第三讲　一维投影法

一维投影法

迭代格式

$$x^{(k+1)} = x^{(k)} + \alpha_k p_k, \quad k = 0, 1, 2, ...$$

- $p_k \in \mathbb{R}^n$ 代表搜索方向, $\alpha_k \in \mathbb{R}$ 代表搜索步长.
- 由于这类方法是沿着一维空间 (即 $\text{span}\{p_k\}$) 来更新迭代解, 因此称为 一维投影法 (One-dimensional projection methods).
- 搜索方向 p_k 和搜索步长 α_k 的不同选取方法就会生成不同的一维投影法. 典型方法有: 最速下降法, Kaczmarz 方法, 坐标下降法和 Cimmino 方法等.

最速下降法与共轭梯度法

(对称正定情形)

线性方程组与二次规划问题

定理 设 $A \in \mathbb{R}^{n\times n}$ 对称正定, 则 x_* 是 $Ax = b$ 的解当且仅当 x_* 是**最小化问题**

$$\min_{x\in\mathbb{R}^n} \quad \Phi(x) \triangleq \frac{1}{2}x^{\mathsf{T}}Ax - b^{\mathsf{T}}x$$

的解, 即 $\Phi(x)$ 的最小值点. (板书)

线性方程组与二次规划问题

定理 设 $A \in \mathbb{R}^{n\times n}$ 对称正定, 则 x_* 是 $Ax = b$ 的解当且仅当 x_* 是**最小化问题**

$$\min_{x\in\mathbb{R}^n} \quad \Phi(x) \triangleq \frac{1}{2}x^{\mathsf{T}}Ax - b^{\mathsf{T}}x$$

的解, 即 $\Phi(x)$ 的最小值点. (板书)

求解 $Ax = b$ $\Longleftrightarrow$ 计算 $\Phi(x)$ 的最小值点

搜索方向/下降方向

根据多元函数的 Taylor 展开公式, 我们有

$$\Phi(x) = \Phi\big(x^{(0)}\big) + \big(x - x^{(0)}\big)^{\mathsf{T}} \nabla \Phi\big(x^{(0)}\big) + o\big(\|x - x^{(0)}\|\big).$$

记 p 为 $x - x^{(0)}$ 所在的方向, 即 $p = \dfrac{x - x^{(0)}}{\|x - x^{(0)}\|}$, 则

$$\big(x - x^{(0)}\big)^{\mathsf{T}} \nabla \Phi\big(x^{(0)}\big) = \|x - x^{(0)}\| \cdot p^{\mathsf{T}} \nabla \Phi\big(x^{(0)}\big).$$

搜索方向/下降方向

根据多元函数的 Taylor 展开公式, 我们有

$$\Phi(x) = \Phi\big(x^{(0)}\big) + \big(x - x^{(0)}\big)^{\mathsf{T}} \nabla \Phi\big(x^{(0)}\big) + o\big(\|x - x^{(0)}\|\big).$$

记 p 为 $x - x^{(0)}$ 所在的方向, 即 $p = \dfrac{x - x^{(0)}}{\|x - x^{(0)}\|}$, 则

$$\big(x - x^{(0)}\big)^{\mathsf{T}} \nabla \Phi\big(x^{(0)}\big) = \|x - x^{(0)}\| \cdot p^{\mathsf{T}} \nabla \Phi\big(x^{(0)}\big).$$

因此, 只要满足 $p^{\mathsf{T}} \nabla \Phi\big(x^{(0)}\big) < 0$, 则 p 就是 **下降方向**.

步长

由于 A 对称正定, 直接计算可得

$$\begin{aligned}\Phi\big(x^{(0)}+\alpha p_1\big) &= \frac{1}{2}\big(x^{(0)}+\alpha p_1\big)^{\mathsf{T}} A\big(x^{(0)}+\alpha p_1\big) - b^{\mathsf{T}}\big(x^{(0)}+\alpha p_1\big)\\ &= \frac{1}{2}\alpha^2 p_1^{\mathsf{T}} A p_1 + \alpha p_1^{\mathsf{T}} A x^{(0)} + \frac{1}{2}\big(x^{(0)}\big)^{\mathsf{T}} A x^{(0)} - b^{\mathsf{T}} x^{(0)} - \alpha b^{\mathsf{T}} p_1\\ &= \frac{1}{2}\alpha^2 p_1^{\mathsf{T}} A p_1 - \alpha p_1^{\mathsf{T}} r_0 + \Phi\big(x^{(0)}\big),\end{aligned}$$

其中 $r_0 = b - Ax^{(0)}$.

这是关于 α 的一元二次函数, 且二次项系数为正, 所以

$$\alpha_1 = \operatorname*{argmin}_{\alpha>0} \Phi\big(x^{(0)}+\alpha p_1\big) = \frac{p_1^{\mathsf{T}} r_0}{p_1^{\mathsf{T}} A p_1}.$$

最速下降法

最速下降方向与最速下降法

下降速度最快的方向 $\rightarrow p^{\mathsf{T}}\nabla\Phi\big(x^{(0)}\big)$ 最小.

由 Cauchy-Schwarz 不等式可知

$$\big|p^{\mathsf{T}}\nabla\Phi\big(x^{(0)}\big)\big| \leq \|p\| \cdot \|\nabla\Phi\big(x^{(0)}\big)\|,$$

等号当且仅当 p 与 $\nabla\Phi\big(x^{(0)}\big)$ 共线时成立.

- 因此当 $p = -\dfrac{\nabla\Phi\big(x^{(0)}\big)}{\|\nabla\Phi\big(x^{(0)}\big)\|}$ 时, $p^{\mathsf{T}}\nabla\Phi\big(x^{(0)}\big)$ 达到最小.
- 我们称该下降方向为**最速下降方向**, 相应的线搜索方法为**最速下降法**.
- 由于最速下降方向就是 $\Phi(x)$ 在当前迭代点处的负梯度方向, 因此也称为**负梯度方向法** 或 **梯度下降法**.

最速下降法的迭代格式

最速下降法的一般格式可表示为

$$x^{(k)} = x^{(k-1)} + \alpha_k p_k, \quad k = 1, 2, \ldots,$$

其中 $p_k = -\nabla\Phi\left(x^{(k-1)}\right) = b - Ax^{(k-1)} = r_{k-1}$, $\alpha_k = \dfrac{p_k^{\mathsf{T}} r_{k-1}}{p_k^{\mathsf{T}} A p_k} = \dfrac{r_{k-1}^{\mathsf{T}} r_{k-1}}{r_{k-1}^{\mathsf{T}} A r_{k-1}}$

实际计算时, 我们无需对下降方向 p_k 进行单位化

最速下降法

算法 最速下降法 (Steepest Descent Algorithm)

1: **给定初值** $x^{(0)}$, **令** $k = 1$
2: **while** not converge **do**
3: 　**计算负梯度方向** (**残量**) $r_{k-1} = b - Ax^{(k-1)}$ **和步长** $\alpha_k = (r_{k-1}^{\mathsf{T}} r_{k-1})/(r_{k-1}^{\mathsf{T}} A r_{k-1})$
4: 　**计算** $x^{(k)} = x^{(k-1)} + \alpha_k p_k$, **其中** $p_k = r_{k-1}$
5: 　**令** $k = k + 1$
6: **end while**

最速下降法的收敛性

定理 设 $A \in \mathbb{R}^{n\times n}$ 对称正定, 则对任意初值 $x^{(0)}$, 最速下降法都收敛, 且

$$\frac{\left\|x^{(k)}-x_*\right\|_A}{\left\|x^{(k-1)}-x_*\right\|_A} \leq \frac{\kappa-1}{\kappa+1},$$

其中 κ 为 A 的谱条件数, 范数 $\|x\|_A \triangleq \sqrt{x^{\mathsf{T}}Ax}$.

(留作课外自习, 可以参见矩阵计算或最优化方法的相关教材)

最速下降法举例

例 用最速下降法求解 $Ax = b$, 其中 $A = \begin{bmatrix} 15 & 2 \\ 2 & 15 \end{bmatrix}$, $b = \begin{bmatrix} 17 \\ 17 \end{bmatrix}$, 初值 $x^{(0)} = \begin{bmatrix} -0.5 \\ 0 \end{bmatrix}$.

k	x	relres
1	$[0.94896898, 1.06454864]$	3.54e-02
2	$[0.99757851, 0.99838567]$	1.61e-03
3	$[0.99991762, 1.00010420]$	5.71e-05
4	$[0.99999609, 0.99999739]$	2.61e-06
5	$[0.99999987, 1.00000017]$	9.21e-08

"relres" 表示相对残量

$$\text{relres} = \frac{\|b - Ax^{(k)}\|_2}{\|b - Ax^{(0)}\|_2}$$

最速下降法的局部最优性

由步长的选取方法可知, 最速下降法的迭代解具有下面的最优性

$$x^{(k)} = \underset{x \in x^{(k-1)} + \mathrm{span}\{p_k\}}{\operatorname{argmin}} \Phi(x)$$

由于舍弃了 Taylor 展开式中的高阶项, 因此最速下降方向是局部最优的

注记

早在 1829 年, Cauchy 就提出了求解 (与积分近似有关) 非线性方程的最速下降法. Riemann 等都曾做过相关研究工作. 我们这里介绍的是由 Kantorovitch 于 1945 年提出的求解对称正定线性方程组的最速下降法迭代格式.

最速下降法的不足

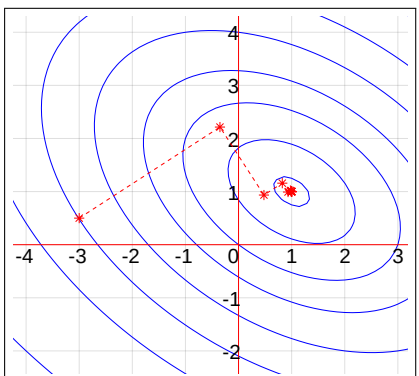
最速下降法的不足

最速下降法每次选取下降方向时只考虑局部最优, 无法保证下降方向 $p_1, p_2, \ldots$ 线性无关, 这意味着迭代过程可能会出现 “之” 字型, 即同一个下降方向可能会多次出现, 这就会导致收敛速度变得非常缓慢.

最速下降法的不足

例 用最速下降法求解 $Ax = b$, 其中 $A = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}$, $b = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$, 初值 $x^{(0)} = \begin{bmatrix} -3 \\ 0.5 \end{bmatrix}$.

k	x	relres
1	$[-0.3498, 2.2148]$	2.70e-01
2	$[0.4784, 0.9348]$	1.30e-01
3	$[0.8240, 1.1584]$	3.52e-02
4	$[0.9320, 0.9915]$	1.70e-02
$\vdots$	$\vdots$	$\vdots$
14	$[1.0000, 1.0000]$	6.41e-07



共轭方向

为了改善收敛性质, 需要对下降方向进行改进, 由此, **共轭方向法** 应运而生.

定义 设 $A \in \mathbb{R}^{n\times n}$ 对称正定, 若非零向量 $x, y \in \mathbb{R}^n$ 满足

$$y^{\mathsf{T}} A x = 0,$$

则称 x 和 y 是 **A-共轭**的, 也称 **A-正交**的.

为什么 A-共轭

为什么选取 A-共轭的向量作为下降方向

设 $p_1, \ldots, p_n$ 相互 A-共轭, 则 $p_1, \ldots, p_n$ 线性无关, 因此构成 $\mathbb{R}^n$ 的一组基.

对任意给定的初值 $x^{(0)}$, 我们可以设

$$x_* - x^{(0)} = \alpha_1 p_1 + \alpha_2 p_2 + \cdots + \alpha_n p_n.$$

两边分别左乘 $p_k^{\mathsf{T}} A$, 可得

$$\alpha_k = \frac{p_k^{\mathsf{T}} A\left(x_* - x^{(0)}\right)}{p_k^{\mathsf{T}} A p_k} = \frac{p_k^{\mathsf{T}}\left(b - A x^{(0)}\right)}{p_k^{\mathsf{T}} A p_k}, \quad k = 1, 2, \ldots, n.$$

这意味着可通过 $p_1, \ldots, p_n$ 和右端项 b, 以及 $x^{(0)}$ 将方程组的解表示出来.

为什么 A-共轭

为什么选取 A-共轭的向量作为下降方向

设 $p_1, \ldots, p_n$ 相互 A-共轭, 则 $p_1, \ldots, p_n$ 线性无关, 因此构成 $\mathbb{R}^n$ 的一组基.

对任意给定的初值 $x^{(0)}$, 我们可以设

$$x_* - x^{(0)} = \alpha_1 p_1 + \alpha_2 p_2 + \cdots + \alpha_n p_n.$$

两边分别左乘 $p_k^{\mathsf{T}} A$, 可得

$$\alpha_k = \frac{p_k^{\mathsf{T}} A\left(x_* - x^{(0)}\right)}{p_k^{\mathsf{T}} A p_k} = \frac{p_k^{\mathsf{T}}\left(b - A x^{(0)}\right)}{p_k^{\mathsf{T}} A p_k}, \quad k = 1, 2, \ldots, n.$$

这意味着可通过 $p_1, \ldots, p_n$ 和右端项 b, 以及 $x^{(0)}$ 将方程组的解表示出来.

若采用一组正交基作为下降方向, 也可得到类似结论, 但 α_i 的表达式中包含 x_*

共轭方向法

考虑到算法的实用性, 当 n 很大时, 通常不需计算所有的 α_k, 而只要计算前面部分, 得到一个满足精度要求的近似解即可. 因此, 我们把 $x_* = x^{(0)} + \alpha_1 p_1 + \cdots$ 看作是一个迭代过程:

$$x^{(k)} = x^{(k-1)} + \alpha_k p_k, \quad k = 1, 2, \ldots.$$

共轭方向法

考虑到算法的实用性, 当 n 很大时, 通常不需计算所有的 α_k, 而只要计算前面部分, 得到一个满足精度要求的近似解即可. 因此, 我们把 $x_* = x^{(0)} + \alpha_1 p_1 + \cdots$ 看作是一个迭代过程:

$$x^{(k)} = x^{(k-1)} + \alpha_k p_k, \quad k = 1, 2, \ldots.$$

共轭方向法

以相互 A-共轭的向量作为下降方向的方法就称为 **共轭方向法**.

共轭方向法的全局最优性

相比于最速下降法的局部最优性:

$$x^{(k)} = \underset{x \in x^{(k-1)} + \mathrm{span}\{p_k\}}{\mathrm{argmin}} \Phi(x),$$

共轭方向法具有全局最优性.

定理 设 $x^{(k)}$ 是由共轭方向法得到的迭代解, 则

$$x^{(k)} = \underset{x \in x^{(0)} + \mathrm{span}\{p_1, p_2, \ldots, p_k\}}{\mathrm{argmin}} \Phi(x)$$

有限步终止

显然, 在不考虑舍入误差的情况下, 共轭方向法具有有限步终止性质.

定理 设 $A \in \mathbb{R}^{n\times n}$ 对称正定, 如果不考虑舍入误差, 则共轭方向法至多经过 n 次迭代后, 近似解 $x^{(k)}$ 就是精确解 x_*.

怎么构造共轭方向?

如何构造相互 A-共轭的下降方向

- 首先需要指出的是, 这样的向量组并不唯一.
  事实上, 与 Gram-Schmidt 正交化过程类似, 任何一组线性无关的向量组都可以构造出一组相互 A-共轭的向量组.
- 目前最成功的是共轭梯度 (Conjugate Gradient) 下降方向.
  它是在最速下降方向的基础上, 通过递推方法来构造的, 可以看作是将最速下降方向进行 A-共轭化, 或者是对最速下降方向的改进.

共轭梯度下降方向的构造

1. 给定 $x^{(0)}$, 取 $\Phi(x)$ 在 $x^{(0)}$ 处的负梯度方向:

$$p_1 = -\nabla\Phi\left(x^{(0)}\right) = r_0 = b - Ax^{(0)}$$

共轭梯度下降方向的构造

❶ 给定 $x^{(0)}$, 取 $\Phi(x)$ 在 $x^{(0)}$ 处的负梯度方向:

$$p_1 = -\nabla\Phi\left(x^{(0)}\right) = r_0 = b - Ax^{(0)}$$

❷ 以 p_1 为下降方向, 计算近似解 $x^{(1)}$:

$$x^{(1)} = x^{(0)} + \alpha_1 p_1, \text{ 其中 } \alpha_1 = \frac{p_1^{\mathsf{T}} r_0}{p_1^{\mathsf{T}} A p_1}$$

共轭梯度下降方向的构造

❶ 给定 $x^{(0)}$, 取 $\Phi(x)$ 在 $x^{(0)}$ 处的负梯度方向:

$$p_1 = -\nabla\Phi\left(x^{(0)}\right) = r_0 = b - Ax^{(0)}$$

❷ 以 p_1 为下降方向, 计算近似解 $x^{(1)}$:

$$x^{(1)} = x^{(0)} + \alpha_1 p_1, \text{ 其中 } \alpha_1 = \frac{p_1^{\mathsf{T}} r_0}{p_1^{\mathsf{T}} A p_1}$$

❸ 计算 $\Phi(x)$ 在 $x^{(1)}$ 处的负梯度方向: $-\nabla\Phi\left(x^{(1)}\right) = r_1 = b - Ax^{(1)} = r_0 - \alpha_1 A p_1$, 将其与 p_1 进行 A-共轭化, 得到下降方向 p_2:

$$p_2 = r_1 + \beta_1 p_1, \text{ 其中 } \beta_1 = -\frac{r_1^{\mathsf{T}} A p_1}{p_1^{\mathsf{T}} A p_1}$$

共轭梯度下降方向的构造

1. 给定 $x^{(0)}$, 取 $\Phi(x)$ 在 $x^{(0)}$ 处的负梯度方向:

$$p_1 = -\nabla\Phi\left(x^{(0)}\right) = r_0 = b - Ax^{(0)}$$

2. 以 p_1 为下降方向, 计算近似解 $x^{(1)}$:

$$x^{(1)} = x^{(0)} + \alpha_1 p_1, \text{ 其中 } \alpha_1 = \frac{p_1^{\mathsf{T}} r_0}{p_1^{\mathsf{T}} A p_1}$$

3. 计算 $\Phi(x)$ 在 $x^{(1)}$ 处的负梯度方向: $-\nabla\Phi\left(x^{(1)}\right) = r_1 = b - Ax^{(1)} = r_0 - \alpha_1 A p_1$, 将其与 p_1 进行 A-共轭化, 得到下降方向 p_2:

$$p_2 = r_1 + \beta_1 p_1, \text{ 其中 } \beta_1 = -\frac{r_1^{\mathsf{T}} A p_1}{p_1^{\mathsf{T}} A p_1}$$

- 以此类推, 我们就可以得到相互 A-共轭的下降方向 $p_1, p_2, \ldots, p_k, \ldots$

怎么简化计算量?

在计算 p_{k+1} 时, 需要将 r_k 与所有 $p_1, p_2, \ldots, p_k$ 进行 A-共轭化, 计算成本很大.

怎么简化计算量?

在计算 p_{k+1} 时, 需要将 r_k 与所有 $p_1, p_2, \ldots, p_k$ 进行 A-共轭化, 计算成本很大.

事实上, 由于 A 对称正定, 我们只需将 r_k 与 p_k 进行 A-共轭化即可, 即

$$p_{k+1} = r_k + \beta_k p_k, \quad \text{其中} \quad \beta_k = -\frac{r_k^{\mathsf{T}} A p_k}{p_k^{\mathsf{T}} A p_k}.$$

这样不仅可以简化计算, 同时可以证明 p_{k+1} 与 $p_1, p_2, \ldots, p_{k-1}$ 都 A-共轭.

共轭搜索方向的性质

定理 设 $A \in \mathbb{R}^{n\times n}$ 对称正定, 共轭梯度法迭代 m 步后 $(m < n)$, 对任意 k $(k \leq m)$ 有

(1) $r_k^{\mathsf{T}} r_i = 0, \quad i = 0, 1, 2, \ldots, k-1;$

(2) $r_k^{\mathsf{T}} p_i = 0, \quad i = 1, 2, \ldots, k;$

(3) $p_{k+1}^{\mathsf{T}} A p_i = 0, \quad i = 1, 2, \ldots, k;$

(4) $\mathrm{span}\{r_0, r_1, \ldots, r_k\} = \mathrm{span}\{p_1, p_2, \ldots, p_{k+1}\} = \mathrm{span}\{r_0, Ar_0, \ldots, A^k r_0\}.$

共轭梯度法迭代格式

于是共轭梯度法可描述为

$$\begin{cases} x^{(k)} = x^{(k-1)} + \alpha_k p_k, \quad \text{其中} \quad \alpha_k = \dfrac{p_k^{\mathsf{T}} r_{k-1}}{p_k^{\mathsf{T}} A p_k}, \\ r_k = b - A x^{(k)} = r_{k-1} - \alpha_k A p_k, \\ p_{k+1} = r_k + \beta_k p_k, \quad \text{其中} \quad \beta_k = -\dfrac{r_k^{\mathsf{T}} A p_k}{p_k^{\mathsf{T}} A p_k}, \end{cases} \qquad k = 1, 2, \ldots,$$

进一步简化计算: 利用 r_k 的正交性和 p_k 的 A-共轭性, 可得

推论 共轭梯度法中的 α_k 和 β_k 可以通过下面的公式计算

$$\alpha_k = \frac{r_{k-1}^{\mathsf{T}} r_{k-1}}{p_k^{\mathsf{T}} A p_k}, \quad \beta_k = \frac{r_k^{\mathsf{T}} r_k}{r_{k-1}^{\mathsf{T}} r_{k-1}}.$$

共轭梯度法的算法描述

算法 共轭梯度法 (Conjugate Gradient Algorithm)

1: **给定初值** $x^{(0)}$

2: **计算** $r_0 = b - Ax^{(0)}$, **并令** $p_1 = r_0$, $k = 1$

3: **while** not converge **do**

4: 　　**计算** $x^{(k)} = x^{(k-1)} + \alpha_k p_k$, 　**其中** 　$\alpha_k = \dfrac{r_{k-1}^{\mathsf{T}} r_{k-1}}{p_k^{\mathsf{T}} A p_k}$

5: 　　**计算** $r_k = r_{k-1} - \alpha_k A p_k$

6: 　　**计算** $p_{k+1} = r_k + \beta_k p_k$, 　**其中** 　$\beta_k = \dfrac{r_k^{\mathsf{T}} r_k}{r_{k-1}^{\mathsf{T}} r_{k-1}}$

7: **end while**

共轭梯度法的收敛性

定理 设 $A \in \mathbb{R}^{n\times n}$ 对称正定, 则对任意初值 $x^{(0)}$, 共轭梯度法都收敛, 且

$$\frac{\left\|x^{(k)} - x_*\right\|_A}{\left\|x^{(0)} - x_*\right\|_A} \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k,$$

其中 κ 为 A 的谱条件数, 范数 $\|x\|_A \triangleq \sqrt{x^{\mathsf{T}}Ax}$.

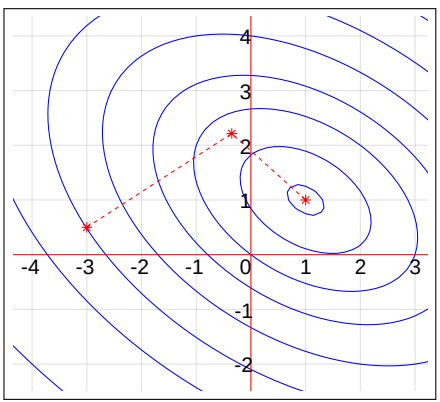
由此可知, 共轭梯度法每次迭代的误差下降率为 $\dfrac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}$.

共轭梯度法举例（一）

例 用共轭梯度法求解 $Ax = b$, 其中 $A = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}$, $b = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$, 初值 $x^{(0)} = \begin{bmatrix} -3 \\ 0.5 \end{bmatrix}$.

k	x	relres
1	$[-0.3498, 2.2148]$	2.70e-01
2	$[1.0000, 1.0000]$	4.39e-17

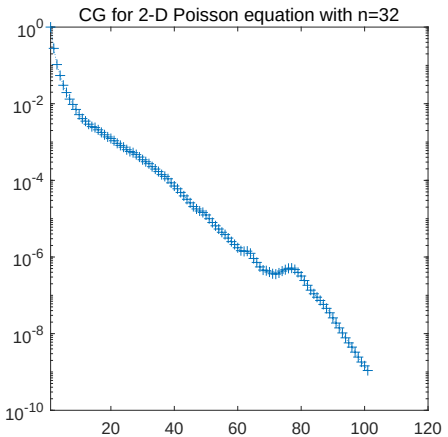
(Iter_CG_01.m)

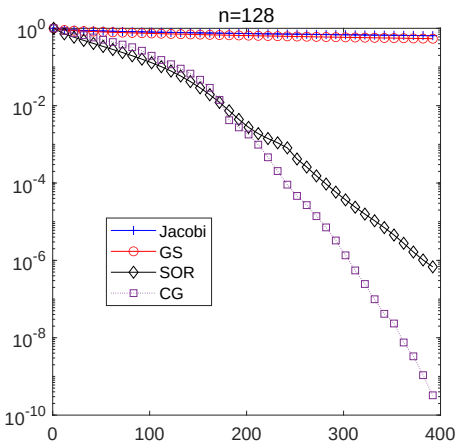


共轭梯度法举例（二）

例 用共轭梯度法求解离散二维 Poission 方程.

(Iter_CG_gallery.m), (Iter_CG_Jacobi_GS_SOR_Poisson2D.m)





Kaczmarz 方法

(超定线性方程组, $A \in \mathbb{R}^{m\times n}$)

Kaczmarz 方法

在 Kaczmarz 方法中, 我们选取 A 的行作为搜索方向.

- 用 $\hat{a}_i$ 表示 A 的第 i 行的转置, 即 A^{T} 的第 i 列: $\hat{a}_i = A^{\mathsf{T}} e_i = A(i,:)^{\mathsf{T}}$, 于是 Kaczmarz 方法的迭代格式可以描述为

$$x^{(k+1)} = x^{(k)} + \alpha_{i_k} \hat{a}_{i_k}, \quad k = 0, 1, 2, \ldots,$$

  其中 $i_k \in \{1, 2, \ldots, n\}$, 即每次迭代选取 A 的某一行作为搜索方向.
- 不同的选取方法就会导致不同的算法, 比如按顺序选取, 即第一次迭代取第一行, 第二次迭代取第二行, 以此类推, n 次迭代后再回到第一行, 循环重复, 即 $i_k = (k+1) \bmod n$.
- 另一种常用的选取方法是按一定的概率随机选取其中的一行, 这就是随机 Kaczmarz 方法.

步长

我们通过极小化误差 $\|x^{(k+1)} - x_*\|_2$ 来确定步长 α_{i_k} (板书)

步长

我们通过极小化误差 $\|x^{(k+1)} - x_*\|_2$ 来确定步长 α_{i_k} (板书)

$$\alpha_{i_k} = \frac{b_{i_k} - \hat{a}_{i_k}^{\mathsf{T}} x^{(k)}}{\|\hat{a}_{i_k}\|^2}.$$

- 由此可见, Kaczmarz 方法的每次迭代只依赖 A 的某一行 (搜索方向和步长), 而不是整个矩阵.
- 另外, 需要注意的是, Kaczmarz 方法中的步长可以是负数.

原始 Kaczmarz 方法

原始的 Kaczmarz 方法 (1937) 是按顺序循环选取 A 的行作为搜索方向, 即 $i_k = (k+1) \bmod n$. 所以算法可以描述为

$$x^{(k+1)} = x^{(k)} + \frac{b_{i_k} - \hat{a}_{i_k}^{\mathsf{T}} x^{(k)}}{\|\hat{a}_{i_k}\|^2} \hat{a}_{i_k}, \quad \text{其中 } i_k = (k+1) \bmod n,\ k = 0, 1, 2, \ldots.$$

✍ 这种选取方法简单直观, 而且能保证算法收敛, 但 **收敛速度** 较难估计.

算法描述

算法 Kaczmarz **方法**

1: **给定初值** $x^{(0)}$, **令** $k=0$
2: **while** not convergence **do**
3: $\quad i_k = (k+1) \bmod n$
4: $\quad x^{(k+1)} = x^{(k)} + \dfrac{b_{i_k} - \hat{a}_{i_k}^{\mathsf{T}} x^{(k)}}{\|\hat{a}_{i_k}\|^2} \hat{a}_{i_k}$
5: $\quad k = k+1$
6: **end while**

收敛性

根据 α_{i_k} 的最优性, 我们可得

$$\|x^{(k+1)} - x_*\|_2^2 = \|x^{(k)} - x_*\|_2^2 - \frac{(e_i^{\mathsf{T}} r_k)^2}{\|\hat{a}_{i_k}\|^2}.$$

因此误差是严格递减的, 除非 $e_i^{\mathsf{T}} r_k = 0$ (即 r_k 的第 i 个分量为 0).

由此可知, 若线性方程组相容, 则 **Kaczmarz 方法是收敛的**.

收敛性

事实上, 将每次迭代格式看作是一个映射, 即

$$x^{(k+1)} = x^{(k)} + \frac{b_{i_k} - \hat{a}_{i_k}^{\mathsf{T}} x^{(k)}}{\|\hat{a}_{i_k}\|^2} \hat{a}_{i_k} \quad \Longrightarrow \quad f_i(x) = x + \frac{b_i - \hat{a}_i^{\mathsf{T}} x}{\|\hat{a}_i\|^2} \hat{a}_i, \quad i = 1, 2, \ldots, m.$$

则经过一轮迭代 (即 m 次) 后, 记得到的近似解为 $x^{(k+1)}$, 则

$$x^{(k+1)} = F(b, x^{(k)}),$$

其中 $F = f_1 \circ f_2 \circ \cdots \circ f_m$.

可以证明, 存在矩阵 $Q \in \mathbb{R}^{n\times n}$ 和 $R \in \mathbb{R}^{n\times m}$, 使得

$$F(b, x) = Qx + Rb,$$

其中 R 只与 A 有关. 在一定条件下可以给出该算法的收敛性.

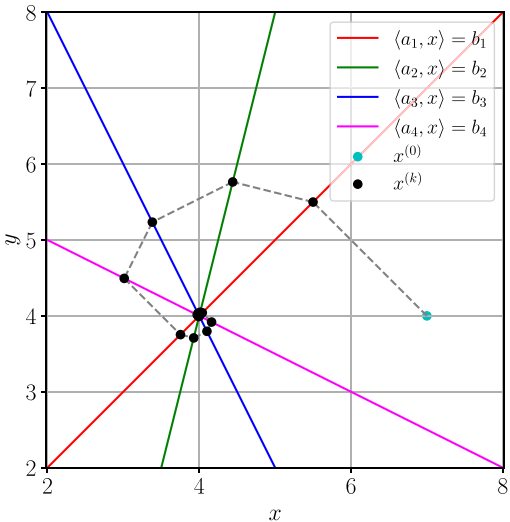
Kaczmarz 方法

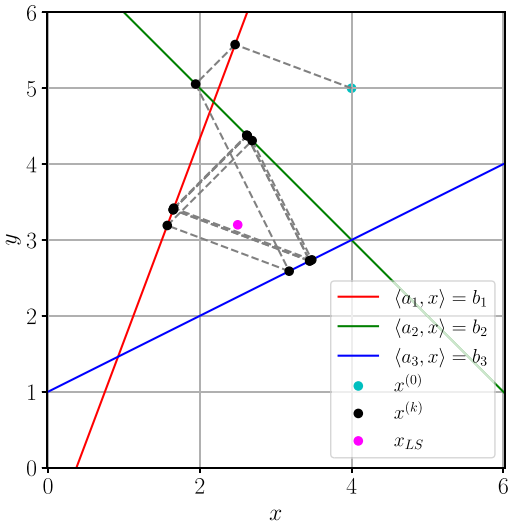
Kaczmarz 方法的几何意义

Kaczmarz 方法有着明确的几何意义.

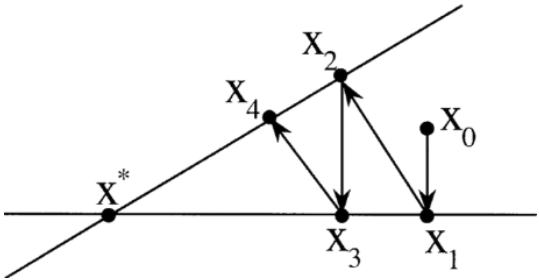
- 事实上, 假定方程组是相容的, 则方程组的精确解 x_* 可以看作是 m 个超平面 $\mathbb{H}_i = \{x : \hat{a}_i^{\mathsf{T}} x = b_i\}$ $(i = 1, 2, \ldots, m)$ 的交点,
- Kaczmarz 方法则是不断将当前迭代解依次映射 (正交投影) 到这些超平面, 从而使得迭代解越来越接近交点 (即 x_*).
  下页左图以 $m = 4$, $n = 2$ 为例, 展示了 Kaczmarz 方法的迭代过程, 其中四条线分别代表四个超平面 $\mathbb{H}_i$.
- 但如果方程是不相容的, 则 Kaczmarz 方法可能无法收敛到最小二乘解, 见下页右图.

Kaczmarz 方法


$\langle a_1, x\rangle = b_1$
$\langle a_2, x\rangle = b_2$
$\langle a_3, x\rangle = b_3$
$\langle a_4, x\rangle = b_4$
$x^{(0)}$
$x^{(k)}$
x
y


$\langle a_1, x\rangle = b_1$
$\langle a_2, x\rangle = b_2$
$\langle a_3, x\rangle = b_3$
$x^{(0)}$
$x^{(k)}$
x_{LS}
x
y

Kaczmarz 方法 ($m = n = 2$)



Kaczmarz 方法

Kaczmarz 方法的特点

- 每次迭代的运算量非常小, 特别是当 A 非常稀疏时, 计算 α_k 的成本很低, 而且由于 $\hat{a}_{i_k}$ 的稀疏性, 每次迭代只需更新 $x^{(k)}$ 的个别分量;
- 由于是依次使用 A 的行, 在开始迭代时并不需要整个 A 的数据, 因此属于在线 (online) 算法, 即可以一边收集数据一边实时求解, 这也是 Kaczmarz 方法在医学图像处理 (比如计算机断层扫描, CT) 中比较流行的原因 (在某些精确 3D CT 扫描中每秒可能会产生上百 G 的数据).
- Kaczmarz 方法可以用来求解相容的超定 (overdetermined) 或欠定 (underdetermined) 线性方程组, 即 $A \in \mathbb{R}^{m\times n}$, 其中 $m > n$ 或 $m < n$.

Kaczmarz 方法

Kaczmarz 方法与 Gauss-Seidel 方法

事实上, Kaczmarz 方法可以看成将 Gauss-Seidel 方法作用到第二类正规方程组上

$$AA^{\mathsf{T}}y = b, \quad x = A^{\mathsf{T}}y, \quad b \in \mathrm{Ran}(A).$$

松弛 Kaczmarz 方法

$$x^{(k+1)} = x^{(k)} + \frac{b_{i_k} - \hat{a}_{i_k}^{\mathsf{T}} x^{(k)}}{\|\hat{a}_{i_k}\|^2} \hat{a}_{i_k} \quad \Longrightarrow \quad x^{(k+1)} = x^{(k)} + \omega_k \frac{b_{i_k} - \hat{a}_{i_k}^{\mathsf{T}} x^{(k)}}{\|\hat{a}_{i_k}\|^2} \hat{a}_{i_k}$$

收敛性, Herman, Lent & Lutz 1978

如果线性方程组相容, 则当松弛参数满足

$$0 < \liminf_{k\to\infty} \omega_k < \limsup_{k\to\infty} \omega_k < 2,$$

则松弛 Kaczmarz 方法收敛.

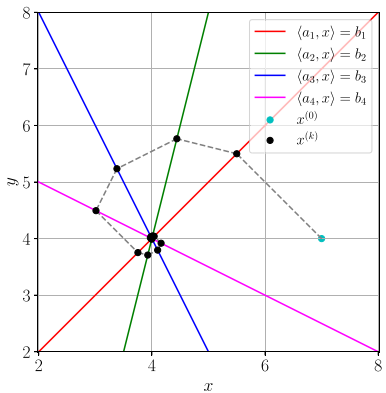
不相容情形, Censor, Eggermoent & Gordon 1983

当初值满足收敛条件情况下, 迭代序列能够收敛到最小范数加权最小二乘解.

替代超平面 Kaczmarz 方法

由正交投影性质可知

$$x^{(k+1)} = \operatorname*{argmin}_{x \in \mathbb{H}_{i_k}} \|x - x^{(k)}\|_2^2 \quad \Longrightarrow \quad \begin{aligned} & x^{(k+1)} = \operatorname{argmin} \|x - x^{(k)}\|_2^2, \\ & \text{s.t. } \eta_k^{\mathsf{T}}(Ax - b) = 0 \end{aligned}$$



$$x^{(k+1)} = x^{(k)} + \frac{\eta_k^{\mathsf{T}}(b - Ax^{(k)})}{\|A^{\mathsf{T}}\eta_k\|_2^2} A^{\mathsf{T}}\eta_k$$

其中 η_k 可以是任意随机变量.

Kaczmarz 方法

- Gauss Kaczmarz 方法: η_k 满足正态分布 (Gauss 抽样)
- 随机 Kaczmarz 方法: $\eta_k = e_{i_k}$, 其中 i_k 满足随机分别
- (贪婪) 分块 Kaczmarz 方法: $\eta_k = \sum_{i\in\tau_k}(e_i^{\mathsf{T}} r_k)e_i$, 取残量中数值大的部分.
- 基于残差的替代超平面 Kaczmarz 方法: $\eta_k = r_k$

加速/斜投影 Kaczmarz 方法

$$\eta_k = r_k + \beta_k \eta_{k-1}, \quad \beta_k = \underset{\beta}{\operatorname{argmin}} \|x^{(k+1)} - x_*\|_2^2 = -\frac{\eta_{k-1}^{\mathsf{T}} A A^{\mathsf{T}} r_k}{\|A^{\mathsf{T}} \eta_{k-1}\|_2^2}$$

斜投影 Kaczmarz 方法

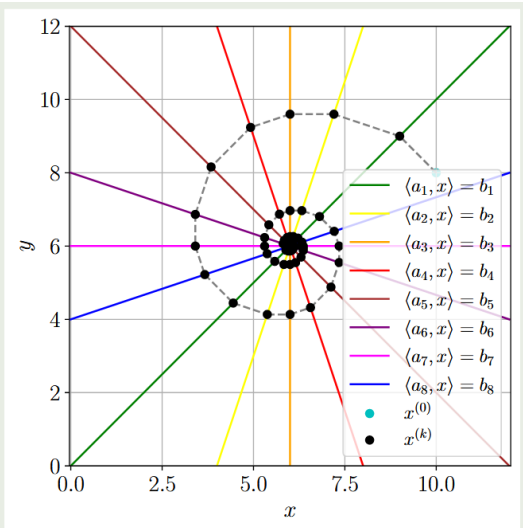
$$x^{(k+1)} = x^{(k)} + \alpha_k A^{\mathsf{T}} \eta_k = x^{(k)} + \alpha_k A^{\mathsf{T}} (r_k + \beta_k \eta_{k-1})$$

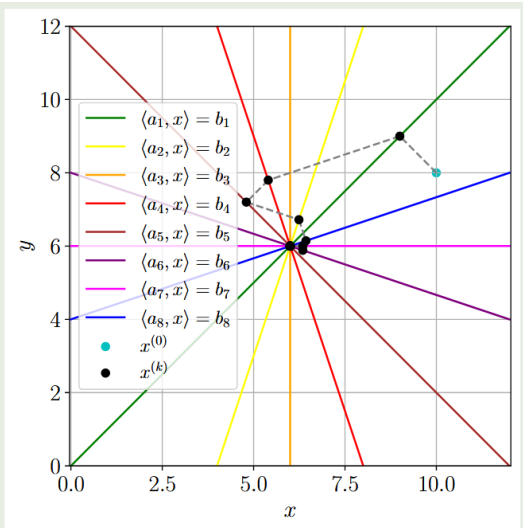
$$\alpha_k = \frac{\eta_k^{\mathsf{T}} (b - A x^{(k)})}{\|A^{\mathsf{T}} \eta_k\|_2^2}$$

随机 Kaczmarz 方法

按不同的概率分布选取行

随机 Kaczmarz 方法





简单随机 Kaczmarz 方法

简单随机 Kaczmarz 方法 (Simple Randomized Kaczmarz, SRK) : 均匀分布

$$\mathbb{P}(i_k = i) = \frac{1}{m}$$

Schmidt 2015

设 $Ax = b$ 是相容的, 给定初值 $x^{(0)} \in \mathrm{Ran}(A^{\mathsf{T}})$, 由简单随机 Kaczmarz 方法生成的迭代序列 $\{x^{(k)}\}$ 在期望意义下收敛到最小范数解 $x_* = A^{\dagger}b$, 且有

$$\mathbb{E}\big[\|x^{(k+1)} - x_*\|_2^2\big] \leq \left(1 - \frac{\sigma_{\min}^2(A)}{m\|A\|_{\infty}^2}\right) \|x^{(k)} - x_*\|_2^2.$$

简单随机 Kaczmarz 方法

注记

在实际应用中, 我们可以通过设置准随机数 (quasirandom numbers) 来随机选取行, 从而克服普通随机数序列在样本点上分布不均匀的问题: 普通的随机数序列 (如伪随机数序列) 往往会产生一些局部区域的样本点过于密集, 而其他区域则样本点稀疏; 准随机序列则通过特定的算法生成, 使得这些样本点分布更均匀, 从而在数值计算中能提高计算精度, 常用的有 Sobol 序列与 Halton 序列等.

基于行范数的概率准则

随机 Kaczmarz 方法 (Randomized Kaczmarz, RK): **以行范数为概率准则**

$$\mathbb{P}(i_k = i) = \frac{\|\hat{a}_i\|_2^2}{\|A\|_F^2}$$

Strohmer & Vershynin 2007

在一定条件下, 可以证明 RK 方法在期望意义下是线性收敛的, 且

$$\mathbb{E}\big[\|x^{(k+1)} - x_*\|_2^2\big] \leq \left(1 - \frac{\sigma_{\min}^2(A)}{\|A\|_F^2}\right) \|x^{(k)} - x_*\|_2^2,$$

贪婪随机 Kaczmarz 方法

贪婪随机 Kaczmarz 方法 (Greedy Randomized Kaczmarz, GRK): 先确定一个子集 $\mathbb{Z}_k \subset \{1, 2, \ldots, m\}$, 然后在子集中按残量大小作为概率准则

$$\mathbb{P}(i_k = i) = \frac{|e_i^{\mathsf{T}} \tilde{r}_k|^2}{\|\tilde{r}_k\|_2^2}$$

其中

$$\tilde{r}_k(i) = \begin{cases} r_k(i), & i \in \mathbb{Z}_k \\ 0, & \text{otherwise.} \end{cases}$$

与 SRK 和 RK 相比, GRK 通常具有更快的收敛速度.

分块 Kaczmarz 方法

分块 Kaczmarz 方法

对 A 的行和 b 进行分块

$$A = \begin{bmatrix} A_{\tau_1} \\ A_{\tau_2} \\ \vdots \\ A_{\tau_p} \end{bmatrix}, \quad b = \begin{bmatrix} b_{\tau_1} \\ b_{\tau_2} \\ \vdots \\ b_{\tau_p} \end{bmatrix},$$

其中 $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_p\}$ 是指标集 $\{1, 2, \ldots, m\}$ 的一个划分.

分块 Kaczmarz 方法: **依次** 在每个分块上进行投影

$$x^{(k+1)} = x^{(k)} + A_{\tau_{i_k}}^{\dagger}(b_{\tau_{i_k}} - A_{\tau_{i_k}} x^{(k)}), \quad i_k = (k+1) \mod p$$

随机分块 Kaczmarz 方法

对 A 的行和 b 进行分块

$$A = \begin{bmatrix} A_{\tau_1} \\ A_{\tau_2} \\ \vdots \\ A_{\tau_p} \end{bmatrix}, \quad b = \begin{bmatrix} b_{\tau_1} \\ b_{\tau_2} \\ \vdots \\ b_{\tau_p} \end{bmatrix},$$

其中 $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_p\}$ 是指标集 $\{1, 2, \ldots, m\}$ 的一个划分.

随机分块 Kaczmarz 方法: 每次 **随机** 选一个分块, 在上面进行投影

$$x^{(k+1)} = x^{(k)} + A_{\tau_{i_k}}^{\dagger}(b_{\tau_{i_k}} - A_{\tau_{i_k}} x^{(k)})$$

随机平均 Kaczmarz 方法

Randomized Kaczmarz with Averaging, RKA (2021): **每次随机选取多行, 然后做平均, 适合并行计算**

$$x^{(k+1)} = x^{(k)} + \frac{1}{|\tau_k|} \sum_{i \in \tau_k} w_i \frac{b_i - \hat{a}_i^{\mathsf{T}} x^{(k)}}{\|\hat{a}_i\|^2} \hat{a}_i$$

更多一维投影方法

- (随机) 拓展 Kaczmarz 方法
- (随机) 非线性 Kaczmarz 方法
- 求解张量问题的各类 (随机) Kaczmarz 方法
- 量子 Kaczmarz 方法, Quantum Kaczmarz Algorithm
- (随机) 坐标下降法, Randomized Coordinate Descent Method
- (随机) Cimmino 方法
- Sketch-and-project methods ($\eta_k \to S_k$, sketching matrix)


谢 谢
THANK YOU