第3章 矩阵处理

MATLAB是由早期专门用于矩阵运算的计算软件发展而来,矩阵是MATLAB最基本、最重要的数据对象,MATLAB的大部分运算或命令都是在矩阵运算的意义下执行的,而且这种运算定义在复数域上。正因为如此,使得MATLAB的矩阵运算功能非常丰富,许多含有矩阵运算的复杂计算问题,在MATLAB中很容易得到解决。因为向量可以看成是仅有一行或一列的矩阵,单个数据(标量)可以看成是仅含一个元素的矩阵,故向量和单个数据都可以作为矩阵的特例来处理。本章介绍创建特殊矩阵的方法、矩阵结构变换和矩阵求值的方法、特征值与特征向量的求解方法以及稀疏矩阵的概念与操作。

3.1 特殊矩阵的生成

在数值计算中,经常需要用到一些特殊形式的矩阵,如零矩阵、幺矩阵、单位矩阵等,这些特殊矩阵在应用中具有通用性。还有一类特殊矩阵在某些特定领域中得到应用,如范德蒙矩阵、希尔伯特矩阵、帕斯卡矩阵等。大部分特殊矩阵可以利用第2章介绍的建立矩阵的方法来实现,但MATLAB中提供了一些函数,利用这些函数可以更方便地生成特殊矩阵。

3.1.1 通用的特殊矩阵

1. 零矩阵/幺矩阵

为了提高运行效率,在编写脚本时,通常先创建零矩阵或幺矩阵,然后再修改指定元素的值。MATLAB提供了如下函数,用于创建零矩阵/幺矩阵。

(1) zeros函数:生成全0矩阵,即零矩阵。

(2) ones函数:生成全1矩阵,即幺矩阵。

(3) eye函数:生成单位矩阵,即对角线上的元素为1、其余元素为0的矩阵。

(4) true函数:生成全1逻辑矩阵。

(5) false函数:生成全0逻辑矩阵。

这几个函数的调用格式相似,下面以生成零矩阵的zeros函数为例进行说明。zeros函数的调用格式如下。

- zeros(m):生成m×m零矩阵,m省略时,生成一个值为0的标量。
- zeros(m,n):生成m×n零矩阵。
- zeros(m,TypeName):生成m×m零矩阵,矩阵元素为TypeName指定类型,可以为以下字符串之一:'double'、'single'、'int8'、'uint8'、'int16'、'uint16'、'int32'、'uint32'、'int64'或'uint64',默认为double型。
- zeros(m, 'like', p):生成m×m零矩阵,矩阵元素的类型与变量p一致。

【例3-1】 分别建立3×3、2×3的零矩阵。

(1) 建立一个3×3的零矩阵。

```
>> S = zeros(3)
S =
     0     0     0
     0     0     0
     0     0     0
```

(2) 建立一个2×3的零矩阵。

```
>> T1 = zeros(2,3);
```

建立一个2×3的零矩阵也可以采用以下命令:

```
>> T1 = zeros([2,3]);
```

如果需指定矩阵中的元素为整型，则使用命令：

```
>> T2 = zeros(2,3,'int16');
```

如果需指定矩阵中的元素为复型，则可以使用命令：

```
>> x = 1 + 2i;
>> T3 = zeros(2,3,'like',x)
T3 =
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
```

2. 随机矩阵

在 MATLAB 中，提供了 4 个生成随机矩阵的函数。

(1) rand(m，n)函数：生成一组值在(0，1)区间均匀分布的随机数，构建 m×n 矩阵。

(2) randi(imax，m，n)函数：生成一组值在[1，imax]区间均匀分布的随机整数，构建 m×n 矩阵。

(3) randn(m，n)函数：生成一组均值为 0、方差为 1 的标准正态分布随机数，构建 m×n 矩阵。

(4) randperm(n，k)函数：将[1，n]区间的整数随机排列，构建一个向量，参数 k 指定向量的长度。

MATLAB 默认生成的是伪随机数列，然后从伪随机数列中依次取出多个数，按函数的输入参数指定的方式构建矩阵。因此，在不同程序中调用同一个生成随机矩阵的函数，得到的随机矩阵是相同的。若需要生成不同的随机数列，则在调用以上函数之前调用 rng 函数。rng 函数的调用格式为：

```
rng(seed)
```

其中，参数 seed 作为生成随机数的种子，可取 0(默认值)或正整数，取'shuffle'时指定用当前时间作为生成随机数的种子。

假设已经得到了一组在(0，1)区间均匀分布的随机数 x，若想得到一组在(a，b)区间内均匀分布的随机数 y，可以用 $y_i=a+(b-a)x_i$ 计算得到。假设已经得到了一组标准正态分布随机数 x，如果想得到一组均值为 μ、方差为 σ^2 的随机数 y，可用 $y_i=\mu+\sigma x_i$ 计算得到。

【例 3-2】 建立随机矩阵。

(1) 在区间(10，30)内均匀分布的 4 阶随机矩阵。

(2) 均值为 0.6、方差为 0.1 的 4 阶正态分布随机矩阵。

命令如下：

```
>> a = 10;
>> b = 30;
>> x = a + (b - a) * rand(4)
x =
   15.0856   16.9997   19.4658   20.9945
   26.2857   13.9319   17.0332   28.3439
   14.8705   15.0217   26.6166   15.7168
   28.5853   22.3209   21.7053   25.1440
>> y = 0.6 + sqrt(0.1) * randn(4)
y =
    0.8641    0.2370    0.3891    0.4612
    0.5229    0.6332    0.6592    0.0325
    0.6682    0.8284    0.5739    0.8658
    0.2313    1.4176   -0.0113    0.3192
```

3.1.2 面向特定应用的特殊矩阵

MATLAB提供了若干能生成其元素值具有一定规律的特殊矩阵的函数,这类特殊矩阵在特定领域中是很有用的。下面介绍几个常用函数的功能和用法。

1. 魔方矩阵

魔方矩阵又称幻方、九宫图、纵横图,将自然数1,2,3,…,n^2,排列成n行n列的方阵,使每行、每列及两条主对角线上的n个数的和都等于$n(n^2+1)/2$。

在MATLAB中,函数magic(n)生成一个n阶魔方矩阵(n≥3)。

【例3-3】 将101～125等25个数填入一个5行5列的表格中,使其每行每列及对角线的和均为565。

一个5阶魔方矩阵的每行、每列及对角线的和均为65,对其每个元素都加100后这些和变为565。完成其功能的命令如下:

```
>> M = 100 + magic(5)
M =
   117   124   101   108   115
   123   105   107   114   116
   104   106   113   120   122
   110   112   119   121   103
   111   118   125   102   109
```

2. 范得蒙矩阵

范得蒙(Vandermonde)矩阵的最后一列全为1,倒数第二列为一个指定的向量,其他各列是其后列向量与倒数第二列向量的点乘积。

在MATLAB中,函数vander(V)生成一个以向量V为基础向量的范得蒙矩阵。例如:

```
>> A = vander([5, - 2,1,6])
A =
   125    25     5    1
   - 8     4   - 2    1
     1     1     1    1
   216    36     6    1
```

3. 希尔伯特矩阵

希尔伯特(Hilbert)矩阵是一种数学变换矩阵,它的每个元素$h_{ij}=1/(i+j-1)$。

在MATLAB中,函数hilb(n)生成一个n阶希尔伯特矩阵。希尔伯特矩阵是一个高度病态的矩阵,即任何一个元素发生微小变动,整个矩阵的行列式的值和逆矩阵都会发生巨大变化,病态程度和阶数相关。MATLAB提供了一个专门求希尔伯特矩阵的逆矩阵的函数invhilb(n),当n<15时,invhilb(n)生成希尔伯特矩阵的精确逆矩阵;当n≥15时,invhilb(n)生成希尔伯特矩阵的近似逆矩阵。

【例3-4】 求4阶希尔伯特矩阵及其逆矩阵。

命令如下:

```
>> format rat        % 以有理形式输出
>> H = hilb(4)
H =
       1             1/2           1/3           1/4
       1/2           1/3           1/4           1/5
       1/3           1/4           1/5           1/6
       1/4           1/5           1/6           1/7
>> H = invhilb(4)
```

```
H =
      16        -120         240        -140
    -120        1200       -2700        1680
     240       -2700        6480       -4200
    -140        1680       -4200        2800
>> format          %恢复默认输出格式
```

4. 托普利兹矩阵

托普利兹(Toeplitz)矩阵除第一行和第一列元素外,其他元素都与该元素左上角的元素相同。

在 MATLAB 中,函数 toeplitz(x,y)生成一个以 x 为第一列、y 为第一行的托普利兹矩阵。这里 x、y 均为向量,x、y 长度可以不同,但 x 和 y 的第 1 个元素必须相同。只有一个输入参数的函数 toeplitz(x)用向量 x 生成一个对称的托普利兹矩阵。例如:

```
>> T = toeplitz([5,6,7],[5,18,16,12])
T =
     5    18    16    12
     6     5    18    16
     7     6     5    18
```

5. 伴随矩阵

设多项式 $p(x)$为:

$$p(x)=a_n x^n+a_{n-1}x^{n-1}+\cdots+a_1 x+a_0$$

称矩阵

$$\boldsymbol{A}=\begin{bmatrix} -\dfrac{a_{n-1}}{a_n} & -\dfrac{a_{n-2}}{a_n} & -\dfrac{a_{n-3}}{a_n} & \cdots & -\dfrac{a_1}{a_n} & -\dfrac{a_0}{a_n} \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix}$$

为多项式 $p(x)$的伴随矩阵,$p(x)$称为 $\boldsymbol{A}$ 的特征多项式,方程 $p(x)=0$ 的根称为 $\boldsymbol{A}$ 的特征值。

MATLAB 生成伴随矩阵的函数是 compan(p),其中 p 是一个多项式的系数向量,高次幂系数排在前,低次幂排在后。例如,为了求多项式 $3x^4+10x^3-7x^2+1$ 的伴随矩阵,可使用命令:

```
>> format rat
>> p = [3,10,-7,0,1];
>> compan(p)
ans =
    -10/3     7/3     0     -1/3
      1       0       0       0
      0       1       0       0
      0       0       1       0
>> format
```

6. 帕斯卡矩阵

二次项$(x+y)^n$ 展开后的系数随 n 的增大组成一个三角形表,称为杨辉三角形。由杨辉三角形表组成的矩阵称为帕斯卡(Pascal)矩阵。帕斯卡矩阵的第 1 行元素和第 1 列元素都为 1,其余元素是同行的前一列元素和上一行的同列元素相加而得,即 $p_{1j}=1$,$p_{i1}=1$,$p_{ij}=p_{i,j-1}+p_{i-1,j}(i>1,j>1)$。

在 MATLAB 中,函数 pascal(n)生成一个 n 阶帕斯卡矩阵。

【例 3-5】 求$(x+y)^4$的展开式。

命令如下:

```
>> T4 = pascal(5)
T4 =
     1     1     1     1     1
     1     2     3     4     5
     1     3     6    10    15
     1     4    10    20    35
     1     5    15    35    70
```

矩阵次对角线上的元素 1,4,6,4,1 即为展开式的系数,即

$$(x+y)^4 = x^4 + 4x^3y + 6x^2y^2 + 4xy^3 + y^4$$

3.2 矩阵变换

矩阵变换是指对一个矩阵进行某种运算与处理,其结果还是一个矩阵,包括求矩阵的对角阵、三角阵、转置矩阵、矩阵旋转、矩阵求逆等。

3.2.1 对角矩阵与三角矩阵

1. 对角矩阵

只有对角线上有非 0 元素的矩阵称为对角矩阵,对角线上的元素相等的对角矩阵称为数量矩阵,对角线上的元素都为 1 的对角矩阵称为单位矩阵。矩阵的对角线有许多性质,如转置运算时对角线元素不变,相似变换时对角线的和不变等。在研究矩阵时,很多时候需要将矩阵的对角线上的元素提取出来形成一个列向量,而有时又需要用一个向量构造一个对角阵。

(1) 提取矩阵的对角线元素。设 A 为 m×n 矩阵,函数 diag(A)用于提取矩阵 A 主对角线元素,生成一个具有 min(m,n)个元素的列向量。例如:

```
>> A = [1,2,3;11,12,13;110,120,130]
A =
     1     2     3
    11    12    13
   110   120   130
>> d = diag(A)
d =
     1
    12
   130
```

函数 diag(A)还有一种形式 diag(A,k),其功能是提取第 k 条对角线的元素。主对角线为第 0 条对角线;与主对角线平行,往上为第 1 条、第 2 条……第 n 条对角线,往下为第−1 条、第−2 条……第−n 条对角线。例如,对于上面建立的 A 矩阵,提取其主对角线两侧对角线的元素,命令如下:

```
>> d1 = diag(A,1)
d1 =
     2
    13
>> d2 = diag(A, - 1)
d2 =
    11
   120
```

(2) 构造对角矩阵。设 V 为具有 m 个元素的向量，diag(V,k)的功能是生成一个 n×n (n=m+|k|)对角阵，其第 k 条对角线的元素即为向量 V 的元素。例如：

```
>> diag(10:2:14, -1)
ans =
     0     0     0     0
    10     0     0     0
     0    12     0     0
     0     0    14     0
```

默认 k 为 0，其主对角线元素即为向量 V 的元素。例如：

```
>> diag(10:2:14)
ans =
    10     0     0
     0    12     0
     0     0    14
```

【例 3-6】 先建立 5×5 矩阵 A，然后将 A 的第 1 行元素乘以 1，第 2 行乘以 2……第 5 行乘以 5。

用一个对角矩阵左乘一个矩阵时，相当于用对角阵的第 1 个元素乘以该矩阵的第 1 行，用对角阵的第 2 个元素乘以该矩阵的第 2 行，以此类推。因此，只需按要求构造一个对角矩阵 D，并用 D 左乘 A 即可。命令如下：

```
>> A=[17,0,1,0,15;23,5,7,14,16;4,0,13,0,22;10,12,19,21,3;…
      11,18,25,2,19];
>> D=diag(1:5);
>> D*A                    %用D左乘A,对A的每行乘以一个指定常数
ans =
    17     0     1     0    15
    46    10    14    28    32
    12     0    39     0    66
    40    48    76    84    12
    55    90   125    10    95
```

如果要对 A 的每列元素乘以同一个数，可以用一个对角阵右乘矩阵 A。

2. 三角矩阵

三角矩阵又进一步分为上三角矩阵和下三角矩阵。所谓上三角矩阵，即矩阵的对角线以下的元素全为 0 的一种矩阵，而下三角矩阵则是对角线以上的元素全为 0 的一种矩阵。

与矩阵 A 对应的上三角矩阵 B 是与 A 具有相同的行数和列数的一个矩阵，并且 B 的对角线以上(含对角线)的元素和 A 对应相等，而对角线以下的元素等于 0。求矩阵 A 的上三角矩阵的 MATLAB 函数是 triu(A)。例如，提取矩阵 A 的上三角元素，形成新的矩阵 B，命令如下：

```
>> A=randi(99,5,5)
A =
    76    70    82    44    49
    74     4    69    38    45
    39    28    32    76    64
    65     5    95    79    71
    17    10     4    19    75
>> B=triu(A)
B =
    76    70    82    44    49
     0     4    69    38    45
     0     0    32    76    64
     0     0     0    79    71
     0     0     0     0    75
```

triu 函数也有另一种形式 triu(A,k),其功能是求矩阵 A 的第 k 条对角线以上的元素。例如,提取上述矩阵 A 的第 2 条对角线以上的元素,形成新的矩阵 B1,命令如下:

```
>> B1 = triu(A,2)
B1 =
     0     0    82    44    49
     0     0     0    38    45
     0     0     0     0    64
     0     0     0     0     0
     0     0     0     0     0
```

在 MATLAB 中,提取矩阵 A 的下三角矩阵的函数是 tril(A)和 tril(A,k),其用法与 triu(A)和 triu(A, k)函数相同。

3.2.2 矩阵的转置与旋转

1. 矩阵的转置

所谓转置,即把源矩阵的第 1 行变成目标矩阵第 1 列,第 2 行变成第 2 列,以此类推。显然,一个 m 行 n 列的矩阵经过转置运算后,形成一个 n 行 m 列的矩阵。

在 MATLAB 中,转置运算使用运算符“.'”,或调用转置运算函数 transpose。例如:

```
>> A = randi(9,2,3)
A =
     3     6     2
     7     2     5
>> B = A.'   % 或 B = transpose(A)
B =
     3     7
     6     2
     2     5
```

复共轭转置运算是针对含复数元素的矩阵,除了将矩阵转置,还对原矩阵中的复数元素的虚部符号求反。复共轭转置运算使用运算符“'”,或调用函数 ctranspose。例如:

```
>> A = [3,4-1i,2+2i; 7i,1+1i,6-1i]
A =
   3.0000 + 0.0000i   4.0000 - 1.0000i   2.0000 + 2.0000i
   0.0000 + 7.0000i   1.0000 + 1.0000i   6.0000 - 1.0000i
>> B1 = A.'   % 或 B1 = transpose(A)
B1 =
   3.0000 + 0.0000i   0.0000 + 7.0000i
   4.0000 - 1.0000i   1.0000 + 1.0000i
   2.0000 + 2.0000i   6.0000 - 1.0000i
>> B2 = A'           % 或 B2 = ctranspose(A)
B2 =
   3.0000 + 0.0000i   0.0000 - 7.0000i
   4.0000 + 1.0000i   1.0000 - 1.0000i
   2.0000 - 2.0000i   6.0000 + 1.0000i
```

矩阵 B2 中的元素与矩阵 A 的对应元素虚部符号相反。

2. 矩阵的旋转与翻转

(1) 矩阵的旋转。在 MATLAB 中,利用函数 rot90(A, k)可以很方便地以 90°为单位对矩阵 A 进行旋转,选项 k 指定旋转 k 倍 90°,默认 k 为 1。当 k 为正整数时,将矩阵 A 按逆时针方向进行旋转;当 k 为负整数时,将矩阵 A 按顺时针方向进行旋转。例如:

```
>> A = rand(3,2)
A =
```

```
    0.5060    0.9593
    0.6991    0.5472
    0.8909    0.1386
>> B = rot90(A, -1)        % 按顺时针方向旋转 90°
B =
    0.8909    0.6991    0.5060
    0.1386    0.5472    0.9593
>> C = rot90(A)            % 按逆时针方向旋转 90°
C =
    0.9593    0.5472    0.1386
    0.5060    0.6991    0.8909
```

(2) 矩阵的翻转。矩阵的翻转分左右翻转和上下翻转。对矩阵实施左右翻转是将原矩阵的第 1 列和最后 1 列调换,第 2 列和倒数第 2 列调换,以此类推。MATLAB 对矩阵 A 实施左右翻转的函数是 fliplr(A)。例如:

```
>> A = randi(99,2,5)
A =
    78    13    47    34    79
    93    57     2    17    31
>> B = fliplr(A)
B =
    79    34    47    13    78
    31    17     2    57    93
```

与矩阵的左右翻转类似,矩阵的上下翻转是将原矩阵的第 1 行与最后 1 行调换,第 2 行与倒数第 2 行调换,以此类推。MATLAB 对矩阵 A 实施上下翻转的函数是 flipud(A)。

【例 3-7】 将 n 阶方阵 $\boldsymbol{A}$ 副对角线元素置零。

命令 1:

```
A - flipud(diag(diag(flipud(A))))
```

命令 2:

```
m = eye(size(A))
k = ~flipud(m)
k. * A
```

命令 3:

```
m = eye(size(A))
A - flipud(m). * A
```

3.2.3 矩阵的逆与伪逆

1. 矩阵的逆

若方阵 $\boldsymbol{A}$、$\boldsymbol{B}$ 满足等式

$$\boldsymbol{AB} = \boldsymbol{BA} = \boldsymbol{I} \quad (\boldsymbol{I} \text{ 为单位矩阵})$$

则称 $\boldsymbol{A}$ 为 $\boldsymbol{B}$ 的逆矩阵,当然,$\boldsymbol{B}$ 也为 $\boldsymbol{A}$ 的逆矩阵。这时 $\boldsymbol{A}$、$\boldsymbol{B}$ 都称为可逆矩阵(或非奇异矩阵、满秩矩阵),否则称为不可逆矩阵(或奇异矩阵、降秩矩阵)。

MATLAB 提供的 inv(A)函数可以用来计算方阵的逆矩阵。若 A 为奇异矩阵、接近奇异矩阵或降秩矩阵时,系统将会给出警告信息。inv(A)函数等效于 A^(-1)。

【例 3-8】 求方阵 $\boldsymbol{A}$ 的逆矩阵并赋值给 $\boldsymbol{B}$,且验证 $\boldsymbol{A}$ 与 $\boldsymbol{B}$ 是互逆的。

命令如下:

```
>> A = [1 -1 1;5 -4 3;2 1 1];
```

```
>> B = inv(A)
B =

    -1.4000    0.4000    0.2000
     0.2000   -0.2000    0.4000
     2.6000   -0.6000    0.2000
>> C = A * B
C =
     1.0000    0.0000    0.0000
    -0.0000    1.0000    0.0000
    -0.0000    0.0000    1.0000
>> D = B * A
D =
     1.0000    0.0000    0.0000
    -0.0000    1.0000    0.0000
    -0.0000    0.0000    1.0000
>> C == D
ans =
  3×3 logical 数组
   0   0   0
   0   0   0
   1   0   0
```

在理想情况下，AB=BA。但是，由于计算机采用二进制存储和处理数据，影响浮点运算的精度，因此 AB 和 BA 非常接近，并不相等，命令 C==D 的结果为逻辑 0(假)。如果执行以下命令，就可以看到两个变量的值是不同的。

```
>> format long
>> C - D
ans =
    1.0e-14 *
  -0.044408920985006   -0.005551115123126    0.016653345369377
  -0.210942374678780    0.044408920985006   -0.005551115123126
                   0   -0.022204460492503    0.022204460492503
```

结果表明，虽然 C−D 的值不等于 0(所以 C 不等于 D)，但 C−D 的值是 10^{-14} 数量级的结果，所以可以认为 C−D 的值接近于 0，即 C 约等于 D。

2. 矩阵的伪逆

如果矩阵 $\boldsymbol{A}$ 不是一个方阵，或者 $\boldsymbol{A}$ 是一个非满秩的方阵时，矩阵 $\boldsymbol{A}$ 没有逆矩阵，但可以找到一个与 $\boldsymbol{A}$ 的转置矩阵 $\boldsymbol{A}'$ 同型的矩阵 $\boldsymbol{B}$，使得：

$$\boldsymbol{ABA}=\boldsymbol{A}$$

$$\boldsymbol{BAB}=\boldsymbol{B}$$

此时称矩阵 $\boldsymbol{B}$ 为矩阵 $\boldsymbol{A}$ 的伪逆，也称为广义逆矩阵。

在 MATLAB 中，求一个矩阵伪逆的函数是 pinv(A)。例如：

```
>> A = [3,1,1,1;1,3,1,1;1,1,3,1];
>> B = pinv(A)
B =
    0.3929   -0.1071   -0.1071
   -0.1071    0.3929   -0.1071
   -0.1071   -0.1071    0.3929
    0.0357    0.0357    0.0357
```

若 A 是一个奇异矩阵，无一般意义上的逆矩阵，但可以求 A 的伪逆矩阵。例如：

```
>> A = [0,0,0;0,1,0;0,0,1];
>> pinv(A)
ans =
```

```
     0     0     0
     0     1     0
     0     0     1
```

本例中，A 的伪逆矩阵和 A 相等，这是一个巧合。一般说来，矩阵的伪逆矩阵和自身是不同的。

3.3 矩阵求值

矩阵求值是指对一个矩阵进行某种运算，其结果是一个数值，包括求矩阵的行列式值、秩、迹、范数、条件数等。

3.3.1 方阵的行列式

将一个方阵看作一个行列式，并对其按行列式的规则求值，这个值就称为矩阵所对应的行列式的值。

在 MATLAB 中，求方阵 A 所对应的行列式的值的函数是 det(A)。例如：

```
>> A = [1,2,3; 2,1,0; 12,5,9]
A =
     1     2     3
     2     1     0
    12     5     9
>> dA = det(A)
dA =
   -33
```

看一个例子，求矩阵 A 的行列式值。

```
>> A = [1,2,3;4,5,6;7,8,9];
>> det(A)
ans =
  -9.5162e-16
```

在这里，A 的元素是整数。数学上用对角线法则求 A 的行列式值，结果也为整数，这里应该为 0，但这里的 det 函数返回结果并不是 0，而只是一个很接近 0 的数(约等于 0)。

对于高阶矩阵，用对角线法则是不现实的，所以 det 函数使用 lu 函数基于高斯消去法获取的三角形因子来计算行列式，这很容易出现浮点舍入误差。下面看一个例子：

```
>> A = [1,2,3;4,5,6;7,8,9];
>> [L,U] = lu(A);
>> s = det(L); % 其结果总为 1 或 -1
>> d1 = s * prod(diag(U))
d1 =
  -9.5162e-16
```

结果显示，两种不同方法得到的行列式值相等。这说明在分析 MATLAB 的执行结果时要从算法实现上去分析，尽管应用 MATLAB 时不需要详细了解算法的实现过程，但还是需要具有算法思维的意识和习惯。

由于实数在计算机中表示的误差，一般来说对实数不要进行“相等”或“不等于”的判断，而要进行“约等于”的判断，即两个数相减取绝对值小于一个很小的数。例如，判断 A 的行列式是否为 0：

```
>> d1 == 0
ans =
  logical
   0
```

```
>> abs(d1)<=1e-10
ans =
  logical
   1
```

应该采用第二种方法,其中判断的精度可以根据实际情况做出选择。再看一个例子:

```
>> sqrt(2)*sqrt(2)==2
ans =
  logical
   0
>> sqrt(2)*sqrt(2)-2
ans =
   4.4409e-16
>> abs(sqrt(2)*sqrt(2)-2)<1e-6
ans =
  logical
   1
```

在 MATLAB 中类似的问题还有很多,其分析方法是一样的。

3.3.2 矩阵的秩与迹

1. 矩阵的秩

矩阵线性无关的行数或列数称为矩阵的秩。一个 $m\times n$ 矩阵 $\boldsymbol{A}$ 可以看成由 m 个行向量组成或由 n 个列向量组成。通常,对于一组向量 $\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_p$,若存在一组不全为零的数 k_i $(i=1,2,\cdots,p)$,使

$$k_1\boldsymbol{x}_1+k_2\boldsymbol{x}_2+\cdots+k_p\boldsymbol{x}_p=\boldsymbol{0}$$

成立,则称这 p 个向量线性相关,否则称线性无关。对于 $m\times n$ 矩阵 $\boldsymbol{A}$,若 m 个行向量中有 r $(r\leqslant m)$个行向量线性无关,而其余为线性相关,则称 r 为矩阵 $\boldsymbol{A}$ 的行秩;类似地,可定义矩阵 $\boldsymbol{A}$ 的列秩。矩阵的行秩和列秩总是相等的,因此将行秩和列秩统称为矩阵的秩,也称为该矩阵的奇异值。

在 MATLAB 中,求矩阵秩的函数是 rank(A)。例如:

```
>> A=[1,2,3; 2,1,0; 12,5,9];
>> r=rank(A)
r=
     3
>> A1=[1,2,3; 4,8,12; 12,5,9];
>> r=rank(A1)
r =
     2
```

说明 A 是一个满秩矩阵,A1 是一个不满秩矩阵。A1 又称为奇异矩阵。

2. 矩阵的迹

矩阵的迹即矩阵的主对角线元素之和,也等于矩阵的特征值之和。

在 MATLAB 中,函数 trace(A)用于求矩阵的迹。例如:

```
>> A=[1,2,3; 2,1,0; 12,5,9];
>> trace(A)
ans=
        11
```

3.3.3 向量和矩阵的范数

在求解线性方程组时,由于实际的观测和测量误差以及计算过程中的舍入误差的影响,所

求得的数值解与精确解之间存在一定的差异，为了了解数值解的精确程度，必须对解的误差进行估计，线性代数中常采用范数进行线性变换的误差分析。范数有多种方法定义，其定义不同，范数值也就不同，因此，讨论向量和矩阵的范数时，一定要弄清是求哪一种范数。

1. 向量的范数

设向量 $\boldsymbol{V}=(v_1,v_2,\cdots,v_n)$，向量的 3 种常用范数定义如下。

(1) 1-范数：向量元素的绝对值之和，即

$$\|\boldsymbol{V}\|_1=\sum_{i=1}^{n}|v_i|$$

(2) 2-范数：向量元素平方和的平方根，即

$$\|\boldsymbol{V}\|_2=\sqrt{\sum_{i=1}^{n}v_i^2}$$

(3) ∞-范数：所有向量元素绝对值的最大值，即

$$\|\boldsymbol{V}\|_\infty=\max_{1\leqslant i\leqslant n}\{|v_i|\}$$

在 MATLAB 中，函数 norm 用于计算向量的范数，其基本调用格式如下：

(1) norm(v)：求向量 v 的 2-范数。

(2) norm(v,p)：求向量 v 的 p-范数，针对 3 种常用范数，p 可取值为 1、2(默认值)、Inf。

例如：

```
>> va = randi(19,1,5) - 10
va =
     0     9    -3     2    -5
>> nva = [norm(va,1),norm(va),norm(va,inf)]     % 求向量 V 的 3 种范数
nva =
   19.0000   10.9087    9.0000
```

从几何意义上讲，两向量之差的 2-范数代表两个点之间的欧几里得距离。以平面直角坐标系为例，将坐标点(x,y)表示为两个元素的向量，可以使用 norm 函数来计算两点之间的距离。例如：

```
>> a = [0, 4];
>> b = [ - 3, 2];
>> d = norm(b - a)
d =
    3.6056
```

即 $d=\sqrt{(-3-0)^2+(2-4)^2}=\sqrt{13}$。

2. 矩阵的范数

设 $\boldsymbol{A}$ 是一个 $m\times n$ 矩阵，$\boldsymbol{V}$ 是一个含有 n 个元素的列向量，定义：

$$\|\boldsymbol{A}\|=\max\{\|\boldsymbol{AV}\|\},\quad \|\boldsymbol{V}\|=1$$

因为 $\boldsymbol{A}$ 是一个 $m\times n$ 矩阵，而 $\boldsymbol{V}$ 是一个含有 n 个元素的列向量，所以 $\boldsymbol{AV}$ 是一个含有 m 个元素的列向量。在前面已经定义了 3 种不同的向量范数，按照上式也可以定义 3 种矩阵范数，这样定义的矩阵范数 $\|\boldsymbol{A}\|$ 称为 $\boldsymbol{A}$ 从属于向量的范数。

上式只给出了矩阵范数的基本定义，未给出具体计算方法，完全按照上式是难以计算一个矩阵的某种具体范数的。从属于 3 种向量范数的矩阵范数计算公式是(a_{ij} 是矩阵 $\boldsymbol{A}$ 的元素)：

(1) 1-范数：组成矩阵的各个列向量元素的绝对值之和的最大值，即

$$\|\boldsymbol{A}\|_1=\max_{\|\boldsymbol{V}\|=1}\{\|\boldsymbol{AV}\|_1\}=\max_{1\leqslant j\leqslant n}\left\{\sum_{i=1}^{m}|a_{ij}|\right\}$$

(2) 2-范数：$\boldsymbol{A}'\boldsymbol{A}$ 最大特征值的平方根，即 $\|\boldsymbol{A}\|_2=\max\limits_{\|\boldsymbol{V}\|=1}\{\|\boldsymbol{AV}\|_2\}=\sqrt{\lambda_1}$，其中，$\lambda_1$ 为 $\boldsymbol{A}'\boldsymbol{A}$ 的最大特征值。

(3) ∞-范数：组成矩阵的各个行向量元素的绝对值之和的最大值，即

$$\|\boldsymbol{A}\|_\infty=\max_{\|\boldsymbol{V}\|=1}\{\|\boldsymbol{AV}\|_\infty\}=\max_{1\leqslant i\leqslant m}\left\{\sum_{j=1}^{n}|a_{ij}|\right\}$$

例如：

```
>> A = [1,2,3,4; -9,0,2,5]
A =
      1     2     3     4
     -9     0     2     5
>> nA = [norm(A,1),norm(A),norm(A,inf)] % 求矩阵 A 的范数
nA =
      10.0000   10.6519   16.0000
```

3.3.4 矩阵的条件数

在求解线性方程组 $\boldsymbol{AX}=\boldsymbol{b}$ 时，一般认为，系数矩阵 $\boldsymbol{A}$ 中个别元素的微小扰动不会引起解向量的很大变化。这样的假设在工程应用中非常重要，因为一般系数矩阵的数据是由实验数据获得的，并非精确值，但与精确值误差不大。由上面的假设可以得出如下结论：当参与运算的系数与实际精确值误差很小时，所获得的解与问题的精确解误差也很小。遗憾的是，上述假设并非总是正确的。对于有的系数矩阵，个别元素的微小扰动会引起解的很大变化。在计算数学中，称这种矩阵为病态矩阵，而称解不因其系数矩阵的微小扰动而发生大的变化的矩阵为良性矩阵。当然，良性与病态是相对的，需要一个参数来描述，条件数就是用来描述矩阵的这种性能的一个参数。

矩阵 $\boldsymbol{A}$ 的条件数等于 $\boldsymbol{A}$ 的范数与 $\boldsymbol{A}$ 的逆矩阵的范数的乘积，即 $\text{cond}(\boldsymbol{A})=\|\boldsymbol{A}\|\cdot\|\boldsymbol{A}^{-1}\|$，这样定义的条件数总是大于或等于 1 的。设条件数用 k 表示，解的相对误差用 $\varepsilon(x)$ 表示，$\boldsymbol{A}$ 的相对误差用 $\varepsilon(\boldsymbol{A})$ 表示，$\boldsymbol{b}$ 的相对误差用 $\varepsilon(\boldsymbol{b})$ 表示，则数学上有这样的结论：

$$\varepsilon(x)\approx k[\varepsilon(\boldsymbol{A})+\varepsilon(\boldsymbol{b})]$$

即解的相对误差约等于 $\boldsymbol{A}$ 和 $\boldsymbol{b}$ 的相对误差的 k 倍，当 k 很大时，即使 $\boldsymbol{A}$ 和 b 的相对误差很小，解的相对误差也可能很大。由此可知，舍入误差对解的影响的大小取决于条件数的大小。条件数越接近于 1，矩阵的性能越好，方程组称为良态方程组；反之，矩阵的性能越差，方程组称为病态方程组。

矩阵有 3 种范数，相应地可以定义 3 种条件数。在 MATLAB 中，计算矩阵 A 的 3 种条件数的函数是：

(1) cond(A,1)用于计算 $\boldsymbol{A}$ 的 1-范数下的条件数，即

$$\text{cond}(\boldsymbol{A},1)=\|\boldsymbol{A}\|_1\|\boldsymbol{A}^{-1}\|_1$$

(2) cond(A)或 cond(A,2)用于计算 $\boldsymbol{A}$ 的 2-范数下的条件数，即

$$\text{cond}(\boldsymbol{A})=\|\boldsymbol{A}\|_2\|\boldsymbol{A}^{-1}\|_2$$

(3) cond(A,inf)用于计算 $\boldsymbol{A}$ 的∞-范数下的条件数，即

$$\text{cond}(\boldsymbol{A},\text{inf})=\|\boldsymbol{A}\|_\infty\|\boldsymbol{A}^{-1}\|_\infty$$

例如：

```
>> A = [1,2,3;3, -4,5; -5,6,7];
>> cA = cond(A)
```

```
cA =
    5.4598
>> B = [1,2,5; - 2, - 7,5; - 5,1, - 2];
>> cB = cond(B)
cB =
    2.1901
```

矩阵 $\boldsymbol{B}$ 的条件数比矩阵 $\boldsymbol{A}$ 的条件数更接近于 1，因此，矩阵 $\boldsymbol{B}$ 的性能要好于矩阵 $\boldsymbol{A}$。

3.4 矩阵的特征值与特征向量

矩阵的特征值与特征向量在科学研究和工程计算中应用广泛。例如，机械中的振动问题、电磁振荡中的某些临界值的确定等问题，往往归结成求矩阵的特征值与特征向量的问题。

设 $\boldsymbol{A}$ 是 n 阶方阵，如果存在数 λ 和 n 维非零向量 $\boldsymbol{x}$，使得 $\boldsymbol{Ax}=\lambda\boldsymbol{x}$ 成立，则称 λ 是矩阵 $\boldsymbol{A}$ 的一个特征值或本征值，称向量 $\boldsymbol{x}$ 为矩阵 $\boldsymbol{A}$ 属于特征值 λ 的特征向量或本征向量，简称 $\boldsymbol{A}$ 的特征向量或 $\boldsymbol{A}$ 的本征向量。

在 MATLAB 中，计算矩阵 A 的特征值和特征向量的函数是 eig(A)，常用的调用格式有如下 3 种。

(1) V=eig(A)：求矩阵 A 的全部特征值，构成向量 V。

(2) [X,D]=eig(A)：求矩阵 A 的全部特征值，构成对角阵 D，并产生矩阵 X，X 各列是相应的特征向量，满足 AX=XD。

(3) [X,D]=eig(A,'nobalance')：与第 2 种格式类似，但第 2 种格式中先对 A 作相似变换后求矩阵 A 的特征值和特征向量，而格式 3 直接求矩阵 A 的特征值和特征向量。

矩阵 A 的特征向量有无穷多个，eig 函数只找出其中的 n 个，A 的其他特征向量，均可由这 n 个特征向量的线性组合表示。

例如：

```
>> A = [1,1,0.5;1,1,0.25;0.5,0.25,2];
>> [V,D] = eig(A)
V =
     0.7212      0.4443     0.5315
    - 0.6863      0.5621     0.4615
    - 0.0937    - 0.6976     0.7103
D =
    - 0.0166           0          0
          0      1.4801          0
          0           0     2.5365
```

求得的 3 个特征值是－0.0166、1.4801 和 2.5365，各特征值对应的特征向量为 V 的各列向量。

下面验证特征值和特征向量。

```
>> A * V
ans =
    - 0.0120     0.6576     1.3481
      0.0114     0.8320     1.1705
      0.0016   - 1.0325     1.8018
>> V * D
ans =

    - 0.0120     0.6576     1.3481
      0.0114     0.8320     1.1705
      0.0016   - 1.0325     1.8018
```

```
>> A * V == V * D
ans =
  3×3 logical 数组
   0   0   0
   0   0   0
   0   0   0
```

请读者思考,看起来 A * V 与 V * D 的结果是一样的,但为什么“等于”关系运算结果为 0 呢? 即这里 A * V 与 V * D 为何不相等?

从理论上讲,特征值分解可满足 AV=VD。但是,由于计算机采用有限的二进制位存储和处理数据,影响浮点运算的精度,因此 AV 与 VD 非常接近,并不相等。看下列结果。

```
>> format long
>> A * V - V * D
ans =
   1.0e-15 *
   0.083266726846887   -0.111022302462516   0.222044604925031
   0.204697370165263    0.222044604925031   0.444089209850063
   0.203396327558281    0.222044604925031   0.222044604925031
>> format
```

A * V−V * D 的结果是很小的数,可以理解为约等于 0,即 A * V≈V * D,这是由计算机解题的特点所决定的。

3.5 稀疏矩阵的操作

稀疏矩阵是指具有大量的零元素,而仅含少量的非零元素的矩阵。通常,一个 $m \times n$ 实矩阵需要占据 $m \times n$ 个存储单元。这对于一个 m、n 较大的矩阵来说,无疑将要占据相当大的内存空间。对稀疏矩阵来说,若将大量的零元素也存储起来,显然是对内存资源的一种浪费。为此,MATLAB 为稀疏矩阵提供了方便、灵活、有效的存储技术。

3.5.1 矩阵存储方式

MATLAB 的矩阵有两种存储方式:完全存储方式和稀疏存储方式。

1. *完全存储方式*

完全存储方式是将矩阵的全部元素按列逐个存储。迄今为止,介绍的矩阵存储方式都是按这个方式存储的,完全存储方式对稀疏矩阵也适用。例如,$m \times n$ 的实矩阵需要 $m \times n$ 个存储单元,而复矩阵需要 $m \times 2n$ 个存储单元。例如:

```
>> A = [1,0,0,0,0; 0,5,0,0,0; 2,0,0,7,0]
A =
     1     0     0     0     0
     0     5     0     0     0
     2     0     0     7     0
>> B = [1,0,0; 0,5 + 3i,0; 2i,0,0]
B =
   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   5.0000 + 3.0000i   0.0000 + 0.0000i
   0.0000 + 2.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
>> whos
  Name        Size              Bytes  Class     Attributes
  A           3×5                 120  double
  B           3×3                 144  double    complex
```

通常,一个 double 类型数据占 8 字节,所以这里的 3×5 矩阵 A 占 120 字节,3×3 复矩阵

B 占 144 字节。

2. 稀疏存储方式

稀疏存储方式仅存储矩阵所有的非零元素的值及其位置，即行号和列号。显然，这对于具有大量零元素的稀疏矩阵来说是十分有效的。例如：

```
>> A = [1,0,0,0,0; 0,5,0,0,0; 2,0,0,7,0]
A =
     1     0     0     0     0
     0     5     0     0     0
     2     0     0     7     0
```

矩阵 A 的稀疏存储方式如下：

```
(1,1)      1
(3,1)      2
(2,2)      5
(3,4)      7
```

其中，括号内为元素的行列位置，其后面为元素值。在 MATLAB 中，稀疏存储方式也是按列顺序存储的，先存储矩阵第 1 列非零元素，再存储矩阵第 2 列非零元素，以此类推。在工作空间观测分别用这两种方式存储的变量，可以看到稀疏存储方式会有效地节省存储空间。

稀疏矩阵是指矩阵的零元素较多、具有稀疏特征的矩阵，稀疏存储矩阵是指采用稀疏方式存储的矩阵。稀疏矩阵不一定非得采用稀疏存储方式，也可以采用完全存储方式。

3.5.2 生成稀疏矩阵

1. 将完全存储方式转化为稀疏存储方式

函数 A＝sparse(S)将矩阵 S 转化为稀疏存储方式的矩阵 A。当矩阵 S 是稀疏存储方式时，则函数调用相当于 A＝S。

【例 3-9】 设

$$\boldsymbol{X}=\begin{bmatrix}2&0&0&0&0\\0&0&0&0&0\\0&0&0&5&0\\0&1&0&0&-1\\0&0&0&0&-5\end{bmatrix}$$

将 $\boldsymbol{X}$ 转化为稀疏存储方式。

命令如下：

```
>> X = [2,0,0,0,0;0,0,0,0,0;0,0,0,5,0;0,1,0,0,-1;0,0,0,0,-5];
>> A = sparse(X)
A =
   (1,1)        2
   (4,2)        1
   (3,4)        5
   (4,5)       -1
   (5,5)       -5
```

A 就是 X 的稀疏存储方式。

sparse 函数还有其他一些调用格式。

(1) sparse(m,n)：生成一个 m×n 的所有元素都是 0 的稀疏矩阵。

(2) sparse(u,v,S)：其中 u、v、S 是 3 个等长的向量。S 是要建立的稀疏矩阵的非零元素，u(i)、v(i)分别是 S(i)的行和列下标，该函数建立一个 max(u)行、max(v)列并以 S 为稀疏

元素的稀疏矩阵。

此外,还有一些和稀疏矩阵操作有关的函数。例如:

(1) [u,v,S]=find(A)——返回矩阵 A 中非零元素的下标和元素。这里产生的 u、v、S 可作为 sparse(u,v,S)的参数。

(2) full(A)——返回和稀疏存储矩阵 A 对应的完全存储方式矩阵。

例如:

```
>> A = sparse([1,2,4,5],[1,3,2,8],[1,10,-2,40])
A =
   (1,1)         1
   (4,2)        -2
   (2,3)        10
   (5,8)        40
>> B = full(A)
B =
     1     0     0     0     0     0     0     0
     0     0    10     0     0     0     0     0
     0     0     0     0     0     0     0     0
     0    -2     0     0     0     0     0     0
     0     0     0     0     0     0     0    40
>> whos
  Name        Size                Bytes  Class     Attributes
  A           5x8                   136  double    sparse
  B           5x8                   320  double
```

B 是 A 对应的完全存储矩阵。可以看出,稀疏存储矩阵 A 所占的内存字节数比相应的完全存储矩阵 B 要少得多。

(3) issparse(A)——用于判断矩阵 A 是否为稀疏矩阵。

2. *产生稀疏存储矩阵*

sparse 函数可以将一个完全存储方式的矩阵转化为稀疏存储方式,但在实际应用时,如果要构建一个稀疏存储方式的大矩阵,按照上述方法,必须先建立该矩阵的完全存储方式矩阵,然后使用 sparse 函数进行转化,这显然是不可取的。能否只把要建立的稀疏矩阵的非零元素及其所在行和列的位置表示出来,然后直接产生其稀疏存储方式呢? MATLAB 提供了 spconvert 函数,其调用格式为:

```
B = spconvert(A)
```

该函数将 A 所描述的一个矩阵转化为一个稀疏存储矩阵。其中,A 为一个 m×3 或 m×4 的矩阵,其每行表示一个非零元素,m 是非零元素的个数。A(i,1)表示第 i 个非零元素所在的行,A(i,2)表示第 i 个非零元素所在的列,A(i,3)表示第 i 个非零元素值的实部,A(i,4)表示第 i 个非零元素值的虚部。若矩阵的全部元素都是实数,则无需第 4 列。

【例 3-10】 根据表示稀疏矩阵的矩阵 $\boldsymbol{A}$,生成一个稀疏矩阵 $\boldsymbol{B}$。

$$\boldsymbol{A}=\begin{bmatrix}2 & 2 & 1\\ 3 & 1 & -1\\ 4 & 3 & 3\\ 5 & 3 & 8\\ 6 & 6 & 12\end{bmatrix}$$

命令如下:

```
>> A = [2,2,1;3,1,-1;4,3,3;5,3,8;6,6,12];
```

```
>> B = spconvert(A)
B =
   (3,1)       -1
   (2,2)        1
   (4,3)        3
   (5,3)        8
   (6,6)       12
```

注意，矩阵 A 并非稀疏存储矩阵，B 才是稀疏存储矩阵。

3. 带状稀疏存储函数

用 spdiags 函数产生带状稀疏存储矩阵。为便于理解，先看下列具有稀疏特征的带状矩阵。

$$\boldsymbol{X}=\begin{bmatrix}11 & 0 & 0 & 12 & 0 & 0\\ 0 & 21 & 0 & 0 & 22 & 0\\ 0 & 0 & 31 & 0 & 0 & 32\\ 41 & 0 & 0 & 42 & 0 & 0\\ 0 & 51 & 0 & 0 & 52 & 0\end{bmatrix}$$

希望产生一个稀疏存储矩阵 $\boldsymbol{A}$。

首先找出 $\boldsymbol{X}$ 矩阵的特征数据。包括矩阵的大小：5×6；有 3 条对角线，它们的位置与值依次为：第 1 条位于主对角线下方第 3 条，记 $d_1=-3$，该对角线元素值为 0,0,0,41,51；第 2 条为主对角线，记 $d_2=0$，元素值为 11,21,31,42,52；第 3 条位于主对角线上方第 3 条，记 $d_3=3$，元素值为 12,22,32,0,0。于是，将带状对角线的值构成下列矩阵 $\boldsymbol{B}$，将带状的位置构成向量 $\boldsymbol{d}$：

$$\boldsymbol{B}=\begin{bmatrix}0 & 11 & 12\\ 0 & 21 & 22\\ 0 & 31 & 32\\ 41 & 42 & 0\\ 51 & 52 & 0\end{bmatrix},\quad \boldsymbol{d}=\begin{bmatrix}-3\\ 0\\ 3\end{bmatrix}$$

然后利用 spdiags 函数产生一个稀疏存储矩阵。

```
>> B = [0,0,0,41,51;11,21,31,42,52;12,22,32,0,0]'
B =
     0    11    12
     0    21    22
     0    31    32
    41    42     0
    51    52     0
>> d = [-3;0;3]
d =
    -3
     0
     3
>> A = spdiags(B,d,5,6)      % 产生一个稀疏存储矩阵 A
A =
   (1,1)       11
   (4,1)       41
   (2,2)       21
   (5,2)       51
   (3,3)       31
   (1,4)       12
   (4,4)       42
   (2,5)       22
```

```
   (5,5)        52
   (3,6)        32
```

spdiags 函数用于产生带状稀疏矩阵的稀疏存储矩阵,其调用格式如下:

```
A = spdiags(B,d,m,n)
```

其中,m、n 为原带状矩阵的行数与列数。B 为 r×p 矩阵,这里 r=min(m,n),p 为原带状矩阵所有非零对角线的条数,矩阵 B 的第 i 列即为原带状矩阵的第 i 条非零对角线。d 为具有 p 个元素的列向量,d_i 存储该原带状矩阵的第 i 条对角线的位置。d_i 的取法:若是主对角线,取 $d_i=0$,若位于主对角线的下方第 s 条对角线,取 $d_i=-s$,若位于主对角线的上方第 s 条对角线,则取 $d_i=s$。矩阵 B 的第 i 个列向量的构成方法是:若原矩阵对角线上元素个数等于 r,则取全部元素;若非零对角线上元素个数小于 r,则用零元素填充,凑足 r 个元素。填充零元素的原则:若 m<n(行数小于列数),则当 $d_i<0$,在对角线元素前面填充零元素;当 $d_i>0$,则在对角线元素后面填充零元素。若 m≥n(行数大于或等于列数),则当 $d_i<0$ 时,在对角线元素后面填充零元素;当 $d_i>0$ 时,在对角线元素前面填充零元素。

spdiags 函数的其他调用格式如下。

(1) [B, d]=spdiags(A):从原带状矩阵 A 中提取全部非零对角线元素赋给矩阵 B 及其这些非零对角线的位置向量 d。

(2) B=spdiags(A,d):从原带状矩阵 A 中提取由向量 d 所指定的那些非零对角线元素构成矩阵 B。

(3) E=spdiags(B, d, A):在原带状矩阵 A 中将由向量 d 所指定的那些非零对角线元素用矩阵 B 替代,构成一个新的矩阵带状矩阵 E。

4. 单位矩阵的稀疏存储

单位矩阵只有对角线元素为 1,其他元素都为 0,是一种具有稀疏特征的矩阵。函数 eye 生成一个完全存储方式的单位矩阵。MATLAB 还提供了一个生成稀疏存储方式的单位矩阵的 speye 函数,speye(m,n)返回一个 m×n 的稀疏单位矩阵。例如:

```
>> s = speye(3,5)
s =
   (1,1)        1
   (2,2)        1
   (3,3)        1
```

5. 获取稀疏矩阵非零元素信息

稀疏矩阵的一个重要信息是非零元素,可以使用以下函数实现操作。

(1) nnz(A):返回矩阵 A 中的非零元素的个数。

(2) nonzeros(A):返回由矩阵 A 的所有非零元素构成的列向量。

例如:

```
>> A = [0,0,3;0,5,0;0,0,9];
>> nnz(A)
ans =
     3
>> nonzeros(A)
ans =
     5
     3
     9
```

3.5.3 稀疏矩阵的运算

稀疏存储矩阵只是矩阵的存储方式不同，它的运算规则与普通矩阵是一样的，所以，在运算过程中，稀疏存储矩阵可以直接参与运算。当参与运算的对象不全是稀疏存储矩阵时，所得结果一般是完全存储形式。例如：

```
>> A = [0,0,3;0,5,0;0,0,9];
>> B = sparse(A);
>> B * B            % 两个稀疏存储矩阵相乘，结果仍为稀疏存储矩阵
ans =
     (2,2)        25
     (1,3)        27
     (3,3)        81
>> rand(3) * B      % 完全存储矩阵与稀疏存储矩阵相乘，结果为完全存储矩阵
ans =
         0    1.1191    7.4328
         0    3.7563    7.3128
         0    1.2755    9.7739
```

【例 3-11】 求下列三对角线性代数方程组的解。

$$\begin{bmatrix} 2 & 3 & & & \\ 1 & 4 & 1 & & \\ & 1 & 6 & 4 & \\ & & 2 & 6 & 2 \\ & & & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 3 \\ 2 \\ 1 \\ 5 \end{bmatrix}$$

A 是一个 5×5 的带状稀疏矩阵，命令如下：

```
>> B = [1,1,2,1,0; 2,4,6,6,1; 0,3,1,4,2].';        % 取 A 对角线上元素构成 B
>> d = [-1;0;1];                                   % 生成带状位置向量
>> A = spdiags(B,d,5,5);                           % 生成稀疏存储的系数矩阵
>> b = [0;3;2;1;5];                                % 方程右边参数向量
>> x = (inv(A) * b)'                               % 求解
x =
   -0.1667    0.1111    2.7222   -3.6111    8.6111
```

也可以采用完全存储方式来存储系数矩阵，命令如下：

```
>> A1 = full(A);
>> x1 = (inv(A1) * b)'
x1 =
   -0.1667    0.1111    2.7222   -3.6111    8.6111
```

从本例可以看出，无论用完全存储还是用稀疏存储，所得到的解线性代数方程组的解是一样的。

3.6 应用实战 3

【例 3-12】 设

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{R}_{3\times3} & \boldsymbol{O}_{3\times2} \\ \boldsymbol{O}_{2\times3} & \boldsymbol{S}_{2\times2} \end{bmatrix}$$

又设 λ_i 为 $\boldsymbol{R}$ 的特征值，λ_j 为 $\boldsymbol{S}$ 的特征值，$\boldsymbol{x}_i=(\alpha_1,\alpha_2,\alpha_3)'$ 是 $\boldsymbol{R}$ 对应于 λ_i 的特征向量，$\boldsymbol{y}_j=(\beta_1,\beta_2)'$ 是 $\boldsymbol{S}$ 对应于 λ_j 的特征向量，试验证：

(1) λ_i、λ_j 为 $\boldsymbol{A}$ 的特征值。

(2) $\boldsymbol{p}_i=(\alpha_1,\alpha_2,\alpha_3,0,0)'$ 是 $\boldsymbol{A}$ 对应于 λ_i 的特征向量，$\boldsymbol{q}_j=(0,0,0,\beta_1,\beta_2)'$ 是 $\boldsymbol{A}$ 对应于 λ_j 的特征向量。

首先建立 R、S 矩阵，再构建 A 矩阵，再调用 eig 函数求矩阵 R、S、A 的特征向量矩阵和特征值矩阵。命令如下：

```
>> R = [ - 1,2,0;2, - 4,1;1,1, - 6];
>> S = [1,2;2,3];
>> A = [R,zeros(3,2);zeros(2,3),S];
>> [X1,d1] = eig(R)
X1 =

    0.8553    0.4517    0.1899
    0.4703  - 0.8395  - 0.5111
    0.2173  - 0.3021    0.8383
d1 =
    0.0996         0         0
         0  - 4.7165         0
         0         0  - 6.3832
>> [X2,d2] = eig(S)
X2 =
  - 0.8507    0.5257
    0.5257    0.8507
d2 =
  - 0.2361         0
         0    4.2361
>> [X3,d3] = eig(A)
X3 =
    0.8553    0.4517    0.1899         0         0
    0.4703  - 0.8395  - 0.5111         0         0
    0.2173  - 0.3021    0.8383         0         0
         0         0         0  - 0.8507  - 0.5257
         0         0         0    0.5257  - 0.8507
d3 =
    0.0996         0         0         0         0
         0  - 4.7165         0         0         0
         0         0  - 6.3832         0         0
         0         0         0  - 0.2361         0
         0         0         0         0    4.2361
```

从命令执行结果可以看出，A 矩阵的特征值由 R 矩阵的特征值和 S 矩阵的特征值组成，关于 A 矩阵每个特征值的特征向量，前 3 个特征向量的前 3 个元素是 R 的特征向量的元素，后两个特征向量的后两个元素是 S 的特征向量的元素，运算结果与结论相符。当然这里只是验证了结论，不算严格的数学证明。

练习题

一、选择题

1. 建立三阶幺矩阵 A 的语句是(　　)。

A. A=one(3)　　B. A=ones(3,1)

C. A=one(3,3)　　D. A=ones(3,3)

2. 建立五阶由两位随机整数构成的矩阵 A，其语句是(　　)。

A. A=fix(10+89 * rand(5))　　B. A=fix(20+90 * rand(5,5))

C. A=fix(10+90 * rand(5))　　D. A=fix(10+100 * rand(5))

3. 建立三阶魔方阵 M 的语句是(　　)。

A. M=magic(3)　　B. M=MAGIC(3)

C. M=Magic(3)　　D. M=magic(1,3)

4. 产生以$(x+y)^5$展开后的系数构成的对角阵 P,可以采用的语句是(　　)。

A. P=diag(flipud(pascal(6)))

B. P=diag(diag(flipud(pascal(6))))

C. P=diag(flipud(pascal(5)))

D. P=diag(diag(flipud(pascal(5))))

5. 将矩阵 A 对角线元素加 30 的命令是(　　)。

A. A+30 * eye(size(A))　　B. A+30 * eye(A)

C. A+30 * ones(size(A))　　D. A+30 * eye(4)

6. 求矩阵 A 的范数的函数是(　　)。

A. trace(A)　　B. cond(A)

C. rank(A)　　D. norm(A)

7. 语句"[X,D]=eig(A)"被执行后,D 是一个(　　)。

A. 三角阵　　B. 对角阵

C. 数量矩阵　　D. 单位阵

8. 语句"A=sparse([0,2,5; 2,0,1])"执行后,输出结果的最后一行是(　　)。

A. (2,1)　2　　B. (1,2)　2

C. (1,3)　5　　D. (2,3)　1

9. 【多选】下列命令对中,结果互为相同的是(　　)。

A. x=(-2: 2)'与 x=[-2: 2]'

B. x=diag(diag(ones(3)))与 x=eye(3)

C. x=triu(A,1)+tril(A,-1)与 x=A-diag(diag(A))

D. x=rot90(A)与 x=fliplr(A)

10. 【多选】矩阵的迹等于矩阵的对角线元素之和,以下方法中能求矩阵迹的有(　　)。

A. trace(A)　　B. sum(diag(A))

C. prod(eig(A))　　D. sum(eig(A))

二、问答题

1. 求下列矩阵的主对角元素、上三角阵、下三角阵、逆矩阵、行列式的值、秩、迹、范数、条件数。

(1) $\boldsymbol{A}=\begin{bmatrix}1 & -1 & 2 & 3\\ 0 & 9 & 3 & 3\\ 7 & -5 & 0 & 2\\ 23 & 6 & 8 & 3\end{bmatrix}$　　(2) $\boldsymbol{B}=\begin{bmatrix}3 & \pi/2 & 45\\ 32 & 76 & \sqrt{37}\\ 5 & 72 & 4.5\times10^{-4}\\ e^2 & 0 & 97\end{bmatrix}$

2. 产生均值为 1、方差为 0.2 的 500 个正态分布的随机数,请写出相应的命令。

3. 多项式的伴随矩阵的特征值与多项式方程的根有何关系? 请用命令验证。

4. 从矩阵 A 提取主对角线元素,并以这些元素构成对角阵 B,请写出相应的命令。

5. 稀疏矩阵与稀疏存储矩阵有何区别?

操作题

1. 建立一个方阵 $\boldsymbol{A}$,求 $\boldsymbol{A}$ 的逆矩阵和 $\boldsymbol{A}$ 的行列式的值。

2. 先生成 $\boldsymbol{A}$ 矩阵,然后将 $\boldsymbol{A}$ 左旋 90°后得到 $\boldsymbol{B}$,右旋 90°后得到 $\boldsymbol{C}$。

$$\boldsymbol{A}=\begin{bmatrix}1 & 4 & 7 & 10\\ 2 & 5 & 8 & 11\\ 3 & 6 & 9 & 12\end{bmatrix},\quad \boldsymbol{B}=\begin{bmatrix}10 & 11 & 12\\ 7 & 8 & 9\\ 4 & 5 & 6\\ 1 & 2 & 3\end{bmatrix},\quad \boldsymbol{C}=\begin{bmatrix}3 & 2 & 1\\ 6 & 5 & 4\\ 9 & 8 & 7\\ 12 & 11 & 10\end{bmatrix}$$

3. 产生五阶希尔伯特矩阵 $\boldsymbol{H}$ 和五阶帕斯卡矩阵 $\boldsymbol{P}$,且求其行列式的值 Hh 和 Hp 以及它们的条件数 Th 和 Tp,判断哪个矩阵性能更好,为什么?

4. 已知:

$$\boldsymbol{A}=\begin{bmatrix}-29 & 6 & 18\\ 20 & 5 & 12\\ -8 & 8 & 5\end{bmatrix}$$

求 $\boldsymbol{A}$ 的特征值及特征向量,并分析其数学意义。

5. 下面是一个线性病态方程组:

$$\begin{bmatrix}1/2 & 1/3 & 1/4\\ 1/3 & 1/4 & 1/5\\ 1/4 & 1/5 & 1/6\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}=\begin{bmatrix}0.95\\ 0.67\\ 0.52\end{bmatrix}\quad\left(\boldsymbol{AX}=\begin{bmatrix}b_1\\ b_2\\ b_3\end{bmatrix}\right)$$

(1) 用矩阵求逆法求方程的解。

(2) 将方程右边向量元素 b_3 改为 0.53,再求解,并比较 b_3 的变化和解的相对变化。

(3) 计算系数矩阵 $\boldsymbol{A}$ 的条件数并分析结论。