Linear Regression

A number of slides and screenshots from : Andrew Ng's course on machine learning and Sebastian Raschka's course on deep learning
Both can be found for free on youtube

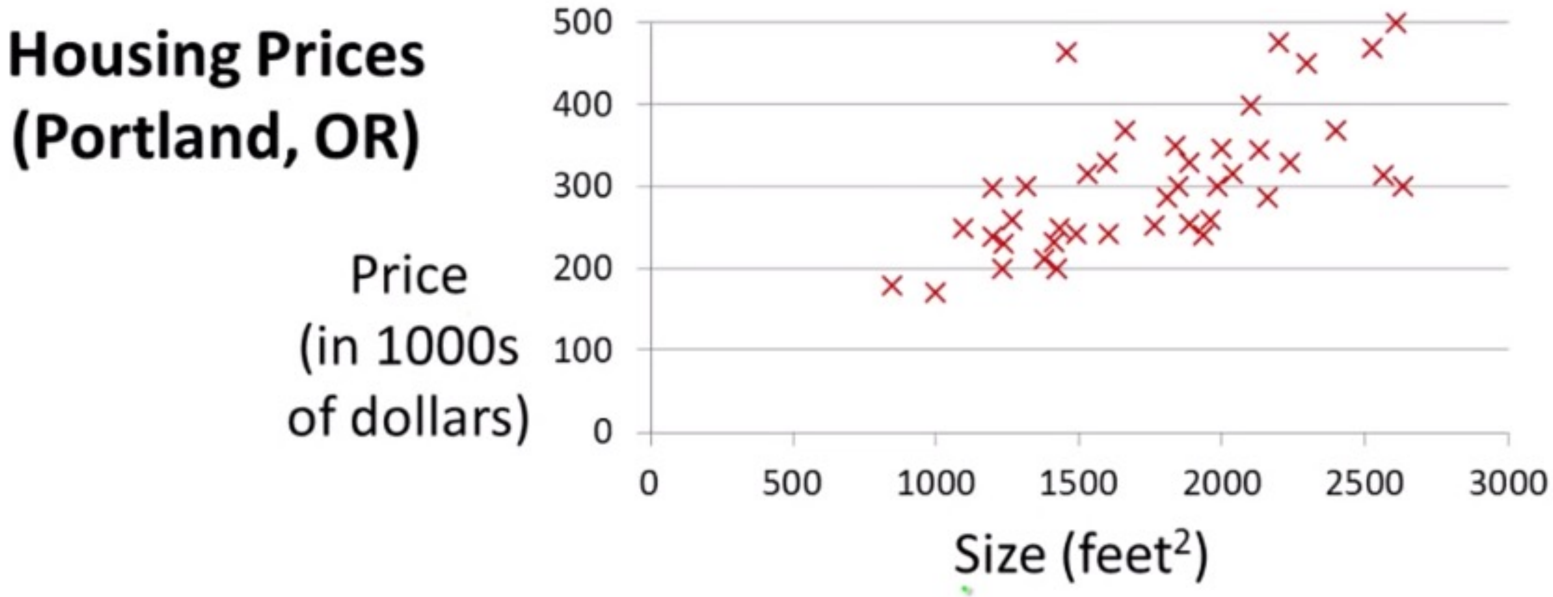
What does regression mean ?

- Seen in intro, but :
- Regression means predictiong real-valued outputs.
- An essential type of supervised machine learning task (trying to give the right « answer » for each example in the data).
- Often contrasted with classification.

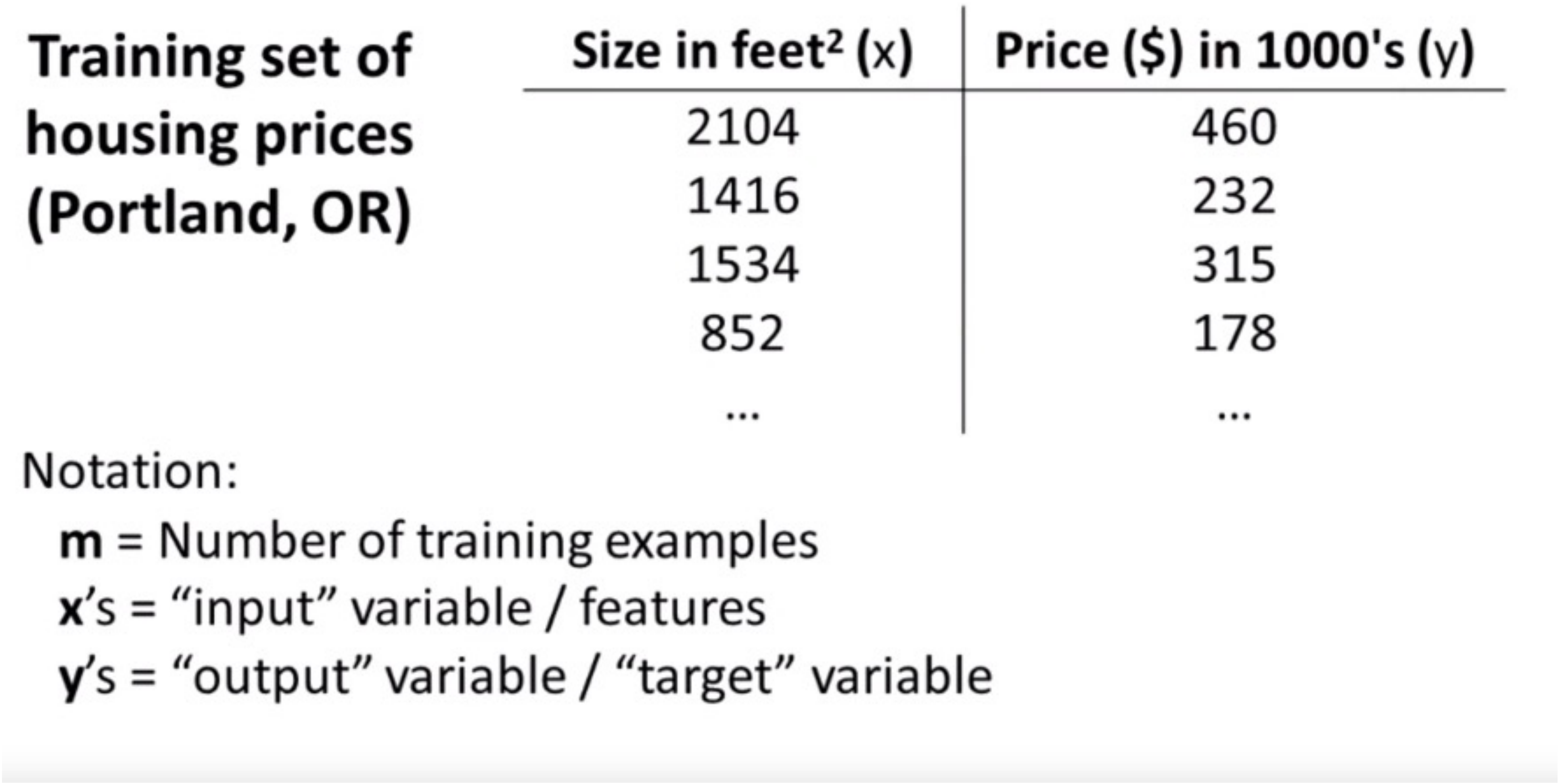
- Example :
- Predicting height => many many real-valued outputs are possible…
- Vs. Predicting a « height class » : short medium-height tall

Dataset and problem example

- Imagine we want to create an ML algorithm to predict the price of a house, using only as information the size of the house. This is the dataset we can use to train our algorithm.



Training Set and Notation



Training set of housing prices (Portland, OR)

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

Notation:

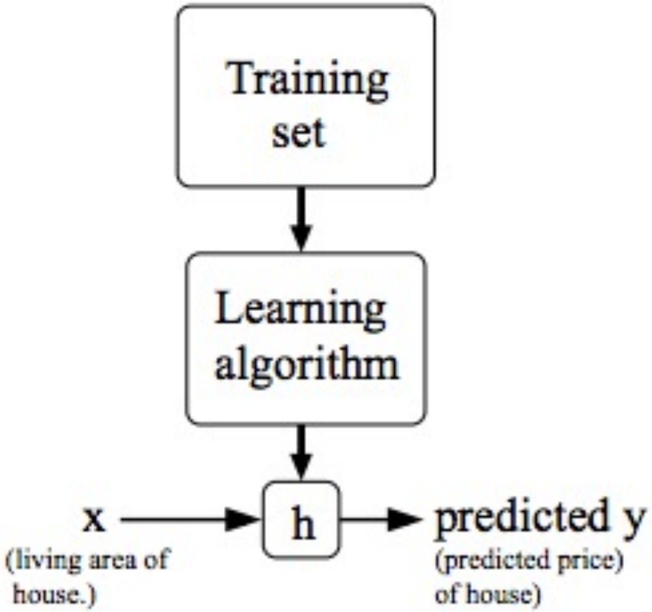
m = Number of training examples
x's = "input" variable / features
y's = "output" variable / "target" variable

The supervised learning workflow

- h: hypothesis
- h is a function which maps x’s to y’s
- Our goal will be to find the function which takes

x as input and predicts the correct y for that

x.

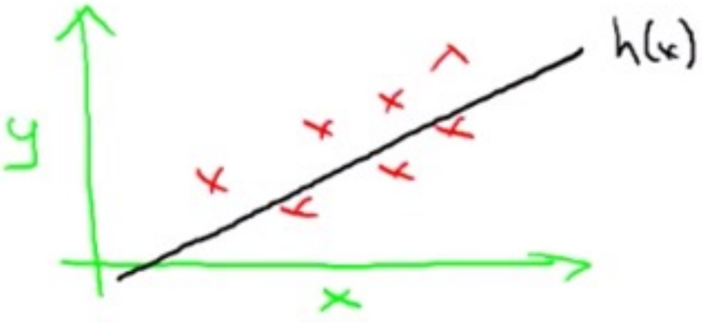


Model h

- To start with, we will use a simple model, a function which is the equation of a line (maybe you remember y = ax + b from school ?)

$$h(x) = \theta_0 + \theta_1 x$$

- This model will predict that y is some straight line function :



If this seems a bit odd to you…

- Remember we want our function to predict the examples we have in our training set correctly,
- which our simple model will probably not do very well….
- What if we can’t get to all the points using a straight line ?
- Don’t worry for now, this is still a good starting point !

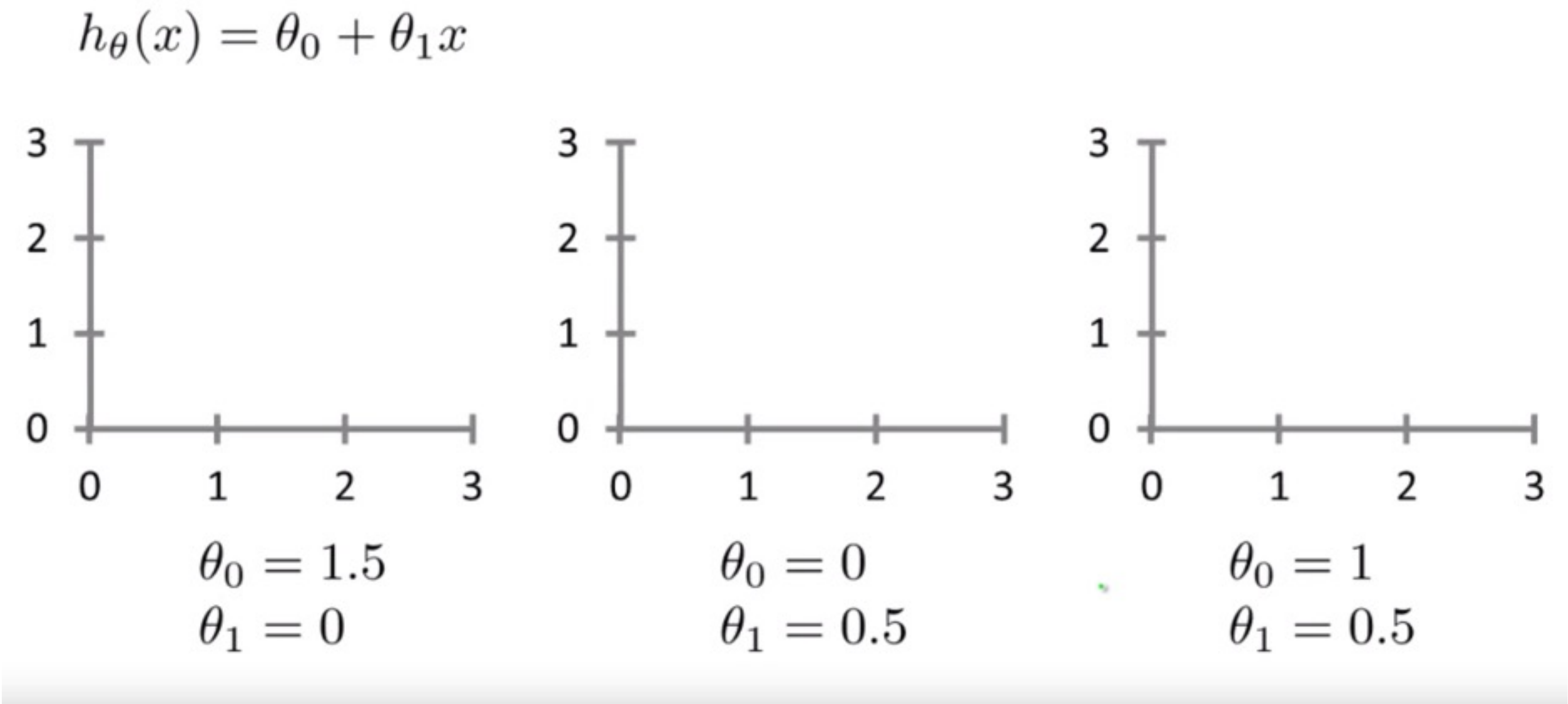
Cost function

- This is a second function we will use to judge how well our straight line is fitting the data and to find the best possible straight line.

- $h(x) = \theta_0 + \theta_1 x$

- $\theta_{i's}$ are what we call **parameters** and we want to find the right combination of those parameters to get the best line.
- So how do we choose the right parameters ?

Different parameter choices/hypotheses



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

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

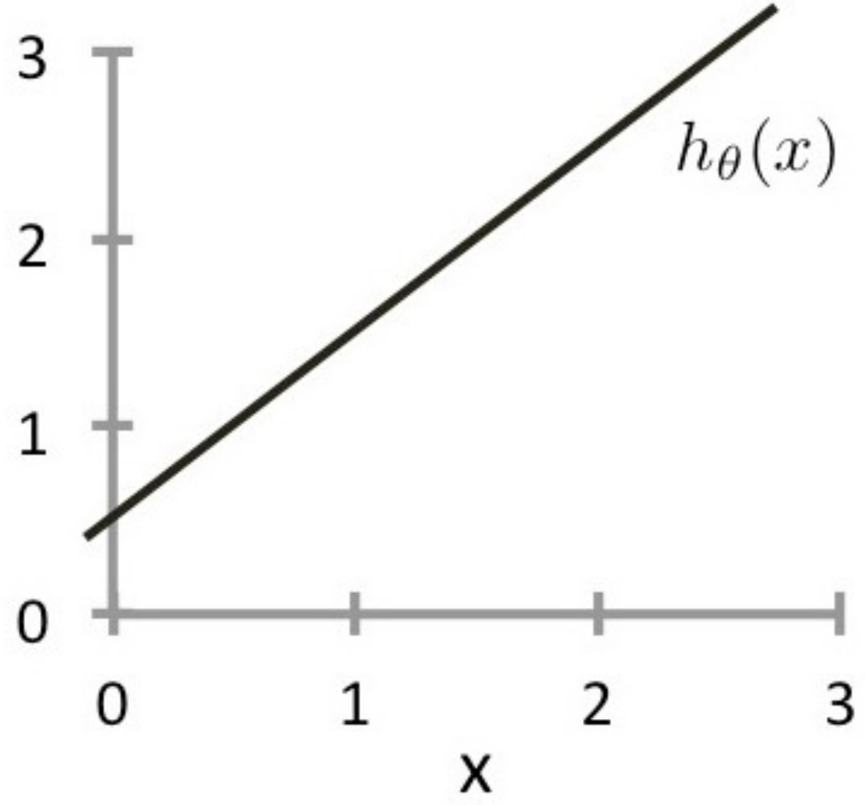
$\theta_0 = 0$
$\theta_1 = 0.5$

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

Exercise

- Look at the plot of $h(x) = \theta_0 + \theta_1 x$

- What are the values of θ_0 and θ_1 ?



Minimization Problem

- We want to choose θ_0 and θ_1 so that
- $h(x)$is close to y for out training examples (x, y)…
- So this is actually a **minimization problem**,
- where we want to minimize $(h(x) - y)^2$ by tweaking our parameters θ_0 and θ_1

Cost function = Quantifying the model's error

- The previous slide only took into account the error for a single example…
- So for all of our examples m the average error is :

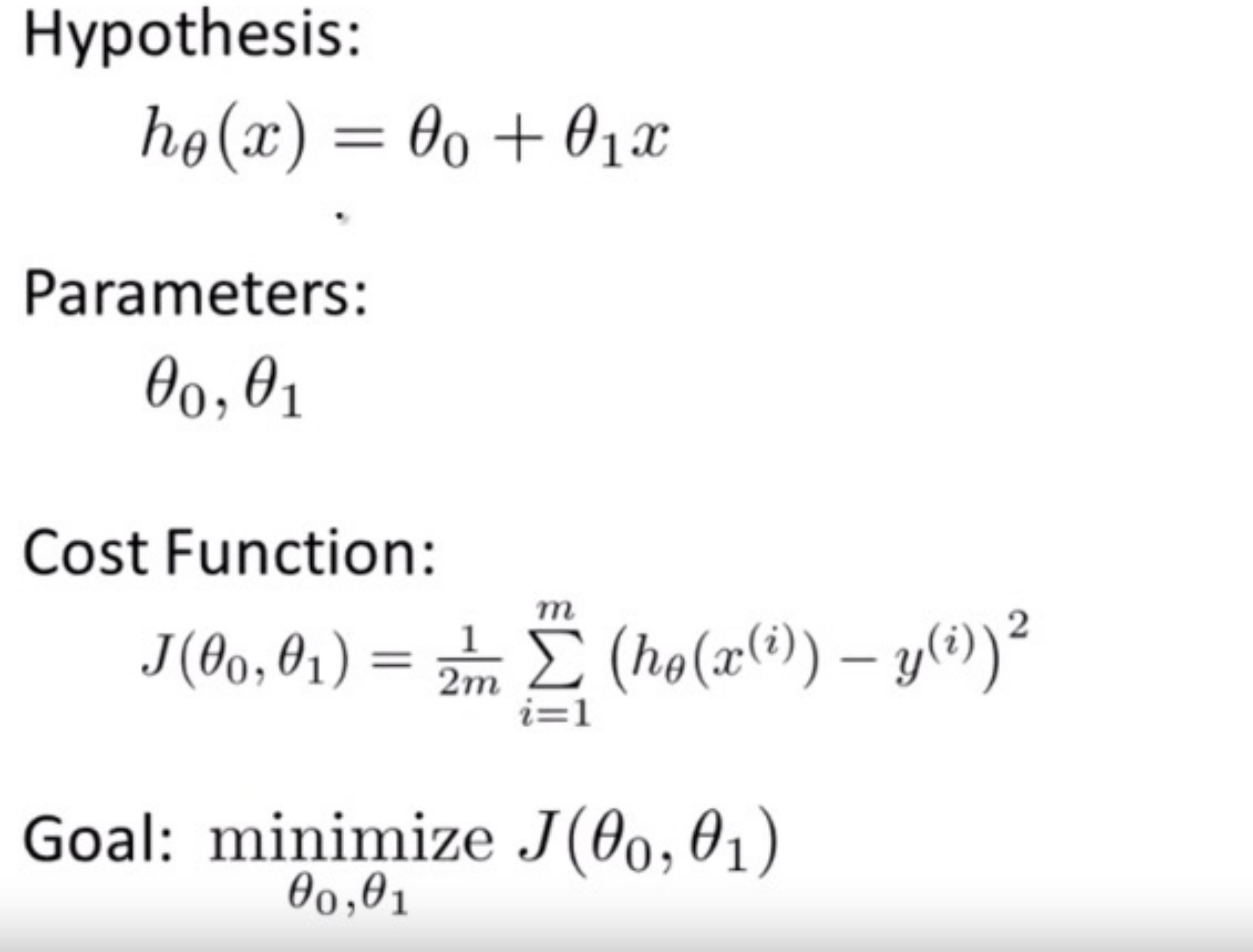
$$J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h\left(x^{(i)}\right) - y^{(i)})^2$$

The 2 is just there to make the math easier but doesn't change anything fundamentally, you can regard this as the average error.

- This function is known as the MSE (we'll see how it works in a few slides) and is the most commonly used:

Mean Squared Error

To recap



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

https://www.coursera.org/learn/machine-learning/home/welcome

Cost Function Intuition

- Let’s use a simplified model hypothesis to understand what’s going on:

$$h(x) \ = \theta_1 x$$

- Our objective is now to minimize

$$J(\theta_1)$$

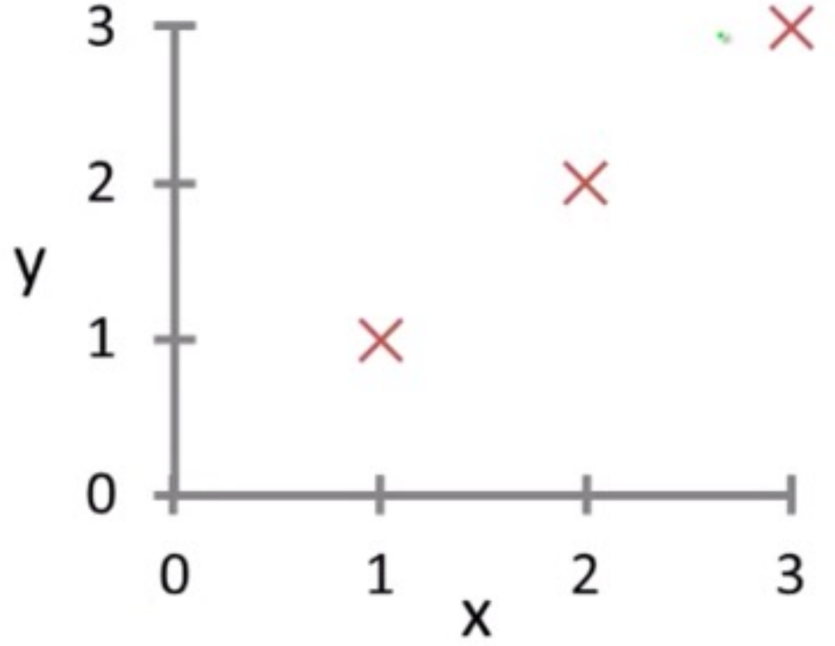
- And our cost function looks like

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

Hypothesis function vs. Cost function

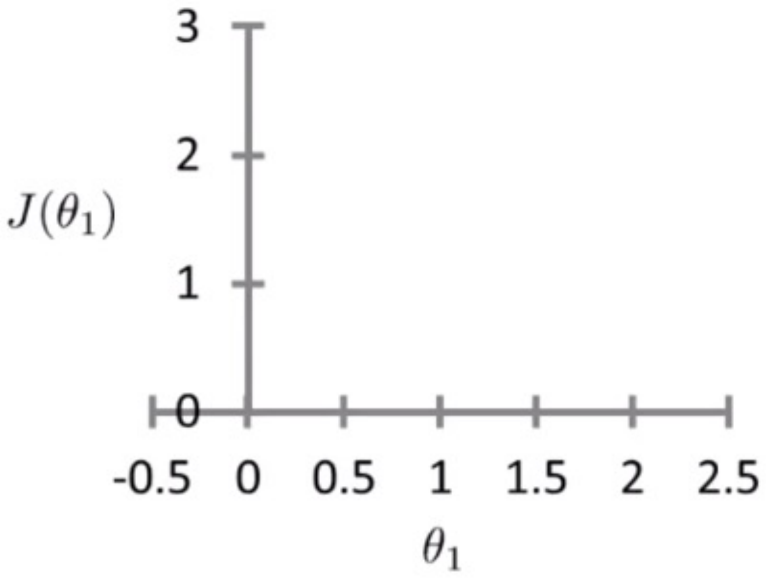
- If the points below represent our training data and $\theta = 1$, what does our hypothesis (line) look like ?
- What is the cost ? Let's find out !

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



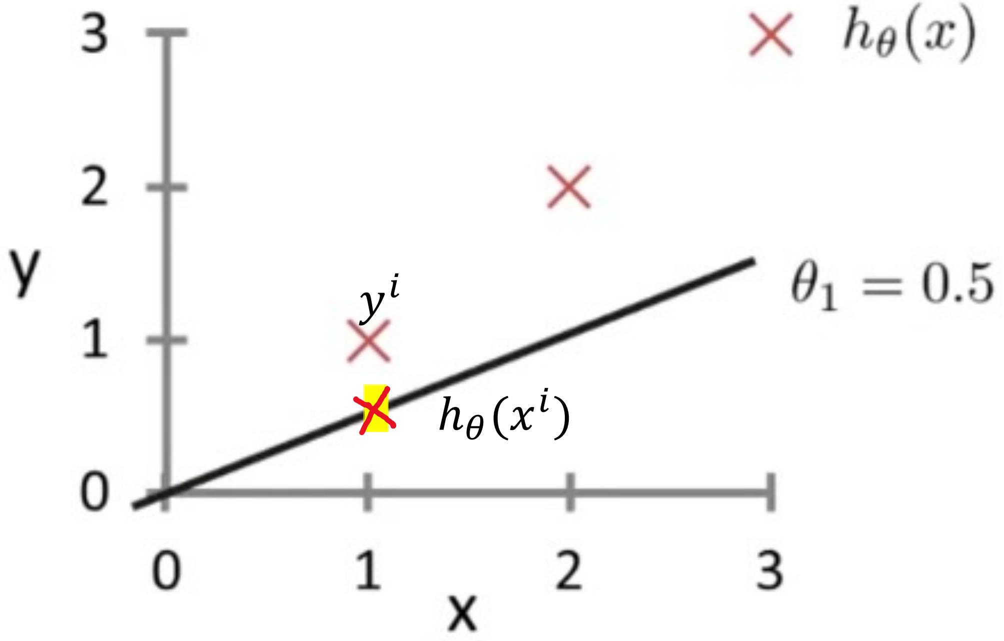
Hypothesis function vs. Cost function

- $J(\theta_1 = 1) = 0$
- We can now plot our error rate
- Notice that the values for θ_1 are on the horizontal axis. This is not the same graph as before !!



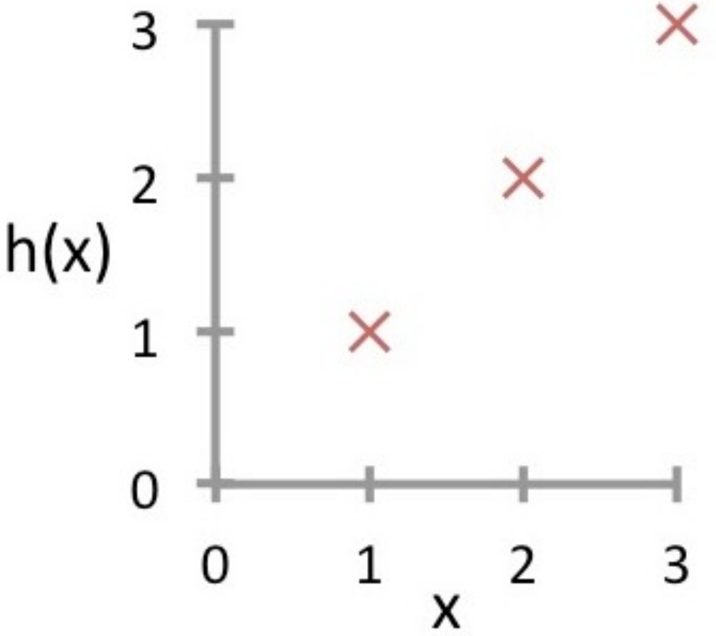
Hypothesis function vs. Cost function

- Now let’s look at $\theta_1 = 0.5$
- And compute $J(\theta_1 = 0.5)$ (approx. 0.58)
- The error for each point is actually the height wich seperates the data point and the line for a given x.



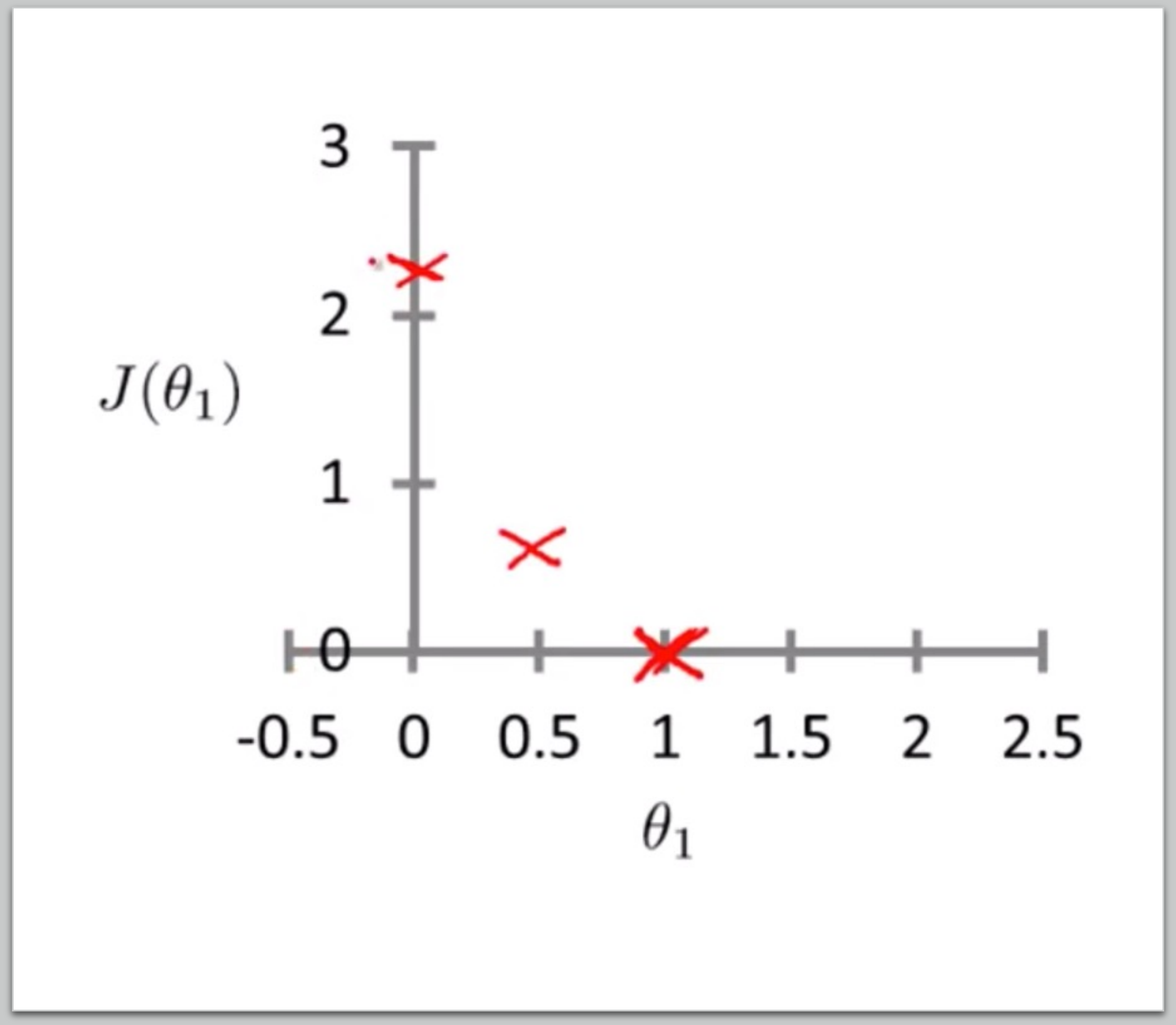
Your turn !

- Suppose this is our training set. $m = 3$.
- Given the same hypothesis and cost
- functions as before, what is $J(0)$?
- ie. $\theta_1 = 0$
- Should be approx. 2.3



Hypothesis function vs. Cost function

- We could continue plotting points but we'll stop here.
- With the error calculated for the different values of θ_1, we start to see part of the general shape of the function
- It turns out the function is convex/looks like a parabola.



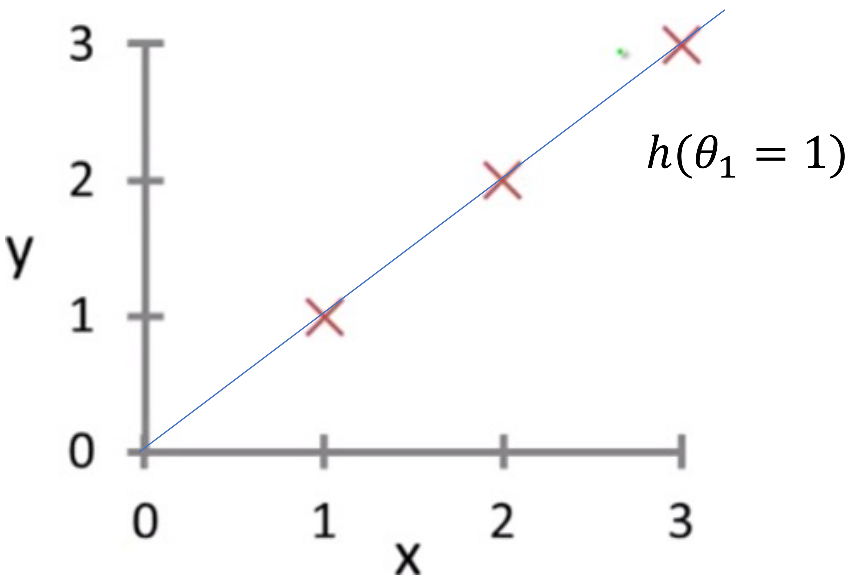
Quick recap

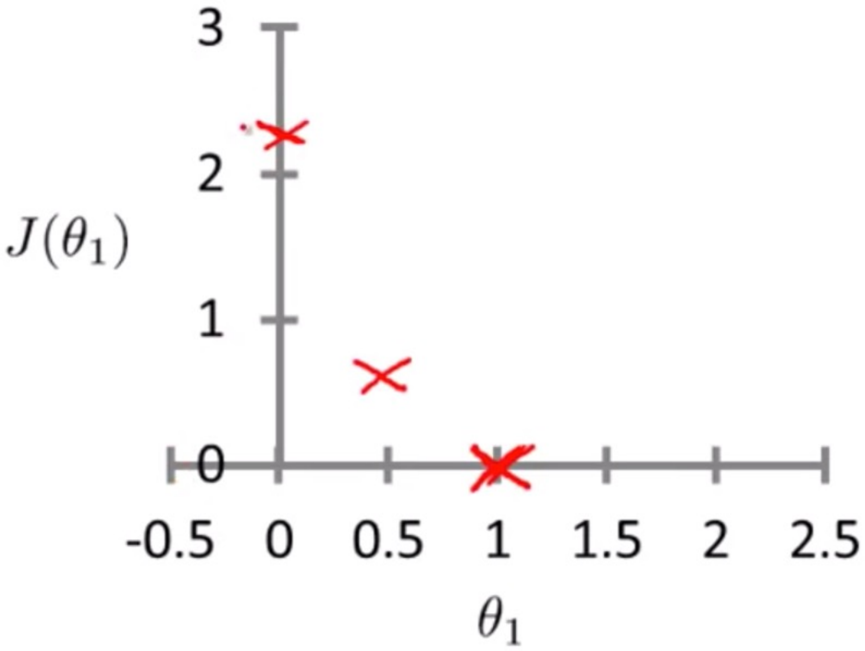
- Each value of $\theta_1$plotted corresponds to a different hypothesis / model / straight line on the data point graphs shown previously.

- For each value we can compute a value $J(\theta_1)$ to trace out the cost function.

- Now remember, we wanted to find the value of θ_1 which minimized $J(\theta_1)$… Looking at the graph we can now do so !

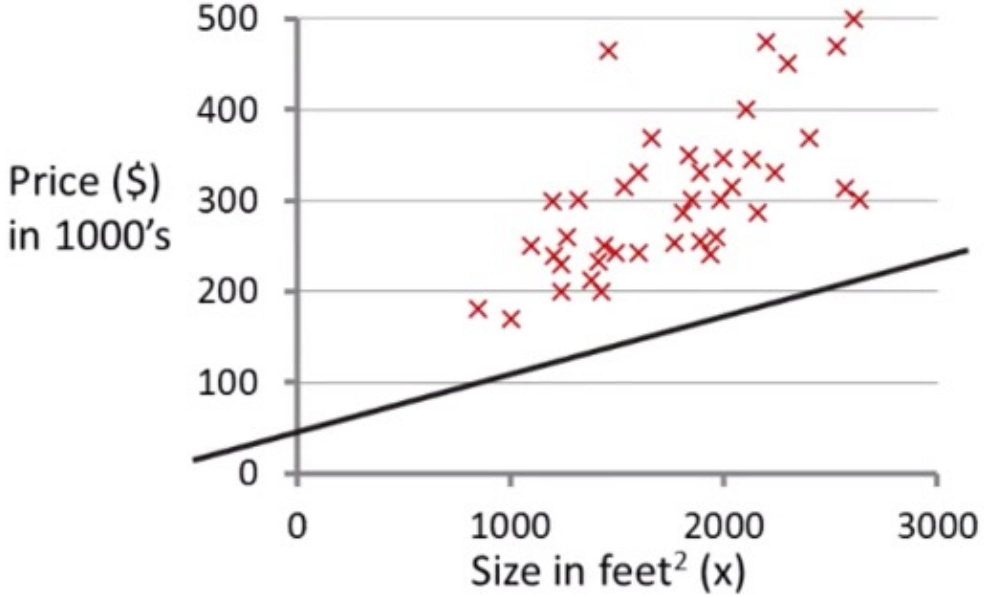
- No surprise, the value of θ_1 which minimizes the error, is associated with the model which fits the data perfectly





Back to 2 parameters

- Now we use our original, 2 parameter hypothesis to draw our line.
- For :
- $\theta_0 = 50$
- $\theta_1 = 0.06$
- We get this straight line as our model



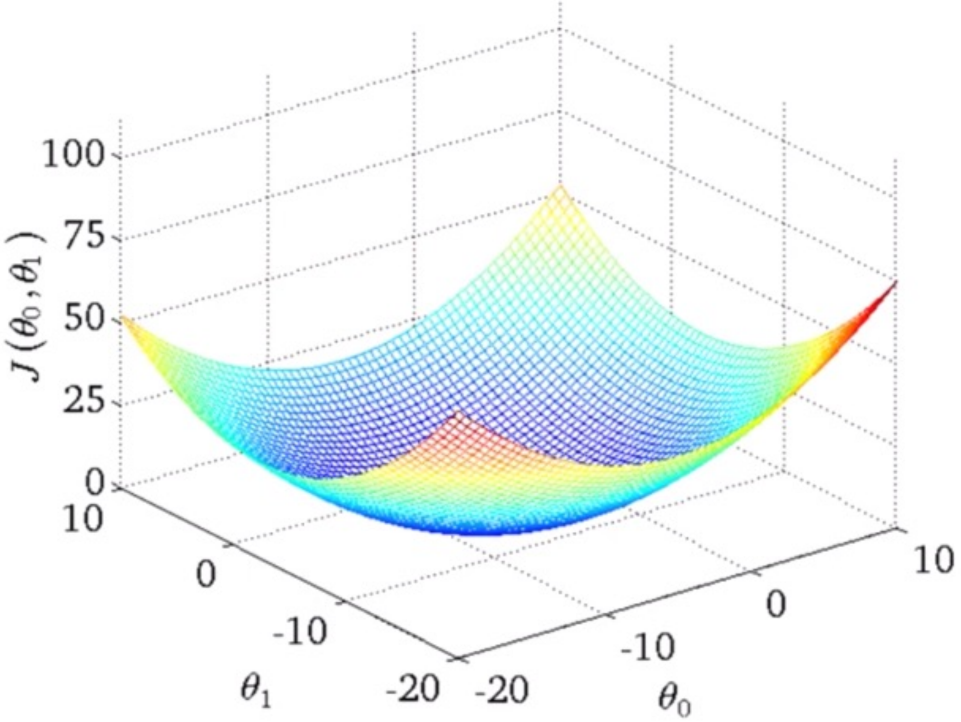
$h_\theta(x) = 50 + 0.06x$

Corresponding Cost function

- Now we have two parameters, the error graph will be slightly harder to plot as it has 3 dimensions:

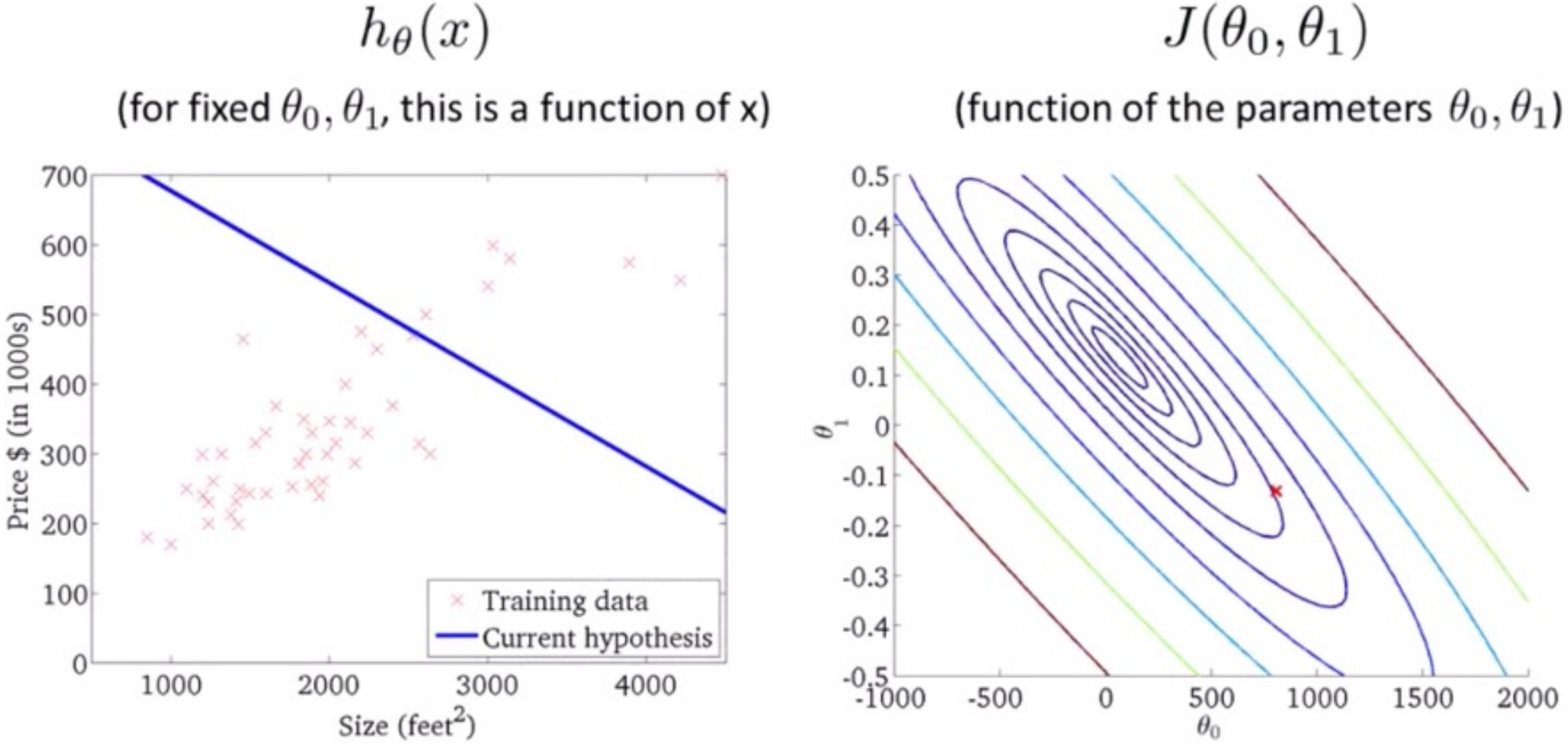
$$\theta_1, \theta_2, cost$$

- Indeed , $J(\theta_1, \theta_2)$ now has 2 inputs,
- So it will like this in 3D:



Contour Plots

- To stay in 2D, you will see the cost function represented by a contour plot :



$h_\theta(x)$

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

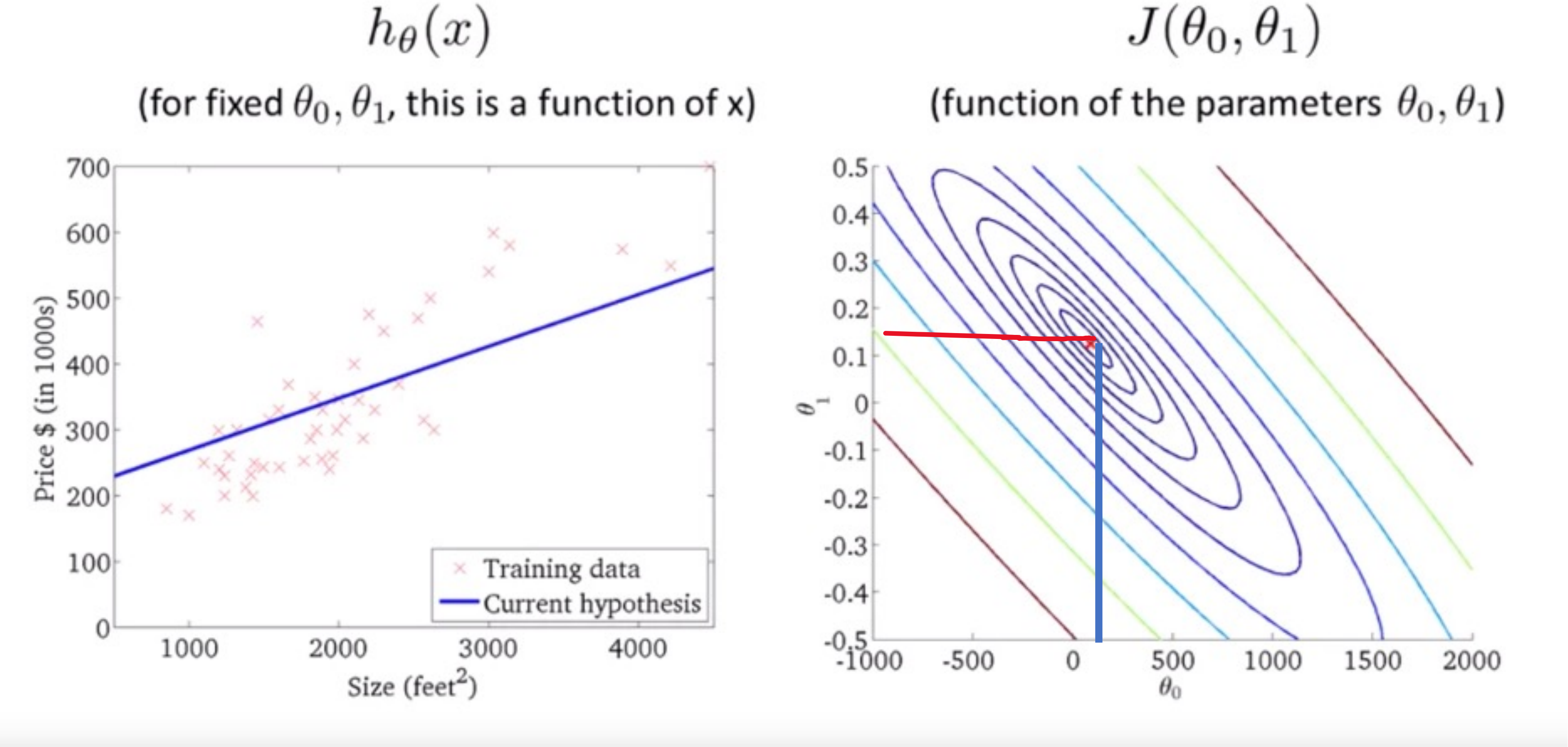
$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

The ovals/ellipses show the set of points which take on the same value for given values of θ_0, θ_1

Countour Plots

- The minimum is at the center of all the « ellipses ».
- This shows a model very close to the minimum.



Gradient Descent

- Now we know how to evaluate a model, using a cost function, how do we make the model *learn* the optimal parameters ?
- In other words, how do we minimize the cost function without testing all the different possible models ?

- The algorithm used to do this is called *Gradient Descent*, and is essential to most machine learning algorithms, not just linear regression !

Gradient Descent

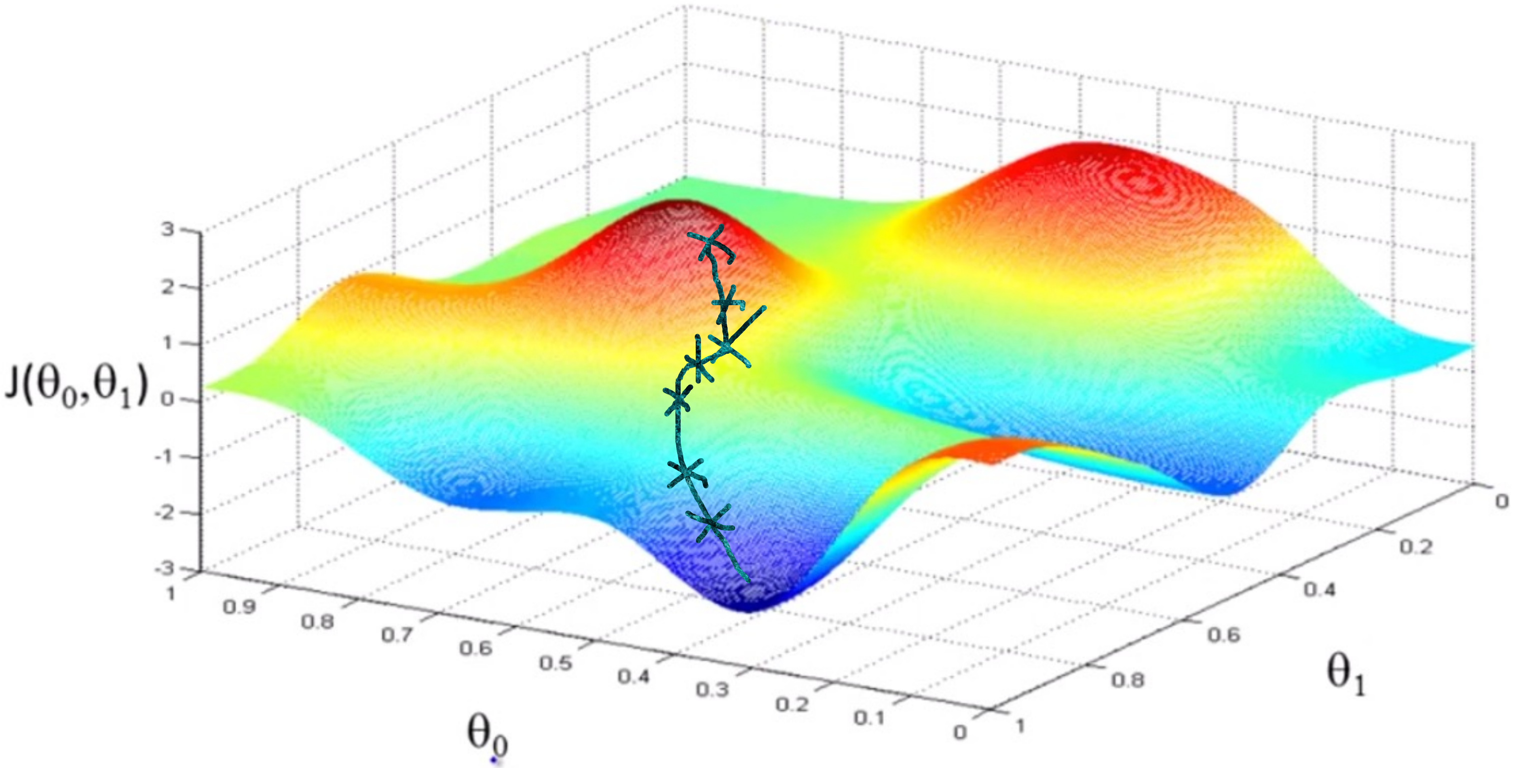
- We have some function $J(\theta_1, \theta_2)$
- Which we want to minimize… (ie. Find the minimum for)

- Outline :

  - Start with some inital guess, some random values for θ_1, θ_2
  - Keep updating θ_1, θ_2 a little bit to reduce $J(\theta_1, \theta_2)$ until we hopefully end up at a minimum

GD intuition

- This is your cost function in 3D
- Imagine you start somewhere near the top of one of the « hills » and your goal is to walk in the direction which will take you down to the bottom the fastest.



GD formula

```
repeat until convergence {
    $\theta_j := \theta_j - \alpha \boxed{\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)}$    (for $j = 0$ and $j = 1$)
}
```

- This is the update formula for each of the parameters
- := signifies assignment
- α is a number called the *learning rate.* If α is very large, then it corresponds to an aggressive learning procedure and big steps being taken « downhill » and vice versa.
- $\boxed{\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)}$ is a derivative term, for which we need to do a tiny bit of calculus !

GD Intuition

- Why does this update make sense ?
- Why are we putting those 2 terms together ?

- Let’s try and get a basic understanding of derivatives before we go any further.

Derivatives

- Disclaimer : This is not necessary for you guys to understand completely, but just so you have an inkiling of where the result comes from, we will go over certain essential points about derivatives.

Derivatives

- The derivative describes how the output of a function varies with regard to a very tiny tiny tiny variation in input, to the point where we consider almost/pretty much no variation in input…
- How can we describe how the ouput varies if the input is fixed…? We will take a look at this paradox.

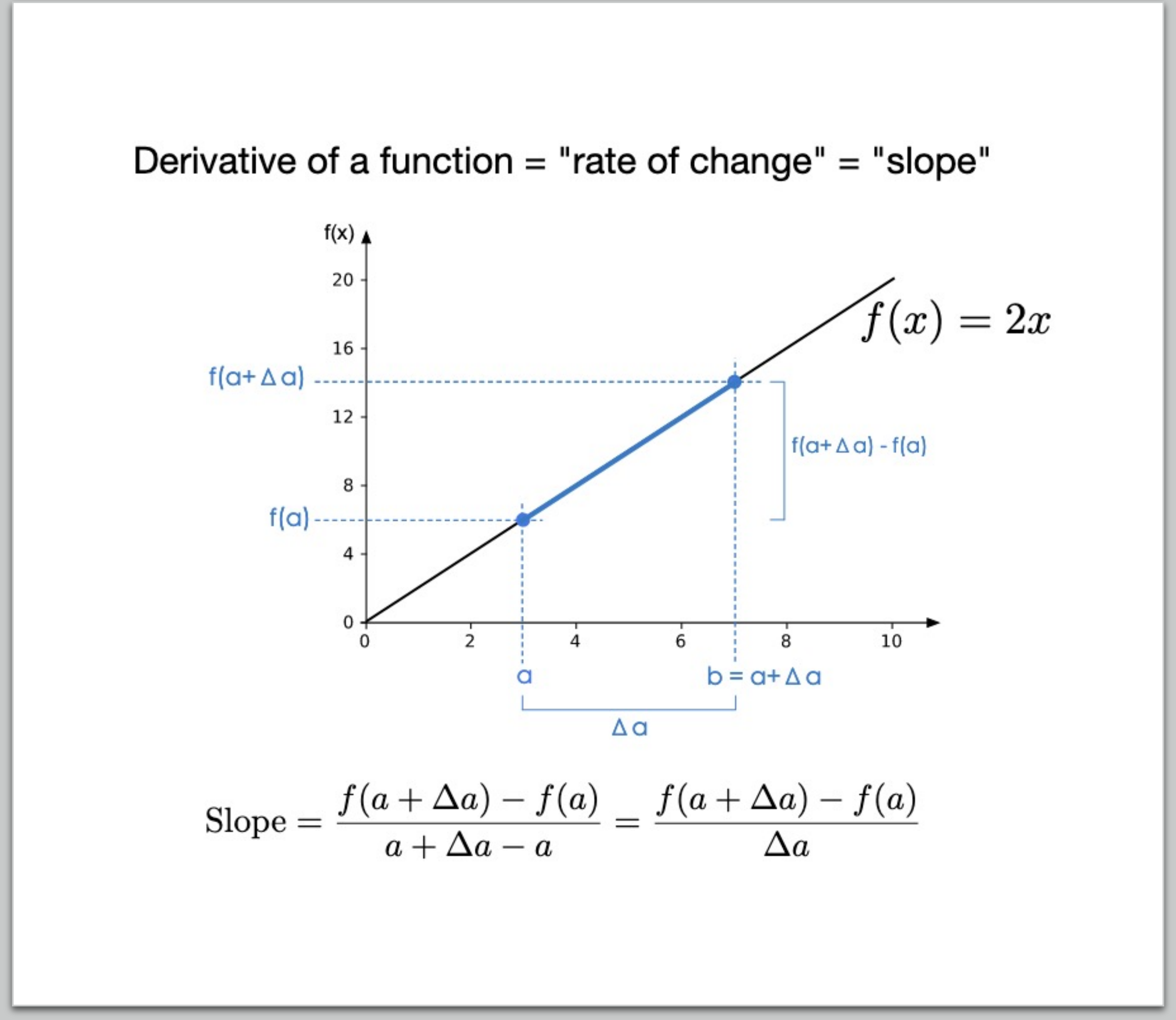
- But to start, let’s first look at a not so tiny change in input
- to familarise ourselves with describing how a function’s output varies with regard to the input.

Derivatives

- Let’s go through the calculation of the slope, using the formula
- $\frac{f(7)-f(3)}{7-3} = \frac{14-6}{4} = \frac{8}{4} = 2$
- Slope is equal to 2

- AKA : if we change the input by 1 unit,
- the output will change by 2 units => this is why it’s called the “**rate of change**”

- The **slope** tells us how the output changes relative to the input.
- What would the line look like if we had a *negative slope* ?



Derivative of a function = "rate of change" = "slope"

$$\text{Slope} = \frac{f(a+\Delta a) - f(a)}{a + \Delta a - a} = \frac{f(a+\Delta a) - f(a)}{\Delta a}$$

Derivatives

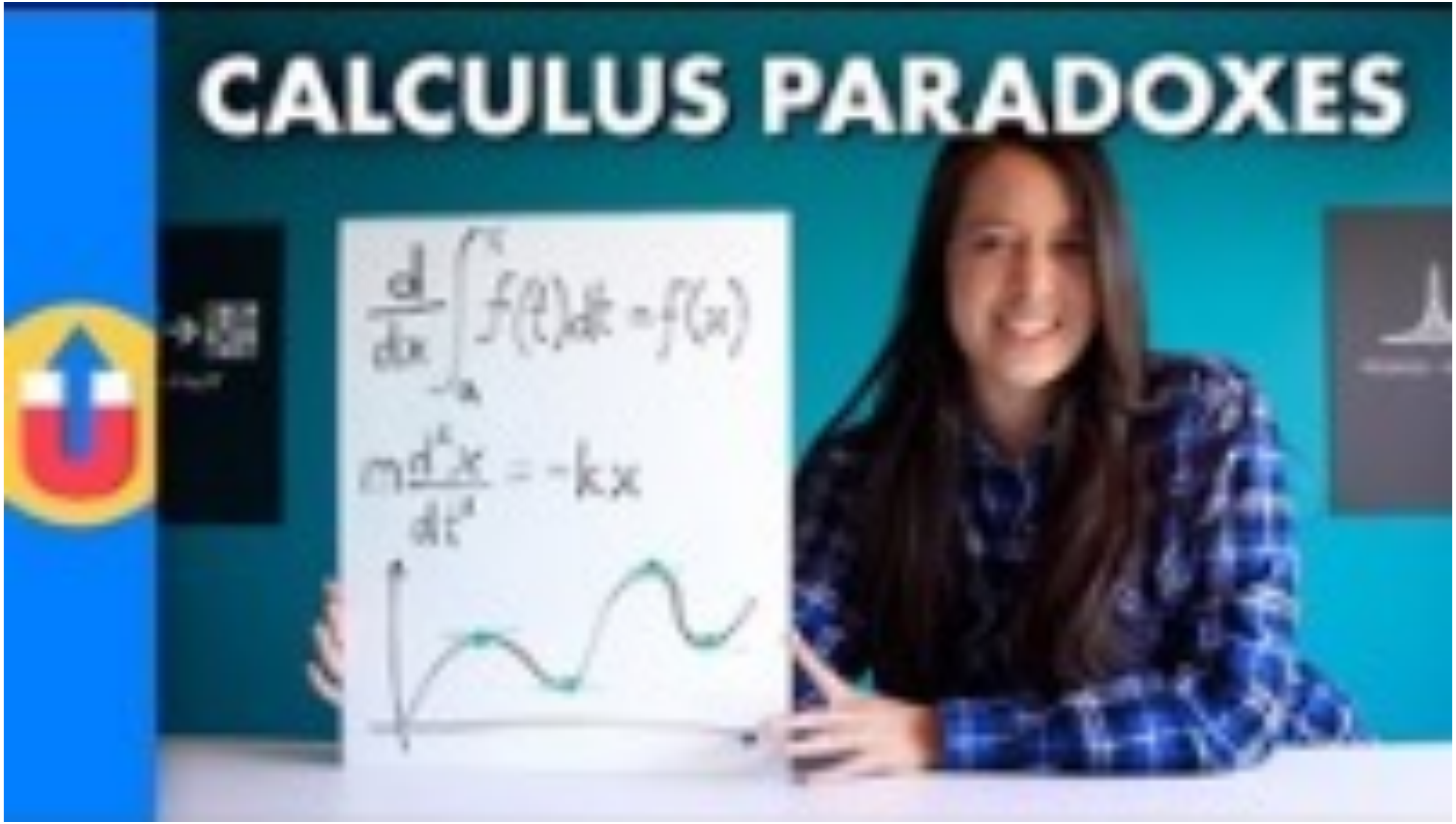
- Okay, so what happens as Δx becomes very very small (ie. very very close to 0) ?

- This is referred to as the « instantaneous rate of change »….

- This notion is quite paradoxical

Derivatives : Paradox

- Zeno’s Nerf Gun (8:46)



Derivatives : notation and using the limit

- So how does a tiny change in x affect the output ?

Lagrange notation

$$f'(x) = \frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

Leibniz Notation

<u>Example 1:</u> $f(x) = 2x$

$$\begin{aligned}\frac{df}{dx} &= \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}\\ &= \lim_{\Delta x \to 0} \frac{2x + 2\Delta x - 2x}{\Delta x}\\ &= \lim_{\Delta x \to 0} \frac{2\Delta x}{\Delta x}\\ &= \lim_{\Delta x \to 0} 2.\end{aligned}$$

Derivatives: a more elaborate function

$$f'(x) = \frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

<u>Example 2:</u> $f(x) = x^2$

$$\begin{aligned}
\frac{df}{dx} &= \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x} \\
&= \lim_{\Delta x \to 0} \frac{x^2 + 2x\Delta x + (\Delta x)^2 - x^2}{\Delta x} \\
&= \lim_{\Delta x \to 0} \frac{2x\Delta x + (\Delta x)^2}{\Delta x} \\
&= \lim_{\Delta x \to 0} 2x + \Delta x.
\end{aligned}$$

- In fact, as Δx approaches 0, the derivative
- Approaches 2x.

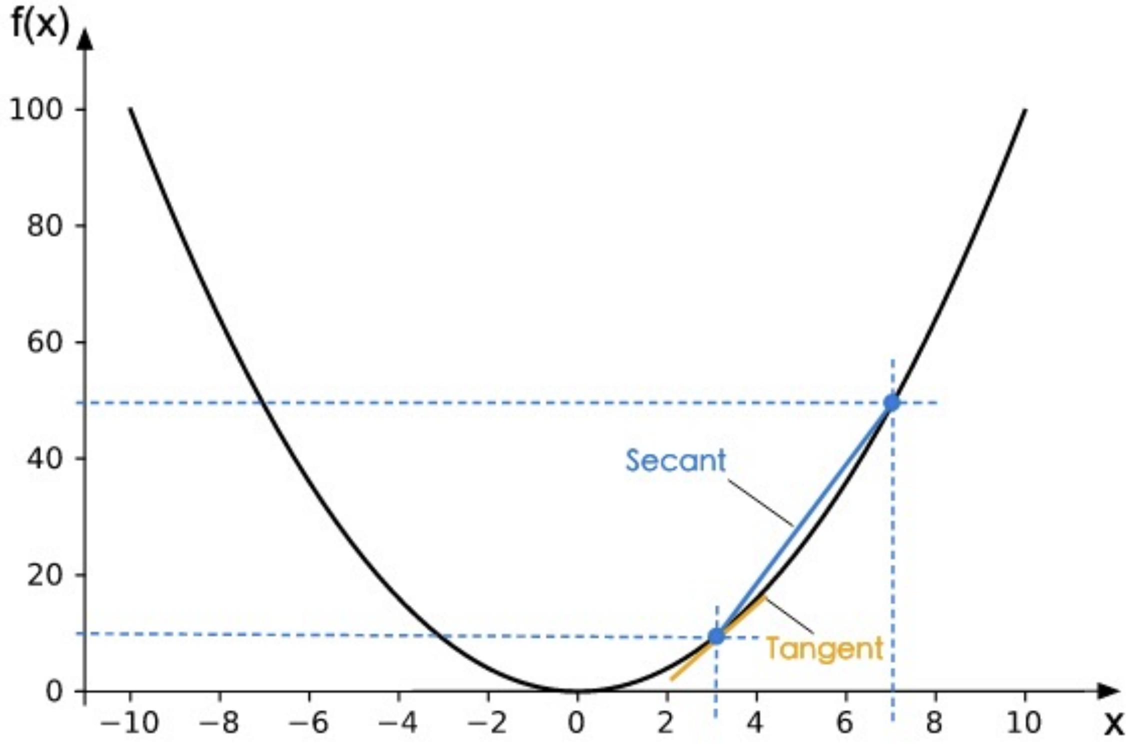
Rate of change (Optional)

- Putting the limit aside, this equation makes us calculate a ratio between a change in ouput and a change in input => our « rate of change » or « slope » from before
- So, for x^2 , if we change the input by 1 unit, the output changes by 2x + 1 units

- Let’s verify this with examples :
- $x = 1$
- $f(x+1) = f(1) + 1(2x + \Delta x) = 1 + 1(2\times 1 + 1) = 4 = 2^2$
- $f(x+2) = f(1) + 2(2x + \Delta x) = 1 + 2(2\times 1 + 2) = 9 = 3^2$
- $f(x+3) = f(1) + 3(2x + \Delta x) = 1 + \ 3(2\times 1 + 3) = 16 = 4^2$
- Etc…
- We’ve found a formula to compute how any change in input will modify the output for our function !

The derivative, visual intuition

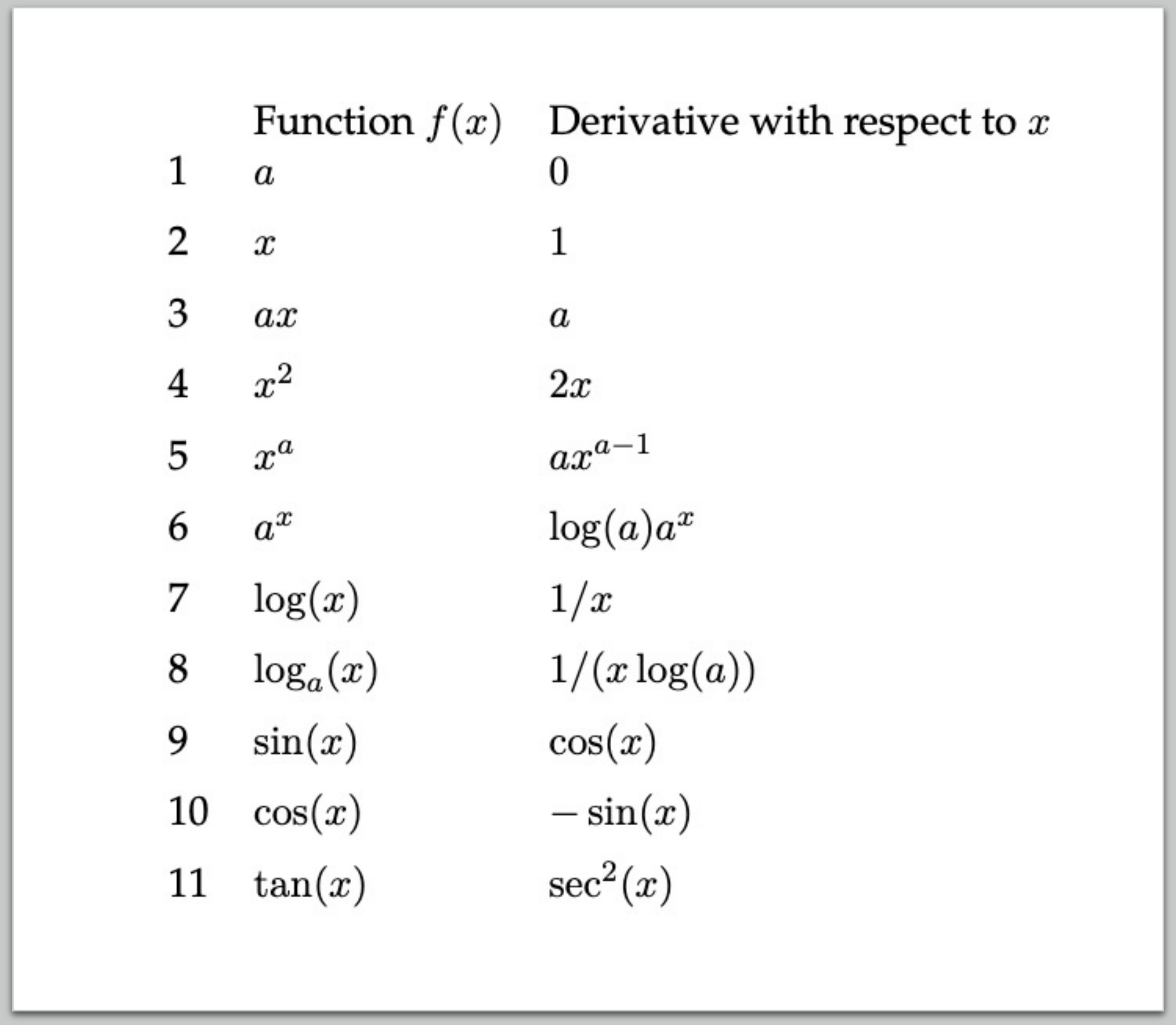
- As we bring the 2nd point closer on the plotted function, that secant line gets closer and closer to the line tangent to our point in x = 3
- The **slope** of this line is our **derivative**, our « *instantaneous rate of change* » !



Derivative rules 1 (Optional)

- Just so you are aware, no need to learn these by heart. But useful if you want to try and derive a function on your own !

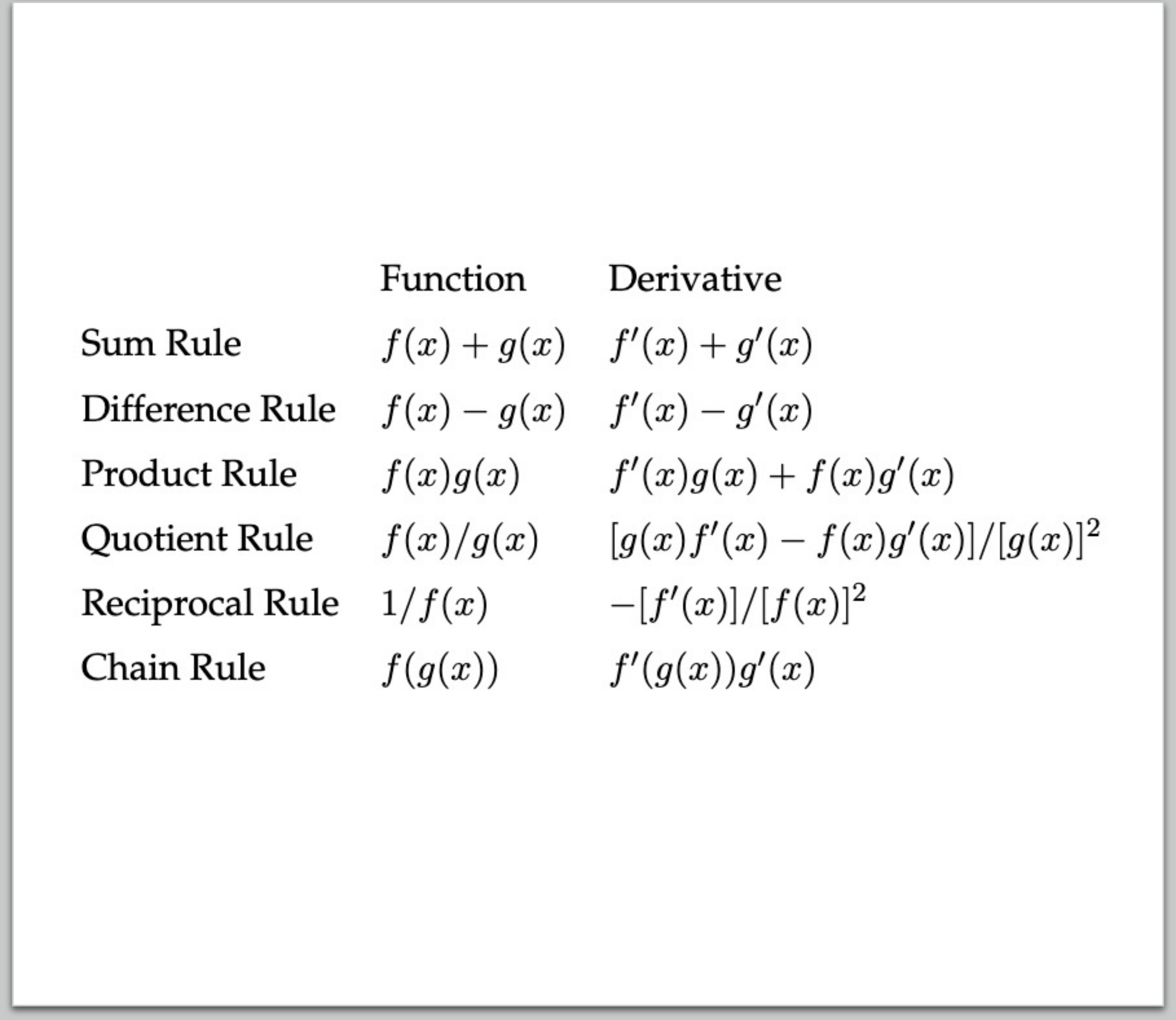
- You can find « cheatsheets » online if you need to.



	Function $f(x)$	Derivative with respect to x
1	a	0
2	x	1
3	ax	a
4	x^2	$2x$
5	x^a	ax^{a-1}
6	a^x	$\log(a)a^x$
7	$\log(x)$	$1/x$
8	$\log_a(x)$	$1/(x\log(a))$
9	$\sin(x)$	$\cos(x)$
10	$\cos(x)$	$-\sin(x)$
11	$\tan(x)$	$\sec^2(x)$

Derivative rules 2(optional)

- More useful rules



	Function	Derivative
Sum Rule	$f(x) + g(x)$	$f'(x) + g'(x)$
Difference Rule	$f(x) - g(x)$	$f'(x) - g'(x)$
Product Rule	$f(x)g(x)$	$f'(x)g(x) + f(x)g'(x)$
Quotient Rule	$f(x)/g(x)$	$[g(x)f'(x) - f(x)g'(x)]/[g(x)]^2$
Reciprocal Rule	$1/f(x)$	$-[f'(x)]/[f(x)]^2$
Chain Rule	$f(g(x))$	$f'(g(x))g'(x)$

Quick but useful example using the rules (optional)

- $f(a) = \frac{(ax\ -y)^2}{2}\ (x\ and\ y\ are\ constants)$
- Power rule : $x^2 \Rightarrow 2x$ (the one we saw earlier remember !)
- Scalar rule: $ax \Rightarrow a$ (we also saw this earlier !)

- Chain rule: $\frac{d}{dx}f\big(g(x)\big) \Rightarrow \frac{df}{dg}\times\frac{dg}{dx}$

(Optional)

- $f(a) = \frac{(ax\ -y)^2}{2}$
- Let’s decompose this into 3 functions:
  - $g(a) = ax\ \ - y$
  - $h(X) = X^2\ where\ X = (ax - y)$
  - $i(Z) = \frac{Z}{2}\ where\ Z = (ax - y)^2$

- Let’s derive these functions 1 by 1 using the rules in the previous slides.

(Optional)

- $g'(a) = \frac{d}{da}(ax - y) = x$
- $h'(X) = \frac{\mathrm{d}}{dX}(\mathrm{X}^2) = \ 2\mathrm{X}$
- $i'(Z) = \frac{d}{dZ}\left(\frac{1}{2}Z\right) = \frac{1}{2}$

- Using the chaine rule, we multiply these dervatives to get the derivative of our original function :

$$f'(a) = \frac{di}{dZ} \times \frac{dh}{dX} \times \frac{dg}{dx}$$

$$= \frac{1}{2} \times 2(ax - y) \times x$$

$$= (ax \ - y) \times x$$

GD Intuition

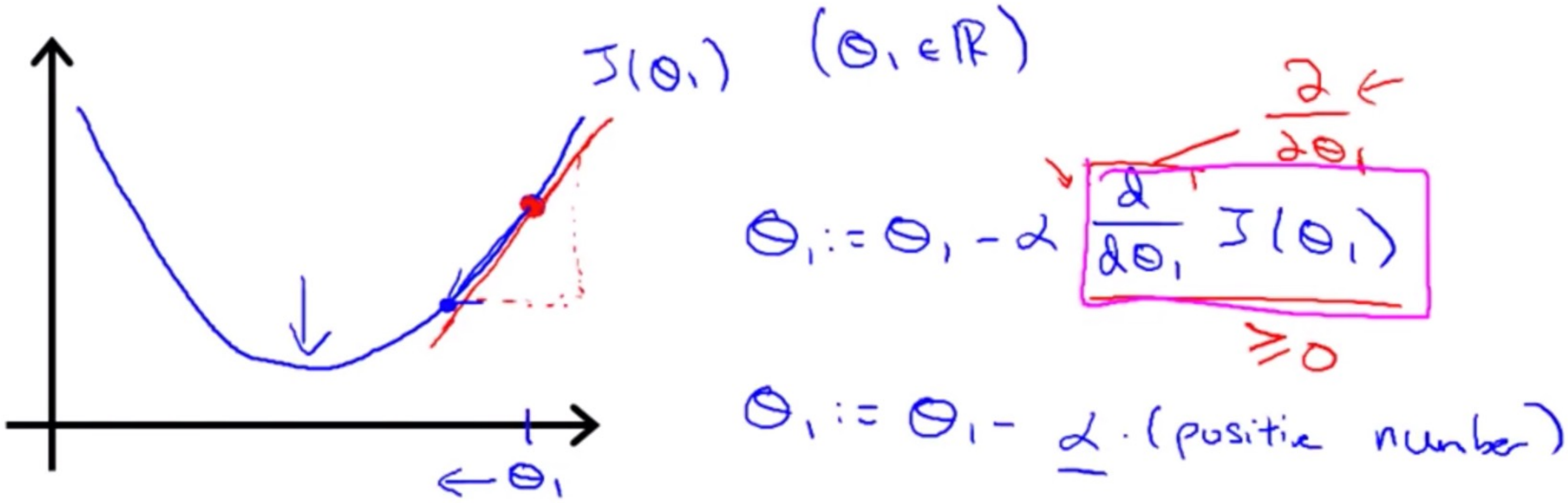
- Now we have a basic undrestanding of derivatives, let’s use a simpler example, whith a cost function of only 1 parameter.

- $J(\theta_1)$ instead of $J(\theta_1, \theta_2)$
- Same as previously when we wanted some intuition about the cost function.

- Let’s look at a couple scenarios to see what Gradient Descent does to our parameter.

When the derivative is positive…

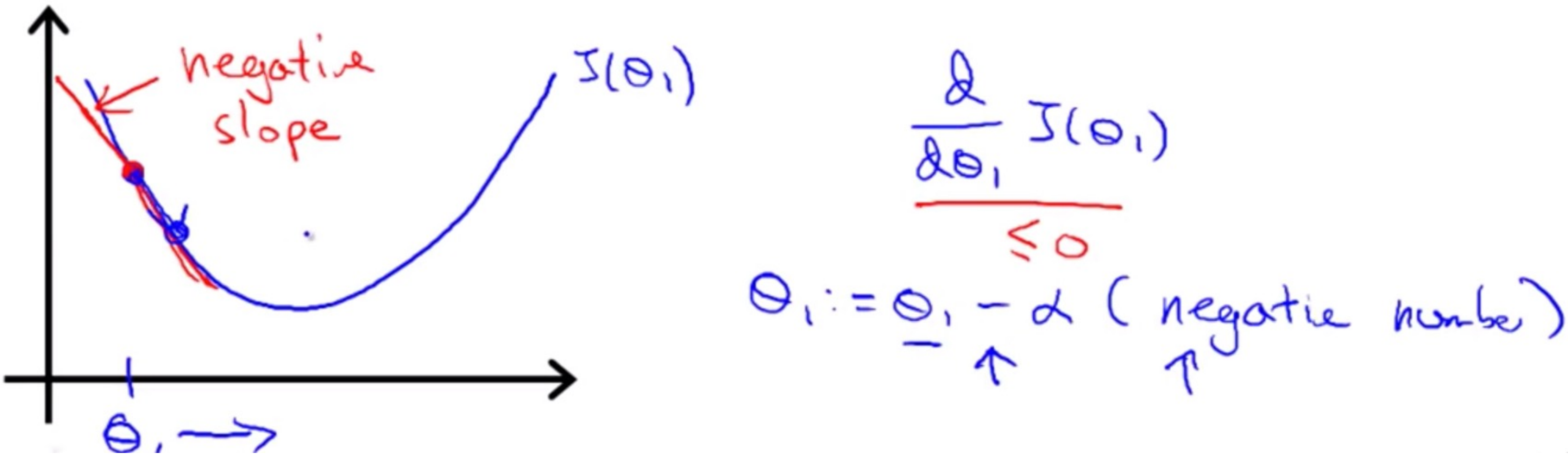
- Remember, our cost function looks like a parabola.
- When θ_1 is too high, let’s see if Gradient Descent reduces it and brings it closer to the « sweet spot », where tht cost is minimized
- Let’s see if it does the right thing :


$J(\theta_1)$
$(\theta_1 \in \mathbb{R})$
$\frac{\partial}{\partial \theta_1}$
$\theta_1 := \theta_1 - \alpha \boxed{\frac{d}{d\theta_1} J(\theta_1)}$
≥ 0
$\theta_1 := \theta_1 - \underline{\alpha} \cdot (\text{positive number})$
$\leftarrow \theta_1$

See Andrew Ng’s course

When the derivative is negative…

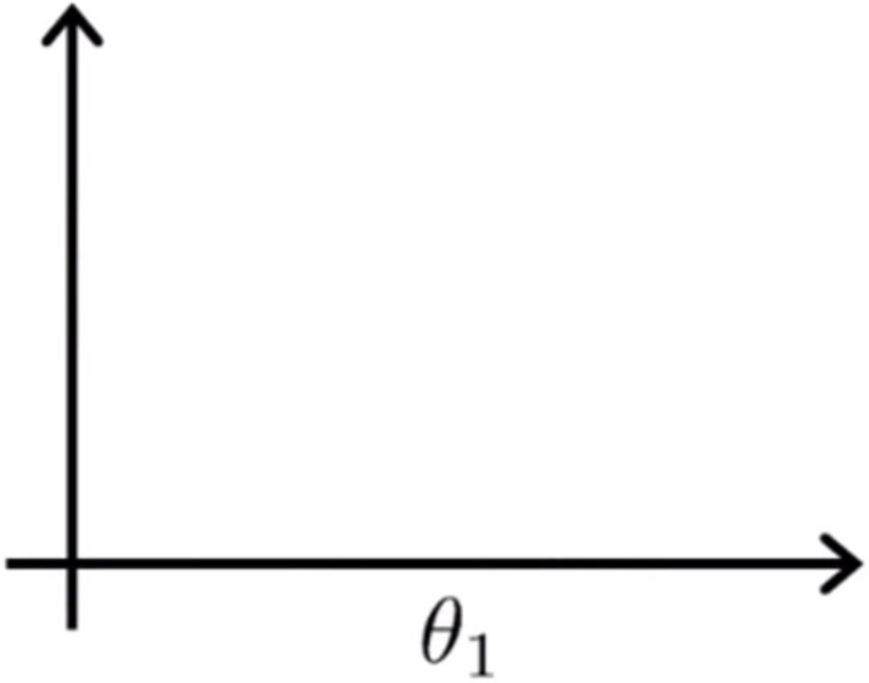
- When $\theta_1$is too low, let’s see if Gradient Descent increases it and brings it closer to the « sweet spot », where tht cost is minimized
- Again, let’s see if it does the right thing :



negative slope
$J(\theta_1)$
$\theta_1 \longrightarrow$
$\frac{\partial}{\partial\theta_1} J(\theta_1)$
≤ 0
$\theta_1 := \underline{\theta_1} - \alpha\,($negatie numbe$)$

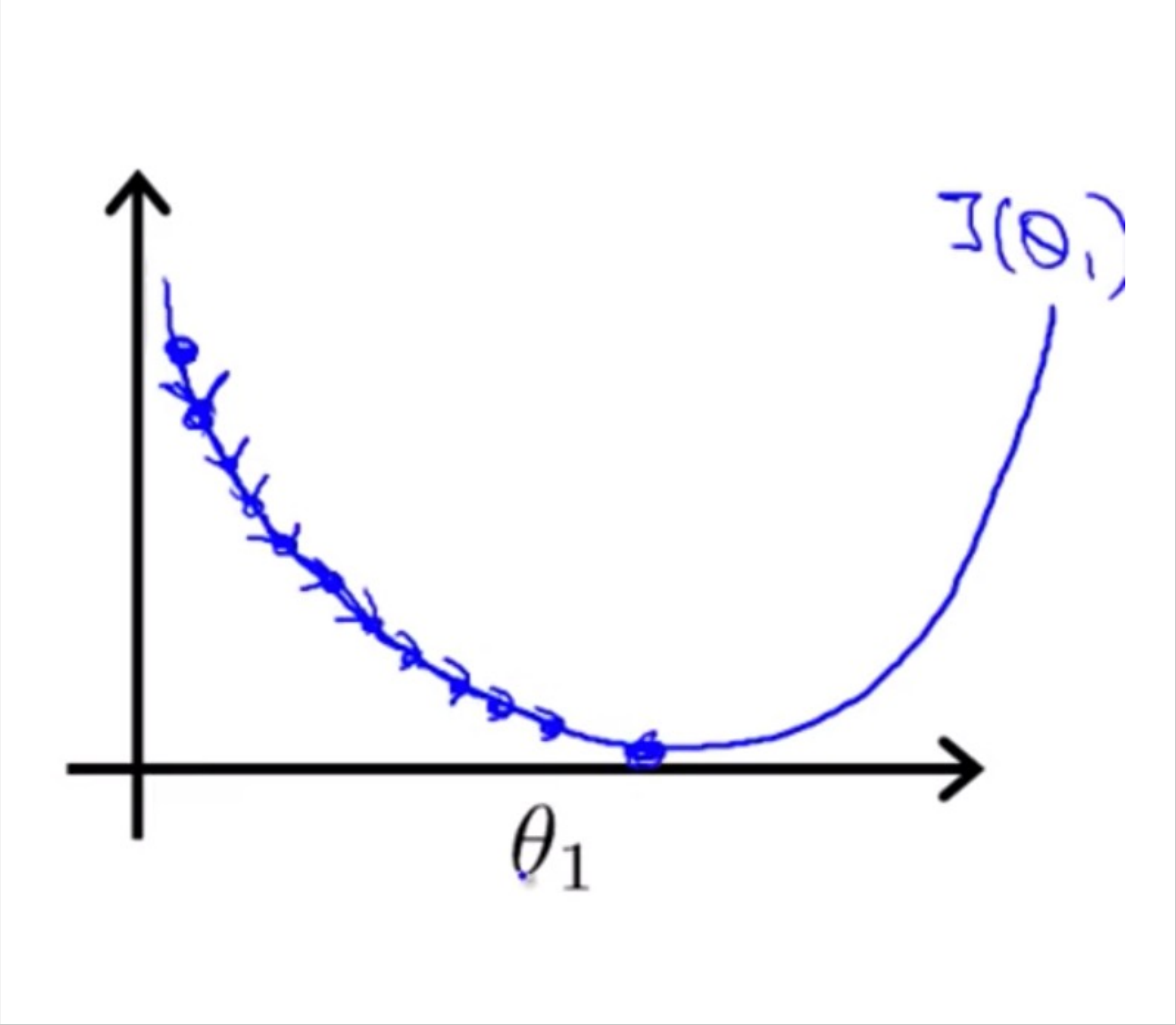
Okay so now what about α ?

- Remember the update rule : $\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$
- How does α influence how we update our parameter θ_1 ?
- If α is too small :



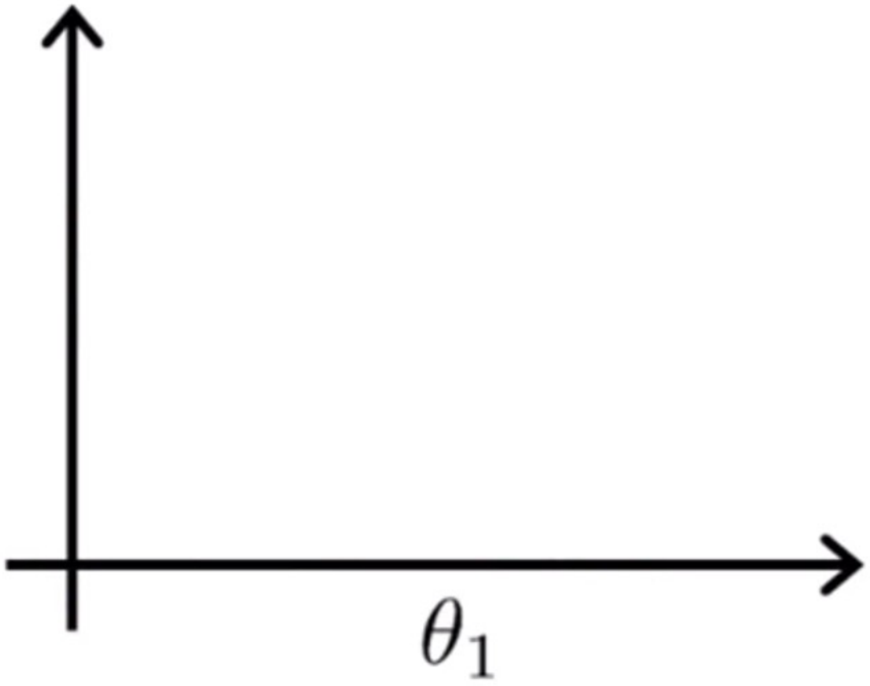
If too alpha too small

- Many small steps will be taken , which maks Gradient Descent very slow

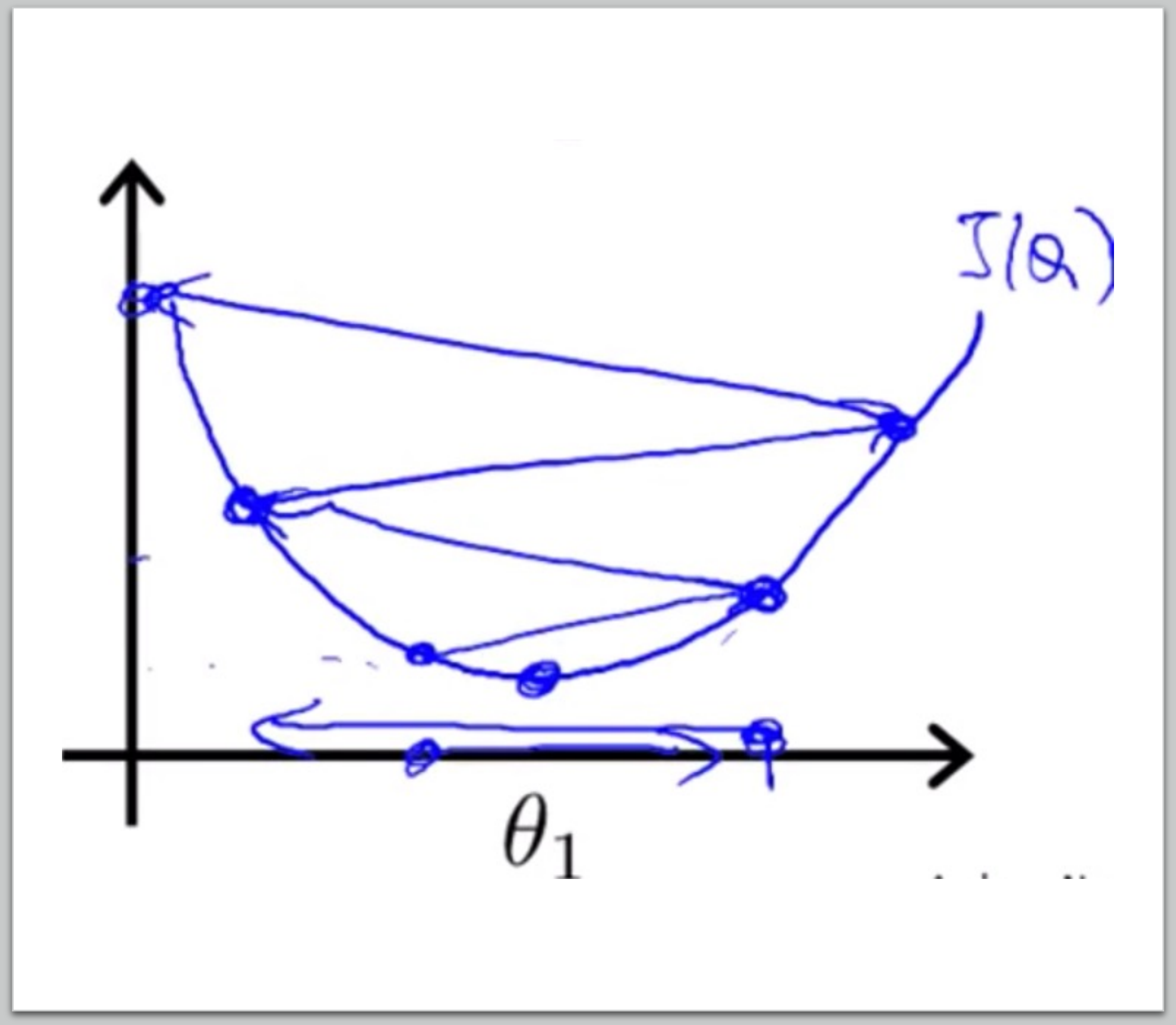


If alpha too large…

- Gradient descent may « overshoot », pass the minimum. It may even never converge (never find the minimum) and keep jumping around.

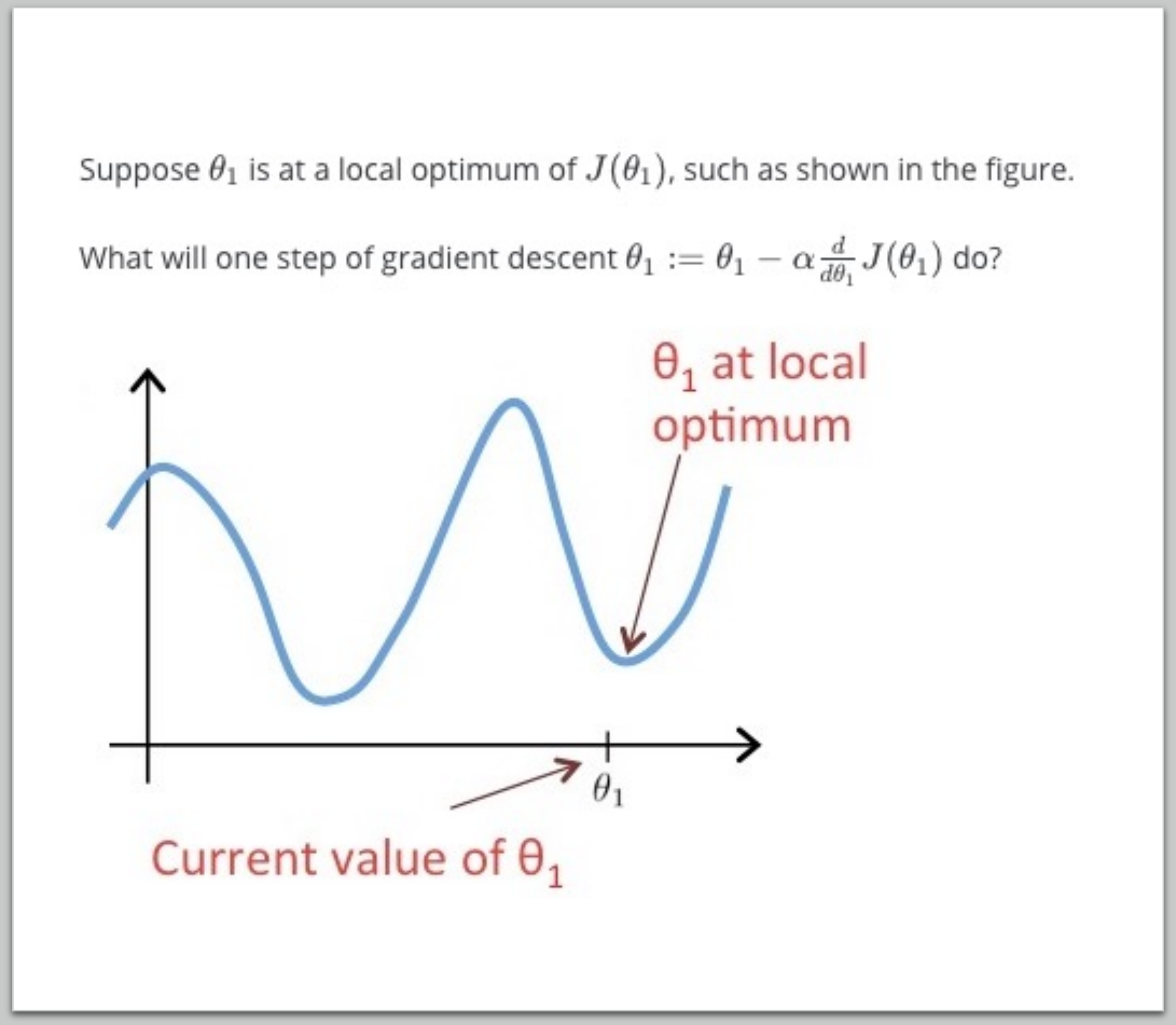


If alpha too large



Question

- Leave $\theta_1$unchanged ?
- Change θ_1 in a random direction ?
- Move θ_1 in the direction of the global minimum of $J(\theta_1)$?
- Decrease θ_1 ?

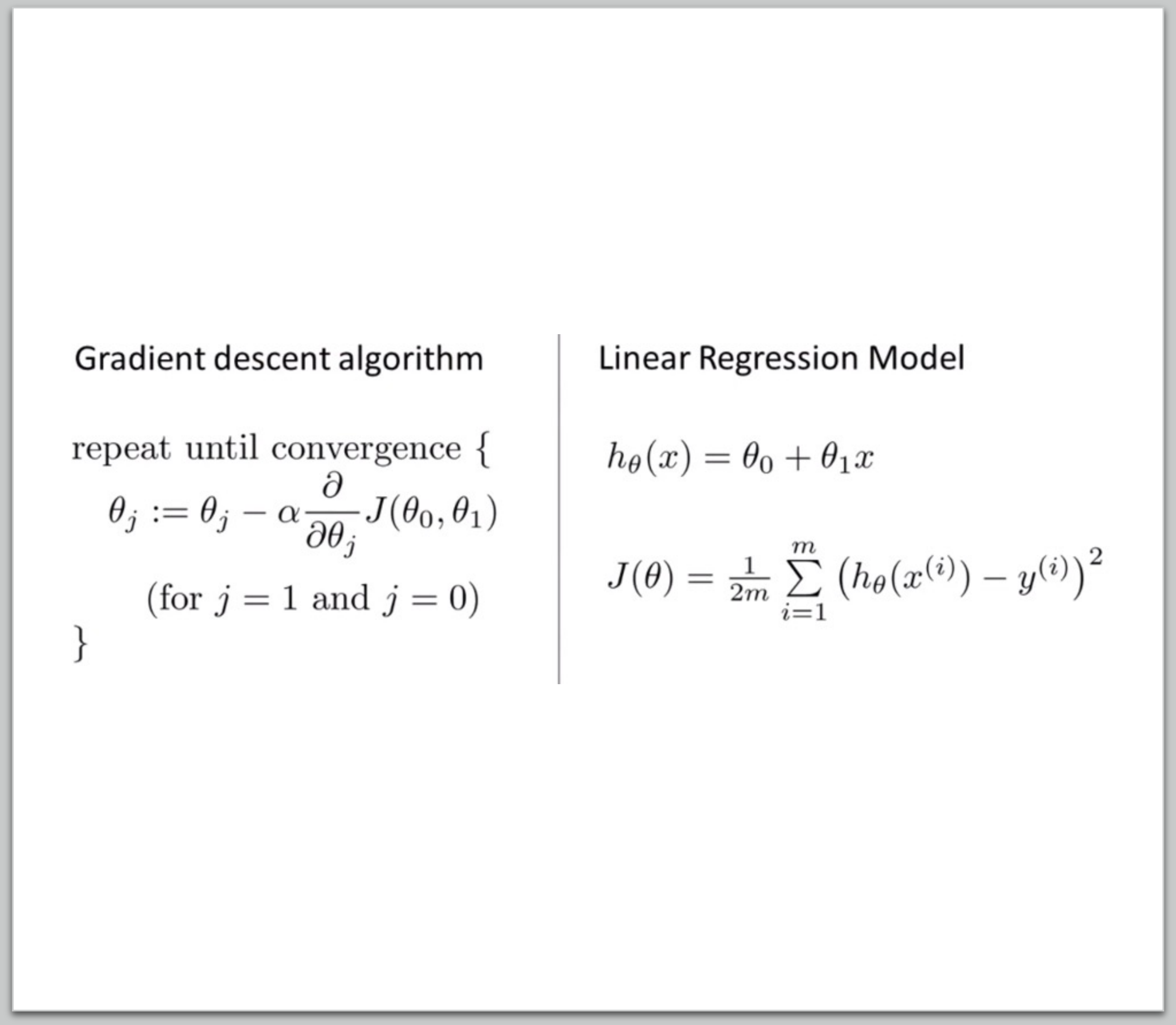


Suppose θ_1 is at a local optimum of $J(\theta_1)$, such as shown in the figure.

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

Piecing everything together

- You now have an intuition of what the algorithm is doing, so let's go through the update together
- This is all we need :
    - A hypothesis function (our model)
    - A cost function (to tell us how well/bad our model is doing)
    - Gradient Descent (to update our parameters and get closer to a better model)



Gradient descent algorithm

```
repeat until convergence {
    $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$
        (for $j = 1$ and $j = 0$)
}
```

Linear Regression Model

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

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

Computing the derivative term
Derivatives vs. Partial derivatives

- Instead of having a cost function with a single input, we are back to 2 inputs, our 2 parameters $\theta_1$and θ_2.

- This means that to know the function's « **rate of change** », we now have to look at how **each parameter** *impacts or affects* the function.

- How does a tiny change in $\theta_1$change $J(\theta_1, \theta_2)$?

- How does a tiny change in $\theta_2$change $J(\theta_1, \theta_2)$?

- We need to compute the *partial derivatives* of the cost function.

Derivatives vs. Partial derivatives

- *Partial Derivative :*

This comes down to calculating the derivative for each variable, treating the other variable as a constant, something fixed.

- The partial derivative is sometimes referred to as *the slope of a slice of a 3D graph,* but we won’t get into this. (if you want to know more)