MATH20007h.01 数值算法与案例分析 (H) Fall 2025

Lecture 12 共轭梯度法 — 2025.12.01

教授：邵美悦

Scribe: 路人甲

目录

1 共轭梯度法 (CG)

1.1 SD 方法回顾

在上节课，我们讲解了利用最速下降法 (SD) 来求解线性方程组 $Ax = b$，其中 A 是 Hermite 矩阵。这一类方法的基本思想是将求解线性方程组问题转化为二次优化问题：

$$\min_{x\in\mathbb{C}^n} \phi(x) = (x - x_\star)^\star A(x - x_\star).$$

SD 方法的迭代格式为

$$x_{k+1} = x_k + \alpha_k r_k, \quad \alpha_k = \frac{r_k^\star r_k}{r_k^\star A r_k},$$

其中 $r_k = b - Ax_k$ 是当前点 x_k 处的梯度。这一方法的几何本质是：我们每一步迭代都最小化误差 $x_k - x_\star$ 的 $A-$范数。在 $A-$内积下，我们每次在负梯度 $-r_k$ 的方向上走到极致，因此下一步的误差 $x_{k+1} - x_\star$ 一定要与 r_k 正交，如图 1 所示。

利用这种方法，我们也可以推导出步长 α_k：它一定是 $x_k - x_\star$ 在 r_k 方向上的投影长度——在 $A-$内

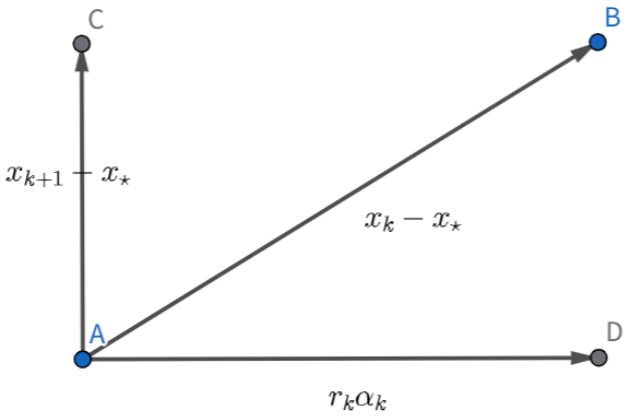

图 1: 最速下降法 (SD) 的几何解释

积意义下的投影长度。因此

$$\begin{aligned} x_{k+1} - x_\star &= x_k - x_\star - r_k \frac{\langle r_k, x_k - x_\star \rangle_A}{\langle r_k, r_k \rangle_A} \\ &= x_k - x_\star - r_k \frac{r_k^\star A(x_k - x_\star)}{r_k^\star A r_k} \\ &= x_k - x_\star + r_k \frac{r_k^\star r_k}{r_k^\star A r_k}. \end{aligned}$$

因此

$$\alpha_k = \frac{r_k^\star r_k}{r_k^\star A r_k}.$$

共轭梯度法和最速下降法非常类似，但它在每一步迭代中并不是沿着当前梯度方向前进，而是沿着一组 $A-$共轭方向前进。我们将在接下来的内容中详细介绍共轭梯度法。

1.2 共轭梯度法的基本思想

SD 的一大缺点是：它每一步迭代都贪心地选择局部最优方向，但这种选择并不一定是全局最优的，因此它的下降是“锯齿式”的，会重复在已经探索过的方向上搜索，因此在 n 步以内不一定收敛。而 Krylov 子空间方法则是全局最优的，因此它在 n 步以内一定收敛。

共轭梯度法 (CG) 结合了 SD 和 Krylov 子空间方法。它的优化目标和 SD 一样，都是最小化二次型

$$\phi(x) = (x - x_\star)^\star A(x - x_\star),$$

亦即最小化误差的 $A-$范数——这和 SD 一样。但与 SD 不同的是，但它每一步做的都是全局搜索：在第 k 步迭代中，它会在仿射空间 $x_0 + \mathcal{K}_k(A, r_0)$ 中寻找最优解

$$x_k = \arg \min_{x \in x_0 + \mathcal{K}_k(A, r_0)} \phi(x).$$

因此，CG 方法在 n 步以内一定收敛。

现在，我们来讨论一下 CG 方法到底是怎么收敛的。我们知道，Krylov 子空间

$$\mathcal{K}_k(A, r_0) = \text{span}\{r_0, Ar_0, A^2r_0, \ldots, A^{k-1}r_0\}$$

我们可以用数学归纳法证明

$$\mathcal{K}_k(A, r_0) = \text{span}\{r_0, r_1, r_2, \ldots, r_{k-1}\},$$

其中 $r_i = b - Ax_i$ 是第 i 步迭代的残差。这个公式我们在上节课已经证明过了，不再赘述。为了最小化误差的 $A-$范数，假定我们每一步迭代都能找到一组 $A-$内积下的正交基

$$\mathcal{K}_k(A, r_0) = \text{span}\{p_0, p_1, \ldots, p_{k-1}\},$$

这些正交基被称为是 **共轭方向**。我们可以将初始误差 $x_\star - x_0$ 在这组基下展开：

$$x_\star - x_0 = p_0\alpha_0 + p_1\alpha_1 + p_2\alpha_2 + \cdots + p_{n-1}\alpha_{n-1}.$$

我们在第 k 步迭代时，只知道前 $k-1$ 个正交基，因此我们最多只能消去前 $k-1$ 项。一个最朴素的想法是：我们每一步都把当前误差在当前基方向上的投影消去掉。这样一来，第 k 步迭代时，误差就变成了

$$\begin{aligned}
x_\star - x_1 &= p_1\alpha_1 + p_2\alpha_2 + \cdots + p_{n-1}\alpha_{n-1},\\
x_\star - x_2 &= p_2\alpha_2 + p_3\alpha_3 + \cdots + p_{n-1}\alpha_{n-1},\\
&\vdots\\
x_\star - x_k &= p_k\alpha_k + p_{k+1}\alpha_{k+1} + \cdots + p_{n-1}\alpha_{n-1}.
\end{aligned}$$

因此，到第 n 步迭代时，误差就被完全消去，算法收敛。

直观地看来，SD 总是沿着当前最陡的梯度方向（垂直于等高线）前进，但因其在标准欧氏距离下垂直，而在实际地形中（$A-$内积下）并不垂直，导致左右振荡，进展缓慢。而 CG 则通过调整方向，使其在地形本身的度量（$A-$内积）下相互垂直，从而每一步都沿着“适配地形”的方向前进，避免重复，直达谷底。

1.3 共轭方向的构造

现在，我们来讨论一下算法实现的细节。我们的算法的大体框架如算法 1 所示。

现在，我们的目标就是确定步长 α_k 和共轭方向 p_k 的计算方法。步长 α_k 非常好确定：如果我们要最小化误差的 $A-$范数，那么我们只需要让下一步的误差 $x_{k+1} - x_\star$ 与当前共轭方向 p_k 正交即可。如图 2 所示。

Algorithm 1: 共轭梯度法 (CG) 的朴素实现

Input: Hermite 正定阵 $A \in \mathbb{C}^{n\times n}$，右端项 $b \in \mathbb{C}^n$，初始点 $x_0 \in \mathbb{C}^n$，容忍误差 $\varepsilon > 0$

Output: 线性方程组 $Ax = b$ 的近似解 x_n

```
p_0 = r_0 = b − Ax_0;
for i = 1 : n do
    α_i =?;
    x_{k+1} = x_k + p_k α_k;
    r_{k+1} = b − Ax_{k+1};
    if ‖r_{k+1}‖ < ε then
        break;
    end
    p_{k+1} =?;
end
```

于是，类似 SD 方法的推导，我们有

$$\alpha_k = \frac{p_k^\star r_k}{p_k^\star A p_k}.$$

同时，残差的更新公式也可以写成

$$r_{k+1} = r_k - Ap_k\alpha_k.$$

这个公式我们在上节课也已经证明过了，不再赘述。

于是问题的关键就落在了共轭方向的构造。第一个正交基 p_0 很容易构造：我们直接让它和初始残差 r_0 相等即可

$$p_0 = r_0 = b - Ax_0.$$

接下来，我们需要构造后续的共轭方向 $p_1, p_2, \ldots$。我们希望这些方向满足

$$\langle p_i, p_j\rangle_A = p_i^\star A p_j = \delta_{i,j}.$$

一个比较自然的想法是：我们可以用 Gram-Schmidt 正交化的方法来构造这些共轭方向。例如，假设我们已经得到了前 k 个共轭方向 $p_0, p_1, \ldots, p_{k-1}$，那么我们可以取

$$p_k = r_k - \sum_{i=0}^{k-1} p_i \frac{\langle r_k, p_i\rangle_A}{\langle p_i, p_i\rangle_A},$$

然而，这样做的问题是：每一步迭代我们都需要计算前面所有的共轭方向，计算量非常大。我们有没有办法可以简化运算呢？

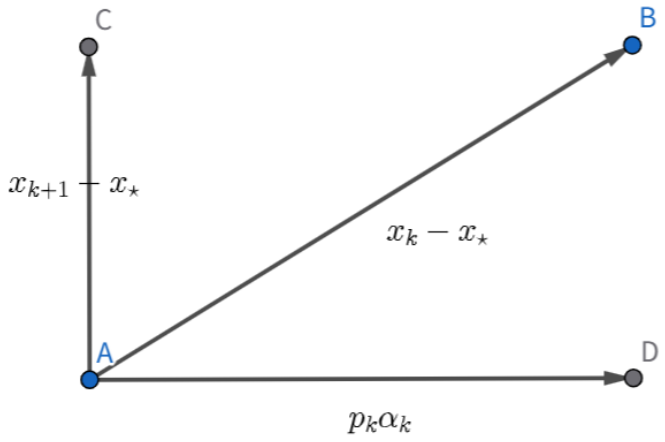

图 2: CG 步长的选择

我们先来看第一项：

$$p_1 = r_1 + p_0 \boxed{\frac{p_0^\star A r_1}{p_0^\star A p_0}} = r_1 + p_0\beta_0.$$

由迭代过程，我们知道 $p_0 = r_0$ 且 $r_1 = r_0 - Ap_0\alpha_0 = r_0 - Ar_0\alpha_0$，因此

$$\begin{aligned}
\beta_0 &= -\frac{r_0^\star A^\star r_1}{r_0^\star A r_0} \\
&= \frac{(r_1 - r_0)^\star}{\alpha_0} \cdot \frac{r_1}{r_0^\star A r_0} \\
&= -\frac{r_0^\star r_1 - r_1^\star r_1}{\alpha_0 r_0^\star A r_0} \\
&= -\frac{r_0^\star r_1}{\alpha_0 r_0^\star A r_0} + \frac{r_1^\star r_1}{\alpha_0 r_0^\star A r_0} \\
&= -\frac{r_0^\star (r_0 - A r_0\alpha_0)}{\alpha_0 r_0^\star A r_0} + \frac{r_1^\star r_1}{\alpha_0 r_0^\star A r_0}
\end{aligned}$$

把 $\alpha_0 = (r_0^\star r_0)/(r_0^\star A r_0)$ 代入上式的第一项的分子，我们得到

$$r_0^\star (r_0 - A r_0\alpha_0) = r_0^\star r_0 - r_0^\star A r_0 \cdot \frac{r_0^\star r_0}{r_0^\star A r_0} = 0.$$

而分母

$$\alpha_0 r_0^\star A r_0 = \frac{r_0^\star r_0}{r_0^\star A r_0} \cdot r_0^\star A r_0 = r_0^\star r_0 \neq 0.$$

因此

$$\beta_0 = \frac{r_1^\star r_1}{r_0^\star r_0}.$$

在上面，我们得到了一个重要的洞察：**计算新的共轭方向时，我们只需要用到前一个残差，而不需要用到之前的所有残差**。这为我们的计算提供了极大的便利。

上面的推理有一个副产品：残量在标准正交基下正交——$r_0^\star r_1 = 0$. 这似乎不大妙，因为我们知道 SD 方法中残量也是正交的。但这并无大碍，因为 CG 的搜索方向是 p_k，不是 r_k——我们的眼睛盯着的是 $A-$内积下的正交基，我们只要保证我们在 $A-$内积下没有重复搜索就行了。

仿照上面的过程，我们可以证明：

$$\beta_k = \frac{r_{k+1}^\star r_{k+1}}{r_k^\star r_k} \tag{1}$$

$$p_{k+1} = r_{k+1} + p_k \beta_k \tag{2}$$

$$r_i^\star r_j = 0, \quad \forall i \neq j \tag{3}$$

$$\alpha_k = \frac{r_k^\star r_k}{p_k^\star A p_k} \tag{4}$$

其中 (4) 是 (3) 的直接推论：

$$\alpha_k = \frac{p_k^\star r_k}{p_k^\star A p_k} = \frac{r_k^\star r_k}{p_k^\star A p_k}.$$

由于 p_k 是 r_k 和之前的残差 r_i 的线性组合，而 r_k 与之前的残差正交，因此 $p_k^\star r_k = r_k^\star r_k$. 因此 (4) 成立。

现在，我们就来证明剩余三个公式。我们用数学归纳法。假设对于前 k 步迭代，(1) (2) (3) 都成立。那么我们来证明第 $k+1$ 步迭代时，这三个公式也成立。

首先，我们来证明 (3)。我们知道第 $k+1$ 步迭代的残差为

$$r_{k+1} = r_k - A p_k \alpha_k.$$

因此，对于任意 $0 \leq j \leq k$，我们有

$$\begin{aligned} r_{k+1}^\star r_j &= (r_k - A p_k \alpha_k)^\star r_j \\ &= r_k^\star r_j - \alpha_k p_k^\star A r_j \\ &= 0. \end{aligned}$$

其中第三个等号是因为：当 $j = k$ 时，两项都是 $r_k^\star r_k$，因此差为零；当 $j < k$ 时，由归纳假设第一项为零，而 p_k 是最后出现的正交基，与之前的残差在 $A-$内积下正交，因此第二项也为零。

接下来，我们来证明 (1) 和 (2)。我们知道

$$p_{k+1} = r_{k+1} - \sum_{i=0}^{k} p_i \frac{\langle r_{k+1}, p_i \rangle_A}{\langle p_i, p_i \rangle_A}.$$

由于 r_{k+1} 与之前的残差在标准内积下正交，而 $A p_i$ 是之前残差 $\{r_0, r_1, \ldots, r_{i+1}\}$ 的线性组合，因此对于 $i < k$，我们有

$$\langle r_{k+1}, p_i \rangle_A = r_{k+1}^\star (A p_i) = 0.$$

因此上式可以简化为

$$p_{k+1} = r_{k+1} - p_k \frac{\langle r_{k+1}, p_k \rangle_A}{\langle p_k, p_k \rangle_A}.$$

再由 $r_{k+1} = r_k - Ap_k\alpha_k$，我们有

$$\langle r_{k+1}, p_k \rangle_A = r_{k+1}^\star A p_k = r_{k+1}^\star \frac{r_k - r_{k+1}}{\alpha_k} = -\frac{r_{k+1}^\star r_{k+1}}{\alpha_k}.$$

分母

$$\langle p_k, p_k \rangle_A = p_k^\star A p_k = \frac{r_k^\star r_k}{\alpha_k}.$$

因此

$$\beta_{k+1} = \frac{r_{k+1}^\star r_{k+1}}{r_k^\star r_k}$$

$$p_{k+1} = r_{k+1} + p_k\beta_k.$$

这就完成了证明。

综上所述，我们得到了共轭梯度法的完整算法，如算法 2 所示。

Algorithm 2: 共轭梯度法 (CG)

Input: Hermite 正定阵 $A \in \mathbb{C}^{n\times n}$，右端项 $b \in \mathbb{C}^n$，初始点 $x_0 \in \mathbb{C}^n$，容忍误差 $\varepsilon > 0$

Output: 线性方程组 $Ax = b$ 的近似解 x_n

$p_0 = r_0 = b - Ax_0$;
for $k = 0 : n-1$ **do**
  $\alpha_k = \frac{r_k^\star r_k}{p_k^\star \boxed{Ap_k}}$;
  $x_{k+1} = x_k + p_k\alpha_k$;
  $r_{k+1} = r_k - \boxed{Ap_k}\alpha_k$;
  if $\|r_{k+1}\| < \varepsilon$ **then**
    break;
  end
  $\beta_k = \frac{r_{k+1}^\star r_{k+1}}{r_k^\star r_k}$;
  $p_{k+1} = r_{k+1} + p_k\beta_k$;
end

算法中框起的部分 $\boxed{Ap_k}$ 在计算中是可以复用的，因此我们在每一步迭代中只需要进行一次矩阵–向量乘法运算。当然，残差项不一定每一步都要按 $r_{k+1} = r_k - Ap_k\alpha_k$ 来计算——事实上，如果我们时不时地用 $r_{k+1} = b - Ax_{k+1}$ 来计算，可以减少数值误差的累积，并有助于恢复残差在标准基下的正交性——不过这会增加一些计算量。

1.4 预条件

在实际应用中，矩阵 A 往往是病态的，这会导致 CG 方法收敛缓慢。为了加速 CG 方法的收敛，我们可以引入预条件矩阵 M，它是一个 Hermite 正定矩阵，并且近似于 A 的逆。我们将原始问题 $Ax = b$ 转化为预条件问题

$$M^{-1}Ax = M^{-1}b.$$

这是很坏的做法，因为 $M^{-1}A$ 很可能不是 Hermite 矩阵，因此 CG 方法不再适用——事实上我们在高代课上证明过，两个 Hermite 矩阵的乘积可以是 $\mathbb{C}^{n\times n}$ 中的任何矩阵。

更好的做法是对原始问题进行对称预条件化：

$$(M^{-1/2}AM^{-1/2})(M^{1/2}x) = M^{-1/2}b.$$

这样，预条件化后的矩阵 $M^{-1/2}AM^{-1/2}$ 仍然是 Hermite 正定矩阵，因此我们可以直接对其应用 CG 方法。我们不妨记 $\tilde{A} = M^{-1/2}AM^{-1/2}$，$\tilde{x} = M^{1/2}x$，$\tilde{b} = M^{-1/2}b$，$\tilde{r} = \tilde{b} - \tilde{A}\tilde{x}$.

问题又来了：如果我们按照算法 2 来实现 PCG 方法，我们需要计算平方根 $M^{1/2}$ 和 $M^{-1/2}$，这在实际中是很难做到的。有没有办法可以避免计算矩阵的平方根呢？我们就需要对 CG 方法做一些修正。预条件化后的 CG 方法称为预条件共轭梯度法 (PCG)。

第一步我们需要计算 $\tilde{r}_0^\star\tilde{r}_0$ 和 $\tilde{p}_0^\star\tilde{A}\tilde{p}_0$. 我们代入这些量的定义，有

$$\tilde{r}_0^\star\tilde{r}_0 = (M^{-1/2}r_0)^\star(M^{-1/2}r_0) = r_0^\star M^{-1}r_0,$$

这是个好兆头，我们把平方根给消掉了。同理，

$$\tilde{p}_0^\star\tilde{A}\tilde{p}_0 = (M^{-1/2}p_0)^\star(M^{-1/2}AM^{-1/2})(M^{-1/2}p_0) = p_0^\star M^{-1}AM^{-1}p_0.$$

因此，递归地做下去，我们可以把所有的平方根都消掉。最终，我们得到 PCG 方法的完整算法，如算法 3 所示。

算法中框起的部分都是可以不断复用的，因此每一步迭代我们都只需要进行三次矩阵–向量乘法运算。

2 收敛性分析

Algorithm 3: 预条件共轭梯度法 (PCG)

Input: Hermite 正定阵 $A \in \mathbb{C}^{n\times n}$，右端项 $b \in \mathbb{C}^n$，预条件矩阵 $M \in \mathbb{C}^{n\times n}$，初始点 $x_0 \in \mathbb{C}^n$，容忍误差 $\varepsilon > 0$

Output: 线性方程组 $Ax = b$ 的近似解 x_n

$r_0 = b - Ax_0$;

$p_0 = M^{-1}r_0$;

for $k = 0 : n-1$ **do**

  $\alpha_k = \dfrac{r_k^\star \boxed{M^{-1}r_k}}{p_k^\star \boxed{Ap_k}}$;

  $x_{k+1} = x_k + p_k\alpha_k$;

  $r_{k+1} = r_k - \boxed{Ap_k}\alpha_k$;

  if $\|r_{k+1}\| < \varepsilon$ **then**

    break;

  end

  $\beta_k = \dfrac{r_{k+1}^\star \boxed{M^{-1}r_{k+1}}}{r_k^\star \boxed{M^{-1}r_k}}$;

  $p_{k+1} = \boxed{M^{-1}r_{k+1}} + p_k\beta_k$;

end