计算方法

[1] 数学科学学院　中国科学技术大学

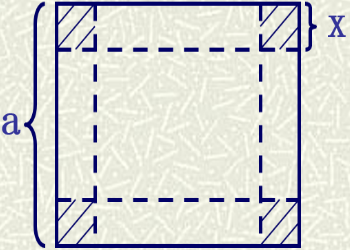
§10.4.1 数学模型

非线性优化问题

如果最优化问题中的**目标函数**或**约束条件**中有关于决策变量的**非线性函数**，则称它是*非线性优化*问题或*非线性规划*问题.

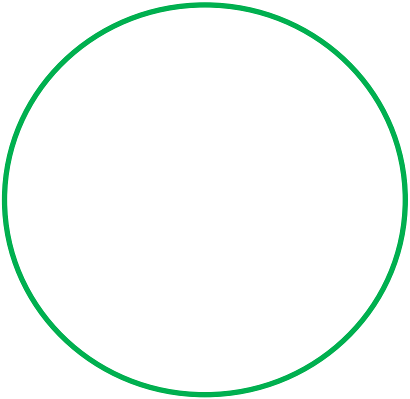
例 如图所示，如何截取x使铁皮所围成的容积最大?

$$v=(a-2x)^2\cdot x$$



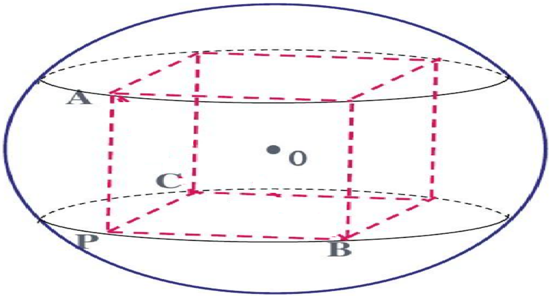
例 10.5

在半径为 R 的圆内有一内接三角形 $\triangle ABC$, 其顶点 A, B, C 可以在圆上移动, 求 $\triangle ABC$ 面积的最大值.



例 10.6

现有一块椭球形的石材, 其半轴分别为 $a, b, c > 0$. 若要从中加工出一块长方体的石砖, 要求长方体各边平行于椭球的半轴且浪费最少, 问该如何设计切割方案?



§10.4.1 数学模型

不失一般性，**非线性优化问题**可写成:

$$\boxed{\begin{aligned} &\min f(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n \\ &\text{s. t.} \begin{cases} c_i(\mathbf{x}) \geqslant 0, & i \in \mathcal{I} \\ c_i(\mathbf{x}) = 0, & i \in \mathcal{E}. \end{cases} \end{aligned}}$$

其中 $f(\mathbf{x}), c_i(\mathbf{x})$ 是有 n 个自变量的连续实函数, $\mathcal{E}, \mathcal{I}$ 分别是**等式约束和不等式约束**的有限指标集.

§10.4.1 数学模型

如果非线性优化问题中的目标函数和约束条件中的函数均为凸函数，则称它为**凸优化**问题.

如果目标函数的自变量为离散变量，譬如整数或有限集合中的元素，则称它为**离散优化**. 整数规划问题、旅行商问题、图着色问题、最小生成树等都是离散优化问题.

§10.4.2 函数极值

定义 10.9

设函数 $f: D \to \mathbb{R}$，其中区域 $D \subset \mathbb{R}^n$. 如果对于 D 的内点 $\mathbf{x}^*$, 存在领域 $U(\mathbf{x}^*) \subset D$, 当 $\mathbf{x} \in U(\mathbf{x}^*)$ 时, 有 $f(\mathbf{x}) \geqslant f(\mathbf{x}^*)$ $(f(\mathbf{x}) \leqslant f(\mathbf{x}^*))$, 则称函数 f 在 D 的内点 $\mathbf{x}^*$ 取***局部极小值*** (***局部极大值***), 简称***极小值*** (***极大值***). 如果对 $\mathbf{x} \in U(\mathbf{x}^*) \backslash \{\mathbf{x}^*\}$ 有严格不等式 $f(\mathbf{x}) > f(\mathbf{x}^*)$ $(f(\mathbf{x}) < f(\mathbf{x}^*))$, 则称函数 f 在 $\mathbf{x}^*$ 取***严格局部极小值*** (***严格局部极大值***).

定义 10.10

函数的局部极大值与局部极小值统称为函数的***局部极值***, 简称***极值***, 函数取极值的点称为***极值点***.

§10.4.2 函数极值

定理 10.4

设函数 $f: D \to \mathbb{R}$ 在内点 $\mathbf{x}^*$ 取极值且 $Jf(\mathbf{x}^*)$ 存在，则 $Jf(\mathbf{x}^*) = \mathbf{0}$，即

$$\frac{\partial f}{\partial x_i}(\mathbf{x}^*) = 0, \quad i = 1, 2, \cdots, n.$$

定义 10.11

设函数 $f: D \to \mathbb{R}$，在 D 中使得 $Jf(\mathbf{x}) = \mathbf{0}$ 的一切内点称为函数 f 的**驻点**.

§10.4.2 函数极值

显然，**极值点一定是驻点**，但一般来说**驻点未必是极值点**. 若记

$$
Jf(\mathbf{x}) = \left(\frac{\partial f(\mathbf{x})}{\partial x_1}, \cdots, \frac{\partial f(\mathbf{x})}{\partial x_n} \right), Hf(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1^2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_n^2} \end{pmatrix},
$$

分别称 $Jf(\mathbf{x})$ 和 $Hf(\mathbf{x})$ 为函数 f 的 **Jacobian 矩阵**和 **Hessian 矩阵**.

§10.4.2 函数极值

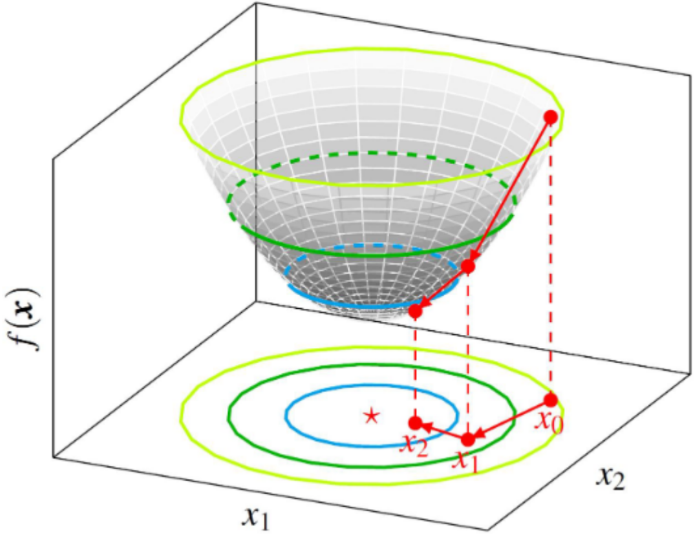
定理 10.5

设函数 $f(\mathbf{x})$ 在 D 上具有连续的二阶偏导数，即 $f(\mathbf{x}) \in C^2(D)$.

(1) (必要性) 若 $f(\mathbf{x})$ 在 $\mathbf{x}^*$ 取局部极小（大）值，则 $Jf(\mathbf{x}^*) = \mathbf{0}$ 且 $Hf(\mathbf{x}^*)$ 半正（负）定；

(2) (充分性) 若 $Jf(\mathbf{x}^*) = \mathbf{0}$ 且 $Hf(\mathbf{x}^*)$ 正（负）定，则 $f(\mathbf{x})$ 在 $\mathbf{x}^*$ 取得严格局部极小（大）值.

判别函数内部局部极值点的二阶必要条件和充分条件.


$f(\boldsymbol{x})$
x_0
x_1
x_2
x_1
x_2

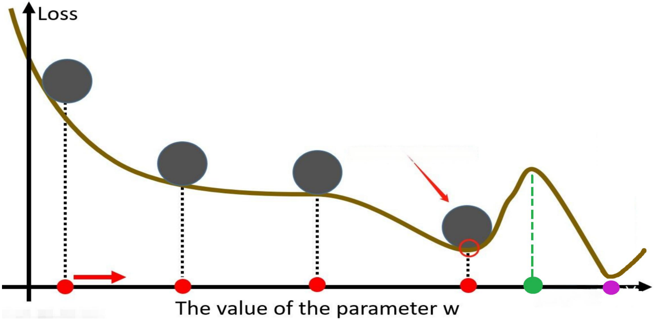
§10.4.2 函数极值

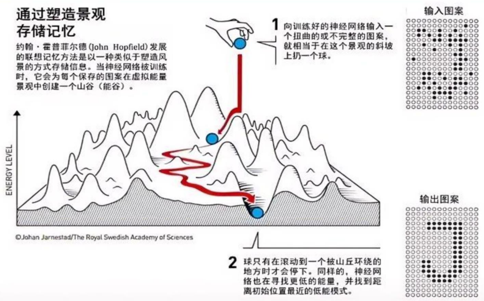
定义 10.12

设 $\mathbf{x}^* \in D$, 若对所有 $\mathbf{x} \in D$, 都有 $\boxed{f(x) \geqslant f(\mathbf{x}^*)}$ $(f(x) \leqslant f(\mathbf{x}^*))$, 则称 $\mathbf{x}^*$ 为 $f(\mathbf{x})$ 在 D 上的**全局极小值点** (**全局极大值点**). 若对所有 $\mathbf{x} \in D \backslash \mathbf{x}^*$, 都有 $f(x) > f(\mathbf{x}^*)$ $(f(x) < f(\mathbf{x}^*))$, 则称 $\mathbf{x}^*$ 为 $f(\mathbf{x})$ 在 D 上的**全局严格极小值点** (**全局严格极大值点**)

全局极值点可能在定义域的边界点上取到. 然而, 对于这些边界上的极值点, 并**没有实用的判别条件**. 因此, **寻找全局极值点是一个相当困难的任务**.

幸好, 在许多实际问题中, 局部极值点就已能满足应用的需求, 故**非线性优化主要研究寻找局部或　　极值点**.


Loss
The value of the parameter w


通过塑造景观
存储记忆
约翰·霍普菲尔德(John Hopfield)发展的联想记忆方法是以一种类似于塑造风景的方式存储信息。当神经网络被训练时，它会为每个保存的图案在虚拟能量景观中创建一个山谷（能谷）。
1 向训练好的神经网络输入一个扭曲的或不完整的图案，就相当于在这个景观的斜坡上扔一个球。
输入图案
ENERGY LEVEL
©Johan Jarnestad/The Royal Swedish Academy of Sciences
2 球只有在滚动到一个被山丘环绕的地方时才会停下。同样的，神经网络也在寻找更低的能量，并找到距离初始位置最近的低能模式。
输出图案

§10.4.2 函数极值

凸优化问题的极值

对于**定义在凸集上的凸函数** $f(\mathbf{x})$，可以证明: $f(\mathbf{x})$ 的**局部极值点就是全局极值点**，且所有极值点构成一个凸集. 进一步，若 $f(\mathbf{x})$ 是**严格凸函数**，则 $f(\mathbf{x})$ 的**全局极值点是唯一**的.

§10.4.3 迭代法

求解**非线性性优化**问题的一个自然的想法是: 1) 令 $\nabla f(\mathbf{x})=\mathbf{0}$, 求出 $f(\mathbf{x})$ 的所有**驻点**; 2) 对每一**驻点**, **利用局部极值的充分条件进行判别**, 找出所要的解.

对某些较简单的函数, 这样做是可行的, 但是, 对于一般的 n 元函数, 由条件 $\nabla f(\mathbf{x})=\mathbf{0}$ 得到的方程通常是一个**非线性方程组, 求解非常困难**. 当函数 $f(\mathbf{x})$ 不可微分时, 就更不能用上述方法了.

因此, **求解非线性性优化问题常使用迭代法**.

§10.4.3 迭代法

迭代法的**基本思路**是：为求函数 $f(\mathbf{x})$ 的最优解，首先给定一个**初始估计** $\mathbf{x}_0$，然后**按某种规则**找出比 $\mathbf{x}_0$ 更好的一个解 $\mathbf{x}_1$，即 $f(\mathbf{x}_1) < f(\mathbf{x}_0)$，重复上述过程，得到一个解序列 $\{\mathbf{x}_k\}_{k=0}^{\infty}$. 若这个解序列有极限 $\mathbf{x}^*$，即

$$\lim_{k\to\infty} \|\mathbf{x}_k - \mathbf{x}^*\| = 0,$$

则称它收敛于 $\mathbf{x}^*$.

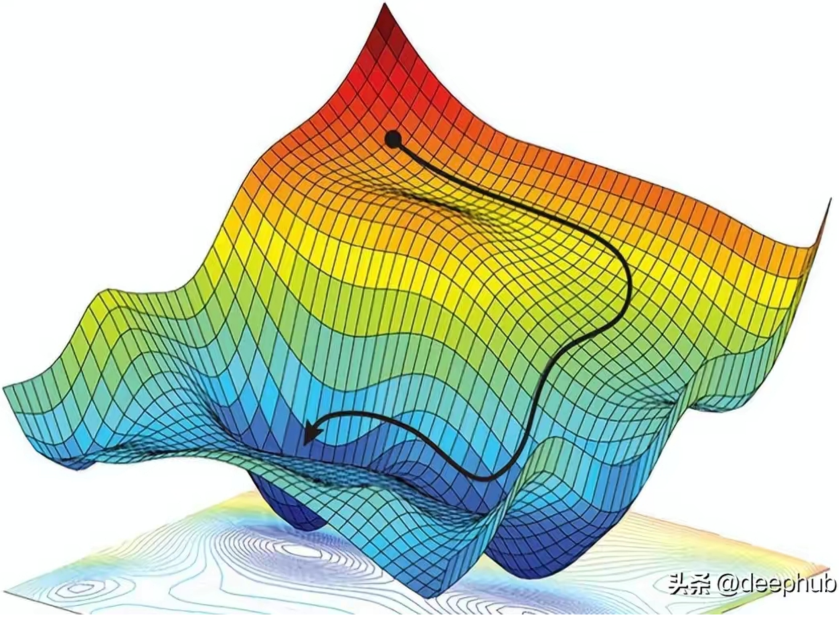
迭代法的关键是：**设计有效的规则**，**使得所产生的解序列 $\{\mathbf{x}_k\}_{k=0}^{\infty}$ 收敛于该问题的最优解之一**.

§10.4.3 迭代法

若由某种算法所产生的解序列 $\{\mathbf{x}_k\}_{k=0}^{\infty}$ 使目标函数 $f(\mathbf{x}_k)$ 逐步减少，则称它为**下降算法**．现假定已迭代到点 $\mathbf{x}_k$，若从该点出发沿任何方向移动都不能使目标函数 $f(\mathbf{x})$ 值下降，则 $\mathbf{x}_k$ 是一个**局部极小点**，迭代停止；否则，从该点出发至少存在一个方向使目标函数值有所下降，则可选择使目标函数值下降的某一方向 $\mathbf{p}_k$ 作为搜索方向，选择一个合适的步长 λ_k，得到一个新的迭代点

$$\boxed{\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{p}_k,}$$

其中称 $\mathbf{p}_k$ 为**搜索方向**，λ_k 为**步长**或**步长因子**．


头条 @deephub

§10.4.3 迭代法

下降迭代算法的计算步骤如下:

1. 选定某一初始点 $\mathbf{x}_0$, 并令 $k \leftarrow 0$;
2. **确定搜索方向** $\mathbf{p}_k$;
3. 从 $\mathbf{x}_k$ 出发, 沿搜索方向 $\mathbf{p}_k$ 求**步长** λ_k, 以产生下一个迭代点 $\mathbf{x}_{k+1}$;
4. 检查新点 $\mathbf{x}_{k+1}$ 是否为极小点或近似极小点. 若是, 则算法终止; 否则, 令 $k \leftarrow k+1$, 重复步骤 2-4.

§10.4.3 迭代法

选取搜索方向 $\mathbf{p}_k$ 是最关键的一步，各种算法的**主要差别之一在于确定搜索方向的方法不同**.

确定步长 λ_k 亦可采用不同的方法. 最简单的一种是令它等于某一常数，譬如 $\lambda_k = 1$，这样做计算简便，但不能保证使目标函数值下降；第二种称为可接收点算法，只要能使目标函数值下降，可任意选择步长 λ_k；第三种方法是沿着搜索方向使目标函数值下降最多，即求解一个子优化问题

$$\boxed{\min_{\lambda \in \mathbb{R}} f(\mathbf{x}_k + \lambda \mathbf{p}_k),}$$

称这一过程为*最优一维搜索*或*精确一维搜索*，所确定的步长为*最佳步长*.

§10.4.3 迭代法

最优一维搜索有一个重要的性质: **在搜索方向上所得到的最优点处, 目标函数的梯度和该搜索方向正交**.

定理 10.6

设目标函数 $f(\mathbf{x})$ 具有一阶连续偏导数, $\mathbf{x}_{k+1}$ 按如下规则产生:

$$\begin{cases} \lambda_k = \arg\min\limits_{\lambda\in\mathbb{R}} f(\mathbf{x}_k + \lambda \mathbf{p}_k), \\ \mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{p}_k, \end{cases}$$

则有

$$\nabla f(\mathbf{x}_{k+1})^{\mathrm{T}} \mathbf{p}_k = 0.$$

定义 10.13

设序列 $\{\mathbf{x}_k\}_{k=0}^{\infty}$ 收敛于 $\mathbf{x}^*$, 若存在常数 $c>0$, $\alpha \geqslant 1$ 及整数 $k_0>0$, 使得当 $k>k_0$ 时, 均有

$$\|\mathbf{x}_{k+1}-\mathbf{x}^*\| \leqslant c\|\mathbf{x}_k-\mathbf{x}^*\|^{\alpha}$$

成立, 则称 $\{\mathbf{x}_k\}_{k=0}^{\infty}$ 的**收敛阶**为 α, 或 $\{\mathbf{x}_k\}_{k=0}^{\infty}$ 是α **阶收敛**. 当 $\alpha=2$ 时, 称为**二阶收敛**; 当 $1<\alpha<2$ 时, 称为**超线性收敛**; 当 $\alpha=1$ 时, 称为**线性收敛**或**一阶收敛**.

通常, **线性收敛速度是比较慢**的, 二阶收敛是很快的, **超线性收敛**介于它们之间.

若一个算法具有超线性或更高的收敛速度, 一般就认为它是一个很好的算法了.

§10.4.3 迭代法

算法的**收敛准则**. 常用的有以下几种:

(1) **依据相邻两次迭代的绝对误差**

$$\|\mathbf{x}_{k+1} - \mathbf{x}_k\| < \varepsilon_1, \quad |f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k)| < \varepsilon_2;$$

(2) **依据相邻两次迭代的相对误差**

$$\frac{\|\mathbf{x}_{k+1} - \mathbf{x}_k\|}{\|\mathbf{x}_k\|} < \varepsilon_3, \quad \frac{|f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k)|}{|f(\mathbf{x}_k)|} < \varepsilon_4;$$

(3) **依据目标函数梯度的模**

$$\|\nabla f(\mathbf{x}_k)\| < \varepsilon_5,$$

其中 $\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4, \varepsilon_5$ 为事先设定的精度参数. 此外, 为了防止程序陷入无限循环, 还可设置最大迭代次数 N.

§10.5.1 数学模型

所谓**一维搜索**，又称**线性搜索**，**是指单变量函数的最优化**，它是多变量函数优化的基础.

在用迭代法求函数的极小值时，经常用到一维搜索，即**沿着某一已知方向求目标函数的极小值点**. 另外，如果优化问题中的目标函数的导数或偏导数不存在或难以计算时，也常采用搜索的方法求解.

$$\boxed{\min_{\lambda\in\mathbb{R}} f(\mathbf{x}_k+\lambda\mathbf{p}_k)}$$

§10.5.1 数学模型

所谓**一维搜索**，又称**线性搜索**，**是指单变量函数的最优化**，它是多变量函数优化的基础.

在用迭代法求函数的极小值时，经常用到一维搜索，即**沿着某一已知方向求目标函数的极小值点**. 另外，如果优化问题中的目标函数的导数或偏导数不存在或难以计算时，也常采用搜索的方法求解.

一维搜索的主要步骤：**首先确定包含问题最优解的搜索区间**；然后采用某种分割技术或插值方法**缩小这个区间**，**进行搜索求解**.

§10.5.1 数学模型

定义 10.14

设 $\varphi:\mathbb{R}\mapsto\mathbb{R}$ 是一个连续函数, $\lambda^*\in[0,+\infty)$, 且有

$$\varphi(\lambda^*)=\min_{\lambda\geqslant 0}\varphi(\lambda). \qquad \min f(\mathbf{x}_k+\lambda\mathbf{p}_k)$$

若存在闭区间 $[a,b]\subset[0,+\infty)$, 使 $\lambda^*\in[a,b]$, 则称 $[a,b]$ 是一维优化问题 $\min\limits_{\lambda\geqslant 0}\varphi(\lambda)$ 的**搜索区间**.

定义 10.14

设 $\varphi : \mathbb{R} \mapsto \mathbb{R}$ 是一个连续函数，$\lambda^* \in [0, +\infty)$，且有

$$\varphi(\lambda^*) = \min_{\lambda \geqslant 0} \varphi(\lambda). \qquad \min f(\mathbf{x}_k + \lambda \mathbf{p}_k)$$

若存在闭区间 $[a, b] \subset [0, +\infty)$，使 $\lambda^* \in [a, b]$，则称 $[a, b]$ 是一维优化问题 $\min_{\lambda \geqslant 0} \varphi(\lambda)$ 的**搜索区间**.

确定搜索区间的一种简单方法叫**进退法**，它的基本思想为：从定义域内某一点出发，按一定步长，试图确定某一区间，使得函数值呈现 "**高 - 低 - 高**" 的形状．一个方向不成功，就退回来，再沿相反方向进行搜索．

进退法的计算步骤如下:

1. 设置初始值. $\lambda_0 \in [0, +\infty)$, $h_0 > 0$, 加倍系数 $t > 1$, 常取 $t = 2$, 计算 $\varphi(\lambda_0)$, $k \leftarrow 0$;
2. 比较目标函数值. 令 $\lambda_{k+1} \leftarrow \lambda_k + h_k$, 计算 $\varphi(\lambda_{k+1})$. 若 $\varphi(\lambda_{k+1}) < \varphi(\lambda_k)$, 进入下一步; 否则, 转步骤 4;
3. 更新搜索步长. 令 $h_{k+1} \leftarrow t \times h_k$, $\lambda \leftarrow \lambda_k$, $\lambda_k \leftarrow \lambda_{k+1}$, $k \leftarrow k + 1$, 转步骤 2;
4. 反向搜索. 若 $k = 0$, 转换搜索方向, 令 $h_k \leftarrow (-h_k)$, $\lambda_k \leftarrow \lambda_{k+1}$, 转步骤 2; 否则, 停止迭代, 并令

$$a = \min\{\lambda, \lambda_{k+1}\}, \quad b = \max\{\lambda, \lambda_{k+1}\},$$

输出 $[a, b]$.

一维搜索方法主要针对单峰函数在单峰区间上的优化问题.

定义 10.15

设 $\varphi:\mathbb{R}\mapsto\mathbb{R}$ 是一个连续函数, 若存在 $[a,b]$ 及 $\lambda^*\in[a,b]$, 使 $\varphi(\lambda)$ 在 $[a,\lambda^*]$ 上严格递减, 在 $[\lambda^*,b]$ 上严格递增, 则称 $[a,b]$ 是函数 $\varphi(\lambda)$ 的**单峰区间**, $\varphi(\lambda)$ 是区间 $[a,b]$ 上的**单峰函数**.

引理 10.2

设 $\varphi:\mathbb{R}\mapsto\mathbb{R}$ 是一个连续函数, $[a,b]$ 是 $\varphi(\lambda)$ 的单峰区间, $\alpha_1,\alpha_2\in[a,b]$, 且 $\alpha_1<\alpha_2$.

(1) 若 $\varphi(\alpha_1)\leqslant\varphi(\alpha_2)$, 则 $[a,\alpha_2]$ 是 $\varphi(\lambda)$ 的单峰区间;

(2) 若 $\varphi(\alpha_1)\geqslant\varphi(\alpha_2)$, 则 $[\alpha_1,b]$ 是 $\varphi(\lambda)$ 的单峰区间.

§10.5.1 数学模型

如果 $\varphi(\lambda)$ 是 $[a,b]$ 上的单峰函数，则可通过比较 $\varphi(\lambda)$ 的函数值，来缩小搜索区间. 设 $\varphi(\lambda)=f(\mathbf{x}_k+\lambda\mathbf{p}_k)$, $\varphi(\lambda)$ 是搜索区间 $[a,b]$ 上的单峰函数. 记第 k 次迭代所产生的搜索区间为 $[a_k,b_k]$，取两个试探点 $\alpha_k,\beta_k\in[a_k,b_k]$，且 $\alpha_k<\beta_k$，计算 $\varphi(\alpha_k)$ 和 $\varphi(\beta_k)$，根据引理10.2，知

(1) 若 $\varphi(\alpha_k)\leqslant\varphi(\beta_k)$，则令 $a_{k+1}\leftarrow a_k$, $b_{k+1}\leftarrow\beta_k$;

(2) 若 $\varphi(\alpha_k)>\varphi(\beta_k)$，则令 $a_{k+1}\leftarrow\alpha_k$, $b_{k+1}\leftarrow b_k$.

反复执行上述步骤，搜索区间逐渐变小，当达到所需的精度时，算法终止.

问题：如何确定试探点 α_k,β_k 的位置.

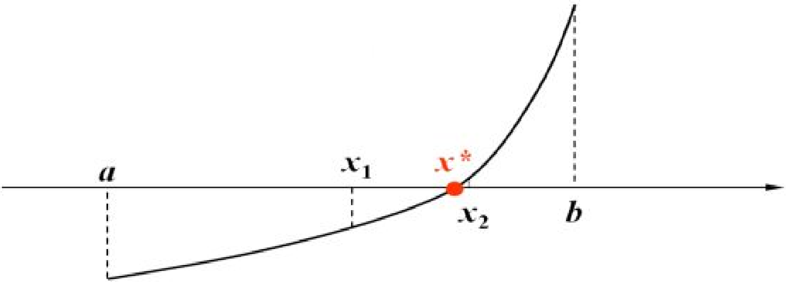
§10.5.4 二分法

$\varphi(\lambda^*) = \min \varphi(\lambda)$

二分法: 通过计算函数导数值来缩短搜索区间. 设初始搜索区间为 $\boxed{[a_0, b_0] = [a, b]}$, 第 k 步时的搜索区间为 $[a_k, b_k]$, 满足 $\varphi'(a_k) \leqslant 0$, $\varphi'(b_k) \geqslant 0$, 取中点

$$c_k = \frac{a_k + b_k}{2}.$$

若 $\varphi'(c_k) \geqslant 0$, 则令 $a_{k+1} = a_k$, $b_{k+1} = c_k$; 否则, 令 $a_{k+1} = c_k$, $b_{k+1} = b_k$, 从而得到新的搜索区间 $\boxed{[a_{k+1}, b_{k+1}]}$. 重复上述步骤, 直到搜索区间的长度小于事先设定的精度为止. 容易看出, 二分法每次迭代都将区间缩短一半, 故**二分法的收敛速度也是线性的**.



a
x_1
x^*
x_2
b

§10.5.2 黄金分割法

要求试探点 α_k, β_k 满足以下条件:

(1) α_k 和 β_k 到搜索区间 $[a_k, b_k]$ 的端点距离相等，即

$$b_k - \alpha_k = \beta_k - a_k; \tag{4}$$

(2) 每次迭代，**搜索区间长度的缩短率相等**，即

$$\boxed{b_{k+1} - a_{k+1} = \tau(b_k - a_k).} \tag{5}$$

将式 (4) 与式 (5) 联立, 可得

$$\alpha_k = a_k + (1-\tau)(b_k - a_k), \tag{6}$$

$$\beta_k = a_k + \tau(b_k - a_k). \tag{7}$$

现考虑 $\varphi(\alpha_k) \leqslant \varphi(\beta_k)$ 的情形, 则新的搜索区间为 $[a_{k+1}, b_{k+1}] = [a_k, \beta_k]$. 为进一步缩短搜索区间, 需取新的试探点 $\alpha_{k+1}, \beta_{k+1}$. 由式 (7) 知,

$$\begin{aligned}
\beta_{k+1} &= a_{k+1} + \tau(b_{k+1} - a_{k+1}) \\
&= a_k + \tau(\beta_k - a_k) \\
&= a_k + \tau[a_k + \tau(b_k - a_k) - a_k] \\
&= a_k + \tau^2(b_k - a_k).
\end{aligned}$$

若令 $\tau^2 = 1 - \tau$, 则

$$\beta_{k+1} = a_k + (1-\tau)(b_k - a_k) = \alpha_k.$$

此时, 新的试探点 β_{k+1} 不需要重新计算, 只要取 α_k 即可. 从而在每次迭代中, 仅需取一个新的试探点即可. 类似地, 考虑 $\varphi(\alpha_k) > \varphi(\beta_k)$ 的情形, 由式 (6) 知, 新的试探点 $\alpha_{k+1} = \beta_k$, 它也不需要重新计算.

由于 $\tau^2 = 1 - \tau$, 且 $\tau > 0$, 解得

$$\boxed{\tau = \frac{-1+\sqrt{5}}{2} \approx 0.618.}$$

因 0.618 等于**黄金分割率**, 故上述方法常称为***黄金分割法***或 **0.618 法**.

黄金分割法的计算步骤如下:

1. 选取初始数据. 确定初始搜索区间 $[a_0, b_0] = [a, b]$ 和精度要求 $\delta > 0$. 计算初始试探点 α_0, β_0, 并令 $k \leftarrow 0$;
2. 比较函数值. 若 $\varphi(\alpha_k) \leqslant \varphi(\beta_k)$, 进入下一步; 否则, 转步骤 4;
3. 若 $\beta_k - a_k \leqslant \delta$, 则停止计算, 输出 α_k; 否则, 令 $a_{k+1} \leftarrow a_k$, $b_{k+1} \leftarrow \beta_k$, $\beta_{k+1} \leftarrow \alpha_k$, $\varphi(\beta_{k+1}) \leftarrow \varphi(\alpha_k)$, $\alpha_{k+1} \leftarrow a_{k+1} + 0.382(b_{k+1} - a_{k+1})$, 计算 $\varphi(\alpha_{k+1})$, 转步骤 5;
4. 若 $k - \alpha_k \leqslant \delta$, 则停止计算, 输出 β_k; 否则, 令 $a_{k+1} \leftarrow \alpha_k$, $b_{k+1} \leftarrow b_k$, $\alpha_{k+1} \leftarrow \beta_k$, $\varphi(\alpha_{k+1}) \leftarrow \varphi(\beta_k)$, $\beta_{k+1} \leftarrow a_{k+1} + 0.618(b_{k+1} - a_{k+1})$, 计算 $\varphi(\beta_{k+1})$, 进入下一步;
5. 令 $k \leftarrow k + 1$, 转步骤 2.

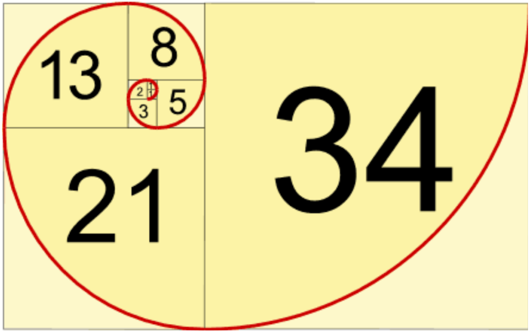
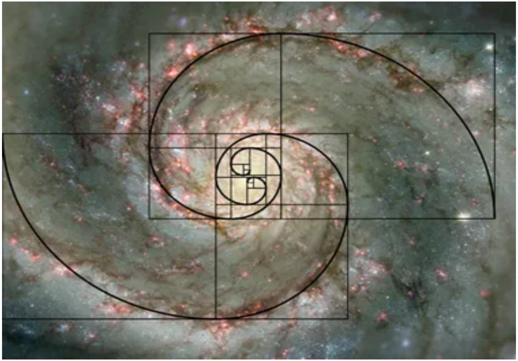
每次迭代后, 搜索区间的缩短率为 τ. 若搜索初始区间为 $[a, b]$, 则经过 n 迭代后, 搜索区间的长度为 $\tau^n(b - a)$, 算法是收敛的, 且收敛速度是线性的.

斐波那契数列 $\{F_k\}$，　即

$$F_0 = F_1 = 1,$$

$$F_{k+1} = F_k + F_{k-1}, \quad k = 1, 2, \cdots .$$


种子的排列
松果上的螺旋线条，顺时针数8条；反向再数就变成了13条. 是不是很神奇？


13
8
2
1
3
5
21
34


斐波那契法：搜索区间长度的缩短率不是采用黄金分割数，而是采用**斐波那契数列** $\{F_k\}_{k=0}^{\infty}$，即

$$\boxed{\begin{aligned} &F_0 = F_1 = 1, \\ &F_{k+1} = F_k + F_{k-1}, \quad k = 1, 2, \cdots. \end{aligned}}$$

在该方法中，试探点 α_k, β_k 的取法为

$$\alpha_k = a_k + \left(1 - \frac{F_{n-k}}{F_{n-k+1}}\right)(b_k - a_k),$$

$$\beta_k = a_k + \frac{F_{n-k}}{F_{n-k+1}}(b_k - a_k).$$

每次缩短率满足

$$\boxed{b_{k+1} - a_{k+1} = \frac{F_{n-k}}{F_{n-k+1}}(b_k - a_k),}$$

其中 n 是计算函数值的次数. 若要求经过 n 次迭代后, 所得的区间长度不超过 δ, 即 $b_n - a_n \leqslant \delta$, 那么

$$\begin{aligned} b_n - a_n &= \frac{F_1}{F_2}(b_{n-1} - a_{n-1}) \\ &= \frac{F_1}{F_2} \cdot \frac{F_2}{F_3} \cdots \frac{F_{n-1}}{F_n}(b_1 - a_1) \\ &= \frac{1}{F_n}(b_1 - a_1). \end{aligned}$$

从而有 $F_n \geqslant \dfrac{b_1 - a_1}{\delta}$.

斐波那契数列的通项公式

$$F_k = \frac{1}{\sqrt{5}}\left\{\left(\frac{1+\sqrt{5}}{2}\right)^{k+1} - \left(\frac{1-\sqrt{5}}{2}\right)^{k+1}\right\}, \quad k = 0, 1, 2, \cdots.$$

因此, 对于给定的初始搜索区间 $[a, b]$ 及精度参数 δ, 可先确定 F_n 的下界, 然后利用通项公式求出满足要求的最小 n, 再通过采用与黄金分割法类似的迭代过程, 求出最终的搜索区间.

容易看出

$$\lim_{k\to\infty}\frac{F_{k-1}}{F_k}=\frac{\sqrt{5}-1}{2}=\tau,$$

故**斐波那契法的区间缩短率与黄金分割法相同**, 也是线性收敛的. 可以证明: **斐波那契法是用分割方法求一维极小化问题的最优策略**, 而**黄金分割法是近似最优**的. 但由于后者简单易行, 因而得到了广泛的应用.

二分法: 通过计算函数导数值来缩短搜索区间. 设初始搜索区间为 $[a_0, b_0] = [a, b]$, 第 k 步时的搜索区间为 $[a_k, b_k]$, 满足 $\varphi'(a_k) \leqslant 0$, $\varphi'(b_k) \geqslant 0$, 取中点

$$c_k = \frac{a_k + b_k}{2}.$$

若 $\varphi'(c_k) \geqslant 0$, 则令 $a_{k+1} = a_k$, $b_{k+1} = c_k$; 否则, 令 $a_{k+1} = c_k$, $b_{k+1} = b_k$, 从而得到新的搜索区间 $[a_{k+1}, b_{k+1}]$. 重复上述步骤, 直到搜索区间的长度小于事先设定的精度为止. 容易看出, 二分法每次迭代都将区间缩短一半, 故**二分法的收敛速度也是线性的**.

插值法: 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

$$\varphi(\lambda^*) = \min_{\lambda \geqslant 0} \varphi(\lambda)$$

的极小点. **当函数具有较好的解析性质时, 插值法比直接方法**, 譬如黄金分割法和斐波那契法等, **效果更好**.

考虑利用一点处的函数值、一阶和二阶导数值来**构造二次插值函数**. 设二次插值多项式为

$$q(x) = ax^2 + bx + c,$$

则 $q(x)$ 在

$$x = -\frac{b}{2a}$$

处取到极小值, 故**可作为计算近似极小点的公式**.

若已知函数在 λ 处的函数值、一阶和二阶导数值, 即 $\varphi(\lambda), \varphi'(\lambda), \varphi''(\lambda)$, 则 $q(x)$ 需满足插值条件

$$
\begin{aligned}
q(\lambda) &= a\lambda^2 + b\lambda + c = \varphi(\lambda), \\
q'(\lambda) &= 2a\lambda + b = \varphi'(\lambda), \\
q''(\lambda) &= 2a = \varphi''(\lambda).
\end{aligned}
$$

解得 $a = \dfrac{\varphi''(\lambda)}{2}, \quad b = \varphi'(\lambda) - \varphi''(\lambda)\lambda$. 从而有迭代计算公式

$$\boxed{\lambda_{k+1} = \lambda_k - \frac{\varphi'(\lambda_k)}{\varphi''(\lambda_k)},} \quad k = 0, 1, \cdots ,$$

称为**一点二次插值法**, 或称为**牛顿法**.

可以证明: 当初始值 λ_0 充分靠近 λ^* 时, 由牛顿法产生的迭代序列 $\{\lambda_k\}_{k=1}^{\infty}$ 是收敛的, **且收敛速度是二阶的**(λ^* **为单根时**).

§10.5.4 插值法

牛顿法的优点是速度速度快, 但是需要用到二阶导数值, 计算代价高. 因此, 一个自然的想法是: **用两个点 λ_1, λ_2 处的函数值及其中一个点的导数值** $\varphi'(\lambda_1)$(或 $\varphi'(\lambda_2)$) 来构造二次插值函数. 通过插值条件并求解, 可得计算公式:

$$\lambda_{k+1} = \lambda_k - \frac{(\lambda_k - \lambda_{k-1})\varphi'(\lambda_k)}{2\left[\varphi'(\lambda_k) - \dfrac{\varphi(\lambda_k) - \varphi(\lambda_{k-1})}{\lambda_k - \lambda_{k-1}}\right]}, \quad k = 1, \cdots,$$

称为***两点二次插值法***, 其收敛速度的阶为 $(1+\sqrt{5})/2 \approx 1.618$.

类似地, 还有**三点二次插值法**, **二点三次插值法**等.

§10.5.5 其他方法

一维搜索过程是**非线性优化**方法的**基本组成部分**，前述通过求解一维搜索最优化问题的方法，譬如黄金分割法，牛顿法等，统称为***精确一维搜索***，需要花费很大的计算量．特别地，当迭代点远离问题的解时，精确得求解一个一维子问题通常不是十分有效．

因此，提出了***非精确一维搜索***，即选取步长 λ_k 使得目标函数 $f(\mathbf{x})$ 有可接收的**下降量** $f(\mathbf{x}_k) - f(\mathbf{x}_k + \lambda_k \mathbf{p}_k) > \delta$．**非精确一维搜索**包括 Armijo-Goldstein 方法，Wolfe-Powell 方法等．

§10.5.5 其他方法

在非线性优化中，一个可用来代替一维线搜索的方法是*信赖域方法* (Trust-Region Methods)，其**基本思想**是：先设定一个可信的步长，然后利用局部 n 维二次模型寻找最优下降方向和步长. 因步长受到使 Taylor 展开式有效的信赖域限制，故又称*限步长法*. **信赖域方法具有快速的局部收敛性**，又有理想的总体收敛性，是一个很好的方法.

不失一般性，**无约束非线性优化**问题可写成

$$\min f(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n \tag{8}$$

其中 $f(\mathbf{x}) : \mathbb{R}^n \mapsto \mathbb{R}$ 是有 n 个自变量的连续函数.

求解无约束最优化问题最简单的方法是**最速下降法**，它以负梯度方向作为极小化算法的下降方向，故又称为**梯度法**.

§10.6.1 最速下降法

设函数 $f(\mathbf{x})$ 在 $\mathbf{x}_k$ 附件连续可微，且 $\nabla f(\mathbf{x}_k) \neq 0$. 利用 Taylor 展开式

$$\boxed{f(\mathbf{x}) = f(\mathbf{x}_k) + (\mathbf{x} - \mathbf{x}_k)^{\mathrm{T}} \nabla f(\mathbf{x}_k) + o(\|\mathbf{x} - \mathbf{x}_k\|)}$$

可知，若记 $\mathbf{x} - \mathbf{x}_k = \lambda \mathbf{p}_k$, $\boxed{\mathbf{g}_k = \nabla f(\mathbf{x}_k)}$, 则满足 $\mathbf{p}_k^{\mathrm{T}} \mathbf{g}_k < 0$ 的方向 $\mathbf{p}_k$ 都是下降方向. 当 λ 的值固定后，若 $\mathbf{p}_k^{\mathrm{T}} \mathbf{g}_k$ 的值越小，即 $-\mathbf{p}_k^{\mathrm{T}} \mathbf{g}_k$ 的值越大，则函数 $f(\mathbf{x})$ 的值下降的越快. 另外，由 Cauchy-Schwartz 不等式知最优化问题

$$\min_{\mathbf{p}_k \in \mathbb{R}^n} \mathbf{p}_k^{\mathrm{T}} \mathbf{g}_k, \quad \text{subject to } \|\mathbf{p}_k\| = \|\mathbf{g}_k\|,$$

当且仅当 $\mathbf{p}_k = -\mathbf{g}_k$ 时，$\mathbf{p}_k^{\mathrm{T}} \mathbf{g}_k$ 的值最小，称 $-\mathbf{g}_k$ 为**最速下降方向**.

§10.6.1 最速下降法

最速下降法的计算步骤如下:

1. 设定初始值 $\mathbf{x}_0 \in \mathbb{R}^n$ 及精度参数 $\varepsilon > 0$, 并令 $k \leftarrow 0$;
2. **计算梯度** $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$, 令 $\mathbf{p}_k = -\mathbf{g}_k$;
3. 若 $\|\mathbf{g}_k\| < \varepsilon$, 则算法终止; 否则, 利用**一维搜索**求步长因子 λ_k, 使得

$$f(\mathbf{x}_k + \lambda_k \mathbf{p}_k) = \min_{\lambda \geqslant 0} f(\mathbf{x}_k + \lambda \mathbf{p}_k);$$

4. 计算 $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \lambda_k \mathbf{p}_k$, 令 $k \leftarrow k+1$, 转步骤 2.

§10.6.1 最速下降法

可以证明: 若 $f(\mathbf{x}) \in C^1$, 在最速下降法中采用**精确一维搜索**, 则产生的迭代点序列 $\{\mathbf{x}_k\}_{k=1}^{\infty}$ 的每一个聚点都是驻点. 进一步, 若 $\lim\limits_{k\to\infty} \mathbf{x}_k = \mathbf{x}^*$ 且 $f(\mathbf{x})$ 在 $\mathbf{x}^*$ 的某一邻域内二次连续可微, 存在 $\varepsilon > 0$ 和 $M > m > 0$, 使得当 $\|\mathbf{x} - \mathbf{x}^*\| < \varepsilon$ 时, 有

$$m\|\mathbf{y}\|^2 \leqslant \mathbf{y}^{\mathrm{T}} G(\mathbf{x})\mathbf{y} \leqslant M\|\mathbf{y}\|^2, \quad \forall \mathbf{y} \in \mathbb{R}^n,$$

其中 $G(\mathbf{x}) = \nabla^2 f(\mathbf{x})$, 则**最速下降法至少是线性收敛的**.

§10.6.1 最速下降法

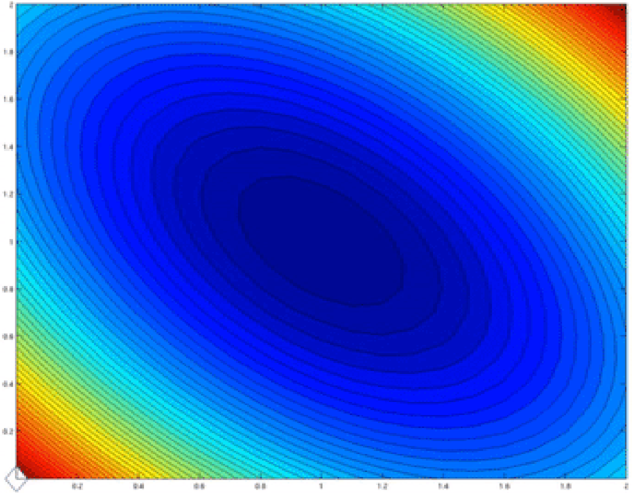
由于最速下降方向是函数的局部性质，对许多实际问题，最速下降法并非 “**最速下降**”，而是下降非常缓慢.

数值实验表明，当目标函数的等值线或面接近于一个圆或球时，最速下降法下降较快；而当目标函数的等值线或面接近于一个扁长的椭圆或椭球时，最速下降法开始几步下降较快，后来就出现锯齿现象，下降十分缓慢. 事实上，因采用一维精确搜索，由定理10.6知

$$\mathbf{g}_{k+1}^{\mathrm{T}}\mathbf{p}_k = \mathbf{p}_{k+1}^{\mathrm{T}}\mathbf{p}_k = 0,$$

即在**相邻的两个迭代点上，两个搜索方向是相互正交的**，这便是产生锯齿状现象的原因. 当接近极小点时，步长越小，前进越慢.

$$f(x) = \frac{1}{2}x^T \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} x - \begin{pmatrix} 1.5 & 1.5 \end{pmatrix} x, \quad x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$$



§10.6.2 牛顿法

最速下降法仅用到了函数 $f(\mathbf{x})$ 的一阶局部信息. 若目标函数 $f(\mathbf{x})$ 是二次连续可微的, 则在 $\mathbf{x}_k$ 附件可用 $f(\mathbf{x})$ 的二次 Taylor 展开式

$$q_k(\mathbf{h}) = f(\mathbf{x}_k) + \nabla f(\mathbf{x}_k)^{\mathrm{T}}\mathbf{h} + \frac{1}{2}\mathbf{h}^{\mathrm{T}}\nabla^2 f(\mathbf{x}_k)\mathbf{h}$$

来近似 $f(\mathbf{x})$, 其中 $\mathbf{h} = \mathbf{x} - \mathbf{x}_k$. 若矩阵 $\nabla^2 f(\mathbf{x}_k)$ 正定, 则 n 维二次函数 $q_k(\mathbf{x})$ 在

$$\frac{\partial q_k(\mathbf{h})}{\partial \mathbf{h}} = \mathbf{0} \Leftrightarrow \nabla^2 f(\mathbf{x}_k)\mathbf{h} = -\nabla f(\mathbf{x}_k) \Leftrightarrow \mathbf{h} = -\left[\nabla^2 f(\mathbf{x}_k)\right]^{-1}\nabla f(\mathbf{x}_k)$$

处取到极小值. 因此, 可设计迭代格式

$$\boxed{\mathbf{x}_{k+1} = \mathbf{x}_k - \left[\nabla^2 f(\mathbf{x}_k)\right]^{-1}\nabla f(\mathbf{x}_k),} \quad k = 0, 1, \cdots,$$

称为**牛顿迭代法**.

§10.6.2 牛顿法

牛顿迭代法是在椭球范数下的最速下降法. 事实上, 对于局部近似函数 $f(\mathbf{x}_k+\mathbf{h}) \approx f(\mathbf{x}_k)+\mathbf{g}_k^{\mathrm{T}}\mathbf{h}$, 考虑极小化问题

$$\min_{\mathbf{h}\in\mathbb{R}^n} \frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{h}}{\|\mathbf{h}\|}$$

的最优解, 其中 $\|\cdot\|$ 是某种向量范数. 当采用 l_2 范数时, 即 $\|\mathbf{x}\|^2=\mathbf{x}^{\mathrm{T}}\mathbf{x}$, 可得

$$\mathbf{h}=-\mathbf{g}_k,$$

所得的方法是最速下降法. 当采用椭球范数时, 即 $\|\mathbf{x}\|^2=\mathbf{x}^{\mathrm{T}}G_k\mathbf{x}$, 可得

$$\mathbf{h}=-G_k^{-1}\mathbf{g}_k,$$

所得的方法是牛顿迭代法.

§10.6.2 牛顿法

当 $f(\mathbf{x})$ 是正定二次函数时, $q_k(\mathbf{x}) = f(\mathbf{x})$, 故牛顿迭代法一步就可达最优解. 而对于非二次函数, 牛顿迭代法不能保证经过有限次迭代求得最优解, 但由于目标函数在极小点附件可近似于二次函数, 故当初始点靠近极小点时, 牛顿迭代法的收敛速度一般是快的.

可以证明: 若 $f(\mathbf{x}) \in C^2$, $\mathbf{x}_k$ 充分靠近 $\mathbf{x}^*$, 如果 $\nabla^2 f(\mathbf{x}^*)$ 正定, 且 $G(x) = \nabla^2 f(\mathbf{x})$ 的每一个元素的函数满足 Lipschitz 条件, 则牛顿迭代法所得的序列 $\{\mathbf{x}_k\}_{k=1}^{\infty}$ 收敛于 $\mathbf{x}^*$, 且具有二阶的收敛速度.

§10.6.2 牛顿法

当初始点远离最优解时, G_k 不一定正定, 牛顿方向不一定是下降方向, 收敛性不能保证, 这说明恒取步长为 1 的牛顿迭代法是不合适的. 因此, 应该引入某种一维搜索来确定步长因子, 这种方法称为**带步长因子的牛顿法**, 计算步骤如下:

1. 设定初始值 $\mathbf{x}_0 \in \mathbb{R}^n$ 及精度参数 $\varepsilon > 0$, 并令 $k \leftarrow 0$;
2. 计算梯度 $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$. 若 $\|\mathbf{g}_k\| < \varepsilon$, 则算法终止; 否则, 进入下一步;
3. 计算矩阵 G_k, 并求解线性方程组 $G_k\mathbf{p} = -\mathbf{g}_k$ 得牛顿方向 $\mathbf{p}_k$;
4. 利用一维搜索求步长因子 λ_k, 使得
$$f(\mathbf{x}_k + \lambda_k\mathbf{p}_k) = \min_{\lambda \geqslant 0} f(\mathbf{x}_k + \lambda\mathbf{p}_k);$$
5. 计算 $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \lambda_k\mathbf{p}_k$, 令 $k \leftarrow k+1$, 转步骤 2.

§10.6.3 共轭方向法

共轭方向法：一种**介于最速下降法与牛顿法之间的方法**，它仅需利用一阶导数信息，但克服了最速下降法收敛慢的缺点，又避免了存储和计算牛顿法所需要的二阶导数信息.

共轭方向法是从研究二次函数的极小化问题产生的，是一种迭代求解大型稀疏正定线性方程组的有效方法，同时它还可以推广到处理非二次函数的极小化问题.

简介 共轭方向法和共轭梯度法

共轭梯度法是介于最速下降法与牛顿法之间的一个方法，它仅需利用一阶导数信息，但克服了最速下降法收敛慢的缺点，又避免了牛顿法需要存储和计算Hesse矩阵并求逆的缺点，共轭梯度法不仅是解决大型线性方程组最有用的方法之一，也是解大型非线性最优化最有效的算法之一.

(1) 最初是由计算数学家Hestenes和几何学家Stiefel于1952年为求正定系数矩阵线性方程组而独立提出的.他们合作的著名文章*Method of conjugate gradients for solving linear systems* 被认为是共轭梯度法的奠基性文章。

(2) 1964年，Fletcher和Reeves将此方法推广到非线性最优化，得到了求解一般函数极小值的共轭梯度法.

(3) 共轭梯度法的收敛性分析的早期工作主要由Fletcher、Powell、Beale等学者给出.

(4) **Nocedal、Gilbert、Nazareth、Al-Baali、Storey、Dai、Yuan和Han等学者在收敛性方面得到了不少新成果.**

§10.6.3 共轭方向法

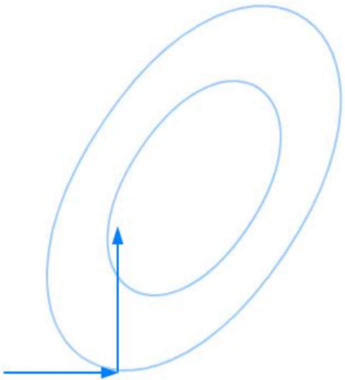
定义 10.16

设 G 是 n 阶正定方阵, $\mathbf{p}_1$, $\mathbf{p}_2$ 是 n 维非零向量, 如果 $\mathbf{p}_1^{\mathrm{T}}G\mathbf{p}_2=0$, 则称 $\mathbf{p}_1$ 与 $\mathbf{p}_2$ 是*G-共轭的*. 类似地, 设 $\mathbf{p}_1,\mathbf{p}_2,\cdots,\mathbf{p}_n$ 是一组 n 维非零向量, 如果

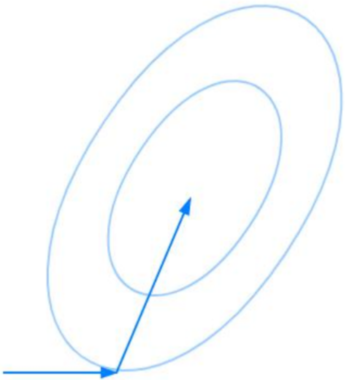
$$\mathbf{p}_i^{\mathrm{T}}G\mathbf{p}_j=0,\quad \forall 1\leqslant i\neq j\leqslant n,$$

则称 $\mathbf{p}_1,\mathbf{p}_2,\cdots,\mathbf{p}_n$ 是 G-共轭的.

显然, 如果 $\mathbf{p}_1,\mathbf{p}_2,\cdots,\mathbf{p}_n$ 是 G-共轭的, 则它们是线性无关的. 当 $G=I$ 时, G-共轭性就是通常的正交性.



正交



共轭

§10.6.3 共轭方向法

一般共轭方向法的计算步骤如下:

1. 设定初始值 $\mathbf{x}_0 \in \mathbb{R}^n$ 及初始方向 $\mathbf{g}_0$, 计算 $\mathbf{p}_0$, 使得 $\mathbf{p}_0^{\mathrm{T}}\mathbf{g}_0 < 0$, 并令 $k \leftarrow 0$;
2. 计算 λ_k 和 $\mathbf{x}_{k+1}$, 使得

$$f(\mathbf{x}_k + \lambda_k \mathbf{p}_k) = \min_{\lambda \geqslant 0} f(\mathbf{x}_k + \lambda \mathbf{p}_k),$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k \mathbf{p}_k;$$

3. 计算 $\mathbf{p}_{k+1}$, 使得 $\mathbf{p}_{k+1}^{\mathrm{T}} G \mathbf{p}_j = 0$, 其中 $j = 0, 1, 2, \cdots, k$;
4. 令 $k \leftarrow k+1$, 转步骤 2.

§10.6.3 共轭方向法

共轭方向法: 只要执行精确一维搜索, 就具有 n 次终止性.

定理 10.7

(共轭方向基本定理) 设 $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathrm{T}}G\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{x} + c$ 是正定二次函数, 其中 G 是 n 阶正定方阵, $\mathbf{b}$是 n 维向量, c 是常数, 则共轭方向法至多经过 n 步精确线性搜索终止; 且每一个 $\mathbf{x}_{k+1}$ 都是 $f(\mathbf{x})$ 在点 $\mathbf{x}_0$ 和方向 $\mathbf{p}_0, \mathbf{p}_1, \cdots, \mathbf{p}_k$ 所张成的线性流形

$$\left\{\mathbf{x} \,\middle|\, \mathbf{x} = \mathbf{x}_0 + \sum_{j=0}^{k} \lambda_j \mathbf{p}_j, \quad \forall \lambda_j \in \mathbb{R}\right\}$$ 中的极小点.

在精确线性搜索的条件下, 利用 $\mathbf{g}_{k+1} - \mathbf{g}_k = G(\mathbf{x}_{k+1} - \mathbf{x}_k) = \lambda_k G\mathbf{p}_k$ 可知, 共轭方向法的梯度 $\mathbf{g}_{k+1}$ 满足

$$\mathbf{g}_{k+1}^{\mathrm{T}}\mathbf{p}_j = 0, \quad j = 0, 1, \cdots, k.$$

§10.6.3 共轭方向法

若记 $\mathbf{x}^* = \mathbf{x}_0 + \sum_{i=0}^{n-1} y_i^* \mathbf{p}_i$, $\mathbf{x} = \mathbf{x}_0 + \sum_{i=0}^{n-1} y_i \mathbf{p}_i$, 则二次函数 $f(\mathbf{x})$ 可改写成

$$q(\mathbf{y}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}^*)^{\mathrm{T}} G(\mathbf{x}-\mathbf{x}^*) + \tilde{c} = \frac{1}{2}(\mathbf{y}-\mathbf{y}^*)^{\mathrm{T}} S^{\mathrm{T}} G S(\mathbf{y}-\mathbf{y}^*) + \tilde{c},$$

其中 $S = (\mathbf{p}_0, \mathbf{p}_1, \cdots, \mathbf{p}_{n-1})$. 利用 G-共轭性条件, 得

$$q(\mathbf{y}) = \frac{1}{2}\sum_{i=0}^{n-1}(y_i - y_i^*)^2 d_{ii},$$

其中 $d_{ii} = \mathbf{p}_i^{\mathrm{T}} G \mathbf{p}_i$.

共轭性意味着存在一个恰当的坐标变换 S, 使得 G 在新的坐标系下是一个对角矩阵 $S^T G S$, 新的变量在二次函数中是相互分离的. 于是, 一个共轭方向法是在新坐标系中的一个交替变量法.

§10.6.3 共轭方向法

共轭梯度法是最著名的共轭方向法，它使得最速下降方向具有共轭性，从而提高算法的有效性和可靠性.

由于解正定线性方程组 $A\mathbf{x} = \mathbf{b}$ **等价**于极小化一个正定二次函数

$$\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\mathbf{x}^{\mathrm{T}}A\mathbf{x} - \mathbf{b}^{\mathrm{T}}\mathbf{x},$$

共轭梯度法首先是作为解线性方程组的方法被提出来的.

§10.6.3 共轭方向法

考虑正定二次函数

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

$f(\mathbf{x})$ 的梯度为 $\nabla f(\mathbf{x}) = G\mathbf{x} + \mathbf{b}$. 共轭梯度法有一个非常特殊的性质：在构造共轭方向时，$\mathbf{p}_k$ 仅依赖于 $\mathbf{p}_{k-1}$. 这意味着在实现算法时，可以减少存储量和计算时间，是一个很好的性质. 另外，在搜索方向 $\mathbf{p}_k$ 上，希望函数值是下降的，故一个自然的选择是

$$\boxed{\mathbf{p}_k = -\mathbf{g}_k + \beta_{k-1}\mathbf{p}_{k-1},}$$

其中 $\mathbf{g}_k = \nabla f(\mathbf{x_k}) = G\mathbf{x_k} + \mathbf{b}$, β_{k-1} 是待定系数.

§10.6.3 共轭方向法

因 $\mathbf{p}_k$ 与 $\mathbf{p}_{k-1}$ 需满足 G-共轭条件，上式两端左乘以 $\mathbf{p}_{k-1}^{\mathrm{T}}G$，故有

$$\beta_{k-1} = \frac{\mathbf{g}_k^{\mathrm{T}} G \mathbf{p}_{k-1}}{\mathbf{p}_{k-1}^{\mathrm{T}} G \mathbf{p}_{k-1}}, \quad k = 1, 2, \cdots, n-1.$$

对于 $\mathbf{p}_0$，直接取 $\mathbf{p}_0 = -\mathbf{g}_0$. 可以证明 $\mathbf{p}_0, \mathbf{p}_1, \cdots, \mathbf{p}_{n-1}$ 是 G-共轭的，且每一次迭代所产生的搜索方向 $\mathbf{p}_k$ 和梯度方向 $\mathbf{g}_k$ 都包含于

$$\mathcal{K}(\mathbf{g}_0; k) \triangleq \operatorname{span}\left\{\mathbf{g}_0, G\mathbf{g}_0, \cdots, G^k \mathbf{g}_0\right\},$$

称 $\mathcal{K}(\mathbf{g}_0; k)$ 为 $\mathbf{g}_0$ 的 k 次 Krylov 子空间.

§10.6.3 共轭方向法

定理 10.8

设 $f(\mathbf{x})=\frac{1}{2}\mathbf{x}^{\mathrm{T}}G\mathbf{x}+\mathbf{b}^{\mathrm{T}}\mathbf{x}+c$ 是正定二次函数，其中 G 是 n 阶正定方阵，$\mathbf{b}$ 是 n 维向量，c 是常数，则采用精确线性搜索的共轭梯度法经过 $m\leqslant n$ 步终止，且对任意的 $k\leqslant m$ 有

$$\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_i=0,\quad i=0,1,\cdots,k-1, \tag{9}$$

$$\mathbf{p}_k^{\mathrm{T}}G\mathbf{p}_i=0,\quad i=0,1,\cdots,k-1, \tag{10}$$

$$\mathbf{p}_i^{\mathrm{T}}\mathbf{g}_i=-\mathbf{g}_i^{\mathrm{T}}\mathbf{g}_i,\quad i=0,1,\cdots,k-1, \tag{11}$$

$$\mathrm{span}\left\{\mathbf{g}_0,\mathbf{g}_1,\cdots,\mathbf{g}_k\right\}=\mathcal{K}(\mathbf{g}_0;k), \tag{12}$$

$$\mathrm{span}\left\{\mathbf{p}_0,\mathbf{p}_1,\cdots,\mathbf{p}_k\right\}=\mathcal{K}(\mathbf{g}_0;k). \tag{13}$$

注意：$\mathbf{p}_0$ 必须取 $-\mathbf{g}_0$，否则上述定理是不成立的.

§10.6.3 共轭方向法

式 (9) 表明梯度向量序列 $\{\mathbf{g}_i\}$ 是相互正交的，而式 (10) 表明搜索方向序列 $\{\mathbf{p}_i\}$ 才是 G-共轭的. 由式 (11) 知，共轭梯度法的每一步都是值下降的，精确一维搜索的步长 λ_k 可写成:

$$\lambda_k = -\frac{\mathbf{p}_k^{\mathrm{T}}\mathbf{g}_k}{\mathbf{p}_k^{\mathrm{T}}G\mathbf{p}_k} = \frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_k}{\mathbf{p}_k^{\mathrm{T}}G\mathbf{p}_k}.$$

§10.6.3 共轭方向法

利用 $G\mathbf{p}_k = (\mathbf{g}_{k+1} - \mathbf{g}_k)/\lambda_k$ 及式 (9) 和 (11), 可得计算搜索方向的步长 β_k 几个常用公式

$$
\begin{aligned}
\beta_k &= \frac{\mathbf{g}_{k+1}^{\mathrm{T}}(\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{p}_k^{\mathrm{T}}(\mathbf{g}_{k+1} - \mathbf{g}_k)} \ \text{(Crowder-Wolfe 公式)} \\
&= \frac{\mathbf{g}_{k+1}^{\mathrm{T}}\mathbf{g}_{k+1}}{\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_k} \ \text{(Fletcher-Reeves 公式)} \\
&= \frac{\mathbf{g}_{k+1}^{\mathrm{T}}(\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_k} \ \text{(Polak-Ribiere-Polyak 公式)} \\
&= -\frac{\mathbf{g}_{k+1}^{\mathrm{T}}\mathbf{g}_{k+1}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{g}_k} \ \text{(Dixon 公式)}.
\end{aligned}
$$

共轭梯度法仅比最速下降法稍微复杂一点, 但却具有二次终止性, 所以共轭梯度法是一个很有效的方法.

§10.6.3 共轭方向法

共轭梯度法可推广至一般的目标函数 $f(\mathbf{x})$，此时，步长 λ_k 可通过精确一维搜索或非精确一维搜索得到，而 β_k 的计算公式保持不变．但是，需要注意的是，n 步以后共轭梯度法所产生的搜索方向 $\mathbf{p}_n$ 不再与 $\mathbf{p}_0, \mathbf{p}_1, \cdots, \mathbf{p}_{n-1}$ 满足 G-共轭条件，故需重新取最速下降方向作为搜索方向，即

$$\mathbf{p}_{cn} = -\mathbf{g}_{cn}, \quad c = 1, 2, \cdots,$$

这种方法称为***再开始共轭梯度法***．

§10.6.3 共轭方向法

再开始共轭梯度法的计算步骤如下:

1. 设定初始值 $\mathbf{x}_0 \in \mathbb{R}^n$ 及精度参数 $\varepsilon > 0$, 计算 $\mathbf{g}_0 = \nabla f(\mathbf{x}_0)$, 令 $k \leftarrow 0$;
2. 若 $\|\mathbf{g}_0\| < \varepsilon$, 则算法终止; 否则, 令 $\mathbf{p}_0 \leftarrow -\mathbf{g}_0$;
3. 利用一维搜索求步长因子 λ_k, 使得
$$f(\mathbf{x}_k + \lambda_k \mathbf{p}_k) = \min_{\lambda \geqslant 0} f(\mathbf{x}_k + \lambda \mathbf{p}_k),$$
令 $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \lambda_k \mathbf{p}_k$, 计算 $\mathbf{g}_{k+1} = \nabla f(\mathbf{x}_{k+1})$;
4. 若 $\|\mathbf{g}_{k+1}\| < \varepsilon$, 则算法终止; 否则, 进入下一步;
5. 若 $k = n-1$, 令 $\mathbf{x}_0 \leftarrow \mathbf{x}_{k+1}$, 转步骤 1; 否则, 进入下一步;
6. 计算 $\beta_k = \mathbf{g}_{k+1}^{\mathrm{T}} \mathbf{g}_{k+1} / \mathbf{g}_k^{\mathrm{T}} \mathbf{g}_k$, 令 $\mathbf{p}_{k+1} \leftarrow -\mathbf{g}_{k+1} + \beta_k \mathbf{p}_k$, $k \leftarrow k+1$;
7. 若 $\mathbf{p}_k^{\mathrm{T}} \mathbf{g}_k > 0$, 令 $\mathbf{x}_0 \leftarrow \mathbf{x}_k$, 转步骤 1; 否则, 转步骤 3.

§10.6.3 共轭方向法

当步长因子 λ_k 也可以采用非精确一维搜索时, β_k 的几种计算公式的效果是不同的.

共轭梯度法在一定条件下是收敛的. 可以证明: 若 $f(\mathbf{x})$ 在有界水平集 $L=\{\mathbf{x}\in\mathbb{R}^n\,|\,f(\mathbf{x})\leqslant f(\mathbf{x}_0)\}$ 上连续可微, 则采用 Fletcher-Reeves 公式和精确一维搜索的共轭梯度法产生的序列 $\{\mathbf{x}_k\}$ 至少有一个聚点是驻点, 即

(1) 当 $\{\mathbf{x}_k\}$ 是有穷点列时, 其最后一个点 $\mathbf{x}^*$ 是 $f(\mathbf{x})$ 的驻点;

(2) 当 $\{\mathbf{x}_k\}$ 是无穷点列时, 它必有极限点, 且其任一极限点都是 $f(\mathbf{x})$ 的驻点.

采用其他公式和搜索策略也有类似的结论.

§10.6.3 共轭方向法

当 $f(\mathbf{x})$ 是正定二次函时，采用精确一维搜索的共轭梯度法至多在 n 次迭代后终止．而对于一般的非线性函数 $f(\mathbf{x})$，若将 n 次迭代视为一次大的迭代，则共轭梯度法应该与牛顿法有类似的收敛速度．可以证明：设 $f(\mathbf{x}) \in C^3$，且存在常数 $m, M > 0$，使得

$$m\|\mathbf{y}\|^2 \leqslant \mathbf{y}^{\mathrm{T}}\nabla^2 f(\mathbf{x})\mathbf{y} \leqslant M\|\mathbf{y}\|^2, \quad \forall \mathbf{y} \in \mathbb{R}^n, \mathbf{x} \in L,$$

其中 $L = \{\mathbf{x} \in \mathbb{R}^n \,|\, f(\mathbf{x}) \leqslant f(\mathbf{x}_0)\}$ 是有界水平集，则采用 Fletcher-Reeves 公式和精确一维搜索的再开始共轭梯度法产生的序列 $\{\mathbf{x}_k\}$ 是 n 步二阶收敛的，即存在常数 $c > 0$，使得

$$\lim_{k\to\infty} \sup \frac{\|\mathbf{x}_{kn+n} - \mathbf{x}^*\|}{\|\mathbf{x}_{kn} - \mathbf{x}^*\|^2} \leqslant c < \infty.$$

§10.6.4 拟牛顿法

拟牛顿法: 利用目标函数 $f(\mathbf{x})$ 及一阶导数 $\nabla f(\mathbf{x})$ 的信息来构造近似的 Hessian 矩阵, 从而实现加快收敛速度的目标.

设 $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ 是二次连续可微的函数, 则 $f(\mathbf{x})$ 在 $\mathbf{x}_{k+1}$ 附件的二次近似函数为

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

其中 $\mathbf{g}_{k+1} = \nabla f(\mathbf{x}_{k+1})$, $G_{k+1} = \nabla^2 f(\mathbf{x}_{k+1})$, 从而有

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

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

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

显然, 当 $f(\mathbf{x})$ 是二次函数时, 上述近似关系取等号.

§10.6.4 拟牛顿法

在拟牛顿法中，要求构造出来的 Hessian 矩阵的逆近似 H_{k+1} 满足这种关系，即

$$H_{k+1}\mathbf{y}_k = \mathbf{s}_k, \tag{14}$$

称为*拟牛顿条件*或*拟牛顿方程*. 若记 $B_{k+1} = H_{k+1}^{-1}$，则上述条件也可写为

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

是关于 Hessian 矩阵的拟牛顿条件.

§10.6.4 拟牛顿法

一般拟牛顿法的计算步骤如下:

1. 设定初始值 $\mathbf{x}_0 \in \mathbb{R}^n$ 及精度参数 $\varepsilon > 0$, $H_0 \in \mathbb{R}^{n\times n}$, 并令 $k \leftarrow 0$;
2. 计算梯度 $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$, 若 $\|\mathbf{g}_k\| < \varepsilon$, 则算法终止; 否则, 计算 $\mathbf{p}_k = -H_k\mathbf{g}_k$;
3. 利用一维搜索求步长因子 λ_k, 使得
$$f(\mathbf{x}_k + \lambda_k\mathbf{p}_k) = \min_{\lambda\geqslant 0} f(\mathbf{x}_k + \lambda\mathbf{p}_k);$$
4. 校正 H_k 产生 H_{k+1}, 使得拟牛顿条件 (14) 成立;
5. 计算 $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \lambda_k\mathbf{p}_k$, 令 $k \leftarrow k+1$, 转步骤 2.

类似地, 拟牛顿法也可采用近似 Hessian 矩阵 B_k 进行. 在实际应用中, 初始矩阵 H_0 可取单位阵, 此时, 拟牛顿法的第一次迭代等价于一个最速下降迭代.

§10.6.4 拟牛顿法

拟牛顿法是在椭球范数 $\|\cdot\|_{H_k^{-1}}$ 下的最速下降法，搜索方向

$$\mathbf{p}_k = -H_k\mathbf{g}_k$$

是 $f(\mathbf{x})$ 从 $\mathbf{x}_k$ 点出发的最速下降方向. 因在每一次迭代中，度量矩阵 H_k^{-1} 都是变化的，故方法也称为*变尺度法*.

与牛顿法相比，拟牛顿法有以下优点:

(1) 仅需一阶导数信息;
(2) H_k 保持正定，使得方法具有下降性质;
(3) 每次迭代需要 $O(n^2)$ 次乘法运算.

§10.6.4 拟牛顿法

如何校正 H_k 产生 H_{k+1}, 使得拟牛顿条件 (14) 成立, 并保持正定性. 设 H_{k+1} 是 H_k 通过对称秩二校正得到的, 即

$$H_{k+1} = H_k + a\mathbf{u}\mathbf{u}^{\mathrm{T}} + b\mathbf{v}\mathbf{v}^{\mathrm{T}},$$

其中 $a, b \in \mathbb{R}$, $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{n\times 1}$. 代入拟牛顿条件 (14) 得

$$H_k\mathbf{y}_k + a\mathbf{u}\mathbf{u}^{\mathrm{T}}\mathbf{y}_k + b\mathbf{v}\mathbf{v}^{\mathrm{T}}\mathbf{y}_k = \mathbf{s}_k.$$

显然, 向量 $\mathbf{u}$ 和 $\mathbf{v}$ 不能由上式唯一确定, 但是 $\mathbf{u}$ 和 $\mathbf{v}$ 可取为

$$\mathbf{u} = \mathbf{s}_k, \quad \mathbf{v} = H_k\mathbf{y}_k,$$

这时, 只要 a 和 b 满足 $a\mathbf{u}^{\mathrm{T}}\mathbf{y}_k = 1$, $b\mathbf{v}^{\mathrm{T}}\mathbf{y}_k = -1$, 拟牛顿条件 (14) 就成立了. 解得

$$H_{k+1} = H_k + \frac{\mathbf{s}_k\mathbf{s}_k^{\mathrm{T}}}{\mathbf{s}_k^{\mathrm{T}}\mathbf{y}_k} - \frac{H_k\mathbf{y}_k\mathbf{y}_k^{\mathrm{T}}H_k}{\mathbf{y}_k^{\mathrm{T}}H_k\mathbf{y}_k}, \tag{15}$$

称为 **DFP 校正公式**, 是由 Davidon, Fletcher 和 Powell 发展出来的.

§10.6.4 拟牛顿法

DFP 校正公式具有很多重要的性质:

(1) 校正保持正定性, 故下降性质成立;
(2) 每次迭代需要 $3n^2+O(n)$ 次乘法运算;
(3) 具有超线性的收敛速度;
(4) 对于凸函数, 当采用精确一维搜索时, 方法具有总体收敛性.

§10.6.4 拟牛顿法

利用 $H_{k+1} \longleftrightarrow B_{k+1}$, $\mathbf{s}_k \longleftrightarrow \mathbf{y}_k$ 之间的对偶关系及矩阵逆的秩一校正公式, 可得

$$H_{k+1} = H_k + \left(1 + \frac{\mathbf{y}_k^{\mathrm{T}} H_k \mathbf{y}_k}{\mathbf{s}_k^{\mathrm{T}} \mathbf{y}_k}\right) \frac{\mathbf{s}_k \mathbf{s}_k^{\mathrm{T}}}{\mathbf{s}_k^{\mathrm{T}} \mathbf{y}_k} - \frac{\mathbf{s}_k \mathbf{y}_k^{\mathrm{T}} H_k + H_k \mathbf{y}_k \mathbf{s}_k^{\mathrm{T}}}{\mathbf{s}_k^{\mathrm{T}} \mathbf{y}_k}, \quad (16)$$

称为 **BFGS 校正公式**, 是由 Broyden, Fletcher, Goldfarb 和 Shanno 发展出来的. BFGS 校正公式具备与 DFP 校正公式相同的性质, 并且对于凸函数, 当采用非精确一维搜索时, 仍具有总体收敛性. 在实际应用中, BFGS 校正公式的结果也优于 DFP 校正公式, 是最常用的拟牛顿方法之一.

§10.6.4 拟牛顿法

一般地, 当初始点靠近极小点时, 拟牛顿迭代法在一定条件下是线性收敛的, 很多时候甚至是超线性收敛的.

对于凸函数, 拟牛顿法还有整体收敛性. 可以证明: 当 $f(\mathbf{x})$ 是一致凸的二阶连续可微函数时, 采用精确一维搜索和 DFP 校正公式的拟牛顿法整体收敛; 当 $f(\mathbf{x})$ 是凸的二阶连续可微函数时, 采用非精确一维搜索的 Wolfe-Powell 准则和 BFGS 校正公式的拟牛顿法整体收敛.

§10.6.5 随机梯度下降法

在许多实际应用中，譬如深度学习，压缩感知，低秩矩阵填充等，优化变量的个数可能会很大，如几十万甚至更多，这时即使采用梯度下降法，每一步迭代所需的计算量也是相当可观的．因此，人们提出了**随机梯度下降法**，基本思想是：梯度向量可以视为期望，而期望可以使用小规模的样本来近似估计.设优化的目标函数为 $f(\mathbf{x})$，优化变量 $\mathbf{x}=(x_1,x_2,\cdots,x_n)^{\mathrm{T}}$，先任取它的一个有 m 个元素的子集 $\{x_{i1},x_{i2},\cdots,x_{im}\}$，$m$ 远小于 n，然后计算 $f(\mathbf{x})$ 关于变量 $x_{i1},x_{i2},\cdots,x_{im}$ 的梯度并乘以一个称为**学习率**的常数 r，其他变量视为常数，执行一次梯度下降．重复以上步骤，直至算法满足收敛准则.