Iterative 6

王胤雅

25114020018

yinyawang25@m.fudan.edu.cn

2025 年 11 月 12 日

PROBLEM I Let a matrix A and its preconditioner M be SPD. Observing that $M^{-1}A$ is self-adjoint with respect to the A inner product, write an algorithm similar to Algorithm 9.1 for solving the preconditioned linear system

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

using the A-inner product. The algorithm should employ only one matrix-by-vector product per CG step.

SOLUTION. $M^{-1}A$ is self-adjiont with A inner product: $(M^{-1}Ax, y)_A = (AM^{-1}Ax, y) = y^TAM^{-1}Ax = y^T(A^T(M^{-1})^T)Ax = (AM^{-1}y)^TAx = (Ax, M^{-1}Ay) = (x, M^{-1}A)_A$

Algorithm 1 Preconditioned Conjugate Gradient using the A-inner product

Given: SPD matrices A and M, initial guess x_0
Compute $r_0 := b - Ax_0$
Compute $z_0 := M^{-1}r_0$
Set $p_0 := z_0$
for $j = 0, 1, 2, \ldots$ until convergence **do**
    Compute $q_j := Ap_j$
    $\alpha_j := \dfrac{(Az_j, z_j)}{(M^{-1}q_j, q_j)}$
    $x_{j+1} := x_j + \alpha_j p_j$
    $r_{j+1} := r_j - \alpha_j q_j$
    $z_{j+1} := M^{-1}r_{j+1}$
    $\beta_j := \dfrac{(Az_{j+1}, z_{j+1})}{(Az_j, z_j)}$
    $p_{j+1} := z_{j+1} + \beta_j p_j$
end for

□

PROBLEM II In order to save operations, the two matrices $D^{-1}E$ and $D^{-1}E^T$ must be stored when computing $\widehat{A}v$ by Algorithm 9.3. This exercise considers alternatives.

1. Consider the matrix $B \equiv D\widehat{A}D$. Show how to implement an algorithm similar to Algorithm 9.3 for multiplying a vector v by B. The requirement is that only ED^{-1} must be stored.

2. The matrix B in the previous question is not the proper preconditioned version of A by the preconditioning (9.6). CG is used on an equivalent system involving B but a further preconditioning by a diagonal must be applied. Which one? How does the resulting algorithm compare in terms of cost and storage with an algorithm based on 9.3?

3. It was mentioned in Section 9.2.2 that $\widehat{A}$ needed to be further preconditioned by D^{-1}. Consider the split-preconditioning option: CG is to be applied to the preconditioned system associated with
$$C = D^{1/2}\widehat{A}D^{1/2}.$$
Defining $\widehat{E} = D^{-1/2}ED^{-1/2}$, show that
$$C = (I-\widehat{E})^{-1}D_2(I-\widehat{E})^{-T} + (I-\widehat{E})^{-1} + (I-\widehat{E})^{-T},$$
where D_2 is a certain matrix to be determined. Then write an analogue of Algorithm 9.3 using this formulation. How does the operation count compare with that of Algorithm 9.3?

SOLUTION. 1. 首先把你给出的式子按规范写出并说明记号。设
$$A = D_0 - E - E^T, \qquad \widehat{A} = (D-E)^{-1}A(D-E^T)^{-1}.$$
则有
$$\begin{aligned}\widehat{A} &= (D-E)^{-1}\big(D_0 - E - E^T\big)(D-E^T)^{-1}\\ &= (D-E)^{-1}\big[\,D_0 - 2D + (D-E) + (D-E^T)\,\big](D-E^T)^{-1}\\ &\equiv (D-E)^{-1}D_1(D-E^T)^{-1} + (D-E)^{-1} + (D-E^T)^{-1}, \qquad (1)\end{aligned}$$
其中我们定义
$$D_1 \equiv D_0 - 2D.$$
式(1) 正是将 $\widehat{A}$ 展开成三项之和的形式：中间项带有两边的三角因子，而另外两项则是单边的逆因子。

定义
$$B \equiv D\widehat{A}D.$$
把 (1) 左右各乘以 D，得到
$$B = D(D-E)^{-1}D_1(D-E^T)^{-1}D + D(D-E)^{-1}D + D(D-E^T)^{-1}D.$$
令
$$F \equiv ED^{-1}, \qquad I - F = I - ED^{-1}.$$
注意到
$$D(D-E)^{-1} = (I-F)^{-1}, \qquad (D-E^T)^{-1}D = (I-F^T)^{-1},$$

于是

$$B = (I-F)^{-1}D_1(I-F^T)^{-1} + (I-F)^{-1}D + D(I-F^T)^{-1}. \tag{2}$$

由此可设计只需存储 $F = ED^{-1}$ 的乘法算法.

伪码（对任意向量 v 计算 $w = Bv$）:

Algorithm 2 用 $F = ED^{-1}$ 计算 $w = Bv$

1: $\widehat{v} := Dv$
2: $z := (I - F^T)^{-1}\widehat{v}$
3: $w := (I-F)^{-1}\big(\widehat{v} + D_1D^{-1}z\big)$
4: $w := w + z$

以上步骤只用到了矩阵与向量的三种操作：对 $(I-F)$ 或 $(I-F^T)$ 的三角解、对角矩阵乘法（D 与 D_1 作用向量）以及向量加法。因此只需存储 $F = ED^{-1}$ 以及 D 和 D_1 的对角条目.

2. 下证明证明存在可逆矩阵 P 使得

$$D^{-1}B = P^{-1}(M^{-1}A)P,$$

从而说明 $D^{-1}B$ 与 $M^{-1}A$ 相似，具有相同的特征值。

取

$$P = (D-E^T)^{-1}D, \qquad P^{-1} = D^{-1}(D-E^T).$$

由定义可知

$$M^{-1}A = (D-E^T)^{-1}D(D-E)^{-1}A.$$

计算

$$\begin{aligned} P^{-1}(M^{-1}A)P &= \big[D^{-1}(D-E^T)\big]\big[(D-E^T)^{-1}D(D-E)^{-1}A\big]\big[(D-E^T)^{-1}D\big] \\ &= D^{-1}D(D-E)^{-1}A(D-E^T)^{-1}D \\ &= (D-E)^{-1}A(D-E^T)^{-1}D. \end{aligned}$$

上式第二步使用了 $(D-E^T)(D-E^T)^{-1} = I$ 以及 $D^{-1}D = I$。

由 $B = D\,A\hat{}D = D(D-E)^{-1}A(D-E^T)^{-1}D$，可得

$$D^{-1}B = (D-E)^{-1}A(D-E^T)^{-1}D.$$

因此

$$D^{-1}B = P^{-1}(M^{-1}A)P.$$

因此，

$$D^{-1}B = P^{-1}(M^{-1}A)P, \qquad P = (D-E^T)^{-1}D.$$

由于相似矩阵具有相同的谱（特征值集合），因此 $D^{-1}B$ 与 $M^{-1}A$ 在谱性质和收敛行为（例如预条件共轭梯度法的收敛性）上是等价的。

下比较基于 $D^{-1}B$ 的实现与 Algorithm 9.3 的成本与存储：令 n 为未知量维数，记

$$m \equiv \mathrm{nnz}(E)$$

为矩阵 E 中的非零元个数（严格下三角部分）。在稀疏三角解中主要成本与 m 成正比；对角缩放与向量加法的成本为 $O(n)$。用常数 c 表示每个非零在三角求解中造成的平均乘加次数（常数级数）。

下面给出两种算法计算一次矩阵–向量乘（Algorithm 9.3：计算 $\widehat{A}v$，以及基于 $D^{-1}B$ 的实现：计算 $y = D^{-1}Bv$）的典型步骤与成本估计。Algorithm 9.3：计算 $w = \widehat{A}v$：

(a) $\widehat{v} \leftarrow D^{-1}v.$ $\quad O(n)$（对角缩放）

(b) $z \leftarrow (I - D^{-1}E^T)^{-1}\widehat{v}.$ $\quad$ 三角解，约 $c\,m$

(c) $t \leftarrow \widehat{v} + D_1 z.$ $\quad O(n)$（对角乘 + 向量加）

(d) $w \leftarrow (I - D^{-1}E)^{-1}t.$ $\quad$ 三角解，约 $c\,m$

(e) $w \leftarrow w + z.$ $\quad O(n)$

因此主项 FLOP 约为

$$\mathrm{FLOP}_{9.3} \approx 2c\,m + O(n).$$

基于 $D^{-1}B$（存 $F = ED^{-1}$）的实现, 计算 $y = D^{-1}Bv$: 典型实现可为：

(a) 对角缩放 $\widehat{v} := Dv$，成本 $O(n)$；

(b) 上三角系统 $(I - F^T)^{-1}\widehat{v}$，成本 $\approx c\,\mathrm{nnz}(E)$；

(c) 对角操作 $D_1 D^{-1} z$ 与向量加法，成本 $O(n)$；

(d) 下三角系统 $(I - F)^{-1}(\cdot)$，成本 $\approx c\,\mathrm{nnz}(E)$；

(e) 向量加法 $w := w + z$，成本 $O(n)$。

(f) $w := D^{-1}w$，成本 $O(n)$。

因此总 FLOP 约为

$$\mathrm{FLOP}(Bv) \approx 2c\,\mathrm{nnz}(E) + O(n),$$

与 Algorithm 9.3 相同阶。唯一的差别在于前后的对角缩放次序, 可能多或少一次 $O(n)$ 操作，对总体复杂度影响可忽略。

存储比较（稀疏矩阵与对角向量）：两种方法所需的主要存储项为稀疏矩阵数据（value + 索引）与若干长度为 n 的对角向量。若同时显式存储矩阵与其转置（以便高效做三角解与乘法），两种方法的稀疏存储量如下：

- Algorithm 9.3：存储 $D^{-1}E$ 与 $(D^{-1}E)^T$，非零数目约为 $2m$（数值）以及相应的索引；另外存对角向量 D、D_1（各 n 个元素）。
- 基于 $D^{-1}B$：存储 $F = ED^{-1}$ 与 F^T，非零数目同样约为 $2m$；另外存对角向量 D、D_1。

因此以非零元计的稀疏存储量在两种实现间为同阶：

$$\text{稀疏存储} = O(m), \qquad \text{额外对角向量} = O(n).$$

因此，基于 $D^{-1}B$ 的实现与书中 Algorithm 9.3 在主计算量和存储量上是同阶的：两者的主项均为两次稀疏三角解，复杂度约为 $O(m)$（取 $m = \mathrm{nnz}(E)$），并各自需要若干 $O(n)$ 的对角向量存储與操作。两者的差别仅在常数因子（是否多出一次对角缩放、是否显式存转置等），因此在实际工程中可根据存储与实现便利性选择任一方案，而不必担心阶上性能劣化。

3. 已知

$$\widehat{A} = (D-E)^{-1}D_1(D-E^T)^{-1} + (D-E)^{-1} + (D-E^T)^{-1},$$

其中 D_1 是由 A 的对角或题中给出的定义得到的矩阵（例如 $D_1 = D_0 - 2D$）。

$$C \equiv D^{1/2}\widehat{A}D^{1/2}.$$

令

$$\widehat{E} \equiv D^{-1/2}ED^{-1/2}.$$

则有缩放恒等式

$$D - E = D^{1/2}(I - \widehat{E})D^{1/2}, \qquad (D-E)^{-1} = D^{-1/2}(I-\widehat{E})^{-1}D^{-1/2},$$

以及类似地

$$(D-E^T)^{-1} = D^{-1/2}(I-\widehat{E}^T)^{-1}D^{-1/2}.$$

将上述三式代入 $\widehat{A}$ 并左右乘以 $D^{1/2}$，得到

$$\begin{aligned} C &= D^{1/2}\Big[(D-E)^{-1}D_1(D-E^T)^{-1} + (D-E)^{-1} + (D-E^T)^{-1}\Big]D^{1/2} \\ &= \big[D^{1/2}(D-E)^{-1}\big]D_1\big[(D-E^T)^{-1}D^{1/2}\big] + D^{1/2}(D-E)^{-1}D^{1/2} + D^{1/2}(D-E^T)^{-1}D^{1/2} \\ &= (I-\widehat{E})^{-1}\,D^{-1/2}D_1D^{-1/2}\,(I-\widehat{E})^{-T} + (I-\widehat{E})^{-1} + (I-\widehat{E})^{-T}. \end{aligned}$$

于是令

$$D_2 \equiv D^{-1/2}D_1D^{-1/2}$$

可得所需分解

$$C = (I-\widehat{E})^{-1}D_2(I-\widehat{E})^{-T} + (I-\widehat{E})^{-1} + (I-\widehat{E})^{-T}$$

基于上面的分解，对任意向量 v 计算 $y = Cv$ 的步骤与原 Algorithm 9.3 完全类似，只是把 $D^{-1}E$ 换成了 $\widehat{E}$、把 D_1 换成了 D_2，并在必要时做 $D^{\pm 1/2}$ 的缩放。一个直接的伪码如下：

Algorithm 3 用 $\widehat{E} = D^{-1/2}ED^{-1/2}$ 计算 $y = Cv$

1: $z \leftarrow (I - \widehat{E}^T)^{-1}v$
2: $t \leftarrow D_2\, z + v$
3: $u \leftarrow (I-\widehat{E})^{-1}t$
4: $w \leftarrow u + z$

运算量与原 Algorithm 9.3 比较：

- **主要开销**：两者的主要开销均来自对形如 $(I-\cdot)^{-1}$ 的稀疏三角求解（前代/后代）。在 split 形式中需求解的是 $(I-\widehat{E})$ 与 $(I-\widehat{E}^T)$，其稀疏图与原来 $(I-D^{-1}E)$、$(I-D^{-1}E^T)$ 相同，因此每次三角求解的代价仍约为 $O(c\cdot\mathrm{nnz}(E))$，其中 c 为每个非零对应的平均乘加常数。

- **对角乘与缩放**：中间需要进行对角乘 $D_2 z$（$O(n)$），以及可能的 $D^{\pm 1/2}$ 缩放（$O(n)$）；这些为低阶项。
- **总 FLOP**：阶上与 Algorithm 9.3 相同，均为

$$\text{FLOP} \approx 2c\,\text{nnz}(E) + O(n),$$

  即两次稀疏三角解加若干 $O(n)$ 的对角操作与向量加法。
- **预处理一次性成本**：构造 $\widehat{E} = D^{-1/2} E D^{-1/2}$ 与 $D_2 = D^{-1/2} D_1 D^{-1/2}$，成本为一次性 $O(\text{nnz}(E) + n)$。

存储比较

- 需存储稀疏矩阵 $\widehat{E}$，其非零数与 E 相同，故稀疏存储为 $O(\text{nnz}(E))$；还需存对角向量 $D^{1/2}$ 及 D_2（$O(n)$）。
- 与 Algorithm 9.3 存储 $D^{-1}E$ 或 ED^{-1} 相比，存储阶相同；区别仅为是否显式存放 $D^{\pm 1/2}$ 与是否预先计算并存储 $\widehat{E}$，这些差别为常数项。

综上，采用 $C = D^{1/2}\widehat{A}D^{1/2}$ 并将 E 缩放为 $\widehat{E}$ 后，得到的矩阵-向量乘法形式与 Algorithm9.3 在结构上完全相似：主要仍为两次稀疏三角求解加若干对角乘与向量相加。故在渐近运算量与存储量上，与 Algorithm 9.3 同阶，差别仅为常数因子与一次性预处理（计算并存储 $D^{\pm 1/2}$、$\widehat{E}$、D_2 等）。

□

PROBLEM III Let $M = LU$ be a preconditioner for a matrix A. Show that the left, right, and split preconditioned matrices all have the same eigenvalues. Does this mean that the corresponding preconditioned iterations will converge in

1. exactly the same number of steps?
2. roughly the same number of steps for any matrix?
3. roughly the same number of steps, except for ill-conditioned matrices?

SOLUTION. 先证明特征值相同：令三种预处理后的矩阵为

左预处理：$M^{-1}A$，右预处理：AM^{-1}，分裂（split）预处理：$L^{-1}AU^{-1}$.

注意到 $M = LU$ 可逆，且

$$M^{-1} = U^{-1}L^{-1}.$$

首先证明 $M^{-1}A$ 与 AM^{-1} 相似：

$$AM^{-1} = A(U^{-1}L^{-1}) = (LU)(U^{-1}L^{-1}A) = L\big(UU^{-1}L^{-1}A\big) = L\big(M^{-1}A\big)L^{-1}.$$

因此

$$AM^{-1} = L\,(M^{-1}A)\,L^{-1},$$

即 AM^{-1} 与 $M^{-1}A$ 相似，从而它们具有相同的特征值。

再证明分裂预处理与 $M^{-1}A$ 相似。由 $M^{-1} = U^{-1}L^{-1}$ 得

$$M^{-1}A = U^{-1}L^{-1}A.$$

两边左乘 U 并右乘 U^{-1}：

$$U(M^{-1}A)U^{-1} = U(U^{-1}L^{-1}A)U^{-1} = L^{-1}AU^{-1},$$

即

$$L^{-1}AU^{-1} = U\left(M^{-1}A\right)U^{-1}.$$

因此 $L^{-1}AU^{-1}$ 与 $M^{-1}A$ 相似，故它也具有与前两者相同的特征值。

综上，三种预处理矩阵互为相似变换，因此具有完全相同的特征值集合（代数重数也相同）。

1. 不成立。反例：取

$$A = \begin{pmatrix} 1 & 100 \\ 0 & 1 \end{pmatrix}, \quad M = I.$$

左预处理 $M^{-1}A = A$ 与右预处理 $AM^{-1} = A$ 的特征值都是 $1, 1$。但 A 是非正规矩阵，GMRES 迭代步数依赖非对角元素大小。左/右预处理对应的残差方向不同，因此迭代步数可以完全不同。$\Rightarrow$ 特征值相同不保证步数完全相同。

2. 不成立。反例：对于任意非正规或病态矩阵 A，即使左/右/分裂预处理矩阵特征值相同，迭代收敛速度依然可能差别很大。例如上例矩阵 $A = \begin{pmatrix} 1 & 100 \\ 0 & 1 \end{pmatrix}$ 已经说明，对非正规矩阵，步数不一定大致相同。

3. 成立（在 SPD 或接近正规矩阵的情况下）。假设 A 为对称正定（SPD）矩阵，M 也是 SPD。左预处理 $M^{-1}A$、右预处理 AM^{-1}、分裂预处理 $L^{-1}AU^{-1}$ 都是对称正定或相似于 SPD 矩阵。对 SPD 矩阵的 CG 迭代收敛步数仅依赖条件数 κ：

$$|0x_k - x^*|0_A \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k |0x_0 - x^*|0_A$$

由于三种预处理矩阵谱相同，条件数相同，迭代步数也大致相同。

□