Practical Optimization Algorithms

Xiang Xu

School of Mathematical Sciences
Zhejiang University

May 6, 2022

Chapter VII: 非线性最小二乘问题

GAUSS-NEWTON 法

记 $F(x) = (F_1(x), F_2(x), \cdots, F_m(x))$, 非线性最小乘问题可以表示为

$$\min_{x \in R^n} \quad f(x) = \frac{1}{2}\|F(x)\|^2 = \frac{1}{2}\sum_{i=1}^{m} F_i^2(x) \tag{7.1}$$

GAUSS-NEWTON 法

记 $F(x) = (F_1(x), F_2(x), \cdots, F_m(x))$, 非线性最小乘问题可以表示为

$$\min_{x\in R^n} \quad f(x) = \frac{1}{2}\|F(x)\|^2 = \frac{1}{2}\sum_{i=1}^{m} F_i^2(x) \tag{7.1}$$

- 可以套用无约束优化问题的数值方法如牛顿法、拟牛顿法等方法求解

Gauss-Newton 法

记 $F(x) = (F_1(x), F_2(x), \cdots, F_m(x))$, 非线性最小乘问题可以表示为

$$\min_{x\in R^n} \quad f(x) = \frac{1}{2}\|F(x)\|^2 = \frac{1}{2}\sum_{i=1}^{m} F_i^2(x) \tag{7.1}$$

- 可以套用无约束优化问题的数值方法如牛顿法、拟牛顿法等方法求解
- 基于问题 (7.1)的特殊性, 在这些优化算法的基础上, 建立更适合本类问题的求解算法.

GAUSS-NEWTON 法

对应的梯度与Hessian阵分别为

$$\begin{aligned} g(x) &= \nabla f(x) = J(x)^T F(x) = \sum_{i=1}^m F_i(x) \nabla F_i(x) \\ G(x) &= \nabla^2 f(x) = \sum_{i=1}^m \nabla F_i(x)(\nabla F_i(x))^T + \sum_{i=1}^m F_i(x) \nabla^2 F_i(x) \\ &= J(x)^T J(x) + \sum_{i=1}^m F_i(x) \nabla^2 F_i(x) \\ &= J(x)^T J(x) + S(x) \end{aligned}$$

其中

$$J(x) = F'(x) = (\nabla F_1(x), \cdots, \nabla F_m(x))^T, S(x) = \sum_{i=1}^m F_i(x) \nabla^2 F_i(x).$$

GAUSS-NEWTON 法

利用牛顿型迭代算法, 我们便得到求解非线性最小二乘问题的迭代算法:

$$x_{k+1} = x_k - (J_k^T J_k + S_k)^{-1} J_k^T F(x_k)$$

GAUSS-NEWTON 法

利用牛顿型迭代算法, 我们便得到求解非线性最小二乘问题的迭代算法:

$$x_{k+1} = x_k - (J_k^T J_k + S_k)^{-1} J_k^T F(x_k)$$

- 在标准假设下, 容易得到该算法的收敛性质

GAUSS-NEWTON 法

利用牛顿型迭代算法, 我们便得到求解非线性最小二乘问题的迭代算法:

$$x_{k+1} = x_k - (J_k^T J_k + S_k)^{-1} J_k^T F(x_k)$$

- 在标准假设下, 容易得到该算法的收敛性质
- 缺点是 $S(x)$ 中 $\nabla^2 F_i(x)$的计算量较大

GAUSS-NEWTON 法

利用牛顿型迭代算法, 我们便得到求解非线性最小二乘问题的迭代算法:

$$x_{k+1} = x_k - (J_k^T J_k + S_k)^{-1} J_k^T F(x_k)$$

- 在标准假设下, 容易得到该算法的收敛性质
- 缺点是 $S(x)$ 中 $\nabla^2 F_i(x)$的计算量较大
- 如果忽略这一项, 便得到求解非线性最小二乘问题的 GAUSS-NEWTON 迭代算法

$$x_{k+1} = x_k + p_k^{GN}, \text{ 其中 } p_k^{GN} = -[J_k^T J_k]^{-1} J_k F(x_k)$$

Gauss-Newton 法

利用牛顿型迭代算法, 我们便得到求解非线性最小二乘问题的迭代算法:

$$x_{k+1} = x_k - (J_k^T J_k + S_k)^{-1} J_k^T F(x_k)$$

- 在标准假设下, 容易得到该算法的收敛性质
- 缺点是 $S(x)$ 中 $\nabla^2 F_i(x)$的计算量较大
- 如果忽略这一项, 便得到求解非线性最小二乘问题的 Gauss-Newton 迭代算法

$$x_{k+1} = x_k + p_k^{GN}, \text{ 其中 } p_k^{GN} = -[J_k^T J_k]^{-1} J_k F(x_k)$$

- 容易验证p_k^{GN}是如下优化问题的最优解

$$\min_{p\in R^n} \frac{1}{2}\|F(x_k) + J_k d\|^2$$

GAUSS-NEWTON 法

利用牛顿型迭代算法, 我们便得到求解非线性最小二乘问题的迭代算法:

$$x_{k+1} = x_k - (J_k^T J_k + S_k)^{-1} J_k^T F(x_k)$$

- 在标准假设下, 容易得到该算法的收敛性质
- 缺点是 $S(x)$ 中 $\nabla^2 F_i(x)$的计算量较大
- 如果忽略这一项, 便得到求解非线性最小二乘问题的 GAUSS-NEWTON 迭代算法

$$x_{k+1} = x_k + p_k^{GN}, \text{ 其中 } p_k^{GN} = -[J_k^T J_k]^{-1} J_k F(x_k)$$

- 容易验证p_k^{GN}是如下优化问题的最优解

$$\min_{p \in R^n} \frac{1}{2} \|F(x_k) + J_k d\|^2$$

- 若向量函数 $F(x)$ 的 Jacobian 矩阵是列满的, 则可以保证 Gauss - Newton方向是下降方向.

Gauss-Newton 法

利用牛顿型迭代算法, 我们便得到求解非线性最小二乘问题的迭代算法:

$$x_{k+1} = x_k - (J_k^T J_k + S_k)^{-1} J_k^T F(x_k)$$

- 在标准假设下, 容易得到该算法的收敛性质
- 缺点是 $S(x)$ 中 $\nabla^2 F_i(x)$的计算量较大
- 如果忽略这一项, 便得到求解非线性最小二乘问题的 Gauss-Newton 迭代算法

$$x_{k+1} = x_k + p_k^{GN}, \text{ 其中 } p_k^{GN} = -[J_k^T J_k]^{-1} J_k F(x_k)$$

- 容易验证p_k^{GN}是如下优化问题的最优解

$$\min_{p \in R^n} \frac{1}{2} \|F(x_k) + J_k d\|^2$$

- 若向量函数 $F(x)$ 的 Jacobian 矩阵是列满的, 则可以保证 Gauss - Newton方向是下降方向.
- 若采取单位步长,算法的收敛性难以保证. 但如果在算法中引入线搜索步长规则, 则可以得到如下的收敛性定理

Gauss-Newton 法

定理: 收敛性

设水平集 $\mathcal{L}(x_0)$ 有界, $J(x) = F'(x)$ 在 $\mathcal{L}(x_0)$ 上 Lipschitz 连续且 满足一致性条件

$$\|J(x)y\| \geq \alpha\|y\|, \ \forall y \in R^n \tag{7.2}$$

其中, $\alpha > 0$. 则在Wolfe步长规则下

$$\begin{cases} f(x_k + \alpha_k p_k) \leq f_k + \sigma_1 \alpha_k g_k^T p_k, \\ g(x_k + \alpha_k p_k)^T p_k \geq \sigma_2 g_k^T p_k \end{cases} \tag{7.3}$$

其中$0 < \sigma_1 < \sigma_2 < 1$. Gauss-Newton 算法产生的迭代点列 $\{x_k\}$ 收敛到 (7.1)的一个稳定点. 即

$$\lim_{k\to\infty} J(x_k)^T F(x_k) = 0.$$

GAUSS-NEWTON 法

定理：收敛率

设单位步长的 Gauss-Newton 算法产生的迭代点列 $\{x_k\}$ 收敛 到 (7.1) 的局部极小点 x^*, 而且 $J(x^*)^T J(x^*)$ 正定. 则当 $J(x)^T J(x)$, $S(x)$, $[J(x)^T J(x)]^{-1}$ 在 x^* 的邻域内 Lipschitz 连续时, 对充分大的 k , 有

$$\|x_{k+1} - x_k\| \leq \|[J(x^*)^T J(x^*)]^{-1}\| \|S(x^*)\| \|x_k - x^*\| + \mathcal{O}(\|x_k - x^*\|^2).$$

Levenberg-Marquardt 方法

- Gauss-Newton 算法在迭代过程中要求矩阵 $J(x_k)$列满秩, 而这一条件限制了它的应用.

Levenberg-Marquardt 方法

- Gauss-Newton 算法在迭代过程中要求矩阵 $J(x_k)$列满秩, 而这一条件限制了它的应用.
- Levenberg-Marquardt 方法通过求解下述优化模 型来获取搜索方向

$$p_k = \arg\min_{p \in R^n} \ \|J_k p + F_k\|^2 + \mu_k \|p\|^2, \text{ 其中 } \mu_k > 0.$$

Levenberg-Marquardt 方法

- Gauss-Newton 算法在迭代过程中要求矩阵 $J(x_k)$列满秩, 而这一条件限制了它的应用.
- Levenberg-Marquardt 方法通过求解下述优化模 型来获取搜索方向

$$p_k = \arg\min_{p\in R^n} \ \|J_k p + F_k\|^2 + \mu_k\|p\|^2, \text{ 其中 } \mu_k > 0.$$

- 由最优性条件知p_k 满足

$$\nabla(\|J_k p + F_k\|^2 + \mu_k\|p\|^2) = 2[(J_k^T J_k + \mu_k I)^{-1} p + J_k^T F_k] = 0$$

得到

$$p_k = -(J_k^T J_k + \mu_k I)^{-1} J_k^T F_k \tag{7.4}$$

Levenberg-Marquardt 方法

- Gauss-Newton 算法在迭代过程中要求矩阵 $J(x_k)$列满秩, 而这一条件限制了它的应用.
- Levenberg-Marquardt 方法通过求解下述优化模 型来获取搜索方向

$$p_k = \arg\min_{p\in R^n} \ \|J_kp + F_k\|^2 + \mu_k\|p\|^2, \text{ 其中 } \mu_k > 0.$$

- 由最优性条件知p_k 满足

$$\nabla(\|J_kp + F_k\|^2 + \mu_k\|p\|^2) = 2[(J_k^TJ_k + \mu_kI)^{-1}p + J_k^TF_k] = 0$$

得到

$$p_k = -(J_k^TJ_k + \mu_kI)^{-1}J_k^TF_k \tag{7.4}$$

- 若$g_k = J_k^TF_k \neq 0$, 则对任意的$\mu_k > 0$

$$g_k^Tp_k = -(J_k^TF_k)^T(J_k^TJ_k + \mu_kI)^{-1}(J_k^TF_k) < 0$$

所以 p_k 是 $f(x)$在x_k点的下降方向.

Levenberg-Marquardt 方法

全局收敛的L-M Algorithm

Step 1. 取$\rho, \sigma \in (0,1)$和$\mu_0 > 0$, $k = 1$
Step 2. 若$g(x_k) = 0$, 停止
Step 3. 求解方程组$(J_k^T J_k + \mu_k)p_k = -J_k^T F_k$
Step 4. 由Armijio搜索求步长. 令m_k是满足下面不等式的最小非负整数m:

$$f(x_k + \rho^m p_k) \leq f_k + \sigma \rho^m g_k^T p_k$$

令$\alpha_k = \rho^{m_k}$.
Step 5. $x_{k+1} := x_k + \alpha_k p_k$, $k = k + 1$, 按照某种方式更新μ_k, 转Step 2.

Levenberg-Marquardt 方法

全局收敛的L-M Algorithm

Step 1. 取$\rho, \sigma \in (0,1)$和$\mu_0 > 0$, $k = 1$
Step 2. 若$g(x_k) = 0$, 停止
Step 3. 求解方程组$(J_k^T J_k + \mu_k) p_k = -J_k^T F_k$
Step 4. 由Armijio搜索求步长. 令m_k是满足下面不等式的最小非负整数m:

$$f(x_k + \rho^m p_k) \le f_k + \sigma \rho^m g_k^T p_k$$

令$\alpha_k = \rho^{m_k}$.
Step 5. $x_{k+1} := x_k + \alpha_k p_k$, $k = k + 1$, 按照某种方式更新μ_k, 转Step 2.

- 注意到算法中搜索方向p_k的取值 其实是与μ_k有关的, 严格意义上讲, p_k应记为 $p_k(\mu_k)$.

LEVENBERG-MARQUARDT 方法

全局收敛的L-M ALGORITHM

Step 1. 取$\rho, \sigma \in (0,1)$和$\mu_0 > 0$, $k=1$
Step 2. 若$g(x_k)=0$, 停止
Step 3. 求解方程组$(J_k^T J_k + \mu_k)p_k = -J_k^T F_k$
Step 4. 由Armijio搜索求步长. 令m_k是满足下面不等式的最小非负整数m:

$$f(x_k + \rho^m p_k) \leq f_k + \sigma \rho^m g_k^T p_k$$

令$\alpha_k = \rho^{m_k}$.
Step 5. $x_{k+1} := x_k + \alpha_k p_k$, $k = k+1$, 按照某种方式更新μ_k, 转Step 2.

- 注意到算法中搜索方向p_k的取值 其实是与μ_k有关的, 严格意义上讲, p_k应记为 $p_k(\mu_k)$.
- 因此 L-M 方法的关键是在迭代过程中如何调整参数μ_k.

LEVENBERG-MARQUARDT 方法

引理

$\|p_k(\mu)\|$关于$\mu > 0$单调不增，且当$\mu \to \infty$时，$\|p_k(\mu)\| \to 0$.

引理

$p_k(\mu)$ 与 $-g_k$的夹角θ关于μ单调不增.

引理

$(J_k^T J_k + \mu I)$的条件数关于$\mu > 0$单调不增.

Levenberg-Marquardt 方法

在具体的L-M 算法中, 我们用类似于调整信赖域半径的策略来调整参数μ.

Levenberg-Marquardt 方法

在具体的L-M 算法中，我们用类似于调整信赖域半径的策略来调整参数μ.

- 首先，在当前迭代点定义一个二次函数

$$q_k(p) = f_k + (J_k^T F_k)^T p + \frac{1}{2} p^T (J_k^T J_k) p$$

基于当前给出的μ计算$p_k = -(J_k^T J_k + \mu_k I)^{-1} J_k F_k$.

Levenberg-Marquardt 方法

在具体的L-M 算法中，我们用类似于调整信赖域半径的策略来调整参数μ.

- 首先，在当前迭代点定义一个二次函数

$$q_k(p) = f_k + (J_k^T F_k)^T p + \frac{1}{2} p^T (J_k^T J_k) p$$

基于当前给出的μ计算$p_k = -(J_k^T J_k + \mu_k I)^{-1} J_k F_k$.

- 然后考虑$q_k(p)$和目标函数的增量

$$\Delta q_k(p_k) = q_k(p_k) - q_k(0) = (J_k^T F_k)^T p_k + \frac{1}{2} p_k^T (J_k^T J_k) p_k$$

$$\Delta f(p_k) = f(x_{k+1}) - f(x_k)$$

Levenberg-Marquardt 方法

在具体的L-M 算法中，我们用类似于调整信赖域半径的策略来调整参数μ.

- 首先，在当前迭代点定义一个二次函数
$$q_k(p) = f_k + (J_k^T F_k)^T p + \frac{1}{2} p^T (J_k^T J_k) p$$
基于当前给出的μ计算$p_k = -(J_k^T J_k + \mu_k I)^{-1} J_k F_k$.
- 然后考虑$q_k(p)$和目标函数的增量
$$\Delta q_k(p_k) = q_k(p_k) - q_k(0) = (J_k^T F_k)^T p_k + \frac{1}{2} p_k^T (J_k^T J_k) p_k$$
$$\Delta f(p_k) = f(x_{k+1}) - f(x_k)$$
- 计算η_k(两个增量之比)
$$\eta_k = \frac{\Delta f(p_k)}{\Delta q_k(p_k)}$$
然后根据η_k 的值调整μ_k

Levenberg-Marquardt 方法

在具体的L-M 算法中，我们用类似于调整信赖域半径的策略来调整参数μ.

- 首先，在当前迭代点定义一个二次函数

$$q_k(p) = f_k + (J_k^T F_k)^T p + \frac{1}{2} p^T (J_k^T J_k) p$$

  基于当前给出的μ计算$p_k = -(J_k^T J_k + \mu_k I)^{-1} J_k F_k$.

- 然后考虑$q_k(p)$和目标函数的增量

$$\Delta q_k(p_k) = q_k(p_k) - q_k(0) = (J_k^T F_k)^T p_k + \frac{1}{2} p_k^T (J_k^T J_k) p_k$$

$$\Delta f(p_k) = f(x_{k+1}) - f(x_k)$$

- 计算η_k(两个增量之比)

$$\eta_k = \frac{\Delta f(p_k)}{\Delta q_k(p_k)}$$

  然后根据η_k 的值调整μ_k

- 最后根据调整后的μ_k计算p_k，并进行线搜索，进而完 成 L-M 算法的一个迭代步

Levenberg-Marquardt 方法

Levenberg-Marquardt 方法

- 当 η_k 接近 1 时, 二次函数$q_k(p)$ 在x_k点拟合目标函数比较好, 用 L-M 方法求解非线性最小乘问题时, 参数μ应取得小一些.

LEVENBERG-MARQUARDT 方法

- 当 η_k 接近 1 时，二次函数$q_k(p)$ 在x_k点拟合目标函数比较好，用 L-M 方法求解非线性最小乘问题时，参数μ应取得小一些.
- 当 η_k 接近 0 时，二次函数$q_k(p)$在 x_k点拟合目标函数比较差，需要减小p_k的模长. 根据引理，应增大参数μ的取值来限制 p_k 的模长.

Levenberg-Marquardt 方法

- 当 η_k 接近 1 时，二次函数$q_k(p)$ 在x_k点拟合目标函数比较好，用 L-M 方法求解非线性最小乘问题时，参数μ应取得小一些.
- 当 η_k 接近 0 时，二次函数$q_k(p)$在 x_k点拟合目标函数比较差，需要减小p_k的模长. 根据引理，应增大参数μ的取值来限制 p_k 的模长.
- 而当比值 η_k 既不接近于 0 也不接近于 1，则认为参 数 μ_k 选取得当，不做调整.

Levenberg-Marquardt 方法

- 当 η_k 接近 1 时, 二次函数$q_k(p)$ 在x_k点拟合目标函数比较好, 用 L-M 方法求解非线性最小乘问题时, 参数μ应取得小一些.
- 当 η_k 接近 0 时, 二次函数$q_k(p)$在 x_k点拟合目标函数比较差, 需要减小p_k的模长. 根据引理, 应增大参数μ的取值来限制 p_k 的模长.
- 而当比值 η_k 既不接近于 0 也不接近于 1, 则认为参 数 μ_k 选取得当, 不做调整.

参数 μ_k 的一个更新规则如下

$$\mu_{k+1} := \begin{cases} 0.1\mu_k, & \text{当 } \eta_k > 0.75 \\ \mu_k, & \text{当 } 0.25 \le \eta_k \le 0.75 \\ 10\mu_k, & \text{当 } \eta_k < 0.25 \end{cases} \tag{7.5}$$

Levenberg-Marquardt 方法

定理： L-M算法收敛性

设 $\{x_k\}$ 是由L-M算法产生无穷迭代序列，若 $\{x_k, \mu_k\}$ 的某一聚点 (x^*, μ^*)满足 $J(x^*)^T J(x^*) + \mu I$正定，则 $\nabla f(x^*) = J(x^*)^T F(x^*) = 0$.

LEVENBERG-MARQUARDT 方法

定理： L-M算法收敛性

设 $\{x_k\}$ 是由L-M算法产生无穷迭代序列, 若 $\{x_k, \mu_k\}$ 的某一聚点 (x^*, μ^*)满足 $J(x^*)^T J(x^*) + \mu I$正定, 则 $\nabla f(x^*) = J(x^*)^T F(x^*) = 0$.

定理： L-M算法收敛速度

设 $\{x_k\}$ 是由L-M算法产生无穷迭代序列收敛到x^*是(7.1)的一个局部最优解. 若 $J(x^*)^T J(x^*)$非奇异, $(\frac{1}{2} - \sigma)J(x^*)^T J(x^*) - \frac{1}{2}S(x^*)$正定，且$G(x) = J(x)^T J(x) + S(x)$在$x^*$附近一致连续, $\mu_k \to 0$, 则当k充分大时, $\alpha_k = 1$, 且

$$\limsup_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \leq \|[J(x^*)^T J(x^*)]^{-1}\| \|S(x^*)\|.$$

Thanks For Your Attention