第五讲　对称特征值问题

对称特征值问题的常用算法有 (直接法)

- Jacobi **迭代**: 最古老, 收敛速度较慢, 但精度较高, 适合并行计算。
- Rayleigh **商迭代**: 一般具有三次收敛性, 但需要解方程组。
- **对称 QR 迭代**: 对于对称三对角矩阵, **若只计算特征值, 则速度最快** (运算量为 $O(n^2)$), 如果还需要计算特征向量, 则运算量约为 $6n^3$。
- **分而治之法** (Divide-and-Conquer) : 同时计算特征值和特征向量的一种快速算法. 基本思想是将大矩阵分解成小矩阵, 然后利用递归思想求特征值. 在最坏的情形下, 运算量为 $O(n^3)$. 在实际应用中, 平均为 $O(n^{2.3})$. 如果使用快速多极子算法 (FMM) 后, 理论上的运算量可降低到 $O(n\log^p n)$, 其中 p 是一个较小的整数, 这使得分而治之算法成为目前**计算对称三对角矩阵的特征值和特征向量的首选方法**。

- **对分法和反迭代**: 对分法主要用于求解对称三对角矩阵在某个区间中的特征值, 运算量约为 $O(kn)$, 其中 k 为所需计算的特征值的个数. 反迭代用于计算特征向量, 在最佳情况下, 即特征值 "适当分离" 时, 运算量约为 $O(kn)$, 但在最差情况下, 即特征值成串地紧靠在一起时, 运算量约为 $O(k^2n)$, 而且不能保证特征向量的精度 (虽然实际上它几乎是精确的)。

✍ 除了 Jacobi 迭代和 Rayleigh 商迭代外, 其余算法都需要先将对称矩阵三对角化, 这个过程大约需花费 $\frac{4}{3}n^3$ 的工作量, 如果需要计算特征向量的话, 则运算量约为 $\frac{8}{3}n^3$.

1 Jacobi 迭代

基本思想

通过一系列的 Jacobi 旋转 将 A 正交相似于一个对角矩阵，即:

$$A^{(0)} = A, \quad A^{(k+1)} = J_k^{\mathsf{T}} A^{(k)} J_k, \quad k = 0, 1, \ldots,$$

且 $A^{(k)}$ 收敛到一个对角矩阵, 其中 J_k 为 Jacobi 旋转, 即 Givens 变换:

$$J_k = G(i_k, j_k, \theta_k) = \begin{array}{c} \\ \\ \\ i_k \\ \\ j_k \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{cccc} i_k & & j_k & \end{array} \\ \left[\begin{array}{c|ccc|c} I & & & & \\ \hline & \cos\theta_k & \cdots & -\sin\theta_k & \\ & \vdots & \ddots & \vdots & \\ & \sin\theta_k & \cdots & \cos\theta_k & \\ \hline & & & & I \end{array}\right] \end{array}$$

引理 设 $A \in \mathbb{R}^{2\times 2}$ 是对称矩阵, 则存在 Givens 变换 $G \in \mathbb{R}^{2\times 2}$ 使得 $G^{\mathsf{T}}AG$ 为对角阵. (板书)

为了使得 $A^{(k)}$ 收敛到一个对角矩阵, 其非对角元素必须趋向于 0.

记 $\mathrm{off}(A)$ 为所有非对角元素的平方和, 即

$$\mathrm{off}(A) = \sum_{i \neq j} a_{ij}^2 = \|A\|_F^2 - \sum_{i=1}^{n} a_{ii}^2 ,$$

我们的目标就是使得 $\mathrm{off}(A)$ 尽快趋向于 0.

引理 设 $A = [a_{ij}]_{n\times n} \in \mathbb{R}^{n\times n}$ 是对称矩阵, $\hat{A} = [\hat{a}_{ij}]_{n\times n} = J^{\mathsf{T}} A J$, $J = G(i, j, \theta)$, 其中 θ 的选取使得 $\hat{a}_{ij} = \hat{a}_{ji} = 0$, 则

$$\mathrm{off}(\hat{A}) = \mathrm{off}(A) - 2a_{ij}^2$$

(板书)

算法 1.1 Jacobi 迭代算法

Given a symmetric matrix $A \in \mathbb{R}^{n\times n}$
if eigenvectors are desired **then**
    set $J = I$ and $shift = 1$
end if
while not converge **do**
    choose an index pair (i, j) such that $a_{ij} \neq 0$
    $\tau = (a_{ii} - a_{jj})/(2a_{ij})$
    $t = \text{sign}(\tau)/(|\tau| + \sqrt{1+\tau^2})$ % 计算 $\tan\theta$
    $c = 1/\sqrt{1+t^2}$, $s = c \cdot t$ % 计算 Givens 变换
    $A = G(i, j, \theta)^{\intercal} A G(i, j, \theta)$ % 实际计算时不需要做矩阵乘积
    if $shift = 1$ **then**
        $J = J \cdot G(i, j, \theta)$
    end if
end while

a_{ij} 的选取问题

直观选取方法: 所有非对角元素中绝对值最大的一个 $\Longrightarrow$ 经典 Jacobi 算法

算法 1.2 经典 Jacobi 迭代算法

1: Given a symmetric matrix $A \in \mathbb{R}^{n\times n}$
2: **while** off$(A)>$ *tol* **do**
3: choose (i,j) such that $|a_{ij}| = \max_{k\neq l} |a_{kl}|$
4: $\tau = (a_{ii} - a_{jj})/(2a_{ij})$
5: $t = \text{sign}(\tau)/(|\tau| + \sqrt{1+\tau^2})$
6: $c = 1/\sqrt{1+t^2}, \quad s = c\cdot t$
7: $A = G(i,j,\theta)^{\intercal} A G(i,j,\theta)$
8: $J = J \cdot G(i,j,\theta)$ % if eigenvectors are desired
9: **end while**

可以证明, 经典 Jacobi 算法至少是线性收敛的.

定理 对于经典 Jacobi 算法 1.2, 有

$$\operatorname{off}(A^{(k+1)}) \leq \left(1 - \frac{1}{N}\right) \operatorname{off}(A^{(k)}), \quad N = \frac{n(n-1)}{2}.$$

故 k 步迭代后, 有

$$\operatorname{off}(A^{(k)}) \leq \left(1 - \frac{1}{N}\right)^k \operatorname{off}(A^{(0)}) = \left(1 - \frac{1}{N}\right)^k \operatorname{off}(A).$$

事实上, 经典 Jacobi 算法最终是二次局部收敛的.

定理 经典 Jacobi 算法 1.2 是 N 步局部二次收敛的, 即对足够大的 k, 有

$$\operatorname{off}\left(A^{(k+N)}\right) = O\left(\operatorname{off}^2\left(A^{(k)}\right)\right).$$

✍ 经典 Jacobi 算法的每一步都要寻找绝对值最大的非对角元, 比较费时.

改进: 逐行扫描, 这就是 **循环 Jacobi 迭代方法**

算法 1.3 循环 Jacobi 迭代算法 (逐行扫描)

Given a symmetric matrix $A \in \mathbb{R}^{n\times n}$
while off$(A)>$ *tol* **do**
  for $i = 1$ to $n-1$ **do**
    for $j = i+1$ to n **do**
      if $a_{ij} \neq 0$ **then**
        $\tau = (a_{ii} - a_{jj})/(2a_{ij})$
        $t = \text{sign}(\tau)/(|\tau| + \sqrt{1+\tau^2}), \quad c = 1/\sqrt{1+t^2}, \quad s = c \cdot t$
        $A = G(i,j,\theta)^{\mathsf{T}} A G(i,j,\theta)$
        $J = J \cdot G(i,j,\theta)$ % if eigenvectors are desired
      end if
    end for
  end for
end while

2 Rayleigh 商迭代

在反迭代方法中, 以 Rayleigh 商作为位移.

关于 Rayleigh 商迭代的收敛性, 我们有下面的结论.

定理 设 $A \in \mathbb{R}^{n\times n}$ 对称, 且特征值都是单重的. 则当误差足够小时, Rayleigh 商迭代中每步迭代所得的正确数字的位数增至三倍, 即 Rayleigh 商迭代是局部三次收敛的.

关于 RQI 算法的全局收敛性, 可参见文献 [Parlett '98].

3 对称 QR 迭代

将带位移的隐式 QR 方法运用到对称矩阵, 就得到对称 QR 迭代方法.

基本步骤

1. 对称三对角化: 利用 Householder 变换, 将 A 化为对称三对角矩阵, 即计算正交矩阵 Q 使得 $T = QAQ^{\mathsf{T}}$ 为对称三对角矩阵;
2. 使用带 (单) 位移的隐式 QR 迭代算法计算 T 的特征值与特征值向量;
3. 计算 A 的特征向量.

对称三对角化

任何一个对称矩阵 $A \in \mathbb{R}^{n\times n}$ 都可以通过正交变换转化成一个对称三对角矩阵 T. 这个过程可以通过 Householder 变换来实现, 也可以通过 Givens 变换来实现.

对称 QR 迭代算法的运算量

- 三对角化 $4n^3/3 + O(n^2)$, 若需计算特征向量, 则为 $8n^3/3 + O(n^2)$;
- 对 T 做带位移的隐式 QR 迭代, 每次迭代的运算量为 $6n$;
- 计算特征值, 假定每个平均迭代 2 步, 则总运算量为 $12n^2$;
- 若要计算 T 的所有特征值和特征向量, 则运算量为 $6n^3 + O(n^2)$;
- 若只要计算 A 的所有特征值, 运算量为 $4n^3/3 + O(n^2)$;
- 若计算 A 的所有特征值和特征向量, 则运算量为 $26n^3/3 + O(n^2)$;

位移的选取 — Wilkinson 位移

位移的好坏直接影响到算法的收敛速度. 我们可以通过下面的方式来选取位移. 设

$$A^{(k)} = \begin{bmatrix} a_1^{(k)} & b_1^{(k)} & & \\ b_1^{(k)} & \ddots & \ddots & \\ & \ddots & \ddots & b_{n-1}^{(k)} \\ & & b_{n-1}^{(k)} & a_n^{(k)} \end{bmatrix},$$

一种简单的位移选取策略就是令 $\sigma_k = a_n^{(k)}$. 事实上, $a_n^{(k)}$ 就是收敛到特征向量的迭代向量的 Rayleigh 商. 这种位移选取方法几乎对所有的矩阵都有三次渐进收敛速度, 但也存在不收敛的例子, 故我们需要对其做改进.

Wilkinson 位移

取 $\begin{bmatrix} a_{n-1}^{(k)} & b_{n-1}^{(k)} \\ b_{n-1}^{(k)} & a_n^{(k)} \end{bmatrix}$ 的最接近 $a_n^{(k)}$ 的特征值作为位移.

通过计算可得 Wilkinson 位移为

$$\sigma = a_n^{(k)} + \delta - \operatorname{sign}(\delta)\sqrt{\delta^2 + \left(b_{n-1}^{(k)}\right)^2}, \quad \text{其中} \quad \delta = \frac{1}{2}(a_{n-1}^{(k)} - a_n^{(k)}).$$

出于稳定性方面的考虑, 我们通常用下面的计算公式

$$\sigma = a_n^{(k)} - \frac{\left(b_{n-1}^{(k)}\right)^2}{\delta + \operatorname{sign}(\delta)\sqrt{\delta^2 + \left(b_{n-1}^{(k)}\right)^2}}$$

定理 采用 Wilkinson 位移的 QR 迭代是整体收敛的, 且至少是线性收敛. 事实上, 几乎对所有的矩阵都是渐进三次收敛的.

例 带 Wilkinson 位移的隐式 QR 迭代算法收敛性演示.

Matlab 代码: `Eig_TriQR.m`