最小二乘问题(Least-Squares Problems)

考虑一个实际问题，如果我们有实验数据$(t_j，y_j)$，这里t_j可以是标量或向量，y_j一般认为是标量，$j=1,2,\cdots,m$。理论和经验告诉我们，这组实验数据应该符合规律

$$\phi(x;t),$$

其中，$x=(x_1,x_2,\cdots,x_n)\in\mathbb{R}^n$是待定参数。那么对于每一个数据，我们有

$$r_j(x)\overset{\text{def}}{=}\phi(x;t_j)-y_j$$

称为模型的残量，显然，为了得到所谓"最优"的模型，我们希望通过调整待定参数x来最小化残量。一种自然的做法是构建无约束优化问题，目标函数具有如下形式：

$$f(x)=\frac{1}{2}\sum_{j=1}^{m}r_j^2(x),\tag{10.1}$$

则相应优化问题称为最小二乘问题。其中r_j是$\mathbb{R}^n\to\mathbb{R}$的光滑函数。我们将$r_j$看作残量(residual)，并且通常假设$m>>n$。

最小二乘问题的背景应用是大家熟悉的，最典型的一个应用就是上面提到的数据回归或函数拟合。这里强调一下f取这种形式的优点在于我们可以将向量函数

$$r(x)=(r_1(x),r_2(x),\cdots,r_m(x))^T\tag{10.2}$$

看作一个$\mathbb{R}^n\to\mathbb{R}^m$的映射，则

$$f(x)=\frac{1}{2}\|r(x)\|_2^2.$$

且$r(x)$的Jacobi矩阵$J(x)$为

$$J(x)=\left[\frac{\partial r_j}{\partial x_i}\right]_{\substack{j=1,2,\cdots,m\\i=1,2,\cdots,n}}=\begin{bmatrix}\nabla r_1(x)^T\\\nabla r_2(x)^T\\\vdots\\\nabla r_m(x)^T\end{bmatrix},\tag{10.3}$$

其中$\nabla r_j(x),j=1,2,\cdots,m$是$r_j$的梯度。于是$f$的梯度和Hessian有如下形式：

$$\nabla f(x)=\sum_{j=1}^{m}r_j(x)\nabla r_j(x)=J(x)^Tr(x),\tag{10.4}$$

$$\begin{aligned}\nabla^2 f(x)&=\sum_{j=1}^{m}\nabla r_j(x)\nabla r_j(x)^T+\sum_{j=1}^{m}r_j(x)\nabla^2 r_j(x)\\&=J(x)^TJ(x)+\sum_{j=1}^{m}r_j(x)\nabla^2 r_j(x).\end{aligned}\tag{10.5}$$

在一般的计算中，计算$J(x)$的代价要远小于计算$\nabla^2 f(x)$。因此最小二乘法的一个核心思想就是尽可能只计算$J(x)$而不去直接计算$\nabla^2 f(x)$。比如在(10.5)中，只计算第一项，而舍弃第二项。当然这么做导致的后果是问题依赖的，也需要仔细分析。

背景

最小二乘的一个主要应用是数据拟合。

例10.1 我们考虑一个药物对病人的某个可测量结果$\phi(x;t)$和五个指定的用药参数$x=(x_1,x_2,x_3,x_4,x_5)$（用量、浓度等等...）以及服用时间之间的关系。经验公式告诉我们这些参数和疗效之间的关系为：

$$\phi(x;t)=x_1+tx_2+t^2x_3+x_4e^{-x_5t}. \tag{10.6}$$

现对一个指定药品和m个病人，我们做了一组实验并得到了实际的结果数据$y_j,j=1,2,\cdots,m$。那么我们就可以利用最小二乘法去最小化目标函数：

$$\frac{1}{2}\sum_{j=1}^{m}\left[\phi(x;t_j)-y_j\right]^2, \tag{10.7}$$

从而得到目标公式的最优参数x^*。这个其实是统计学上定点回归(fixed-regressor)的一个特例。这里平方的本质是求模型结果$\phi(x)$和实测结果y之差

$$r_j(x)=\phi(x;t)-y_j, \tag{10.8}$$

的l_2范数（的平方），它使得目标函数更加光滑，但是也使得误差的效应扩散。也有回归模型讨论l_∞范数或l_1范数，但此时目标函数相应变为

$$\max_{j=1,2,\cdots,m}\left|\phi(x;t_j)-y_j\right|, \tag{10.9}$$

和

$$\sum_{j=1}^{m}\left|\phi(x;t_j)-y_j\right|, \tag{10.10}$$

失去了光滑性，给计算带来额外的困难。本章节我们只讨论l_2范数的情况。

线性最小二乘问题

若$\phi(x)$是一个关于x的线性函数，则对应的最小二乘问题称为线性最小二乘问题。此时残量函数(10.8)也是线性的，我们可以将其写作

$$r(x)=Jx-y,$$

其中J是矩阵，y是向量，且都与x无关。所以目标函数f简化为

$$f(x)=\frac{1}{2}\|Jx-y\|^2, \tag{10.13}$$

其中$y=r(0)$。而相应地，

$$\nabla f(x)=J^T(Jx-y),\nabla^2 f(x)=J^TJ.$$

显然，此时f是凸的，因此我们只需要求它的稳定点$\nabla f(x^*)=0$即可找到它的全局最小值点x^*。也即，以下方程

$$J^TJx^*=J^Ty. \tag{10.14}$$

的解。这个方程被称为(10.3)的正规方程组(the normal equations)。

注：这里等价于一个超定方程组的广义解，或求长方阵J的广义逆。

和之前讨论的类似，对(10.14)主要有两条路线可以求解，其一是基于Cholesky分解：

- 计算系数矩阵J^TJ和右端项J^Ty；
- 计算J^TJ的Cholesky分解R^TR；（注：$R^T\neq J$，因此必须先计算J^TJ再通过Cholesky分解计算R。）

- 依次求解三角方程组：$R^T\tilde{y} = J^T y$和$Rx^* = \tilde{y}$得x^*。

这个方法很容易实现，但缺点也很明显。就是$J^T J$的条件数(condition number)是J的平方。因此相对原始问题(10.13)，直接求解(10.14)有可能会损失大量的浮点运算精度（数值不稳定）。当J是病态(ill conditioned)时，J^J甚至会接近数值不正定。

注：另一种理解是，J是一个$m \times n$的长方阵，但$J^T J$是$n \times n$的，这里丢失了信息的自由度，本质上是做了空间投影，引入了额外的投影误差。

第二个方法则采用QR分解。首先，我们回顾一下高等代数的结论：欧氏范数（l_2范数）在正交变换下是不变的。也即

$$\|Jx - y\| = \|Q^T(Jx - y)\|, \qquad (10.16)$$

其中，Q是$m \times m$阶正交阵。假设存在正交分解（具体算法参见：Gene H. Golub, C. F. Van Loan, Ch. 5, p. 136, Matrix Computations, Johns Hopkins University Press, 2012, Baltimore.）：

$$J\Pi = Q \begin{bmatrix} R \\ 0 \end{bmatrix} = [Q_1 Q_2] \begin{bmatrix} R \\ 0 \end{bmatrix} = Q_1 R, \quad (10.17)$$

其中

- Π是$n \times n$阶交换阵(permutation matrix)，交换阵必然是正交阵；
- Q是$m \times m$阶正交阵；
- Q_1是Q的头n列，而Q_2是后$m - n$列；
- R是$n \times n$阶上三角矩阵(upper triangular matrix)，并且对角元均为正。

结合(10.16)和(10.17)，我们有（正交阵性质$Q^T Q = I$，以及正交阵的算子作用不改变l_2范数。）

$$\begin{aligned} \|Jx - y\|_2^2 &= \left\| \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} \left(J\Pi\Pi^T x - y \right) \right\|_2^2 \\ &= \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} \left(\Pi^T x \right) - \begin{bmatrix} Q_1^T y \\ Q_2^T y \end{bmatrix} \right\|_2^2 \\ &= \left\| R\left(\Pi^T x \right) - Q_1^T y \right\|_2^2 + \left\| Q_2^T y \right\|_2^2. \quad (10.18) \end{aligned}$$

注意这里第二项和x无关，因此我们只需设法将第一项消成零，即

$$x^* = \Pi R^{-1} Q_1^T y.$$

这一步在实际计算中是先求解三角方程组

$$Rz = Q_1^T y,$$

然后

$$x^* = \Pi z.$$

这个方法实际的舍入误差是由J而不是$J^T J$的条件数决定的。

第三条道路，则是基于对J的奇异值分解(singular-value decomposition, SVD)。（具体算法参见Matrix Computations的第8章）对J做奇异值分解，有

$$J = U \begin{bmatrix} S \\ 0 \end{bmatrix} V^T = [U_1 U_2] \begin{bmatrix} S \\ 0 \end{bmatrix} V^T = U_1 S V^T, \quad (10.19)$$

其中

- U是$m \times m$阶正交阵；
- U_1是U的前n列，而U_2是后$m - n$列；

- V是$n \times n$阶正交阵；
- S是$n \times n$阶对角阵，其对角元$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n > 0$。

（这里

$$J^T J = V S^2 V^T,$$

所以V的第j列向量就是$J^T J$关于特征值$\sigma_j^2, j = 1, 2, \cdots, n$的特征向量。）

和第二种方法类似，我们有

$$\begin{aligned} \|Jx - y\|_2^2 &= \left\| \begin{bmatrix} S \\ 0 \end{bmatrix} \left(V^T x\right) - \begin{bmatrix} U_1^T \\ U_2^T \end{bmatrix} y \right\|_2^2 \\ &= \left\| S\left(V^T x\right) - U_1^T y \right\|_2^2 + \|U_2^T y\|_2^2. \quad (10.20) \end{aligned}$$

同样地，我们只需考虑第一项变零的情形，也即

$$x^* = V S^{-1} U_1^T y.$$

记U的第i列为$u_i \in \mathbb{R}^m$，V的第i列为$v_i \in \mathbb{R}^n$，则

$$x^* = \sum_{i=1}^{n} \frac{u_i^T y}{\sigma_i} v_i. \qquad (10.21)$$

尽管奇异值分解的代价较大，但它也提供了额外的信息。当σ_i较小时，x^*对$u_i^T y$的扰动是非常敏感的。特别J接近秩亏(rank-deficient)时，也即$\sigma_n / \sigma_1 << 1$时，为了获得敏感性信息而付出这个代价是值得的。

以上这三种方法各有各的适用场合。当$m >> n$时，或者我们只有$J^T J$信息，缺少J信息时，基于Cholesky分解的算法是合适的。然而，若$J^T J$的条件数不好时，我们必须做额外的预处理。基于QR分解的算法，用于条件数较小，具有更高的鲁棒性和稳定性。而基于SVD的算法具有最好的鲁棒性和数值稳定性（代价也最大）。当J接近秩亏时，某一些奇异值数值上就是零，而此时

$$x^* = \sum_{\sigma_i \neq 0} \frac{u_i^T y}{\sigma_i} v_i + \sum_{\sigma_i = 0} \tau_i v_i \qquad (10.22)$$

（τ_i可以是任意系数）仍然是(10.20)的最优解。我们通常将τ_i都设成零。甚至对某些特别小的σ_i，我们也可以直接将其当作零来处理。

注：具体采用何种策略在实际问题中主要取决于以下几个因素：

- J是否大规模稀疏；
- J和$J^T J$的存储量对比；
- J是否列满秩；
- 模型是否参数敏感...

并没有确定的判定原则，一切要从具体问题的分析入手。

非线性最小二乘问题

高斯-牛顿法(Gauss-Newton)

对于一般性问题(10.1)，我们仍然从其局部梯度(10.4)和Hessian(10.5)考虑设计算法。最直接的想法是应用线搜索Newton法的一个修正——Gauss-Newton法。这里标准的Newton法会求解Newton方程组：

$$\nabla^2 f(x_k)p_k^N = -\nabla f(x_k),$$

而Gauss-Newton则用以下方程组来寻找搜索方向：

$$J_k^T J_k p_k^{\mathrm{GN}} = -J_k^T r_k. \tag{10.23}$$

注：舍弃了(10.5)的第二项。

这个方法有如下优点：

首先，

$$\nabla^2 f_k \approx J_k^T J_k, \tag{10.24}$$

但它的计算量实际和

$$\nabla f_k = J_k^T r_k$$

相当。

注：这一点是问题依赖的。

其次，有很多实际问题中，(10.5)的第一项是强占优的，此时，Gauss-Newton法的收敛率也接近Newton法。第三，注意到当J_k是满秩时，∇f_k非零，所以p_k^{GN}始终是下降方向，这一点很适合线搜索。由(10.4)和(10.23)，我们有

$$\left(p_k^{\mathrm{GN}}\right)^T \nabla f_k = \left(p_k^{\mathrm{GN}}\right)^T J_k^T r_k = -(p_k^{\mathrm{GN}})^T J_k^T J_k p_k^{\mathrm{GN}} = -\left\|J_k p_k^{\mathrm{GN}}\right\|^2 \leq 0. \tag{10.25}$$

而当上式最后一项取等号时，由J_k满秩，我们立刻有$\nabla f_k = 0$，也即x_k是稳定点。最后一点是，Gauss-Newton法的每一步，实际上是一个线性最小二乘问题

$$\min_p \frac{1}{2}\|J_k p + r_k\|^2. \tag{10.26}$$

因此我们之前讨论过的线性最小二乘算法均可以用上。而当我们采用QR方法，或者SVD方法时，我们实际上不必去计算$J^T J$，而只需要J即可。想象一下我们如果用CG法去求解(10.26)，我们需要求解一个系数矩阵为$J_k^T J_k$的线性方程组，于是我们实际上只需要实现算子作用$J_k^T J_k$（即矩阵乘向量运算），而这一点我们可以通过先左乘J_k，再左乘J_k^T来实现。

但这一点也不是绝对的，当$m >> n$时，直接存储J可能不是一个好办法（这是个$m \times n$阶矩阵，想象一下你有100万个数据），此时先把$J^T J$和$J^T r$计算出来可能更好。我们通过一个$j = 1, 2, \cdots, m$的遍历，可以得到

$$J^T J = \sum_{i=1}^m (\nabla r_j)(\nabla r_j)^T, J^T r = \sum_{i=1}^m r_j(\nabla r_j). \tag{10.27}$$

当p_k^{GN}给出搜索方向之后，第三章介绍的线搜索策略我们就可以直接引入，比如Armijo或Wolfe条件。

注：这里跳过了一段关于Gauss-Newton法全局收敛性的证明，需要的同学自己看书。

Levenberg-Marquardt法

Gauss-Newton法的一个主要缺陷是当$J(x)$变得数值奇异时，作为线搜索方向可能出现收敛失败。而一个自然的解决思路是引入信任域策略。这就是Levenberg-Marquardt方法的思路。除此之外，它和Gauss-Newton法一致。对于迭代的每一步，它的子问题是

$$\min_p \frac{1}{2}\|J_k p + r_k\|_2^2, \quad \text{s.t.} \|p\|_2 \leq \Delta_k, \qquad (10.31)$$

这里$\Delta_k > 0$是信任域半径。也即我们实际上选择的二次模型是

$$m_k(p) = \frac{1}{2}\|r_k\|_2^2 + p^T J_k^T r_k + \frac{1}{2} p^T J_k^T J_k p. \quad (10.32)$$

以下讨论忽略下标k。由第四章的分析，我们知道当Gauss-Newton法的解p^{GN}落在信任域半径内时，它也是(10.31)的解。否则，存在$\lambda > 0$使得$p = p^{\mathrm{LM}}$满足$\|p\| = \Delta$，并且

$$\left(J^T J + \lambda J\right) p = -J^T r. \qquad (10.33)$$

因此下面的引理可以看作定理4.1的一个推论。

引理 10.2 向量p^{LM}是信任域子问题

$$\min_p \|Jp + r\|_2^2, \quad \text{s.t.} \|p\|_2 \leq \Delta$$

的解当且仅当p^{LM}可行且存在$\lambda \geq 0$使得

$$\left(J^T J + \lambda I\right) p^{\mathrm{LM}} = -J^T r, \qquad (10.34a)$$
$$\lambda \left(\Delta - \left\|p^{\mathrm{LM}}\right\|_2\right) = 0. \qquad (10.34b)$$

证明 略。

注意(10.33)实际上是线性最小二乘问题

$$\min_p \frac{1}{2}\left\| \begin{bmatrix} J \\ \sqrt{\lambda} I \end{bmatrix} p + \begin{bmatrix} r \\ 0 \end{bmatrix} \right\|_2^2. \qquad (10.35)$$

的正规方程组，和Gauss-Newton法类似的，我们这里考虑一下如何避免计算$J^T J$并对上述方程系数做Cholesky分解。

Levenberg-Marquardt 方法的实现

对引理10.2中的Δ，我们可以用第四章中介绍的求根方法寻找对应的λ，这里当$\lambda^{(l)}$为正时，对$B = J^T J$我们有正定性保证，但仍然可以通过(10.35)设计一种特殊的QR分解来提高效率和稳定性：

$$\begin{bmatrix} R_\lambda \\ 0 \end{bmatrix} = Q_\lambda^T \begin{bmatrix} J \\ \sqrt{\lambda} I \end{bmatrix}, \qquad (10.36)$$

其中Q_λ是正交阵，R_λ是上三角阵。

（注：算法上，即通过一系列正交变换，也即Householder变换，使得$\begin{bmatrix} J \\ \sqrt{\lambda} I \end{bmatrix}$上三角化，注意这是个长方阵，因此下方会出现零矩阵块。）

这种分解得到的R_λ满足

$$R_\lambda^T R_\lambda = J^T J + \lambda I.$$

具体的做法是通过Householder变换完成下面的分解：

$$J = Q \begin{bmatrix} R \\ 0 \end{bmatrix}. \tag{10.37}$$

于是我们就得到

$$\begin{bmatrix} R \\ 0 \\ \sqrt{\lambda} I \end{bmatrix} = \begin{bmatrix} Q^T & \\ & I \end{bmatrix} \begin{bmatrix} J \\ \sqrt{\lambda} I \end{bmatrix}. \tag{10.38}$$

左端的矩阵下方有一个子阵λI对应了n个非零，而这些非零元可以继续用$n(n+1)/2$次Givens旋转变换利用用R的对角元消去。具体做法是：

- 用R的第n行对$\sqrt{\lambda} I$的第n行做旋转，消去$\sqrt{\lambda} I$的(n,n)元；
- 用R的第$n-1$行对$\sqrt{\lambda} I$的第$n-1$行做旋转，消去$\sqrt{\lambda} I$的$(n-1,n-1)$元，但同时会将其$(n-1,n)$元填充为非零。再用R的第n行对$\sqrt{\lambda} I$的第$n-1$行做旋转，消去新增的非零元$(n-1,n)$；
- 如此自下而上，消去全部$\sqrt{\lambda} I$块。

如果我们把所有的Givens变换的算子叠加作用都用$\bar{Q}_\lambda$表示，那么有

$$\bar{Q}_\lambda^T \begin{bmatrix} R \\ 0 \\ \sqrt{\lambda} I \end{bmatrix} = \begin{bmatrix} R_\lambda \\ 0 \\ 0 \end{bmatrix},$$

而最终完成了(10.36)的分解，其中

$$Q_\lambda = \begin{bmatrix} Q & \\ & I \end{bmatrix} \bar{Q}_\lambda.$$

注意在求根过程中，当λ变化时，我们不需要重做Householder变换部分（主要计算量），因此当$m >> n$时，节省了大量的时间。

最小二乘问题的一个特点是数据往往极差很大。比如某些数据可能是10^4级别的，而另一些却是10^{-6}级别。这会导致额外的舍入误差和数值不稳定。一个解决的思路是用椭球型的(ellipsoidal)信任域去代替球型的(spherical)信任域，也即将信任域问题改为：

$$\min_p \frac{1}{2} \|J_k p + r_k\|_2^2, \quad \text{s.t.} \|D_k p\|_2 \le \Delta_k, \tag{10.39}$$

这里对角阵D_k实际上起着预处理阵的效果，它的构建也和预处理类似（并没有特别好的一般性方法）。注意(10.39)的解实际上满足的方程是

$$\left(J_k^T J_k + \lambda D_k^2\right) p_k^{\mathrm{LM}} = -J_k^T r_k, \tag{10.40}$$

或等价地，线性最小二乘问题为：

$$\min_p \left\| \begin{bmatrix} J_k \\ \sqrt{\lambda} D_k \end{bmatrix} p + \begin{bmatrix} r_k \\ 0 \end{bmatrix} \right\|_2^2. \tag{10.41}$$

这里D_k可能在迭代过程中不断调整。这个算法的具体讨论已经超出本书范围，并进入优化算法的科研前沿，有兴趣的同学请自行参阅Seber和Wild[280]等文献。

当$J(x)$矩阵是稀疏的时候，我们要注意在整个算法过程中，特别是求解(10.31)或(10.39)过程中要保持矩阵的稀疏性。这时可以考虑引入CG法的策略。这时用$J_k^T J_k$代替真实的Hessian阵就有了更进一步的意义。

LM方法的全局收敛性基本上是信任域收敛性的一个特列，我们不再详述，感兴趣的同学自己看书（这里跳过了一节）。

高残量问题的方法

如果$r(x)$相对而言不是一个小量，那么在$\nabla f^2(x)$的第二项就不能舍弃。于是我们之前讨论的Gauss-Newton和Levenberg-Marquardt方法都失去了前提。这时唯有回归Newton法或拟Newton法。然而后者的工作量确实很大。一个合理的策略，就是我们之前也提到过的混合(hybrid)法。也就是当残量变小时，启动GN或LM方法，而当残量变大时，启动Newton或拟Newton法。这些问题都已经涉及前沿领域，发表的方法很多，但是否符合你的实际需求则需要自己判断。Fletcher和Xu在[101]中给出的思路是维持一个正定的Hessian阵的近似，B_k。如果GN方法在一步当中对f有一个显著的下降（用一个固定的因子决定），则采用一步GN方法的结果，并将B_k用$J_k^T J_k$覆盖（因为拟Newton法需要上一步的Hessian近似）；否则，就以B_k计算一个拟Newton方向，并用线搜索方法得到x_{k+1}。无论如何，我们都始终采用类似BFGS的方法迭代得到新的B_{k+1}。这个方法在实际计算中表现不错。

第二个混合的思路是直接对

$$S_k = \sum_{j=1}^{m} r_j(x_k)\nabla^2 r_j(x_k)$$

展开类似拟Newton方法的递推估计，比如Dennis, Gay 和 Welsch [90]给出的公式如下：

$$S_{k+1} = S_k + \frac{\left(y^{\#} - S_k s\right) y^T + y\left(y^{\#} - S_k s\right)^T}{y^T s} - \frac{\left(y^{\#} - S_k s\right)^T s}{\left(y^T s\right)^2} y y^T \quad (10.43)$$

其中

$$\begin{aligned} s &= x_{k+1} - x_k, \\ y &= J_{k+1}^T r_{k+1} - J_k^T r_k, \\ y^{\#} &= J_{k+1}^T r_{k+1} - J_k^T r_{k+1}. \end{aligned}$$

正交距离回归(Orthogonal Distance Regression)

这种思想基于这样的假设，即我们的回归是"正确的"。也就是$\phi(x;t_j)$和y_j的不一致主要是由输入和输出的扰动决定的。这个时候，我们设法在最小化"误差投影"的意义下得到参数回归估计。也就是，假设

$$y_j = \phi(x;t_j+\delta_j) + \epsilon_j, j = 1,2,\cdots,m, \quad (10.44)$$

这里δ_j和ϵ_j都是未知的小扰动。而回归的具体模型为

$$\min_{x,\delta_j,\epsilon_j} \frac{1}{2}\sum_{j=1}^{m} \omega_j^2\epsilon^2 + d_j^2\delta_j^2, \quad \text{s.t.}(10.44). \quad (10.45)$$

这里ω_j和d_j称为权重(weight)，一般从模型或经验获得（嗯？！神经网络警告...）注意这里(10.45)将(10.44)当作约束。

要解决上述问题，我们实际上需要求解的无约束问题如下（其实一直有一点超纲，这些都是约束优化的方法）：

$$\min_{x,\delta} F(x,\delta) = \frac{1}{2}\sum_{j=1}^{m} w_j^2[y_j - \phi(x;t_j+\delta_j)]^2 + d_j^2\delta_j^2 = \frac{1}{2}\sum_{j=1}^{2m} r_j^2(x,\delta), \quad (10.46)$$

其中$\delta = (\delta_1,\delta_2,\cdots,\delta_m)^T$，以及

$$r_j(x,\delta) = \begin{cases} w_j\left[\phi(x;t_j+\delta_j) - y_j\right], & j = 1,2,\cdots,m. \\ d_{j-m}\delta_{j-m}, & j = m+1,\cdots,2m. \end{cases} \quad (10.47)$$

现在（10.46）已经成为一个$m+n$维（δ也是未知量）的最小二乘问题，我们可以用之前提到的所有方法去求解。但是直接求解这个问题的代价有点大。

好在(10.46)的Jacobi矩阵是有特殊结构可以在CG或LM方法中加以利用的。注意到

$$\frac{\partial r_j}{\partial \delta_i}=\frac{\partial\left[\phi(x;t_j+\delta_j)-y_j\right]}{\partial \delta_i}=0, i,j=1,2,\cdots,m, i\neq j,$$

且

$$\frac{\partial r_j}{\partial x_i}=0, j=m+1,m+2,\cdots,2m, i=1,2,\cdots,n.$$

以及对$j=1,2,\cdots,m$和$i=1,2,\cdots,m$，有

$$\frac{\partial r_{m+j}}{\partial \delta_j}=\begin{cases} d_j & i=j, \\ 0 & \text{其它}. \end{cases}$$

因此，我们可以对Jacobi矩阵做如下分块

$$J(x,\delta)=\begin{bmatrix}\hat{J} & V\\ 0 & D\end{bmatrix}, \tag{10.48}$$

其中V和D是$m\times m$阶对角阵，$\hat{J}$是$m\times n$阶矩阵，是函数$w_j\phi(t_j+\delta_j;x)$关于x的偏导数（Jacobi阵）。Boggs，Byrd和Schnabel[30]建议用LM方法去求解(10.46)，而对应(10.48)，将步长和残量都做对应划分：

$$p=\begin{bmatrix}p_x\\ p_\delta\end{bmatrix}, r=\begin{bmatrix}\hat{r_1}\\ \hat{r}_2\end{bmatrix},$$

于是对应的正规方程组为

$$\begin{bmatrix}\hat{J}^T\hat{J}+\lambda I & \hat{J}^TV\\ V\hat{J} & V^2+D^2+\lambda I\end{bmatrix}\begin{bmatrix}p_x\\ p_\delta\end{bmatrix}=-\begin{bmatrix}\hat{J}^T\hat{r}_1\\ V\hat{r}_1+D\hat{r}_2\end{bmatrix}. \tag{10.49}$$

注意到右下角这块$V^2+D^2+\lambda I$是一个对角阵，因此p_δ的计算是容易的。也就是我们实际上只需要求解一个$n\times n$阶的线性系统即可。