13.2 梯度下降法的优化

梯度下降法：求解函数极值的方法

- 函数在某一点的**梯度**：**向量**
- **梯度方向**：函数值**变化最快**的方向

二元函数

$$\nabla = \begin{pmatrix} \frac{\partial f(w,b)}{\partial w} \\ \frac{\partial f(w,b)}{\partial b} \end{pmatrix}$$

多元函数

$$\nabla Loss(W) = \begin{pmatrix} \frac{\partial Loss(W)}{\partial w_0} \\ \frac{\partial Loss(W)}{\partial w_1} \\ \cdots \\ \frac{\partial Loss(W)}{\partial w_n} \end{pmatrix}$$

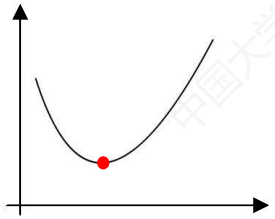
- 沿着**损失函数梯度方向**更新权值

$$W = W - \eta \nabla Loss(W)$$

$$W^{(k+1)} = W^{(k)} - \eta \nabla Loss(W^{(k)})$$

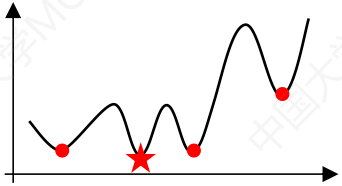
- **梯度为零**时，停止迭代

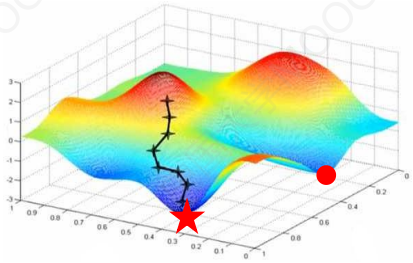
驻点：导数为0的点

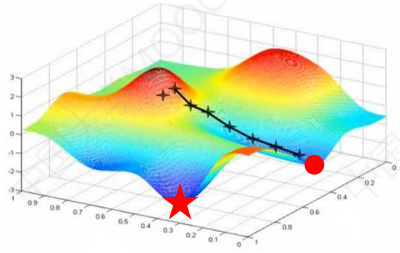


非凸函数

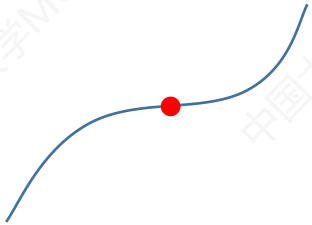
局部极小值点

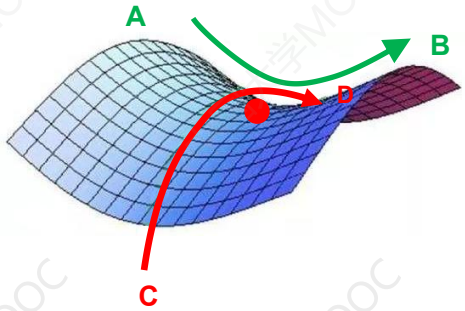


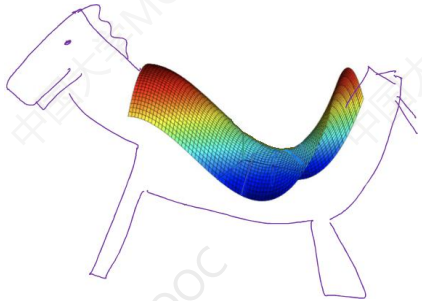




鞍点







多层神经网络使用梯度下降法，**无法保证一定可以收敛于全局最小值点**

口 小批量梯度下降法

一元线性回归

$$Loss = \frac{1}{2}\sum_{i=1}^{t}(y_i - \hat{y}_i)^2 = \frac{1}{2}\sum_{i=1}^{t}(y_i - (wx_i + b))^2$$

$$w^{(k+1)} = w^{(k)} - \eta \frac{\partial Loss(w^{(k)}, b^{(k)})}{w^{(k)}}$$

$$b^{(k+1)} = b^{(k)} - \eta \frac{\partial Loss(w^{(k)}, b^{(k)})}{\partial b^{(k)}}$$

影响小批量梯度下降法的主要因素

- 小批量样本的选择
- 批量大小
- 学习率
- 梯度

小批量样本的选择

$$Loss = \frac{1}{2}\sum_{i=1}^{t}(y_i - \hat{y}_i)^2 = \frac{1}{2}\sum_{i=1}^{t}(y_i - (wx_i + b))^2$$

在**每轮**训练之前，**打乱样本顺序**

数据集的**连续的样本之间有高度的相关性**

	WBC	RBC	HGB	HCT	MCV	MCH	MCHC	PLT	LYMPHF	NEUTP	MONOP
1	7.3	4.7	151.0	45.6	92.6	31.7	342.0	215.0	32.1	57.2	9.1
2	9.0	4.0	151.8	44.0	8.5	30.6	356.6	266.9	32.6	5.7	5.1
3	6.5	4.0	153.9	50.5	9.3	33.0	337.4	207.7	33.9	5.2	4.9
4	7.7	4.9	150.5	41.8	8.6	32.6	343.7	273.9	34.6	6.3	5.3
5	7.3	4.0	149.1	42.7	8.6	31.6	350.7	221.0	34.2	6.4	4.7
6	7.5	4.6	151.1	45.1	9.2	32.4	363.8	289.1	31.6	5.2	5.3
7	8.8	4.2	145.7	47.3	8.7	31.6	357.4	262.7	31.5	5.7	5.5
8	7.8	5.0	151.7	44.1	8.6	30.9	354.0	284.2	31.3	5.9	5.9
9	8.7	4.1	146.4	44.0	8.8	31.1	354.5	277.4	35.0	6.2	6.4
10	8.2	4.8	149.0	49.1	8.6	32.7	340.0	219.4	31.9	6.2	4.9
11	8.2	4.2	153.8	43.4	8.8	31.4	363.3	255.9	31.1	5.6	5.5
12	8.4	4.7	150.1	48.3	8.8	32.6	352.7	261.5	29.9	6.1	4.7
13	9.0	4.7	145.1	45.2	8.9	31.5	345.2	294.8	34.2	5.4	5.7
14	7.9	4.7	148.8	41.1	9.4	30.1	352.0	256.0	30.6	5.5	5.1
15	8.5	4.0	152.3	41.6	8.5	32.2	353.0	227.5	29.5	5.8	5.8
16	8.0	4.8	150.3	48.3	9.0	29.7	344.7	223.1	33.6	5.7	6.1
17	8.6	4.0	145.4	46.8	8.6	29.2	360.3	287.7	28.5	5.2	5.6
18	8.6	4.2	145.1	40.7	8.5	30.5	344.1	247.0	31.8	5.9	6.4
19	7.7	4.3	150.6	47.5	9.3	29.1	361.8	212.1	32.2	5.8	6.2
20	7.9	4.1	145.2	51.1	9.1	31.8	346.8	259.8	33.3	5.3	5.6
21	7.0	5.0	153.5	42.6	8.6	30.4	357.0	294.4	31.6	5.2	5.7
22	7.5	4.1	146.4	46.9	8.5	29.1	342.9	294.9	28.9	5.3	4.5
23	8.2	4.8	145.9	40.8	8.6	30.9	339.8	292.1	29.7	6.5	6.5
24	8.2	4.2	147.5	44.1	9.3	32.8	339.9	200.6	28.7	6.0	6.1
25	8.6	4.1	148.3	51.1	8.9	31.5	338.6	202.4	28.5	6.5	5.9
26	6.7	4.8	148.9	46.2	9.4	30.4	344.1	242.4	29.4	5.2	5.1
27	7.2	4.5	145.8	49.8	8.5	29.1	358.3	218.4	30.0	5.4	5.7
28	8.4	4.2	151.2	48.1	9.0	31.1	342.8	200.5	34.3	5.4	4.6
29	6.5	4.6	146.3	40.8	8.5	29.9	362.6	224.3	29.2	5.8	5.7
30	6.7	4.2	150.1	40.2	8.5	32.6	364.3	244.9	32.5	6.2	5.4
31	7.1	4.0	147.6	40.2	8.7	32.3	345.7	281.0	29.8	6.5	5.3
32	6.8	4.2	148.1	49.8	8.8	29.0	340.7	219.1	29.4	5.5	5.2
33	6.9	4.0	149.2	42.7	9.1	32.1	345.9	223.8	32.3	6.3	4.8
34	9.0	5.0	145.3	45.8	8.9	32.8	354.9	216.1	29.7	6.4	5.2
35	8.8	4.7	147.8	43.3	8.7	31.7	356.6	237.0	31.0	6.1	5.5
36	6.5	4.2	152.5	45.5	8.7	29.7	359.3	234.3	34.9	5.7	4.7
37	7.0	4.3	150.3	50.1	8.8	32.2	342.2	272.3	28.5	6.3	6.1
38	7.6	4.0	146.4	43.5	9.4	29.2	338.9	292.7	30.8	5.3	4.6
39	6.9	4.3	147.6	50.1	8.6	31.1	359.3	280.0	28.1	6.1	4.7

- **小批量中样本的数量**（mini-batch size）
  - 批量中的**样本数量越多**，**梯度方向越准确**，迭代次数越少

    批量梯度下降法：小批量样本=整个数据集
  - 批量中的**样本数量越少**，**随机性越大**，迭代次数越多

    随机梯度下降法：小批量样本数=1
  - 充分利用处理器资源，进行**并行计算**

    2的幂数：32、64、128、256

学习率

$$w^{(k+1)} = w^{(k)} - \eta \frac{\partial Loss(w^{(k)}, b^{(k)})}{w^{(k)}}$$

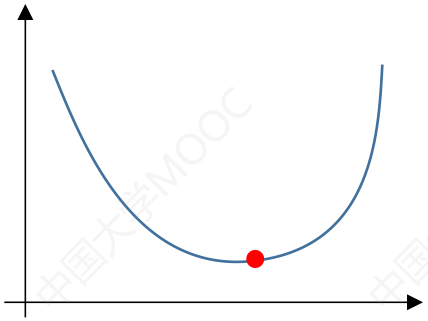
$$b^{(k+1)} = b^{(k)} - \eta \frac{\partial Loss(w^{(k)}, b^{(k)})}{\partial b^{(k)}}$$

固定学习率

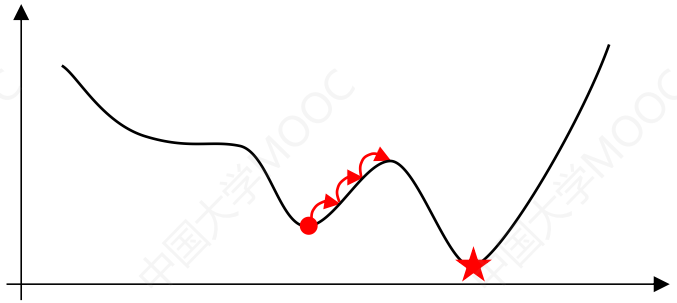
学习率设置**过小**，收敛速度慢；
学习率设置**过大**，震荡，无法收敛

动态调整学习率：在训练过程中，动态的调整学习率

- **学习率衰减**（Learning Rate Decay）/**学习率退火**（Learning Rate Annealing）
  - **开始训练**时，设置**较大**学习率，加快收敛速度
  - **迭代过程**中，学习率随着迭代次数**逐渐减小**，避免震荡



凸函数



非凸函数

- **调节学习率**——非凸函数
  - **周期性**的增大学习率
  - **自适应**调整学习率
  - 自适应调整**每个参数**的学习率

$$w^{(k+1)} = w^{(k)} - \eta \frac{\partial Loss(w^{(k)}, b^{(k)})}{w^{(k)}}$$

$$b^{(k+1)} = b^{(k)} - \eta \frac{\partial Loss(w^{(k)}, b^{(k)})}{\partial b^{(k)}}$$

- **自适应学习率算法**
  - **AdaGrad**
    - 全局学习率 η
    - 各个**参数自适应**的调整学习率 $\frac{\eta}{\sqrt{\sum_{r=1}^{t}(grad_r)^2 + \varepsilon}}$
    - 学习率随着迭代次数的增加而**单调减小**
  - **RMSprop**、和**AdaDelta**
    - 改进了AdaGrad算法**调整学习率**的方法
    - 各参数的学习率**不是单调递减**的

梯度估计

$$w^{(k+1)} = w^{(k)} - \eta \frac{\partial Loss(w^{(k)}, b^{(k)})}{w^{(k)}}$$

$$b^{(k+1)} = b^{(k)} - \eta \frac{\partial Loss(w^{(k)}, b^{(k)})}{\partial b^{(k)}}$$

当前梯度
历史梯度 → 参数更新

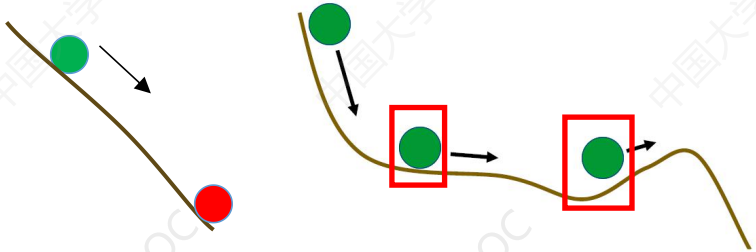
动量梯度下降法（Momentum）

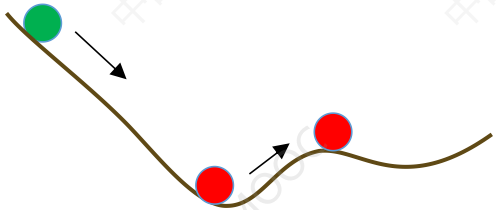
动量=质量×速度

在更新参数时，可以在一定程度上**保留之前的更新方向**

$$\begin{cases} W^{(k+1)} = W^{(k)} + v^{(k+1)} \\ v^{(k+1)} = \alpha v^{(k)} - \eta \nabla Loss\,(W^{(k+1)}, X^{(i)}, Y^{(i)}) \end{cases}$$

梯度方向不变：步长更大，收敛加快
梯度方向改变：步长变小，更新变慢



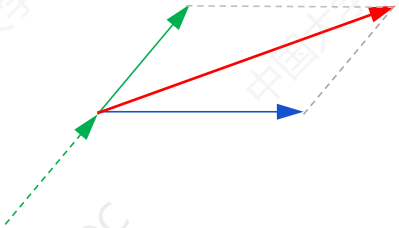


牛顿加速梯度算法（Nesterov Accelerated Gradient，NAG）

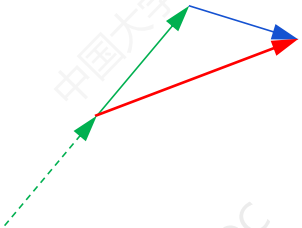
- 根据当前更新方向，估算下一步的梯度方向
- 在新位置计算梯度，修正梯度方向

$$\begin{cases} W^{(k+1)} = W^{(k)} + v^{(k+1)} \\ v^{(k+1)} = \alpha v^{(k)} - \eta \nabla Loss(W^{(k+1)} - \mu v^{(k)}, X^{(i)}, Y^{(i)}) \end{cases}$$

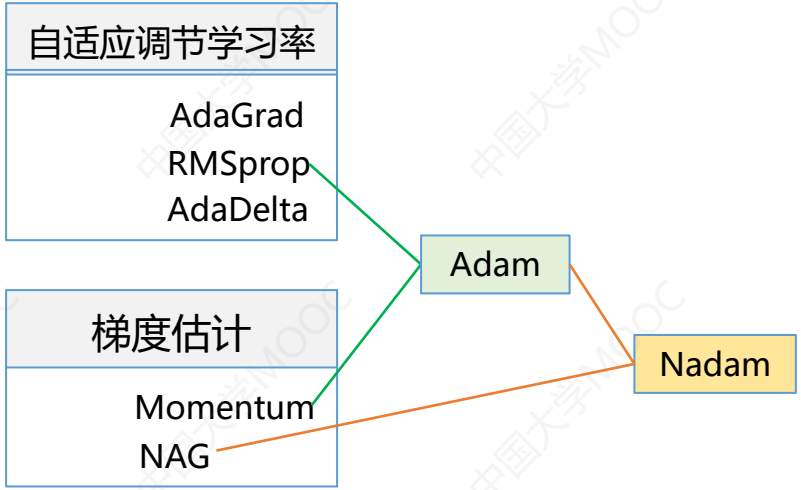
Momentum

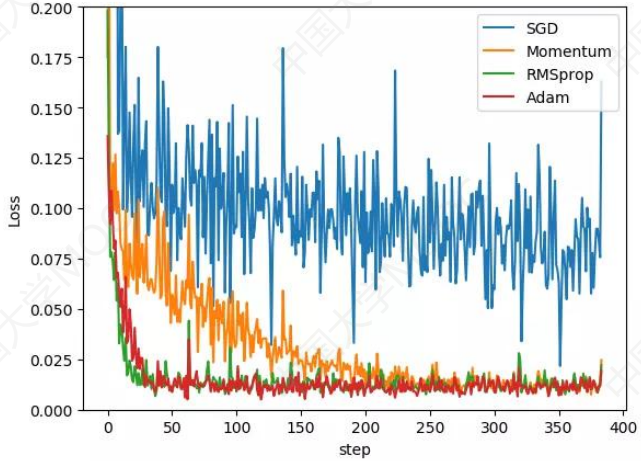


NAG



自适应学习率+梯度估计


自适应调节学习率
AdaGrad
RMSprop
AdaDelta
梯度估计
Momentum
NAG
Adam
Nadam


SGD
Momentum
RMSprop
Adam
Loss
step
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0
50
100
150
200
250
300
350
400