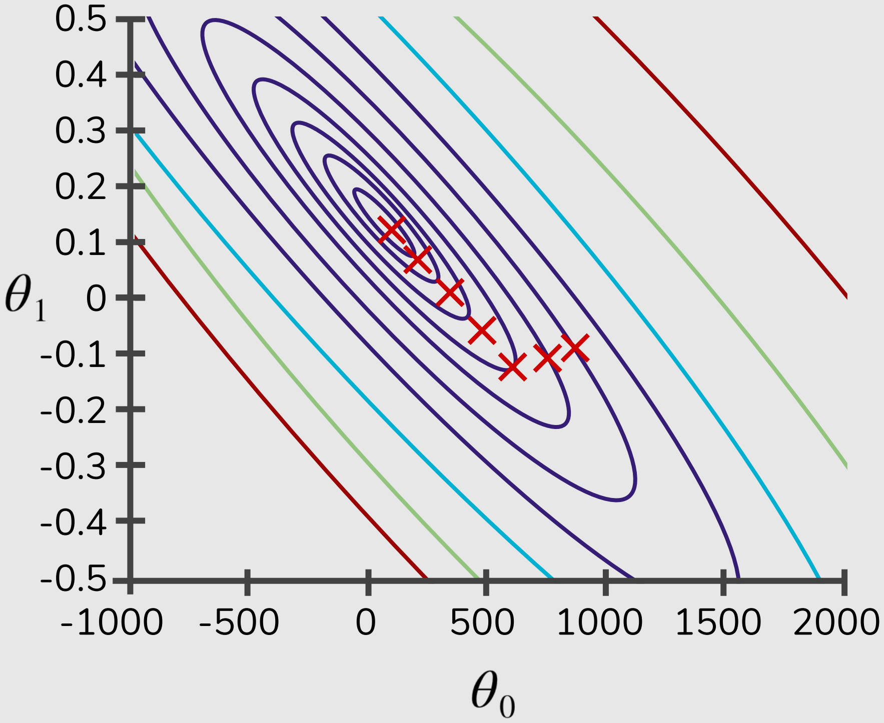
$h_\theta(x)$

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

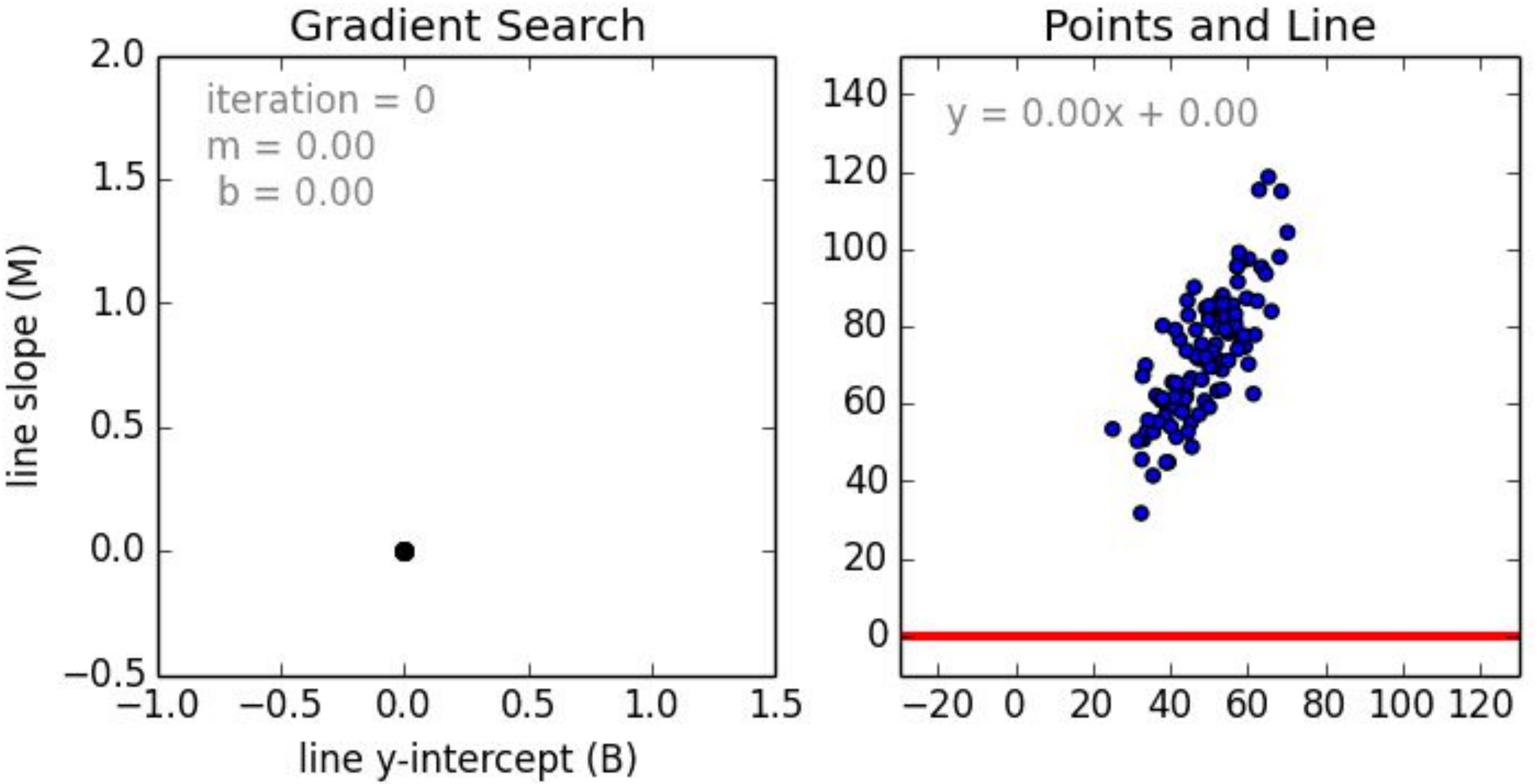


$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

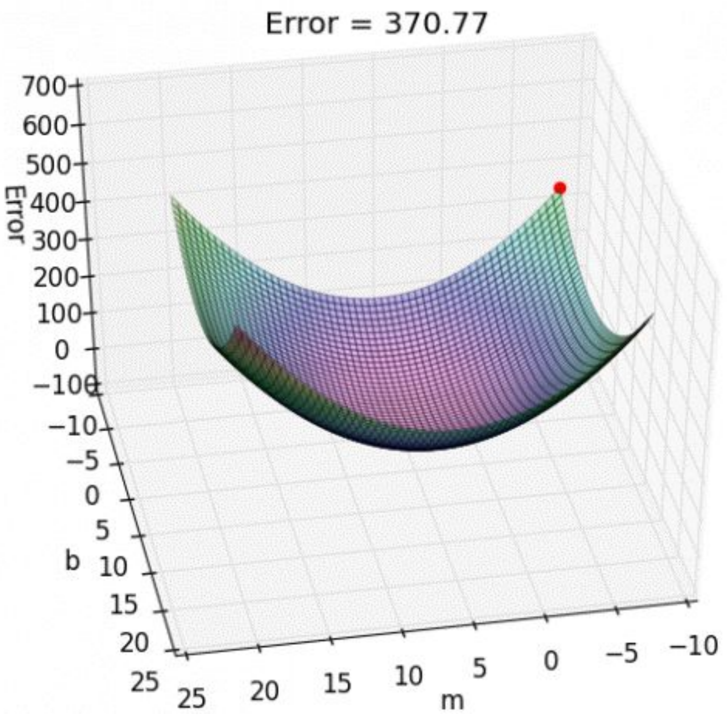


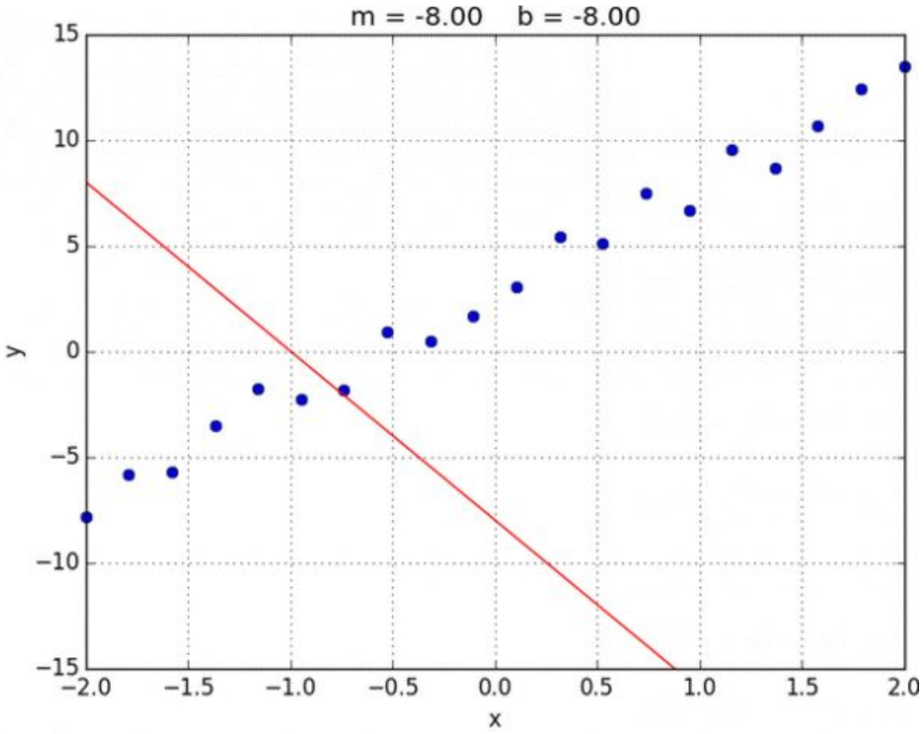
$$h_\theta(x) = \theta_0 + \theta_1 x \Rightarrow y = b + mx$$



Credit: https://github.com/mattnedrich/GradientDescentExample/raw/master/gradient_descent_example.gif

$$y = b + mx$$





Credit: https://alykhantejani.github.io/a-brief-introduction-to-gradient-descent/

"Batch" Gradient Descent

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

“Batch” Gradient Descent

“Batch”: Each step of gradient descent uses **all the training examples**.

- Stochastic Gradient Descent
- Mini-batch Gradient Descent

“Batch” Gradient Descent

repeat until convergence {

$$\left.\begin{aligned}\theta_0 &:= \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \\ \theta_1 &:= \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})\, x^{(i)}\end{aligned}\right\}$$

update θ_0 and θ_1 simultaneously

}

Stochastic Gradient Descent

Each step of gradient descent uses **one training example**.

repeat until convergence {

for $i := 1, \ldots, m$ {

$$\theta_0 := \theta_0 - \alpha(h_\theta(x^{(i)}) - y^{(i)})$$

$$\theta_1 := \theta_1 - \alpha(h_\theta(x^{(i)}) - y^{(i)})x^{(i)}$$

}

}

Mini-batch Gradient Descent

Each step of gradient descent uses b **training examples**.

Say $b = 10$, $m = 1000$.
repeat until convergence {

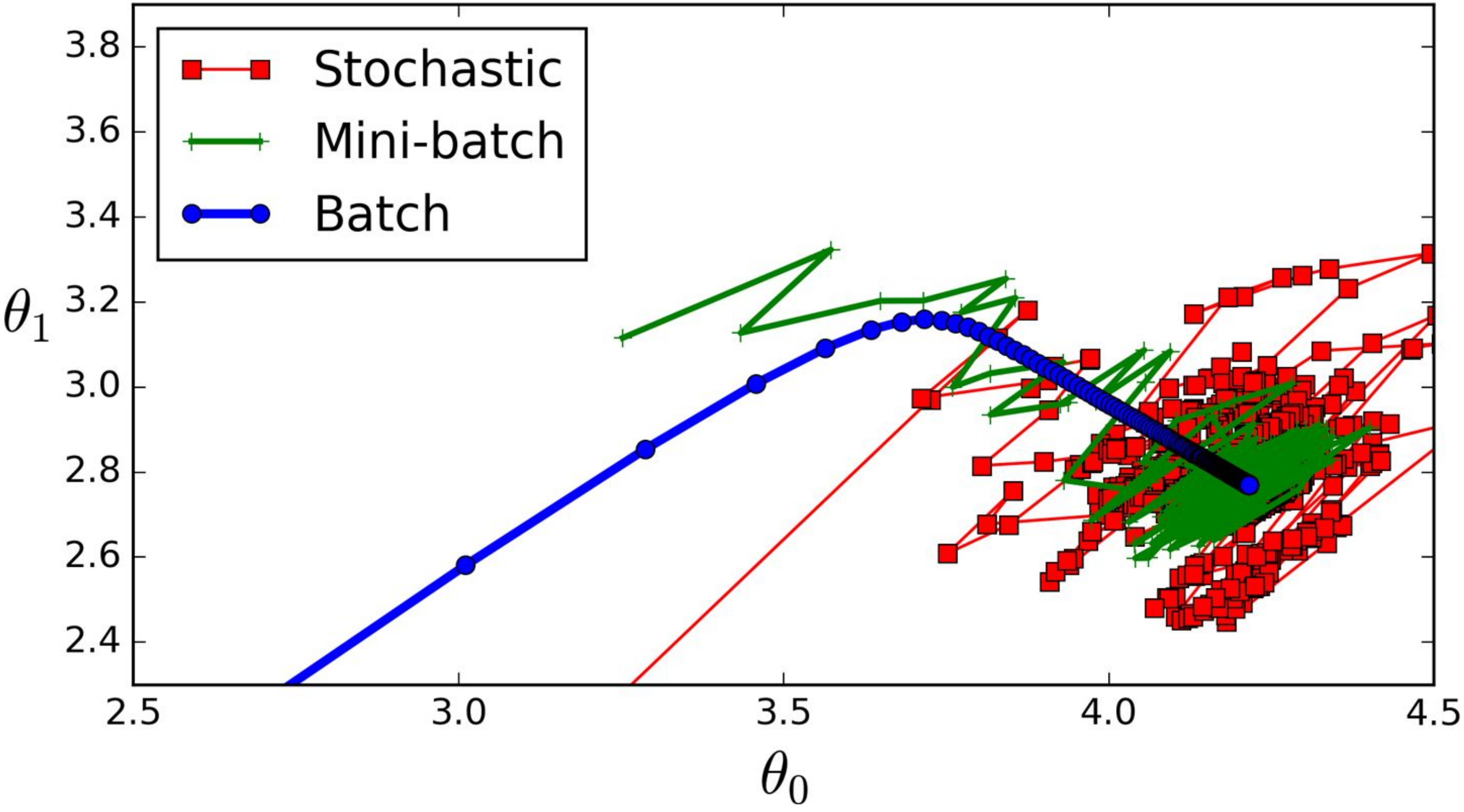
for $i \coloneqq 1, 11, 21 \ldots, 991$ {

$$\theta_0 := \theta_0 - \alpha \frac{1}{10} \sum_{i=k}^{i+9} (h_\theta(x^{(k)}) - y^{(k)})$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{10} \sum_{i=k}^{i+9} (h_\theta(x^{(k)}) - y^{(k)})\, x^{(k)}$$

} }

Batch vs. Stochastic vs. Mini-batch



Linear Regression with multiple variables

Multiple ~~Variables~~ Features

Size in feet^2 (x)	Price $(\$)$ in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	…

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

Multiple ~~Variables~~ Features

Size in feet^2 x_1	Number of bedrooms x_2	Number of floors x_3	Age of home (years) x_4	Price ($) in 1000's y
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	2	36	178
…	…	…	…	…

Notation:

n = number of features

$x^{(i)}$ = input (features) of i^{th} training example

$x_j^{(i)}$ = value of features j in i^{th} training example

Hypothesis

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

Hypothesis

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

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4$$

Hypothesis

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

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4$$

$$h_\theta(x) = 80 + 0.1 x_1 + 10 x_2 + 3 x_3 - 2 x_4$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

For convenience of notation, define $x_0 = 1$.

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

For convenience of notation, define $x_0 = 1$.

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

For convenience of notation, define $x_0 = 1$.

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$$

$$\begin{bmatrix} \theta_0 & \theta_1 & \cdots & \theta_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$$

$$h_\theta(x) = \theta^T x$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

For convenience of notation, define $x_0 = 1$.

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$$

$$\begin{bmatrix} \theta_0 & \theta_1 & \cdots & \theta_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$$

$$h_\theta(x) = \theta^T x$$

Multivariate linear regression.

Hypothesis: $h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

Parameters: $\theta_0, \theta_1, \ldots, \theta_n$

Cost Function: $J(\theta_0, \theta_1, \ldots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$

Gradient Descent:

repeat {

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

} (simultaneously update for every $j = 0, 1, \ldots, n$)

Gradient Descent

Previously ($n = 1$):

repeat {

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

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

(simultaneously update θ_0, θ_1)

}

Gradient Descent

Previously ($n = 1$):

repeat {

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

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

(simultaneously update θ_0, θ_1)

}

New Algorithm ($n \geq 1$):

repeat {

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

(simultaneously update θ_j for $j = 0, 1, \ldots, n$)

}

Gradient Descent

Previously ($n = 1$):

repeat {

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

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

(simultaneously update θ_0, θ_1)

}

New Algorithm ($n \geq 1$):

repeat {

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

(simultaneously update θ_j for $j = 0, 1, \ldots, n$)

}

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)}$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)}$$

...

References

Machine Learning Books

- Hands-On Machine Learning with Scikit-Learn and TensorFlow, Chap. 2 & 4
- Pattern Recognition and Machine Learning, Chap. 3

Machine Learning Courses

- https://www.coursera.org/learn/machine-learning, Week 1 & 2