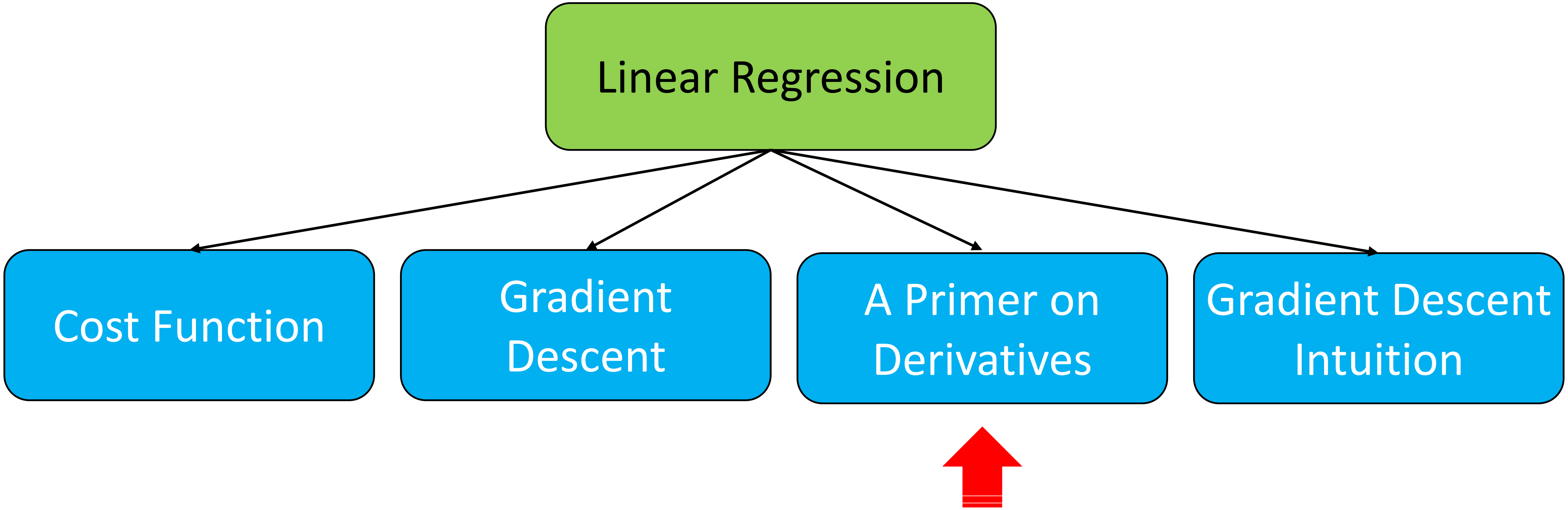
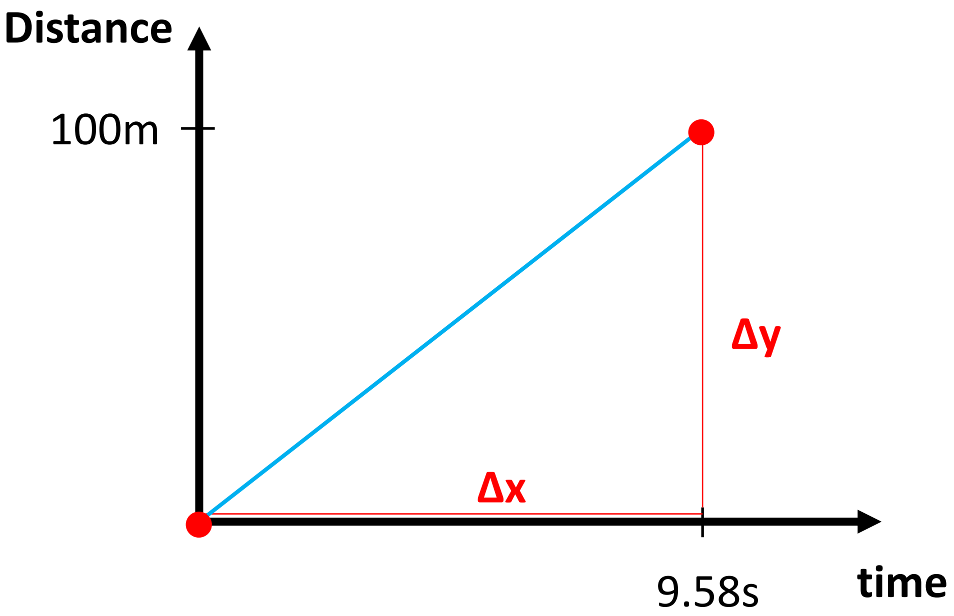
Outline


Linear Regression
Cost Function
Gradient Descent
A Primer on Derivatives
Gradient Descent Intuition

Who is Usain Bolt?

- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!



What is the ***average speed*** of Usain Bolt?

= Change in Distance/Change in Time

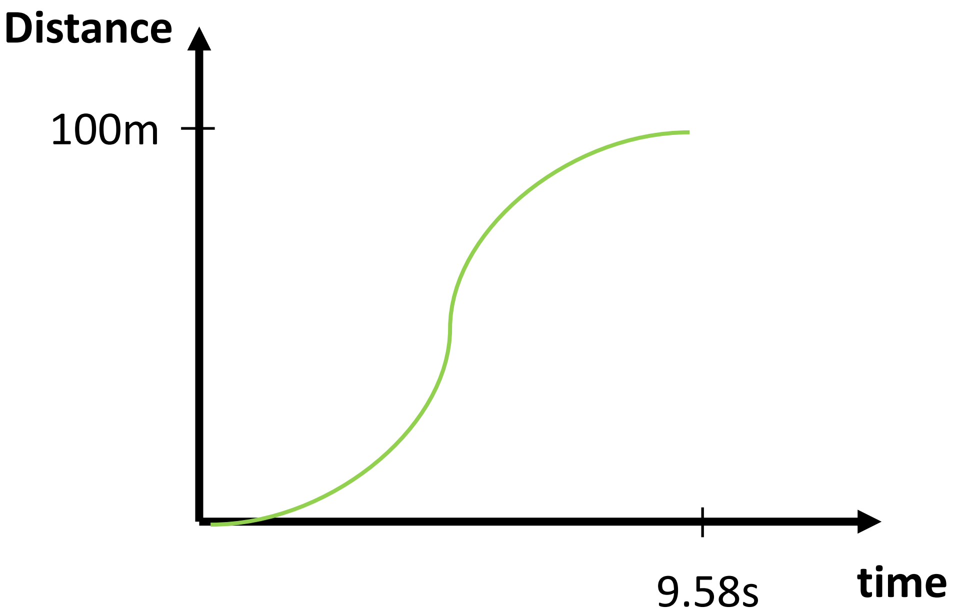
= $\Delta y/\Delta x$

= 100/9.58

= 10.43m/s

Average Speed vs. Instantenous Speed

- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!

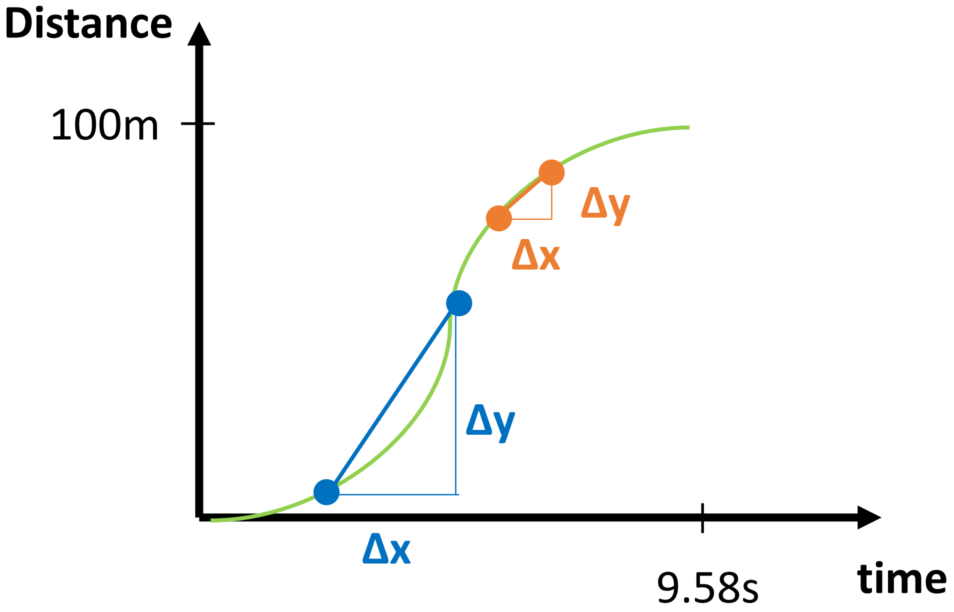


But, this ***average speed*** is different than ***instantenous speed!***

Bolt will not instantly go 100m in 9.58s, but rather start off a little slower, then accelerate, then decelerate a little towards the end

Average Speed vs. Instantenous Speed

- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!

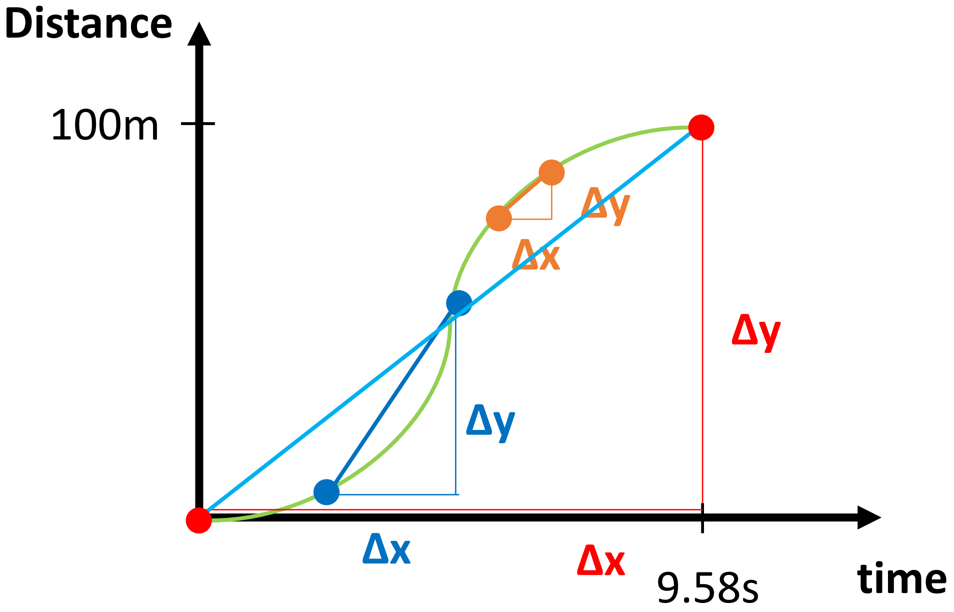


But, this ***average speed*** is different than ***instantenous speed!***

This way, **Δy**/**Δx** != **Δy**/**Δx** (*this is opposite to having a line whereby it does not matter which two points to take on it since the slope will be always the same*)

Average Speed vs. Instantenous Speed

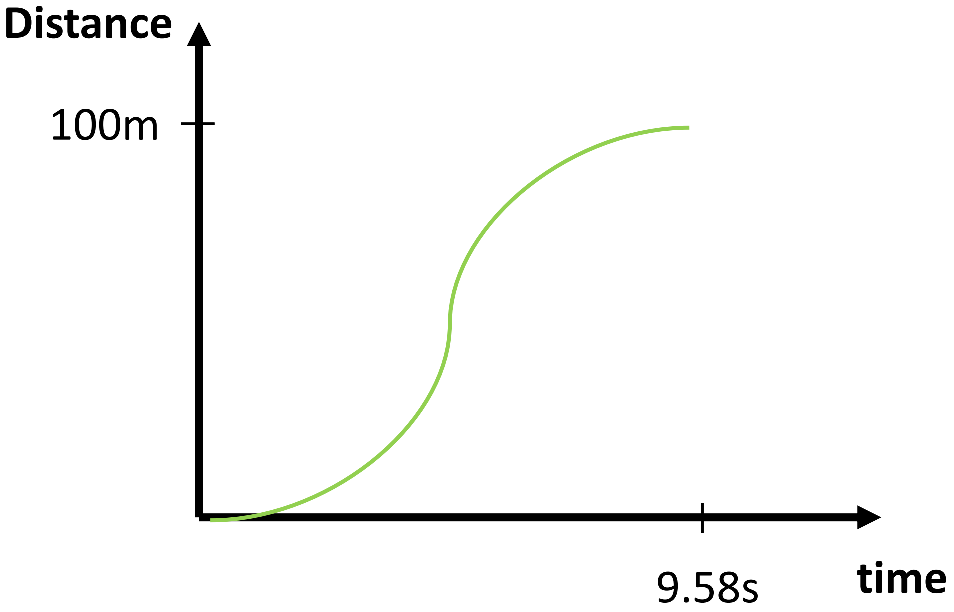
- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!



Consequently, at any given moment in time, a slope on the green function (e.g., **Δy/Δx** or **Δy/Δx**) will be different than the *average slope* on the blue line (i.e., **Δy/Δx**)

Instantenous Speed

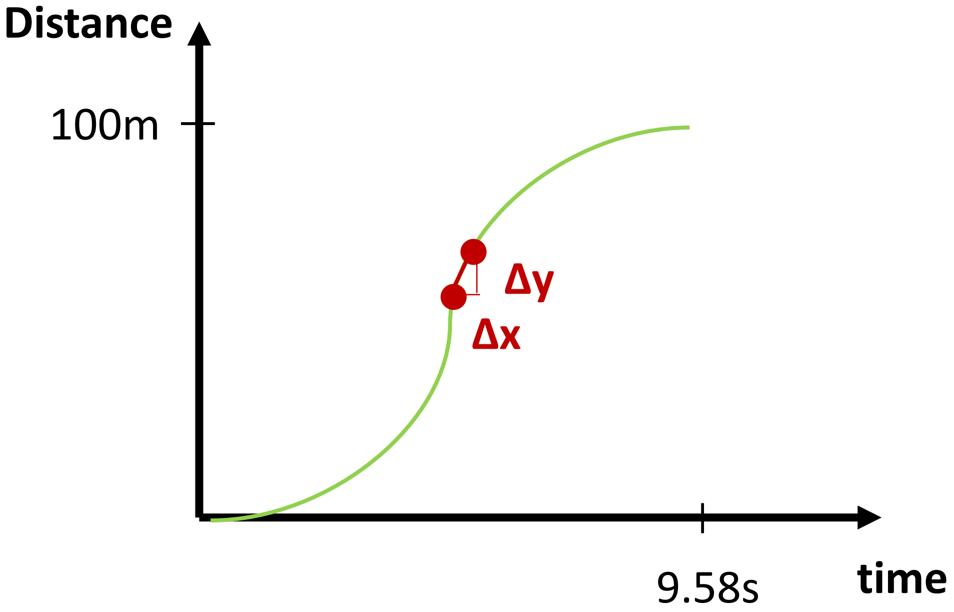
- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!



So, what is Bolt's instantaneous (i.e., NOT average) speed?

Instantenous Speed

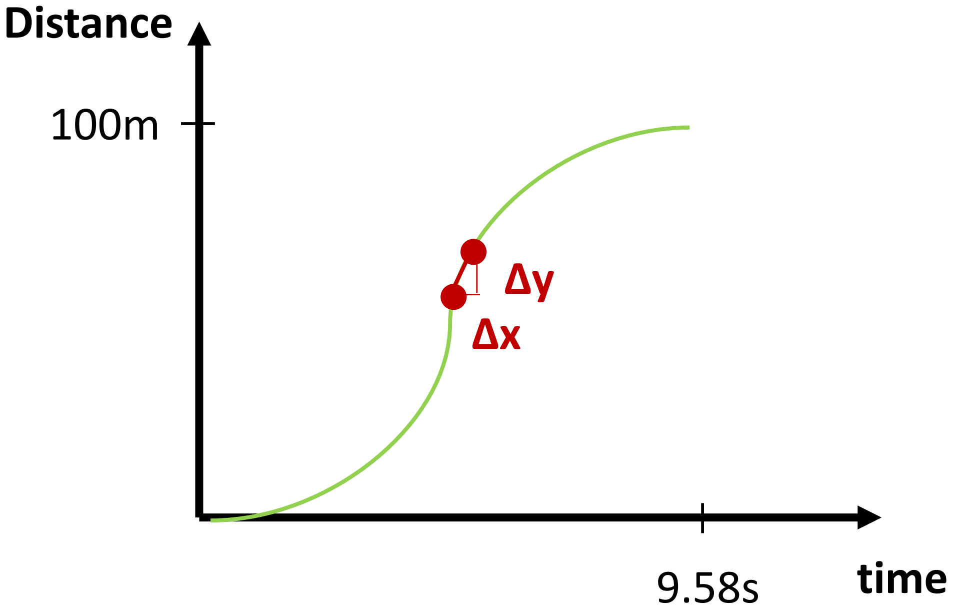
- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!



We can compute the slope around the steepest point if we are interested about the *fastest* instantaneous speed!

Instantenous Speed

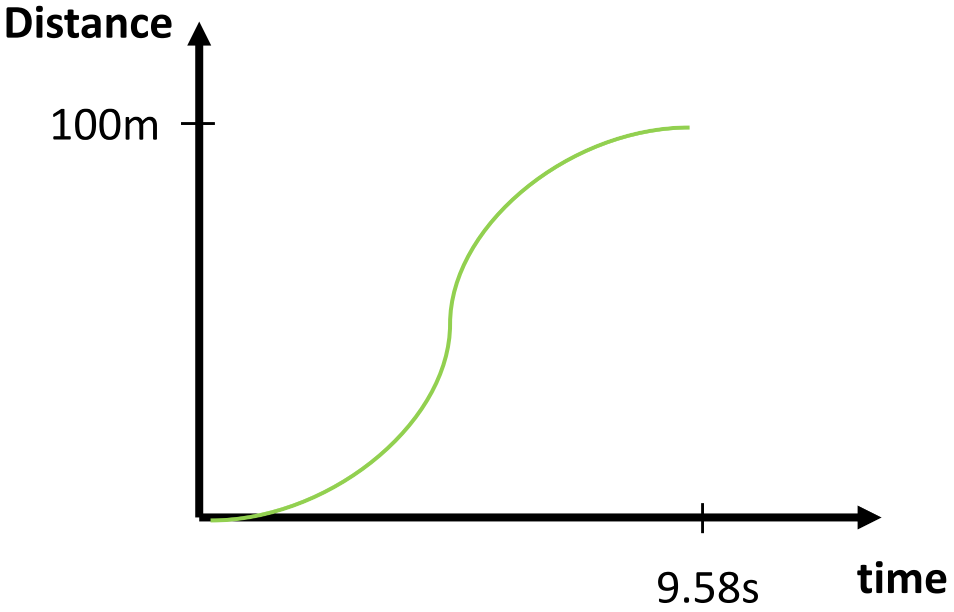
- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!



But that would be only an approximation because the slope of the curve is *constantly* changing

Instantenous Speed

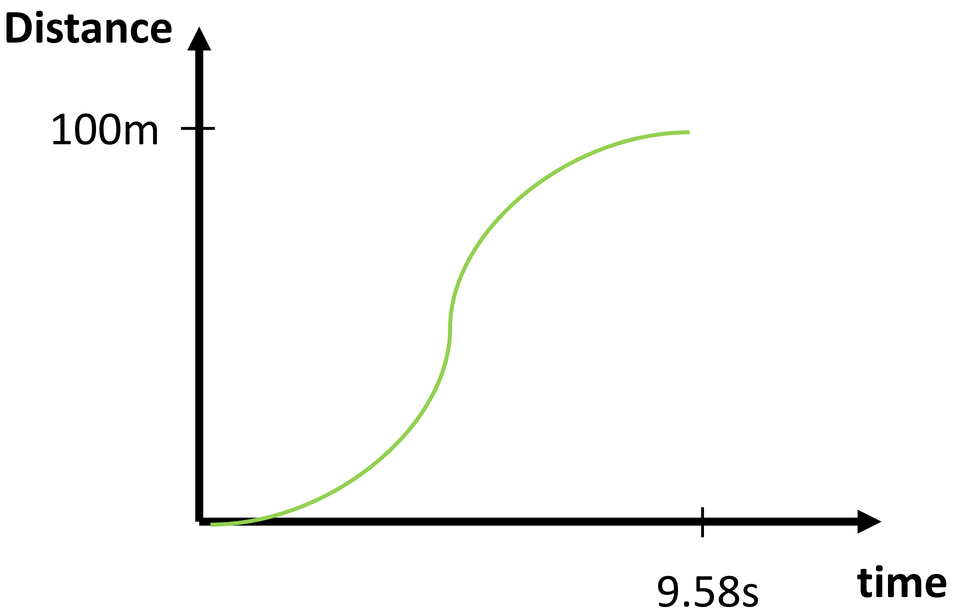
- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!



We can achieve a better approximation by measuring the slope with a smaller & smaller change in x, which yields a smaller & smaller change in y

Instantenous Speed

- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!

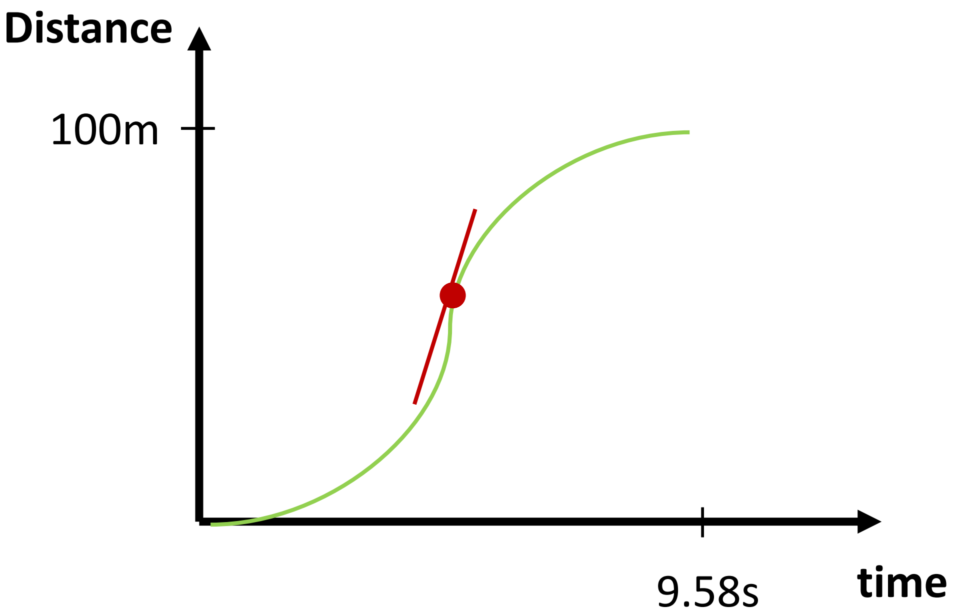


Said differently, we can take the limit of **Δy/Δx** as **Δx** approaches zero:

$$\lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x}$$

Instantenous Speed

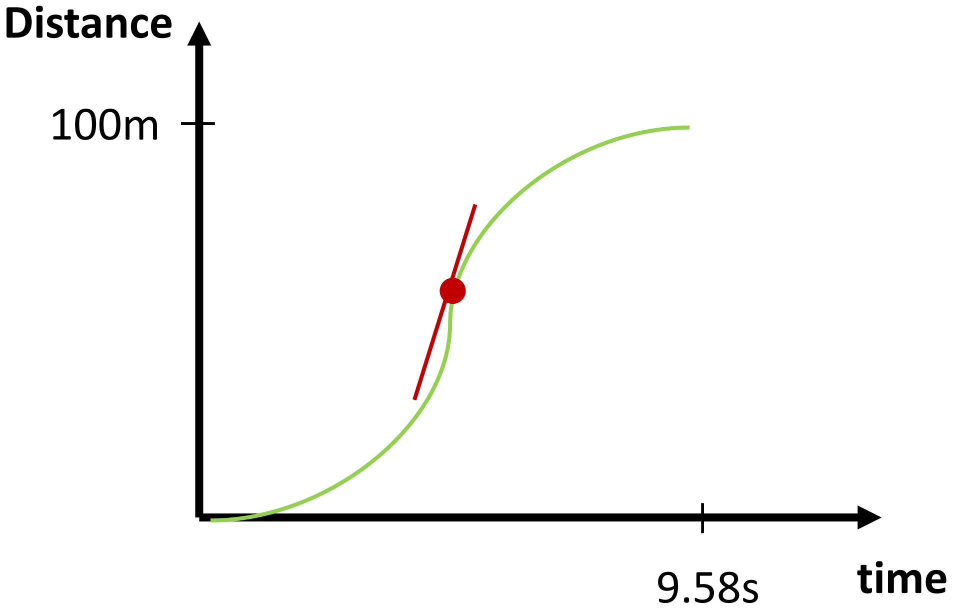
- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!



By doing this, we will approach the instantaneous rate of change (which is the slope of the tangent line – the red line- on the green function)

The Derivative is the Instantaneous Slope

- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!



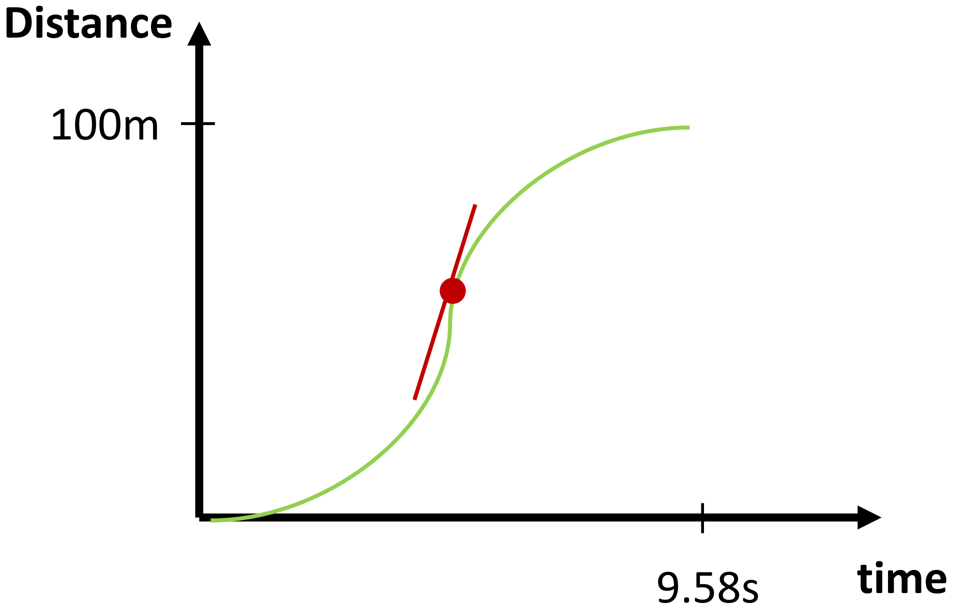
This ***instantaneous slope*** is what mathematicians denote as the ***derivative*** and write as:

$$\lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} = dy/dx$$

This is an infinitely small change in y (***d*** stands for ***differential***)

The Derivative is the Instantaneous Slope

- Usain Bolt is regarded widely as the greatest sprinter of all time
  - He can run 100meters in 9.58seconds!



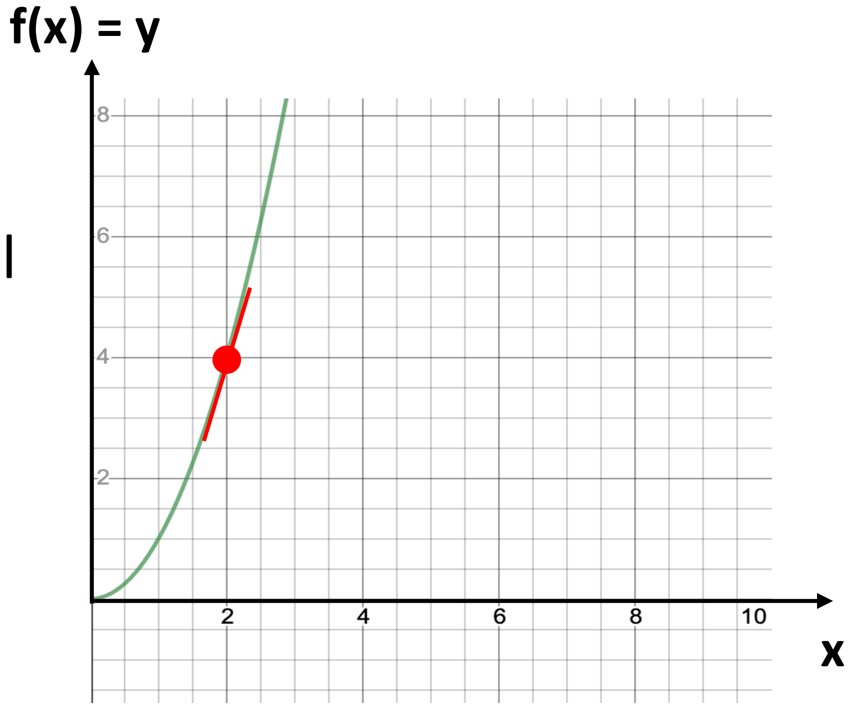
This ***instantaneous slope*** is what mathematicians denote as the ***derivative*** and write as:

$$\lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} = dy/dx$$

And, this is an infinitely small change in x (***d*** stands for ***differential***)

Derivative of a Univariate Function

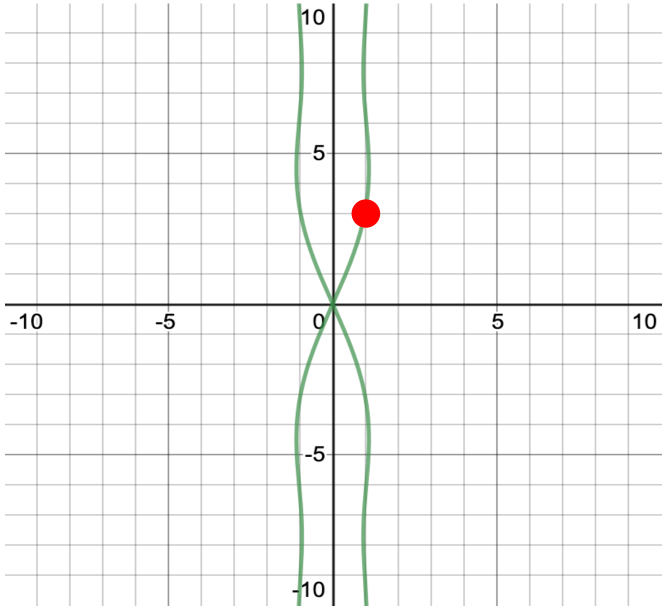
- What is the instantaneous rate of change at a point (say, **2**) on a function (say, $f(x) = x^2$)?
  - It is the slope of the tangent line at point **2**
  - Which is the derivative at point **2**
  - Which is "a super small change in y"/ "a super small change in x" at point **2**
  - Which is $\lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x}(\mathbf{2})$
  - Which is $\frac{dx}{dy}(2)$



Derivative of a Multivariate Function

- What is the instantaneous rate of change at a point (*say*, (**1**, **3**)) on a function that involves multiple variables (*say*, $f(x, y) = x^2y + \sin(y)$)?
  - It is the ***partial*** derivative at point (**1**, **3**)
  - Which can be computed as:
    - The derivative of $f(x, y)$ with respect to x while y is held constant:
      - $\frac{\partial f}{\partial x}(\mathbf{1}, \mathbf{3}) = \frac{\partial f}{\partial x}(x^2 . 3 + \sin(3))$ = 2x.3 + 0 = 6x = 6
    - And the derivative of $f(x, y)$ with respect to y while x is held constant:
      - $\frac{\partial f}{\partial y}(\mathbf{1}, \mathbf{3}) = \frac{\partial f}{\partial y}(1^2 . y + \sin(y))$ = 1 + cos(y) = 1 + cos(3)

We do not use *d* with multi-variable functions, but rather ∂



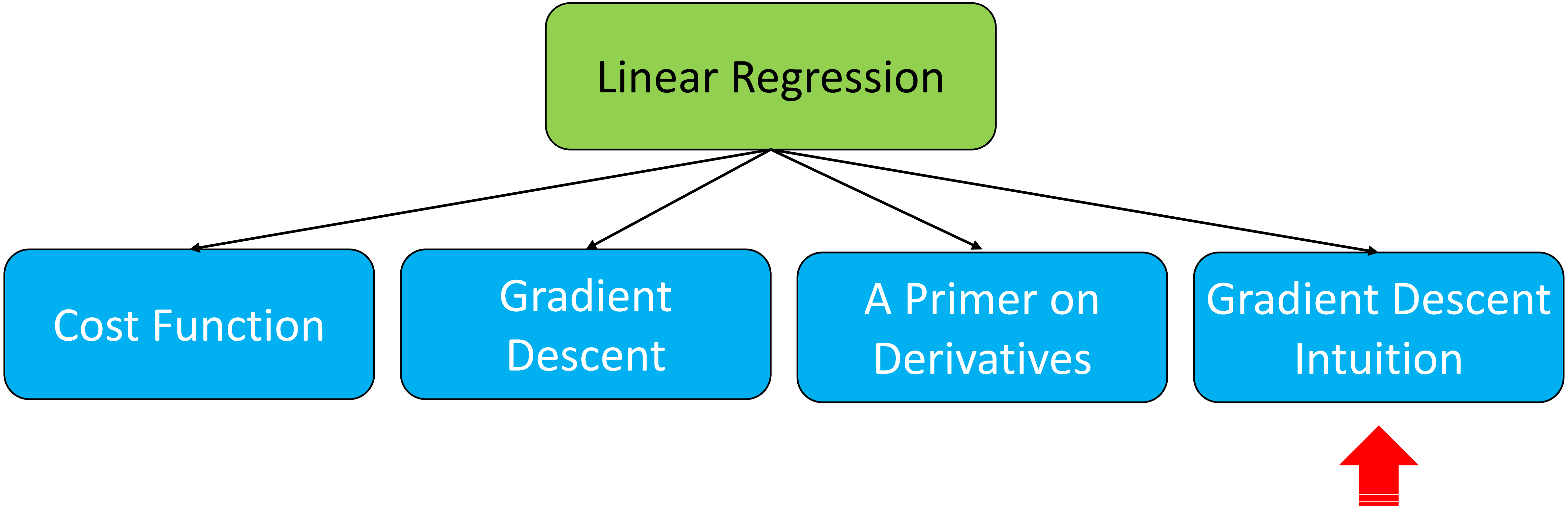
Gradient

- ***Gradient*** is a way of packing together all the partial derivative information of a function
  - Consider $f(x, y) = x^2 y + \sin(y)$
    - $\frac{\partial f}{\partial x} = 2xy$
    - $\frac{\partial f}{\partial y} = x^2 + cos(y)$
  - Gradient puts these two partial derivatives together in a vector as follows:

Called *nabla*,
but often pronouced
as *del* or *gradient*

$$\nabla f(x, y) = \nabla\, x^2 y + \sin(y) = \begin{bmatrix} 2xy \\ x^2 + \cos(y) \end{bmatrix}$$

Outline


Linear Regression
Cost Function
Gradient Descent
A Primer on Derivatives
Gradient Descent Intuition

Gradient Descent For Linear Regression

- **Outline**:
  - Have some cost function $\boldsymbol{J(\theta_0, \dots, \theta_{n-1})}$
  - Start off with some guesses for $\theta_0, \dots, \theta_{n-1}$
    - It does not really matter what values you start off with, but a common choice is to set them all initially to zero
  - Keep changing $\theta_0, \dots, \theta_{n-1}$ to reduce $\boldsymbol{J(\theta_0, \dots, \theta_{n-1})}$ until we hopefully end up at a minimum location
    - When you are at a certain position on the surface of $\boldsymbol{J}$, look around, then take a little step in the direction of *the steepest descent*, then repeat

Gradient Descent For Linear Regression

- **Outline**:
  - Have some cost function $\boldsymbol{J(\theta_0, \dots, \theta_{n-1})}$
  - Start off with some guesses for $\theta_0, \dots, \theta_{n-1}$
    - It does not really matter what values you start off with, but a common choice is to set them all initially to zero
  - Repeat until convergence{

Partial Derivative

$$\theta_j = \theta_j - \alpha \frac{\partial \boldsymbol{J(\theta_0, \dots, \theta_{n-1})}}{\partial \theta_j}$$

}

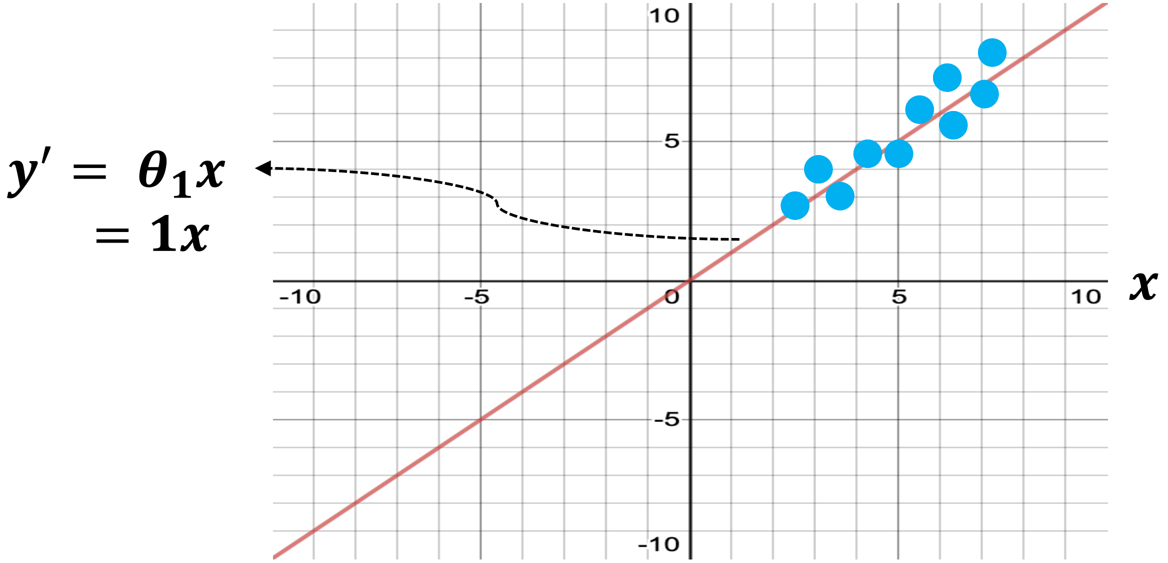
Learing Rate

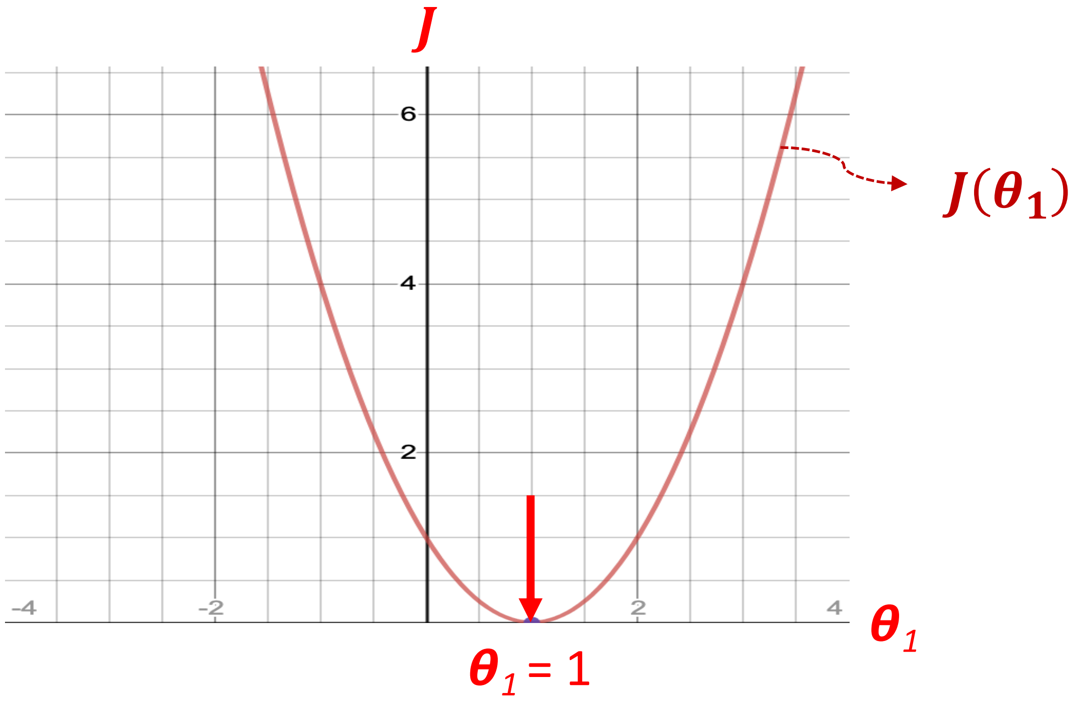
What do α and ∂ do?

The Impact of Partial Derviative

- For simplicity, let us assume our optimization objective is to $\underset{\theta_0,\theta_1}{\text{minimize}}\, J(\theta_1)$, thus, $\boldsymbol{\theta}_0 = 0$

$y' = h_\theta(x)$



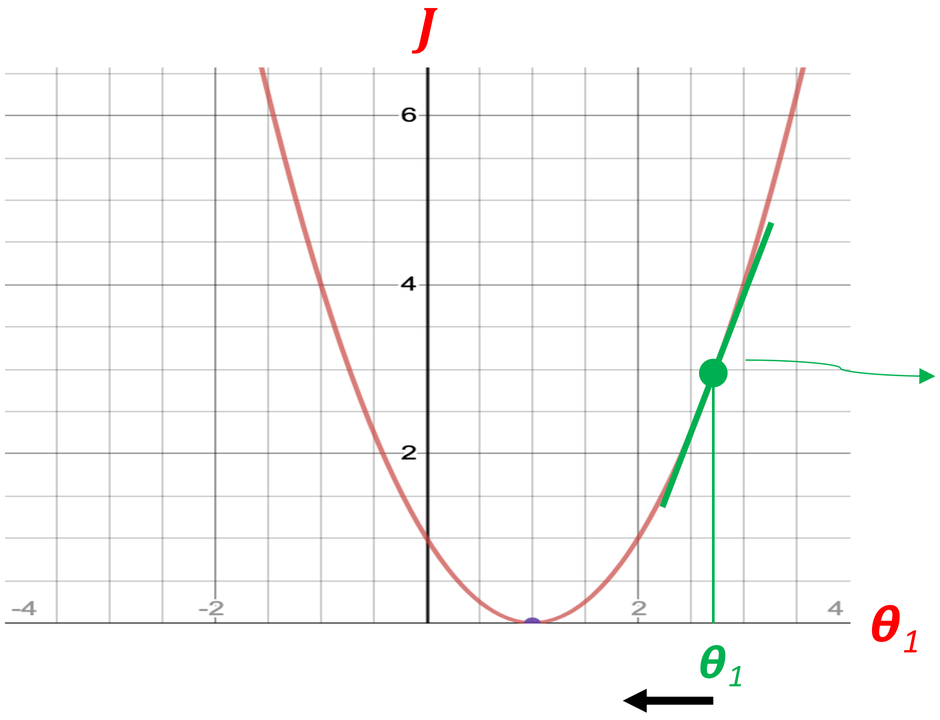


$h_\theta(x)$ is the ***Hypothesis Function***

$J(\theta_1)$ is the ***Cost Function***

The Impact of Partial Derviative

- For simplicity, let us assume our optimization objective is to $\underset{\theta_0,\theta_1}{\text{minimize}}\, \boldsymbol{J(\theta_1)}$, thus, $\boldsymbol{\theta}_0 = 0$

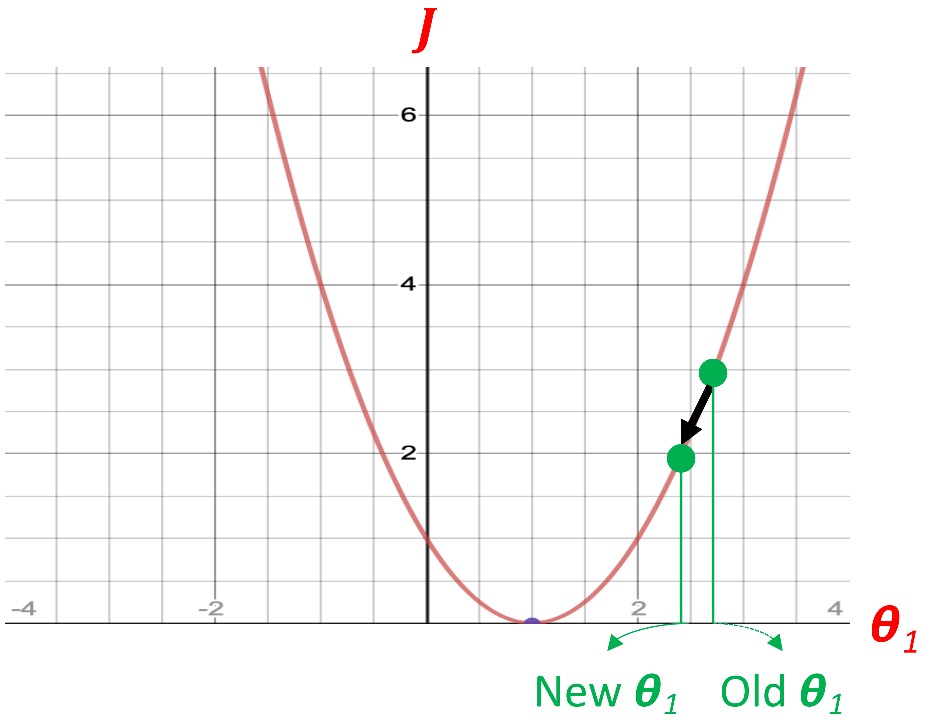


$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}$$

$$\underbrace{= \theta_1 - \alpha\, (Positive\ Number)}_{\text{Decrease } \theta_1 \text{ by a certain value}}$$

The Impact of Partial Derviative

- For simplicity, let us assume our optimization objective is to $\underset{\theta_0,\theta_1}{\text{minimize}}\, \boldsymbol{J(\theta_1)}$, thus, $\boldsymbol{\theta}_0 = 0$



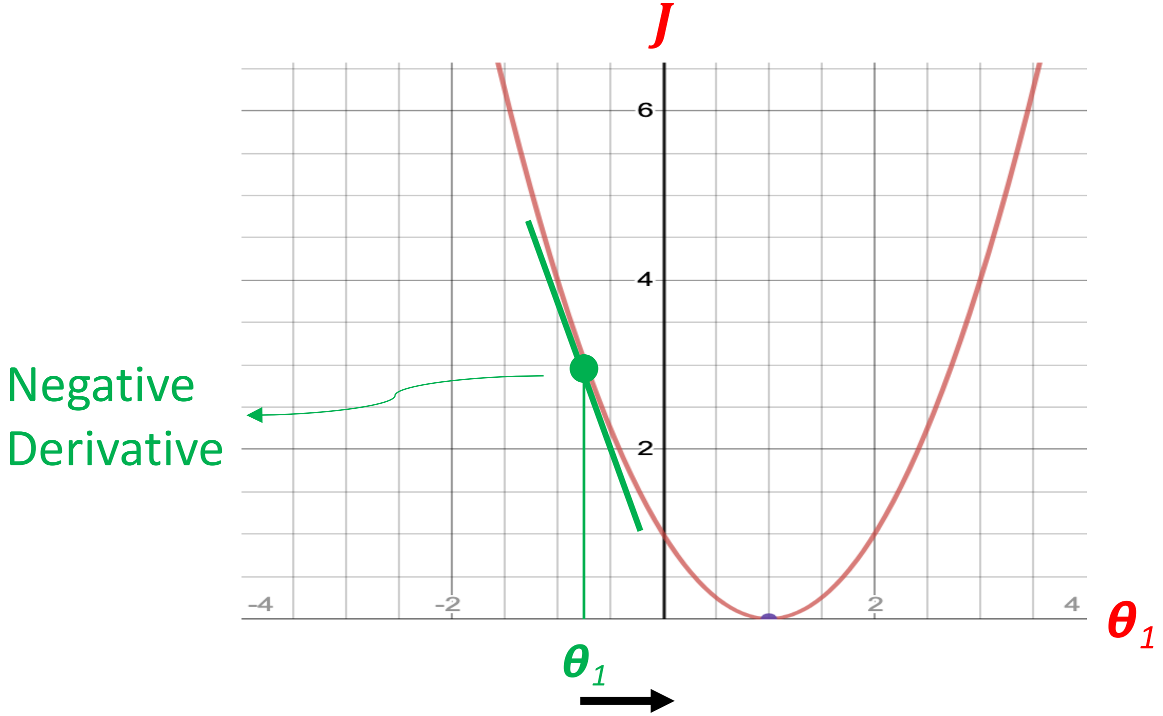
$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}$$

$$\underbrace{= \theta_1 - \alpha\, (Positive\ Number)}_{}$$

Decrease θ_1 by a certain value

The Impact of Partial Derviative

- For simplicity, let us assume our optimization objective is to $\underset{\theta_0,\theta_1}{\text{minimize}}\, \boldsymbol{J(\theta_1)}$, thus, $\boldsymbol{\theta}_0 = 0$



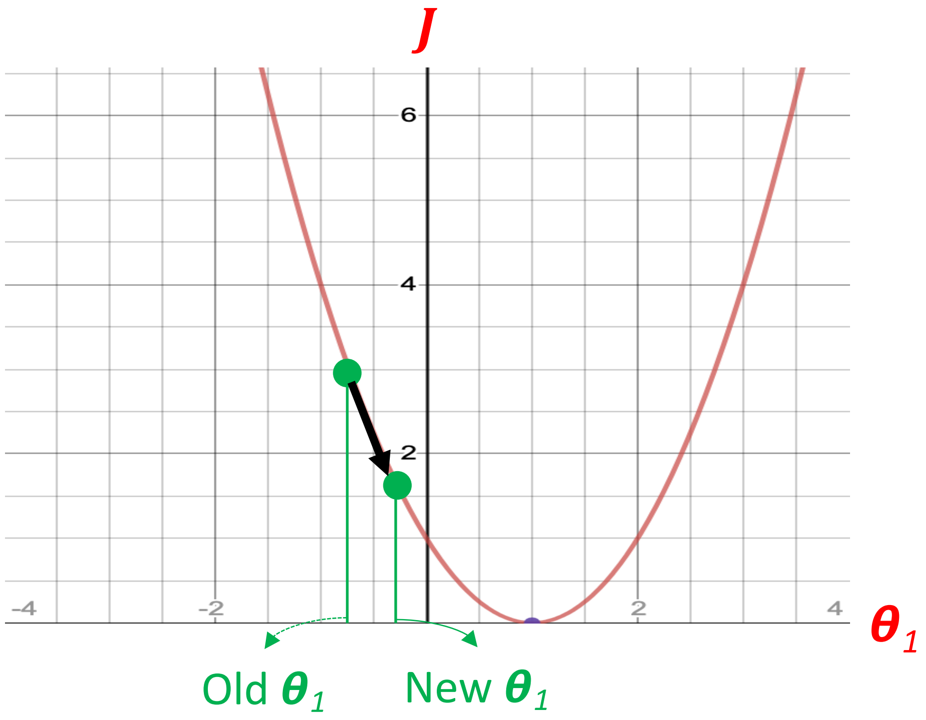
$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}$$

$$\underbrace{= \theta_1 - \alpha\, (Negative\ Number)}_{}$$

Increase θ_1 by a certain value

The Impact of Partial Derviative

- For simplicity, let us assume our optimization objective is to $\underset{\theta_0,\theta_1}{\text{minimize}}\, \boldsymbol{J(\theta_1)}$, thus, $\boldsymbol{\theta}_0 = 0$

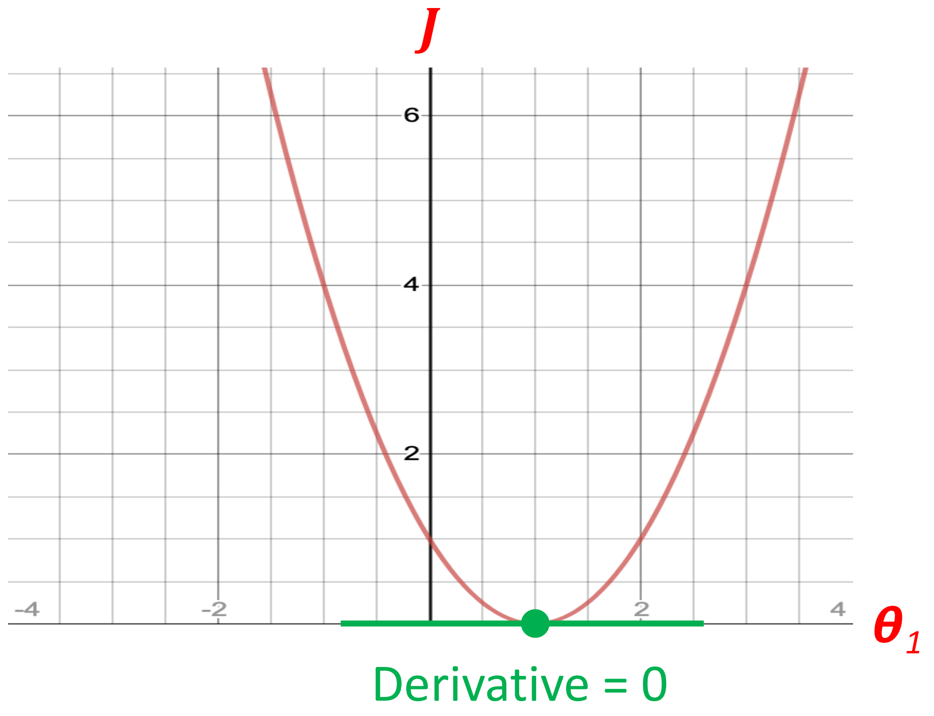


$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}$$

$$\underbrace{= \theta_1 - \alpha\, (Negative\ Number)}_{\text{Increase } \theta_1 \text{ by a certain value}}$$

The Impact of Partial Derviative

- For simplicity, let us assume our optimization objective is to $\underset{\theta_0,\theta_1}{\text{minimize}}\, \boldsymbol{J(\theta_1)}$, thus, $\boldsymbol{\theta}_0 = 0$



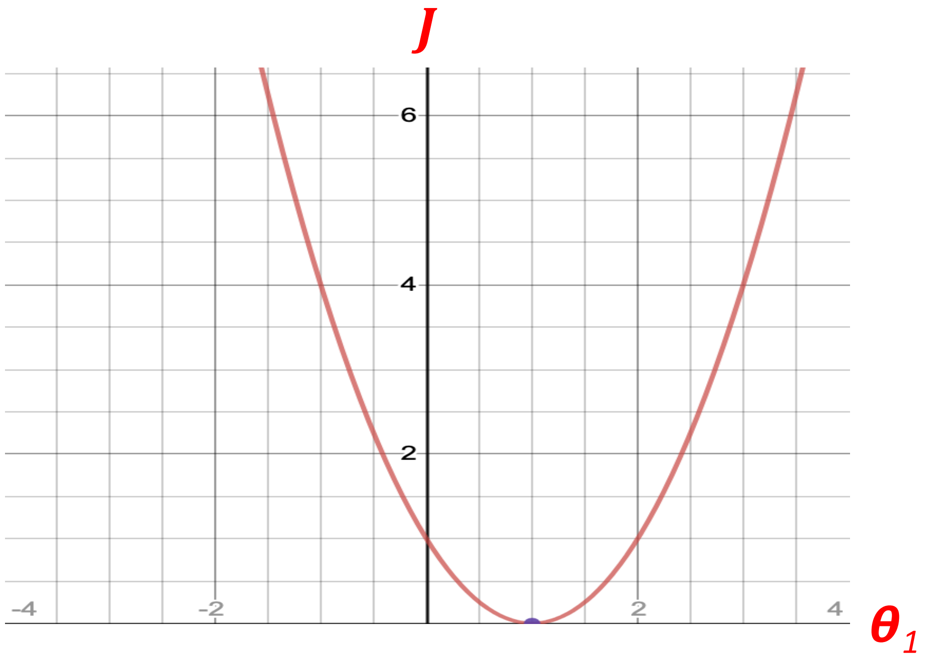
$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}$$

$$\underbrace{= \theta_1 - \alpha\, (Zero)}$$

θ_1 remains the same, hence, gradient descent *converges*

The Impact of Learning Rate

- For simplicity, let us assume our optimization objective is to $\underset{\theta_0,\theta_1}{\text{minimize}}\, \boldsymbol{J(\theta_1)}$, thus, $\boldsymbol{\theta}_0 = 0$



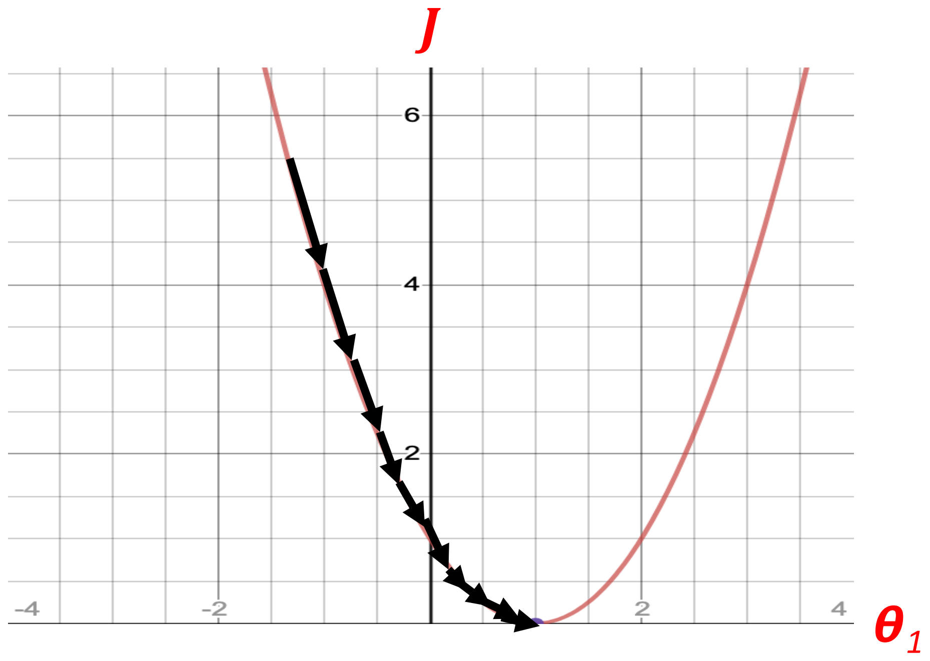
$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}$$

Learing Rate

What happens if α is too small?

The Impact of Learning Rate

- For simplicity, let us assume our optimization objective is to $\underset{\theta_0,\theta_1}{\text{minimize}}\, \boldsymbol{J(\theta_1)}$, thus, $\boldsymbol{\theta}_0 = 0$



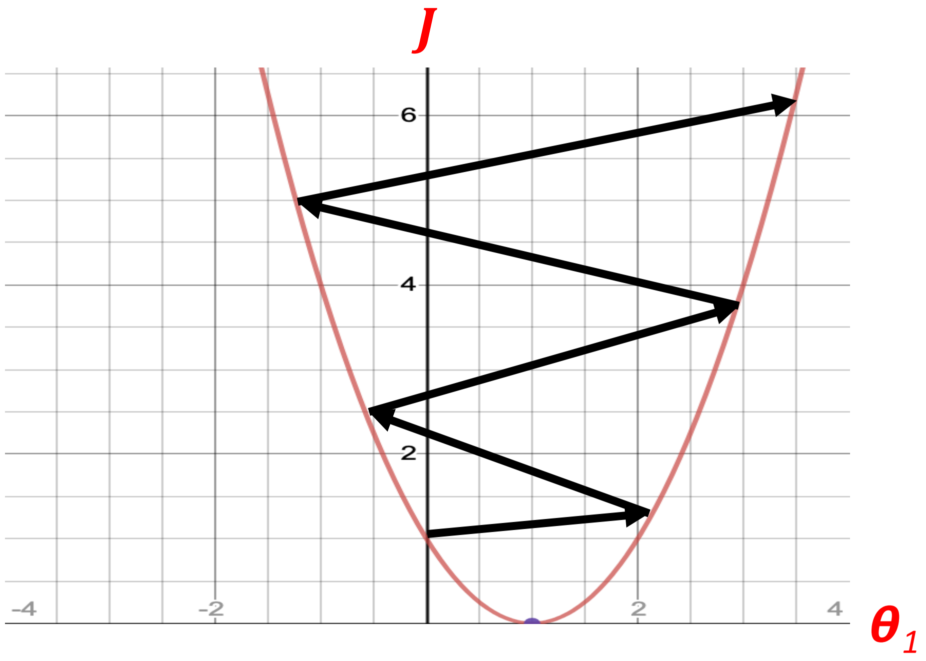
$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}$$

$$= \theta_1 - (Too\ Small\ Number)\ \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}$$

θ_1 changes only a tiny bit on each step, hence, gradient descent *will render slow (will take more time to converge)*

The Impact of Learning Rate

- For simplicity, let us assume our optimization objective is to $\min_{\theta_0,\theta_1} \boldsymbol{J(\theta_1)}$, thus, $\boldsymbol{\theta}_0 = 0$



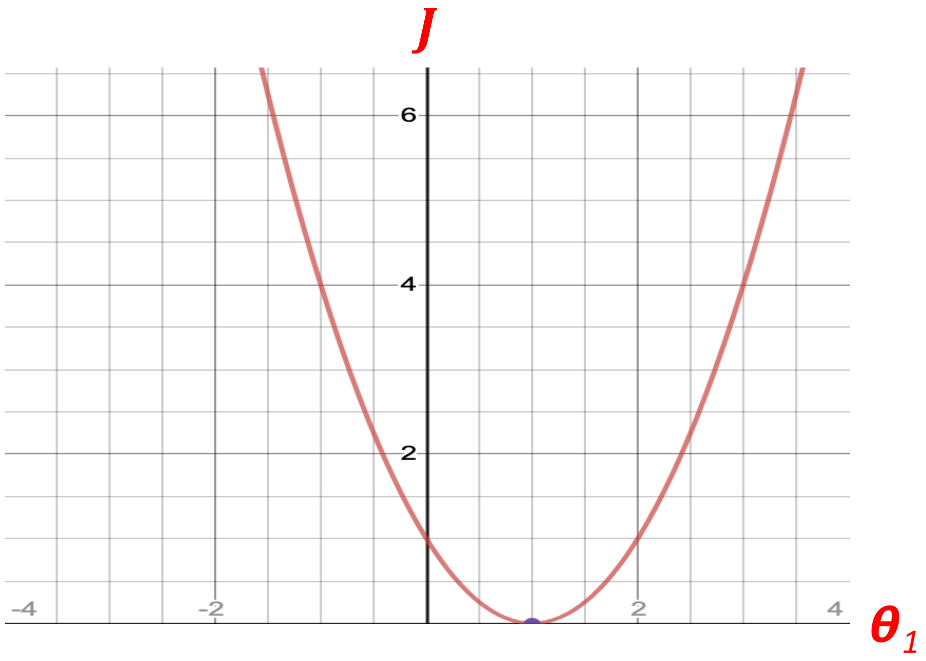
$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}$$

$$\underbrace{= \theta_1 - (Too\ Large\ Number)\ \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}}$$

θ_1 changes a lot (and probably faster) on each step, hence, gradient descent *will potentially overshoot the minimum and, accordingly, fail to converge (or even diverge)*

The Impact of Learning Rate

- For simplicity, let us assume our optimization objective is to $\underset{\theta_0,\theta_1}{\text{minimize}}\, \boldsymbol{J(\theta_1)}$, thus, $\boldsymbol{\theta}_0 = 0$

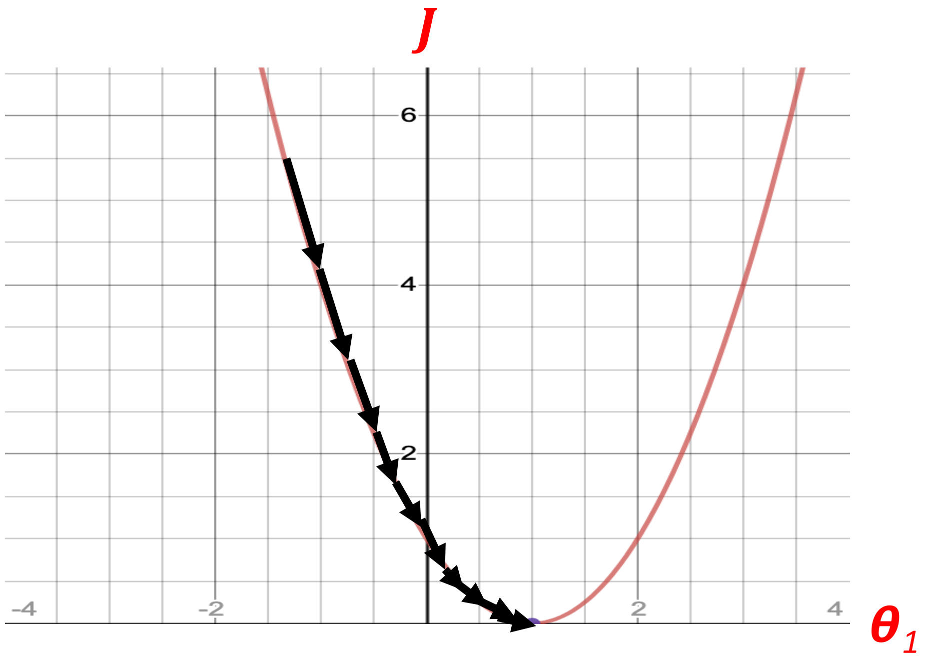


$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}$$

We can set α between 0 and 1 (say, 0.5, or a little more or less, hence, not very small or very large)

The Impact of Learning Rate

- For simplicity, let us assume our optimization objective is to $\underset{\theta_0,\theta_1}{\text{minimize}}\, \boldsymbol{J(\theta_1)}$, thus, $\boldsymbol{\theta}_0 = 0$



$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J(\theta_1)}}{\boldsymbol{d}\, \theta_j}$$

We can also ***fix α*** because as we approach the (global) minimum, gradient descent will automatically start taking smaller steps (i.e., θ_1 will start changing at a slower pace because the derivative will become less steep)

Gradient Descent For Linear Regression

- **Outline**:
  - Have some cost function $\boldsymbol{J(\theta_0, \ldots, \theta_{n-1})}$
  - Start off with some guesses for $\theta_0, \ldots, \theta_{n-1}$
    - It does not really matter what values you start off with, but a common choice is to set them all initially to zero
  - Repeat until convergence{

Partial derivative

$$\theta_j = \theta_j - \alpha \frac{\partial \boldsymbol{J(\theta_0, \ldots, \theta_{n-1})}}{\partial \theta_j}$$

}

***Learing rate**, which controls how big a step we take when we update θ_j*

Now we understand the intuition behind gradient descent and how $\boldsymbol{\alpha}$ and $\boldsymbol{\partial}$ act together to make gradient descent work!

Gradient Descent For Linear Regression

- **Outline** (considering only two varilables θ_0 and θ_1):
  - Have some cost function $\boldsymbol{J(\theta_0, \theta_1)}$
  - Start off with some guesses for θ_0, θ_1
    - It does not really matter what values you start off with, but a common choice is to set them both initially to zero
  - Repeat until convergence{

$$temp_0 = \theta_0 - \alpha \boxed{\frac{\partial\, \boldsymbol{J(\theta_0, \theta_1)}}{\partial\, \theta_0}} \rightarrow \frac{1}{n}\sum_{i=1}^{n}\left(h_\theta(x)^{(i)} - y^{(i)}\right)$$

$$temp_1 = \theta_1 - \alpha \boxed{\frac{\partial\, \boldsymbol{J(\theta_0, \theta_1)}}{\partial\, \theta_1}} \rightarrow \frac{1}{n}\sum_{i=1}^{n}\left(h_\theta(x)^{(i)} - y^{(i)}\right) .x^{(i)}$$

$$\theta_0 = temp_0$$
$$\theta_1 = temp_1$$

}

Gradient Descent For Linear Regression

- **Outline** (considering only two varilables θ_0 and θ_1):
  - Have some cost function $\boldsymbol{J(\theta_0, \theta_1)}$
  - Start off with some guesses for θ_0, θ_1
    - It does not really matter what values you start off with, but a common choice is to set them both initially to zero
  - Repeat until convergence{

$$temp_0 = \theta_0 - \alpha \frac{1}{n} \sum_{i=1}^{n} \left(h_\theta(x)^{(i)} - y^{(i)}\right)$$

$$temp_1 = \theta_1 - \alpha \frac{1}{n} \sum_{i=1}^{n} \left(h_\theta(x)^{(i)} - y^{(i)}\right) . x^{(i)}$$

$$\theta_0 = temp_0$$
$$\theta_1 = temp_1$$

}

Today’s plan: Linear Regression

- Model representation
- Cost function
- Gradient descent
- Features and polynomial regression
- Normal equation

Linear Regression

- **Model representation**
- Cost function
- Gradient descent
- Features and polynomial regression
- Normal equation

<u>Regression</u>

real-valued output

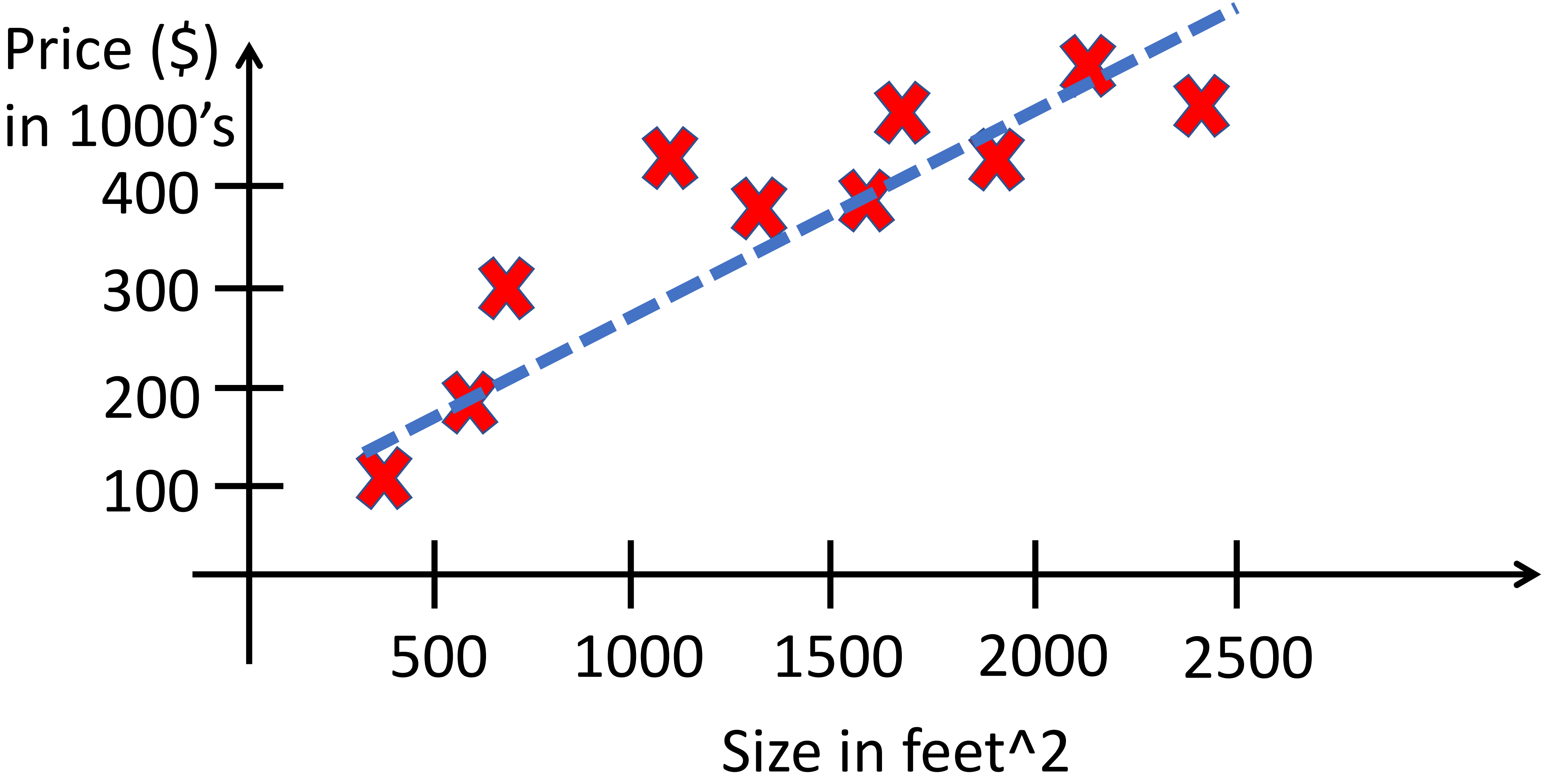
Training set

Learning Algorithm

x → h → y

Size of house | Hypothesis | Estimated price

House pricing prediction



Training set

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

m = 47

- Notation:
  - m = Number of training examples
  - x = Input variable / features
  - y = Output variable / target variable
  - (x, y) = One training example
  - $(x^{(i)}, y^{(i)})$ = i^{th} training example

Examples:

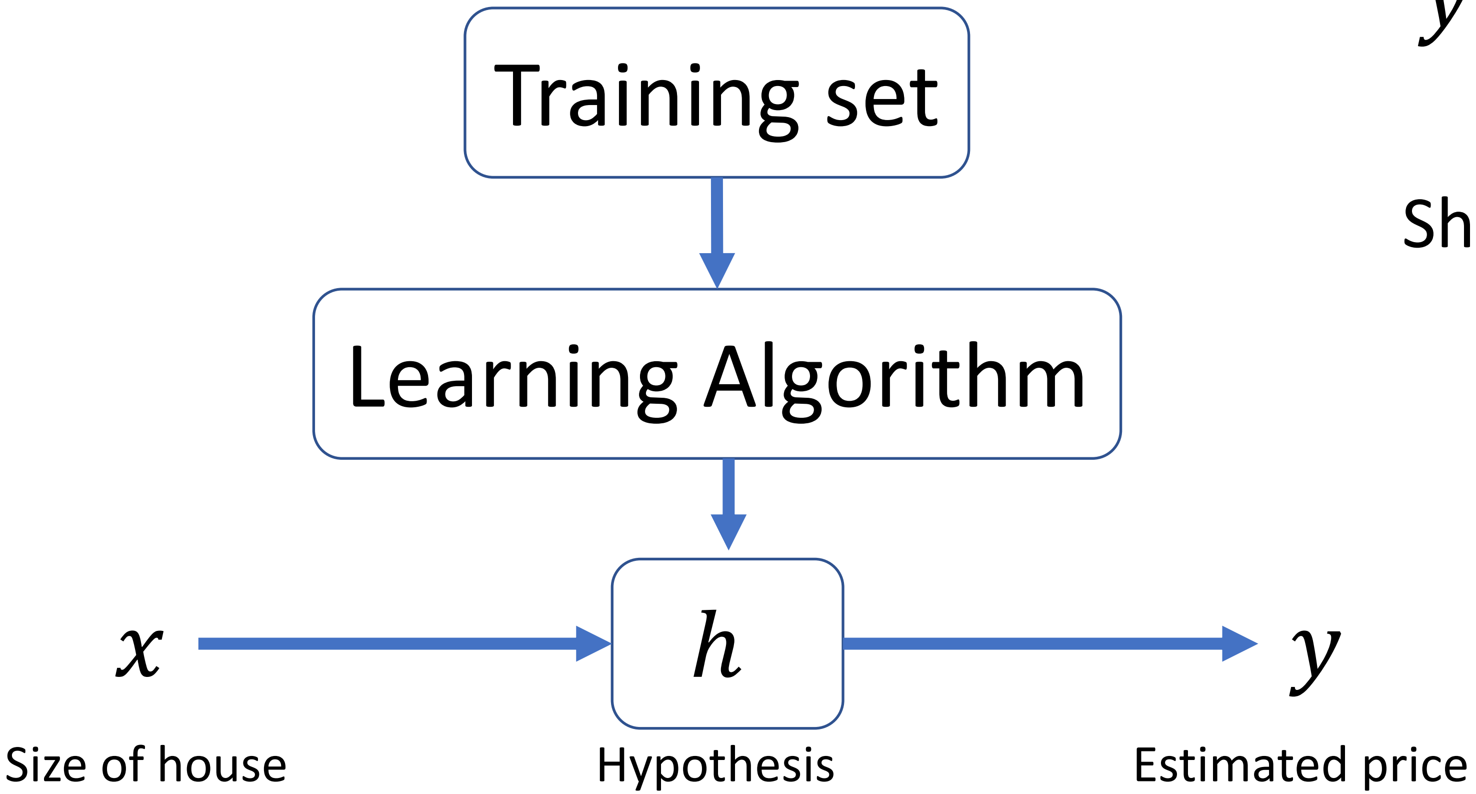
$x^{(1)} = 2104$

$x^{(2)} = 1416$

$y^{(1)} = \ 460$

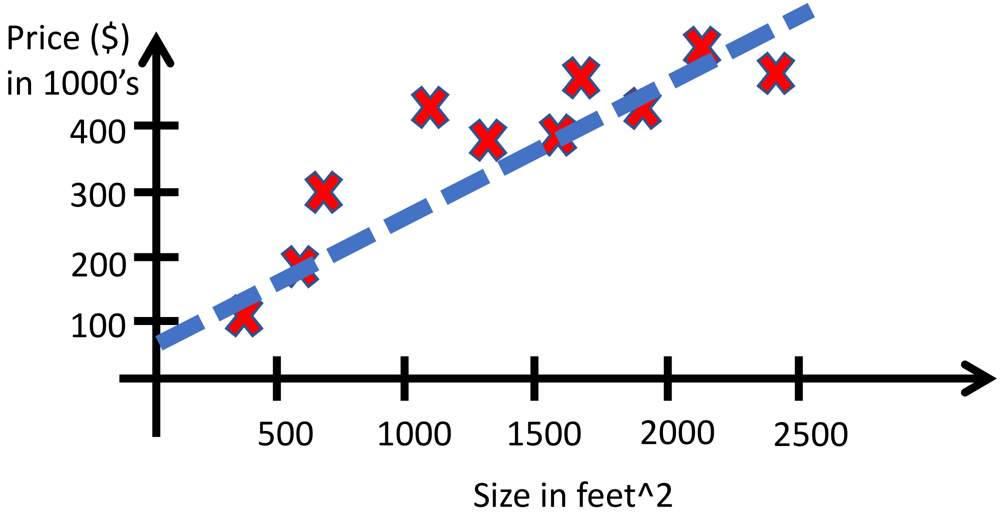
Slide credit: Andrew Ng

Model representation



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

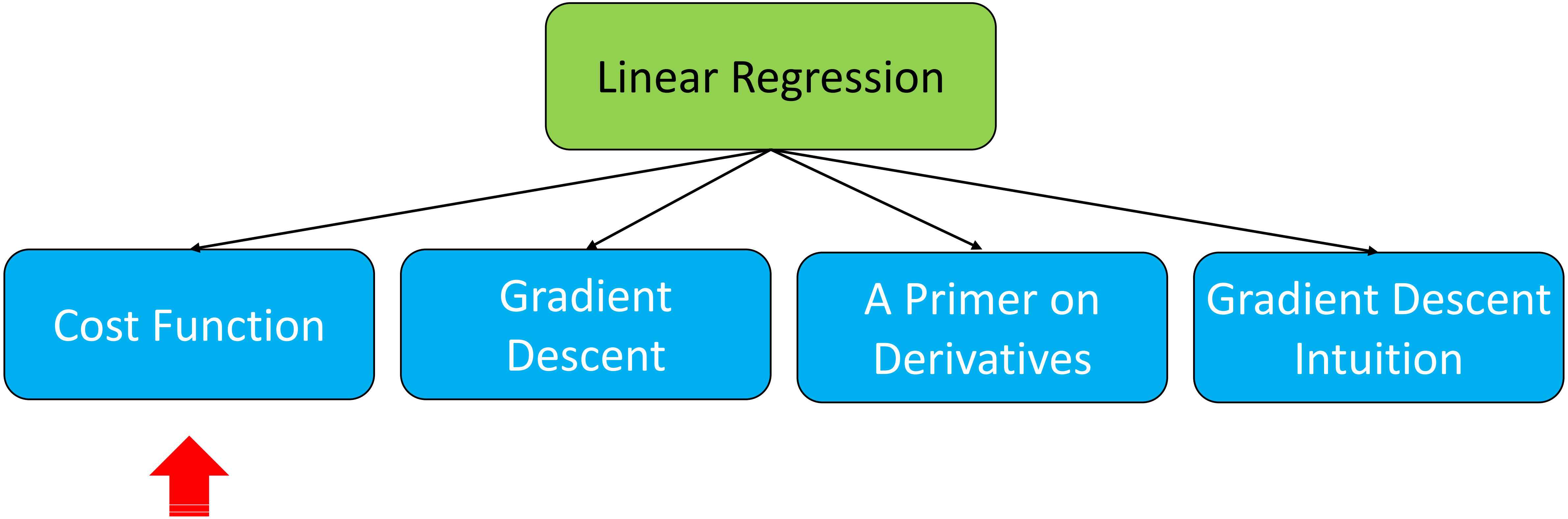
Shorthand $h(x)$



Univariate linear regression

Slide credit: Andrew Ng

Outline


Linear Regression
Cost Function
Gradient Descent
A Primer on Derivatives
Gradient Descent Intuition

Linear Regression

- Model representation
- **Cost function**
- Gradient descent
- Features and polynomial regression
- Normal equation

Training set

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

m = 47

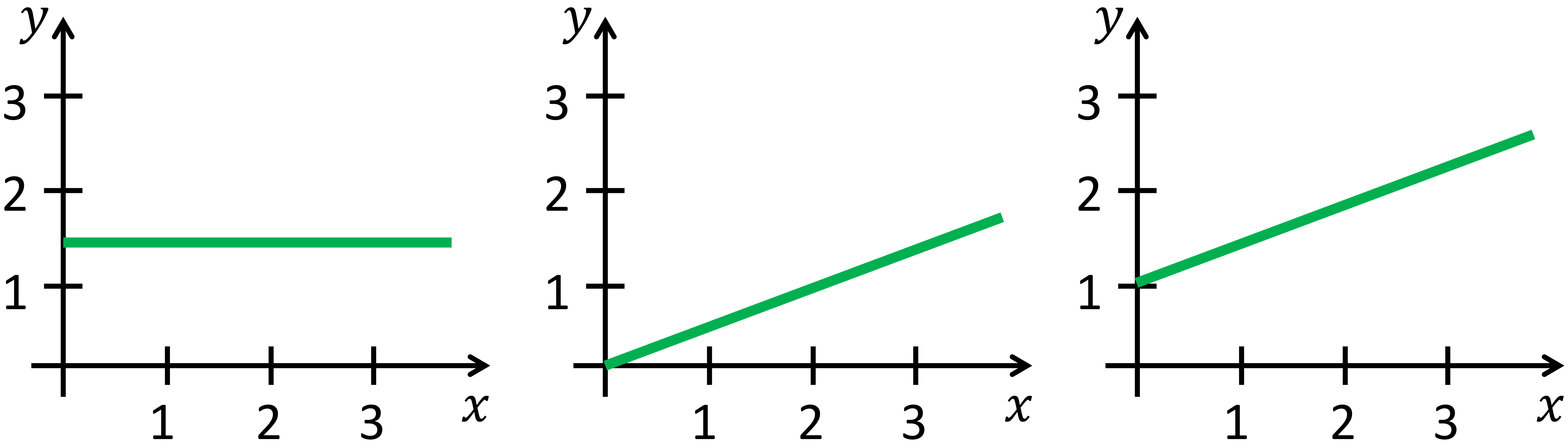
- Hypothesis $h_\theta(x) = \theta_0 + \theta_1 x$

θ_0, θ_1: parameters/weights

How to choose θ_i's?

Slide credit: Andrew Ng

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



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

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

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

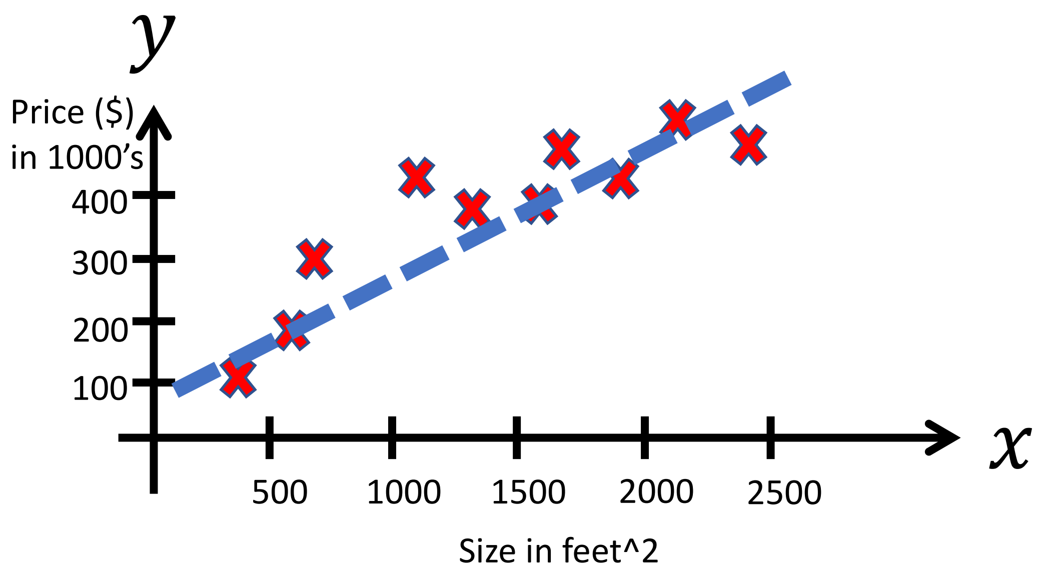
Slide credit: Andrew Ng

Cost function

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

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

$$h_\theta\left(x^{(i)}\right) = \theta_0 + \theta_1 x^{(i)}$$



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

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

Slide credit: Andrew Ng

Simplified

- **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:**

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

- **Hypothesis:**

$$h_\theta(x) = \theta_1 x \qquad \theta_0 = 0$$

- **Parameters:**

$$\theta_1$$

- **Cost function:**

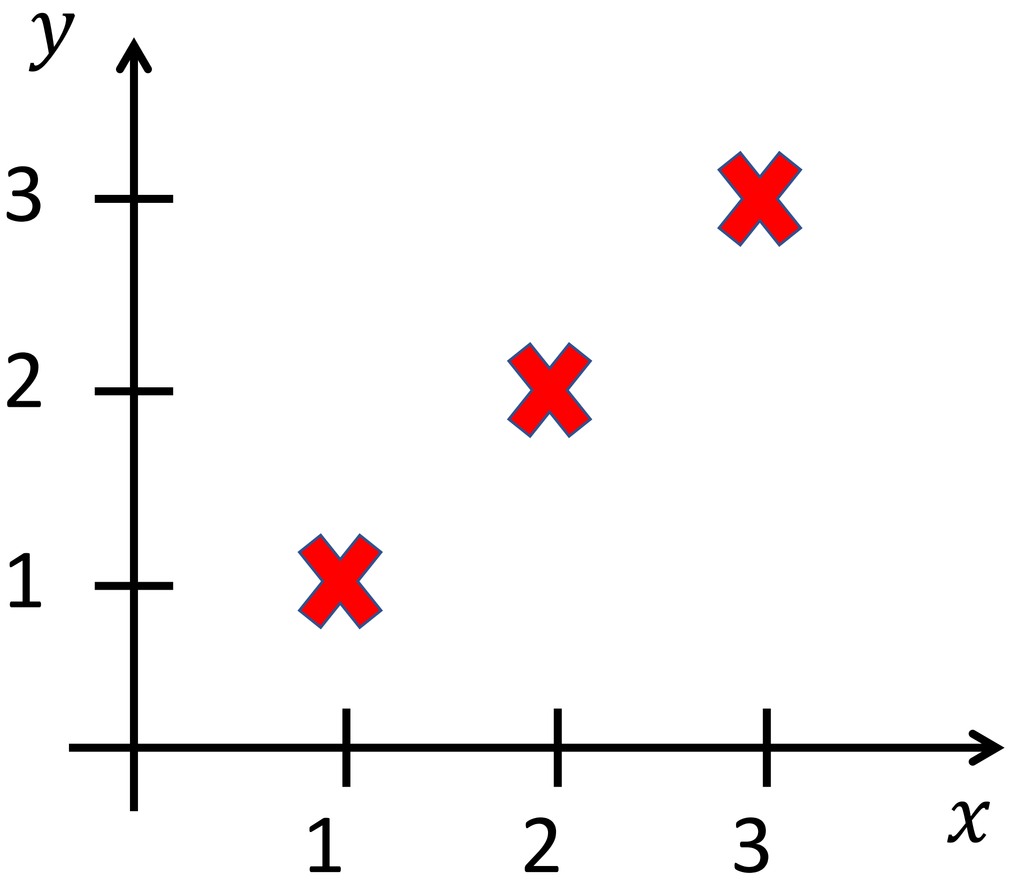
$$J(\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

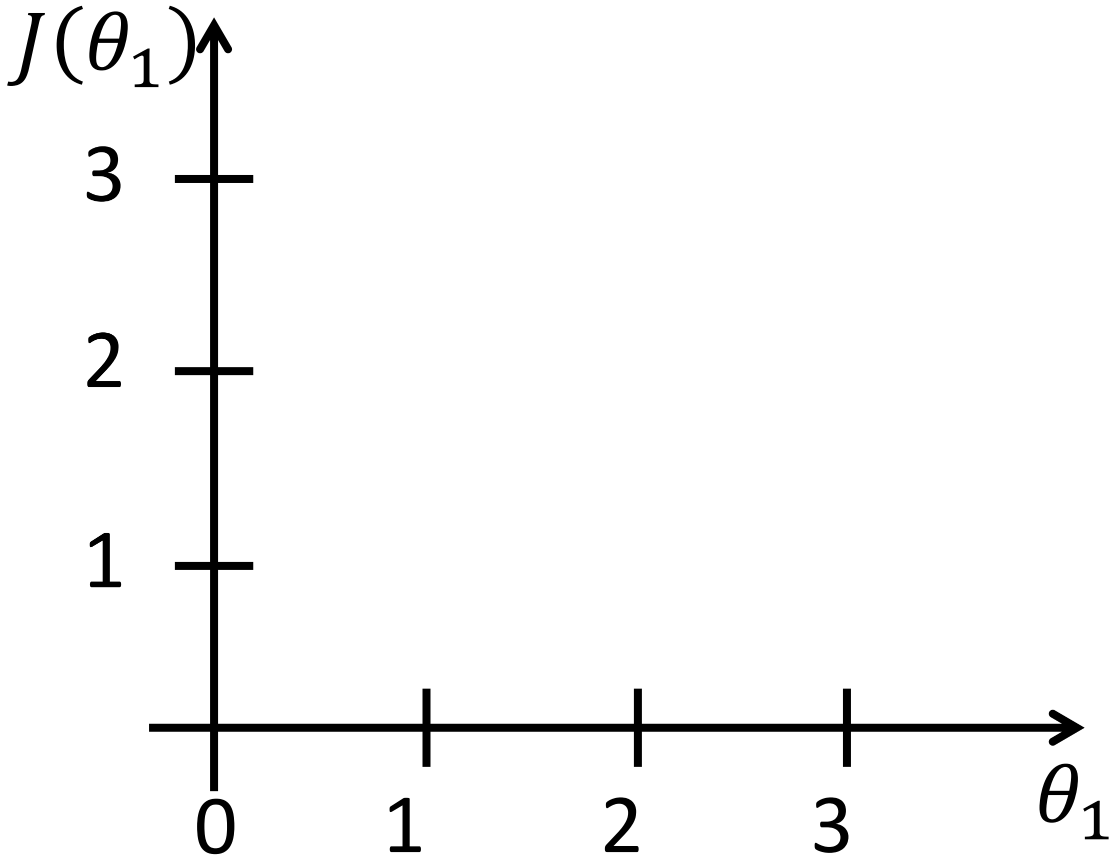
- **Goal:**

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

Slide credit: Andrew Ng

$h_\theta(x)$, function of x 　　 $J(\theta_1)$, function of θ_1

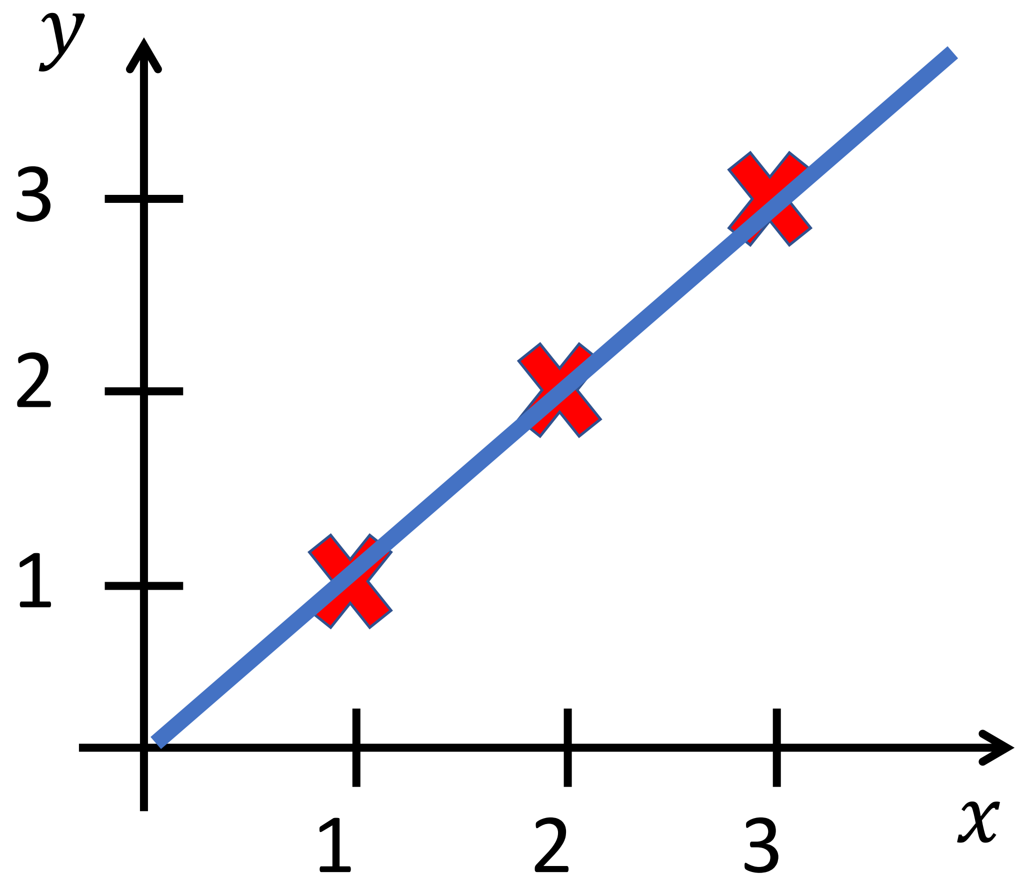


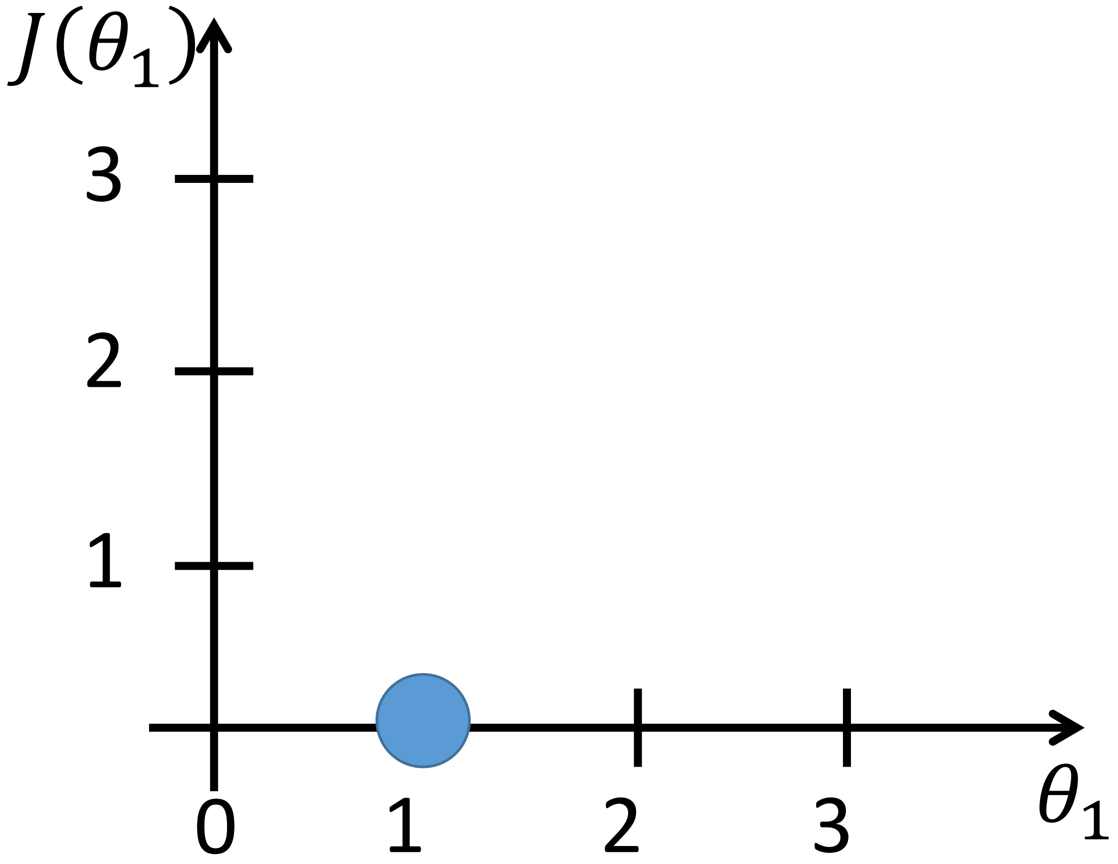


Slide credit: Andrew Ng

$h_\theta(x)$, function of x

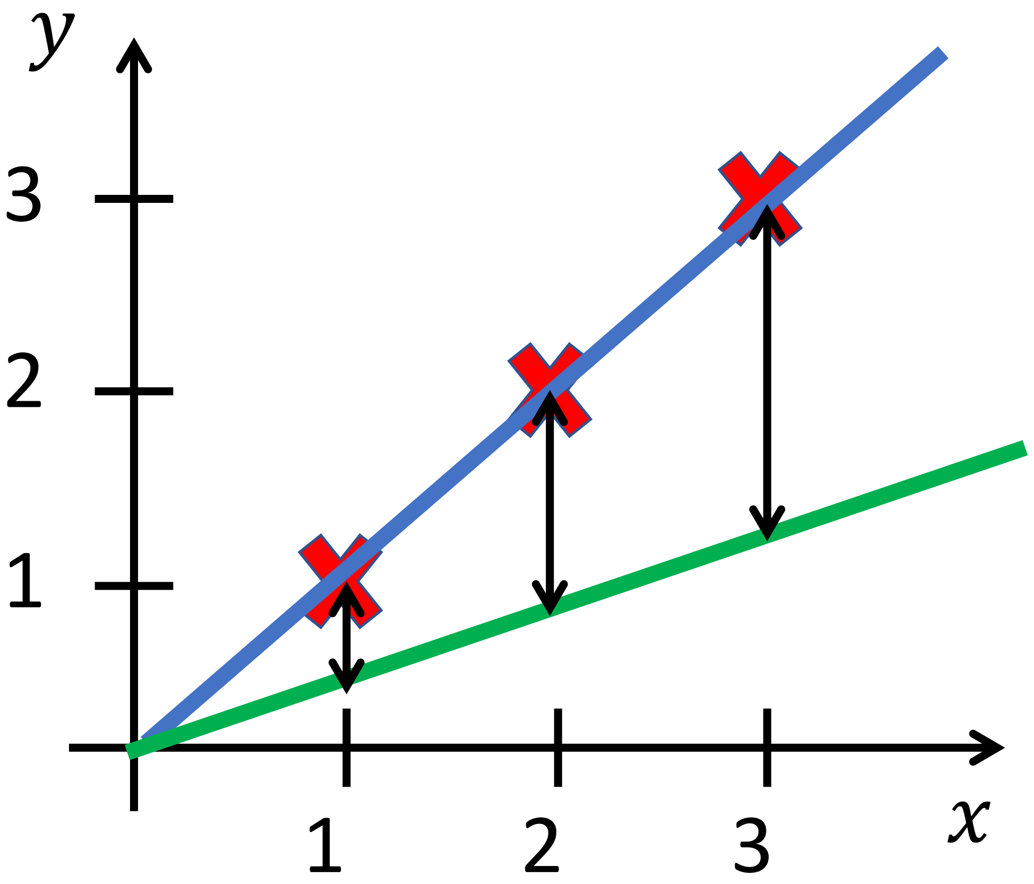
$J(\theta_1)$, function of θ_1



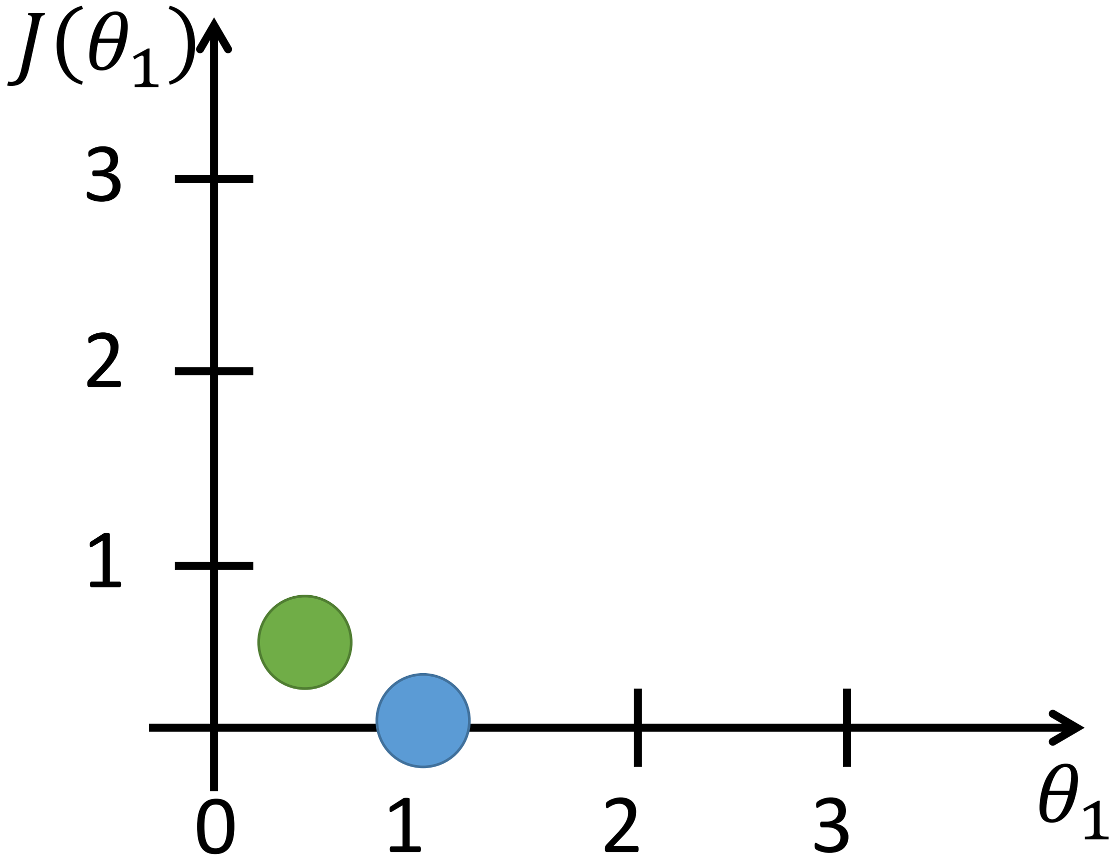


Slide credit: Andrew Ng

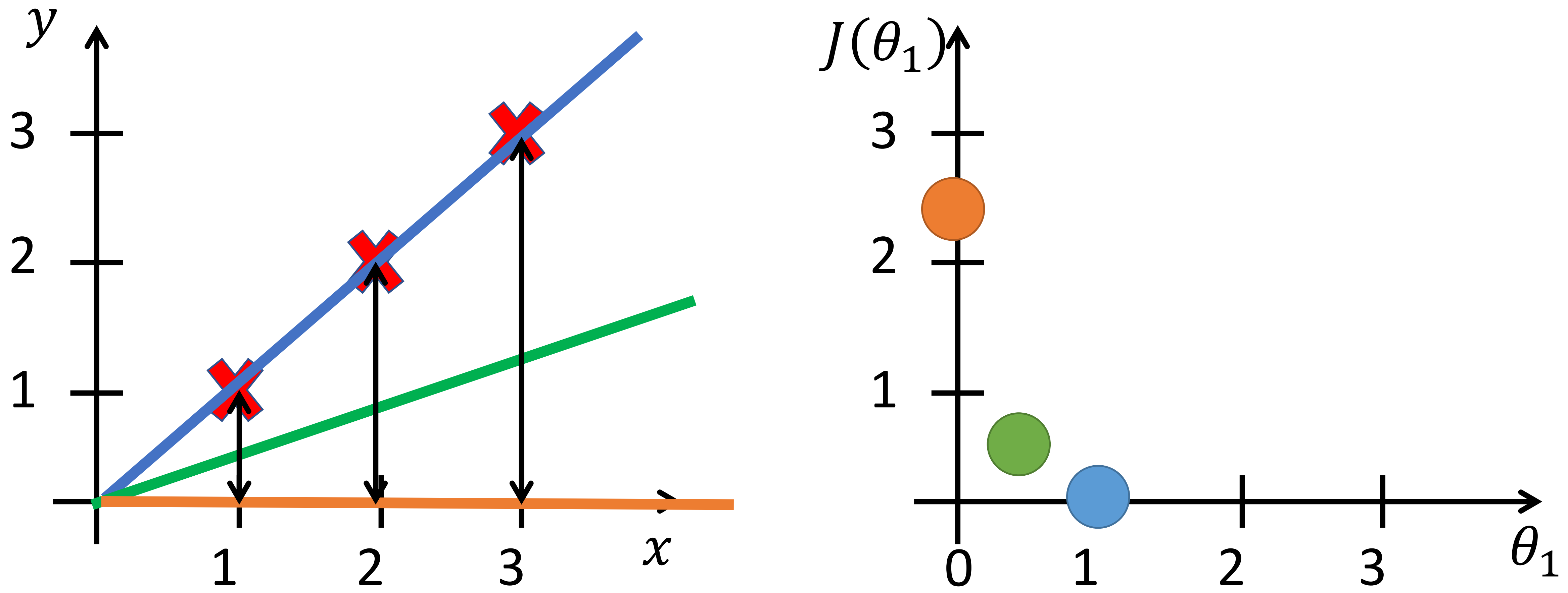
$h_\theta(x)$, function of x



$J(\theta_1)$, function of θ_1

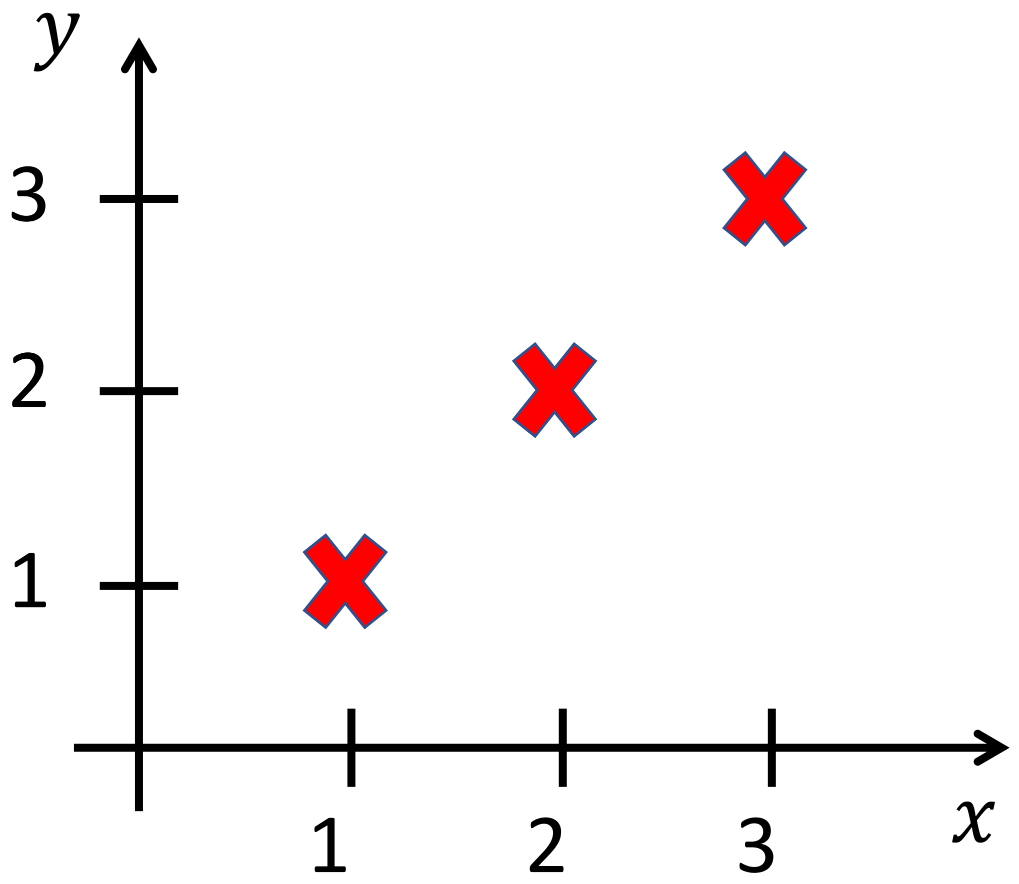


$h_\theta(x)$, function of x $\quad$ $J(\theta_1)$, function of θ_1

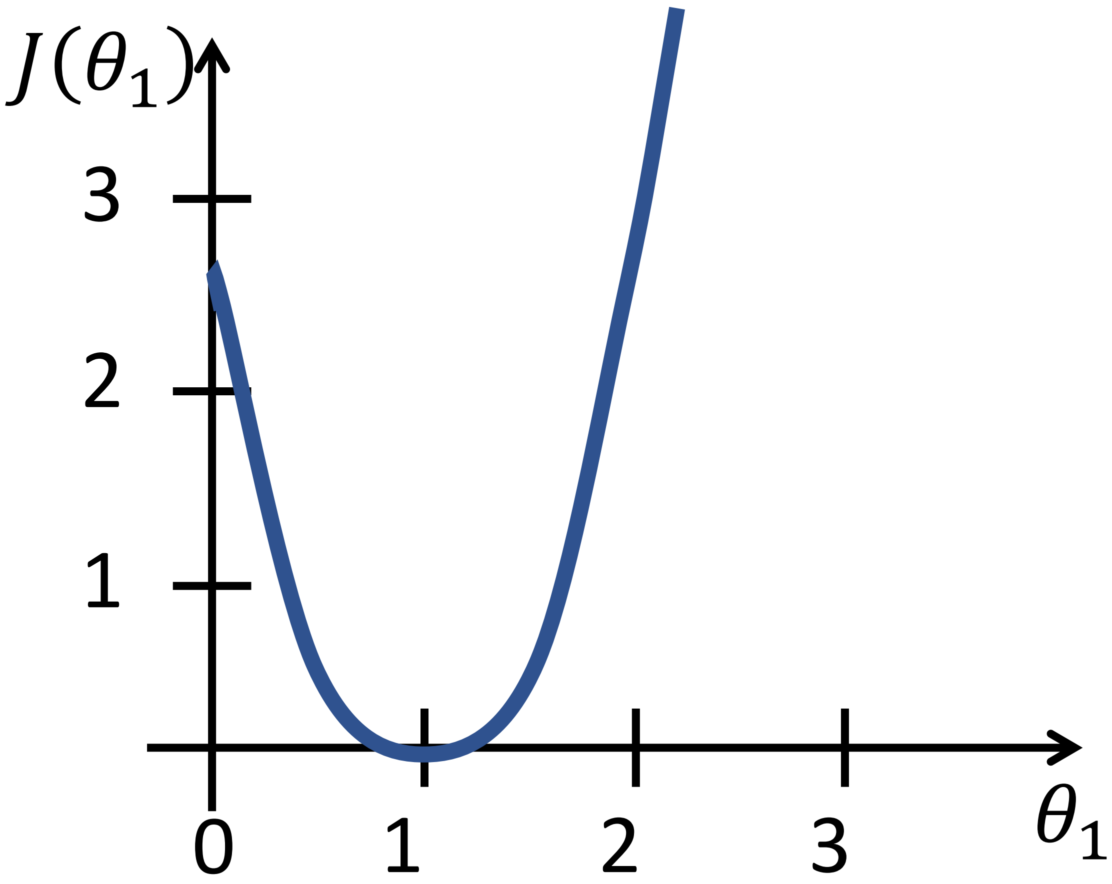


Slide credit: Andrew Ng

$h_\theta(x)$, function of x



$J(\theta_1)$, function of θ_1



Slide credit: Andrew Ng

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

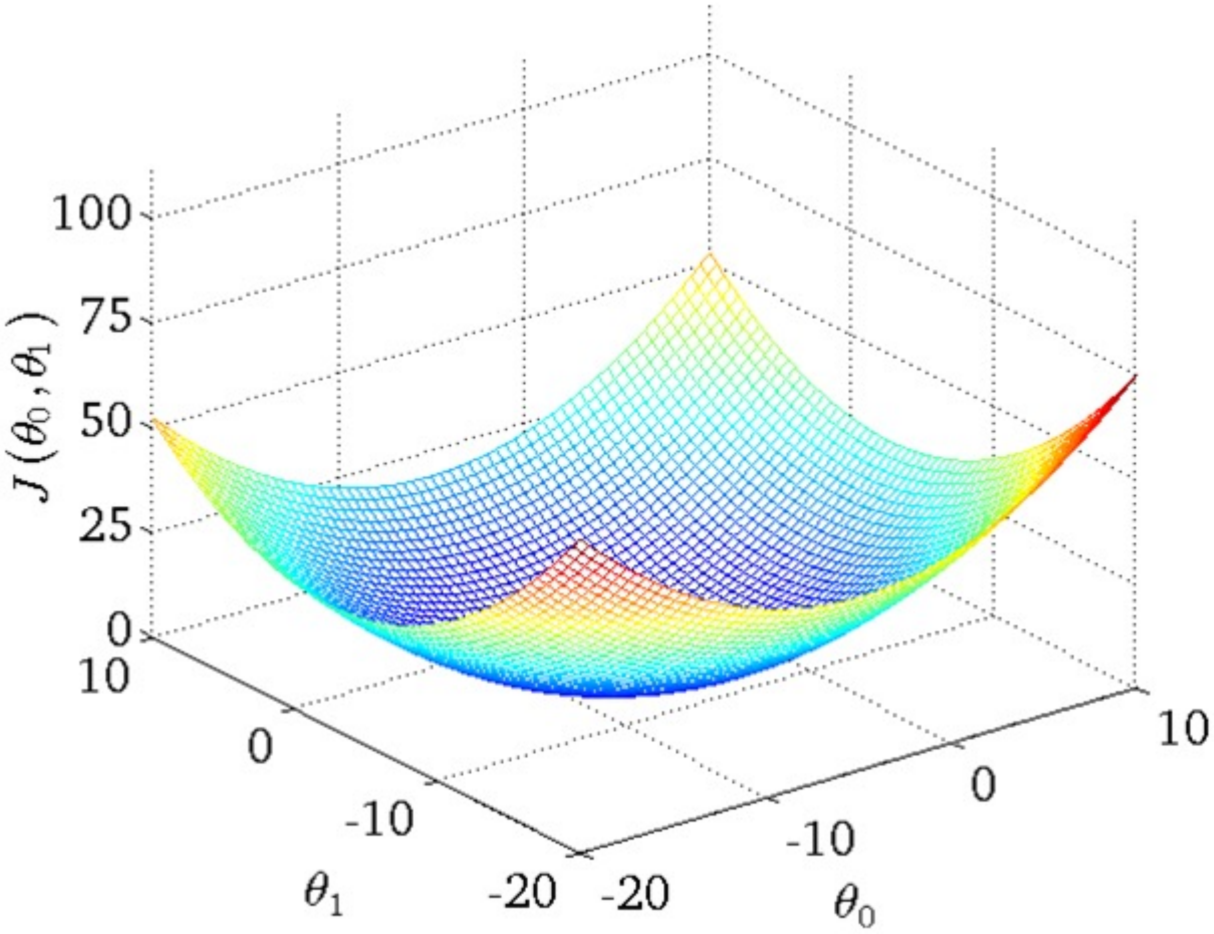
- **Parameters:** θ_0, θ_1

- **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**: $\underset{\theta_0, \theta_1}{\text{minimize}} \; J(\theta_0, \theta_1)$

Slide credit: Andrew Ng

Cost function



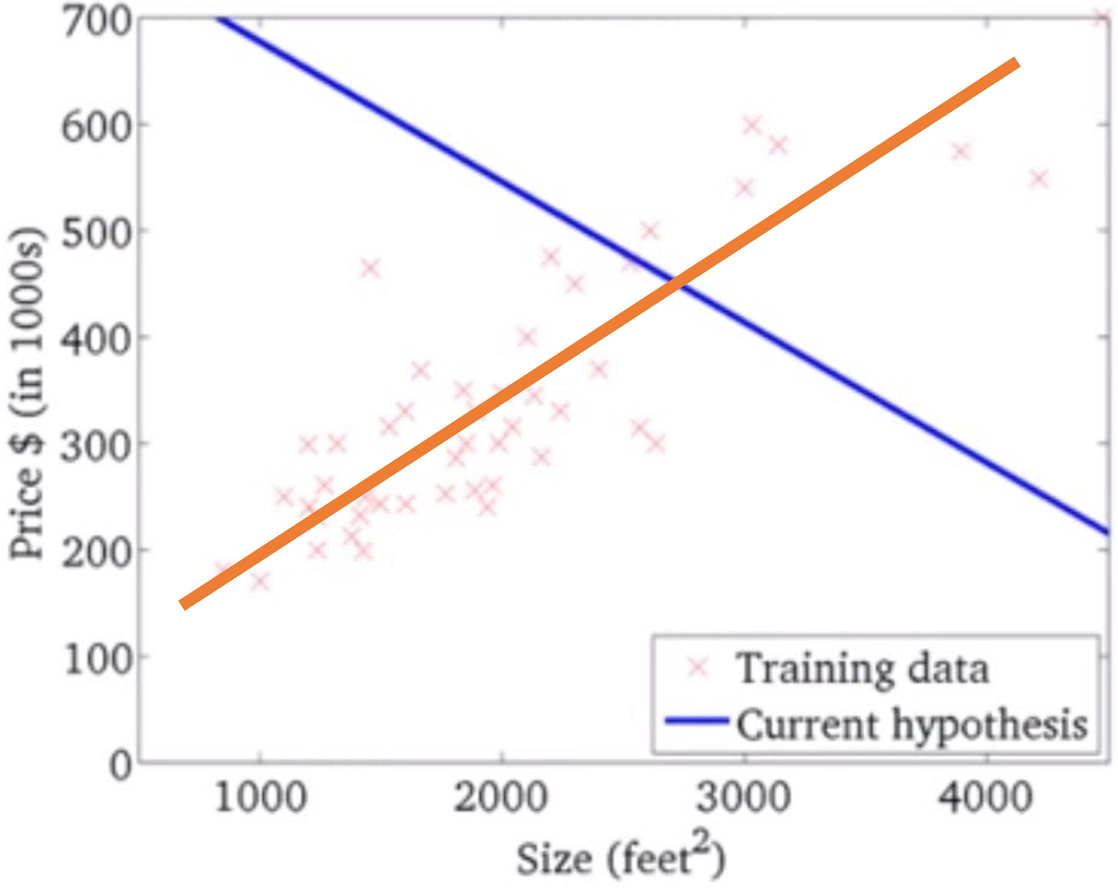
Slide credit: Andrew Ng

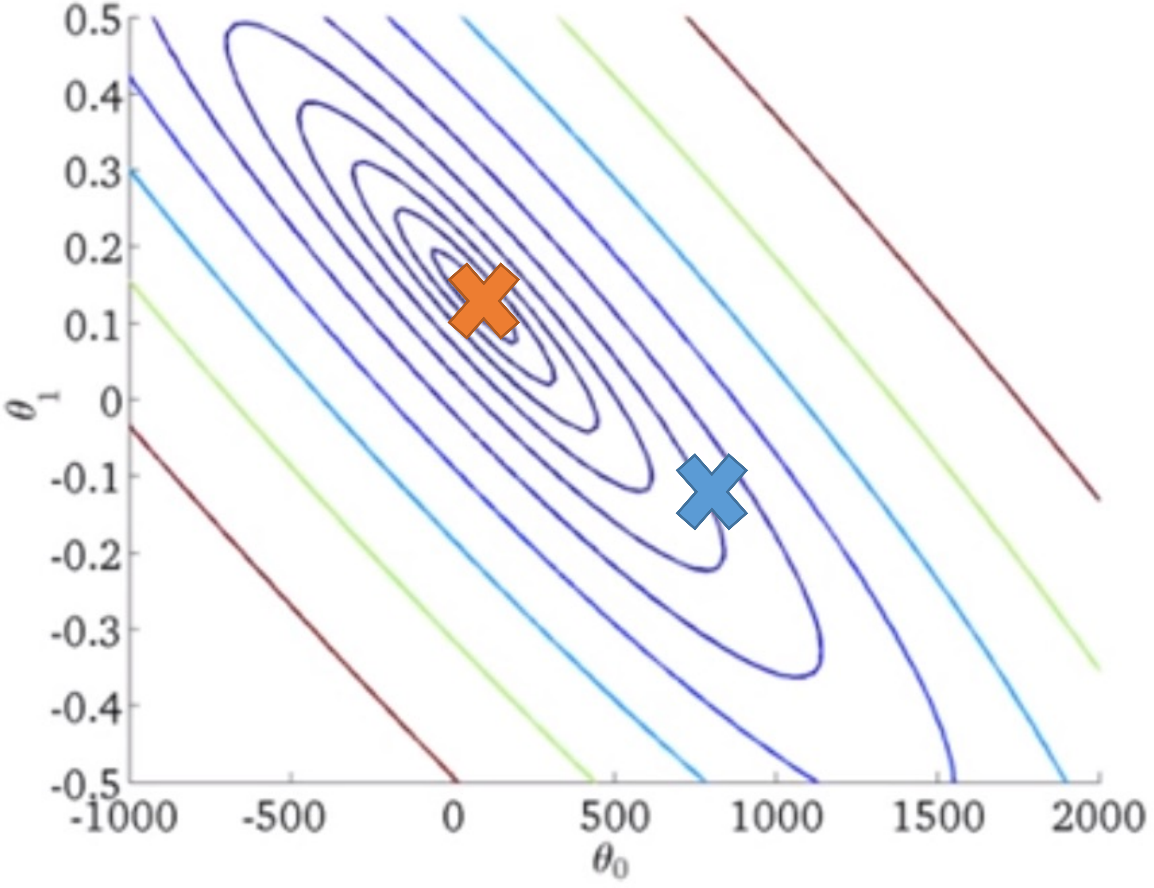
$h_\theta(x)$

(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)

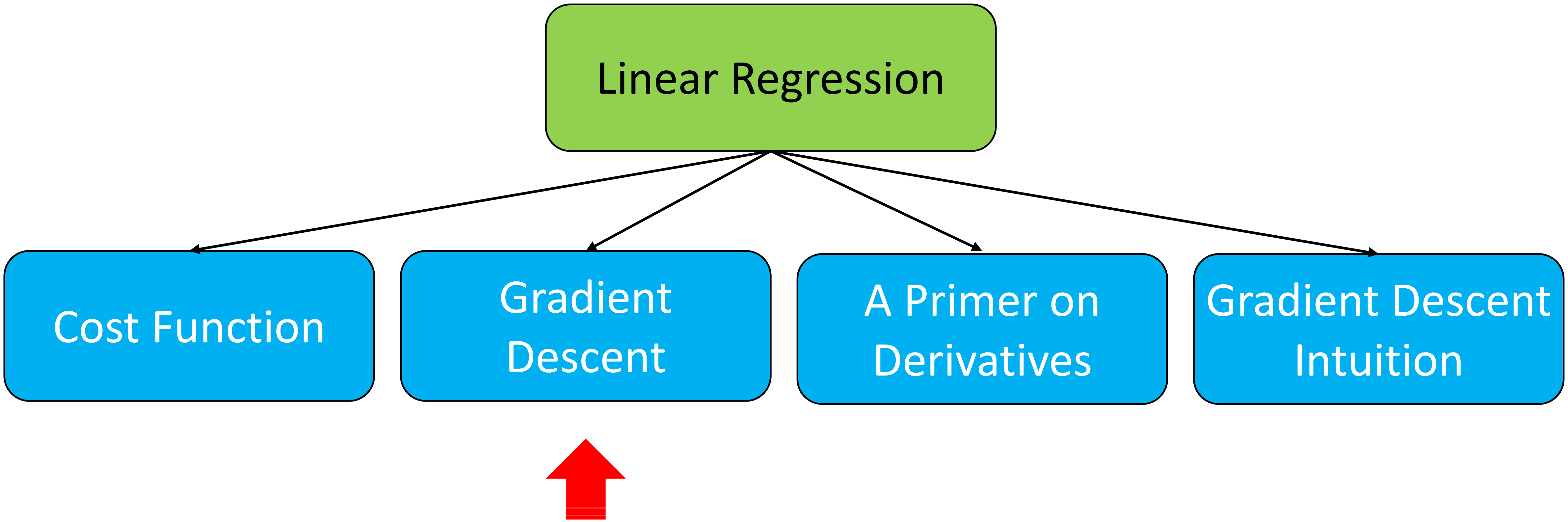




How do we find good θ_0, θ_1 that minimize $J(\theta_0, \theta_1)$?

Slide credit: Andrew Ng

Outline


Linear Regression
Cost Function
Gradient Descent
A Primer on Derivatives
Gradient Descent Intuition

Linear Regression

- Model representation
- Cost function
- **Gradient descent**
- Features and polynomial regression
- Normal equation

Gradient descent

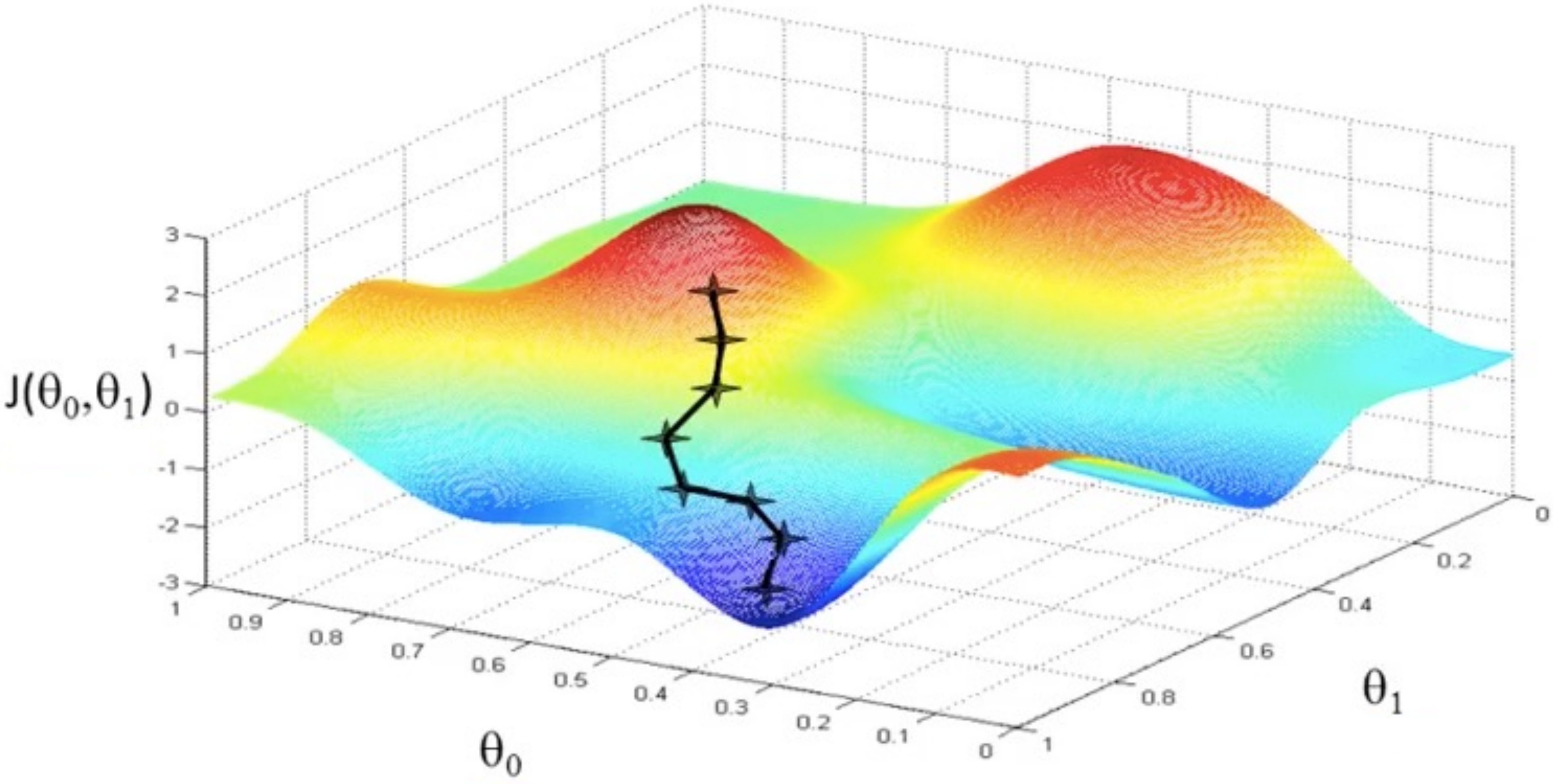
Have some function $J(\theta_0, \theta_1)$

Want $\underset{\theta_0, \theta_1}{\operatorname{argmin}} \; J(\theta_0, \theta_1)$

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 minimum

Slide credit: Andrew Ng



Slide credit: Andrew Ng

Gradient descent

Repeat until convergence{

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

}

α: Learning rate (step size)

$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$: derivative (rate of change)

Slide credit: Andrew Ng

Gradient descent

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:

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

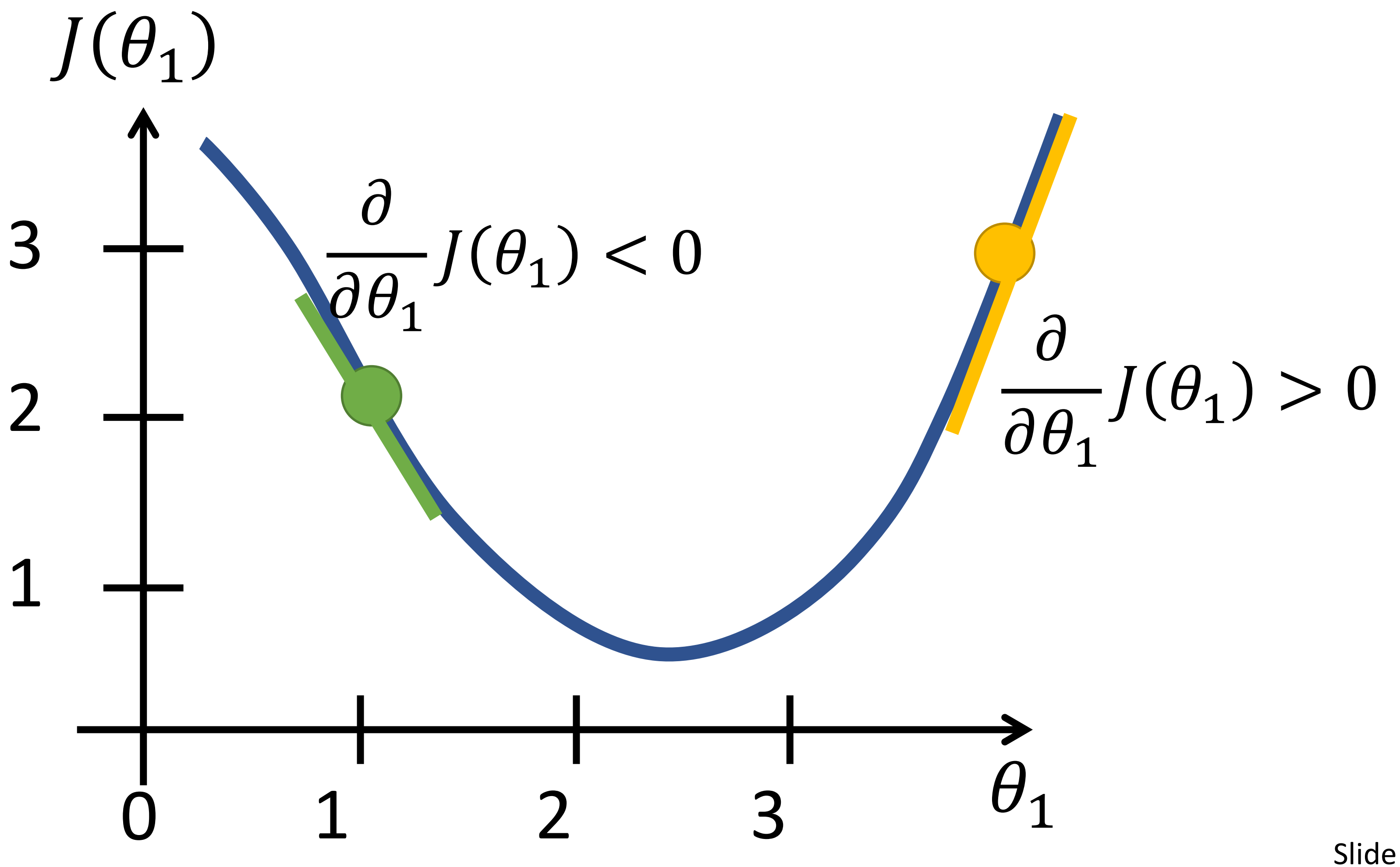
$$\theta_0 := \text{temp0}$$

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

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

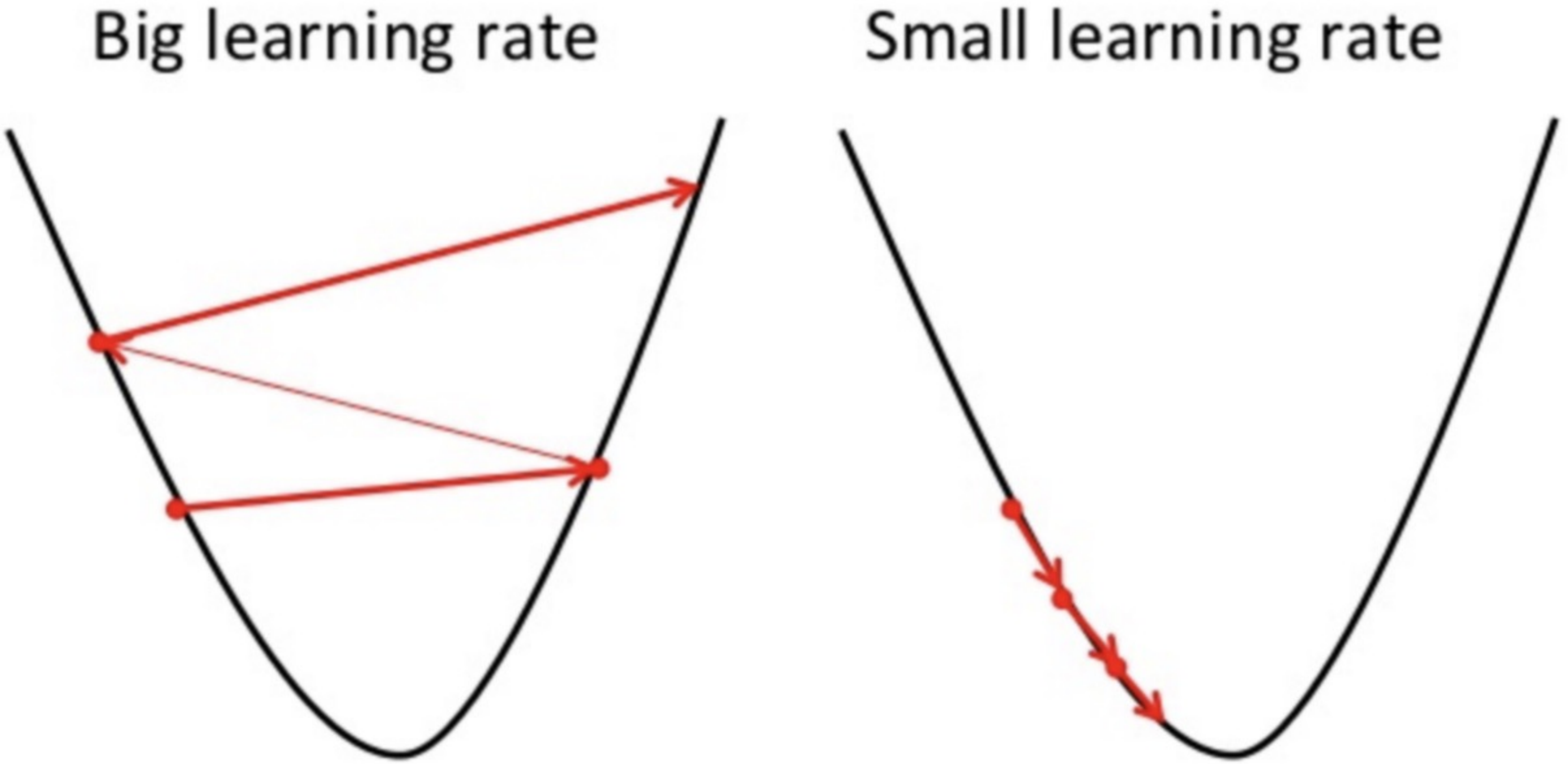
Slide credit: Andrew Ng

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



Slide credit: Andrew Ng

Learning rate



Gradient descent for linear regression

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

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

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

Slide credit: Andrew Ng

Computing partial derivative

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

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

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

Slide credit: Andrew Ng

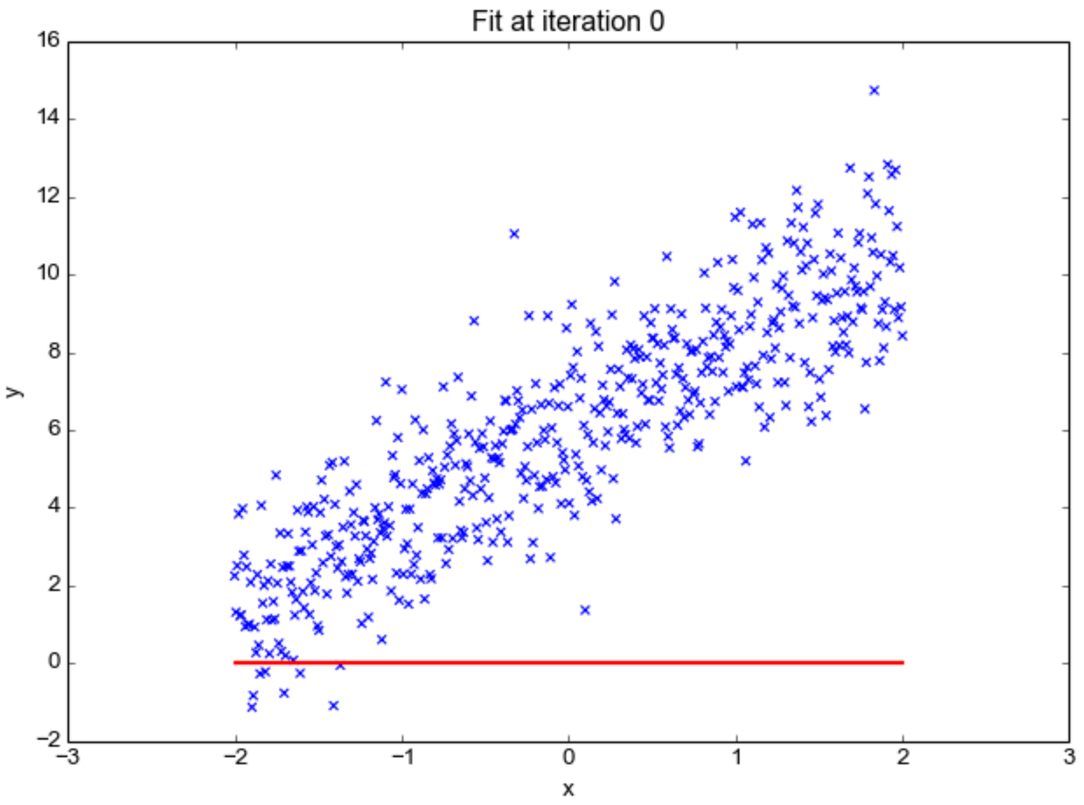
Gradient descent for linear regression

Repeat until convergence{

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

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

}



Update θ_0 and θ_1 simultaneously

Slide credit: Andrew Ng