非对称特征值问题

July 22, 2019

非对称矩阵特征值/特征向量的计算
基本约定 1：$A \in \mathbb{R}^{n\times n}$、非对称、稠密
基本约定 2: $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n| \geq 0$

本讲主要讨论如何计算 A 的全部特征值和/或特征向量
主要介绍以下方法:

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

关于稠密矩阵特征值计算的参考资料有:

- J.H.Wilkinson,TheAlgebraicEigenvalueProblem,1965
- B.N.Parlett,TheSymmetricEigenvalueProblem,2ndEds.,1998
- G.W.Stewart,MatrixAlgorithms,VolII:Eigensystems,2001
- G.H.GolubandC.F.VanLoan,MatrixComputations,2013
- P.Arbenz,Thecourse252-0504-00G,

Numerical Methods for Solving Large Scale Eigenvalue Problems, 2018. (该课程的主页)

1 幂迭代

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算法的一个基本框架.
算法 1.1 幂迭代算法 (Power Iteration)

Choose an initial guess $x(0)$ with $||x(0)||_2 = 1$

set $k = 0$

while not convergence do

$\quad y^{(k+1)} = Ax^{(k)}$

$\quad x^{(k+1)} = y^{(k+1)} / \left\|y^{(k+1)}\right\|_2$

$\quad \mu_{k+1} = \left(x^{(k+1)}, Ax^{(k+1)}\right)$

$\quad k = k + 1$

end while

幂迭代的收敛性
假设 1:$A \in \mathbb{R}^{n\times n}$ 可对角化, 即 $A = V\Lambda V^{-1}$,其中

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

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

$$x^{(0)} = \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n = V\left[\alpha_1, \alpha_2, \ldots, \alpha_n\right]^\top$$

我们假定 $\alpha_1 \neq 0$,即 $x^{(0)}$ 不属于 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

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

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

又 $|\lambda_i/\lambda_1| < 1, i = 2, 3, \ldots, n$,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

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

收敛到 $e_1=[1,0,\ldots,0]^\top$
所以向量 $x^{(k)}=A^k x^{(0)}/\left\|A^k x^{(0)}\right\|_2$ 收敛到 $\pm v_1$, 即 λ_1 的特征向量. 而 $\mu_k=\left(x^{(k)}\right)^* A x^{(k)}$ 则收敛到 $v_1^* A v_1=\lambda_1$ † 幂迭代的收敛快慢取决于 $|\lambda_2/\lambda_1|$ 的大小,$|\lambda_2/\lambda_1|$ 越小,收敛越快.

- 幂迭代只能用于计算 (模) 最大的特征值和其相应的特征向量
- 当 $|\lambda_2/\lambda_1|$ 接近于 1 时, 收敛速度会非常慢
- 如果模最大的特征值是一对共轭复数, 则幂迭代可能会失效.

加速技巧:位移策略
出发点: 加快幂迭代算法的收敛速度 $\Longleftrightarrow$ 尽可能地减小 $|\lambda_2/\lambda_1|$
位移策略:计算 $A-\sigma I$ 的特征值
我们称 σ 为位移, 满足

(1) $\lambda_1-\sigma$ 是 $A-\sigma I$ 的模最大特征值

(2) $\max_{2\leq i\leq n}\left|\frac{\lambda_i-\sigma}{\lambda_1-\sigma}\right|$ 尽可能地小

其中第一个条件保证最后所求得的特征值是我们所要的, 第二个条件用于加快幂迭代的收敛速度.
缺点:(1)σ 很难选取;(2) 加速效果有限
改进: 与反迭代相结合, 能起到很好的加速效果

2 反迭代

用幂迭代求 A^{-1} 的模最小特征值,这就是反迭代
算法 2.1 反迭代算法 (Inverse Iteration)

1: Choose an initial guess $x(0)$ with $||x(0)||_2=1$

2: set $k=0$

3: while not convergence do

4: $\quad y^{(k+1)}=(A-\sigma I)^{-1}x^{(k)}$

5: $\quad x^{(k+1)}=y^{(k+1)}/\left\|y^{(k+1)}\right\|_2$

6: $\quad \mu_{k+1}=\left(x^{(k+1)},Ax^{(k+1)}\right)$

7: $\quad \sigma=\mu_{k+1}, k=k+1$

8: end while

显然:μ_k 收敛到 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:

- 若 σ 与某个特征值 λ_k 非常接近, 则反迭代算法的收敛速度非常快
- 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:

- 每步迭代需要解一个线性方程组 $(A-\sigma I)y^{(k+1)}=x^{(k)}$ 这需要对 $A-\sigma I$ 做 LU 或 PLU 分解
- 与幂迭代一样, 反迭代算法一次只能求一个特征值
- 怎样选取位移 σ? $\rightarrow$ Rayleigh 商动态选取, 自动调整

2.1 Rayleigh 商迭代

出发点:使得 σ 与所求的特征值越靠近越好.

期望能直接给出一个理想位移是不太现实的. 比较现实的方法就是动态调整, 使得位移逐渐靠近某个特征值.
Rayleigh 商迭代: 以 Rayleigh 商 μ_k 为第 k 步的位移
理由:μ_k 会逐渐收敛到某个特征值.

算法 2.2 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 $\sigma=\left(x^{(0)}\right)^* Ax^{(0)}$
4. while not convergence do
5. $\quad y^{(k+1)}=(A-\sigma I)^{-1}x^{(k)}$
6. $\quad x^{(k+1)}=y^{(k+1)}/\left\|y^{(k+1)}\right\|_2$
7. $\quad \mu_{k+1}=\left(x^{(k+1)},Ax^{(k+1)}\right)$
8. $\quad \sigma=\mu_{k+1}$
9. $\quad k=k+1$
10. end while

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.
因此通常应用于 三对角矩阵 的特征值计算

3 正交迭代

出发点:同时计算多个特征值/特征向量
策略: 同时采用多个初始向量, 希望收敛到 A 的一个不变子空间
算法 3.1 正交迭代算法 (Orthogonal Iteration)

1: Choose an initial vector$n \times p$column orthogonal matrix Z_0

2: set $k = 0$

3: while not convergence do

4: compute $Y^{(k+1)} = AZ_{(k)}$

5: $Y_{(k+1)} = Z_{(k+1)}\hat{R}_{k+1}$

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 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
假设 A 是可对角化的, 即 $A = V\Lambda V^{-1}$, 其中 $\Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$, 且 $|\lambda_1| \geq \cdots \geq |\lambda_p| > |\lambda_{p+1}| \geq \cdots \geq |\lambda_n|$. 则可得

$$\operatorname{span}\{Z_k\} = \operatorname{span}\{Y_k\} = \operatorname{span}\{AZ_{k-1}\}, \quad k = 1, 2, \ldots$$

由此可知

$$\operatorname{span}\{Z_k\} = \operatorname{span}\{A^k Z_0\} = \operatorname{span}\{V\Lambda^k V^{-1} Z_0\}$$

我们注意到

$$\Lambda^k V^{-1} Z_0 = \lambda_p^k \begin{bmatrix} (\lambda_1/\lambda_p)^k & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & (\lambda_n/\lambda_p)^k \end{bmatrix} V^{-1} Z_0 \triangleq \lambda_p^k \begin{bmatrix} W_p^{(k)} \\ W_{n-p}^{(k)} \end{bmatrix}$$

由于当 $i > p$ 时有 $|\lambda_i/\lambda_p| < 1$,所以当 k 趋于无穷大时,$W_{n-p}^{(k)}$ 趋向于 0,令 $V = [V_p, V_{n-p}]$,则

$$V\Lambda^k V^{-1} Z_0 = \lambda_p^k [V_p, V_{n-p}] \begin{bmatrix} W_p^{(k)} \\ W_{n-p}^{(k)} \end{bmatrix} = \lambda_p^k \left(V_p W_p^{(k)} + V_{n-p} W_{n-p}^{(k)} \right)$$

所以当 $k \to \infty$ 时,有

$$\begin{aligned} \text{span}\{Z_k\} = \text{span}\{V\Lambda^k V^{-1} Z_0\} &= \text{span}\left\{V_p W_p^{(k)} + V_{n-p} W_{n-p}^{(k)}\right\} \\ &\to \text{span}\left\{V_p W_p^{(k)}\right\} = \text{span}\{V_p\} \end{aligned}$$

即 $\text{span}\{Z_k\}$ 趋向于 A 的一个 p 维不变子空间 $\text{span}\{V_p\}$

定理 给定正整数 $p(1 \le p \le n)$, 考虑算法 3.1, 假设 A 是可对角化的, 且 $|\lambda_1| \ge \cdots \ge |\lambda_p| > |\lambda_{p+1}| \ge \cdots \ge |\lambda_n|$. 则 $\text{span}\{Z_k\}$ 收敛到 A 的一个 p 维不变子空间.
说明:
如果 A 不可对角化, 利用 Jordan 标准型, 可以到同样的结论, 见 [Watkins 2007, Watkins-Elsner 1991].
† 在正交迭代中, 如果我们取 $Z_0 = I$, 则可得到一类特殊的正交迭代算法. 此时, 在一定条件下, 正交迭代会收敛到 A 的 Schur 标准型.

4 QR 迭代

4.1 算法介绍

基本思想:通过不断的正交相似变换, 将 A 转化为 (拟) 上三角形式
算法 4.1 QR 迭代算法 (QR Iteration)

1: Set $A_1 = A$ and $k = 1$

2: while not convergence do

3: $\quad [Q_k, R_k] = qr(A_k)$

4: $\quad$ compute $A_{k+1} = R_k Q_k$

$k = k + 1$

end while

正交相似性在 QR 迭代算法中，我们有

$$A_{k+1} = R_k Q_k = \left(Q_k^\top Q_k\right) R_k Q_k = Q_k^\top \left(Q_k R_k\right) Q_k = Q_k^\top A_k Q_k$$

由这个递推关系可得

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

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

$$A_{k+1} = \tilde{Q}_k^\top A \tilde{Q}_k \tag{1}$$

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

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

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.

$\rightarrow$ 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，$A\tilde{q}_1^{(k)} \rightarrow \lambda_1 \tilde{q}_1^{(k)}$

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

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

结论

A_{k+1} 的第一列的第一个元素收敛到 λ_1，而其他元素都趋向于 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 $A_{k+1}=\tilde{Q}_k^\top A\tilde{Q}_k$ 可知

$$A\tilde{Q}_k=\tilde{Q}_kA_{k+1}=\tilde{Q}_kQ_{k+1}R_{k+1}=\tilde{Q}_{k+1}R_{k+1}$$

所以有

$$\tilde{Q}_{k+1}=A\tilde{Q}_kR_{k+1}^{-1}$$

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

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

观察等式两边矩阵的最后一列，可得

$$\tilde{q}_n^{(k+1)}=c_1\left(A^\top\right)^{-1}\tilde{q}_n^{(k)}(c_1\text{为某个常数})$$

以此类推，可知

$$\tilde{q}_n^{(k+1)}=c\left(A^\top\right)^{-k}\tilde{q}_n^{(1)}(c\text{为某个常数})$$

假定 $|\lambda_1|\geq\cdots\geq|\lambda_{n-1}|>|\lambda_n|>0$ 则 λ_n^{-1} 是 $\left(A^\top\right)^{-1}$ 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

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

所以

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

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

$$A_{k+1}^\top(:,n)=\tilde{Q}_k^\top A^\top\tilde{q}_n^{(k)}\to\lambda_n\tilde{Q}_k^\top\tilde{q}_n^{(k)}=\lambda_ne_n$$

□□

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 迭代算法与正交迭代算法 $(Z_0=I)$ 之间的关系.

定理 假定正交迭代算法 3.1 和 QR 算法 4.1 中所涉及的 QR 分解都是唯一的. A_k 是由 QR 迭代算法 4.1 生成的矩阵, Z_k 是由正交迭代算法 3.1 (取 $Z_0=I$) 生成的矩阵, 则有

$$A_{k+1}=Z_k^\top AZ_k$$

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 $\Lambda=\mathrm{diag}\left(\lambda_1,\lambda_2,\ldots,\lambda_n\right)$，且 $|\lambda_1|>|\lambda_2|>\cdots>|\lambda_n|$. 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收敛到 0

说明：
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 迭代算法演示 (见 $Eig_Q R.m$). 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 Ak 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

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

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

```
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

为了加快 QR 迭代的收敛速度, 可以采用位移策略 和反迭代的思想
算法 4.2 带位移的 QR 迭代算法 (QR Iteration with shift)

1: Set $A_1 = A$ and $k = 1$

2: while not convergence do

3: Choose a shift σ_k

4: $[Q_k, R_k] = qr(A_k - \sigma_k I)$

5: compute $A_{k+1} = R_k Q_k + \sigma_k I$

6: $k = k + 1$

7: end while

正交相似性

$$\begin{aligned} A_{k+1} = R_k Q_k + \sigma_k I &= \left(Q_k^\top Q_k\right) R_k Q_k + \sigma_k I \\ &= Q_k^\top \left(A_k - \sigma_k I\right) Q_k + \sigma_k I \\ &= Q_k^\top A_k Q_k \end{aligned}$$

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 迭代算法得到.
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 迭代算法有二次收敛速度.

例 带位移的 QR 迭代算法演示 ($\square Eig_Q R_s hift.m$).
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 $\sigma_k = A_k(n-1, n-1)$

5 带位移的隐式 QR 迭代

直接实施 QR 方法的困难: 运算量
每一步迭代需要做一次 QR 分解和矩阵乘积, 运算量为 $O\left(n^3\right)$ 即使每计算一个特征值只需迭代一步, 则总运算量为 $O\left(n^4\right)$
我们的目标: 从 $O\left(n^4\right)$ 减小到 $O\left(n^3\right)$

```
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
```

实现方法:两个步骤

(1) 首先通过相似变化将 A 转化成一个上 H essenberg 矩阵

(2) 对这个 Hessenberg 矩阵实施隐式 QR 迭代

隐式 QR 迭代:
在 QR 迭代算法中, 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 $O\left(n^2\right)$ 量级,从而将总运算量降到 $O(n^3)$

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 $QAQ^\top$ 是上 Hessenberg 矩阵
下面我们以一个 5×5 的矩阵 A 为例, 给出具体的转化过程, 采用的工具为 Householder 变换.
第一步:令 $Q_1=\operatorname{diag}\left(I_{1\times 1},H_1\right)$,其中 H_1 是对应于向量 $A(2:5,1)$ 的 Householder 矩阵. 于是可得

$$Q_1A=\begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \end{bmatrix}$$

由于用 Q_1^T 右乘 Q_1A, 不会改变 Q_1A 的第一列元素的值, 故

$$A_1\triangleq Q_1AQ_1^\top=\begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \\ 0 & * & * & * & * \end{bmatrix}$$

第二步:令 $Q_2=\operatorname{diag}\left(I_{2\times 2},H_2\right)$,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 $Q_2^\top$ 右乘 Q_2A_1 时,不会改变 Q_2A_1 前两列元素的值. 因此

$$Q_2A_1=\begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & * & * & * \end{bmatrix}\square A_2\triangleq Q_2A_1Q_2^\top=\begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & * & * & * \end{bmatrix}$$

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

$$Q_3A_2 = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix} \square A_3 \triangleq Q_3A_2Q_3^\top = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}$$

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