计算方法及MATLAB实现

郑勋烨　编著

主审：　高世臣　褚宝增　王祖朝　王翠香

副审：　李少琪　李明霞　赵琳琳　赵俊芳

国防工业出版社

第七章　矩阵特征值计算

7.3　矩阵特征值的 QR 算法

7.3.1 QR **正交三角分解算法**

QR(正交三角) 分解算法是由 西方学者 J.G.Francis 和苏联女学者 V.N.Kublanovskaya(忽必烈诺夫斯卡雅) 于 1961 年独立发现的，取代了 Rutishauser 于 1958 年基于矩阵的三角分解建立起来的 LR 算法，更加稳定.

1955 年人们还在为计算矩阵特征值大伤脑筋， 1965 年基于 QR 算法的程序已经相当成熟. QR 算法对于解决矩阵特征值计算问题具有里程碑意义，至今仍然是最有效和最实用的方法之一.

【**定义** 1.QR **正交三角分解基本算法** (Basic QR Method)】

注意到正交矩阵满足 $Q^TQ = I$，由于矩阵 A 具有正交三角 QR 分解

$$A = QR, \Leftrightarrow R = Q^T A$$

由此可以构造与矩阵 A正交相似 的新的矩阵 B 为

$$B = RQ = Q^T AQ \sim A$$

众所周知，一对相似矩阵 B 与 A 具有完全相同的 全部特征值 和 全部特征向量. 因此如果经过 正交相似变换 获得的新的矩阵 B 能够比较容易地求得 特征值 和 特征向量，则初始矩阵 A 的 特征值 和 特征向量 求解问题就可以简化.

对于新矩阵可以再度进行正交三角 QR 分解得到

$$B = RQ = Q^T A Q = Q' R'$$

同时我们获得一种类似“可交换”性质的等式

$$RQ = Q' R'$$

然而右方的正交三角矩阵 Q', R' 是全新的.

照葫芦画瓢，这样我们将得到一个以矩阵 $A_1 = A$ 为初始矩阵的序列：

$$
\begin{cases}
A_1 & = A & = Q_1 R_1 \\
A_2 & = R_1 Q_1 = Q_1^T A_1 Q_1 & = Q_2 R_2 \\
A_3 & = R_2 Q_2 = Q_2^T A_2 Q_2 & = Q_3 R_3 \\
A_4 & = R_3 Q_3 = Q_3^T A_3 Q_3 & = Q_4 R_4 \\
\dots\dots\dots & \dots\dots\dots & \dots\dots \\
A_k & = R_{k-1} Q_{k-1} = Q_{k-1}^T A_{k-1} Q_{k-1} & = Q_k R_k \\
A_{k+1} & = R_k Q_k = Q_k^T A_k Q_k & = Q_{k+1} R_{k+1} \\
\dots\dots\dots & \dots\dots\dots & \dots\dots
\end{cases}
$$

于是向初始矩阵迭代有

$$
\begin{aligned}
& A_{k+1} = R_k Q_k = Q_k^T A_k Q_k = Q_k^T Q_{k-1}^T A_{k-1} Q_{k-1} Q_k \\
& = \cdots\cdots = Q_k^T Q_{k-1}^T \cdots Q_2^T Q_1^T A_1 Q_1 Q_2 \cdots Q_{k-1} Q_k \\
& = (Q_1 Q_2 \cdots Q_{k-1} Q_k)^T A_1 Q_1 Q_2 \cdots Q_{k-1} Q_k \\
& = \tilde{Q}_k^T A_1 \tilde{Q}_k
\end{aligned}
$$

我们标记连乘积形式的正交矩阵 $\tilde{Q}_k$ 为正交矩阵序列顺序相乘，而上三角矩阵 $\tilde{R}_k$ 为上三角矩阵序列倒序相乘. 即

$$\tilde{Q}_k := Q_1 Q_2 \cdots Q_{k-1} Q_k, \qquad \tilde{R}_k := R_k R_{k-1} \cdots R_2 R_1$$

则显然有性质：

I. 矩阵序列 A_k 中的所有矩阵彼此 正交相似： $A_{k+1} = Q_k^T A_k Q_k$.

II. 矩阵序列 A_k 中的所有矩阵与初始矩阵 $A_1 = A$ 正交相似： $A_{k+1} = \tilde{Q}_k^T A_1 \tilde{Q}_k$.

此外，还有一条非常重要的“幂表达”的性质： $A^k = \tilde{Q}_k \tilde{R}_k$.

事实上，归纳假设 $A^{k-1} = \tilde{Q}_{k-1} \tilde{R}_{k-1}$. 则

$$
\begin{aligned}
\tilde{Q}_k \tilde{R}_k &= Q_1 Q_2 \cdots Q_{k-1} (Q_k R_k) R_{k-1} \cdots R_2 R_1 \\
&= Q_1 Q_2 \cdots Q_{k-1} A_k R_{k-1} \cdots R_2 R_1 \\
&= \tilde{Q}_{k-1} A_k \tilde{R}_{k-1} \\
&= \tilde{Q}_{k-1} \tilde{Q}_{k-1}^T A_1 \tilde{Q}_{k-1} \tilde{R}_{k-1} \\
&= A_1 \tilde{Q}_{k-1} \tilde{R}_{k-1} = A A^{k-1} = A^k
\end{aligned}
$$

再由矩阵 A_k 的正交三角 QR 分解程序,

$$A_k = Q_k R_k, \qquad R_k = Q_k^T A_k$$

标记正交矩阵 Q_k^T 为连乘积形式的 Givens 旋转正交矩阵序列或 Householder 初等反射正交矩阵序列倒序相乘. 即

$$Q_k^T := P_{n-1} \cdots P_2 P_1$$

则有

$$R_k = Q_k^T A_k = P_{n-1} \cdots P_2 P_1 A_k$$

并且

$$
\begin{aligned}
A_{k+1} &= R_k Q_k = Q_k^T A_k Q_k \\
&= P_{n-1} \cdots P_2 P_1 A_k (P_{n-1} \cdots P_2 P_1)^T \\
&= P_{n-1} \cdots P_2 P_1 A_k P_1^T P_2^T \cdots P_{n-1}^T
\end{aligned}
$$

从而矩阵 A_{k+1} 可以通过对 A_k 作两个连续的左右正交变换获得:

I. 前锋矩阵 A_k 左正交变换： $P_{n-1}\cdots P_2P_1A_k = R_k$.

II. 上三角矩阵 R_k 右正交变换：
$A_{k+1} = R_kP_1^TP_2^T\cdots P_{n-1}^T$.

上述算法称为 正交三角分解基本 QR 算法 (QR Method).

【 **定理** 1. **基本** QR **算法的基本收敛** (Essential Convergence of QR Method) 】

标记 $A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$

$$A_k = \begin{pmatrix} a_{11}^{(k)} & a_{12}^{(k)} & \cdots & a_{1n}^{(k)} \\ a_{21}^{(k)} & a_{22}^{(k)} & \cdots & a_{2n}^{(k)} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}^{(k)} & a_{n2}^{(k)} & \cdots & a_{nn}^{(k)} \end{pmatrix}$$

即有：

I. 矩阵序列 A_k 主对角元 收敛于特征值：

$$\lim_{k\to+\infty} a_{ii}^{(k)} = \lambda_i.$$

II. 矩阵序列 A_k 下对角元 收敛于零：

$$\lim_{k\to+\infty} a_{ij}^{(k)} = 0, \qquad i > j.$$

而当 $i < j$ 时，极限 $\lim\limits_{k\to+\infty} a_{ij}^{(k)}$ 未必存在．这就是所谓“基本收敛”．但当矩阵 $A = (a_{ij}) = (a_{ji})$ 是 对称矩阵 时，显然有：

III.　矩阵序列 A_k 上对角元 收敛于零：

$\lim\limits_{k\to+\infty} a_{ij}^{(k)} = 0, \qquad i < j.$

亦即矩阵 $A = (a_{ij}) = (a_{ji})$ 是 对称矩阵 时，矩阵 A_k 收敛于对角矩阵 D ：

$$\lim_{k\to+\infty} A_k = D = Diag[\lambda_1, \lambda_2, \cdots, \lambda_n]$$

【例 1. Givens 旋转变换基本 QR 算法】

对于如下对称矩阵

$$A = \begin{pmatrix} 8 & 2 \\ 2 & 5 \end{pmatrix}$$

存在特征值： $\lambda_1 = 9 > \lambda_2 = 4$. 满足基本 QR 算法的基本收敛条件. 试用系列 Givens 旋转变换矩阵 $P = P(i, j, \theta)$ 的乘积形式的正交矩阵 $Q_k^T = P_{n-1} \cdots P_2 P_1$ 构造基本 QR 算法矩阵序列 A_k . 并计算所用正交矩阵 Q_k^T 和 上三角矩阵 R_k .

解

由 Givens 旋转约化定理，对于非零向量 $x=(x_1,x_2,\cdots,x_i,\cdots,x_j,\cdots,x_n)^T, x_i^2+x_j^2\neq 0$, 存在平面旋转阵 $P=P(i,j,\theta)$ 使分量 x_j 化为 0. 即

$$\begin{cases} Px & = P(x_1,x_2,\cdots,x_i,\cdots,x_j,\cdots,x_n)^T \\ & = P(x_1,x_2,\cdots,x_i',\cdots,0,\cdots,x_n)^T \\ x_i' & = \sqrt{x_i^2+x_j^2} \\ \theta & = \arctan\dfrac{x_j}{x_i} \end{cases}$$

对于非零向量 $x = (8, 2)^T$ ，存在 2 阶 Givens 旋转变换阵 $P(1, 2, \theta_1)$ 使分量 $x_2 = 2$ 化为 0 且 $P(1, 2, \theta_1)x = \alpha_1 = \sigma e_1$.

下面来逐个计算各个参数、向量和矩阵：

$x_1' = \sqrt{x_1^2 + x_2^2} = \sqrt{8^2 + 2^2} = \sqrt{68}$.

取三角函数为

$$c = \cos\theta = \frac{x_i}{x_i'} = \frac{x_1}{\sqrt{x_1^2 + x_2^2}} = \frac{8}{\sqrt{68}}$$

$$s = \sin\theta = \frac{x_j}{x_i'} = \frac{x_2}{\sqrt{x_1^2 + x_2^2}} = \frac{2}{\sqrt{68}}$$

于是取旋转角度为 $\theta_1 = \arctan\dfrac{x_j}{x_i} = \arctan\dfrac{1}{4}$ 时，我们有在此平面旋转变换之下的向量为

$$\begin{aligned} P(1,2,\theta_1)x \quad &= P(1,2,\theta_1)(8,2)^T = (\sqrt{68}, 0)^T = \sigma e_1, \\ \sigma = \sqrt{68} > 0. & \end{aligned}$$

而相应的正交矩阵 Q^T 即 2 阶 Givens 旋转变换矩阵取为

$$Q_1^T = P(1,2,\theta_1) = \begin{pmatrix} \frac{8}{\sqrt{68}} & \frac{2}{\sqrt{68}} \\ -\frac{2}{\sqrt{68}} & \frac{8}{\sqrt{68}} \end{pmatrix}$$

亦即正交矩阵

$$Q_1 = P(1,2,\theta_1)^T = \begin{pmatrix} \dfrac{8}{\sqrt{68}} & -\dfrac{2}{\sqrt{68}} \\ \dfrac{2}{\sqrt{68}} & \dfrac{8}{\sqrt{68}} \end{pmatrix}$$

在此正交变换下对称矩阵 A 约化成为 上三角矩阵 R_1.

即

$$R_1 = Q_1^T A = \begin{pmatrix} \frac{8}{\sqrt{68}} & \frac{2}{\sqrt{68}} \\ -\frac{2}{\sqrt{68}} & \frac{8}{\sqrt{68}} \end{pmatrix} \begin{pmatrix} 8 & 2 \\ 2 & 5 \end{pmatrix}$$

$$= \frac{1}{\sqrt{68}} \begin{pmatrix} 68 & 26 \\ 0 & 36 \end{pmatrix}$$

从而我们有基本 QR 算法矩阵序列

$$A_2 = R_1 Q_1 = \frac{1}{\sqrt{68}} \begin{pmatrix} 68 & 26 \\ 0 & 36 \end{pmatrix} \begin{pmatrix} \frac{8}{\sqrt{68}} & -\frac{2}{\sqrt{68}} \\ \frac{2}{\sqrt{68}} & \frac{8}{\sqrt{68}} \end{pmatrix} =$$

$$\frac{1}{68} \begin{pmatrix} 596 & 72 \\ 72 & 288 \end{pmatrix}$$

$$\approx \begin{pmatrix} 8.7647 & 1.0588 \\ 1.0588 & 4.2353 \end{pmatrix}.$$

可见，矩阵 A_2 依然是 对称矩阵 ，而其 主对角元 更接近于真实特征值即 $\lambda_1 = 9, \lambda_2 = 4$. 上下对角元 则趋向于零：$\lim\limits_{k\to+\infty} a_{ij}^{(k)} = 0, \qquad i < j, i > j$. 总之，矩阵 A_2 更接近于 对角矩阵 $D = Diag[\lambda_1, \lambda_2]$. 基本 QR 算法仅仅迭代一步便初见成效. $\square$

7.3.2 QR 算法的原点位移加速

前述正交三角 QR 算法 (QR Method) 是计算矩阵 A 的 基本算法，收敛速度可能很慢. 类似于幂法和反幂法的加速思想，我们也希望对基本 QR 算法进行加速.

【原点位移 QR 算法的引入】

矩阵 $A=(a_{ij})_{n\times n}\in R^{n\times n}$ 的所有特征值依照模从大到小的次序排列为

$$|\lambda_1|>|\lambda_2|>\cdots>|\lambda_n|>0$$

由基本 QR 算法结论，矩阵序列 A_k 的 主对角元 序列收敛于特征值： $\lim\limits_{k\to+\infty} a_{ii}^{(k)}=\lambda_i$.

特别地，对于最小模的特征值有

$$\lim_{k \to +\infty} a_{nn}^{(k)} = \lambda_n$$

而“收敛速度”依赖于比值

$$r_n = \frac{|\lambda_n|}{|\lambda_{n-1}|}$$

引入参数 p, 则位移矩阵 $B = (A - pI)$ 的所有特征值为

$$\lambda_1 - p, \lambda_2 - p, \cdots, \lambda_n - p$$

现在假定 参数 p 是矩阵 A 的 “最小模” 特征值 λ_n 的 近似值，即满足 分离条件：

$$|\lambda_n - p| \ll |\lambda_i - p|, \qquad i \neq n$$

此时对于位移矩阵 $B = (A - pI)$ 使用 QR 算法，则矩阵序列 $A_k - pI$ 的处在 $(n, n-1)$ 位置的元素将以更快的速度 $r_n = \dfrac{|\lambda_n - p|}{|\lambda_{n-1} - p|}$ 线性收敛于零.

每次执行这样的程序：

I. 选定好每步的位移参数 p_k ，构造位移矩阵 $A_k - p_k I$.

II. 对于位移矩阵 $A_k - p_k I$ 作 正交三角 QR 分解：
$A_k - p_k I = Q_k R_k$.

III. 构造矩阵序列的后继矩阵 $A_{k+1} = R_k Q_k + p_k I$.

照葫芦画瓢，这样我们将得到一个以矩阵 $A_1 = A$ 为初始矩阵的序列：

$$\left\{\begin{array}{lc}
A_1 - p_1 I & = A - p_1 I = Q_1 R_1 \\
A_2 & = R_1 Q_1 + p_1 I = Q_1^T A_1 Q_1 = Q_2 R_2 \\
A_3 & = R_2 Q_2 + p_2 I = Q_2^T A_2 Q_2 = Q_3 R_3 \\
A_4 & = R_3 Q_3 + p_3 I = Q_3^T A_3 Q_3 = Q_4 R_4 \\
\cdots\cdots\cdots & \cdots\cdots\cdots \\
A_k & = R_{k-1} Q_{k-1} + p_k I = Q_{k-1}^T A_{k-1} Q_{k-1} = Q_k R_k \\
A_{k+1} & = R_k Q_k + p_k I = Q_k^T A_k Q_k = Q_{k+1} R_{k+1} \\
\cdots\cdots\cdots & \cdots\cdots\cdots
\end{array}\right.$$

简写为

$$\begin{cases} A_k - p_k I & = Q_k R_k \\ A_{k+1} & = R_k Q_k + p_k I \end{cases}$$

于是有

$$\begin{aligned} & A_{k+1} = R_k Q_k + p_k I = Q_k^T A_k Q_k = Q_k^T Q_{k-1}^T A_{k-1} Q_{k-1} Q_k \\ & = \cdots\cdots = Q_k^T Q_{k-1}^T \cdots Q_2^T Q_1^T A_1 Q_1 Q_2 \cdots Q_{k-1} Q_k \\ & = (Q_1 Q_2 \cdots Q_{k-1} Q_k)^T A_1 Q_1 Q_2 \cdots Q_{k-1} Q_k \\ & = \tilde{Q}_k^T A_1 \tilde{Q}_k \end{aligned}$$

我们标记连乘积形式的正交矩阵 $\tilde{Q}_k$ 为正交矩阵序列顺序相乘，而上三角矩阵 $\tilde{R}_k$ 为上三角矩阵序列倒序相乘. 即

$$\tilde{Q}_k := Q_1 Q_2 \cdots Q_{k-1} Q_k, \qquad \tilde{R}_k := R_k R_{k-1} \cdots R_2 R_1$$

则显然有：

$$\prod_{k=1}^{n} (A - p_k I) = \tilde{Q}_k \tilde{R}_k$$

【例 1. Givens 旋转变换 Rayleigh 商原点位移 QR 算法】

对于如下对称矩阵

$$A = \begin{pmatrix} 8 & 2 \\ 2 & 5 \end{pmatrix}$$

存在特征值： $\lambda_1 = 9 > \lambda_2 = 4$. 满足基本 QR 算法的基本收敛条件. 适当选择位移参数 p_k ，构造原点位移 QR 算法矩阵序列 A_k . 并计算所用正交矩阵 Q_k^T 和上三角矩阵 R_k .

解

我们执行如下“三步走”方案.

I. 选择 位移参数： $p_k = a_{nn}^{(k)}$. 构造位移矩阵 $A_k - p_k I$.

II. 对于位移矩阵 $A_k - p_k I$ 作 正交三角 QR 分解：
$A_k - p_k I = Q_k R_k$.

III. 构造矩阵序列的后继矩阵 $A_{k+1} = R_k Q_k + p_k I$.

I. 选择 位移参数： $p_1 = a_{nn}^{(1)} = 5$. 构造位移矩阵 $A_1 - p_1 I$.

$$A_1 - p_1 I = \begin{pmatrix} 8-5 & 2 \\ 2 & 5-5 \end{pmatrix} = \begin{pmatrix} 3 & 2 \\ 2 & 0 \end{pmatrix}$$

II. 对于位移矩阵 $A_k - p_k I$ 作 正交三角 QR 分解：
$A_k - p_k I = Q_k R_k$.

由 Givens 旋转约化定理，对于非零向量
$x = (x_1, x_2, \cdots, x_i, \cdots, x_j, \cdots, x_n)^T, x_i^2 + x_j^2 \neq 0$, 存在平面旋转阵 $P = P(i, j, \theta)$ 使分量 x_j 化为 0. 即

$$\begin{cases} Px & = P(x_1, x_2, \cdots, x_i, \cdots, x_j, \cdots, x_n)^T \\ & = P(x_1, x_2, \cdots, x_i', \cdots, 0, \cdots, x_n)^T \\ x_i' & = \sqrt{x_i^2 + x_j^2} \\ \theta & = \arctan \dfrac{x_j}{x_i} \end{cases}$$

对于非零向量 $x=(3,2)^T$ ，存在 2 阶 Givens 旋转变换阵 $P(1,2,\theta_1)$ 使分量 $x_2=2$ 化为 0 且 $P(1,2,\theta_1)x=\alpha_1=\sigma e_1$.

下面来逐个计算各个参数、向量和矩阵：

$x_1'=\sqrt{x_1^2+x_2^2}=\sqrt{3^2+2^2}=\sqrt{13}$.

取三角函数为

$$c = \cos\theta = \frac{x_i}{x_i'} = \frac{x_1}{\sqrt{x_1^2 + x_2^2}} = \frac{3}{\sqrt{13}}$$

$$s = \sin\theta = \frac{x_j}{x_i'} = \frac{x_2}{\sqrt{x_1^2 + x_2^2}} = \frac{2}{\sqrt{13}}$$

于是取旋转角度为 $\theta_1 = \arctan\dfrac{x_j}{x_i} = \arctan\dfrac{2}{3}$ 时，我们有在此平面旋转变换之下的向量为

$$\begin{aligned} P(1,2,\theta_1)x \quad &= P(1,2,\theta_1)(3,2)^T = (\sqrt{13},0)^T = \sigma e_1, \\ \sigma = \sqrt{13} > 0. \end{aligned}$$

而相应的正交矩阵 Q^T 即 2 阶 Givens 旋转变换矩阵取为

$$Q_1^T = P(1,2,\theta_1) = \begin{pmatrix} \dfrac{3}{\sqrt{13}} & \dfrac{2}{\sqrt{13}} \\ -\dfrac{2}{\sqrt{13}} & \dfrac{3}{\sqrt{13}} \end{pmatrix}$$

亦即正交矩阵

$$Q_1 = P(1,2,\theta_1)^T = \begin{pmatrix} \dfrac{3}{\sqrt{13}} & -\dfrac{2}{\sqrt{13}} \\ \dfrac{2}{\sqrt{13}} & \dfrac{3}{\sqrt{13}} \end{pmatrix}$$

在此正交变换下位移矩阵 $A_1 - p_1 I$ 约化成为上三角矩阵 R_1. 即

$$R_1 = Q_1^T(A_1 - p_1 I) = \begin{pmatrix} \frac{3}{\sqrt{13}} & \frac{2}{\sqrt{13}} \\ -\frac{2}{\sqrt{13}} & \frac{3}{\sqrt{13}} \end{pmatrix} \begin{pmatrix} 3 & 2 \\ 2 & 0 \end{pmatrix}$$

$$= \frac{1}{\sqrt{13}} \begin{pmatrix} 13 & 6 \\ 0 & -4 \end{pmatrix}$$

III.　构造矩阵序列的后继矩阵 $A_{k+1} = R_k Q_k + p_k I$.

$$A_2 = R_1 Q_1 + p_1 I =$$

$$\frac{1}{\sqrt{13}} \begin{pmatrix} 13 & 6 \\ 0 & -4 \end{pmatrix} \begin{pmatrix} \frac{3}{\sqrt{13}} & -\frac{2}{\sqrt{13}} \\ \frac{2}{\sqrt{13}} & \frac{3}{\sqrt{13}} \end{pmatrix} + \begin{pmatrix} 5 & 0 \\ 0 & 5 \end{pmatrix}$$

$$= \frac{1}{13}\begin{pmatrix} 51 & -8 \\ -8 & -12 \end{pmatrix} + \begin{pmatrix} 5 & 0 \\ 0 & 5 \end{pmatrix} =$$

$$\frac{1}{13}\begin{pmatrix} 51 & -8 \\ -8 & -12 \end{pmatrix} + \frac{1}{13}\begin{pmatrix} 65 & 0 \\ 0 & 65 \end{pmatrix}$$

$$= \frac{1}{13} \begin{pmatrix} 116 & -8 \\ -8 & 53 \end{pmatrix} \approx$$

$$\begin{pmatrix} 8.923076923 & -0.61538461538 \\ -0.61538461538 & 4.076923076923 \end{pmatrix}.$$

可见，矩阵 A_2 依然是对称矩阵，而其主对角元更接近于真实特征值即 $\lambda_1 = 9, \lambda_2 = 4$. 上下对角元则趋向于零: $\lim\limits_{k\to+\infty} a_{ij}^{(k)} = 0, \qquad i < j, i > j$. 总之，矩阵 A_2 更接近于对角矩阵 $D = Diag[\lambda_1, \lambda_2]$. 原点位移 QR 算法仅仅迭代一步便初见成效.

下一步选取 位移参数： $p_2 = a_{nn}^{(2)} = 4.076923076923$. 构造位移矩阵 $A_2 - p_2 I$. 继续迭代计算，我们有原点位移 QR 算法矩阵序列

$$A_3 \approx \begin{pmatrix} 8.999981 & -0.009766 \\ -0.009766 & 4.000019 \end{pmatrix}.$$

$$A_4 \approx \begin{pmatrix} 9.000000 & -3.7 \times 10^{-7} \\ -3.7 \times 10^{-7} & 4.000000 \end{pmatrix}.$$

仅仅迭代几步便几乎已经完美地逼近真实特征值即
$\lambda_1 = 9, \lambda_2 = 4$ ！ $\square$

【注记. 原点位移 QR 算法与基本 QR 算法的收敛速度比较】

矩阵 $A=(a_{ij})_{n\times n}\in R^{n\times n}$ 的所有特征值依照模从大到小的次序排列为

$$|\lambda_1|>|\lambda_2|>\cdots>|\lambda_n|>0$$

基本 QR 算法的“收敛速度”依赖于比值

$$r_n=\frac{|\lambda_n|}{|\lambda_{n-1}|}=\frac{4}{9}$$

引入参数 $p=5$, 则位移矩阵 $B=(A-pI)$ 的所有特征值为

$$\lambda_1-p=9-5=4, \lambda_2-p=4-5=-1$$

原点位移 QR 算法将以更快的速度

$$r_n=\frac{|\lambda_n-p|}{|\lambda_{n-1}-p|}=\frac{1}{4}<\frac{4}{9}$$

收敛.