非线性优化及算法总结

李扶洋

邮箱：951678201@qq.com

2022 年 7 月 18 日

§非线性优化问题

记$F(\mathbf{x})$是以$\mathbf{x}$为参数的**非线性标量函数**，优化问题典型目标是：

$$\mathbf{x}^* = argmin_x F(\mathbf{x}) \tag{1}$$

如果$F(\mathbf{x})$有连续的二级导数，则任意点$\mathbf{x}$处的泰勒展开式为：

$$F(\mathbf{x}+\Delta\mathbf{x}) \approx F(\mathbf{x}) + \Delta\mathbf{x}^T\mathbf{g}(\mathbf{x}) + \frac{1}{2}\Delta\mathbf{x}^T\mathbf{G}(\mathbf{x})\Delta\mathbf{x} + \cdots \tag{2}$$

其中$\mathbf{g}$是F的梯度，$\mathbf{G}$是F的二阶导：

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \mathbf{g} = \left(\frac{\partial F}{\partial \mathbf{x}}\right)^T = \begin{bmatrix} \frac{\partial F}{\partial x_1} \\ \frac{\partial F}{\partial x_2} \\ \vdots \\ \frac{\partial F}{\partial x_n} \end{bmatrix}, \mathbf{G} = \begin{bmatrix} \frac{\partial^2 F}{\partial x_1^2} & \cdots & \frac{\partial^2 F}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 F}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 F}{\partial x_n^2} \end{bmatrix} \tag{3}$$

$\mathbf{g}(\mathbf{x}^*) = 0$时$\mathbf{x}^*$处为驻点（驻点处还可能是最大值点或者鞍点）的充分必要条件，是此点处为最小值的必要条件如果$\mathbf{g}(\mathbf{x}^*) = 0$，由泰勒公式 (2),可以推出额外的条件：

$$F(\mathbf{x}^* + \Delta\mathbf{x}) \approx F(\mathbf{x}^*) + \frac{1}{2}\Delta\mathbf{x}^T\mathbf{G}(\mathbf{x}^*)\Delta\mathbf{x} + \cdots \tag{4}$$

对于任意$\Delta\mathbf{x} \neq 0$,要得到$F(\mathbf{x}^* + \Delta\mathbf{x}) \geq F(\mathbf{x}^*)$，充分条件是：$\mathbf{G}(x^*) > 0$;或者必要条件$\mathbf{G}(x^*) \geq 0$

结论：使得$\mathbf{x}^* = argmin_x F(\mathbf{x})$成立的：

充分条件为$\mathbf{g}(\mathbf{x}^*) = 0$且$\mathbf{G}(x^*) > 0$

必要条件为$\mathbf{g}(\mathbf{x}^*) = 0$且$\mathbf{G}(x^*) \geq 0$

迭代算法

从一个猜想的初始点$\mathbf{x}_0$开始，如 (5)持续迭代更新试图找到最小值点：

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \tag{5}$$

称 (5)中$\mathbf{p}_k$称为**搜索方向**，α_k称为**步长**

问题1：如何选择搜索方向$\mathbf{p}_k$？？

一阶导数方法（梯度下降）

$F(\mathbf{x})$在当前点处$\mathbf{x}_{k+1}$的泰勒展开为：

$$F_{k+1} \equiv F(\mathbf{x}_k + \alpha \mathbf{p}_k) \approx F(\mathbf{x}_k) + \frac{\partial F}{\partial \mathbf{x}}(\mathbf{x} + k + 1 - \mathbf{x}_k) = F_k + {\mathbf{g}_k}^T(\alpha_k \mathbf{p}_k) \tag{6}$$

假定$\alpha_k > 0$,要确保函数递减（$F_{k+1} < F_k$），需要：

$${\mathbf{g}_k}^T \mathbf{p}_k < 0 \tag{7}$$

因此，最陡下降的选择为：$\mathbf{p}_k = -\mathbf{g}_k$， (5)更新为：$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k$

二阶导数方法（牛顿法方向）

通常比一阶导数方法更快收敛到最小值点，假设F二次以上函数，

$\mathbf{g}_{k+1}$在$\mathbf{x}_{k+1}$处的泰勒展开为：

$$\mathbf{g}_{k+1} \equiv g(\mathbf{x}_k + \mathbf{p}_k) \approx \mathbf{g}_k + G_k(\mathbf{x}_{k+1} - \mathbf{x}_k) = \mathbf{g}_k + \mathbf{G}_k \mathbf{p}_k \tag{8}$$

想要$\mathbf{x}_{k+1}$处是最小值，需要$\mathbf{g}_{k+1} \approx 0$,因此：$\mathbf{p}_k = -{\mathbf{G}_k}^{-1}\mathbf{g}_k$

注意：对于复杂的$F(\mathbf{x})$，获得明确的一阶导数和二级导数可能是困难的，可用的办法有：有限差分技术（高代价）或者牛顿近似算法（如，$BFGS$）。另外，复杂$F(\boldsymbol{x})$的构建方法，如：CNN；微分计算需要借助自动微分软件，如：$Pytorch$，JAX等，后续再做介绍

介于一阶二阶之间的折中选择（共轭梯度方法）[参考第14页起内容]

问题2：如何选择步长α_k？？

只讲一个最常用的线性搜索法，**线性搜索**的目标是在$\mathbf{x}_k$，$\mathbf{p}_k$已知的情况下，搜索α_k使得：

$$\min_{\alpha_k \in \mathbb{R}} F(\mathbf{x}_k + \alpha_k \mathbf{p}_k) \tag{9}$$

这里介绍一种简单和广泛被采用的一种线性搜索方法，叫做：**回溯算法**（Backtracking Algorithm）

回溯算法及分析

【初始化】：选择$\gamma \in (0,1)$,$c \in (0,1)$

第一步**【计算回溯步长】**：

$\alpha_k{}^\star = max\gamma^\nu$

约束条件为：$\nu \in \{0,1,2,3\dots\}$

$$F(\mathbf{x}_k + \gamma^\nu \mathbf{p}_k) \leq F(\mathbf{x}_k) + c\gamma^\nu g(\mathbf{x}_k)\mathbf{p}_k \tag{10}$$

第二步**【迭代更新】**：

$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k^\star \mathbf{p}_k$

称 (10)为Armijo-Goldstein不等式，该不等式能保证以下不等式成立：

$$F(\mathbf{x}_k + \alpha_k{}^\star \mathbf{p}_k) < F(\mathbf{x}_k) \tag{11}$$

以下是算法解释分析：

搜索方向$\mathbf{p}_k$必须满足$g(\mathbf{x}_k)\mathbf{p}_k < 0$，任意满足该条件的方向被称作F在$\mathbf{x}_k$处的**严格递减方向**,如果$g(\mathbf{x}_k) \neq 0$，则**严格递减方向**总是存在例如，参考前面介绍的选择$\mathbf{p}_k$的方法，选择$\mathbf{p}_k = -\mathbf{g}_k$:

$$g(\mathbf{x}_k)\mathbf{p}_k = -\|\mathbf{g}_k{}^2\| \tag{12}$$

我们注意到，如果$\mathbf{p}_k$在$\mathbf{x}_k$处是**严格递减方向**，则一定存在$\alpha > 0$，使得$F(\mathbf{x}_k + \alpha\mathbf{p}_k) < F(\mathbf{x}_k), \forall \alpha_k \in (0,\alpha)$。证明如下：

$$g(\mathbf{x}_k) = \lim_{\alpha \to 0} \frac{F(\mathbf{x}_k + \alpha\mathbf{p}_k) - F(\mathbf{x}_k)}{\alpha\mathbf{p}_k}, \forall \alpha_k \in (0,\alpha) \tag{13}$$

进行简单变换：

$$g(\mathbf{x}_k)\alpha\mathbf{p}_k = \lim_{\alpha \to 0} F(\mathbf{x}_k + \alpha\mathbf{p}_k) - F(\mathbf{x}_k), \forall \alpha_k \in (0,\alpha) \tag{14}$$

由于$g(\mathbf{x}_k)\alpha\mathbf{p}_k < 0$,因此，$F(\mathbf{x}_k + \alpha\mathbf{p}_k) < F(\mathbf{x}_k), \forall \alpha_k \in (0,\alpha)$

Armijo-Goldstein不等式被认为是充分递减，很关键的一点是不要选择太小的$\alpha_k{}^\star$，这也是选择γ^ν第一个数字成员的原因，总之，我们总是选择尽可能大的$\alpha_k{}^\star$。通常选择$\gamma \in [0.5, 0.8]$、$c \in [0.001, 0.1]$来让$\alpha_k{}^\star$尽可能大的同时尽可能少在不同点的评估函数值

回溯算法中，有关搜索方向的选择知识扩展

1，使用梯度下降方向：$\mathbf{p}_k = \frac{-\mathbf{g}_k}{\|\mathbf{g}_k{}^2\|}$，注意：步长选择主要对该方法而言，如下牛顿和类牛顿方法中步长通常固定为1

在$\mathbf{g}_k \neq 0$时，该方向可保证$g(\mathbf{x}_k)\mathbf{p}_k < 0$，因此 (10)总是成立

2，使用牛顿方法方向：$\mathbf{p}_k = -{G_k}^{-1}\mathbf{g}_k$

3，使用类牛顿方法方向：$\mathbf{p}_k = -\mathbf{H}\mathbf{g}_k$,其中$\mathbf{H}$是对${G_k}^{-1}$的某种近似

对于牛顿和类牛顿方法方向，$\mathbf{g}(\mathbf{x}_k)\mathbf{p}_k = -\mathbf{g}_k{G_k}^{-1}\mathbf{g}_k = -\mathbf{g}_k\mathbf{H}\mathbf{g}_k$，(10)不总是成立，当且仅当$\mathbf{H}$为正定矩阵式才成立

在回溯算法中，采用梯度下降方向提供一个严格递减的迭代方向，这很好；不过当迭代进入接近驻点是，该算法需要很长一段时间（在这个区域通常会是很小的步长，数据错误和混乱不断出现和迭代）来获得一个有关驻点的中等精确度估计值。另一方面，牛顿方法方向直到非常接近驻点前可能不能保证严格递减方向，但在驻点附近时二阶导数充分条件通常就会出现和满足，因此，一旦接近驻点附近，切换采用牛顿法方向将可实现很快向驻点收敛。在实践中，可以利用综合考量利用以达到效果

回溯算法计算回溯步长很容易编程实现，以下是一段伪代码可做参考：

（1）$F_k = F(\mathbf{x}_k)$,$\mathbf{g}_k = F^{'}(\mathbf{x}_k)$

（2）$\Delta F = c\mathbf{g}_k(-\mathbf{g}_k)$,这里选择搜索方向为$-\mathbf{g}_k$

（3）令$\alpha_k = 1$，$F_{new} = F(\mathbf{x}_k + (-\mathbf{g}_k))$

（4）while $F_{new} > F_k + \alpha_k\Delta F$

$\alpha_k = \gamma\alpha_k$

$F_{new} = F(\mathbf{x}_k + \alpha_k(-\mathbf{g}_k))$

（5）end

Wolfe条件(Wolfe Conditions)

相对于基本的线性回溯算法，Wolfe条件试图获得更精确的线性搜索步长α_k值。考虑选择步长α_k，使其满足：

$$F(\mathbf{x}_k + \alpha_k\mathbf{p}_k) = \min_{\alpha_k \in \mathbb{R}} F(\mathbf{x}_k + \alpha\mathbf{p}_k) \tag{15}$$

这个情况下，由一阶导数条件知道：$\mathbf{g}_{k+1} = \mathbf{g}^T_{\mathbf{x}_k+\alpha_k\mathbf{p}_k}\mathbf{p}_k = 0$ Wolfe条件试图联合 Armijo-Goldstein充分递减条件且试图满足$\mathbf{g}^T_{\mathbf{x}_k+\alpha_k\mathbf{p}_k}\mathbf{p}_k \approx 0$的条件，以得到更加精确的步长，以下是Wolfe条件的具体形式描述

弱Wolfe条件（Weak Wolfe Conditions）：

$$F(\mathbf{x}_k + \alpha_k\mathbf{p}_k) \leq F(\mathbf{x}_k) + c_1\alpha_k\mathbf{g}(\mathbf{x}_k)\mathbf{p}_k \tag{16}$$

$$c_2\mathbf{g}(\mathbf{x}_k)\mathbf{p}_k \leq \mathbf{g}(\mathbf{x}_k + \alpha_k\mathbf{p}_k)\mathbf{p}_k \tag{17}$$

其中，$0 < c_1 < c_2 < 1$

弱Wolfe条件试图使得搜索方向$\mathbf{p}_k$在$\mathbf{x}_{k+1}$处不再是该点处梯度下降的方向（甚至可能是梯度升的方向）

强Wolfe条件（Strong Wolfe Conditions）：

$$F(\mathbf{x}_k + \alpha_k \mathbf{p}_k) \leq F(\mathbf{x}_k) + c_1 \alpha_k \mathbf{g}(\mathbf{x}_k)\mathbf{p}_k \tag{18}$$

$$|\mathbf{g}(\mathbf{x}_k + \alpha_k \mathbf{p}_k)\mathbf{p}_k| \leq c_2 |\mathbf{g}(\mathbf{x}_k)\mathbf{p}_k| \tag{19}$$

其中，$0 < c_1 < c_2 < 1$

强Wolfe条件试图使得搜索方向$\mathbf{p}_k$在$\mathbf{x}_{k+1}$处的一阶倒数接近0

将 Wolfe 条件中的一个或另一个附加于线搜索过程已成为基于线搜索方法的优化的标准做法

下面简要描述一个求解弱Wolfe条件的算法：二分算法

【初始化】：选择$0 < c_1 < c_2 < 1$,令$\alpha = 0$，$\alpha_k = 1$，$\beta = +\infty$

【循环】：

if $F(\mathbf{x}_k + \alpha_k \mathbf{p}_k) > F(\mathbf{x}_k) + c_1 \alpha_k \mathbf{g}_k \mathbf{p}_k$

set $\beta = \alpha_k$ and reset $\alpha_k = \frac{1}{2}(\alpha + \beta)$

else if $\mathbf{g}(\mathbf{x}_k + \alpha_k \mathbf{p}_k)\mathbf{p}_k < c_2 \mathbf{g}(\mathbf{x}_k)\mathbf{p}_k$

set $\alpha = \alpha_k$ and reset

$$\alpha_k = \begin{cases} 2\alpha, if \beta = +\infty \\ \frac{1}{2}(\alpha + \beta), otherwise \end{cases}$$

Else, Stop

【停止循环】

估算类牛顿方法的Hessians矩阵方法

上接第3页 (8)，接下来介绍计算$\mathbf{G}_k$或者$\mathbf{G}_k{}^{-1}$【对应矩阵称为Hessian矩阵或逆矩阵】的方法
使用 (5)，取$\alpha_k = 1$代入$\mathbf{p}_k = -\mathbf{G}_k{}^{-1}\mathbf{g}_k$，得到：

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{G}_k{}^{-1}\mathbf{g}_k \tag{20}$$

对 (20)进行进行简单变换得到：

$$\mathbf{G}_k(\mathbf{x}_{k+1} - \mathbf{x}_k) = -\mathbf{g}_k \tag{21}$$

另外，根据切线定义导数的原理，有以下切线方程成立：

$$\mathbf{G}_{k+1}(\mathbf{x}_{k+1} - \mathbf{x}_k) \approx \mathbf{g}_{k+1} - \mathbf{g}_k \tag{22}$$

用 (22)减去 (21)，得到：

$$(\mathbf{G}_{k+1} - \mathbf{G}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k) = \mathbf{g}_{k+1} \tag{23}$$

在每次迭代中，给出$\mathbf{x}_k$,$\mathbf{G}_k$利用 (21)计算得到$\mathbf{x}_{k+1}$,再利用 (23)可计算得到$\mathbf{G}_{k+1}$。 (23)等价于n个线性方程，但$\mathbf{G}_{k+1}$是包含n^2个未知数的矩阵，求解上式并不能得到唯一的$\mathbf{G}_{k+1}$，为了确定$\mathbf{G}_{k+1}$，需要提供更多的条件，那么怎样的条件合理呢？
下面做简要介绍:三种结果等价的方法供参考
第一种
由于$\mathbf{x}_{k+1}$与$\mathbf{x}_k$比较接近，$\mathbf{G}_{k+1}$也应当“接近”$\mathbf{G}_k$。如何定义和衡量“接近”，从计算的角度而言，“接近”意味着容易表达和计算，下面具体化“接近”为：$\mathbf{G}_{k+1}$为$\mathbf{G}_k$的秩1增量变化，再考虑到$\mathbf{G}_{k+1}$和$\mathbf{G}_k$是二阶导数矩阵Hessian矩阵的近似，应当是对称正定矩阵；因此，使用以下秩1增量对称矩阵更新等式：

$$\mathbf{G}_{k+1} = \mathbf{G}_k + \alpha\mathbf{u}\mathbf{u}^T \tag{24}$$

下面来分析确定合理的$\mathbf{u}$取值，以得到$\mathbf{G}_{k+1}$的估算公式
记$\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$，$\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$，结合 (22)和 (24)则有

$$\mathbf{y}_k = \mathbf{G}_{k+1}\mathbf{s}_k = \mathbf{G}_k\mathbf{s}_k + \alpha\mathbf{u}\mathbf{u}^T\mathbf{s}_k = \mathbf{G}_k\mathbf{s}_k + (\alpha\mathbf{u}^T\mathbf{s}_k)\mathbf{u} \tag{25}$$

如果$\mathbf{u}^T\mathbf{s}_k \neq 0$,可知$\mathbf{u} = \beta(\mathbf{y}_k - \mathbf{G}_k\mathbf{s}_k)$，其中$\beta \in \mathbb{R}$,取$\mathbf{u} = (\mathbf{y}_k - \mathbf{G}_k\mathbf{s}_k)$，可以计算出$\alpha = 1/(\mathbf{y}_k - \mathbf{G}_k\mathbf{s}_k)$，回带入 (24)得到：

$$\mathbf{G}_{k+1} = \mathbf{G}_k + \frac{(\mathbf{y}_k - \mathbf{G}_k\mathbf{s}_k)(\mathbf{y}_k - \mathbf{G}_k\mathbf{s}_k)^T}{(\mathbf{y}_k - \mathbf{G}_k\mathbf{s}_k)^T\mathbf{s}_k} \tag{26}$$

(26)称为**对称秩1更新公式（SR1 Update）**，该算法简单可行，但缺点是不能保证和保留正定性特性

类似的，快速介绍一下秩2增量对称矩阵更新方法：

$$\mathbf{G}_{k+1} = \mathbf{G}_k + \alpha\mathbf{u}\mathbf{u}^T + \beta\mathbf{v}\mathbf{v}^T \tag{27}$$

下面来分析确定合理的$\mathbf{u}$，$\mathbf{v}$取值，得到$\mathbf{G}_{k+1}$的估算公式

记$\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$，$\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$，结合 (22)和 (27)则有

$$\mathbf{y}_k = \mathbf{G}_{k+1}\mathbf{s}_k = \mathbf{G}_k\mathbf{s}_k + \alpha\mathbf{u}\mathbf{u}^T\mathbf{s}_k + \beta\mathbf{v}\mathbf{v}^T\mathbf{s}_k = \mathbf{G}_k\mathbf{s}_k + (\alpha\mathbf{u}^T\mathbf{s}_k)\mathbf{u} + (\beta\mathbf{v}^T\mathbf{s}_k)\mathbf{v} \tag{28}$$

令$\mathbf{u} = \mathbf{y}_k$，$\mathbf{v} = \mathbf{G}_k\mathbf{s}_k$，解出$\alpha$，$\beta$回代入 (27)，得到：

$$\mathbf{G}_{k+1} = \mathbf{G}_k + \frac{\mathbf{y}_k{\mathbf{y}_k}^T}{{\mathbf{y}_k}^T\mathbf{s}_k} - \frac{\mathbf{G}_k\mathbf{s}_k{\mathbf{s}_k}^T\mathbf{G}_k}{{\mathbf{s}_k}^T\mathbf{G}_k\mathbf{s}_k} \tag{29}$$

(29)称为**BFGS更新公式（Broyden-Fletcher-Goldfarb-Shanno (BFGS) update）** Sherman-Morrison-Woodbury公式:

$$(\mathbf{A} + \mathbf{UCV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{C}^{-1} + \mathbf{VA}^{-1}\mathbf{U})^{-1}\mathbf{VA}^{-1} \tag{30}$$

对 (29)运用 (30),得到：

$${\mathbf{G}_{k+1}}^{-1} = (\mathbf{I} - \rho_k\mathbf{s}_k{\mathbf{y}_k}^T){\mathbf{G}_k}^{-1}(\mathbf{I} - \rho_k\mathbf{y}_k{\mathbf{s}_k}^T) + \rho_k\mathbf{s}_k{\mathbf{s}_k}^T \tag{31}$$

其中，$\rho_k = \frac{1}{{\mathbf{y}_k}^T\mathbf{s}_k}$ (31)称为**BFGS逆更新公式**

重要的是，BFGS更新公式能够保持正定性：

$$\mathbf{x}^T{\mathbf{G}_{k+1}}^{-1}\mathbf{x} = (\mathbf{x} - \rho_k\mathbf{s}_k^T\mathbf{x}\mathbf{y}_k){\mathbf{G}_k}^{-1}(\mathbf{x} - \rho_k{\mathbf{s}_k}^T\mathbf{x}\mathbf{y}_k) + \rho_k({\mathbf{s}_k}^T\mathbf{x})^2 \tag{32}$$

上式两个组成部分均为非负，第二部分仅当${\mathbf{s}_k}^T\mathbf{x} = 0$时才为0，在第二个部分为0时，第一部分也为0需要$\mathbf{x} = 0$

接下来，简单提下Davidon-Fletcher-Powell (DFP) 更新法：

对${\mathbf{G}_{k+1}}^{-1}$和${\mathbf{G}_k}^{-1}$，而不是$\mathbf{G}_{k+1}$和$\mathbf{G}_k$直接应用SR2更新方法，即

$${\mathbf{G}_{k+1}}^{-1} = {\mathbf{G}_k}^{-1} + \alpha\mathbf{u}\mathbf{u}^T + \beta\mathbf{v}\mathbf{v}^T \tag{33}$$

切线等式变为：${\mathbf{G}_k}^{-1}\mathbf{y}_k = \mathbf{s}_k$，求解$\alpha$，$\beta$回代得到：

$${\mathbf{G}_{k+1}}^{-1} = {\mathbf{G}_k}^{-1} + \frac{\mathbf{s}_k{\mathbf{s}_k}^T}{{\mathbf{y}_k}^T\mathbf{s}_k} - \frac{{\mathbf{G}_k}^{-1}\mathbf{y}_k{\mathbf{y}_k}^T{\mathbf{G}_k}^{-1}}{{\mathbf{y}_k}^T{\mathbf{G}_k}^{-1}\mathbf{y}_k} \tag{34}$$

对上式，借助Sherman-Morrison-Woodbury公式，可得到:

$$\mathbf{G}_{k+1} = (\mathbf{I} - \rho_k \mathbf{y}_k {\mathbf{s}_k}^T)\mathbf{G}_k(\mathbf{I} - \rho_k \mathbf{s}_k {\mathbf{y}_k}^T) + \rho_k \mathbf{y}_k {\mathbf{y}_k}^T \tag{35}$$

其中，$\rho_k = \frac{1}{{\mathbf{y}_k}^T \mathbf{s}_k}$ DFP更新算法同样可以保留正定性，计算复杂的也和BFGS一样，不过没有BFGS流行

最好，介绍下曲率条件（Curvature condition）：由切线等式：$\mathbf{G}_{k+1}\mathbf{s}_k = \mathbf{y}_k$，暗含：

$$ {\mathbf{y}_k}^T \mathbf{s}_k = {\mathbf{s}_k}^T \mathbf{G}_{k+1} \mathbf{s}_k > 0 \tag{36}$$

因此，${\mathbf{y}_k}^T \mathbf{s}_k > 0$时，一定存在$\mathbf{G}_{k+1}$，使得：$\mathbf{G}_{k+1}\mathbf{s}_k = \mathbf{y}_k$

第二种

下面考虑在忽略对称性但仍考虑秩1更新的情况（BGFS更新公式的第二种推到方法）：

$$\mathbf{G}_{k+1} = \mathbf{G}_k + \mathbf{u}\mathbf{v}^T \tag{37}$$

来分析合理的$\mathbf{u}$,$\mathbf{v}$取值，得到$\mathbf{G}_{k+1}$的估算方法
记$\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$，$\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$，结合 (22)和 (37)则有

$$\mathbf{y}_k = \mathbf{G}_{k+1}\mathbf{s}_k = \mathbf{G}_k\mathbf{s}_k + \mathbf{u}\mathbf{v}^T\mathbf{s}_k \tag{38}$$

如果$\mathbf{v}^T\mathbf{s}_k \neq 0$,得到$\mathbf{u} = (\mathbf{y}_k - \mathbf{G}_k\mathbf{s}_k)(\mathbf{v}^T\mathbf{s}_k)^{-1}$,回带入 (37)得到：

$$\mathbf{G}_{k+1} = \mathbf{G}_k + (\mathbf{y}_k - \mathbf{G}_k\mathbf{s}_k)(\mathbf{v}^T\mathbf{s}_k)^{-1}\mathbf{v}^T \tag{39}$$

只要$\mathbf{s}_k \neq 0$,选择$\mathbf{v} = \mathbf{s}_k$, (39)更新为：

$$\mathbf{G}_{k+1} = \mathbf{G}_k + (\mathbf{y}_k - \mathbf{G}_k\mathbf{s}_k)(\mathbf{s}_k^T\mathbf{s}_k)^{-1}\mathbf{s}_k^T \tag{40}$$

(40)就是**Broyden更新公式**，现在需要把该算法和求解最优化问题结合起来看，这就需要$\mathbf{G}_{k+1}$和$\mathbf{G}_k$满足：1.对称性，2.正定性
现在把对称和正定性纳入进来，考虑如下问题：已知对称正定矩阵$\mathbf{G}_k$和两个向量$\mathbf{s}_k$，$\mathbf{y}_k$，寻找一个对称正定矩阵$\mathbf{G}_{k+1}$，该矩阵需满足$\mathbf{G}_{k+1}\mathbf{s}_k = \mathbf{y}_k$条件。由于$\mathbf{G}_k$是对称正定矩阵，因此存在一个非奇异值$nxn$矩阵$\mathbf{L}_k$，使得$\mathbf{G}_k = \mathbf{L}_k{\mathbf{L}_k}^T$（例如，$\mathbf{L}_k$可以是$\mathbf{G}_k$的下三角Cholesky factor），类似的存在矩阵$\mathbf{J}_{k+1}$使得$\mathbf{G}_{k+1} = \mathbf{J}_{k+1}{\mathbf{J}_{k+1}}^T$。
令：

$$\mathbf{v}_k = {\mathbf{J}_{k+1}}^T\mathbf{s}_k \tag{41}$$

则：

$$\mathbf{J}_{k+1}\mathbf{v}_{kl} = \mathbf{J}_{k+1}{\mathbf{J}_{k+1}}^T\mathbf{s}_k = \mathbf{G}_{k+1}\mathbf{s}_k = \mathbf{y}_k \tag{42}$$

对 (42)应用 (40)Broyden算法，得到：

$$\mathbf{J}_{k+1} = \mathbf{L}_k + \frac{(\mathbf{y}_k - \mathbf{L}_k\mathbf{v}_k){\mathbf{v}_k}^T}{{\mathbf{v}_k}^T\mathbf{v}_k} \tag{43}$$

将 (43)带入 (41),得到：

$$\mathbf{v}_k = {\mathbf{J}_{k+1}}^T\mathbf{s}_k = {\mathbf{L}_k}^T\mathbf{s}_k + \frac{\mathbf{v}_k(\mathbf{y}_k - \mathbf{L}_k\mathbf{v}_k)^T\mathbf{s}_k}{{\mathbf{v}_k}^T\mathbf{v}_k} \tag{44}$$

由上式得知：$\mathbf{v}_k = \alpha {\mathbf{L}_k}^T \mathbf{s}_k$，其中$\alpha \in \mathbb{R}$，回代入 (44)得到：

$$\alpha = [\frac{{\mathbf{s}_k}^T \mathbf{y}_k}{{\mathbf{s}_k}^T \mathbf{G}_k \mathbf{s}_k}]^{\frac{1}{2}} \tag{45}$$

仅当${\mathbf{s}_k}^T \mathbf{y}_k > 0$时，有以下$\mathbf{J}_{k+1}$成立：

$$\mathbf{J}_{k+1} = \mathbf{L}_k + \frac{(\mathbf{y}_k - \alpha \mathbf{G}_k \mathbf{s}_k) {\mathbf{s}_k}^T \mathbf{L}_k}{\alpha {\mathbf{s}_k}^T \mathbf{G}_k \mathbf{s}_k} \tag{46}$$

因此:

$$\mathbf{G}_{k+1} = \mathbf{G}_k + \frac{\mathbf{y}_k {\mathbf{y}_k}^T}{{\mathbf{y}_k}^T \mathbf{s}_k} - \frac{\mathbf{G}_k \mathbf{s}_k {\mathbf{s}_k}^T \mathbf{G}_k}{{\mathbf{s}_k}^T \mathbf{G}_k \mathbf{s}_k} \tag{47}$$

对 (47)运用Sherman-Morrison-Woodbury得到逆的更新公式为：

$${\mathbf{G}_{k+1}}^{-1} = (\mathbf{I} - \rho_k \mathbf{s}_k {\mathbf{y}_k}^T) {\mathbf{G}_k}^{-1} (\mathbf{I} - \rho_k \mathbf{y}_k {\mathbf{s}_k}^T) + \rho_k \mathbf{s}_k {\mathbf{s}_k}^T \tag{48}$$

其中，$\rho_k = \frac{1}{{\mathbf{y}_k}^T \mathbf{s}_k}$ (47)称为BFGS更新公式， (48)称为逆BFGS更新公式。

第三种

另外，（BGFS更新公式的第三种推到方法），基于 (23)，选择$\mathbf{G}_{k+1}$让其充分接近$\mathbf{G}_k$，得到以下一般形式的优化问题描述：

$$minimize_{\mathbf{G}_{k+1}} = \|\mathbf{G}_{k+1} - \mathbf{G}_k\|$$

约束条件为：$\mathbf{G}_{k+1} = \mathbf{G}_{k+1}^T$ 且 $\mathbf{G}_{k+1}\mathbf{s}_k = \mathbf{y}_k$,这里的范式$\|\mathbf{.}\|$可以有很多种选择，不同的范式选择会得到不同的更新公式和算法。比如，使用Weighted Frobenius norm（具体过程省略）得到的BFGS更新公式为：

$$\mathbf{G}_{k+1}{}^{-1} = (\mathbf{I} - \rho_k \mathbf{s}_k \mathbf{y}_k{}^T)\mathbf{G}_k{}^{-1}(\mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k{}^T) + \rho_k \mathbf{s}_k \mathbf{s}_k{}^T \tag{49}$$

其中，$\rho_k = \frac{1}{\mathbf{y}_k{}^T \mathbf{s}_k}$ 对 (49)运用Sherman-Morrison-Woodbury公式，得到逆的更新公式为：

$$\mathbf{G}_{k+1} = \mathbf{G}_k - \frac{1}{\mathbf{s}_k{}^T \mathbf{G}_k \mathbf{s}_k} \mathbf{G}_k \mathbf{s}_k \mathbf{s}_k{}^T \mathbf{G}_k + \rho_k \mathbf{y}_k \mathbf{y}_k{}^T \tag{50}$$

细心可注意到 (50)和 (47)是一样的。

简要对L-BFGS做介绍如下：

对于大型高维度空间优化问题，甚至存储Hessian矩阵都会面临挑战，因此，希望只通过存储一些向量，就能维持一个对Hessian矩阵的近似成为一个现实中不错做法，该方法被称为有限存储（Limited-memory）BFGS (L-BFGS)方法，主要的思路如下：

$$\mathbf{G}_{k+1}{}^{-1} = \mathbf{V}_k{}^T \mathbf{G}_k{}^{-1} \mathbf{V}_k + \rho_k \mathbf{s}_k \mathbf{s}_k{}^T \tag{51}$$

其中,$\mathbf{V}_k = \mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k{}^T$，通过存储一定数量（例如：5个）最新的向量对$(\mathbf{s}_k, \mathbf{y}_k)$隐式的来替代存储$\mathbf{G}_k$,也就是说L-BFGS如下式，仅维护一定数量（例如：m个）最新的向量对：

$$\begin{aligned}
\mathbf{G}_{k+1}{}^{-1} = & \mathbf{V}_k{}^T \mathbf{V}_{k-1}{}^T \mathbf{V}_2{}^T \cdots \mathbf{V}_{k-m}{}^T \mathbf{G}_{k-m}{}^{-1} \mathbf{V}_{k-m} \cdots \mathbf{V}_{k-2} \mathbf{V}_{k-1} \mathbf{V}_k \\
& + \rho_{k-m} \mathbf{V}_k{}^T \mathbf{V}_{k-1}{}^T \mathbf{V}_2{}^T \cdots \mathbf{V}_{k-m}{}^T \mathbf{s}_{k-m} \mathbf{s}_{k-m}{}^T \mathbf{V}_{k-m} \cdots \mathbf{V}_{k-2} \mathbf{V}_{k-1} \mathbf{V}_k \\
& + \rho_{k-m+1} \mathbf{V}_k{}^T \mathbf{V}_{k-1}{}^T \mathbf{V}_2{}^T \cdots \mathbf{V}_{k-m+1}{}^T \mathbf{s}_{k-m+1} \mathbf{s}_{k-m+1}{}^T \mathbf{V}_{k-m+1} \cdots \mathbf{V}_{k-2} \mathbf{V}_{k-1} \mathbf{V}_k \\
& + \cdots \\
& + \rho_k \mathbf{s}_k \mathbf{s}_k{}^T
\end{aligned} \tag{52}$$

$\mathbf{G}_{k-m}{}^{-1} = \mathbf{I}$是一个流行的选择，不过也有存在其他更加合理的选择

共轭梯度算法（Conjugate gradient methods）

上接第3页 (5)，在共轭梯度算法中，搜索方向$\mathbf{p}_k$的产生规则如下：

$$\mathbf{p}_{k+1} = -\mathbf{g}_{k+1} + \beta_k \mathbf{p}_k, \mathbf{p}_0 = -\mathbf{g}_0 \tag{53}$$

这里及以下用CG简称代表共轭梯度算法，β_k称作迭代参数；不同的标量β_k选择对应不同的CG算法

用$\|.\|$表示欧几里德范数，记$\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$，下表提供了不同算法的更新参数选择：

${\beta_k}^{HS} = \frac{\mathbf{g}_{k+1}^T \mathbf{y}_k}{\mathbf{d}_k^T \mathbf{y}_k}$	(1952)	自原始(线性)CG论文，作者：Hestenes and Stiefel
${\beta_k}^{FR} = \frac{\|\mathbf{g}_{k+1}^2\|}{\|\mathbf{g}_k^2\|}$	(1964)	第一个非线性CG方法，作者：Fletcher and Reeves
${\beta_k}^{D} = \frac{\mathbf{g}_{k+1}^T \nabla^2 F(\mathbf{x}_k) \mathbf{d}_k}{\mathbf{d}_k^T \nabla^2 F(\mathbf{x}_k) \mathbf{d}_k}$	(1967)	需要Hessian矩阵，作者：Daniel
${\beta_k}^{PRP} = \frac{\mathbf{g}_{k+1}^T \mathbf{y}_k}{\|\mathbf{g}_k^2\|}$	(1969)	作者：Polak and Ribi‘ere and by Polyak
${\beta_k}^{CD} = \frac{\|\mathbf{g}_{k+1}^2\|}{-\mathbf{d}_k^T \mathbf{g}_k}$	(1987)	CD代表:“Conjugate Descent”，作者：Fletcher
${\beta_k}^{LS} = \frac{\mathbf{g}_{k+1}^T \mathbf{y}_k}{-\mathbf{d}_k^T \mathbf{g}_k}$	(1991)	作者：Liu and Storey
${\beta_k}^{DY} = \frac{\|\mathbf{g}_{k+1}^2\|}{\mathbf{d}_k^T \mathbf{y}_k}$	(1999)	作者： Dai and Yuan
${\beta_k}^{N} = (\mathbf{y}_k - 2\mathbf{d}_k \frac{\|\mathbf{y}_k^2\|}{\mathbf{d}_k^T \mathbf{y}_k}) \frac{\mathbf{g}_{k+1}}{\mathbf{d}_k^T \mathbf{y}_k}$	(2005)	作者： Hager and Zhang

上表中，除了最后一个和第三个，更新参数 β_k的分子有2种形式（$\|\mathbf{g}_{k+1}^2\|$或者$\mathbf{g}_{k+1}^T \mathbf{y}_k$），其分母有3种形式（$\|\mathbf{g}_k^2\|$，$\mathbf{d}_k^T \mathbf{y}_k$或者$-\mathbf{d}_k^T \mathbf{g}_k$），共计6种不同的组合

第一类：$\|\mathbf{g}_{k+1}^2\|$在计算 β_k的分母中,涉及表格中FR，DY和CD方法：强收敛，但容易受干扰而失效（即算法可能采用很多小步长而在迫近最小值上毫无进展）

第二类：$\mathbf{g}_{k+1}^T \mathbf{y}_k$在计算 β_k的分母中,涉及表格中HS，PRP和LS方法：具有内置重启功的具有解决干扰问题[当$\mathbf{x}_{k+1} - \mathbf{x}_k$很小时，$\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$也很小，让$\beta_k$趋近于0，因此，$\beta_k$很小，新的搜索方向$\mathbf{d}_{k+1}$使用最陡峭梯度下降方向$-\mathbf{g}_{k+1}$。这样，HS，PRP和LS方法通过自动调整$\beta_k$来避免干扰],但这些方法通常不收敛

混合CG方法和参数家族

基于以上两类方法的特点，混合利用以上两类的优点，例如：

$\beta_k = max\{0, min\{{\beta_k}^{PRP}, {\beta_k}^{FR}\}\}$

表格中最后一个方法能够确保充分下降且与线性搜索的精确度条件相互独立，在实际使用中又很好的效果
另外，CG方法在整个迭代中，只需要存储2个（例如，FR方法）或者3个（如，PRP）向量，当问题规模n很大时，这是非常大的优势。

了解以上理论和算法后，我们要问：实际问题中$F(\mathrm{x})$具体形式是怎样的，该如何构建，其倒数又该如何计算，使用什么工具？请看下篇