

第五讲 对称特征值问题

- 1 Jacobi 迭代
- 2 Rayleigh 商迭代
- 3 对称 QR 迭代
- 4 分而治之法
- 5 对分法和反迭代
- 6 奇异值分解
- 7 扰动分析
- 8 应用举例






关于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**: 对称矩阵的 QR 方法. 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:

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

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

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



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 $G^T A G$ 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 $\text{off}(A)$ 为所有非对角元素的平方和, 即

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

我们的目标就是使得 $\text{off}(A)$ 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, $\hat{A} = [\hat{a}_{ij}]_{n \times n} = J^T A J$, $J = G(i, j, \theta)$, 其中 θ 的选取使得 $\hat{a}_{ij} = \hat{a}_{ji} = 0$, 则

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

(板书)

算法 1.1 Jacobi 迭代算法

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



a_{ij} 的选取问题

一种直观的选取方法就是使得 a_{ij} 为所有非对角元素中绝对值最大的一个, 这就是经典 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: if eigenvectors are desired then
3:     set  $J = I$  and  $shift = 1$ 
4: end if
5: while  $\text{off}(A) > tol$  do
6:     choose  $(i, j)$  such that  $|a_{ij}| = \max_{k \neq l} |a_{kl}|$ 
7:      $\tau = (a_{ii} - a_{jj}) / (2a_{ij})$ 
8:      $t = \text{sign}(\tau) / (|\tau| + \sqrt{1 + \tau^2})$ 
9:      $c = 1 / \sqrt{1 + t^2}$ ,  $s = c \cdot t$ 
10:     $A = G(i, j, \theta)^T A G(i, j, \theta)$ 
11:    if  $shift = 1$  then
12:         $J = J \cdot G(i, j, \theta)$ 
13:    end if
14: 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

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

故 k 步迭代后, 有

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

(证明见讲义, 留作自习)



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

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



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)

- 1: Given a symmetric matrix $A \in \mathbb{R}^{n \times n}$
- 2: **if** eigenvectors are desired **then**
- 3: set $J = I$ and $shift = 1$
- 4: **end if**
- 5: **while** $\text{off}(A) > tol$ **do**
- 6: **for** $i = 1$ to $n - 1$ **do**
- 7: **for** $j = i + 1$ to n **do**
- 8: **if** $a_{ij} \neq 0$ **then**
- 9: $\tau = (a_{ii} - a_{jj}) / (2a_{ij})$



```
10:       $t = \text{sign}(\tau) / (|\tau| + \sqrt{1 + \tau^2})$ 
11:       $c = 1 / \sqrt{1 + t^2}$ 
12:       $s = c \cdot t$ 
13:       $A = G(i, j, \theta)^\top A G(i, j, \theta)$ 
14:      if  $shift = 1$  then
15:           $J = J \cdot G(i, j, \theta)$ 
16:      end if
17:  end if
18: end for
19: end for
20: end while
```

循环 Jacobi 也具有局部二次收敛性.



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. (证明见讲义, 留作自习)

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 $T = QAQ^T$ 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

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

出于稳定性方面的考虑, 我们通常用下面的计算公式

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



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](#)



4 分而治之法

分而治之法由 Cuppen 于 1981 年首次提出, 但直到 1995 年才出现稳定的实现方式, 是目前计算 **所有特征值和特征向量** 的最快算法.

考虑不可约对称三对角矩阵

$$T = \left[\begin{array}{ccc|ccc} a_1 & b_1 & & & & \\ b_1 & \ddots & \ddots & & & \\ & \ddots & a_{m-1} & b_{m-1} & & \\ & & b_{m-1} & a_m & & \\ \hline & & & b_m & a_{m+1} & b_{m+1} \\ & & & & b_{m+1} & \ddots & \ddots \\ & & & & & \ddots & \ddots & b_{n-1} \\ & & & & & & b_{n-1} & a_n \end{array} \right]$$



$$= \left[\begin{array}{ccc|ccc} a_1 & b_1 & & & & \\ b_1 & \ddots & \ddots & & & \\ & \ddots & a_{m-1} & b_{m-1} & & \\ & & b_{m-1} & a_m - b_m & & \\ \hline & & & a_{m+1} - b_m & b_{m+1} & \\ & & b_{m+1} & \ddots & \ddots & \\ & & & \ddots & \ddots & b_{n-1} \\ & & & & b_{n-1} & a_n \end{array} \right] + \left[\begin{array}{c|c} b_m & b_m \\ \hline b_m & b_m \end{array} \right]$$

$$= \left[\begin{array}{c|c} T_1 & 0 \\ \hline 0 & T_2 \end{array} \right] + b_m v v^\top,$$

其中 $v = [0, \dots, 0, 1, 1, 0, \dots, 0]^\top$.



假定 T_1 和 T_2 的特征值分解已经计算出来

即 $T_1 = Q_1 \Lambda_1 Q_1^T$, $T_2 = Q_2 \Lambda_2 Q_2^T$, 下面考虑 T 的特征值分解.

$$\begin{aligned} T &= \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} + b_m v v^T = \begin{bmatrix} Q_1 \Lambda_1 Q_1^T & 0 \\ 0 & Q_2 \Lambda_2 Q_2^T \end{bmatrix} + b_m v v^T \\ &= \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} \left(\begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix} + b_m u u^T \right) \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}^T, \end{aligned}$$

其中

$$u = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}^T, \quad v = \begin{bmatrix} Q_1^T \text{ 的最后一列} \\ Q_2^T \text{ 的第一列} \end{bmatrix}.$$

令 $\alpha = b_m$, $D = \text{diag}(\Lambda_1, \Lambda_2) = \text{diag}(d_1, d_2, \dots, d_n)$, 并假定 $d_1 \geq d_2 \geq \dots \geq d_n$. 则 T 的特征值与 $D + \alpha u u^T$ 的特征值相同.



考虑 $D + \alpha uu^\top$ 的特征值

设 λ 是 $D + \alpha uu^\top$ 的一个特征值, 若 $D - \lambda I$ 非奇异, 则

$$\det(D + \alpha uu^\top - \lambda I) = \det(D - \lambda I) \cdot \det(I + \alpha(D - \lambda I)^{-1}uu^\top).$$

故 $\det(I + \alpha(D - \lambda I)^{-1}uu^\top) = 0$.

引理 设 $x, y \in \mathbb{R}^n$, 则 $\det(I + xy^\top) = 1 + y^\top x$.

于是

$$\det(I + \alpha(D - \lambda I)^{-1}uu^\top) = 1 + \alpha u^\top (D - \lambda I)^{-1}u = 1 + \alpha \sum_{i=1}^n \frac{u_i^2}{d_i - \lambda} \triangleq f(\lambda)$$

故求 A 的特征值等价于求**特征方程** $f(\lambda) = 0$ 的根.

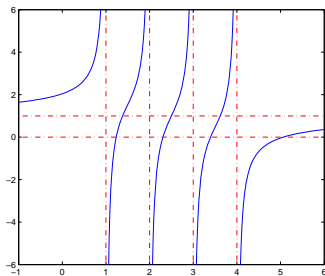


由于

$$f'(\lambda) = \alpha \sum_{i=1}^n \frac{u_i^2}{(d_i - \lambda)^2},$$

当所有的 d_i 都互不相同, 且所有的 u_i 都不为零时, $f(\lambda)$ 在 $\lambda \neq d_i$ 处都是严格单调的.

所以 $f(\lambda)$ 在每个区间 (d_{i+1}, d_i) 内都有一个根, 共 $n - 1$ 个, 另一个根在 (d_1, ∞) (若 $\alpha > 0$) 或 $(-\infty, d_n)$ (若 $\alpha < 0$) 中.



$$(\alpha = 0.5, d_i = 4, 3, 2, 1, u_i = 1)$$



由于 $f(\lambda)$ 在每个区间 (d_{i+1}, d_i) 内光滑且严格单调递增 ($\alpha > 0$) 或递减 ($\alpha < 0$), 所以在实际计算中, 可以使用对分法, 牛顿法及其变形, 或有理逼近等算法来求解. 通常都能很快收敛, 一般只需迭代几步即可.


因此, 计算一个特征值的运算量约为 $O(n)$, 计算 $D + \alpha uu^T$ 的所有特征值的运算量约为 $O(n^2)$.

当所有特征值计算出来后, 我们用下面的引理来计算特征向量.

引理 设 $D \in \mathbb{R}^{n \times n}$ 为对角矩阵, $u \in \mathbb{R}^n$, $\alpha \in \mathbb{R}$, 若 λ 是 $D + \alpha uu^T$ 的特征值, 且 $\lambda \neq d_i, i = 1, 2, \dots, n$, 则 $(D - \lambda I)^{-1}u$ 是其对应的特征向量. (板书)

算法 4.1 计算对称三对角矩阵的特征值和特征向量的分而治之法

```
1: function [Q, Λ] = dc_eig(T)           % T = QΛQ⊤
2: if T is of 1 × 1 then
3:     Q = 1, Λ = T, return
4: end if
5: form  $T = \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} + b_m v v^⊤$ 
6: [Q1, Λ1] = dc_eig(T1), [Q2, Λ2] = dc_eig(T2)
7: form D + αuu⊤ from Λ1, Λ2, Q1, Q2
8: compute the eigenvalues Λ and eigenvectors  $\hat{Q}$  of D + αuu⊤
9: compute the eigenvectors of T with  $Q = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} \cdot \hat{Q}$ 
10: end
```

 在分而治之法中, 计算特征值和计算特征向量是同时进行的.



分而治之法的实施

下面我们详细讨论分而治之算法的几个细节问题:

- (1) 如何减小运算量;
- (2) 如何求解特征方程 $f(\lambda) = 0$;
- (3) 如何稳定地计算特征向量.



(1) 如何减小运算量 — 收缩技巧 (deflation)

分而治之算法的计算复杂性分析如下: 用 $t(n)$ 表示对 n 阶矩阵调用函数 `dc_eig` 的运算量, 则

$$\begin{aligned} t(n) = & 2t(n/2) \quad \text{递归调用 dc_eig 两次} \\ & + O(n^2) \quad \text{计算 } D + \alpha uu^T \text{ 的特征值和特征向量} \\ & + c \cdot n^3 \quad \text{计算 } Q. \end{aligned}$$

如果计算 Q 时使用的是稠密矩阵乘法, 则 $c = 2$; 若不计 $O(n^2)$ 项, 则由递归公式 $t(n) = 2t(n/2) + c \cdot n^3$ 可得 $t(n) \approx c \cdot 4n^3/3$.

事实上, 由于收缩 (deflation) 现象的存在, 常数 c 通常比 1 小得多.



在前面的算法描述过程中, 我们假定 d_i 互不相等且 u_i 不能为零.

事实上, 若 $d_i = d_{i+1}$ 或 $u_i = 0$, 则 d_i 即为 $D + \alpha uu^T$ 的特征值, 这种现象我们称为**收缩 (deflation)**.

在实际计算时, 当 $d_i - d_{i+1}$ 或 $|u_i|$ 小于一个给定的阈值时, 我们就近似认为 d_i 为 $D + \alpha uu^T$ 的特征值, 即出现收缩现象.

在实际计算中, 收缩现象会经常发生, 而且非常频繁, 所以我们可以而且应该利用这种优点加快分而治之算法的速度.

由于主要的计算量集中在计算 Q , 即算法最后一步的矩阵乘积. 如果 $u_i = 0$, 则 d_i 为特征值, 其对应的特征向量为 e_i , 即 \hat{Q} 的第 i 列为 e_i , 故计算 Q 的第 i 列时不需要做任何的计算.

当 $d_i = d_{i+1}$ 时, 也存在一个类似的简化.



(2) 特征方程求解

通常我们可以使用牛顿法来计算特征方程 $f(\lambda) = 0$ 的解.

当 $d_i \neq d_{i+1}$ 且 $u_i \neq 0$ 时, 用牛顿法计算 $f(\lambda)$ 在 (d_{i+1}, d_i) 中的零点 λ_i .

如果 $|u_i|$ 小于给定的阈值时, 我们可直接将 d_i 作为特征值 λ_i 的一个近似.

但当 u_i 很小 (却大于给定的阈值) 时, 此时 $f(\lambda)$ 在区间 $[d_{i+1}, d_i]$ 中的大部分处的斜率几乎为 0 (见下图). 这时, 如果任取 $[d_{i+1}, d_i]$ 中的一个点作为迭代初始点, 经过一次牛顿迭代后, 迭代解可能会跑到区间 $[d_{i+1}, d_i]$ 的外面, 造成不收敛.

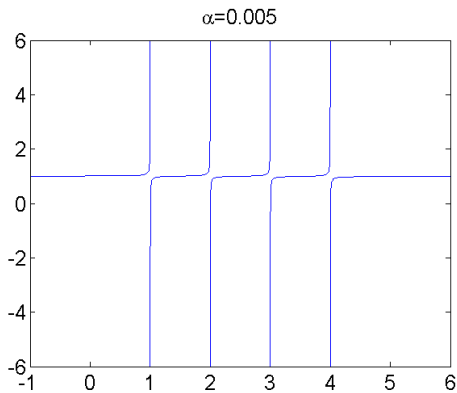


图 4.1 $f(\lambda) = 1 + 0.005 \left(\frac{1}{4-\lambda} + \frac{1}{3-\lambda} + \frac{1}{2-\lambda} + \frac{1}{1-\lambda} \right)$ 的图像



这时需要采用修正的牛顿法. 假设我们已经计算出 λ_i 的一个近似 $\tilde{\lambda}$, 下面我们需要从 $\tilde{\lambda}$ 出发, 利用牛顿迭代计算下一个近似, 直至收敛. 我们知道牛顿法的基本原理是使用 $f(\lambda)$ 在点 $\tilde{\lambda}$ 的切线来近似 $f(\lambda)$, 并将切线的零点作为下一个近似, 即用直线来近似曲线 $f(\lambda)$.

当 u_i 很小时, 这种近似方法会出现问题, 此时不能使用直线来近似 $f(\lambda)$. 这时, 我们可以寻找其它简单函数 $h(\lambda)$ 来近似 $f(\lambda)$, 然后用 $h(\lambda)$ 的零点作为 $f(\lambda)$ 零点的近似, 并不断迭代下去, 直至收敛.

当然, $h(\lambda)$ 需要满足一定的要求:

- (1) 必须容易构造;
- (2) 其零点容易计算;
- (3) 尽可能与 $f(\lambda)$ 相近.



下面给出构造 $h(\lambda)$ 的一种方法.

因为 d_i 和 d_{i+1} 是 $f(\lambda)$ 的奇点, 所以我们令

$$h(\lambda) = \frac{c_1}{d_i - \lambda} + \frac{c_2}{d_{i+1} - \lambda} + c_3,$$

其中 c_1, c_2, c_3 为参数. 显然, $h(\lambda)$ 的零点很容易计算 (与 Newton 法相差无几). 在选取这些参数时, 要使得 $h(\lambda)$ 在 $\tilde{\lambda}$ 附近尽可能地接近 $f(\lambda)$. 记

$$\begin{aligned} f(\lambda) &= 1 + \alpha \sum_{k=1}^n \frac{u_k^2}{d_k - \lambda} = 1 + \alpha \left(\sum_{k=1}^i \frac{u_k^2}{d_k - \lambda} + \sum_{k=i+1}^n \frac{u_k^2}{d_k - \lambda} \right) \\ &\triangleq 1 + \alpha (\Psi_1(\lambda) + \Psi_2(\lambda)). \end{aligned}$$



当 $\lambda \in (d_{i+1}, d_i)$ 时, $\Psi_1(\lambda)$ 为正项的和, $\Psi_2(\lambda)$ 为负项的和, 因此它们都可以较精确地计算. 但如果把它们加在一起时可能会引起对消, 从而失去相对精度. 因此我们也将 $h(\lambda)$ 写成

$$h(\lambda) = 1 + \alpha(h_1(\lambda) + h_2(\lambda)),$$

其中

$$h_1(\lambda) = \frac{c_1}{d_i - \lambda} + \hat{c}_1, \quad h_2(\lambda) = \frac{c_2}{d_{i+1} - \lambda} + \hat{c}_2$$

满足

$$\begin{aligned} h_1(\tilde{\lambda}) &= \Psi_1(\tilde{\lambda}), & h'_1(\tilde{\lambda}) &= \Psi'_1(\tilde{\lambda}), \\ h_2(\tilde{\lambda}) &= \Psi_2(\tilde{\lambda}), & h'_2(\tilde{\lambda}) &= \Psi'_2(\tilde{\lambda}). \end{aligned}$$



即 $h_1(\lambda)$ 和 $h_2(\lambda)$ 分别在点 $\tilde{\lambda}$ 与 $\Psi_1(\lambda)$ 和 $\Psi_2(\lambda)$ 相切. 这在数值插值中是常见的条件. 容易计算可得

$$\begin{cases} c_1 = \Psi'_1(\tilde{\lambda})(d_i - \tilde{\lambda})^2, & \hat{c}_1 = \Psi_1(\tilde{\lambda}) - \Psi'_1(\tilde{\lambda})(d_i - \tilde{\lambda}), \\ c_2 = \Psi'_2(\tilde{\lambda})(d_{i+1} - \tilde{\lambda})^2, & \hat{c}_2 = \Psi_2(\tilde{\lambda}) - \Psi'_2(\tilde{\lambda})(d_{i+1} - \tilde{\lambda}). \end{cases} \quad (5.3)$$

所以, 最后取

$$h(\lambda) = 1 + \alpha(\hat{c}_1 + \hat{c}_2) + \alpha \left(\frac{c_1}{d_i - \lambda} + \frac{c_2}{d_{i+1} - \lambda} \right). \quad (5.4)$$

这就是迭代函数.



算法 4.2 修正的 Newton 算法

- 1: set $k = 0$
 - 2: choose an initial guess $\lambda_0 \in [d_{i+1}, d_i]$
 - 3: **while** not convergence **do**
 - 4: let $\tilde{\lambda} = \lambda_k$ and compute $c_1, c_2, \hat{c}_1, \hat{c}_2$ from (5.3)
 - 5: set $k = k + 1$
 - 6: compute the solution λ_k of $h(\lambda)$ defined by (5.4)
 - 7: **end while**
-



(3) 计算特征向量的稳定算法

设 λ_i 是 $D + \alpha uu^T$ 的特征值, 则根据引理 4.2, 可利用公式 $(D - \lambda_i I)^{-1}u$ 来计算其对应的特征向量. 但遗憾的是, 当相邻的两个特征值非常接近时, 这个公式可能不稳定. 即当 λ_i 与 λ_{i+1} 非常接近时, 它们都靠近 d_{i+1} (这里假定 $\lambda_i \in (d_{i+1}, d_i)$), 在计算 $d_{i+1} - \lambda_i$ 和 $d_{i+1} - \lambda_{i+1}$ 时会存在对消, 这就可能损失有效数字, 产生较大的相对误差, 从而导致 $(D - \lambda_i I)^{-1}u$ 与 $(D - \lambda_{i+1} I)^{-1}u$ 的计算是不准确的, 正交性也会失去. 下面的定理可以解决这个问题.




定理 (Löwner) 设对角阵 $D = \text{diag}(d_1, d_2, \dots, d_n)$ 满足 $d_1 > d_2 > \dots > d_n$. 若矩阵 $\hat{D} = D + \hat{u}\hat{u}^\top$ 的特征值 $\lambda_1, \lambda_2, \dots, \lambda_n$ 满足交错性质

$$\lambda_1 > d_1 > \lambda_2 > d_2 > \dots > \lambda_n > d_n, \quad (5.5)$$

则向量 \hat{u} 的分量满足

$$|\hat{u}_i| = \left(\frac{\prod_{k=1}^n (\lambda_k - d_i)}{\prod_{k=1, k \neq i}^n (d_k - d_i)} \right)^{1/2}. \quad (5.6)$$

(证明见讲义, 留作自习)

 因此, 我们可以采用公式 (5.6) 来计算特征向量. 这样就尽可能地避免了出现分母很小的情形.



5 对分法和反迭代

对分法的基本思想是利用惯性定理来计算所需的部分特征值.

定义 设 A 为对称矩阵, 则其**惯性**定义为

$$\text{Inertia}(A) = (\nu, \zeta, \pi)$$

其中 ν, ζ, π 分别表示 A 的负特征值, 零特征值和正特征值的个数.

定理 (Sylvester 惯性定理) 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, $X \in \mathbb{R}^{n \times n}$ 非奇异, 则 $X^T A X$ 与 A 有相同的惯性.



利用 LU 分解可得 $A - zI = LDL^T$, 其中 L 为非奇异下三角矩阵, D 为对角阵, 则

$$\text{Inertia}(A - zI) = \text{Inertia}(D).$$

由于 D 是对角矩阵, 所以 $\text{Inertia}(D)$ 很容易计算.

设 $\alpha \in \mathbb{R}$, 记 $\text{Negcount}(A, \alpha)$ 为小于 α 的 A 的特征值的个数, 即

$$\text{Negcount}(A, \alpha) = \#(\lambda(A) < \alpha).$$

设 $\alpha_1 < \alpha_2$, 则 A 在区间 $[\alpha_1, \alpha_2)$ 中的特征值个数为

$$\text{Negcount}(A, \alpha_2) - \text{Negcount}(A, \alpha_1).$$

如果 $\alpha_2 - \alpha_1 < \text{tol}$ (其中 $\text{tol} \ll 1$ 为事先给定的阈值), 且 A 在 $[\alpha_1, \alpha_2)$ 中有特征值, 则我们可将 $[\alpha_1, \alpha_2)$ 中的任意一个值作为 A 在该区间中的特征值的近似.



由此我们可以给出下面的对分法.

算法 5.1 计算 A 在 $[a, b)$ 中的所有特征值

- 1: Let tol be a given threshold
- 2: compute $n_a = \text{Negcount}(A, a)$
- 3: compute $n_b = \text{Negcount}(A, b)$
- 4: **if** $n_a = n_b$ **then**
- 5: **return** % 此时 $[a, b)$ 中没有 A 的特征值
- 6: **end if**
- 7: put (a, n_a, b, n_b) onto $worklist$
- 8: % $worklist$ 中的元素是“四元素对”, 即由四个数组成的数对
- 9: **while** $worklist$ not empty **do**
- 10: remove $(low, n_{low}, up, n_{up})$ from the $worklist$
- 11: % $(low, n_{low}, up, n_{up})$ 是 $worklist$ 中的任意一个元素
- 12: **if** $(up - low) < tol$ **then**



```
13:      print "There are  $n_{up} - n_{low}$  eigenvalues in [low, up)"
14:  else
15:      compute  $mid = (low + up)/2$ 
16:      compute  $n_{mid} = \text{Negcount}(A, mid)$ 
17:      if ( $n_{mid} > n_{low}$ ) then
18:          put ( $low, n_{low}, mid, n_{mid}$ ) onto worklist
19:      end if
20:      if ( $n_{up} > n_{mid}$ ) then
21:          put ( $mid, n_{mid}, up, n_{up}$ ) onto worklist
22:      end if
23:  end if
24: end while
```



对分法的主要运算量集中在计算 $\text{Negcount}(A, z)$. 通常是事先将 A 转化成对称三对角矩阵, 这样计算 $A - zI$ 的 LDL^T 分解就非常简单:

$$\begin{aligned} A - zI &= \begin{bmatrix} a_1 - z & b_1 & & \\ b_1 & \ddots & \ddots & \\ & \ddots & \ddots & b_{n-1} \\ & & b_{n-1} & a_n - z \end{bmatrix} \\ &= \begin{bmatrix} 1 & & & \\ l_1 & \ddots & & \\ & \ddots & \ddots & \\ & & l_{n-1} & 1 \end{bmatrix} \begin{bmatrix} d_1 & & & \\ & \ddots & & \\ & & \ddots & \\ & & & d_n \end{bmatrix} \begin{bmatrix} 1 & l_1 & & \\ & \ddots & \ddots & \\ & & \ddots & l_{n-1} \\ & & & 1 \end{bmatrix} \triangleq LDL^T \end{aligned}$$



利用待定系数法,可以得到下面的递推公式

$$d_1 = a_1 - z, \quad d_i = (a_i - z) - \frac{b_{i-1}^2}{d_{i-1}}, \quad i = 2, 3, \dots, n. \quad (5.7)$$

用上面的公式计算 d_i 的运算量约为 $4n$.

注意这里没有选主元,但针对对称三对角矩阵,该算法是非常稳定的,即使当 d_i 有可能很小时,算法依然很稳定.

定理 [Demmel '97] 利用公式 (5.7) 计算所得的 d_i 与精确计算 \hat{A} 的 \hat{d}_i 有相同的符号,故有相同的惯性. 这里 \hat{A} 与 A 非常接近,即

$$\hat{A}(i, i) = a_i, \quad \hat{A}(i, i+1) = b_i(1 + \varepsilon_i),$$

其中 $|\varepsilon_i| \leq 2.5\varepsilon + O(\varepsilon^2)$, 这里的 ε 为机器精度.



- 由于单独调用一次 Negcount 的运算量为 $4n$, 故计算 k 个特征值的总运算量约为 $O(kn)$;
- 当特征值计算出来后, 我们可以使用带位移的反迭代来计算对应的特征向量. 通常只需迭代 1 至 2 次即可, 由于 A 是三对角矩阵, 故计算每个特征向量的运算量为 $O(n)$;
- 当特征值紧靠在一起时, 计算出来的特征向量可能会失去正交性, 此时需要进行再正交化, 可通过 MGS 的 QR 分解来实现.



6 奇异值分解

6.1 二对角化

6.2 Golub-Kahan SVD 算法

6.3 dqds 算法

6.4 Jacobi 算法

奇异值分解 (SVD) 具有十分广泛的应用背景, 因此, 如何更好更快地计算一个给定矩阵的 SVD 是科学与工程计算领域中的一个热门研究课题, 吸引了众多专家进行这方面的研究, 也涌现出了许多奇妙的方法. 本章主要介绍计算 SVD 的常用算法.



对任意矩阵 $A \in \mathbb{R}^{m \times n}$, 其奇异值与对称矩阵 $A^T A$, AA^T 和 $\begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix}$ 的特征值是密切相关的, 故理论上计算对称特征值的算法都可以用于计算奇异值. 但在实际计算中, 我们通常可以利用 SVD 的特殊结构使得算法更加有效和准确.

与计算对称矩阵的特征值类似, 计算一个矩阵 A 的奇异值分解的算法通常分为以下几个步骤 (Jacobi 算法除外):

1. 将 A 二对角化: $B = U_1^T A V_1$, 其中 B 为上二对角矩阵, U_1, V_1 为正交阵;
2. 计算 B 的 SVD: $B = U_2 \Sigma V_2^T$, 其中 Σ 为对角阵, U_2, V_2 为正交阵;
3. 合并得到 A 的 SVD: $A = U_1 B V_1^T = (U_1 U_2) B (V_1 V_2)^T$.




6.1 二对角化

我们知道, 对称矩阵可以通过一系列 Householder 变换转化为对称三对角矩阵. 对于一般矩阵 $A \in \mathbb{R}^{m \times n}$, 我们也可以通过 Householder 变换, 将其转化为二对角矩阵, 即计算正交矩阵 U_1 和 V_1 使得

$$U_1^T A V_1 = B, \quad (5.8)$$

其中 B 是一个实(上)二对角矩阵. 这个过程就称为**二对角化**.

 需要注意的是, 与对称矩阵的对称三对角化不同, A 与 B 是不相似的.



设 $A \in \mathbb{R}^{m \times n}$, 二对角化过程大致如下:

- (1) 首先确定一个 Household 矩阵 $H_1 \in \mathbb{R}^{m \times m}$, 使得 $H_1 A$ 的第一列除第一个元素外, 其它分量都为零, 即

$$H_1 A = \begin{bmatrix} * & * & * & \cdots & * \\ 0 & * & * & \cdots & * \\ 0 & * & * & \cdots & * \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & * & * & \cdots & * \end{bmatrix}.$$



- (2) 再确定一个 Househoid 矩阵 $\tilde{H}_1 \in \mathbb{R}^{(n-1) \times (n-1)}$, 把 $H_1 A$ 的第一行的第 3 至第 n 个元素化为零, 即

$$H_1 A \begin{bmatrix} 1 & 0 \\ 0 & \tilde{H}_1 \end{bmatrix} = \begin{bmatrix} * & * & 0 & \cdots & 0 \\ 0 & * & * & \cdots & * \\ 0 & * & * & \cdots & * \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & * & * & \cdots & * \end{bmatrix}.$$

- (3) 重复上面的过程, 直到把 A 最终化为二对角矩阵.



有了分解 (5.8) 以后, 我们可得

$$A^T A = (U_1 B V_1^T)^T U_1 B V_1^T = V_1 B^T B V_1^T,$$

即 $V_1^T A^T A V_1 = B^T B$. 由于 $B^T B$ 是对称三对角的, 所以这就相当于将 $A^T A$ 三对角化.

整个二对角化过程的运算量约为 $4mn^2 + 4m^2n - 4n^3/3$, 若不需要计算 U_1 和 V_1 , 则运算量约为 $4mn^2 - 4n^3/3$.



二对角矩阵的奇异值分解

设 $B \in \mathbb{R}^{n \times n}$ 是一个二对角矩阵 $B = \begin{bmatrix} a_1 & b_1 & & \\ & \ddots & \ddots & \\ & & \ddots & b_{n-1} \\ & & & a_n \end{bmatrix}$,

则下面三种方法均可将计算 B 的 SVD 转化成计算对称三对角矩阵的特征分解:

- (1) 令 $A = \begin{bmatrix} 0 & B^\top \\ B & 0 \end{bmatrix}$, 置换阵 $P = [e_1, e_{n+1}, e_2, e_{n+2}, \dots, e_n, e_{2n}]$, 则 $T_{ps} = P^\top A P$ 是对称三对角矩阵, 且 T_{ps} 的主对角线元素全为 0, 次对角线元素为 $a_1, b_1, a_2, b_2, \dots, a_{n-1}, b_{n-1}, a_n$. 若 (λ_i, x_i) 是



T_{ps} 的一个特征对, 则

$$\lambda_i = \pm\sigma_i, \quad Px_i = \frac{1}{\sqrt{2}} \begin{bmatrix} v_i \\ \pm u_i \end{bmatrix},$$

其中 σ_i 为 B 一个奇异值, u_i 和 v_i 分别为对应的左和右奇异向量.

(2) 令 $T_{BB^T} = BB^T$, 则

$$T_{BB^T} = \begin{bmatrix} a_1^2 + b_1^2 & a_2 b_1 & & \\ a_2 b_1 & \ddots & \ddots & \\ & \ddots & a_{n-1}^2 + b_{n-1}^2 & a_n b_{n-1} \\ & & a_n b_{n-1} & a_n^2 \end{bmatrix}.$$

T_{BB^T} 的特征值为 B 的奇异值的平方, 且 T_{BB^T} 的特征向量为 B 的左奇异向量.



(3) 令 $T_{B^T B} = B^T B$, 则

$$T_{B^T B} = \begin{bmatrix} a_1^2 & a_1 b_1 & & \\ a_1 b_1 & a_2^2 + b_1^2 & \ddots & \\ & \ddots & \ddots & a_{n-1} b_{n-1} \\ & & a_{n-1} b_{n-1} & a_n^2 + b_{n-1}^2 \end{bmatrix}.$$

$T_{B^T B}$ 的特征值为 B 的奇异值的平方, 且 $T_{B^T B}$ 的特征向量为 B 的右奇异向量.



理论上, 我们可以直接使用 QR 迭代、分而治之法或带反迭代的对分法, 计算三对角矩阵 T_{ps} , T_{BB^T} 和 $T_{B^T B}$ 的特征值和特征向量.

但一般来说, 这种做法并不是最佳的, 原因如下:

- (1) 对 T_{ps} 做 QR 迭代并不划算, 因为 QR 迭代计算所有的特征值和特征向量, 而事实上只要计算正的特征值即可;
- (2) 直接构成 T_{BB^T} 或 $T_{B^T B}$ 是数值不稳定的. 事实上, 这样做可能会使得 B 的小奇异值的精度丢失一半.



下面是一些计算奇异值分解的比较实用的算法.

1. **Golub-Kahan SVD 算法**: 由 Golub 和 Kahan 于 1965 年提出, 是一种十分稳定且高效的计算 SVD 的算法. 主要思想是将带位移的对称 QR 迭代算法隐式地用到 $B^T B$ 上, 在该算法中, 并不需要显示地把 $B^T B$ 计算出来. 该算法也通常就称为 SVD 算法, 是一个基本且实用的算法, 目前仍然是计算小规模矩阵奇异值分解的常用算法.
2. **dqds 算法**: 由 Fernando 和 Parlett 于 1994 年提出, 是计算二对角矩阵所有奇异值的最快算法, 而且能达到很高的相对精度, 包括奇异值很小的情形. 该算法主要基于对 $B^T B$ 的 Cholesky 迭代, 可以看作是 LR 迭代算法的改进. 由于 LR 迭代算法在一定条件下与对称 QR 算法是等价的, 因此该算法也可以看作是 QR 迭代的变形.



3. **分而治之法**: 该算法是计算维数 $n \geq 25$ 的矩阵的所有奇异值和奇异向量的最快算法, 但不能保证小奇异值的相对精度, 即 σ_i 的相对精度为 $O(\varepsilon)\sigma_1$, 而不是 $O(\varepsilon)\sigma_i$.
4. **对分法和反迭代**: 主要用于计算某个区间内的奇异值及对应的奇异向量, 能保证较高的相对精度.
5. **Jacobi 迭代**: 可隐式地对 AA^T 或 $A^T A$ 实施对称 Jacobi 迭代, 能保证较高的相对精度. 最近, Z. Drmač 和 K. Veselić 改进了最初的 Jacobi 算法, 使其变成一个速度快、精度高的实用算法.

我们简要介绍 Golub-Kahan SVD 算法, dqds 算法和 Jacobi 迭代.



6.2 Golub-Kahan SVD 算法

该算法主要思想是将带位移的对称 QR 迭代算法隐式地用到 $B^T B$ 上, 而无需将 $B^T B$ 显式地计算出来.

算法基本框架

Golub-Kahan SVD 算法有时也简称 SVD 算法, 其基本框架是:

- 将矩阵 A 二对角化, 得到上二对角矩阵 B ;
- 用隐式 QR 迭代计算 $B^T B$ 的特征值分解, 即

$$B^T B = Q \Lambda Q^T, \quad \Lambda = \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_n^2). \quad (5.9)$$

- 计算 BQ 的列主元 QR 分解, 即

$$(BQ)P = UR, \quad (5.10)$$

其中 P 是置换矩阵, U 是正交矩阵, R 是上三角矩阵.



由 (5.9) 可知

$$(BQ)^T BQ = \Lambda,$$

因此 BQ 是列正交矩阵 (但不是单位列正交). 再由 (5.10) 可知 $R = U^T(BQ)P$ 也是列正交矩阵. 又 R 是上三角矩阵, 所以 R 必定是对角矩阵. 令 $V = QP$, 则由 (5.10) 可知

$$U^T B V = R.$$

这就是二对角矩阵 B 的奇异值分解.

算法的具体实现可参见相关文献.



6.3 dqds 算法

我们首先介绍针对实对称正定矩阵的 LR 算法, 该算法思想与 QR 迭代算法类似, 但提出时间更早.

算法 6.1 带位移的 LR 算法

- 1: Let T_0 be a given real symmetric positive definite matrix
 - 2: set $i = 0$
 - 3: **while** not converge **do**
 - 4: choose a shift τ_i^2 satisfying $\tau_i^2 < \min\{\lambda(T_i)\}$
 - 5: compute B_i such that $T_i - \tau_i^2 I = B_i^\top B_i$ % Cholesky factorization
 - 6: $T_{i+1} = B_i B_i^\top + \tau_i^2 I$
 - 7: $i = i + 1$
 - 8: **end while**
-



LR 迭代算法在形式上与 QR 迭代算法非常类似. 事实上, 对于不带位移的 LR 迭代算法, 我们可以证明, 两步 LR 迭代等价于一步 QR 迭代.

引理 设 \tilde{T} 是不带位移的 LR 算法迭代两步后生成的矩阵, \hat{T} 是不带位移的 QR 算法迭代一步后生成的矩阵, 则 $\tilde{T} = \hat{T}$.

- (1) LR 算法中要求 T_0 对称正定, 但并不一定是三对角矩阵;
- (2) 由该引理可知, QR 算法与 LR 算法有相同的收敛性.



dqds 算法

该算法是针对三对角的对称正定矩阵 $B^T B$, 其中 B 是二对角矩阵. 在数学上, dqds 算法与 LR 算法是等价的, 但在该算法中, 我们是直接通过 B_i 来计算 B_{i+1} , 从而避免计算中间矩阵 T_{i+1} , 这样也就尽可能地避免了由于计算 $B_i B_i^T$ 而可能带来的数值不稳定性.

下面推导如何从 B_i 直接计算 B_{i+1} . 设

$$B_i = \begin{bmatrix} a_1 & b_1 & & & \\ & a_2 & \ddots & & \\ & & \ddots & b_{n-1} & \\ & & & a_n & \\ & & & & \end{bmatrix}, \quad B_{i+1} = \begin{bmatrix} \tilde{a}_1 & \tilde{b}_1 & & & \\ & \tilde{a}_2 & \ddots & & \\ & & \ddots & \tilde{b}_{n-1} & \\ & & & \tilde{a}_n & \\ & & & & \end{bmatrix}.$$

为了书写方便, 我们记 $b_0 = b_n = \tilde{b}_0 = \tilde{b}_n = 0$. 由 LR 算法 6.1 可知

$$B_{i+1}^T B_{i+1} + \tau_{i+1}^2 I = B_i B_i^T + \tau_i^2 I.$$



比较等式两边矩阵的对角线和上对角线元素, 可得

$$\tilde{a}_k^2 + \tilde{b}_{k-1}^2 + \tau_{i+1}^2 = a_k^2 + b_k^2 + \tau_i^2, \quad k = 1, 2, \dots, n$$

$$\tilde{a}_k \tilde{b}_k = a_{k+1} b_k \quad \text{或} \quad \tilde{a}_k^2 \tilde{b}_k^2 = a_{k+1}^2 b_k^2, \quad k = 1, 2, \dots, n-1.$$

记 $\delta = \tau_{i+1}^2 - \tau_i^2$, $p_k = a_k^2$, $q_k = b_k^2$, $\tilde{p}_k = \tilde{a}_k^2$, $\tilde{q}_k = \tilde{b}_k^2$, 则可得 **qds 算法**:

算法 6.2 qds 算法的单步 ($B_i \rightarrow B_{i+1}$)

- 1: $\delta = \tau_{i+1}^2 - \tau_i^2$
 - 2: **for** $k = 1$ **to** $n - 1$ **do**
 - 3: $\tilde{p}_k = p_k + q_k - \tilde{q}_{k-1} - \delta$
 - 4: $\tilde{q}_k = q_k \cdot (p_{k+1} / \tilde{p}_k)$
 - 5: **end for**
 - 6: $\tilde{p}_n = p_n - \tilde{q}_{n-1} - \delta$
-



qds 算法中的每个循环仅需 5 个浮点运算, 所以运算量较少.

为了提高算法的精确性, 我们引入一个辅助变量 $d_k \triangleq p_k - \tilde{q}_{k-1} - \delta$, 则

$$\begin{aligned}d_k &= p_k - \tilde{q}_{k-1} - \delta \\&= p_k - \frac{q_{k-1}p_k}{\tilde{p}_{k-1}} - \delta \\&= p_k \cdot \frac{\tilde{p}_{k-1} - q_{k-1}}{\tilde{p}_{k-1}} - \delta \\&= p_k \cdot \frac{p_{k-1} - \tilde{q}_{k-2} - \delta}{\tilde{p}_{k-1}} - \delta \\&= \frac{p_k}{\tilde{p}_{k-1}} \cdot d_{k-1} - \delta.\end{aligned}$$

于是就可得到 dqds 算法.



算法 6.3 dqds 算法的单步 ($B_i \rightarrow B_{i+1}$)

- 1: $\delta = \tau_{i+1}^2 - \tau_i^2$
 - 2: $d_1 = p_1 - \delta$
 - 3: **for** $k = 1$ **to** $n - 1$ **do**
 - 4: $\tilde{p}_k = d_k + q_k$
 - 5: $t = p_{k+1} / \tilde{p}_k$
 - 6: $\tilde{q}_j = q_k \cdot t$
 - 7: $d_{k+1} = d_k \cdot t - \delta$
 - 8: **end for**
 - 9: $\tilde{p}_n = d_n$
-

dqds 算法的运算量与 dqs 差不多, 但更精确.



下面的定理显示了 dqds 算法的高精度性质.

定理 以浮点运算对 B 做单步 dqds 迭代, 得到矩阵 \tilde{B} , 该过程等价于

1. 对 B 的每个元素作一个小的相对扰动 (不超过 1.5ϵ), 得到 \tilde{B} ;
2. 对 \tilde{B} 应用精确的 dqds 算法的单步, 得到 \bar{B} ;
3. 对 \bar{B} 的每个元素作一个小的相对扰动 (不超过 ϵ), 得到 \tilde{B} .

因此, B 和 \tilde{B} 的奇异值满足高的相对精度.

关于 dqds 算法中位移的选取, 以及如何判断收敛性, 可参见相关文献



6.4 Jacobi 算法

本节讨论对矩阵 $M = A^T A$ 实施隐式的 Jacobi 算法来计算 A 的奇异值.

我们知道, Jacobi 算法的每一步就是对矩阵作 Jacobi 旋转, 即 $A^T A \rightarrow J^T A^T A J$, 其中 J 的选取将某两个非对角元化为 0. 在实际计算中, 我们只需计算 AJ , 故该算法称为**单边 Jacobi 旋转**.



算法 6.4 单边 Jacobi 旋转的单步

% 对 $M = A^T A$ 作 Jacobi 旋转, 将 $M(i, j), M(j, i)$ 化为 0

- 1: Compute $m_{ii} = (A^T A)_{ii}, m_{ij} = (A^T A)_{ij}, m_{jj} = (A^T A)_{jj}$
- 2: **if** m_{ij} is not small enough **then**
- 3: $\tau = (m_{ii} - m_{jj}) / (2 \cdot m_{ij})$
- 4: $t = \text{sign}(\tau) / (|\tau| + \sqrt{1 + \tau^2})$
- 5: $c = 1 / \sqrt{1 + t^2}$
- 6: $s = c \cdot t$
- 7: $A = AG(i, j, \theta)$ **% $G(i, j, \theta)$ 为 Givens 变换**
- 8: **if** eigenvectors are desired **then**
- 9: $J = J \cdot G(i, j, \theta)$
- 10: **end if**
- 11: **end if**



在上面算法的基础上, 我们可以给出完整的单边 Jacobi 算法.

算法 6.5 单边 Jacobi 算法: 计算 $A = U\Sigma V^T$

```
1: while  $A^T A$  is not diagonal enough do
2:   for  $i = 1$  to  $n - 1$  do
3:     for  $j = i + 1$  to  $n$  do
4:       调用单边 Jacobi 旋转
5:     end for
6:   end for
7: end while
8: compute  $\sigma_i = \|A(:, i)\|_2, i = 1, 2, \dots, n$ 
9:  $U = [u_1, \dots, u_n]$  with  $u_i = A(:, i)/\sigma_i$ 
10:  $V = J$ 
```



Jacobi 算法的特点

- 不需要双对角化, 这样可以避免双对角化引入的误差;
- 可达到相对较高的计算精度;
- **速度较慢**. (目前已有快速的改进算法)

定理 设 $A = DX \in \mathbb{R}^{n \times n}$, 其中 D 为非奇异对角阵, X 非奇异. 设 \hat{A} 是按浮点运算单边 Jacobi 旋转 m 次后所得到的矩阵. 若 A 和 \hat{A} 的奇异值分别为 $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n$ 和 $\hat{\sigma}_1 \geq \hat{\sigma}_2 \geq \dots \geq \hat{\sigma}_n$, 则

$$\frac{|\hat{\sigma}_i - \sigma_i|}{\sigma_i} \leq O(m\varepsilon)\kappa(X).$$

故 X 的条件数越小, 计算矩阵 A 的奇异值时相对误差越小.



7 扰动分析

- 7.1 特征值与 Rayleigh 商
- 7.2 对称矩阵特征值的扰动分析
- 7.3 对称矩阵特征向量的扰动
- 7.4 Rayleigh 商逼近
- 7.5 相对扰动分析



设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, 则有下面的谱分解.

定理 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵. 则存在一个正交矩阵 Q 使得

$$A = Q\Lambda Q^T$$

其中 $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ 是一个实对角矩阵.

这里的 λ_i 就是 A 的特征值, 我们假设 $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$. 令 $Q = [q_1, q_2, \dots, q_n]$, 则 q_i 就是 λ_i 对应的单位正交特征向量.

关于对称矩阵特征值问题的扰动理论, 这里只做一些简单介绍, 若要深入了解这方面的信息, 可以参考相关文献.



7.1 特征值与 Rayleigh 商

定义 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, 向量 $x \in \mathbb{R}^n$ 非零, 则 x 关于 A 的 Rayleigh 商定义为:

$$\rho(x, A) = \frac{x^\top A x}{x^\top x}.$$

有时简记为 $\rho(x)$.

下面是关于 Rayleigh 商的一些基本性质:

- (1) $\rho(\alpha x) = \rho(x), \forall \alpha \in \mathbb{R}, \alpha \neq 0$;
- (2) $\rho(q_i) = \lambda_i, i = 1, 2, \dots, n$;
- (3) 设 $x = \alpha_1 q_1 + \alpha_2 q_2 + \dots + \alpha_n q_n$, 则

$$\rho(x) = \frac{\alpha_1^2 \lambda_1 + \alpha_2^2 \lambda_2 + \dots + \alpha_n^2 \lambda_n}{\alpha_1^2 + \alpha_2^2 + \dots + \alpha_n^2};$$

- (4) $\lambda_n \leq \rho(x) \leq \lambda_1, |\rho(x)| \leq \|A\|_2$.



Courant-Fischer 极小极大定理

实对称矩阵的特征值与 Rayleigh 商之间的一个基本性质是 Courant-Fischer 极小极大定理.

定理 (Courant-Fischer) 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, 其特征值为 $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$, 则有

$$\lambda_k = \max_{U \in \mathbb{S}_k^n} \min_{x \in U, x \neq 0} \frac{x^T A x}{x^T x} = \min_{V \in \mathbb{S}_{n-k+1}^n} \max_{x \in V, x \neq 0} \frac{x^T A x}{x^T x},$$

其中 \mathbb{S}_i^n 表示 \mathbb{R}^n 中所有 i 维子空间构成的集合. 当

$$U = \text{span}\{q_1, \dots, q_k\}, \quad V = \text{span}\{q_k, \dots, q_n\}, \quad x = q_k$$

时, 上式中的等号成立.

(板书)



Rayleigh-Ritz 定理

当 $k = 1$ 和 $k = n$ 时, 就可以得到下面的定理.

定理 (Rayleigh-Ritz) 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, 其特征值为 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, 则有

$$\lambda_1 = \max_{x \in \mathbb{R}^n, x \neq 0} \frac{x^\top A x}{x^\top x}, \quad \lambda_n = \min_{x \in \mathbb{R}^n, x \neq 0} \frac{x^\top A x}{x^\top x}.$$



特征值分隔定理

由极小极大定理, 我们可以得到下面的特征值分隔定理.

定理 (分隔定理) 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, $B = Q^T A Q$, 其中 $Q \in \mathbb{R}^{n \times (n-1)}$ 满足 $Q^T Q = I_{n-1}$. 再设 A 和 B 的特征值分别为

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \quad \text{和} \quad \tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_{n-1},$$

则有

$$\lambda_1 \geq \tilde{\lambda}_1 \geq \lambda_2 \geq \tilde{\lambda}_2 \cdots \geq \tilde{\lambda}_{n-1} \geq \lambda_n.$$



特别地, 在定理 7.4 中, 取 $Q = [e_1, \dots, e_{i-1}, e_{i+1}, \dots, e_n]$, 则可以得到下面的结论.

推论 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, \tilde{A} 是 A 的一个 $n-1$ 阶主子矩阵, A 和 \tilde{A} 的特征值分别为

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \quad \text{和} \quad \tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \dots \geq \tilde{\lambda}_{n-1},$$

则有

$$\lambda_1 \geq \tilde{\lambda}_1 \geq \lambda_2 \geq \tilde{\lambda}_2 \geq \dots \geq \tilde{\lambda}_{n-1} \geq \lambda_n.$$



反复应用上面的推论, 即可得到下面的结论.

推论 设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, \tilde{A} 是 A 的一个 k 阶主子矩阵 ($1 \leq k \leq n-1$), A 和 \tilde{A} 的特征值分别为

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \quad \text{和} \quad \tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_k,$$

则有

$$\lambda_i \geq \tilde{\lambda}_i \geq \lambda_{n-k+i}, \quad i = 1, 2, \dots, k.$$



7.2 对称矩阵特征值的扰动分析

设 $A \in \mathbb{R}^{n \times n}$ 是对称矩阵, 扰动矩阵 $E \in \mathbb{R}^{n \times n}$ 也是对称矩阵, 下面讨论 $A + E$ 的特征值与 A 的特征值之间的关系.

由极小极大定理, 我们可以证明下面的性质.

定理 设 $A \in \mathbb{R}^{n \times n}$ 和 $B = A + E \in \mathbb{R}^{n \times n}$ 都是对称矩阵, 其特征值分别为

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \quad \text{和} \quad \tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_n.$$

假定 E 的最大和最小特征值分别为 μ_1 和 μ_n , 则有

$$\lambda_i + \mu_1 \geq \tilde{\lambda}_i \geq \lambda_i + \mu_n, \quad i = 1, 2, \dots, n.$$

(证明留作练习)



Weyl 定理

根据这个定理, 我们立即可以得到下面的 Weyl 定理.

定理 (Weyl) 设 $A \in \mathbb{R}^{n \times n}$ 和 $B = A + E \in \mathbb{R}^{n \times n}$ 都是对称矩阵, 其特征值分别为 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ 和 $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_n$, 则

$$|\tilde{\lambda}_j - \lambda_j| \leq \|E\|_2, \quad j = 1, 2, \dots, n.$$

该定理的结论可以推广到奇异值情形.



我们首先给出下面的引理.

引理 设 $A \in \mathbb{R}^{m \times n}$ ($m \geq n$) 的奇异值分解为 $A = U\Sigma V$, 其中 $U = [u_1, \dots, u_n] \in \mathbb{R}^{m \times n}$ 为列正交矩阵, $V = [v_1, \dots, v_n] \in \mathbb{R}^{n \times n}$ 为正交矩阵, $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_n)$. 将 U 扩展成 $n \times n$ 的正交矩阵 $[U, \tilde{U}] = [u_1, \dots, u_n, \tilde{u}_1, \dots, \tilde{u}_{m-n}]$, 令

$$H = \begin{bmatrix} 0 & A^\top \\ A & 0 \end{bmatrix} \in \mathbb{R}^{(m+n) \times (m+n)},$$

则 H 对称, 且特征值为 $\pm\sigma_i$ 和 0 (其中 0 至少为 $m - n$ 重特征值), 对应的特征向量分别为 $\frac{\sqrt{2}}{2} \begin{bmatrix} v_i \\ \pm u_i \end{bmatrix}$, $i = 1, 2, \dots, n$, 和 $\begin{bmatrix} 0 \\ \tilde{u}_j \end{bmatrix}$, $j = 1, 2, \dots, m - n$.



由上面的引理和 Weyl 定理 7.8 立即可得

定理 设 $A, B \in \mathbb{R}^{m \times n}$ ($m \geq n$), 它们的奇异值分别为 $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$ 和 $\tilde{\sigma}_1 \geq \tilde{\sigma}_2 \geq \cdots \geq \tilde{\sigma}_n$. 则

$$|\tilde{\sigma}_j - \sigma_j| \leq \|B - A\|_2, \quad j = 1, 2, \dots, n.$$




7.3 对称矩阵特征向量的扰动

定义 设 $A \in \mathbb{R}^{n \times n}$ 的特征值为 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, 则 λ_i 与其余特征值之间的**间隙 (gap)** 定义为

$$\text{gap}(\lambda_i, A) = \min_{j \neq i} |\lambda_j - \lambda_i|.$$

有时简记为 $\text{gap}(\lambda_i)$.

 特征向量的敏感性依赖于其对应的特征值的 gap, 一般来说, gap 越小, 特征向量越敏感.



例 设

$$A = \begin{bmatrix} 1+g & \\ & 1 \end{bmatrix}, \quad E = \begin{bmatrix} 0 & \varepsilon \\ \varepsilon & 0 \end{bmatrix}, \quad (0 < \varepsilon < g)$$

易知 A 的特征值为 $\lambda_1 = 1+g, \lambda_2 = 1$, 特征向量 $q_1 = e_1, q_2 = e_2$.

当 ε 充分小时, $A + E$ 的特征值为 $\hat{\lambda}_{1,2} = 1 + (g \pm \sqrt{g^2 + 4\varepsilon^2})/2$, 单位特征向量为

$$\begin{aligned} \hat{q}_1 &= \beta_1 \cdot \begin{bmatrix} 1 \\ \frac{\sqrt{1 + 4\varepsilon^2/g^2} - 1}{2\varepsilon/g} \end{bmatrix} = \beta_1 \cdot \begin{bmatrix} 1 \\ \frac{\sqrt{(1 + 2\varepsilon^2/g^2)^2 - 4(\varepsilon/g)^4} - 1}{2\varepsilon/g} \end{bmatrix} \\ &\approx \beta_1 \cdot \begin{bmatrix} 1 \\ \frac{(1 + 2\varepsilon^2/g^2) - 1}{2\varepsilon/g} \end{bmatrix} \\ &= \frac{1}{\sqrt{1 + \varepsilon^2/g^2}} \begin{bmatrix} 1 \\ \varepsilon/g \end{bmatrix}, \end{aligned}$$



$$\hat{q}_2 = \beta_2 \cdot \left[\frac{1}{-\sqrt{1 + 4\varepsilon^2/g^2} - 1} \right] \approx \frac{1}{\sqrt{1 + \varepsilon^2/g^2}} \begin{bmatrix} -\varepsilon/g \\ 1 \end{bmatrix},$$

其中 β_1, β_2 为规范化因子.

故特征向量的扰动约为 ε/g , 与特征值的间隙 $\text{gap}(\lambda_i, A) = g$ 成反比.



定理 设 $A = Q\Lambda Q^T$ 和 $A + E = \tilde{Q}\tilde{\Lambda}\tilde{Q}^T$ 分别为对称矩阵 $A \in \mathbb{R}^{n \times n}$ 和 $A + E \in \mathbb{R}^{n \times n}$ 的特征值分解, 其中 $Q = [q_1, q_2, \dots, q_n]$ 和 $\tilde{Q} = [\tilde{q}_1, \tilde{q}_2, \dots, \tilde{q}_n]$ 均为正交矩阵, 且 \tilde{q}_i 为 q_i 对应的扰动特征向量. 用 θ_i 表示 q_i 和 \tilde{q}_i 之间的锐角, 则当 $\text{gap}(\lambda_i, A) > 0$ 时

$$\frac{1}{2} \sin 2\theta_i \leq \frac{\|E\|_2}{\text{gap}(\lambda_i, A)}.$$

类似地, 当 $\text{gap}(\tilde{\lambda}_i, A + E) > 0$ 时

$$\frac{1}{2} \sin 2\theta_i \leq \frac{\|E\|_2}{\text{gap}(\tilde{\lambda}_i, A + E)}.$$

(证明见讲义, 留作自习)



- 🔔 当 $\theta_i \ll 1$ 时, $\frac{1}{2} \sin 2\theta_i \approx \theta_i \approx \sin \theta_i$;
- 🔔 若 $\|E\|_2 \geq \frac{1}{2} \text{gap}(\lambda_i, A)$, 则定理中给出的上界就失去实际意义;
- 🔔 在该定理中, 没有对特征值进行排序;
- 🔔 在实际计算中, 我们通常所知道的是 $\text{gap}(\tilde{\lambda}_i, A + E)$.



7.4 Rayleigh 商逼近

定理 设对称矩阵 $A \in \mathbb{R}^{n \times n}$ 的特征值为 $\lambda_1, \lambda_2, \dots, \lambda_n$.

(1) 若 $x \in \mathbb{R}^n$ 是单位向量, $\beta \in \mathbb{R}$, 则

$$\min_{1 \leq i \leq n} |\lambda_i - \beta| \leq \|Ax - \beta x\|_2; \quad (5.15)$$

(2) 给定非零向量 $x \in \mathbb{R}^n$, 当 $\beta = \rho(x)$ 时, $\|Ax - \beta x\|_2$ 达到最小, 即

$$\min_{\beta \in \mathbb{R}} \|Ax - \beta x\|_2 = \|Ax - \rho(x)x\|_2; \quad (5.16)$$

(3) 令 $r = Ax - \rho(x)x$, 设 λ_i 是离 $\rho(x)$ 最近的特征值, $\text{gap}' = \min_{j \neq i} |\lambda_j - \rho(x)|$, θ 是 x 和 q_i 之间的锐角, 其中 q_i 是 λ_i 对应的单位特征向量, 则

$$\sin \theta \leq \frac{\|r\|_2}{\text{gap}'} \quad \text{且} \quad |\lambda_i - \rho(x)| \leq \frac{\|r\|_2^2}{\text{gap}'}. \quad (5.17)$$



🔔 由(5.15)可知, 在幂迭代和反迭代中可以使用残量 $\|Ax - \tilde{\lambda}x\|_2 < tol$ 作为停机准则, 这里 $\tilde{\lambda}$ 是迭代过程中计算得到的近似特征值. 等式(5.16)则解释了为什么用 Rayleigh 商来近似特征值.

🔔 不等式(5.17)表明 $|\lambda_i - \rho(x)|$ 的值与残量范数 $\|r\|_2$ 的平方成正比, 这个结论是 Rayleigh 商迭代局部三次收敛的基础.



7.5 相对扰动分析


这里主要讨论 A 和 $X^T A X$ 的特征值和特征向量之间的扰动关系, 其中 X 非奇异且满足 $\|X^T X - I\|_2 = \varepsilon$. 这是因为在计算特征向量时, 由于舍入误差的原因, 最后得到的正交矩阵 Q 会带有误差, 从而失去正交性.

定理 (相对 Weyl 定理) 设对称矩阵 A 和 $X^T A X$ 的特征值分别为 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ 和 $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_n$, 令 $\varepsilon = \|X^T X - I\|_2$, 则

$$|\tilde{\lambda}_i - \lambda_i| \leq \varepsilon |\lambda_i| \quad \text{或} \quad \frac{|\tilde{\lambda}_i - \lambda_i|}{|\lambda_i|} \leq \varepsilon \quad (\text{if } \lambda_i \neq 0).$$

(证明见讲义, 留作自习)



 当 X 正交时, $\varepsilon = 0$, 故 X^TAX 与 A 有相同的特征值. 当 X 几乎正交时, ε 很小, 此时 X^TAX 与 A 的特征值几乎相同.

推论 设 G 和 $Y^T GX$ 的奇异值分别为 $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$ 和 $\tilde{\sigma}_1 \geq \tilde{\sigma}_2 \geq \cdots \geq \tilde{\sigma}_n$, 令 $\varepsilon = \max \{ \|X^T X - I\|_2, \|Y^T Y - I\|_2 \}$, 则

$$|\tilde{\sigma}_i - \sigma_i| \leq \varepsilon |\sigma_i| \quad \text{或} \quad \frac{|\tilde{\sigma}_i - \sigma_i|}{|\sigma_i|} \leq \varepsilon \quad (\text{if } \sigma_i \neq 0).$$



下面给出特征向量的相对扰动性质.

定义 设 $A \in \mathbb{R}^{n \times n}$ 的特征值为 $\lambda_1, \lambda_2, \dots, \lambda_n$, 若 $\lambda_i \neq 0$, 则 λ_i 与其他特征值之间的**相对间隙 (relative gap)** 定义为

$$\text{relgap}(\lambda_i, A) = \min_{j \neq i} \frac{|\lambda_j - \lambda_i|}{|\lambda_i|}.$$



定理 设 $A \in \mathbb{R}^{n \times n}$ 和 $X^T A X \in \mathbb{R}^{n \times n}$ 的特征值分解分别为 $A = Q \Lambda Q^T$ 和 $X^T A X = \tilde{Q} \tilde{\Lambda} \tilde{Q}^T$, 其中 $Q = [q_1, q_2, \dots, q_n]$ 和 $\tilde{Q} = [\tilde{q}_1, \tilde{q}_2, \dots, \tilde{q}_n]$ 均为正交矩阵, $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$, $\tilde{\Lambda} = \text{diag}(\tilde{\lambda}_1, \tilde{\lambda}_2, \dots, \tilde{\lambda}_n)$ 且 $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$, $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \dots \geq \tilde{\lambda}_n$. 设 θ_i 表示 q_i 和 \tilde{q}_i 之间的锐角, 令 $\varepsilon_1 = \|I - X^{-T} X^{-1}\|_2$, $\varepsilon_2 = \|X - I\|_2$, 若 $\varepsilon_1 < 1$ 且 $\text{relgap}(\tilde{\lambda}_i, X^T A X) > 0$, 则

$$\frac{1}{2} \sin 2\theta_i \leq \frac{\varepsilon_1}{1 - \varepsilon_1} \cdot \frac{1}{\text{relgap}(\tilde{\lambda}_i, X^T A X)} + \varepsilon_2.$$

(证明见讲义, 留作自习)



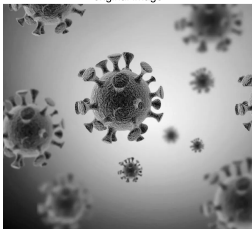
8 应用举例

SVD 与图像压缩

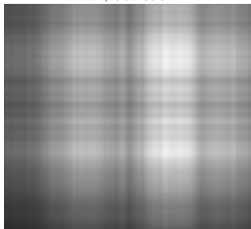
```
1 clear all; close all;
2 A = imread('covid03.jpg');
3 A = rgb2gray(A); A = mat2gray(A);
4
5 [U,D,V] = svd(A); % A = U*D*V'
6
7 KK = [1,50,150,200];
8 for idx = 1:length(KK)
9     k = KK(idx);
10    A1 = U(:,1:k)*D(1:k,1:k)*V(:,1:k)'; % 保留前 k 个奇异值
11    figure, imshow(A1);
12    tit = ['k=',int2str(k)]; title(tit);
13 end
```



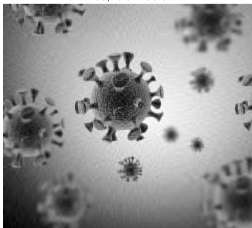
original image



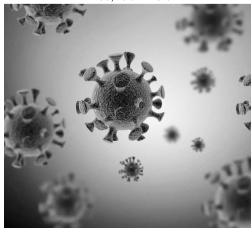
k=1, relerr=3e-01



k=50, relerr=5e-02



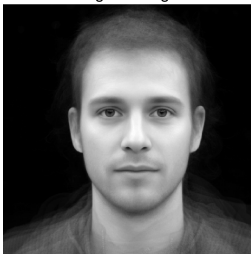
k=150, relerr=2e-02



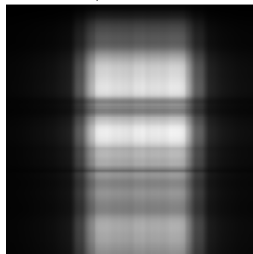
保留前 k 个最大奇异值, 原始图片像素: 809×900



original image



$k=1$, $\text{relerr}=2\text{e-}01$



$k=50$, $\text{relerr}=1\text{e-}02$



$k=150$, $\text{relerr}=5\text{e-}03$



保留前 k 个最大奇异值, 原始图片像素: 512×512