

Foundation of Optimization

最优化原理

李晓鹏

Li Xiao-Peng

电子与信息工程学院

School of Electronic and Information Engineering

Semester-I 2025/26

第三章 无约束优化

- ① 迭代法概述
- ② 一维搜索方法(线搜索方法)
- ③ 最速下降法(梯度下降法)
- ④ 共轭梯度法
- ⑤ 牛顿法(牛顿法和阻尼牛顿法)
- ⑥ 拟牛顿法(变尺度法)
- ⑦ 信赖域法

本章框架

无约束最优化方法

- 一维搜索(求步长)
 - 平分法(二分法)
 - 黄金分割法(0.618法)
 - 牛顿法
- 确定搜索方向
 - 最速下降法
 - 共轭梯度法
 - 牛顿法/阻尼牛顿法
 - 拟牛顿法
 - 信赖域法

本章要求

重点、难点:

本章讨论无约束优化问题的求解算法, 包括精确求解步长的一维搜索方法, 以及求搜索方向的最速下降法、共轭梯度法、牛顿法等。这些方法也是求解约束优化问题的基础。

学习要求:

掌握几种常用的无约束优化方法的基本原理, 并会用这些方法手动求解小规模优化问题(Toy Optimization Problems).

目录

① 迭代法概述

② 一维搜索方法(线搜索方法)

③ 最速下降法(梯度下降法)

④ 共轭梯度法

⑤ 牛顿法(牛顿法和阻尼牛顿法)

⑥ 拟牛顿法(变尺度法)

⑦ 信赖域法

迭代法概述：基本思想

考虑求解如下无约束优化问题：

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

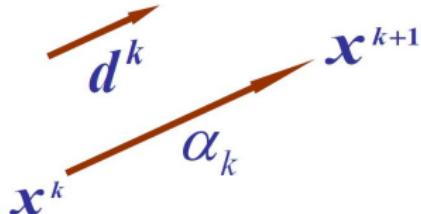
选取初始点 $\boldsymbol{x}^0 \in \mathbb{R}^n$, 按一定规则(迭代规则)产生点序列

$$\boldsymbol{x}^1, \boldsymbol{x}^2, \dots, \boldsymbol{x}^k$$

记为 $\{\boldsymbol{x}^k\}$, 使得 \boldsymbol{x}^k 恰好是优化问题的一个最优解, 或该点列 $\{\boldsymbol{x}^k\}$ 收敛到优化问题的一个最优解.

问题： 如何构造点列 $\{\boldsymbol{x}^k\}$? 也即, 如何由 $\boldsymbol{x}^k \rightarrow \boldsymbol{x}^{k+1}$.

迭代法概述：确定下降方向和步长



迭代的定义：

- 1) 在 x^k 处确定一个方向 d^k ;
- 2) 然后确定沿着射线 $x = x^k + \alpha d^k (\alpha > 0)$ 走多远, 即确定一个步长 α_k , 从而得到新迭代点

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k.$$

通常希望: $f(\mathbf{x}^{k+1}) \leq f(\mathbf{x}^k)$.

下降算法: 按某迭代规则得到的点列 $\{\mathbf{x}^k\}$ 满足 $f(\mathbf{x}^{k+1}) \leq f(\mathbf{x}^k)$.

迭代法概述：下降方向的确定

定义 3.1:

在点 x^k 处, 若对于方向 $p^k \neq 0$, 存在实数 $\delta > 0$, 使得对任意的 $\alpha \in (0, \delta)$ 有:

$$f(x^k + \alpha p^k) < f(x^k).$$

成立, 则称 p^k 为函数 $f(x)$ 在点 x^k 处的一个下降方向.

定理 3.1:

设函数 $f : \Omega \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$ 在点 x^k 处一阶可导, $p \in \mathbb{R}^n$ 为 n 维非零向量, 如果满足

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

则 p 为函数 $f(x)$ 在点 x^k 处的一个下降方向.

迭代法概述：下降方向的确定

证明： $f(\mathbf{x})$ 具有连续的一阶偏导数，由Taylor公式：

$$f(\mathbf{x}^k + \alpha \mathbf{p}) = f(\mathbf{x}^k) + \alpha \nabla f(\mathbf{x}^k)^T \mathbf{p} + o(\alpha).$$

当 $\nabla f(\mathbf{x}^k)^T \mathbf{p} < 0$ 时，有

$$f(\mathbf{x}^k + \alpha \mathbf{p}) < f(\mathbf{x}^k),$$

所以(α 充分小时) \mathbf{p} 是 $f(\mathbf{x})$ 在 \mathbf{x}^k 处的一个下降方向.

因此，满足 $\nabla f(\mathbf{x}^k)^T \mathbf{p} < 0$ 的方向 \mathbf{p} 为 $f(\mathbf{x})$ 在 \mathbf{x}^k 处的下降方向.

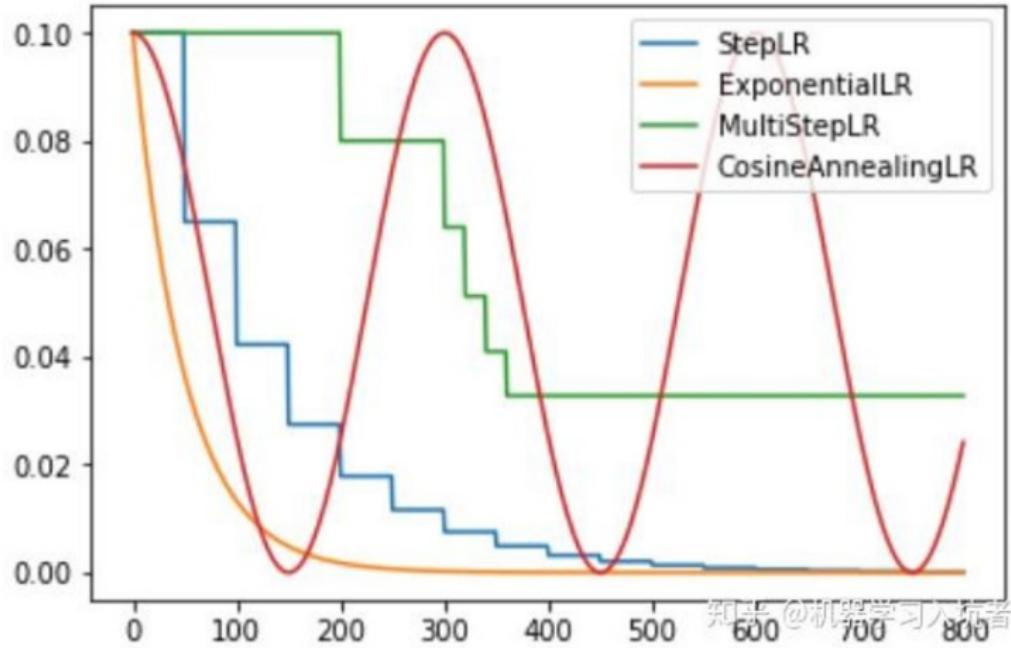
迭代法概述：最优步长的确定

确定了搜索方向 d^k 后，下一步就要确定步长 α_k ，常见的方法有三种：

- 1) 固定步长法：将步长 α_k 固定为一常数（例如令 $\alpha_k = 1$ ）。（固定步长法无法保证目标函数值持续下降。）
- 2) 可接受步长法：在能够保证目标函数值下降或整体下降的前提下，步长 α_k 可任意选取。
- 3) (精确)一维搜索或(精确)线搜索：沿搜索方向使目标函数值下降最多，即沿射线 $x = \mathbf{x}^k + \alpha \mathbf{d}^k (\alpha > 0)$ 求目标函数 $f(\mathbf{x}^k + \alpha \mathbf{d}^k)$ 的极小点。

迭代法概述：最优步长的确定

深度学习步长衰减策略示例



迭代法概述：最优步长的确定

具体而言，(精确)一维搜索或(精确)线搜索在确定步长 α_k 时， $\alpha_k > 0$ 通常满足：

$$f(\mathbf{x}^k + \alpha_k \mathbf{d}^k) = \min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k) \text{ 或 } \alpha_k = \arg \min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k),$$

即选择一个能够使目标函数值下降最多的步长，简称**最优步长**。这其实就是求单变量函数(一元函数)极小点的问题：

$$\min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k) \Leftrightarrow \min_{\alpha > 0} f(\alpha).$$

迭代法概述：迭代法步骤

给定初始点 \mathbf{x}^0 , 令 $k = 0$

- 1) 确定 \mathbf{x}^k 处的下降方向 \mathbf{d}^k ;
- 2) 确定步长 $\alpha_k > 0$ 使得目标函数值下降, 也即:

$$f(\mathbf{x}^k + \alpha_k \mathbf{d}^k) \leq f(\mathbf{x}^k).$$

- 3) 令 $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$;
- 4) 若 \mathbf{x}^{k+1} 满足终止准则, 停止迭代; 否则令 $k = k + 1$, 转Step 1).

迭代法概述：算法设计的关键点

1. **收敛性**: 局部收敛, 全局收敛.
2. **收敛速度**: 线性收敛, 超线性收敛, 二阶收敛.
3. **终止准则**: 迭代结束的条件, 也即逼近最优解的程度的考量.

考虑问题 $\min f(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^n$, 记 \mathbf{x}^* 为其最优解. 给定初始点 \mathbf{x}^0 , 设按某种迭代规则(算法)产生的点列为 $\{\mathbf{x}^k\}$.

迭代法概述：算法设计的关键点

收敛性定义

- 算法收敛：存在正整数 N 满足： $\mathbf{x}^N = \mathbf{x}^*$ 或 $\lim_{k \rightarrow +\infty} \mathbf{x}^k = \mathbf{x}^*$.
- 局部收敛：存在 \mathbf{x}^* 的某邻域 $N_\delta(\mathbf{x}^*)$ ，使得仅当 $\mathbf{x}^0 \in N_\delta(\mathbf{x}^*)$ 时，算法产生的点列 $\{\mathbf{x}^k\}$ 才收敛于 \mathbf{x}^* ，称该算法具有局部收敛性质.
- 全局收敛：在定义域内任取初值 \mathbf{x}^0 ，由算法产生的点列 $\{\mathbf{x}^k\}$ 都收敛于 \mathbf{x}^* ，称该算法具有全局收敛性质.

迭代法概述：算法设计的关键点

收敛速度

- 线性收敛：设 \mathbf{x}^* 为问题 $\min f(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^n$ 的最优解. 由算法A产生的序列 $\{\mathbf{x}^k\}$ 收敛于 \mathbf{x}^* , 即

$$\lim_{k \rightarrow +\infty} \mathbf{x}^k = \mathbf{x}^*.$$

如果存在一个与 k 无关的常数 $\beta \in (0, 1)$ 以及某个正整数 k_0 , 使得当 $k \geq k_0$ 时, 有

$$\|\mathbf{x}^{k+1} - \mathbf{x}^*\|_2 \leq \beta \|\mathbf{x}^k - \mathbf{x}^*\|_2 \quad \dots \dots (I)$$

成立, 则称序列 $\{\mathbf{x}^k\}$ 线性收敛, 或称算法A是线性收敛的.

迭代法概述：算法设计的关键点

不失一般性，可以假设(I)式对于任意 $k \geq 0$ 都成立，于是有：

$$\begin{aligned}\|\mathbf{x}^k - \mathbf{x}^*\|_2 &\leq \beta \|\mathbf{x}^{k-1} - \mathbf{x}^*\|_2 \\ &\leq \beta^2 \|\mathbf{x}^{k-2} - \mathbf{x}^*\|_2 \\ &\leq \cdots \leq \beta^k \|\mathbf{x}^0 - \mathbf{x}^*\|_2\end{aligned}$$

即 $\lim_{k \rightarrow +\infty} \|\mathbf{x}^k - \mathbf{x}^*\|_2 = 0$.

当 $k \rightarrow \infty$ 时，点 \mathbf{x}^k 到 \mathbf{x}^* 的距离，大致以公比为 β 的等比序列(几何序列)减小，因此线性收敛速度相当于相应的等比数列的收敛速度.

迭代法概述：算法设计的关键点

- p 阶收敛：设由算法A产生的序列 $\{\mathbf{x}^k\}$ 收敛于 \mathbf{x}^* . 如果存在一个与 k 无关的常数 $\beta > 0$ 和 $p > 1$, 及某个正整数 k_0 , 使得当 $k \geq k_0$ 时,

$$\|\mathbf{x}^{k+1} - \mathbf{x}^*\|_2 \leq \beta \|\mathbf{x}^k - \mathbf{x}^*\|_2^p$$

成立, 则称序列 $\{\mathbf{x}^k\}$ 为 p 阶收敛的, 或称序列 $\{\mathbf{x}^k\}$ 的收敛阶为 p , 或称算法A是 p 阶收敛的.

【注】

- 当 $1 < p < 2$ 时, 则称序列 $\{\mathbf{x}^k\}$ 为超线性收敛的.
- 当 $p = 2$ 时, 则称序列 $\{\mathbf{x}^k\}$ 为二阶收敛的.
- 该定义中, 数值解与最优解间的差异性用范数 $\|\cdot\|_2$ 来刻画. 可将 $\|\cdot\|_2$ 理解为数值解与最优解间的距离.

迭代法概述：算法设计的关键点

终止准则

假设算法A产生的序列 $\{x^k\}$ 收敛于最优解 x^* . 那么, 算法何时停止迭代?

迭代法概述：算法设计的关键点

终止准则

假设算法A产生的序列 $\{\mathbf{x}^k\}$ 收敛于最优解 \mathbf{x}^* . 那么, 算法何时停止迭代?

由于 \mathbf{x}^* 未知, 所以 $\|\mathbf{x}^k - \mathbf{x}^*\|_2 \leq \epsilon (\epsilon > 0)$ 无法作为停机准则.

若算法可以收敛, 则当 $\|\mathbf{x}^k - \mathbf{x}^*\|_2$ 较小时, $\|\mathbf{x}^{k+1} - \mathbf{x}^k\|_2$ 也较小. 所以终止准则通常取为:

$$\|\mathbf{x}^{k+1} - \mathbf{x}^k\|_2 \leq \epsilon_1 \text{ 或 } \|\mathbf{x}^{k+1} - \mathbf{x}^k\|_2 \leq \epsilon'_1 \|\mathbf{x}^k\|_2$$

其他终止条件如:

$$|f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k)| \leq \epsilon_2 \text{ 或 } |f(\mathbf{x}^{k+1}) - f(\mathbf{x}^k)| \leq \epsilon'_2 |f(\mathbf{x}^k)|.$$

另外若函数一阶可导, 通常可采取: $\|\nabla f(\mathbf{x}^k)\|_2 \leq \epsilon_3$.

目录

- ① 迭代法概述
- ② 一维搜索方法(线搜索方法)
- ③ 最速下降法(梯度下降法)
- ④ 共轭梯度法
- ⑤ 牛顿法(牛顿法和阻尼牛顿法)
- ⑥ 拟牛顿法(变尺度法)
- ⑦ 信赖域法

一维搜索相关概念

前面提到, 为了尽快求得最优解, 很多算法在确定步长 α_k 时通常要
求 $\alpha_k > 0$ 满足:

$$f(\mathbf{x}^k + \alpha_k \mathbf{d}^k) = \min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k) \text{ 或 } \alpha_k = \arg \min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k),$$

即选择一个使函数值下降最多的步长, 简称最优步长.

这其实就是求单变量函数(一元函数)极小点问题:

$$\min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k) \Leftrightarrow \min_{\alpha > 0} f(\alpha),$$

也称为**一维搜索问题**或**线搜索问题**.

一维搜索相关概念

定义 3.2:

- 一维搜索, 是指在一维空间中求最优化问题最优解的**过程**.
- 一维搜索问题, 是指单变量函数(一元函数)极大或极小**问题**.

一维搜索方法(步长求法):

- 解析法: 令 $\frac{df(x)}{dx} = 0$
- 数值解法:
 - 试探法
 - 函数逼近法

一维搜索相关概念：试探法

基本思想

从一个包含极小点的初始区间 $[a, b]$ 开始，按照一定规则往区间内放入试探点，通过比较试探点与区间端点的函数值或导数值，不断缩短包含极小点的搜索区间。

当区间长度缩小到一定程度(达到误差精度要求)，那么区间上任意一点都可以作为极小点的近似。通常取最终区间的中点作为近似极小点。

代表算法：平分法和黄金分割法。

一维搜索相关概念：函数逼近法

基本思想

在包含极小点的搜索区间中不断用简单多项式(通常不超过三次)来近似目标函数，并逐步用多项式的极小点来逼近一维搜索问题的极小点.

代表算法：牛顿法.

一维搜索相关概念：函数逼近法

基本思想

在包含极小点的搜索区间中不断用简单多项式(通常不超过三次)来近似目标函数，并逐步用多项式的极小点来逼近一维搜索问题的极小点.

代表算法：牛顿法.

【注】无论试探法还是函数逼近法，我们都需要事先知道一个包含极小点的初始搜索区间.

一维搜索相关概念：搜索区间

定义 3.3: 搜索区间

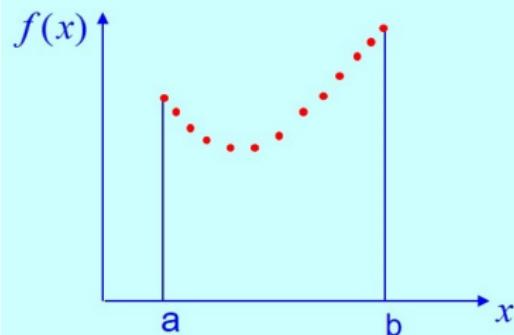
设 $f(x)$ 是定义在一元空间上的函数, $x^* = \arg \min_x f(x)$. 若存在区间 $[a, b]$ 使得 $x^* \in (a, b)$, 则称 $[a, b]$ 为一维搜索问题的搜索空间.

定义 3.4: 单谷函数

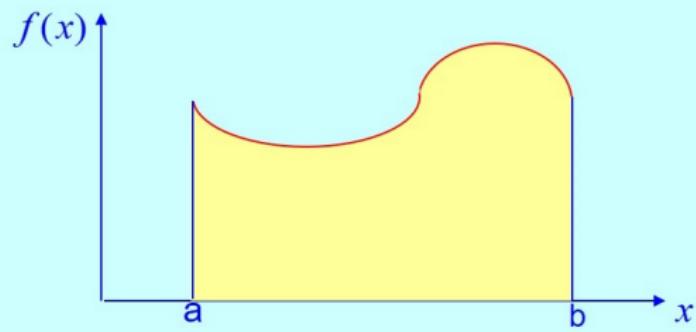
若 x^* 使得 $f(x)$ 在区间 $[a, x^*]$ 上严格递减, 并且 $[x^*, b]$ 上严格递增, 则称 $f(x)$ 为搜索区间 $[a, b]$ 上的单谷函数.

一维搜索相关概念：搜索区间

单谷函数示例：



离散单谷函数



非单谷函数

一维搜索方法：初始搜索区间的确定(法一)

法一

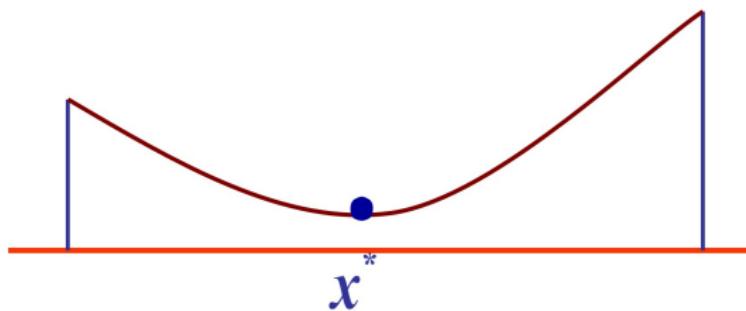
设单谷函数 $f(x)$ 在某定义域内一阶连续可导, 则可按如下方法确定初始搜索区间:

- 1) 任意选一个初始点 x_0 以及步长 Δx ;
- 2) 若 $f'(x_0) < 0$, 则极小点位于 x_0 的右侧, 那么令 $x_1 = x_0 + \Delta x$:
 - 若 $f'(x_1) > 0$, 则令 $[a, b] = [x_0, x_1]$.
 - 若 $f'(x_1) < 0$, 则令 $x_0 = x_1$, 返回Step 2).
- 3) 若 $f'(x_0) > 0$, 则作类似于Step 2)的情况去讨论.

一维搜索方法：初始搜索区间的确定(进退法)

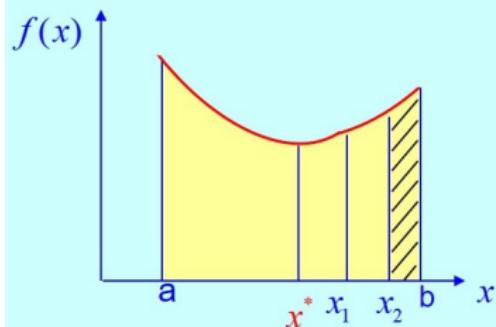
法二(进退法)

设 $f(x)$ 是定义在 $[a, b]$ 上的单谷函数，即 $f(x)$ 在 $[a, b]$ 上有唯一的极小点 x^* ，那么在 x^* 左边 $f(x)$ 严格下降，在 x^* 右边 $f(x)$ 严格上升。

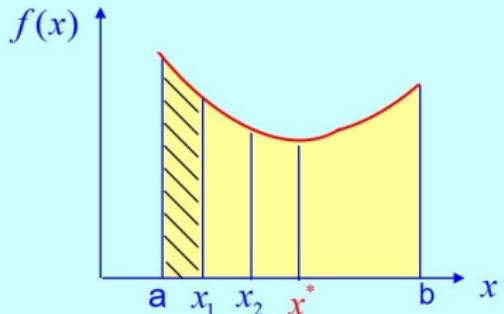


一维搜索方法：初始搜索区间的确定(进退法)

若 $f(x_1) \leq f(x_2)$, 则 $x^* \in [a, x_2]$;



若 $f(x_1) \geq f(x_2)$, 则 $x^* \in [x_1, b]$.



定理 3.2:

- (1) 若 $f(x_1) < f(x_2) < f(x_3)$, 则极小点位于点 x_2 的左侧.
- (2) 若 $f(x_1) > f(x_2) > f(x_3)$, 则极小点位于点 x_2 的右侧.
- (3) 若 $f(x_1) > f(x_2) < f(x_3)$, 则极小点位于点 x_1 和 x_3 之间,
故 $[x_1, x_3]$ 是一个包含极小点的区间.

一维搜索方法：初始搜索区间的确定(进退法)

设从定义域内任取一个初始点 x_1 , 那么如何求 $f(x)$ 的一个包含极小点的区间 $[x_1, x_3]$?

一维搜索方法：初始搜索区间的确定(进退法)

设从定义域内任取一个初始点 x_1 , 那么如何求 $f(x)$ 的一个包含极小点的区间 $[x_1, x_3]$?

结合单谷函数的定义可知, 上述初始区间的问题可归结为:
如何在 x_1 的右边找两个点 $x_2 < x_3$, 使得:

$$f(x_1) \geq f(x_2) \leq f(x_3)$$

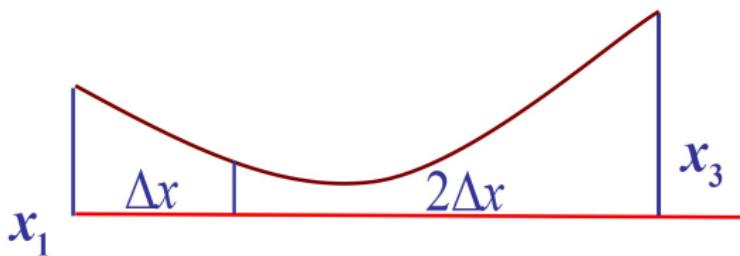
首先, 令 $x_2 = x_1 + \Delta x$, $\Delta x > 0$ 为给定的步长.

下面分两种情况讨论:

一维搜索方法：初始搜索区间的确定(进退法)

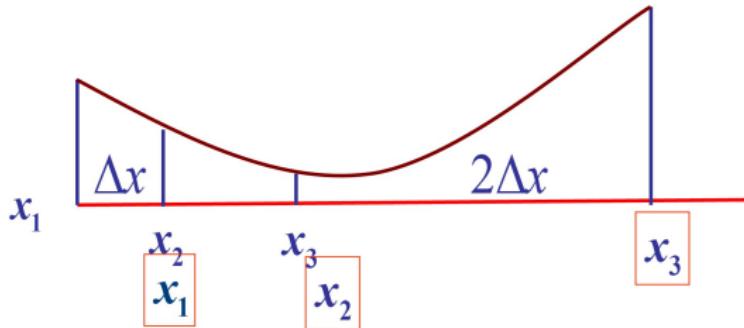
(1) $f(x_1) \geq f(x_2)$

此时 x_1 位于极小点的左侧，并且 x_2 取值小，因此加大步长向右搜索，令 $\Delta x = 2\Delta x$, $x_3 = x_2 + \Delta x$.



若 $f(x_2) \leq f(x_3)$, 则包含极小值点的区间为 $[x_1, x_3]$.

一维搜索方法：初始搜索区间的确定(进退法)

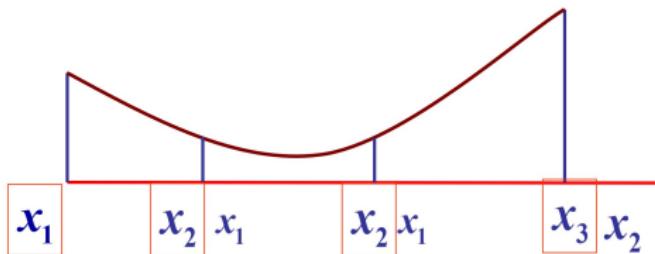


若 $f(x_2) > f(x_3)$, 则取的步长偏小.

令 $x_1 = x_2$, $x_2 = x_3$, $\Delta x = 2\Delta x$, $x_3 = x_2 + \Delta x$, 继续判断, 直到满足 $f(x_2) \leq f(x_3)$.

一维搜索方法：初始搜索区间的确定(进退法)

(2) $f(x_1) < f(x_2)$



此时, x_2 位于极小点的右侧, 则向左搜索.

令 $\Delta x = 2\Delta x$, $x_3 = x_2$, $x_2 = x_1$, $x_1 = x_2 - \Delta x$.

若 $f(x_1) \geq f(x_2)$, 则包含极小值点的区间即为 $[x_1, x_3]$. 否则, 继续向左寻找 x_1 , 直到 $f(x_1) \geq f(x_2)$.

一维搜索方法：初始搜索区间的确定(进退法)

给定初始点 x_1 , 初始步长 $\Delta x > 0$

- 1) $x_2 = x_1 + \Delta x$, 计算 $f(x_1)$ 、 $f(x_2)$, 若 $f(x_1) \geq f(x_2)$, 则转Step 2); 否则, 转Step 3);
- 2) 令 $\Delta x = 2\Delta x$, $x_3 = x_2 + \Delta x$, 计算 $f(x_3)$.
若 $f(x_2) \leq f(x_3)$, 则得区间 $[x_1, x_3]$ 为初始区间, 停止搜索;
若 $f(x_2) > f(x_3)$, 则令 $x_1 = x_2$, $x_2 = x_3$, 转Step 2).
- 3) 令 $x_3 = x_2$, $x_2 = x_1$, $\Delta x = 2\Delta x$, $x_1 = x_2 - \Delta x$, 计算 $f(x_1)$.
若 $f(x_1) \geq f(x_2)$, 则取 $[x_1, x_3]$ 为初始区间, 停止搜索;
若 $f(x_1) < f(x_2)$, 转Step 3).

一维搜索方法：二(平)分法

设 $f(x)$ 为单谷函数且在 $[a_k, b_k]$ 内一阶连续可导,
且 $f'(a_k) < 0$ 和 $f'(b_k) > 0$.

取 $c_k = \frac{(a_k + b_k)}{2}$, 若 $f'(c_k) = 0$, 则 c_k 为极小点;

若 $f'(c_k) > 0$, 则令 $a_{k+1} = a_k, b_{k+1} = c_k$,

若 $f'(c_k) < 0$, 则令 $a_{k+1} = c_k, b_{k+1} = b_k$.

【注】 二分法适用于 $f(x)$ 的导数存在且容易计算的情况下.

一维搜索方法：二(平)分法

二(平)分法算法步骤

- 1) 给定初始区间 $[a_1, b_1]$, 满足 $f'(a_1) < 0, f'(b_1) > 0$, 且允许误差为 $\epsilon > 0$, 令 $k = 1$
- 2) 若 $|b_k - a_k| \leq \epsilon$, 停止运算; 否则, 取 $c_k = \frac{a_k + b_k}{2}$, 转Step 3).
- 3) 计算 $f'(c_k)$.
若 $f'(c_k) = 0$, 则令 $x^* = c_k$, 停止运算;
若 $f'(c_k) < 0$, 则令 $a_{k+1} = c_k, b_{k+1} = b_k$, 转Step 4);
若 $f'(c_k) > 0$, 则令 $a_{k+1} = a_k, b_{k+1} = c_k$, 转Step 4).
- 4) 令 $k = k + 1$, 返回Step 2).

一维搜索方法：二(平)分法

优点：

- 每次以搜索区间的中点作为试探点，计算量小，算法简洁易于实现；
- 总能收敛到一个局部极小点。

缺点：

- 要求目标函数至少一阶连续可导；
- 收敛速度很慢。

一维搜索方法：二(平)分法

例 3.1: 设 $f(x) = x^2 + 1$, 试用平分法求 $[-1, 2]$ 中的极小点, 给定允许误差(精度) $\epsilon = 0.05$.

一维搜索方法：二(平)分法

例 3.1: 设 $f(x) = x^2 + 1$, 试用平分法求 $[-1, 2]$ 中的极小点, 给定允许误差(精度) $\epsilon = 0.05$.

解: $f'(x) = 2x$

k	a_k	b_k	c_k	$f'(c_k)$
1	-1	2	1/2	1 > 0
2	-1	1/2	-1/4	-1/2 < 0
3	-1/4	1/2	1/8	1/4 > 0
4	-1/4	1/8	-1/16	-1/8 < 0
5	-1/16	1/8	1/32	1/16 > 0
6	-1/16	1/32	-1/64	-1/32 < 0
7	-1/64	1/32		

第7步时, 区间长度

$$\left| \frac{1}{32} - \left(-\frac{1}{64} \right) \right| = \frac{3}{64} \\ \approx 0.047 < 0.05$$

近似极小点为:

$$\tilde{x} = \frac{a_7+b_7}{2} = 0.0078125$$

一维搜索方法：二(平)分法

例 3.2: 设 $f(x) = x^2 - 3$, 试用平分法求 $[-3, 5]$ 中的极小点, 给定允许误差(精度) $\epsilon = 0.1$.

一维搜索方法：二(平)分法

例 3.2: 设 $f(x) = x^2 - 3$, 试用平分法求 $[-3, 5]$ 中的极小点, 给定允许误差(精度) $\epsilon = 0.1$.

解: $f'(x) = 2x$

k	a_k	b_k	c_k	$f'(c_k)$
1	-3	5	1	2
2	-3	1	-1	-2
3	-1	1	0	0

一维搜索方法：黄金分割法

黄金分割法的特点：

- 1) 在搜索区间内 $[a_k, b_k]$ 取两个试探点 λ_k, μ_k , 将区间分成三段.
- 2) 通过比较试探点处的函数值, 使搜索区间缩短; 不断迭代使得包含极小点的搜索区间不断缩短, 最终得到极小点或近似极小点。

【注】 黄金分割法对函数的要求较低, 函数可以不连续。因为该算法仅利用函数值而非函数的导数来缩小搜索区间。

一维搜索方法：黄金分割法

记当前的搜索区间为 $[a_k, b_k]$, 设选取的试探点为 λ_k, μ_k , 其中 $\lambda_k < \mu_k$,

若 $f(\lambda_k) \leq f(\mu_k)$, 则 $x^* \in [a_k, \mu_k]$;

若 $f(\lambda_k) > f(\mu_k)$, 则 $x^* \in [\lambda_k, b_k]$.

一维搜索方法：黄金分割法

记当前的搜索区间为 $[a_k, b_k]$, 设选取的试探点为 λ_k, μ_k , 其中 $\lambda_k < \mu_k$,

若 $f(\lambda_k) \leq f(\mu_k)$, 则 $x^* \in [\lambda_k, \mu_k]$;

若 $f(\lambda_k) > f(\mu_k)$, 则 $x^* \in [\mu_k, b_k]$.

【注】函数值大者, 留下作边界.

那么下一个搜索区间取为 $[a_k, \mu_k]$ 或 $[\lambda_k, b_k]$, 为了机会均等, 我们选取对称的试点, 即

$$\mu_k - a_k = b_k - \lambda_k \quad \dots \dots \dots \quad (1)$$

另外, 要求区间缩短率相同, 即

$$b_{k+1} - a_{k+1} = \rho(b_k - a_k) \quad \dots \dots \dots \quad (2)$$

其中 $0 < \rho < 1$ 为区间缩短率.

一维搜索方法：黄金分割法

分情况讨论：

若 $f(\lambda_k) \leq f(\mu_k)$, 则取 $[a_{k+1}, b_{k+1}] = [a_k, \mu_k]$. 再结合式(1)(2)得：

$$\begin{aligned}\lambda_k &= a_k + (1 - \rho)(b_k - a_k), \\ \mu_k &= a_k + \rho(b_k - a_k)\end{aligned} \dots\dots \text{(I)}$$

同理, 当 $f(\lambda_k) > f(\mu_k)$ 时, 可证上两式也成立。

可见, 一旦 $0 < \rho < 1$ 确定, λ_k, μ_k 便可算出。

一维搜索方法：黄金分割法

分情况讨论：

若 $f(\lambda_k) \leq f(\mu_k)$, 则取 $[a_{k+1}, b_{k+1}] = [a_k, \mu_k]$. 再结合式(1)(2)得：

$$\begin{aligned}\lambda_k &= a_k + (1 - \rho)(b_k - a_k), \\ \mu_k &= a_k + \rho(b_k - a_k)\end{aligned} \dots\dots \text{(I)}$$

同理, 当 $f(\lambda_k) > f(\mu_k)$ 时, 可证上两式也成立。

可见, 一旦 $0 < \rho < 1$ 确定, λ_k, μ_k 便可算出。

那么 ρ 应该取多少呢?

一维搜索方法：黄金分割法

类似上一页的做法，再分情况讨论，发现

若 $f(\lambda_k) \leq f(\mu_k)$ ，那么 $[a_{k+1}, b_{k+1}] = [a_k, \mu_k]$ ，进一步有：

$$\mu_{k+1} = a_{k+1} + \rho(b_{k+1} - a_{k+1}) = a_k + \rho^2(b_k - a_k),$$

又注意到 $\lambda_k = a_k + (1 - \rho)(b_k - a_k)$. 因此，当 $\rho^2 = 1 - \rho$ 时，
 $\mu_{k+1} = \lambda_k$ 。此时，第 $k + 1$ 步无需计算 μ_{k+1} 。

一维搜索方法：黄金分割法

类似上一页的做法，再分情况讨论，发现

若 $f(\lambda_k) \leq f(\mu_k)$, 那么 $[a_{k+1}, b_{k+1}] = [a_k, \mu_k]$, 进一步有：

$$\mu_{k+1} = a_{k+1} + \rho(b_{k+1} - a_{k+1}) = a_k + \rho^2(b_k - a_k),$$

又注意到 $\lambda_k = a_k + (1 - \rho)(b_k - a_k)$. 因此, 当 $\rho^2 = 1 - \rho$ 时,
 $\mu_{k+1} = \lambda_k$ 。此时, 第 $k + 1$ 步无需计算 μ_{k+1} 。

同理, 当 $f(\lambda_k) > f(\mu_k)$, 那么 $[a_{k+1}, b_{k+1}] = [\lambda_k, b_k]$,
若 $\rho^2 = 1 - \rho$, $\lambda_{k+1} = \mu_k$, 即第 $k + 1$ 步无需计算 λ_{k+1} 。

【注】函数值小者留下作为新区间的一个试探点。

一维搜索方法：黄金分割法

解方程 $\rho^2 = 1 - \rho$, $\rho > 0$, 得 $\rho = \frac{\sqrt{5}-1}{2} \approx 0.618$, $\rho = \frac{-\sqrt{5}-1}{2}$ (舍).

于是得到试探点的计算公式：

$$\begin{aligned}\lambda_k &= a_k + 0.382(b_k - a_k), \\ \mu_k &= a_k + 0.618(b_k - a_k).\end{aligned}$$

由于区间缩短率 ρ 满足

$$\frac{\rho}{1 - \rho} = \frac{1 - \rho}{\rho}$$

是黄金分割数, 故该方法称为**黄金分割法或0.618法**.

一维搜索方法：黄金分割法

- 1) 设置初始区间 $[a_1, b_1]$ 和允许误差 $\epsilon > 0$, 计算试探点
 $\lambda_1 = a_1 + 0.382(b_1 - a_1)$, $\mu_1 = a_1 + 0.618(b_1 - a_1)$ 及对应的函数值 $f(\lambda_1), f(\mu_1)$, 令 $k = 1$;
- 2) 若 $|b_k - a_k| < \epsilon$, 则停止运算, 取 $x^* = \frac{a_k + b_k}{2}$;
否则, 当 $f(\lambda_k) > f(\mu_k)$ 时, 转Step 3);
当 $f(\lambda_k) \leq f(\mu_k)$ 时, 转Step 4);
- 3) 令 $[a_{k+1}, b_{k+1}] = [\lambda_k, b_k]$, $\lambda_{k+1} = \mu_k$, $f(\lambda_{k+1}) = f(\mu_k)$,
计算 $\mu_{k+1} = a_{k+1} + 0.618(b_{k+1} - a_{k+1})$ 及 $f(\mu_{k+1})$, 转Step 5);
- 4) 令 $[a_{k+1}, b_{k+1}] = [a_k, \mu_k]$, $\mu_{k+1} = \lambda_k$, $f(\mu_{k+1}) = f(\lambda_k)$,
计算 $\lambda_{k+1} = a_{k+1} + 0.382(b_{k+1} - a_{k+1})$ 及 $f(\lambda_{k+1})$, 转Step 5);
- 5) 令 $k = k + 1$, 返回Step 2)。

一维搜索方法：黄金分割法

例 3.3: 用黄金分割法求函数 $f(x) = x^2 - x + 2$ 在区间 $[-1, 3]$ 上的极小点。要求最终区间长度不大于原始区间长度的 0.08 倍。

一维搜索方法：黄金分割法

例 3.3: 用黄金分割法求函数 $f(x) = x^2 - x + 2$ 在区间 $[-1, 3]$ 上的极小点。要求最终区间长度不大于原始区间长度的 0.08 倍。

解：

$$\lambda_k = a_k + 0.382(b_k - a_k), \mu_k = a_k + 0.618(b_k - a_k).$$

初始区间为 $[a_1, b_1] = [-1, 3]$,

误差精度为 $\epsilon = [3 - (-1)] \times 0.08 = 0.32$ 。

一维搜索方法：黄金分割法

第一次迭代: $[a_1, b_1] = [-1, 3]$

$$\lambda_1 = a_1 + 0.382(b_1 - a_1) = 0.528, \quad f(\lambda_1) = 1.751$$

$$\mu_1 = a_1 + 0.618(b_1 - a_1) = 1.472, \quad f(\mu_1) = 2.695$$

$f(\lambda_1) < f(\mu_1)$, 故取下一个搜索区间为 $[-1, 1.472]$.

第二次迭代: $[a_2, b_2] = [-1, 1.472]$

$$\mu_2 = \lambda_1 = 0.528, \quad f(\mu_2) = f(\lambda_1) = 1.751$$

$$\lambda_2 = a_2 + 0.382(b_2 - a_2) = -0.056, \quad f(\lambda_2) = 2.059$$

$f(\lambda_2) > f(\mu_2)$, 故取下一个搜索区间为 $[-0.056, 1.472]$ 。

一维搜索方法：黄金分割法

迭代次数	$[a_k, b_k]$	λ_k	μ_k	$f(\lambda_k)$	$f(\mu_k)$	$ b_k - a_k < \varepsilon$
1	$[-1, 3]$	0.528	1.472	1.751	2.695	否
2	$[-1, 1.472]$	-0.056	0.528	2.059	1.751	否
3	$[-0.056, 1.472]$	0.528	0.888	1.751	1.901	否
4	$[-0.056, 0.888]$	0.305	0.528	1.788	1.751	否
5	$[0.305, 0.888]$	0.528	0.665	1.751	1.777	否
6	$[0.305, 0.665]$	0.443	0.528	1.753	1.751	否
7	$[0.443, 0.665]$					是
$x^* = (0.443 + 0.665) / 2 = 0.554$						

一维搜索方法：黄金分割法

例 3.4: 设 $f(x) = x^2 - 3$, 试用黄金分割法求 $[-3, 5]$ 中的极小点, 给定允许误差(精度) $\epsilon = 0.3$.

一维搜索方法：黄金分割法

例 3.4: 设 $f(x) = x^2 - 3$, 试用黄金分割法求 $[-3, 5]$ 中的极小点, 给定允许误差(精度) $\epsilon = 0.3$.

解:

k	$[a_k, b_k]$	λ_k	μ_k	$f(\lambda_k)$	$f(\mu_k)$	≤ 0.3
1	[-3, 5]	0.056	1.944	-2.997	0.779	
2	[-3, 1.944]	-1.111	0.055	-1.766	-2.997	
3	[-1.111, 1.944]	0.056	0.777	-2.997	-2.396	
4	[-1.111, 0.777]	-0.390	0.056	-2.848	-2.997	
5	[-0.390, 0.777]	0.056	0.331	-2.997	-2.890	
6	[-0.390, 0.331]	-0.115	0.056	-2.987	-2.997	
7	[-0.115, 0.331]	0.056	0.161	-2.997	-2.974	
8	[-0.115, 0.161]	-0.010	0.056	-3.000	-2.997	是

$$f((0.161 - 0.115)/2) = -2.9995$$

一维搜索方法：黄金分割法

优点：

- 对函数要求低，不要求函数可微，只需要计算函数值；
- 新区间里包含原来的试探点，且该试探点在下一步被利用(即每次只需计算一个试探点)，计算量小；
- 试探点的计算公式一致.

收敛性：

黄金分割法是线性收敛速度.

一维搜索方法：牛顿法(函数逼近法)

基本思想：

在极小点附近任取一个初始点，用迭代点处的二阶Taylor多项式作为原函数的近似(即用该函数逼近原函数)，然后求近似函数的极小点。若所求得的点不是原问题的极小点，便让它作为下一个迭代点；依次迭代下去，直到求得原问题的极小点或近似极小点。

【注】 牛顿法适用于二阶连续可导函数求极值问题。

一维搜索方法：牛顿法(函数逼近法)

设当前迭代点为 x^k , 则 $f(x)$ 在 x^k 处的二次近似函数为:

$$g_k(x) = f(x^k) + f'(x^k)(x - x^k) + \frac{1}{2}f''(x^k)(x - x^k)^2.$$

两边对 x 求导, 并令 $g'_k(x) = 0$, 则有:

$$f'(x^k) + f''(x^k)(x - x^k) = 0.$$

解之得: $x = x^k - \frac{f'(x^k)}{f''(x^k)}$.

若该点不是(近似)极小点, 则构造迭代格式令

$$x^{k+1} = x^k - \frac{f'(x^k)}{f''(x^k)},$$

即得牛顿法的迭代公式, 简称牛顿迭代公式.

一维搜索方法：牛顿法(函数逼近法)

- 1) 给定初始点 $x^0 \in \mathbb{R}$, 允许误差 $0 \leq \epsilon \ll 1$, 令 $k = 0$
- 2) 计算 $f'(x^k)$, 如果 $|f'(x^k)| < \epsilon$, 停止运算. 否则计算 $f''(x^k)$, 并令

$$x^{k+1} = x^k - \frac{f'(x^k)}{f''(x^k)}.$$

- 3) 令 $k = k + 1$, 返回 Step 2).

一维搜索方法：牛顿法(函数逼近法)

例 3.5: 用牛顿法求函数 $f(x) = 2x^2 - x - 1$ 的极小点, 初始点 $x^0 = 0$, 允许误差 $\epsilon = 0.05$.

一维搜索方法：牛顿法(函数逼近法)

例 3.5: 用牛顿法求函数 $f(x) = 2x^2 - x - 1$ 的极小点, 初始点 $x^0 = 0$, 允许误差 $\epsilon = 0.05$.

解: $f'(x) = 4x - 1$, $f''(x) = 4$.

迭代次数	x^k	$f'(x^k)$	$f''(x^k)$
0	0	-1	4
1	1/4	0	

可见, 经1步迭代得到极小点 $x^* = \frac{1}{4} = 0.25$.

一维搜索方法：牛顿法(函数逼近法)

例 3.6: 设 $f(x) = x^2 - 3$, 试用牛顿法求 $f(x)$ 的极小点, 初始点 $x^0 = 1$, 允许误差(精度) $\epsilon = 0.3$.

一维搜索方法：牛顿法(函数逼近法)

例 3.6: 设 $f(x) = x^2 - 3$, 试用牛顿法求 $f(x)$ 的极小点, 初始点 $x^0 = 1$, 允许误差(精度) $\epsilon = 0.3$.

解: $f'(x) = 2x$, $f''(x) = 2$.

迭代次数	x^k	$f'(x^k)$	$f''(x^k)$
0	1	2	2
1	0	0	

可见, 经1步迭代得到极小点 $x^* = 0$.

一维搜索方法：牛顿法(函数逼近法)

优点：

- 对于严格凸二次函数求极值，一步迭代即可找到最优解；
- 如果初始点距离极小点较近，算法会很快收敛于极小点。

缺点：

- 对初始点要求高，若初始点距离极小点太远，则迭代过程可能不收敛，也可能收敛到极大点，即牛顿法的收敛性依赖于初始点的选择；
- 对函数要求高，牛顿法适用于二阶连续可导函数求极值问题；
- 需计算二阶导数，计算复杂度高。

目录

- ① 迭代法概述
- ② 一维搜索方法(线搜索方法)
- ③ 最速下降法(梯度下降法)
- ④ 共轭梯度法
- ⑤ 牛顿法(牛顿法和阻尼牛顿法)
- ⑥ 拟牛顿法(变尺度法)
- ⑦ 信赖域法

最速下降法（梯度下降法）

给定函数 $f(\mathbf{x}) = f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T (\mathbf{x} - \mathbf{x}^k) + o(\|\mathbf{x} - \mathbf{x}^k\|_2)$.

问题: 在点 \mathbf{x}^k 处, 沿什么方向 \mathbf{d}^k , $f(\mathbf{x})$ 下降最快?

最速下降法（梯度下降法）

给定函数 $f(\mathbf{x}) = f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T (\mathbf{x} - \mathbf{x}^k) + o(\|\mathbf{x} - \mathbf{x}^k\|_2)$.

问题：在点 \mathbf{x}^k 处，沿什么方向 \mathbf{d}^k , $f(\mathbf{x})$ 下降最快？

分析： $f(\mathbf{x}^k + \alpha \mathbf{d}^k) = f(\mathbf{x}^k) + \alpha \nabla f(\mathbf{x}^k)^T \mathbf{d}^k + o(\alpha)$.

由于： $\nabla f(\mathbf{x}^k)^T \mathbf{d}^k = \|\nabla f(\mathbf{x}^k)\|_2 \|\mathbf{d}^k\|_2 \cos \theta$.

显然当 $\cos \theta = -1$ 时， $\nabla f(\mathbf{x}^k)^T \mathbf{d}^k$ 取极小值.

此时： $\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$.

结论：沿负梯度方向 $-\nabla f(\mathbf{x})$ 下降最快，因此称为**最速下降方向**.

最速下降法（梯度下降法）

- 1) 初始点 $\mathbf{x}^0 \in \mathbb{R}^n$, $0 \leq \epsilon << 1$, $k = 0$
- 2) 计算 $\nabla f(\mathbf{x}^k)$, 如果 $\|\nabla f(\mathbf{x}^k)\|_2 \leq \epsilon$, 停止迭代;
否则, 令 $\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$.
- 3) 沿 \mathbf{d}^k 方向确定步长 α_k .
- 4) 令 $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$, $k = k + 1$, 转 Step 2).

精确一维搜索最速下降法

定义 3.5:

若最速下降法Step 3)中确定的步长 α_k 满足:

$$\alpha_k = \arg \min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k),$$

即 $f(\mathbf{x}^k + \alpha_k \mathbf{d}^k) = \min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k)$. 也即, 若 α_k 经精确一维搜索得到, 则相应的最速下降算法称为**精确一维(线)搜索最速下降法**, 且称 α_k 为精确步长因子.

精确一维搜索最速下降法

最速下降法特点：

d^k 与 d^{k+1} 正交，即相邻两次迭代中搜索方向是正交的。

证明：由于 $\alpha_k = \arg \min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k)$, 则 $\frac{df(\mathbf{x}^k + \alpha \mathbf{d}^k)}{d\alpha} |_{\alpha=\alpha_k} = 0$ 。

即 $(\mathbf{d}^k)^T \nabla f(\mathbf{x}^k + \alpha_k \mathbf{d}^k) = (\mathbf{d}^k)^T \nabla f(\mathbf{x}^{k+1}) = 0$ 。

又 $\mathbf{d}^{k+1} = -\nabla f(\mathbf{x}^{k+1})$, 故 $(\mathbf{d}^k)^T \mathbf{d}^{k+1} = 0$, 即正交。

精确一维搜索最速下降法

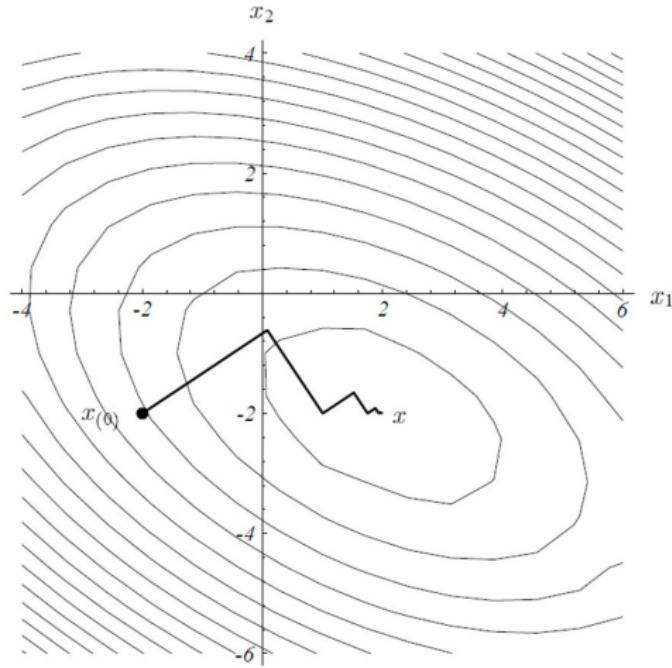
最速下降法特点：

最速下降法向极小点逼近是曲折前进的，这种现象称为锯齿现象。该现象会使得迭代点逼近极小点的过程是“之”字形。

而在实际优化过程中，对于任意初始点，最速下降法都可以很快到达极小点附近，但是越靠近极小点步长越小，导致最速下降法的收敛速度很慢。

精确一维搜索最速下降法

$$\mathbf{x}^0 = [-2, -2]^T, \mathbf{x}^* = [2, -2]^T$$



精确一维搜索最速下降法

例 3.7: 用**精确一维线搜索**最速下降法求解:

$$\min f(\boldsymbol{x}) = x_1^2 + 4x_2^2,$$

取 $\boldsymbol{x}^0 = (1, 1)^T$ 要求迭代两次, 并验证相邻两个搜索方向是否正交.

精确一维搜索最速下降法

例 3.7: 用精确一维线搜索最速下降法求解:

$$\min f(\mathbf{x}) = x_1^2 + 4x_2^2,$$

取 $\mathbf{x}^0 = (1, 1)^T$ 要求迭代两次, 并验证相邻两个搜索方向是否正交.

解: $\nabla f(\mathbf{x}) = \left(\frac{\partial f(\mathbf{x})}{\partial x_1}, \frac{\partial f(\mathbf{x})}{\partial x_2} \right)^T = (2x_1, 8x_2)^T,$

由 $\mathbf{x}^0 = (1, 1)^T$ 得: $\nabla f(\mathbf{x}^0) = (2, 8)^T.$

由于 $\|\nabla f(\mathbf{x}^0)\|_2 = 8.24621 > 0$, 因此需继续迭代.

(1) 取 $\mathbf{d}^0 = -\nabla f(\mathbf{x}^0) = (-2, -8)^T$

下面用精确线搜索求步长.

精确一维搜索最速下降法

因 $f(\mathbf{x}^0 + \alpha \mathbf{d}^0) = f\left(\begin{pmatrix} 1-2\alpha \\ 1-8\alpha \end{pmatrix}\right) = (1-2\alpha)^2 + 4(1-8\alpha)^2$.

令 $\frac{df(\mathbf{x}^0 + \alpha \mathbf{d}^0)}{d\alpha} = 0$ 得 $\alpha_0 = 0.13077$, 进而有:

$$\mathbf{x}^1 = \mathbf{x}^0 + \alpha_0 \mathbf{d}^0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - 0.13077 \begin{pmatrix} 2 \\ 8 \end{pmatrix} = \begin{pmatrix} 0.73846 \\ -0.04616 \end{pmatrix},$$

$$\nabla f(\mathbf{x}^1) = (1.47692, -0.36923)^T,$$

$$\|\nabla f(\mathbf{x}^1)\|_2 = 1.52237,$$

由于 $\|\nabla f(\mathbf{x}^1)\|_2$ 较大, 因此还需继续迭代.

精确一维搜索最速下降法

$$(2) \quad \mathbf{d}^1 = -\nabla f(\mathbf{x}^1) = (-1.47692, 0.36923)^T,$$

然后用精确线搜索求步长, 即令 $\frac{df(\mathbf{x}^1 + \alpha \mathbf{d}^1)}{d\alpha} = 0$ 得: $\alpha_1 = 0.42500$.
则有:

$$\mathbf{x}^2 = \mathbf{x}^1 + \alpha_1 \mathbf{d}^1 = \begin{pmatrix} 0.73846 \\ -0.04616 \end{pmatrix} + 0.42500 \begin{pmatrix} -1.47692 \\ 0.36923 \end{pmatrix} = \begin{pmatrix} 0.11076 \\ 0.11076 \end{pmatrix},$$

$$\nabla f(\mathbf{x}^2) = (0.22152, 0.88608)^T,$$

$$\|\nabla f(\mathbf{x}^2)\|_2 = 0.91335.$$

可见, 用最速下降法迭代两次并没有到达极小点.

精确一维搜索最速下降法

下面验证相邻两个搜索方向是否正交:

由(2)得:

$$\mathbf{d}^2 = -\nabla f(\mathbf{x}^2) = (-0.22152, -0.88608)^T,$$

且:

$$\mathbf{d}^1 = -\nabla f(\mathbf{x}^1) = (-1.47692, 0.36923)^T,$$

$$\mathbf{d}^0 = -\nabla f(\mathbf{x}^0) = (-2, -8)^T,$$

所以:

$$(\mathbf{d}^1)^T \mathbf{d}^0 = (-1.47692)(-2) + (0.36923)(-8) = 0,$$

$$(\mathbf{d}^2)^T \mathbf{d}^1 = (-0.22152)(-1.47692) + (-0.88608)(0.36923) = 0.$$

精确一维搜索最速下降法

例 3.8: 用**精确一维线搜索**最速下降法求解:

$$\min f(\boldsymbol{x}) = x_1^2 - 2x_1x_2 + 4x_2^2 + x_1 - 3x_2$$

取初始点 $\boldsymbol{x}^0 = (1, 1)^T$, 要求迭代2次, 出现的分数不化为小数, 并验证相邻两个搜索方向是否正交。

精确一维搜索最速下降法

例 3.8: 用精确一维线搜索最速下降法求解:

$$\min f(\mathbf{x}) = x_1^2 - 2x_1x_2 + 4x_2^2 + x_1 - 3x_2$$

取初始点 $\mathbf{x}^0 = (1, 1)^T$, 要求迭代2次, 出现的分数不化为小数, 并验证相邻两个搜索方向是否正交。

解: $\nabla f(\mathbf{x}) = \left(\frac{\partial f(\mathbf{x})}{\partial x_1}, \frac{\partial f(\mathbf{x})}{\partial x_2} \right)^T = (2x_1 - 2x_2 + 1, -2x_1 + 8x_2 - 3)^T$,

由 $\mathbf{x}^0 = (1, 1)^T$ 得: $\nabla f(\mathbf{x}^0) = (1, 3)^T$.

由于 $\|\nabla f(\mathbf{x}^0)\|_2 = 3.162 > 0$, 因此需继续迭代.

(1) 取 $\mathbf{d}^0 = -\nabla f(\mathbf{x}^0) = (-1, -3)^T$

下面用精确线搜索求步长.

精确一维搜索最速下降法

$$\begin{aligned}f(\mathbf{x}^0 + \alpha \mathbf{d}^0) &= f\left(\begin{pmatrix} 1 - \alpha \\ 1 - 3\alpha \end{pmatrix}\right) \\&= (1 - \alpha)^2 - 2(1 - \alpha)(1 - 3\alpha) + 4(1 - 3\alpha)^2 + (1 - \alpha) - 3(1 - 3\alpha)\end{aligned}$$

令 $\frac{df(\mathbf{x}^0 + \alpha \mathbf{d}^0)}{d\alpha} = 0$ 得 $\alpha_0 = 5/31$, 进而有:

$$\mathbf{x}^1 = \mathbf{x}^0 + \alpha_0 \mathbf{d}^0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} + \frac{5}{31} \begin{pmatrix} -1 \\ -3 \end{pmatrix} = \begin{pmatrix} 26/31 \\ 16/31 \end{pmatrix},$$

精确一维搜索最速下降法

$$(2) \quad \mathbf{d}^1 = -\nabla f(\mathbf{x}^1) = (-51/31, 17/31)^T,$$

然后用精确线搜索求步长, 即令 $\frac{df(\mathbf{x}^1 + \alpha \mathbf{d}^1)}{d\alpha} = 0$ 得: $\alpha_1 = 5/19$.
则有:

$$\mathbf{x}^2 = \mathbf{x}^1 + \alpha_1 \mathbf{d}^1 = \begin{pmatrix} 26/31 \\ 16/31 \end{pmatrix} + \frac{5}{19} \begin{pmatrix} -51/31 \\ 17/31 \end{pmatrix} = \begin{pmatrix} 239/589 \\ 389/589 \end{pmatrix},$$

精确一维搜索最速下降法

定理 3.3: 收敛性定理 设 $f(\mathbf{x})$ 连续可微, 且水平集有界

$$L = \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) \leq f(\mathbf{x}^0)\}$$

则对于某个 k , 由精确一维搜索最速下降法产生的序列满足:

$$\|\nabla f(\mathbf{x}^k)\|_2 = 0,$$

或

$$\lim_{k \rightarrow \infty} \|\nabla f(\mathbf{x}^k)\|_2 = 0.$$

精确一维搜索最速下降法

优点:

- 计算复杂度低, 存储量小;
- 具有整体收敛性, 对初始点没有特别要求.

缺点:

- 收敛速度慢, 尤其在极小点附近, 该方法产生的点列逼近函数的极小点越来越慢.

导致缺点的原因:

- $d^k = -\nabla f(\mathbf{x}^k)$ 仅反映 $f(\mathbf{x})$ 在 \mathbf{x}^k 处的局部性质.
- d^k 与 d^{k+1} 正交, 相邻两次迭代的搜索方向正交, 致使优化“远快近慢”.

精确一维搜索最速下降法

为了解决最速下降法中相邻两步搜索方向正交的所导致的收敛速度慢的问题，可使用以下改进方法：

- 1) **选择不同初始点**：有利于避免迭代过程中出现锯齿现象。
- 2) **采用不精确的一维搜索**：可使相邻两个迭代点处的梯度不正交，从而改变收敛性。对于最速下降法，有时为了减少计算工作量，采用固定步长，称为固定步长最速下降法。
- 3) **采用加速梯度法**：负梯度方向和 $p^k = \mathbf{x}^k - \mathbf{x}^{k-2}$ 相结合。即，由于最速下降法在极小点附近成“锯齿”状，因此第 k 步的搜索方向取为 p^k ，而第 $k+1$ 和第 $k+2$ 步继续使用负梯度方向。

目录

① 迭代法概述

② 一维搜索方法(线搜索方法)

③ 最速下降法(梯度下降法)

④ 共轭梯度法

⑤ 牛顿法(牛顿法和阻尼牛顿法)

⑥ 拟牛顿法(变尺度法)

⑦ 信赖域法

共轭梯度法

背景和基本思想：

利用目标函数在当前迭代点 x^k 处的负梯度方向 $-\nabla f(x^k)$ 与上一步的搜索方向 d^{k-1} 的适当线性组合，作为 x^k 处的搜索方向 d^k 。

由Taylor公式知，函数在某点附近的性质与二次函数是很接近的。因此，往往先设计算法求解二次模型，即先针对正定二次函数建立有效的算法，然后再推广到一般函数上去。

共轭梯度法

一、共轭方向及其性质

定义 3.6:

设 d_1, d_2, \dots, d_m 是 \mathbb{R}^n 中任意一组非零向量, $Q \in \mathbb{R}^{n \times n}$ 是 n 阶实正定矩阵。若

$$d_i^T Q d_j = 0, \quad (i \neq j)$$

则称 d_1, d_2, \dots, d_m 是关于 Q 共轭的。

【注】 若 $Q = I$, $d_i^T Id_j = 0, \quad (i \neq j)$, 则 d_1, d_2, \dots, d_m 是正交的, 因此共轭是正交的推广、共轭向量组是正交向量组的推广。

共轭梯度法

定理 3.4:

设 Q 为 n 阶正定矩阵, 非零向量组 d_1, d_2, \dots, d_m 关于 Q 共轭, 则必线性无关.

推论 3.1:

设 Q 为 n 阶正定矩阵, 非零向量组 d_1, d_2, \dots, d_n 关于 Q 共轭, 则它们构成 \mathbb{R}^n 的一组基.

推论 3.2:

设 Q 为 n 阶正定矩阵, 非零向量组 d_1, d_2, \dots, d_n 关于 Q 共轭. 若向量 v 与 d_1, d_2, \dots, d_n 均正交, 则 $v = 0$.

共轭梯度法

定理 3.5:

设 Q 为 n 阶正定阵, 向量组 $\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_n$ 关于 Q 共轭, 对正定二次函数

$$\min f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T Q \mathbf{x} + \mathbf{b}^T \mathbf{x} + c,$$

由任意 \mathbf{x}^1 开始, 依次沿 \mathbf{d}_i 进行 **精确线搜索**, 则至多经过 n 次迭代即可求得 $f(\mathbf{x})$ 的极小值点.

共轭梯度法

思考：对于下列正定二次函数求极小问题：

$$\min f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c,$$

如果能够选定这样的搜索方向，那么对于二元二次函数只需依次沿搜索方向 $\mathbf{d}^1, \mathbf{d}^2$ 各进行1次精确一维搜索就可以求到极小点 \mathbf{x}^* ，即

$$\mathbf{x}^* = \mathbf{x}^0 + \alpha_2 \mathbf{d}^2$$

注意到 \mathbf{x}^* 是 $f(\mathbf{x})$ 的极小点，故 \mathbf{x}^* 是 $f(\mathbf{x})$ 的驻点：

$$\nabla f(\mathbf{x}^*) = \mathbf{Q} \mathbf{x}^* + \mathbf{b} = \mathbf{0}.$$

共轭梯度法

也即：

$$0 = \nabla f(\mathbf{x}^*) = Q\mathbf{x}^2 + \mathbf{b} + \alpha_2 Q\mathbf{d}^2 = \nabla f(\mathbf{x}^2) + \alpha_2 Q\mathbf{d}^2.$$

等式两边同时左乘 $(\mathbf{d}^1)^T$ 可得：

$$0 = (\mathbf{d}^1)^T \nabla f(\mathbf{x}^2) + \alpha_2 (\mathbf{d}^1)^T Q\mathbf{d}^2 = 0 + \alpha_2 (\mathbf{d}^1)^T Q\mathbf{d}^2 \Leftrightarrow (\mathbf{d}^1)^T Q\mathbf{d}^2 = 0.$$

这说明，要想两次迭代要得到二元二次函数的极小点，则迭代算法必须满足的条件是：搜索方向 \mathbf{d}^1 和 \mathbf{d}^2 关于 Q 共轭，其中 \mathbf{d}^1 是某个搜索方向。

共轭梯度法

启发：

用迭代法求解正定二次函数极小的问题：

$$\min f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c$$

可转化为构造或寻找关于 \mathbf{Q} 共轭的 n 个方向问题.

问题： 如何构造 n 个关于 \mathbf{Q} 共轭的方向？

共轭梯度法

启发：

用迭代法求解正定二次函数极小的问题：

$$\min f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c$$

可转化为构造或寻找关于 \mathbf{Q} 共轭的 n 个方向问题.

问题： 如何构造 n 个关于 \mathbf{Q} 共轭的方向？

一种做法是结合当前迭代点处的负梯度方向以及上一个迭代点处的下降方向来构造共轭梯度法.

共轭梯度法

二、求解正定二次函数的共轭梯度法

1. 基于待定系数法构造共轭方向

取: $\boxed{\mathbf{d}^1 = -\nabla f(\mathbf{x}^1)}$ 令 $k = 1$.

由精确线搜索得步长 α_k , 进而得: $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$,
令 $\mathbf{d}^{k+1} = -\nabla f(\mathbf{x}^{k+1}) + \lambda_k \mathbf{d}^k$, λ_k 为待定组合系数.

左乘 $(\mathbf{d}^k)^T Q$, 并使 $(\mathbf{d}^k)^T Q \mathbf{d}^{k+1} = 0$, 得:

$$\lambda_k = \frac{(\mathbf{d}^k)^T Q \nabla f(\mathbf{x}^{k+1})}{(\mathbf{d}^k)^T Q \mathbf{d}^k}$$

共轭梯度法

对于上述方式产生的下降方向 d^k , 可以证明:

$$(\mathbf{d}^i)^T \mathbf{Q} \mathbf{d}^{k+1} = 0, i = 1, 2, \dots, n.$$

也即, 若中途不停机的话, 按上述方法得到的 n 个方向 $\mathbf{d}^1, \mathbf{d}^2, \dots, \mathbf{d}^n$ 关于 \mathbf{Q} 是两两共轭的.

此外, 上述共轭梯度法迭代过程中, 序列 $\{\mathbf{x}^k\}$ 所对应的梯度向量组 $\nabla f(\mathbf{x}^k), k = 1, 2, \dots, n$ 是正交向量组.

共轭梯度法

2. 适用于求解正定二次函数的共轭梯度法算法

- 1) 给出 $\mathbf{x}^1 \in \mathbb{R}^n, \mathbf{d}^1 = -\nabla f(\mathbf{x}^1), 0 \leq \epsilon < 1, k = 1$
- 2) 如果 $\|\nabla f(\mathbf{x}^k)\|_2 < \epsilon$, 停止迭代;

否则沿 \mathbf{d}^k 精确线搜索求步长:

$$\alpha_k = \arg \min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k) = -\frac{(\mathbf{d}^k)^T \nabla f(\mathbf{x}^k)}{(\mathbf{d}^k)^T \mathbf{Q} \mathbf{d}^k}.$$

令 $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$, 并计算: $\mathbf{d}^{k+1} = -\nabla f(\mathbf{x}^{k+1}) + \lambda_k \mathbf{d}^k$,

其中 $\lambda_k = \frac{(\mathbf{d}^k)^T \mathbf{Q} \nabla f(\mathbf{x}^{k+1})}{(\mathbf{d}^k)^T \mathbf{Q} \mathbf{d}^k}$.

- 3) 令 $k = k + 1$, 转 Step 2).

共轭梯度法

针对正定二次函数迭代过程中 α_k 的推导过程：

不失一般性，设 $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q\mathbf{x} + \mathbf{b}^T \mathbf{x} + c$ ，其中 Q 为对称正定矩阵，且 $\nabla f(\mathbf{x}) = Q\mathbf{x} + \mathbf{b}$ 。给定任意迭代点 \mathbf{x}^k ，则最佳步长 α_k 满足

$$\alpha_k = \arg \min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k),$$

即

$$\begin{aligned} 0 &= f'(\mathbf{x}^k + \alpha \mathbf{d}^k) = \nabla f(\mathbf{x}^k + \alpha \mathbf{d}^k)^T \mathbf{d}^k \\ &= (Q(\mathbf{x}^k + \alpha \mathbf{d}^k) + \mathbf{b})^T \mathbf{d}^k = (\nabla f(\mathbf{x}^k) + \alpha Q \mathbf{d}^k)^T \mathbf{d}^k \end{aligned}$$

易知 $\alpha_k = -\frac{(\mathbf{d}^k)^T \nabla f(\mathbf{x}^k)}{(\mathbf{d}^k)^T Q \mathbf{d}^k}$ 。

共轭梯度法

3. 收敛性

综上所述, 若中途不停机的话, 这样得到的 n 个方向 $\mathbf{d}^1, \mathbf{d}^2, \dots, \mathbf{d}^n$ 关于 Q 是两两共轭的. 再结合定理3.5, 那么 \mathbf{x}^{n+1} 一定是所求的最优解.

组合系数的选取为如下Hestenes-Stiefel公式(HS):

$$\lambda_k = \frac{(\mathbf{d}^k)^T Q \nabla f(\mathbf{x}^{k+1})}{(\mathbf{d}^k)^T Q \mathbf{d}^k}.$$

可见, 该公式中里面含有Hessian矩阵 Q , 不能直接应用于一般非线性函数求极值情形.

共轭梯度法

4. 组合系数的其他形式—正定二次函数

1. Fletcher-Reeve公式(FR)

$$\lambda_k = \frac{\|\nabla f(\mathbf{x}^{k+1})\|_2^2}{\|\nabla f(\mathbf{x}^k)\|_2^2}$$

2. Dixon-Myers公式(DM)

$$\lambda_k = -\frac{\|\nabla f(\mathbf{x}^{k+1})\|_2^2}{(\mathbf{d}^k)^T \nabla f(\mathbf{x}^k)}$$

3. Polak-Ribiere-Polyak公式(PRP)

$$\lambda_k = \frac{\nabla f(\mathbf{x}^{k+1})^T (\nabla f(\mathbf{x}^{k+1}) - \nabla f(\mathbf{x}^k))}{\|\nabla f(\mathbf{x}^k)\|_2^2}$$

共轭梯度法

三、求解一般函数极值的FR共轭梯度法

- 1) 给定 $\mathbf{x}^0 \in \mathbb{R}^n$, $\mathbf{d}^0 = -\nabla f(\mathbf{x}^0)$, $0 \leq \epsilon < 1$, $k = 0$.
- 2) 如果 $\|\nabla f(\mathbf{x}^k)\|_2 < \epsilon$, 停止迭代, 取 $\mathbf{x}^* = \mathbf{x}^k$; 否则, 由精确线搜索求步长:
$$\alpha_k = \arg \min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k).$$
进而得 $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$, 以及 $\mathbf{d}^{k+1} = -\nabla f(\mathbf{x}^{k+1}) + \lambda_k \mathbf{d}^k$.
- 3) 令 $k = k + 1$, 返回 Step 2).

【注】 对于正定二次函数, HS、FR、DM、PRP等四个公式是等价的; 对应的四种共轭梯度法也是等价的.对于非二次函数, 产生的搜索方向不再相同, 常利用FR、DM、PRP等三个公式, 通常不用HS公式(其中含有Hessian矩阵).经验上PRP效果较好.

共轭梯度法

例 3.9: 用共轭梯度法求解:

$$\min f(\mathbf{x}) = x_1^2 + 4x_2^2$$

取 $\mathbf{x}^0 = (1, 1)^T$, 用Fletcher-Reeve公式(FR)计算 λ_k .

共轭梯度法

例 3.9: 用共轭梯度法求解:

$$\min f(\mathbf{x}) = x_1^2 + 4x_2^2$$

取 $\mathbf{x}^0 = (1, 1)^T$, 用 Fletcher-Reeve 公式(FR)计算 λ_k .

分析: 观察知该函数的极小点是 $\mathbf{x}^* = (0, 0)^T$.

又由 $f(\mathbf{x}) = \frac{1}{2}(x_1, x_2) \begin{pmatrix} 2 & 0 \\ 0 & 8 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ 可看出 $f(\mathbf{x})$ 是一个正定二次函数, 其中 $Q = \begin{pmatrix} 2 & 0 \\ 0 & 8 \end{pmatrix}$. 因此在求最优步长和组合系数时, 有多种方法.

共轭梯度法

解：因共轭梯度法的第一步迭代与最速下降法相同，故由上一节例3.4知

(1) 由 $\mathbf{x}^0 = (1, 1)^T$ 得 $\mathbf{d}^0 = -\nabla f(\mathbf{x}^0) = (-2, -8)^T$.

令 $\frac{df(\mathbf{x}^0 + \alpha \mathbf{d}^0)}{d\alpha} = 0$ 得: $\alpha_0 = 0.13077$, 进而有:

$$\mathbf{x}^1 = \mathbf{x}^0 + \alpha_0 \mathbf{d}^0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - 0.13077 \begin{pmatrix} 2 \\ 8 \end{pmatrix} = \begin{pmatrix} 0.73846 \\ -0.04616 \end{pmatrix}.$$

$$\nabla f(\mathbf{x}^1) = (1.47692, -0.36923)^T, \|\nabla f(\mathbf{x}^1)\|_2 = 1.52237.$$

由于 $\|\nabla f(\mathbf{x}^1)\|_2$ 较大, 因此还需继续迭代.

共轭梯度法

(2) 先构造下一个搜索方向 $\mathbf{d}^1 = -\nabla f(\mathbf{x}^1) + \lambda_0 \mathbf{d}^0$, 其中组合系数

$$\lambda_0 = \frac{\|\nabla f(\mathbf{x}^1)\|_2^2}{\|\nabla f(\mathbf{x}^0)\|_2^2} = \dots = 0.03408$$

或

$$\lambda_0 = \frac{(\mathbf{d}^0)^T \mathbf{Q} \nabla f(\mathbf{x}^1)}{(\mathbf{d}^0)^T \mathbf{Q} \mathbf{d}^0} = \dots = \frac{17.72304}{520} = 0.03408$$

于是有 $\mathbf{d}^1 = -\nabla f(\mathbf{x}^1) + \lambda_0 \mathbf{d}^0$

$$= \begin{pmatrix} -1.47692 \\ 0.36923 \end{pmatrix} + 0.03408 \begin{pmatrix} -2 \\ -8 \end{pmatrix} = \begin{pmatrix} -1.54508 \\ 0.09659 \end{pmatrix}$$

共轭梯度法

沿搜索方向 \mathbf{d}^1 用精确线搜索求步长:

为此, 先计算出表达式 $f(\mathbf{x}^1 + \alpha \mathbf{d}^1)$, 并且令 $\frac{df(\mathbf{x}^1 + \alpha \mathbf{d}^1)}{d\alpha} = 0$ 得:

$$\alpha_1 = 0.47794.$$

于是有

$$\mathbf{x}^2 = \mathbf{x}^1 + \alpha_1 \mathbf{d}^1 = \begin{pmatrix} 0.73846 \\ -0.04616 \end{pmatrix} + 0.47794 \begin{pmatrix} -1.54508 \\ 0.09659 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

因 $\|\nabla f(\mathbf{x}^2)\|_2 = 0$, 所以停止迭代. 最优解为 $\mathbf{x}^* = \mathbf{x}^2 = (0, 0)^T$.

可见: 对于例3.9, 用共轭梯度法经两次迭代求得最优解.

共轭梯度法

定理 3.6: 收敛性定理 设凸函数 $f(\mathbf{x})$ 存在一阶连续偏导数, 且水平集有界

$$L = \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) \leq f(\mathbf{x}^0)\}$$

则由共轭梯度法得到的点列 $\{\mathbf{x}^k\}$ 有如下性质:

- 1) $f(\mathbf{x}^k)$ 严格单调下降, 且 $\lim_{k \rightarrow +\infty} f(\mathbf{x}^k)$ 存在.
- 2) $\{\mathbf{x}^k\}$ 的任意聚点 \mathbf{x}^* 都是 $f(\mathbf{x})$ 的极小点.

共轭梯度法

优点：

- 克服了最速下降法收敛速度慢的缺点, 共轭梯度法的收敛速率不坏于最速下降法;
- 建立在二次模型之上, 具有二次终止性, 即求解 n 元正定二次函数(严格凸二次函数)极值, 最多 n 步迭代便可求出最优解;
- 不用求Hessian矩阵或者逆矩阵, 算法所需存储空间小, 是求解大规模问题的重要方法;
- 对凸函数全局收敛.

共轭梯度法

缺点:

- 误差可能会使 n 步迭代得不到正定二次函数的极小点. \mathbb{R}^n 中共轭方向最多有 n 个, n 步后构造的搜索方向不再是共轭的, 会降低收敛速度;
- 对非二次函数, 由于目标函数的Hessian矩阵不再是常数矩阵, 因而产生的方向不再是共轭方向.

共轭梯度法

四、周期性的共轭梯度法

对于FR算法和PRP算法, 如果初始方向不取负梯度方向或不使用精确线搜索求最优步长, 即使对于二次函数也很难产生 n 个共轭方向.

重新开始技术: 因此用这两个方法时, 通常当迭代 n 步后, 重新设定搜索方向为当前点的负梯度方向, 然后以该方向作为初始方向, 用共轭梯度法产生 n 个新点列. 依次重复下去, 直至收敛.

共轭梯度法

周期性FR共轭梯度法算法

- 1) $\mathbf{x}^0 \in \mathbb{R}^n, 0 \leq \epsilon \ll, k = 0$
- 2) 计算 $\nabla f(\mathbf{x}^k)$, 如果 $\|\nabla f(\mathbf{x}^k)\|_2 < \epsilon$, 停止迭代; 否则转Step 3).
- 3) 若 k 是 n 的倍数, 则令 $\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$;
否则令 $\mathbf{d}^k = -\nabla f(\mathbf{x}^k) + \frac{\|\nabla f(\mathbf{x}^k)\|_2^2}{\|\nabla f(\mathbf{x}^{k-1})\|_2^2} \mathbf{d}^{k-1}$.
- 4) 由精确线搜索求 α_k , 令 $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k, k = k + 1$, 返回Step 2).

目录

- ① 迭代法概述
- ② 一维搜索方法(线搜索方法)
- ③ 最速下降法(梯度下降法)
- ④ 共轭梯度法
- ⑤ 牛顿法(牛顿法和阻尼牛顿法)
- ⑥ 拟牛顿法(变尺度法)
- ⑦ 信赖域法

牛顿法

一、一维牛顿法推广

精确一维搜索中介绍了牛顿法，即用目标函数的二阶泰勒展开式近似代替函数本身，用二次泰勒展开式的极小点近似函数的极小点：

$$x^{k+1} = x^k - \frac{f'(x^k)}{f''(x^k)} = x^k - [f''(x^k)]^{-1} f'(x^k)$$

将它推广到多元函数情况/多维的情形，即得到求解多元函数极小的牛顿迭代算法：

$$\mathbf{x}^{k+1} = \mathbf{x}^k - [\nabla^2 f(\mathbf{x}^k)]^{-1} \nabla f(\mathbf{x}^k)$$

【注】 此时 $\alpha_k = 1$, $\mathbf{d}^k = -[\nabla^2 f(\mathbf{x}^k)]^{-1} \nabla f(\mathbf{x}^k)$ 称为牛顿方向。

牛顿法

牛顿法的理论推导

设 $f(\mathbf{x})$ 是二阶连续可微的, 由略去高阶项的Taylor公式

$$f(\mathbf{x}) \approx f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T (\mathbf{x} - \mathbf{x}^k) + \frac{1}{2} (\mathbf{x} - \mathbf{x}^k)^T \nabla^2 f(\mathbf{x}^k) (\mathbf{x} - \mathbf{x}^k)$$

对右端二次近似函数求导, 且令:

$$\nabla f(\mathbf{x}^k) + \nabla^2 f(\mathbf{x}^k) (\mathbf{x} - \mathbf{x}^k) = \mathbf{0}$$

若 $\nabla^2 f(\mathbf{x}^k)$ 可逆, 则有

$$\mathbf{x} = \mathbf{x}^k - [\nabla^2 f(\mathbf{x}^k)]^{-1} \nabla f(\mathbf{x}^k)$$

若它不是 $f(\mathbf{x})$ 的极值点, 则令它作为 \mathbf{x}^{k+1} , 即

$$\mathbf{x}^{k+1} = \mathbf{x}^k - [\nabla^2 f(\mathbf{x}^k)]^{-1} \nabla f(\mathbf{x}^k)$$

牛顿法

利用目标函数 $f(x)$ 在 x^k 处的二阶Taylor多项式去近似目标函数, 用近似函数的极小点作为下一个迭代点, 如此下去使迭代点列逐渐逼近目标函数的极小点.

对于Hessian矩阵正定的二次函数, 用牛顿法从任意点开始迭代, 只需迭代一次就可以得到其极小点.

定义 3.7: 二次收敛性(二次终止性)

从任意初始点出发, 经有限次迭代总可以达到正定二次函数的极小点, 称这样的算法具有二次收敛性.

牛顿法

定理 3.7:

对于Hessian矩阵正定的二次函数，因此用牛顿法从任意初始点出发，只需迭代一次就可以得到极小点。

证明：不失一般性，设 $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q \mathbf{x} + \mathbf{b}^T \mathbf{x} + c$ ，其中 Q 为对称正定矩阵，则 $f(\mathbf{x})$ 的极小值点为 $\mathbf{x}^* = -Q^{-1}\mathbf{b}$ 。给定初始点 \mathbf{x}^0 ，则：
 $\nabla f(\mathbf{x}^0) = Q\mathbf{x}^0 + \mathbf{b}$ 且 $\nabla^2 f(\mathbf{x}^0) = Q$ ，故

$$\begin{aligned}\mathbf{x}^1 &= \mathbf{x}^0 - [\nabla^2 f(\mathbf{x}^0)]^{-1} \nabla f(\mathbf{x}^0) \\&= \mathbf{x}^0 - Q^{-1}(Q\mathbf{x}^0 + \mathbf{b}) \\&= -Q^{-1}\mathbf{b} = \mathbf{x}^*\end{aligned}$$

牛顿法

牛顿法算法

- 1) $\mathbf{x}^0 \in \mathbb{R}^n, 0 \leq \epsilon << 1, k = 0$
- 2) 计算 $\nabla f(\mathbf{x}^k)$, 如果 $\|\nabla f(\mathbf{x}^k)\| \leq \epsilon$, 停止迭代;
否则计算 $\nabla^2 f(\mathbf{x}^k)$, 并令

$$\mathbf{d}^k = -[\nabla^2 f(\mathbf{x}^k)]^{-1} \nabla f(\mathbf{x}^k)$$

- 3) 令 $\mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{d}^k, k = k + 1$, 返回 Step 2).

【注】 如何手动计算 $A = [\nabla^2 f(\mathbf{x}^k)]^{-1}$?

对 $(A|I)$ 进行初等行变换(左乘些初等矩阵)。当 $(A|I)$ 中 A 变换为 I 时, 则原 I 变换为 A^{-1} , 即 $(I|A^{-1})$ 。

牛顿法

例 3.10: 设 $f(\mathbf{x}) = (6 + x_1 + x_2)^2 + (2 - 3x_1 - 3x_2 - x_1x_2)^2$, 求该函数在点 $\mathbf{x}^1 = (-4, 6)^T$ 处的牛顿方向.

牛顿法

例 3.10: 设 $f(\mathbf{x}) = (6 + x_1 + x_2)^2 + (2 - 3x_1 - 3x_2 - x_1x_2)^2$, 求该函数在点 $\mathbf{x}^1 = (-4, 6)^T$ 处的牛顿方向.

分析: 点 \mathbf{x} 处的牛顿方向为 $\mathbf{d} = -[\nabla^2 f(\mathbf{x})]^{-1} \nabla f(\mathbf{x})$, 故需要计算出 $\nabla f(\mathbf{x})$ 和 $\nabla^2 f(\mathbf{x})$ 的表达式.

解:

$$\frac{\partial f(\mathbf{x})}{\partial x_1} = 2(6 + x_1 + x_2) + 2(2 - 3x_1 - 3x_2 - x_1x_2)(-3 - x_2)$$

$$\frac{\partial f(\mathbf{x})}{\partial x_2} = 2(6 + x_1 + x_2) + 2(2 - 3x_1 - 3x_2 - x_1x_2)(-3 - x_1)$$

故 $\left. \frac{\partial f(\mathbf{x})}{\partial x_1} \right|_{\mathbf{x}=(-4,6)^T} = -344$, $\left. \frac{\partial f(\mathbf{x})}{\partial x_2} \right|_{\mathbf{x}=(-4,6)^T} = 56$, $\nabla f(\mathbf{x}^1) = \begin{pmatrix} -344 \\ 56 \end{pmatrix}$

牛顿法

$$\frac{\partial^2 f(\mathbf{x})}{\partial x_1^2} = 2 + 2(3 + x_2)^2, \quad \frac{\partial^2 f(\mathbf{x})}{\partial x_2^2} = 2 + 2(3 + x_1)^2$$

$$\frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_1} = \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_2} = 2 + 2(3 + x_1)(3 + x_2) + 2(-2 + 3x_1 + 3x_2 + x_1 x_2)$$

故

$$\left. \frac{\partial^2 f(\mathbf{x})}{\partial x_1^2} \right|_{\mathbf{x}=(-4,6)^T} = 164, \quad \left. \frac{\partial^2 f(\mathbf{x})}{\partial x_2^2} \right|_{\mathbf{x}=(-4,6)^T} = 4$$

$$\left. \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_1} \right|_{\mathbf{x}=(-4,6)^T} = -56, \quad \left. \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_2} \right|_{\mathbf{x}=(-4,6)^T} = -56$$

牛顿法

所以

$$\nabla^2 f(\mathbf{x}^1) = \begin{pmatrix} 164 & -56 \\ -56 & 4 \end{pmatrix}$$

进而有

$$[\nabla^2 f(\mathbf{x}^1)]^{-1} = -\frac{1}{620} \begin{pmatrix} 1 & 14 \\ 14 & 41 \end{pmatrix}$$

因此, 所求的牛顿方向为

$$\begin{aligned} \mathbf{d}^1 &= -[\nabla^2 f(\mathbf{x}^1)]^{-1} \nabla f(\mathbf{x}^1) \\ &= \frac{1}{620} \begin{pmatrix} 1 & 14 \\ 14 & 41 \end{pmatrix} \begin{pmatrix} -344 \\ 56 \end{pmatrix} = \frac{1}{155} \begin{pmatrix} 110 \\ -630 \end{pmatrix} = \frac{2}{31} \begin{pmatrix} 11 \\ -63 \end{pmatrix} \end{aligned}$$

牛顿法

例 3.11:用牛顿法求解:

$$\min f(\mathbf{x}) = x_1^2 + 4x_2^2, \quad \text{取初始点为 } \mathbf{x}^0 = (1, 1)^T.$$

牛顿法

例 3.11: 用牛顿法求解:

$$\min f(\mathbf{x}) = x_1^2 + 4x_2^2, \quad \text{取初始点为 } \mathbf{x}^0 = (1, 1)^T.$$

解:

$$\frac{\partial f(\mathbf{x})}{\partial x_1} = 2x_1, \quad \frac{\partial f(\mathbf{x})}{\partial x_2} = 8x_2,$$

$$\frac{\partial^2 f(\mathbf{x})}{\partial x_1^2} = 2, \quad \frac{\partial^2 f(\mathbf{x})}{\partial x_2^2} = 8, \quad \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_1} = \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_2} = 0$$

所以 $\nabla f(\mathbf{x}) = \begin{pmatrix} 2x_1 \\ 8x_2 \end{pmatrix}, \quad \nabla^2 f(\mathbf{x}) = \begin{pmatrix} 2 & 0 \\ 0 & 8 \end{pmatrix}$

代入得 $\nabla f(\mathbf{x}^0) = \begin{pmatrix} 2 \\ 8 \end{pmatrix}, \quad \nabla^2 f(\mathbf{x}^0) = \begin{pmatrix} 2 & 0 \\ 0 & 8 \end{pmatrix}$

牛顿法

所以 $\mathbf{x}^1 = \mathbf{x}^0 - [\nabla^2 f(\mathbf{x}^0)]^{-1} \nabla f(\mathbf{x}^0)$

$$= \begin{pmatrix} 1 \\ 1 \end{pmatrix} - \begin{pmatrix} 2 & 0 \\ 0 & 8 \end{pmatrix}^{-1} \begin{pmatrix} 2 \\ 8 \end{pmatrix}$$

$$= \begin{pmatrix} 1 \\ 1 \end{pmatrix} - \begin{pmatrix} 1/2 & 0 \\ 0 & 1/8 \end{pmatrix} \begin{pmatrix} 2 \\ 8 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

而 $\nabla f(\mathbf{x}) = \begin{pmatrix} 2x_1 \\ 8x_2 \end{pmatrix}$, 故 $\|\nabla f(\mathbf{x}^1)\|_2 = 0$ 。

所以迭代终止, 最优解为: $\mathbf{x}^* = \mathbf{x}^1 = (0, 0)^T$ 。

可见: 用牛顿法求解经一次迭代即可求得极小点。

牛顿法

优点

- 对正定二次函数, 用牛顿法从任意初始点迭代一次即可得到极小点;
- 如果最优解 \mathbf{x}^* 处 $\nabla^2 f(\mathbf{x}^*)$ 正定, 且初始点选取合适, 算法很快收敛.

缺点

- 收敛性与初始点的选取依赖很大, 即不具备全局收敛性. 有可能收敛到鞍点或极大值点;
- 要求函数至少二阶连续可微;
- 每次迭代需要计算Hessian矩阵 $\nabla^2 f(\mathbf{x}^k)$, 计算复杂度高;
- 每次迭代需要求解方程组 $\nabla^2 f(\mathbf{x}^k) \mathbf{d}^k = -\nabla f(\mathbf{x}^k)$, 当方程组为奇异或病态时, 无法求得 \mathbf{d}^k 或 \mathbf{d}^k 不是下降方向.

阻尼牛顿法

基本思想

针对牛顿法的第一个缺点，在求新迭代点时，以 \mathbf{d}^k 作为搜索方向进行一维搜索求步长 α_k ，而非将步长固定为1，也即使

$$\alpha_k = \arg \min_{\alpha > 0} f(\mathbf{x}^k + \alpha \mathbf{d}^k).$$

也即利用精确一维搜索为牛顿迭代公式搜索得最优步长 α_k ，然后令

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k = \mathbf{x}^k - \alpha_k [\nabla^2 f(\mathbf{x}^k)]^{-1} \nabla f(\mathbf{x}^k).$$

【注】 每次迭代目标函数值一定有所下降.

阻尼牛顿法

定理 3.8:

对于Hessian矩阵正定的二次函数, 用阻尼牛顿法从任意初始点出发, 只需迭代一次就可以得到极小点.

证明: 不失一般性, 设 $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q\mathbf{x} + \mathbf{b}^T \mathbf{x} + c$, 其中 Q 为对称正定矩阵, 则 $f(\mathbf{x})$ 的极小值点为 $\mathbf{x}^* = -Q^{-1}\mathbf{b}$. 给定初始点 \mathbf{x}^0 , 则:

$\nabla f(\mathbf{x}^0) = Q\mathbf{x}^0 + \mathbf{b}$ 且 $\nabla^2 f(\mathbf{x}^0) = Q$, 因此 $\mathbf{d}^0 = -Q^{-1}\nabla f(\mathbf{x}^0)$.

另一方面, 最佳步长满足 $\alpha_0 = \arg \min_{\alpha > 0} f(\mathbf{x}^0 + \alpha \mathbf{d}^0)$, 即

$$\begin{aligned} 0 &= f'(\mathbf{x}^0 + \alpha_0 \mathbf{d}^0) = \nabla f(\mathbf{x}^0 + \alpha_0 \mathbf{d}^0)^T \mathbf{d}^0 \\ &= (Q(\mathbf{x}^0 + \alpha_0 \mathbf{d}^0) + \mathbf{b})^T \mathbf{d}^0 = (\nabla f(\mathbf{x}^0) + \alpha_0 Q \mathbf{d}^0)^T \mathbf{d}^0 \end{aligned}$$

易知, $\alpha_0 = -\frac{\nabla f(\mathbf{x}^0)^T \mathbf{d}^0}{(\mathbf{d}^0)^T Q \mathbf{d}^0} = -\frac{\nabla f(\mathbf{x}^0)^T \mathbf{d}^0}{-(\mathbf{d}^0)^T Q Q^{-1} \nabla f(\mathbf{x}^0)} = 1$. 带入牛顿公式则得证.

阻尼牛顿法

阻尼牛顿法算法

- 1) $\mathbf{x}^0 \in \mathbb{R}^n, 0 \leq \epsilon << 1, k = 0$
- 2) 计算 $\nabla f(\mathbf{x}^k)$, 如果 $\|\nabla f(\mathbf{x}^k)\| \leq \epsilon$, 停止迭代; 否则计算 $\nabla^2 f(\mathbf{x}^k)$, 并令

$$\mathbf{d}^k = -\left[\nabla^2 f(\mathbf{x}^k)\right]^{-1} \nabla f(\mathbf{x}^k)$$

- 3) 沿 d^k 进行线搜索, 得最优步长 α_k .
- 4) 令 $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k, k = k + 1$, 转 Step 2).

阻尼牛顿法

定理 3.9：收敛性定理

设 $f(\mathbf{x})$ 二阶连续可微, $\nabla^2 f(\mathbf{x})$ 正定. 记 $\{\mathbf{x}^k\}$ 是由阻尼牛顿法所得迭代点列, 若水平集 $L = \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) \leq f(\mathbf{x}^0)\}$ 有界, 则:

- (1) $\{f(\mathbf{x}^k)\}$ 为严格单调下降数列;
- (2) $\{\mathbf{x}^k\}$ 必有聚点, 且任何聚点 \mathbf{x}^* 满足 $\nabla f(\mathbf{x}^*) = \mathbf{0}$.

阻尼牛顿法

例 3.12: 用阻尼牛顿法求解:

$$\min f(\mathbf{x}) = x_1^2 + 2x_2^2 - 4x_1 - 2x_1x_2, \text{ 取初始点为 } \mathbf{x}^0 = (1, 1)^T$$

阻尼牛顿法

例 3.12: 用阻尼牛顿法求解:

$$\min f(\mathbf{x}) = x_1^2 + 2x_2^2 - 4x_1 - 2x_1x_2, \text{ 取初始点为 } \mathbf{x}^0 = (1, 1)^T$$

解:

$$\frac{\partial f(\mathbf{x})}{\partial x_1} = 2x_1 - 4 - 2x_2, \quad \frac{\partial f(\mathbf{x})}{\partial x_2} = 4x_2 - 2x_1,$$

$$\frac{\partial^2 f(\mathbf{x})}{\partial x_1^2} = 2, \quad \frac{\partial^2 f(\mathbf{x})}{\partial x_2^2} = 4, \quad \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_1} = \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_2} = -2$$

所以 $\nabla f(\mathbf{x}^0) = \begin{pmatrix} -4 \\ 2 \end{pmatrix}, \nabla^2 f(\mathbf{x}^0) = \begin{pmatrix} 2 & -2 \\ -2 & 4 \end{pmatrix}$

故 $[\nabla^2 f(\mathbf{x}^0)]^{-1} = \begin{pmatrix} 1 & 1/2 \\ 1/2 & 1/2 \end{pmatrix}$

阻尼牛顿法

所以

$$\begin{aligned}\mathbf{d}^0 &= -[\nabla^2 f(\mathbf{x}^0)]^{-1} \nabla f(\mathbf{x}^0) \\ &= -\begin{pmatrix} 1 & 1/2 \\ 1/2 & 1/2 \end{pmatrix} \begin{pmatrix} -4 \\ 2 \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \end{pmatrix}\end{aligned}$$

接下来计算步长:

$$\begin{aligned}\mathbf{x}^0 + \alpha \mathbf{d}^0 &= \begin{pmatrix} 1 \\ 1 \end{pmatrix} + \alpha \begin{pmatrix} 3 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 + 3\alpha \\ 1 + \alpha \end{pmatrix} \\ f(\mathbf{x}^0 + \alpha \mathbf{d}^0) &= f(1 + 3\alpha, 1 + \alpha) = 5\alpha^2 - 10\alpha - 3\end{aligned}$$

阻尼牛顿法

令

$$\frac{df(\alpha)}{d\alpha} = 10\alpha - 10 = 0 \quad \text{得: } \alpha_0 = 1.$$

于是

$$\mathbf{x}^1 = \mathbf{x}^0 + \alpha_0 \mathbf{d}^0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} + 1 \begin{pmatrix} 3 \\ 1 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix}$$

注意到

$$\nabla f(\mathbf{x}^1) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \text{即 } \|\nabla f(\mathbf{x}^1)\| = 0$$

因此 $\mathbf{x}^* = \mathbf{x}^1 = \begin{pmatrix} 4 \\ 2 \end{pmatrix}$ 是所求的极小点.

阻尼牛顿法

针对每次迭代都要计算Hessian矩阵、计算量大的改进方法：

- 为减小工作量, 取 m (正整数), 使每 m 次迭代使用同一个Hessian矩阵. 但收敛速度随 m 的增大而下降. 具体而言, 随 $m \rightarrow +\infty$, 算法收敛速度逐渐降低为即线性收敛.
- 采用拟牛顿法, 随着迭代的进程逐渐近似Hessian矩阵 $\nabla^2 f(\mathbf{x}^k)$.

阻尼牛顿法

针对 $\nabla^2 f(\mathbf{x}^k)$ 非正定和奇异的改进方法：

- 当 d^k 为函数上升方向时, 可向其负方向搜索, 但可能出现在 $\pm d^k$ 方向上函数值都不变化的情况.
- 找到尽可能小的 $\mu > 0$ 使 $\nabla^2 f(\mathbf{x}^k) + \mu I$ 正定, 用 $\nabla^2 f(\mathbf{x}^k) + \mu I$ 取代 $\nabla^2 f(\mathbf{x}^k)$ 进行迭代. 且该策略为全局二阶收敛.

- Goldstein-Price法: 若 $\nabla^2 f(\mathbf{x}^k)$ 正定, 则

取 $d^k = -[\nabla^2 f(\mathbf{x}^k)]^{-1} \nabla f(\mathbf{x}^k)$; 否则取 $d^k = -\nabla f(\mathbf{x}^k)$. 同时, 采用非精确一维搜索(Armijo-Goldstein准则)搜索步长 α_k . 在一定条件下, Goldstein-Price法全局收敛. 但当迭代过程中 $\nabla^2 f(\mathbf{x}^k)$ 非正定情况较多时, 收敛速度降为接近线性.

- 采用拟牛顿法, 随着迭代不断近似Hessian矩阵的逆 $[\nabla^2 f(\mathbf{x}^k)]^{-1}$.

目录

- ① 迭代法概述
- ② 一维搜索方法(线搜索方法)
- ③ 最速下降法(梯度下降法)
- ④ 共轭梯度法
- ⑤ 牛顿法(牛顿法和阻尼牛顿法)
- ⑥ 拟牛顿法(变尺度法)
- ⑦ 信赖域法

拟牛顿法(变尺度法)

一. 拟牛顿法基本思想

牛顿方向:

$$\mathbf{d}^k = -[\nabla^2 f(\mathbf{x}^k)]^{-1} \nabla f(\mathbf{x}^k).$$

可以看到, 牛顿法在每次迭代时都需要计算Hessian矩阵 $\nabla^2 f(\mathbf{x}^k)$ 及其逆矩阵, 计算和存储量都较大.

1959年, Davidon提出仅用每次迭代中得到的梯度信息来近似海森矩阵或其逆矩阵的思想, 由此产生了一类非常成功的算法—拟牛顿法.

目前最常用的两个拟牛顿法:

- Davidon-Fletcher-Powell (DFP) 算法
- Broyden-Fletcher-Goldfarb-Shanno(BFGS) 算法

拟牛顿法-DFP算法

DFP算法思想及原理

记 H_k 为 $[\nabla^2 f(\mathbf{x}^k)]^{-1}$ 的近似矩阵, 则新算法中每次的搜索方向为

$$\mathbf{d}^k = -H_k \nabla f(\mathbf{x}^k).$$

由此产生的方法称为变尺度法, H_k 称为尺度矩阵.

思考: 要使新算法比牛顿法计算和存储量小且同时整体收敛性好, 关键在于:

拟牛顿法-DFP算法

DFP算法思想及原理

记 H_k 为 $[\nabla^2 f(\mathbf{x}^k)]^{-1}$ 的近似矩阵, 则新算法中每次的搜索方向为

$$\mathbf{d}^k = -H_k \nabla f(\mathbf{x}^k).$$

由此产生的方法称为变尺度法, H_k 称为尺度矩阵.

思考: 要使新算法比牛顿法计算和存储量小且同时整体收敛性好, 关键在于:

如何构造近似矩阵阵列 $\{H_k\}$.

要求: $\{H_k\}$ 的选取既能逐步逼近 $[\nabla^2 f(\mathbf{x}^k)]^{-1}$, 又无需计算二阶导数.

拟牛顿法-DFP算法

$\{H_k\}$ 需满足以下条件:

C1 H_k 是对称矩阵

C2 H_k 由 H_{k-1} 经简单修正而得:

$$H_k = H_{k-1} + E_k$$

C3 H_k 满足拟牛顿方程

$$H_k \gamma_k = \delta_k$$

其中 $\delta_k = \mathbf{x}^k - \mathbf{x}^{k-1}$, $\gamma_k = \nabla f(\mathbf{x}^k) - \nabla f(\mathbf{x}^{k-1})$

以下给出 $\{H_k\}$ 需满足C3的原因:

拟牛顿法-DFP算法

设 $f(\mathbf{x})$ 是二次连续可微的, 由 Taylor 公式

$$f(\mathbf{x}) \approx f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T (\mathbf{x} - \mathbf{x}^k) + \frac{1}{2} (\mathbf{x} - \mathbf{x}^k)^T \nabla^2 f(\mathbf{x}^k) (\mathbf{x} - \mathbf{x}^k)$$

$$\nabla f(\mathbf{x}) \approx \nabla f(\mathbf{x}^k) + \nabla^2 f(\mathbf{x}^k) (\mathbf{x} - \mathbf{x}^k)$$

令 $\mathbf{x} = \mathbf{x}^{k-1}$, 则:

$$\nabla^2 f(\mathbf{x}^k) (\mathbf{x}^k - \mathbf{x}^{k-1}) \approx \nabla f(\mathbf{x}^k) - \nabla f(\mathbf{x}^{k-1}).$$

令 $\delta_k = \mathbf{x}^k - \mathbf{x}^{k-1}$, $\gamma_k = \nabla f(\mathbf{x}^k) - \nabla f(\mathbf{x}^{k-1})$, 因此 $\nabla^2 f(\mathbf{x}^k) \delta_k \approx \gamma_k$ (对二次函数为等号).

若 $[\nabla^2 f(\mathbf{x}^k)]$ 可逆, 则有 $[\nabla^2 f(\mathbf{x}^k)]^{-1} \gamma_k \approx \delta_k$.

用 H_k 代替上式中的 $[\nabla^2 f(\mathbf{x}^k)]^{-1}$ 并强制上式取等号, 得到拟牛顿方程.

拟牛顿法-DFP算法

对称秩1算法

设校正矩阵的形式为:

$$\mathbf{H}_k = \mathbf{H}_{k-1} + \beta_k \mathbf{u}_k \mathbf{u}_k^T \cdots \cdots (I)$$

其中 $\beta_k \neq 0$, $\mathbf{u}_k = (u_{k1}, u_{k2}, \dots, u_{kn})^T \neq (0, 0, \dots, 0)^T$
代入拟牛顿方程 $\mathbf{H}_k \boldsymbol{\gamma}_k = \boldsymbol{\delta}_k$ 得:

$$\mathbf{H}_{k-1} \boldsymbol{\gamma}_k + \beta_k \mathbf{u}_k \mathbf{u}_k^T \boldsymbol{\gamma}_k = \boldsymbol{\delta}_k$$

即: $\beta_k \mathbf{u}_k (\mathbf{u}_k^T \boldsymbol{\gamma}_k) = \boldsymbol{\delta}_k - \mathbf{H}_{k-1} \boldsymbol{\gamma}_k$

注意到 $\beta_k, \mathbf{u}_k^T \boldsymbol{\gamma}_k$ 都是数, 所以向量 \mathbf{u}_k 与 $\boldsymbol{\delta}_k - \mathbf{H}_{k-1} \boldsymbol{\gamma}_k$ 成比例. 因此不妨取 $\beta_k (\mathbf{u}_k^T \boldsymbol{\gamma}_k) = 1$, 则有 $\mathbf{u}_k = \boldsymbol{\delta}_k - \mathbf{H}_{k-1} \boldsymbol{\gamma}_k$

拟牛顿法-DFP算法

此时

$$\beta_k = \frac{1}{\mathbf{u}_k^T \boldsymbol{\gamma}_k} = \frac{1}{\boldsymbol{\gamma}_k^T \mathbf{u}_k} = \frac{1}{\boldsymbol{\gamma}_k^T (\boldsymbol{\delta}_k - \mathbf{H}_{k-1} \boldsymbol{\gamma}_k)}$$

代入到 (1) 式, 得

$$\mathbf{H}_k = \mathbf{H}_{k-1} + \frac{(\boldsymbol{\delta}_k - \mathbf{H}_{k-1} \boldsymbol{\gamma}_k) (\boldsymbol{\delta}_k - \mathbf{H}_{k-1} \boldsymbol{\gamma}_k)^T}{\boldsymbol{\gamma}_k^T (\boldsymbol{\delta}_k - \mathbf{H}_{k-1} \boldsymbol{\gamma}_k)}$$

秩1校正公式

拟牛顿法-DFP算法

秩1算法

- 1) 给出 $\mathbf{x}^0 \in \mathbb{R}^n$, $\mathbf{H}_0 = \mathbf{I}$, $0 \leq \epsilon \ll 1$, $k = 0$, 取 $\mathbf{d}^0 = -\nabla f(\mathbf{x}^0)$
- 2) 若 $\|\nabla f(\mathbf{x}^k)\|_2 \leq \epsilon$, 停止运算, 此时取 $\mathbf{x}^* = \mathbf{x}^k$;
否则, 进行精确线搜索求 α_k , 即 $\alpha_k = \arg \min f(\mathbf{x}^k + \alpha \mathbf{d}^k)$, 并
令 $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$.
- 3) 计算
$$\begin{aligned}\boldsymbol{\delta}_{k+1} &= \mathbf{x}^{k+1} - \mathbf{x}^k, \quad \boldsymbol{\gamma}_{k+1} = \nabla f(\mathbf{x}^{k+1}) - \nabla f(\mathbf{x}^k), \\ \mathbf{H}_{k+1} &= \mathbf{H}_k + \frac{(\boldsymbol{\delta}_{k+1} - \mathbf{H}_k \boldsymbol{\gamma}_{k+1})(\boldsymbol{\delta}_{k+1} - \mathbf{H}_k \boldsymbol{\gamma}_{k+1})^T}{\boldsymbol{\gamma}_{k+1}^T (\boldsymbol{\delta}_{k+1} - \mathbf{H}_k \boldsymbol{\gamma}_{k+1})}, \\ \mathbf{d}^{k+1} &= -\mathbf{H}_{k+1} \nabla f(\mathbf{x}^{k+1})\end{aligned}$$
- 4) 令 $k = k + 1$, 返回 Step 2).

【注】 第一步迭代与最速下降法相同

拟牛顿法-DFP算法

优点:

- 算法简洁, 具有二次终止性.

缺点:

- 当 H_{k-1} 正定时, 由秩1公式确定的 H_k 不一定正定, 因此不能保证搜索方向 $d^k = -H_k \nabla f(\mathbf{x}^k)$ 是下降方向;
- 秩1公式中分母可能为零或者近似于零, 导致算法不稳定.

拟牛顿法-DFP算法

对称秩2算法(DFP校正公式)

设校正矩阵的形式为:

$$\mathbf{H}_k = \mathbf{H}_{k-1} + \lambda_k \mathbf{u}_k \mathbf{u}_k^T + \beta_k \mathbf{v}_k \mathbf{v}_k^T \cdots \quad (II)$$

其中 $\lambda_k \neq 0$, $\mathbf{u}_k = (u_{k1}, u_{k2}, \dots, u_{kn})^T \neq (0, 0, \dots, 0)^T$
 $\beta_k \neq 0$, $\mathbf{v}_k = (v_{k1}, v_{k2}, \dots, v_{kn})^T \neq (0, 0, \dots, 0)^T$

代入拟牛顿方程 $\mathbf{H}_k \boldsymbol{\gamma}_k = \boldsymbol{\delta}_k$ 得到

$$\mathbf{H}_{k-1} \boldsymbol{\gamma}_k + \lambda_k \mathbf{u}_k \mathbf{u}_k^T \boldsymbol{\gamma}_k + \beta_k \mathbf{v}_k \mathbf{v}_k^T \boldsymbol{\gamma}_k = \boldsymbol{\delta}_k$$

$$\text{即 } \lambda_k \mathbf{u}_k \mathbf{u}_k^T \boldsymbol{\gamma}_k + \beta_k \mathbf{v}_k \mathbf{v}_k^T \boldsymbol{\gamma}_k = \boldsymbol{\delta}_k - \mathbf{H}_{k-1} \boldsymbol{\gamma}_k$$

拟牛顿法-DFP算法

若 $\lambda_k \mathbf{u}_k (\mathbf{u}_k^T \boldsymbol{\gamma}_k) = \boldsymbol{\delta}_k$, $\beta_k \mathbf{v}_k (\mathbf{v}_k^T \boldsymbol{\gamma}_k) = -\mathbf{H}_{k-1} \boldsymbol{\gamma}_k$, 上式成立

注意到 $\lambda_k, \mathbf{u}_k^T \boldsymbol{\gamma}_k, \beta_k, \mathbf{v}_k^T \boldsymbol{\gamma}_k$ 都是数, 那么若令

$$\lambda_k (\mathbf{u}_k^T \boldsymbol{\gamma}_k) = 1, \beta_k (\mathbf{v}_k^T \boldsymbol{\gamma}_k) = -1$$

则有 $\mathbf{u}_k = \boldsymbol{\delta}_k$, $\mathbf{v}_k = \mathbf{H}_{k-1} \boldsymbol{\gamma}_k$.

将这些代入(II)得:

$$\mathbf{H}_k = \mathbf{H}_{k-1} + \frac{\boldsymbol{\delta}_k \boldsymbol{\delta}_k^T}{\boldsymbol{\delta}_k^T \boldsymbol{\gamma}_k} - \frac{\mathbf{H}_{k-1} \boldsymbol{\gamma}_k \boldsymbol{\gamma}_k^T \mathbf{H}_{k-1}}{\boldsymbol{\gamma}_k^T \mathbf{H}_{k-1} \boldsymbol{\gamma}_k}$$

DFP公式

拟牛顿法-DFP算法

- 1) 给出 $\mathbf{x}^0 \in \mathbb{R}^n$, $\mathbf{H}_0 = I$, $0 \leq \epsilon \ll 1$, $k = 0$, 取 $\mathbf{d}^0 = -\nabla f(\mathbf{x}^0)$
- 2) 若 $\|\nabla f(\mathbf{x}^k)\|_2 \leq \epsilon$, 停止运算, 此时取 $\mathbf{x}^* = \mathbf{x}^k$;
否则, 进行精确线搜索求 α_k , 即
$$\alpha_k = \arg \min f(\mathbf{x}^k + \alpha \mathbf{d}^k)$$
, 并令 $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$.
- 3) 计算

$$\begin{aligned}\boldsymbol{\delta}_{k+1} &= \mathbf{x}^{k+1} - \mathbf{x}^k, \quad \boldsymbol{\gamma}_{k+1} = \nabla f(\mathbf{x}^{k+1}) - \nabla f(\mathbf{x}^k), \\ \mathbf{H}_{k+1} &= \mathbf{H}_k + \frac{\boldsymbol{\delta}_{k+1} \boldsymbol{\delta}_{k+1}^T}{\boldsymbol{\delta}_{k+1}^T \boldsymbol{\gamma}_{k+1}} - \frac{\mathbf{H}_k \boldsymbol{\gamma}_{k+1} \boldsymbol{\gamma}_{k+1}^T \mathbf{H}_k}{\boldsymbol{\gamma}_{k+1}^T \mathbf{H}_k \boldsymbol{\gamma}_{k+1}} \\ \mathbf{d}^{k+1} &= -\mathbf{H}_{k+1} \nabla f(\mathbf{x}^{k+1})\end{aligned}$$

- 4) 令 $k = k + 1$, 返回 Step 2.

【注】 第一步迭代与最速下降法相同

拟牛顿法-DFP算法

例 3.13: 用DFP算法求解:

$$\min f(\mathbf{x}) = x_1^2 + 4x_2^2$$

取初始 $\mathbf{x}^0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, $H_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, 精度 $\epsilon = 10^{-4}$

拟牛顿法-DFP算法

例 3.13: 用DFP算法求解:

$$\min f(\mathbf{x}) = x_1^2 + 4x_2^2$$

取初始 $\mathbf{x}^0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, $H_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, 精度 $\epsilon = 10^{-4}$

解: $\nabla f(\mathbf{x}) = \begin{pmatrix} 2x_1 \\ 8x_2 \end{pmatrix}$

(1) $\nabla f(\mathbf{x}^0) = \begin{pmatrix} 2 \\ 8 \end{pmatrix}$, $\mathbf{d}^0 = -\nabla f(\mathbf{x}^0) = \begin{pmatrix} -2 \\ -8 \end{pmatrix}$

$\varphi(\alpha) = f(\mathbf{x}^0 + \alpha \mathbf{d}^0) = (1 - 2\alpha)^2 + 4(1 - 8\alpha)^2$

令 $\varphi'(\alpha) = 0$ 得: $\alpha_0 = \frac{68}{520} = 0.13077$

拟牛顿法-DFP算法

进而有：

$$\boldsymbol{x}^1 = \boldsymbol{x}^0 + \alpha_0 \boldsymbol{d}^0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - 0.13077 \begin{pmatrix} 2 \\ 8 \end{pmatrix} = \begin{pmatrix} 0.73846 \\ -0.04616 \end{pmatrix}$$

$$\nabla f(\boldsymbol{x}^1) = (1.47692, -0.36923)^T,$$

$$\|\nabla f(\boldsymbol{x}^1)\|_2 = 1.52237$$

由于 $\|\nabla f(\boldsymbol{x}^1)\|_2$ 较大，因此还需迭代下去。

(2) 先确定搜索方向，为此需要按公式计算

$$\boldsymbol{H}_1 = \boldsymbol{H}_0 + \frac{\boldsymbol{\delta}_1 \boldsymbol{\delta}_1^T}{\boldsymbol{\delta}_1^T \boldsymbol{\gamma}_1} - \frac{\boldsymbol{H}_0 \boldsymbol{\gamma}_1 \boldsymbol{\gamma}_1^T \boldsymbol{H}_0}{\boldsymbol{\gamma}_1^T \boldsymbol{H}_0 \boldsymbol{\gamma}_1}$$

拟牛顿法-DFP算法

又

$$\delta_1 = \mathbf{x}^1 - \mathbf{x}^0 = \begin{pmatrix} -0.26154 \\ -1.04616 \end{pmatrix}$$

$$\gamma_1 = \nabla f(\mathbf{x}^1) - \nabla f(\mathbf{x}^0) = \begin{pmatrix} -0.52308 \\ -8.36923 \end{pmatrix}$$

故

$$\delta_1^T \gamma_1 = 8.89236$$

$$\gamma_1^T \mathbf{H}_0 \gamma_1 = \gamma_1^T \gamma_1 = 70.31762$$

$$\delta_1 \delta_1^T = \begin{pmatrix} 0.06840 & 0.27361 \\ 0.27361 & 1.09445 \end{pmatrix}$$

$$\mathbf{H}_0 \gamma_1 \gamma_1^T \mathbf{H}_0 = \gamma_1 \gamma_1^T = \begin{pmatrix} 0.27361 & 4.37778 \\ 4.37778 & 70.04401 \end{pmatrix}$$

拟牛顿法-DFP算法

所以

$$\mathbf{H}_1 = \begin{pmatrix} 1.00380 & -0.03149 \\ -0.03149 & 0.12697 \end{pmatrix}$$
$$\mathbf{d}^1 = -\mathbf{H}_1 \nabla f(\mathbf{x}^1) = \begin{pmatrix} -1.49416 \\ 0.09340 \end{pmatrix}$$

利用 $\frac{df(\mathbf{x}^1 + \alpha \mathbf{d}^1)}{d\alpha} = 0$ 得: $\alpha_1 = 0.49423$.

所以 $\mathbf{x}^2 = \mathbf{x}^1 + \alpha_1 \mathbf{d}^1 = \begin{pmatrix} 0.00000 \\ 0.00000 \end{pmatrix}$

因 $\|\nabla f(\mathbf{x}^2)\|_2 = 0$, 于是停止迭代, 最优解为 $\mathbf{x}^* = \mathbf{x}^2 = (0, 0)^T$.

拟牛顿法-DFP算法

定理 3.10：收敛定理

设 $f(\mathbf{x})$ 一次连续可微, $\mathbf{x}^0 \in \mathbb{R}^n$, 假设水平集 $\{\mathbf{x} | f(\mathbf{x}) \leq f(\mathbf{x}^0)\}$ 有界, 则对由 DFP 算法产生的点列 $\{\mathbf{x}^k\}$ 具有如下结论:

1. 若 $\{\mathbf{x}^k\}$ 是有穷点列且 \mathbf{x}^k 是 $f(\mathbf{x})$ 的驻点.
2. 若 $\{\mathbf{x}^k\}$ 是无穷点列, 它必有极限且任一极限是 $f(\mathbf{x})$ 的驻点.

拟牛顿法-DFP算法

优点

- 若目标函数 $f(\mathbf{x})$ 是二次严格凸函数, 则由 DFP 算法产生的搜索方向 $\mathbf{d}^1, \mathbf{d}^2, \dots, \mathbf{d}^k$ 关于矩阵 $\nabla^2 f(\mathbf{x})$ 共轭的. 因此, DFP 算法也是一种共轭方向法, 故其具有二次终止性;
- 若目标函数 $f(\mathbf{x})$ 是至少一阶连续可微的严格凸函数, 则 DFP 算法是全局收敛的;
- 若 \mathbf{H}_k 正定, $\nabla f(\mathbf{x}^k) \neq 0$, 则 \mathbf{H}_{k+1} 正定;
- DFP 算法克服了秩 1 校正中分母为 0 或者近似为 0 的缺点.
- 对非二次函数, DFP 算法的效果也很好, 它比最速下降法和共轭梯度法要有效的多, 收敛速度是超线性的.

拟牛顿法-DFP算法

缺点

- 需要的存储量较大, 大约需要 $O(n^2)$ 个存储单元, 因此对于大规模问题, 与共轭梯度法相比, 计算量、存储量较大;
- 实际运算中, 由于舍入误差的存在以及一维搜索的不精确, DFP算法的稳定性和计算效率都会受到很大的影响, 数值计算的稳定性不如后面将介绍的BFGS算法;
- 若求步长时不采用精确一维搜索, 则计算效率不如BFGS算法好.

拟牛顿法-BFGS算法

BFGS算法思想及原理:

在对牛顿方向修正时, 若不考虑构造 $[\nabla^2 f(\mathbf{x}^k)]^{-1}$ 的近似矩阵, 而是考虑Hessian矩阵 $\nabla^2 f(\mathbf{x}^k)$ 的近似矩阵 B_k , 同样可构造出另一类变尺度法/拟牛顿算法, 其中BFGS算法是其中之一.

记 B_k 为 $[\nabla^2 f(\mathbf{x}^k)]$ 的近似矩阵, 则新算法中每次搜索方向满足

$$B_k \mathbf{d}^k + \nabla f(\mathbf{x}^k) = \mathbf{0}.$$

思考: 要使新算法比牛顿法计算和存储量小, 且整体收敛性好, 关键在于:

拟牛顿法–BFGS算法

BFGS算法思想及原理:

在对牛顿方向修正时, 若不考虑构造 $[\nabla^2 f(\mathbf{x}^k)]^{-1}$ 的近似矩阵, 而是考虑Hessian矩阵 $\nabla^2 f(\mathbf{x}^k)$ 的近似矩阵 B_k , 同样可构造出另一类变尺度法/拟牛顿算法, 其中BFGS算法是其中之一.

记 B_k 为 $[\nabla^2 f(\mathbf{x}^k)]$ 的近似矩阵, 则新算法中每次搜索方向满足

$$B_k \mathbf{d}^k + \nabla f(\mathbf{x}^k) = \mathbf{0}.$$

思考: 要使新算法比牛顿法计算和存储量小, 且整体收敛性好, 关键在于: 如何构造近似矩阵阵列 $\{B_k\}$.

要求: $\{B_k\}$ 的选取既能逐步逼近 $\nabla^2 f(\mathbf{x}^k)$, 又无需计算二阶导数.

拟牛顿法-BFGS算法

需具备以下条件:

C1 B_k 是对称矩阵

C2 B_k 由 B_{k-1} 经简单修正而得:

$$B_k = B_{k-1} + C_k$$

C3 B_k 满足拟牛顿方程.

$$B_k \delta_k = \gamma_k$$

其中 $\delta_k = \mathbf{x}^k - \mathbf{x}^{k-1}$, $\gamma_k = \nabla f(\mathbf{x}^k) - \nabla f(\mathbf{x}^{k-1})$

以下给出需满足C3的原因:

拟牛顿法-BFGS算法

设 $f(\mathbf{x})$ 是二次连续可微的, 由 Taylor 公式

$$f(\mathbf{x}) \approx f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T (\mathbf{x} - \mathbf{x}^k) + \frac{1}{2} (\mathbf{x} - \mathbf{x}^k)^T \nabla^2 f(\mathbf{x}^k) (\mathbf{x} - \mathbf{x}^k)$$
$$\nabla f(\mathbf{x}) \approx \nabla f(\mathbf{x}^k) + \nabla^2 f(\mathbf{x}^k) (\mathbf{x} - \mathbf{x}^k)$$

令 $\mathbf{x} = \mathbf{x}^{k-1}$, 则: $\nabla^2 f(\mathbf{x}^k) (\mathbf{x}^k - \mathbf{x}^{k-1}) \approx \nabla f(\mathbf{x}^k) - \nabla f(\mathbf{x}^{k-1})$

令 $\delta_k = \mathbf{x}^k - \mathbf{x}^{k-1}$, $\gamma_k = \nabla f(\mathbf{x}^k) - \nabla f(\mathbf{x}^{k-1})$, 因此,
 $\nabla^2 f(\mathbf{x}^k) \delta_k \approx \gamma_k$ (对二次函数为等式).

用 B_k 代替上式中的 $[\nabla^2 f(\mathbf{x}^k)]$ 并强制上式取等号, 得到拟牛顿方程 $B_k \delta_k = \gamma_k$

拟牛顿法-BFGS算法

设校正矩阵的形式为：

$$B_k = B_{k-1} + \lambda_k u_k u_k^T + \beta_k v_k v_k^T \quad \dots \dots \quad (I)$$

其中 $\lambda_k \neq 0$, $u_k = (u_{k1}, u_{k2}, \dots, u_{kn})^T \neq (0, 0, \dots, 0)^T$
 $\beta_k \neq 0$, $v_k = (v_{k1}, v_{k2}, \dots, v_{kn})^T \neq (0, 0, \dots, 0)^T$

代入拟牛顿方程 $B_k \delta_k = \gamma_k$

得到 $B_{k-1} \delta_k + \lambda_k u_k u_k^T \delta_k + \beta_k v_k v_k^T \delta_k = \gamma_k$

即 $\lambda_k u_k u_k^T \delta_k + \beta_k v_k v_k^T \delta_k = \gamma_k - B_{k-1} \delta_k$

拟牛顿法-BFGS算法

即 $\lambda_k \mathbf{u}_k \mathbf{u}_k^T \boldsymbol{\delta}_k + \beta_k \mathbf{v}_k \mathbf{v}_k^T \boldsymbol{\delta}_k = \boldsymbol{\gamma}_k - \mathbf{B}_{k-1} \boldsymbol{\delta}_k$

可见若 $\lambda_k \mathbf{u}_k (\mathbf{u}_k^T \boldsymbol{\delta}_k) = \gamma_k$, $\beta_k \mathbf{v}_k (\mathbf{v}_k^T \boldsymbol{\delta}_k) = -\mathbf{B}_{k-1} \boldsymbol{\delta}_k$ 上式成立

注意到 $\lambda_k, \mathbf{u}_k^T \boldsymbol{\delta}_k, \beta_k, \mathbf{v}_k^T \boldsymbol{\delta}_k$ 都是数, 那么若令 $\lambda_k (\mathbf{u}_k^T \boldsymbol{\delta}_k) = 1$,
 $\beta_k (\mathbf{v}_k^T \boldsymbol{\delta}_k) = -1$, 则有 $\mathbf{u}_k = \boldsymbol{\gamma}_k, \mathbf{v}_k = \mathbf{B}_{k-1} \boldsymbol{\delta}_k$

将这些代入(II)得:

$$\mathbf{B}_k = \mathbf{B}_{k-1} + \frac{\boldsymbol{\gamma}_k \boldsymbol{\gamma}_k^T}{\boldsymbol{\gamma}_k^T \boldsymbol{\delta}_k} - \frac{\mathbf{B}_{k-1} \boldsymbol{\delta}_k \boldsymbol{\delta}_k^T \mathbf{B}_{k-1}}{\boldsymbol{\delta}_k^T \mathbf{B}_{k-1} \boldsymbol{\delta}_k}$$

BFGS校正公式

拟牛顿法-BFGS算法

BFGS算法

- 1) 给出 $\mathbf{x}^0 \in \mathbb{R}^n, \mathbf{B}_0 = \mathbf{I}, 0 \leq \epsilon << 1, k = 0$
取 $\mathbf{d}^0 = -\nabla f(\mathbf{x}^0)$
- 2) 若 $\|\nabla f(\mathbf{x}^k)\|_2 \leq \epsilon$, 停止运算, 此时取 $\mathbf{x}^* = \mathbf{x}^k$;
否则, 进行精确线搜索求 α_k , 即
 $\alpha_k = \arg \min f(\mathbf{x}^k + \alpha \mathbf{d}^k)$, 并令 $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$.
- 3) 计算
 $\boldsymbol{\delta}_{k+1} = \mathbf{x}^{k+1} - \mathbf{x}^k, \boldsymbol{\gamma}_{k+1} = \nabla f(\mathbf{x}^{k+1}) - \nabla f(\mathbf{x}^k),$
$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{\boldsymbol{\gamma}_{k+1} \boldsymbol{\gamma}_{k+1}^T}{\boldsymbol{\gamma}_{k+1}^T \boldsymbol{\delta}_{k+1}} - \frac{\mathbf{B}_k \boldsymbol{\delta}_{k+1} \boldsymbol{\delta}_{k+1}^T \mathbf{B}_k}{\boldsymbol{\delta}_{k+1}^T \mathbf{B}_k \boldsymbol{\delta}_{k+1}}$$

解方程组 $\mathbf{B}_{k+1} \mathbf{d}^{k+1} + \nabla f(\mathbf{x}^{k+1}) = \mathbf{0}$ 得 \mathbf{d}^{k+1}
- 4) 令 $k = k + 1$, 返回Step 2).

【注】第一步迭代与最速下降法相同

拟牛顿法-BFGS算法

BFGS算法与DFP算法的比较

BFGS算法与DFP算法都属于拟牛顿算法, 二者性质有一定类似性.BFGS算法的数值稳定性比DFP算法好, 而且求步长若不用精确一维搜索, 仍然是超线性收敛的.

目录

① 迭代法概述

② 一维搜索方法(线搜索方法)

③ 最速下降法(梯度下降法)

④ 共轭梯度法

⑤ 牛顿法(牛顿法和阻尼牛顿法)

⑥ 拟牛顿法(变尺度法)

⑦ 信赖域法

信赖域法

上一节学习了牛顿法, 传统的牛顿法属于局部收敛算法(收敛与初始点选取有关)。为了得到下降迭代算法, 对此做了改进, 即当 $\nabla f(\mathbf{x}^k) \neq 0$, $\nabla^2 f(\mathbf{x}^k)$ 可逆时, 沿搜索方向 $\mathbf{d}^k = -\nabla^2 f(\mathbf{x}^k)^{-1} \nabla f(\mathbf{x}^k)$ 作一维搜索, 从而得到一个精确步长, 这就是阻尼牛顿法, 该方法具有全局收敛性。

这一节, 将介绍另一种具有全局收敛性的新方法—信赖域法。

信赖域法

基本思想

首先用迭代点 x^k 处的某二次近似函数 $p_k(x)$ 作为原函数的近似，然后给定一个邻域半径 Δ_k ，并由此定义 x^k 的一个邻域 $\Omega_k = N_{\Delta_k}(x^k)$ ，假定在 Ω_k 内上述二次函数是原函数的一个较好的近似，接下来求 $p_k(x)$ 在 Ω_k 内的极小点 x ，并判断在此邻域内 $p_k(x)$ 对 $f(x)$ 的近似程度：

- 若近似程度好，则将 x 作为下一个迭代点 x^{k+1} ，并判断它是否是原函数的极小点。若是，停止运算；否则，在 x^{k+1} 的一个邻域内用某个二次函数对 $f(x)$ 作近似。
- 若近似程度不好，则缩小 Δ_k ，构造 x^k 的新邻域，在此邻域内用 $p_k(x)$ 再对 $f(x)$ 做近似，求在此邻域内的极小点并判断其近似程度。

信赖域法

设 $f(\mathbf{x})$ 是二次连续可微的, 由 Taylor 公式

$$f(\mathbf{x}) \approx f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T (\mathbf{x} - \mathbf{x}^k) + \frac{1}{2} (\mathbf{x} - \mathbf{x}^k)^T \nabla^2 f(\mathbf{x}^k) (\mathbf{x} - \mathbf{x}^k)$$

记 $p_k(\mathbf{x}) = f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T (\mathbf{x} - \mathbf{x}^k) + \frac{1}{2} (\mathbf{x} - \mathbf{x}^k)^T \mathbf{B}_k (\mathbf{x} - \mathbf{x}^k)$,
其中 \mathbf{B}_k 是 $\nabla^2 f(\mathbf{x}^k)$ 本身或其近似矩阵。

令 $\mathbf{s} = \mathbf{x} - \mathbf{x}^k$, 则 $p_k(\mathbf{x})$ 变为: $q_k(\mathbf{s}) = f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T \mathbf{s} + \frac{1}{2} \mathbf{s}^T \mathbf{B}_k \mathbf{s}$ 。

那么, 求 $\min_{\|\mathbf{x} - \mathbf{x}^k\|_2 \leq \Delta_k} p_k(\mathbf{x})$ 便转化为求解 $\min_{\|\mathbf{s}\|_2 \leq \Delta_k} q_k(\mathbf{s})$, 其
中 $\mathbf{s}_k^* = \mathbf{x}_k^* - \mathbf{x}^k$ 。

若记 $\mathbf{x}_k^* = \arg \min_{\|\mathbf{x} - \mathbf{x}^k\|_2 \leq \Delta_k} p_k(\mathbf{x})$, $\mathbf{s}_k^* = \arg \min_{\|\mathbf{s}\|_2 \leq \Delta_k} q_k(\mathbf{s})$,
则 $\mathbf{s}_k^* = \mathbf{x}_k^* - \mathbf{x}^k$, $q_k(\mathbf{s}_k^*) = p_k(\mathbf{x}_k^*)$ 。

信赖域法

设 Δf_k 是在第 k 步的实际下降量, 即

$$\Delta f_k = f(\mathbf{x}^k) - f(\mathbf{x}_k^*)$$

设 Δq_k 是在第 k 步的预测下降量, 即

$$\Delta q_k = f(\mathbf{x}^k) - p_k(\mathbf{x}_k^*) = f(\mathbf{x}^k) - q_k(\mathbf{s}_k^*)。$$

定义比值: $r_k = \frac{\Delta f_k}{\Delta q_k}$ 来衡量函数的近似程度。

若 r_k 越接近1, 表明近似程度越好, 故取 \mathbf{x}_k^* 为下一个迭代点, 并加大信赖域半径。

若 r_k 接近于0, 说明近似程度不好, 仍然取 \mathbf{x}_k 作为下一个迭代点, 并缩小邻域半径。

信赖域法

- 1) 给定 $\mathbf{x}^0 \in \mathbb{R}^n$, $\mathbf{B}_0 = \mathbf{I}$, $0 \leq \epsilon << 1$, $k = 0$, 给定邻域半径 $\Delta_0 > 0$
- 2) 若 $\|\nabla f(\mathbf{x}^k)\|_2 \leq \epsilon$, 停止运算; 否则, 转Step 3)。
- 3) 求解 $\min_{\|\mathbf{s}\|_2 \leq \Delta_k} q_k(\mathbf{s})$ 得其最优解 \mathbf{s}_k^* 。 (求解该子问题的方法有Lagrange乘子法、截断共轭梯度法等。)
- 4) 计算 $\Delta f_k = f(\mathbf{x}^k) - f(\mathbf{x}^k + \mathbf{s}_k^*)$, $\Delta q_k = f(\mathbf{x}^k) - q_k(\mathbf{s}_k^*)$ 。
- 5) 若 $\frac{\Delta f_k}{\Delta q_k} \leq 0.25$, 令 $\mathbf{x}^{k+1} = \mathbf{x}^k$, $\mathbf{B}_{k+1} = \mathbf{B}_k$; 否则,
令 $\mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{s}_k^*$, 并按BFGS公式修正 \mathbf{B}_k , 即

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{\boldsymbol{\gamma}_{k+1} \boldsymbol{\gamma}_{k+1}^T}{\boldsymbol{\gamma}_{k+1}^T \boldsymbol{\delta}_{k+1}} - \frac{\mathbf{B}_k \boldsymbol{\delta}_{k+1} \boldsymbol{\delta}_{k+1}^T \mathbf{B}_k}{\boldsymbol{\delta}_{k+1}^T \mathbf{B}_k \boldsymbol{\delta}_{k+1}}.$$

信赖域法

6) 令

$$\Delta_{k+1} = \begin{cases} \frac{1}{2}\Delta_k & \text{若 } \frac{\Delta f_k}{\Delta q_k} \leq 0.25 \\ \Delta_k & \text{若 } 0.25 < \frac{\Delta f_k}{\Delta q_k} < 0.75 \\ 2\Delta_k & \text{若 } \frac{\Delta f_k}{\Delta q_k} \geq 0.75 \end{cases}$$

7) 令 $k = k + 1$, 返回Step 2);

优点

- 具有全局收敛性；
- 避免计算Hessian矩阵，同时不要求Hessian矩阵 $\nabla^2 f(\mathbf{x}^k)$ 正定；
- 不需要进行线搜索求步长。