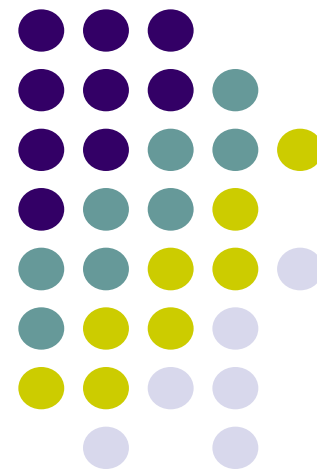


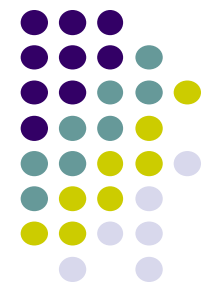
最优化方法（III）

张宏鑫

2014-04-10

浙江大学计算机学院



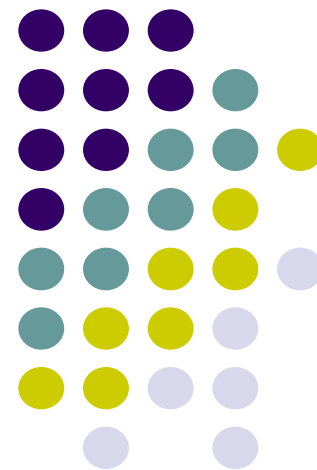


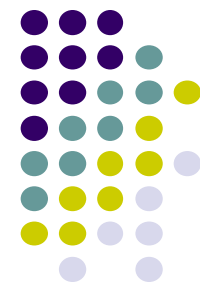
内容

- 线性规划
- 非线性优化

- 主要参考书：
 - 《线性规划》
 - 张建中，许绍吉，科学出版社
 - 《最优化理论与方法》
 - 袁亚湘，孙文瑜，科学出版社
 - 《数学规划》
 - 黄红选，韩继业，清华大学出版社

二、非线性最优化





引言

- 最优化的一般形式为

$$\begin{aligned} &\text{Min } f(\mathbf{x}) \\ &\text{s.t. } \mathbf{x} \in X \end{aligned}$$

$f(\mathbf{x})$ 为目标函数， $X \subset E^n$ 为可行域。

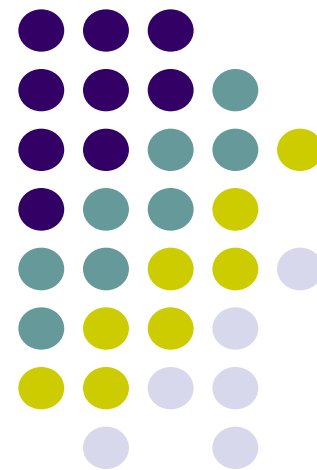
如 $X = E^n$ ，则以上最优化问题为**无约束最优化问题**

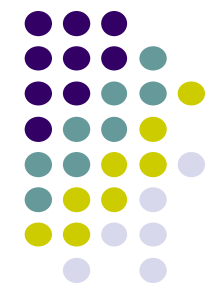
约束最优化问题通常写为

$$\begin{aligned} &\text{Min } f(\mathbf{x}) \\ &\text{s.t. } c_i(\mathbf{x}) = 0, i \in E, \\ &\quad c_i(\mathbf{x}) \geq 0, i \in I, \end{aligned}$$

其中 E, I 分别为等式约束的指标集和不等式约束的指标集， $c_i(\mathbf{x})$ 是约束函数。

1. 无约束非线性最优化





非线性的数据无处不在

Google 财经
谷歌 中国版

上证指数

查询 股票筛选器 新!

例如“601988”、“zgyh”或“中国银行”

上证指数

2,851.43

+41.31 (1.47%)
实时: 1:45PM CST

开盘价: 2,812.99
最高价: 2,853.89
最低价: 2,810.85
成交量: 9416.60万

市值: 0.00
52周最高价: 2,952.04
52周最低价: 1,664.93
平均成交量: 0.00

市盈率: -
预期市盈率: -
贝塔系数: -
每股收益: -

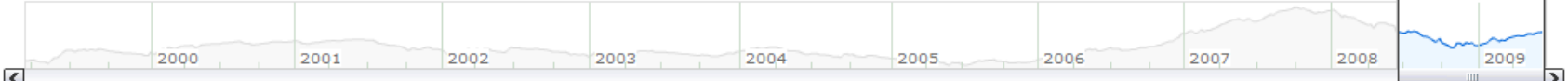
股息: -
收益率: -
总股本: 0.00
机构持股率: -

比较 设置

历史价格 链接至图表

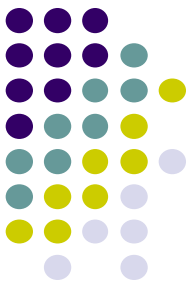
在这里输入代码 添加 深证成指

缩放: 二日 五日 一个月 三个月 半年 年初至今 一年 五年 十年 最大
2008年6月19日 - 2009年6月18日 -88.28 (-3%)

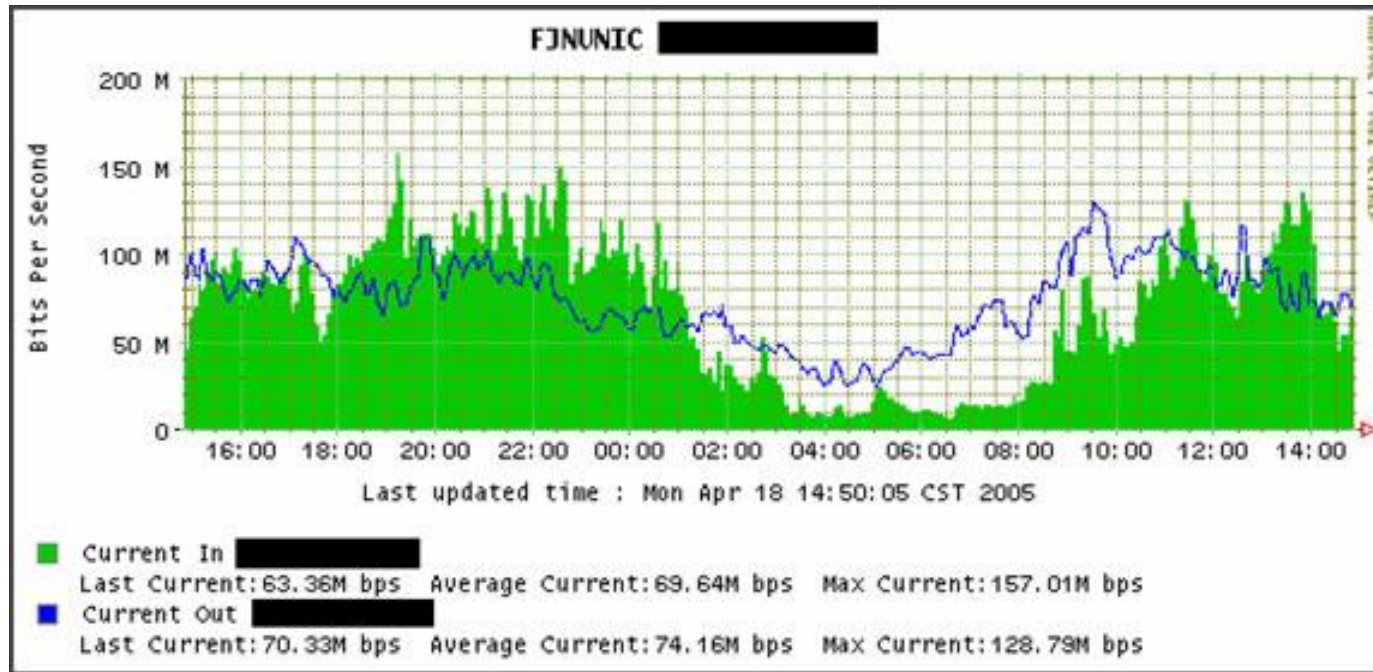


提示: 您可以拖动图表。

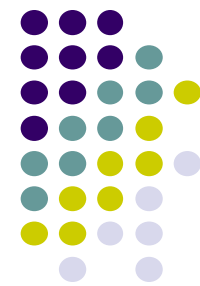
实时数据由 SHA 提供 已向后复权。 - 免责声明



非线性的数据无处不在



网络流量分析



1.1 无约束问题的最优条件

- $\min f(\mathbf{x}), \mathbf{x} \in R^n$ 的最优性条件

- **局部极小** 若存在 $\delta > 0$, 使得对所有满足 $\|x - x^*\| < \delta$ 的 x , 都有

$$f(x) \geq f(x^*),$$

则称 x^* 为 f 的局部极小点。

如所有满足 $\|x - x^*\| < \delta$ 的 x , 都有 $f(x) > f(x^*)$,

则称 x^* 为 f 的严格局部极小点。

- **全局极小** 若存在 $\delta > 0$, 使得对所有 x , 都有 $f(x) \geq f(x^*)$,

则称 x^* 为 f 的总体极小点。

如所有 x , 都有 $f(x) > f(x^*)$,

则称 x^* 为 f 的严格总体极小点。



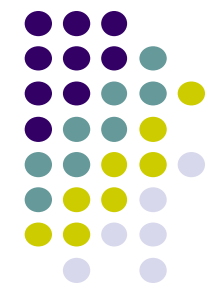
1.1 无约束问题的最优条件

- $\min f(\mathbf{x}), \mathbf{x} \in R^n$ 的最优性条件。

设 $g(x) = \nabla f(x)$, $G(x) = \Delta f(x)$ 分别为 f 的一阶和二阶导数。

定理（一阶必要条件）：设 $f : D \subset R^n \rightarrow R^1$ 在开集 D 上连续可微，
若 $x^* \in D$ 是局部极小点，则 $g(x^*) = 0$ 。

定理（二阶必要条件）：设 $f : D \subset R^n \rightarrow R^1$ 在开集 D 上二阶连续可微，
若 $x^* \in D$ 是局部极小点，则 $g(x^*) = 0, G(x^*) \geq 0$



1.1 无约束问题的最优条件

$g(x^*) = 0$, 则 x 称为函数 f 的平稳点。平稳点有可能是极小点, 也可能为极大点, 也可能不是极值点 (鞍点)。

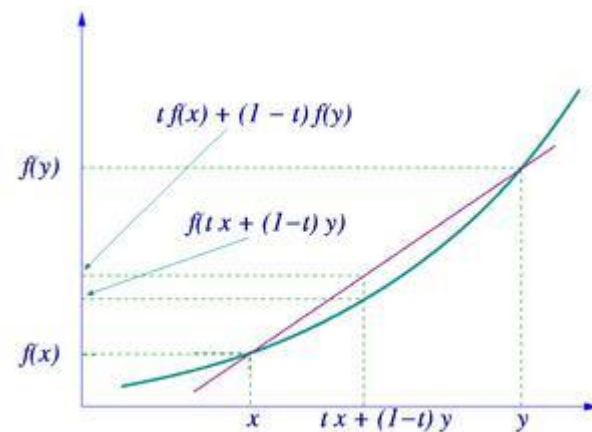
定理 (二阶充分条件) : 设 $f : D \subset R^n \rightarrow R^1$ 在开集 D 上二阶连续可微, 若 $x^* \in D$ 是严格局部极小点的充分条件是, 则 $g(x^*) = 0$, 且 $G(x^*)$ 为正定矩阵。

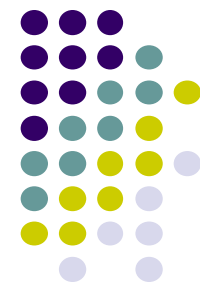
定理 (凸充分性定理) : 设 $f : D \subset R^n \rightarrow R^1$ 是凸函数且一阶连续可微, 若 x^* 是总体极小点的充要条件是 $g(x^*) = 0$ 。

问题: 什么是凸函数?

定义1. $f((1-\alpha)x + \alpha y) < (1-\alpha)f(x) + \alpha f(y)$

定义2. 函数的二阶导数 $G(x) \geq 0$





1.2 最优化方法的结构

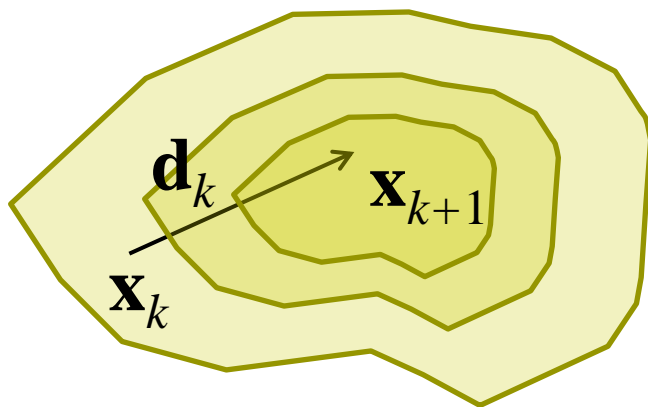
- 迭代优化方法的基本思想：
 - 给定一个初始点 \mathbf{x}_0 ,
 - 按照某一迭代规则产生一个点列 $\{\mathbf{x}_k\}$, 使得
 - 当 $\{\mathbf{x}_k\}$ 是有穷点列时, 其最后一个点是最优化模型问题的最优解。
 - 当 $\{\mathbf{x}_k\}$ 是无穷点列时, 其极限点为最优解。
- 一个**好的**算法应具备的典型特征为：
 - 迭代点 \mathbf{x}_k 能**稳定**地接近局部极小点 \mathbf{x}^* 的邻域, 然后迅速收敛于 \mathbf{x}^*
 - 当给定的某种**收敛**准则满足时, 迭代即终止。

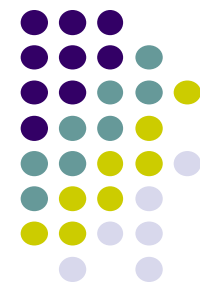


最优化方法的结构

给定初始点 \mathbf{x}_0 ,

1. 确定搜索方向 \mathbf{d}_k , 即依照一定规则构造 f 在 \mathbf{x}_k 点处的下降方向为搜索方向
2. 确定步长因子 α_k , 使目标函数值有某种意义下降
3. 令 $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$
 - a) 若 \mathbf{x}_{k+1} 满足某种终止条件, 则停止迭代, 得到近似最优解,
 - b) 否则, 重复以上步骤。





收敛速度

- 收敛速度也是衡量最优化方法有效性的重要方面。

若存在实数 $\alpha > 0$ 及一个与迭代次数 k 无关的常数 $q > 0$, 使得

$$\lim_{k \rightarrow \infty} \frac{\|\mathbf{x}_{k+1} - \mathbf{x}^*\|}{\|\mathbf{x}_k - \mathbf{x}^*\|^\alpha} = q,$$

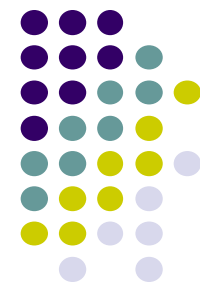
则称算法产生的迭代点列 $\{\mathbf{x}_k\}$ 具有 $Q - \alpha$ 阶收敛速度。特别地

- (1) 当 $\alpha=1$, $q > 0$ 时, $\{\mathbf{x}_k\}$ 具有 $Q -$ 线性收敛速度。
- (2) 当 $1 < \alpha < 2$, $q > 0$ 时 (或者 $\alpha=1, q = 0$), $\{\mathbf{x}_k\}$ 具有 $Q -$ 超线性收敛速度。
- (3) 当 $\alpha=2$, $q > 0$ 时, $\{\mathbf{x}_k\}$ 具有 $Q -$ 二阶收敛速度。



收敛速度

- 一般认为，具有超线性和二阶收敛速度的方法是比较快速的。
- 但对于任何一个算法，收敛性和收敛速度的理论结果并不保证算法在实际执行时一定有好的实际计算结果。
 - 忽略了误差；函数计算不满足限制条件
- 需要选择有代表性的检验函数进行数值计算。



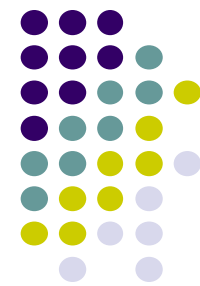
收敛速度

定理：如果序列 $\{x_k\}$ 超线性收敛到 x^* ，那么

$$\lim_{k \rightarrow \infty} \frac{\|x_{k+1} - x_k\|}{\|x_k - x^*\|} = 1,$$

但反之一般不成立。

该定理表明，可以用 $\|x_{k+1} - x_k\|$ 来代替 $\|x_k - x^*\|$ 给出终止条件，并且该估计随着 k 的增加而改善。



终止条件

$$\|x_{k+1} - x_k\| \leq \varepsilon, \text{ 或 } |f(x_k) - f(x_{k+1})| \leq \varepsilon.$$

有时 $\|x_{k+1} - x_k\|$ 是小的, 但 $|f(x_k) - f(x_{k+1})|$ 仍然很大 (或者相反)。

应同时使用两式, *Himmelblau* 提出:

$$\text{当 } \|x_k\| > \varepsilon_2, |f(x_k)| > \varepsilon_2 \text{ 时, 采用 } \frac{\|x_{k+1} - x_k\|}{\|x_k\|} \leq \varepsilon_1, \frac{|f(x_k) - f(x_{k+1})|}{|f(x_k)|} \leq \varepsilon_1,$$

否则采用 $\|x_{k+1} - x_k\| \leq \varepsilon_1$, 或 $|f(x_k) - f(x_{k+1})| \leq \varepsilon_1$

对于有一阶导数信息, 且收敛不太快的算法, 可采用 $\|g_k\| \leq \varepsilon_3$, 其中 $g_k = \nabla f(x_k)$ 。

但由于平稳点也可能是鞍点, 因此可与上式结合使用。

一般地, 可取 $\varepsilon_1 = \varepsilon_2 = 10^{-5}, \varepsilon_3 = 10^{-4}$ 。



1.3 一维搜索

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k \rightarrow \varphi(\alpha) = f(\mathbf{x}_k + \alpha \mathbf{d}_k)$$

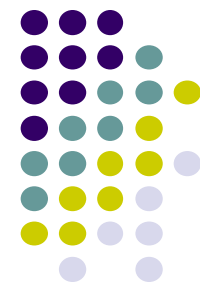
- 单变量函数的最优化。

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k,$$

其关键就是构造搜索方向 \mathbf{d}_k 和步长因子 α_k 。设 $\varphi(\alpha) = f(\mathbf{x}_k + \alpha \mathbf{d}_k)$ ，这样，确定 α_k ，使得 $\varphi(\alpha_k) < \varphi(0)$ 。这就是关于 α 的一维搜索问题。

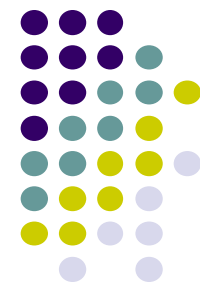
若 α_k 使得目标函数沿方向达到极小，即 $\varphi(\alpha_k) = \min_{\alpha > 0} \varphi(\alpha)$ ，则称这样的一维搜索为最优一维搜索（或精确一维搜索）， α_k 为最优步长因子。若取 α_k 使得目标函数得到可以接受的下降量，则称为近似一维搜索，或不精确一维搜索。

实际中，精确的最优步长因子一般不能求到，求几乎精确的最优步长因子需花费想到大的工作量，因而花费计算量较少的不精确一维搜索受到重视。



一维搜索的主要结构

1. 确定包含问题最优解的**搜索区间**
 2. 再用某种分割技术或插值方法**缩小这个区间**，进行搜索求解
- 搜索区间：包含最优值的闭区间。
 - 确定搜索区间的简单方法——进退法。
 - 从一点出发，试图确定出函数值呈现“高一低一高”的三点。一个方向不成功，就退回来，再沿相反方向寻找。



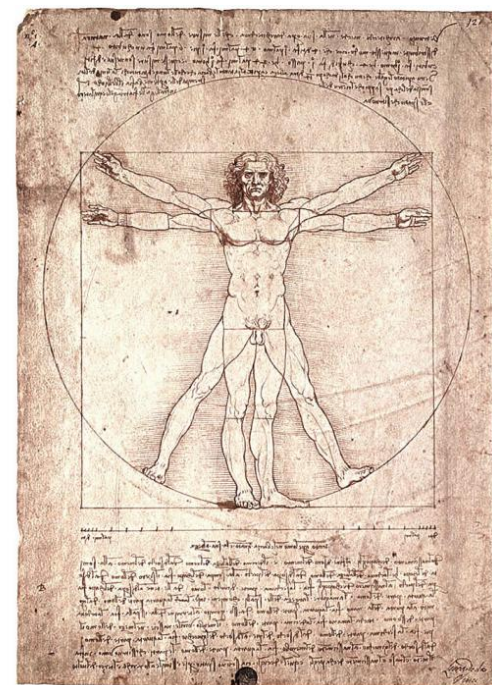
一维区间搜索的进退法

- 确定搜索区间的简单方法——进退法。
 - 从一点出发，试图确定出函数值呈现“高一低一高”的三点。一个方向不成功，就退回来，再沿相反方向寻找。
 1. 选取初始值 α_0 , h_0 , 加倍系数 $t > 1$ (一般 $t = 2$), $k = 0$
 2. 如 $\varphi(\alpha_k + h_k) < \varphi(\alpha_k)$, 则 $h_{k+1} = th_k$, $\alpha_{k+1} = \alpha_k + h_{k+1}$, $k++$, 返回 2
 3. 若 $k = 0$, 转换搜索方向 (即 $h_k = -h_k$, 转 2);
否则, 停止迭代, 输出 $a = \min\{\alpha_0, \alpha_{k+1}\}$, $b = \max\{\alpha_0, \alpha_{k+1}\}$



一维搜索：黄金分割法

- 基本思想：通过取试探点和进行函数值比较，使包含极小点的搜索区间不断缩小，当缩短到一定程度后，该区间上的任意一点均可看作极小值的近似
- 该方法用途广泛
 - 尤其适合导数表达式复杂或写不出的情况





一维搜索：黄金分割法（0.618法）

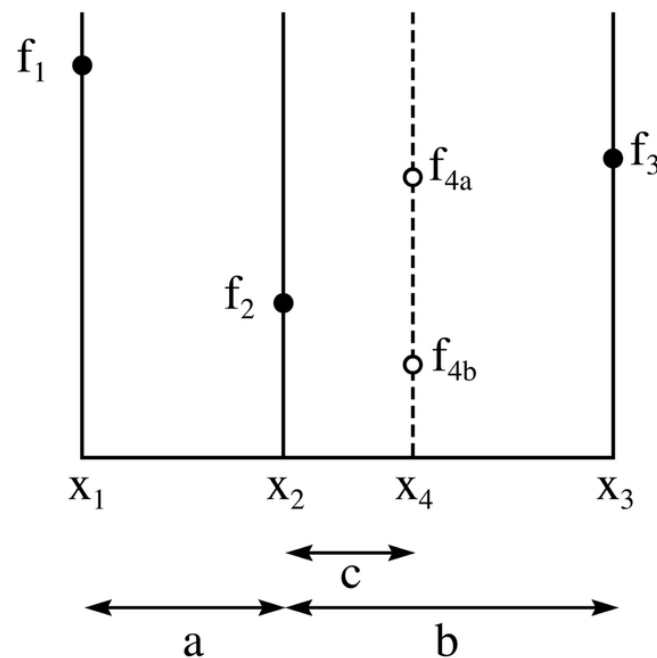
设 $\varphi(\alpha) = f(x_k + \alpha d_k)$ 是搜索区间上的单峰函数。

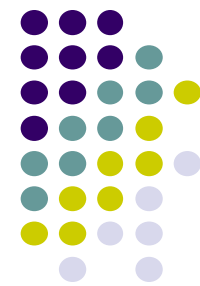
设第 k 次迭代时搜索区间为 $[a_k, b_k]$,

取两个试探点 λ_k, μ_k ($\lambda_k < \mu_k$), 计算 $\varphi(\lambda_k)$ 和 $\varphi(\mu_k)$ 。

(1) 若 $\varphi(\lambda_k) \leq \varphi(\mu_k)$, 则令 $a_{k+1} = a_k, b_{k+1} = \mu_k$ (易证此时仍为单峰函数)

(2) 若 $\varphi(\lambda_k) > \varphi(\mu_k)$, 则令 $a_{k+1} = \lambda_k, b_{k+1} = b_k$





一维搜索：黄金分割法（0.618法）

设 $\varphi(\alpha) = f(x_k + \alpha d_k)$ 是搜索区间上的单峰函数。

设第 k 次迭代时搜索区间为 $[a_k, b_k]$,

取两个试探点 λ_k, μ_k ($\lambda_k < \mu_k$), 计算 $\varphi(\lambda_k)$ 和 $\varphi(\mu_k)$ 。

(1) 若 $\varphi(\lambda_k) \leq \varphi(\mu_k)$, 则令 $a_{k+1} = a_k, b_{k+1} = \mu_k$ (易证此时仍为单峰函数)

(2) 若 $\varphi(\lambda_k) > \varphi(\mu_k)$, 则令 $a_{k+1} = \lambda_k, b_{k+1} = b_k$

要求试探点 λ_k, μ_k 满足下列条件:

1. λ_k, μ_k 到搜索区间 $[a_k, b_k]$ 的端点等距, 即 $\lambda_k - a_k = b_k - \mu_k$ (或 $b_k - \lambda_k = \mu_k - a_k$)

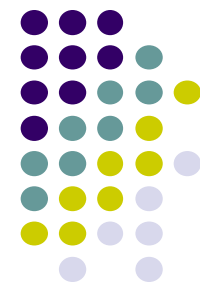
2. 每次迭代, 搜索区间长度缩短率相同, 即 $b_{k+1} - a_{k+1} = \tau(b_k - a_k)$

则可推出 $\lambda_k = a_k + (1 - \tau)(b_k - a_k)$, $\mu_k = a_k + \tau(b_k - a_k)$ 。

3. $\lambda_k = \mu_{k+1}$

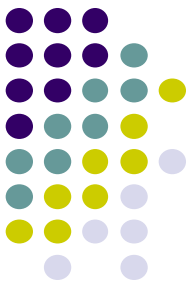
可推出 $\tau = \frac{\sqrt{5}-1}{2} \approx 0.618$

不难知道第 n 次迭代后的区间长度为 $\tau^{n-1}(b_1 - a_1)$, 收敛速度为线性。



改进

- 实际上所遇到的函数不一定是单峰函数，这时搜索出的值有可能大于初始区间的端点值。
- 改进：每次缩小区间时，不只比较两个内点处的函数值，而是比较两个内点和两个端点处的函数值。
 - 当左边第一个或第二个点是这四个点中函数值最小的点时，丢弃右端点；
 - 否则，丢弃左端点。



Fibonacci法

- 与0.618法的主要区别之一：搜索区间缩短率不是采用黄金分割数，而是采用Fibonacci数。
 - Fibonacci数满足：

$$F_0=F_1=1, F_{k+1}=F_k+F_{k-1}, k=1, 2, \dots$$

$$1, 1, 2, 3, 5, 8, 13, \dots$$

Fibonacci分割方法：

$$\lambda_k = a_k + (1 - \tau)(b_k - a_k), \quad \mu_k = a_k + \tau(b_k - a_k), \quad \tau = \frac{F_{n-k-1}}{F_{n-k+1}}$$

Fibonacci法



$$\lambda_k = a_k + (1 - \tau)(b_k - a_k), \quad \mu_k = a_k + \tau(b_k - a_k), \quad \tau = \frac{F_{n-k-1}}{F_{n-k+1}}$$

要求经过 n 次计算后，最后的区间长度不超过 δ ，即

$$b_n - a_n \leq \delta.$$

$$\text{另一方面, } b_n - a_n = \frac{F_1}{F_2}(b_{n-1} - a_{n-1}) = \frac{F_1}{F_2} \frac{F_2}{F_3} \cdots \frac{F_{n-1}}{F_n}(b_1 - a_1) = \frac{1}{F_n}(b_1 - a_1)$$

$$\text{故可得 } F_n \geq \frac{(b_1 - a_1)}{\delta}$$

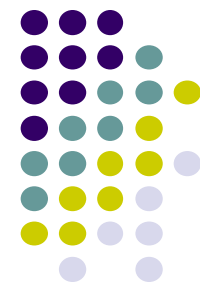
可以证明 $n \rightarrow \infty$ ，Fibonacci法与0.618法的区间缩短率相同。

因而Fibonacci法线性收敛。

同时可以证明：

Fibonacci法是分割方法求一维极小化问题的最优策略，

而0.618法是分割方法求一维极小化问题的近似最优解。



1.3.2 插值法

- 插值法是一类重要的搜索方法，其基本思想是：
 - 在搜索区间中不断用低次(通常不超过三次)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的极小点。
- 当函数具有比较好的解析性质时，插值方法比直接方法(0.618法或Fibonacci法)效果更好。

二次插值法： 一点二次插值（牛顿法）



- 利用一点处的函数值、一阶和二阶导数值构造二次插值函数

$q(\alpha) = a\alpha^2 + b\alpha + c$, 则 $\alpha = -\frac{b}{2a}$ 为近似极小点的计算公式。

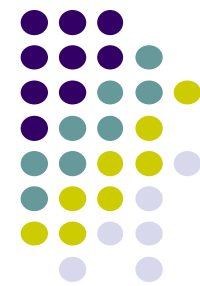
设端点处值为 $\varphi(\alpha_1)$, $\varphi'(\alpha_1)$, $\varphi''(\alpha_1)$ 。则可推出一维搜索的近似极小点为

$$\bar{\alpha} = -\frac{b}{2a} = \alpha_1 - \frac{\varphi'(\alpha_1)}{\varphi''(\alpha_1)}$$

写成迭代形式为

$$\alpha_{k+1} = \alpha_k - \frac{\varphi'(\alpha_k)}{\varphi''(\alpha_k)}$$

- 牛顿法的优点是收敛速度快，具有局部二阶收敛速度。



二点二次插值法

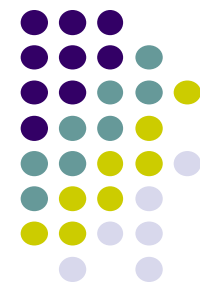
- 给出两点的函数值和其中一点的导数值，构造二次插值函数。类似的可得

$$\bar{\alpha} = -\frac{b}{2a} = \alpha_1 - \frac{(\alpha_1 - \alpha_2)\varphi'(\alpha_1)}{2\left[\varphi'(\alpha_1) - \frac{\varphi(\alpha_1) - \varphi(\alpha_2)}{\alpha_1 - \alpha_2}\right]}$$

写成迭代形式为

$$\alpha_{k+1} = \alpha_k - \frac{(\alpha_k - \alpha_{k-1})\varphi'(\alpha_k)}{2\left[\varphi'(\alpha_k) - \frac{\varphi(\alpha_k) - \varphi(\alpha_{k-1})}{\alpha_k - \alpha_{k-1}}\right]}$$

- 可证明二点二次插值法的收敛阶为**1.618**，超线性收敛。



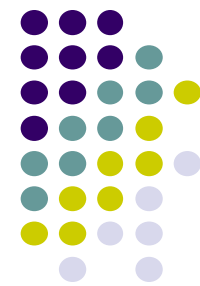
三点二次法（抛物线法）

$$\bar{\alpha} = -\frac{b}{2a} = \frac{1}{2}(\alpha_1 + \alpha_2) + \frac{1}{2} \frac{(\phi_1 - \phi_2)(\phi_2 - \phi_3)(\phi_3 - \phi_1)}{(\alpha_2 - \alpha_3)\phi_1 + (\alpha_3 - \alpha_1)\phi_2 + (\alpha_1 - \alpha_2)\phi_3}。$$

$$\phi_1 > \phi_2, \phi_3 > \phi_2$$

迭代时，从 $\alpha_1, \alpha_2, \alpha_3, \bar{\alpha}$ 中选择目标函数最小的点及其左右两点，进行下一步的迭代。

- 超线性收敛，收敛阶近似为1.32



二点三次插值法

- 用三次多项式来逼近。比二次插值法有较好的收敛效果，但通常要求计算导数值，且计算量较大。一般当导数易求时，用三次插值法较好。

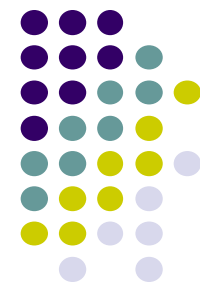
用函数 $\varphi(\alpha)$ 在 a, b 两点的函数值 $\varphi(a), \varphi(b)$ 和导数值 $\varphi'(a), \varphi'(b)$ 来构造三次函数，初值条件 $a < b, \varphi'(a) < 0, \varphi'(b) > 0$ 。经过推导

$$\bar{\alpha} = a + (b - a) \frac{(w - \varphi'(a) - z)}{\varphi'(b) - \varphi'(a) + 2w},$$

$$\text{其中 } z = 3 \frac{\varphi(b) - \varphi(a)}{b - a} - \varphi'(a) - \varphi'(b), \quad w^2 = z^2 - \varphi'(a)\varphi'(b)$$

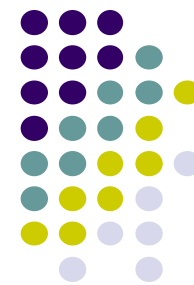
在迭代时，若 $f(\bar{\alpha}) < 0$ ，则令 $a = \bar{\alpha}$ ；若 $f(\bar{\alpha}) > 0$ ，令 $b = \bar{\alpha}$ 。

- 收敛速度为2阶，一般优于抛物线法。



1.3.3 不精确的一维搜索方法

- 精确一维搜索往往需要花费很大的时间。
 - 当迭代点远离问题的解时，精确求解通常不十分有效。
 - 很多最优化方法，如牛顿法和拟牛顿法，其收敛速度并不依赖于精确一维搜索过程。
- 只要保证目标函数有满意的下降，可大大节省计算量。



Armijo-Goldstein准则

设区间 $J = \{\alpha > 0 \mid \varphi(\alpha) < \varphi(0)\}$. 要保证目标函数下降, 同时 f 的下降不是太小。一个合理的要求:

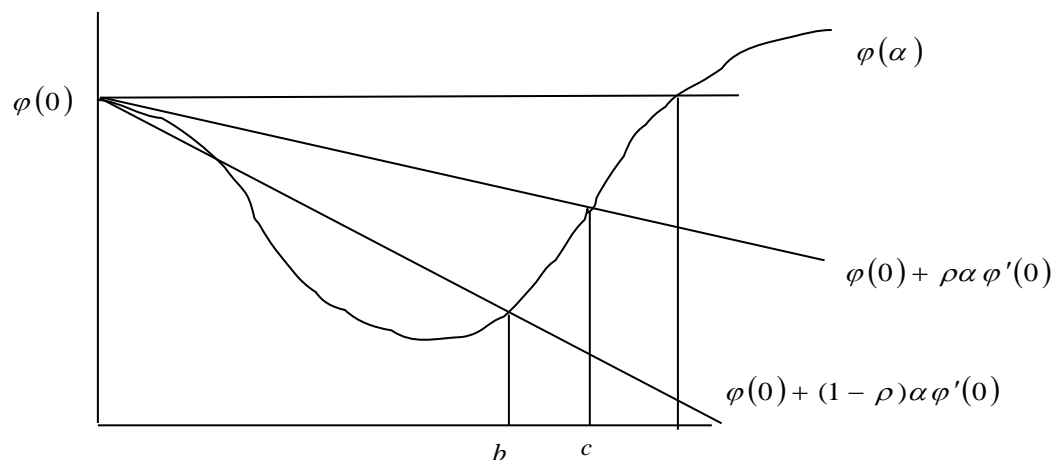
$$\varphi(\alpha) \leq \varphi(0) + \rho \alpha \varphi'(0), \text{ 其中 } 0 < \rho < 0.5.$$

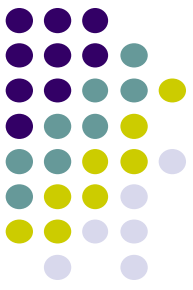
为了避免 α 取得过小, 加上另一个要求

$$\varphi(\alpha) \geq \varphi(0) + (1 - \rho) \alpha \varphi'(0).$$

上述两式称为 *Armijo - Goldstein* 不精确线性搜索准则。满足上述

要求的 α 构成区间 $J_2 = [b, c]$, 称为可接受区间, α 称为可接受步长因子。

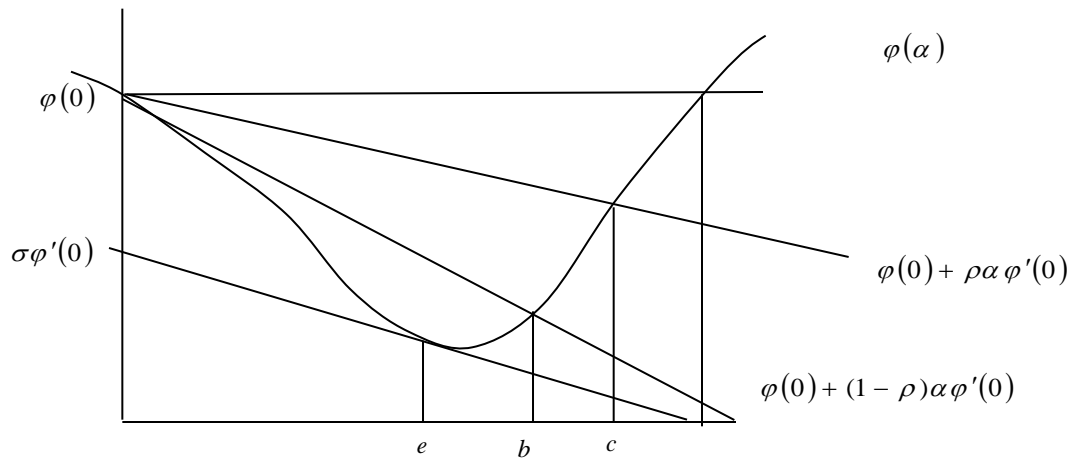




Wolfe-Powell准则

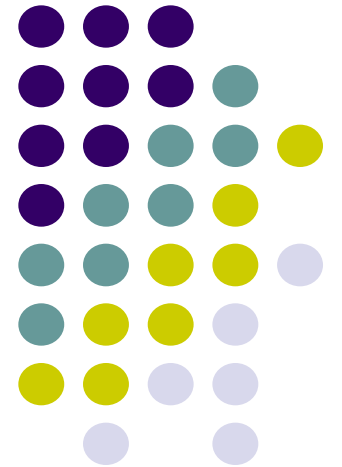
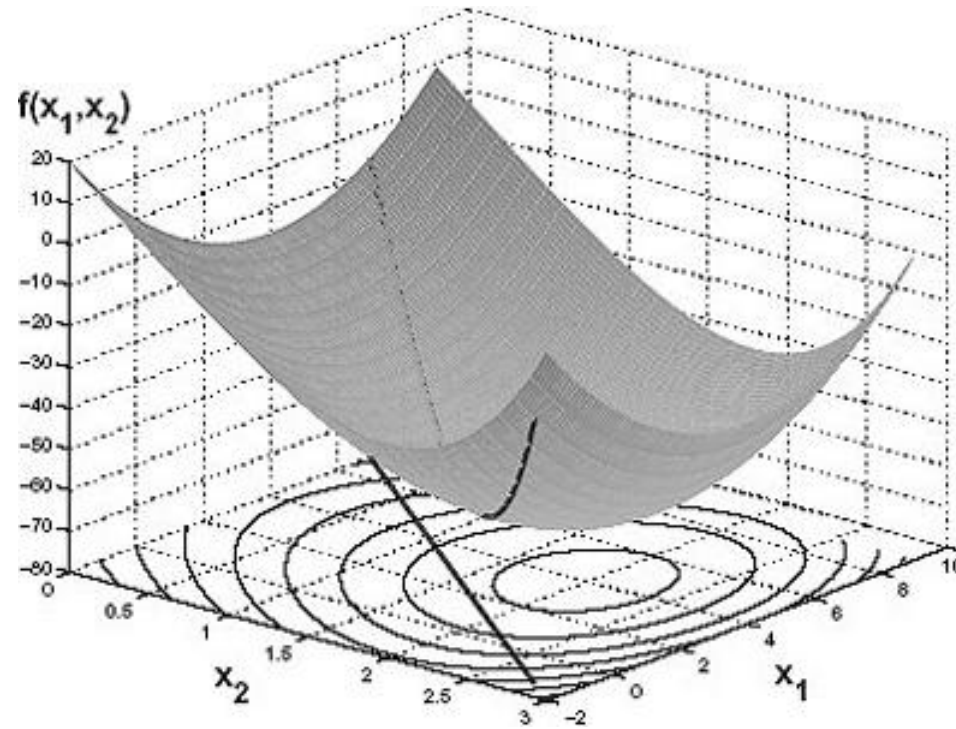
- Armijo-Goldstein准则有可能把最优步长因子排除在可接受区间外，因此更改第二个条件为

$\varphi(\alpha) \geq \varphi(0) + (1 - \rho)\alpha\varphi'(0) \Rightarrow |\varphi'(\alpha)| \leq \sigma |\varphi'(0)|$, 其中 $\rho < \sigma < 1$ 。



- 一般地， σ 值越小（0.1），线性搜索越精确。不过，计算量也越大。通常取 $\rho = 0.1$, $\sigma = 0.4$ 。

1.4 牛顿型方法



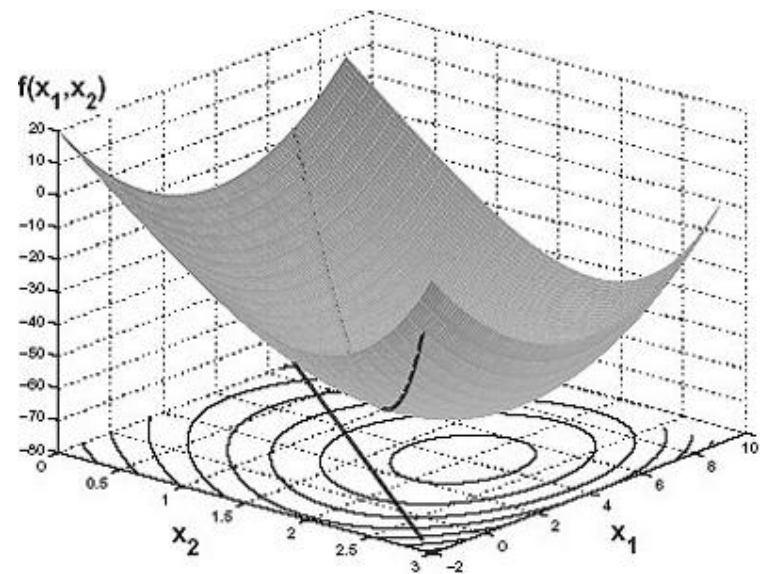


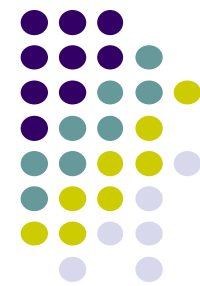
1.4.1 最速下降法

- 以负梯度方向作为极小化算法的搜索方向，即

$$\mathbf{d}_k = -\mathbf{g}_k$$

- 具有总体收敛性：
 - 产生的迭代点列 $\{\mathbf{x}_k\}$ 的每一个聚点都是平稳点。
- 最速下降方向仅是局部性质
 - 对于许多问题并非下降方向，而且下降非常缓慢
 - 接近极小点时，步长越小前进越慢
- 线性收敛





1.4.2 牛顿法

- 牛顿法的基本思想是利用目标函数的二次**Taylor**展开，并将其极小化。

函数 $f(x)$ 在 x_k 处的二次 *Taylor* 展开为

$$f(x_k + s) \approx q^{(k)}(s) = f(x_k) + \nabla f(x_k)^T s + \frac{1}{2} s^T \nabla^2 f(x_k) s$$

其中 $s = x - x_k$. 将 $q^{(k)}(s)$ 极小化，得到

$$x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k) = x_k - G_k^{-1} g_k$$

- 对于正定二次函数，一步即可得最优解。
- 由于目标函数在极点附件近似于二次函数，故在初始点接近极小点时，牛顿法收敛速度较快。
- 牛顿法具有局部收敛性，为二阶收敛。

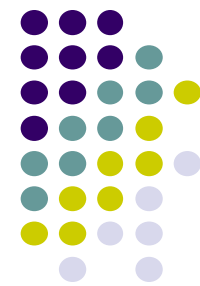


牛顿法

- 当初始点远离最优解时， G_k 不一定是正定的，则牛顿方向不一定为下降方向，其收敛性不能保证。
- 这说明恒取步长因子为1是不合适的，应该采用一维搜索（仅当步长因子 $\{\alpha_k\}$ 收敛1时，牛顿法才是二阶收敛的）。

$$\text{迭代公式为： } \mathbf{d}_k = -G_k^{-1} \mathbf{g}_k, \mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$$

- 带步长因子的牛顿法是总体收敛的。



1.4.3 修正牛顿法

- 牛顿法的主要困难在于Hesse阵 G_k 不正定。这时二次型模型不一定有极小点，甚至没有平稳点。
- Goldstein and Price (1967)当 G_k 非正定时，采用最速下降方向结合方向，给出

$$\mathbf{d}_k = \begin{cases} \mathbf{d}_k^N = -G_k^{-1} \mathbf{g}_k, & \text{angle} \langle \mathbf{d}_k^N, -\mathbf{g}_k \rangle < \frac{\pi}{2} \\ -\mathbf{g}_k, & \text{otherwise} \end{cases}$$

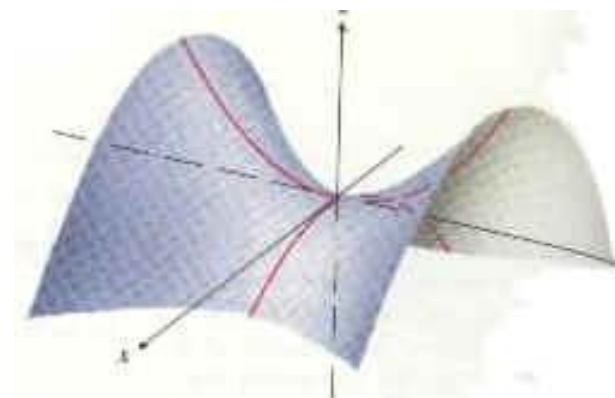


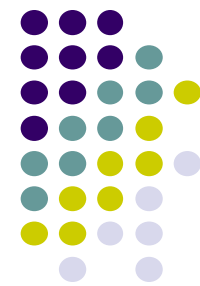
负曲率方向法

- 在鞍点处 $g_k=0$ 而 G_k 不是正半定时，修正牛顿法失效
 - 采用负曲率方向作为搜索方向，可使目标函数值下降
- 若 $G(\mathbf{x})$ 至少有一个负特征值，则 \mathbf{x} 叫做 **不定点**
- 若 \mathbf{x} 是一个不定点，方向 \mathbf{d} 满足

$$\mathbf{d}^T G(\mathbf{x}) \mathbf{d} < 0,$$

则 \mathbf{d} 为 $f(\mathbf{x})$ 在 \mathbf{x} 处的负曲率方向





Gill-Murray稳定牛顿法

- 基本思想是:

- 当 G_k 不定矩阵时, 采用修改Cholesky分解强迫矩阵正定;
- 当 \mathbf{g}_k 趋于0时, 采用负曲率方向使函数值下降

$$\overline{G}_k = G_k + E_k = L_k D_k L_k^T$$

构造负曲率方向:

1. 令 $w_j = d_{jj} - e_{jj}, j = 1, \dots, n$

2. 求下标 t , 使得 $w_t = \min \{w_j \mid j = 1, \dots, n\}$

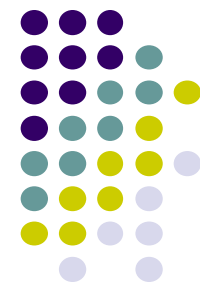
3. 若 $w_t \geq 0$, 则停止; 否则求解 $L_k^T d = e_t$, 求得方向 d 为负曲率方向。



Gill-Murray稳定牛顿法

1. 给定初始点 $x_0, \varepsilon > 0, k = 1$
2. 计算 g_k 和 G_k
3. 对 G_k 进行修改 *Cholesky* 分解, $L_k D_k L_k^T = G_k + E_k$
4. 若 $\|g_k\| > \varepsilon$, 则解方程 $L_k D_k L_k^T d = -g_k$, 求出搜索方向 d_k , 转 6。
5. 构造负曲率方向, 若求不出方向 (即 $w_i \geq 0$), 则停止;
否则, 求出方向 d_k , 再令 $d_k = \begin{cases} -d_k, & \text{若 } g_k^T d_k > 0 \\ d_k, & \text{其他} \end{cases}$
6. 一维搜索, 使得 $f(x_k + \alpha_k d_k) = \min_{\alpha \geq 0} f(x_k + \alpha d_k)$, 并令
$$x_{k+1} = x_k + \alpha_k d_k$$
7. 若满足终止条件, 则停止; 否则, $k = k + 1$, 转 2。

可证明该方法总体收敛!



Goldfeld修正牛顿法

- 使牛顿方向偏向最速下降方向。更明确地说，将Hesse阵 G_k 改变成 $G_k + v_k I$ ，使得 $G_k + v_k I$ 正定。比较理想的是， v_k 为使 $G_k + v_k I$ 正定的最小 v 。

利用 *Gill*和 *Murray*的修改 *Cholesky*分解算法确定 v_k 即

$$G_k + E = LDL^T \text{ (} L \text{为下三角阵, } D \text{为对角阵)}$$

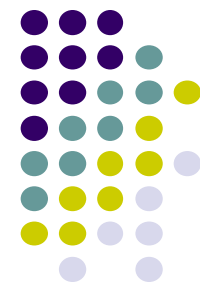
若 $E = 0$, 则 $v_k = 0$;

若 $E \neq 0$, 则利用 *Gerschgorin*圆盘定理计算 v_k 的一个上界 b_1 ,

$$b_1 = \left| \min_{1 \leq i \leq n} \left\{ (G_k)_{ii} - \sum_{j \neq i} |(G_k)_{ij}| \right\} \right| \geq \left| \min_i \lambda_i \right|$$

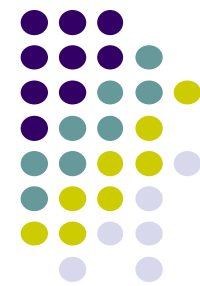
另外，令 $b_2 = \max_i \{(E)_{ii}\}$ 也是 v 的一个上界。

最后，取 $v_k = \min \{b_1, b_2\}$



信赖域方法

- 不仅可以用来代替一维搜索，而且也可以解决Hessen矩阵不正定等困难
- 主要思想：
 - 首先选择一个步长 r ，使得在 $\|\mathbf{x}-\mathbf{x}_k\|<r$ 范围内(信赖域)
 - 目标函数用 n 维二次模型来逼近，并以此选择一个搜索方向 \mathbf{s}_k ，取
$$\mathbf{x}_{k+1}=\mathbf{x}_k+\mathbf{s}_k$$
- 具有牛顿法的快速局部收敛性，又具有理想的总体收敛性。



Levenberg-Marquardt方法

最重要的一类的信赖域是取 l_2 范数，此时原模型等效于

$$\min q^{(k)}(\mathbf{s}) = f_k + \mathbf{g}_k^T \mathbf{s} + \frac{1}{2} \mathbf{s}^T \mathbf{G}_k \mathbf{s}, \quad s.t. \|\mathbf{s}\|_2 \leq h_k$$

引入 *Lagrange* 函数 $L(\mathbf{s}, \mu) = q^{(k)}(\mathbf{s}) + \frac{1}{2} \mu (\mathbf{s}^T \mathbf{s} - h_k^2)$ 。

根据约束最优化的最优性条件知： $\nabla_s L = 0, \mu \geq 0$

从而可推出

$$\begin{aligned} L(s, \mu_k) &= q^{(k)}(s) + \frac{1}{2} \mu_k (s^T s - h_k^2) = q^{(k)}(s) + \frac{1}{2} \mu_k (s^T s - h_k^2) + s_k^T \nabla_s L(s_k, \mu_k) \\ &= f_k + \mathbf{g}_k^T s + \frac{1}{2} s^T \mathbf{G}_k s + \frac{1}{2} \mu_k (s^T s - h_k^2) + s_k^T \nabla_s L(s_k, \mu_k) + s_k^T [\mathbf{g}_k + \mathbf{G}_k s_k + \mu_k s_k] \\ &= q^{(k)}(s_k) + \frac{1}{2} (s - s_k)^T (\mathbf{G}_k + \mu_k \mathbf{I})(s - s_k) \end{aligned}$$



Levenberg-Marquardt方法

信赖域采用 l_2 范数定义, 原模型等效于

$$\min q^{(k)}(\mathbf{s}) = f_k + \mathbf{g}_k^T \mathbf{s} + \frac{1}{2} \mathbf{s}^T \mathbf{G}_k \mathbf{s}, \quad s.t. \|\mathbf{s}\|_2 \leq h_k$$

引入 *Lagrange* 函数 $L(\mathbf{s}, \mu) = q^{(k)}(\mathbf{s}) + \frac{1}{2} \mu (\mathbf{s}^T \mathbf{s} - h_k^2)$

根据约束最优化的最优性条件知: $\nabla_{\mathbf{s}} L = 0, \mu \geq 0$

$$L(\mathbf{s}, \mu_k) = q^{(k)}(\mathbf{s}_k) + \frac{1}{2} (\mathbf{s} - \mathbf{s}_k)^T (\mathbf{G}_k + \mu_k \mathbf{I})(\mathbf{s} - \mathbf{s}_k)$$

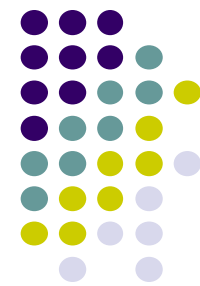
可证明:

总体解的二阶必要条件为 $(\mathbf{G}_k + \mu_k \mathbf{I})$ 半正定。

总体解严格最小的充分条件为 $(\mathbf{G}_k + \mu_k \mathbf{I})$ 正定。

因此, *LM* 方法都是要确定一个 $\mu_k \geq 0$, 使得 $(\mathbf{G}_k + \mu_k \mathbf{I})$ 正定,

并用 $\mathbf{g}_k + (\mathbf{G}_k + \mu_k \mathbf{I})\mathbf{s} = 0$ (即 $\nabla_{\mathbf{s}} L = 0$) 求解 \mathbf{s}_k 。同时可以证明 $\|\mathbf{s}\|_2$ 随 μ 单调减小。

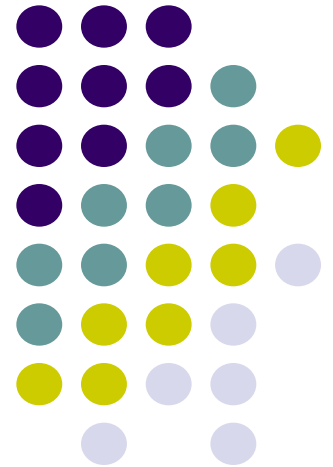


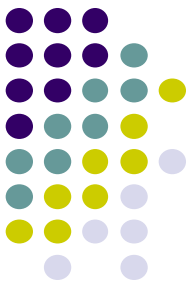
Levenberg-Marquardt方法

1. 给定初始点 $x_0, \mu_0 > 0, k = 1$
2. 计算 g_k 和 G_k
3. 若 $\|g_k\| < \varepsilon$, 停止。
4. 分解 $G_k + \mu_k I$, 若不正定, 置 $\mu_k = 4\mu_k$, 重复 4 直到正定。
5. 解 $(G_k + \mu_k I)s = -g_k$, 求出 s_k
6. 求 $f(x_k + s_k), q^{(k)}(s_k)$ 和 $r_k = \frac{\Delta f_k}{\Delta q^{(k)}}$
7. 若 $r_k < 0.25$, 置 $\mu_{k+1} = 4\mu_k$; 若 $r_k > 0.75$, 置 $\mu_{k+1} = \mu_k / 2$; 否则, $\mu_{k+1} = \mu_k$
8. 若 $r_k \leq 0$, 置 $x_{k+1} = x_k$; 否则, 置 $x_{k+1} = x_k + s_k$
9. 令 $k = k + 1$, 转 2。

1.5 共轭方向法

介于最速下降法与牛顿法之间的一个方法，仅需利用一阶导数，但克服了最速下降法收敛慢地缺点，又避免存储和计算牛顿法所需要的二阶导数信息。





共轭

- 设 G 是 $n \times n$ 对称正定矩阵，若给定 n 维非零向量 \mathbf{d}_1 和 \mathbf{d}_2 ，满足

$$\mathbf{d}_1^T G \mathbf{d}_2 = 0,$$

则称向量 $\mathbf{d}_1, \mathbf{d}_2$ 是 G -共轭的

- 类似，设 n 维非零向量 $\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_m$ ，如果 $\mathbf{d}_i^T G \mathbf{d}_j = 0$ ($i \neq j$)，则称 $\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_m$ 是 G -共轭的

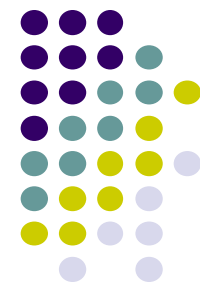


一般共轭方向法

1. 给定初始点 \mathbf{x}_0 , $k = 0$
2. 计算 $\mathbf{g}_0 = \mathbf{g}(\mathbf{x}_0)$
3. 计算 \mathbf{d}_0 , 使得 $\mathbf{d}_0^T \mathbf{g}_0 < 0$
4. 一维搜索得到 $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$
5. 计算 \mathbf{d}_{k+1} , 使得 $\mathbf{d}_{k+1}^T G \mathbf{d}_j = 0, j = 0, 1, \dots, k$
6. $k = k + 1$, 转 4。

共轭方向法的基本性质:

- 共轭方向 $\mathbf{d}_0, \mathbf{d}_1, \dots, \mathbf{d}_n$, 对于所有 $i \leq n-1$, 有
$$\mathbf{g}_{i+1}^T \mathbf{d}_j = 0, (j=0, \dots, i)$$
- 对于正定二次函数, 共轭方向法至多经过 n 步精确线性搜索终止。



共轭梯度法

考虑目标函数：
$$f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T G \mathbf{x} + \mathbf{b}^T \mathbf{x} + \mathbf{c}$$

令 $\mathbf{d}_0 = -\mathbf{g}_0$, 根据一维精确搜索的性质, $\mathbf{g}_1^T \mathbf{d}_0 = 0$

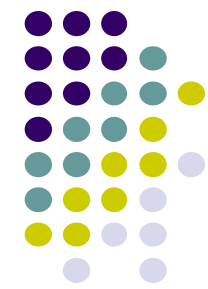
令

$$\mathbf{d}_1 = -\mathbf{g}_1 + \beta_0 \mathbf{d}_0 \quad (*)$$

选择 β_0 使得 $\mathbf{d}_1^T G \mathbf{d}_0 = 0$. 则 (*) 式两边乘 $\mathbf{d}_0^T G$, 可得

$$\beta_0 = \frac{\mathbf{g}_1^T G \mathbf{d}_0}{\mathbf{d}_1^T G \mathbf{d}_0} \xrightarrow{\text{二次函数}} = \frac{\mathbf{g}_1^T (\mathbf{g}_1 - \mathbf{g}_0)}{\mathbf{d}_1^T (\mathbf{g}_1 - \mathbf{g}_0)} = \frac{\mathbf{g}_1^T \mathbf{g}_1}{\mathbf{g}_0^T \mathbf{g}_0}$$

- 共轭梯度法仅比最速下降法稍微复杂一点, 但具有二次终止性 (对于二次函数, 算法在有限步终止)。



共轭梯度法

$$\beta_0 = \frac{\mathbf{g}_1^T G \mathbf{d}_0}{\mathbf{d}_1^T G \mathbf{d}_0} \xrightarrow{\text{二次函数}} = \frac{\mathbf{g}_1^T (\mathbf{g}_1 - \mathbf{g}_0)}{\mathbf{d}_1^T (\mathbf{g}_1 - \mathbf{g}_0)} = \frac{\mathbf{g}_1^T \mathbf{g}_1}{\mathbf{g}_0^T \mathbf{g}_0}$$

类似地利用共轭法的基本性质进行推导，可得 一般在第 $k+1$ 次迭代时，

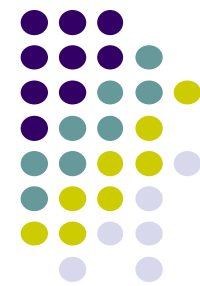
$$\mathbf{d}_{k+1} = -\mathbf{g}_{k+1} + \beta_k \mathbf{d}_k, \text{ 其中}$$

$$\beta_k = \frac{\mathbf{g}_{k+1}^T (\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{d}_k^T (\mathbf{g}_{k+1} - \mathbf{g}_k)} \text{ (Crowder - Wolfe 公式)}$$

$$= \frac{\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}}{\mathbf{g}_k^T \mathbf{g}_k} \text{ (Fletcher - Reeves 公式)}$$

$$= -\frac{\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}}{\mathbf{d}_k^T \mathbf{g}_k} \text{ (Dixon 公式)}$$

- 共轭梯度法仅比最速下降法稍微复杂一点，但具有二次终止性（对于二次函数，算法在有限步终止）。

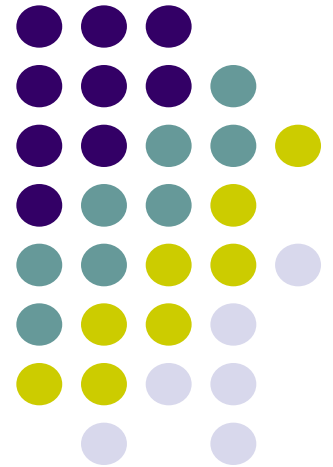


再开始FR共轭梯度法

- 对于一般非二次函数， n 步后共轭梯度法产生的搜索方向不再共轭。
- 再开始共轭梯度法
 - n 步后周期性采用最速下降方向作为搜索方向
 - 在精确线性搜索条件下，总体收敛。
 - 满足Lipschitz条件，线性搜索由Wolfe-Powell准则确定，且下降方向与负梯度方向小于90度，总体收敛。
 - 具有 n 步二阶收敛性。（ n 步可求得二次凸函数的极小点，相当牛顿法的执行一步）

1.6 拟牛顿法

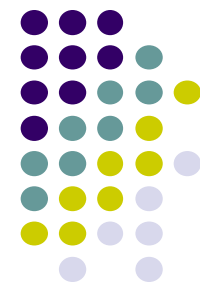
Hesse矩阵的计算工作量大，能否不算或者少算？





拟牛顿法

- **Hesse**矩阵的计算工作量大，有时目标函数的**Hesse**阵很难计算。
- 拟牛顿法利用目标函数和一阶导数，来构造目标函数的曲率近似，而不需要明显形成**Hesse**阵，同时具有收敛速度快的优点。



拟牛顿条件

f 在 \mathbf{x}_{k+1} 附近的二次近似为

$$f(\mathbf{x}) \approx f(\mathbf{x}_{k+1}) + \mathbf{g}_{k+1}^T (\mathbf{x} - \mathbf{x}_{k+1}) + \frac{1}{2} (\mathbf{x} - \mathbf{x}_{k+1})^T G_{k+1} (\mathbf{x} - \mathbf{x}_{k+1}),$$

对上式求导, 有

$$\mathbf{g}(\mathbf{x}) \approx \mathbf{g}_{k+1} + G_{k+1} (\mathbf{x} - \mathbf{x}_{k+1}),$$

令 $\mathbf{x} = \mathbf{x}_k$, $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$, $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$, 得

$$G_{k+1}^{-1} \mathbf{y}_k \approx \mathbf{s}_k$$

对于二次函数 f , 上述关系式精确成立。

要求在拟牛顿法中构造出 *Hesse* 逆近似 H_{k+1} , 满足

$$H_{k+1} \mathbf{y}_k = \mathbf{s}_k,$$

称为拟牛顿条件。或构造 *Hesse* 近似

$$B_{k+1} \mathbf{s}_k = \mathbf{y}_k$$

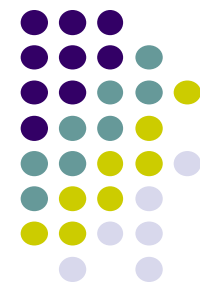


一般的拟牛顿算法

一般拟牛顿算法：

1. 给初值 $\mathbf{x}_0 \in R^n, H_0 \in R^{n \times n}, 0 \leq \varepsilon \leq 1, k = 0$.
2. 若 $\|\mathbf{g}_k\| \leq \varepsilon$, 则停止；否则，计算 $\mathbf{d}_k = -H_k \mathbf{g}_k$
3. 沿方向 \mathbf{d}_k 线性搜索求 α_k , 令 $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$
4. 校正 H_k 产生 H_{k+1} , 使得拟牛顿条件满足。
5. $k = k + 1$, 转步 2。

- 优点：
 - 仅需一阶导数
 - H_k 保持正定，具有下降性。
 - 迭代每次需要 $O(n^2)$ 次乘法。（牛顿法 $O(n^3)$ 次乘法）



对称秩一校正(SR1校正)

希望有 $H_{k+1} = H_k + E_k$, 其中 E_k 为一个低阶矩阵。在秩一校正情形下,

$$E_k = uv^T.$$

根据拟牛顿条件有, $(H_k + uv^T)y_k = s_k \Rightarrow (v^T y_k)u = s_k - H_k y_k$, 故

u 必在方向 $s_k - H_k y_k$ 上。

取 $u = \frac{1}{v^T y_k} (s_k - H_k y_k)$, 因为 $Hesse$ 为对称阵, 故取 $v = (s_k - H_k y_k)$ 。

那么 $H_{k+1} = H_k + \frac{(s_k - H_k y_k)(s_k - H_k y_k)^T}{(s_k - H_k y_k)^T y_k}$, 称为对称秩一校正 (SR1校正)。

性质

1. 对于二次函数, 不需要精确一维搜索, 具有 n 步终止性质, $H_n = G^{-1}$
2. 不能保持正定性, 仅当 $(s_k - H_k y_k)^T y_k > 0$ 时。但这个条件往往很难满足。这使得 SR1 校正正在应用中受到限制。



DFP校正

设秩二校正为 $H_{k+1} = H_k + auu^T + bvv^T$.

若要拟牛顿条件 $(H_k + auu^T + bvv^T)y_k = s_k$ 成立,

对于 u, v 一个明显的取法为 $u = s_k, v = H_k y_k$.

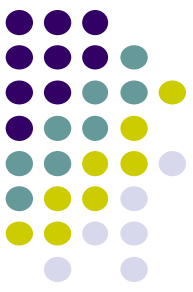
同时令: $au^T y_k = 1, bv^T y_k = -1$, 可知 (*) 恒成立。

$$\text{那么 } a = \frac{1}{s_k^T y_k}, b = -\frac{1}{y_k^T H_k y_k}$$

$$\text{那么 } H_{k+1} = H_k + \frac{s_k^T s_k}{s_k^T y_k} - \frac{H_k y_k y_k^T H_k^T}{y_k^T H_k y_k}, \text{ 称为 DFP 校正.}$$

性质

1. 对于二次函数, 精确一维搜索, 具有 n 步终止性质, $H_n = G^{-1}$
2. 对于二次函数, $H_0 = I$ 时, 产生共轭方向与共轭梯度。
3. 校正保持正定性, 下降性质成立。
4. 具有超线性收敛。
5. 采用精确线性搜索时, 对于凸函数, 总体收敛。



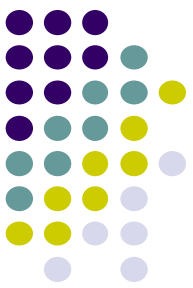
L-BFGS算法

- 一种非常流行和成功的拟牛顿法
- Limited memory Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm
- 有大量开源的实现

$$H_{i+1} = H_i + \frac{s_i s_i^T}{s_i^T q_i} - \frac{H_i q_i q_i^T H_i}{q_i^T H_i q_i} + [\nabla f(X_{i+1}) - \nabla f(X_i)]^T H_i [\nabla f(X_{i+1}) - \nabla f(X_i)] w w^T \quad \dots [14]$$

where

$$w = \frac{s_i}{s_i^T q_i} - \frac{H_i q_i}{q_i^T H_i q_i} \\ = H_i + \frac{X_{i+1} - X_i}{(X_{i+1} - X_i)^T [\nabla f(X_{i+1}) - \nabla f(X_i)]} - \frac{H_i [\nabla f(X_{i+1}) - \nabla f(X_i)]}{[\nabla f(X_{i+1}) - \nabla f(X_i)]^T H_i [\nabla f(X_{i+1}) - \nabla f(X_i)]} \quad \dots [15]$$



L-BFGS算法

Two loops approach $y_k = g_{k+1} - g_k$ $\rho_k = \frac{1}{y_k^T s_k}$

$$q = g_k$$

$$\text{For } i = k - 1, k - 2, \dots, k - m$$

$$\alpha_i = \rho_i s_i^T q$$

$$q = q - \alpha_i y_i$$

$$H_k = y_{k-1}^T s_{k-1} / y_{k-1}^T y_{k-1}$$

$$z = H_k q$$

$$\text{For } i = k - m, k - m + 1, \dots, k - 1$$

$$\beta_i = \rho_i y_i^T z$$

$$z = z + s_i (\alpha_i - \beta_i)$$

$$\text{Stop with } H_k g_k = z$$