

MATLAB 在数学建模中有广泛的应用。本章主要介绍 MATLAB 建模的基础知识，包括数组和变量、矩阵、符号运算、关系运算和逻辑运算等。

学习目标：

- 熟悉 MATLAB 中的数组和变量、矩阵
- 熟悉 MATLAB 的符号运算、关系运算和逻辑运算

2.1 数组

MATLAB 中数组可以说无处不在，任何变量在 MATLAB 中都是以数组形式存储和运算的。

2.1.1 数组的定义

数组指相同数据类型的元素按一定顺序排列的集合，就是把有限个类型相同的变量用一个名字命名，然后用编号区分变量的集合，这个名字称为数组名，编号称为下标。组成数组的各个变量称为数组的分量，也称为数组的元素，有时也称为下标变量。

按照数组元素个数和排列方式，MATLAB 中的数组可以分为以下几种。

- (1) 没有元素的空数组 (empty array)。
- (2) 只有一个元素的标量 (scalar)，它实际上是一行一列的数组。
- (3) 只有一行或者一列元素的向量 (vector)，分别叫作行向量和列向量，也统称为一维数组。
- (4) 普通的具有多行多列元素的二维数组。
- (5) 超过二维的多维数组 (具有行、列和页等多个维度)。

按照数组的存储方式，MATLAB 中的数组可以分为普通数组和稀疏数组 (常称为稀疏矩阵)。稀疏矩阵适用于那些大部分元素为 0、只有少部分非 0 元素的数组的存储，主要是为了提高数据存储和运算的效率。

2.1.2 数组的创建

MATLAB 中一般使用方括号 ([])、逗号 (,) 或空格，以及分号 (;) 来创建数组，方括号中给出数组的所有元素，同一行中的元素间用逗号或空格分隔，不同行之间用分号分隔。

1. 创建空数组

空数组是 MATLAB 中特殊的数组。它不含有任何元素。空数组可以用于数组声明、数组清空及各种特殊的运算场合（如特殊的逻辑运算等）。

创建空数组很简单，只需要把变量赋值为空的方括号即可。

【例 2-1】创建空数组 A 。

解：在命令行窗口中输入

```
>> A=[]  
A =  
    []
```

2. 创建一维数组

一维数组包括行向量和列向量，是所有元素排列在一行或一列中的数组。实际上，一维数组可以看作二维数组在某一方向（行或列）尺寸退化为 1 的特殊形式。

创建一维行向量，只需要把所有用空格或逗号分隔的元素用方括号括起来即可；而创建一维列向量，则需要在方括号括起来的元素之间用分号分隔。不过，更常用的办法是用转置运算符（'），把行向量转置为列向量。

【例 2-2】创建行向量和列向量。

解：在命令行窗口中输入

```
>> clear                                %从当前工作区中删除所有变量，并将它们从系统内存中释放  
>> A=[1 2 3]  
A =  
     1     2     3  
>> B=[1; 2; 3]  
B =  
     1  
     2  
     3  
>> C=A'  
C =  
     1  
     2  
     3
```

当要创建的一维数组是某个等差数列时，可以通过冒号来创建，也可以通过 MATLAB 提供的 `linspace` 函数来创建，格式如下：

<code>Var=start_var:step:stop_var</code>	%创建一个一维行向量 Var，它的第一个元素是 start_var，然后 %依次递增（step 为正）或递减（step 为负），直到向量中的最 %后一个元素与 stop_var 差的绝对值小于或等于 step 的绝对 %值为止，不指定 step 时，step 默认为 1
<code>Var=linspace(start_var,stop_var,n)</code>	%创建一个一维行向量 Var，第一个元素是 start_var，最后一 %个元素是 stop_var，形成总共 n 个元素的等差数列。不指定 %n 时，n 默认为 100

要注意，利用冒号创建等差的一维数组时，可能取不到 `stop_var` 元素。

【例 2-3】创建一维等差数组。**解：**在命令行窗口输入

```
>> clear
>> A=1:4
A =
     1     2     3     4
>> B=1:2:4
B =
     1     3
>> C=linspace(1,2,4)
C =
 1.0000  1.3333  1.6667  2.0000
```

类似 linspace 函数，MATLAB 中还有创建等比一维数组的 logspace 函数：

```
Var=logspace(start_var,stop_var,n) %产生从 start_var 到 stop_var 的共包含 n 个元素的等比一维
%数组 Var，不指定 n 时，n 默认为 50
```

【例 2-4】创建一维等比数组。**解：**在命令行窗口输入

```
>> clear
>> A=logspace(0,log10(32),6)
A =
 1.0000  2.0000  4.0000  8.0000 16.0000 32.0000
```

综上，创建一维数组可能用到方括号、逗号或空格、分号、冒号、函数 linspace 和 logspace 及转置符号等，读者需要完全掌握。

3. 创建二维数组

常规创建二维数组的方法实际上和创建一维数组的方法类似，就是综合运用方括号、逗号、空格，以及分号等。

方括号把所有元素括起来，不同行元素之间用分号间隔，同一行元素之间用逗号或者空格间隔，按照逐行排列的方式顺序书写每个元素。

当然，在创建每行或每列元素时，可以利用冒号和函数的方法，只是要特别注意创建二维数组时，要保证每行（或每列）具有相同数目的元素。

【例 2-5】创建二维数组。**解：**在命令行窗口中输入

```
>> clear
>> A=[1 2 3; 2 5 6; 1 4 5]
A =
     1     2     3
     2     5     6
     1     4     5
>> B=[1:5;linspace(3,10,5);3 5 2 6 4]
B =
 1.0000  2.0000  3.0000  4.0000  5.0000
```

```

    3.0000    4.7500    6.5000    8.2500   10.0000
    3.0000    5.0000    2.0000    6.0000    4.0000
>> C=[1:3];[linspace(2,3,3)];[3 5 6]]
C =
    1.0000    2.0000    3.0000
    2.0000    2.5000    3.0000
    3.0000    5.0000    6.0000

```

提示：通过函数拼接一维数组也可以创建二维数组，或者利用 MATLAB 内部函数直接创建特殊的二维数组，本章后续内容会逐步介绍。

4. 创建三维数组

1) 使用下标创建三维数组

在 MATLAB 中，习惯将二维数组的第一维称为“行”，第二维称为“列”，而对于三维数组，其第三维则习惯性地称为“页”。

在 MATLAB 中，将三维或者三维以上的数组统称为多维数组，下面以三维数组为例来介绍如何创建高维数组。

【例 2-6】使用下标引用的方法创建三维数组。

解：在命令行窗口中依次输入

```

>> clear
>> A(3,3,2)=1; %创建一个 3 行 3 列 2 页的三维数组 A，其中第 2 页第三行第三列元素为 1，其余元素为 0
>> for i=1:3
    for j=1:3
        for k=1:2
            A(i,j,k)=i+j+k;
        end
    end
end
>> A(:, :, 1) %第 1 页元素
ans =
     3     4     5
     4     5     6
     5     6     7
>> A(:, :, 2) %第 2 页元素
ans =
     4     5     6
     5     6     7
     6     7     8

```

继续在命令行窗口中输入

```
>> B(3,4,:)=2:3; %创建一个 3 行 4 列 2 页的三维数组 B
```

在命令行窗口输入变量名，查看变量值

```

>> B(:, :, 1)
ans =
     0     0     0     0

```

```

    0    0    0    0
    0    0    0    2
>> B(:,:,2)
ans =
    0    0    0    0
    0    0    0    0
    0    0    0    3

```

从上面结果可以看出，当使用下标的方法来创建高维数组时，需要使用各自对应的维度数值，没有指定的数值则在默认情况下为 0。

2) 使用低维数组创建三维数组

由于三维数组中“包含”二维数组，因此可以通过二维数组来创建各种三维数组。

【例 2-7】使用低维数组来创建高维数组。

解：在命令行窗口中依次输入

```

>> clear
>> A=[1,2,3; 4,5,6; 7,8,9];           %创建二维数组
>> B(:,:,1)=A;
>> B(:,:,2)=2*A;
>> B(:,:,3)=3*A;

```

在命令行窗口输入变量名，查看变量值

```

>> B
B(:,:,1) =
     1     2     3
     4     5     6
     7     8     9
B(:,:,2) =
     2     4     6
     8    10    12
    14    16    18
B(:,:,3) =
     3     6     9
    12    15    18
    21    24    27

```

3) 使用创建函数创建三维数组

下面介绍利用 MATLAB 的创建函数 `cat`、`repmat` 和 `reshape` 创建三维数组。其中，`cat` 用于连接数组；`repmat` 用于复制并堆砌数组；`reshape` 用于修改数组的大小，如将二维数组修改为三维数组。它们的调用格式为

```

B=cat(dim,A1,A2,A3...)    %dim 表示创建数组的维度，A1,A2,A3 表示各维度上的数组
B=repmat(A,[m n p...])    %A 表示复制的数组模块，[m n p...]表示该数组模块在各个维度上的复制个数
B=reshape(A,[m n p ...])  %A 是待重组的矩阵，[m n p ...]表示数组各维的维度

```

【例 2-8】使用函数创建高维数组。

解：使用 `cat` 创建高维数组。在命令行窗口中依次输入

```

>> clear
>> A=[1,2,3,;4,5,6;7,8,9];
>> B=cat(3, A, 2*A, 3*A);

```

在命令行窗口输入变量名称, 查看变量值

```
>> B
B(:,:,1) =
     1     2     3
     4     5     6
     7     8     9
B(:,:,2) =
     2     4     6
     8    10    12
    14    16    18
B(:,:,3) =
     3     6     9
    12    15    18
    21    24    27
```

使用 `repmat` 来创建数组。在命令行窗口中依次输入

```
>> C=[1,2,3; 4,5,6; 7,8,9];
>> D1=repmat(C, 2, 3);
>> D2=repmat(C, [1 2 3]);
```

在命令行窗口输入变量名, 查看变量值

```
>> D1
D1 =
     1     2     3     1     2     3     1     2     3
     4     5     6     4     5     6     4     5     6
     7     8     9     7     8     9     7     8     9
     1     2     3     1     2     3     1     2     3
     4     5     6     4     5     6     4     5     6
     7     8     9     7     8     9     7     8     9
>> D2
D2(:,:,1) =
     1     2     3     1     2     3
     4     5     6     4     5     6
     7     8     9     7     8     9
D2(:,:,2) =
     1     2     3     1     2     3
     4     5     6     4     5     6
     7     8     9     7     8     9
D2(:,:,3) =
     1     2     3     1     2     3
     4     5     6     4     5     6
     7     8     9     7     8     9
```

使用 `reshape` 来创建数组。在命令行窗口中依次输入

```
>> E=[1,2,3,4; 5,6,7,8; 9,10,11,12];
>> F1=reshape(E,2,2,3);
>> F2=reshape(E,2,3,2);
>> F3=reshape(E,3,2,2);
```

在命令行窗口输入变量名, 查看变量值

```

>> F1
F1(:,:,1) =
     1     9
     5     2
F1(:,:,2) =
     6     3
    10     7
F1(:,:,3) =
    11     8
     4    12
>> F2
F2(:,:,1) =
     1     9     6
     5     2    10
F2(:,:,2) =
     3    11     8
     7     4    12
>> F3
F3(:,:,1) =
     1     2
     5     6
     9    10
F3(:,:,2) =
     3     4
     7     8
    11    12

```

5. 创建低维标准数组

MATLAB 还提供多种函数来生成某些标准数组，直接使用这些函数可以创建一些特殊的数组。

【例 2-9】使用标准数组函数创建低维数组。

解：在命令行窗口中依次输入

```

>> clear
>> A=zeros(3,2);           %创建全 0 数组
>> B=ones(2,4);            %创建全 1 数组
>> C=eye(4);               %创建单位矩阵
>> D=magic(5);             %创建幻方矩阵
>> randn('state',0);       %设置伪随机数发生器状态为初始状态
>> E=randn(1,2);           %产生符合正态分布的随机数
>> F=gallery(5);           %生成测试矩阵

```

在命令行窗口输入变量名，查看变量值

```

>> A
A =
     0     0
     0     0
     0     0
>> B
B =

```

```

    1    1    1    1
    1    1    1    1
>> C
C =
    1    0    0    0
    0    1    0    0
    0    0    1    0
    0    0    0    1
>> D
D =
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
>> E
E =
   -0.4326   -1.6656
>> F
F =
    -9     11    -21     63   -252
    70    -69    141   -421   1684
   -575    575  -1149   3451  -13801
   3891  -3891   7782  -23345   93365
   1024  -1024   2048   -6144  24572

```

并不是所有的标准函数都可以创建多种矩阵,例如 `eye`、`magic` 等就不能创建高维数组。同时,每个标准函数的参数都有各自的要求,例如 `gallery` 中只能选择 3 或者 5。

6. 创建高维标准数组

下面介绍如何使用标准数组函数来创建高维标准数组。

【例 2-10】使用标准数组函数创建高维数组。

解:在命令行窗口中输入

```

>> clear
>> rand('state',1111); %设置随机数发生器初始状态
>> D1=randn(2,3,5);
>> D2=ones(2,3,4);

```

在命令行窗口输入变量名,查看变量值

```

>> D1
D1(:,:,1) =
    0.8156    1.2902    1.1908
    0.7119    0.6686   -1.2025
D1(:,:,2) =
   -0.0198   -1.6041   -1.0565
   -0.1567    0.2573    1.4151
D1(:,:,3) =
   -0.8051    0.2193   -2.1707

```

```
0.5287    -0.9219    -0.0592
D1(:, :, 4) =
   -1.0106    0.5077    0.5913
    0.6145    1.6924   -0.6436
D1(:, :, 5) =
    0.3803   -0.0195    0.0000
   -1.0091   -0.0482   -0.3179
>> D2
D2(:, :, 1) =
     1     1     1
     1     1     1
D2(:, :, 2) =
     1     1     1
     1     1     1
D2(:, :, 3) =
     1     1     1
     1     1     1
D2(:, :, 4) =
     1     1     1
     1     1     1
```

限于篇幅，不再详细讲解各函数的参数和使用方法，请读者自行阅读相关帮助文件。

2.1.3 多维数组及其操作

MATLAB 中把超过二维的数组称为多维数组，多维数组实际上是二维数组的扩展。下面讲述 MATLAB 中多维数组的操作。

1. 多维数组的属性

MATLAB 中提供了多个函数，可以获得多维数组的尺寸、维度、占用内存和数据类型等多种属性，具体如表 2-1 所示。

表 2-1 MATLAB 中获取多维数组属性的函数

数 组 属 性	函 数 用 法	函 数 功 能
尺寸	size(A)	按照“行—列—页”的顺序，返回数组A各个维上的大小
维度	ndims(A)	返回数组A具有的维度值
内存占用/数据类型等	whos	返回当前工作区中各个变量的详细信息

【例 2-11】通过 MATLAB 函数获取多维数组的属性。

解：在命令行窗口输入

```
>> clear
>> A=cat(4,[9 2;6 5], [7 1;8 4]);
>> size(A)                                %获取数组 A 的尺寸属性
ans =
     2     2     1     2
>> ndims(A)                              %获取数组 A 的维度属性
```

```
ans =
     4
>> whos
      Name      Size      Bytes  Class  Attributes
      A         4-D         64     double
      ans        1x1          8     double
```

2. 多维数组的操作

MATLAB 中提供了可以对多维数组进行索引、重排和计算的函数。

1) 多维数组的索引

MATLAB 中索引多维数组的方法包括多下标索引和单下标索引。

对于 n 维数组, 可以用 n 个下标索引访问到一个特定位置的元素, 而用数组或者冒号来代表其中某一维, 则可以访问指定位置的多个元素。单下标索引方法则只通过一个下标来定位多维数组中某个元素的位置。

MATLAB 中按照“行—列—页”优先级逐渐升高的顺序把多维数组的所有元素线性存储, 这样就可以知道一个特定的单下标对应的多维数组下标的位置了。

【例 2-12】多维数组的索引访问, 其中 A 是一个随机生成的 $4 \times 5 \times 3$ 的多维数组。

解: 在命令行窗口输入

```
>> clear
>> A=randn(4,5,3)
A(:,:,1) =
   -1.3617    0.5528    0.6601   -0.3031    1.5270
    0.4550    1.0391   -0.0679    0.0230    0.4669
   -0.8487   -1.1176   -0.1952    0.0513   -0.2097
   -0.3349    1.2607   -0.2176    0.8261    0.6252
A(:,:,2) =
    0.1832    0.1352   -0.1623   -0.8757   -0.1922
   -1.0298    0.5152   -0.1461   -0.4838   -0.2741
    0.9492    0.2614   -0.5320   -0.7120    1.5301
    0.3071   -0.9415    1.6821   -1.1742   -0.2490
A(:,:,3) =
   -1.0642   -1.5062   -0.2612   -0.9480    0.0125
    1.6035   -0.4446    0.4434   -0.7411   -3.0292
    1.2347   -0.1559    0.3919   -0.5078   -0.4570
   -0.2296    0.2761   -1.2507   -0.3206    1.2424
>> A(3,2,2) %访问 A 的第 3 行第 2 列第 2 页的元素
ans =
    0.2614
>> A(27) %访问 A 的第 27 个元素 (即第 3 行第 2 列第 2 页的元素)
ans =
    0.2614
```

$A(27)$ 是通过单下标索引来访问多维数组 A 的元素。多维数组 A 有 3 页, 每页有 $4 \times 5 = 20$ 个元素, 所以第 27 个元素在第二页上, 而每页列方向上有 4 个元素, 根据“行—列—页”优先原则, 第 27 个元素代表的就是第 2 页上第 2 列第 3 行的元素, 即 $A(27)$ 相当于 $A(3,2,2)$ 。

2) 多维数组的维度操作

多维数组的维度操作包括对多维数组的形状的重排和维度的重新排序。

reshape 函数可以改变多维数组的形状,但操作前后 MATLAB 按照“行—列—页”优先级对多维数组进行线性存储的方式不变,许多多维数组在某一维度上只有一个元素,可以利用 squeeze 函数来消除这种单值维度。

reshape 函数的调用格式如下:

```
B = reshape(A,sz)    %使用大小向量 sz 重构 A 以定义 size(B)。例如, reshape(A,[2,3]) 将 A 重构为
                    %一个 2×3 矩阵。sz 必须至少包含 2 个元素, prod(sz) 必须与 numel(A) 相同
B = reshape(A,sz1,...,szN) %将 A 重构为一个 sz1×...×szN 数组,其中 sz1,...,szN 指示每个维
                    %度的大小。可以指定 [] 的单个维度大小,以便自动计算维度大小,以使 B
                    %中的元素数与 A 中的元素数相匹配。例如,如果 A 是一个 10×10 矩阵,
                    %则 reshape(A,2,2,[]) 将 A 的 100 个元素重构为一个 2×2×25 数组
```

【例 2-13】利用 reshape 函数改变多维数组的形状。

解: 在命令行窗口中输入

```
>> clear
>> A = magic(4)
A =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
>> B = reshape(A,[],8)
B =
    16     9     2     7     3     6    13    12
     5     4    11    14    10    15     8     1
>> C = reshape(A,8,[])
C =
    16     3
     5    10
     9     6
     4    15
     2    13
    11     8
     7    12
    14     1
```

按照指定的顺序重新定义多维数组的维度顺序时可以采用 permute 函数。重新定义后的多维数组把原来在某一维度上的所有元素移动到新的维度上,和 reshape 不同的是,permute 会改变多维数组线性存储的位置。

另外,ipermute 可以被看作是 permute 的逆函数,当 B=permute(A,dims)时,ipermute(B,dims)刚好返回多维数组 A。

【例 2-14】对多维数组维度的重新排序。

解: 在命令行窗口中输入

```
>> clear
>> A=randn(3,3,2)
A(:,:,1) =
    0.4227   -1.2128    0.3271
   -1.6702    0.0662    1.0826
```

```
0.4716    0.6524    1.0061
A(:,:,2) =
   -0.6509   -1.3218   -0.0549
    0.2571    0.9248    0.9111
   -0.9444    0.0000    0.5946
>> B=permute(A,[3 1 2])
B(:,:,1) =
    0.4227   -1.6702    0.4716
   -0.6509    0.2571   -0.9444
B(:,:,2) =
   -1.2128    0.0662    0.6524
   -1.3218    0.9248    0.0000
B(:,:,3) =
    0.3271    1.0826    1.0061
   -0.0549    0.9111    0.5946
>> ipermute(B,[3 1 2])
ans(:,:,1) =
    0.4227   -1.2128    0.3271
   -1.6702    0.0662    1.0826
    0.4716    0.6524    1.0061
ans(:,:,2) =
   -0.6509   -1.3218   -0.0549
    0.2571    0.9248    0.9111
   -0.9444    0.0000    0.5946
```

3) 多维数组参与数学计算

多维数组参与数学计算时,既可以针对某一维度的向量,也可以针对单个元素,或者针对某一特定页面上的二维数组。

- (1) `sum`、`mean` 等函数可以对多维数组中第 1 个不为 1 的维度上的向量进行计算。
- (2) `sin`、`cos` 等函数则对多维数组中的每个单独元素进行计算。
- (3) `eig` 等针对二维数组的运算函数则需要用指定的页面上的二维数组作为输入函数。

【例 2-15】多维数组参与数学运算。

解: 在命令行窗口中输入

```
>> clear
>> A=randn(2,5,2)
A(:,:,1) =
    0.3502    0.9298   -0.6904    1.1921   -0.0245
    1.2503    0.2398   -0.6516   -1.6118   -1.9488
A(:,:,2) =
    1.0205    0.0012   -2.4863   -2.1924    0.0799
    0.8617   -0.0708    0.5812   -2.3193   -0.9485
>> sum(A)
ans(:,:,1) =
    1.6005    1.1696   -1.3419   -0.4197   -1.9733
ans(:,:,2) =
    1.8822   -0.0697   -1.9051   -4.5117   -0.8685
>> sin(A)
ans(:,:,1) =
```

```

    0.3431    0.8015   -0.6368    0.9291   -0.0245
    0.9491    0.2375   -0.6064   -0.9992   -0.9294
ans(:,:,2) =
    0.8524    0.0012   -0.6094   -0.8129    0.0798
    0.7590   -0.0708    0.5490   -0.7327   -0.8125
>> eig(A(:,[1 2],1))
ans =
    1.3746
   -0.7846

```

2.2 矩阵

在数学中,矩阵是指排列的二维数据表格,它来自于方程组的系数及常数所构成的方阵,矩阵的一个重要用途是解线性方程组。线性方程组中未知量的系数可以排成一个矩阵,加上常数项,则称为增广矩阵。矩阵的另一个重要用途是表示线性变换。

MATLAB 的强大功能之一体现在可以直接处理矩阵,而其首要任务就是输入待处理的矩阵。下面介绍几种基本的矩阵生成方式。

2.2.1 实数矩阵输入

任何矩阵(向量)都可以直接按行输入每个元素:同一行中的元素用逗号(,)或者用空格符来分隔,且空格个数不限;不同的行用分号(;)分隔。所有元素处于一对方括号([])内;当矩阵是多维(三维以上)且方括号内的元素是维数较低的矩阵时,可以用多重方括号。

【例 2-16】实数矩阵的输入。

解:在命令行窗口中输入

```

>> clear
>> A = [11 12 1 3 5 7 9 10]
A =
    11    12     1     3     5     7     9    10
>> B = [2.32 3.43;4.37 5.98]
B =
    2.3200    3.4300
    4.3700    5.9800
>> C = [1 2 3 4 5]
C =
     1     2     3     4     5
>> D = [1 2 3;2 3 4;3 4 5]
D =
     1     2     3
     2     3     4
     3     4     5
>> E = [ ]
E =
    [ ]
%生成一个空矩阵

```

2.2.2 复数矩阵输入

复数矩阵有单个元素生成和整体生成两种生成方式。

【例 2-17】复数矩阵的输入。

解：在命令行窗口中输入

```
>> clear
%%%单个元素生成%%%
>> a=2.7
a =
    2.7000
>> b=13/25
b =
    0.5200
>> c=[1,3*a+i*b,b*sqrt(a); sin(pi/6),3*a+b,3]
c =
    1.0000 + 0.0000i    8.1000 + 0.5200i    0.8544 + 0.0000i
    0.5000 + 0.0000i    8.6200 + 0.0000i    3.0000 + 0.0000i

%%%整体生成%%%
>> A=[1 2 3;4 5 6]
A =
     1     2     3
     4     5     6
>> B=[11 12 13;14 15 16]
B =
    11    12    13
    14    15    16
>> C=A+i*B
C =
    1.0000 +11.0000i    2.0000 +12.0000i    3.0000 +13.0000i
    4.0000 +14.0000i    5.0000 +15.0000i    6.0000 +16.0000i
```

2.2.3 符号矩阵的生成

在 MATLAB 中输入符号向量或者矩阵的方法和输入数值类型的向量或者矩阵在形式上很像,只不过要用到符号定义函数 `syms`,使用时首先定义一些必要的符号变量,再像定义普通矩阵一样输入符号矩阵。

【例 2-18】符号矩阵的输入。

解：在命令行窗口中输入

```
>> clear
>> syms a b c
>> M1 = sym('Classical') %创建符号变量 Classical
M1 =
    Classical
```

```
>> M2 = sym('Jazz')
M2 =
    Jazz
>> M3 = sym('Blues')
M3 =
    Blues
>> syms_matrix = [a b c; M1,M2,M3; 2 3 5]
syms_matrix =
[      a,      b,      c]
[ Classical, Jazz, Blues]
[      2,      3,      5]
```

说明：无论矩阵是用分数形式还是浮点形式表示的，将矩阵转换成符号矩阵后，都将以最接近原值的有理数形式或者函数形式表示。

2.2.4 大型矩阵的生成

对于大型矩阵，通常创建一个 M 文件存储矩阵，以便于后续修改。

【例 2-19】用 M 文件创建大型矩阵。

解：在编辑器窗口中输入以下程序，并保存为 test.m。

```
tes=[ 456    468    873     2   579    55
      21    687     54   488     8    13
      65   4567     88    98    21     5
      456     68   4589   654     5   987
      5488    10     9     6    33    77]
```

在命令行窗口中输入

```
>> test
tes =
      456      468      873         2      579      55
       21      687       54      488         8      13
       65     4567       88       98       21         5
       456        68     4589     654         5     987
      5488        10         9         6        33        77

>> size(tes)
ans =
     5     6
%显示 tes 的大小
%表示 tes 有 5 行 6 列
```

2.2.5 矩阵的数学函数

MATLAB 以矩阵为基本的数据运算单位，它能够很好地与 C 语言进行混合编程，本节主要讨论一些常见的矩阵数学函数。

1. 三角函数

常用三角函数如表 2-2 所示。

表 2-2 常用三角函数

序 号	函 数 名 称	函 数 用 法
1	正弦函数 (以弧度为单位)	$Y=\sin(X)$
2	双曲正弦函数	$Y=\sinh(X)$
3	余弦函数 (以弧度为单位)	$Y=\cos(X)$
4	双曲余弦函数	$Y=\cosh(X)$
5	反正弦函数 (以弧度为单位)	$Y=\operatorname{asin}(X)$
6	反双曲正弦函数	$Y=\operatorname{asinh}(X)$
7	反余弦函数 (以弧度为单位)	$Y=\operatorname{acos}(X)$
8	反双曲余弦函数	$Y=\operatorname{acosh}(X)$
9	正切函数 (以弧度为单位)	$Y=\tan(X)$
10	双曲正切函数	$Y=\tanh(X)$
11	反正切函数 (以弧度为单位)	$Y=\operatorname{atan}(X)$
12	反双曲正切函数	$Y=\operatorname{atanh}(X)$

【例 2-20】函数应用示例。

解：在命令行窗口中依次输入

```
>> clear
>> X=magic(2)
X =
     1     3
     4     2
>> Y1=sin(X)                                %计算矩阵正弦
Y1 =
     0.8415     0.1411
    -0.7568     0.9093
>> Y2=cos(X)                                %计算矩阵余弦
Y2 =
     0.5403    -0.9900
    -0.6536    -0.4161
>> Y3=sinh(X)                                %计算矩阵双曲正弦
Y3 =
     1.1752    10.0179
    27.2899     3.6269
>> Y4=cosh(X)                                %计算矩阵双曲余弦
Y4 =
     1.5431    10.0677
    27.3082     3.7622
>> Y5=asin(X)                                %计算矩阵反正弦
Y5 =
     1.5708 + 0.0000i     1.5708 - 1.7627i
     1.5708 - 2.0634i     1.5708 - 1.3170i
>> Y6=acos(X)                                %计算矩阵反余弦
Y6 =
     0.0000 + 0.0000i     0.0000 + 1.7627i
```

```

0.0000 + 2.0634i    0.0000 + 1.3170i
>> Y7=asinh(X)      %计算矩阵反双曲正弦
Y7 =
    0.8814    1.8184
    2.0947    1.4436
>> Y8=acosh(X)      %计算矩阵反双曲余弦
Y8 =
     0    1.7627
    2.0634    1.3170
>> Y9=tan(X)        %计算矩阵正切
Y9 =
    1.5574   -0.1425
    1.1578   -2.1850
>> Y10=tanh(X)      %计算矩阵双曲正切
Y10 =
    0.7616    0.9951
    0.9993    0.9640
>> Y11=atan(X)      %计算矩阵反正切
Y11 =
    0.7854    1.2490
    1.3258    1.1071
>> Y12=atanh(X)     %计算矩阵反双曲正切
Y12 =
    Inf + 0.0000i    0.3466 + 1.5708i
    0.2554 + 1.5708i    0.5493 + 1.5708i

```

2. 指数函数和对数函数

在矩阵中，常用的指数函数和对数函数包括 `exp`、`expm` 和 `logm`。

(1) 指数函数的调用格式为

```

Y=exp(X)           %为矩阵 X 中的每个元素返回指数  $e^x$ ，可以接受任意维度的矩阵作为输入
Y=expm(X)          %计算矩阵 X 的指数并返回给 Y 值，输入参数 X 必须为方阵

```

`exp` 函数分别计算每个元素的指数，`expm` 函数计算矩阵指数。若输入矩阵是上三角矩阵或下三角矩阵，两函数计算结果中主对角线位置的元素是相等的，其余元素则不相等。

【例 2-21】 对矩阵分别用 `expm` 函数和 `exp` 函数计算魔方矩阵及其上三角矩阵的指数。

解：在命令行窗口依次输入

```

>> clear
>> A=magic(3)
A =
     8     1     6
     3     5     7
     4     9     2
>> B=expm(A)        %对矩阵 A 求指数
B =
 1.0e+06 *
    1.0898    1.0896    1.0897
    1.0896    1.0897    1.0897
    1.0896    1.0897    1.0897

```

```

>> C=exp(A)                                %对矩阵 A 的每个元素求指数
C =
    1.0e+03 *
    2.9810    0.0027    0.4034
    0.0201    0.1484    1.0966
    0.0546    8.1031    0.0074

>> B=triu(A)                                %抽取矩阵 A 中的元素构成上三角矩阵
B =
     8     1     6
     0     5     7
     0     0     2

>> expm(B)                                %求上三角矩阵的指数
ans =
    1.0e+03 *
    2.9810    0.9442    4.0203
         0    0.1484    0.3291
         0         0    0.0074

>> exp(B)                                %求上三角矩阵每个元素的指数
ans =
    1.0e+03 *
    2.9810    0.0027    0.4034
    0.0010    0.1484    1.0966
    0.0010    0.0010    0.0074

```

对上三角矩阵 B 分别用 expm 和 exp 计算，主对角线位置元素相等，其余元素则不相等。

(2) 对数函数的调用格式为

```

L=logm(A)                                %计算矩阵 A 的对数并返回 L 值，输入参数 A 必须为方阵
[L,exitflag]=logm(A)                    %exitflag 是一个标量值，描述函数的退出状态。exitflag 为 0 时，表示函
                                         %数成功完成计算；为 1 时，需要计算太多的矩阵平方根，但此时返回的结果依
                                         %然是准确的

```

logm 函数是 expm 函数的逆运算。

【例 2-22】先对方阵计算指数，再对结果计算对数，得到原矩阵。

解：在命令行窗口依次输入

```

>> clear
>> X=[1,0,1;1,0,-2;-1,0,1];
>> Y=expm(X)                                %对矩阵计算指数
Y =
    1.4687         0    2.2874
    3.1967    1.0000   -1.8467
   -2.2874         0    1.4687

>> XX=logm(Y)                                %对所得结果计算对数，得到的矩阵 xx 等于矩阵 X
XX =
    1.0000   -0.0000    1.0000
    1.0000    0.0000   -2.0000
   -1.0000    0.0000    1.0000

```

由于 logm 函数是 expm 函数的逆运算，因此得到的结果与原矩阵相等。

3. 复数函数

复数函数包括复数的创建、复数的模、复数的共轭函数等。

1) 复数的创建函数 complex

函数的调用格式为

```
c=complex(a,b) %用两个实数 a 和 b 创建复数 c, c=a+bi。c 与 a、b 是同型的数组或矩阵。如果 b 是全 0
%的, c 也依然是一个复数, 例如 c=complex(1,0) 返回复数 1, isreal(c) 等于 false,
%而 1+0i 则返回实数 1
c=complex(a) %输入参数 a 作为复数 c 的实部, c 的虚部为 0, 但 isreal(a) 返回 false, 表示 c 是一
%个复数
```

【例 2-23】创建复数 $3+2i$ 和 $3+0i$ 。

解：在命令行窗口依次输入

```
>> clear
>> a=complex(3,2) %创建复数 3+2i
a =
    3.0000 + 2.0000i
>> b=complex(3,0) %用 complex 创建复数 3+0i
b =
    3.0000 + 0.0000i
>> c=3+0i %直接创建复数 3+0i
c =
     3
>> b==c %b 与 c 的值相等
ans =
    logical
     1
>> isreal(b) %b 是复数
ans =
    logical
     0
>> isreal(c) %c 是实数
ans =
    logical
     1
```

虽然 b 与 c 相等, 但 b 是由 complex 创建的, 属于复数, 而 c 是实数。

2) 求矩阵的模 abs

该函数是 MATLAB 中十分常用的数值计算函数, 其调用格式为

```
Y=abs(X) %Y 是与 X 同型的矩阵, 如果 X 中的元素是实数, 函数返回其绝对值, 如果 X 中的元素是复数,
%函数返回复数模值, 即 sqrt(real(X).^2+imag(X).^2)
```

【例 2-24】求复数 $3+2i$ 的幅值。

解：在命令行窗口依次输入

```
>> clear
>> a=abs(3+2i) %求复数 3+2i 的幅值
a =
    3.6056
```

3) 求复数的共轭 conj

函数的调用格式为

```
Y=conj(Z) %返回 Z 中元素的复共轭值, conj(Z)=real(Z)-i*imag(Z)
```

【例 2-25】求复数 $3+2i$ 的共轭值。

解：在命令行窗口依次输入

```
>> clear
>> z=3+2i;
>> conj(z) %求 3+2i 的共轭值
ans =
    3.0000 - 2.0000i
```

复数 Z 的共轭，其实部与 Z 的实部相等，虚部是 Z 的虚部的相反数。

2.3 符号运算

MATLAB 中数值运算的操作对象是数值，而符号运算的操作对象是非数值的符号。通过 MATLAB 的符号运算功能，可以求解科学计算中符号解析表达的精确解，这在自然科学与工程计算的理论分析中有着极其重要的作用与实用价值。

2.3.1 符号对象

符号对象是一种新的数据类型（sym 类型），用来存储代表非数值的字符符号（通常是英文字母及其字符串）。符号对象可以是符号常量（符号形式的数）、符号变量、符号函数及各种符号表达式（符号数学表达式、符号方程与符号矩阵）。

在 MATLAB 中，符号对象可利用函数 sym、syms 来建立，而利用函数 class 来测试建立的操作对象为何种操作对象类型、是否为符号对象类型（即 sym 类型）。

在一个 MATLAB 程序中，作为符号对象的符号常量、符号变量、符号函数及符号表达式，首先需要使用函数 sym、syms 加以规定，即创建。

1. 函数 sym 的调用格式

```
S=sym(A) %由 A 来建立一个符号对象 S，其类型为 sym 类型
S=sym('A') %如果 A（不带单引号）是一个数字（值）或数值矩阵和数值表达式，则输出是将数值对象
%转换成的符号对象。如果 A（带单引号）是一个字符串，输出则是将字符串转换成的符号
%对象
S=sym(A,flag) %转换成的符号对象应符合 flag 格式
S=sym('A',set) %转换成的符号对象应按 set 指定的要求
```

flag 可取以下选项：

- 'd'——最接近的十进制浮点精确表示；
- 'e'——带估计误差（数值计算时 0）的有理数表示；
- 'f'——十六进制浮点表示；
- 'r'——为默认设置，是最接近有理数表示的形式。这种形式是指用两个正整数 p 、 q 构成的 p/q 、 $p\pi/q$ 、

- $\text{sqrt}(p)$ 、 2^p 、 10^q 表示的形式之一。
- set 可取以下“限定性”选项：
- 'positive'——限定 A 为正的实型符号变量；
 - 'real'——限定 A 为实型符号变量；
 - 'integer'——限定 A 为整型符号变量；
 - 'rational'——限定 A 为有理数符号变量。

2. 函数syms的调用格式

```
syms s1 s2 s3 ... set           %按 set 指定的要求建立 1 个或多个符号对象 s1、s2...
```

3. 函数class的调用格式

```
str=class(object)              %返回指代数据对象类型的字符串。数据对象类型如表 2-3 所示
```

表 2-3 数据对象类型

数 据 对 象	说 明
cell	CELL数组
char	字符数组
double	双精度浮点数值类型
int8	8 位带符号整型数组
int16	16位带符号整型数组
int32	32位带符号整型数组
sparse	实（或复）稀疏矩阵
struct	结构数组
uint8	8 位不带符号整型数组
uint16	16位不带符号整型数组
uint32	32位不带符号整型数组
<class_name>	用户定义的对象类型
<java_class>	Java对象的Java类型
sym	符号对象类型

【例 2-26】对数值量 1/4 创建符号对象并检测数据的类型。

解：在命令行窗口依次输入以下语句

```
>> a=1/4;
>> b='1/4';
>> c=sym(1/4);
>> d=sym('1/4');
>> classa=class(a)
classa =
    'double'
>> classb=class(b)
classb =
    'char'
```

```
>> classc=class(c)
classc =
    'sym'
>> classd=class(d)
classd =
    'sym'
```

即 a 是双精度浮点数值类型； b 是字符类型； c 与 d 都是符号对象类型。

【例 2-27】 创建符号对象，观察符号对象形成中的差异。

解：在命令行窗口依次输入

```
>> a1=[1/3, pi/7, sqrt(5), pi+sqrt(5)]
a1 =
    0.3333    0.4488    2.2361    5.3777
>> a2=sym([1/3, pi/7, sqrt(5), pi+sqrt(5)])
a2 =
    [ 1/3, pi/7, 5^(1/2), 189209612611719/35184372088832]
>> a3=sym([1/3, pi/7, sqrt(5), pi+sqrt(5)], 'e')
a3 =
    [ 1/3 - eps/12, pi/7 - (13*eps)/165, (137*eps)/280 + 5^(1/2),
    189209612611719/35184372088832]
>> y=sym(2*sin(x)*cos(x))
y =
    2*cos(x)*sin(x)
```

2.3.2 符号变量

符号变量通常是指一个或几个特定的字符，不是指符号表达式，虽然可以将一符号表达式赋值给一个符号变量。符号变量有时也叫作自由变量。符号变量与 MATLAB 数值运算的数值变量名的命名规则相同：

- 变量名可以由英文字母、数字和下画线组成；
- 变量名应以英文字母开头；
- 组成变量名的字符长度不大于 31 个；
- MATLAB 区分英文字母大小写。

在 MATLAB 中，同样可以用函数 `sym` 或 `syms` 来建立符号变量。

【例 2-28】 用函数 `sym` 与 `syms` 建立符号变量 α 、 β ，并检测数据的类型。

解：用函数 `sym` 创建符号对象，在命令行窗口中输入

```
>> a=sym('alpha')
a =
    alpha
>> b=sym('beta')
b =
    beta
>> classa=class(a)
classa =
    'sym'
>> classb=class(b)
```

```
classb =
    'sym'
```

用函数 `syms` 创建符号对象并检测数据的类型，在命令行窗口中输入

```
>> syms alpha beta;
>> classa=class(alpha)
classa =
    'sym'
>> classb=class(beta)
classb =
    'sym'
```

语句执行完后可以确认数据对象 α 、 β 是符号对象类型。

【例 2-29】 求矩阵 $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ 的行列式值、逆和特征根。

解：在命令行窗口中输入

```
>> syms a11 a12 a21 a22
>> A=[a11,a12;a21,a22]
A =
    [ a11, a12]
    [ a21, a22]
>> DA=det(A)
DA =
    a11*a22 - a12*a21
>> IA=inv(A)
IA =
    [ a22/(a11*a22 - a12*a21), -a12/(a11*a22 - a12*a21)]
    [ -a21/(a11*a22 - a12*a21),  a11/(a11*a22 - a12*a21)]
>> EA=eig(A)
EA =
    a11/2 + a22/2 - (a11^2 - 2*a11*a22 + a22^2 + 4*a12*a21)^(1/2)/2
    a11/2 + a22/2 + (a11^2 - 2*a11*a22 + a22^2 + 4*a12*a21)^(1/2)/2
```

2.3.3 符号表达式及函数

在 MATLAB 数值运算中，数值表达式是由常量、数值变量、数值函数或数值矩阵用运算符连接而成的数学关系式。而在 MATLAB 符号运算中，符号表达式是由符号常量、符号变量、符号函数用运算符或专用函数连接而成的符号对象。

符号表达式有两类：符号函数与符号方程。符号函数不带等号，而符号方程是带等号的。在 MATLAB 中，同样用函数 `sym` 来建立符号表达式。

1. 符号表达式与符号方程的建立

【例 2-30】 建立符号函数 f_1 、 f_2 、 f_3 、 f_4 并检测符号对象的类型。

解：利用 `syms` 与 `sym` 创建符号函数并检测数据的类型。在命令行窗口中输入

```
>> syms n x T wc p z;
>> f1=n*x^n/x;
```

```
>> f2=sym(log(T)^2*T+p);
>> f3=sym(w+sin(a*z));
>> classf1=class(f1)
>> f4=pi+atan(T*wc);
classf1 =
    'sym'
>> classf2=class(f2)
classf2 =
    'sym'
>> classf3=class(f3)
classf3 =
    'sym'
>> classf4=class(f4)
classf4 =
    'sym'
```

【例 2-31】建立符号方程 e_1 、 e_2 、 e_3 、 e_4 并检测符号对象的类型。

解：利用 `syms` 与 `sym` 创建符号方程并检测数据的类型。在命令行窗口中输入

```
>> syms a b c x y t p Dy
>> e1=sym(a*x^2+b*x+c==0)
>> e2=sym(log(t)^2*t==p)
>> e3=sym(sin(x)^2+cos(x)==0)
>> e4=sym(Dy-y==x)
>> classe1=class(e1)
classe1 =
    'sym'
>> classe2=class(e2)
classe2 =
    'sym'
>> classe3=class(e3)
classe3 =
    'sym'
>> classe4=class(e4)
classe4 =
    'sym'
```

2. 符号表达式中的符号变量

在微积分、函数表达式化简、求解方程中，确定自变量是必不可少的。在不指定自变量的情况下，按照数学常规，自变量通常都是小写英文字母，并且为字母表末尾的几个，如 t 、 w 、 x 、 y 、 z 等。

在 MATLAB 中，可以用函数 `symvar` 按这种数学习惯来确定一个符号表达式中的自变量，这对于按照特定要求进行某种计算是非常有实用价值的。

函数 `symvar` 的调用格式为

<code>symvar(f,n)</code>	%按数学习惯确定符号函数 f 中的 n 个自变量。当 $n=1$ 时，从 f 中找出在字母表中与 x 最近的字母；若有两个字母与 x 的距离相等，取较后的一个。当 n 为默认值时，将给出 f 中所有的符号变量
<code>symvar(e,n)</code>	%按数学习惯确定符号方程 e 中的 n 个自变量。其余同 <code>symvar(f,n)</code>

【例 2-32】确定符号函数 f_1 、 f_2 中的自变量。

解：在命令行窗口中依次输入

```
>> syms k m n w y z;
>> f=n*y^n+m*y+w;
>> ans1= symvar (f,1)
ans1 =
    y
>> f2=m*y+n*log(z)+exp(k*y*z);
>> ans2= symvar (f2,2)
ans2 =
    [ y, z]
```

【例 2-33】确定符号方程 e_1 、 e_2 中的自变量。

解：在命令行窗口中依次输入

```
>> clear
>> syms a b c x p q t w;
>> e1=sym(a*x^2+b*x+c==0);
>> ans1=symvar(e1,1)
ans1 =
    x
>> e2=sym(w*(sin(p*t+q))==0);
>> ans2=symvar(e2)
ans2 =
    [ p, q, t, w]
```

2.3.4 符号变量代换

函数 subs 可以实现符号变量代换，其调用格式为

```
subs(S, old, new)    %将符号表达式 S 中的 old 变量替换为 new。old 一定是符号表达式 S 中的符号变量，
                    %new 可以是符号变量、符号常量、双精度数值与数值数组等
subs(S, new)         %用 new 置换符号表达式 S 中的自变量
```

【例 2-34】已知 $f=ax^n+by+k$ ，试对其进行：①符号变量替换： $a=\sin t$ 、 $b=\ln w$ 、 $k=ce^{-dt}$ ；②符号常量替换： $n=5$ 、 $k=\pi$ ；③数值数组替换： $k=1:1:4$ 。

解：在命令行窗口中依次输入

```
>> clear
>> syms a b c d k n x y w t;
>> f=a*x^n+b*y+k
f =
    k + a*x^n + b*y
>> f1=subs(f,[a b],[sin(t) log(w)])           %替换 a、b
f1 =
    k + x^n*sin(t) + y*log(w)
>> f2=subs(f,[a b k],[sin(t) log(w) c*exp(-d*t)]) %替换 a、b、k
f2 =
    c*exp(-d*t) + x^n*sin(t) + y*log(w)
>> f3=subs(f,[n k],[5 pi])                     %替换 n、k
```

```
f3 =
      a*x^5 + pi + b*y
>> f4=subs(f1,k,1:4)           %替换为数组
f4 =
      [ x^n*sin(t) + y*log(w) + 1, x^n*sin(t) + y*log(w) + 2, x^n*sin(t) + y*log(w)
      + 3, x^n*sin(t) + y*log(w) + 4]
```

若要对符号表达式进行两个变量的数值数组替换, 可以用循环程序来实现。

【例 2-35】已知 $f=asinx+k$, 试求当 $a=1:2$ 与 $x=0:30:60(^{\circ})$ 时函数 f 的值。

解: 在命令行窗口中依次输入

```
>> clear
>> syms a k x;
>> f=a*sin(x)+k;
>> for a=1:2;
      for x=0:pi/6:pi/3;
          f1=a*sin(x)+k
      end
  end
```

程序运行第一组 (当 $a=1$ 时) 后的结果:

```
f1 =
      k
f1 =
      k + 1/2
f1 =
      k + 3^(1/2)/2
```

程序运行第二组 (当 $a=2$ 时) 后的结果:

```
f1 =
      k
f1 =
      k + 1
f1 =
      k + 3^(1/2)
```

2.3.5 符号对象转换为数值对象

多数符号运算的目的是计算表达式的数值解, 于是需要将符号表达式的解析解转换为数值解。使用函数 `double` 可以得到双精度数值解; 使用函数 `digits` 可以得到指定精度的精确数值解; 使用函数 `vpa` 可以精确计算表达式的值; 使用函数 `numeric` 可以将符号对象转换为数值形式。联合使用 `digits` 与 `vpa` 两个函数可以实现解析解的数值转换。这几个函数调用格式为

<code>double(C)</code>	%将符号常量 C 转换为双精度数值
<code>digits(D)</code>	%设置有效数字个数为 D 的近似解精度
<code>R=vpa(E)</code>	%必须与函数 <code>digits(D)</code> 连用, 在其设置下, 求得符号表达式 E 的设定精度的数值解, 返回的数值解为符号对象类型
<code>R=vpa(E,D)</code>	%求得符号表达式 E 的 D 位精度的数值解, 返回的数值解是符号对象类型
<code>N=numeric(E)</code>	%将不含变量的符号表达式 E 转换为双精度浮点数值形式, 与 <code>N=double(sym(E))</code> 相同

【例 2-36】计算以下三个符号常量的值： $c_1 = \sqrt{2} \ln 7$ 、 $c_2 = \pi \sin \frac{\pi}{5} e^{1.3}$ 、 $c_3 = e^{\sqrt{8}\pi}$ ，并将结果转换为双精度型数值。

解：在命令行窗口中依次输入

```
>> clear
>> syms c1 c2 c3;
>> c1=sym(sqrt(2)*log(7));
>> c2=sym(pi*sin(pi/5)*exp(1.3));
>> c3=sym(exp(pi*sqrt(8)));
>> ans1=double(c1)
ans1 =
    2.7519
>> ans2=double(c2)
ans2 =
    6.7757
>> ans3=double(c3)
ans3 =
    7.2283e+003
>> class(ans1)
ans =
    'double'
>> class(ans2)
ans =
    'double'
>> class(ans3)
ans =
    'double'
```

即 $c_1=2.7519$ 、 $c_2=6.7757$ 、 $c_3=7.2283e^3$ ，并且它们都是双精度型数值。

【例 2-37】计算符号常量 $c_1 = e^{\sqrt{79}\pi}$ 的值，并将结果转换为指定精度为 8 位与 18 位的精确数值解。

解：在命令行窗口中依次输入

```
>> clear
>> c=sym(exp(pi*sqrt(79)));
>> c1=double(c)
c1 =
    1.3392e+12
>> ans1=class(c1)
ans1 =
    'double'
>> c2=vpa(c1,8)
c2 =
    1.3391903e+12
>> ans2=class(c2)
ans2 =
    'sym'
>> digits 18
>> c3=vpa(c1)
```

```

c3 =
    1339190288739.15283
>> ans3=class(c3)
ans3 =
    'sym'

```

2.3.6 符号表达式的化简

MATLAB 提供了多个对符号表达式进行化简的函数, 诸如因式分解、合并同类项、符号表达式的展开、符号表达式的化简与通分等, 它们都是表达式的恒等变换。

1. 函数factor

函数 factor 用于符号表达式的因式分解, 其调用格式为

```
factor(E)    %对符号表达式 E 进行因式分解, 如果 E 包含的所有元素为整数, 则计算其最佳因式分解式。对
              %于大于 252 的整数的分解, 可使用语句 factor(sym('N'))
```

【例 2-38】已知 $f=x^3+x^2-x-1$, 试对其进行因式分解。

解: 在命令行窗口中依次输入

```

>> syms x;
>> f=x^3+x^2-x-1;
>> f1=factor(f)
f1 =
    [ x - 1, x + 1, x + 1]

```

即 $f=x^3+x^2-x-1=(x-1)(x+1)^2$ 。

2. 函数expand

借助函数 expand 对符号表达式展开, 调用格式为

```
expand(E)    %将符号表达式 E 展开, 常用在多项式、三角函数、指数函数与对数函数的展开中
```

【例 2-39】已知 $f=(x+y)^3$, 试将其展开。

解: 在命令行窗口中依次输入

```

>> syms x y;
>> f=(x+y)^3;
>> f1=expand(f)
f1 =
    x^3 + 3*x^2*y + 3*x*y^2 + y^3

```

即 $f=(x+y)^3=x^3+3x^2y+3xy^2+y^3$ 。

3. 函数collect

借助函数 collect 对符号表达式合并同类项, 调用格式有两种

```
collect(E,v)    %将符号表达式 E 中 v 的同幂次项合并
collect(E)      %将符号表达式 E 中默认变量的同类项系数合并, 默认变量由函数 symvar 确定
```

【例 2-40】已知 $f = -axe^{-cx} + be^{-cx}$ ，试对其同类项进行合并。

解：在命令行窗口中依次输入

```
>> syms a b c x;
>> f=-a*x*exp(-c*x)+b*exp(-c*x);
>> f1=collect(f,exp(-c*x))
f1 =
      (b - a*x)*exp(-c*x)
```

即 $f = -axe^{-cx} + be^{-cx} = (b - ax)e^{-cx}$ 。

4. 函数simplify

借助函数 simplify 对符号表达式进行化简，调用格式为

```
simplify(E) %将符号表达式 E 运用多种恒等式变换进行综合化简
```

【例 2-41】试对 $e_1 = \sin^2 x + \cos^2 x$ 与 $e_2 = e^{c \ln(\alpha + \beta)}$ 进行综合化简。

解：在命令行窗口中依次输入

```
>> syms x n c alph beta;
>> e10=sin(x)^2+cos(x)^2;
>> e1=simplify(e10)
e1 =
      1
>> e20=exp(c*log(alph+beta));
>> e2=simplify(e20)
e2 =
      (alph + beta)^c
```

即 $e_1 = \sin^2 x + \cos^2 x = 1$ 和 $e_2 = e^{c \ln(\alpha + \beta)} = (\alpha + \beta)^c$ 。

5. 函数numden

借助函数 numden 可以对符号表达式进行通分，调用格式为

```
[N,D]=numden(E) %将符号表达式 E 通分，分别返回 E 通分后的分子 N 与分母 D，并且转换成的分子与分母都是整系数的最佳多项式形式。只需要再计算 N/D 即求得符号表达式 E 通分的结果。若无等号左边的输出参数，则仅返回 E 通分后的分子 N
```

【例 2-42】已知 $f = \frac{x}{ky} + \frac{y}{px}$ ，试对其进行通分。

解：在命令行窗口中依次输入

```
>> syms k p x y;
>> f=x/(k*y)+y/(p*x);
>> [n,d]=numden(f)
n =
      p*x^2 + k*y^2
d =
      k*p*x*y
>> f1=n/d
f1 =
      (p*x^2 + k*y^2)/(k*p*x*y)
```

```
>> numden(f)
ans =
    p*x^2 + k*y^2
```

即

$$f = \frac{x}{ky} + \frac{y}{px} = \frac{px^2 + ky^2}{kpxy}$$

当无等号左边的输出参数时, 仅返回通分后的分子。

6. 函数 horner

函数 horner 对符号表达式进行嵌套型分解, 调用格式为

```
horner(E) %将符号表达式 E 转换成嵌套形式表达式
```

【例 2-43】已知 $f = -ax^4 + bx^3 - cx^2 + x + d$, 试将其转换成嵌套形式表达式。

解: 在命令行窗口中依次输入

```
>> syms a b c d x;
>> f = -a*x^4 + b*x^3 - c*x^2 + x + d;
>> f1 = horner(f)
f1 =
    d - x*(x*(c - x*(b - a*x)) - 1)
```

即 $f = -ax^4 + bx^3 - cx^2 + x + d = d - x(x(c - x(b - ax)) - 1)$ 。

2.3.7 符号运算的其他函数

1. 函数 char

借助函数 char 可以将数值对象、符号对象转换为字符对象, 调用格式为

```
char(S) %将数值对象或符号对象 S 转换为字符对象
```

【例 2-44】试将数值对象 $c = 123456$ 与符号对象 $f = x + y + z$ 转换成字符对象。

解: 在命令行窗口中依次输入

```
>> syms a b c x y z;
>> c = 123456;
>> ans1 = class(c)
ans1 =
    'double'
>> c1 = char(sym(c))
c1 =
    '123456'
>> ans2 = class(c1)
ans2 =
    'char'
>> f = sym(x + y + z);
>> ans3 = class(f)
```

```
ans3 =
    'sym'
>> f1=char(f)
f1 =
    'x + y + z'
>> ans4=class(f1)
ans4 =
    'char'
```

即原数值对象与符号对象均被转换成字符对象。

2. 函数pretty

借助函数 pretty 以习惯的方式显示符号表达式，调用格式为

```
pretty(E) %以习惯的“书写”方式显示符号表达式 E (包括符号矩阵)
```

【例 2-45】试将 MATLAB 符号表达式 $f=a*x/b+c/(d*y)$ 与 $\text{sqrt}(b^2-4*a*c)$ 以习惯的“书写”方式显示。

解：在命令行窗口中依次输入

```
>> syms a b c d x y;
>> f=a*x/b+c/(d*y);
>> f1=sqrt(b^2-4*a*c);
>> pretty(f)
c    a x
--- + ---
d y    b
>> pretty(f1)
      2
sqrt(b  - 4 a c)
```

即 $f = \frac{ax}{b} + \frac{c}{dy}$ 与 $f_1 = \sqrt{b^2 - 4ac}$ 。

3. 函数clear

函数 clear 用于清除 MATLAB 工作空间变量与函数，调用格式为

```
clear
```

这是一个不带输入参数的命令，其功能是清除 MATLAB 工作空间中保存的变量与函数。通常置于程序之首，以免原有 MATLAB 工作空间中保存的变量与函数影响新的程序。

2.3.8 两种特定的符号运算函数

MATLAB 两种特定的符号函数运算是指复合函数运算与反函数运算。

1. 复合函数运算

设 z 是 y (自变量) 的函数 $z=f(y)$ ，而 y 又是 x (自变量) 的函数 $y=j(x)$ ，则 z 对 x 的函数 $z=f(j(x))$ 叫作 z 对 x 的复合函数。求 z 对 x 的复合函数 $z=f(j(x))$ 的过程叫作复合函数运算。

MATLAB 求复合函数的函数为 `compose`。函数调用格式为

<code>compose(f,g)</code>	%当 $f=f(x)$ 与 $g=g(y)$ 时返回复合函数 $f(g(y))$, 即用 $g=g(y)$ 代入 $f(x)$ 中的 x , 且 % x 为函数命令 <code>symvar()</code> 确定的 f 的自变量, y 为 <code>symvar()</code> 确定的 g 的自变量
<code>compose(f,g,z)</code>	%当 $f=f(x)$ 与 $g=g(y)$ 时返回以 z 为自变量的复合函数 $f(g(z))$, 即用 $g=g(y)$ %代入 $f(x)$ 中的 x , 且 $g(y)$ 中的自变量 y 改换为 z
<code>compose(f,g,x,z)</code>	%同 <code>compose(f,g,z)</code>
<code>compose(f,g,t,z)</code>	%当 $f=f(t)$ 与 $g=g(y)$ 时返回以 z 为自变量的复合函数 $f(g(z))$, 即用 $g=g(y)$ 代入 % $f(t)$ 中的 t , 且 $g(y)$ 中的自变量 y 改换为 z
<code>compose(f,h,x,y,z)</code>	%同 <code>compose(f,g,z)</code>
<code>compose(f,g,t,u,z)</code>	%当 $f=f(t)$ 与 $g=g(u)$ 时返回以 z 为自变量的复合函数 $f(g(z))$, 即用 $g=g(u)$ 代入 % $f(t)$ 中的 t , 且 $g(u)$ 中的自变量 u 改换为 z

【例 2-46】已知 $f = \ln \frac{x}{t}$ 与 $g = u \times \cos y$, 求其复合函数 $f(\varphi(x))$ 与 $f(g(z))$ 。

解: 在命令行窗口中依次输入

```
>> syms f g t u x y z;
>> f=log(x/t);
>> g=u*cos(y);
>> cfg=compose(f,g)
cfg =
    log((u*cos(y))/t)
>> cfgt=compose(f,g,z)
cfgt =
    log((u*cos(z))/t)
>> cfgxz=compose(f,g,x,z)
cfgxz =
    log((u*cos(z))/t)
>> cfgtz=compose(f,g,t,z)
cfgtz =
    log(x/(u*cos(z)))
>> cfgxyz=compose(f,g,x,y,z)
cfgxyz =
    log((u*cos(z))/t)
>> cfgxyz=compose(f,g,t,u,z)
cfgxyz =
    log(x/(z*cos(y)))
```

2. 反函数运算

设 y 是 x (自变量) 的函数 $y=f(x)$, 若将 y 当作自变量, x 当作函数, 则函数 $x=j(y)$ 叫作函数 $f(x)$ 的反函数, 而 $f(x)$ 叫作直接函数。

在同一坐标系中, 直接函数 $y=f(x)$ 与反函数 $x=j(y)$ 表示同一图形。通常把 x 当作自变量, 而把 y 当作函数, 故反函数 $x=j(y)$ 写为 $y=j(x)$ 。

MATLAB 提供的求反函数的函数为 `finverse`, 调用格式为

```
g=finverse(f,v) %求符号函数 f 的自变量为 v 的反函数 g
g=finverse(f)    %求符号函数 f 的反函数 g,符号函数 f 有自变量 x,函数 g 也是符号函数,且有 g(f(x))=x
```

【例 2-47】求函数 $y = ax + b$ 的反函数。

解：在命令行窗口中依次输入

```
>> syms a b x y;
>> y=a*x+b;
>> g=finverse(y)
g =
    -(b - x)/a
>> compose(y,g)
ans =
    x
```

即反函数为 $y = \frac{-(b-x)}{a}$ ，且 $g(f(x)) = x$ 。

2.4 关系运算和逻辑运算

前面介绍了算术运算，下面介绍关系运算和逻辑运算。关系运算用于比较两个操作数，而逻辑运算则是对简单逻辑表达式进行复合运算。关系运算和逻辑运算的返回结果都是逻辑类型(1 代表逻辑真，0 代表逻辑假)。

2.4.1 关系运算

在程序中经常需要比较两个量的大小关系，以决定程序下一步的工作。比较两个量的运算符称为关系运算符。MATLAB 中的关系运算符如表 2-4 所示。

表 2-4 MATLAB 中的关系运算符

运 算 符	名 称	示 例	使 用 说 明
<	小于	A<B	(1) A、B 都是标量，结果是 1（真）或 0（假）的标量； (2) A、B 中若一个为标量、一个为数组，则标量将与数组各元素逐一比较，结果为与运算数组行列相同的数组，其中各元素取值为 1 或 0； (3) A、B 均为数组时，必须行、列数分别相同，A 与 B 各对应元素相比较，结果为与 A 或 B 行列相同的数组，其中各元素取值为 1 或 0； (4) == 和 ~ = 运算对参与比较的量同时比较实部和虚部，其他运算只比较实部
<=	小于或等于	A<=B	
>	大于	A>B	
>=	大于或等于	A>=B	
==	恒等于	A==B	
~=	不等于	A ~= B	

需要明确指出的是，MATLAB 的关系运算虽可看成矩阵的关系运算，但严格地讲，把关系运算定义在数组基础之上更为合理。从表 2-4 中所列的法则不难发现，关系运算是元素一对一的运算结果。数组的关系运算向下可兼容一般高级语言中所定义的标量关系运算。

当操作数是数组形式时，关系运算符总是对被比较的两个数组的各个对应元素进行比较，因此要求被比较的数组必须具有相同的尺寸。

【例 2-48】MATLAB 中的关系运算。

解：在命令行窗口输入

```
>> 5>=4
ans =
    logical
     1
>> x=rand(1,4)
x =
    0.8147    0.9058    0.1270    0.9134
>> y=rand(1,4)
y =
    0.6324    0.0975    0.2785    0.5469
>> x>y
ans =
    1×4 logical 数组
     1     1     0     1
```

注意:

- (1) 比较两个数是否相等的关系运算符是两个等号“==”，而单个等号“=”在 MATLAB 中是变量赋值的符号；
- (2) 比较两个浮点数是否相等时需要注意，由于浮点数的存储形式决定了相对误差的存在，在程序设计中最好不要直接比较两个浮点数是否相等，而是采用大于、小于的比较运算将待确定值限制在一个满足需要的区间之内。

2.4.2 逻辑运算

关系运算返回的结果是逻辑类型（逻辑真或逻辑假），这些简单的逻辑数据可以通过逻辑运算符组成复杂的逻辑表达式，这在程序设计中经常用于进行分支选择或者确定循环终止条件。

MATLAB 中的逻辑运算有 3 类：逐个元素的逻辑运算、捷径逻辑运算和逐位逻辑运算。只有前两种逻辑运算返回逻辑类型的结果。

1. 逐个元素的逻辑运算

逐个元素的逻辑运算符有三种：逻辑与（&）、逻辑或（|）和逻辑非（~）。前两个是双目运算符，必须有两个元素参与运算，逻辑非是单目运算符，只对单个元素进行运算，其意义和示例如表 2-5 所示。

表 2-5 逐个元素的逻辑运算符

运 算 符	说 明	示 例
&	逻辑与：双目逻辑运算符 参与运算的两个元素值为逻辑真或非零时，返回逻辑真，否则返回逻辑假	1&0返回0 1&false返回0 1&1返回1
	逻辑或：双目逻辑运算符 参与运算的两个元素都为逻辑假或零时，返回逻辑假，否则返回逻辑真	1 0返回1 1 false返回1 0 0返回0
~	逻辑非：单目逻辑运算符 参与运算的元素为逻辑真或非零时，返回逻辑假，否则返回逻辑真	~ 1返回0 ~ 0返回1

注意：这里逻辑与和逻辑非运算，都是逐个元素进行双目运算，因此如果参与运算的是数组，就要求两个数组具有相同的尺寸。

【例 2-49】逐个元素的逻辑运算示例。

解：在命令行窗口输入

```
>> x=rand(1,3)
x =
    0.9575    0.9649    0.1576
>> y=x>0.5
y =
1×3 logical 数组
     1     1     0
>> m=x<0.96
m =
1×3 logical 数组
     1     0     1
>> y&m
ans =
1×3 logical 数组
     1     0     0
>> y|m
ans =
1×3 logical 数组
     1     1     1
>> ~y
ans =
1×3 logical 数组
     0     0     1
```

2. 捷径逻辑运算

MATLAB 中捷径逻辑运算符有两个：逻辑与（&&）和逻辑或（||）。实际上它们的运算功能和前面讲过的逐个元素的逻辑运算符相似，只不过在一些特殊情况下，捷径逻辑运算符会少一些逻辑判断的操作。

当参与逻辑与运算的两个元素都同为逻辑真（非零）时，逻辑与运算才返回逻辑真（1），否则都返回逻辑假（0）。

&&运算符就是利用这一特点，当参与运算的第一个元素为逻辑假时，直接返回假，而不再去计算第二个元素。&运算符在任何情况下都要计算两个元素的结果，然后进行逻辑与运算。

||运算符的情况类似，当第一个元素为逻辑真时，||直接返回逻辑真，而不再去计算第二个元素。|运算符任何情况下都要计算两个元素的结果，然后进行逻辑或运算。

捷径逻辑运算符如表 2-6 所示。

表 2-6 捷径逻辑运算符

运 算 符	说 明
&&	捷径逻辑与：当第一个元素为假，直接返回假，否则同&
	捷径逻辑或：当第一个元素为真，直接返回真，否则同

因此，捷径逻辑运算符比相应的逐个元素的逻辑运算符的运算效率更高，在实际编程中，一般都用捷径逻辑运算符。

【例 2-50】捷径逻辑运算。

解：在命令行窗口中输入

```
>> x=0
x =
    0
>> x~=0&&(1/x>2)
ans =
    logical
    0
>> x~=0&(1/x>2)
ans =
    logical
    0
```

3. 逐位逻辑运算

逐位逻辑运算函数能够对二进制形式非负整数进行逐位逻辑运算，并将逐位运算后的二进制数值转换成十进制数值输出。MATLAB 中逐位逻辑运算函数如表 2-7 所示。

表 2-7 MATLAB中逐位逻辑运算函数

函 数	说 明
bitand(a,b)	逐位逻辑与：a和b的二进制数位上都为1则返回1，否则返回0，并将逐位逻辑运算后的二进制数值转换成十进制数值输出
bitor(a,b)	逐位逻辑或：a和b的二进制数位上都为0则返回0，否则返回1，并将逐位逻辑运算后的二进制数值转换成十进制数值输出
bitcmp(a)	逐位逻辑非：将a扩展成n位二进制形式，当扩展后的二进制数位上都为1则返回0，否则返回1，并将逐位逻辑运算后的二进制数值转换成十进制数值输出
bitxor(a,b)	逐位逻辑异或：a和b的二进制数位上相同则返回0，否则返回1，并将逐位逻辑运算后的二进制数值转换成十进制数值输出

【例 2-51】逐位逻辑运算函数。

解：在命令行窗口输入

```
>> m=8;n=2;
>> mm=bitxor(m,n);
>> dec2bin(m)
ans =
    1000
>> dec2bin(n)
ans =
     10
>> dec2bin(mm)
ans =
    1010
```

2.4.3 常用函数

除上面的关系与逻辑运算操作符外，MATLAB 提供了大量的其他关系与逻辑函数，具体如表 2-8 所示。

表 2-8 其他关系与逻辑函数

函 数	说 明
xor(x,y)	异或运算。x或y非零（真）返回1，x和y都是零（假）或都是非零（真）返回0
any(x)	如果在一个向量x中，任何元素非零，返回1；矩阵x中的每列有非零元素，返回1
all(x)	如果在一个向量x中，所有元素非零，返回1；矩阵x中的每列所有元素非零，返回1

【例 2-52】关系与逻辑函数的应用。

解：在命令行窗口输入

```
>> A=[0 0 3;0 3 3]
A =
     0     0     3
     0     3     3
>> B=[0 -2 0;1 -2 0]
B =
     0    -2     0
     1    -2     0
>> C=xor(A,B)
C =
   2×3 logical 数组
     0     1     1
     1     0     1
>> D=any(A)
D =
   1×3 logical 数组
     0     1     1
>> E=all(A)
E =
   1×3 logical 数组
     0     0     1
```

除了这些函数，MATLAB 还提供了大量的函数，用于测试特殊值或条件的存在，返回逻辑值，如表 2-9 所示，限于篇幅本书不再介绍。

表 2-9 测试函数

测试函数	说 明	测试函数	说 明
finite	元素有限，返回真值	isieee	计算机执行IEEE算术运算，返回真值
isempty	参量为空，返回真值	isinf	元素无穷大，返回真值
isglobal	参量是一个全局变量，返回真值	isletter	元素为字母，返回真值
ishold	当前绘图保持状态是ON，返回真值	isnan	元素为不定值，返回真值

续表

测试函数	说 明	测试函数	说 明
isreal	参量无虚部，返回真值	isstudent	MATLAB为学生版，返回真值
isspace	元素为空格字符，返回真值	isunix	计算机为UNIX系统，返回真值
isstr	参量为一个字符串，返回真值	isvms	计算机为VMS系统，返回真值

2.4.4 运算符的优先级

和其他高级语言一样，当用多个运算符和运算量写出一个 MATLAB 表达式时，运算符的优先次序是一个必须明确的问题。表 2-10 列出了 MATLAB 中运算符的优先次序。

表 2-10 MATLAB中运算符的优先次序

优 先 次 序	运 算 符
高 ↓ 低	' (转置共轭)、^ (矩阵乘幂)、!' (转置)、.^ (数组乘幂)
	~ (逻辑非)
	、/ (右除)、\ (左除)、. (数组乘)、./ (数组右除)、.\ (数组左除)
	+, -, : (冒号运算)
	<, <=, >, >=, == (恒等于)、~= (不等于)
	& (逻辑与)
	(逻辑或)
	&& (捷径逻辑与)
	(捷径逻辑或)

在表 2-10 中，从上到下优先次序为由高到低，同一行的各运算符具有相同的优先级，同时在同一级别中遵循有括号先进行括号运算的原则。

2.5 本章小结

数组、变量和矩阵是 MATLAB 语言中必不可少的要件，其中数组是 MATLAB 中各种变量存储和运算的通用数据结构。MATLAB 把数组、变量、矩阵当成基本的运算量，除了传统的数学运算，MATLAB 支持关系和逻辑运算。