

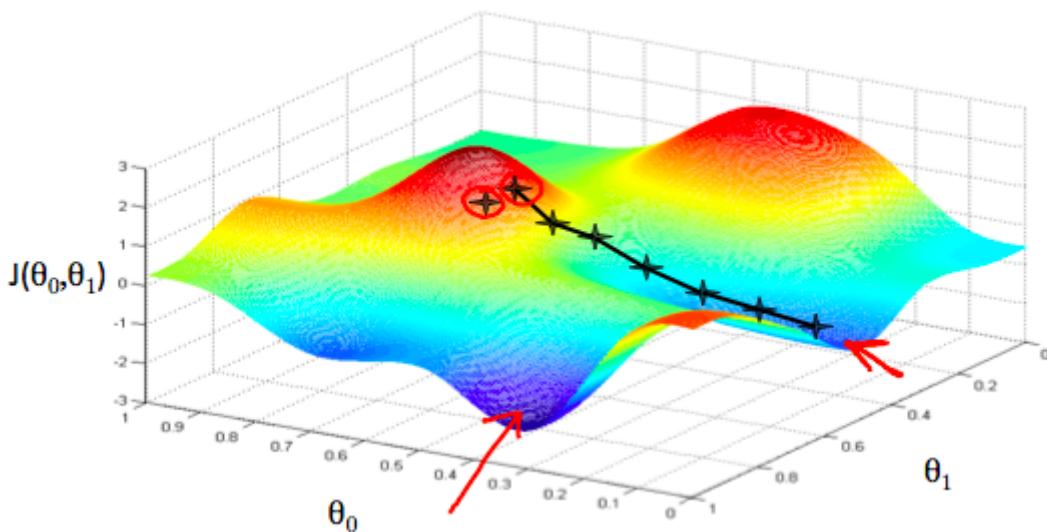
## 3. Parameter Learning

### 3.1 Gradient Descent

So we have our hypothesis function and we have a way of measuring how well it fits into the data. Now we need to **estimate the parameters in the hypothesis function**. That's where gradient descent comes in.

Imagine that we graph our hypothesis function based on its fields  $\theta_0$  and  $\theta_1$  (actually we are graphing the cost function as a function of the parameter estimates). We are not graphing  $x$  and  $y$  itself, but the **parameter range** of our **hypothesis function** and the **cost** resulting from selecting a particular set of parameters.

We put  $\theta_0$  on the  $x$  axis and  $\theta_1$  on the  $y$  axis, with the cost function on the vertical  $z$  axis. The points on our graph will be the result of the cost function using our hypothesis with those specific theta parameters. The graph below depicts such a setup.



We will know that we have succeeded when our cost function is at the very bottom of the pits in our graph, i.e. when its value is the minimum. The red arrows show the minimum points in the graph.

The way we do this is by taking the **derivative** (the tangential line to a function) of our **cost function**. The slope of the tangent is the derivative at that point and it will give us a direction to move towards. We make steps down the cost function in the direction with the steepest descent. The size of each step is determined by the parameter  $\alpha$ , which is called the **learning rate**.

For example, the distance between each 'star' in the graph above represents a step determined by our parameter  $\alpha$ . A smaller  $\alpha$  would result in a smaller step and a larger  $\alpha$  results in a larger step. The direction in which the step is taken is determined by the **partial derivative** of  $J(\theta_0, \theta_1)$ . Depending on where one starts on the graph, one could end up at different points. The image above shows us two different starting points that end up in two different places.

The gradient descent algorithm is:

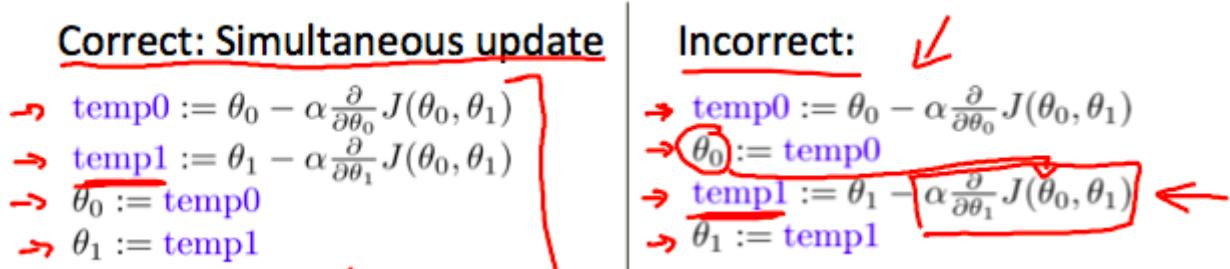
You can actually use the markdown mode for the cell and use the normal HTML code, as in

Repeat until convergence:

$$\left. \begin{array}{l} \\ \theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \\ \end{array} \right\}$$

where  $j = 0, 1$  represents the feature index number.

At each iteration  $j$ , one should simultaneously update the parameters  $\theta_1, \theta_2, \dots, \theta_n$ . Updating a specific parameter prior to calculating another one on the  $j^{th}$  iteration would yield a wrong implementation:



The gradient descent is a general ML algorithm, not used just in linear regression. We will later use it to minimise other functions as well (not just the cost function  $J$  for the linear regression; i.e.  $J$  can be an arbitrary function that the gradient descent is able to minimise).

To recap the problem solved with gradient descent algorithm (note: only two parameters are considered for the sake of brevity):

Have some function  $J(\theta_0, \theta_1)$   $J(\theta_0, \theta_1, \theta_2, \dots, \theta_n)$

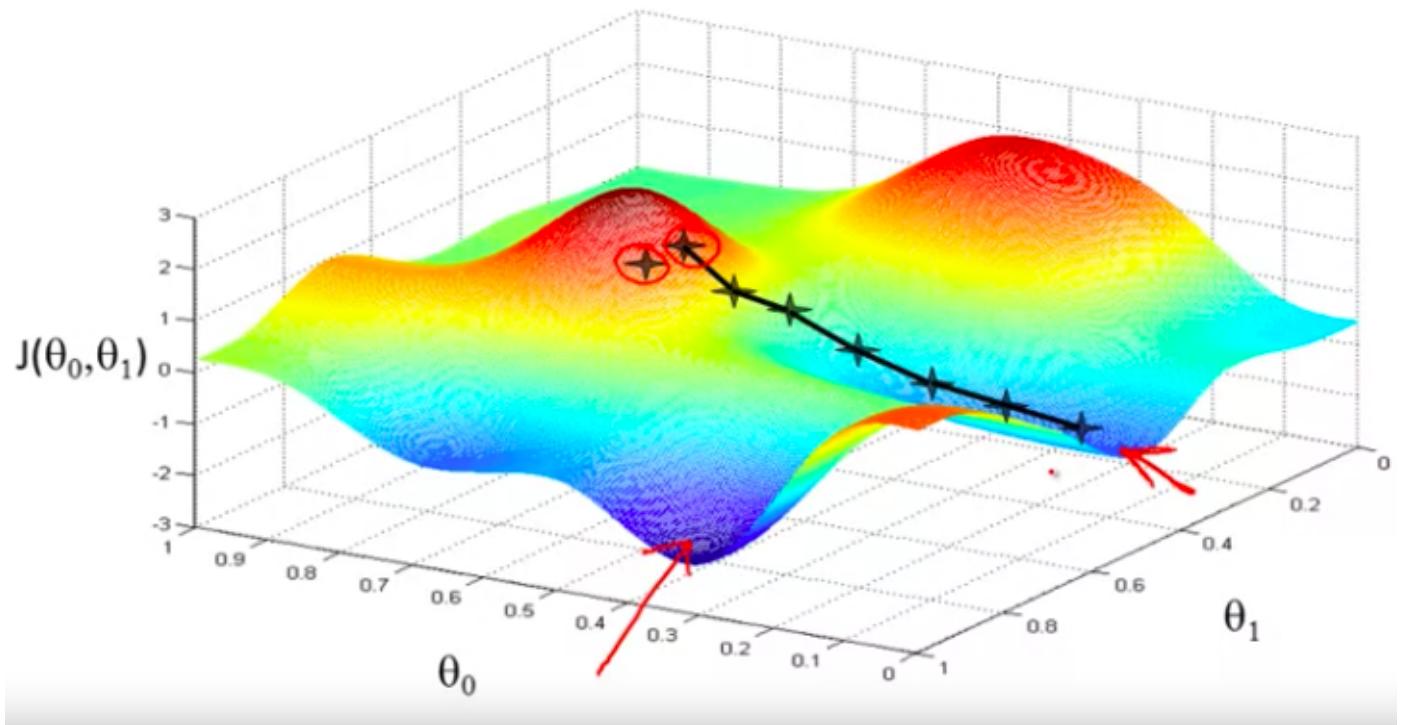
Want  $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$   $\min_{\theta_0, \dots, \theta_n} J(\theta_0, \dots, \theta_n)$

## Outline:

- Start with some  $\underline{\theta_0}, \underline{\theta_1}$  (say  $\theta_0 = 0, \theta_1 = 0$ )
- Keep changing  $\underline{\theta_0}, \underline{\theta_1}$  to reduce  $J(\theta_0, \theta_1)$   
until we hopefully end up at a minimum

The final minimum the algorithm converges to might be a local minimum.

The gradient descent algorithm has an interesting property: depending on how the parameters get initialised it might converge to different local optima (note the different positions of the two red arrows):



**Question:** Suppose  $\theta_0 = 1, \theta_1 = 2$  and we simultaneously update  $\theta_0$  and  $\theta_1$  using the rule

$$\theta_j := \theta_j + \sqrt{\theta_0 \theta_1} \quad (\text{for } j = 0 \text{ and } j = 1)$$

What are the resulting values of  $\theta_0$  and  $\theta_1$ ?

$$\begin{aligned}\theta_0 &= 1 + \sqrt{2} \\ \theta_1 &= 2 + \sqrt{2}\end{aligned}$$

### 3.2 Gradient Descent Intuition

In this video we explored the scenario where we used one parameter  $\theta_1$  and plotted its cost function to implement a gradient descent.

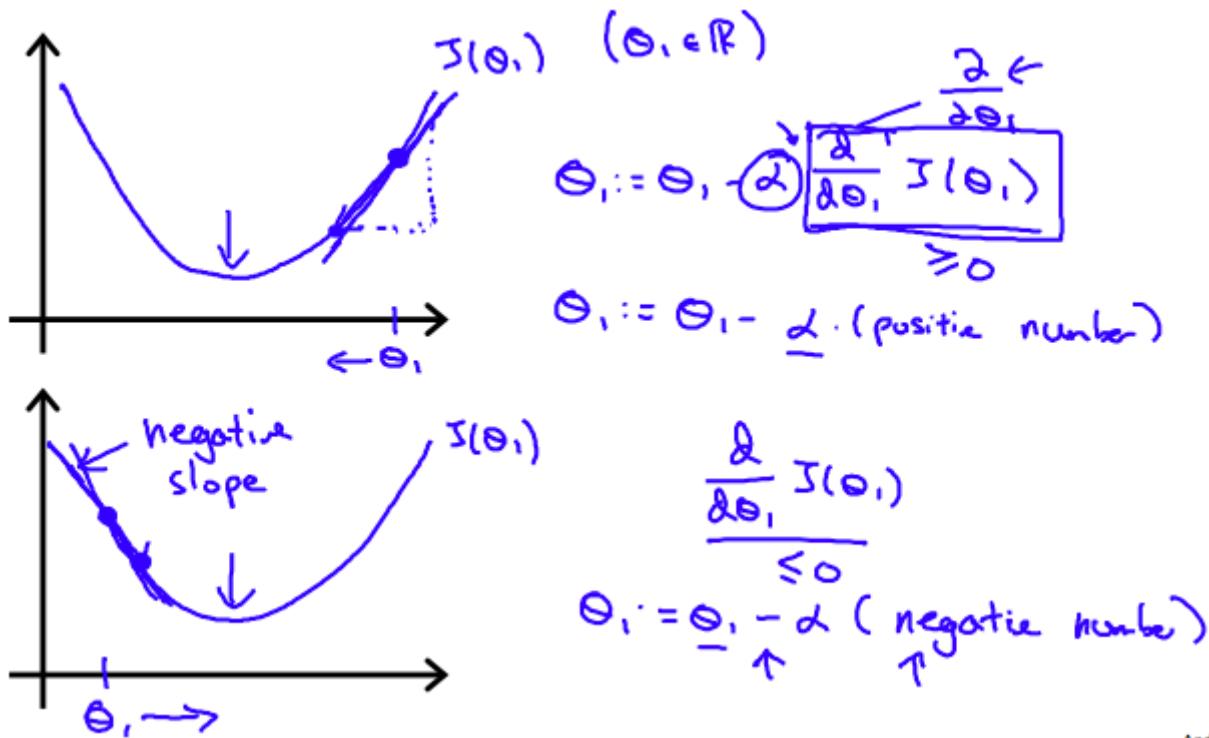
$$\min_{\theta_1} J(\theta_1) \quad \theta_1 \in \mathbb{R}.$$

Our formula for a single parameter was:

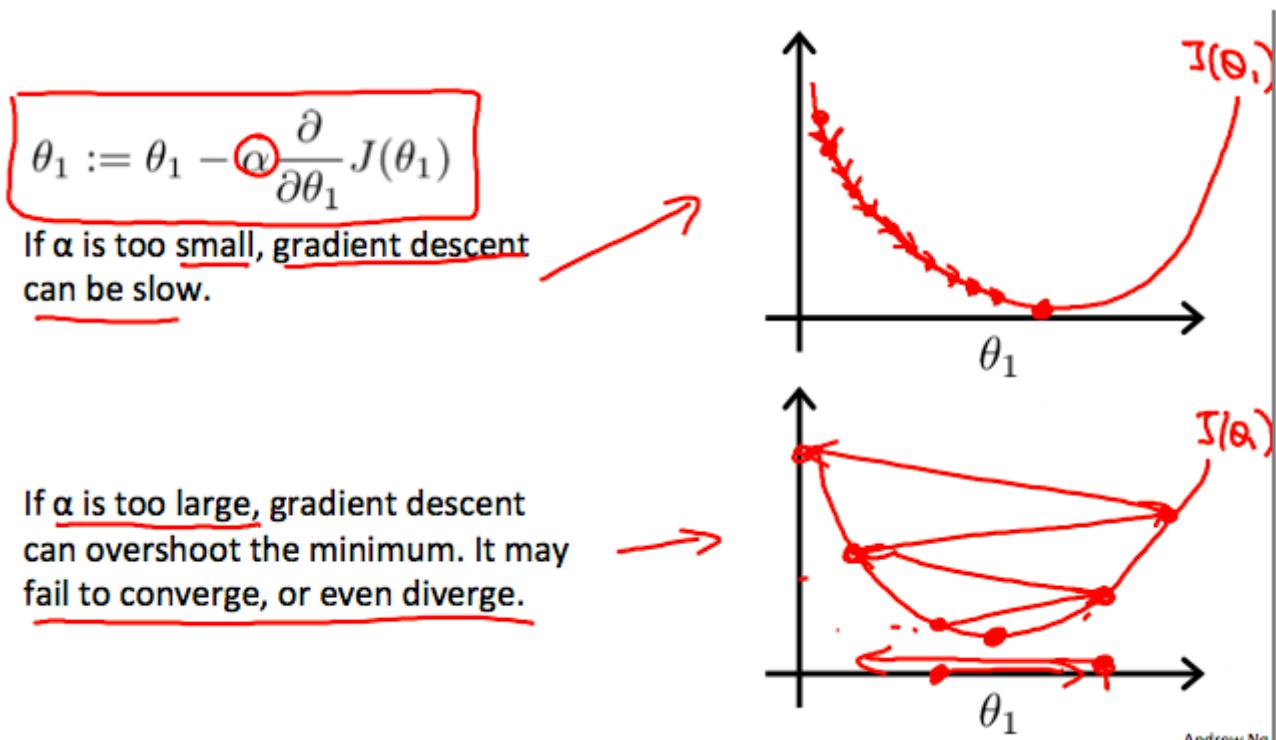
Repeat until convergence:

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

Regardless of the slope's sign for  $\frac{\partial}{\partial \theta_1} J(\theta_1)$ ,  $\theta_1$  eventually converges to its minimum value. The following graph shows that when the slope is negative, the value of  $\theta_1$  increases and when it is positive, the value of  $\theta_1$  decreases:



On a side note, we should adjust our parameter  $\alpha$  to ensure that the gradient descent algorithm converges in a reasonable time. Failure to converge or too much time to obtain the minimum value imply that our step size is wrong.



How does gradient descent converge with a fixed step size  $\alpha$ ?

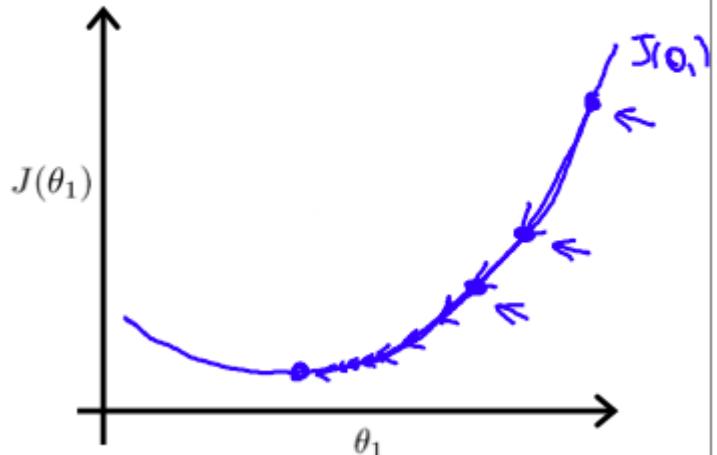
The intuition behind the convergence is that  $\frac{\partial}{\partial \theta_1} J(\theta_1)$  approaches 0 as we approach the bottom of our convex function. At the minimum, the derivative will always be 0 and thus we get:

$$\theta_1 := \theta_1 - \alpha * 0$$

Gradient descent can converge to a local minimum, even with the learning rate  $\alpha$  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  $\alpha$  over time.



As we approach the local minimum, by definition, the derivative will get smaller and smaller...

To recap:

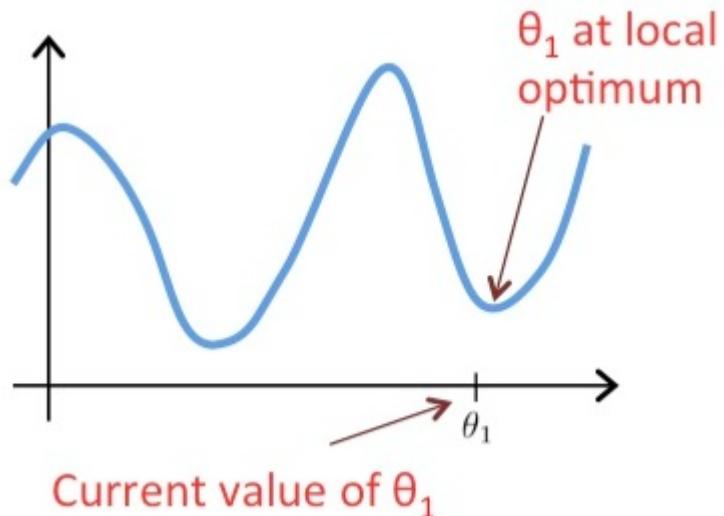
## Gradient descent algorithm

```

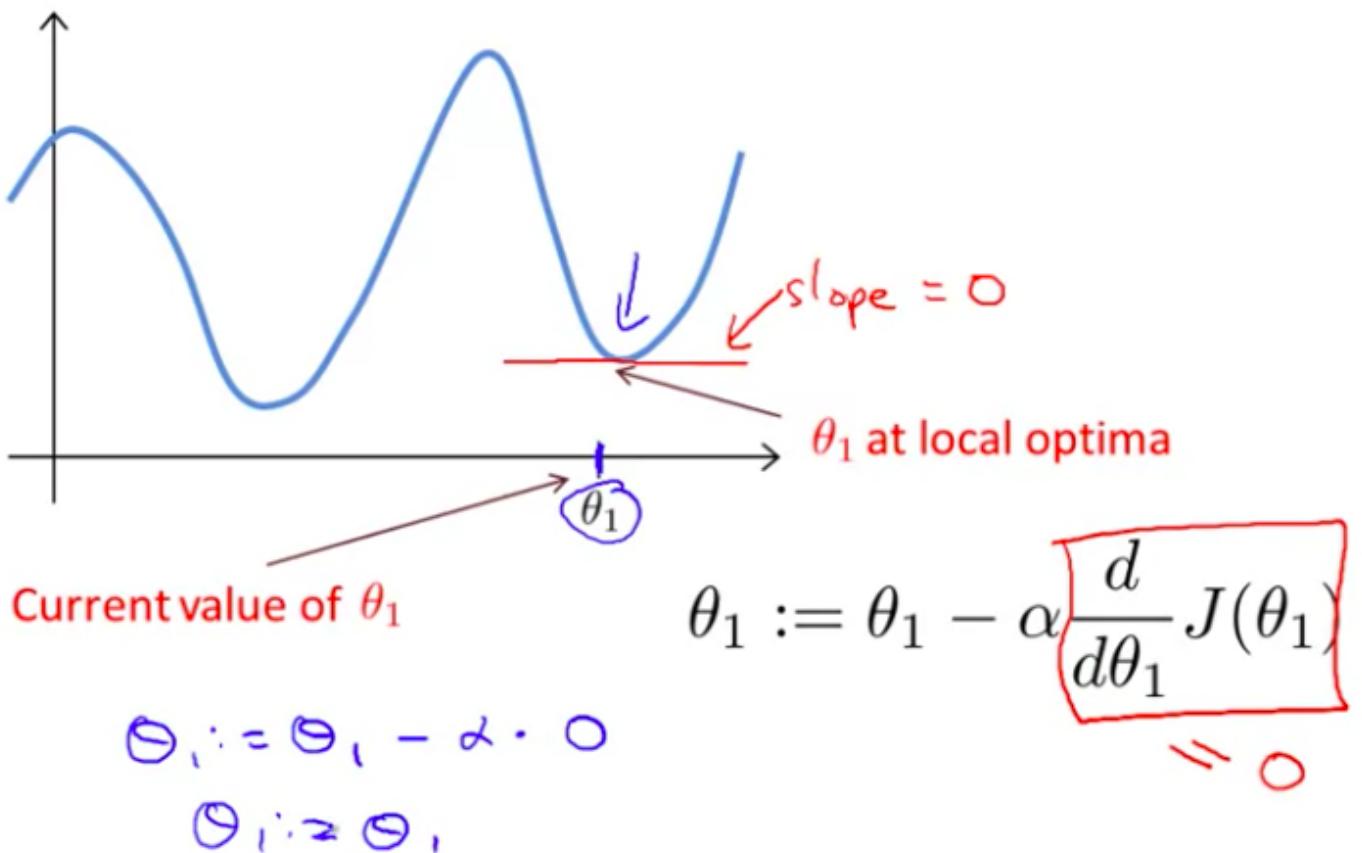
repeat until convergence {
    →  $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$  } (simultaneously update
        learning rate                                j = 0 and j = 1)
        ↑                                     ↑
        Derivative
  
```

Note: the **learning rate  $\alpha$**  is always a **positive number**.

**Question:** Suppose  $\theta_1$  is at a local optimum (i.e. minimum) of  $J(\theta_1)$ , such as shown in the figure below. What will one step of gradient descent  $\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$  do?



Answer: Leave  $\theta_1$  unchanged



### 3.3 Gradient Descent for Linear Regression

## 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

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

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

When specifically applied to the case of linear regression, a new form of the gradient descent equation can be derived. We can substitute our actual cost function and our actual hypothesis function and modify the equation to:

---

```
repeat until convergence: {
     $\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x_i) - y_i)$ 
     $\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m ((h_{\theta}(x_i) - y_i)x_i)$ 
}
```

where  $m$  is the size of the training set,  $\theta_0$  is a constant that ill be changing simultaneously with  $\theta_1$ , and  $x_i$  and  $y_i$  are values of the given training set (i.e. the data).

Note that we have separated out the two cases for  $\theta_j$  into 2 separate equations for  $\theta_0$  and  $\theta_1$ ; and that for  $\theta_1$  we are multiplying  $x_i$  at the end due to the derivative. The following is a derivation of  $\frac{\partial}{\partial \theta_j} J(\theta)$  for a single example:

$$\begin{aligned} \frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} (h_{\theta}(x) - y)^2 \\ &= 2 \cdot \frac{1}{2} (h_{\theta}(x) - y) \cdot \frac{\partial}{\partial \theta_j} (h_{\theta}(x) - y) \\ &= (h_{\theta}(x) - y) \cdot \frac{\partial}{\partial \theta_j} \left( \sum_{i=0}^n \theta_i x_i - y \right) \\ &= (h_{\theta}(x) - y) x_j \end{aligned}$$

How the partial derivatives of the gradient descent for linear regression are calculated (please refer to this link <https://math.stackexchange.com/questions/70728/partial-derivative-in-gradient-descent-for-two-variables> (<https://math.stackexchange.com/questions/70728/partial-derivative-in-gradient-descent-for-two-variables>)):

COST FUNCTION:

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

(m of items in learning set)

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

for  
linear  
regression

Goal of GRADIENT DESCENT:

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

Each step in the gradient descent can be described as:

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

For  $\theta_j = \theta_0$ :

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

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

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

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

Partial derivative of the expression within parentheses with respect to  $\theta_0$   
 (treat other terms as constants)

For  $\theta_j = \theta_1$ :

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

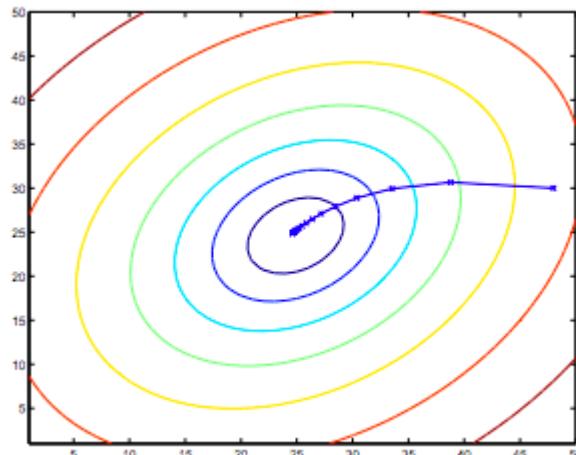
$$= \frac{1}{m} \sum_{i=1}^m [(\theta_0 + \theta_1 x^{(i)} - y^{(i)}) \cdot x^{(i)}] =$$

$$= \frac{1}{m} \sum_{i=1}^m [(h(x^{(i)}) - y^{(i)}) \cdot x^{(i)}]$$

The point of all this is that if we start with a guess for our hypothesis and then repeatedly apply these gradient descent equations, our hypothesis will become more and more accurate.

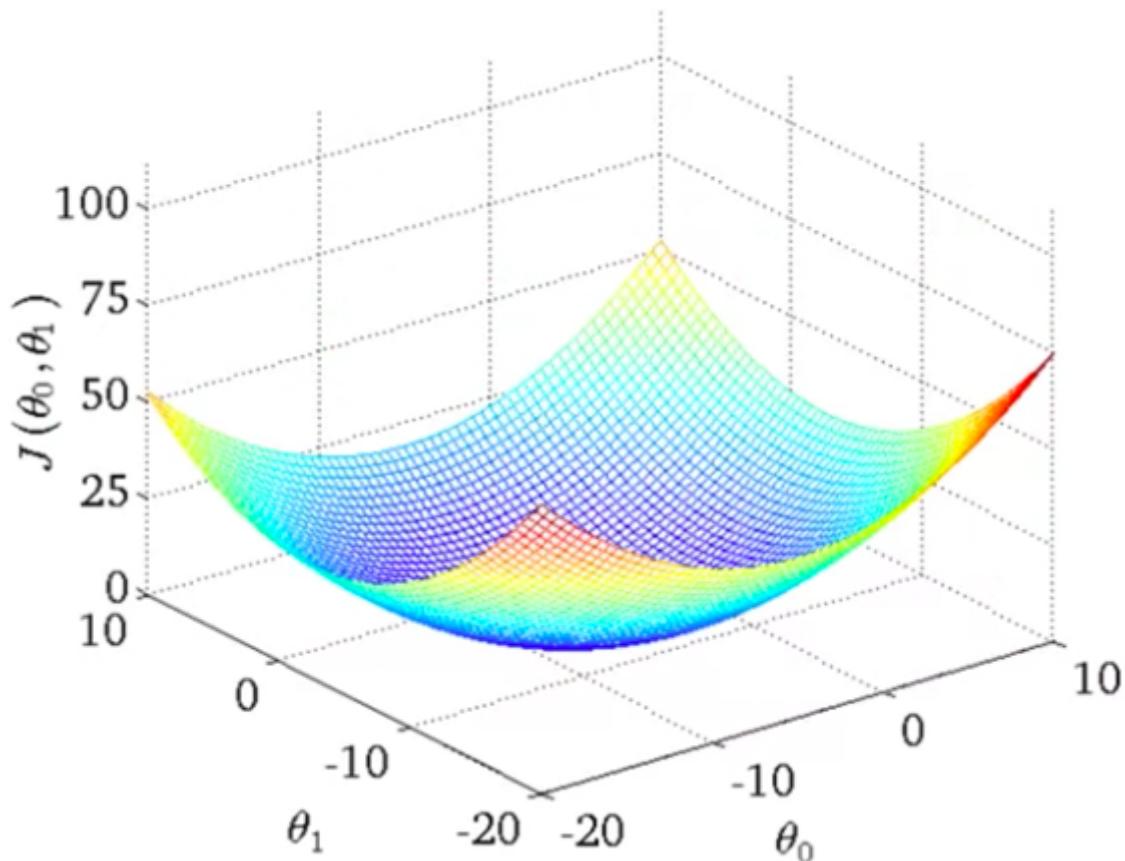
So, this is simply gradient descent on the original cost function  $J$ . This method looks at every example in the entire training set on every step, and is called **batch gradient descent**. At each step of the gradient descent algorithm we are computing sums of all training examples to update the parameters of the model. There are other versions of gradient descent that are not batch versions, i.e. they do not look at the entire training set but only at a small subset within it

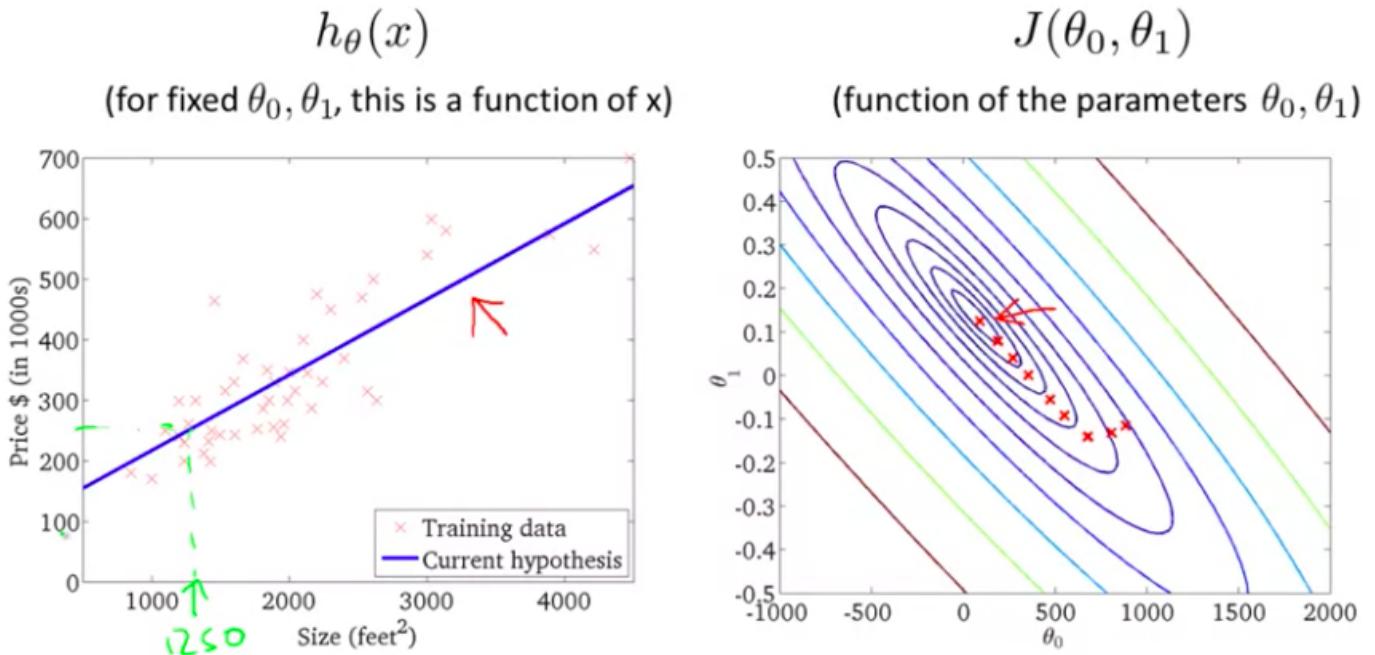
Note that, while gradient descent can be susceptible to local minima in general, the optimization problem we have posed here for linear regression has only one global, and no other local, optima; thus gradient descent always converges (assuming the learning rate  $\alpha$  is not too large) to the global minimum. Indeed,  $J$  is a **convex quadratic function**. Here is an example of gradient descent as it is run to minimize a quadratic function.



The ellipses shown above are the contours of a quadratic function. Also shown is the trajectory taken by gradient descent, which was initialized at (48,30). The x's in the figure (joined by straight lines) mark the successive values of  $\theta$  that gradient descent went through as it converged to its minimum.

The cost function for gradient descent of linear regression is always going to be a bowl-shaped (convex) function and there is **only one global optimum** (no local optima) to which the gradient descent will converge.





**Question:** Which of the following are true statements?

- To make gradient descent converge, we must slowly decrease  $\alpha$  over time: FALSE
- Gradient descent is guaranteed to find the global minimum for any function  $J(\theta_0, \theta_1)$ : FALSE
- Gradient descent can converge even if  $\alpha$  is kept fixed. (But  $\alpha$  cannot be too large, or else it may fail to converge.): TRUE
- For the specific choice of cost function  $J(\theta_0, \theta_1)$  used in linear regression, there are no local optima (other than the global optimum): TRUE

If you have taken an advanced course in **linear algebra**, you might know that there is a solution for **numerically solving** for the minimum of the cost function  $J$  without having to use an iterative algorithm like gradient descent. This method is called the **normal equation method**. However, it turns out that the gradient descent scales better to larger datasets than the normal equation method.

## Graded Quiz: Linear Regression with One Variable

### Question 1

Consider the problem of predicting how well a student does in her second year of college/university, given how well she did in her first year.

Specifically, let  $x$  be equal to the number of "A" grades (including A-, A and A+ grades) that a student receives in their first year of college (freshmen year). We would like to predict the value of  $y$ , which we define as the number of "A" grades they get in their second year (sophomore year).

Here each row is one training example. Recall that in linear regression, our hypothesis is  $h(x) = \theta_0 + \theta_1 x$  and we use  $m$  to denote the number of training examples.

<b>x</b>	<b>y</b>
3	2
1	2
0	1
4	3

For the training set given above (note that this training set may also be referenced in other questions in this quiz), what is the value of  $m$ ? In the box below, please enter your answer (which should be a number between 0 and 10).

Answer:  $m = 4$

### Question 2

Consider the following training set of  $m = 4$  training examples:

x	y
1	0.5
2	1
4	2
0	0

Consider the linear regression model  $h_{\theta}(x) = \theta_0 + \theta_1 x$ . What are the values of  $\theta_0, \theta_1$  that you would expect to obtain upon running gradient descent on this model? (Linear regression will be able to fit this data perfectly.)

Answer:

$$\theta_0 = 0$$

$$\theta_1 = 0.5$$

### Question 3

Suppose we set  $\theta_0 = -1, \theta_1 = 0.5$ . What is  $h_{\theta}(4)$ ?

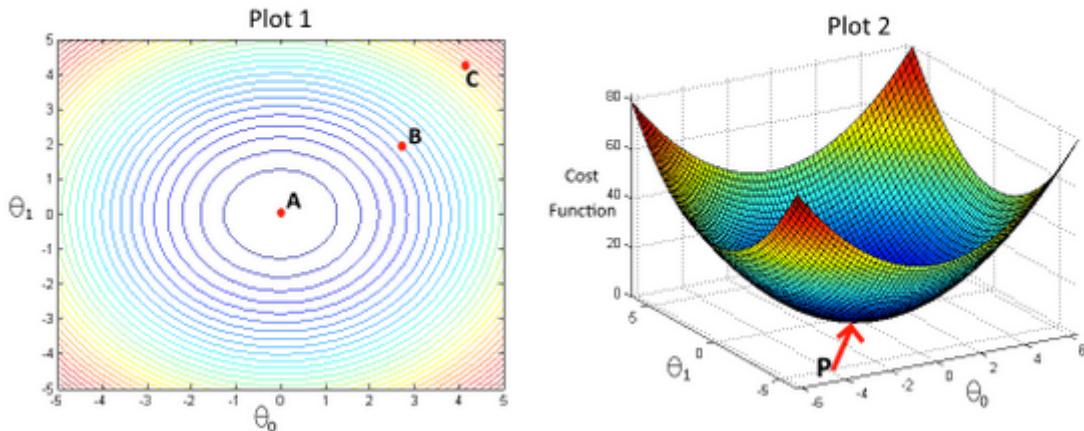
Answer:

$$h_{\theta}(4) = -1 + 0.5 * 4 = 1$$

### Question 4

In the given figure, the cost function  $J(\theta_0, \theta_1)$  has been plotted against  $\theta_0$  and  $\theta_1$ , as shown in 'Plot 2'. The contour plot for the same cost function is given in 'Plot 1'. Based on the figure, choose the correct options (check all that apply).

### Plots for Cost Function $J(\theta_0, \theta_1)$



Answers:

- Point P (the global minimum of Plot 2) corresponds to Point A in Plot 1.
- If we start from point B, gradient descent with a well-chosen learning rate will eventually help us reach at or near point A, as the value of the cost function  $J(\theta_0, \theta_1)$  is minimum at A.

### Question 5

Suppose that for some linear regression problem (say, predicting housing prices as in the lecture), we have some training set, and for our training set we managed to find some  $\theta_0, \theta_1$  such that  $J(\theta_0, \theta_1) = 0$ . Which of the statements below must then be true? (Check all that apply.)

Answers:

- Our training set can be fit perfectly by a straight line, i.e. all of our training examples lie perfectly on some straight line.