1 知识准备

鞍点：

鞍点 （saddle point）的数学含义是： 目标函数在此点上的梯度（一阶导数）值为 0， 但从该点出发的一个方向是函数的极大值点，而在另一个方向是函数的极小值点。

判断鞍点的一个充分条件是：函数在一阶导数为零处（驻点）的Hessian矩阵为不定矩阵。

半正定矩阵： 所有特征值为非负，或主子式全部非负。

半负定矩阵：所有特征值为非正，或主子式负正相间。

不定矩阵：特征值有正有负，或主子式不满足上面的两种情况。

泰勒展开式：

泰勒公式是将一个在x=x0处具有n阶导数的函数f(x)利用关于(x-x0)的n次[多项式](https://baike.baidu.com/item/%E5%A4%9A%E9%A1%B9%E5%BC%8F" \t "https://blog.csdn.net/qq_38906523/article/details/_blank)来逼近函数的方法。



2无约束求解

# 2.1 梯度下降法(Gradient Descent)

2.1.1

随机梯度下降  [stochastic gradient descent](https://www.google.com.hk/search?safe=strict&q=stochastic+gradient+descent&sa=X&ved=0ahUKEwjllrODzsHSAhWBzLwKHXwvB0sQ4QIIFygA)  SGD：通过每个样本来迭代更新一次



for i in range(nb\_epochs):

np.random.shuffle(data)

for example in data:

params\_grad = evaluate\_gradient(loss\_function, example, params)

params = params - learning\_rate \* params\_grad

## 批量梯度下降 Mini-Batch Gradient Descent：**每一次利用一小批样本，即 n 个样本进行计算**



for i in range(nb\_epochs):

np.random.shuffle(data)

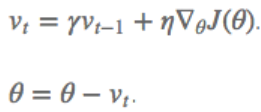
for batch in get\_batches(data, batch\_size=50):

params\_grad = evaluate\_gradient(loss\_function, batch, params)

params = params - learning\_rate \* params\_grad

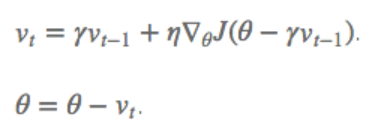
## **2.1.2 动量优化Momentum**

当前权值的改变会受到上一次权值改变的影响，类似于小球向下滚动的时候带上了惯性。这样可以加快小球向下滚动的速度，可以使得梯度方向不变的维度上速度变快，梯度方向有所改变的维度上的更新速度变慢，这样就可以加快收敛并减小震荡



## 牛顿加速梯度 **Nesterov Accelerated Gradient**

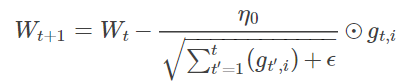
Momentum计算的是当前位置的反向梯度，Nesterov Momentum 计算的是按照上次更新方向走一小步后的反向梯度。



# **2.1.3** 自适应学习率

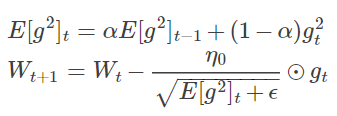
## **AdaGrad算法**

Adagrad算法在每一步的计算的时候，根据历史梯度对学习率进行修改，G包含了针对所有历史梯度的平方和



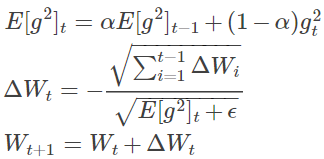
**RMSProp算法**

改变梯度累积为指数衰减的移动平均以丢弃遥远的过去历史。



**AdaDelta算法**

AdaGrad算法和RMSProp算法都需要指定全局学习率，AdaDelta算法结合两种算法每次参数的更新步长即

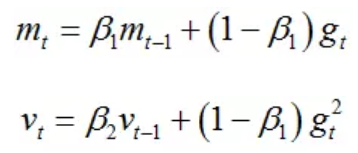


**Adam算法Adaptive Moment Estimation**

AdaGrad和AdaDelta在SGD基础上增加了二阶动量。把一阶动量和二阶动量都用起来，就是Adam了

两部分：对历史平方梯度v(t)乘上一个衰减因子和一个历史梯度m(t)

SGD的一阶动量，加上AdaDelta的二阶动量



在迭代初始阶段，mtmt和vtvt有一个向初值的偏移（过多偏向0），因此需要对一阶和二阶动量进行偏置校正（bias correction）

