最优化方法及其 Matlab 程序设计

马昌凤

2010 年 3 月 22 日

目 录

第二章　线搜索技术

通从本章开始, 介绍无约束优化问题的一些常用数值方法. 我们考虑下面的无约束优化模型

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

众所周知, 研究上述无约束优化问题的数值方法, 不仅是出于实际问题的需要, 同时也是研究约束优化问题数值方法的基础. 本章主要讨论一维线搜索算法及其收敛性分析.

我们在第 1 章论及无约束优化问题迭代算法的一般框架时, 其中有下面的一个迭代步:

通过某种搜索方式确定步长因子 α_k, 使得

$$f(x_k + \alpha_k d_k) < f(x_k). \tag{2.1}$$

这实际上是 (n 个变量的) 目标函数 $f(x)$ 在一个规定的方向上移动所形成的单变量优化问题, 也就是所谓的“线搜索”或“一维搜索”技术. 令

$$\phi(\alpha) = f(x_k + \alpha d_k), \tag{2.2}$$

这样, 搜索式 (2.1) 等价于求步长 α_k 使得

$$\phi(\alpha_k) < \phi(0).$$

线搜索有精确线搜索和非精确线搜索之分. 所谓精确线搜索, 是指求 α_k 使目标函数 f 沿方向 d_k 达到极小, 即

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

或

$$\phi(\alpha_k) = \min_{\alpha>0} \phi(\alpha).$$

若 $f(x)$ 是连续可微的, 那么由精确线搜索得到的步长因子 α_k 具有如下性质:

$$\nabla f(x_k+\alpha_k d_k)^T d_k = 0 \ (\text{亦即 } g_{k+1}^T d_k = 0). \tag{2.3}$$

上述性质在后面的算法收敛性分析中将起着重要的作用.

所谓非精确线搜索, 是指选取 α_k 使目标函数 f 得到可接受的下降量, 即 $\Delta f_k = f(x_k) - f(x_k+\alpha_k d_k) > 0$ 是可接受的.

精确线搜索的基本思想是: 首先确定包含问题最优解的搜索区间, 然后采用某种插值或分割技术缩小这个区间, 进行搜索求解. 下面给出搜索区间的定义.

定义 2.1 设 ϕ 是定义在实数集上一元实函数, $\alpha^* \in [0,+\infty)$, 并且

$$\phi(\alpha^*) = \min_{\alpha\geq 0} \phi(\alpha). \tag{2.4}$$

若存在区间 $[a,b] \subset [0,+\infty)$, 使 $\alpha^* \in (a,b)$, 则称 $[a,b]$ 是极小化问题 (2.4) 的搜索区间. 进一步, 若 α^* 使得 $\phi(\alpha)$ 在 $[a,\alpha^*]$ 上严格递减, 在 $[\alpha^*,b]$ 上严格递增, 则称 $[a,b]$ 是 $\phi(\alpha)$ 的单峰区间, $\phi(\alpha)$ 是 $[a,b]$ 上的单峰函数.

下面介绍一种确定搜索区间并保证具有近似单峰性质的数值算法—进退法, 其基本思想是从一点出发, 按一定步长, 试图确定函数值呈现“高-低-高”的三点, 从而得到一个近似的单峰区间.

算法 2.1 (**进退法**)

步 1　选取 $\alpha_0 \geq 0$, $h_0 > 0$. 计算 $\phi_0 := \phi(\alpha_0)$. 置 $k := 0$.

步 2　令 $\alpha_{k+1} = \alpha_k + h_k$, 计算 $\phi_{k+1} := \phi(\alpha_{k+1})$. 若 $\phi_{k+1} < \phi_k$, 转步 3, 否则转步 4.

步 3　加大步长. 令 $h_{k+1} := 2h_k$, $\alpha := \alpha_k$, $\alpha_k := \alpha_{k+1}$, $\phi_k := \phi_{k+1}$, $k := k + 1$, 转步 2.

步 4　反向搜索或输出. 若 $k = 0$, 令 $h_1 := h_0$, $\alpha := \alpha_1$, $\alpha_1 := \alpha_0$, $\phi_1 := \phi_0$, $k := 1$, 转步 2; 否则停止迭代, 令

$$a = \min\{\alpha, \alpha_{k+1}\}, \quad a = \max\{\alpha, \alpha_{k+1}\}.$$

输出 $[a, b]$.

2.1　精确线搜索及其 Matlab 实现

精确线搜索分为两类. 一类是使用导数的搜索, 如插值法, 牛顿法及抛物线法等; 另一类是不用导数的搜索, 如 0.618 法, 分数法及成功-失败法等. 本书仅介绍 0.618 法和二次插值逼近法.

2.1.1　黄金分割法

黄金分割法也称为 0.618 法, 其基本思想是通过试探点函数值得比较, 是包含极小点的搜索区间不断缩小. 该方法仅需要计算函数值, 适用范围广, 使用方便. 下面我们来推导 0.618 法的计算公式.

设

$$\phi(s) = f(x_k + sd_k),$$

其中 $\phi(s)$ 是搜索区间 $[a_0, b_0]$ 上的单峰函数. 在第 i 次迭代时搜索区间为 $[a_i, b_i]$. 取两个试探点为 $p_i, q_i \in [a_i, b_i]$ 且 $p_i < q_i$. 计算 $\phi(p_i$ 和 $\phi(q_i)$. 根据单峰函数的性质, 可能会出现如下两种情形之一:

(1) 若 $\phi(p_i) \le \phi(q_i)$, 则令 $a_{i+1} := a_i$, $b_{i+1} := q_i$;

(2) 若 $\phi(p_i) > \phi(q_i)$, 则令 $a_{i+1} := p_i$, $b_{i+1} := b_i$.

我们要求两个试探点 p_i 和 q_i 满足下述两个条件:

(a) $[a_i, q_i]$ 与 $[p_i, b_i]$ 的长度相同, 即 $b_i - p_i = q_i - a_i$;

(b) 区间长度的缩短率相同, 即 $b_{i+1} - a_{i+1} = t(b_i - a_i)$.

从而可得

$$p_i = a_i + (1-t)(b_i - a_i), \quad q_i = a_i + t(b_i - a_i). \tag{2.5}$$

现在考虑情形 (1). 此时, 新的搜索区间为

$$[a_{i+i}, b_{i+1}] = [a_i, q_i].$$

为了进一步缩短搜索区间, 需取新的试探点 p_{i+1}, q_{i+1}. 由 (2.5) 得

$$\begin{aligned} q_{i+1} &= a_{i+1} + t(b_{i+1} - a_{i+1}) \\ &= a_i + t(q_i - a_i) \\ &= a_i + t^2(b_i - a_i). \end{aligned}$$

若令

$$t^2 = 1 - t, \quad t > 0, \tag{2.6}$$

则

$$q_{i+1} = a_i + (1 - t)(b_i - a_i) = p_i.$$

这样, 新的试探点 q_{i+1} 就不需要重新计算. 类似地, 对于情形 (2), 也有相同的结论.

解方程 (2.6) 得区间长度缩短率为

$$t = \frac{\sqrt{5} - 1}{2} \approx 0.618.$$

因此, 我们可以写出 0.618 法的计算步骤如下.

算法 2.2 (0.618 法)

步 0　确定初始搜索区间 $[a_0, b_0]$ 和容许误差 $\varepsilon > 0$. 计算初始试探点

$$p_0 = a_0 + 0.382(b_0 - a_0), \quad q_0 = a_0 + 0.618(b_0 - a_0)$$

及相应的函数值 $\phi(p_0)$, $\phi(q_0)$. 置 $i := 0$.

步 1　若 $\phi(p_i) \le \phi(q_i)$, 转步 2; 否则, 转步 3.

步 2　计算左试探点. 若 $|q_i - a_i| \le \varepsilon$, 停算, 输出 p_i. 否则, 令

$$a_{i+1} := a_i, \quad b_{i+1} := q_i, \quad \phi(q_{i+1}) := \phi(p_i),$$
$$q_{i+1} := p_i, \quad p_{i+1} := a_{i+1} + 0.382(b_{i+1} - a_{i+}).$$

计算 $\phi(p_{i+1})$, $i := i + 1$, 转步 1.

步 3　计算右试探点. 若 $|b_i - p_i| \le \varepsilon$, 停算, 输出 q_i. 否则, 令

$$a_{i+1} := p_i, \quad b_{i+1} := b_i, \quad \phi(p_{i+1}) := \phi(q_i),$$
$$p_{i+1} := q_i, \quad q_{i+1} := a_{i+1} + 0.618(b_{i+1} - a_{i+}).$$

计算 $\phi(q_{i+1})$, $i := i + 1$, 转步 1.

值得说明的是, 由于每次迭代搜索区间的收缩率是 $t = 0.618$, 故 0.618 法只是线性收敛的, 即这一方法的计算效率并不高. 但该方法每次迭代只需计算一次函数值的优点足以弥补这一缺憾.

下面给出用 0.618 法求单变量函数 ϕ 在单峰区间上近似极小点的 Matlab 程序.

程序 2.1 (0.618 法程序) 用 0.618 法求单变量函数 ϕ 在单峰区间 $[a,b]$ 上的近似极小点.

```
function [s,phis,k,G,E]=golds(phi,a,b,delta,epsilon)
%输入: phi是目标函数, a, b 是搜索区间的两个端点
%       delta, epsilon分别是自变量和函数值的容许误差
%输出: s, phis分别是近似极小点和极小值,  G是nx4矩阵,
%       其第k行分别是a,p,q,b的第k次迭代值[ak,pk,qk,bk],
%       E=[ds,dphi], 分别是s和phis的误差限.
t=(sqrt(5)-1)/2;  h=b-a;
phia=feval(phi,a); phib=feval(phi,b);
p=a+(1-t)*h;  q=a+t*h;
phip=feval(phi,p); phiq=feval(phi,q);
k=1;  G(k,:)=[a, p, q, b];
while(abs(phib-phia)>epsilon)|(h>delta)
    if(phip<phiq)
```

```
        b=q;  phib=phiq; q=p; phiq=phip;
        h=b-a; p=a+(1-t)*h; phip=feval(phi,p);
    else
        a=p; phia=phip; p=q; phip=phiq;
        h=b-a;  q=a+t*h;  phiq=feval(phi,q);
    end
    k=k+1;  G(k,:)=[a, p, q, b];
end
ds=abs(b-a); dphi=abs(phib-phia);
if(phip<=phiq)
    s=p;  phis=phip;
else
    s=q;  phis=phiq;
end
E=[ds,dphi];
```

例 2.1 用 0.618 法程序 2.1 求函数 $\phi(x)=x^2-\sin(x)$ 在 $[0,1]$ 上的极小点. 取容许误差 $\delta=10^{-4}$, $\varepsilon=10^{-5}$.

解 在命令窗口输入如下命令

```
f=inline('s^2-sin(s)');
[s,phis,k,G,E]=golds(f,0,1,1e-4,1e-5)
```

回车后即得如下数值结果:

表 2.1 用 0.618 法求单变量函数极小点的数值结果.

迭代次数	近似极小点 $(\bar{s})$	$\lvert b_k-a_k\rvert$ 的值	$\lvert\phi(b_k)-\phi(a_k)\rvert$ 的值
21	0.450183	6.6107×10^{-5}	1.1075×10^{-9}

2.1.2　抛物线法

二次插值法的基本思想是: 在搜索区间中不断地使用二次多项式去

近似目标函数, 并逐步用插值多项式的极小点去逼近线搜索问题

$$\min_{s>0} \phi(s) = f(x_k + sd_k)$$

的极小点. 下面我们详细介绍这一方法.

设已知三点

$$s_0,\ \ s_1 = s_0 + h,\ \ s_2 = s_0 + 2h,\ \ (h > 0)$$

处的函数值 ϕ_0, ϕ_1, ϕ_2 且满足

$$\phi_1 < \phi_0, \quad \phi_1 < \phi_2.$$

上述条件保证了函数 ϕ 在区间 $[s_0, s_2]$ 上是单峰函数. 则满足上述条件的二次 Lagrabge 插值多项式为

$$q(s) = \frac{(s-s_1)(s-s_2)}{2h^2}\phi_0 - \frac{(s-s_0)(s-s_2)}{h^2}\phi_1 + \frac{(s-s_0)(s-s_1)}{2h^2}\phi_2.$$

$q(s)$ 的一阶导数为

$$q'(s) = \frac{2s-s_1-s_2}{2h^2}\phi_0 - \frac{2s-s_0-s_2}{h^2}\phi_1 + \frac{2s-s_0-s_1}{2h^2}\phi_2. \tag{2.7}$$

令 $q'(s)=0$ 解得

$$\begin{aligned}\bar{s} &= \frac{(s_1+s_2)\phi_0-2(s_0+s_2)\phi_1+(s_0+s_1)\phi_2}{2(\phi_0-2\phi_1+\phi_2)}\\ &= \frac{(2s_0+3h)\phi_0-2(2s_0+2h)\phi_1+(2s_0+h)\phi_2}{2(\phi_0-2\phi_1+\phi_2)}\\ &= s_0+\frac{(3\phi_0-4\phi_1+\phi_2)h}{2(\phi_0-2\phi_1+\phi_2)} := s_0+\bar{h},\end{aligned} \tag{2.8}$$

这里

$$\bar{h}=\frac{(4\phi_1-3\phi_0-\phi_2)h}{2(2\phi_1-\phi_0-\phi_2)}. \tag{2.9}$$

又 $q(s)$ 的二阶导数分别为

$$q''(s)=\frac{\phi_0}{h^2}-\frac{2\phi_1}{h^2}+\frac{\phi_2}{h^2}=\frac{\phi_0-2\phi_1+\phi_2}{h^2}>0.$$

故 $q(s)$ 为凸二次函数, 从而 $s_{\min}$ 是 $q(s)$ 的全局极小点.

注意到, $\bar{s}=s_0+\bar{h}$ 比 s_0 更好地逼近 s^*. 故可用 $\bar{s}$, $\bar{h}$ 分别替换 s_0 和 h 并重复上述计算过程, 求出新的 $\bar{s}$ 和新的 $\bar{h}$. 重复这一迭代过程, 直到得

到所需的精度为止. 值得说明的是, 这一算法中目标函数的导数在 (2.7) 中隐式地用来确定用来确定二次插值多项式的极小点, 而算法的程序实现中并不需要使用导数值.

算法 2.3 (**二次插值法**)

步 0　由算法 2.1 确定三点 $s_0 < s_1 < s_2$, 对应的函数值 ϕ_0, ϕ_1, ϕ_2 满足

$$\phi_1 < \phi_0, \quad \phi_1 < \phi_2.$$

设定容许误差 $\varepsilon > 0$.

步 1　若 $|s_2 - s_0| < \varepsilon$, 停算, 输出 $s^* \approx s_1$.

步 2　计算插值点. 根据 (2.8) 计算 $\bar{s}$ 和 $\bar{\phi} := \phi(\bar{s})$. 若 $\phi_1 \le \bar{\phi}$, 转步 4, 否则, 转步 3.

步 3　若 $s_1 > \bar{s}$, 则 $s_2 := s_1$, $s_1 := \bar{s}$, $\phi_2 := \phi_1$, $\phi_1 := \bar{\phi}$, 转步 1. 否则, $s_0 := s_1$, $s_1 := \bar{s}$, $\phi_0 := \phi_1$, $\phi_1 := \bar{\phi}$, 转步 1.

步 4　若 $s_1 < \bar{s}$, 则 $s_2 := \bar{s}$, $\phi_2 := \bar{\phi}$, 转步 1. 否则, $s_0 := \bar{s}$, $\phi_0 := \bar{\phi}$, 转步 1.

下面给出用二次插值法求单变量函数 ϕ 在单峰区间上近似极小点的 Matlab 程序.

程序 2.2 (二次插值法程序) 求函数 $\phi(s)$ 在区间 $[a,b]$ 上的局部极小值, 从初始点 s_0 开始, 然后在区间 $[a,s_0]$ 和 $[s_0,b]$ 上进行搜索.

```
function [s,phis,ds,dphi,S]=qmin(phi,a,b,delta,epsilon)
%输入: phi 是目标函数, a和b是搜索区间的端点
%       delta,epsilon是容许误差
%输出: s是近似极小点, phis是对应的近似极小值; k是迭代次数
%       ds是迭代终止时的步长, dphi是|phi(s1)-phi(s)|;  S是迭代向量
s0=a; maxj=20; maxk=30; big=1e6; err=1; k=1;
S(k)=s0; cond=0;  h=1; ds=0.00001;
if (abs(s0)>1e4), h=abs(s0)*(1e-4); end
while (k<maxk & err>epsilon &cond~=5)
    f1=(feval(phi,s0+ds)-feval(phi,s0-ds))/(2*ds);
```

```
if(f1>0), h=-abs(h); end
s1=s0+h;      s2=s0+2*h;       bars=s0;
phi0=feval(phi,s0);  phi1=feval(phi,s1);
phi2=feval(phi,s2);  barphi=phi0; cond=0;
j=0;  %确定h使得phi1<phi0且phi1<phi2
while(j<maxj&abs(h)>delta&cond==0)
    if (phi0<=phi1),
        s2=s1; phi2=phi1; h=0.5*h;
        s1=s0+h; phi1=feval(phi,s1);
    else if (phi2<phi1),
            s1=s2; phi1=phi2; h=2*h;
            s2=s0+2*h; phi2=feval(phi,s2);
        else
            cond=-1;
        end
```

```
        end
        j=j+1;
        if(abs(h)>big|abs(s0)>big), cond=5; end
    end
    if(cond==5)
        bars=s1; barphi=feval(phi,s1);
    else
        %二次插值求phis
        d=2*(2*phi1-phi0-phi2);
        if(d<0),
            barh=h*(4*phi1-3*phi0-phi2)/d;
        else
            barh=h/3; cond=4;
        end
        bars=s0+barh;    barphi=feval(phi,bars);
```

```
        h=abs(h);  h0=abs(barh);
        h1=abs(barh-h);  h2=abs(barh-2*h);
        %确定下一次迭代的h值
        if(h0<h), h=h0; end
        if(h1<h), h=h1; end
        if(h2<h), h=h2; end
        if(h==0), h=barh; end
        if(h<delta), cond=1; end
        if(abs(h)>big|abs(bars)>big), cond=5; end
        err=abs(phi1-barphi);
        s0=bars;  k=k+1; S(k)=s0;
    end
    if(cond==2&h<delta), cond=3; end
end
s=s0;  phis=feval(phi,s);
```

```
ds=h;  dphi=err;
```

例 2.2 用二次插值法程序 2.2 求函数 $\phi(x)=x^2-\sin(x)$ 在 $[0,1]$ 上的极小点. 取容许误差 $\delta=10^{-4}$, $\varepsilon=10^{-6}$.

解 在命令窗口输入如下命令

```
[s,phis,k,ds,dphi,S]=qmin(inline('s^2-sin(s)'),0,1,1e-4,1e-6)
```

回车后即得如下数值结果:

表 2.2 用二次插值法求单变量函数极小点的数值结果.

迭代次数	近似极小点 $(\bar{s})$	$\lvert\bar{s}-s_1\rvert$ 的值	$\lvert\phi(\bar{s})-\phi(s_1)\rvert$ 的值
4	0.450184	1.5340×10^{-5}	1.6984×10^{-9}

2.2　非精确线搜索及其 Matlab 实现

线搜索技术是求解许多优化问题下降算法的基本组成部分, 但精确线搜索往往需要计算很多的函数值和梯度值, 从而耗费较多的计算资源.

特别是当迭代点远离最优点时, 精确线搜索通常不是十分有效和合理的. 对于许多优化算法, 其收敛速度并不依赖于精确搜索过程. 因此, 既能保证目标函数具有可接受的下降量又能使最终形成的迭代序列收敛的非精确线搜索变得越来越流行. 本书着重介绍非精确线搜索中的 Wolfe 准则和 Armijo 准则.

1. Wolfe 准则

Wolfe 准则是指: 给定 $\rho \in (0, 0.5)$, $\sigma \in (\rho, 0.5)$, 求 α_k 使得下面两个不等式同时成立:

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \rho \alpha_k g_k^T d_k, \tag{2.10}$$

$$\nabla f(x_k + \alpha_k d_k)^T d_k \geq \sigma g_k^T d_k, \tag{2.11}$$

其中 $g_k = g(x_k) = \nabla f(x_k)$. 条件 (2.11) 有时也用另一个更强的条件

$$|\nabla f(x_k + \alpha_k d_k)^T d_k| \leq -\sigma g_k^T d_k \tag{2.12}$$

来代替. 这样, 当 $\sigma > 0$ 充分小时, 可保证 (2.12) 变成近似精确线搜索. (2.10) 和 (2.12) 也称为强 Wolfe 准则.

强 Wolfe 准则表明, 由该准则得到的新的迭代点 $x_{k+1} = x_k + \alpha_k d_k$ 在 x_k 的某一邻域内且使目标函数值有一定的下降量.

由于 $g_k^T d_k < 0$, 可以证明 Wolfe 准则的有限终止性, 即步长 α_k 的存在性. 我们有下面的定理.

定理 2.1 设 $f(x)$ 有下界且 $g_k^T d_k < 0$, 令 $\rho \in (0, 0.5)$, $\sigma \in (\rho, 1)$. 则存在一个区间 $[a, b]$, $0 < a < b$, 使每个 $\alpha \in [a, b]$ 均满足 (2.10) 和 (2.12).

2. Armijo 准则

Armijo 准则是指: 给定 $\beta \in (0, 1)$, $\sigma \in (0, 0.5)$. 令步长因子 $\alpha_k = \beta^{m_k}$, 其中 m_k 是满足下列不等式的最小非负整数:

$$f(x_k + \beta^m d_k) \leq f(x_k) + \sigma \beta^m g_k^T d_k. \tag{2.13}$$

可以证明, 若 $f(x)$ 是连续可微的且满足 $g_k^T d_k < 0$, 则 Armijo 准则是有限终止的, 即存在正数 σ, 使得对于充分大的正整数 m, (2.13) 式成立.

为了程序实现的方便, 我们将 Armijo 准则写成下列详细的算法步骤.

算法 2.4 (**Armijo 准则**)

步 0　给定 $\beta \in (0,1)$, $\sigma \in (0, 0.5)$. 令 $m := 0$.

步 1　若不等式

$$f(x_k + \beta^m d_k) \leq f(x_k) + \sigma \beta^m g_k^T d_k$$

成立, 置 $m_k := m$, $x_{k+1} := x_k + \beta^{m_k} d_k$, 停算. 否则, 转步 2.

步 2　令 $m := m + 1$, 转步 1.

下面给出 Armijo 准则的 Matlab 程序.

程序 2.3 (**Armijo 准则程序**) Armijo 搜索规则是许多非线性优化算法都必须执行的步骤, 把它编制成可重复利用的程序模块是很有意义的.

```
function mk=armijo(xk,dk )
beta=0.5;  sigma=0.2;
m=0; mmax=20;
while (m<=mmax)
```

```
        if(fun(xk+beta^m*dk)<=fun(xk)+sigma*beta^m*gfun(xk)'*dk)
            mk=m;   break;
        end
        m=m+1;
    end
    alpha=beta^mk
    newxk=xk+alpha*dk
    fk=fun(xk)
    newfk=fun(newxk)
```

说明 程序 2.3 中 fun 和 gfun 分别是指目标函数和它的梯度函数的子程序. 执行上述过程时这两个子程序必须事先准备好.

我们用上述程序来求解下面的问题.

例 2.3 考虑无约束优化问题

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

设当前迭代点 $x_k = (-1, 1)^T$, 下降方向 $d_k = (1, -2)^T$. 利用程序 2.3 求步长因子 α_k.

解 首先编写好目标函数及其梯度两个 m 文件 fun.m 和 gfun.m:

```
% 目标函数
function f=fun(x)
f=100*(x(1)^2-x(2))^2+(x(1)-1)^2;
% 梯度
function gf=gfun(x)
gf=[400*x(1)*(x(1)^2-x(2))+2*(x(1)-1), -200*(x(1)^2-x(2))]';
```

然后在 Matlab 命令窗口输入下列代码

```
xk=[-1,1]'; dk=[1,-2]'; mk=armijo(xk,dk)
```

可得

$$m_k = 2;\ \alpha_k = 0.25;\ x_{k+1} = (-0.75, 0.5)^T;\ f(x_k) = 4;\ f(x_{k+1}) = 3.4531.$$

2.3　线搜索法的收敛性

下面我们给出线搜索法的收敛性结果. 所谓“线搜索法”是指用线搜索技术求步长因子的无约束优化问题下降类算法的简称. 其一般的算法框架是:

算法 2.5 (线搜索法算法框架)

步 0　初始化. 选取有关参数及初始迭代点 $x_0 \in \mathbb{R}^n$. 设定容许误差 $\varepsilon \ll 1$. 令 $k := 0$.

步 1　检验终止判别准则. 计算 $g_k = \nabla f(x_k)$. 若 $\|g_k\| \leq \varepsilon$, 输出 $x^* \approx x_k$, 停算.

步 2　确定下降方向 d_k, 使满足 $g_k^T d_k < 0$.

步 3　确定步长因子 α_k. 可在下列“精确”与“非精确”两种线搜索技术中选用其一:

(1)　用前面介绍的 0.618 法或二次插值法等精确线搜索技术求

$$\alpha_k = \arg\min_{\alpha>0} f(x_k + \alpha d_k). \tag{2.14}$$

(2) 用前面介绍的 Wolfe 准则或 Armijo 准则等非精确线搜索技术求 α_k.

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

值得说明的是, 为了保证这类算法的收敛性, 除了在步 3 要选用适当的线搜索技术外, 搜索方向 d_k 也需满足一定的条件, 即对于所有的 k, d_k 与 $-g_k$ 的夹角 θ_k 满足

$$0 \leq \theta_k \leq \frac{\pi}{2} - \mu, \quad \mu \in \left(0, \frac{\pi}{2}\right). \tag{2.15}$$

显然, 夹角 θ_k 的余弦为

$$\cos\theta_k = \frac{-g_k^T d_k}{\|g_k\|\|d_k\|}. \tag{2.16}$$

下面的定理描述了基于 Wolfe 准则或 Armijo 准则的非精确线搜索法的收敛性.

定理 2.2 设 $\{x_k\}$ 是由算法 2.5 产生的序列, $f(x)$ 有下界且对任意的

$x_0 \in \mathbb{R}^n$, $\nabla f(x)$ 在水平集

$$L(x_0) = \{x \in \mathbb{R}^n | \, f(x) \leq f(x_0)\}$$

上存在且一致连续. 若下降方向 d_k 满足条件 (2.15), 则

(1) 采用 Wolfe 准则求搜索步长 α_k 时, 有 $\{\|g_k\|\} \to 0$, $k \to \infty$.

(2) 采用 Armijo 准则求搜索步长 α_k 时, $\{x_k\}$ 的任何聚点 x^* 都满足 $\nabla f(x^*) = 0$.

证 (1) 用反证法. 设存在子列 (仍记指标为 k), 使得 $\|g_k\| \geq \varepsilon > 0$. 注意到 d_k 是下降方向, 由 Wolfe 准则的条件 (2.10) 知, $\{f(x_k)\}$ 是单调下降的. 又 $f(x_k)$ 有下界, 故 $f(x_k)$ 得极限存在, 因此有 $f(x_k) - f(x_{k+1}) \to 0$. 令 $s_k = \alpha_k d_k$, 则由 (2.10) 和 (2.15) 可得 $\cos\theta_k \geq \sin\mu$ 及

$$0 \leq -g_k^T(\alpha_k d_k) = -g_k^T s_k \leq \frac{1}{\rho}\big(f(x_k) - f(x_{k+1})\big) \to 0. \tag{2.17}$$

故

$$0 \leq \|g_k\|\|s_k\| \sin\mu \leq \|g_k\|\|s_k\| \cos\theta_k = -g_k^T s_k \to 0.$$

注意到 $\|g_k\| \geq \varepsilon > 0$, 故由上式必有 $\|s_k\| \to 0$. 又由于 $\nabla f(x)$ 在水平集 $L(x_0)$ 上是一致连续的, 我们有

$$\nabla f(x_{k+1})^T s_k = g_k^T s_k + o(\|s_k\|),$$

即

$$\lim_{k\to\infty} \frac{\nabla f(x_{k+1})^T s_k}{g_k^T s_k} = 1,$$

这与 (2.11) 式及 $\sigma < 1$ 矛盾. 因而必有 $\|g_k\| \to 0$.

(2) 用反证法. 假设 x^* 是序列 $\{x_k\}$ 的聚点且 $\nabla f(x^*) \neq 0$. 由定理的条件可得 $f(x_k) \to f(x^*)$ 及 $f(x_k) - f(x_{k+1}) \to 0$. 又由 Armijo 准则的条件 (2.13), 我们有

$$-\sigma g_k^T s_k \to 0, \quad g_k^T s_k \to 0,$$

此处 $s_k = \beta^{m_k} d_k$. 若 $g(x_k) \to 0$ 不成立, 则由上式可得 $\|s_k\| \to 0$. 由于在 Armijo 准则的条件 (2.13) 中, m_k 是使不等式成立的最小非负整数, 因此对于 $\beta^{m_k-1} = \beta^{m_k}/\beta$, 不等式变为

$$f(x_k + \beta^{m_k-1} d_k) - f(x_k) > \sigma \beta^{m_k-1} g_k^T d_k.$$

注意到 $\beta^{m_k-1}d_k = s_k/\beta$, 故上式即

$$f\left(x_k + \frac{1}{\beta}s_k\right) - f(x_k) > \sigma g_k^T\left(\frac{s_k}{\beta}\right). \tag{2.18}$$

若令 $p_k = \dfrac{s_k}{\|s_k\|}$, 则 $\dfrac{s_k}{\beta} = \dfrac{\|s_k\|}{\beta}p_k$. 由 $\|s_k\| \to 0$ 可知, $\alpha_k' = \dfrac{\|s_k\|}{\beta} \to 0$ 且 (2.18) 式可改写为

$$\frac{f(x_k + \alpha_k' p_k) - f(x_k)}{\alpha_k'} > \sigma g_k^T p_k.$$

因 $\|p_k\| = 1$, 故 $\{\|p_k\|\}$ 有界, 从而存在收敛的子列, 仍记为 $\{\|p_k\|\} \to p^*$, $\|p^*\| = 1$. 对上式两边取极限得

$$\nabla f(x^*)^T p^* \geq \sigma \nabla f(x^*)^T p^*.$$

由此可得

$$\nabla f(x^*)^T p^* \geq 0. \tag{2.19}$$

另一方面, 注意到

$$p_k = \frac{s_k}{\|s_k\|} = \frac{d_k}{\|d_k\|},$$

故有

$$-g_k^T p_k = -g_k^T \left(\frac{d_k}{\|d_k\|} \right) = \|g_k\| \cos \theta_k \geq \|g_k\| \sin \mu.$$

对上式取极限得

$$-\nabla f(x^*)^T p^* \geq \|\nabla f(x^*)\| \sin \mu > 0,$$

即

$$\nabla f(x^*)^T p^* < 0,$$

这与 (2.19) 式矛盾. 故必有 $\nabla f(x^*) = 0$. □

下面的定理描述了基于精确线搜索技术的无约束优化问题下降类算法的收敛性.

定理 2.3 设 $\{x_k\}$ 是由算法 2.5 产生的序列, $f(x)$ 有下界且对任意的 $x_0 \in \mathbb{R}^n$, $\nabla f(x)$ 在水平集

$$L(x_0) = \{x \in \mathbb{R}^n | \, f(x) \leq f(x_0)\}$$

上存在且一致连续. 若下降方向 d_k 满足条件 (2.15) 且搜索步长 α_k 满足精确线搜索条件 (2.14). 则若非 $g_k = 0$, 必有 $g_k \to 0\,(k \to \infty)$.

证 只需证明 $g_k \neq 0$ 时必有 $g_k \to 0\,(k \to \infty)$ 成立. 用反证法. 若 $g_k \to 0$ 不成立, 则存在常数 $\varepsilon > 0$ 和一个子列 (仍记为该序列本身) 使得 $\|g_k\| \geq \varepsilon$.

首先, 由算法 2.5 的步 4 知, $\{f(x_k)\}$ 是单调下降的, 再注意到 $f(x_k)$ 有下界, 故序列 $\{f(x_k)\}$ 的极限存在, 从而

$$f(x_{k+1}) - f(x_k) \to 0, \quad k \to \infty. \tag{2.20}$$

另一方面, 我们有

$$\frac{-g_k^T d_k}{\|d_k\|} = \|g_k\| \cos\theta_k \geq \varepsilon \sin\mu \equiv \varepsilon_0. \tag{2.21}$$

由泰勒展开式得

$$\begin{aligned} f(x_k+\alpha d_k) &= f(x_k)+\alpha g(\xi_k)^T d_k \\ &= f(x_k)+\alpha g_k^T d_k+\alpha[g(\xi_k)-g_k]^T d_k \\ &\le f(x_k)+\alpha\|d_k\|\left[\frac{g_k^T d_k}{\|d_k\|}+\|g(\xi_k)-g_k\|\right], \end{aligned} \tag{2.22}$$

其中 ξ_k 是连接 x_k 与 $x_k+\alpha d_k$ 线段上的某一点. 由于 $g(x)=\nabla f(x)$ 在水平集 $L(x_0)$ 上一致连续, 故存在 $\bar{\alpha}>0$, 使当 $0\le\alpha\|d_k\|\le\bar{\alpha}$ 时,

$$\|g(\xi_k)-g_k\|\le\frac{1}{2}\varepsilon_0,\quad \forall\, k\ge 0. \tag{2.23}$$

在 (2.22) 中令 $\alpha=\bar{\alpha}/\|d_k\|$, 并利用 (2.21) 和 (2.23) 得

$$\begin{aligned} f\left(x_k+\frac{\bar{\alpha}}{\|d_k\|}d_k\right) &\le f(x_k)+\bar{\alpha}\left[\frac{g_k^T d_k}{\|d_k\|}+\|g(\xi_k)-g_k\|\right] \\ &\le f(x_k)+\bar{\alpha}\left[-\varepsilon_0+\frac{1}{2}\varepsilon_0\right] \\ &= f(x_k)-\frac{1}{2}\bar{\alpha}\varepsilon_0. \end{aligned}$$

由精确搜索条件得

$$f(x_{k+1}) \leq f\left(x_k + \frac{\bar{\alpha}}{\|d_k\|} d_k\right) \leq f(x_k) - \frac{1}{2}\bar{\alpha}\varepsilon_0.$$

这与 (2.20) 式矛盾, 从而 $g_k \to 0$. □