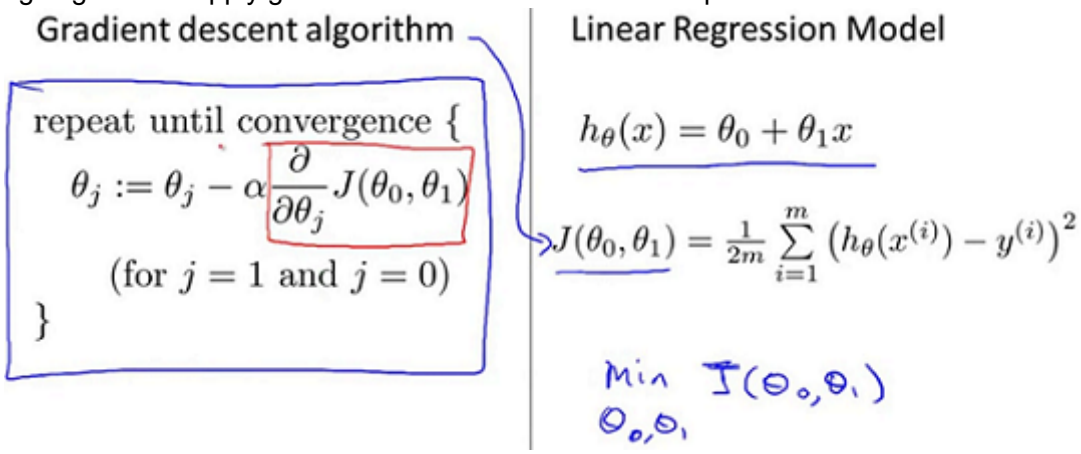
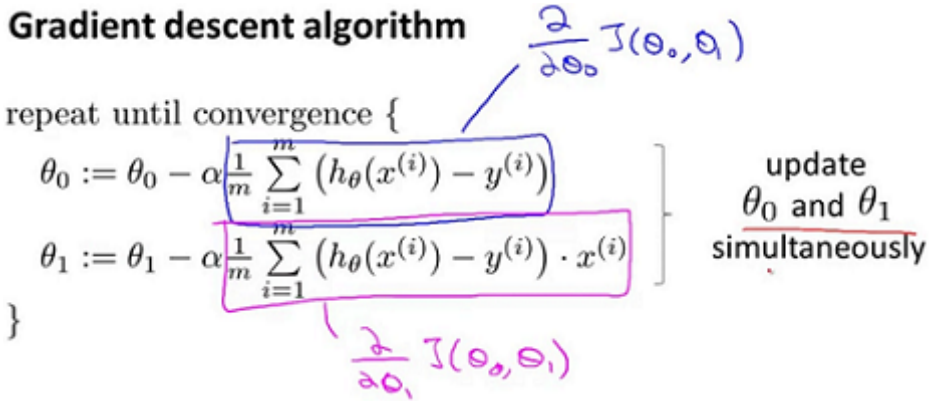
Gradient Descent For Linear Regression

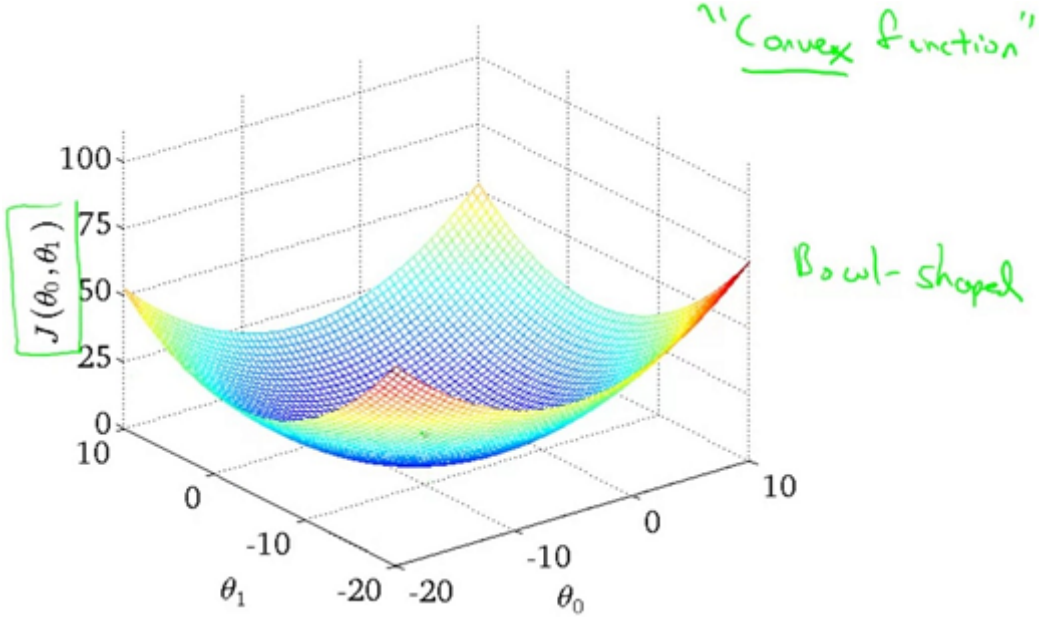
What we're going to do is apply gradient descent to minimize our squared error cost function.



Turns out we need to figure out what is this partial derivative for two cases for $J = 0$ and $J = 1$. So we want to figure out what is this partial derivative for both the θ_0 case and the θ_1 case (computing the partial derivative terms requires multivariate calculus), and the gradient descent for linear regression which is gonna repeat until convergence, θ_0 and θ_1 get updated as we know $\theta - \alpha$ times the derivative term.



In gradient descent depending on where we initialize it, we can end up at different local optima. But, it turns out that the cost function for linear regression is always going to be a bow shaped function. The techincal term for this is called a **convex function**. Informally a convex function means a bowl shaped function and this function doesn't have any **local optima** except for the one **global optimum**.



Finally, it turns out that the algorithm that we just went over is sometimes called batch gradient descent.

The term batch gradient descent refers to the fact that in every step of gradient descent, we're looking at all of the training examples.

"Batch" Gradient Descent

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

Video Question: Which of the following are true statements? Select all that apply.

- To make gradient descent converge, we must slowly decrease α over time.
- Gradient descent is guaranteed to find the global minimum for any function $J(\theta_0, \theta_1)$.

> Gradient descent can converge even if α is kept fixed. (But α cannot be too large, or else it may fail to converge.)

> 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).

Summary

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 :

$$\text{repeat until convergence: } \{$$

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

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

$$\}$$

where m is the size of the training set, θ_0 a constant that will be changing simultaneously with θ_1 and x_i, y_i are values of the given training set (data).

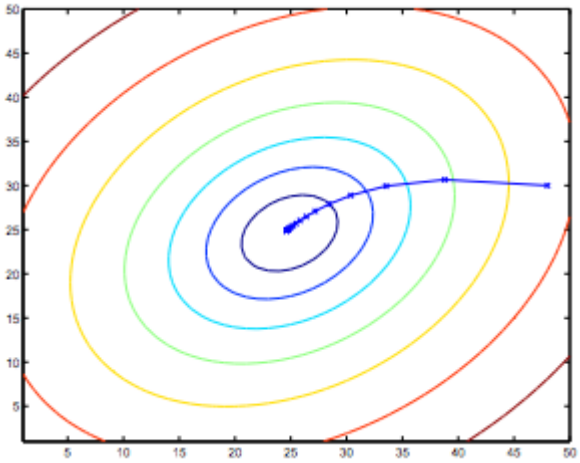
Note that we have separated out the two cases for θ_j into separate equations for θ_0 and θ_1: and that for θ_1 we are multiplying x_i at the end due to the derivative. The following is a derivation of $\frac{\partial}{\partial \theta_j} J(\theta)$ for a single example:

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

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.

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 α 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 θ that gradient descent went through as it converged to its minimum.