typora-copy-images-to: picture

优化迭代方法统一论

一：问题引入：线性回归问题

现在，我有如下一组关于病人收缩压的数据，包括患者姓名，性别，年龄，体重等信息，每一种信息为表中的一列。数据其中第6列为病人的收缩压。根据已有的这些数据记录，我需要对新的病例进行预测，那么怎么办呢？按照机器学习的方法，是首先对已有的数据进行训练，得到一个模型，然后利用该模型对新的未知病例进行预测。

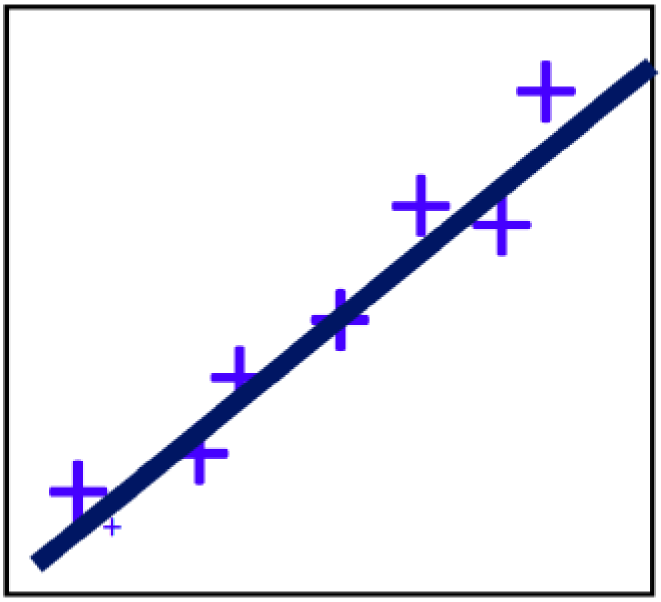
	1 name	2 sex	3 age	4 wgt	5 smoke	6 sys	7 dia	8 trial1	9 trial
1 YPL-320	'SMITH'	'm'	38	176	1	124	93	18	
2 GLI-532	'JOHNSON'	'm'	43	163	0	109	77	11	
3 PNI-258	'WILLIAMS'	'f'	38	131	0	125	83	-99	
4 MIJ-579	'JONES'	'f'	40	133	0	117	75	6	
5 XLK-030	'BROWN'	'f'	49	119	0	122	80	14	
6 TFP-518	'DAVIS'	'f'	46	142	0	121	70	19	
7 LPD-746	'MILLER'	'f'	33	142	1	130	88	0	
8 ATA-945	'WILSON'	'm'	40	180	0	115	82	-99	
9 VNL-702	'MOORE'	'm'	28	183	0	115	78	2	
10 LQW-768	'TAYLOR'	'f'	31	132	0	118	86	11	
11 QFY-472	'ANDERS...	'f'	45	128	0	114	77	8	
12 UJG-627	'THOMAS'	'f'	42	137	0	115	68	4	
13 XUE-826	'JACKSON'	'm'	25	174	0	127	74	-99	
14 TRW-072	'WHITE'	'm'	39	202	1	130	95	8	

符号说明：

1.$\left\{\left(x^{(i)}, y^{(i)}\right)\right\}$是一个训练样本，其中上角标$i$表示样本的编号；

2.$\left\{\left(x^{(i)}, y^{(i)}\right); i=1, \cdots, N\right\}$是训练样本集，共有$N$个样本；

3.$\left\{\left(x_1^{(i)}, x_2^{(i)}, y^{(i)}\right)\right\} \rightarrow\left\{\left(\mathbf{x}^{(i)}, y^{(i)}\right)\right\}, \mathbf{x}^{(i)}=\left[\begin{array}{c}x_1^{(i)} \\ x_2^{(i)}\end{array}\right]$，将多个影响因素组合成一个向量表示。其中$\mathbf{x}^{(i)}$表示特征，$y^{(i)}$表示预测值（标签值）。



上图便是我们熟悉的线性回归模型，只不过是一维情况下的示意图。在实际的机器学习过程中，影响y的因素肯定不只有一个，就拿上述收缩压的例子来讲，影响收缩压的因素就有性别，年龄等诸多因素。因此，一维情形下的线性回归模型肯定不能够满足要求。这就引出了多维情形下的线性回归模型。

以下对一维和多维情形下的线性回归问题进行对比观察：

- 对于一维的线性回归，试图学习：$f(x) = wx + b$，使得$f\left(x^{(i)}\right) \approx y^{(i)}$
- 对于多维的线性回归，试图学习：$f(\mathrm{x}) = w^T\mathrm{x} + b$，使得$f\left(\mathrm{x}^{(i)}\right) \approx y^{(i)}$，其中输入为向量，输出是标量。$w^T\mathrm{x}$代表向量内积（或者称为向量点乘），最终的结果是一个具体的数字（标量）。在线性代数中，向量默认是列向量。

接下来，核心的问题就在于怎么学到w和b?

二：无约束优化梯度分析法

2.1 定义无约束优化问题

自变量为标量的函数f: $R \rightarrow R$:

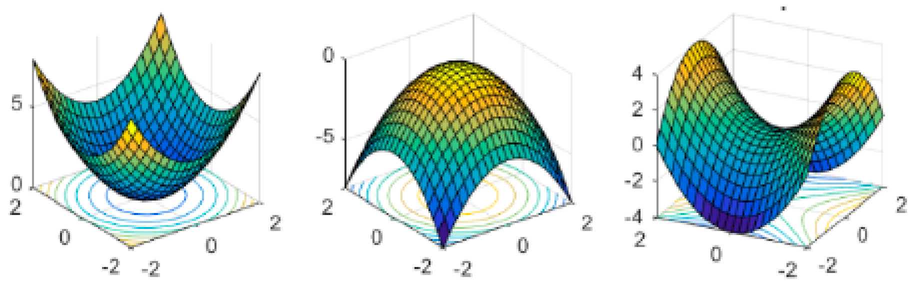
$$\min f(x) \quad x \in R$$

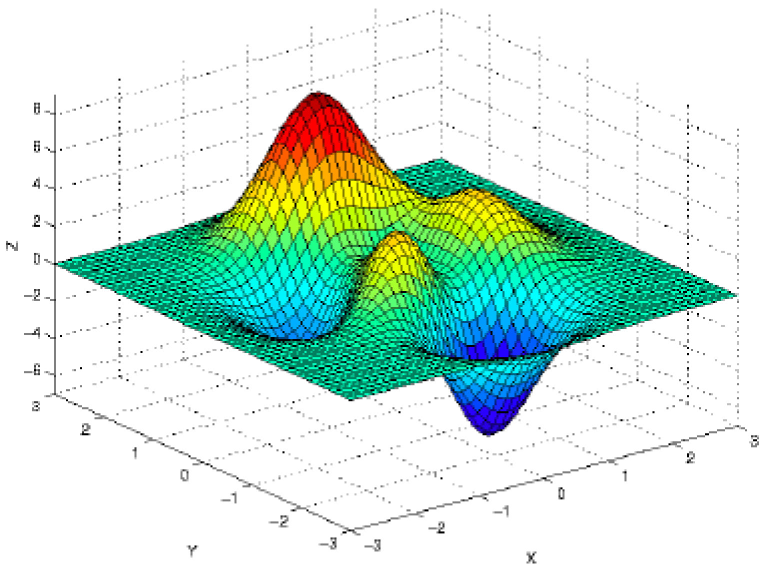
自变量为向量的函数f: $R^n \rightarrow R$:

$$min f(\mathrm{x}) \quad \mathrm{x} \in R^n$$

通过将一维和多维情形下的优化函数进行对比，我们可以清楚的明白，优化问题就是要求一个函数的最小值。在一维情况下，自变量为标量，而在多元情况下，自变量变成向量，但是最优的函数值依旧是标量。在实际应用中，一元的情况很少见，最常见到的就是多元的情况，而且自变量x的维度有可能非常高。

优化问题可能的极值点情况：





第一个图有极小值，第二个图有极大值，第三个图有鞍点(saddle point)，可以类比（$y = x^3 \quad x = 0$）的情况。第四张图中，既有极大值也有极小值，而且有局部极大（小）值。在实际的应用中，最常出现的是最后一种图，当维度很高时，我们有时候根本就不可能知道函数到底是什么样子的，也无法可视化。而且我们往往只能找到函数的局部极值，很难找到函数的全局最值（客观条件所限）。但是能够找到函数的局部极值也是非常有意义的。

2.2 梯度和Hessian矩阵

同样采用一阶和二阶对照的角度来理解

一阶导数和梯度：$f'(x);\quad \mathbf{g}(\mathbf{x}) = \nabla f(\mathbf{x}) = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f(\mathbf{x})}{\partial x_1} \\ \vdots \\ \frac{\partial f(\mathbf{x})}{\partial x_n} \end{bmatrix}$

注解：

1. 导数的大小代表了函数在某个方向上变化的快慢；梯度的方向为函数值增加最快的方向。梯度本身是一个n维向量。
2. （一阶导数为对x（标量）求导，二阶导数为$\mathbf{x}$(n维的向量）求导，结果为f对每一个x单独求导，然后组成一个向量（列向量）。）

二阶导数和Hessian矩阵：

$$f''(x); \quad \mathbf{H}(\mathbf{x}) = \nabla^2 f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1^2} & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_n} \cdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_1} & \frac{\partial^2 f(\mathbf{x})}{\partial x_2^2} & & \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_1} & \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_n^2} \end{bmatrix} = \nabla(\nabla f(\mathbf{x}))^T$$

注解：

1. 在多维情况下，二阶导数即为Hessian矩阵，在梯度的基础上再求一次导。是一个n$*$n的矩阵。
2. Hessian矩阵其实是一个实对称矩阵，对角元相等。

2.3 二次型

2.3.1 定义

给定矩阵$\mathbf{A} \in \mathbb{R}^{n \times n}$，函数

$$\mathbf{x}^T \mathbf{A} \mathbf{x} = \sum_{i=1}^{n} x_i (\mathbf{A}\mathbf{x})_i = \sum_{i=1}^{n} x_i \left(\sum_{j=1}^{n} a_{ij} x_j \right) = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j a_{ij}$$

被称为二次型。

- 给定对称矩阵$\mathbf{A} \in \mathbb{R}^{n \times n}$，如果对于所有的$\mathbf{x} \in R^n$,有$\mathbf{X}^T \mathbf{A} \mathbf{x} \geq 0$，则为半正定矩阵，此时特征值$\lambda(\mathbf{A}) \geq 0$.
- 如果对于所有的$\mathbf{x} \in \mathbb{R}^n, \mathbf{x} \neq 0$，有$\mathbf{X}^T \mathbf{A} \mathbf{x} > 0$，则为正定矩阵。反之，如果小于0，则为负定矩阵，否则为不定矩阵。

上式注解：

1.A是一个实对称矩阵。

2.$\mathbf{A}\mathbf{x}$的乘积可以看作是一个列向量，然后与$\mathbf{x}^T$相乘。这样其实就是两个列向量做点乘，结果是一个具体的数（标量）。

3.可以类比$x*2*x>0$,此时对于任意的x(x不为0)，函数值均大于0，此时2为正数。

2.3.2 具体计算

- 向量a与x无关，则$\nabla\left(\mathbf{a}^T\mathbf{x}\right)=\mathbf{a}, \nabla^2\left(\mathbf{a}^T\mathbf{x}\right)=\mathbf{0}$
- 对称矩阵$\mathbf{A}$与x无关，则$\nabla\left(\mathbf{x}^T\mathbf{A}\mathbf{x}\right)=\mathbf{2Ax}, \nabla^2\left(\mathbf{x}^T\mathbf{A}\mathbf{x}\right)=2\mathbf{A}$ （可类比$(ax^2)'=2ax; (ax^2)''=2a.$
- 最小二乘：

$$
\begin{aligned}
f(\mathrm{x}) &= \|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2 \\
&= (\mathbf{A}\mathbf{x}-\mathbf{b})^T(\mathbf{A}\mathbf{x}-\mathbf{b}) \\
&= \mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x}-\mathbf{x}^T\mathbf{A}^T\mathbf{b}-\mathbf{b}^T\mathbf{A}\mathbf{x}+\mathbf{b}^T\mathbf{b} \\
&= \mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x}-2\mathbf{b}^T\mathbf{A}\mathbf{x}+\mathbf{b}^T\mathbf{b} \\
\nabla f(\mathbf{x}) &= 2\mathbf{A}^T\mathbf{A}\mathbf{x}-2\mathbf{A}^T\mathbf{b}
\end{aligned}
$$

2.4 泰勒级数

2.4.1 泰勒级数展开（标量和向量）

- 输入为标量的泰勒级数展开

$$
f\left(x_k+\delta\right) \approx f\left(x_k\right)+f'\left(x_k\right)\delta+\frac{1}{2}f''\left(x_k\right)\delta^2+\cdots+\frac{1}{k!}f^k\left(x_k\right)\delta^k+\cdots
$$

- 输入为向量的泰勒级数展开

$$
f\left(\mathbf{x}_k+\boldsymbol{\delta}\right) \approx f\left(\mathbf{x}_k\right)+\mathbf{g}^T\left(\mathbf{x}_k\right)\boldsymbol{\delta}+\frac{1}{2}\boldsymbol{\delta}^T\mathbf{H}\left(\mathbf{x}_k\right)\boldsymbol{\delta}
$$

注解

1. 理解向量情况时，与标量情况进行对照理解。
2. $\mathbf{g}^T\left(\mathbf{x}_k\right)$为梯度的转置（由列向量转变为行向量），相当于求一阶导数。$\mathbf{H}\left(\mathbf{x}_k\right)$为Hessian矩阵，详单与求二阶导数。因为后边的高阶项数值太小，因此只保留到二阶项。
3. $\boldsymbol{\delta}$可正可负，代表x周边很小的一个值。

2.4.2 泰勒级数和极值

标量情况

- 输入为标量的泰勒级数展开：（保留到二阶项）

$$
f\left(x_k+\delta\right) \approx f\left(x_k\right)+f'\left(x_k\right)\delta+\frac{1}{2}f''\left(x_k\right)\delta^2
$$

- 严格的局部极小点是指：$f\left(x_k+\delta\right)>f\left(x_k\right)$

- 称满足$f'(x)=0$的点为平稳点（候选点）
- 函数在x_k由严格局部极小值的条件为$f'(x)=0$且$f''(x)>0.$

向量情况（一定对照标量情况理解）

- 输入为向量的泰勒级数展开：（保留到二阶项）

$$f\left(\mathbf{x}_k+\boldsymbol{\delta}\right)\approx f\left(\mathbf{x}_k\right)+\mathbf{g}^T\left(\mathbf{x}_k\right)\boldsymbol{\delta}+\frac{1}{2}\boldsymbol{\delta}^T\mathbf{H}\left(\mathbf{x}_k\right)\boldsymbol{\delta}$$

- 称满足$g(\mathbf{x_k})=0$的点为平稳点（候选点），此时如果$\mathbf{H(x_k)}$为正定矩阵，则$\mathbf{x_k}$为一严格局部极小点；如果为负定矩阵，则为严格局部极大点；如果为不定矩阵，则为鞍点（saddle point）。

通过2.4.1和2.4.2的分析，我们可以发现，当我们想要求函数的极小值时，首先需要找到一阶导数为0的点，然后再判断这些点处二阶导数的情况。但是实际中，当求解梯度为0时存在一些局限性。比如：

计算$f(x)=x^4+\sin\left(x^2\right)-\ln(x)e^x+7$的导数。

$$\begin{aligned}
f'(x)&=4x^{(4-1)}+\frac{d\left(x^2\right)}{dx}\cos\left(x^2\right)-\frac{d(\ln x)}{dx}e^x-\ln(x)\frac{d\left(e^x\right)}{dx}+0\\
&=4x^3+2x\cos\left(x^2\right)-\frac{1}{x}e^x-\ln(x)e^x
\end{aligned}$$

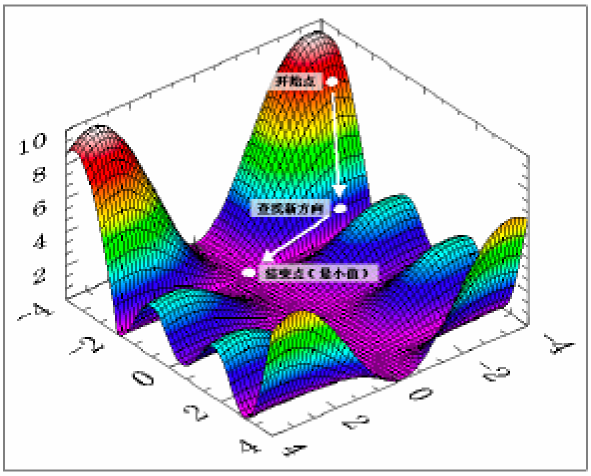
从上面的结果中可以看出，当$f'(x)=0$时，很难通过直接求导等于0的方法求出显式解。此时，我们就需要采用另外的方法来解决这个问题，此时，无约束优化迭代法应运而生。

三：无约束优化迭代法

3.1 迭代法的基本结构（最小化$f(x)$）

1. 选择一个初始点，设置一个收敛容忍度ϵ，计数$k=0$
2. 决定搜索方向$\mathbf{d_k}$，使得函数下降。（核心步骤）算法预算法最本质的区别就在于搜索方向的不同
3. 决定步长α_k，使得$f(\mathbf{x_k}+\alpha_\mathbf{k}\mathbf{d_k})$对于$\alpha_\mathbf{k}\geq\mathbf{0}$最小化，构建$\mathbf{x_{k+1}}=\mathbf{x_k}+\alpha_\mathbf{k}\mathbf{d_k}$
4. 如果$||\mathbf{d_k}||_2<\epsilon$,则停止迭代（说明梯度已经非常小了，这时已经非常接近极值点了）；否则继续迭代

α_k太大，则容易在最低值处震荡，甚至冲过最低点导致不收敛。如果太小，则收敛速度会很慢，在实际应用中，这个值就是需要调的参数。



3.2 梯度下降法

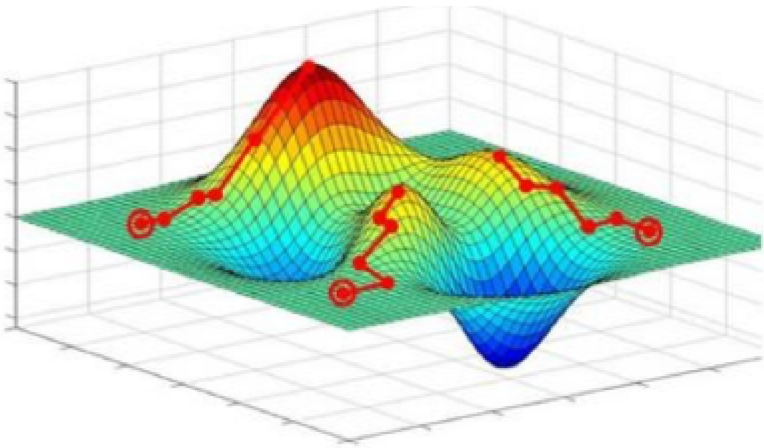
- 方向选取：$\mathbf{d_k} = -\mathbf{g(x_k)}$(**最重要**)

原因分析：

我们展开泰勒级数，只保留一阶项，则$\mathbf{f(x_k + d_k)} \approx \mathbf{f(x_k)} + \mathbf{g^T(x_k)d_k}$,既然要使得函数值下降，则必须要使得$\mathbf{f(x_k + d_k)} < \mathbf{f(x_k)}$，也即是要求$\mathbf{g^T(x_k)d_k} < 0$，这就说明是两个向量的内积小于0，相当于两个向量的夹角大于90度（$-1 \leq cos(\theta) \leq 1$）。当夹角为180度时，两个向量的内积最小（绝对值最大），此时$\mathbf{d_k} = -\mathbf{g(x_k)}$，$\mathbf{f(x_k + d_k)}$下降最多。

注释

1. 两个向量的内积$\mathbf{a} \cdot \mathbf{b} = \mathbf{a^T b} = ||\mathbf{a}||||\mathbf{b}||\mathbf{cos}(\theta)$



2. 在保留一阶项的时候，梯度下降法是最优的方法，所选取的负梯度方向为最优的方向；但是这并不代表负梯度方向就是全局最优的方向，因为我们把二阶项给舍弃了。

3.3 牛顿法

3.3.1 牛顿法介绍

方向：$\mathbf{d_k}=-\mathbf{H^{-1}(x_k)g(x_k)}$

方向选取的依据：

$$f\left(\mathbf{x}_k+\mathbf{d}_k\right)=f\left(\mathbf{x}_k\right)+\mathbf{g}^T\left(\mathbf{x}_k\right)\mathbf{d}_k+\frac{1}{2}\mathbf{d}_k^T\mathbf{H}\left(\mathbf{x}_k\right)\mathbf{d}_k$$

在上面这个式子中，$\mathbf{x_k}$是已知的，$\mathbf{d_k}$是未知的。我们的目的是找到一个$\mathbf{d_k}$使得$\mathbf{f(x_k+d_k)}$最小，因此我们对$\mathbf{d_k}$求导，得到：

$$\frac{\partial f\left(\mathbf{x}_k+\mathbf{d}_k\right)}{\partial \mathrm{d}_k}=\mathbf{0}\Rightarrow\mathbf{g}\left(\mathbf{x}_k\right)+\mathbf{H}\left(\mathbf{x}_k\right)\mathbf{d}_k=\mathbf{0}$$

如果Hessian正定，则有$\mathbf{d}_k=-\mathbf{H}^{-1}\left(\mathbf{x}_k\right)\mathbf{g}\left(\mathbf{x}_k\right)$。

注：需要强制要求Hessian矩阵正定。原因如下：

（1）把$\mathbf{d_k}$的结果表达式代入，可得：$f\mathbf{(x_k+d_k)}=f\mathbf{(x_k)-1/2g^T(x_k)H^{-1}(x_k)g(x_k)}$,只有当$\mathbf{H^{-1}(x_k)}$正定，也就是$\mathbf{H(x_k)}$正定时，才能保证$f\mathbf{(x_k+d_k)}<f\mathbf{(x_k)}$,即函数值下降。

（2）只有当H正定时，才能保证H可逆，才能求得$\mathbf{d_k}$。

3.3.2 应用牛顿法的困难点

1. 在实际工程中，Hessian矩阵$\mathbf{H}$很难求，而$\mathbf{H^{-1}}$更加难求。而且$\mathbf{H}$本身可能就不是正定矩阵。
2. 解决办法：
   - 修正牛顿法：当Hessian矩阵不是正定矩阵时，可以对Hessian矩阵进行修正：$\mathbf{H(x_k)+E}$，典型方法$\mathbf{E}=\delta\mathbf{I}$，$\delta>0$很小。这样做的目的是：通过添加一个单位阵，让$\mathbf{E}$中最小的特征值也大于0，这就就可以保证修正后的Hessian矩阵是正定的，然后再求逆矩阵。
   - 拟牛顿法

3.4 拟牛顿法

3.4.1 核心思想

- 统一看待梯度下降法和牛顿法：

$$\mathbf{d_k}=-\mathbf{S_k g_k}$$

其中：$\mathbf{S}_k = \begin{cases} \mathbf{I} & \text{steepest} \\ \mathbf{H}_k^{-1} & \text{Newton} \end{cases}$

- 由于牛顿法的困难之处在于$\mathbf{H}^{-1}$很难求，那么我们可以尝试这样的思路，不直接求$\mathbf{H_k^{-1}}$，而是尝试用一个正定矩阵去逼近$\mathbf{H_k^{-1}}$。
- 定义$\delta_\mathbf{k} = \mathbf{x_{k+1}} - \mathbf{x_k}, \gamma_\mathbf{k} = \mathbf{g_{k+1}} - \mathbf{g_k}$
- 用于近似$\mathbf{H_k^{-1}}$的矩阵应满足这样的条件：$\mathbf{S_{k+1}}\gamma_\mathbf{k} = \delta_\mathbf{k}$
  - 理解方式：$\frac{\mathbf{g_{k+1}} - \mathbf{g_k}}{\mathbf{x_{k+1}} - \mathbf{x_k}} = \mathbf{H_k}$,因此，就可以得到$\mathbf{S_{k+1}} := \frac{\delta_\mathbf{k}}{\gamma_\mathbf{k}} = \mathbf{H_k^{-1}}$，当满足$\mathbf{S_{k+1}}\gamma_\mathbf{k} = \delta_\mathbf{k}$时，$\mathbf{S_{k+1}}$可用来近似$\mathbf{H}^{-1}$
  - 注意：关于$\mathbf{S_{k+1}}$的推导是不严谨的，仅仅通过上述方法用于理解思想。（即一阶导数再求导，便可以得到二阶导数）
- 只有$\delta_\mathbf{k}$和$\gamma_\mathbf{k}$是不可能计算出$\mathbf{S_{k+1}}$的(因为$\delta_\mathbf{k}$和$\gamma_\mathbf{k}$都是向量，不能直接做除法)，因此，我们继续考虑使用迭代的方法。

3.4.2 DFP法

- 给定初始$\mathbf{S_0} = \mathbf{I}$
- $\mathbf{S}_{k+1} = \mathbf{S}_k + \Delta\mathbf{S}_k, k = 0, 1, \cdots$
- $\Delta\mathbf{S}_k = \alpha\mathbf{u}\mathbf{u}^T + \beta\mathbf{v}\mathbf{v}^T$,因此

$$\mathbf{S}_{k+1} = \mathbf{S}_k + \alpha\mathbf{u}\mathbf{u}^T + \beta\mathbf{v}\mathbf{v}^T$$

- 两边同时乘$\gamma_\mathbf{k}$,有$\delta_k = \mathbf{S}_k\gamma_k + \underbrace{(\alpha\mathbf{u}^T\gamma_k)}_{1}\mathbf{u} + \underbrace{(\beta\mathbf{v}^T\gamma_k)}_{-1}\mathbf{v} = \mathbf{S}_k\gamma_k + \mathbf{u} - \mathbf{v}$，令

  $\alpha\mathbf{u}^T\gamma_k = 1, \beta\mathbf{v}^T\gamma_k = -1$（类似待定系数法）
- 解得：$\alpha = \frac{1}{\mathbf{u}^T\gamma_k}, \beta = -\frac{1}{\mathbf{v}^T\gamma_k}$且$\mathbf{u} - \mathbf{v} = \boldsymbol{\delta}_k - \mathbf{S}_k\gamma_k$，可得$\mathbf{u} = \boldsymbol{\delta}_k$；$\mathbf{v} = \mathbf{S}_k\gamma_k$，从而最终解得DFP更新公式：

$$\mathbf{S}_{k+1} = \mathbf{S}_k + \frac{\delta_k\boldsymbol{\delta}_k^T}{\delta_k^T\gamma_k} - \frac{\mathbf{S}_k\gamma_k\gamma_k^T\mathbf{S}_k}{\gamma_k^T\mathbf{S}_k\gamma_k}$$

  注意：$\mathbf{S_k}$是对称矩阵，其转置和自身相等。

3.4.3 BFGS法

思想与DFP方法一致，区别在于$\Delta\mathbf{S_k}$的选取不一致。一般来讲，BFGS法在数值上更稳定一些。

更新公式：

$$\text{Broyden-Fletcher-Goldfarb-Shanno (BFGS)}: \mathbf{S}_0 = \mathbf{I}$$
$$\mathbf{S}_{k+1} = \mathbf{S}_k + \left(1 + \frac{\gamma_k^T \mathbf{S}_k \gamma_k}{\delta_k^T \gamma_k}\right) \frac{\delta_k \boldsymbol{\delta}_k^T}{\delta_k^T \gamma_k} - \frac{\delta_k \gamma_k^T \mathbf{S}_k + \mathbf{S}_k \gamma_k \boldsymbol{\delta}_k^T}{\delta_k^T \gamma_k}$$

3.5 步长选取问题

第一种方法：每次迭代选择固定的步长。这种方法在实际中最常用，例如$\alpha_k = \alpha = 0.1$。

第二种方法：每次选取最优步长。例如，对于二次型问题：$f(\mathbf{x}) = \mathbf{x^T A x} + \mathbf{2b^T x} + c$,

需要解：$\min\limits_{\alpha \geq 0} f(\mathbf{x} + \alpha \mathbf{d})$,令$h(\alpha) = f(\mathbf{x} + \alpha \mathbf{d})$，则$\frac{\partial h(\alpha)}{\partial \alpha} = 0 \Rightarrow \alpha = -\frac{\mathbf{d}^T \nabla f(\mathbf{x})}{\mathbf{2d}^T \mathbf{Ad}}$。该$\alpha$即为每次迭代时的最优步长。

推导计算

$$\begin{aligned} h(\alpha) &= (\mathbf{x} + \alpha \mathbf{d})^T \mathbf{A} (\mathbf{x} + \alpha \mathbf{d}) + 2\mathbf{b^T} (\mathbf{x} + \alpha \mathbf{d}) + c \\ &= (\mathbf{x^T A} + \alpha \mathbf{d^T A})(\mathbf{x} + \alpha \mathbf{d}) + 2\mathbf{b^T} (\mathbf{x} + \alpha \mathbf{d}) + c \\ &= \mathbf{x^T A x} + \alpha \mathbf{d^T A x} + \alpha \mathbf{x^T A d} + \alpha^2 \mathbf{d^T A d} + 2\mathbf{b^T x} + 2\alpha \mathbf{b^T d} + c \end{aligned}$$
$$\frac{\partial h(\alpha)}{\partial \alpha} = \mathbf{d^T A x} + \mathbf{x^T A d} + 2\alpha \mathbf{d^T A d} + 2\mathbf{b^T d} = 0$$
$$2\mathbf{x^T A d} + 2\alpha \mathbf{d^T A d} + 2\mathbf{b^T d} = 0$$
$$\alpha = \frac{\mathbf{x^T A d} + \mathbf{b^T d}}{-\mathbf{d^T A d}} = \frac{\mathbf{d^T A x} + \mathbf{d^T b}}{-\mathbf{d^T A d}} = \frac{\mathbf{d^T}(2\mathbf{Ax} + 2\mathbf{b})}{-2\mathbf{d^T A d}} = \frac{\mathbf{d^T} \nabla f(\mathbf{x})}{-2\mathbf{d^T A d}}$$

注：当采用梯度下降法时，$\mathbf{d} = -\mathbf{g} = -\nabla f(\mathbf{x})$,$\alpha = \frac{||\nabla f((x)||_2^2}{\mathbf{2d^T A d}}$；当采用牛顿法或者拟牛顿法时，$\mathbf{d} = -\mathbf{S g}$.

通过以上求解，可以得到每次迭代时的最优步长。

四：线性回归求解

4.1 利用梯度等于0直接求解

对于一个线性回归问题，我们试图学习到这样一个模型：$f(\mathbf{x}) = \mathbf{w^T x} + b$，使得$f(\mathbf{x}^{(i)}) \approx y^{(i)}$。关键在于如何学习得到$\mathbf{w}$和$b$。

- 令$\overline{\mathbf{w}} = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$,$\mathbf{X} = \begin{bmatrix} \mathbf{x}^{(1)T} & 1 \\ \vdots & \vdots \\ \mathbf{x}^{(N)T} & 1 \end{bmatrix}_{N \times (d+1)}$,则有$\mathbf{y} \approx \mathbf{X} \overline{\mathbf{w}}$。
- 损失函数：$||\mathbf{y} - \mathbf{X}\overline{\mathbf{w}}||_2^2$，我们的目标在于求解使得损失函数最小的$\overline{\mathbf{w}}$和$b$。即：

$$\min_{\overline{\mathbf{w}},b} ||\mathbf{y}-\mathbf{X}\overline{\mathbf{w}}||_2^2$$

- 损失函数对$\overline{\mathbf{w}}$求导，令导函数为0可得：

$$g(\overline{\mathbf{w}})=0 \Rightarrow 2\mathbf{X}^T(\mathbf{X}\overline{\mathbf{w}}-\mathbf{y})=0 \Rightarrow \overline{\mathbf{w}}^*=\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

这样便可以直接求得最优参数，但是我们也观察到结果中存在求逆的步骤。求逆的运算量特别大，在实际工程中一般都会避免，并且，也不一定在任何情形下均可以求逆。因此，我们可以采用梯度下降法来进行迭代。

4.2 梯度下降法求解

- 梯度下降法

$$\begin{aligned}\mathbf{g}(\overline{\mathbf{w}})&=2\mathbf{X}^T(\mathbf{X}\overline{\mathbf{w}}-\mathbf{y})\\&=2\sum_{i=1}^{N}\mathbf{X}^{(i)}\left(\overline{\mathbf{w}}^T\mathbf{X}^{(i)}-y^{(i)}\right)\\\overline{\mathbf{w}}&\leftarrow\overline{\mathbf{w}}-\alpha\mathbf{g}(\overline{\mathbf{w}})\end{aligned}$$

注意：其中$\mathbf{X}^{(i)}=[\mathbf{x}^{(i)T},1]^T$.是一个列向量。

- 随机梯度下降法(SGD),在实际中很常用。其实就是把梯度下降法中的求和运算去掉，每次更新时，只选择一个样本进行计算。

$$\left\{i=1:N, 2\mathbf{X}^{(i)}\left(\overline{\mathbf{w}}^T\mathbf{X}^{(i)}-y^{(i)}\right)\right\}$$

- 当$||\mathbf{g}(\overline{\mathbf{w}})||_2<\epsilon$时，停止迭代。