

## 第3章 矩 阵 运 算

MATLAB中的大多数运算可以直接对矩阵应用。除了在第 2.4节中讨论的算术运算 +、-、\*、^、/、\ 外，还有用于转置和共轭的运算符、有理数运算符和逻辑运算符。

MATLAB 学生版的用户应该知道矩阵中的元素总数极限是 16384。

此外，矩阵有算术函数和逻辑函数，有些函数仅能在二维矩阵中使用。

### 3.1 加法和减法

如果矩阵 **A** 和 **B** 具有相同的维数，那么就可以定义两个矩阵的和 **A+B** 和两个矩阵的差 **A-B**。矩阵 **A ± B**，即元素  $a_{ij...p} \pm b_{ij...p}$ 。在 MATLAB 中，一个  $m \times n$  矩阵 **A** 和一个标量，即一个  $1 \times 1$  矩阵 **s** 之间也能进行加和减运算。矩阵 **A+s** 得到与 **A** 相同的维数，元素为  $a_{ij}+s$ 。

#### 例3.1

假设 **A** 和 **B** 定义如下：

$$\mathbf{A} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}$$

MATLAB 命令

```
Add=A+B, Sub=AB, Add100=A+100
```

得到结果：

Add =

```
6      8
10     12
```

Sub =

```
-4     -4
-4     -4
```

Add100 =

```
101    102
103    104
```

### 3.2 乘法

如果矩阵 **A** 的列数等于矩阵 **B** 的行数，那么矩阵相乘，即 **C=AB**，就被定义为二维矩阵。如果不是这种情况，MATLAB 就返回一个错误信息。只有一个例外就是这两个矩阵之一是  $1 \times 1$ ，如一个标量，那么 MATLAB 是可以接受的。在 MATLAB 中，乘法的运算符是 \*，因此，命令是 **C=A\*B**。

元素  $c_{ij}$  是 **A** 的第  $i$  行和 **B** 的第  $j$  列的点积。点积的定义可参见命令集 23 和附录 B。矩阵 **C** 有与 **A** 相同的行数和与 **B** 相同的列数。

对于方阵，也定义了积  $BA$ ，但其结果通常与  $AB$  不同。

### 例3.2

(a) 假设  $A$  和  $B$  如同例3.1，命令

```
A, B, MultAB= $A*B$ , MultBA= $B*A$ 
```

在屏幕上显示如下的结果：

```
A =
    1    2
    3    4
```

```
B =
    5    6
    7    8
```

```
MultAB =
    19    22
    43    50
```

```
MultBA =
    23    34
    31    46
```

(b) 令  $x$  和  $y$  为：

```
x = (1 2 3)    y =  $\begin{pmatrix} 1 \\ 10 \\ 100 \end{pmatrix}$ 
```

命令  $S=x*y$ ， $M=y*x$ ，结果为：

```
S =
    321
```

```
M =
    1    2    3
   10   20   30
  100  200  300
```

MATLAB 也包含其他乘积。命令  $\text{dot}(x, y)$  得到具有相同元素数量的两个向量  $x$  和  $y$  的点积，也称为标量积或内积。如果点积为零，则两个向量是正交的。如果  $A$  和  $B$  具有相同的维数，则定义两个矩阵  $A$  和  $B$  的点积，在 MATLAB 中定义为列方式。其结果是一个行向量，其元素是第 1 列、第 2 列等的点积，可参见附录 B。

### 命令集23 点积

```
dot(x, y)    得到向量  $x$  和  $y$  的点积
dot(A, B)    得到一个长度为  $n$  的行向量，这里的元素是  $A$  和  $B$  对应列的点积。矩
              阵  $A$  和  $B$  必须具有相同的维数  $n \times n$ 。多维矩阵可参见 helpdesk。
dot(A, B, dim) 在  $dim$  数组中给出  $A$  和  $B$  的点积。
```

对于各具三个元素的两个向量  $\mathbf{x}$  和  $\mathbf{y}$ , 命令 `cross(x, y)` 给出向量积或叉积, 即:

$$\mathbf{x} \times \mathbf{y} = (x_2y_3 - x_3y_2 \quad x_3y_1 - x_1y_3 \quad x_1y_2 - x_2y_1)$$

对向量  $\mathbf{x}$  和  $\mathbf{y}$ , 向量  $\mathbf{x} \times \mathbf{y}$  是正交的。

`cross` 命令也可以应用于  $3 \times n$  矩阵, 其结果是一个  $3 \times n$  矩阵, 这里的第  $i$  列是  $\mathbf{A}$  和  $\mathbf{B}$  中的第  $i$  列的叉积。

#### 命令集24 叉积

<code>cross(x, y)</code>	得到向量 $\mathbf{x}$ 和 $\mathbf{y}$ 的叉积。
<code>cross(A, B)</code>	得到一个 $3 \times n$ 矩阵, 其中的列是 $\mathbf{A}$ 和 $\mathbf{B}$ 对应列的叉积。矩阵 $\mathbf{A}$ 和 $\mathbf{B}$ 必须具有相同的维数 $3 \times n$ 。
<code>cross(A, B, dim)</code>	在 $dim$ 数组中给出向量 $\mathbf{A}$ 和 $\mathbf{B}$ 的叉积。 $\mathbf{A}$ 和 $\mathbf{B}$ 必须具有相同的维数, <code>size(A, dim)</code> 和 <code>size(B, dim)</code> 必须是 3。

#### 例3.3

假设:

$$\mathbf{x} = (1 \ 0 \ 0) \quad \mathbf{y} = (0 \ 1 \ 0)$$

命令 `crossprod=cross(x, y)` 得到:

$$\text{crossprod} = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}$$

对  $\mathbf{x}$  和  $\mathbf{y}$ , 它是正交的, 即:

$$\text{scalar1} = \text{dot}(\mathbf{x}, \text{crossprod}), \quad \text{scalar2} = \text{dot}(\mathbf{y}, \text{crossprod})$$

得:

$$\text{scalar1} = 0$$

$$\text{scalar2} = 0$$

在 MATLAB 中, 有一个完成二维矩阵卷积的函数。可以使用 FIR 滤波器(有限脉冲响应)作为一个自变量, 这部分内容在 `helpdesk` 中描述。

#### 命令集25 矩阵的卷积

<code>conv2(A, B)</code>	返回矩阵 $\mathbf{A}$ 和 $\mathbf{B}$ 的二维卷积
<code>conv2(hcol, hrow, A)</code>	矩阵 $\mathbf{A}$ 与向量 $\mathbf{hcol}$ 列方式和向量 $\mathbf{hrow}$ 行方式的卷积。
<code>conv2(..., format)</code>	得到一个卷积的特殊形式。参数 <code>format</code> 必须是下列字符串之一: ' same ' 返回最接近中心的部分卷积, 其维数与 $\mathbf{A}$ 相同。 ' valid ' 仅返回不考虑边缘补零计算的部分卷积。
<code>convn(A, B)</code>	返回矩阵 $\mathbf{A}$ 和 $\mathbf{B}$ 的多维卷积。
<code>convn(..., format)</code>	得到卷积的一个特殊形式, 如上所示。

Kronecker张量积可以用于创建大的矩阵，它由命令 `kron(A, B)` 得到。如果 **A** 是一个  $m \times n$  矩阵，**B** 是一个  $k \times r$  矩阵，那么这个命令就返回一个  $m \cdot k \times r \cdot n$  的矩阵。

#### 命令集26 张量积

`kron(A, B)` 得到**A**和**B**的Kronecker张量积。

##### 例3.4

假设：

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

命令 `K=kron(A, B)` 的结果为：

$$\mathbf{K} = \begin{pmatrix} 2 & 4 & 6 & 0 & 0 & 0 \\ 2 & 0 & 2 & 0 & 0 & 0 \\ -1 & -2 & -3 & 1 & 2 & 3 \\ -1 & 0 & -1 & 1 & 0 & 1 \end{pmatrix}$$

### 3.3 除法

在MATLAB中，有两个矩阵除法的符号，左除 `\` 和右除 `/`。如果 **A** 是一个非奇异方阵，那么 `A \ B` 和 `B / A` 对应 **A** 的逆与 **B** 的左乘和右乘，即分别等价于命令 `inv(A)*B` 和 `B*inv(A)`。可是，MATLAB执行它们时是不同的，如例3.5所示。**A** 的逆，`inv(A)` 或 `A \ I` 在第7.1节中介绍。

如果 **A** 是一个方阵，那么 `X=A \ B` 是矩阵方程  $\mathbf{AX}=\mathbf{B}$  的解  $\mathbf{A}^{-1}\mathbf{B}$ ，这里的 **X** 具有与 **B** 相同的维数。在  $\mathbf{B}=\mathbf{b}$  是一个列向量这样一个特殊情况下，`x=A \ b` 是线性系统  $\mathbf{AX}=\mathbf{b}$  的解。参见第7.2节。

如果 **A** 是一个  $m > n$  的  $m \times n$  矩阵，`X=A \ B` 得到矩阵方程  $\mathbf{AX}=\mathbf{B}$  的最小二乘解，参见第7.7节。

矩阵方程  $\mathbf{XA}=\mathbf{B}$  的解是  $\mathbf{X}=\mathbf{B}/\mathbf{A}$ ，它等同于  $(\mathbf{A} \setminus \mathbf{B})'$ ，即右除可以由左除定义。这里，撇号 `'` 表示转置，这将在第3.4节中进行说明。

##### 例3.5

(a) 设 **A** 和 **B** 如例3.1一样定义，命令

`A, B, Right=B/A, Left=A \ B` 得到：

$$\mathbf{A} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

$$\mathbf{B} = \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}$$

$$\mathbf{Right} = \begin{pmatrix} -1 & 2 \\ -2 & 3 \end{pmatrix}$$

```
Left =
    -3    -4
     4     5
```

如果输入  $\text{Right} = B * \text{inv}(A)$  和  $\text{Left} = \text{inv}(A) * B$ , 则得到

```
Right =
   -1.0000    2.0000
   -2.0000    3.0000
```

```
Left =
   -3.0000   -4.0000
    4.0000    5.0000
```

这分别与用 / 和 \ 计算的矩阵结果是一致的, 但浮点格式表明它们的计算过程是不一样的。

(b) 设下列  $A$  和  $b$  :

$$A = \begin{pmatrix} 1 & 3 & 5 \\ 1 & 2 & 4 \\ 0 & 5 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 22 \\ 17 \\ 13 \end{pmatrix}$$

系统  $Ax=b$  的解在 MATLAB 中写作  $x=A \backslash b$ , 得到:

```
x =
    1.0000
    2.0000
    3.0000
```

(c) 使用如上的  $A$  和  $b$ , 检查求解系统  $Ax=b$  的运算次数。

命令 `flops(0); x=inv(A)*b; flops` 给出结果:

```
ans =
    109
```

命令 `flops(0); X=A\b; fl` 输出结果:

```
ans =
    72
```

因此, 在 MATLAB 中求解一个系统用左除比用逆和乘法所需的运算次数要少。命令 `flops` 的定义参见第 2.5 节。

### 3.4 转置和共轭

一个重要的运算是转置和共轭转置, 它在 MATLAB 中用撇 ' 表示。在课本中, 这种运算经常用 \* 和  $\mu$  表示

如果  $A$  是一个实数, 那么它被转置时, 第 1 行变成第 1 列, 第 2 行变成第 2 列, 依此类推, 一个  $m \times n$  矩阵变为一个  $n \times m$  矩阵。如果矩阵是方阵, 那么这个矩阵在主对角线反映出来。

如果矩阵  $A$  的元素  $a_{ij}$  是复数, 那么所有元素也是共轭的。矩阵  $A'$  在项  $(i, j)$  上含有  $\overline{a_{ji}}$ 。

如果仅希望转置, 在撇号之前输入一点 .',  $A.'$  表示转置, 其结果与 `conj(A')` 相同。如果  $A$  是实数, 那么  $A'$  与  $A.'$  相同。

#### 例 3.6

假设  $A$  和  $b$  与例 3.5(b) 相同。

`Transp=A', Transpb=b'`, 得到:

```
Transp =
     1     1     0
```

```

3      2      5
5      4      1

```

```

Transpb =
22      17      13

```

### 3.5 元素操作算术运算

算术运算也可以元素与元素逐次进行。矩阵的维数要相同，可以是多维的。如果运算是由一点进行的，那么这个运算实行的是元素方式。

对于加法和减法，数组运算和矩阵运算没有差别。数组运算符是：

```

+      -      .*      ./      .\      .^

```

注意，`.`并没有列出，这个点在那种情况下具有不同的含义。这个操作符仅给出转置，与相反，给出了共轭转置，详见第3.4节。

#### 例3.7

假设定义如下矩阵：

$$\mathbf{A} = \begin{pmatrix} 1 & 2 \\ -1 & 5 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 7 & 2 \\ 1 & 0 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} 1+2i & 5-2i \\ 3+i & 1+3i \end{pmatrix}$$

(a)  $\mathbf{A}.*\mathbf{B}$ 得：

```

ans =
7      4
-1     0

```

(b)  $\mathbf{B}./\mathbf{A}$ 得：

```

ans =
7      1
-1     0

```

(c)  $\mathbf{B}.^2$ 得：

```

ans =
49     4
1      0

```

(d)  $\mathbf{A}.^B$ 得：

```

ans =
1      4
-1     1

```

(e) 基数是标量，而指数是一个矩阵， $2.^{[1 \ 2 \ 3 \ 4]}$ 得：

```

ans=
2      4      8     16

```

(f)  $\mathbf{C}.'$ 得：

```

ans =
1.0000 + 2.0000i    3.0000 + 1.0000i
5.0000 - 2.0000i    1.0000 + 3.0000i

```

可参见例13.1，它也使用了数组运算。

### 3.6 元素操作函数

在MATLAB中预定义的数学标准函数(见第2.4节)是基于矩阵对元素的运算。如果  $f$  是这样 一个函数， $A$  是带元素  $a_{ij}$  的一个矩阵，那么  $f(A)_{ij}=f(a_{ij})$ 。如果元素是复数，那么根据这个函数 产生的矩阵也可以是复数，矩阵的维数没有改变。

例3.8

令  $A$ 、 $B$  和  $C$  为：

$$A = \begin{pmatrix} 0 & -3 \\ 5 & 1 \\ 4 & -6 \end{pmatrix} \quad B = \begin{pmatrix} \pi & 0 \\ \pi/2 & \pi/4 \end{pmatrix} \quad C = \begin{pmatrix} 1+i & -\pi i \\ 0 & 2-i \end{pmatrix}$$

(a) `abs(A)` 得：

```
ans =
     0     3
     5     1
     4     6
```

(b) `cos(B)` 得：

```
ans =
-1.0000    1.0000
 0.0000    0.7071
```

(c) `sin(abs(C))` 得：

```
ans =
 0.9878    0.0000
     0    0.7867
```

数组运算符和数组函数在 MATLAB 中十分有用，用户可以定义自己的数组函数，并把它 们存放在 M 文件中，可参见第 2.9 节。

例3.9

函数 `sincos(x)=sin(x)cos(x)` 是一个非标准 MATLAB 函数，可是，你可以定义自己的函数 `sincos` 并存放在文件 `sincos.m` 中。

```
function y=sincos(x)
y=sin(x).*cos(x);
```

可如下调用 `sincos`:

```
y1 = sincos(pi), y2 = sincos([0 pi/4 pi/2])
```

```
y1 =
-1.2246e-16
```

```
y2 =
     0    0.5000    0.0000
```

看到，应该为0值的 $y_1$ 是一个十分小的数。事实上， $eps$ 是较大的。如果用一个向量作为一个自变量来调用  $sincos$ ，因为  $\sin$  和  $\cos$  返回向量，所以其结果是一个向量。当绘制函数图形时，这是十分有用的。

M文件的应用可参见第12章和第13章。

### 3.7 矩阵的乘方与函数

对于二维方阵， $A$ 的 $p$ 次乘方可以用  $A^p$  实现。如果 $p$ 是一个正整数，那么这个幂可以由许多矩阵乘法运算定义。对于  $p=0$ ，得到与  $A$  维数相同的同一个矩阵；当  $p<0$  时，如果  $A^{-1}$  存在，可定义  $A^p$ ，它是与  $\text{inv}(A)^{-p}$  相同。

象  $\exp(A)$  和  $\text{sqrt}(A)$  那样的 MATLAB 表达式可视为数组运算（参见第3.6节），即它们是对  $A$  中元素逐个运算。

MATLAB 也能处理方阵函数。例如  $A^{1/2}$  ( $A$  的平方根) 或  $e^A$ 。举例如下：

$$e^A = I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \cdots$$

#### 命令集27 矩阵函数

<code>expm(A)</code>	使用Pade近似法计算 $e^A$ ，这是一个内部函数。
<code>expml(A)</code>	使用一个M文件和与内部函数相同的算法计算 $e^A$ 。
<code>expm2(A)</code>	使用泰勒级数计算 $e^A$ 。
<code>expm3(A)</code>	使用特征值和特征向量计算 $e^A$ 。
<code>logm(A)</code>	计算 $A$ 的对数。
<code>sqrtm(A)</code>	计算 $A^{1/2}$ 。当 $A$ 是对称正定阵时，平方根是唯一的。
<code>funm(A, fcn)</code>	计算由字符串 <code>fcn</code> 指定的 $A$ 的矩阵函数，参见第 5.1.4 节。字符串 <code>fcn</code> 可以是任意的基本函数，如 <code>sin</code> 、 <code>cos</code> 等等，参见第2.4节。例如， <code>expm(A)=funm(A,'exp')</code> 。
<code>[F, E]=funm(A, fcn)</code>	计算如上矩阵函数，但返回结果矩阵 $F$ 和剩余近似值矩阵 $E$ 。
<code>polyvalm(p,A)</code>	估算矩阵 $A$ 的一个多项式。向量 $p$ 含有多项式的系数。详见第10.1节。

很重要的一点是要区别 `expm` 和 `exp`、`logm` 和 `log` 等等。

#### 例3.10

假设：

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}$$

比较 `exp` 和 `expm`：

```
Elementwise=exp(A), Operatorwise=expm(A)
```

得：

```
Elementwise =
    2.7183    1.0000
    1.0000    7.3891
```



```
Operatorwise =
    2.7183    0
    0    7.3891
```

### 3.8 关系运算符

MATLAB有用于比较矩阵的六个关系运算符，也可以对矩阵与一个标量进行比较，即矩阵中的每个元素与标量进行比较。

关系运算符如下：

<	小于
<=	小于等于
>	大于
>=	大于等于
==	等于
~=	不等于

关系运算符比较对应的元素，产生一个仅包含 1 和 0 的具有相同维数的矩阵。其元素是：

1 比较结果是真  
0 比较结果是假

在一个表达式中，算术运算符优先级最高，其次是关系运算符，最低级别是逻辑运算符。圆括号可以改变其顺序。

#### 例3.11

(a) 对预定义变量pi的值和通过命令rat获得的pi的近似值进行比较。

```
[t, n]=rat(pi), piapprox=t/n;
format long, piapprox, pi, piapprox==pi
```

得：

```
t =
    355

n =
    113

piapprox =
    3.14159292035398

ans =
    3.14159265358979

ans =
    0
```

(b) 假设：

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 4 \\ 1 & 1 & 1 \\ 2 & 3 & 1 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{pmatrix}$$

A中的元素有大于B中对应的元素吗？

```
Greater=A>B
```

得：

```
Greater =
    0     0     1
    0     0     0
    0     1     0
```

即A中的项(1, 3)和(3, 2)的值大于B中对应项的值。

(c) 令A如例(b)中假设，A中的元素有大于1的吗？

```
GreaterThanOne=A>1
```

得：

```
GreaterThanOne =
    0     1     1
    0     0     0
    1     1     0
```

### 3.9 逻辑运算符

在MATLAB中有四种逻辑运算符：

```
&      与
|      或
~      非
xor     异或
```

逻辑运算符的运算优先级最低。在一个表达式中，关系运算符和算术运算符的运算级别要高于逻辑运算符。

**xor**和**or**之间的差别在于：表达式中至少有一个是真，那么 **or**是真；**xor**是表达式中有一个是真但不能两者均为真时才为真。

运算符**&**和**|**比较两个相同维数的矩阵，如同前一节一样，它也能使一个标量与一个矩阵进行比较。逻辑运算符是按元素比较的。零元素表示逻辑值假，任何其他值的元素表示逻辑值真。其结果是一个包含1和0的矩阵。

#### 命令集28 逻辑运算符

<b>A&amp;B</b>	返回一个与A和B相同维数的矩阵。在这个矩阵中，A和B对应元素都为非零时，则对应项为1；有一个为零的项则为0。
<b>A B</b>	返回一个与A和B相同维数的矩阵。在这个矩阵中，A和B对应元素只要有一个为非零，则对应项为1；两个矩阵均为零时，则为0。
<b>A</b>	返回一个与A和B相同维数的矩阵。在这个矩阵中，A是零时，则对应项为1；A是非零时，则对应项为0。
<b>xor(A, B)</b>	返回一个与A和B相同维数的矩阵。在这个矩阵中，如果A和B均为零或均为非零时，则对应项为0；如果A或B是非零但不是两者同时为非零时，则对应项为1。

## 3.10 逻辑函数

在MATLAB中有几个逻辑函数。在以下定义的函数中，假设  $A$  是一个  $m \times n$  的矩阵， $x$  是一个向量。

在一些计算中，很重要的一点是要在给定的矩阵中以一定的特征定位。例如，在部分选主元的高斯消去法中，必须在工作列中寻找最大的项。MATLAB命令 `find` 可以用于这种情况。

**命令集29** 查找非零元素

<code>find(x)</code>	返回一个 $x$ 中包含非零元素的下标的向量。如果所有的元素都是零，那么返回一个空矩阵，即 <code>[]</code> 。
<code>find(A)</code>	返回一个长的列向量，表示 $A$ 中包含非零元素的下标向量。下述命令更可取。
<code>[u, v]=find(A)</code>	返回向量 $u$ 和 $v$ ，它们包含 $A$ 中的非零元素的下标，即 $A$ 中元素 $(u_k, v_k)$ 为非零。
<code>[u,v,b]=find(A)</code>	返回包含 $A$ 中非零元素的下标向量 $u$ 和 $v$ 以及一个包含对应非零元素的向量。 $A$ 中元素 $(u_k, v_k)$ 为非零并且能在 $b_k$ 中找到。

## 例3.12

假设  $x$  和  $A$  是：

$$x = (3 \quad -4 \quad 0 \quad 6.1 \quad 0) \quad A = \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix} \quad y = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 4 \end{pmatrix}$$

(a) `ind=find(x)`，`indcol=find(A)`

```
ind =
     1     2     4
```

```
indcol =
     1
     4
```

即向量  $x$  中元素 1、2 和 4 是非零值。要获得 `indcol`，也可以输入：

```
find(y)
```

(b) 命令 `find` 可以与关系运算符一起使用，这样使命令更有用。例如 `index=find(x>0.5)` 返回：

```
index=
     1     4
```

如果输入 `greaterThan=x(index)`，得到：

```
greaterThan=
     3.0000     6.1000
```

这就是用向量 `index` 寻找  $x$  中所有大于 0.5 的元素。

如果仅想知道  $x$  中有多少大于 0.5 的元素时，可以输入：`length(find(x>0.5))` 针对上题数据，得到：

```
ans=
```

```
2
```

(c) 要获得A中所有非零元素的索引, 输入:

```
[index1, index2]=find(A)
```

得:

```
index1 =
```

```
1
```

```
2
```

```
index2 =
```

```
1
```

```
2
```

即元素(1, 1)和(2, 2)是非零值。

MATLAB有any和all两个函数, 用于测试矩阵和向量的逻辑条件。其结果是逻辑值, 0或1, 真或假。它们在if语句中有特殊的用途, 详见第12.1节。

### 命令集30 逻辑函数(一)

any(x)	如果x中的有一个元素为非零值, 那么返回1; 否则, 返回0。
any(A)	对A进行列运算, 根据相应列是否包含非零元素, 返回一个带1和0的行向量。
all(x)	如果所有的元素都是非零值, 返回1; 否则, 返回0。
all(A)	对A进行列操作, 根据相应列是否所有元素都为非零值, 返回带1和0的一个行向量。

如果这两个函数之一对一个矩阵进行两次操作, 例如any(any(A))和all(all(A))则返回一个标量1或0。

#### 例3.13

(a) 如果all(x<=5)返回1, 则实向量x中所有的元素都小于或等于5。如果返回0, 则至少有一个元素大于5。如果all(all(A<=5))返回1, 则一个矩阵A的所有元素小于或等于5。

(b) 对一个实方阵A, 如果all(all(A==A'))返回1, 则A是对称的。

(c) 如果any(any(tril(A, -1)))返回0, 则方阵A是上三角阵。否则, 在A中的下三角阵中至少有一个非零元素。一个等价的命令是all(all(A==triu(A))), 如果A是上三角阵, 它就返回1。

也有下列逻辑函数:

### 命令集31 逻辑函数(二)

isnan(A)	返回一个维数与A相同的矩阵, 在这个矩阵中, 对应A中有'NaN'处为1, 其他地方为0。
----------	---

<code>isinf(A)</code>	返回一个维数与 A 相同的矩阵，在这个矩阵中，对应 A 中有 'inf' 处为 1，其他地方为 0。
<code>isempty(A)</code>	如果 A 是一个空矩阵，返回 1；否则返回 0。
<code>isequal(A, B)</code>	如果 A 和 B 是相同的，即有相同的维数和相同的内容，则返回 1。
<code>isreal(A)</code>	如果 A 是一个不带虚部的实矩阵，则返回 1；否则，返回零。
<code>isfinite(A)</code>	返回一个与 A 维数相同的矩阵。在这个矩阵中，A 中元素是有限的，则对应元素为 1；否则，为零。

也有针对字符串和其他数据类型的逻辑函数，详见第 5 章。