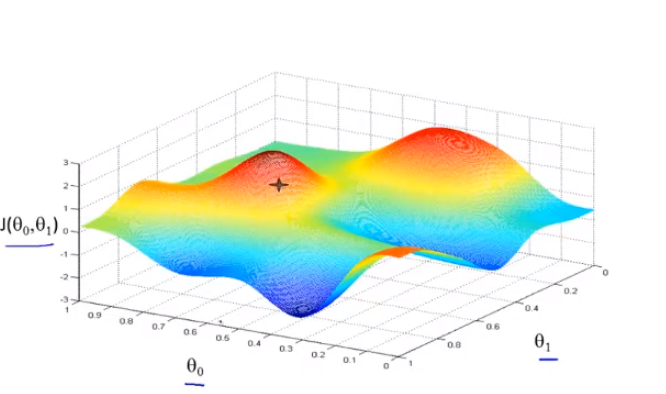
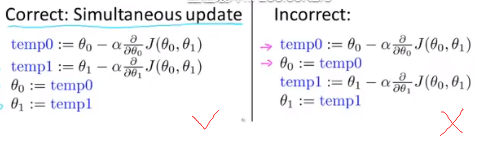
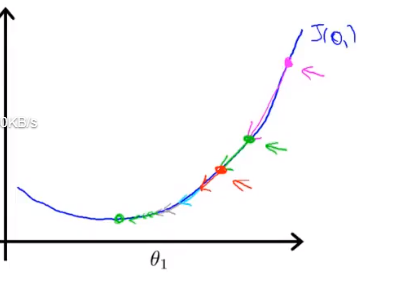
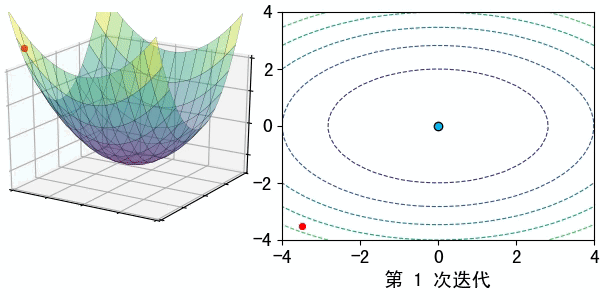
**使用梯度下降法最小化代价函数*J(θ)***



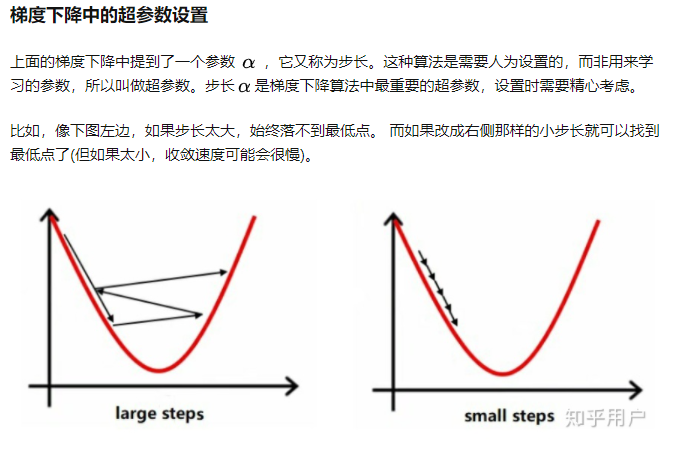
clipboard.png





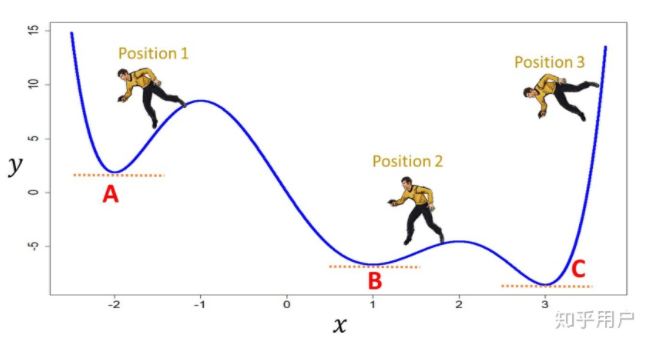


**问题1：步长α的设置问题**



**问题2：局部最优解**

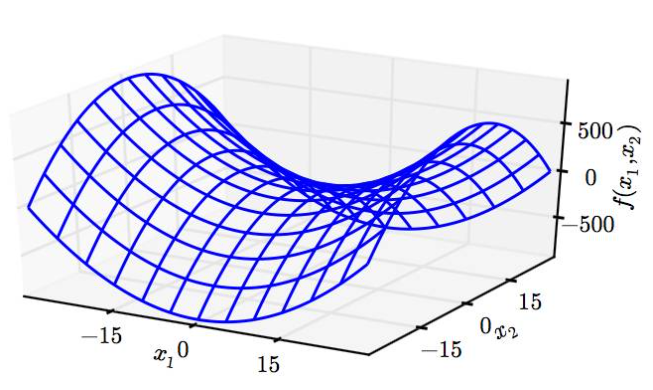
如果目标函数有多个极小值点（很多个低谷），那么如果开始位置不理想，很可能导致最终卡在一个局部极小值。比如下图的例子



需要多次用不同初始值运行算法，关键损失函数的最小值，选择损失函数最小化的初值。

局部最优解很少能找到，只能接近。

鞍点：



**批量梯度下降法**

这是梯度下降法的基本类型，这种方法 **使用整个数据集（** *the complete dataset* **）去计算代价函数的梯度** 。每次使用全部数据计算梯度去更新参数， **批量梯度下降法会很慢** ，并且很难处理不能载入内存（ *don’t fit in memory* ）的数据集。**批量梯度下降法不适合大数据集**

**随机梯度下降法**

随机梯度下降法的第一步是随机化整个数据集。在每次迭代仅选择一个训练样本去计算代价函数的梯度，然后更新参数。即使是大规模数据集，随机梯度下降法也会很快收敛。**随机梯度下降法得到结果的准确性可能不会是最好的，**因为每一轮梯度更新都完全与上一轮的数据和梯度无关**，但是计算结果的速度很快。**

**优化方式：随机平均梯度下降法：**在内存中为每一个样本都维护一个旧的梯度，随机选择第i个样本来更新此样本的梯度，其他样本的梯度保持不变，然后求得所有梯度的平均值，进而更新了参数，如此，每一轮更新仅需计算一个样本的梯度，计算成本等同于SG，但收敛速度快得多

**小批量梯度下降法**

小批量梯度下降法是最广泛使用的一种算法，该算法每次使用*m*个训练样本（称之为一批）进行训练，能够更快得出准确的答案。**小批量梯度下降法不是使用完整数据集，在每次迭代中仅使用*m*个训练样本去计算代价函数的梯度**。一般小批量梯度下降法所选取的样本数量在50到256个之间，视具体应用而定。

回归

回归在数学上来说是给定一个点集，能够用一条曲线去拟合之，如果这个曲线是一条直线，那就被称为线性回归，如果曲线是一条二次曲线，就被称为二次回归