第十二章 优化算法

第 33 讲 无约束优化算法：线搜索和一阶优化算法

黄定江

DaSE @ ECNU

djhuang@dase.ecnu.edu.cn

1. 33.1 无约束优化算法概述
2. 33.2 精确和非精确线搜索方法
3. 33.3 一阶优化算法

本讲和下一讲讨论无约束优化问题

$$\min \ f(\boldsymbol{x}) \tag{1}$$

的求解算法，其中 $f\colon \mathbb{R}^n \to \mathbb{R}$ 的可微函数。

- 根据前一章对无约束优化问题最优性条件的讨论可知，对于可微的优化问题，可使得目标函数 $f(\boldsymbol{x})$ 的梯度等于零，求得平稳点；然后用充分条件进行判别，便可求出所要的最优解。
- 然而，对于一般的 n 元函数 $f(\boldsymbol{x})$ 来说，通常 $\nabla f(\boldsymbol{x}) = 0$ 是一个非线性方程组，求它的解析解相当困难。
- 对于不可微优化问题，更无法使用这样的方法。
- 为此，常使用**迭代法**进行求解。

33.1.1 迭代法

迭代法的基本思想是：

- 首先给定一个**初始估计**$\boldsymbol{x}^{(0)}$；
- 然后按某种规则（即**算法**）找出比 $\boldsymbol{x}^{(0)}$ 更好的解 $\boldsymbol{x}^{(1)}$(对极小化问题，$f(\boldsymbol{x}^{(1)}) < f(\boldsymbol{x}^{(0)})$；对极大化问题，$f(\boldsymbol{x}^{(1)}) > f(\boldsymbol{x}^{(0)})$)；
- 再按此种规则找出比 $\boldsymbol{x}^{(1)}$ 更好的解 $\boldsymbol{x}^{(2)}, \cdots$；

如此，即可得到一个解的序列 $\{\boldsymbol{x}^{(k)}\}$。若这个解序列收敛于最优解 $\boldsymbol{x}^*$，即

$$\lim_{k\to\infty} \|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\| = 0$$

则该算法就是求解优化问题的一种迭代方法。

注：由于计算机只能进行有限次迭代，一般只能得到近似解。当满足所要求的精度时，即可停止迭代。

迭代法收敛速度

评价一个算法的好坏，不仅要考虑它产生的点列是否能收敛到问题的最优解，而且还要考虑它的**收敛速度**。设 $\boldsymbol{x}^{(k)}$ 为算法产生的迭代点列且收敛于 $\boldsymbol{x}^*$，对充分大的 k，

- 若满足有

$$\frac{\left\|\boldsymbol{x}^{(k+1)}-\boldsymbol{x}^*\right\|}{\left\|\boldsymbol{x}^{(k)}-\boldsymbol{x}^*\right\|}\leq a, a\in(0,1), \tag{2}$$

  则称算法是 **Q-线性收敛**的；

- 若满足

$$\lim_{k\to\infty}\frac{\left\|\boldsymbol{x}^{(k+1)}-\boldsymbol{x}^*\right\|}{\left\|\boldsymbol{x}^{(k)}-\boldsymbol{x}^*\right\|}=0, \tag{3}$$

  则称算法是 **Q-超线性收敛**的；

- 若满足

$$\lim_{k\to\infty}\frac{\left\|\boldsymbol{x}^{(k+1)}-\boldsymbol{x}^*\right\|}{\left\|\boldsymbol{x}^{(k)}-\boldsymbol{x}^*\right\|}=1, \tag{4}$$

则称算法是 **Q-次线性收敛**的；

- 若满足

$$\frac{\left\|\boldsymbol{x}^{(k+1)}-\boldsymbol{x}^*\right\|}{\left\|\boldsymbol{x}^{(k)}-\boldsymbol{x}^*\right\|^2}\leq a, a>0, \tag{5}$$

则称算法是 **Q-二次收敛**的。

一般地，若一个算法具有超线性或更高的收敛速度，就认为它是一个很好的算法。

迭代法基本步骤

一般地，最优化的迭代方法是下降算法，具有如下求解结构：

1. 选定某一初始点 $\boldsymbol{x}^{(0)}$，并令 $k := 0$；
2. 依据一定规则，确定**搜索方向**$\boldsymbol{p}^{(k)}$（通常的是下降方向）；
3. 从 $\boldsymbol{x}^{(k)}$ 出发，沿方向 $\boldsymbol{p}^{(k)}$ 求**步长**（步长因子）λ_k，以产生下一个迭代点 $\boldsymbol{x}^{(k+1)}$；
4. 检查得到的新点 $\boldsymbol{x}^{(k+1)}$ 是否为极小点或近似极小点。若是，则停止迭代。否则，令 $k := k + 1$，转回第二步继续进行迭代。

具体迭代过程，可参见下图1。

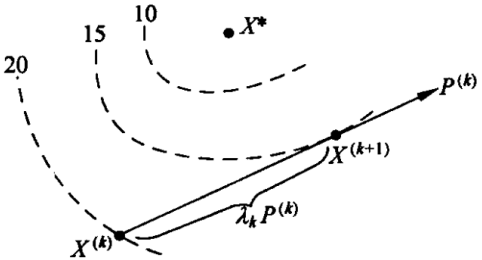

图 1

在以上步骤中，存在两个**关键问题**：

- 如何确定搜索方向 $\boldsymbol{p}^{(k)}$
- 如何确定步长 λ_k

下面分别进行论述。

33.1.2 关于确定步长

我们结合无约束优化问题的线搜索类优化算法来进行论述。线搜索类算法的数学表述为：

- 给定当前迭代点 $\boldsymbol{x}^{(k)}$，首先通过某种算法选取向量 $\boldsymbol{p}^{(k)}$，之后确定正数 λ_k，则下一步的迭代点可写作

$$\boldsymbol{x} = \boldsymbol{x}^{(k)} + \lambda_k \boldsymbol{p}^{(k)},$$

这里 $\boldsymbol{p}^{(k)}$ 为迭代点 $\boldsymbol{x}^{(k)}$ 处的搜索方向，λ_k 为相应的步长。通常要求 $\boldsymbol{p}^{(k)}$ 是一个下降方向，即 $(\boldsymbol{p}^{(k)})^T \nabla f(\boldsymbol{x}^{(k)}) < 0$。这个下降性质保证了沿着此方向搜索函数 f 的值会减小。线搜索类算法的关键是如何选取一个好的方向 $\boldsymbol{p}^{(k)} \in \mathbb{R}^n$ 以及合适的步长 λ_k。

- 确定步长 λ_k 的过程，实际上是解决在确定搜索方向之后，在该方向走多远的问题？通常，选取 $\boldsymbol{p}^{(k)}$ 的方法千差万别，但选取 λ_k 的方法在不同的算法中非常相似：把目标函数 $f(\boldsymbol{x})$ 限制在射线 $\{\boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)} | \lambda > 0\}$ 上得到一个关于 λ 的一元函数 $f(\boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)})$。线搜索的目标是选取合适的 λ_k，使得 $f(\boldsymbol{x}^{(k)} + \lambda_k \boldsymbol{p}^{(k)})$ 尽可能减小。

关于确定步长

但这一工作并不容易：λ_k 应该使得 f 充分下降，与此同时不应该在寻找 λ_k 上花费过多的计算量。我们需要权衡这两个方面，由此产生两类方法：

- 精确线搜索，即沿射线 $\boldsymbol{x} = \boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)}$ 求使得目标函数 $f(\boldsymbol{x})$ 的极小的 λ_k，换言之

$$\lambda_k = \arg\min_{\lambda} \ f(\boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)})$$

  这样确定的步长为**最佳步长**。

- 非精确线搜索，不要求 λ_k 是 $f(\boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)})$ 的极小点，只要步长 λ_k 能使目标函数值下降充分即可。

33.1.3 关于搜索方向

根据利用目标函数的信息不同，确定搜索方向的方法也有差异。我们将其分为如下两类：

- **一阶方法**：其本质是仅仅使用目标函数的一阶导数（梯度）信息选取下降方向 $\boldsymbol{p}^{(k)}$。该类方法也称为梯度类算法，适用于不需要很高精度的大数据优化问题，例如：机器学习、深度学习；
- **二阶方法**：其本质是利用目标函数的二阶导数（Hessian 矩阵）信息来构造下降方向 $\boldsymbol{p}^{(k)}$。该类方法包括牛顿法、拟牛顿法等，适用于需要高精度的优化问题，例如：科学计算。

我们可以得到如下优化算法总览图：

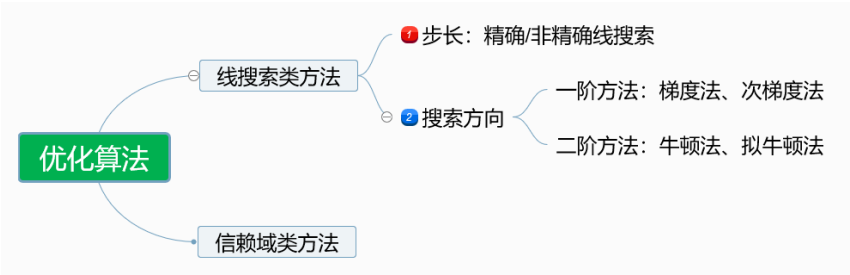

图 2

注：限于篇幅，本课程主要介绍线搜索类方法，这在传统机器学习和深度学习的模型优化中也更为常用。

33.2.1 精确线搜索

前面已提及线搜索分为精确和非精确线搜索，我们先介绍精确线搜索，其本质上是求如下优化问题：

$$\lambda_k = \arg\min_{\lambda} f(\boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)}).$$

这样得到的最优解具有很好的性质：某种程度上，下一个迭代点的函数值在该方向上已经不能够再下降（达到最优）。如下定理所述。

定理 1

设目标函数 $f(\boldsymbol{x})$ 具有一阶连续偏导数，$\boldsymbol{x}^{(k+1)}$ 按照下述规则产生

$$\begin{cases} \lambda_k: \min\limits_{\lambda} f(\boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)}) \\ \boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \lambda_k \boldsymbol{p}^{(k)} \end{cases}$$

则有

$$\nabla f(\boldsymbol{x}^{(k+1)})^T \boldsymbol{p}^{(k)} = 0 \tag{6}$$

证明.

构造函数 $\varphi(\lambda) = f(\boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)})$，则得

$$\begin{cases} \varphi(\lambda_k) = \min\limits_{\lambda} \varphi(\lambda) \\ \boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \lambda_k \boldsymbol{p}^{(k)} \end{cases}$$

即 λ_k 为 $\varphi(\lambda)$ 的极小点。此外 $\varphi'(\lambda) = \nabla f(\boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)})^T \boldsymbol{p}^{(k)}$。

由 $\varphi'(\lambda)|_{\lambda=\lambda_k} = 0$，可得

$$\nabla f(\boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)})^T \boldsymbol{p}^{(k)} = \nabla f(\boldsymbol{x}^{(k+1)})^T \boldsymbol{p}^{(k)} = 0$$

□

式(6) 的几何意义见下图3。

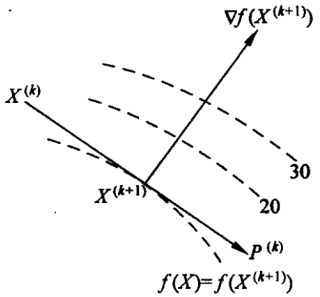

图 3: 精确线搜索的几何意义

图中 $\boldsymbol{p}^{(k)}$ 就是前一迭代点 $\boldsymbol{x}^{(k)}$ 的下降方向，当使用精确线搜索，达到 $k+1$ 个迭代点 $\boldsymbol{x}^{(k+1)}$ 时，必然与该点的梯度方向垂直。若不然，梯度方向与搜索方向为钝角，则意味着函数值还可以继续下降，这与最小值矛盾。

精确线搜索方法包括：**斐波那契法**和 **0.618 法**等。

实际上，由于线搜索本质上是单变量函数的最优化问题。因此，一维优化方法均可用于此，包括：

- 试探法（“成功 -失败”法，斐波那契法，0.618 法等）；
- 插值法（抛物线插值法，三次插值法等）；
- 微积分中的求根法（切线法，二分法等）。

需要指出的是，尽管使用精确线搜索算法时我们可以在多数情况下得到优化问题的解，但这样选取的步长通常需要很大计算量，在实际应用中较少使用。因此上述方法我们不详细展开。

33.2.2 非精确线搜索

下面我们考虑非精确线搜索，它不要求步长是最小值点，而是仅仅要求它是满足某些不等式性质的近似解．由于非精确线搜索算法结构简单，在实际应用中较为常见。

在非精确线搜索算法中，选取 λ_k 需要满足一定的要求，这些要求被称为线搜索准则。一般，若选取不合适的线搜索准则将会导致算法无法收敛。为便于理解这一点，我们给出一个例子。

例 1

考虑一维无约束优化问题

$$\min_x f(x) = x^2,$$

迭代初始点 $x^0 = 1$. 由于问题是一维的, 下降方向只有 $\{-1, +1\}$ 两种. 我们选取 $d^k = -\operatorname{sign}\left(x^k\right)$, 且只要求选取的步长满足迭代点处函数值单调下降, 即 $f\left(x^k + \lambda_k d^k\right) < f\left(x^k\right)$. 考虑选取如下两种步长:

$$\lambda_{k,1} = \frac{1}{3^{k+1}}, \quad \lambda_{k,2} = 1 + \frac{2}{3^{k+1}},$$

（续）通过简单计算可以得到

$$x_1^k = \frac{1}{2}\left(1 + \frac{1}{3^k}\right), \quad x_2^k = \frac{(-1)^k}{2}\left(1 + \frac{1}{3^k}\right).$$

显然, 序列 $\left\{f\left(x_1^k\right)\right\}$ 和序列 $\left\{f\left(x_2^k\right)\right\}$ 均单调下降, 但序列 $\left\{x_1^k\right\}$ 收敛的点不是极小值点, 序列 $\left\{x_2^k\right\}$ 则在原点左右振荡, 不存在极限.

出现上述情况的原因是在迭代过程中函数值 $f\left(x^k\right)$ 的下降量不够充分, 以至于算法无法收敛到极小值点. 为了避免这种情况发生, 必须引入一些更合理的线搜索准则来确保迭代的收敛性。

33.2.3 Armijo 准则

首先引入 Armijo 准则, 它是一个常用的线搜索准则. 引入 Armijo 准则的目的是保证每一步迭代充分下降.

定义 1

(Armijo 准则) 设 $\boldsymbol{d}^k$ 是点 $\boldsymbol{x}^k$ 处的下降方向, 若

$$f\left(\boldsymbol{x}^k+\lambda \boldsymbol{d}^k\right) \leqslant f\left(\boldsymbol{x}^k\right)+c_1 \lambda \nabla f\left(\boldsymbol{x}^k\right)^{\mathrm{T}} \boldsymbol{d}^k,$$

则称步长 λ 满足 *Armijo* 准则, 其中 $c_1 \in (0,1)$ 是一个常数.

Armijo 准则有非常直观的几何含义, 它指的是点 $(\lambda, \phi(\lambda))$ 必须在直线

$$l(\lambda) = \phi(0) + c_1 \lambda \nabla f\left(\boldsymbol{x}^k\right)^{\mathrm{T}} \boldsymbol{d}^k$$

的下方. 如图 4 所示, 区间 $[0, \lambda_1]$ 中的点均满足 Armijo 准则.

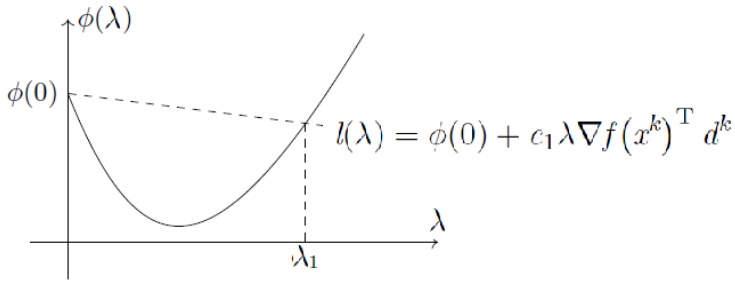

图 4

- 我们注意到 $\boldsymbol{d}^k$ 为下降方向, 这说明 $l(\lambda)$ 的斜率为负, 选取符合 Armijo 准则的 λ 确实会使得函数值下降.
- 在实际应用中, 参数 c_1 通常选为一个很小的正数, 例如 $c_1 = 10^{-3}$, 这使得 Armijo 准则非常容易得到满足. 但是仅仅使用 Armijo 准则并不能保证迭代的收敛性,
- 这是因为 $\lambda = 0$ 显然满足条件, 而这意味着迭代序列中的点固定不变, 研究这样的步长是没有意义的. 为此, Armijo 准则需要配合其他准则共同使用.

33.2.4 Wolfe 准则

为了克服 Armijo 准则的缺陷，我们需要引入其他准则来保证每一步的步长不会太小. 为此我们引入 Armijo-Wolfe 准则, 简称 Wolfe 准则.

定义 2

（*Wolfe* 准则）设 $\boldsymbol{d}^k$ 是点 $\boldsymbol{x}^k$ 处的下降方向, 若

$$f\left(\boldsymbol{x}^k+\lambda \boldsymbol{d}^k\right) \leqslant f\left(\boldsymbol{x}^k\right)+c_1 \lambda \nabla f\left(\boldsymbol{x}^k\right)^{\mathrm{T}} \boldsymbol{d}^k,$$

$$\nabla f\left(\boldsymbol{x}^k+\lambda \boldsymbol{d}^k\right)^{\mathrm{T}} \boldsymbol{d}^k \geqslant c_2 \nabla f\left(\boldsymbol{x}^k\right)^{\mathrm{T}} \boldsymbol{d}^k,$$

则称步长 λ 满足 *Wolfe* 准则, 其中 $c_1, c_2 \in (0,1)$ 为给定的常数且 $c_1 < c_2$.

在 Wolfe 准则中, 第一个不等式即是 Armijo 准则, 而第二个不等式则是 Wolfe 准则的本质要求. 注意到 $\nabla f\left(\boldsymbol{x}^k+\lambda \boldsymbol{d}^k\right)^{\mathrm{T}} \boldsymbol{d}^k$ 恰好就是 $\phi(\lambda)$ 的导数, Wolfe 准则实际要求 $\phi(\lambda)$ 在点 λ 处切线的斜率不能小于 $\phi'(0)$ 的 c_2 倍. 如图5 所示,

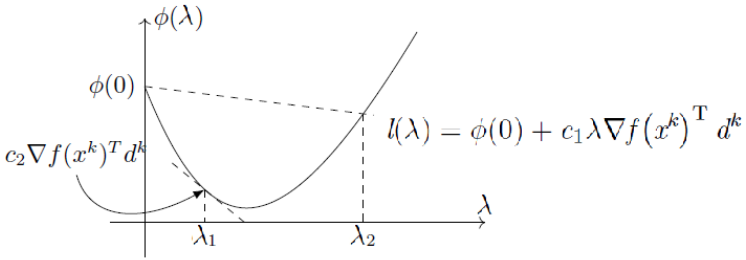

图 5

- 在区间 $[\lambda_1, \lambda_2]$ 中的点均满足 Wolfe 准则. 注意到在 $\phi(\lambda)$ 的极小值点 λ^* 处有 $\phi'(\lambda^*) = \nabla f\left(\boldsymbol{x}^k + \lambda^* \boldsymbol{d}^k\right)^{\mathrm{T}} \boldsymbol{d}^k = 0$, 因此 λ^* 永远满足第二个不等式.
- 而选择较小的 c_1 可使得 λ^* 同时满足第一个不等式条件, 即 Wolfe 准则在绝大多数情况下会包含线搜索子问题的精确解. 在实际应用中, 参数 c_2 通常取为 0.9。

33.2.5 非精确线搜索算法

在优化算法的实现中, 寻找一个满足 Armijo 准则的步长是比较容易的, 一个最常用的算法是**回退法**. 给定初值 $\hat{\lambda}$, 回退法通过不断以指数方式缩小试探步长, 找到第一个满足 Armijo 准则的点.

回退法的基本过程如下所示：

Algorithm 1 回退法

1: 选择初始步长 $\hat{\lambda}$, 参数 $\gamma, c \in (0,1)$. 初始化 $\lambda \leftarrow \hat{\lambda}$
2: **while** $f\left(\boldsymbol{x}^k + \lambda \boldsymbol{d}^k\right) > f\left(\boldsymbol{x}^k\right) + c\lambda \nabla f\left(\boldsymbol{x}^k\right)^{\mathrm{T}} \boldsymbol{d}^k$ **do**
3: $\quad \lambda \leftarrow \gamma\lambda$.
4: **end while**
5: 输出 $\lambda_k = \lambda$.

具体来说, 回退法选取

$$\lambda_k = \gamma^{j_0}\hat{\lambda},$$

其中

$$j_0 = \min\left\{j = 0, 1, \cdots \mid f\left(\boldsymbol{x}^k + \gamma^j\hat{\lambda}\boldsymbol{d}^k\right) \leqslant f\left(\boldsymbol{x}^k\right) + c_1\gamma^j\hat{\lambda}\nabla f\left(\boldsymbol{x}^k\right)^{\mathrm{T}}\boldsymbol{d}^k\right\}$$

参数 $\gamma \in (0, 1)$ 为一个给定的实数.

注:

- 该算法被称为回退法是因为 λ 的试验值是由大至小的, 它可以确保输出的 λ_k 能尽量地大.
- 此外该算法不会无限进行下去, 因为 $\boldsymbol{d}^k$ 是一个下降方向，当 λ 充分小时，Armijo 准则总是成立的.

回退法的实现简单、原理直观，所以它是最常用的线搜索算法之一．然而，回退法的缺点也很明显：

- 第一，回退法以指数的方式缩小步长，因此对初值 $\hat{\lambda}$ 和参数 γ 的选取比较敏感，当 γ 过大时每一步试探步长改变量很小，此时回退法效率比较低，当 γ 过小时回退法过于激进，导致最终找到的步长太小，错过了选取大步长的机会．为了提高回退法的效率，还有其他类型基于多项式插值的线搜索算法．
- 第二，它无法保证找到满足 Wolfe 准则的步长，但对一些优化算法而言，找到满足 Wolfe 准则的步长是十分必要的。为此，Fletcher 提出了一个用于寻找满足 Wolfe 准则的算法．这个算法比较复杂，有较多细节，这里不展开阐述。

上一小节，我们讨论了确定步长的相关方法。本小节开始考虑确定搜索方向的问题。

- 我们期望的搜索方向应当能够保证函数值在局部范围内下降，而且在局部范围内尽可能“最优”。
- 根据对目标函数的近似程度不同，这些方法可分为一阶方法（梯度类方法）和二阶方法（牛顿类方法）。

本小节，我们将一起探讨被机器学习领域广泛应用的一阶方法（梯度类方法）。

33.3.1 梯度下降法

假定无约束优化问题中的目标函数 $f(\boldsymbol{x})$ 有一阶连续偏导数，具有极小点 $\boldsymbol{x}^*$。设 $\boldsymbol{x}^{(k)}$ 表示极小点的第 k 次近似，为了探讨当前的搜索方向应满足的性质，不妨设此时方向为 $\boldsymbol{p}^k$。那么第 $k+1$ 次近似点 $\boldsymbol{x}^{(k+1)}$ 可以表示为

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)} \quad (\lambda \geq 0)$$

其中 λ 为步长。

我们期望函数值下降：$f(\boldsymbol{x}^{(k+1)}) < f(\boldsymbol{x}^{(k)})$，即 $f(\boldsymbol{x}^{(k)} + \lambda \boldsymbol{p}^{(k)}) < f(\boldsymbol{x}^{(k)})$。

在小范围内，$f(\boldsymbol{x}^{(k)}+\lambda \boldsymbol{p}^{(k)})$ 可以由 $\boldsymbol{x}^{(k)}$ 点处的泰勒级数很好地近似，即：

$$f(\boldsymbol{x}^{(k)}+\lambda \boldsymbol{p}^{(k)})=f(\boldsymbol{x}^{(k)})+\lambda \nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{p}^{(k)}+\boldsymbol{o}(\lambda)$$

其中

$$\lim_{\lambda \to 0^+} \frac{\boldsymbol{o}(\lambda)}{\lambda}=0$$

因此，对于充分小的 λ，只要

$$\nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{p}^{(k)}<0 \tag{7}$$

即可保证 $f(\boldsymbol{x}^{(k+1)})=f(\boldsymbol{x}^{k}+\lambda \boldsymbol{p}^{(k)})<f(\boldsymbol{x}^{(k)})$。

但是满足（7）的方向 $\boldsymbol{p}^{(k)}$ 太多，该如何选择？

为了使目标函数值能得到尽量大的改善，必须寻求使 $\nabla f(\boldsymbol{x}^{(k)})^T\boldsymbol{p}^{(k)}$ 取最小值的 $\boldsymbol{p}^{(k)}$。由线性代数学知道

$$\nabla f(\boldsymbol{x}^{(k)})^T\boldsymbol{p}^{(k)} = \|\nabla f(\boldsymbol{x}^{(k)})\| \cdot \|\boldsymbol{p}^{(k)}\| \cos\theta \tag{8}$$

式中 θ 为向量 $\nabla f(\boldsymbol{x}^{(k)})$ 与 $\boldsymbol{p}^{(k)}$ 的夹角。

当 $\boldsymbol{p}^{(k)}$ 与 $\nabla f(\boldsymbol{x}^{(k)})$ 反向时，$\theta = 180°, \cos\theta = -1$。这时式(7) 成立，而且其左端取最小值。我们称方向

$$\boldsymbol{p}^{(k)} = -\nabla f(\boldsymbol{x}^{(k)})$$

为**负梯度方向**，它是使函数值下降最快的方向（在 $\boldsymbol{x}^{(k)}$ 的某一小范围内）。这便得到梯度下降法。

梯度下降法具体步骤

现将用**梯度下降法**解无约束优化问题的步骤简要总结如下：

Algorithm 2 梯度下降法

1: 给定初始近似点 $\boldsymbol{x}^{(0)}$ 及精度 $\varepsilon > 0$，计算 $\nabla f(\boldsymbol{x}^{(0)})$。

2: 若 $\|\nabla f(\boldsymbol{x}^{(0)})\|^2 \leq \varepsilon$，则 $f(\boldsymbol{x}^{(0)})$ 即为近似极小点；

3: 若 $\|\nabla f(\boldsymbol{x}^{(0)})\|^2 > \varepsilon$，利用线搜索确定步长 λ_0，并计算

$$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} - \lambda_0 \nabla f(\boldsymbol{x}^{(0)}).$$

4: 一般地，设已迭代到点 $\boldsymbol{x}^{(k)}$，计算 $\nabla f(\boldsymbol{x}^{(k)})$。若 $\|\nabla f(\boldsymbol{x}^{(k)})\|^2 \leq \varepsilon$，则 $\boldsymbol{x}^{(k)}$ 即为所求的近似解；若 $\|\nabla f(\boldsymbol{x}^{(k)})\|^2 > \varepsilon$，则求步长 λ_k，并确定下一个近似点

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \lambda_k \nabla f(\boldsymbol{x}^{(k)})$$

如此继续，直至达到要求的精度为止。

在选定搜索方向之后，还要确定步长 λ。可能采取如下方法：

- 有时可以采取合适的固定步长的方式；
- 也可采用可接受点算法，就是取某一 λ 进行试算，看是否满足不等式

$$f(\boldsymbol{x}^{(k)} - \lambda \nabla f(\boldsymbol{x}^{(k)})) < f(\boldsymbol{x}^{(k)}) \tag{9}$$

  若上述不等式成立，就可以迭代下去。否则，缩小 λ 使满足不等式(9)。
- 还可以采取精确线搜索的方式计算出最优的步长。

若步长是通过精确线搜索得到的，此时的梯度法就是所谓的**最速下降法**。

例 2

试用梯度法求

$$f(\boldsymbol{x}) = (x_1 - 1)^2 + (x_2 - 1)^2$$

的极小点，已知 $\varepsilon = 0.1$。

解

取初始点 $\boldsymbol{x}^{(0)} = (0, 0)^T$

$$\nabla f(\boldsymbol{x}) = [2(x_1 - 1), 2(x_2 - 1)]^T$$

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

$$\|\nabla f(\boldsymbol{x}^{(0)})\|^2 = (-2)^2 + (-2)^2 = 8 > \varepsilon$$

令 $\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} - \lambda_0 \nabla f(\boldsymbol{x}^{(0)}) = \begin{pmatrix} 2\lambda_0 \\ 2\lambda_0 \end{pmatrix}$，代入 $f(x)$，可得：

$$f(\boldsymbol{x}^{(1)}) = (2\lambda_0 - 1)^2 + (2\lambda_0 - 1)^2$$

要使得上式最小，令 $\mathrm{d}f(\boldsymbol{x}^{(1)})/\mathrm{d}\lambda_0 = 0$，可得

$$\lambda_0 = \frac{1}{2}$$

因此，

$$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} - \lambda_0 \nabla f(\boldsymbol{x}^{(0)}) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$\nabla f(\boldsymbol{x}^{(1)}) = [2(1-1), 2(1-1)]^T = (0, 0)^T$$

故 $\boldsymbol{x}^{(1)}$ 即为极小点。

有时还可采用**固定步长**求解，甚至可以得到如下收敛定理。

定理 2

设函数 $f(x)$ 为凸的梯度 L-利普希茨连续函数, $f^* = f(x^*) = \inf_x f(x)$ 存在且可达。如果步长 λ_k 取为常数 λ 且满足 $0 < \lambda \leq \frac{1}{L}$，那么由梯度下降法得到的点列 $\left\{x^{(k)}\right\}$ 的函数值收敛到最优值，且在函数值的意义下收敛速度为 $\mathcal{O}\left(\frac{1}{k}\right)$。

证明.

因为函数 f 是利普希茨可微函数, 对任意的 x, 根据梯度 L- 利普希茨连续的性质：

$$f(x - \lambda \nabla f(x)) \leqslant f(x) - \lambda \left(1 - \frac{L\lambda}{2}\right) \|\nabla f(x)\|^2.$$

现在记 $\tilde{x} = x - \lambda \nabla f(x)$，我们有

$$\begin{aligned} f(\tilde{x}) \leqslant f(x) - \frac{\lambda}{2}\|\nabla f(x)\|^2 &\leqslant f^* + \nabla f(x)^T (x - x^*) - \frac{\lambda}{2}\|\nabla f(x)\|^2 \text{（凸性）} \\ &= f^* + \frac{1}{2\lambda}\left(\|x - x^*\|^2 - \|x - x^* - \lambda \nabla f(x)\|^2\right) \\ &= f^* + \frac{1}{2\lambda}\left(\|x - x^*\|^2 - \|\tilde{x} - x^*\|^2\right). \end{aligned}$$

证明（续）.

在上式中取 $x = x^{(i-1)}, \tilde{x} = x^{(i)}$ 并将不等式对 $i = 1, 2, \cdots, k$ 求和得到

$$\begin{aligned}\sum_{i=1}^{k}\left(f\left(x^{(i)}\right) - f^*\right) &\leqslant \frac{1}{2\lambda}\sum_{i=1}^{k}\left(\left\|x^{(i-1)} - x^*\right\|^2 - \left\|x^{(i)} - x^*\right\|^2\right)\\ &= \frac{1}{2\lambda}\left(\left\|x^{(0)} - x^*\right\|^2 - \left\|x^{(k)} - x^*\right\|^2\right)\\ &\leqslant \frac{1}{2\lambda}\left\|x^{(0)} - x^*\right\|^2.\end{aligned}$$

易知 $f\left(x^{(i)}\right)$ 是非增的, 所以

$$f\left(x^{(k)}\right) - f^* \leqslant \frac{1}{k}\sum_{i=1}^{k}\left(f\left(x^{(i)}\right) - f^*\right) \leqslant \frac{1}{2k\lambda}\left\|x^{(0)} - x^*\right\|^2.$$

例 3

试求 $f(\boldsymbol{x}) = x_1^2 + 25x_2^2$ 的极小点。

解

取初始点 $\boldsymbol{x}^{(0)} = (2, 2)^T$，固定步长 $\lambda = 0.01$，其迭代过程如表1 所示。

步骤	点	x_1	x_2	$\frac{\partial f(\boldsymbol{x}^{(k)})}{\partial x_1}$	$\frac{\partial f(\boldsymbol{x}^{(k)})}{\partial x_2}$	$\|\nabla f(\boldsymbol{x}^{(k)})\|$
0	$\boldsymbol{x}^{(0)}$	*2*	*2*	*4*	*100*	*100.08*
1	$\boldsymbol{x}^{(1)}$	*1.96*	*1.00*	*3.92*	*50*	*50.15*
2	$\boldsymbol{x}^{(2)}$	*1.92*	*0.50*	*3.84*	*25*	*25.29*
3	$\boldsymbol{x}^{(3)}$	*1.88*	*0.25*	*3.76*	*12.5*	*13.06*
...	...	...	...	...	...	...
200	$\boldsymbol{x}^{(200)}$	3.45×10^{-2}	6.22×10^{-61}	6.89×10^{-2}	3.11×10^{-59}	*0.07*

Table 1

通过这个例子，我们可以观察到：

- 通过迭代在开头几步，目标函数值下降较快，但接近极小点 $\boldsymbol{x}^*$ 时，收敛速度就不理想了。特别是当目标函数的等值线椭圆比较扁平时，收敛速度就更慢了。
- 因此，在实用中，常将梯度法和其它方法（后面介绍的二阶方法）联合起来应用。在前期使用梯度法，而在接近极小点时，则使用收敛较快的其它方法。

应用：梯度法求解 LASSO 问题

LASSO 问题的形式为

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2 + \mu\|\boldsymbol{x}\|_1$$

目标函数 $f(\boldsymbol{x})$ 中 $\mu\|\boldsymbol{x}\|_1$ 不光滑，在某些点处无法求出梯度，因此不能直接对原始问题使用梯度法求解。

考虑到目标函数的不光滑项为 $\|\boldsymbol{x}\|_1$，它实际上是 $\boldsymbol{x}$ 各个分量绝对值的和。因此，在实际应用中，人们常考虑利用如下一维光滑函数近似：

$$l_\delta(x) = \begin{cases} \frac{1}{2\delta}x^2, & |x| < \delta, \\ |x| - \frac{\delta}{2}, & \text{其他}. \end{cases}$$

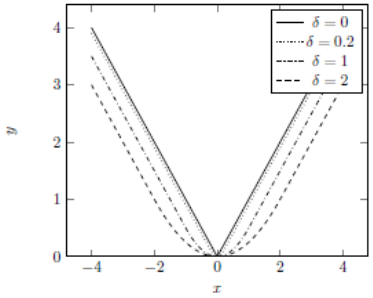

图 6: 绝对值函数的光滑近似

易知，当 $\delta \to 0$ 时, 光滑函数 $l_\delta(x)$ 和绝对值函数 $|x|$ 会越来越接近。

这样便可构造光滑化的 LASSO 问题为

$$\min_{\boldsymbol{x}} f_\delta(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2 + \mu L_\delta(\boldsymbol{x}),$$

其中 δ 为给定的光滑化参数，以及

$$L_\delta(\boldsymbol{x}) = \sum_{i=1}^{n} l_\delta\left(x_i\right).$$

这时容易计算出 $f_\delta(\boldsymbol{x})$ 的梯度为

$$\nabla f_\delta(\boldsymbol{x}) = \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) + \mu \nabla L_\delta(\boldsymbol{x}),$$

其中 $\nabla L_\delta(\boldsymbol{x})$ 是逐个分量定义的:

$$\left(\nabla L_\delta(\boldsymbol{x})\right)_i = \begin{cases} \operatorname{sign}\left(x_i\right), & |x_i| > \delta, \\ \frac{x_i}{\delta}, & |x_i| \leqslant \delta. \end{cases}$$

现在我们讨论步长的问题，显然 $f_\delta(\boldsymbol{x})$ 的梯度是 L-利普希茨连续的，且相应常数为 $L = \left\|\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right\|_2 + \frac{\mu}{\delta}$. 根据定理2，若采用固定步长则需满足 $0 < \lambda \leq \frac{1}{L}$ 才能保证算法收敛。如果 δ 过小, 那么我们需要选取充分小的步长 λ 使得梯度法收敛。

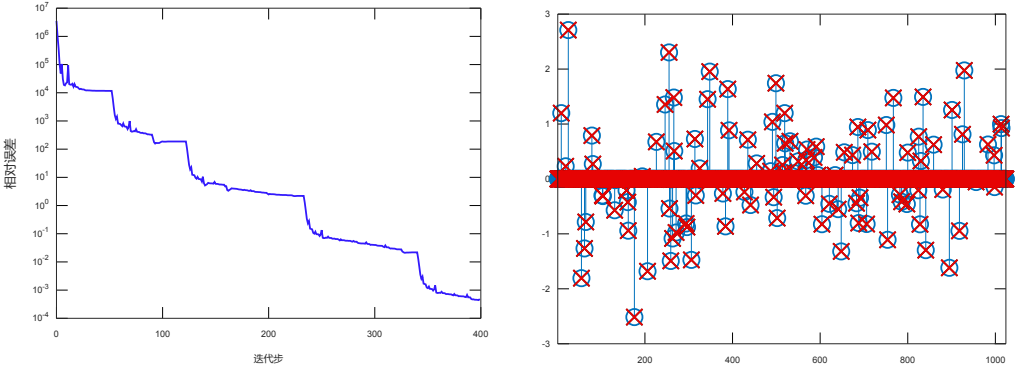

图 7: 光滑化 LASSO 问题求解结果：左图为相对误差；右图蓝色圆圈"◯" 标记为真值 $\boldsymbol{x}$ 的各分量，红色交叉"×" 标记对应梯度法所求解

33.3.2 共轭梯度法引入

思考：

- 负梯度方向真的是最理想的搜索方向吗？
- 由于负梯度方向的最速下降性，很容易使人们认为负梯度方向是理想的搜索方向。
- 必须指出，$\boldsymbol{x}$ 点处的负梯度方向 $-\nabla f(\boldsymbol{x})$，仅**在 $\boldsymbol{x}$ 点附近**才具有这种“最速下降”的性质，而对于整个极小化过程来说，那就是另外一回事了。
- 例如，一般二元二次凸函数的等值线为一族共心椭圆，当用最速下降法趋近极小点时，其搜索路径呈**直角锯齿状**（如下图所示）。

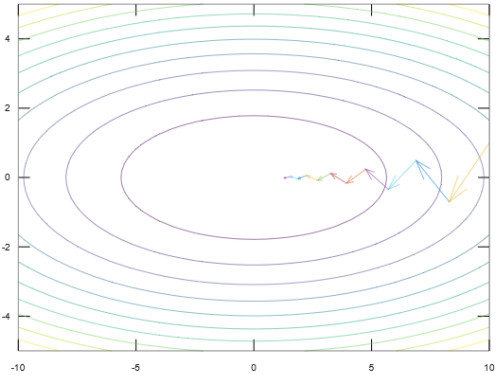

图 8

实际上，对于正定二次函数，我们还可以寻找到更好的下降方向，即接下来要介绍的**共轭梯度法**。

1. 共轭方向及其性质

(1) 共轭方向

设 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 是 n 维向量，若有

$$\boldsymbol{x}^T\boldsymbol{y} = 0$$

就称 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 正交。再设 $\boldsymbol{A}$ 为 $n \times n$ **对称正定阵**，如果 $\boldsymbol{x}$ 和 $\boldsymbol{A}\boldsymbol{y}$ 正交，即有

$$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y} = 0 \tag{10}$$

则称 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 关于 $\boldsymbol{A}$ 共轭，或 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 为 $\boldsymbol{A}$ **共轭** ($\boldsymbol{A}$ 正交)。

一般地，设 $\boldsymbol{A}$ 为 $n \times n$ 对称正定阵，若非零向量组 $\boldsymbol{p}^{(1)}, \boldsymbol{p}^{(2)}, \cdots, \boldsymbol{p}^{(n)}$ 满足条件

$$(\boldsymbol{p}^{(i)})^T \boldsymbol{A} \boldsymbol{p}^{(j)} = 0 \quad (i \neq j; \quad i, j = 1, 2, \cdots, n) \tag{11}$$

则称该向量组为 $\boldsymbol{A}$ 共轭。

- 如果 $\boldsymbol{A} = \boldsymbol{I}$（单位阵），则上述条件即为通常的正交条件。
- 因此，$\boldsymbol{A}$ 共轭概念实际上是通常正交概念的推广。

定理 3

设 $\boldsymbol{A}$ 为 $n \times n$ 对称正定阵，$\boldsymbol{p}^{(1)}, \boldsymbol{p}^{(2)}, \cdots, \boldsymbol{p}^{(n)}$ 为 $\boldsymbol{A}$ 共轭的非零向量，则这一组向量线性无关。

证明.

设向量 $\boldsymbol{p}^{(1)}, \boldsymbol{p}^{(2)}, \cdots, \boldsymbol{p}^{(n)}$ 之间存在如下线性关系

$$\alpha_1 \boldsymbol{p}^{(1)} + \alpha_2 \boldsymbol{p}^{(2)} + \cdots + \alpha_n \boldsymbol{p}^{(n)} = 0$$

对 $i = 1, 2, \cdots, n$, 用 $(\boldsymbol{p}^{(i)})^T \boldsymbol{A}$ 左乘上式得

$$\alpha_i (\boldsymbol{p}^{(i)})^T \boldsymbol{A} \boldsymbol{p}^{(i)} = 0$$

但 $\boldsymbol{p}^{(i)} \neq 0, \boldsymbol{A}$ 为正定，即

$$(\boldsymbol{p}^{(i)})^T \boldsymbol{A} \boldsymbol{p}^{(i)} > 0$$

故必有 $\alpha_i = 0, \quad i = 1, 2, \cdots, n$，从而 $\boldsymbol{p}^{(1)}, \boldsymbol{p}^{(2)}, \cdots, \boldsymbol{p}^{(n)}$ 线性无关。 □

2. 正定二次函数的优化问题

(2) 正定二次函数的共轭梯度法

现在利用共轭的关系，求解一类特殊的无约束优化问题

$$\min\ f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^T\boldsymbol{x} + c \tag{12}$$

式中 $\boldsymbol{A}$ 为 $n \times n$ 对称正定阵；$\boldsymbol{x}, \boldsymbol{b}$ 为 n 维向量；c 为常数。问题式(12) 称为**正定二次函数极小问题**，它在最优化问题中起到极其重要的作用。

定理 4

设向量 $\boldsymbol{p}^{(i)}, i=0,1,2,\cdots,n-1$, 为 $\boldsymbol{A}$ 共轭，则从任一点 $\boldsymbol{x}^{(0)}$ 出发，相继以 $\boldsymbol{p}^{(0)}, \boldsymbol{p}^{(1)}, \cdots, \boldsymbol{p}^{(n-1)}$ 为搜索方向的下述算法

$$\begin{cases} \min\limits_{\lambda} f(\boldsymbol{x}^{(k)}+\lambda \boldsymbol{p}^{(k)}) = f(\boldsymbol{x}^{(k)}+\lambda_k \boldsymbol{p}^{(k)}) \\ \boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)}+\lambda_k \boldsymbol{p}^{(k)} \end{cases}$$

经 n 次一维搜索收敛于问题式(12) 的极小点 $\boldsymbol{x}^*$。

该定理说明：

- 求解正定二次函数的优化问题，只需要找到一组（n 个）共轭方向即可。
- 在理想的情况下，二次正定函数的优化问题只要迭代 n 步就会终止，并找到最优解 $\boldsymbol{x}^*$。

3. 二维情形下共轭梯度法的探索

下面以二维正定二次函数的情况进行说明：

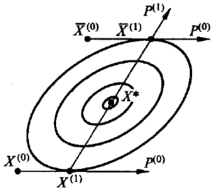

图 9

二维正定二次函数的等值线，在极小点附近可用一族**公心椭圆**来代表（如上图）。其中 $\boldsymbol{p}^{(0)}$ 与 $\boldsymbol{p}^{(1)}$ 即为一组共轭方向。因为 $\boldsymbol{p}^{(1)}$ 方向经过极小点。所以，在第二次迭代时，利用精确线搜索必将找到最优点 $\boldsymbol{x}^*$。

那么如何寻找到这些共轭方向，自然是求解正定二次优化问题的关键。在正式给出共轭梯度法之前，仍然以二维情形进行思考。

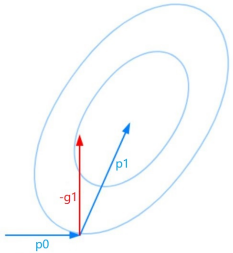

图 10

可以看出第二次的共轭方向 $\boldsymbol{p}_1$ 不再是负梯度方向 $-\boldsymbol{g}_1$，而是负梯度方向 $-\boldsymbol{g}_1$ 与前一次共轭方向 $\boldsymbol{p}_0$ 的线性组合。

这样，我们可以假定：

$$\boldsymbol{p}_1 = -\boldsymbol{g}_1 + \beta_0 \boldsymbol{p}_0.$$

其中初始共轭方向 $\boldsymbol{p}_0$ 取为负梯度方向 $-\boldsymbol{g}_0$。由 $\boldsymbol{p}_0$ 和 $\boldsymbol{p}_1$ 共轭，便可推出

$$\beta_0 = \frac{\boldsymbol{g}_1^T \boldsymbol{A} \boldsymbol{p}_0}{\boldsymbol{p}_0^T \boldsymbol{A} \boldsymbol{p}_0} = \frac{\boldsymbol{g}_1^T (\boldsymbol{g}_1 - \boldsymbol{g}_0)}{\boldsymbol{p}_0^T (\boldsymbol{g}_1 - \boldsymbol{g}_0)} = \frac{\boldsymbol{g}_1^T \boldsymbol{g}_1}{\boldsymbol{g}_0^T \boldsymbol{g}_0}.$$

其中第三个等式是根据精确线搜索的性质 $\boldsymbol{g}_1^T \boldsymbol{p}_0 = 0$ 得到的。

猜想：这对 n 维空间的正定二次函数也是成立的。

4. 共轭梯度法具体步骤

如果猜想成立，则很容易总结出**共轭梯度法**如下：

Algorithm 3 共轭梯度法

1: 选择初始近似 $\boldsymbol{x}^{(0)}$，给出允许误差 $\varepsilon > 0$；

2: 计算

$$\boldsymbol{p}^{(0)} = -\nabla f(\boldsymbol{x}^{(0)})$$

3: 一般地，假定已得出 $\boldsymbol{x}^{(k)}$ 和 $\boldsymbol{p}^{(k)}$，则利用精确线搜索可计算其第 $k+1$ 次近似 $\boldsymbol{x}^{(k+1)}$

4: 若 $\|\nabla f(\boldsymbol{x}^{(k+1)})\|^2 \leq \varepsilon$，停止计算，$\boldsymbol{x}^{(k+1)}$ 即为要求的近似解。否则，若 $k < n-1$，则

$$\boldsymbol{p}^{(k+1)} = -\nabla f(\boldsymbol{x}^{(k+1)}) + \beta_k \boldsymbol{p}^{(k)} \tag{13}$$

$$\beta_k = \frac{\nabla f(\boldsymbol{x}^{(k+1)})^T \nabla f(\boldsymbol{x}^{(k+1)})}{\nabla f(\boldsymbol{x}^{(k)})^T \nabla f(\boldsymbol{x}^{(k)})} \tag{14}$$

计算 β_k 和 $\boldsymbol{p}^{(k+1)}$，并转向第 3 步。

5. 搜索方向的共轭性

下面证明我们的猜想，即由上述共轭梯度法得到的搜索方向 $\boldsymbol{p}^{(0)}, \boldsymbol{p}^{(1)}, \cdots, \boldsymbol{p}^{(n-1)}$ 确实为 $\boldsymbol{A}$ 共轭。为证明这一点，只需要考虑前 k 次迭代成立的条件下（$k \leq 1$ 时容易验证成立），第 $k+1$ 次迭代这一结论依然成立。下面先证明几个子定理，然后再证明搜索方向的共轭性。

定理 5

第 $k+1$ 个迭代点的梯度方向与前面的搜索方向正交，即：$\nabla f(\boldsymbol{x}^{(k+1)})^T \boldsymbol{p}^{(j)} = 0,\ j = 0, \cdots, k$。

证明.

- $j = k$ 时，由精确线搜索性质，显然成立。
- $j < k$ 时，

$$
\begin{aligned}
\nabla f(\boldsymbol{x}^{(k+1)})^T \boldsymbol{p}^{(j)} &= \nabla f(\boldsymbol{x}^{(j+1)})^T \boldsymbol{p}^{(j)} + \sum_{i=j+1}^{k} (\nabla f(\boldsymbol{x}^{(i+1)})^T \boldsymbol{p}^{(j)} - \nabla f(\boldsymbol{x}^{(i)})^T \boldsymbol{p}^{(j)}) \\
&= \sum_{i=j+1}^{k} (\nabla f(\boldsymbol{x}^{(i+1)})^T - \nabla f(\boldsymbol{x}^{(i)})^T) \boldsymbol{p}^{(j)} \quad \text{（精确线搜索）} \\
&= \sum_{i=j+1}^{k} (A(x^{(i+1)} - x^{(i)}))^T \boldsymbol{p}^{(j)} \\
&= \sum_{i=j+1}^{k} \lambda_i (\boldsymbol{p}^{(i)})^T A \boldsymbol{p}^{(j)} = 0 \quad \text{（相互共轭）}
\end{aligned}
\tag{15}
$$

定理 6

第 $k+1$ 个迭代点的梯度方向与前面的梯度方向正交，即：

$$\nabla f(\boldsymbol{x}^{(k+1)})^T \nabla f(\boldsymbol{x}^{(j)}) = 0,\ j = 0, \cdots, k.$$

证明.

根据共轭梯度法中搜索方向的迭代公式：

$$\begin{aligned} \nabla f(\boldsymbol{x}^{(k+1)})^T \boldsymbol{p}^{(j)} &= \nabla f(\boldsymbol{x}^{(k+1)})^T (-\nabla f(\boldsymbol{x}^{(j)}) + \beta_{j-1} \boldsymbol{p}^{(j-1)}) \\ &= -\nabla f(\boldsymbol{x}^{(k+1)})^T \nabla f(\boldsymbol{x}^{(j)}) + \beta_{j-1} \nabla f(\boldsymbol{x}^{(k+1)})^T \boldsymbol{p}^{(j-1)} \\ &= -\nabla f(\boldsymbol{x}^{(k+1)})^T \nabla f(\boldsymbol{x}^{(j)}) \quad （上一定理结论） \end{aligned} \tag{16}$$

根据上一定理的结论可知，整个式子为 0。故由最后一个等式知结论成立。 □

定理 7

共轭梯度法产生的搜索方向 $\boldsymbol{p}^{(0)}, \boldsymbol{p}^{(1)}, \cdots, \boldsymbol{p}^{(n-1)}$ 为 $\boldsymbol{A}$ 共轭。

证明.

仍然考虑第 $k+1$ 次迭代的情形（为简化描述，下用 $\boldsymbol{g}_k$ 表示 $\nabla f(\boldsymbol{x}^{(k)})$）：

- 先证当 $j=k$ 时，$(\boldsymbol{p}^{(j)})^T \boldsymbol{A} \boldsymbol{p}^{(k+1)} = 0$.
  因为 $\boldsymbol{p}^{(k+1)} = -\boldsymbol{g}_{k+1} + \beta_k \boldsymbol{p}^{(k)}$，所以

$$
\begin{aligned}
(\boldsymbol{p}^{(k)})^T \boldsymbol{A} \boldsymbol{p}^{(k+1)} &= -(\boldsymbol{p}^{(k)})^T \boldsymbol{A} \boldsymbol{g}_{k+1} + \frac{\boldsymbol{g}_{k+1}^T \boldsymbol{g}_{k+1}}{\boldsymbol{g}_k^T \boldsymbol{g}_k} (\boldsymbol{p}^{(k)})^T \boldsymbol{A} \boldsymbol{p}^{(k)} \\
&= -\frac{(\boldsymbol{g}_{k+1} - \boldsymbol{g}_k)^T \boldsymbol{g}_{k+1}}{\lambda_k} + \frac{\boldsymbol{g}_{k+1}^T \boldsymbol{g}_{k+1}}{\boldsymbol{g}_k^T \boldsymbol{g}_k} (\boldsymbol{p}^{(k)})^T \boldsymbol{A} \boldsymbol{p}^{(k)} \qquad (17) \\
&= -\frac{\boldsymbol{g}_{k+1}^T \boldsymbol{g}_{k+1}}{\lambda_k} + \frac{\boldsymbol{g}_{k+1}^T \boldsymbol{g}_{k+1}}{\boldsymbol{g}_k^T \boldsymbol{g}_k} (\boldsymbol{p}^{(k)})^T \boldsymbol{A} \boldsymbol{p}^{(k)}
\end{aligned}
$$

  根据精确线搜索知 $\lambda_k = -\frac{(\boldsymbol{p}^{(k)})^T \boldsymbol{g}_k}{(\boldsymbol{p}^{(k)})^T \boldsymbol{A} (\boldsymbol{p}^{(k)})} = \frac{\boldsymbol{g}_k^T \boldsymbol{g}_k}{(\boldsymbol{p}^{(k)})^T \boldsymbol{A} (\boldsymbol{p}^{(k)})}$。因此，上式为 0。

证明（续）.

- 现考虑 $j<k$ 时，$(\boldsymbol{p}^{(j)})^T\boldsymbol{A}\boldsymbol{p}^{(k+1)}=0.$

  同理

$$\begin{aligned}(\boldsymbol{p}^{(j)})^T\boldsymbol{A}\boldsymbol{p}^{(k+1)} &= -(\boldsymbol{p}^{(j)})^T\boldsymbol{A}\boldsymbol{g}_{k+1}+\frac{\boldsymbol{g}_{k+1}^T\boldsymbol{g}_{k+1}}{\boldsymbol{g}_k^T\boldsymbol{g}_k}(\boldsymbol{p}^{(j)})^T\boldsymbol{A}\boldsymbol{p}^{(k)}\\ &= -\frac{(\boldsymbol{g}_{j+1}-\boldsymbol{g}_j)^T\boldsymbol{g}_{k+1}}{\lambda_j}+\frac{\boldsymbol{g}_{k+1}^T\boldsymbol{g}_{k+1}}{\boldsymbol{g}_k^T\boldsymbol{g}_k}(\boldsymbol{p}^{(j)})^T\boldsymbol{A}\boldsymbol{p}^{(k)}\end{aligned} \tag{18}$$

根据定理 6 知等式右边第一部分为 0，又由于 $\boldsymbol{p}^{(j)},\ j=0,\cdots,k$ 为 $\boldsymbol{A}$ 共轭，所以第二部分为 0。故得证。 □

6. 关于共轭梯度法的实现

- 共轭梯度法还有很多其他等价形式。
- 应当指出，从理论上说，对于正定二次函数的情形，进行 n 次迭代即可达到极小点。
- 但是，在实际计算中，由于数据的四舍五入以及计算误差的积累，往往做不到这一点。
- 此外，由于 n 维问题的共轭方向最多只有 n 个，在 n 步以后继续如上进行是没有意义的。
- 因此，在实际应用时，如迭代到 n 步还不收敛，就将 $\boldsymbol{x}^{(n)}$ 作为新的初始近似，重新开始迭代。根据实际经验，采用这种再开始的办法，一般都可得到较好的效果。

例 4

试用共轭梯度法求下述二次函数的极小点

$$f(\boldsymbol{x}) = \frac{3}{2}x_1^2 + \frac{1}{2}x_2^2 - x_1x_2 - 2x_1$$

解

将 $f(\boldsymbol{x})$ 化成式(12) 的形式，得

$$\boldsymbol{A} = \begin{pmatrix} 3 & -1 \\ -1 & 1 \end{pmatrix}$$

现从 $\boldsymbol{x}^{(0)} = (-2, 4)^T$ 开始，由于 $\nabla f(\boldsymbol{x}) = [(3x_1 - x_2 - 2), (x_2 - x_1)]^T$ 故

$$\nabla f(\boldsymbol{x}^{(0)}) = (-12, 6)^T, \qquad \boldsymbol{p}^{(0)} = -\nabla f(\boldsymbol{x}^{(0)}) = (12, -6)^T$$

$$\lambda_0 = -\frac{\nabla f(\boldsymbol{x}^{(0)})^T \boldsymbol{p}^{(0)}}{(\boldsymbol{p}^{(0)})^T \boldsymbol{A} \boldsymbol{p}^{(0)}} = -\frac{(-12, 6)\begin{pmatrix} 12 \\ -6 \end{pmatrix}}{(12, -6)\begin{pmatrix} 3 & -1 \\ -1 & 1 \end{pmatrix}\begin{pmatrix} 12 \\ -6 \end{pmatrix}} = \frac{180}{612} = \frac{5}{17}$$

于是

$$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \lambda_0 \boldsymbol{p}^{(0)} = \begin{pmatrix} -2 \\ 4 \end{pmatrix} + \frac{5}{17} \begin{pmatrix} 12 \\ -6 \end{pmatrix} = \left(\frac{26}{17}, \frac{38}{17} \right)^T$$

$$\nabla f(\boldsymbol{x}^{(1)}) = \left(\frac{6}{17}, \frac{12}{17} \right)^T$$

$$\beta_0 = \frac{\nabla f(\boldsymbol{x}^{(1)})^T \nabla f(\boldsymbol{x}^{(1)})}{\nabla f(\boldsymbol{x}^{(0)})^T \nabla f(\boldsymbol{x}^{(0)})} = \frac{\left(\frac{6}{17}, \frac{12}{17} \right) \begin{pmatrix} \frac{6}{17} \\ \frac{12}{17} \end{pmatrix}}{(-12, 6) \begin{pmatrix} -12 \\ 6 \end{pmatrix}} = \frac{1}{289}$$

$$\boldsymbol{p}^{(1)} = -\nabla f(\boldsymbol{x}^{(1)}) + \beta_0 \boldsymbol{p}^{(0)} = -\begin{pmatrix} \frac{6}{17} \\ \frac{12}{17} \end{pmatrix} + \frac{1}{289} \begin{pmatrix} 12 \\ -6 \end{pmatrix} = \left(-\frac{90}{289}, -\frac{210}{289} \right)^T$$

如图11 表明了本例的搜索方向。

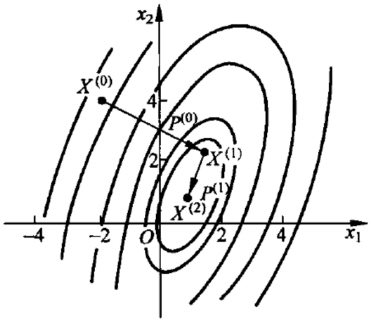

图 11

再计算

$$\begin{aligned}\lambda_1 &= -\frac{\nabla f(\boldsymbol{x}^{(1)})^T \boldsymbol{p}^{(1)}}{(\boldsymbol{p}^{(1)})^T \boldsymbol{A} \boldsymbol{p}^{(1)}} \\ &= -\frac{\left(\frac{6}{17}, \frac{12}{17}\right)\left(-\frac{90}{289}, -\frac{210}{289}\right)^T}{\left(-\frac{90}{289}, -\frac{210}{289}\right)\begin{pmatrix} 3 & -1 \\ -1 & 1 \end{pmatrix}\left(-\frac{90}{289}, -\frac{210}{289}\right)^T} \\ &= \frac{6 \times 17 \times 90 + 12 \times 17 \times 210}{(-60, -120)(-90, -210)^T} = \frac{17(6 \times 90 + 12 \times 210)}{60 \times 90 + 120 \times 210} = \frac{17}{10}\end{aligned}$$

故

$$\boldsymbol{x}^{(2)} = \boldsymbol{x}^{(1)} + \lambda_1 \boldsymbol{p}^{(1)} = \begin{pmatrix} \frac{26}{17} \\ \frac{38}{17} \end{pmatrix} + \frac{17}{10}\begin{pmatrix} -\frac{90}{289} \\ -\frac{210}{289} \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

易知其梯度为 0，这就是 $f(\boldsymbol{x})$ 的极小点。

33.3.3 次梯度算法

在实际应用中经常会遇到不可微的函数，对于这类函数我们无法在每个点处求出梯度，但往往它们的最优值都是在不可微点处取到的．为了能处理这种情形，这一节介绍次梯度算法。现在我们在问题 (1) 中假设 $f(\boldsymbol{x})$ 为凸函数, 但不一定可微. 对凸函数可以在定义域的内点处定义次梯度 $\boldsymbol{g} \in \partial f(\boldsymbol{x})$. 类比梯度法的构造, 我们有如下次梯度算法的迭代格式：

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \lambda_k \boldsymbol{g}^k, \quad \boldsymbol{g}^k \in \partial f\left(\boldsymbol{x}^k\right), \tag{19}$$

其中 $\lambda_k > 0$ 为步长.

关于次梯度算法中的步长通常有如下四种选择:

1. 固定步长 $\lambda_k = \lambda$;
2. 固定 $\left\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\right\|$, 即 $\lambda_k \left\|\boldsymbol{g}^k\right\|$ 为常数;
3. 消失步长 $\lambda_k \to 0$ 且 $\sum_{k=0}^{\infty} \lambda_k = +\infty$;
4. 选取 λ_k 使其满足某种线搜索准则.

- 次梯度算法 (19) 的构造虽然是受梯度法的启发, 但在很多方面次梯度算法有其独特性质.
- 首先, 次微分 $\partial f(\boldsymbol{x})$ 是一个集合, 在次梯度算法的构造中只要求从这个集合中选出一个次梯度即可, 但在实际中不同的次梯度取法可能会产生截然不同的效果;
- 其次, 对于梯度法, 判断一阶最优性条件只需要验证 $\|\nabla f(\boldsymbol{x}^*)\|$ 是否充分小即可, 但对于次梯度算法, 此时有 $0 \in \partial f(\boldsymbol{x}^*)$, 而这个条件在实际应用中往往是不易直接验证的, 这导致我们不能使用它作为次梯度算法的停机条件;
- 此外, 步长选取在次梯度法中的影响非常大, 因此，次梯度算法的收敛性分析，相比于梯度法较为复杂一些。这里不对次梯度算法的收敛性进行展开叙述。

应用：次梯度法求解正定矩阵补全

正定矩阵补全问题是一种特殊的矩阵恢复问题, 它的具体形式为

$$\begin{aligned} \text{find } & \boldsymbol{X} \in \mathcal{S}^n, \\ \text{s.t. } & X_{ij} = M_{ij}, \quad (i,j) \in \Omega, \\ & \boldsymbol{X} \succeq 0. \end{aligned}$$

其中 Ω 是已经观测的分量位置集合. 问题本质上是一个目标函数为常数的半定规划问题, 但由于其特殊性我们可以使用次梯度算法求解.

考虑两个集合

$$C_1 = \{\boldsymbol{X} \mid X_{ij} = M_{ij}, (i,j) \in \Omega\},$$

$$C_2 = \{\boldsymbol{X} \mid \boldsymbol{X} \succeq 0\},$$

因此，求解正定矩阵补全问题等价于寻找闭凸集 C_1 和 C_2 的交集. 定义欧几里得距离函数

$$d_j(\boldsymbol{X}) = \inf_{\boldsymbol{Y} \in C_j} \|\boldsymbol{X} - \boldsymbol{Y}\|_F,$$

则可将这个问题转化为无约束非光滑优化问题

$$\min f(\boldsymbol{X}) = \max\{d_1(\boldsymbol{X}), d_2(\boldsymbol{X})\}$$

由次梯度计算规则可知

$$\partial f(\boldsymbol{X}) = \begin{cases} \partial d_1(\boldsymbol{X}), & d_1(\boldsymbol{X}) > d_2(\boldsymbol{X}) \\ \partial d_2(\boldsymbol{X}), & d_1(\boldsymbol{X}) < d_2(\boldsymbol{X}) \\ \operatorname{conv}\left(\partial d_1(\boldsymbol{X}) \cup \partial d_2(\boldsymbol{X})\right), & d_1(\boldsymbol{X}) = d_2(\boldsymbol{X}) \end{cases}$$

而又根据固定分量的函数极小值求次梯度的例子, 我们可以求得距离函数的一个次梯度为

$$G_j = \begin{cases} 0, & \boldsymbol{X} \in C_j, \\ \frac{1}{d_j(\boldsymbol{X})}\left(\boldsymbol{X} - \mathcal{P}_{C_j}(\boldsymbol{X})\right), & \boldsymbol{X} \notin C_j, \end{cases}$$

其中 $\mathcal{P}_{C_j}(\boldsymbol{X}) = \underset{Y \in C_j}{\arg\min} \|\boldsymbol{Y} - \boldsymbol{X}\|_F$ 为 $\boldsymbol{X}$ 到 C_j 的投影. 对于集合 $C_1, \boldsymbol{X}$ 在它上面的投影为

$$\left(\mathcal{P}_{\mathrm{C}_1}(\boldsymbol{X})\right)_{ij} = \begin{cases} \boldsymbol{M}_{ij}, & (i,j) \in \Omega \\ \boldsymbol{X}_{ij}, & (i,j) \notin \Omega \end{cases}$$

对于集合 $C_2, \boldsymbol{X}$ 在它上面的投影为

$$\mathcal{P}_{\mathrm{C}_2}(\boldsymbol{X}) = \sum_{i=1}^{n} \max\left(0, \lambda_i\right) \boldsymbol{q}_i \boldsymbol{q}_i^{\mathrm{T}}$$

其中 $\lambda_i, \boldsymbol{q}_i$ 分别是 $\boldsymbol{X}$ 的第 i 个特征值和特征向量.

在这里注意, 为了比较 $d_1(\boldsymbol{X})$ 和 $d_2(\boldsymbol{X})$ 的大小关系, 我们在计算次梯度时还是要将 $\boldsymbol{X}$ 到两个集合的投影分别求出, 之后再选取距离较大的一个计算出次梯度。因此，完整的次梯度计算过程为：

1. 给定点 $\boldsymbol{X}$, 根据上式计算出 $\boldsymbol{X}$ 到 C_1 和 C_2 的投影, 分别记为 $\boldsymbol{P}_1$ 和 $\boldsymbol{P}_2$ ；
2. 比较 $d_j(\boldsymbol{X}) = \|\boldsymbol{X} - \boldsymbol{P}_j\|_F, j = 1, 2$, 较大者记为 $\hat{j}$;
3. 计算次梯度 $G = \frac{\boldsymbol{X} - \boldsymbol{P}_{\hat{j}}}{d_{\hat{j}}(\boldsymbol{X})}$。

本讲小结

线搜索

- 无约束优化算法概述：迭代法
- 精确线搜索
- 非精确线搜索

一阶优化算法

- 梯度下降法
- 共轭梯度法：共轭方向、正定二次函数的共轭梯度法
- 次梯度算法

我们从一般性的角度讨论了无约束优化算法，需要解决步长和下降方向的问题。步长是属于线搜索的范畴，我们进行了详细的探讨。下降方向在这里主要是介绍了一阶的方法。那么如何利用二阶信息呢？