

第8章 特征值和特征向量

MATLAB中的命令计算特征值和特征向量很方便，可以得到不同的子结果和分解，这在线性代数教学时很有用。注意，本章中的命令只能对二维矩阵操作。

8.1 特征值和特征向量的计算

假设 A 是一个 $m \times n$ 的矩阵， A 的特征值问题就是找到方程组的解：

$$Ax = \lambda x$$

其中 λ 是一个标量， x 是一个长度为 n 的列向量。标量 λ 是 A 的特征值， x 是相对应的特征向量。对于实数矩阵 A 来说，特征值和特征向量可能是复数。一个 $n \times n$ 的矩阵有 n 个特征值，表示为 $\lambda_1, \lambda_2, \dots, \lambda_n$ 。

MATLAB中用命令eig来确定矩阵 A 的特征值和特征向量。特征向量的规格化，就是每个特征向量的欧几里得范数为1；参见7.6节。

命令eig自动完成对矩阵 A 的平衡化。这就要求MATLAB找出一个相似变换矩阵 Q ，满足条件 \tilde{A} 。求 $\tilde{A} = Q^{-1}AQ$ 的特征值比求 A 的特征值条件更好些。万一 A 有一个和机器误差大小一样的元素，平衡化对于计算过程是没有好处的。带有参数nobalance的命令eig可用来计算没有这个变换矩阵的特征值和特征向量。

命令集79 特征值和特征向量

eig(A)	求包含矩阵 A 的特征值的向量。
[X,D]=eig(A)	产生一个矩阵 A 的特征值在对角线上的对角矩阵 D 和矩阵 X ，它们的列是相应的特征向量，满足 $AX=XD$ 。为了得到有更好条件特征值的矩阵要进行相似变换。
[X,D]= eig(A,'nobalance')	不经过平衡处理求得矩阵 A 的特征值和特征向量，也就是不进行平衡相似变换。
balance(A)	求平衡矩阵。
[T,B]=balance(A)	找到一个相似变换矩阵 T 和矩阵 B ，使得它们满足 $B=T^{-1}AT$ 。 B 是用命令balance求得的平衡矩阵。
eigs(A)	返回一个由矩阵 A 的部分特征值组成的向量，和命令eig一样，但是不返回全部的特征值。如果不带有参量，则计算出最大的特征值。当计算所有特征值时，如果矩阵 A 的秩不小于6，则计算出6个特征值来。
eigs(f,n)	求出矩阵 A 的部分特征值。在使用一个矩阵列的线性运算符时，字符串 f 中包含的是 M 文件的文件名， n 指定问题的阶次。用这种方法来求特征值比开始就用运算符来求要快。

`eigs(A,B,k,sigma)` 求矩阵A的部分特征值，矩阵B的大小和A相同；如果没有给出B=eye(size(A))，那么k就是要计算的特征值的个数；如果k没有给出，就用小于6的数或者A的秩；变量sigma是一个实数或者复数的移位参数，或者下列文本字符串中的一个，文本字符串指明需要的是哪种特征值：

- ‘lm’ 最大的特征值(缺省)
- ‘sm’ 最小的特征值
- ‘lr’ 最大的实数部分
- ‘sr’ 最小的实数部分
- ‘be’ 同时求得最大和最小的实数部分

`condeig(A)` 返回一个由矩阵A的特征值条件数组成的向量。

`[V,D,s]=condeig(A)` 返回[V,D]=eig(A)和s=condeig(A)。

如果A是实数矩阵，MATLAB在计算中用QR因式分解；否则用QZ因式分解。

左特征向量是满足下面条件的非零行向量y：

$$yA = y$$

如果用命令eig对A作用，也可以计算出左特征向量，因为：

$$A'y' = \bar{\lambda}y'$$

这里的撇号'代表矩阵的转置和共轭复数(见3.4节)，λ上的短杠表示共轭复数。矩阵特征值的集合称为矩阵的谱，谱半径ρ(A)定义为max(abs(eig(A)))。矩阵A的特征值的乘积等于det(A)，和等于trace(A)，这是矩阵A主对角线上元素的和。

如果X一个列向量为A的特征向量的矩阵，并且它的秩为n，那么特征向量线性无关。如果不是这样，则称矩阵为缺陷阵。如果X'X=I，则特征向量正交，这对于对称矩阵是成立的。

例8.1

矩阵A定义为：

$$A = \begin{pmatrix} -9 & -3 & -16 \\ 13 & 7 & 16 \\ 3 & 3 & 10 \end{pmatrix}$$

(a) 运行命令[Evect, Evalue]=eig(A)，得到结果为：

```
Evect =
-0.7071    -0.5774    -0.5774
 0.7071     0.5774    -0.5774
 0.0000     0.5774     0.5774

Evalue =
-6.0000         0         0
 0    10.0000         0
 0         0     4.0000
```

可知特征值都是非零数，矩阵是满秩的，可以用 therank=rank(Evect) 来确认：

```
therank=
```

令 $M = \text{Evec}' * \text{Evec}$ 得：

$$M = \begin{pmatrix} 1.0000 & 0.8165 & -0.0000 \\ 0.8165 & 1.0000 & 0.3333 \\ -0.0000 & 0.3333 & 1.0000 \end{pmatrix}$$

可知特征向量没有相互正交。

```
(b) determinant = prod(diag(Evalue)), ...
    determinant2 = det(A)
```

给出结果为：

```
determinant =
-240.0000
```

```
determinant2 =
-240
```

可知行列式等于特征值的积。

```
(c) theTrace = trace(A), theTrace2 = sum(diag(Evalue))
```

结果为：

```
theTrace =
8
```

```
theTrace2 =
8.0000
```

可知矩阵的迹等于特征值的和。

如果矩阵 A 是实数矩阵，但是有复数特征值，那么这些特征值是以共轭复数的形式出现的。

如果 $[X, D] = \text{eig}(A)$ ，可以用命令 `cdf2rdf` 将矩阵 D 转换为一个实数块对角矩阵。在对角线上用一个 2×2 实数块代替共轭复数对。

命令集80 复对角矩阵变成实对角矩阵

`[Y, E] = cdf2rdf(X, D)` 将复对角矩阵 D 变成实对角矩阵 E ， Y 的列不是 A 的特征向量。

例8.2

假设矩阵 A 为：

$$A = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

则运行 `[X, D] = eig(A)` 可得：

$$X = \begin{pmatrix} 0.7071 & 0.7071 & 0 \\ 0 + 0.7071i & 0 - 0.7071i & 0 \\ 0 & 0 & 1.0000 \end{pmatrix}$$

$$D = \begin{pmatrix} 0 + 1.0000i & 0 & 0 \\ 0 & 0 - 1.0000i & 0 \\ 0 & 0 & 3.0000 \end{pmatrix}$$

X和**D**都是复数矩阵，运行命令：`[Y,E]=cdf2rdf(X,D)`，结果为：

```
Y =
    0.7071         0         0
         0    0.7071         0
         0         0    1.0000
```

```
E =
     0     1     0
    -1     0     0
     0     0     3
```

所得的矩阵**E**正好是**A**矩阵。

注意 特征向量是特征多项式 $\det(\mathbf{I} - \mathbf{A})=0$ 的根，其中**I**是单位矩阵。用命令`poly`来求特征多项式，参见11.1节。命令`roots`也可求得特征值，但是用命令`eig`求得的特征值更准确，精度更高。

广义特征值问题就是找到方程组 $\mathbf{Ax} = \mathbf{Bx}$ 的重要解，其中**B**也是一个 $n \times n$ 的矩阵。值和向量**x**分别称为广义特征值和广义特征向量。

如果**B**是一个奇异矩阵，则用**QZ**算法来求解。

标准和广义特征值问题都属于矩阵多项式特征问题，都可以用命令`polyeig`来求它们的解。

命令集81 广义特征值和广义特征向量

<code>eig(A,B)</code>	返回一个含有广义特征值的向量， A 和 B 都是方阵。
<code>[X,D]=eig(A,B)</code>	返回一个对角线上为广义特征值的对角矩阵 D 和矩阵 X ， X 的列是相对应的特征向量，因此有 $\mathbf{AX}=\mathbf{BXD}$ 。
<code>[X,v]=</code> <code>polyeig(A0,A1,...,</code> <code>AK)</code>	给出度为 k 的特征问题 $(\mathbf{A}_0 + \mathbf{A}_1\lambda + \mathbf{A}_2\lambda^2 + \dots + \mathbf{A}_k\lambda^k)\mathbf{x} = \mathbf{0}$ 的特征值和特征向量。向量 v 的长度为 nk ，包含有特征值； $n \times nk$ 的矩阵 X 的列是特征向量。如果有 $\mathbf{A}_0 = \mathbf{A}$ 和 $\mathbf{A}_1 = -\mathbf{I}$ ，那么这就是标准特征值问题。

为了检查特征值的条件或者它的敏感性，可以计算出条件数 $\text{cond}(\mathbf{X}) = \|\mathbf{X}\| \|\mathbf{X}^{-1}\|$ ，矩阵**X**的列是**A**的特征向量。条件数大表示坏条件，也就是对扰动很敏感。

为了检查特征向量的条件或者它的敏感性可以查看特征值，多个重复的特征值或者特征彼此相差很小就表示是坏条件问题。

例8.3

假设：

$$\mathbf{A} = \begin{pmatrix} 3.75 & -0.5 & -0.375 & 0.495 & -1.37 \\ 0.25 & 2.5 & 0.375 & -0.495 & -0.63 \\ 1.25 & -0.5 & 2.875 & 0.495 & -2.12 \\ 0.25 & -0.5 & -0.625 & 2.505 & 0.37 \\ 0.25 & -0.5 & -0.625 & 0.495 & 2.38 \end{pmatrix}$$

运行命令`[XX, DD]=eig(A)`，结果为：

```

XX =
-0.0000    0.0000 - 0.4472i    0.0000 + 0.4472i   -0.4472    0.4472
 0.5000   -0.0000 + 0.4472i   -0.0000 - 0.4472i   -0.4472   -0.4472
 0.5000    0.0000 - 0.4472i    0.0000 + 0.4472i   -0.4472    0.4472
-0.5000    0.0000 - 0.4472i    0.0000 + 0.4472i   -0.4472   -0.4472
-0.5000    0.0000 - 0.4472i    0.0000 + 0.4472i   -0.4472    0.4472

```

显然两个特征向量，列2和列3是复数。

```

DD =
4.0000         0         0         0         0
      0   3.0000 + 0.0000i         0         0         0
      0         0   3.0000 - 0.0000i         0         0
      0         0         0         2.0000         0
      0         0         0         0         2.0100

```

从上可以看出特征值为2, 2.01, 3, 3和4, 可以知道这个特征值问题是一个坏条件问题。

输入 `badMatrix=cond(XX)` 可求得条件数：

```

badMatrix =
5.0156e+07

```

将这个数和例8.2中矩阵的特征值条件数 `niceMatrix=cond(X)` 比较：

```

niceMatrix =
1.0000

```

它们是不同的。

8.2 上海森伯形式、QR和QZ因式分解

如果只求特征值和特征向量，推荐用上一节中提到的方法。然而，有时要求更详细了解计算过程，可用在这一节和下一节中定义的命令来满足这样的要求。

如果矩阵 **H** 的第一子对角线下元素都是零，则它是一个上海森伯(Hessenberg)矩阵。如果矩阵是对称矩阵，则它的海森伯形式是对角三角阵。MATLAB可以通过相似变换将矩阵变换成这种形式。

命令集82 上海森伯形式

```

hess(A)          返回矩阵A的上海森伯形式。
[P,H]=hess(A)    返回一个酉矩阵P和上海森伯矩阵H，使A=P H P和PP'=I。

```

在MATLAB中，QR算法是计算矩阵所有特征值的一种有效的数学方法，也可以用命令 `eig` 来求。在用这种方法时，建议将矩阵转换成相似的上海森伯形式，参见例 8.4。

QR算法是基于QR因式分解的一种算法，每个 $m \times n$ 的矩阵A可以表示成：

```
A=QR
```

其中Q是一个 $m \times m$ 的酉矩阵，R是一个 $m \times n$ 的上三角矩阵。如果A是一个方阵，R也还是这样的矩阵。当用命令 `qr` 时，会返回矩阵Q和R，也可参见例7.7。

命令集83 QR因式分解

```
[Q,R]=qr(A)          产生一个m×m的酉矩阵Q和一个m×n的上三角矩阵
```

```
[Q,R,P]=qr(A)
```

R，使得 $A=QR$ 。

产生一个大小为 $m \times m$ 、列正交的西矩阵 **Q**，一个对角线元素递减的 $m \times n$ 的上三角矩阵 **R** 和一个置换矩阵 **P**，使得 $AP=QR$ 。

```
[Q,R]=qrinsert  
(Q,R,j,b)
```

由于在矩阵 **A** 的 j 列后插入一个额外的列 b 而得到新的 **QR** 因式分解，**Q** 和 **R** 是对矩阵 **A** 进行 **QR** 因式分解得到的矩阵。如果 $j=n+1$ ，那么 b 就插入在矩阵 **A** 的最后一列。

```
[Q,R]=  
qrdelete(Q,R,j)
```

由于去掉矩阵 **A** 的第 j 列而得到新的 **QR** 分解，**Q** 和 **R** 是对矩阵 **A** 进行 **QR** 分解得到的矩阵。

```
[Q1,R1]=  
qrupdate(Q,R,x,y)
```

给出 $A+xy'$ 的 **QR** 分解，也就是用秩为 1 的矩阵改变 **A** 的 **QR** 分解。

如果 **A** 是上海森伯矩阵，则 **Q** 也是一样。对于 **QR** 算法，下面给出一些简短的描述：

QR 算法：

- 1) 令 $A_0=A$ ， $k=0$ ；
- 2) 找到 A_k 的分解： $A_k=Q_k R_k$ ；
- 3) 迭代计算下一个矩阵： $A_{k+1}=Q_k R_k$ ，令 $k=k+1$ ；
- 4) 返回到 2。

这种方法也称为不移位的 **QR** 方法，就是在某种约定下逼近于上三角矩阵。因为所有的矩阵 A_k 和 $A_0=A$ 相似，所以有和原始矩阵相同的特征值，即最后的上三角矩阵的对角线元素就是 **A** 的特征值。

如果矩阵一开始就转换成有接近一半元素是零的上海森伯形式，就可以减少可观的计算步骤。**QR** 方法作为 MATLAB 的一个内建函数，为了加快逼近速度也可进行移位。

例 8.4

用不移位的 **QR** 因式分解算法，计算例 8.1 中矩阵 **A** 的特征值

$$A = \begin{pmatrix} -9 & -3 & -16 \\ 13 & 7 & 16 \\ 3 & 3 & 10 \end{pmatrix}$$

正确的特征值为 $\lambda_1=10$ ， $\lambda_2=4$ 和 $\lambda_3=-6$ 。

第 1 步：

```
A0 = hess(A); [Q0,R0] = qr(A0); A1 = R0*Q0
```

返回得到：

```
A1 =  
    1.7992    26.8770   -12.6126  
    2.3625     4.5085    -0.1434  
         0     4.9518     1.6923
```

第 2 步： $[Q1,R1]=qr(A1)$ ； $A2=R1*Q1$ ，得到：

```
A2 =
    17.6077    11.3432    5.0128
   -15.3516   -13.6557   -5.8721
         0     1.0748    4.0480
```

在计算开始时，看不出要逼近什么样的矩阵。但是在计算了 10 步以后，就可看出主对角线以下的元素较小。

```
[Q9,R9] = qr(A9); A10 = R9*Q9
```

结果为：

```
A10 =
    10.1297    22.6238    15.3505
    -0.0924    -6.1616    -5.8036
         0     0.0562    4.0319
```

注意，在整个计算过程中一直保留着上海森伯形式。

上例中的迭代过程可用 MATLAB 的内建编程语言简明地写出。在 12.2 节中有相关的例子。

QZ 算法是用来计算复数矩阵的复数特征向量对和广义特征值的。在 MATLAB 中，依据下面的命令集 84 来调用命令 qz。

命令集 84 QZ 算法

```
[C,D,Q,Z,V]= 得到对角线元素是广义特征值的上三角矩阵C、D和广义特征向量矩阵V。
qz(A,B)       矩阵Q和Z是变换矩阵，使得 QAZ=C和QBZ=D。
```

QZ 方法是基于 QZ 分解的方法。

8.3 舒尔分解和奇异值分解

如果 A 是一个方阵，则有一个这样的酉矩阵 U ，使得：

$$U^{-1}AU = U'AU = T$$

其中 T 是上三角矩阵。这是一个相似变换，因此矩阵 A 和 T 有相同的特征值。因为 T 是一个对角矩阵，因此其对角线元素就是它的特征值。

如果 A 是实数矩阵且对称，那么 T 有对角形式， U 的列就是 A 的特征向量。

如果 A 是实数矩阵，但是有复数特征值，那么 T 就是一个复数矩阵。为了避免复杂的计算，用 2×2 的实数矩阵来代表每一对共轭复数特征值，参见例 8.2。这样 T 就是一个块三角实数矩阵。MATLAB 中用命令 `schur` 来进行实数和复数矩阵 A 的舒尔分解。

如果 A 是实数矩阵，那么 `schur(A)` 返回实数的舒尔形式，但是如果 A 是复数矩阵，则给出复数形式。它们的差别在于实数形式中在对角线上用 2×2 的矩阵块来表示共轭复数特征值对，而复数形式给出的对角线元素是复数。函数 `rsf2csf` 可以将实数形式转换成复数形式。

命令集 85 舒尔分解

```
schur(A)          给出矩阵A的舒尔分解，也就是如上所述的矩阵T。
[U,T]=schur(A)    给出矩阵A的舒尔分解和一个酉矩阵U，使得A=UTU'。
[V,S]=rsf2csf(U,T) 将实数舒尔形式矩阵U和T转变成复数舒尔形式矩阵V和S。
```

例8.5

将A1, A2, A3定义为：

$$A1 = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \quad A2 = \begin{pmatrix} 2 & 1 \\ 0 & 2 \end{pmatrix} \quad A3 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

运行的命令为：

```
Sch1 = schur(A1), Sch2 = schur(A2)}, ...
[U,Sch3] = schur(A3)
```

结果为：

```
Sch1 =
     1     0
     0     3
```

```
Sch2 =
     2     1
     0     2
```

```
U =
     1     0
     0     1
```

```
Sch3 =
     0     1
    -1     0
```

因为有复数特征值，所以可以看出矩阵 Sch3不是一个上三角矩阵。为了验证一下可以输入：[V,S]=rsf2csf(U,Sch3)，得到：

```
V =
      0 + 0.7071i    0.7071
    -0.7071         0 - 0.7071i
```

```
S =
      0 + 1.0000i    0
      0             0 - 1.0000i
```

这里的特征值是*i*和 - *i*。

MATLAB还可以计算奇异值分解，即SVD和矩阵的奇异值。这些数都是非负数，在某种特定情况下，它们和矩阵的特征值相同。命令svds和命令eigs相似，也返回一些奇异值。

命令集86 SVD分解

svd(A)	返回一个包含矩阵A奇异值的向量。
[U,S,V]=svd(A)	返回一个对角矩阵S和大小分别为 $m \times m$ 和 $n \times n$ 的酉矩阵U和V，矩阵S的大小和A相同，也是 $m \times n$ 的，而且奇异值在矩阵的对角线上。奇异值是非负数且按降序


```
[U,S,V]=svd(A,0)
```

```
svds(A,k,0)
```

```
gsvd(A)
```

排列。这些矩阵满足 $A=USV$ 和 $U^T AV=S$ 。

得到一个“有效大小”的分解，只计算出矩阵 U 的前 n 列。矩阵 S 的大小为 $n \times n$ 。

计算出 k 个最大的奇异值和相应矩阵 A 的向量。如果 k 没有给出，则缺省值为 5。如果 0 作为最后一个参量值，则计算最小值；否则计算最大值。

给出广义奇异分解，参见 `gsvd` 的帮助可得更多相关信息。

在 MATLAB 中，矩阵 A 的伪逆可以用命令 `pinv(A)` 来求，也可用 SVD 分解来求矩阵的伪逆，参见 7.1 节。

如果 s_i 是奇异值，那么 $\|A\|_2 = \max s_i = s_1$ ， $\|A^{-1}\|_2 = (\min s_i)^{-1} = s_n^{-1}$ 和 $\text{cond}(A) = s_1/s_n$ ，其中 s_n 是最小奇异值。对于非奇异矩阵和满秩的长方形矩阵 ($m > n$)，最后一个表达式都成立。

例8.6

假设矩阵 A, B ：

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{pmatrix} \quad B = \begin{pmatrix} 1 & -1 & 0 \\ 1 & 2 & 0 \\ 1 & 3 & 0 \end{pmatrix}$$

(a) 求奇异值分解：

```
[Ua,Sa,Va] = svd(A), [Ub,Sb,Vb] = svd(B)
```

$U_a =$

```
0.3231    -0.8538    0.4082
0.5475    -0.1832   -0.8165
0.7719     0.4873    0.4082
```

$S_a =$

```
4.0791         0
         0    0.6005
         0         0
```

$V_a =$

```
0.4027   -0.9153
0.9153    0.4027
```

$U_b =$

```
-0.1641    0.9668    0.1961
0.5653    0.2551   -0.7845
0.8084    0.0178    0.5883
```

$S_b =$

```
3.9116         0         0
         0    1.3036         0
         0         0         0
```

```
Vb =  
    0.3092    0.9510         0  
    0.9510   -0.3092         0  
         0         0    1.0000
```

可以看出矩阵A和B有两个非零的奇异值，也就是它们的秩为2。

(b) 这些矩阵的逆是没有办法求的，但是伪逆可以用 `PseudoA=pinv(A)` 和 `PseudoB=pinv(B)` 来求：

```
PseudoA =  
    1.3333    0.3333   -0.6667  
   -0.5000    0.0000    0.5000
```

```
PseudoB =  
    0.6923    0.2308    0.0769  
   -0.2692    0.0769    0.1923  
         0         0         0
```

(c) 矩阵A和B的范数和条件数为：

```
normA = norm(A), normB = norm(B), ...  
condA = cond(A), condB = cond(B)
```

```
normA =  
    4.0791
```

```
normB =  
    3.9116
```

```
condA =  
    6.7930
```

```
condB =  
    Inf
```

可以看出欧几里得范数等于最大奇异值，条件数等于 s_1/s_{n_0} 。