

# Professional English – VI

## Listening

Chixiao Chen

[cxchen@fudan.edu.cn](mailto:cxchen@fudan.edu.cn)

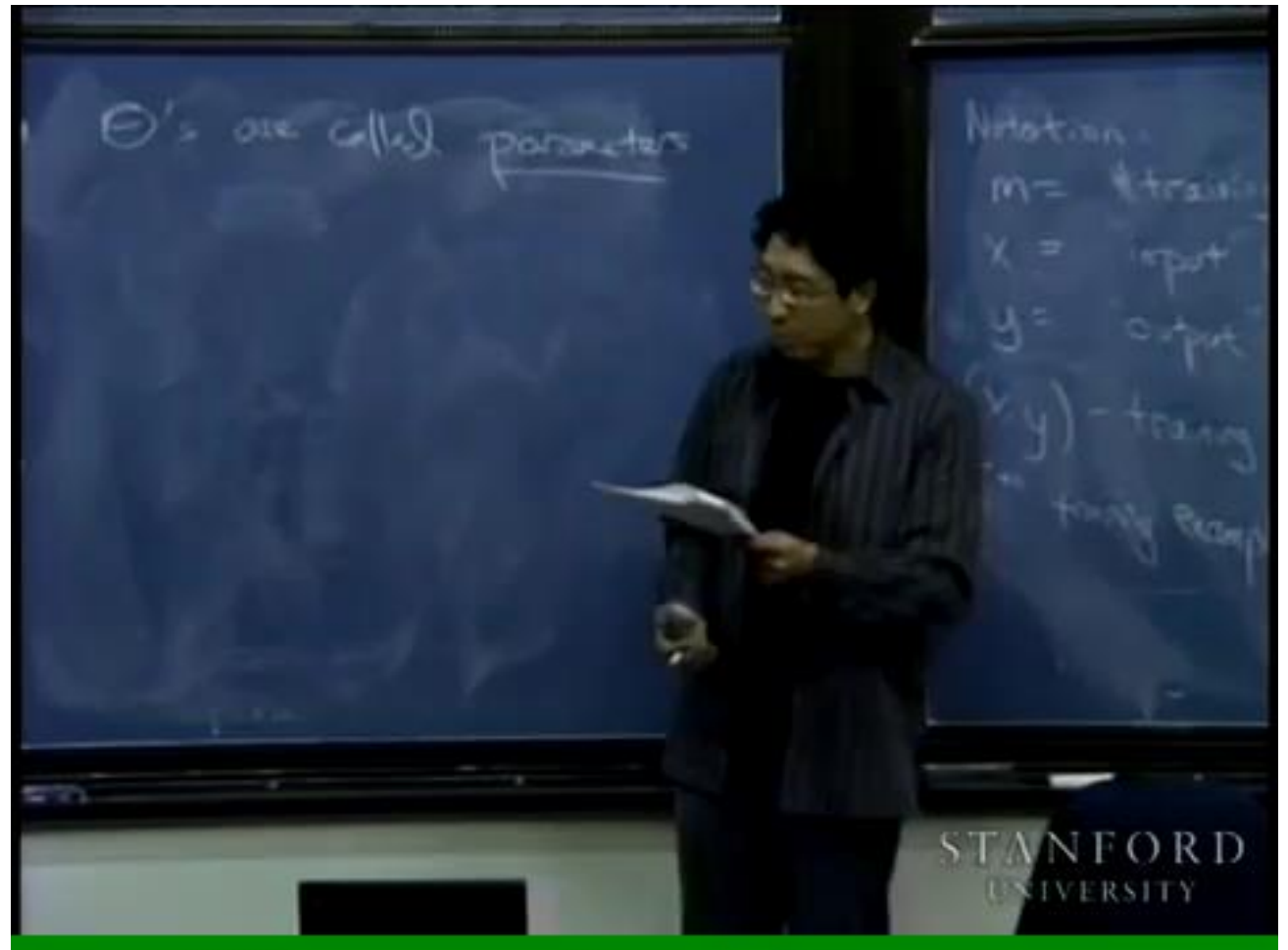
# Announcement

- We have a homework assignment last time. Due date: 12/31, 2019  
Please send to : [faet\\_english@126.com](mailto:faet_english@126.com)
- We have our final exam on 12/27, 2019.
- One more thing.

# Listening Comprehension

- Prof. Ng from Stanford University in the video talks about how to solve a mathematical problem analytically with certain parameters.

Hints: Please take notes!



# Please answer the following Question:

- What the objective of the mathematical problem ?
  - Hint: minimizing ...
- What analogy does the instructor use to describe the problem?
- Which kinds of parameters might cause different results?
- Which Greek alphabet is used in the derivation and what does it represent ?



How do we so that  
about all the houses. All right, so one  
reasonable thing to do seems to be, well, .



How do we choose the parameters  $\theta$  so that our hypothesis  $H$  will make accurate predictions about all the houses. All right, so one reasonable thing to do seems to be, well, we have a training set.



We have

, minus

. So the sum from  $i$  equals  
, of price

, minus

.



We have  $M$  training examples. So the sum from  $i$  equals one through  $M$  of my  $M$  training examples, of price predicted on the  $i$ -th house in my training set, minus the actual target variable, minus actual price on the  $i$ -th training example.





For those of you who have taken sort of \_\_\_\_\_, or  
maybe \_\_\_\_\_, some of you may have seen things  
like these before and see  
.



For those of you who have taken sort of linear algebra classes, or maybe basic statistics classes, some of you may have seen things like these before and see least square regression or least mean squares.



The first algorithm I'm going to talk about is  
where the basic idea is

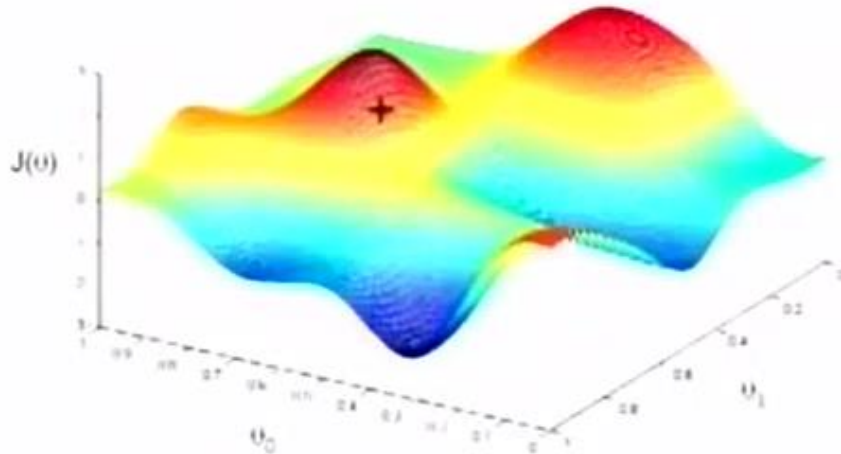
. Maybe  
vector of .

,  
to be the



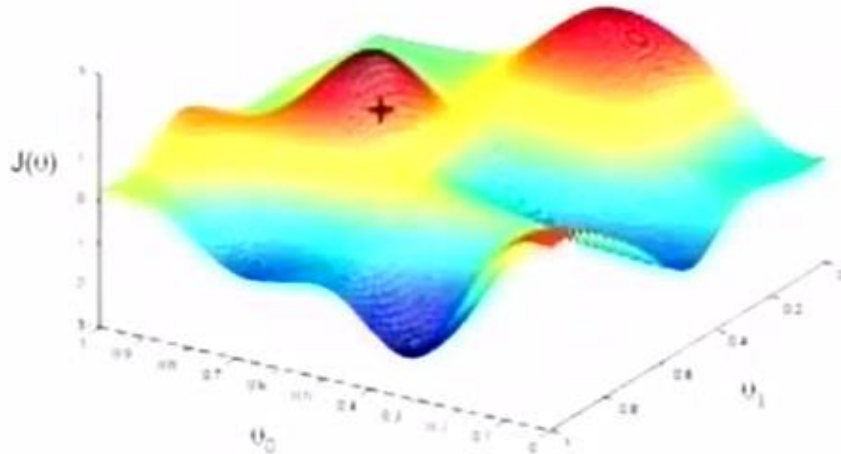
The first algorithm I'm going to talk about is a search algorithm, where the basic idea is we'll start with some value of my parameter vector  $\theta$ . Maybe initialize my parameter vector  $\theta$  to be the vector of all zeros.

## Gradient Descent



You see on the display a plot and the axes,  $\theta_0$  and  $\theta_1$ . That's usually – the parameters, which is represented by  $\theta$ . So  $J(\theta)$  is the cost function, and the axes of this function, or  $\theta_0$  and  $\theta_1$ , are the parameters, written down here below.

## Gradient Descent



You see on the display a plot and the axes, the horizontal axes are  $\theta_0$  and  $\theta_1$ . That's usually – minimize  $J$  of  $\theta$ , which is represented by the height of this plot. So the surface represents the function  $J$  of  $\theta$  and the axes of this function, or the inputs of this function are the parameters  $\theta_0$  and  $\theta_1$ , written down here below.



Here's . I'm going to  
. It could be or some  
. Let's say we start from that point denoted by the star, by the  
, and now I want you to imagine that  
. Imagine or  
something, and this is of, like, a hill in some park.



Here's the gradient descent algorithm. I'm going to choose some initial point. It could be vector of all zeros or some randomly chosen point. Let's say we start from that point denoted by the star, by the cross, and now I want you to imagine that this display actually shows a 3D landscape. Imagine you're all in a hilly park or something, and this is the 3D shape of, like, a hill in some park.





Imagine you can \_\_\_\_\_, right, and  
\_\_\_\_\_ and ask, \_\_\_\_\_, what would  
\_\_\_\_\_? Okay, just imagine that  
\_\_\_\_\_ and you're standing there, and would look around  
ask, "If I take a small step, \_\_\_\_\_,  
that would \_\_\_\_\_?"



Imagine you can stand on that hill, right, and look all 360 degrees around you and ask, if I were to take a small step, what would allow me to go downhill the most? Okay, just imagine that this is physically a hill and you're standing there, and would look around ask, "If I take a small step, what is the direction of steepest descent, that would take me downhill as quickly as possible?"



You can , okay, and you sort of keep going until  
 , J of theta.  
is that where you end up – in this case,  
we ended up at this point .



You can take another step, okay, and you sort of keep going until you end up at a local minimum of this function,  $J$  of  $\theta$ . One property of gradient descent is that where you end up – in this case, we ended up at this point on the lower left hand corner of this plot.



It                      that with                      , you  
can actually end up at                      . Okay,  
so this is                      , and we'll  
                    . So                      gradient descent can sometimes depend  
on                      , theta zero and theta one.



It turns out that with a slightly different initial starting point, you can actually end up at a completely different local optimum. Okay, so this is a property of gradient descent, and we'll come back to it in a second. So be aware that gradient descent can sometimes depend on where you initialize your parameters,  $\theta_0$  and  $\theta_1$ .



Start with some  $\Theta$ . (Say  $\Theta = \vec{0}$ )  
Keep changing  $\Theta$  to reduce  $J(\Theta)$ .

---

Gradient descent:

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



We're going to update the parameters theta as theta I minus the partial derivative with respect to theta I, J of Theta. Okay, so this is how we're going to update the I parameter, theta I, how we're going to update Theta I on each iteration of gradient descent.



Start with some  $\Theta$ . (Say  $\Theta = \vec{0}$ )  
 Keep changing  $\Theta$  to reduce  $J(\Theta)$ .

---

Gradient descent:  
 $\Theta_i := \Theta_i - \alpha \frac{\partial}{\partial \Theta_i} J(\Theta)$



Just  $\Theta_i$ , I use  $\Theta$  to denote  
 $\Theta_i$  equal to  $\Theta$ .  
 . If I write  $\Theta_i$ , then what I'm saying is,  
 this is part of  $\Theta$ , or this is part of an algorithm  
 where we take the value of  $B$ ,  $\Theta_i$ , and  
 use that to  $\Theta$ .

Start with some  $\Theta$ . (Say  $\Theta = \vec{0}$ )  
Keep changing  $\Theta$  to reduce  $J(\Theta)$ .

---

Gradient descent:  
 $\Theta_i := \Theta_i - \alpha \frac{\partial}{\partial \Theta_i} J(\Theta)$



Just a point of notation, I use this colon equals notation to denote setting a variable on the left hand side equal to the variable on the right hand side. If I write A colon equals B, then what I'm saying is, this is part of a computer program, or this is part of an algorithm where we take the value of B, the value on the right hand side, and use that to overwrite

.

$$\frac{\partial}{\partial \theta_i} J(\theta) = \frac{\partial}{\partial \theta_i} \frac{1}{2} (h_{\theta}(x) - y)^2$$



If we have only one-half of , of only one training example this is what J of theta is going to be.

then J of theta is going to be . So if you have , then

$$\frac{\partial}{\partial \theta_i} J(\theta) = \frac{\partial}{\partial \theta_i} \frac{1}{2} (h_{\theta}(x) - y)^2$$



If we have only one training example then  $J$  of  $\theta$  is going to be one-half of square  $\theta$ , of  $X$  minus  $Y$ , square. So if you have only one training example comprising one pair,  $X$ ,  $Y$ , then this is what  $J$  of  $\theta$  is going to be.

$$\begin{aligned}
 \frac{\partial}{\partial \theta_i} J(\theta) &= \frac{\partial}{\partial \theta_i} \frac{1}{2} (h_{\theta}(x) - y)^2 \\
 &= 2 \cdot \frac{1}{2} (h_{\theta}(x) - y) \cdot \frac{\partial}{\partial \theta_i} (h_{\theta}(x) - y) \\
 &= (h_{\theta}(x) - y) \cdot \frac{\partial}{\partial \theta_i} (\theta_0 x_0 + \dots + \theta_n x_n - y)
 \end{aligned}$$

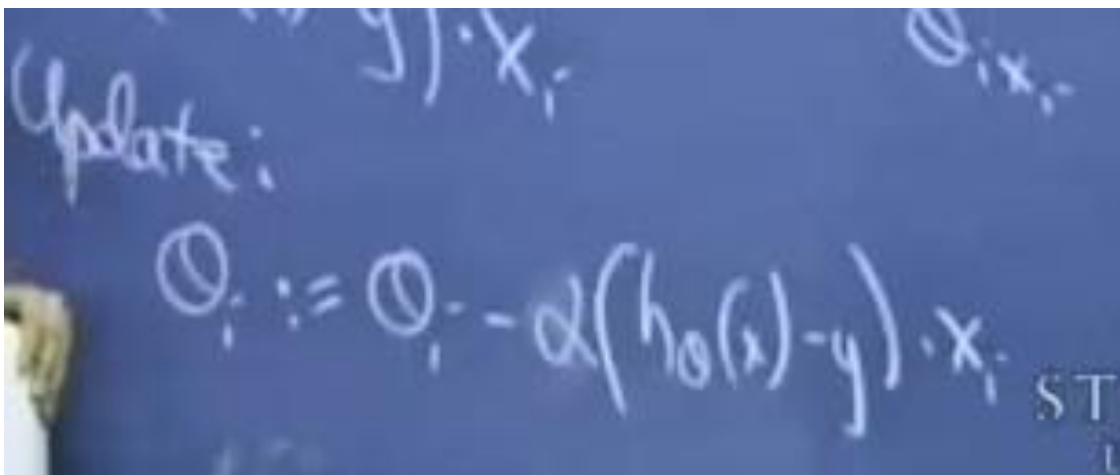


So taking derivatives, you have  $\frac{\partial}{\partial \theta_i} J(\theta)$ . So the derivative of  $J(\theta)$  with respect to  $\theta_i$  is  $(h_{\theta}(x) - y) \cdot \frac{\partial}{\partial \theta_i} (h_{\theta}(x) - y)$ . So you have two times one-half times theta of X minus Y, and then by the chain rule, we also have  $\frac{\partial}{\partial \theta_i} (\theta_0 x_0 + \dots + \theta_n x_n - y)$ . Right, the two and the one-half cancel out. So this leaves [blah blah blah ...]

$$\begin{aligned}
 \frac{\partial}{\partial \theta_i} J(\theta) &= \frac{\partial}{\partial \theta_i} \frac{1}{2} (h_{\theta}(x) - y)^2 \\
 &= 2 \cdot \frac{1}{2} (h_{\theta}(x) - y) \cdot \frac{\partial}{\partial \theta_i} (h_{\theta}(x) - y) \\
 &= (h_{\theta}(x) - y) \cdot \frac{\partial}{\partial \theta_i} (\theta_0 x_0 + \dots + \theta_n x_n - y)
 \end{aligned}$$



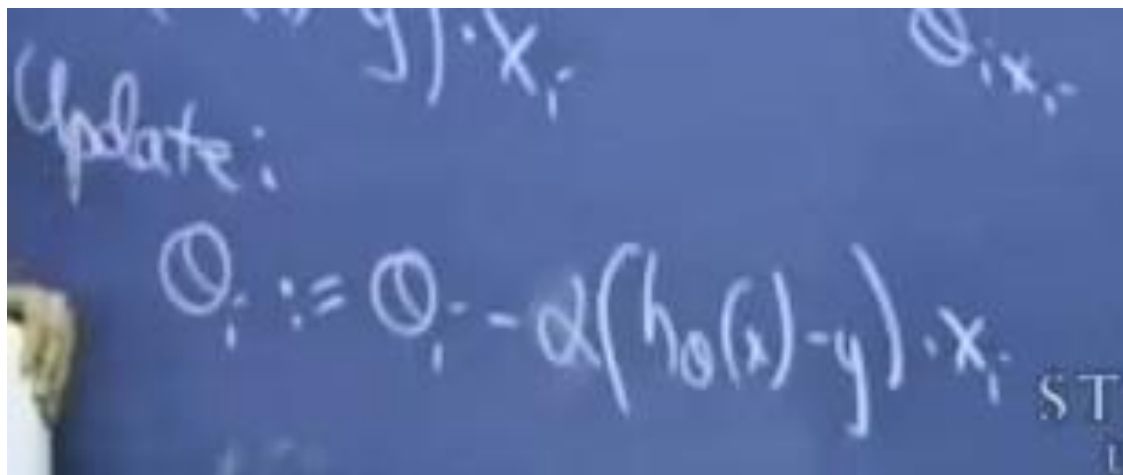
So taking derivatives, you have one-half something squared. So the two comes down. So you have two times one-half times theta of X minus Y, and then by the chain rule derivatives, we also must apply this by the derivative of what's inside the square. Right, the two and the one-half cancel. So this leaves [blah blah blah ...]



Update:

$$\theta_i := \theta_i - \alpha(h_0(x) - y) \cdot x_i$$

The Greek alphabet alpha here is called  $\alpha$ , and this parameter alpha is the learning rate. So you're standing on the hill. You decided to take a step in the direction of the negative gradient, and so this parameter alpha is the learning rate – how large a step you take in each iteration.



Update:

$$\theta_i := \theta_i - \alpha(h_0(x) - y) \cdot x_i$$

STU



The Greek alphabet alpha here is a parameter of the algorithm called the learning rate, and this parameter alpha controls how large a step you take. So you're standing on the hill. You decided what direction to take a step in, and so this parameter alpha controls how aggressive – how large a step you take in this direction of steepest descent.