Practical Optimization Algorithms
实用优化算法

徐 翔

数学科学学院
浙江大学

Mar 18, 2022

第二讲: LINE SEARCH METHODS (线搜索方法)

GENERAL DESCRIPTION

- 一般迭代格式为$x_{k+1} = x_k + \alpha_k p_k$ 关键是构造搜索方向p_k和步长因子α_k.

General Description

- 一般迭代格式为$x_{k+1} = x_k + \alpha_k p_k$ 关键是构造搜索方向p_k和步长因子α_k.
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$, 沿着$p_k$, 确定步长因子$\alpha_k$使得 $\varphi(\alpha_k) < \varphi(0)$.

General Description

- 一般迭代格式为$x_{k+1} = x_k + \alpha_k p_k$ 关键是构造搜索方向p_k和步长因子α_k.
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$, 沿着$p_k$, 确定步长因子$\alpha_k$使得 $\varphi(\alpha_k) < \varphi(0)$.
    - $\alpha_k = \arg\min_{\alpha>0} \varphi(\alpha)$ 称为最优线搜索或精确线搜索，或最优一维搜索.

General Description

- 一般迭代格式为$x_{k+1} = x_k + \alpha_k p_k$ 关键是构造搜索方向p_k和步长因子α_k.
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$, 沿着$p_k$, 确定步长因子$\alpha_k$使得 $\varphi(\alpha_k) < \varphi(0)$.
  - $\alpha_k = \arg\min_{\alpha>0} \varphi(\alpha)$ 称为最优线搜索或精确线搜索，或最优一维搜索.
  - 如果α_k, 使目标函数f得到可接受的下降量, 即使得下降量 $f(x_k) - f(x_k + \alpha_k p_k) > 0$ 是可以接受的, 则称这样的一维搜索为近似一维搜索，或不精确一维搜索.

GENERAL DESCRIPTION

- 一般迭代格式为$x_{k+1} = x_k + \alpha_k p_k$ 关键是构造搜索方向p_k和步长因子α_k.
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$, 沿着$p_k$, 确定步长因子$\alpha_k$使得 $\varphi(\alpha_k) < \varphi(0)$.
  - $\alpha_k = \arg\min_{\alpha>0} \varphi(\alpha)$ 称为最优线搜索或精确线搜索，或最优一维搜索.
  - 如果α_k, 使目标函数f得到可接受的下降量, 即使得下降量 $f(x_k) - f(x_k + \alpha_k p_k) > 0$ 是可以接受的, 则称这样的一维搜索为近似一维搜索，或不精确一维搜索.
- 一维搜索主要结构:

GENERAL DESCRIPTION

- 一般迭代格式为$x_{k+1} = x_k + \alpha_k p_k$ 关键是构造搜索方向p_k和步长因子α_k.
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$, 沿着$p_k$, 确定步长因子$\alpha_k$使得 $\varphi(\alpha_k) < \varphi(0)$.
  - $\alpha_k = \arg\min_{\alpha>0} \varphi(\alpha)$ 称为最优线搜索或精确线搜索, 或最优一维搜索.
  - 如果α_k, 使目标函数f得到可接受的下降量, 即使得下降量 $f(x_k) - f(x_k + \alpha_k p_k) > 0$ 是可以接受的, 则称这样的一维搜索为近似一维搜索, 或不精确一维搜索.
- 一维搜索主要结构:
  - 首先确定包含问题最优解得搜索区间,

GENERAL DESCRIPTION

- 一般迭代格式为$x_{k+1} = x_k + \alpha_k p_k$ 关键是构造搜索方向p_k和步长因子α_k.
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$, 沿着$p_k$, 确定步长因子$\alpha_k$使得 $\varphi(\alpha_k) < \varphi(0)$.
  - $\alpha_k = \arg\min_{\alpha>0} \varphi(\alpha)$ 称为最优线搜索或精确线搜索，或最优一维搜索.
  - 如果α_k, 使目标函数f得到可接受的下降量, 即使得下降量 $f(x_k) - f(x_k + \alpha_k p_k) > 0$ 是可以接受的, 则称这样的一维搜索为近似一维搜索，或不精确一维搜索.
- 一维搜索主要结构:
  - 首先确定包含问题最优解得搜索区间,
  - 采用某种分割技术或插值方法缩小这个区间, 进行搜索.

General Description

- 一般迭代格式为$x_{k+1} = x_k + \alpha_k p_k$ 关键是构造搜索方向p_k和步长因子α_k.
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$, 沿着$p_k$, 确定步长因子$\alpha_k$使得 $\varphi(\alpha_k) < \varphi(0)$.
  - $\alpha_k = \arg\min_{\alpha>0} \varphi(\alpha)$ 称为最优线搜索或精确线搜索，或最优一维搜索.
  - 如果α_k, 使目标函数f得到可接受的下降量, 即使得下降量 $f(x_k) - f(x_k + \alpha_k p_k) > 0$ 是可以接受的, 则称这样的一维搜索为近似一维搜索，或不精确一维搜索.
- 一维搜索主要结构:
  - 首先确定包含问题最优解得搜索区间,
  - 采用某种分割技术或插值方法缩小这个区间, 进行搜索.
- 设α^* 是满足 $\varphi(\alpha^*) = \min_{\alpha\geq 0} \varphi(\alpha)$. 如果存在$[a,b] \subset [0,\infty)$, 使得$\alpha^* \in [a,b]$, 则称$[a,b]$是一维极小化$\min_{\alpha\geq 0} \varphi(\alpha)$的搜索区间.

GENERAL DESCRIPTION

- 一般迭代格式为$x_{k+1} = x_k + \alpha_k p_k$ 关键是构造搜索方向p_k和步长因子α_k.
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$, 沿着$p_k$, 确定步长因子$\alpha_k$使得 $\varphi(\alpha_k) < \varphi(0)$.
    - $\alpha_k = \arg\min_{\alpha>0} \varphi(\alpha)$ 称为最优线搜索或精确线搜索，或最优一维搜索.
    - 如果α_k, 使目标函数f得到可接受的下降量, 即使得下降量 $f(x_k) - f(x_k + \alpha_k p_k) > 0$ 是可以接受的, 则称这样的一维搜索为近似一维搜索，或不精确一维搜索.
- 一维搜索主要结构:
    - 首先确定包含问题最优解得搜索区间,
    - 采用某种分割技术或插值方法缩小这个区间, 进行搜索.
- 设α^* 是满足 $\varphi(\alpha^*) = \min_{\alpha\geq 0} \varphi(\alpha)$. 如果存在$[a, b] \subset [0, \infty)$, 使得$\alpha^* \in [a, b]$, 则称$[a, b]$是一维极小化$\min_{\alpha\geq 0} \varphi(\alpha)$的搜索区间.
- 确定搜索区间的一种简单方法：进退法。基本思想是从一点出发，按一定步长，试图确定出函数值呈现"高-低-高"三点. 一个方向不成功, 就退回来, 再沿相反方向寻找.

General Description

进退法搜索

1. 选取初始数据.

General Description

进退法搜索

1. 选取初始数据. 给定α_0, $h_0 > 0$, 加倍系数$t > 1$, 计算$\varphi(\alpha_0)$, 设$k = 0$;

GENERAL DESCRIPTION

进退法搜索

1. 选取初始数据. 给定α_0, $h_0 > 0$, 加倍系数$t > 1$, 计算$\varphi(\alpha_0)$, 设$k = 0$;
2. 比较目标函数值.

General Description

进退法搜索

1. 选取初始数据. 给定α_0, $h_0 > 0$, 加倍系数$t > 1$, 计算$\varphi(\alpha_0)$, 设$k = 0$;
2. 比较目标函数值. 令$\alpha_{k+1} = \alpha_k + h_k$, 计算$\varphi_{k+1} = \varphi(\alpha_{k+1})$,

GENERAL DESCRIPTION

进退法搜索

1. 选取初始数据. 给定α_0, $h_0 > 0$, 加倍系数$t > 1$, 计算$\varphi(\alpha_0)$, 设$k = 0$;
2. 比较目标函数值. 令$\alpha_{k+1} = \alpha_k + h_k$, 计算$\varphi_{k+1} = \varphi(\alpha_{k+1})$, 如果$\varphi_{k+1} < \varphi_k$, 转步3, 否则转步4

GENERAL DESCRIPTION

进退法搜索

1. 选取初始数据. 给定α_0, $h_0 > 0$, 加倍系数$t > 1$, 计算$\varphi(\alpha_0)$, 设$k = 0$;
2. 比较目标函数值. 令$\alpha_{k+1} = \alpha_k + h_k$, 计算$\varphi_{k+1} = \varphi(\alpha_{k+1})$,
   如果$\varphi_{k+1} < \varphi_k$, 转步3, 否则转步4
3. 加大搜索步长.

GENERAL DESCRIPTION

进退法搜索

1. 选取初始数据. 给定α_0, $h_0 > 0$, 加倍系数$t > 1$, 计算$\varphi(\alpha_0)$, 设$k = 0$;
2. 比较目标函数值. 令$\alpha_{k+1} = \alpha_k + h_k$, 计算$\varphi_{k+1} = \varphi(\alpha_{k+1})$, 如果$\varphi_{k+1} < \varphi_k$, 转步3, 否则转步4
3. 加大搜索步长. 令$h_{k+1} = th_k$, $\alpha = \alpha_k$, $\alpha_k = \alpha_{k+1}$, $\varphi_k = \varphi_{k+1}$, $k = k + 1$, 转步2.

GENERAL DESCRIPTION

进退法搜索

1. 选取初始数据. 给定α_0, $h_0 > 0$, 加倍系数$t > 1$, 计算$\varphi(\alpha_0)$, 设$k = 0$;
2. 比较目标函数值. 令$\alpha_{k+1} = \alpha_k + h_k$, 计算$\varphi_{k+1} = \varphi(\alpha_{k+1})$, 如果$\varphi_{k+1} < \varphi_k$, 转步3, 否则转步4
3. 加大搜索步长. 令$h_{k+1} = th_k$, $\alpha = \alpha_k$, $\alpha_k = \alpha_{k+1}$, $\varphi_k = \varphi_{k+1}$, $k = k + 1$, 转步2.
4. 反向探索.

GENERAL DESCRIPTION

进退法搜索

1. 选取初始数据. 给定α_0, $h_0 > 0$, 加倍系数$t > 1$, 计算$\varphi(\alpha_0)$, 设$k = 0$;
2. 比较目标函数值. 令$\alpha_{k+1} = \alpha_k + h_k$, 计算$\varphi_{k+1} = \varphi(\alpha_{k+1})$, 如果$\varphi_{k+1} < \varphi_k$, 转步3, 否则转步4
3. 加大搜索步长. 令$h_{k+1} = th_k$, $\alpha = \alpha_k$, $\alpha_k = \alpha_{k+1}$, $\varphi_k = \varphi_{k+1}$, $k = k + 1$, 转步2.
4. 反向探索. 若$k = 0$, 转换探索方向, 令$h_k := -h_k$, $\alpha_k = \alpha_{k+1}$, 转步2;

General Description

进退法搜索

1. 选取初始数据. 给定α_0, $h_0 > 0$, 加倍系数$t > 1$, 计算$\varphi(\alpha_0)$, 设$k = 0$;
2. 比较目标函数值. 令$\alpha_{k+1} = \alpha_k + h_k$, 计算$\varphi_{k+1} = \varphi(\alpha_{k+1})$,
如果$\varphi_{k+1} < \varphi_k$, 转步3, 否则转步4
3. 加大搜索步长. 令$h_{k+1} = th_k$, $\alpha = \alpha_k$, $\alpha_k = \alpha_{k+1}$, $\varphi_k = \varphi_{k+1}$, $k = k+1$,
转步2.
4. 反向探索. 若$k = 0$, 转换探索方向, 令$h_k := -h_k$, $\alpha_k = \alpha_{k+1}$, 转步2;
否则, 停止迭代, 令

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

General Description

进退法搜索

1. 选取初始数据. 给定α_0, $h_0 > 0$, 加倍系数$t > 1$, 计算$\varphi(\alpha_0)$, 设$k = 0$;
2. 比较目标函数值. 令$\alpha_{k+1} = \alpha_k + h_k$, 计算$\varphi_{k+1} = \varphi(\alpha_{k+1})$,
   如果$\varphi_{k+1} < \varphi_k$, 转步3, 否则转步4
3. 加大搜索步长. 令$h_{k+1} = th_k$, $\alpha = \alpha_k$, $\alpha_k = \alpha_{k+1}$, $\varphi_k = \varphi_{k+1}$, $k = k+1$,
   转步2.
4. 反向探索. 若$k = 0$, 转换探索方向, 令$h_k := -h_k$, $\alpha_k = \alpha_{k+1}$, 转步2;
   否则, 停止迭代, 令

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

定义单峰/谷函数(unimodal function)

设$\varphi : R \to R$, $[a, b] \subset R$, 若存在$\alpha^* \in [a, b]$, 使得$\varphi(\alpha)$ 在$[a, \alpha^*]$上严格递减,
在$[\alpha^*, b]$上严格递增, 则称$[a, b]$是函数φ的单峰区间(或单谷区间).

精确一维搜索

算法2.1

给定$x_0 \in R^n$, $0 \leq \varepsilon \ll 1$;

精确一维搜索

算法2.1

给定$x_0 \in R^n$, $0 \le \varepsilon \ll 1$;

for $k = 0,\ 1,\ \cdots$

精确一维搜索

算法2.1

给定$x_0 \in R^n$, $0 \le \varepsilon \ll 1$;

for $k = 0,\ 1,\ \cdots$

  计算搜索方向p_k;

精确一维搜索

算法2.1

给定$x_0 \in R^n$, $0 \leq \varepsilon \ll 1$;

for $k = 0,\ 1,\ \cdots$

　　计算搜索方向p_k;

　　计算步长α_k, 使得 $f(x_k + \alpha_k p_k) = \min\limits_{\alpha \geq 0} f(x_k + \alpha p_k)$;

精确一维搜索

算法2.1

给定$x_0 \in R^n$, $0 \le \varepsilon \ll 1$;

for $k = 0,\ 1,\ \cdots$

计算搜索方向p_k;

计算步长α_k, 使得 $f(x_k + \alpha_k p_k) = \min\limits_{\alpha \ge 0} f(x_k + \alpha p_k)$;

$x_{k+1} = x_k + \alpha_k p_k$;

精确一维搜索

算法2.1

给定$x_0 \in R^n$, $0 \leq \varepsilon \ll 1$;

for $k = 0, 1, \cdots$

计算搜索方向p_k;

计算步长α_k, 使得 $f(x_k + \alpha_k p_k) = \min_{\alpha \geq 0} f(x_k + \alpha p_k)$;

$x_{k+1} = x_k + \alpha_k p_k$;

if $\|\nabla f(x_k)\| \leq \varepsilon$

精确一维搜索

算法2.1

给定$x_0 \in R^n$, $0 \le \varepsilon \ll 1$;

for $k = 0,\ 1,\ \cdots$

  计算搜索方向p_k;

  计算步长α_k, 使得 $f(x_k + \alpha_k p_k) = \min_{\alpha \ge 0} f(x_k + \alpha p_k)$;

  $x_{k+1} = x_k + \alpha_k p_k$;

  if $\|\nabla f(x_k)\| \le \varepsilon$

    stop;

精确一维搜索

算法2.1

给定$x_0 \in R^n$, $0 \le \varepsilon \ll 1$;

for $k = 0,\ 1,\ \cdots$

  计算搜索方向p_k;

  计算步长α_k, 使得 $f(x_k + \alpha_k p_k) = \min\limits_{\alpha \ge 0} f(x_k + \alpha p_k)$;

  $x_{k+1} = x_k + \alpha_k p_k$;

  if $\|\nabla f(x_k)\| \le \varepsilon$

    stop;

  end (if)

end (for)

精确一维搜索

算法2.1

给定$x_0 \in R^n$, $0 \le \varepsilon \ll 1$;

for $k = 0,\ 1,\ \cdots$

  计算搜索方向p_k;

  计算步长α_k, 使得 $f(x_k + \alpha_k p_k) = \min_{\alpha \ge 0} f(x_k + \alpha p_k)$;

  $x_{k+1} = x_k + \alpha_k p_k$;

  if $\|\nabla f(x_k)\| \le \varepsilon$

    stop;

  end (if)

end (for)

定义向量之间的夹角

设$\theta_k = \langle p_k, \nabla f(x_k)\rangle$表示向量$p_k$和向量$\nabla f(x_k)$之间的夹角，则有

$$\cos\theta_k = \cos\langle p_k, \nabla f(x_k)\rangle = \frac{p_k^T \nabla f(x_k)}{\|p_k\|\|\nabla f(x_k)\|}.$$

精确线性搜索的收敛性

定理

设$\alpha_k > 0$是精确线性搜索的解, $\|\nabla^2 f(x_k + \alpha p_k)\| \leq M$, 则有

$$f(x_k) - f(x_k + \alpha_k p_k) \geq \frac{1}{2M} \|\nabla f(x_k)\|^2 \cos^2 \theta_k$$

精确线性搜索的收敛性

定理

设$\alpha_k > 0$是精确线性搜索的解, $\|\nabla^2 f(x_k + \alpha p_k)\| \leq M$, 则有

$$f(x_k) - f(x_k + \alpha_k p_k) \geq \frac{1}{2M}\|\nabla f(x_k)\|^2 \cos^2 \theta_k$$

证明

由假设可知, 对于任意的α满足

$$f(x_k + \alpha p_k) \leq f(x_k) + \alpha p_k^T \nabla f(x_k) + \frac{\alpha^2}{2} M \|p_k\|^2$$

精确线性搜索的收敛性

定理

设$\alpha_k > 0$是精确线性搜索的解, $\|\nabla^2 f(x_k + \alpha p_k)\| \leq M$, 则有

$$f(x_k) - f(x_k + \alpha_k p_k) \geq \frac{1}{2M} \|\nabla f(x_k)\|^2 \cos^2 \theta_k$$

证明

由假设可知, 对于任意的α满足

$$f(x_k + \alpha p_k) \leq f(x_k) + \alpha p_k^T \nabla f(x_k) + \frac{\alpha^2}{2} M \|p_k\|^2$$

不妨取$\alpha = \bar{\alpha} = -p_k^T \nabla f(x_k)/(M\|p_k\|^2)$, 则有

精确线性搜索的收敛性

定理

设$\alpha_k > 0$是精确线性搜索的解，$\|\nabla^2 f(x_k + \alpha p_k)\| \le M$，则有

$$f(x_k) - f(x_k + \alpha_k p_k) \ge \frac{1}{2M}\|\nabla f(x_k)\|^2 \cos^2\theta_k$$

证明

由假设可知，对于任意的α满足

$$f(x_k + \alpha p_k) \le f(x_k) + \alpha p_k^T \nabla f(x_k) + \frac{\alpha^2}{2} M\|p_k\|^2$$

不妨取$\alpha = \bar{\alpha} = -p_k^T \nabla f(x_k)/(M\|p_k\|^2)$，则有

$$f(x_k) - f(x_k + \alpha_k p_k) \ge f(x_k) - f(x_k + \bar{\alpha} p_k) \ge -\bar{\alpha} p_k^T \nabla f(x_k) - \frac{\bar{\alpha}^2}{2} M\|p_k\|^2$$

$$= \frac{1}{2}\frac{(p_k^T \nabla f(x_k))^2}{M\|p_k\|^2} = \frac{1}{2M}\|\nabla f(x_k)\|^2 \frac{(p_k^T \nabla f(x_k))^2}{\|p_k\|^2\|\nabla f(x_k)\|^2} = \frac{1}{2M}\|\nabla f(x_k)\|^2 \cos^2\theta_k$$

精确线性搜索的收敛性

定理

- 设f是连续可微函数, 任意的极小化算法2.1产生的$\{x_k\}$满足

$$(i) f(x_{k+1}) \leq f(x_k), \forall k; \quad (ii) p_k^T \nabla f(x_k) \leq 0.$$

- 假设x^*是$\{x_k\}$的聚点, K_1是满足$\lim_{k \in K_1} x_k = x^*$的指标集. 假设存在$M > 0$, 使得$\|p_k\| < M$, $\forall k \in K_1$. 设$\bar{p}$是序列 $\{p_k\}$的任意一个聚点, 则

$$\nabla f(x^*)^T \bar{p} = 0.$$

- 进一步, 如果再设$f(x)$在D上二次连续可微, 则有

$$\bar{p} \nabla^2 f(\bar{x}) \bar{p} \geq 0.$$

精确线性搜索的收敛性

定理

设$\nabla f(x)$在水平集$L = \{x \in R^n | f(x) \leq f(x_0)\}$上存在且一致连续，算法2.1 中选取的方向$p_k$与负梯度$-\nabla f(x_k)$的夹角$\theta_k$满足

$$\theta_k \leq \frac{\pi}{2} - \mu, \quad \text{对某个}\mu > 0$$

则或者对某个k有$\nabla f(x_k) = 0$，或者有$f(x_k) \to -\infty$，或者有$\nabla f(x_k) \to 0$.

定理：收敛速度

- 假设算法2.1产生的序列$\{x_k\}$收敛到$f(x)$的极小值点x^*.
- 如果$f(x)$在x^*的某个邻域内二次连续可微，且存在$\varepsilon > 0$和$M > m > 0$，使得当$\|x - x^*\| < \varepsilon$时，有 $m\|y\|^2 \leq y^T G(x) y \leq M\|y^2\|, \forall y \in R^n$,
- 则 $\{x_k\}$线性收敛.

0.618法、FIBONACCI法和二分法

- 基本思想： 通过取试探点进行函数值比较，使得包含极小值点的搜索区间不断缩短, 当区间长度缩短到一定程度时, 区间上个点均接近极小值.

0.618法、FIBONACCI法和二分法

- 基本思想： 通过取试探点进行函数值比较，使得包含极小值点的搜索区间不断缩短, 当区间长度缩短到一定程度时, 区间上个点均接近极小值. 仅需计算函数值，不需要计算导数值， 适用于非光滑及导数表达式复杂的或写不出的情形。

0.618法、Fibonacci法和二分法

- 基本思想： 通过取试探点进行函数值比较，使得包含极小值点的搜索区间不断缩短, 当区间长度缩短到一定程度时, 区间上个点均接近极小值. 仅需计算函数值，不需要计算导数值， 适用于非光滑及导数表达式复杂的或写不出的情形。
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$，是搜索区间$[a_1, b_1]$上的单峰函数.

0.618法、FIBONACCI法和二分法

- 基本思想： 通过取试探点进行函数值比较，使得包含极小值点的搜索区间不断缩短, 当区间长度缩短到一定程度时, 区间上个点均接近极小值. 仅需计算函数值，不需要计算导数值， 适用于非光滑及导数表达式复杂的或写不出的情形。
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$，是搜索区间$[a_1, b_1]$上的单峰函数.
- 假设在k次迭代时搜索区间为$[a_k, b_k]$. 取两个试探点$\lambda_k, \mu_k \in [a_k, b_k]$, 且$\lambda_k < \mu_k$,要求满足下列条件：

0.618法、FIBONACCI法和二分法

- 基本思想: 通过取试探点进行函数值比较，使得包含极小值点的搜索区间不断缩短, 当区间长度缩短到一定程度时, 区间上个点均接近极小值. 仅需计算函数值，不需要计算导数值，适用于非光滑及导数表达式复杂的或写不出的情形。
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$，是搜索区间$[a_1, b_1]$上的单峰函数.
- 假设在k次迭代时搜索区间为$[a_k, b_k]$. 取两个试探点$\lambda_k, \mu_k \in [a_k, b_k]$, 且$\lambda_k < \mu_k$,要求满足下列条件:
  1. λ_k和μ_k到搜索区间$[a_k, b_k]$两端点等距，即 $b_k - \lambda_k = \mu_k - a_k$.
  2. 每次迭代，搜索区间长度缩短率相同, 即$b_{k+1} - a_{k+1} = \tau(b_k - a_k)$.

0.618法、FIBONACCI法和二分法

- 基本思想：通过取试探点进行函数值比较，使得包含极小值点的搜索区间不断缩短, 当区间长度缩短到一定程度时, 区间上个点均接近极小值. 仅需计算函数值，不需要计算导数值，适用于非光滑及导数表达式复杂的或写不出的情形。
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$，是搜索区间$[a_1, b_1]$上的单峰函数.
- 假设在k次迭代时搜索区间为$[a_k, b_k]$. 取两个试探点$\lambda_k, \mu_k \in [a_k, b_k]$, 且$\lambda_k < \mu_k$,要求满足下列条件：
    1. λ_k和μ_k到搜索区间$[a_k, b_k]$两端点等距，即 $b_k - \lambda_k = \mu_k - a_k$.
    2. 每次迭代，搜索区间长度缩短率相同, 即$b_{k+1} - a_{k+1} = \tau(b_k - a_k)$.
- 如果$\varphi(\lambda_k) \le \varphi(\mu_k)$, 则令$a_{k+1} = a_k$, $b_{k+1} = \mu_k$.
  如果$\varphi(\lambda_k) > \varphi(\mu_k)$, 则令$a_{k+1} = \lambda_k$, $b_{k+1} = b_k$.

0.618法、FIBONACCI法和二分法

- 基本思想：通过取试探点进行函数值比较，使得包含极小值点的搜索区间不断缩短，当区间长度缩短到一定程度时，区间上个点均接近极小值．仅需计算函数值，不需要计算导数值，适用于非光滑及导数表达式复杂的或写不出的情形。
- 设$\varphi(\alpha) = f(x_k + \alpha p_k)$，是搜索区间$[a_1, b_1]$上的单峰函数.
- 假设在k次迭代时搜索区间为$[a_k, b_k]$. 取两个试探点$\lambda_k, \mu_k \in [a_k, b_k]$，且$\lambda_k < \mu_k$,要求满足下列条件：
  1. λ_k和μ_k到搜索区间$[a_k, b_k]$两端点等距，即 $b_k - \lambda_k = \mu_k - a_k$.
  2. 每次迭代，搜索区间长度缩短率相同，即$b_{k+1} - a_{k+1} = \tau(b_k - a_k)$.
- 如果$\varphi(\lambda_k) \le \varphi(\mu_k)$，则令$a_{k+1} = a_k$，$b_{k+1} = \mu_k$.
  如果$\varphi(\lambda_k) > \varphi(\mu_k)$，则令$a_{k+1} = \lambda_k$，$b_{k+1} = b_k$.
- $\tau = \frac{\sqrt{5}-1}{2} \approx 0.618$. (黄金分割法)
  $\lambda_k = a_k + 0.382(b_k - a_k)$, $\quad \mu_k = a_k + 0.618(b_k - a_k)$.

0.618法、FIBONACCI法和二分法

- Fibonacci法中τ不在是常数而是$\tau_k = \frac{F_{n-k}}{F_{n-k+1}}$, 其中

0.618法、Fibonacci法和二分法

- Fibonacci法中τ不在是常数而是$\tau_k = \frac{F_{n-k}}{F_{n-k+1}}$, 其中
- Fibonacci数列 $F_0 = F_1 = 1, \quad F_{k+1} = F_k + F_{k-1}, k = 1, 2 \cdots ,$.

0.618法、Fibonacci法和二分法

- Fibonacci法中τ不在是常数而是$\tau_k = \frac{F_{n-k}}{F_{n-k+1}}$, 其中
- Fibonacci数列 $F_0 = F_1 = 1, \quad F_{k+1} = F_k + F_{k-1}, \, k = 1, 2 \cdots ,$.
- $\lambda_k = a_k + (1 - \tau_k)(b_k - a_k) = a_k + \frac{F_{n-k-1}}{F_{n-k+1}}(b_k - a_k)$
  $\mu_k = a_k + \tau_k(b_k - a_k) = a_k + \frac{F_{n-k}}{F_{n-k+1}}(b_k - a_k)$

0.618法、FIBONACCI法和二分法

- Fibonacci法中τ不在是常数而是$\tau_k = \frac{F_{n-k}}{F_{n-k+1}}$, 其中
- Fibonacci数列 $F_0 = F_1 = 1, \quad F_{k+1} = F_k + F_{k-1}, k = 1, 2 \cdots$,.
- $\lambda_k = a_k + (1 - \tau_k)(b_k - a_k) = a_k + \frac{F_{n-k-1}}{F_{n-k+1}}(b_k - a_k)$
  $\mu_k = a_k + \tau_k(b_k - a_k) = a_k + \frac{F_{n-k}}{F_{n-k+1}}(b_k - a_k)$
- 假设$F_k \approx r^k$, 有 $r^{k+1} = r^k + r^{k-1}$ 可以推出 $r = \frac{\sqrt{5}-1}{2}$.即 Fibonacci法渐进行为就是黄金分割法.

0.618法、Fibonacci法和二分法

- Fibonacci法中τ不在是常数而是$\tau_k = \frac{F_{n-k}}{F_{n-k+1}}$, 其中
- Fibonacci数列 $F_0 = F_1 = 1,\quad F_{k+1} = F_k + F_{k-1}, k = 1, 2 \cdots ,$.
- $\lambda_k = a_k + (1 - \tau_k)(b_k - a_k) = a_k + \frac{F_{n-k-1}}{F_{n-k+1}}(b_k - a_k)$
  $\mu_k = a_k + \tau_k(b_k - a_k) = a_k + \frac{F_{n-k}}{F_{n-k+1}}(b_k - a_k)$
- 假设$F_k \approx r^k$, 有 $r^{k+1} = r^k + r^{k-1}$ 可以推出 $r = \frac{\sqrt{5}-1}{2}$.即 Fibonacci法渐进行为就是黄金分割法.
- 事实上, 可以证明Fibonacci法是分割方法求解一维极小化问题的最优策略, 而黄金分割法是近似最优法.

0.618法、FIBONACCI法和二分法

- Fibonacci法中τ不在是常数而是$\tau_k = \frac{F_{n-k}}{F_{n-k+1}}$, 其中
- Fibonacci数列 $F_0 = F_1 = 1, \quad F_{k+1} = F_k + F_{k-1}, k = 1, 2\cdots,$.
- $\lambda_k = a_k + (1-\tau_k)(b_k - a_k) = a_k + \frac{F_{n-k-1}}{F_{n-k+1}}(b_k - a_k)$
  $\mu_k = a_k + \tau_k(b_k - a_k) = a_k + \frac{F_{n-k}}{F_{n-k+1}}(b_k - a_k)$
- 假设$F_k \approx r^k$, 有 $r^{k+1} = r^k + r^{k-1}$ 可以推出 $r = \frac{\sqrt{5}-1}{2}$.即 Fibonacci法渐进行为就是黄金分割法.
- 事实上, 可以证明Fibonacci法是分割方法求解一维极小化问题的最优策略, 而黄金分割法是近似最优法.
- 二分法$\lambda_k = \mu_k = \frac{a_k+b_k}{2}$.

0.618法、Fibonacci法和二分法

- Fibonacci法中τ不在是常数而是$\tau_k = \frac{F_{n-k}}{F_{n-k+1}}$, 其中
- Fibonacci数列 $F_0 = F_1 = 1, \quad F_{k+1} = F_k + F_{k-1}, k = 1, 2 \cdots ,$.
- $\lambda_k = a_k + (1 - \tau_k)(b_k - a_k) = a_k + \frac{F_{n-k-1}}{F_{n-k+1}}(b_k - a_k)$
  $\mu_k = a_k + \tau_k(b_k - a_k) = a_k + \frac{F_{n-k}}{F_{n-k+1}}(b_k - a_k)$
- 假设$F_k \approx r^k$, 有 $r^{k+1} = r^k + r^{k-1}$ 可以推出 $r = \frac{\sqrt{5}-1}{2}$.即 Fibonacci法渐进行为就是黄金分割法.
- 事实上, 可以证明Fibonacci法是分割方法求解一维极小化问题的最优策略, 而黄金分割法是近似最优法.
- 二分法$\lambda_k = \mu_k = \frac{a_k+b_k}{2}$.
- 分割法都是线性收敛的方法。

插值法

- 基本思想： 在搜索区间中不断使用低次多项式来近似目标函数，并逐步用插值多项式的极小点来逼近一维搜索问题$\min\limits_{\alpha} \varphi(\alpha)$的极小点.

插值法

- 基本思想：在搜索区间中不断使用低次多项式来近似目标函数，并逐步用插值多项式的极小点来逼近一维搜索问题$\min_{\alpha} \varphi(\alpha)$的极小点.
- 当函数解析性质比较好时，插值法比分割法效果更好.

插值法

- 基本思想： 在搜索区间中不断使用低次多项式来近似目标函数，并逐步用插值多项式的极小点来逼近一维搜索问题$\min_{\alpha} \varphi(\alpha)$的极小点.
- 当函数解析性质比较好时，插值法比分割法效果更好.
- 二次插值法（单点，二点，三点），局部二阶收敛、超线性收敛

插值法

- 基本思想：在搜索区间中不断使用低次多项式来近似目标函数，并逐步用插值多项式的极小点来逼近一维搜索问题$\min_{\alpha} \varphi(\alpha)$的极小点.
- 当函数解析性质比较好时，插值法比分割法效果更好.
- 二次插值法（单点，二点，三点），局部二阶收敛、超线性收敛
- 三次插值法（二点），局部二阶收敛

单点插值法(牛顿法)

- 考虑利用某一点处的函数值、一阶导数值、二阶导数值构造二次函数

单点插值法(牛顿法)

- 考虑利用某一点处的函数值、一阶导数值、二阶导数值构造二次函数
- 设$q(\alpha) = a\alpha^2 + b\alpha + c$
  满足 $q(\alpha_1) = \varphi(\alpha_1)$, $q'(\alpha_1) = \varphi'(\alpha_1)$, $q''(\alpha_1) = \varphi''(\alpha_1)$.

单点插值法(牛顿法)

- 考虑利用某一点处的函数值、一阶导数值、二阶导数值构造二次函数
- 设$q(\alpha) = a\alpha^2 + b\alpha + c$
  满足 $q(\alpha_1) = \varphi(\alpha_1)$, $q'(\alpha_1) = \varphi'(\alpha_1)$, $q''(\alpha_1) = \varphi''(\alpha_1)$.
- 直接求解$q(\alpha)$的最小值可得：$\bar{\alpha} = -\frac{b}{2a} = \alpha_1 - \frac{\varphi'(\alpha_1)}{\varphi''(\alpha_1)}$.

单点插值法(牛顿法)

- 考虑利用某一点处的函数值、一阶导数值、二阶导数值构造二次函数
- 设$q(\alpha) = a\alpha^2 + b\alpha + c$
  满足 $q(\alpha_1) = \varphi(\alpha_1)$, $q'(\alpha_1) = \varphi'(\alpha_1)$, $q''(\alpha_1) = \varphi''(\alpha_1)$.
- 直接求解$q(\alpha)$的最小值可得：$\bar{\alpha} = -\frac{b}{2a} = \alpha_1 - \frac{\varphi'(\alpha_1)}{\varphi''(\alpha_1)}$.
- 本质上是牛顿法。（具有局部的二次收敛性）

单点插值法(牛顿法)

定理(牛顿迭代法的局部二次收敛性)

假设$\varphi: R \to R$, $\varphi \in C^2$, $\varphi'(\alpha^*) = 0$, $\varphi''(\alpha^*) \neq 0$, 则当初始点$\alpha_0$比较靠近$\alpha^*$时, 由牛顿迭代法产生的序列

$$\alpha_{k+1} = \alpha_k - (\varphi''(\alpha_k))^{-1}\varphi'(\alpha_k), \quad k = 0, 1, 2, \cdots$$

是收敛的, 即$\alpha_k \to \alpha^*$. 如果$\varphi \in C^3$, 则

$$\lim_{k\to\infty} \frac{|\alpha_{k+1} - \alpha^*|}{|\alpha_k - \alpha^*|^2} = |\frac{1}{2}\varphi''(\alpha^*)^{-1}\varphi'''(\alpha^*)|,$$

这表明$|\alpha_{k+1} - \alpha^*| = \mathcal{O}(|\alpha_k - \alpha^*|^2)$.

不精确一维搜索法

- 一维搜索是最优化方法的基本组成部分
- 精确的一维搜索花费巨大
- 很多最优化方法，例如牛顿法/拟牛顿法，收敛速度不依赖于精确一维搜索过程

不精确一维搜索法

Armijo condition: 首先保证 α_k 能够使目标函数f产生足够下降 sufficient decrease

$$f(x_k + \alpha p_k) \le f(x_k) + c_1 \alpha \nabla^T(x_k) p_k \tag{2.1}$$

for some constant $c_1 \in (0, 1)$. In practice, c_1 is chosen to be quite small, say $c_1 = 10^{-4}$.

不精确一维搜索法

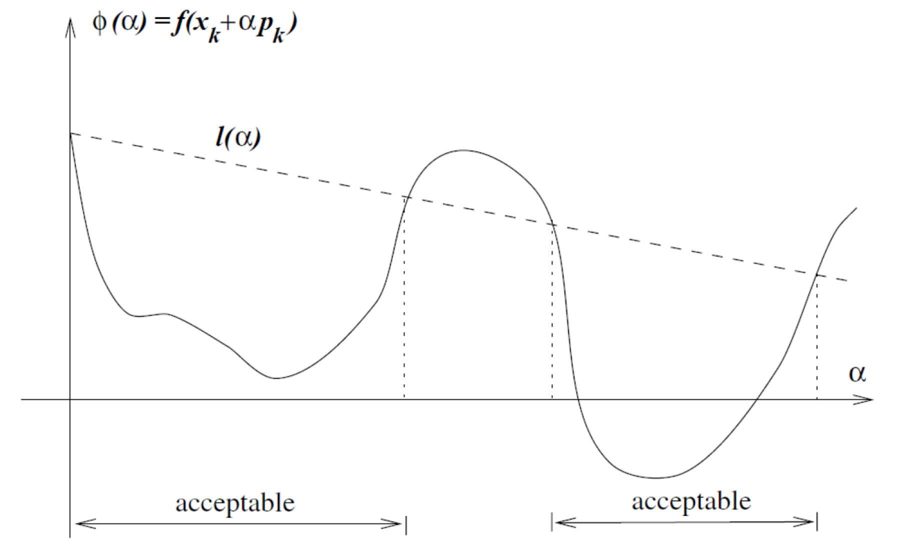
Armijo condition: 首先保证 α_k 能够使目标函数f产生足够下降 sufficient decrease

$$f(x_k + \alpha p_k) \leq f(x_k) + c_1 \alpha \nabla^T (x_k) p_k \tag{2.1}$$

for some constant $c_1 \in (0, 1)$. In practice, c_1 is chosen to be quite small, say $c_1 = 10^{-4}$.

(2.1) means that the reduction in f should be proportional to both the step length α_k and the directional derivative $\nabla f^T (x_k) p_k$.

Demo: Sufficient Decrease Condition



The Wolfe Condition

The Wolfe Condition

- The sufficient decrease condition is not enough by itself to ensure that the algorithm makes reasonable progress because it is satisfied for all sufficiently small α.

The Wolfe Condition

- The sufficient decrease condition is not enough by itself to ensure that the algorithm makes reasonable progress because it is satisfied for all sufficiently small α.
- To rule out unacceptably short steps we introduce a second requirement, called the *curvature condition*, which requires α_k to satisfy

$$\big(\nabla f(x_k + \alpha_k p_k)\big)^T p_k \geq c_2(\nabla f(x_k))^T p_k \tag{2.2}$$

for some constant $c_2 \in (c_1, 1)$, where c_1 (通常很小) is the constant from (2.1), i.e.,

$$f(x_k + \alpha p_k) \leq f(x_k) + c_1 \alpha \nabla^T(x_k) p_k$$

The Wolfe Condition

- The sufficient decrease condition is not enough by itself to ensure that the algorithm makes reasonable progress because it is satisfied for all sufficiently small α.
- To rule out unacceptably short steps we introduce a second requirement, called the *curvature condition*, which requires α_k to satisfy

$$\left(\nabla f(x_k+\alpha_k p_k)\right)^T p_k \ge c_2(\nabla f(x_k))^T p_k \tag{2.2}$$

  for some constant $c_2 \in (c_1, 1)$, where c_1 (通常很小) is the constant from (2.1), i.e.,

$$f(x_k+\alpha p_k) \le f(x_k) + c_1\alpha\nabla^T(x_k)p_k$$

- Typical values of $c_2 \approx 0.9$ when the search direction p_k is chosen by a Newton or quasi-Newton method, or $c_2 \approx 0.1$ when p_k is obtained from a nonlinear conjugate gradient method.

The Wolfe Condition

The Wolfe Condition

- Note that the left-hand-side is simply the derivative $\phi'(\alpha_k)$, so the curvature condition ensures that the slope of ϕ at α_k is greater than c_2 times the initial step slope $\phi'(0)$, i.e., $\phi'(\alpha_k) \geq c_2\phi'(0)$.

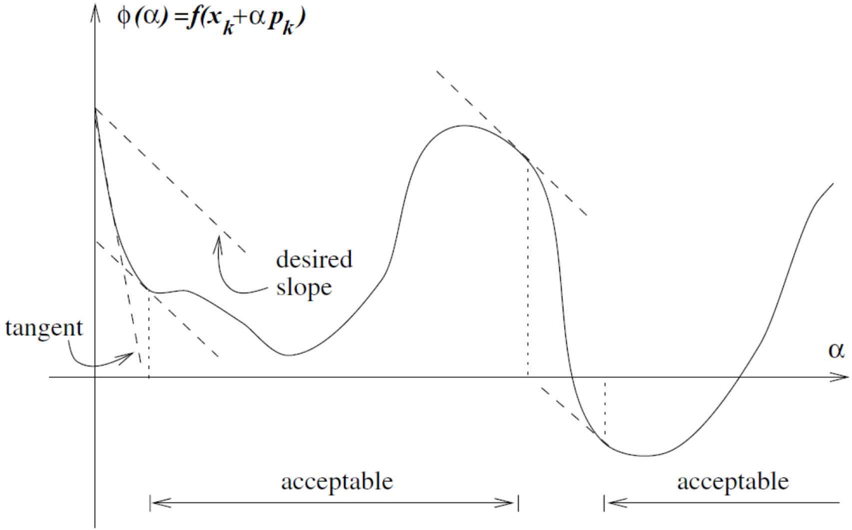
THE WOLFE CONDITION

- Note that the left-hand-side is simply the derivative $\phi'(\alpha_k)$, so the curvature condition ensures that the slope of ϕ at α_k is greater than c_2 times the initial step slope $\phi'(0)$, i.e., $\phi'(\alpha_k) \geq c_2\phi'(0)$.
- This make sense because if the slope $\phi'(\alpha)$ is strongly negatives, we have indication that we can reduce f significantly by moving further along the chosen direction.

The Wolfe Condition

- Note that the left-hand-side is simply the derivative $\phi'(\alpha_k)$, so the curvature condition ensures that the slope of ϕ at α_k is greater than c_2 times the initial step slope $\phi'(0)$, i.e., $\phi'(\alpha_k) \geq c_2\phi'(0)$.
- This make sense because if the slope $\phi'(\alpha)$ is strongly negatives, we have indication that we can reduce f significantly by moving further along the chosen direction.
- On the other hand, if $\phi'(\alpha_k)$ is only slightly negative or even positive, it is a sign that we cannot expect much more decrease in f in this direction, so it makes sense to terminate the line search.

THE WOLFE CONDITION



The Wolfe Condition

The sufficient decrease and the curvature conditions are known collectively as the Wolfe conditions. We restate them here for future reference:

The Wolfe Condition

The sufficient decrease and the curvature conditions are known collectively as the Wolfe conditions. We restate them here for future reference:

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k (\nabla f(x_k))^T p_k \tag{2.3a}$$

$$(\nabla f(x_k + \alpha_k p_k))^T p_k \geq c_2 (\nabla f(x_k))^T p_k \tag{2.3b}$$

The Wolfe Condition

The sufficient decrease and the curvature conditions are known collectively as the Wolfe conditions. We restate them here for future reference:

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k (\nabla f(x_k))^T p_k \tag{2.3a}$$

$$(\nabla f(x_k + \alpha_k p_k))^T p_k \geq c_2 (\nabla f(x_k))^T p_k \tag{2.3b}$$

The Wolfe conditions are scale-invariant in a broad sense:

The Wolfe Condition

The sufficient decrease and the curvature conditions are known collectively as the Wolfe conditions. We restate them here for future reference:

$$f(x_k + \alpha_k p_k) \le f(x_k) + c_1 \alpha_k (\nabla f(x_k))^T p_k \tag{2.3a}$$

$$(\nabla f(x_k + \alpha_k p_k))^T p_k \ge c_2 (\nabla f(x_k))^T p_k \tag{2.3b}$$

The Wolfe conditions are scale-invariant in a broad sense:

- Multiplying the objective function by a constant or making an affine change of variables does not alter them.

The Wolfe Condition

The sufficient decrease and the curvature conditions are known collectively as the Wolfe conditions. We restate them here for future reference:

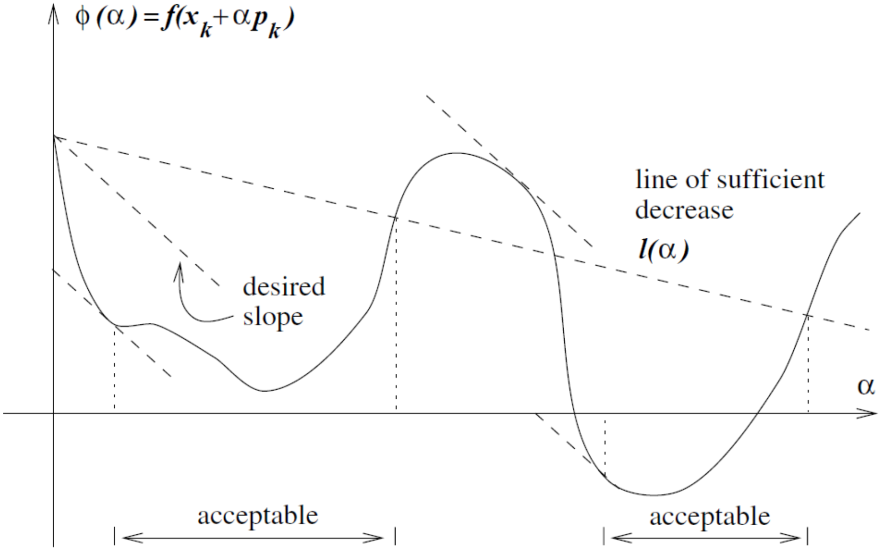
$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k (\nabla f(x_k))^T p_k \qquad (2.3a)$$

$$(\nabla f(x_k + \alpha_k p_k))^T p_k \geq c_2 (\nabla f(x_k))^T p_k \qquad (2.3b)$$

The Wolfe conditions are scale-invariant in a broad sense:

- Multiplying the objective function by a constant or making an affine change of variables does not alter them.
- They can be used in most line search methods, and are particularly important in the implementation of quasi-Newton methods.

THE WOLFE CONDITION



Strong Wolfe Condition

- A step length may satisfy the Wolfe conditions without being particularly close to a minimizer of ϕ.

Strong Wolfe Condition

- A step length may satisfy the Wolfe conditions without being particularly close to a minimizer of ϕ.
- We can, however, modify the curvature condition to force α_k to lie in at least a broad neighborhood of a local minimizer or stationary point of ϕ.

Strong Wolfe Condition

- A step length may satisfy the Wolfe conditions without being particularly close to a minimizer of ϕ.
- We can, however, modify the curvature condition to force α_k to lie in at least a broad neighborhood of a local minimizer or stationary point of ϕ.
- The strong Wolfe conditions require α_k to satisfy

$$f(x_k + \alpha_k p_k) \le f(x_k) + c_1 \alpha_k (\nabla f(x_k))^T p_k \tag{2.4a}$$

$$|(\nabla f(x_k + \alpha_k p_k))^T p_k| \le c_2 |(\nabla f(x_k))^T p_k| \tag{2.4b}$$

with $0 < c_1 < c_2 < 1$.

STRONG WOLFE CONDITION

- A step length may satisfy the Wolfe conditions without being particularly close to a minimizer of ϕ.
- We can, however, modify the curvature condition to force α_k to lie in at least a broad neighborhood of a local minimizer or stationary point of ϕ.
- The strong Wolfe conditions require α_k to satisfy

$$f(x_k + \alpha_k p_k) \le f(x_k) + c_1 \alpha_k (\nabla f(x_k))^T p_k \tag{2.4a}$$

$$|(\nabla f(x_k + \alpha_k p_k))^T p_k| \le c_2 |(\nabla f(x_k))^T p_k| \tag{2.4b}$$

  with $0 < c_1 < c_2 < 1$.
- The only difference with the Wolfe condition is that we no longer allow the derivative $\phi'(\alpha_k)$ to be too positive. Hence, we exclude points that are far from stationary points of ϕ.

The Wolfe Condition

The following theorem shows that there exist step lengths that satisfy the Wolfe conditions for every function f that is smooth and bounded below.

Theorem

Suppose that $f : \mathcal{R}^n \to \mathcal{R}$ is continuously differentiable. Let p_k be a descent direction at x_k , and assume that f is bounded below along the ray $\{x_k + \alpha p_k | \alpha > 0\}$. Then if $0 < c_1 < c_2 < 1$, there exist intervals of step lengths satisfy the Wolfe conditions (2.3) *and the strong Wolfe conditions* (2.4).

The Goldstein Condition

The Goldstein conditions ensure that the step length α achieves sufficient decrease but is not too short:

$$f(x_k) + (1-c)\alpha_k(\nabla f(x_k))^T p_k \leq f(x_k + \alpha_k p_k) \leq f(x_k) + c\alpha_k(\nabla f(x_k))^T p_k, \tag{2.5}$$

with $0 < c < \frac{1}{2}$.

The Goldstein Condition

The Goldstein conditions ensure that the step length α achieves sufficient decrease but is not too short:

$$f(x_k) + (1-c)\alpha_k(\nabla f(x_k))^T p_k \le f(x_k + \alpha_k p_k) \le f(x_k) + c\alpha_k(\nabla f(x_k))^T p_k, \tag{2.5}$$

with $0 < c < \frac{1}{2}$.

- The second equality is the sufficient decrease condition (2.1)

THE GOLDSTEIN CONDITION

The Goldstein conditions ensure that the step length α achieves sufficient decrease but is not too short:

$$f(x_k) + (1-c)\alpha_k(\nabla f(x_k))^T p_k \leq f(x_k + \alpha_k p_k) \leq f(x_k) + c\alpha_k(\nabla f(x_k))^T p_k, \tag{2.5}$$

with $0 < c < \frac{1}{2}$.

- The second equality is the sufficient decrease condition (2.1)
- The first inequality is introduced to control the step length from below.

The Goldstein Condition

- A disadvantage of the Goldstein conditions vs the Wolfe conditions is that the first inequality in (2.5) may exclude all minimizer of ϕ.

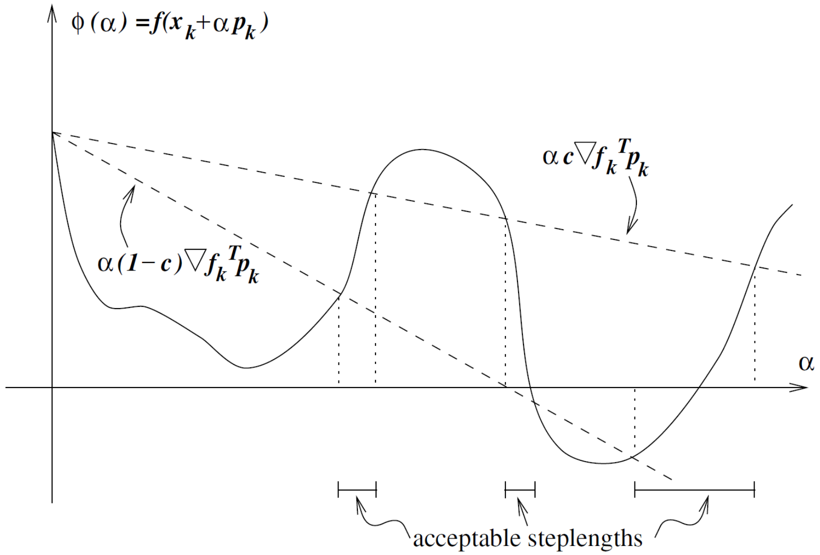
The Goldstein Condition

- A disadvantage of the Goldstein conditions vs the Wolfe conditions is that the first inequality in (2.5) may exclude all minimizer of ϕ.
- However, the Goldstein and Wolfe conditions have much in common and their convergence theories are quite similar.

The Goldstein Condition

- A disadvantage of the Goldstein conditions vs the Wolfe conditions is that the first inequality in (2.5) may exclude all minimizer of ϕ.
- However, the Goldstein and Wolfe conditions have much in common and their convergence theories are quite similar.
- The Goldstein conditions are often used in Newton-type methods but are no well suited for quasi-Newton methods, which maintain a positive definite Hessian approximation.

The Goldstein Condition



Sufficient Decrease and Backtracking

Algorithm Backtracking Line Search (回溯线搜索)

Choose $\bar{\alpha} > 0$, $\rho \in (0,1)$, $c \in (0,1)$, Set $\alpha \leftarrow \bar{\alpha}$;
Do until $f(x_k + \alpha p_k) \le f(x_k) + c\alpha(\nabla f(x_k))^T p_k$

$$\alpha \leftarrow \rho\alpha;$$

End(do)
Terminate with $\alpha_k = \alpha$

Sufficient Decrease and Backtracking

- In this procedure, the initial step length $\bar{\alpha}$ is chosen to be 1 in Newton and quasi-Newton methods (牛顿法或拟牛顿法), but can have different values in other algorithms such as steepest descent or conjugate gradient (最速下降法或共轭梯度法).
- An acceptable step length will be found after a finite number of trials (有限步停止), because α_k will eventually become small enough that the sufficient decrease condition holds.
- In practice, the contraction factor ρ (ρ_k) is often allowed to vary at each iteration of the line search.
- For example, it can be chosen by safeguarded interpolation. We need ensure only that at each iteration we have $\rho \in [\rho_{low}, \rho_{hi}]$, for some fixed constants $0 < \rho_{low} < \rho_{hi} < 1$.

Sufficient Decrease and Backtracking

- The backtracking approach ensures either that the selected step length α_k is some fixed value (the initial choice $\bar{\alpha}$), or else that it is short enough to satisfy the sufficient decrease condition but not too short.

SUFFICIENT DECREASE AND BACKTRACKING

- The backtracking approach ensures either that the selected step length α_k is some fixed value (the initial choice $\bar{\alpha}$), or else that it is short enough to satisfy the sufficient decrease condition but not too short.
- The latter claim holds because the accepted value α_k is within a factor ρ of the previous trial value, α_k/ρ, which was rejected for violating the sufficient decrease condition, that is, for being too long.

Sufficient Decrease and Backtracking

- The backtracking approach ensures either that the selected step length α_k is some fixed value (the initial choice $\bar{\alpha}$), or else that it is short enough to satisfy the sufficient decrease condition but not too short.
- The latter claim holds because the accepted value α_k is within a factor ρ of the previous trial value, α_k/ρ, which was rejected for violating the sufficient decrease condition, that is, for being too long.
- This simple and popular strategy for terminating a line strategy for terminating a line search is well suited for Newton methods but is less appropriate for quasi-Newton and conjugate gradient methods.

Step-Length Selection Algorithms

We now consider techniques for finding a minimum of the one-dimensional function

$$\phi(\alpha) = f(x_k + \alpha p_k) \tag{2.6}$$

or for simply finding a step length α_k satisfying one of the termination conditions we described. （包括Wolfe条件和Goldstein条件）

Step-Length Selection Algorithms

- If f is a convex quadratic function $f(x) = \frac{1}{2}x^T Qx - b^T x$, its one-dimensional minimizer along the ray $x_k + \alpha p_k$ can be computed analytically and is given by

$$\alpha_k = \frac{(\nabla f(x_k))^T p_k}{p_k Q p_k}$$

STEP-LENGTH SELECTION ALGORITHMS

- If f is a convex quadratic function $f(x) = \frac{1}{2}x^T Qx - b^T x$, its one-dimensional minimizer along the ray $x_k + \alpha p_k$ can be computed analytically and is given by

$$\alpha_k = \frac{(\nabla f(x_k))^T p_k}{p_k Q p_k}$$

- For general nonlinear functions, it is necessary to use an iterative procedure.

Step-Length Selection Algorithms

All the line search procedures requires an initial estimate α_0 and generate a sequence α_k that:

- terminates with a step length satisfied by the user (for example, the Wolfe conditions)
- or determines that such a step length does not exist.

Step-Length Selection Algorithms

All the line search procedures requires an initial estimate α_0 and generate a sequence α_k that:

- terminates with a step length satisfied by the user (for example, the Wolfe conditions)
- or determines that such a step length does not exist.

Typical procedure consist of two phases:

- a bracketing phase that finds an interval $[\bar{a}, \bar{b}]$ containing acceptable step lengths
- a selection phase that zooms in to locate the final step length.

INTERPOLATION

The selection phase usually

- reduces the bracketing interval during its search for the desired length
- interpolates 插值 some of the the function and derivative information gathered on earlier steps to guess the location of the minimizer.

Interpolation

The selection phase usually

- reduces the bracketing interval during its search for the desired length
- interpolates 插值 some of the the function and derivative information gathered on earlier steps to guess the location of the minimizer.

Reduce the bracketing interval

Interpolation

The selection phase usually

- reduces the bracketing interval during its search for the desired length
- interpolates 插值 some of the the function and derivative information gathered on earlier steps to guess the location of the minimizer.

Reduce the bracketing interval

- Rewrite the sufficient decrease condition in the notation of (2.6) as

$$\phi(\alpha_k) \leq \phi(0) + c_1 \alpha_k \phi(0) \tag{2.7}$$

Interpolation

The selection phase usually

- reduces the bracketing interval during its search for the desired length
- interpolates 插值 some of the the function and derivative information gathered on earlier steps to guess the location of the minimizer.

Reduce the bracketing interval

- Rewrite the sufficient decrease condition in the notation of (2.6) as

$$\phi(\alpha_k) \leq \phi(0) + c_1 \alpha_k \phi(0) \tag{2.7}$$

- Suppose that the initial guess α_0 is given. If we have

$$\phi(\alpha_0) \leq \phi(0) + c_1 \alpha_0 \phi(0) \tag{2.8}$$

this step length satisfies the condition, and we terminate the search.

INTERPOLATION

The selection phase usually

- reduces the bracketing interval during its search for the desired length
- interpolates 插值 some of the the function and derivative information gathered on earlier steps to guess the location of the minimizer.

Reduce the bracketing interval

- Rewrite the sufficient decrease condition in the notation of (2.6) as

$$\phi(\alpha_k) \leq \phi(0) + c_1 \alpha_k \phi(0) \tag{2.7}$$

- Suppose that the initial guess α_0 is given. If we have

$$\phi(\alpha_0) \leq \phi(0) + c_1 \alpha_0 \phi(0) \tag{2.8}$$

  this step length satisfies the condition, and we terminate the search.

- Otherwise, we know that the interval $[0, \alpha_0]$ contains acceptable step length.

INTERPOLATION

Interpolation

- We construct a quadratic approximation $\phi_q(\alpha)$ to approach ϕ so that it satisfies the interpolation conditions $\phi_q(0) = \phi(0)$, $\phi_q'(0) = \phi'(0)$, and $\phi_q(\alpha_0) = \phi(\alpha_0)$ as follow:

$$\phi_q(\alpha) = \Big(\frac{\phi(\alpha_0) - \phi(0) - \alpha_0\phi'(0)}{\alpha_0^2}\Big)\alpha^2 + \phi'(0)\alpha + \phi(0)$$

Interpolation

Interpolation

- We construct a quadratic approximation $\phi_q(\alpha)$ to approach ϕ so that it satisfies the interpolation conditions $\phi_q(0) = \phi(0)$, $\phi_q'(0) = \phi'(0)$, and $\phi_q(\alpha_0) = \phi(\alpha_0)$ as follow:

$$\phi_q(\alpha) = \Big(\frac{\phi(\alpha_0) - \phi(0) - \alpha_0\phi'(0)}{\alpha_0^2}\Big)\alpha^2 + \phi'(0)\alpha + \phi(0)$$

- The new trial value α_1 is defined as the minimizer of this quadratic, that is

$$\alpha_1 = -\frac{\phi'(0)\alpha_0^2}{2[\phi(\alpha_0) - \phi(0) - \phi'(0)\alpha_0]}$$

Interpolation

Interpolation

- We construct a quadratic approximation $\phi_q(\alpha)$ to approach ϕ so that it satisfies the interpolation conditions $\phi_q(0) = \phi(0)$, $\phi_q'(0) = \phi'(0)$, and $\phi_q(\alpha_0) = \phi(\alpha_0)$ as follow:

$$\phi_q(\alpha) = \Big(\frac{\phi(\alpha_0) - \phi(0) - \alpha_0\phi'(0)}{\alpha_0^2}\Big)\alpha^2 + \phi'(0)\alpha + \phi(0)$$

- The new trial value α_1 is defined as the minimizer of this quadratic, that is

$$\alpha_1 = -\frac{\phi'(0)\alpha_0^2}{2[\phi(\alpha_0) - \phi(0) - \phi'(0)\alpha_0]}$$

- If the sufficient decrease condition is satisfied at α_1, we terminate the search. Otherwise...

Interpolation

- Otherwise, we construct a cubic function that satisfies $\phi_c(0) = \phi(0), \phi_c'(0) = \phi'(0), \phi_c(\alpha_0) = \phi(\alpha_0)$ and $\phi_c(\alpha_1) = \phi(\alpha_1)$ as follow:

$$\phi_c(\alpha) = a\alpha^3 + b\alpha^2 + \phi'(0)\alpha + \phi(0),$$

Interpolation

- Otherwise, we construct a cubic function that satisfies $\phi_c(0) = \phi(0), \phi_c'(0) = \phi'(0), \phi_c(\alpha_0) = \phi(\alpha_0)$ and $\phi_c(\alpha_1) = \phi(\alpha_1)$ as follow:

$$\phi_c(\alpha) = a\alpha^3 + b\alpha^2 + \phi'(0)\alpha + \phi(0),$$

where

$$\begin{pmatrix} a \\ b \end{pmatrix} = \frac{1}{\alpha_0^2\alpha_1^2(\alpha_1 - \alpha_0)} \begin{pmatrix} \alpha_0^2 & -\alpha_1^2 \\ -\alpha_0^3 & \alpha_1^3 \end{pmatrix} \begin{pmatrix} \phi(\alpha_1) - \phi(0) - \phi'(0)\alpha_1 \\ \phi(\alpha_0) - \phi(0) - \phi'(0)\alpha_0 \end{pmatrix}$$

INTERPOLATION

- Otherwise, we construct a cubic function that satisfies $\phi_c(0) = \phi(0), \phi_c'(0) = \phi'(0), \phi_c(\alpha_0) = \phi(\alpha_0)$ and $\phi_c(\alpha_1) = \phi(\alpha_1)$ as follow:

$$\phi_c(\alpha) = a\alpha^3 + b\alpha^2 + \phi'(0)\alpha + \phi(0),$$

where

$$\begin{pmatrix} a \\ b \end{pmatrix} = \frac{1}{\alpha_0^2\alpha_1^2(\alpha_1 - \alpha_0)} \begin{pmatrix} \alpha_0^2 & -\alpha_1^2 \\ -\alpha_0^3 & \alpha_1^3 \end{pmatrix} \begin{pmatrix} \phi(\alpha_1) - \phi(0) - \phi'(0)\alpha_1 \\ \phi(\alpha_0) - \phi(0) - \phi'(0)\alpha_0 \end{pmatrix}$$

- By differentiating $\phi_c(x)$, we see that the minimizer α_2 of ϕ_c lies in the interval $[0, \alpha_1]$ and is given by

$$\alpha_2 = \frac{-b + \sqrt{b^2 - 3a\phi'(0)}}{3a}.$$

Interpolation

INTERPOLATION

- If necessary, above process is repeated, using a cubic interpolant of $\phi(0)$, $\phi'(0)$ and the two most recent values of ϕ, until an α that satisfies the sufficient decrease condition is located.

INTERPOLATION

- If necessary, above process is repeated, using a cubic interpolant of $\phi(0)$, $\phi'(0)$ and the two most recent values of ϕ, until an α that satisfies the sufficient decrease condition is located.
- If the computation of directional derivative can be done simultaneously with the function at little cost, we can design an alternative strategy based on cubic interpolation of the value of ϕ and ϕ' at the most recent values of α. (即使用$\phi(\alpha_k)$, $\phi'(\alpha_k)$, $\phi(\alpha_{k+1})$, $\phi'(\alpha_{k+1})$计算α_{k+2}).

INTERPOLATION

- If necessary, above process is repeated, using a cubic interpolant of $\phi(0)$, $\phi'(0)$ and the two most recent values of ϕ, until an α that satisfies the sufficient decrease condition is located.
- If the computation of directional derivative can be done simultaneously with the function at little cost, we can design an alternative strategy based on cubic interpolation of the value of ϕ and ϕ' at the most recent values of α. (即使用$\phi(\alpha_k)$, $\phi'(\alpha_k)$, $\phi(\alpha_{k+1})$, $\phi'(\alpha_{k+1})$计算α_{k+2}).
- Advantages: Cubic interpolation provides a good model for functions with significant changes of curvature and usually produces a quadratic rate of convergence of the iteration to the minimizing value of α.

Initial Step Length

- For Newton and quasi-Newton methods the step $\alpha_0 = 1$ should always be used as the initial trial step length.

INITIAL STEP LENGTH

- For Newton and quasi-Newton methods the step $\alpha_0 = 1$ should always be used as the initial trial step length.
- This choice ensures that unit step lengths are taken whenever they satisfy the termination conditions and allows the rapid rate-of-convergence properties of these methods to take effect.

Initial Step Length

- For Newton and quasi-Newton methods the step $\alpha_0 = 1$ should always be used as the initial trial step length.
- This choice ensures that unit step lengths are taken whenever they satisfy the termination conditions and allows the rapid rate-of-convergence properties of these methods to take effect.
- For methods that do not produce well-scaled search directions, such as the steepest descent and conjugate gradient methods, it is important to use current information about the problem and the algorithm to make the initial guess.

Initial Step Length

- A popular strategy

INITIAL STEP LENGTH

- A popular strategy is to assume that the first-order change in the function at iterate x_k will be the same as that obtained at the previous step.

INITIAL STEP LENGTH

- A popular strategy is to assume that the first-order change in the function at iterate x_k will be the same as that obtained at the previous step.
  In other words, we choose the initial guess α_0, so that $\alpha_0 \nabla f(x_k)^T p_k = \alpha_{k-1} \nabla f(x_{k-1})^T p_{k-1}$, that is,

$$\alpha_0 = \alpha_{k-1} \frac{\nabla f(x_{k-1})^T p_{k-1}}{\nabla f(x_k)^T p_k} \tag{2.9}$$

Initial Step Length

- Another useful strategy:

Initial Step Length

- Another useful strategy: interpolate a quadratic to the data $f(x_{k-1}), f(x_k)$, and $\phi'(0) = \nabla f(x_{k-1})^T p_{k-1}$ and define α_0 to be its minimizer.

INITIAL STEP LENGTH

- Another useful strategy: interpolate a quadratic to the data $f(x_{k-1}), f(x_k)$, and $\phi'(0) = \nabla f(x_{k-1})^T p_{k-1}$ and define α_0 to be its minimizer.
- This strategy yields

$$\alpha_0 = \frac{2(f(x_k) - f(x_{k-1}))}{\phi'(0)} \tag{2.10}$$

Initial Step Length

- Another useful strategy: interpolate a quadratic to the data $f(x_{k-1}), f(x_k)$, and $\phi'(0) = \nabla f(x_{k-1})^T p_{k-1}$ and define α_0 to be its minimizer.
- This strategy yields

$$\alpha_0 = \frac{2(f(x_k) - f(x_{k-1}))}{\phi'(0)} \tag{2.10}$$

- It can be shown that if $x_k \to x^*$ superlinearly, then the ratio in this expression converges to 1.

INITIAL STEP LENGTH

- Another useful strategy: interpolate a quadratic to the data $f(x_{k-1}), f(x_k)$, and $\phi'(0) = \nabla f(x_{k-1})^T p_{k-1}$ and define α_0 to be its minimizer.
- This strategy yields

$$\alpha_0 = \frac{2(f(x_k) - f(x_{k-1}))}{\phi'(0)} \tag{2.10}$$

- It can be shown that if $x_k \to x^*$ superlinearly, then the ratio in this expression converges to 1. If we adjust the choice (2.10) by setting

$$\alpha_0 \leftarrow \min(1, 1.01\alpha_0)$$

INITIAL STEP LENGTH

- Another useful strategy: interpolate a quadratic to the data $f(x_{k-1}), f(x_k)$, and $\phi'(0) = \nabla f(x_{k-1})^T p_{k-1}$ and define α_0 to be its minimizer.
- This strategy yields

$$\alpha_0 = \frac{2(f(x_k) - f(x_{k-1}))}{\phi'(0)} \tag{2.10}$$

- It can be shown that if $x_k \to x^*$ superlinearly, then the ratio in this expression converges to 1. If we adjust the choice (2.10) by setting

$$\alpha_0 \leftarrow \min(1, 1.01\alpha_0)$$

we find that the unit step length $\alpha_0 = 1$ will eventually always be tried and accepted, and the superlinear convergence properties of Newton and quasi-Newton methods will be observed.

A Line Search Algorithm

Algorithm 1: (Line Search Algorithm for Wolfe Conditions)

Set $\alpha_0 \leftarrow 0$, choose $\alpha_{\max} > 0$ and $\alpha_1 \in (0, \alpha_{\max})$, $i \leftarrow 1$

A LINE SEARCH ALGORITHM

ALGORITHM 1: (Line Search Algorithm for Wolfe Conditions)

Set $\alpha_0 \leftarrow 0$, choose $\alpha_{\max} > 0$ and $\alpha_1 \in (0, \alpha_{\max})$, $i \leftarrow 1$

Repeat

Evaluate $\phi(\alpha_i)$;

A Line Search Algorithm

Algorithm 1: (Line Search Algorithm for Wolfe Conditions)

Set $\alpha_0 \leftarrow 0$, choose $\alpha_{\max} > 0$ and $\alpha_1 \in (0, \alpha_{\max})$, $i \leftarrow 1$

Repeat

Evaluate $\phi(\alpha_i)$;

If $\phi(\alpha_i) > \phi(0) + c_1\alpha_i\phi'(0)$ **or** $[\phi(\alpha_i) \geq \phi(\alpha_{i-1})$ and $i > 1\ ]$

Set $\alpha_* \leftarrow$ **zoom**(α_{i-1}, α_i) and **stop**

A LINE SEARCH ALGORITHM

ALGORITHM 1: (Line Search Algorithm for Wolfe Conditions)

Set $\alpha_0 \leftarrow 0$, choose $\alpha_{\max} > 0$ and $\alpha_1 \in (0, \alpha_{\max})$, $i \leftarrow 1$
Repeat
  Evaluate $\phi(\alpha_i)$;
    If $\phi(\alpha_i) > \phi(0) + c_1 \alpha_i \phi'(0)$ **or** $[\phi(\alpha_i) \geq \phi(\alpha_{i-1})$ and $i > 1\ ]$
      Set $\alpha_* \leftarrow$ **zoom**(α_{i-1}, α_i) and **stop**
  Evaluate $\phi'(\alpha_i)$;

A Line Search Algorithm

Algorithm 1: (Line Search Algorithm for Wolfe Conditions)

Set $\alpha_0 \leftarrow 0$, choose $\alpha_{\max} > 0$ and $\alpha_1 \in (0, \alpha_{\max})$, $i \leftarrow 1$

Repeat

  Evaluate $\phi(\alpha_i)$;

   If $\phi(\alpha_i) > \phi(0) + c_1 \alpha_i \phi'(0)$ **or** $[\phi(\alpha_i) \geq \phi(\alpha_{i-1})$ and $i > 1\ ]$

    Set $\alpha_* \leftarrow$ **zoom**(α_{i-1}, α_i) and **stop**

  Evaluate $\phi'(\alpha_i)$;

   If $|\phi'(\alpha_i)| \leq -c_2 \phi'(0)$

    Set $\alpha_* \leftarrow \alpha_i$ and **stop**;

A Line Search Algorithm

Algorithm 1: (Line Search Algorithm for Wolfe Conditions)

Set $\alpha_0 \leftarrow 0$, choose $\alpha_{\max} > 0$ and $\alpha_1 \in (0, \alpha_{\max})$, $i \leftarrow 1$
Repeat
  Evaluate $\phi(\alpha_i)$;
    If $\phi(\alpha_i) > \phi(0) + c_1\alpha_i\phi'(0)$ **or** $[\phi(\alpha_i) \geq \phi(\alpha_{i-1})$ and $i > 1\ ]$
      Set $\alpha_* \leftarrow$ **zoom**(α_{i-1}, α_i) and **stop**
  Evaluate $\phi'(\alpha_i)$;
    If $|\phi'(\alpha_i)| \leq -c_2\phi'(0)$
      Set $\alpha_* \leftarrow \alpha_i$ and **stop**;
    If $\phi'(\alpha_i) \geq 0$ or $\phi'(\alpha_i) < c_2\phi'(0)$
      Set $\alpha_* \leftarrow$ **zoom**(α_{i-1}, α_i) and **stop**;

A LINE SEARCH ALGORITHM

ALGORITHM 1: (Line Search Algorithm for Wolfe Conditions)

Set $\alpha_0 \leftarrow 0$, choose $\alpha_{\max} > 0$ and $\alpha_1 \in (0, \alpha_{\max})$, $i \leftarrow 1$
Repeat
  Evaluate $\phi(\alpha_i)$;
    If $\phi(\alpha_i) > \phi(0) + c_1\alpha_i\phi'(0)$ **or** $[\phi(\alpha_i) \geq \phi(\alpha_{i-1})$ and $i > 1$]
      Set $\alpha_* \leftarrow$ **zoom**(α_{i-1}, α_i) and **stop**
  Evaluate $\phi'(\alpha_i)$;
    If $|\phi'(\alpha_i)| \leq -c_2\phi'(0)$
      Set $\alpha_* \leftarrow \alpha_i$ and **stop**;
    If $\phi'(\alpha_i) \geq 0$ or $\phi'(\alpha_i) < c_2\phi'(0)$
      Set $\alpha_* \leftarrow$ **zoom**(α_{i-1}, α_i) and **stop**;
  Choose $\alpha_{i+1} \in (\alpha_i, \alpha_{\max})$;
  $i \leftarrow i + 1$;
End(repeat)

A Line Search Algorithm

Algorithm 2: (Zoom)

Repeat

Interpolate (using quadratic, cubic or bisection) to find a trial step length α_j between $\alpha_{\text{low}}, \alpha_{\text{high}}$

A Line Search Algorithm

Algorithm 2: (Zoom)

Repeat

Interpolate (using quadratic, cubic or bisection) to find a trial step length α_j between $\alpha_{\text{low}}, \alpha_{\text{high}}$

Evaluate $\phi(\alpha_j)$;

If $\phi(\alpha_j) > \phi(0) + c_1 \alpha_i \phi'(0)$ **or** $[\phi(\alpha_i) \geq \phi(\alpha_{\text{low}})\ ]$

Set $\alpha_{\text{high}} \leftarrow \alpha_j$;

A Line Search Algorithm

Algorithm 2: (Zoom)

Repeat

Interpolate (using quadratic, cubic or bisection) to find a trial step length α_j between $\alpha_{\text{low}}, \alpha_{\text{high}}$

Evaluate $\phi(\alpha_j)$;

If $\phi(\alpha_j) > \phi(0) + c_1\alpha_i\phi'(0)$ **or** $[\phi(\alpha_i) \geq \phi(\alpha_{\text{low}})\ ]$

Set $\alpha_{\text{high}} \leftarrow \alpha_j$;

else

Evaluate $\phi'(\alpha_i)$;

A Line Search Algorithm

Algorithm 2: (Zoom)

Repeat

Interpolate (using quadratic, cubic or bisection) to find a trial step length α_j between $\alpha_{\text{low}}, \alpha_{\text{high}}$

Evaluate $\phi(\alpha_j)$;

If $\phi(\alpha_j) > \phi(0) + c_1\alpha_i\phi'(0)$ **or** $[\phi(\alpha_i) \geq \phi(\alpha_{\text{low}})\ ]$

Set $\alpha_{\text{high}} \leftarrow \alpha_j$;

else

Evaluate $\phi'(\alpha_i)$;

If $|\phi'(\alpha_i)| \leq -c_2\phi'(0)$

Set $\alpha_* \leftarrow \alpha_j$ and **stop**;

A Line Search Algorithm

Algorithm 2: (Zoom)

Repeat

Interpolate (using quadratic, cubic or bisection) to find a trial step length α_j between $\alpha_{\text{low}}, \alpha_{\text{high}}$

Evaluate $\phi(\alpha_j)$;

If $\phi(\alpha_j) > \phi(0) + c_1\alpha_i\phi'(0)$ **or** $[\phi(\alpha_i) \geq \phi(\alpha_{\text{low}})\ ]$

Set $\alpha_{\text{high}} \leftarrow \alpha_j$;

else

Evaluate $\phi'(\alpha_i)$;

If $|\phi'(\alpha_i)| \leq -c_2\phi'(0)$

Set $\alpha_* \leftarrow \alpha_j$ and **stop**;

If $\phi'(\alpha_j)(\alpha_{\text{high}} - \alpha_{\text{low}}) \geq 0$

Set $\alpha_{\text{high}} \leftarrow \alpha_{\text{low}}$;

A Line Search Algorithm

Algorithm 2: (Zoom)

Repeat

Interpolate (using quadratic, cubic or bisection) to find a trial step length α_j between $\alpha_{\text{low}}, \alpha_{\text{high}}$

Evaluate $\phi(\alpha_j)$;

If $\phi(\alpha_j) > \phi(0) + c_1\alpha_i\phi'(0)$ **or** $[\phi(\alpha_i) \geq \phi(\alpha_{\text{low}})\ ]$

Set $\alpha_{\text{high}} \leftarrow \alpha_j$;

else

Evaluate $\phi'(\alpha_i)$;

If $|\phi'(\alpha_i)| \leq -c_2\phi'(0)$

Set $\alpha_* \leftarrow \alpha_j$ and **stop**;

If $\phi'(\alpha_j)(\alpha_{\text{high}} - \alpha_{\text{low}}) \geq 0$

Set $\alpha_{\text{high}} \leftarrow \alpha_{\text{low}}$;

$\alpha_{\text{low}} \leftarrow \alpha_j$;

End(repeat)

Convergence of Line Search Methods

- We discuss requirements on the search direction in this section.

CONVERGENCE OF LINE SEARCH METHODS

- We discuss requirements on the search direction in this section.
- Focusing on one key property: the angle between p_k and the steepest descent direction $-\nabla f(x_k)$, defined by θ_k

$$\cos\theta_k = \frac{-\nabla f(x_k)^T p_k}{\|\nabla f(x_k)\|\|p_k\|} \tag{2.11}$$

Convergence of Line Search Methods

Theorem (Zoutendijk)

Convergence of Line Search Methods

Theorem (Zoutendijk)

- Consider any iteration of the form (2.19), where p_k is a descent direction and α_k satisfies the Wolfe conditions (2.3).

Convergence of Line Search Methods

Theorem (Zoutendijk)

- Consider any iteration of the form (2.19), where p_k is a descent direction and α_k satisfies the Wolfe conditions (2.3).
- Suppose that $f(x)$ is bounded below in $\mathcal{R}^n$ and that $f(x)$ is continuously differentiable in an open set $\mathcal{N}$ containing the level set $\mathcal{N} \equiv \{x| : f(x) \leq f(x_0)\}$, where x_0 is the starting point of the iteration.

Convergence of Line Search Methods

Theorem (Zoutendijk)

- Consider any iteration of the form (2.19), where p_k is a descent direction and α_k satisfies the Wolfe conditions (2.3).
- Suppose that $f(x)$ is bounded below in $\mathcal{R}^n$ and that $f(x)$ is continuously differentiable in an open set $\mathcal{N}$ containing the level set $\mathcal{N} \equiv \{x| : f(x) \leq f(x_0)\}$, where x_0 is the starting point of the iteration.
- Assume also that the gradient ∇f is Lipschitz continuous on $\mathcal{N}$, that is, there exists a constant $L > 0$ such that

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \quad \forall x, y \in \mathcal{N}. \tag{2.12}$$

CONVERGENCE OF LINE SEARCH METHODS

Theorem (Zoutendijk)

- Consider any iteration of the form (2.19), where p_k is a descent direction and α_k satisfies the Wolfe conditions (2.3).
- Suppose that $f(x)$ is bounded below in $\mathcal{R}^n$ and that $f(x)$ is continuously differentiable in an open set $\mathcal{N}$ containing the level set $\mathcal{N} \equiv \{x| : f(x) \leq f(x_0)\}$, where x_0 is the starting point of the iteration.
- Assume also that the gradient ∇f is Lipschitz continuous on $\mathcal{N}$, that is, there exists a constant $L > 0$ such that

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \quad \forall x, y \in \mathcal{N}. \tag{2.12}$$

- **Then**

$$\sum_{k \geq 0} \cos^2(\theta_k)\|\nabla f(x_k)\|^2 < \infty \tag{2.13}$$

which is called Zoutendijk condition.

Convergence of Line Search Methods

Remark

Convergence of Line Search Methods

Remark

- Similar results to this theorem hold when the Goldstein condition or strong Wolfe conditions are used in place of the Wolfe conditions.

Convergence of Line Search Methods

Remark

- Similar results to this theorem hold when the Goldstein condition or strong Wolfe conditions are used in place of the Wolfe conditions.
- The Zoutendijk condition (2.13) implies that

$$\cos^2(\theta_k)\|\nabla f(x_k)\|^2 \to 0. \tag{2.14}$$

Convergence of Line Search Methods

Remark

- Similar results to this theorem hold when the Goldstein condition or strong Wolfe conditions are used in place of the Wolfe conditions.
- The Zoutendijk condition (2.13) implies that

$$\cos^2(\theta_k)\|\nabla f(x_k)\|^2 \to 0. \tag{2.14}$$

- This limit can be used in turn to derive global convergence results for line search algorithms.

Convergence of Line Search Methods

Remark

Convergence of Line Search Methods

Remark

- If the search direction p_k is chosen that the angle θ_k is bounded away from 90°, there is a positive constant δ such that

$$\cos\theta_k \geq \delta > 0, \forall k \tag{2.15}$$

It follows immediately from (2.14) that

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0. \tag{2.16}$$

CONVERGENCE OF LINE SEARCH METHODS

REMARK

- If the search direction p_k is chosen that the angle θ_k is bounded away from $90°$, there is a positive constant δ such that

$$\cos\theta_k \geq \delta > 0, \forall k \tag{2.15}$$

  It follows immediately from (2.14) that

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0. \tag{2.16}$$

- In other words, we can be sure that the gradient norms $\|\nabla f(x_k)\|$ converge to zero, provided that the search direction are never too close to orthogonality with the gradient.

A Simple Example

A simple condition we could impose on is to require in f , that is,

$$f(x_k + \alpha_k p_k) < f(x_k)$$

- This requirement is not enough to produce convergence to x^*

A Simple Example

A simple condition we could impose on is to require in f , that is,

$$f(x_k + \alpha_k p_k) < f(x_k)$$

- This requirement is not enough to produce convergence to x^*
- For instance, the minimum function value is $f^* = -1$

A Simple Example

A simple condition we could impose on is to require in f , that is,

$$f(x_k + \alpha_k p_k) < f(x_k)$$

- This requirement is not enough to produce convergence to x^*
- For instance, the minimum function value is $f^* = -1$
- but a sequence of iterates $\{x_k\}$ for which $f(x_k) = 5/k, k = 0, 1, \cdots$ yields a decrease at each iteration but has a limiting function value of zero.

A Simple Example

A simple condition we could impose on is to require in f , that is,

$$f(x_k + \alpha_k p_k) < f(x_k)$$

- This requirement is not enough to produce convergence to x^*
- For instance, the minimum function value is $f^* = -1$
- but a sequence of iterates $\{x_k\}$ for which $f(x_k) = 5/k, k = 0, 1, \cdots$ yields a decrease at each iteration but has a limiting function value of zero.
- The insufficient reduction in f at each iteration cause it to fail to converge to the minimizer of this convex function.

A Simple Example

A simple condition we could impose on is to require in f , that is,

$$f(x_k + \alpha_k p_k) < f(x_k)$$

- This requirement is not enough to produce convergence to x^*
- For instance, the minimum function value is $f^* = -1$
- but a sequence of iterates $\{x_k\}$ for which $f(x_k) = 5/k, k = 0, 1, \cdots$ yields a decrease at each iteration but has a limiting function value of zero.
- The insufficient reduction in f at each iteration cause it to fail to converge to the minimizer of this convex function.

To avoid this behavior we need to enforce a sufficient decrease condition.

Thanks For Your Attention