最优化方法及其 Matlab 程序设计

第三章 最速下降法和牛顿法

本章讨论无约束优化问题

$$\min_{x\in\mathbb{R}^n} \ f(x) \tag{3.1}$$

的最速下降法和牛顿法及其改进算法. 其中最速下降法是求解无约束优化问题最简单和最古老的方法之一, 虽然时至今日它不再具有实用性, 但它却研究其它无约束优化算法的基础, 许多有效算法都是以它基础通过改进或修正而得到的. 此外, 牛顿法也是一种经典的无约束优化算法, 并且因其收敛速度快以及具有自适应性等优点而至今仍受到科技工作者的青睐.

§3.1 最速下降方法及其 Matlab 实现

在第 2 章关于无约束优化问题下降类算法的一般框架时提及, 用不同的方式确定搜索方向或搜索步长, 就会得到不同的算法. 最速下

降法是用负梯度方向

$$d_k = -\nabla f(x_k) \tag{3.2}$$

作为搜索方向的 (因此也称为梯度法). 设 $f(x)$ 在 x_k 附近连续可微, d_k 为搜索方向向量, $g_k = \nabla f(x_k)$. 由泰勒展开式得

$$f(x_k + \alpha d_k) = f(x_k) + \alpha g_k^T d_k + o(\alpha), \quad \alpha > 0.$$

那么目标函数 $f(x)$ 在 x_k 处沿方向 d_k 下降的变化率为

$$\begin{aligned} \lim_{\alpha \to 0} \frac{f(x_k + \alpha d_k) - f(x_k)}{\alpha} &= \lim_{\alpha \to 0} \frac{\alpha g_k^T d_k + o(\alpha)}{\alpha} \\ &= g_k^T d_k = \|g_k\| \|d_k\| \cos \bar{\theta}_k, \end{aligned}$$

其中 $\bar{\theta}_k$ 是 g_k 与 d_k 的夹角. 显然, 对于不同的方向 d_k, 函数变化率取决于它与 g_k 夹角的余弦值. 要使变化率最小, 只有 $\cos \bar{\theta}_k = -1$, 即 $\bar{\theta}_k = \pi$ 时才能达到, 亦即 d_k 应该取 (3.2) 中的负梯度方向, 这也是将

负梯度方向叫作最速下降方向的由来. 下面给出最速下降法的具体计算步骤.

算法 3.1 (最速下降法)

步 0 选取初始点 $x_0 \in \mathbb{R}^n$, 容许误差 $0 \leq \varepsilon \ll 1$. 令 $k := 1$.

步 1 计算 $g_k = \nabla f(x_k)$. 若 $\|g_k\| \leq \varepsilon$, 停算, 输出 x_k 作为近似最优解.

步 2 取方向 $d_k = -g_k$.

步 3 由线搜索技术确定步长因子 α_k.

步 4 令 $x_{k+1} := x_k + \alpha_k d_k$, $k := k + 1$, 转步 1.

说明 步 3 中步长因子 α_k 的确定即可使用精确线搜索方法, 也可以使用非精确线搜索方法, 在理论上都能保证其全局收敛性. 若采用

精确线搜索方法, 即

$$f(x_k + \alpha_k d_k) = \min_{\alpha \geq 0} f(x_k + \alpha d_k),$$

那么 α_k 应满足

$$\phi'(\alpha) = \frac{\mathrm{d}}{\mathrm{d}\alpha} f(x_k + \alpha d_k)|_{\alpha=\alpha_k} = \nabla f(x_k + \alpha_k d_k)^T d_k = 0.$$

由 (3.2) 有

$$\nabla f(x_{k+1})^T \nabla f(x_k) = 0,$$

即新点 x_{k+1} 处的梯度与旧点 x_k 处的梯度是正交的, 也就是说迭代点列所走的路线是锯齿型的, 故其收敛速度是很缓慢的 (至多线性收敛速度).

由 $d_k = -g_k$ 及 (2.16), 即

$$cos\theta_k = \frac{-g_k^T d_k}{\|g_k\|\|d_k\|} = \frac{-g_k^T(-g_k)}{\|g_k\|\|-g_k\|} = 1, \ \Rightarrow \ \theta_k = 0,$$

故条件 (2.15) 必然满足 $(0 \le \theta_k \le \frac{\pi}{2} - \mu,\ \mu > 0)$. 从而直接应用定理 2.2 和定理 2.3 即得到最速下降法的全局收敛性定理:

定理 3.1 设目标函数 $f(x)$ 连续可微且其梯度函数 $\nabla f(x)$ 是 Lipschitz 连续的, $\{x_k\}$ 由最速下降法产生, 其中步长因子 α_k 由精确线搜索, 或由 Wolfe 准则, 或由 Armijo 准则产生. 则有

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0.$$

下面的定理给出了最速下降法求解严格凸二次函数极小值问题时的收敛速度估计, 其证明可参阅有关文献, 此处省略不证.

定理 3.2 设矩阵 $H \in \mathbb{R}^{n\times n}$ 对称正定, $c \in \mathbb{R}^n$. 记 λ_1 和 λ_n 分别是 H 最大和最小特征值, $\kappa = \lambda_1/\lambda_n$. 考虑如下极小化问题

$$\min f(x) = c^T x + \frac{1}{2} x^T H x.$$

设 $\{x_k\}$ 是用精确线搜索的最速下降法求解上述问题所产生的迭代序列, 则对于所有的 k, 下面的不等式成立

$$\|x_{k+1} - x^*\|_H \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)\|x_k - x^*\|_H, \tag{3.3}$$

其中, x^* 是问题的唯一解, $\|x\|_H = \sqrt{x^T H x}$.

由上面的定理可以看出, 若条件数 κ 接近于 1 (即 H 的最大特征值和最小特征值接近时), 最速下降法是收敛很快的. 但当条件数 κ 较大时 (即 H 近似于病态时), 算法的收敛速度是很缓慢的.

下面我们给出基于 Armijo 非精确线搜索的最速下降法 Matlab 程序.

程序 3.1 (最速下降法程序)

```
function [x,val,k]=grad(fun,gfun,x0)
%功能: 用最速下降法求解无约束问题: min f(x)
%输入: x0是初始点, fun, gfun分别是目标函数和梯度
```

```
%输出： x，val分别是近似最优点和最优值， k是迭代次数.
maxk=5000;   %最大迭代次数
rho=0.5;sigma=0.4;
k=0;  epsilon=1e-5;
while(k<maxk)
    g=feval(gfun,x0);  %计算梯度
    d=-g;    %计算搜索方向
    if(norm(d)<epsilon), break; end
    m=0; mk=0;
    while(m<20)   %Armijo搜索
        if(feval(fun,x0+rho^m*d)<feval(fun,x0)+sigma*rho^m*g'*d)
            mk=m; break;
        end
        m=m+1;
    end
    x0=x0+rho^mk*d;
    k=k+1;
end
x=x0;
val=feval(fun,x0);
```

例 3.1 利用程序 3.1 求解无约束优化问题

$$\min_{x\in\mathbb{R}^2} f(x) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2.$$

该问题有精确解 $x^* = (1,1)^T$, $f(x^*) = 0$.

解 首先建立两个分别计算目标函数和梯度的 m 文件:

```
function f=fun(x)
f=100*(x(1)^2-x(2))^2+(x(1)-1)^2;
function g=gfun(x)
g=[400*x(1)*(x(1)^2-x(2))+2*(x(1)-1), -200*(x(1)^2-x(2))]';
```

我们利用程序 3.1, 终止准则取为 $\|\nabla f(x_k)\| \le 10^{-5}$. 取不同的初始点, 数值结果如下表.

表 3.1 最速下降法的数值结果.

初始点 (x_0)	迭代次数 (k)	目标函数值 $(f(x_k))$
$(0.0, 0.0)^T$	1159	1.1630×10^{-10}
$(2.0, 1.0)^T$	611	1.1416×10^{-10}
$(1.0, -1.0)^T$	1551	1.2251×10^{-10}
$(-1.0, -1.0)^T$	1499	9.2536×10^{-11}
$(-1.2, 1)^T$	1435	1.1985×10^{-10}
$(10, -10)^T$	1024	1.0156×10^{-10}

由上表可以看出, 最速下降法的收敛速度是比较缓慢的.

说明 上述程序的调用方式是:

```
x0=[-1.2 1]';
[x,val,k]=grad('fun','gfun',x0)
```

其中 fun, gfun 分别是求目标函数值及其梯度的 M 函数文件.

§3.2 牛顿法及其 Matlab 实现

跟最速下降法一样, 牛顿法也是求解无约束优化问题最早使用的经典算法之一. 其基本思想是用迭代点 x_k 处的一阶导数 (梯度) 和二

阶导数 (Hesse 阵) 对目标函数进行二次函数近似, 然后把二次模型的极小点作为新的迭代点, 并不断重复这一过程, 直至求得满足精度的近似极小点.

下面来推导牛顿法的迭代公式. 设 $f(x)$ 的 Hesse 阵 $G(x) = \nabla^2 f(x)$ 连续, 截取其在 x_k 处的泰勒展开式的前三项得

$$q_k(x) = f_k + g_k^T(x - x_k) + \frac{1}{2}(x - x_k)^T G_k(x - x_k),$$

其中 $f_k = f(x_k)$, $g_k = \nabla f(x_k)$, $G_k = \nabla^2 f(x_k)$. 求二次函数 $q_k(x)$ 的稳定点, 得

$$\nabla q_k(x) = g_k + G_k(x - x_k) = 0.$$

若 G_k 非奇异, 那么解上面的线性方程组 (记其解为 x_{k+1}) 即得牛顿法的迭代公式

$$x_{k+1} = x_k - G_k^{-1} g_k. \tag{3.4}$$

在迭代公式 (3.4) 中每步迭代需要求 Hesse 阵的逆 G_k^{-1}, 在实际计算中可通过 先解 $G_k d = -g_k$ 得 d_k, 然后令 $x_{k+1} = x_k + d_k$ 来避免求逆. 这样, 可以写出基本牛顿法的步骤如下:

算法 3.2 (基本牛顿法)

步 0 给定终止误差值 $0 \leq \varepsilon \ll 1$, 初始点 $x_0 \in \mathbb{R}^n$. 令 $k := 0$.

步 1 计算 $g_k = \nabla f(x_k)$. 若 $\|g_k\| \leq \varepsilon$, 停算, 输出 $x^* \approx x_k$.

步 2 计算 $G_k = \nabla^2 f(x_k)$, 并求解线性方程组得解 d_k:

$$G_k d = -g_k.$$

步 3 令 $x_{k+1} = x_k + d_k$, $k := k + 1$, 转步 1.

牛顿法最突出的优点是收敛速度快, 具有局部二阶收敛性. 下面的定理表明了这一性质.

定理 3.3 设函数 $f(x)$ 有二阶连续偏导数, 在局部极小点 x^* 处, $G(x^*) = \nabla^2 f(x^*)$ 是正定的且 $G(x)$ 在 x^* 的一个邻域内是 Lipschitz 连续的. 如果初始点 x_0 充分靠近 x^*, 那么对一切 k, 牛顿迭代公式 (3.4) 是适定的, 并当 $\{x_k\}$ 为无穷点列时, 其极限为 x^*, 且收敛阶至少是二阶的.

证 由 $G(x^*)$ 的正定性及 f 二次连续可微可知, 存在 x^* 的一个邻域 $U(x^*)$, 使得对任意的 $x \in U(x^*)$, 都有 $G(x)$ 是一致正定的. 特别地, $\|G(x)^{-1}\|$ 在 $U_1(x^*)$ 上有界, 即存在常数 $M \geq 0$, 使得 $\|G(x)^{-1}\| \leq M$, $\forall\, x \in U_1(x^*)$. 又由 $G(x)$ 的连续性可知, 存在邻域 $U(x^*)$, 使得

$$\|G(x) - G(x^*)\| \leq \frac{1}{4M}, \quad \forall\, x \in U(x^*) \subseteq U_1(x^*).$$

因此, 当 $x_0 \in U(x^*)$ 时, 有

$$\begin{aligned}
\|x_1 - x^*\| &= \|x_0 - x^* - G_0^{-1} g_0\| \\
&\le \|G_0^{-1}\| \|g(x_0) - g(x^*) - G_0(x_0 - x^*)\| \\
&\le \|G_0^{-1}\| \left\| \int_0^1 G(x^* + \tau(x_0 - x^*))(x_0 - x^*) \mathrm{d}\tau - G_0(x_0 - x^*) \right\| \\
&\le M \int_0^1 \|G(x^* + \tau(x_0 - x^*)) - G(x_0)\| \|x_0 - x^*\| \mathrm{d}\tau \\
&\le M \left(\int_0^1 \|G(x^* + \tau(x_0 - x^*)) - G(x^*)\| \mathrm{d}\tau \right. \\
&\qquad \left. + \int_0^1 \|G(x_0) - G(x^*)\| \mathrm{d}\tau \right) \|x_0 - x^*\| \\
&\le \frac{1}{2} \|x_0 - x^*\|. \qquad (3.5)
\end{aligned}$$

上式特别说明, $x_1 \in U(x^*)$. 类似地, 利用归纳法原理, 可证明对于所

有的 $k \geq 1$ 有

$$\|x_{k+1} - x^*\| \leq \frac{1}{2}\|x_k - x^*\|.$$

因此 $\{x_k\} \subset U(x^*)$, 且 $x_k \to x^* \, (k \to \infty)$. 进一步, 类似于 (3.5) 的推导, 可得

$$\begin{aligned} \|x_{k+1} - x^*\| &\leq M \int_0^1 \|G(x^* + \tau(x_k - x^*)) - G(x_k)\| \|x_k - x^*\| \mathrm{d}\tau \\ &= o(\|x_k - x^*\|), \end{aligned}$$

即 $\{x_k\}$ 超线性收敛于 x^*. 若 $G(x)$ 在 x^* 的一个邻域内 Lipschitz 连续,

则由上式得

$$\begin{aligned}\|x_{k+1}-x^*\| &\le M\Big(\int_0^1 \|G(x^*+\tau(x_k-x^*))-G(x^*)\|\mathrm{d}\tau \\ &\qquad + \int_0^1 \|G(x^*)-G(x_k)\|\mathrm{d}\tau\Big)\|x_k-x^*\| \\ &\le LM\left(\int_0^1 \tau\mathrm{d}\tau+1\right)\|x_k-x^*\|^2 \\ &= \frac{3}{2}LM\|x_k-x^*\|^2,\end{aligned}$$

即 $\{x_k\}$ 二次收敛于 x^*. 证毕. □

上述定理指出, 初始点需要足够“靠近”极小点, 否则有可能导致算法不收敛. 由于实际问题的精确极小点一般是不知道的, 因此初始点的选取给算法的实际操作带来了很大的困难. 为了克服这一困难, 可引入线搜索技术以得到大范围收敛的算法, 即所谓的阻尼牛顿法.

我们给出一个基于 Armijo 搜索的阻尼牛顿法如下:

算法 3.3 (阻尼牛顿法)

步 0 给定终止误差值 $0 \leq \varepsilon \ll 1$, $\delta \in (0,1)$, $\sigma \in (0, 0.5)$. 初始点 $x_0 \in \mathbb{R}^n$. 令 $k : 0$.

步 1 计算 $g_k = \nabla f(x_k)$. 若 $\|g_k\| \leq \varepsilon$, 停算, 输出 $x^* \approx x_k$.

步 2 计算 $G_k = \nabla^2 f(x_k)$, 并求解线性方程组得解 d_k:

$$G_k d = -g_k \tag{3.6}$$

步 3 记 m_k 是满足下列不等式的最小非负整数 m:

$$f(x_k + \delta^m d_k) \leq f(x_k) + \sigma \delta^m g_k^T d_k. \tag{3.7}$$

步 4 令 $\alpha_k = \delta^{m_k}$, $x_{k+1} = x_k + \alpha_k d_k$, $k := k + 1$, 转步 1.

我们给出算法 3.3 的全局收敛性定理如下:

定理 3.4 设函数 $f(x)$ 二次连续可微且存在常数 $\gamma > 0$, 使得

$$d^T \nabla^2 f(x) d \geq \gamma \|d\|^2, \ \forall\, d \in \mathbb{R}^n, x \in L(x_0), \tag{3.8}$$

其中 $L(x_0) = \{x | f(x) \leq f(x_0)\}$. 设 $\{x_k\}$ 是由算法 3.3 产生的无穷点列, 则该点列收敛于 f 在水平集 $L(x_0)$ 中的唯一全局极小点.

证 由条件 (3.8) 知, f 是水平集 $L(x_0)$ 上的一致凸函数, 因此一定存在唯一的全局极小点 x^*, 且 x^* 是 $\nabla f(x) = 0$ 的唯一解.

由于 f 是凸函数, 故水平集 $L(x_0)$ 是一个有界闭凸集. 注意到算法 3.3 的步 3, 序列 $\{f(x_k)\}$ 是单调下降的. 故显然有 $\{x_k\} \subset L(x_0)$.

由 (3.6) 和 (3.8) 不难得到

$$\|g_k\| \leq \beta \|d_k\|,$$

其中 $\beta > 0$ 是某个常数. 设 $\bar{x}$ 是 $\{x_k\}$ 的任意一个极限点, 则有 $f_k \to f(\bar{x})$. 记 θ_k 是负梯度方向 $-g_k$ 与牛顿方向 $d_k = -G_k^{-1} g_k$ 的夹角. 由

上式和 (3.6), (3.8) 可推得

$$\begin{aligned}\cos\theta_k &= \frac{-g_k^T d_k}{\|g_k\|\|d_k\|} = \frac{d_k^T G_k d_k}{\|g_k\|\|d_k\|} \geq \frac{\gamma\|d_k\|^2}{\|g_k\|\|d_k\|} \\ &= \frac{\gamma\|d_k\|}{\|g_k\|} \geq \frac{\gamma}{\beta} > 0.\end{aligned}$$

则由定理 2.2 可得

$$\lim_{k\to\infty} g_k = 0,$$

即 $\{x_k\}$ 的极限点都是稳定点. 由 f 的凸性知, 稳定点亦即 (全局) 极小点. 故由极小点的唯一性知, $\{x_k\}$ 收敛于 f 在水平集 $L(x_0)$ 上的全局极小点. 证毕. □

为了分析算法 3.3 的收敛速度, 需要用到下面的引理, 其详细的证明过程可参见文献 [1].

引理 3.1 设函数 $f:\mathbb{R}^n \to \mathbb{R}$ 二次连续可微, 点列 $\{x_k\}$ 由算法

3.3 产生. 设 $\{x_k\} \to x^*$ 且 $g(x^*) = 0$, $G(x^*)$ 正定. 那么, 若

$$\lim_{k\to\infty} \frac{\|G(x_k)d_k + g_k\|}{\|d_k\|} = 0, \tag{3.9}$$

则 (1) 当 k 充分大时, 步长因子 $\alpha_k \equiv 1$. (2) 点列 $\{x_k\}$ 超线性收敛于 x^*.

定理 3.5 设定理 3.4 的条件成立, 点列 $\{x_k\}$ 由算法 3.3 产生. 则 $\{x_k\}$ 超线性收敛于 f 的全局极小点 x^*. 此外, 若 Hesse 阵 $G(x)$ 在 x^* 处 Lipschitz 连续, 则收敛阶至少是二阶的.

证 定理 3.4 已证明 $\{x_k\} \to x^*$ 且 $g(x^*) = \nabla f(x^*) = 0$. 又由算法 3.3 的步 2 显然有条件 (3.9) 成立. 故由引理 3.1(1) 可知对于充分大的 k, $\alpha_k = 1$ 满足算法中的线搜索式. 因此, 由定理 3.3 立即得到 $\{x_k\}$ 至少二阶收敛于 x^*. 证毕. □.

下面我们给出基于 Armijo 非精确线搜索的阻尼牛顿法 Matlab 程

序.

程序 3.2 (阻尼牛顿法程序)

```
function [x,val,k]=dampnm(fun,gfun,Hess,x0)
%功能: 用阻尼牛顿法求解无约束问题: min f(x)
%输入: x0是初始点, fun, gfun, Hess 分别是求
%       目标函数值,梯度,Hesse 阵的函数
%输出:  x, val分别是近似最优点和最优值,  k是迭代次数.
maxk=100;   %给出最大迭代次数
rho=0.55;sigma=0.4;
k=0;  epsilon=1e-5;
while(k<maxk)
    gk=feval(gfun,x0); %计算梯度
    Gk=feval(Hess,x0);  %计算Hesse阵
    dk=-Gk\gk; %解方程组Gk*dk=-gk, 计算搜索方向
    if(norm(gk)<epsilon), break; end  %检验终止准则
    m=0; mk=0;
    while(m<20)   % 用Armijo搜索求步长
        if(feval(fun,x0+rho^m*dk)<feval(fun,x0)+sigma*rho^m*gk'*dk)
            mk=m; break;
```

```
        end
        m=m+1;
    end
    x0=x0+rho^mk*dk;
    k=k+1;
end
x=x0;
val=feval(fun,x);
```

例 3.2 利用程序 3.2 求解无约束优化问题

$$\min_{x\in\mathbb{R}^2} f(x) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2.$$

该问题有精确解 $x^* = (1,1)^T$, $f(x^*) = 0$.

解 除了例 3.1 中建立的两个计算目标函数和梯度的 m 文件之外, 还需建立求 Hesse 阵的 m 文件:

```
function He=Hess(x)
n=length(x);
He=zeros(n,n);
```

```
He=[1200*x(1)^2-400*x(2)+2, -400*x(1);
    -400*x(1),                200    ];
```

我们利用程序 3.2, 终止准则取为 $\|\nabla f(x_k)\| \leq 10^{-5}$. 取不同的初始点, 数值结果如下表.

表 3.2 阻尼牛顿法的数值结果.

初始点 (x_0)	迭代次数 (k)	目标函数值 $(f(x_k))$
$(0,0)^T$	13	9.6238×10^{-15}
$(0.5,0.5)^T$	11	3.5183×10^{-19}
$(2,2)^T$	14	1.6322×10^{-14}
$(-1,-1)^T$	20	3.6221×10^{-17}
$(1,10)^T$	1	4.9309×10^{-28}
$(10,10)^T$	47	3.3426×10^{-17}
$(20,20)^T$	73	3.0386×10^{-17}

由上表可以看出, 阻尼牛顿法的收敛速度是比较令人满意的.

说明 上述程序的调用方式是:

```
x0=[-1.2 1]';
[x,val,k]=dampnm('fun','gfun','Hess',x0)
```

其中 fun, gfun, Hess 分别是求目标函数值和它的梯度及其 Hesse 阵的 M 函数文件.

§3.3 修正牛顿法及其 Matlab 实现

从上一节的分析可知, 牛顿法具有不低于二阶的收敛速度, 这是它的优点. 但该算法要求目标函数的 Hesse 阵 $G(x) = \nabla^2 f(x)$ 在每个迭代点 x_k 处是正定的, 否则难以保证牛顿方向 $d_k = -G_k^{-1} g_k$ 是 f 在 x_k 处的下降方向. 为克服这一缺陷, 可对牛顿法进行修正. 修正的途径之一是将牛顿法和最速下降法结合起来, 构造所谓的“牛顿-最速下降混合算法”, 其基本思想是: 当 $\nabla^2 f(x_k)$ 正定时, 采用牛顿方向作为搜索方向; 否则, 若 $\nabla^2 f(x_k)$ 奇异, 或者虽然非奇异但牛顿方向不是下降方向, 则采用负梯度方向作为搜索方向. 我们写出详细的计算步骤如下:

算法 3.4 (牛顿-最速下降混合算法)

步 0 给定初始点 $x_0 \in \mathbb{R}^n$, 终止误差 $0 \leq \varepsilon \ll 1$. 令 $k := 0$.

步 1 计算 $g_k = \nabla f(x_k)$. 若 $\|g_k\| \leq \varepsilon$, 停算, 输出 x_k 作为近似极小点.

步 2 计算 $G_k = \nabla^2 f(x_k)$. 解方程组

$$G_k d + g_k = 0. \tag{3.10}$$

若 (4.10) 有解 d_k 且满足 $g_k^T d_k < 0$, 转步 3; 否则, 令 $d_k = -g_k$ 转步 3.

步 3 由线搜索技术确定步长因子 α_k.

步 4 令 $x_{k+1} = x_k + \alpha_k d_k$, $k := k + 1$, 转步 1.

对于算法 3.4, 利用定理 2.2, 定理 2.3 以及引理 3.1 不难证明下面的收敛性定理.

定理 3.6 设对任意的 $x_0 \in \mathbb{R}^n$, 水平集 $L(x_0) = \{x \,|\, f(x) \leq$

$f(x_0)\}$ 有界, 且函数 f 在包含 $L(x_0)$ 的一个有界闭凸集上二次连续可微. $\{x_k\}$ 由采用精确线搜索, 或 Wolfe 准则, 或 Armijo 准则确定步长因子的算法 3.4 产生的迭代序列, 且存在 $\{x_k\}$ 的一个极限点 x^*, 使得 $G(x^*)$ 正定. 则有

$$\lim_{k\to\infty} \inf \|g_k\| = 0,$$

以及 $\{x_k\}$ 超线性收敛于 x^*. 进一步, 若 $G(x)$ 在 x^* 处是 Lipschitz 连续的, 则收敛速度至少是二阶的.

上述的修正牛顿法克服了牛顿法要求 Hesse 阵 $G(x_k) = \nabla^2 f(x_k)$ 正定的缺陷. 克服这一缺陷还有其它的方法和途径. 例如, 引进阻尼因子 $\mu_k \geq 0$, 即在每一迭代步适当地选取参数 μ_k 使得矩阵 $A_k = G(x_k) + \mu_k I$ 正定. 我们写出算法步骤如下:

算法 3.5 (修正牛顿法)

步 0 给定参数 $\delta \in (0,1)$, $\tau \in [0,1]$, $\sigma \in (0,0.5)$, 终止误差 $0 \leq \varepsilon \ll 1$. 初始点 $x_0 \in \mathbb{R}^n$. 令 $k := 0$.

步 1 计算 $g_k = \nabla f(x_k)$, $\mu_k = \|g_k\|^{1+\tau}$. 若 $\|g_k\| \leq \varepsilon$, 停算, 输出 x_k 作为近似极小点.

步 2 计算 Hesee 阵 $G_k = \nabla^2 f(x_k)$. 解线性方程组

$$(G_k + \mu_k I)d = -g_k, \tag{3.11}$$

得解 d_k.

步 3 令 m_k 是满足下列不等式的最小非负整数 m:

$$f(x_k + \delta^m d_k) \leq f(x_k) + \sigma \delta^m g_k^T d_k. \tag{3.12}$$

令 $\alpha_k = \delta^{m_k}$, $x_{k+1} := x_k + \alpha_k d_k$.

步 4 令 $k := k+1$, 转步 1.

下面的定理给出了算法 3.5 的全局收敛性, 其证明可参看文献 [?].

定理 3.7 设函数 $f:\mathbb{R}^n\to\mathbb{R}$ 有下界且二次连续可微, $G(x)=\nabla^2 f(x)$ 半正定且 Lipschitz 连续. 则由算法 3.5 产生的迭代序列 $\{x_k\}$ 的任何极限点都是问题 (3.1) 的解.

已有研究证明, 当目标函数的 Hesse 阵 $G(x)=\nabla^2 f(x)$ 在极小点 x^* 处奇异时, 牛顿法的收敛速度可能会降低为线性收敛速度. 下面我们给出奇异解的概念.

定义 3.1 若在问题 (3.1) 的极小点 x^* 处 Hesse 阵 $G(x^*)$ 奇异, 则称 x^* 是问题 (3.1) 的奇异解.

当问题 (3.1) 有奇异解时, 其解可能不唯一, 此时我们用 X 表示其解集, 即

$$X=\{x^*\mid f(x^*)=\min f(x),\ x\in\mathbb{R}^n\}.$$

定义 3.2 设 $x^* \in X$, 函数 $\theta : \mathbb{R}^n \to \mathbb{R}_+$. 若存在 x^*的邻域 $U(x^*)$ 及常数 $\gamma > 0$, 使得

$$\theta(x) \geq \gamma \mathrm{dist}(x, X), \quad x \in U(x^*), \tag{3.13}$$

其中 $\mathrm{dist}(x, X)$ 表示点 x 到集合 X 的距离, 则称函数 θ 在邻域 $U(x^*)$ 内对问题 (3.1) 解集合 X 提供了一个局部误差界.

下面的定理给出了算法 3.5 的局部收敛速度的估计, 其证明可参看文献 [4].

定理 3.8 设定理 3.7 的条件成立. 若由算法 3.5 产生的迭代序列 $\{x_k\}$ 有子列 $\{x_k : k \in K\}$ 收敛于 $x^* \in X$, 且函数 $\|g(x)\|$ 在 x^* 的某邻域内对问题 (3.1) 提供了一个局部误差界, 则当 $k \in K$ 充分大时, $\alpha_k \equiv 1$, 且子列 $\{x_k : k \in K\}$ 二阶收敛于 x^*, 即存在常数 $\beta > 0$

使得

$$\mathrm{dist}(x_{k+1}, X) \leq \beta \mathrm{dist}(x_k, X)^2.$$

下面我们给出算法 3.5 (修正牛顿法) 的 Matlab 程序.

程序 3.3 (修正牛顿法 Matlab 程序)

```
function [x,val,k]=revisenm(fun,gfun,Hess,x0)
% 功能: 用修正牛顿法求解无约束问题: min f(x)
%输入: x0是初始点, fun, gfun, Hess 分别是求
%       目标函数值,梯度,Hesse 阵的函数
%输出: x, val分别是近似最优点和最优值, k是迭代次数.
n=length(x0); maxk=150;
rho=0.55;sigma=0.4; tau=0.0;
k=0;  epsilon=1e-5;
while(k<maxk)
    gk=feval(gfun,x0); % 计算梯度
    muk=norm(gk)^(1+tau);
    Gk=feval(Hess,x0);  % 计算Hesse阵
    Ak=Gk+muk*eye(n);
    dk=-Ak\gk; %解方程组Gk*dk=-gk, 计算搜索方向
```

```
    if(norm(gk)<epsilon), break; end  %检验终止准则
    m=0; mk=0;
    while(m<20)   %用Armijo搜索求步长
        if(feval(fun,x0+rho^m*dk)<feval(fun,x0)+sigma*rho^m*gk'*dk)
            mk=m; break;
        end
        m=m+1;
    end
    x0=x0+rho^mk*dk;
    k=k+1;
end
x=x0;
val=feval(fun,x);
```

例 3.3 利用程序 3.3 求解无约束优化问题

$$\min_{x\in\mathbb{R}^2} f(x) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2.$$

该问题有精确解 $x^* = (1,1)^T$, $f(x^*) = 0$.

解 前面例 3.1 和例 3.2 中已经建立的计算目标函数及其梯度和 Hesee 阵的 m 文件, 可以重新利用. 取终止准则值为 $\|\nabla f(x_k)\| \leq 10^{-5}$, 利用程序 3.3, 取不同的初始点, 数值结果如下表.

表 3.3 修正牛顿法的数值结果.

初始点 (x_0)	迭代次数 (k)	目标函数值 $(f(x_k))$
$(0,0)^T$	16	4.7808×10^{-12}
$(0.5,0.5)^T$	10	2.4524×10^{-15}
$(2,2)^T$	17	1.3250×10^{-19}
$(-1,-1)^T$	23	2.3697×10^{-12}
$(1,10)^T$	1	4.9309×10^{-28}
$(10,10)^T$	46	4.5469×10^{-18}
$(20,20)^T$	76	9.1654×10^{-13}

由上表可以看出, 修正牛顿法的收敛速度不如阻尼牛顿法快, 因为矩阵 $A_k = G_k + \mu_k I$ 只是 Hesee 阵 G_k 的一个近似.

说明 上述程序的调用方式是:

```
x0=[-1.2 1]';
[x,val,k]=revisenm('fun','gfun','Hess',x0)
```

Back
Close

其中 fun, gfun, Hess 分别是求目标函数值和它的梯度及其 Hesse 阵的 M 函数文件.