数值算法课程纲要

大数据学院　邵美悦

最后更新: 2024 年 12 月 17 日

目录

1 浮点运算

在计算机中, 数字通常是以浮点格式存储的. 浮点数 x 可以表示为

$$x = (-1)^s \cdot m \cdot \beta^e,$$

其中 $s \in \{0, 1\}$ 是符号位, m 是一个 p 位小数, p 是浮点数的精度, β 是基底 (一般为 2), e 是指数位. 在 IEEE 双精度格式下, 一个浮点数由 64 个比特表示, 精度 $p = 53$. 在浮点运算中有舍入误差的存在, 设 $x \in \mathbb{R}$, $y \in \mathbb{R}$, 则

$$\mathrm{fl}(x \circ y) = (x \circ y)(1 + \delta), \quad |\delta| \leq \boldsymbol{u},$$

其中 $\circ \in \{+, -, \times, \div\}$, $\boldsymbol{u}$ 在使用就近舍入情况下为 $\beta^{1-p}/2$.

在矩阵计算中, 向量内积运算是一类非常重要的运算. 设 $x \in \mathbb{R}^n$, $y \in \mathbb{R}^n$, 则

$$\mathrm{fl}(x^\top y) = (x + \Delta x)^\top y = x^\top (y + \Delta y), \quad |\Delta x| \leq \gamma_n, \quad |\Delta y| \leq \gamma_n,$$

其中 γ_n 有多种不同取法, 常用的有 $(1 + \boldsymbol{u})^n - 1$, $n\boldsymbol{u}/(1 - n\boldsymbol{u})$. 由此可以推得矩阵向量乘的向后误差分析:

$$\mathrm{fl}(Ax) = (A + E)x, \quad |E| \leq \gamma_n |A|.$$

2 线性方程组直接法

2.1 三角方程组求解

设 $L \in \mathbb{C}^{n \times n}$ 是非奇异下三角矩阵, $b \in \mathbb{C}^n$, 三角方程组 $Lx = b$ 可以通过回代法求解. 算法按顺序求出 x_1, x_2, $\ldots$, x_n, 其中

$$x_i = \left(b_i - \sum_{j=1}^{i-1} \ell_{ij} x_j \right) \Big/ \ell_{ii}.$$

该算法的复杂度为 $n^2 + O(n)$. 在矩阵为实数的情况下, 可以证明算法的向后误差满足

$$(L + \Delta L)x = b, \quad |\Delta L| \leq \gamma_n |L|.$$

上三角矩阵的求解也可以得到类似的结论.

2.2 LU 分解

设 $A \in \mathbb{C}^{n\times n}$, A 的 LU 分解即用一系列初等行变换将 A 消成上三角矩阵. 即有

$$L_{n-1}^{-1}\cdots L_2^{-1}L_1^{-1}A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \cdots & a_{2n}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \cdots & a_{3n}^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn}^{(n-1)} \end{bmatrix}$$

为一个上三角矩阵. 我们将这个上三角矩阵记为 U, 并记

$$L \triangleq L_1L_2\cdots L_{n-1} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & \cdots & 0 \\ l_{31} & l_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & 1 \end{bmatrix},$$

则可得 $A = LU$. 该算法的复杂度为 $2n^3/3 + O(n^2)$, 并不是所有非奇异矩阵都有 LU 分解, 矩阵 A 存在 LU 分解的充要条件是 A 的所有顺序主子矩阵都非奇异.

在 LU 分解过程中, 我们称 $a_{kk}^{(k-1)}$ 为主元. 如果 $a_{kk}^{(k-1)} = 0$, 则算法就无法进行下去. 而如果 $|a_{kk}^{(k-1)}|$ 的值很小, 由于舍入误差的存在, 也可能会给计算结果带来很大的影响. 我们可以通过选主元来解决这个问题, 即寻找置换矩阵 P_1, P_2 使得 $P_1AP_2 = LU$. 选主元的目的是在每一步选择一个较大的元素作为主元, 以此来提高算法的稳定性. 选主元的策略主要有以下两种:

1. 部分选主元: 在第 k 步, 查找第 k 列中从 k 行到 n 行的元素，找到绝对值最大的元素 a_{pk}. 如果 a_{pk} 位于第 k 行, 则不需要交换; 否则, 将第 p 行和第 k 行交换.

2. 全选主元: 在第 k 步, 查找从 k 行到 n 行以及从 k 列到 n 列的所有元素, 找到绝对值最大的元素 a_{pq}. 如果 a_{pq} 位于第 k 行和第 k 列, 则不需要交换; 否则先交换行, 再交换列.

全选主元比部分选主元更稳定, 但其工作量较大, 在实际应用中通常使用部分选主元算法. 如果使用部分选主元算法, 很容易得到 L 所有的元素的绝对值都不超过 1.

实数情形下如果 LU 分解能够顺利完成, 则其向后误差满足

$$A + \Delta A = \hat{L}\hat{U}, \quad |A| \leq \gamma_n |\hat{L}||\hat{U}|,$$

其中 $\hat{L}$, $\hat{U}$ 是计算得到的 L, U. 综合三角方程组的舍入误差分析, 我们可以得到实数情况下如果用 LU 分解去求解线性方程组 $Ax = b$, 其向后误差满足

$$(A + \Delta A)x = b, \quad |A| \leq \left(3\gamma_n + \gamma_n^2\right)|\hat{L}||\hat{U}| \leq \gamma_{3n}|\hat{L}||\hat{U}|.$$

如果两边取无穷范数可以得到

$$\|\Delta A\|_\infty \leq \gamma_n \|\hat{L}\|_\infty \|\hat{U}\|_\infty.$$

我们希望向后误差 ΔA 相对于 A 是一个小量, 我们可以定义增长因子为

$$\rho = \max_{i,j,k} \frac{|a_{i,j}^{(k)}|}{\|A\|_\infty},$$

很容易得到 $\|U\|_\infty \leq n\rho\|A\|_\infty$. 在部分选主元的情况下还有 $\|L\|_\infty \leq n$, 我们可以得到

$$(A + \Delta A)x = b, \quad \|\Delta A\|_\infty \leq \gamma_{3n} n^2 \rho \|A\|_\infty.$$

所以若 ρ 比较小, 则部分选主元 LU 分解是向后稳定的. 但理论上并不能保证 ρ 比较小, 可以构造出 ρ 随矩阵维数 n 指数增长的例子.

2.3 其他分解

1. Cholesky 分解: 正定矩阵 $A \in \mathbb{C}^{n\times n}$ 可以被分解为 $A = R^*R$, 其中 R 是对角元为正的上三角矩阵. 该分解可以通过平方根法完成, 且运算量为 $n^3/3 + O(n^2)$, 大约只有 LU 分解的一半.

2. LDL 分解: 对称不定矩阵 $PAP^* = LDL^*$, 其中 L 是单位下三角矩阵, D 是分块对角矩阵 (对角块至多是 2 阶的), P 是置换矩阵.

3 最小二乘问题

3.1 Householder 变换和 Givens 变换

以下假定向量 $x, y \in \mathbb{C}^n$.

1. Householder 变换: Householder 变换形如

$$H = I - 2\frac{ww^*}{w^*w},$$

容易验证 H 是一个 Hermite 酉矩阵, 且 H 可以分解为两个投影矩阵, 一个是向 $\mathrm{span}\{w\}$ 上投影, 另一个是向 $\mathrm{span}\{w\}^\perp$ 上投影,

$$H = \left(I - \frac{ww^*}{w^*w}\right) - \frac{ww^*}{w^*w}.$$

若 $\|x\|_2 = \|y\|_2$, 且 x^*y 是实数, 则可以证明矩阵

$$H = I - 2\frac{(x-y)(x-y)^*}{(x-y)^*(x-y)}$$

满足 $Hx = y$.

2. Givens 变换: 先考虑实数情形, 设 $a, b \in \mathbb{R}$. 容易得到若

$$c = \frac{a}{\sqrt{a^2+b^2}}, \quad s = \frac{b}{\sqrt{a^2+b^2}},$$

则

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sqrt{a^2+b^2} \\ 0 \end{bmatrix}$$

复数时可以类似地构造矩阵

$$\begin{bmatrix} \overline{c} & \overline{s} \\ -s & c \end{bmatrix}.$$

当对矩阵使用 Givens 变换时, 需要用到

$$G(i,j) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & -s & \cdots & c & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix},$$

即将单位阵的 (i,i) 和 (j,j) 位置上的元素用 c 代替, 而 (i,j) 和 (j,i) 位置上的元素分别用 s 和 $-s$ 代替.

3.2 QR 分解

以下假定矩阵 $A \in \mathbb{C}^{m\times n}$, $m \geq n$

1. Householder QR:

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} = H_1 \cdots H_n \begin{bmatrix} R \\ 0 \end{bmatrix},$$

其中

$$H_i = \begin{bmatrix} I_{i-1} & \\ & \tilde{H}_i \end{bmatrix},$$

$\tilde{H}_i$ 是一个 Householder 变换, 整个消元过程如下所示:

$$\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \to \begin{bmatrix} \times & \times & \times & \times \\ \circ & \times & \times & \times \\ \circ & \times & \times & \times \\ \circ & \times & \times & \times \end{bmatrix} \to \begin{bmatrix} \times & \times & \times & \times \\ \circ & \times & \times & \times \\ \circ & \circ & \times & \times \\ \circ & \circ & \times & \times \end{bmatrix} \to \begin{bmatrix} \times & \times & \times & \times \\ \circ & \times & \times & \times \\ \circ & \circ & \times & \times \\ \circ & \circ & \circ & \times \end{bmatrix}.$$

2. Givens QR: Givens QR 与 Householder QR 类似, 它使用一系列 Givens 变换将 A 的下三角部分消为 0, 其常用于具有特殊结构的矩阵的 QR 分解.

3. CGS 和 MGS: CGS (Classical Gram–Schmidt) 和 MGS (Modified Gram–Schmidt) 都通过 Gram–Schmidt 正交化过程来进行 QR 分解.

   - CGS 的第 k 步
     (a) 计算 $r_{ik} = \langle a_k, q_i \rangle, \quad i = 1, \ldots, k-1.$
     (b) 计算 $\tilde{q}_k = a_k - r_{1k}q_1 - r_{2k}q_2 - \cdots - r_{k-1,k}q_{k-1}.$
     (c) 计算 $r_{kk} = \|\tilde{q}_k\|, \quad q_k = \tilde{q}_k / r_{kk}.$
   - MGS 的第 k 步
     (a) 计算 $\tilde{q}_k = a_k.$
     (b) 计算 $r_{ik} = \langle \tilde{q}_k, q_i \rangle, \quad \tilde{q}_k = \tilde{q}_k - r_{ik}q_i, \quad i = 1, \ldots, k-1.$
     (c) 计算 $r_{kk} = \|\tilde{q}_k\|, \quad q_k = \tilde{q}_k / r_{kk}.$

   CGS 的第 k 步投影过程相当于

   $$(I - q_1 q_1^* - \cdots - q_{k-1} q_{k-1}^*) a_k = (I - Q_{k-1} Q_{k-1}^*) a_k,$$

   其中 $Q_{k-1} = [q_1, \ldots, q_{k-1}]$. 而 MGS 的第 k 步投影过程相当于

   $$(I - q_{k-1} q_{k-1}^*) \cdots (I - q_1 q_1^*) a_k.$$

   实际中, MGS 的算法计算得到的 $\hat{Q}$ 正交性更好, 两者通常需要重正交化来提高正交性.

4. Cholesky QR:

   $$R^* R = A^* A, \quad Q = A R^{-1}$$

   对 A^*A 进行 Cholesky 分解, 容易验证 $Q = AR^{-1}$ 是酉矩阵. 该方法相较于之前几种方法速度最快, 但计算得到的 $\hat{Q}$ 正交性与 $(\kappa_2(A))^2$ 相关, 并且当 A 条件数很大时, Cholesky 分解可能失败. 实际使用需要加位移, 以及重正交化的方式提高精度.

3.3 Arnoldi 过程

设 $A \in \mathbb{C}^{n\times n}$, $b \in \mathbb{C}^n$. Krylov 子空间 $\mathcal{K}_m(A, v)$ 定义为

$$\mathcal{K}_m(A, v) = \mathrm{span}\{v, Av, \ldots, A^{m-1}v\}.$$

该空间是矩阵计算中非常重要的一个概念, 它常用于大型稀疏线性方程组或者特征值的求解, 而我们需要该空间的一组正交基. 如果直接对 $[v, Av, \ldots, A^{m-1}v]$ 进行 QR 分解, 由于其条件数较大, 可能导致信息的丢失. 我们需要通过 Arnoldi 过程计算其一组正交基,

1. 对初始向量 v 归一化 $v_1 = v/\|v\|_2$.

2. 第 k 步计算 $w = Av_k$, 令其与 v_1, …, v_k 正交, 以 MGS 为例,

$$h_{ij} = \langle w_j, v_i\rangle, \quad w_j = w_j - h_{ij}v_i, \quad i = 1, \ldots, k.$$

3. 对 w 归一化, $h_{k+1,k} = \|w_k\|_2$, $v_{k+1} = w_k/h_{k+1,k}$.

令 $\underline{H}_m$ 是上面算法得到的 $(m+1)\times m$ 维的上 Hessenberg 矩阵, 它的对应元素就是 h_{ij}, H_m 是 $\underline{H}_m$ 去掉最后一行, 即

$$\underline{H}_m = \left[\begin{array}{ccccc} h_{11} & h_{12} & \cdots & \cdots & h_{1m} \\ h_{21} & h_{22} & \cdots & \cdots & h_{2m} \\ & h_{32} & \ddots & & \vdots \\ & & \ddots & \ddots & \vdots \\ & & & h_{m,m-1} & h_{mm} \\ \hdashline & & & & h_{m+1,m} \end{array}\right], \quad H_m = \begin{bmatrix} h_{11} & h_{12} & \cdots & \cdots & h_{1m} \\ h_{21} & h_{22} & \cdots & \cdots & h_{2m} \\ & h_{32} & \ddots & & \vdots \\ & & \ddots & \ddots & \vdots \\ & & & h_{m,m-1} & h_{mm} \end{bmatrix}.$$

根据上面的等式我们有

$$\begin{aligned} AV_m &= V_m H_m + h_{m+1,m} v_{m+1} e_m^* \\ &= V_{m+1}\underline{H}_m. \end{aligned}$$

该过程等价于对

$$[v_1, Av_1, Av_2, \ldots, Av_m]$$

进行 QR 分解, 实际执行中还可以使用 Householder 正交化方法. Arnoldi 过程可能出现中断, 即 $h_{m+1,m} = 0$ 的情况. 此时算法找到了 A 的一个不变子空间, 这在很多情况下都是有益的.

当 $A = A^*$ 时, H_m 为实对称矩阵, 此时 Arnoldi 过程可以简化为一个三项递推过程, 该过程称为 Lanczos 过程.

3.4 最小二乘问题求解

最小二乘问题需要求解

$$\min_x \|Ax - b\|_2.$$

我们主要考虑 $A \in \mathbb{R}^{m\times n}$ 是满秩的情况, 主要求解方法有一下几种:

1. 法方程法: 利用 Cholesky 分解求解 $A^*Ax = A^*x$. 当问题较为病态时, 计算 A^*A 会显著增加条件数, 不宜用于实际求解, 因此该方法仅用于良态的最小二乘问题的计算.

2. 增广系统法: 直接求解对称系统

   $$Mz = d,$$

   其中

   $$M = \begin{bmatrix} I & A \\ A^* & 0 \end{bmatrix}, \quad z = \begin{bmatrix} r \\ x \end{bmatrix}, \quad d = \begin{bmatrix} b \\ 0 \end{bmatrix}.$$

3. QR 分解法: 设 $A = QR$, 其中 $Q \in \mathbb{C}^{m\times n}$, $R \in \mathbb{C}^{m\times n}$, 则最小二乘解为 $x = R^{-1}Q^*b$.

4. 奇异值分解法: 设 A 的奇异值分解为 $U\Sigma V^*$, 其中 $U \in \mathbb{C}^{m\times n}$, Σ, $V \in \mathbb{C}^{n\times n}$, 则最小二乘解为 $x = V\Sigma^{-1}U^*b$.

3.5 最小二乘问题变体

1. 秩亏损最小二乘问题: 矩阵 A 秩亏损的情况, 可以使用选主元 QR 分解 $AP = QR$ 求解, 其中 P 是一个排列矩阵.

2. 带约束最小二乘问题: 带线性等式约束的最小二乘问题指求解

$$\min_{Cx=d} \|Ax - b\|_2,$$

   其中 $C \in \mathbb{R}^{p\times n}$ 是行满秩矩阵. 当 $\mathrm{rank}(C) = p$ 且 $\mathrm{Ker}(A) \cap \mathrm{Ker}(C) = \{0\}$ 时解唯一. 可以对 C 进行 RQ 分解进行消元, 然后求解.

3. Tikhonov 正则化方法: 当 A 条件数过大时, 通过加入正则化参数 $\lambda > 0$, 求解

$$\min_x \|Ax - b\|_2^2 + \lambda\|x\|_2^2,$$

   来得到原问题的近似解.

4 非对称特征值算法

本节假设 $A \in \mathbb{C}^{n\times n}$.

4.1 乘幂法

最基本的乘幂法可用来求矩阵模最大的特征值以及特征向量,

1. 计算 $y_k = Ax_k$

2. 归一化 y_k:

$$x_{k+1} = \frac{y_k}{\|y_k\|_2}$$

3. 通过计算 Rayleigh 商计算近似特征值: $\lambda_{k+1} = x_{k+1}^* A x_{k+1}$

反幂法: 如果我们将乘幂法作用在 A^{-1} 上, 则可求出 A 的模最小的特征值. 而通过位移 μ 的选择, 我们可以计算出矩阵靠近 μ 的特征值和特征向量,

1. 计算 $y_k = (A - \mu I)^{-1} x_k$

2. 归一化 y_k:

$$x_{k+1} = \frac{y_k}{\|y_k\|_2}$$

3. 通过计算 Rayleigh 商计算近似特征值: $\lambda_{k+1} = x_{k+1}^* A x_{k+1}$

Rayleigh 商迭代, 如果将位移选择为前一次计算的 Rayleigh 商, 就可以得到 Rayleigh 商迭代,

1. 计算 $y_k = (A - \lambda_k I)^{-1} x_k$

2. 归一化 y_k:

$$x_{k+1} = \frac{y_k}{\|y_k\|_2}$$

3. 通过计算 Rayleigh 商计算近似特征值: $\lambda_{k+1} = x_{k+1}^* A x_{k+1}$

Rayleigh 商迭代具有局部二次收敛性. 如果 A 是对称的, 则能达到局部三次收敛. 在 Rayleigh 商迭代中, 由于每次迭代的位移是不同的, 因此每次迭代需要求解一个不同的线性方程组, 这使得运算量增加.

4.2 子空间迭代法

乘幂法只能同时计算一个特征对. 如果想同时计算多个特征对, 我们可以采用多个初始向量进行迭代. 而子空间迭代算法就是基于这种思想, 它能够计算 A 的一个不变子空间, 从而可以同时计算出多个特征值. 设 $Q_0 \in \mathbb{R}^{n \times p}$,

1. 计算 $Y_k = A Q_k$

2. 对 Y_k 进行 QR 分解得到 $Y_k = Q_{k+1} R_{k+1}$.

上式中的 Q_k 将收敛到 A 的一个不变子空间, 其对应的特征值为 $Q_k^* A Q_k$ 的特征值.

4.3 QR 算法

QR 算法的基本思想是通过不断的正交相似变换, 使得 A 趋向于一个上三角形式 (或拟上三角形式).

1. 计算 QR 分解 $A_k = Q_k R_k$

2. $A_{k+1} = R_k Q_k$.

可以证明 QR 算法等价于子空间迭代法中选择 $Q_0 = I_n$.

上面的算法每一步迭代都需要做一次 QR 分解和矩阵乘积, 运算量为 $O(n^3)$, 而且收敛的速度很慢, 会需要很多步迭代. 为了解决这两个问题, 我们需要以下操作:

1. 对 A 使用正交变换将其约化到上 Hessernberg 型: 使用 $n-2$ 个 Householder 变换使得

$$H_{n-2} \cdots H_1 A H_1^* \cdots H_{n-2}^*,$$

为上 Hessernberg 矩阵, 消元过程如下图所示.

$$\begin{pmatrix} \times & \overbrace{\times & \times & \times & \times} \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{pmatrix} \to \begin{pmatrix} \times & \times & \overbrace{\times & \times & \times} \\ \times & \times & \times & \times & \times \\ \circ & \times & \times & \times & \times \\ \circ & \times & \times & \times & \times \\ \circ & \times & \times & \times & \times \end{pmatrix} \to \begin{pmatrix} \times & \times & \times & \overbrace{\times & \times} \\ \times & \times & \times & \times & \times \\ \circ & \times & \times & \times & \times \\ \circ & \circ & \times & \times & \times \\ \circ & \circ & \times & \times & \times \end{pmatrix} \to \begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \circ & \times & \times & \times & \times \\ \circ & \circ & \times & \times & \times \\ \circ & \circ & \circ & \times & \times \end{pmatrix}$$

容易验证该变换等价于 Arnoldi 过程, 初始向量选为 e_1, (可能出现中断). 接下来我们只需要对一个上 Hessernberg 矩阵进行 QR 算法, 这样一次迭代的工作量就可以降到 $O(n^2)$.

2. 选择位移加速收敛: 选择位移 μ_k, 然后

$$\begin{aligned} A_k - \mu_k I &= Q_k R_k, \\ A_{k+1} &= R_k Q_k + \mu_k I. \end{aligned}$$

若 μ_k 是 A 的一个特征值, 容易得到

$$A_{k+1} = \begin{bmatrix} \tilde{A}_{k+1} & * \\ & \mu_k \end{bmatrix}$$

此时一步 QR 算法就将使特征值 μ_k 收敛. 由此可见, 如果 μ_k 与 A 的某个特征值非常接近, 则收敛速度通常会很快.

(a) a_{nn}, 该位移可视为 Rayleigh 商, 因为 $a_{nn} = e_n^* A e_n$.

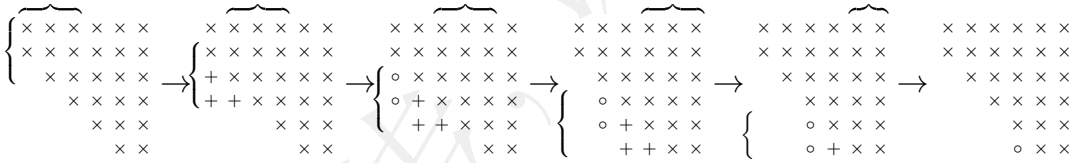
(b) Wilkinson 位移: 选择子矩阵 $\begin{bmatrix} a_{n-1,n-1} & a_{n-1,n} \\ a_{n,n-1} & a_{nn} \end{bmatrix}$ 最接近 a_{nn} 的特征值作为位移.

(c) Francis 位移: 选择子矩阵 $\begin{bmatrix} a_{n-1,n-1} & a_{n-1,n} \\ a_{n,n-1} & a_{nn} \end{bmatrix}$ 的特征值作为双位移.

若 $A \in \mathbb{R}^{n\times n}$, 当使用 Francis 位移时, 子矩阵的特征值可能是一对共轭复数, 为了避免复数运算, 我们会使用隐式双重步位移迭代.

隐式双重步位移迭代依赖于隐式 Q 定理: 设 $H = Q^* A Q \in \mathbb{C}^{n\times n}$ 是一个不可约上 Hessenberg 矩阵, 其中 $Q \in \mathbb{C}^{n\times n}$ 是正交矩阵, 则 Q 的第 2 至第 n 列均由 Q 的第一列所唯一确定 (可相差一个符号).

隐式双重步位移迭代整个过程如下图所示:



设位移选为 μ, $\overline{\mu}$, $(A-\mu I)(A-\overline{\mu} I) = QR$. 算法先构造 Householder 矩阵 H_1 使得 H_1 第一列与 Q 第一列相同, 再使用一系列 Householder 变换将 $H_1^* A H_1$ 重新变为上 Hessenberg 矩阵.

4.4 特征向量的计算

当我们通过 QR 算法计算出矩阵的 Schur 型后, 我们还需要求解其特征向量. 简单起见, 我们设 R 为上三角矩阵, λ 是其单特征值, 则 $R - \lambda I$ 可以写成

$$\begin{bmatrix} R_{11} - \lambda I & R_{12} & R_{13} \\ 0 & 0 & R_{23} \\ 0 & 0 & R_{33} - \lambda I \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = 0.$$

令 $y_2 = 1$, $y_3 = 0$, 此时只需要求解三角方程组

$$(R_{11} - \lambda I)y = R_{12}$$

就可以得到特征值 λ 对应的特征向量. 得到特征向量后, 我们需要对其归一化以免特征向量组成的矩阵条件数过大.

QR 算法中, 如果矩阵的某个次对角线 $a_{i+1,1}$ 很小, 我们可以将其设成 0, 这时 A 就可以写成分块上三角形式, 其中两个对角块都是上 Hessenberg 矩阵. 我们可以对这两个规模相对较小的矩阵使用 QR 算法, 从而可以节省运算量. 经典的判断准则是

$$|a_{i+1,i}| \leq \boldsymbol{u}(|a_{ii}| + |a_{i+1,i+1}|).$$

而如今常用 aggressive early deflation 方法来加速收敛.

4.5 Schur 型中特征值的交换

我们有时需要 Schur 型中的矩阵按照某种顺序排列, 这时我们就需要特征值的交换. 以 2×2 的实矩阵为例, 我们可以构造出正交阵 Q 使得

$$Q^\top \begin{bmatrix} a & d \\ 0 & b \end{bmatrix} Q = \begin{bmatrix} b & d \\ 0 & a \end{bmatrix},$$

其中 Q 为

$$\frac{1}{\sqrt{d^2 + (a-b)^2}} \begin{bmatrix} d & a-b \\ b-a & d \end{bmatrix}.$$

如果涉及到 2×2 块特征值的交换, 则需要使用全选主元的 LU 分解来求解一个 Sylvester 方程.

4.6 特征值扰动

1. 单特征值的扰动: 设 (λ, x) 是 A 的一个单特征值对, E 是一个小扰动, $(\tilde{\lambda}, \tilde{x})$ 是 $A+E$ 的一个特征值对, 并且有 $x^*(\tilde{x}-x)=0$, 则

$$\tilde{\lambda} = \lambda + \frac{1}{y^*x} y^*Ex + O(\|E\|^2),$$
$$\tilde{x} = x - X_\perp (X_\perp^*(A-\lambda I)X_\perp)^{-1} X_\perp^* Ex + O(\|E\|^2),$$

   其中 $X_\perp$ 的列是 $\mathrm{span}\{x\}^\perp$ 的正交基.

2. Bauer–Fike 定理: 设 $A = X\Lambda X^{-1}$, 其中 Λ 是对角矩阵. 如果 $\tilde{\lambda}$ 是 $A+E$ 的特征值, 那么存在一个 A 的特征值 λ 使得

$$|\tilde{\lambda} - \lambda| \le \kappa_2(X)\|E\|_2.$$

3. Weyl 不等式: 设 A, E 都是 Hermite 矩阵, 则

$$\max_{1\le k\le n} |\lambda_k(A+E) - \lambda_k(A)| \le \|E\|_2.$$

4. Hoffman–Wielandt 不等式: 设 A, E 都是 Hermite 矩阵, 则

$$\sum_{k=1}^{n} \big(\lambda_k(A+E) - \lambda_k(A)\big)^2 \le \|E\|_{\mathsf{F}}^2$$

5 对称特征值算法

本节假设 $A = A^\top \in \mathbb{R}^{n\times n}$.

5.1 对称 QR 算法

基本步骤为:

1. 对 A 使用正交变换将其约化到对称三对角矩阵 T.
2. 使用带位移的隐式 QR 算法计算 T 的特征值与特征值向量.

3. 计算 A 的特征向量.

在对称 QR 算法中如果使用 Wilkinson 位移, 算法是整体收敛的, 且常常是渐进三次收敛的.

5.2 Jacobi 算法

Jacobi 算法使用一系列 Givens 变换使得 A 的非对角线元素逐渐减小到 0, 从而使得矩阵 A 对角化. 其基本操作是从 A 中选出一个子矩阵, 然后构造 Given 变换使得

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^\top \begin{bmatrix} a_{ii} & a_{ij} \\ a_{ji} & a_{jj} \end{bmatrix} \begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

为对角阵, 即

$$a_{ij}(c^2 - s^2) + (a_{ii} - a_{jj})cs = 0.$$

如果令

$$\tau = \frac{a_{jj} - a_{ii}}{2a_{ij}}, \quad t = \frac{s}{c},$$

我们需要求解方程

$$t^2 + 2\tau t - 1 = 0,$$

为了保证算法的收敛性, 我们会取

$$t = \frac{\operatorname{sign}(\tau)}{|\tau| + \sqrt{1 + \tau^2}}.$$

从而 $c = 1/\sqrt{1 + t^2}$, $s = ct$. 经过上面的变换后, 非对角线元素的平方和会减少 $2a_{ij}^2$. 常用的选取子矩阵的方法有:

1. 选取 a_{ij} 为所有非对角线元素中绝对值最大的一个, 这就是经典 Jacobi 算法. 可以证明经典 Jacobi 算法是渐近二次收敛的.

2. 逐行扫描来选取 a_{ij}, 这就是循环 Jacobi 迭代算法.

该方法能够达到很高的精度, 但计算速度较慢.

5.3 分而治之法

计算对称三对角矩阵的特征值时, 可以使用分而治之法 (Divide-and-Conquer). 对称三对角矩阵 T 可以写成

$$T = \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} + \rho z z^\top.$$

如果我们已经得到 T_1, T_2 的特征值分解, 上面的问题就可以转化为求解对角阵加上一个秩一矩阵的特征值求解.

下面我们考虑矩阵 $D + \rho z z^\top$ 的特征值求解, 其中 D 是一个对角阵. 根据 $\det(\lambda I - D - \rho z z^\top) = 0$ 等价于

$$\det\left(\begin{bmatrix} \lambda I - D & \rho z \\ z^\top & 1 \end{bmatrix}\right) = 1 - \rho \sum_{i=1}^{n} \frac{z_i^2}{\lambda - d_i} = f(\lambda) = 0,$$

特征值的求解就等价于求特征方程 $f(\lambda)$ 的根, 这可以通过牛顿法等求根方法来实现. 如果 D 对角线元素各不相同, 并且 $\rho \neq 0$, z 的各元素非零, (λ, u) 是 $D + \rho z z^\top$ 的特征对, 则有

1. $z^\top u \neq 0$, $\lambda I - D$ 非奇异.
2. $u = \alpha(\lambda I - D)^{-1} z$.

如果 $z_i = 0$, 则 d_i 是一个特征值. 而当 $d_i = d_{i+1}$ 时, 可以使用 Givens 变换将 z_{i+1} 消为 0, 容易得到 d_i 也是一个特征值.

分而治之法于 1981 年首次提出, 但直到 1995 年才由 Gu 和 Eisenstat 给出了一种快速稳定的实现方式. 该方法速度比 QR 算法快, 但精度没有 QR 算法高.

除了以上几种方法外, 还有 dqds 算法, 二分法, MRRR 算法等计算对称特征值的算法.

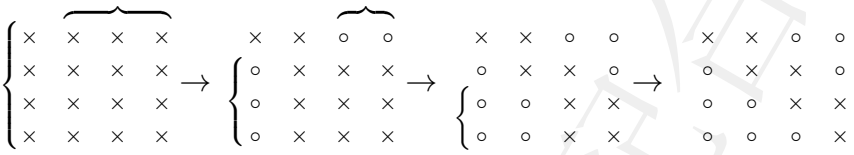
5.4 奇异值分解算法

矩阵 $A \in \mathbb{C}^{m \times n}$ 的奇异值与 A^*A, AA^*, 以及

$$\begin{bmatrix} 0 & A^* \\ A & 0 \end{bmatrix}$$

的特征值密切相关, 奇异值求解的算法也从这几个矩阵特征值求解算法发展而来. Golub–Kahan SVD 算法通过计算 A^*A 的特征值, 来得到奇异值, 主要有以下几个步骤:

1. 将矩阵 A 二对角化, 得到上二对角矩阵 B. 消元过程如下图所示:



   上述过程实际上是对

$$\begin{bmatrix} 0 & A^* \\ A & 0 \end{bmatrix}$$

   行列重排后矩阵的三对角化过程. 当 m 比 n 大很多时, 也可以先对 A 先进行 QR 分解, 再对 R 二对角化.

2. 用隐式 QR 迭代计算 B^*B 的特征值分解, 得到 $B^*B = Q\Lambda Q^*$.

3. 计算 BQ 的列主元 QR 分解, 即

$$(BQ)P = UR.$$

   其中 P 是置换矩阵, U 是正交矩阵, R 是上三角矩阵.

反对称阵 $A = -A^\top \in \mathbb{R}^{n\times n}$ 的特征值问题可以转化为奇异值问题求解.

6 矩阵方程与矩阵函数

6.1 Sylvester 方程

Sylvester 方程需要解

$$AX - XB = C,$$

其中 $A \in \mathbb{C}^{m\times m}$, $B \in \mathbb{C}^{n\times n}$, $C \in \mathbb{C}^{m\times n}$, 未知矩阵 $X \in \mathbb{C}^{m\times n}$. 该方程可以通过以下几种方式求解:

1. 将方程转化为求解

$$(I_n \otimes A - B^\top \otimes I_m)\,\mathrm{vec}(X) = \mathrm{vec}(C),$$

直接求解的时间复杂度为 $O(m^3n^3)$.

2. Bartels–Stewart 算法: 设 A, B^* 的 Schur 分解为 $Q_1T_1Q_1^*$, $Q_2T_2Q_2^*$. 进而方程可以转化为求解

$$T_1Y - YT_2^* = \tilde{C},$$

其中 $Y = Q_1^*XQ_2$, $\tilde{C} = Q_1^*CQ_2$. 此时 $I_n \otimes T_1 - \overline{T}_2 \otimes I_m$ 为上三角矩阵, 易于求解.

3. 只对 A 或 B 其中一个进行 Schur 分解. 以 A 为例, 设 $A = Q_1T_1Q_1^*$. 此时只需求解

$$T_1Y - YB = \tilde{C},$$

其中 $Y = Q_1^*X$, $\tilde{C} = Q_1^*C$. 接下来可以从 Y 的最后一行开始, 一行一行求解 Y, 每次需要求解一个和 B 有关的线性方程组.

6.2 矩阵函数

矩阵计算中常用的矩阵函数有: A^{-1}, $A^{1/2}$, $\exp(A)$, $\log(A)$, $p(A)$, 其中 $p(\lambda)$ 是多项式. 设 A 的 Schur 分解为 $A = QTQ^*$, 则 $f(A) = Qf(T)Q^*$. 此时我们可以用 Schur–Parlett 算法计算上三角矩阵 T 的矩阵函数.

以分块上三角矩阵为例, 设

$$f\left(\begin{bmatrix} T_{11} & T_{12} \\ & T_{22} \end{bmatrix}\right) = \begin{bmatrix} f(T_{11}) & X \\ & f(T_{22}) \end{bmatrix}.$$

如果已经计算出 $f(T_{11})$, $f(T_{22})$, 则根据 $f(T)T = Tf(T)$ 可以得到

$$T_{11}X - XT_{22} = f(T_{11})T_{12} - T_{12}f(T_{22}),$$

我们通过解一个 Sylvester 方程得到 X. 由于 $f(T)$ 对角线易求, 我们可以逐列求得 $f(T)$, 但实际中常使用分块求解的方式.

如果 T_{11} 和 T_{22} 中含有相近特征值, 则上述 Sylvester 方程条件数较大. 我们可以通过 Schur 型特征值交换算法来使得相近的特征值位于同一对角块中, 对于对角块使用有理函数逼近来计算, 其余部分则可以使用上述 Schur–Parlett 算法计算.

6.3 指数函数

指数函数可以使用 scaling and squaring 方法计算, 即利用

$$\exp(A) = \exp\left(\frac{A}{2^k}\right)^{2^k}$$

当 k 足够大时, $A/2^k$ 的范数很小, 此时可以利用泰勒展开或者 Padé 逼近计算 $\exp(A/2^k)$, 然后通过 k 次矩阵乘法计算出 $\exp(A)$.

7 稀疏线性方程组算法

当 A 为大型稀疏矩阵时, 用第 2 章中的算法求解 $Ax = b$ 将变得很困难, 我们可以使用迭代法降低复杂度.

7.1 矩阵分裂迭代法

设 $A = M - N$, 其中 M 非奇异为 A 的一个近似, 则迭代法

$$x^{(k+1)} = M^{-1}Nx^{(k)} + M^{-1}b = Bx^{(k)} + g,$$

被称为矩阵分裂迭代法.

将矩阵 A 写成 $D - L - U$, 其中 D 为 A 的对角线部分, $-L$ 和 $-U$ 分别为 A 的严格下三角和严格上三角部分. 我们可以得到以下两种迭代法.

1. Jacobi 迭代法: 取 $M = D$, $N = L + U$, 对应的迭代矩阵为 $D^{-1}(L + U)$, 写成分量形式为:

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1, j\neq i}^{n} a_{ij}x_j^{(k)}\right), \quad i = 1, 2, \ldots, n.$$

2. Gauss–Seidel 迭代法: 取 $M = D - L$, $N = U$, 对应的迭代矩阵为 $(D-L)^{-1}U$, 写成分量形式为:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right), \quad i = 1, 2, \ldots, n.$$

Gauss–Seidel 迭代法可以视作 Jacobi 迭代的改进, 每次迭代会使用新得到的分量值进行本次迭代.

除了这两种方法外, 还有 SOR 迭代法, HSS 迭代法等矩阵分裂迭代法.

关于矩阵分裂迭代法的收敛性, 我们有以下结果.

1. 对任意迭代初始向量 $x^{(0)}$, 迭代法都收敛的充要条件是 $\rho(B) < 1$.
2. 对于严格对角占优矩阵 A, Jacobi 迭代和 Gauss–Seidel 迭代都收敛.
3. 对于正定矩阵 A, Gauss–Seidel 迭代收敛.

7.2 Krylov 子空间方法

1. GMRES: 该算法一般用于求解非对称的线性方程组. 算法在仿射空间 $x_0 + \mathcal{K}_m(A, r_0)$ 中寻找近似解 x_m 满足

$$x_m = \underset{x \in x_0 + \mathcal{K}_m(A, r_0)}{\arg\min} \|b - Ax\|_2,$$

其中 x_0 是初始解, r_0 是初始残量 $b - Ax_0$. 我们根据 Arnoldi 过程可以产生 $\mathcal{K}_{m+1}(A, r_0)$ 一组正交基 V_{m+1}, 并且有 $AV_m = V_{m+1}\underline{H}_m$, 进而问题可以转化为求解

$$y_m = \arg\min_y \|\beta e_1 - \underline{H}_m y\|_2,$$
$$x_m = x_0 + V_m y_m.$$

这是一个易于求解的最小二乘问题, 可以通过 Givens 变换求解. 如果矩阵 $A = A^*$, GMRES 可以简化为 MINRES.

2. CG: 该算法一般用于求解对称正定的线性方程组. 以 $A \in \mathbb{R}^{n\times n}$ 为例, 设

$$f(x) = \frac{1}{2}x^\top Ax - b^\top x,$$

容易知道 $f(x)$ 的最小值点就是 $Ax = b$ 的解. CG 算法通过寻找一组 A-内积下正交的向量 $p_0, p_1, \ldots$ 作为每步前进的方向最小化 $f(x)$. 迭代更新的方式为

$$x_{k+1} = x_k + \alpha_k p_k, \quad r_{k+1} = r_k - \alpha_k A p_k, \quad p_{k+1} = r_{k+1} + \beta_k p_k,$$

其中 α_k 使得 $r_{k+1}^\top p_k = 0$, β_k 使得 $p_{k+1}^\top A p_k = 0$, 所以

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

可以证明

$$r_k^\top p_j = 0, \quad r_k^\top r_j = 0, \quad j = 0, \ldots, k-1,$$

所以两个参数可以改写成

$$\alpha_k = \frac{r_k^\top r_k}{p_k^\top A p_k}, \quad \beta_k = \frac{r_{k+1}^\top r_{k+1}}{r_k^\top r_k}.$$

关于 CG 算法的收敛性我们以下结论:

(a) 设 $x_* = A^{-1}b$, CG 算法的第 k 步的解 x_k 满足

$$x_k = \mathop{\arg\min}_{x\in x_0+\mathcal{K}_m(A,r_0)} \|x - x_*\|_A$$

(b)

$$\|x_k - x_*\|_A \le 2\left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1}\right)^k \|x_0 - x_*\|_A$$

该结论可以由 Chebyshev 多项式的性质得到.

除了以上几种算法, 还有 FOM, BiCGSTAB 等求解线性方程组的 Krylov 子空间方法. 而实际求解中我们会使用预条件矩阵来加速收敛.

8 其他稀疏矩阵算法

8.1 特征值

本小节假设 $A = A^*$.

1. Rayleigh–Ritz 投影: 设有一组正交基 $V \in \mathbb{C}^{n\times k}$, 如果存在复数 θ, 向量 $\tilde{x} \neq 0$, 使得
$$V^*(A\tilde{x} - \theta\tilde{x}) = 0, \quad \tilde{x} = Vy,$$
则 θ 被称为 Ritz 值, $\tilde{x}$ 被称为 Ritz 向量.

2. Lanczos 算法: 该算法一般用于求解 A 模较大的特征值. 算法使用 Lanczos 过程计算得到 $AV_m = V_mT_m + \beta_m v_{m+1}e_m^*$, 然后计算 T_m 模较大的特征值 θ 以及对应的特征向量 y, 进而将 (θ, V_my) 作为矩阵 A 的近似特征对.

3. FEAST 算法: 该算法可以用于求解 A 在给定区域的特征值. 如图 1 所示, 求

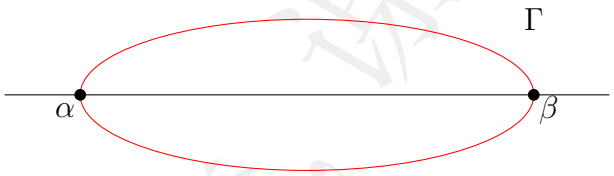


图 1: FEAST 算法示意图

解区间 (α, β) 内的特征值, 先用一条曲线 Γ 包围 (α, β), 根据复变函数的知识可以得到
$$f(A) = \frac{1}{2\pi \mathrm{i}} \int_\Gamma (zI - A)^{-1}\,\mathrm{d}z = Q_{\text{in}}Q_{\text{in}}^*,$$
其中 Q_{in} 是 A 位于区间 (α, β) 内的特征值对应的特征向量. 围道积分的可以通过梯形积分公式或者 Gauss 积分公式来计算, 近似得到 $f(A)$ 后便可以通过子空间迭代得到特征值以及特征向量.

4. Davidson 算法: 设有一组初始正交基 V, 通过 Rayleigh–Ritz 投影得到一组 Ritz 值 Θ 和 Ritz 向量 Q, 算法计算残量
$$R = AQ - Q\Theta,$$

将 $[Q, R]$ 作为下一次迭代的初始向量.

5. PINVIT: 反幂法 $x_{k+1} = A^{-1}x_k$ 可以改写成

$$x_{k+1} = x_k - A^{-1}r_k, \quad r_k = Ax_k - \lambda_k x_k,$$

其中 λ_k 是第 k 步的 Rayleigh 商. PINVIT 用一个容易求解的矩阵 M 代替迭代式中的 A, 即

$$x_{k+1} = x_k - M^{-1}r_k.$$

当 M 满足一定条件时, 上述算法是收敛的.

8.2 最小二乘问题和奇异值算法

Paige–Saunders 二对角化: 设 u_1 为单位向量, 设 $\alpha_1 v_1 = A^* u_1$, 其中 α_1 是归一化系数, 则整个过程为

1. $\beta_{i+1}u_{i+1} = Av_i - \alpha_i u_i$
2. $\alpha_{i+1}v_{i+1} = A^*u_{i+1} - \beta_{i+1}v_i$

最后得到的向量满足

$$AV_k = U_{k+1}B_k, \quad A^*U_k = V_k\tilde{B}_k^*,$$

其中

$$B_k = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix},$$

$\tilde{B}_k$ 是 B_k 的前 k 行组成的矩阵. 上述过程可以视为对矩阵

$$\begin{bmatrix} 0 & A \\ A^* & 0 \end{bmatrix}$$

初始向量选为 $[u_1^*, 0]^*$, 进行 Lanczos 过程.

求解最小二乘问题 $\min_x \|Ax - b\|_2$, 可以令 $u_1 = b/\|b\|_2$, 利用上述过程可以将问题转化为求解

$$\min_y \|AV_k y - b\|_2 = \min_y \|B_k y - \|b\|e_1\|_2.$$

若要求解 A 的奇异值, 则可以用 B_k 的奇异值代替.

8.3 矩阵函数

本节假设 $A = A^*$.

1. $f(A)b$: 设有 Lanczos 过程为 $AV_m = V_m T_m + t_{m+1,m} v_{m+1} e_m^*$, 其中初始向量 $v_1 = b/\|b\|_2$. 可以将上述过程截断, 用 $V_m T_m$ 近似 AV_m, 然后可以得到 $f(A)V_m = V_m f(T_m)$, 故

$$f(A)b = \|b\|_2 V_m f(T_m) e_1.$$

2. $v^* f(A) v$: 进行 Lanczos 过程时, 初始向量选为 $v_1 = v/\|v\|_2$, 则有近似

$$v^* f(A) v = \|v\|_2^2 e_1 f(T_m) e_2.$$

3. $u^* f(A) v$: 利用极化恒等式

$$\langle u, v\rangle = \frac{1}{4}\left(\|u+v\|^2 - \|u-v\|^2 + \mathrm{i}\|u+\mathrm{i}v\|^2 + \mathrm{i}\|u-\mathrm{i}v\|^2\right).$$

将问题转化为第二种形式求解.