第四讲　非对称特征值问题

1 幂迭代法

2 反迭代法

3 正交迭代法

4 QR 迭代法

5 带位移的隐式 QR 迭代法

6 特征向量计算, 广义特征值问题, 应用: 多项式求根

基本约定

- 基本约定 1：$A \in \mathbb{R}^{n\times n}$、非对称、稠密
- 基本约定 2：$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n|$
- 基本约定 3：计算 A 的**全部特征值和/或特征向量**

主要介绍以下方法

- 幂迭代法
- 反迭代法 (位移策略，Rayleigh 商迭代)
- 正交迭代方法
- QR 迭代法

关于稠密矩阵特征值计算的相关参考资料

- Wilkinson, **The Algebraic Eigenvalue Problem**, 1965
- Parlett, **The Symmetric Eigenvalue Problem**, 2nd Eds., 1998
- Stewart, **Matrix Algorithms, Vol II: Eigensystems**, 2001
- Golub and Van Loan, **Matrix Computations**, 4th Eds., 2013
- Bai, Demmel, Dongarra, Ruhe and van der Vorst,
  Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide, 2000

▷ Golub and van der Vorst, **Eigenvalue computation in the 20th century**, JCAM, 2000

1 幂迭代法

1.1 算法介绍

1.2 收敛性分析

1.3 位移策略

1.1 算法介绍

幂迭代 是计算特征值和特征向量的一种简单易用的算法.

虽然简单, 但它却建立了计算特征值和特征向量的一个基本框架.

算法 1 幂迭代法 (Power Iteration)

1: Choose an initial guess $x^{(0)}$ with $\|x^{(0)}\|_2 = 1$
2: set $k = 0$
3: **while** not convergence **do**
4: $\quad y^{(k+1)} = Ax^{(k)}$
5: $\quad x^{(k+1)} = y^{(k+1)}/\|y^{(k+1)}\|_2$
6: $\quad \mu_{k+1} = \langle Ax^{(k+1)}, x^{(k+1)} \rangle$ % 内积
7: $\quad k = k + 1$
8: **end while**

1.2 收敛性分析

假设 1: $A \in \mathbb{R}^{n\times n}$ **可对角化, 即** $A = V\Lambda V^{-1}$, **其中**

$$\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n), \quad V = [v_1, \ldots, v_n] \in \mathbb{C}^{n\times n}, \quad \|v_i\|_2 = 1$$

假设 2: $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$.

由于 V 的列向量组构成 $\mathbb{C}^n$ 的一组基, 因此 $x^{(0)}$ 可表示为

$$x^{(0)} = \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n = V[\alpha_1, \alpha_2, \ldots, \alpha_n]^{\mathsf{T}}.$$

我们假定 $\alpha_1 \neq 0$, 即 $x^{(0)}$ 不属于 $\mathrm{span}\{v_2, v_3, \ldots, v_n\}$

(由于 $x^{(0)}$ 是随机选取的, 从概率意义上讲, 这个假设通常是成立的).

于是我们可得

$$A^k x^{(0)} = (V\Lambda V^{-1})^k V \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} = V\Lambda^k \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} = V \begin{bmatrix} \alpha_1\lambda_1^k \\ \alpha_2\lambda_2^k \\ \vdots \\ \alpha_n\lambda_n^k \end{bmatrix} = \alpha_1\lambda_1^k V \begin{bmatrix} 1 \\ \frac{\alpha_2}{\alpha_1}\left(\frac{\lambda_2}{\lambda_1}\right)^k \\ \vdots \\ \frac{\alpha_n}{\alpha_1}\left(\frac{\lambda_n}{\lambda_1}\right)^k \end{bmatrix}$$

又 $\left|\dfrac{\lambda_i}{\lambda_1}\right| < 1, i = 2, 3, \ldots, n,$ 所以

$$\lim_{k\to\infty} \left(\frac{\lambda_i}{\lambda_1}\right)^k = 0, \quad i = 2, 3, \ldots, n.$$

故当 k 趋向于无穷大时, 向量

$$\left[1,\ \frac{\alpha_2}{\alpha_1}\left(\frac{\lambda_2}{\lambda_1}\right)^k,\ \ldots,\ \frac{\alpha_n}{\alpha_1}\left(\frac{\lambda_n}{\lambda_1}\right)^k\right]^{\mathsf{T}}, \quad k = 0, 1, 2, \ldots$$

收敛到 $e_1 = [1, 0, \ldots, 0]^{\mathsf{T}}$.

所以向量 $x^{(k)} = \dfrac{A^k x^{(0)}}{\|A^k x^{(0)}\|_2}$ 收敛到 $\pm v_1$, 即 λ_1 的特征向量.

而 $\mu_k = (Ax^{(k)}, x^{(k)})$ 则收敛到 $v_1^* A v_1 = \lambda_1$.

幂迭代的收敛速度

幂迭代的收敛快慢取决于 $\dfrac{|\lambda_2|}{|\lambda_1|}$ 的大小, $\dfrac{|\lambda_2|}{|\lambda_1|}$ 越小, 收敛越快.

幂迭代的不足

1. 当 $\dfrac{|\lambda_2|}{|\lambda_1|}$ 接近于 1 时, 收敛速度会非常缓慢.

   如果模最大的特征值不唯一 (比如共轭复数), 则幂迭代可能会失效.

2. 幂迭代只能用于计算 (模) 最大的特征值 (有时也称为主特征值) 和其对应的特征向量.

收缩 (Deflation)

如果需要计算其他特征值, 比如模第二大特征值 λ_2, 则可以在模最大特征值 λ_1 计算出来后, 采用 **收缩 (Deflation)** 技术: 构造酉矩阵 U, 使得

$$U^*AU = \begin{bmatrix} \lambda_1 & A_{12} \\ 0 & A_{22} \end{bmatrix}.$$

然后将幂迭代作用到 A_{22} 上, 就可以求出 λ_2.

以此类推, 可以依次求出所有特征值 (这里假定特征值互不相同).

思考: 上面的收缩技术中的 U 怎么选取?

1.3 位移策略

幂迭代的加速方法

加快幂迭代法的收敛速度 $\Longleftrightarrow$ 尽可能地减小 $\dfrac{|\lambda_2|}{|\lambda_1|}$

位移策略 (shift): 计算 $A-\sigma I$ 的特征值

选取位移 σ, 满足

(1) $\lambda_1-\sigma$ 是 $A-\sigma I$ 的模最大的特征值 $\quad\rightarrow\quad$ 确保计算模最大特征值 λ_1

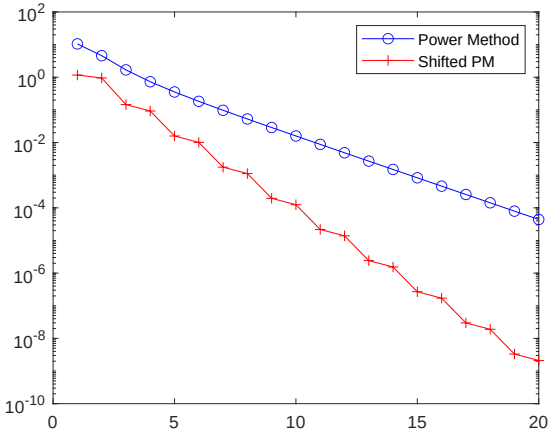
(2) $\max\limits_{2\le i\le n}\left|\dfrac{\lambda_i-\sigma}{\lambda_1-\sigma}\right|$ 尽可能地小 $\quad\rightarrow\quad$ 加速

例 设 $A = X\Lambda X^{-1}$, 分别用幂迭代法和带位移幂迭代法计算 A 的主特征值.

```
itermax = 20; % 最大迭代步数
Lam = [9, 5, 3, 1]; % 给定特征值
n = length(Lam); X = rand(n);
A = (X*diag(Lam))/X; % 以 Lam 为特征值的矩阵
x0 = ones(n,1)/sqrt(n); % 迭代初始向量

% power method
x = x0;
for k = 1 : itermax
    x = A*x; x = x / norm(x); % 近似特征向量
    err(k) = abs(dot(A*x,x) - 9); % 近似特征值的误差
end

% shifted power method
x = x0;
sigma = 3;
B = A - sigma*eye(n);
for k = 1 : itermax
    x = B*x; x = x / norm(x); % 近似特征向量
    err_s(k) = abs(dot(B*x,x) + sigma - 9); % 近似特征值的误差
end
```



几点说明

- 缺点: σ 很难选取; 加速效果有限.
- 与反迭代相结合, 能起到很好的加速效果

2 反迭代方法

2.1 算法介绍

2.2 Rayleigh 商迭代

2.1 算法介绍

将幂迭代作用到 A^{-1} 上，从而计算 A 的模最小特征值, 这就是 反迭代法

反迭代法通常与位移策略结合使用.

算法 2 带位移的反迭代法 (Inverse Iteration)

1: Choose a scalar σ and an initial vector $x^{(0)}$ with $\|x^{(0)}\|_2 = 1$
2: set $k = 0$
3: **while** not convergence **do**
4: $\quad y^{(k+1)} = (A - \sigma I)^{-1} x^{(k)}$
5: $\quad x^{(k+1)} = y^{(k+1)} / \|y^{(k+1)}\|_2$
6: $\quad \mu_{k+1} = \langle A x^{(k+1)}, x^{(k+1)} \rangle$
7: $\quad k = k + 1$
8: **end while**

- μ_k 收敛到距离 σ 最近的特征值, $x^{(k)}$ 收敛到对应的特征向量
- 理论上, 反迭代 + 位移策略, 可以计算矩阵的任意一个特征值

优点

(1) 若 σ 与某个特征值 λ_k 非常接近, 则反迭代的收敛速度非常快.

(2) 只要选取合适的位移 σ, 就可以计算 A 的任意一个特征值.

缺点

(1) 每步迭代需要解一个线性方程组 $(A-\sigma I)y^{(k+1)} = x^{(k)}$

(2) 与幂迭代一样, 反迭代法一次只能求一个特征值.

(3) 怎样选取位移 σ? $\rightarrow$ Rayleigh 商: 动态选取, 自动调整

2.2 Rayleigh 商迭代

为什么 Rayleigh 商

μ_k 会逐渐收敛到某个特征值 ← 幂迭代的收敛性

Rayleigh 商迭代: 以 Rayleigh 商为位移的反迭代, 简称 RQI

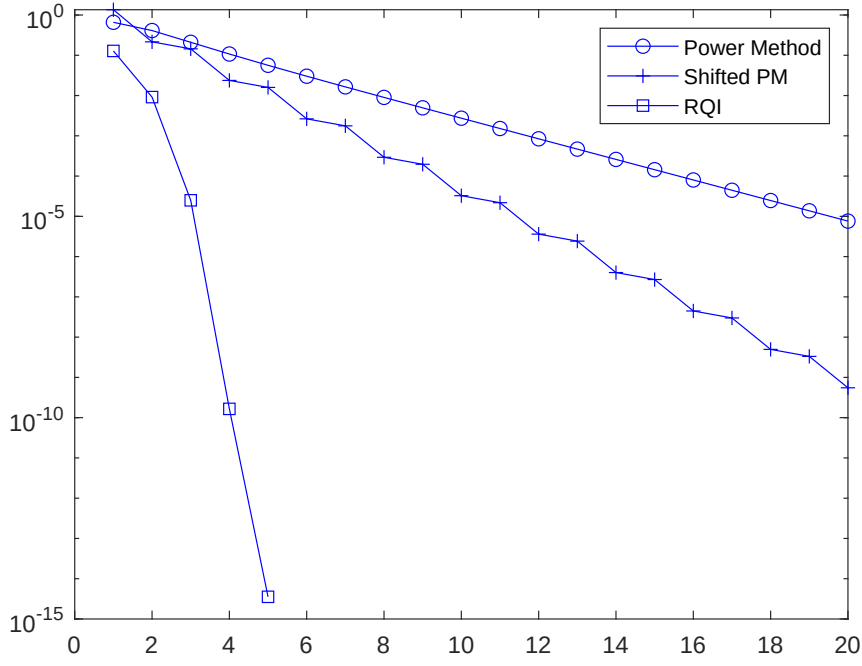
算法 3 Rayleigh 商迭代 (Rayleigh Quotient Iteration, RQI)

1: Choose an initial vector $x^{(0)}$ with $\|x^{(0)}\|_2 = 1$
2: set $k = 0$
3: compute $\sigma = (x^{(0)})^* A x^{(0)}$
4: **while** not converge **do**
5: $\quad y^{(k+1)} = (A - \sigma I)^{-1} x^{(k)}$
6: $\quad x^{(k+1)} = y^{(k+1)} / \|y^{(k+1)}\|_2$
7: $\quad \mu_{k+1} = \langle x^{(k+1)}, A x^{(k+1)} \rangle$
8: $\quad \sigma = \mu_{k+1}$ % 动态位移
9: $\quad k = k + 1$
10: **end while**

例 设 $A = X\Lambda X^{-1}$, 其中 Λ 为对角矩阵, 用 Rayleigh 商迭代计算 A 的特征值.

(见 Eig_Rayleigh.m)

```
x = x0;
sigma = dot(A*x,x);
for k = 1 : itermax
    x = (A-sigma*eye(n)) \ x;
    x = x / norm(x);
    sigma = dot(A*x,x);
    [tmp,idx] = min(abs(Lam - sigma)); % 找出与 sigma 最近的特征值
    err_R(k) = abs( sigma - Lam(idx)); % 与最近特征值之间的误差
end
```


Power Method
Shifted PM
RQI
10^{0}
10^{-5}
10^{-10}
10^{-15}
0
2
4
6
8
10
12
14
16
18
20

RQI 算法的收敛性

一般来说, 如果 Rayleigh 商迭代收敛到 A 的一个单特征值, 则至少是二次收敛的, 即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, 详情见后面的**对称特征值问题**.

缺点:

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, 这使得运算量大大增加.



RQI 非常适合 三对角矩阵 的特征值计算.

3 正交迭代

为什么正交迭代

出发点: 同时计算多个特征值/特征向量

策略: 采用多个初始向量, 希望收敛到 A 的一个**不变子空间**



正交迭代 (orthogonal iteration) 也称为 subspace iteration 或 simultaneous iteration.

算法 4 正交迭代法 (Orthogonal Iteration)

1: Choose an $n \times p$ column orthogonal matrix Z_0
2: set $k = 0$
3: **while** not convergence **do**
4: compute $Y_{k+1} = AZ_k$
5: $Y_{k+1} = Z_{k+1}\hat{R}_{k+1}$ % QR 分解
6: $k = k + 1$
7: **end while**

说明

在算法中使用 QR 分解是为了保持 Z_k 的列正交性, 使得其列向量组构成子空间 $\text{span}\{A^k Z_0\}$ 的一组正交基. 一方面提高算法的数值稳定性, 另一方面避免所有列都收敛到最大特征值所对应的特征向量.

收敛性分析

定理 设 A 可对角化, 给定正整数 $p\,(1 \le p \le n)$, 且 $|\lambda_1| \ge \cdots \ge |\lambda_p| > |\lambda_{p+1}| \ge \cdots \ge |\lambda_n|$, 则 $\mathrm{span}\{Z_k\}$ 收敛到 A 的一个 p 维不变子空间.

(留作课外自习)

注记

- 如果 A 不可对角化, 利用 Jordan 标准型, 可以到同样的结论.
- 如果取 $Z_0 = I$, 在一定条件下, 收敛到 A 的 Schur 分解.

4 QR 迭代

4.1 算法介绍

4.2 QR 迭代与幂迭代的关系

4.3 QR 迭代与反迭代的关系

4.4 QR 迭代与正交迭代的关系

4.5 QR 迭代的收敛性

4.6 带位移的 QR 迭代

4.1 算法介绍

基本思想

通过一系列正交相似变换, 使得 A 趋向于拟上三角形式 (实 Schur 分解)

算法 5 QR 迭代法 (QR Iteration)

1: Set $A_1 = A$ and $k = 1$
2: **while** not convergence **do**
3: $[Q_k, R_k]$ = `qr`(A_k) % QR 分解
4: compute $A_{k+1} = R_k Q_k$
5: $k = k + 1$
6: **end while**

基本性质: 保持正交相似性

在 QR 迭代法中, 我们有

$$A_{k+1} = R_k Q_k = (Q_k^{\mathsf{T}} Q_k) R_k Q_k = Q_k^{\mathsf{T}} (Q_k R_k) Q_k = Q_k^{\mathsf{T}} A_k Q_k.$$

由这个递推关系可得

$$A_{k+1} = Q_k^{\mathsf{T}} A_k Q_k = \cdots = Q_k^{\mathsf{T}} Q_{k-1}^{\mathsf{T}} \cdots Q_1^{\mathsf{T}} A Q_1 \cdots Q_{k-1} Q_k.$$

记 $\tilde{Q}_k = Q_1 \cdots Q_{k-1} Q_k = \left[\tilde{q}_1^{(k)}, \tilde{q}_2^{(k)}, \ldots, \tilde{q}_n^{(k)}\right]$, 则

$$A_{k+1} = \tilde{Q}_k^{\mathsf{T}} A \tilde{Q}_k \tag{4.1}$$

即 A_{k+1} 与 A 正交相似.

4.2 QR 迭代与幂迭代的关系

记 $\tilde{R}_k = R_k R_{k-1} \cdots R_1$, 则有

$$\begin{aligned}\tilde{Q}_k \tilde{R}_k &= \tilde{Q}_{k-1}(Q_k R_k)\tilde{R}_{k-1} = \tilde{Q}_{k-1}(A_k)\tilde{R}_{k-1} \\ &= \tilde{Q}_{k-1}(\tilde{Q}_{k-1}^{\mathsf{T}} A \tilde{Q}_{k-1})\tilde{R}_{k-1} \\ &= A\tilde{Q}_{k-1}\tilde{R}_{k-1},\end{aligned}$$

由此递推下去, 即可得

$$\tilde{Q}_k \tilde{R}_k = A^{k-1}\tilde{Q}_1 \tilde{R}_1 = A^{k-1} Q_1 R_1 = A^k$$

故

$$\tilde{Q}_k \tilde{R}_k e_1 = A^k e_1$$

假设 $|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$, 则当 k 充分大时, $A^k e_1$ 收敛到 A 的模最大特征值 λ_1 所对应的特征向量.

$\Longrightarrow$ 故 $\tilde{Q}_k$ 的第一列 $\tilde{q}_1^{(k)}$ 也收敛到 λ_1 所对应的特征向量

因此, 当 k 充分大时, $A\tilde{q}_1^{(k)} \to \lambda_1 \tilde{q}_1^{(k)}$

由 $A_{k+1} = \tilde{Q}_k^{\mathsf{T}} A \tilde{Q}_k$ 可知, A_{k+1} 的第一列

$$A_{k+1}(:,1) = \tilde{Q}_k^{\mathsf{T}} A \tilde{q}_1^{(k)} \to \lambda_1 \tilde{Q}_k^{\mathsf{T}} \tilde{q}_1^{(k)} = \lambda_1 e_1$$

结论

A_{k+1} 的第一列的第一个元素收敛到 λ_1, 而其它元素都趋向于 0.
收敛速度取决于 $|\lambda_2/\lambda_1|$ 的大小.

4.3 QR 迭代与反迭代的关系

观察 $\tilde{Q}_k$ 的最后一列. 由 $A_{k+1} = \tilde{Q}_k^{\mathsf{T}} A \tilde{Q}_k$ 可知

$$A\tilde{Q}_k = \tilde{Q}_k A_{k+1} = \tilde{Q}_k Q_{k+1} R_{k+1} = \tilde{Q}_{k+1} R_{k+1},$$

所以有 $\tilde{Q}_{k+1} = A\tilde{Q}_k R_{k+1}^{-1}$.

由于 $\tilde{Q}_{k+1}$ 和 $\tilde{Q}_k$ 都是正交矩阵, 上式两边转置后求逆, 可得

$$\tilde{Q}_{k+1} = \left(\tilde{Q}_{k+1}^{\mathsf{T}}\right)^{-1} = \left(\left(R_{k+1}^{-1}\right)^{\mathsf{T}} \tilde{Q}_k^{\mathsf{T}} A^{\mathsf{T}}\right)^{-1} = \left(A^{\mathsf{T}}\right)^{-1} \tilde{Q}_k R_{k+1}^{\mathsf{T}}.$$

观察等式两边矩阵最后一列, 可得 $\tilde{q}_n^{(k+1)} = c_1 \left(A^{\mathsf{T}}\right)^{-1} \tilde{q}_n^{(k)}$ (c_1 为某常数)

依此类推, 可知

$$\tilde{q}_n^{(k+1)} = c\left(A^{\mathsf{T}}\right)^{-k}\tilde{q}_n^{(1)} \qquad (c \text{ 为某常数})$$

假定 $|\lambda_{n-1}| > |\lambda_n| > 0$, 则 λ_n^{-1} 是 $\left(A^{\mathsf{T}}\right)^{-1}$ 的模最大特征值.

由幂迭代可知, $\tilde{q}_n^{(k+1)}$ 收敛到 λ_n^{-1} 所对应的特征向量, 即

$$\left(A^{\mathsf{T}}\right)^{-1}\tilde{q}_n^{(k+1)} \to \lambda_n^{-1}\tilde{q}_n^{(k+1)} \quad (k \to \infty)$$

所以

$$A^{\mathsf{T}}\tilde{q}_n^{(k)} \to \lambda_n\tilde{q}_n^{(k)} \quad (k \to \infty)$$

由 $A_{k+1} = \tilde{Q}_k^{\mathsf{T}}A\tilde{Q}_k$ 可知, A_{k+1}^{T} 的最后一列

$$A_{k+1}^{\mathsf{T}}(:, n) = \tilde{Q}_k^{\mathsf{T}}A^{\mathsf{T}}\tilde{q}_n^{(k)} \to \lambda_n\tilde{Q}_k^{\mathsf{T}}\tilde{q}_n^{(k)} = \lambda_n e_n.$$

结论

A_{k+1} 的最后一行的最后一个元素收敛到 λ_n, 而其它元素都趋向于 0. 收敛速度取决于 $|\lambda_n/\lambda_{n-1}|$ 的大小.

4.4 QR 迭代与正交迭代的关系

下面的定理给出了 QR 迭代与正交迭代 ($Z_0 = I$) 之间的关系.

定理 假定正交迭代法和 QR 算法中所涉及的 QR 分解都是唯一的, 设 A_k 是由 QR 迭代法生成的矩阵, Z_k 是由正交迭代法 (取 $Z_0 = I$) 生成的矩阵, 则有

$$A_{k+1} = Z_k^{\mathsf{T}} A Z_k.$$

4.5 QR 迭代的收敛性

定理 设 $A = V\Lambda V^{-1} \in \mathbb{R}^{n\times n}$, 其中 $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$, 且 $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n| > 0$. 若 V^{-1} 的所有顺序主子矩阵都非奇异（即 V^{-1} 存在 LU 分解）, 则 A_k 的对角线以下的元素均收敛到 0.

(板书)

注记

需要指出的是, 由于 D_k 的元素不一定收敛, 故 A_{k+1} 对角线以上（不含对角线）的元素不一定收敛, 但这不妨碍 A_{k+1} 的对角线元素收敛到 A 的特征值（即 A_{k+1} 的对角线元素是收敛的）.

例 QR 迭代法演示 (见 Eig_QR.m). 设

$$A = X \begin{bmatrix} 9 & & & \\ & 5 & & \\ & & 3 & \\ & & & 1 \end{bmatrix} X^{-1},$$

其中 X 是由 MATLAB 随机生成的非奇异矩阵.

在迭代过程中, 对于 A_k 的下三角部分中元素, 如果其绝对值小于某个阈值 tol, 则直接将其设为 0, 即

$$a_{ij}^{(k)} = 0 \quad \text{if} \quad i > j \text{ and } |a_{ij}^{(k)}| < \text{tol}.$$

这里我们取 $\text{tol} = 10^{-6} \max\limits_{1 \le i,j \le n} \{|a_{ij}^{(k)}|\}$, 迭代过程如下:

```
iter_max = 100;            % 最大迭代步数
Lam = [9, 5, 3, 1];        % 特征值
n = length(Lam);
rng(2015); X = rand(n);    % 随机矩阵
A = (X*diag(Lam))/X        % 以 Lam 为特征值的矩阵
tol = max(abs(A(:)))/1e6; % 下三角部分中小于 tol 的将直接设为 0

for k = 1 : iter_max
    [Q,R] = qr(A);
    A = R*Q;
    L = A - triu(A);       % A 的下三角部分
    L(find(abs(L)<tol)) = 0; % 将绝对值小于 tol 的设为 0
    A = L + triu(A);
    fprintf('k = %d\n',k); A % 输出 A_k

    % 收敛判断
    if (max(abs(L(:))) < tol)
        break;
    end
end
```

```
A =
   6.5629e+00   3.1505e+00   2.4882e+00  -4.5006e+00
   3.1564e+00   4.6079e+00   1.4346e+00  -2.9295e+00
  -3.5367e-02   9.7647e+00   7.7607e+00  -8.7044e+00
   3.7514e+00   2.4217e+00   5.2685e-01  -9.3141e-01
A_7 =
   1.0079e+01   2.0598e+00  -8.7382e-02  -1.4010e+01
  -2.6356e+00   3.9694e+00   5.3709e+00   2.8474e+00
  -1.0317e-02  -1.8888e-02   2.9523e+00  -1.4913e+00
            0  -1.4296e-05   1.3377e-03   9.9898e-01
A_8 =
   9.8306e+00   3.5979e+00  -1.4282e+00   1.4272e+01
  -1.1084e+00   4.1983e+00   5.1778e+00   7.8545e-01
  -2.9432e-03  -1.2199e-02   2.9714e+00   1.5095e+00
            0            0  -4.5563e-04   9.9966e-01
A_12 =
   9.0830e+00   4.6472e+00  -2.4491e+00   1.3798e+01
  -7.2867e-02   4.9207e+00   4.7783e+00   3.7229e+00
  -2.9534e-05  -1.5694e-03   2.9963e+00   1.5315e+00
            0            0            0   1.0000e+00
```

```
A_13 =
   9.0460e+00   4.6811e+00   2.4859e+00  -1.3767e+01
  -3.9787e-02   4.9562e+00  -4.7591e+00  -3.8330e+00
            0   9.3992e-04   2.9978e+00   1.5328e+00
            0            0            0   1.0000e+00
A_22 =
   9.0002e+00   4.7219e+00  -2.5302e+00   1.3729e+01
  -1.9625e-04   4.9998e+00   4.7355e+00   3.9669e+00
            0            0   3.0000e+00   1.5346e+00
            0            0            0   1.0000e+00

A_28 =
   9.0000e+00   4.7221e+00  -2.5304e+00   1.3729e+01
            0   5.0000e+00   4.7354e+00   3.9675e+00
            0            0   3.0000e+00   1.5346e+00
            0            0            0   1.0000e+00
```

4.6 带位移的 QR 迭代

可以采用 位移策略 和 反迭代 思想来加快 QR 迭代的收敛速度.

算法 6 带位移的 QR 迭代法 (QR Iteration with shift)

1: Set $A_1 = A$ and $k = 1$
2: **while** not convergence **do**
3: Choose a shift σ_k
4: $[Q_k, R_k]$ = `qr`$(A_k - \sigma_k I)$ % QR 分解
5: Compute $A_{k+1} = R_k Q_k + \sigma_k I$
6: $k = k + 1$
7: **end while**

正交相似性: $A_{k+1} = R_k Q_k + \sigma_k I = (Q_k^{\mathsf{T}} Q_k) R_k Q_k + \sigma_k I$

$= Q_k^{\mathsf{T}} (A_k - \sigma_k I) Q_k + \sigma_k I = Q_k^{\mathsf{T}} A_k Q_k$

位移 σ_k 的选取

在前面的分析可知, $A_{k+1}(n,n)$ 收敛到 A 的模最小特征值.

若 σ_k 就是 A 的一个特征值, 则 $A_k - \sigma_k I$ 的模最小特征值为 0, 故 QR 算法迭代一步就收敛. 此时

$$A_{k+1} = R_k Q_k + \sigma_k I = \begin{bmatrix} A_{k+1}^{(n-1)\times(n-1)} & * \\ 0 & \sigma_k \end{bmatrix}.$$

A 的其它特征值可通过对 $A_{k+1}^{(n-1)\times(n-1)}$ 使用带位移 QR 迭代法得到.

通常, 如果 σ_k 与 A 的某个特征值非常接近, 则收敛速度通常会很快. 由于 $A_k(n,n)$ 收敛到 A 的一个特征值, 所以在实际使用中, 一个比较直观的位移选择策略是 $\sigma_k = A_k(n,n)$. 事实上, 这样的位移选取方法通常会使得 QR 迭代法有二次收敛速度.

例 带位移的 QR 迭代法演示 (见 Eig_QR_shift.m).

所有数据和设置与例 4.1 相同, 在迭代过程中, 取 $\sigma_k = A_k(n,n)$.

如果 $A_k(n,n)$ 已经收敛, 则取 $\sigma_k = A_k(n-1,n-1)$.

```
L = A - triu(A); % A 的下三角部分
if (max(abs(L(:))) < tol)
    return;
end

% 确定位移 sigma
idx = n;
while (idx > 1)
    if ( sum(abs(L(idx,:)))>0 )
        sigma = A(idx,idx); break;
    else
        idx = idx - 1;
    end
end

% 开始迭代
for k = 1 : iter_max
```

```
    [Q,R] = qr(A-sigma*eye(n));
    A = R*Q + sigma*eye(n);

    L = A - triu(A);        % A 的下三角部分
    L(find(abs(L)<tol)) = 0; % 将绝对值小于 tol 的设为 0
    A = L + triu(A);

    fprintf('k = %d, sigma=%.4e\n',k,sigma);

    % 收敛判断
    if (max(abs(L(:))) < tol)
        break;
    end

    % 确定位移
    while (idx>1)
        if (sum(abs(L(idx,:)))>0)
            sigma = A(idx,idx); break;
        end
        idx = idx - 1;
    end
end
```

```
A =
   6.5629e+00   3.1505e+00   2.4882e+00  -4.5006e+00
   3.1564e+00   4.6079e+00   1.4346e+00  -2.9295e+00
  -3.5367e-02   9.7647e+00   7.7607e+00  -8.7044e+00
   3.7514e+00   2.4217e+00   5.2685e-01  -9.3141e-01
A_5 =
   5.5186e+00  -3.0411e-01   4.4529e+00  -5.1700e+00
  -4.9782e+00   8.5660e+00   3.0148e+00   1.3331e+01
  -3.9116e-02  -1.7945e-03   2.9153e+00  -1.4587e+00
            0            0            0   1.0000e+00
A_7 =
   9.4467e+00   4.2553e+00  -2.0222e+00  -1.4068e+01
  -4.6678e-01   4.5533e+00   4.9737e+00  -2.5126e+00
            0            0   3.0000e+00  -1.5346e+00
            0            0            0   1.0000e+00
A_10 =
   9.0000e+00  -4.7221e+00   2.5304e+00   1.3729e+01
            0   5.0000e+00   4.7354e+00  -3.9676e+00
            0            0   3.0000e+00  -1.5346e+00
            0            0            0   1.0000e+00
```

5 带位移的隐式 QR 迭代

5.1 上 Hessenberg 矩阵

5.2 隐式 QR 迭代法

5.3 位移的选取

5.4 收缩

直接实施 QR 方法的困难: 运算量太大

每一步迭代需要做一次 QR 分解和矩阵乘积, 运算量为 $\mathcal{O}(n^3)$.

即使每计算一个特征值只需迭代一步, 则总运算量为 $\mathcal{O}(n^4)$.



目标: 运算量从 $\mathcal{O}(n^4)$ 减小到 $\mathcal{O}(n^3)$.

实现方法: 两步走

第一步: 首先通过正交相似变换将 A 转化成一个 上 Hessenberg 矩阵

第二步: 对这个 Hessenberg 矩阵实施 隐式 QR 迭代

隐式 QR 迭代

在 QR 迭代法中, 我们并 不进行显式的 QR 分解和矩阵乘积, 而是通过 特殊手段 来实现从 A_k 到 A_{k+1} 的迭代, 并且将运算量控制在 $\mathcal{O}(n^2)$ 量级, 从而将总运算量降到 $\mathcal{O}(n^3)$.

5.1 上 Hessenberg 矩阵

上 Hessenberg 矩阵: $H = [h_{ij}] \in \mathbb{R}^{n\times n}$, 当 $i > j+1$ 时, 有 $h_{ij} = 0$

定理 设 $A \in \mathbb{R}^{n\times n}$, 则存在正交矩阵 $Q \in \mathbb{R}^{n\times n}$, 使得 QAQ^{T} 是上 Hessenberg 矩阵.

以 5×5 矩阵为例, 给出具体的转化过程, 采用的工具为 Householder 变换.

$$A = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \end{bmatrix} \quad \to \quad \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}$$

第一步： 令 $Q_1 = \mathrm{diag}(I_{1\times 1}, H_1)$, 其中 H_1 是对应于向量 $A(2:5,1)$ 的 Householder 矩阵. 于是可得

$$Q_1 A = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \end{bmatrix}.$$

由于用 Q_1^{T} 右乘 $Q_1 A$ 时, 不会改变 $Q_1 A$ 第一列元素的值, 故

$$A_1 \triangleq Q_1 A Q_1^{\mathsf{T}} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \end{bmatrix}.$$

第二步： 令 $Q_2 = \mathrm{diag}(I_{2\times 2}, H_2)$, 其中 H_2 是对应于向量 $A_1(3:5,2)$ 的 Householder 矩阵, 则用 Q_2 左乘 A_1 时, 不会改变 A_1 的第一列元素的值. 用 Q_2^{T} 右乘 Q_2A_1 时, 不会改变 Q_2A_1 前两列元素的值. 因此,

$$Q_2A_1 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & * & * & * \end{bmatrix} \quad \text{和} \quad A_2 \triangleq Q_2A_1Q_2^{\mathsf{T}} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & * & * & * \end{bmatrix}.$$

第三步: 令 $Q_3 = \mathrm{diag}(I_{3\times 3}, H_3)$, 其中 H_3 是对应于向量 $A_2(4:5,3)$ 的 Householder 矩阵, 则有

$$Q_3A_2 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix} \quad \text{和} \quad A_3 \triangleq Q_3A_2Q_3^{\mathsf{T}} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

这时, 我们就将 A 转化成一个上 Hessenberg 矩阵, 即 $QAQ^{\mathsf{T}} = A_3$ 其中 $Q = Q_3Q_2Q_1$ 是正交矩阵, A_3 是上 Hessenberg 矩阵.

上 Hessenberg 化算法

算法 7 上 Hessenberg 化 (Upper Hessenberg Reduction)

1: Set $Q = I$

2: **for** $k = 1$ to $n - 2$ **do**

3: compute Householder matrix H_k with respect to $A(k+1:n, k)$

4: $$\begin{aligned} A(k+1:n, k:n) &= H_k \cdot A(k+1:n, k:n) \\ &= A(k+1:n, k:n) - \beta_k v_k \left(v_k^{\mathsf{T}} A(k+1:n, k:n)\right) \end{aligned}$$

5: $$\begin{aligned} A(1:n, k+1:n) &= A(1:n, k+1:n) \cdot H_k^{\mathsf{T}} \\ &= A(1:n, k+1:n) - \beta_k A(1:n, k+1:n) v_k v_k^{\mathsf{T}} \end{aligned}$$

6: $$\begin{aligned} Q(k+1:n, k:n) &= H_k \cdot Q(k+1:n, k:n) \\ &= Q(k+1:n, k:n) - \beta_k v_k \left(v_k^{\mathsf{T}} Q(k+1:n, k:n)\right) \end{aligned}$$

7: **end for**

几点说明

- 在实际计算时, 我们不需要显式地形成 Householder 矩阵 H_k.
- 上述算法的运算量大约为 $\frac{14}{3}n^3 + \mathcal{O}(n^2)$.

  如不计算特征向量, 则 Q 不用计算, 运算量约为 $\frac{10}{3}n^3 + \mathcal{O}(n^2)$.

上 Hessenberg 矩阵性质（一）

定理 设 $A \in \mathbb{R}^{n\times n}$ 是非奇异上 Hessenberg 矩阵, 其 QR 分解为 $A = QR$, 则 $\tilde{A} \triangleq RQ$ 也是上 Hessenberg 矩阵, 即**上 Hessenberg 矩阵在 QR 迭代中保持形状不变**. (板书)

✍ **注：若 A 是奇异的, 也可以通过选取适当的 Q, 使得上述结论成立, 留作练习.**

上 Hessenberg 矩阵性质（二）

定理 设 $A \in \mathbb{R}^{n\times n}$ 是上 Hessenberg 矩阵且下次对角线元素均非零, 即 $a_{i+1,i} \neq 0, i = 1, 2, \ldots, n-1$. 设其 QR 分解为 $A = QR$, 则 $\tilde{A} \triangleq RQ$ 的下次对角线元素也都非零.

(留作练习)

✍ 注: 在实际计算时, 我们只需考虑**不可约**情形，因此可以假设 A 的下次对角线均非零.

5.2 隐式 QR 迭代法

在 QR 迭代中, 先做 QR 分解 $A_k = Q_k R_k$, 然后计算 $A_{k+1} = R_k Q_k$.

☞ 在不对 A_k 进行 QR 分解的前提下, 直接计算出 A_{k+1}. 这就是 隐式 QR 迭代

定理 (Implicit Q Theorem) 设 $H = Q^{\mathsf{T}} A Q \in \mathbb{R}^{n\times n}$ 是一个不可约上 Hessenberg 矩阵, 其中 $Q \in \mathbb{R}^{n\times n}$ 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). (板书)

- QR 迭代: $A_{k+1} = R_k Q_k = Q_k^{\mathsf{T}} A_k Q_k$
- 隐式 QR 迭代: $\tilde{A}_{k+1} = \tilde{Q}_k^{\mathsf{T}} A_k \tilde{Q}_k$, 其中 $\tilde{Q}_k$ 的第一列与 Q_k 第一列相等

由隐式 Q 定理可知 $\tilde{Q}_k = W Q_k$, 其中 $W = \mathrm{diag}(1, \pm 1, \ldots, \pm 1)$. 于是

$$\tilde{A}_{k+1} \tilde{Q}_k^{\mathsf{T}} A_k \tilde{Q}_k = W^{\mathsf{T}} Q_k^{\mathsf{T}} A_k Q_k W = W^{\mathsf{T}} A_{k+1} W.$$

即 $\tilde{A}_{k+1}$ 与 A_{k+1} 的对角线元素相等, 非对角线元素至多相差一个符号.

隐式 QR 迭代

在 QR 迭代法中, 如果我们直接令 $A_{k+1} = \tilde{Q}_k^{\mathsf{T}} A_k \tilde{Q}_k$, 则其收敛性与原 QR 迭代法没有任何区别!

隐式 QR 迭代的实现

举例说明如何利用隐式 Q 定理, 由 A_1 得到 A_2, 即实现隐式 QR 迭代.

设 $A \in \mathbb{R}^{5\times 5}$ 是一个不可约上 Hessenberg 矩阵, 即

$$A_1 = A = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

第一步： 构造一个 Givens 变换

$$G_1^{\mathsf{T}} \triangleq G(1,2,\theta_1) = \begin{bmatrix} c_1 & s_1 & & \\ -s_1 & c_1 & & \\ & & & I_3 \end{bmatrix} \quad (c_1, s_1 \text{ 待定})$$

于是有

$$G_1^{\mathsf{T}} A = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix} \quad \text{和} \quad A^{(1)} \triangleq G_1^{\mathsf{T}} A G_1 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ + & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

与 A_1 相比较, $A^{(1)}$ 在 $(3,1)$ 位置上多出一个非零元, 我们把它记为 " $+$ ", 并称之为 **bulge**. 在下面的计算过程中, 我们的目标就是将其 "赶" 出矩阵, 从而得到一个新的上 Hessenberg 矩阵, 即 A_2.

第二步： 为了消去这个 bulge, 我们可以构造 Givens 变换

$$G_2^{\mathsf{T}} \triangleq G(2,3,\theta_2) = \begin{bmatrix} 1 & & & & \\ & c_2 & s_2 & & \\ & -s_2 & c_2 & & \\ & & & & I_2 \end{bmatrix} \text{ 使得 } G_2^{\mathsf{T}} A^{(1)} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

为了保持与原矩阵的相似性, 需要再右乘 G_2, 所以

$$A^{(2)} \triangleq G_2^{\mathsf{T}} A^{(1)} G_2 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & + & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

此时, bugle 从 $(3,1)$ 位置被 “赶” 到 $(4,2)$ 位置.

第三步： 与第二步类似, 构造 Givens 变换

$$G_3^{\mathsf{T}} \triangleq G(3,4,\theta_3) = \begin{bmatrix} I_2 & & & \\ & c_3 & s_3 & \\ & -s_3 & c_3 & \\ & & & 1 \end{bmatrix} \text{ 使得 } G_3^{\mathsf{T}} A^{(2)} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

这时

$$A^{(3)} \triangleq G_3^{\mathsf{T}} A^{(2)} G_3 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & + & * & * \end{bmatrix}.$$

于是, bugle 又从 $(4,2)$ 位置又被 “赶” 到 $(5,3)$ 位置.

第四步： 再次构造 Givens 变换

$$G_4^{\mathsf{T}} \triangleq G(4,5,\theta_4) = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & c_4 & s_4 \\ & & & -s_4 & c_4 \end{bmatrix} \text{ 使得 } G_4^{\mathsf{T}} A^{(3)} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}$$

于是

$$A^{(4)} \triangleq G_4^{\mathsf{T}} A^{(3)} G_4 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}.$$

现在, bulge 被 “赶” 出矩阵, $A^{(4)}$ 就是我们所要的矩阵!

算法分析, 以及 c_1, s_1 的取值

- 根据前面的计算过程, 有

$$A^{(4)} = G_4^{\mathsf{T}} G_3^{\mathsf{T}} G_2^{\mathsf{T}} G_1^{\mathsf{T}} A_1 G_1 G_2 G_3 G_4 = \tilde{Q}_1^{\mathsf{T}} A_1 \tilde{Q}_1,$$

  其中 $\tilde{Q}_1 = G_1 G_2 G_3 G_4 \Longrightarrow A^{(4)} = \tilde{Q}_1^{\mathsf{T}} A_1 \tilde{Q}_1$

- 通过直接计算可知, $\tilde{Q}_1$ 的第一列为

$$[c_1, s_1, 0, 0, 0]^{\mathsf{T}}.$$

- 如果将其取为 A_1 的第一列 $[a_{11}, a_{21}, 0, \ldots, 0]^{\mathsf{T}}$ 单位化后的向量, 则 $\tilde{Q}_1$ 的第一列与 Q_1 的第一列相同!

带位移隐式 QR 迭代

针对带位移的隐式 QR 迭代法, 我们取 $A_1 - \sigma_1 I$ 的第一列

$$[a_{11} - \sigma_1, a_{21}, 0, \ldots, 0]^{\mathsf{T}}$$

单位化后的向量作为 G_1 的第一列即可.

运算量

如果 $A \in \mathbb{R}^{n\times n}$ 是上 Hessenberg 矩阵, 则使用上面的算法, 带位移隐式 QR 迭代中**每一步的运算量为** $6n^2 + \mathcal{O}(n)$.

5.3 位移的选取

基本准则: 位移越离某个特征值越近越好.

直观取法

如果位移 σ 与某个特征值非常接近, 则 $A_k(n,n)-\sigma$ 就非常接近于 0.

这说明 $A_k(n,n)$ 通常会首先收敛到 A 的一个特征值.

所以 $\sigma = A_k(n,n)$ 是一个不错的选择.

☹ 但是, 如果这个特征值是复数, 这种位移选取方法就可能失效.

双位移策略

设 $\sigma \in \mathbb{C}$ 是 A 的某个复特征值 λ 的一个很好的近似, 则 $\bar{\sigma}$ 也应该是 $\bar{\lambda}$ 的一个很好近似. 因此我们可以考虑**双位移**策略, 即先以 σ 为位移迭代一次, 然后再以 $\bar{\sigma}$ 为位移迭代一次, 不断交替, 这样就有

$$\begin{aligned} A_1 - \sigma I &= Q_1 R_1, \\ A_2 &= R_1 Q_1 + \sigma I, \\ A_2 - \bar{\sigma} I &= Q_2 R_2, \\ A_3 &= R_2 Q_2 + \bar{\sigma} I. \end{aligned}$$

容易验证

$$A_3 = Q_2^{\mathsf{T}} A_2 Q_2 = Q_2^* Q_1^* A_1 Q_1 Q_2 = Q^* A_1 Q,$$

其中 $Q = Q_1 Q_2$.

我们注意到 σ 可能是复的, 所以 Q_1 和 Q_2 都可能是复矩阵. 但我们却可以选取适当的 Q_1 和 Q_2, 使的 $Q = Q_1Q_2$ 是实正交矩阵.

引理 在双位移 QR 迭代中, 我们可以选取酉矩阵 Q_1 和 Q_2 使得 $Q = Q_1Q_2$ 是实矩阵.

证明概要. 由于 $Q_2R_2 = A_2 - \bar{\sigma}I = R_1Q_1 + (\sigma - \bar{\sigma})I$, 所以

$$\begin{aligned} Q_1Q_2R_2R_1 &= Q_1\big(R_1Q_1 + (\sigma - \bar{\sigma})I\big)R_1 \\ &= Q_1R_1Q_1R_1 + (\sigma - \bar{\sigma})Q_1R_1 \\ &= (A_1 - \sigma I)^2 + (\sigma - \bar{\sigma})(A_1 - \sigma I) \\ &= A_1^2 - (\sigma + \bar{\sigma})A_1 + \bar{\sigma}\sigma I, \\ &= A_1^2 - 2\mathrm{Re}(\sigma)A_1 + |\sigma|^2 I \in \mathbb{R}^{n\times n} \end{aligned}$$

又 Q_1Q_2 是酉矩阵, R_2R_1 是上三角矩阵, 故 $(Q_1Q_2)(R_2R_1)$ 是实矩阵 $A_1^2 - 2\mathrm{Re}(\sigma)A_1 + |\sigma|^2 I = (A_1 - \sigma I)(A_1 - \bar{\sigma}I)$ 的 QR 分解. 所以 Q_1Q_2 和 R_2R_1 都可以是实矩阵.

双位移策略的实现

由前面的结论可知, 存在 Q_1 和 Q_2, 使得 $Q = Q_1Q_2$ 是实矩阵, 从而

$$A_3 = Q^{\mathsf{T}} A_1 Q$$

也是实矩阵. 因此我们希望 不计算 A_2, 而是直接从 A_1 得到 A_3.

实现方法

根据隐式 Q 定理: 只要找到一个实正交矩阵 Q, 使得其第一列与

$$A_1^2 - 2\mathrm{Re}(\sigma)A_1 + |\sigma|^2 I$$

的第一列平行, 并且 $A_3 = Q^{\mathsf{T}} A_1 Q$ 是上 Hessenberg 矩阵即可.

易知, $A_1^2 - 2\mathrm{Re}(\sigma)A_1 + |\sigma|^2 I$ 的第一列为

$$\begin{bmatrix} a_{11}^2 + a_{12}a_{21} - 2\mathrm{Re}(\sigma)a_{11} + |\sigma|^2 \\ a_{21}(a_{11} + a_{22} - 2\mathrm{Re}(\sigma)) \\ a_{21}a_{32} \\ 0 \\ \vdots \end{bmatrix}$$

所以 Q 的第一列是上述向量的单位化.



其它过程可以通过隐式 QR 迭代来实现. 但此时的 "bulge" 是一个 2×2 小矩阵. 因此, 在双位移隐式 QR 迭代过程中, 需使用 Householder 变换.

需要指出的是, 双位移 QR 迭代法中的运算都是实数运算.

双位移隐式 QR 迭代举例

设 $A \in \mathbb{R}^{6\times 6}$ 是一个不可约上 Hessenberg 矩阵, 即

$$A_1 = A = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix}.$$

第一步： 构造一个正交矩阵 $H_1 = \begin{bmatrix} \tilde{H}_1^{\mathsf{T}} & 0 \\ 0 & I_3 \end{bmatrix}$, 其中 $\tilde{H}_1 \in \mathbb{R}^{3\times 3}$, 使得其第一列与 $A_1^2 - 2\mathrm{Re}(\sigma)A_1 + |\sigma|^2 I$ 的第一列平行. 于是有

$$H_1^{\mathsf{T}} A = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ + & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix} \text{ 和 } A^{(1)} \triangleq H_1^{\mathsf{T}} A H_1 = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ + & * & * & * & * & * \\ + & + & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix}.$$

与 A_1 相比较, $A^{(1)}$ 在 $(3,1)$, $(4,1)$ 和 $(4,2)$ 位置上出现 **bulge**. 在下面的计算过程中, 我们的目标就是把它们 "赶" 出矩阵, 从而得到一个新的上 Hessenberg 矩阵.

第二步： 令 $H_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \tilde{H}_2^{\mathsf{T}} & 0 \\ 0 & 0 & I_2 \end{bmatrix}$, 其中 $\tilde{H}_2 \in \mathbb{R}^{3\times 3}$ 是对应于 $A(2:4,1)$ 的 Householder 变换, 使得

$$H_2^{\mathsf{T}} A^{(1)} = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & + & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix} \text{ 和 } A^{(2)} \triangleq H_2^{\mathsf{T}} A^{(1)} H_2 = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & + & * & * & * & * \\ 0 & + & + & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix}$$

这时, 我们将 bugle 向右下角方向 “赶” 了一个位置.

第三步： 与第二步类似, 令 $H_3 = \begin{bmatrix} I_2 & 0 & 0 \\ 0 & \tilde{H}_3^{\mathsf{T}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$, 其中 $\tilde{H}_3 \in \mathbb{R}^{3\times 3}$ 是对应于 $A(3:5,2)$ 的 Householder 变换, 使得

$$H_3^{\mathsf{T}} A^{(2)} = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & + & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix} \text{ 和 } A^{(3)} \triangleq H_3^{\mathsf{T}} A^{(2)} H_3 = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & + & * & * & * \\ 0 & 0 & + & + & * & * \end{bmatrix}$$

此时, bugle 又被向右下角方向 “赶” 了一个位置.

第四步： 令 $H_4 = \begin{bmatrix} I_3 & 0 \\ 0 & \tilde{H}_4^{\mathsf{T}} \end{bmatrix}$, 其中 $\tilde{H}_4 \in \mathbb{R}^{3\times 3}$ 是对应于 $A(4:6,3)$ 的 Householder 变换, 使得

$$H_4^{\mathsf{T}} A^{(3)} = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & + & * & * \end{bmatrix} \text{ 和 } A^{(4)} \triangleq H_4^{\mathsf{T}} A^{(3)} H_4 = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & + & * & * \end{bmatrix}$$

第五步： 只需构造一个 Givens 变换 $G_5 = \begin{bmatrix} I_4 & 0 \\ 0 & G(4,5,\theta)^{\mathsf{T}} \end{bmatrix}$, 使得

$$G_5^{\mathsf{T}} A^{(4)} = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix} \text{ 和 } A^{(5)} \triangleq G_5^{\mathsf{T}} A^{(4)} G_5 = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix}$$

现在, bulge 已经被全部消除, 且

$$A^{(5)} = Q^{\mathsf{T}} A Q,$$

其中 $Q = H_1 H_2 H_3 H_4 G_5$. 通过直接计算可知, Q 的第一列即为 H_1 的第一列. 根据隐式 Q 定理, 可以直接令 $A_3 \triangleq A^{(5)} = Q^{\mathsf{T}} A Q$

位移的选取

取 A_k 的右下角矩阵

$$\begin{bmatrix} A_k(n-1,n-1) & A_k(n-1,n) \\ A_k(n,n-1) & A_k(n,n) \end{bmatrix}$$

的复共轭特征值作为双位移. 这样选取的位移就是 Francis 位移 .

- 若上述矩阵的两个特征值都是实的, 则选取其中模较小的特征值做单位移
- 采用 Francis 位移的 QR 迭代会使 A_k 的右下角收敛到一个上三角矩阵 (两个实特征值) 或 2 阶矩阵 (复共轭特征值), 而且通常会有二次收敛性. 实际计算中, 一个特征值一般平均只需迭代两步.

收敛性判断

看 $A_k(i+1,i)$ ($A_k(n-1,n-2)$ 或 $A_k(n,n-1)$) 是否趋向于 0.

需要指出的是, 采用 Francis 位移的 QR 迭代并不是对所有矩阵都收敛.

- 例如:

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

对于上面的矩阵, 采用 Francis 位移的 QR 迭代法无效.

多重位移策略

另外, 也可以考虑多重位移策略, 可参见 [Watkins '07].

5.4 收缩

收缩技术

- 在隐式 QR 迭代过程中, 当矩阵 A_k 的某个下次对角线元素 $a_{i+1,i}$ 很小时, 我们可以将其设为 0.
- 由于 A_{k+1} 是上 Hessenberg 矩阵, 这时 A_{k+1} 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.
- 因此我们可以将隐式 QR 迭代作用在这两个规模相对较小的矩阵上, 从而可以大大节约运算量.

6 特征向量的计算

- 解特征方程 $(\lambda I - A)x = 0$
- 带位移反迭代
- ……

7 广义特征值问题

7.1 广义特征值基本理论

7.2 广义 Schur 分解

7.3 QZ 迭代法

7.1 广义特征值基本理论

什么是广义特征值

设 $A, B \in \mathbb{R}^{n\times n}$, 若存在 $\lambda \in \mathbb{C}$ 和非零向量 $x \in \mathbb{C}^n$ 使得

$$Ax = \lambda Bx$$

则称 λ 为矩阵对 (A, B) 的特征值, x 为相应的特征向量.

计算矩阵对 (A, B) 的特征值和特征向量就是 广义特征值问题

当 B 非奇异时, 广义特征值问题就等价于标准特征值问题

$$B^{-1}Ax = \lambda x \quad \text{或} \quad AB^{-1}y = \lambda y, \quad \text{其中 } y = Bx$$

正则矩阵对

λ 是 (A,B) 的一个特征值当且仅当

$$\det(A-\lambda B)=0 \quad \rightarrow \quad \text{特征方程}$$

若特征方程对所有 $\lambda\in\mathbb{C}$ 都成立, 则称矩阵对 (A,B) 是 **奇异矩阵对**, 否则称为 **正则矩阵对**.

- 若 B 非奇异, 则特征方程是 n 次多项式, 因此恰好有 n 个特征值
- 若 B 奇异, 则特征方程的次数低于 n, 因此方程的解的个数小于 n.

  注意到 $\lambda\neq 0$ 是 (A,B) 的特征值当且仅当 $\mu=\lambda^{-1}$ 是 (B,A) 的特征值.

  因此, 若 B 奇异, 则 $\mu=0$ 是 (B,A) 的特征值. 把 $\lambda=\mu^{-1}=\infty$ 作为 (A,B) 的特征值.

  所以, 广义特征值不是分布在 $\mathbb{C}$ 上, 而是分布在 $\mathbb{C}\cup\{\infty\}$ 上.

7.2 广义 Schur 分解

若 U, V 非奇异, 则 (U^*AV, U^*BV) 与 (A, B) 具有相同的特征值. 因此这种变换称为**矩阵对的等价变换**. 如果 U, V 是酉矩阵, 则称为**酉等价变换**.

定理 (广义 Schur 分解) 设 $A, B \in \mathbb{C}^{n\times n}$, 则存在酉矩阵 $Q, Z \in \mathbb{C}^{n\times n}$, 使得

$$Q^*AZ = R_A, \quad Q^*BZ = R_B,$$

其中 $R_A, R_B \in \mathbb{C}^{n\times n}$ 都是上三角矩阵. 此时矩阵对 (A, B) 的特征值为

$$\lambda_i = \frac{R_A(i,i)}{R_B(i,i)}, \quad i = 1, 2, \ldots, n.$$

当 $R_B(i,i) = 0$ 时, 对应的特征值 $\lambda_i = \infty$.

广义实 Schur 分解

与实 Schur 分解类似, 当 A, B 都是实矩阵时, 我们有相应的**广义实 Schur 分解**.

定理 (广义实 Schur 分解) 设 $A, B \in \mathbb{R}^{n\times n}$, 则存在正交矩阵 $Q, Z \in \mathbb{R}^{n\times n}$, 使得

$$Q^{\mathsf{T}} A Z = T_A, \quad Q^{\mathsf{T}} B Z = T_B,$$

其中 $T_A, T_B \in \mathbb{R}^{n\times n}$ 都是拟上三角矩阵.

7.3 QZ 迭代法

- QZ 迭代法是用于计算 (A, B) 的广义 Schur 分解的算法, 是 QR 算法的自然推广, 本质上可以看作是将 QR 算法作用到矩阵 AB^{-1} 上, 但在具体实施时需要做一些优化, 以提高执行效率.
- 详细算法可参见 [Kressner '05, Xu-Qian '11].

8 应用：多项式求根

考虑 n 次多项式

$$q_n(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_1 x + c_0, \quad c_i \in \mathbb{R}.$$

- 由代数学基本定理可知, $p_n(x)$ 在复数域中有且仅有 n 的零点
- $n \geq 5$ 时, 不存在求根公式
- 非线性迭代方法求解
- MATLAB 中的 `roots` 命令: 通过特征值计算方法求出所有零点

友矩阵

$$A = \begin{bmatrix} 0 & & & -c_0 \\ 1 & 0 & & -c_1 \\ & \ddots & \ddots & \vdots \\ & & 1 & -c_{n-1} \end{bmatrix}$$

多项式 $q_n(x)$ 的零点 $\Longleftrightarrow$ A 的特征值

- 无需上 Hessenberg 化
- A 非常稀疏, 但经过一步 QR 迭代后, 上三角部分的零元素会消失, 总运算量仍是 $\mathcal{O}(n^3)$
- 快速 QR 方法 : 利用 A 的特殊结构, 运算量降为 $\mathcal{O}(n^2)$

  - 将 A 写成一个酉矩阵与秩一矩阵之差, 参见相关文献


谢 谢
THANK YOU