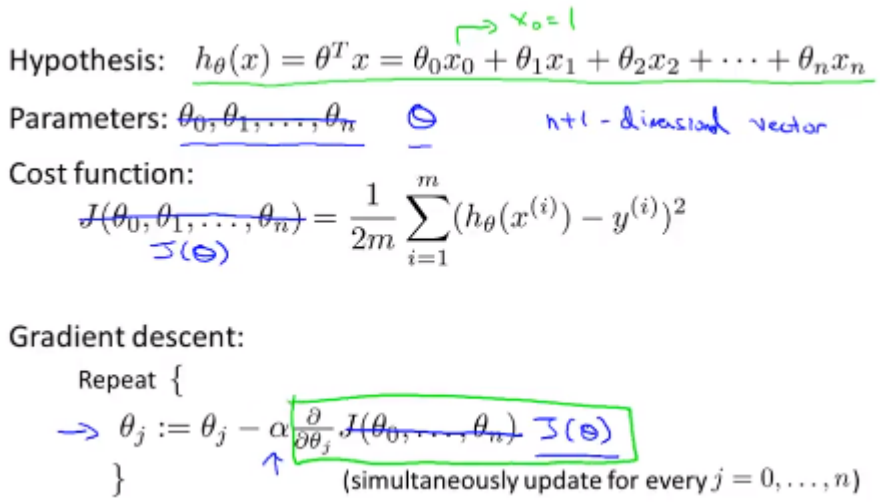
梯度下降法的使用

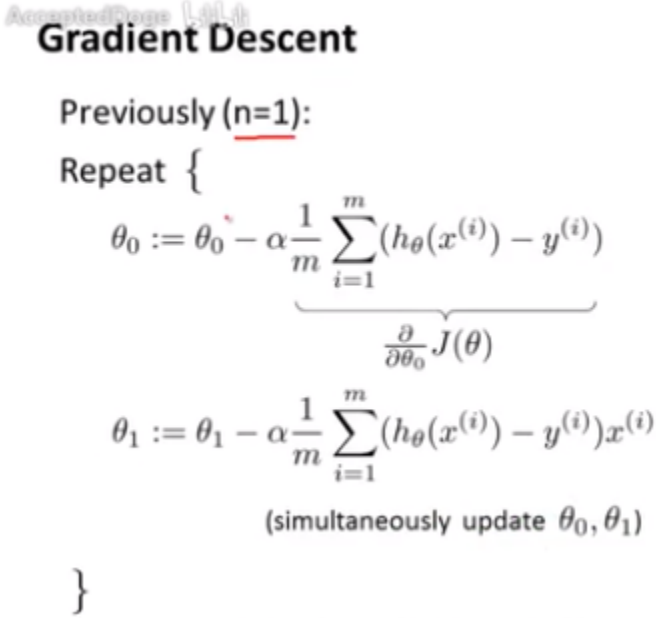
问题：如何使用梯度下降来解决多特征的线性回归问题？



如上图所示，假设有多元线性回归，并约定 x0=1 该模型的参数是从 θ0 到 θn，注意不要认为这是 n+1 个的单独参数，你可以把这 n+1 个 θ 想象成其本身就是一个 n+1 维向量的向量，代价函数是从 θ0 到 θn 的函数 j 并给出了误差项平方的和，但同样不要把函数 J 相成是带有一个 n+1 维向量的函数，这就是梯度下降法。

梯度下降基本概念：不停用 θj 减去学习率 α 与对应导数的乘积。

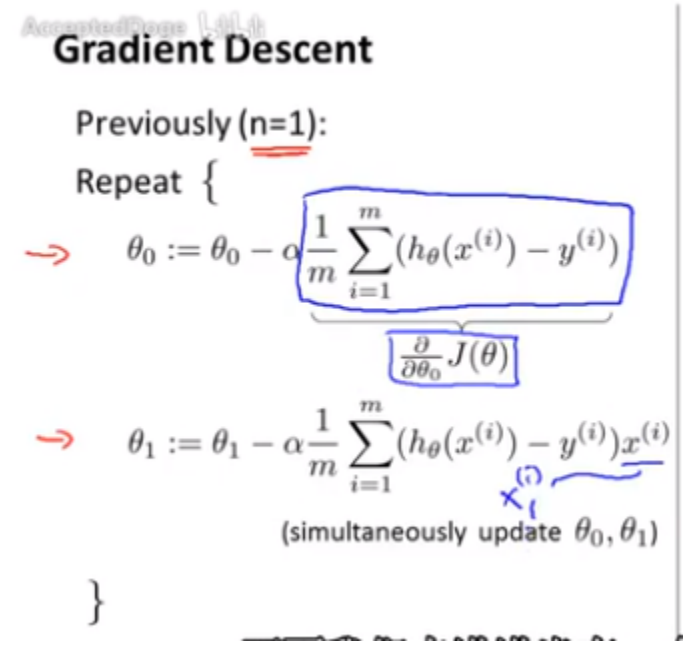
当 n=1 时，梯度下降的情况



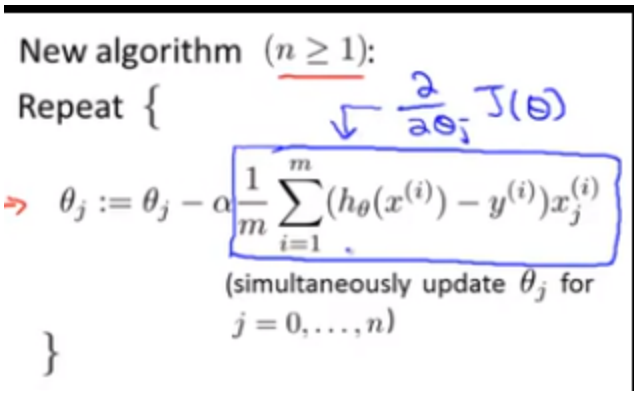
我们有两条针对参数 θ0 和 θ1 不同的更新规则，

$$\alpha \underbrace{\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})}_{\frac{\partial}{\partial\theta_0}J(\theta)}$$

这个相当于 θ0 的偏导数，同样对参数 θ1，我们有另一个更新规则，当我们只有一个特征的时候我们称该特征维 x(i)，但现在我们在新符号里，我们会上标(i)下标 1 来表示我们的特征，以上就说当我们仅有一个特征的时候的算法。

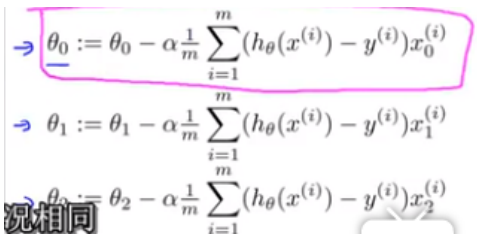


当有一个以上特征时候的算法，现有数目远大于 1 的很多特征，梯度下降更新规则如下

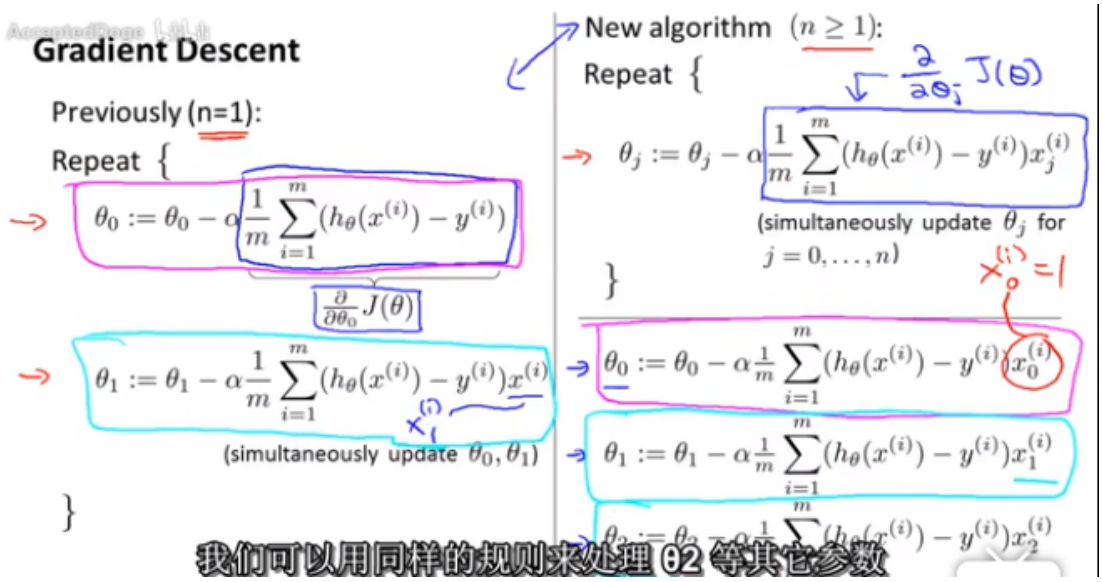


得到多元线性回归的梯度下降算法。

有两个或者两个以上数的特征，同时我们有对 θ1，θ2，θ3 的三条更新规则，当然可能还有其他参数，



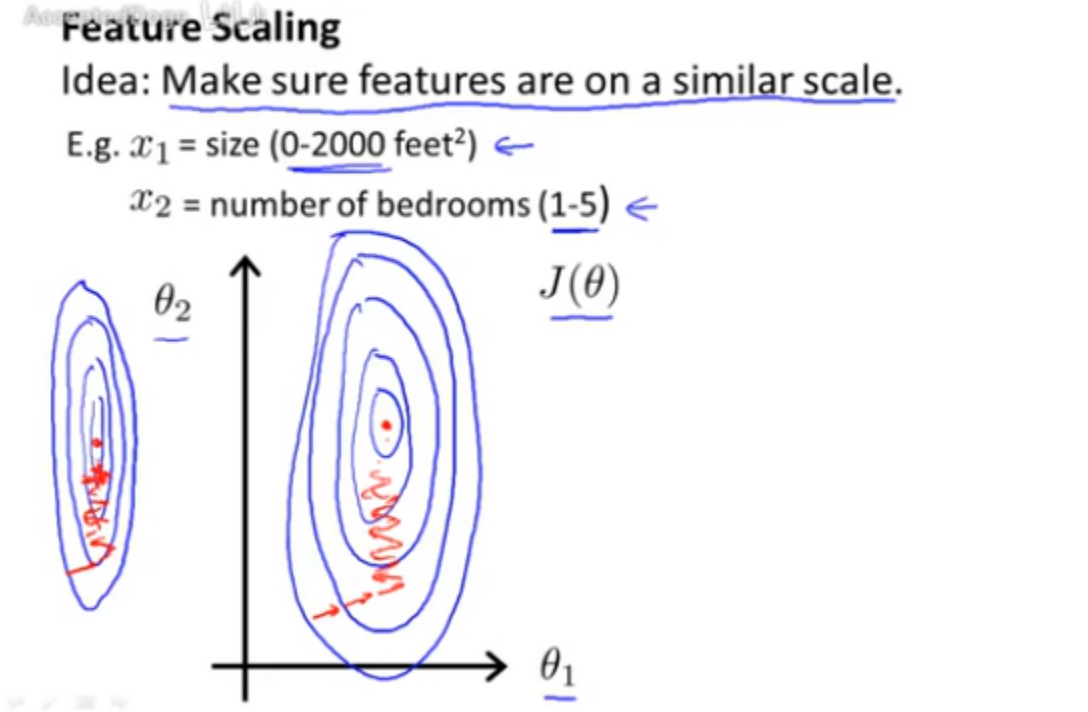
Θ0 跟 n=1 的情况是一样的，之所以时等价是因为在我们标记约定里有 x(i)0=1.也就是用红圈圈起来的两项（$X^{(i)}_0 = X_0^{i)}$），Θ1 的更新规则跟之前对参数 Θ 的更新是等价的，同样可以用这样的规则来处理 Θ2 等其他参数。



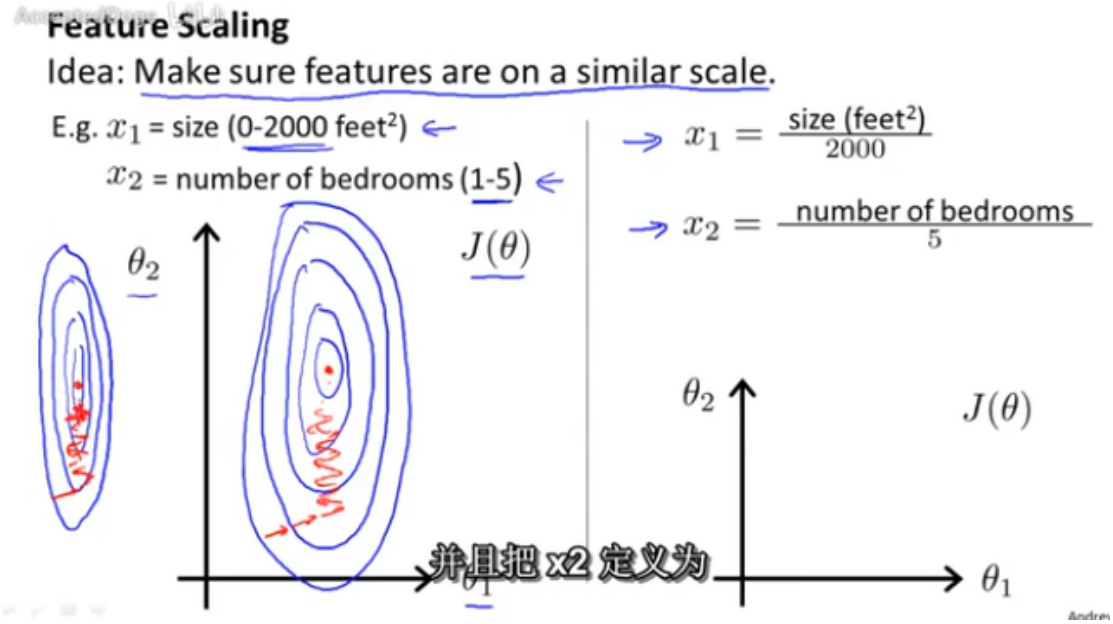
梯度下降运算中的实用技巧

1、 特征缩放

假设你有一个具有两个特征的问题，其中 x1 是房屋的大小，他的取值在 0~2000,x2 是卧室的数量，可能这个值的取值在 1-5，画出 J(Θ)图：



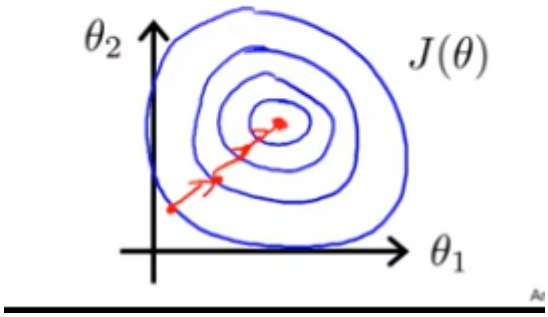
X1 很大，x2 有范围（假设 1-5），通过梯度下降来获得全局最小值，花费的时间可能会很长。一种有效的方法：特征收敛。



它把 x 定义为房子的面积大小，除以 2000 把 x2 定义为卧室的数量除以 5，表示代价函数轮廓偏移没那么严重。

$$x_1 = \frac{\text{size (feet}^2)}{2000}$$

$$x_2 = \frac{\text{number of bedrooms}}{5}$$



能够花更短的时间找到最小值。通过特征缩放，消耗掉这些值的范围，最终得到得值 x1 和 x2 都在 0 和 1 之间，这样的到的梯度下降算法会更快的收敛。

一般的我们通过特征值约束到-1 到+1 的范围内。你的特征值 x0 总是等于 1，因此在这个范围之内，但对其他特征可能需要除以不同的数让他们处于同一范围内只要接近-1--+1 之间都可以（比如 -2 到 5），但是如果有一个值 x3 范围在-100 到+100 之间，这个范围很大不同了，不是一个好特征，同样的如果 x4 范围在-0.0001 到 0.0001 之间比-1 到+1 小得多的范围，这个特征也不太好。一般范围-3 到+3，-1/3 到+1/3 之间是可以的.

2、 均值归一化

Mean normalization

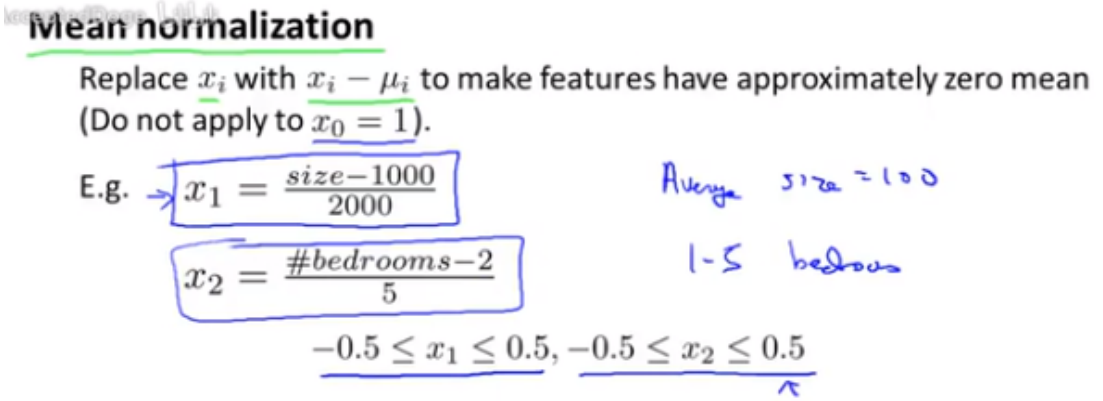
Replace x_i with $x_i - \mu_i$ to make features have approximately zero mean (Do not apply to $x_0 = 1$).

E.g. $x_1 = \frac{size - 1000}{2000}$

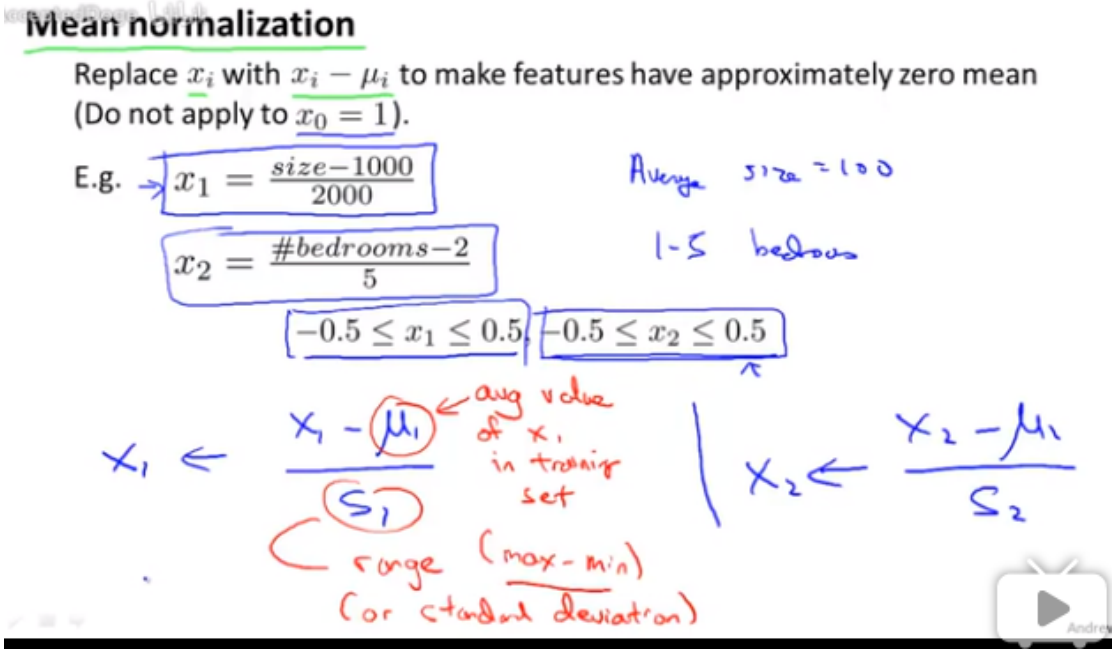
$x_2 = \frac{\#bedrooms - 2}{5}$

$-0.5 \leq x_1 \leq 0.5, -0.5 \leq x_2 \leq 0.5$

用 xi-ui 来替换，让特征值具有为 0 的平均值，假设房子大小取值介于 0-2000,并且假设房子面积的平均值是等于 1000，用上图公司将 x1 的值减去平均值 u1 再除以 2000,类似的如果你的房子有 5 间卧室，平均一套房子有两间卧室，可以使用这个公式来归一化你的第二个特征 x2，在这两种情况下算出新的特征值 x1 和 x2,它们的范围可以在-0.5 到 0.5 之间，实际上 x2 的值大于 0.5,但是很接近



更一般的可以使用这个公式（x1-u1）/S1.来替换原来的特征值 x1，u1 是在训练集中 x1 的平均值，而 S1 是特征值的范围（最大值减去最小值==变量的标准差），对于特征值 S2 同样可以用这个 特征值减去平均值再除以范围来替换原特征，范围的意思依然是最大值减去最小值，这类公式把你的特征变成这样的大概的范围。



梯度下降算法实用技巧—学习率 α

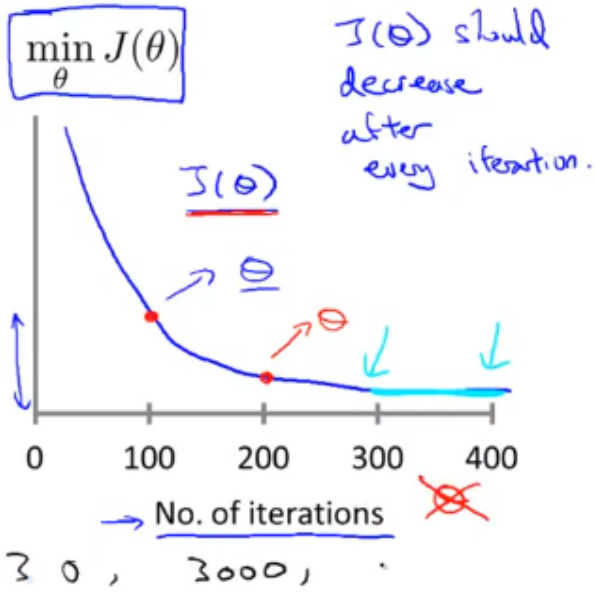
1、 如何确定梯度下降是否正常工作？

2、 如何选择学习率？

梯度下降做的工作就说为你找到一个 θ 值，并希望它能够最小化代价函数 J(θ),

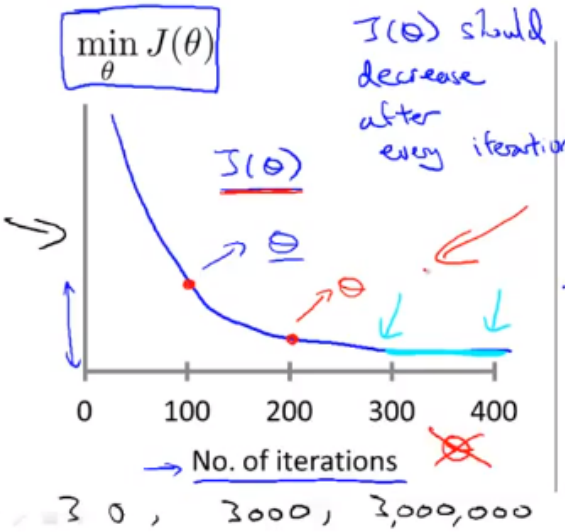
X 轴代表梯度下降算法的迭代步数，当运行完 100 步的梯度下降迭代之后无论我得到的 θ 值多大，在 100 步迭代之后会得到一个 θ 值，计算出代价函数 J(θ)的值，那个点的垂直高度也就是梯度下降算法迭代 200 次之后得到的 θ，计算出 J(θ)，这条曲线显示的是梯度下降算法迭代过程中代价函数 J(θ)的值，如果梯度下降正常工作，每一步迭代之后看起来 J(θ)会下降多少，当到达 400 步是曲线看起来很平坦了，梯度下降算法已经收敛了，对于每一个特定的问题梯度下降需要迭代的次数相差很大。

Making sure gradient descent is working correctly.



我们需要这个曲线画出大家函数随迭代步数增加的变化曲线，也可以进行自动收敛测试

Making sure gradient descent is working correctly.

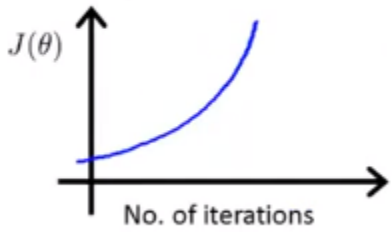


- Example automatic convergence test:
- Declare convergence if $J(\theta)$ decreases by less than 10^{-3} in one iteration.

ε

Andrew Ng

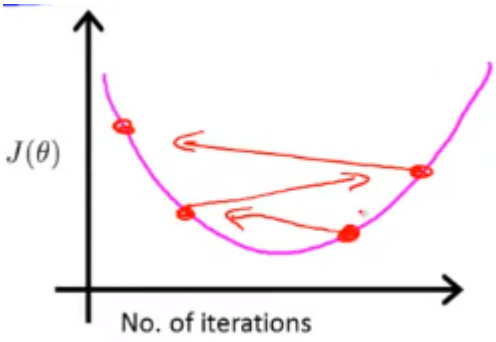
Making sure gradient descent is working correctly.



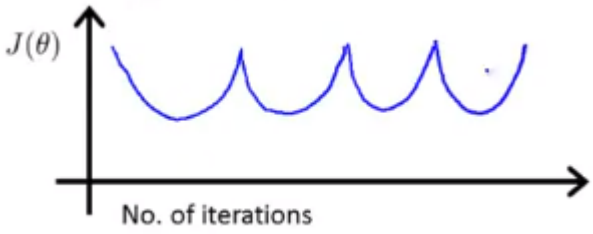
Gradient descent not working.

Use smaller α.

这个曲线图很明显没有收敛，梯度下降没有正常工作，这样的曲线图需要使用较小的学习率 α，如果 J(θ)在上升，常见的问题是你在最下化这样的一个函数，冲过最小值，达到另一个点，如果学习率太大可能会再次冲过最小值达到另一个点



梯度下降算法将会不断的冲过最小值，最后会得到越来越糟糕的结果，得到一个越来越大的代价函数 J(θ),遇到这一样的问题通常是使用较小的 α 值。
可能你也会看到这种情况



解决办法同样是选择较小的 α 值
总结：尝试一系列的值（0.0001,0.001），每隔 10 被取一个值，绘制 J(θ)图