第二章 MATLAB的基本数学功能

目 录

2.1 数据类型

MATLAB中定义了很多种数据类型:

2.1.1 数值类型

2.1.2 逻辑类型

2.1.3 字符和字符串类型

2.1.4 单元数组类型

2.1.5 结构体类型

2.1.6 函数句柄类型

在MATLAB中有15种基本数据类型，

每种基本数据类型均以数组/矩阵的形式出现。

- Integer type (8)
- Floating-point type (2)
- Logical type
- Char type
- Cell array
- Structure type
- Function handle

2.1.1 数值类型

数值类型包含

- 整数 integer
- 浮点数 float
- 复数 complex
- Inf (infinite)
- NaN (not a number)

（1）整数类型

Function	signed /unsigned	Byte per element	Range
int8(x)	signed (有符号的)	1	$-2^7 \sim 2^7-1$
int16(x)		2	$-2^{15} \sim 2^{15}-1$
int32(x)		4	$-2^{31} \sim 2^{31}-1$
int64(x)		8	$-2^{63} \sim 2^{63}-1$
uint8(x)	unsigned (无符号的)	1	$0 \sim 2^8-1$
uint16(x)		2	$0 \sim 2^{16}-1$
uint32(x)		4	$0 \sim 2^{32}-1$
uint64(x)		8	$0 \sim 2^{64}-1$

（1）整数类型

从浮点数向整型数的转换函数

函数	说明
round	向最接近的整数取整， 如果小数为0.5，则取绝对值大的整数
fix	向0取整
floor	不大于该数的最接近整数
ceil	不小于该数的最接近整数

* 关于数制转化的讨论

常用进制表示方法：

- 十进制：适合于人的认知与习惯
- 二进制：适合于电脑的计算、存储等
- 八进制：方便与二进制进行转换
- 十六进制：方便与二进制进行转换
- 十二进制
- 六十进制

* 关于数制转化的讨论

➢ 二进制

电脑中使用二进制的优点：

- ✓ 技术实现简单：逻辑电路的开、关对应“1”和“0”
- ✓ 运算规则简单：二进制的和、积运算组合分别只有3种规则
- ✓ 适合逻辑运算：逻辑的“真”、“假”对应“1”和“0”
- ✓ 易于进行转换：与十进制、八进制、十六进制转化方便
- ✓ 抗干扰能力强：适合于电脑储存

* 关于数制转化的讨论

➢ 八进制、十六进制

- ✓ 分别对应3、4位二进制数的组合，方便与二进制数进行转换；
- ✓ 八进制计数在早期计算机中常见，适用于12位和36位的计算机系统（或者其他位数为3的倍数的计算机系统）；
- ✓ 十六进制则更适用于8位、16位、32位、64位（位数为2的幂）的计算机系统；
- ✓ 相较二进制数，长度缩短，可以方便的读写。

中国古代：1斤=16两，半斤八两

国外：1磅=16盎司

* 关于数制转化的讨论

➢ 十二进制

- ✓ 在计算机中使用较少，在生活中有较多应用
- ✓ 钟表一圈为12小时，一年有12个月，24个节气
- ✓ 1英尺=12英寸

➢ 六十进制

- ✓ 在计算机中使用较少，在生活中有较多应用，尤其是与时间有关的问题
- ✓ 1小时=60分钟，1分钟=60秒，1甲子=60年
- ✓ 圆周为360度

（2）浮点数类型

- 在MATLAB中，浮点数包括单精度浮点数（single）和双精度浮点数（double）
- 双精度浮点采用8个字节，即64位来表示，

  第63位表示符号，0为正，1为负

  第52-62位表示指数部分

  第0-51位表示小数部分。
- 单精度浮点数采用4个字节，即32位来表示

  第31位为符号位，0为正，1为负，

  第23-30位为指数部分

  0-22位为小数部分

（2）浮点数类型

- 双精度浮点数是MATLAB中默认的数据类型。
- 单精度浮点数比双精度浮点数能够表示的数值范围和数值精度都小。

名称	存储空间	表示范围	转换函数
单精度浮点数	4字节	$-3.40282\times10^{38}\sim-1.17549\times10^{-38}$ $1.17549\times10^{-38}\sim3.40282\times10^{38}$	single()
双精度浮点数	8字节	$-1.79769\times10^{308}\sim-2.22507\times10^{-308}$ $2.22507\times10^{-308}\sim1.79769\times10^{308}$	double()

补充知识：关于计算机存储

什么是位、字节、字、KB、MB

- 位：位(bit)"是电子计算机中最小的数据单位。每一位的状态只能是0或1。
- 字节：8个二进制位构成1个"字节(Byte)"，它是存储空间的基本计量单位。1个字节可以储存1个英文字母或者半个汉字，换句话说，1个汉字占据2个字节的存储空间。
- 字：由若干个字节构成，字的位数叫做字长，不同档次的机器有不同的字长。例如一台8位机，它的1个字就等于1个字节，字长为8位。如果是一台16位机，那么，它的1个字就由2个字节构成，字长为16位。字是计算机进行数据处理和运算的单位。

补充知识：关于计算机存储

什么是位、字节、字、KB、MB

- KB：K一般表示1000(kilo-)，在计算机中K表示1024，也就是2^{10}。1KB表示1K个Byte，也就是1024个字节。
- MB：M(兆)一般表示是10^6(mega) 。在二进制中，MB也表示到了百万级的数量级，但1MB不正好等于1000000字节，而是1048576字节，即 1MB = 2^{20} Bytes = 1048576Bytes。

1KB=1024B，1MB=1024KB，

1GB=1024MB， 1TB=1024GB，

PB、EB、ZB、YB、BB、NB、DB (相差2^{10}=1024倍)

（3）复数类型

- 复数是对实数的扩展，包含实部和虚部两部分，虚部的单位是-1的平方根。
- 在MATLAB中，采用i或j表示虚部的单位。
- 可以采用赋值语句直接产生复数，也可以采用函数complex( )产生复数。

$$i/j=\sqrt{-1}$$

（3）复数类型

函 数	说 明
complex(a, b)	创建复数，a为实部，b为虚部
real(z)	得到复数z的实部
imag(z)	得到复数z的虚部
abs(z)	得到复数z的模
angle(z)	得到复数z的角度
conj(z)	得到复数z的共轭复数

（4）Inf和NaN

- Inf和-Inf 分别表示正无穷大和负无穷大。除法运算中除数为0或者运算结果溢出都会导致inf或-inf的运行结果。
- 用NaN（Not a Number）

  来表示一个既不是实数也不是复数的数值。

2.1.2 逻辑类型

- 在MATLAB中逻辑类型包含true和false，分别由1和0表示。
- 在MATLAB中用函数logical( )将任何非零的数值转换为true（即1），将数值0转换为false（即0）。

2.1.3 字符和字符串类型

- 在MATLAB中，字符型数据类型用**char**表示。
- 字符和字符串不进行区分，将单个字符也看成字符串，都用单引号（'）括起来。
- 字符串中的每个字符占用2个字节的存储空间。
- 字符串和字符串处理函数：包括字符串的比较、查找和替换等等。

2.1.4 单元数组类型

单元数组和结构体数组的特点：

允许用户将不同但是相关的数据类型集成到一个单一的变量

单元数组每个元素都以单元的形式存在。

采用大括号（{ }）建立单元数组，也可以采用函数cell( )来建立单元数组。

在获取单元数组的元素时，也采用大括号表示下标。

2.1.4 单元数组类型

单元数组的生成

A、直接生成

```
>> a={1 , 'wind gone' , 100+200i , [90 85 33 ; 87 49 293 ; 32 23 299 ; -200 89 87]}
a =
    [1]    'wind gone'    [1.0000e+002 +2.0000e+002i]    [4x3 double]
```

B、使用cell函数生成单元数组

cell(M，N)生成一个M×N的置空单元数组

```
>> b=cell(2 , 2)
b =
    []    []
    []    []
```

```
b{1,1}=[34]
b =
    [34]    []
    []    []
```

2.1.4 单元数组类型

单元数组的操作

单元数组内容的显示

A、celldisp(C, ‘name’)函数

显示单元数组C的内容，其显示的变量名称为name

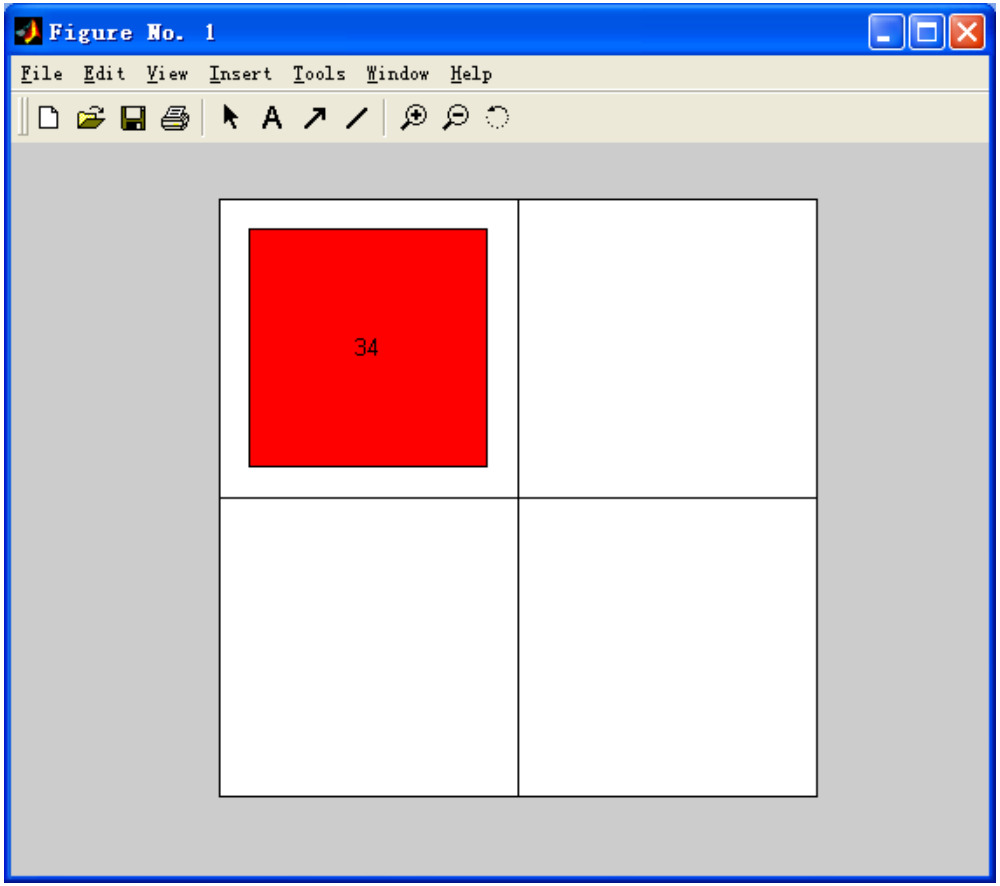
```
 b{1 , 1}=[34]
b =
    [34]     []
    []     []
>> celldisp(b , 'ww')
ww{1 , 1} =        ww{ 1, 2} =
     34                    []

ww{ 2, 1} =       ww{2 , 2} =
         []              []
```

2.1.4 单元数组类型

B、cellplot( )

使用彩色图形来显示单元型变量的结构形式

>> cellplot(b)



2.1.4 单元数组类型

cell 1,1

3	4	2
9	7	6
8	5	1

cell 1,2

'Anne Smith'
'9/12/94 '
'Class II '
'Obs. 1 '
'Obs. 2 '

cell 1,3

.25+3i	8-16i
34+5i	7+.92i

cell 2,1

[1.43 2.98 5.67]

cell 2,2

7	2	14
8	3	45
52	16	3

cell 2,3

'text'	4 2 1 5
[4 2 7]	.02 + 8i

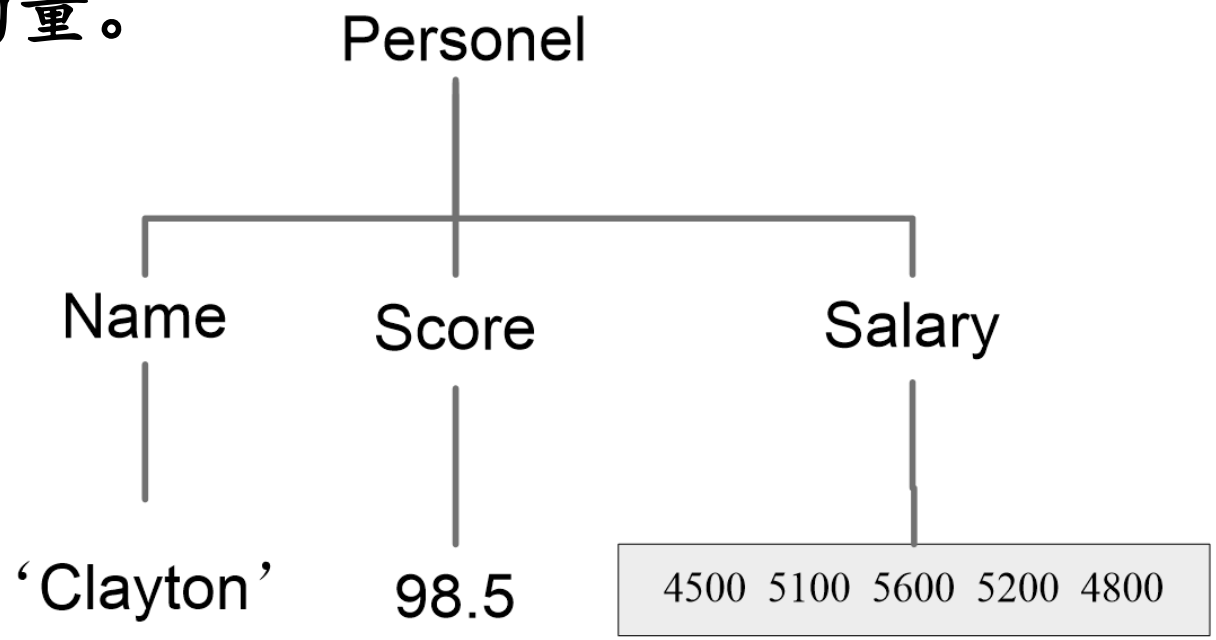
2.1.5 结构体类型

结构体类型是一种由若干属性（field）组成的MATLAB数组，其中的每个属性可以是任意数据类型。

结构体类似于C语言的结构体数据。每个成员变量用指针操作符“.”表示，例如A.name表示结构体变量A的name成员变量

2.1.5 结构体类型

下图表示了一个结构体（Personel），它包括3个属性（Name、Score和Salary），其中Name是一个字符串，Score是一个数值，Salary是一个1×5的向量。



2.1.5 结构体类型

- struct( ) 产生结构体变量；
- rmfield( ) 删除结构体中的成员变量；
- isstruct( ) 判断是否为结构体变量；
- isfield( ) 判断是否为结构体变量的成员变量;
- fieldnames( ) 获得结构体的成员变量名字；
- orderfields(S) 按照成员变量的字母顺序排序;
- getfield( ) 取得存储在结构体中的成员变量的值；
- setfield( ) 用于给结构体的成员变量设置新值；
- struct2cell( ) 将结构体变量转换为单元数组 。

2.1.5 结构体类型

结构型变量的生成

A、直接输入法

```
>> student.test=[3 6 6 67 77 4];
>> student.name='Huang ming';
>> student.weight=67;
>> student.height=1.68;
>> student.num=05322;
>> student
```

```
student =
    test: [3 6 6 67 77 4]
    name: 'Huang ming'
    weight: 67
    height: 1.6800
    num: 5322
```

2.1.5 结构体类型

B、使用struct函数生成结构型变量

```
>> struct_array = struct('countrys', {{'china', 'american'}},
    'strengths', [10000 9000])
struct_array =
    countrys: {'china'  'american'}
    strengths: [10000 9000]
```

注意struct输入的格式
小括号,中括号,大括号

2.1.5 结构体类型

结构型变量的操作

A、添加

>> student(1).gender='male';

给结构变量student第一组成员增加gender类，并赋值

B、删除　使用rmfield函数

```
>> student
student =
      test: [3 6 6 67 77 4]
      name: 'Huang ming'
    weight: 67
    height: 1.6800
       num: 5322
    gender: 'male'
```

```
>> student=rmfield(student,'gender')
student =
      test: [3 6 6 67 77 4]
      name: 'Huang ming'
    weight: 67
    height: 1.6800
       num: 5322
```

2.1.6 函数句柄类型

- 在MATLAB中，通过函数句柄来间接调用函数，函数句柄的数据类型为function_handle。
- 函数句柄可以通过符号@后面跟函数命令来创建

例如：程序fhandle=@cos，创建了函数cos( )的函数句柄，以后就可以通过函数句柄fhandle来间接调用函数cos( )。

2.1.6 函数句柄类型

引入函数句柄是为了：

- 使泛函指令工作更可靠；
- 反复调用情况下更显效率；
- 使“函数调用”像“变量调用”一样方便灵活；
- 提高函数调用速度，提高软件重用性，扩大子函数和私用函数的可调用范围；
- 迅速获得同名重载函数的位置、类型信息。

MATLAB中函数句柄的使用使得函数也可以成为输入变量，并且能很方便的调用，提高函数的可用性和独立性。

2.2 运算符和常量、变量

2.2.1 算数运算符

2.2.2 关系运算符

2.2.3 逻辑运算符

2.2.4 运算优先级

2.2.5 常量和变量

2.2.1 算数运算符

算数运算符的用法和功能如下表所示

运算符	用法	功能描述
+	A+B+A	一元运算符正号； 加法，把矩阵A和B相加； A、B必须是具有相同大小的矩阵，或者其中之一为标量，标量可以和任何一个矩阵相加。
-	A-B-A	一元运算符负号； 减法，把矩阵A和B相减； A、B必须是具有相同大小的矩阵，或者其中之一为标量，标量可以和任何一个矩阵相减。
.*	A.*B	元素相乘，把A和B对应的元素相乘； A、B必须是具有相同大小的矩阵，或者其中之一为标量，标量可以和任何一个矩阵相乘

运算符	用法	功能描述
./	A./B	元素右除法，矩阵A除以矩阵B的对应元素； A、B必须是具有相同大小的矩阵，或者其中之一为标量，标量可以和任何一个矩阵相除。
.\	A.\B	元素左除法，矩阵B除以矩阵A的对应元素； A、B必须是具有相同大小的矩阵，或者其中之一为标量，标量可以和任何一个矩阵相除。
.^	A.^B	元素的乘方，矩阵A的元素做矩阵B中对应元素的乘方； A、B必须是具有相同大小的矩阵，或者其中之一为标量，标量可以和任何一个矩阵乘方
.’	A.’	矩阵转秩（虚部不变号）

运算符	用法	功能描述
*	A*B	矩阵乘法，对于非标量的矩阵A和B，矩阵A的列长度必须和矩阵B的行长度一致； 标量可以和任何一个矩阵相乘。
/	A/B	矩阵右除法，粗略的相当于B*inv(A)，准确的说相当于(A'\B')'； 方程X*A=B的解
\	A\B	矩阵左除法，方程A*X=B的解
^	A^B	矩阵乘方，具体用法见后
'	A'	矩阵共轭转秩（虚部变号）

算术运算符

标量和数组运算	矩阵运算	功　能
+	+	加
-	-	减
.*	*	乘
./	/	左除
.\	\	右除
.^	^	乘方
.’	.’	转置
’	’	共轭转置

补充说明A^B的用法如下：

- 当A和B都为矩阵时，此运算无定义；
- 当A和B都是标量时，表示标量A的B次幂；
- 当A是标量且B为矩阵时，计算中涉及与A和B相关的特征值和特征向量；
- 当A为方阵且B为正整数时，表示矩阵A的B次乘积；

- 当**A**为方阵且**B**为负整数时，表示矩阵**A**逆的负**B**次乘积；
- 当**A**为可对角化的方阵且**B**为非整数时，有如下表达式：

$$A\hat{}B = V\begin{bmatrix} \lambda_1{}^{B} & & \\ & \ddots & \\ & & \lambda_n{}^{B} \end{bmatrix} V^{-1}$$

A=magic(3)，B=[1 2 3; 4 5 6; 7 8 9],

计算两个矩阵的元素乘积和矩阵乘积

A=magic(3)

```
A=   8   1   6
     3   5   7
     4   9   2
```

C=A.*B %矩阵元素乘积

```
C=   8    2   18
    12   25   42
    28   72   18
```

B=[1 2 3; 4 5 6; 7 8 9]

```
B=   1   2   3
     4   5   6
     7   8   9
```

D=A*B % 矩阵乘积

```
D=   54   69   84
     72   87  102
     54   69   84
```

例：计算3/B，具体代码如下：

```
B=[1 2 3; 4 5 6; 7 8 9];
C=3./B
```

运行结果如下：

```
C=
    3.0000    1.5000    1.0000
    0.7500    0.6000    0.5000
    0.4286    0.3750    0.3333
```

例：计算A.’，具体代码如下：

```
A=magic(3);
C=A.’
```

运行结果如下：

```
C=
     8     3     4
     1     5     9
     6     7     2
```

2.3.2 关系运算符

关系运算符的用法和功能如下表所示

运算符	说明
<	小于
<=	小于等于
>	大于
>=	大于等于
==	等于
~=	不等于

注意：关系运算符只针对两个相同长度的矩阵，或其中之一是标量的情况进行运算。

- 对于两个相同长度的矩阵，是指两个矩阵的对应元素进行比较，返回具有相同长度的矩阵；
- 对于比较双方之一为标量的情况，是指这个标量与另一个矩阵的每元素进行运算。

关系运算的结果只有0和1两种情况：

- ✓ 0表示不满足条件
- ✓ 1表示满足条件

例：显示矩阵A=magic(3)中哪些元素值大于4，具体代码如下：

```
A=magic(3)
A=
     8     1     6
     3     5     7
     4     9     2
```

```
C=A>B
C=
     1     0     1
     0     1     1
     0     1     0
```

```
B=ones(3)*4
B=
     4     4     4
     4     4     4
     4     4     4
```

或者采用代码：

```
C=A>4
```

2.3.3 逻辑运算符

元素方式和比特方式等逻辑运算符。

元素方式逻辑运算符：

A= [0 1 1 0 1];
B= [1 1 0 0 1];

运算符	功能描述	例子
逻辑与 (&)	两个操作数同时为1，运算结果为1；否则为0；	A&B=01001
逻辑或 (|)	两个操作数同时为0，运算结果为0；否则为1；	A|B=11101
逻辑非 (~)	操作数为0时，运算结果为1； 操作数为1时，运算结果为0；	~A=10010
逻辑异或 (xor)	两个操作数相同时，运算结果为0； 两个操作数不同时，运算结果为1；	xor(A,B)=10100

- 元素方式逻辑运算符**&** 、**|**和**~**与函数**and()**、**or()**和**not()**是等价的
- 比特方式逻辑运算符只接受逻辑和非负整数类型的输入变量，它是针对输入变量的二进制进行逻辑运算

比特方式逻辑运算符的用法和功能如下表所示，

A = 28　　11100

B = 200　　11001000

函数名	功能描述	例子
bitand	位与，两个非负整数的对应位与操作	bitand(A, B)=8 (binary 1000)
bitor	位或，两个非负整数的对应位或操作	bitor(A, B)=220 (binary 11011100)
bitcmp	位补，指定位数（不少于输入变量二进制的最大位数）的补操作	bitcmp(A,5)=3 (binary 00011) bitcmp(A,7)=99 (binary 1100011)
bitxor	位异或，两个非负整数的对应位异或操作	bitxor(A,B)=212 (binary 11010100)

2.3.4 运算优先级

- 运算符的优先级决定表达式求值顺序；
- 具有相同优先级的运算符从左到右依次进行运算；
- 不同优先级的运算符采用先进行优先高的运算。

运算符	优先等级
括号	最高运算级
转秩(.'), 幂(.^), 复共轭转秩('), 矩阵幂(^)	
一元正号(+), 一元负号(-), 逻辑非(~)	
元素相乘(.*), 元素右除(./), 元素左除(.\) 矩阵相乘(*), 矩阵右除(/), 矩阵左除(\)	
加法(+), 减法(-)	
冒号运算符(:)	
小于(<), 小于等于(<=), 大于(>), 大于等于(>=), 等于(=), 不等于(~=)	
逻辑与(&)	
逻辑或(|)	最低运算级

依次下降

例：计算C=A./B.^2，其中A=[3 9 5], B=[2 1 5]
具体代码如下：

```
A= [3  9  5];
B= [2  1  5];
C= A./B.^2
```

运行结果如下：

```
C=
        0.7500 9.0000 0.20000
```

先运行B.^2

2.1.5 常量和变量

常量	说明
ANS或ans	默认变量名，保存最后一次的运算结果
Pi	圆周率
INF或inf	无穷大
NaN或nan	不定值 0/0
bitmax	最大的正整数
realmax	最大的正实数
realmin	最小的正实数
eps	浮点数的相对误差
i或j	复数单位，-1的平方根
nargin	函数的输入参数个数
nargout	函数的输出参数个数
varargin	可变的输入参数个数
varargout	可变的输出参数个数

nargin是用来判断输入变量个数的函数，这样就可以针对不同的情况执行不同的功能。

例子：函数test1的功能是输出a和b的和。如果只输入一个变量，则认为另一个变量为0，如果两个变量都没有输入，则默认两者均为0。

```
function y=test1(a,b)
if nargin==0
    a=0;b=0;
elseif nargin==1
    b=0;
end
y=a+b;
```

2.1.5 常量和变量

在MATLAB中，对变量的命名规则：

1. 变量名的长度不超过31个字符，超过的字符，系统将忽略不计。
2. 变量名区分大小写，例如变量名var1和Var1是不同的。
3. 变量名必须以字母开头，变量名中可以包含字母、数字和下划线。

2.3 数组和矩阵分析

MATLAB语言最基本和最重要的功能就是进行矩阵运算，所有的数值功能都以矩阵为基本单元来实现。本节将对数组和矩阵及其运算进行详细的介绍。

2.3 数组和矩阵分析

2.3.1 数组及其函数

2.3.2 矩阵的创建

2.3.3 矩阵的基本操作

2.3.4 矩阵的基本数值运算

2.3.5 特殊矩阵的生成

2.3.6 矩阵的特征值和线性代数

2.3.7 稀疏矩阵

2.3.8 矩阵的分解

2.3.1 数组及其函数

下面介绍MATLAB中如何建立数组，以及数组的常用操作等，包括数组的算术运算、关系运算和逻辑运算，以及数组信息的获取等。

- 一维数组相当于向量
- 二维数组相当于矩阵

矩阵是数组的子集

(1) 数组的建立和操作

- 在MATLAB中，一般使用方括号（[ ]）、逗号（,）、空格及分号（;）来创建数组。
- 数组中同一行的元素之间用逗号或空格进行分割，不同行之间用分号进行分割。
- 需要注意，这些符号都必须在英文输入状态下输入。

(1) 数组的建立和操作

- 空数组是MATLAB中最特殊的数组，不含有任何元素，可以用于数组的声明或者清空等。创建空数组非常简单，只要把变量赋值为一对方括号即可。([ ])
- 数组是有方向的，一维数组包括行向量和列向量，行向量是以行方向分布的，列向量是以列方向分布的。
- 创建一维行向量，把所有用空格或逗号分割的元素用方括号括起来。
- 创建一维列向量，把所有用分号分割的元素用方括号括起来。

(2) 数组的算术运算

- 数组运算是从数组的单个元素出发，针对每个元素进行的运算。在MATLAB中，一维数组的基本算术运算有：＋（加）、－（减）、.*（乘）、./（左除）、.\（右除）和^（乘方）等。

数组的加减运算规则：

- 若数组A和B的维数相同，则可以执行加减运算，相应元素相加减。
- 如果A和B的维数不相同，则MATLAB将给出错误信息，提示用户两个数组的维数不匹配。

(2) 数组的算术运算

- 数组的乘法和除法分别用“.*”和“./”表示。如果数组A和B具有相同的维数，则数组的乘法表示数组A和B中对应的元素相乘，数组的除法表示数组A和B中对应的元素相除。

- 右除和左除的关系为：A./B=B.\A，

  其中A是被除数，B是除数。

(3) 数组的关系运算

- 当参与比较的量是两个维数相同的数组时，比较两数组相同位置的元素，并给出比较结果。最终的关系运算的结果是一个维数与原矩阵相同的数组，由0或1组成。
- 当参与比较的一个是标量，而另一个是数组时，则把标量与数组的每一个元素逐个比较，最终的关系运算的结果是一个维数与原数组相同的数组。

(4) 数组的逻辑运算

- 提供了3种逻辑运算符，

  &（逻辑与）

  |（逻辑或）

  ～（逻辑非）

- 在逻辑运算中，

  非零元素则为逻辑真，用1表示

  零元素为逻辑假，用0表示。

(5) 数组信息的获取

下面介绍如何获取数组的信息，

数组大小

维度

数据类型

内存占用

数组的元素查找和排序

(5) 数组信息的获取

“is……”

- isempty(A)：该函数检测数组是否为空，如果为空，返回值为1，否则，返回值为0。
- isscalar(A): 检测数组是否为单个元素的标量。
- isvector(A): 检测数组是否为行向量或列向量。
- isrow(A): 检测数组是否为行向量。
- iscolumn(A): 检测数组是否为列向量。
- issparse(A): 检测数组是否为稀疏矩阵。

(5) 数组信息的获取

```
V = rand(1,5);
iscolumn(V)
ans =
0

V1 = V';
iscolumn(V1)
ans =
1
```

(5) 数组信息的获取

最常用的检测数组大小的函数是size( )和length( )。

- size( ) 获取数组的行数和列数，
- length( ) 获取一维数组的长度，如果是二维数组，则返回行数和列数中的较大者 。
- ndims( ) 计算数组的维度 。

(5) 数组信息的获取

- 采用函数**whos**来获取数组的大小，以及占用内存的多少。对于数组中不同的数据类型，占用的内存也不一样。
- 数组元素的查找采用函数**find( )**，返回关系表达式为真的元素的下标。
- 数组的排序使用函数**sort( )**，该函数默认按照升序排列，返回值为排序后的数组，和原数组维数相同。

```
>> a=magic(3)
a =
     8     1     6
     3     5     7
     4     9     2

>> find(a)
ans =
     1
     2
     3
     4
     5
     6
     7
     8
     9
```

```
>> a=magic(3);
>> b=a-3
b =
     5    -2     3
     0     2     4
     1     6    -1

>> find(b)
ans =
     1
     3
     4
     5
     6
     7
     8
     9
```

```
>> a=rand(3)

a =

    0.9501    0.4860    0.4565
    0.2311    0.8913    0.0185
    0.6068    0.7621    0.8214

>> b=sort(a)

b =

    0.2311    0.4860    0.0185
    0.6068    0.7621    0.4565
    0.9501    0.8913    0.8214
```

默认按列排序！！！

2.3.2 矩阵的创建

矩阵的创建有多种方式：

(1) 在命令窗口中直接输入矩阵，比较适合创建比较小的矩阵。

把矩阵的元素放到方括号里面，每行的元素用空格或逗号分割，每列用分号分割。

- 注意：每行的元素数必须相等，每列的元素数也必须相等。

2.3.2 矩阵的创建

(2) 通过语句和函数生成矩阵，

例如函数eye( )用于生成单位矩阵。

(3) 通过M文件来建立矩阵，或从外部数据文件中导入矩阵，

例如通过函数imread( )读取图片，从而得到图像数据的二维矩阵。

2.3.2 矩阵的创建

- 矩阵的元素按照列进行保存，先第一列，再第二列，直到结束。
- 矩阵中的元素可以采用单下标获取，也可以采用双下标获取。单下标和双下标之间，可以通过函数进行转换。
- 在程序中，对矩阵中的元素进行赋值。如果行或者列超出矩阵的大小，则MATLAB自动扩充矩阵的大小，然后再进行赋值，扩充部分用零填充。

2.3.3 矩阵的基本操作

在MATLAB中，矩阵是基本的计算单元，有很多关于矩阵操作的函数。

- 矩阵的扩展
- 块操作
- 矩阵元素删除
- 转置
- 旋转
- 翻转
- 改变矩阵的大小

(1) 矩阵的扩展

可以通过数组的扩展，将多个小矩阵转换为大的矩阵。（catenate）

- C=cat(DIM, A, B)：该函数在DIM维度上进行矩阵A和B的连接，返回值为连接后的矩阵。
- C=vertcat(A, B)：该函数在水平方向上连接数组A和B，相当于cat(1, A, B)。
- C=horzcat(A, B)：该函数在垂直方向上连接数组A和B，相当于cat(2, A, B)。

```
A= [1 3; 5 7];
B= [2 4; 6 8];
cat(1, A, B)
```

结果显示为：

ans=

1	3
5	7
2	4
6	8

```
A= [1 3; 5 7];
B= [2 4; 6 8];
cat(2, A, B)
```

结果显示为：

ans=

1	3	2	4
5	7	6	8

(2) 矩阵的块操作

通过函数repmat( )进行数据块的复制，(Replicate Matrix)

- B=repmat(A, m, n)：该函数产生大的矩阵B，把矩阵A当作单个元素，产生由m行和n列的矩阵A组成的大矩阵B。
- B=repmat(A, m)：该函数产生大的矩阵B，把矩阵A当作单个元素，产生m行和m列的矩阵A组成的大矩阵B。

A= [1 2; 3 4];

B= repmat(A, 2, 1)

结果显示为：

ans=

1	2
3	4

1	2
3	4

(2) 矩阵的块操作

采用函数blkdiag( )将多个矩阵作为对角块，产生新的矩阵。(block diagonal)

- Y=blkdiag(A, B)：该函数将矩阵A和B作为对角块，产生新的矩阵Y。
- Y=blkdiag(A, B, …)：该函数将多个矩阵作为对角块，产生新的矩阵。

```
a=1;
b=[2,2;3,3];
c=[4,4;5,5;6,6];
d=8;
out = blkdiag(a,b,c,d)
```

结果为

```
out =

     1     0     0     0     0     0
     0     2     2     0     0     0
     0     3     3     0     0     0
     0     0     0     4     4     0
     0     0     0     5     5     0
     0     0     0     6     6     0
     0     0     0     0     0     8
```

(2) 矩阵的块操作

- 采用函数kron( )进行矩阵的块操作

  (Kronecker，德国数学家与逻辑学家)

- Y=kron(A, B)：将矩阵B和A的各个元素相乘，得到更高阶的矩阵

$$\begin{bmatrix} A(1,1)*B & A(1,2)*B & A(1,3)*B \\ A(2,1)*B & A(2,2)*B & A(2,3)*B \end{bmatrix}$$

```
X = [1 2;3 4;5 6];
Y = [2 4;6 8];
kron(X,Y)
```

结果为：

```
ans =
```

X(1,1) *Y →	2	4	4	8
	6	8	12	16
X(2,1) *Y →	6	12	8	16
	18	24	24	32
	10	20	12	24
	30	40	36	48

(3) 矩阵中元素的删除

- 利用空矩阵删除矩阵元素。
- 空矩阵为一对方括号 [ ]。矩阵赋值为空矩阵的语句为X=[ ]。

- 注意：X=[ ]与clear X不同，clear是将X从工作空间中删除，而空矩阵则存在于工作空间中，只是维数为0。

(4) 矩阵的转置

- 矩阵的转置命令为**ctranspose**，或采用转置操作符（’）。如果矩阵中含有复数，则进行矩阵转置后，复数转化为共轭复数。
- 矩阵的真正转置为**A.’**，即使为复数，也不转换为共轭。也可以采用函数**transpose(A)**来实现，两者完全一致。

(5) 矩阵的旋转

矩阵的旋转可以采用转置的方法，也可以采用函数rot90( ), (rotate)

- B=rot90(A)：该函数将矩阵逆时针旋转90度。
- B=rot90(A, k)：该函数将矩阵逆时针旋转90度的k倍，k的默认值为1。

(6) 矩阵的翻转

- 对矩阵实施左右翻转是将原矩阵的第一列和最后一列调换，第二列和倒数第二列调换，依次类推。在MATLAB中，对矩阵进行左右翻转的函数是**fliplr(A)**，实现矩阵的左右翻转。
- 对矩阵进行上下翻转是将原矩阵的第一行和最后一行调换，第二行和倒数第二行调换，依次类推。在MATLAB中，对矩阵进行上下翻转的函数是**flipud(A)**，实现矩阵的上下翻转。

(6) 矩阵的翻转

还可以采用函数flipdim( )进行矩阵的翻转，该函数的调用格式为：

- **flipdim(A, k)**，该函数在指定的方向k进行矩阵的翻转。

  当k=1时，相当于flipud(A)，

  当k=2时，相当于fliplr(A)。

```
>> a=[1 2 3; 4 5 6; 7 8 9]

a =

     1     2     3
     4     5     6
     7     8     9

>> b=fliplr(a)

b =

     3     2     1
     6     5     4
     9     8     7

>> c=flipud(a)

c =

     7     8     9
     4     5     6
     1     2     3
```

(7) 矩阵尺寸的改变

在矩阵总元素保持不变的前提下，用函数reshape()改变矩阵的尺寸：

- Y=reshape(X, m, n)，将矩阵转换为m行n列的二维矩阵。矩阵的总元素数不变。

把一个3x4的矩阵变形为一个2x6的矩阵：

A=

1	4	7	10
2	5	8	11
3	6	9	12

B=reshape(A, 2,6)

B=

1	3	5	7	9	11
2	4	6	8	10	12

2.3.4 矩阵的基本数值运算

矩阵的基本数值计算包括：

- 矩阵的加法、减法、乘法和除法
- 矩阵元素的查找、排序、求和、求积

(1) 矩阵的加减运算

- 假定有两个矩阵A和B，则可以由A+B和A-B实现矩阵的加减运算，要求矩阵A和B的维数必须相同。
- 矩阵的加法和减法是矩阵中对应元素加减。
- 如果其中有一个为标量，则将矩阵中的每一个元素和该标量进行加减运算。

(2) 矩阵的乘法

- 矩阵A和B的乘法为A*B，要求矩阵A的列数和矩阵B的行数必须相等。
- 矩阵A和B的点乘为A.*B，表示矩阵A和B中对应元素相乘，要求矩阵A和B具有相同的维数，返回结果和原矩阵有相同的维数。如果A和B的维数不满足要求，系统会给出出错信息，提示两个数组的维数不匹配。

(3) 矩阵的除法

矩阵的除法有左除和右除两种，分别用“\”和“/”表示。通常矩阵的除法用来求解方程组的解。

- 矩阵A和B的左除为X=A\B，表示方程组A*X=B的解。
- 矩阵A和B的右除为X=B/A，表示方程组X*A=B的解。
- 如果X不存在或不唯一，则系统显示警告信息。
- 此外，还有矩阵的点除，采用“./”或“.\”表示，表示两个矩阵中对应元素相除。

(4) 矩阵元素的查找

采用函数find( )进行矩阵元素的查找。函数find( )通常和关系运算、逻辑运算相结合，能够对矩阵中的元素进行查找：

- **i=find(A)**：该函数查找矩阵中的非零元素，函数返回这些元素的单下标。
- **[i , j]=find(A)**：该函数查找矩阵中的非零元素，函数返回这些元素的双下标。

```
x=    [3    0    0
       0    0    0
       0    6    0];
[i, j]=find(x)
```

结果为

```
i=
     1
     3
j=
     1
     2
```

```
k=find(x)

k=???

k=

     1

     6
```

(5) 矩阵元素的排序

矩阵元素的排序使用函数sort( )，该函数默认按照升序排列，返回排序后的矩阵。

- **Y=sort(X)**：该函数对矩阵按照升序进行排列。当X为向量时，返回由小到大排序后的向量；当X为矩阵时，返回X中各列按照由小到大排序后的矩阵。

(5) 矩阵元素的排序

- **Y=sort(X, DIM)**：该函数返回在给定的维数上的按照由小到大的顺序排序后的结果，

  当DIM=1时，按照列进行排序，

  当DIM=2时，按照行进行排序。

- **Y=sort(X, DIM, 'MODE')**：该函数可以指定排序的方式。

  MODE默认值为‘ascend’，即按照升序进行排列；

  MODE为‘descend’时，对矩阵进行降序排列。

思考：如果没有descend选项，如何进行降序排列？？？

A=

3	5	9
2	5	6
14	9	7

B=sort(A)　%对A中各个列向量元素进行升序排列

B=

2	5	6
3	5	7
14	9	9

(6) 矩阵元素的求和

- 进行矩阵中元素求和时采用函数sum( )：
- **Y=sum(X)**：该函数对矩阵X的元素求和，返回矩阵中各列元素的和组成的向量。
- **Y=sum(X, DIM)**：该函数返回在给定的维数DIM上的元素的和，

  DIM=1时，计算矩阵各列元素的和

  DIM=2时，得到矩阵各行元素的和。

(7) 矩阵元素的求积

- 进行矩阵中元素求积时采用函数prod( )：
- **Y=prod(X)**：该函数对矩阵的元素求积，返回矩阵X中各列元素的积组成的向量。
- **Y=prod(X, DIM)**：该函数返回在给定的维数上的元素的积，

  DIM=1时，计算矩阵各列元素的积，

  DIM=2时，得到矩阵各行元素的积。

M=

8	1	6
3	5	7
4	9	2

B=prod(M)

B=

96	45	84

B=prod(M, 2)

B=

48
105
72

(8) 矩阵元素的差分

- 利用函数diff( )计算矩阵的差分：
- Y=diff(X)：该函数计算矩阵各列的差分。
- Y=diff(X, N)：该函数计算矩阵各列的N阶差分。
- Y=diff(X, N, DIM)：该函数计算矩阵在方向DIM上的N阶差分。

  DIM=1时，计算矩阵各列元素的差分，

  DIM=2时，得到矩阵各行元素的差分。

2.3.5 特殊矩阵的生成

在MATLAB中，有许多用于创建矩阵的函数。通过这些函数可以创建二维矩阵，甚至更高维的矩阵。下面对这些产生通用特殊矩阵的函数进行介绍。

(1) 全零矩阵

采用函数zeros( )产生全零矩阵：

- A=zeros(N)：该函数产生N行N列的全零矩阵。
- A=zeros(M, N)：该函数产生M行N列的全零矩阵。
- A=zeros(M, N, P,…)：该函数产生M*N*P*…的全零矩阵。
- A=zeros(size(B))：该函数产生和矩阵B维数相同的全零矩阵。

B=zeros(2, 3)

B=

0	**0**	**0**
0	**0**	**0**

A=[1 2 3; 4 5 6];

B=zeros(size(A))

B=

0	**0**	**0**
0	**0**	**0**

(2) 全1矩阵

采用函数ones( )产生全1矩阵，该函数的调用格式和函数zeros( )基本一致。

(3) 单位矩阵

采用函数eye( )产生单位矩阵：

- A=eye(N)：该函数产生N行N列的单位矩阵。
- A=eye(M, N)：该函数产生M行N列的矩阵，对角线元素为1，其余元素均为0。
- A=eye(size(B))：该函数产生和矩阵B维数相同的单位矩阵。

```
Y=eye(3, 2)
Y=
     1     0
     0     1
     0     0
```

```
A=   [1     2     3
      2     3     4
      3     4     5
      4     5     6]
Y=eye(size(A))

Y=
     1     0     0
     0     1     0
     0     0     1
     0     0     0
```

(4) 0～1间均匀分布的随机矩阵

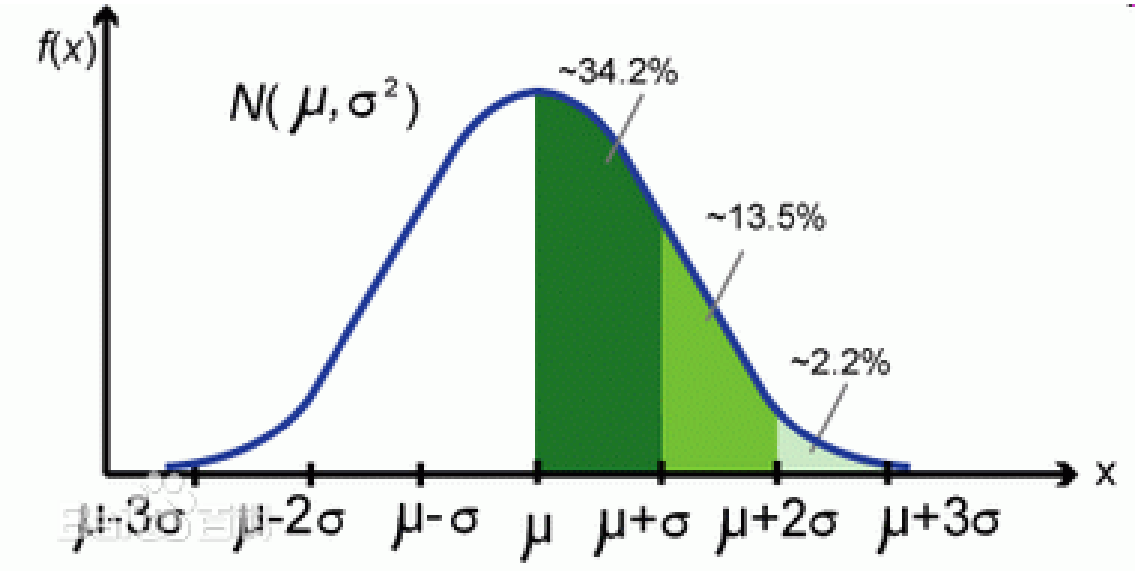
采用函数rand( )产生0~1之间均匀分布的随机矩阵：

- A=rand(N)：该函数产生N行N列的0~1之间均匀分布的随机矩阵。
- A=rand(M, N)：该函数产生M行N列的0~1之间均匀分布的随机矩阵。
- A=rand(M, N, P,…)：该函数产生M*N*P*…的0~1之间均匀分布的随机矩阵。
- A=rand(size(B))：该函数产生和矩阵B维数相同的0~1之间均匀分布的随机矩阵。

```
A=rand(3)

A=
    0.2897    0.7271    0.5681
    0.3412    0.3093    0.3704
    0.5341    0.8385    0.7027

A=[1 2; 3 4; 5 6];
B=rand(size(A))

B=
    0.5226    0.9797
    0.8801    0.2714
    0.1730    0.2523
```

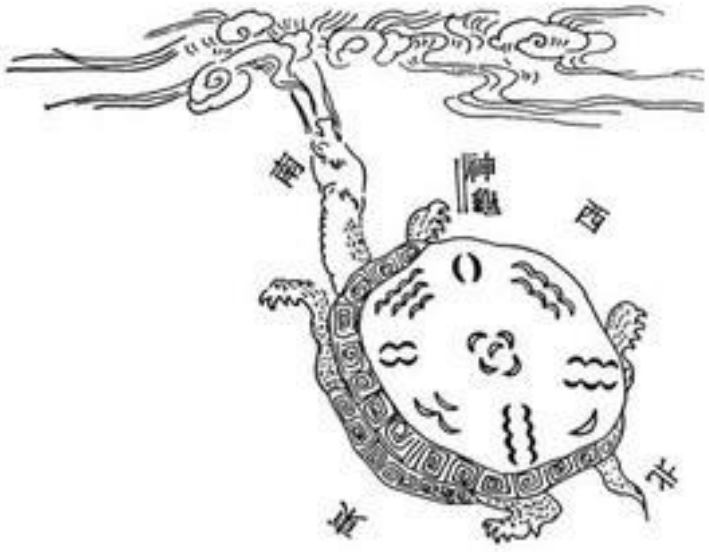
(5) 标准正态分布随机矩阵

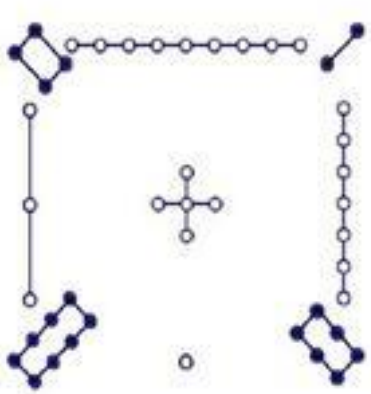
- 采用函数randn( )产生均值为0，方差为1的标准正态分布随机矩阵。该函数的调用格式和函数rand( )基本一致。



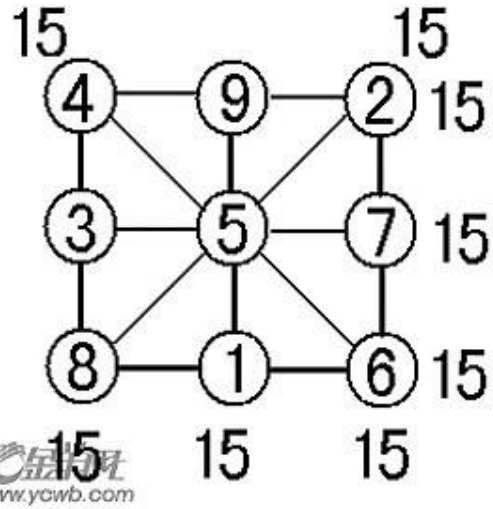
(6) 魔方矩阵

魔方又称幻方、纵横图、九宫图，最早记录于我国古代的洛书。据说夏禹治水时，河南洛阳附近的大河里浮出了一只乌龟，背上有一个很奇怪的图形，古人认为是一种祥瑞，预示着洪水将被夏禹王彻底制服。后人称之为"洛书"或"河图"，又叫河洛图。





洛书



(6) 魔方矩阵

- 魔方矩阵中每行、每列及两条对角线上的元素和都相等。对于n阶魔方阵，其元素由1,2,3,…,n^2组成，共n^2个整数。
- 通过函数**magic(n)**，求n阶魔方矩阵。

```
M=magic(4)

M=
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

(7) 范得蒙矩阵

范德蒙矩阵是法国数学家范德蒙(Vandermonde, AlexandreTheophile, 1735～1796) 提出的一种各列为几何级数的矩阵。可应用于纠错编码等方面。

$$V = \begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & \dots & \alpha_1^{n-1} \\ 1 & \alpha_2 & \alpha_2^2 & \dots & \alpha_2^{n-1} \\ 1 & \alpha_3 & \alpha_3^2 & \dots & \alpha_3^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_m & \alpha_m^2 & \dots & \alpha_m^{n-1} \end{bmatrix}$$

(7) 范得蒙矩阵

- 范得蒙（Vandermonde）矩阵最后一列全为1，倒数第二列为一个指定的向量，其他各列是其后列与倒数第二列的点乘积。可以用一个指定向量生成一个范得蒙矩阵。

- 通过函数vander(V)生成以向量V为基础向量的范得蒙矩阵。

(8) 希尔伯特矩阵

希尔伯特矩阵是一种病态矩阵，矩阵中任何一个元素发生微小的变化，整个矩阵的值和逆矩阵都发生巨大的变化，病态程度和阶数相关。

$$H_{ij} = \frac{1}{i+j-1}. \qquad H = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{bmatrix}.$$

- 在一台相当于10位十进制字长的计算机上对希尔伯特矩阵求逆或解方程组时,如n≥8，则所得解答连一位准确数字都没有

(8) 希尔伯特矩阵

- 通过函数**hilb( )**生成希尔伯特（Hilbert）矩阵。该函数的调用格式为：hilb(n)，产生n阶的希尔伯特矩阵。

- 通过函数函数**invhilb( )**求希尔伯特矩阵的逆矩阵，该函数的调用格式为：invhilb(n)，该函数产生n阶希尔伯特矩阵的逆矩阵。

H=hilb(4)

H=

1.0000	0.5000	0.3333	0.2500
0.5000	0.3333	0.2500	0.2000
0.3000	0.2500	0.2000	0.1667
0.2500	0.2000	0.1667	0.1429

David Hilbert，(1862～1943)，德国著名数学家。

他于1900年在巴黎第二届国际数学家大会上，希尔伯特发表了题为《数学问题》的著名讲演。他根据过去特别是十九世纪数学研究的成果和发展趋势，提出了23个最重要的数学问题。

这23个问题统称希尔伯特问题，对这些问题的研究有力推动了20世纪数学的发展，在世界上产生了深远的影响。

(9) 托普利兹矩阵

托普利兹（Toeplitz）矩阵的斜对角均为等值的常数

$$A=\begin{bmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{-n+1} \\ a_1 & a_0 & a_{-1} & \ddots & & \vdots \\ a_2 & a_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_{-1} & a_{-2} \\ \vdots & & \ddots & a_1 & a_0 & a_{-1} \\ a_{n-1} & \cdots & \cdots & a_2 & a_1 & a_0 \end{bmatrix}$$

(9) 托普利兹矩阵

托普利兹（Toeplitz）矩阵除第一行和第一列外，其它每个元素都与左上角的元素相同。通过函数toeplitz( )生成托普利兹矩阵：

- **toeplitz(x)**：该函数用向量x生成一个对称托普利兹矩阵。
- **toeplitz(x, y)**：该函数产生一个以x为第一列，y为第一行的托普利兹矩阵。x和y均为向量，两者不必等长。需要注意的是，向量x和y的第一个元素必须相等。

T=toeplitz(1:4; 2:5)

T=

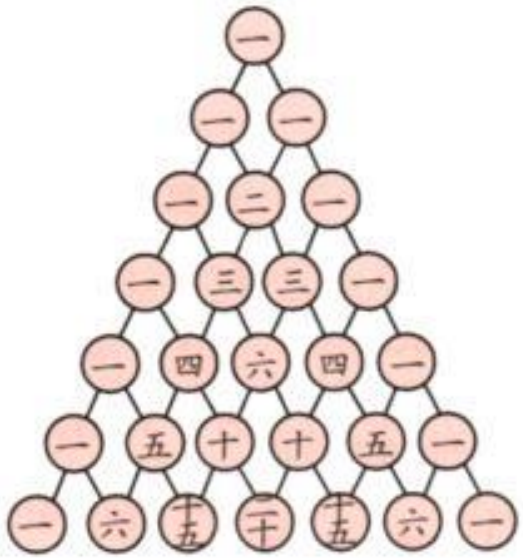
1	3	4	5
3	1	3	4
3	2	1	3
4	3	2	1

(10) 帕斯卡矩阵

二次项展开后的系数随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。

pascal(n)，产生一个n阶的帕斯卡矩阵。

$$S_5 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 3 & 6 & 10 & 15 \\ 1 & 4 & 10 & 20 & 35 \\ 1 & 5 & 15 & 35 & 70 \end{pmatrix}.$$



- 二项式系数

2.3.6 矩阵的特征和线性代数

下面介绍矩阵的一些基本操作，包括矩阵的特征值、三角阵、对角阵、等，以及矩阵的一些特征，例如矩阵的秩、矩阵的迹和矩阵的范数等。

(1) 方阵的行列式

行列式在数学中，是由解线性方程组产生的一种算式。

- 把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。
- 采用函数det( )求方阵的行列式，

$$A=\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \longrightarrow |A|=\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix}$$

```
A=
     1     2     3
     4     5     6
     7     8     9
D=det(A)
D=
     0

A(3,3)=0      %将A中元素数值改变
D=det(A)
D=
    27
```

(2) 特征值、特征向量和特征多项式

设A是n阶矩阵，如果数λ和n维非零列向量x使得关系式 $Ax=\lambda x$ 成立，那么，这样的数λ称为方阵A的特征值，非零向量x称为A对应于特征值λ的特征向量。

对关系式进行变换：$(A-\lambda E)x=0$ 其中E为单位矩阵，这是n个未知数n个方程的齐次线性方程组，它有非零解的充要条件是系数行列式为0，即$|A-\lambda E|=0$。

带入具体的数字或符号，可以看出该式是以λ为未知数的一元n次方程，称为方阵A的特征方程，左端 $|A-\lambda E|$ 是λ的n次多项式，也称为方阵A的特征多项式。

(2) 特征值、特征向量和特征多项式

计算矩阵A的特征值和特征向量的函数是eig( )

- E=eig(A)：求矩阵A的全部特征值，组成向量E。
- [V, D]=eig(A)：该函数计算矩阵的特征值和特征向量，返回值V和D为两个方阵。方阵V的每一列为一个特征向量，方阵D为对角矩阵，对角线上的元素为特征值。

```
>> B=[3 -2 -0.9 0; -2 4 -1 0; 0 0 -1 0; -0.5 -0.5 0.1 1]
B =
    3.0000   -2.0000   -0.9000         0
   -2.0000    4.0000   -1.0000         0
         0         0   -1.0000         0
   -0.5000   -0.5000    0.1000    1.0000
>> d=eig(B)
d =
    1.0000
    5.5616
    1.4384
   -1.0000
```

(3) 对角阵

只有对角线上有非0元素的矩阵称为对角矩阵，

对角线上的元素都为1的对角矩阵称为单位矩阵。

函数diag( ):

- 获取矩阵的对角线元素，
- 生成对角线为某一向量的矩阵，其他位置为0

❖ **diag(A)**：该函数用于提取矩阵A的主对角线元素，产生一个列向量。

❖ **diag(A, k)**：该函数提取第k条对角线的元素，组成一个列向量。

```
>> a=magic(3)
a =
     8     1     6
     3     5     7
     4     9     2
>> b=diag(a)
b =
     8
     5
     2
>> c=diag(a,1)
c =
     1
     7
>> d=diag(a,-2)
d =
     4
```

```
>> v=[1 3 8]
v =
     1     3     8
>> x=diag(v)
x =
     1     0     0
     0     3     0
     0     0     8
>> y=diag(v,1)
y =
     0     1     0     0
     0     0     3     0
     0     0     0     8
     0     0     0     0
```

(4) 上三角阵和下三角阵

- 三角阵可以分为上三角阵和下三角阵，所谓上三角阵，即矩阵的对角线以下的元素全为0的矩阵，而下三角阵则是对角线以上的元素全为0的一种矩阵。
- 由于带三角矩阵的矩阵方程容易求解，在解多元线性方程组时，总是将其系数矩阵通过初等变换化为三角矩阵来求解；又如三角矩阵的行列式就是其对角线上元素的乘积，很容易计算。
- 有鉴于此，在数值分析等分支中三角矩阵十分重要。一个可逆矩阵A可以通过**LU分解**变成一个下三角矩阵L与一个上三角矩阵U的乘积。

(4) 上三角阵和下三角阵

在MATLAB中，通过函数triu( )获取矩阵的上三角矩阵，该函数的调用格式为：

- **triu(A)**：返回矩阵A的上三角矩阵。
- **triu(A, k)**：返回矩阵A的第k条对角线以上的元素。
- 在MATLAB中，采用函数**tril( )**求矩阵的下三角矩阵，该函数的调用格式和函数triu( )完全相同。

(5) 矩阵的逆和伪逆

- 对于方阵A，如果存在一个与其同阶的方阵B，使得A*B=B*A=E，则称A和B互为逆矩阵。采用函数inv( )求方阵的逆矩阵。
- 如果矩阵A不是一个方阵，或者是一个非满秩方阵时，没有逆矩阵，但可以找到一个与A的转置矩阵同型的矩阵B，使得：

$$\begin{cases} A \times B \times A = A \\ B \times A \times B = B \end{cases}$$

(5) 矩阵的逆和伪逆

- 此时，称矩阵B为矩阵A的伪逆，也称为广义逆矩阵。在MATLAB中，求矩阵的广义逆矩阵的函数是**pinv( )**。该函数的调用格式为：pinv(A)，该函数计算矩阵A的广义逆矩阵。

对于任意一个 3x3 矩阵：

X =

1	4	5
8	2	9
1	3	4

在 MATLAB 命令窗口中输入： R = inv (X)

计算结果(即 X 矩阵的逆)为：

R =

19.0000	1.0000	-26.0000
23.0000	1.0000	-31.0000
-22.0000	-1.0000	30.0000

A =

64	2	3	61	60	6
9	55	54	12	13	51
17	47	46	20	21	43
40	26	27	37	36	30
32	34	35	29	28	38
41	23	22	44	45	19
49	15	14	52	53	11
8	58	59	5	4	62

在 MATLAB 命令窗口中输入：B = pinv(A)，可得：

B =

Columns 1 through 8

0.0177	-0.0165	-0.0164	0.0174	0.0173	-0.0161	-0.0160	0.0170
-0.0121	0.0132	0.0130	-0.0114	-0.0112	0.0124	0.0122	-0.0106
-0.0055	0.0064	0.0060	-0.0043	-0.0040	0.0049	0.0045	-0.0028
-0.0020	0.0039	0.0046	-0.0038	-0.0044	0.0064	0.0070	-0.0063
-0.0086	0.0108	0.0115	-0.0109	-0.0117	0.0139	0.0147	-0.0141
0.0142	-0.0140	-0.0149	0.0169	0.0178	-0.0176	-0.0185	0.0205

(6) 矩阵的秩

- 矩阵的秩包括行秩和列秩，矩阵的行向量、列向量组成的极大无关组中行向量的个数。
- 矩阵的秩反映了矩阵中各行向量之间和各列向量之间的线性关系。
- 对于满秩矩阵，秩等于行数或列数，其各行向量或列向量都线性无关。
- 通过函数rank( )求矩阵的秩。该函数的调用格式为：**rank(A)**，该函数求矩阵的秩。

(7) 矩阵的迹

- 矩阵的迹等于矩阵的对角线元素之和，也等于矩阵的特征值之和。

- 在MATLAB中，通过函数trace( )求矩阵的迹。该函数的调用格式为：trace(A)，求矩阵的迹。

(8) 矩阵的范数

- 矩阵的范数常用的有3种。在MATLAB中，求矩阵范数的函数为norm( )，调用格式为：
- norm(X)或norm(X, 2)：计算矩阵的2-范数，返回矩阵的最大奇异值。
- norm(X, 1)：计算矩阵的1-范数，返回矩阵的的列向元素和的最大值。
- norm(X, inf)：计算矩阵的∞-范数，返回矩阵的行向元素和的最大值。

A =

1	2	3
4	5	6
7	8	9

在 MATLAB 命令窗口中输入：n = norm (A)，

计算结果(即 A 矩阵的范数)为：n = 16.8481。

在 MATLAB 命令窗口中输入：n = norm (A,1)，

计算结果(即 A 矩阵的 1 阶范数)为：n = 18。

在 MATLAB 命令窗口中输入：n = norm (A,2)，

计算结果(即 A 矩阵的最大奇异值)为：n = 16.8481。

在 MATLAB 命令窗口中输入：n = norm (A, inf)，

计算结果(即 A 矩阵的无穷大范数)为：n = 24。

(9) 矩阵的条件数

- 矩阵的条件数是用来判断矩阵病态的一个量，矩阵的条件数越大，表明该矩阵越病态，否则该矩阵越良态。
- **cond(X, 1)**：该函数计算矩阵X的1-范数下的条件数。
- **cond(X)或cond(X, 2)**：该函数计算矩阵X的2-范数数下的条件数。
- **cond(X, inf)**：该函数计算矩阵X的∞-范数下的条件数。

(1) 对于任意一个 5×5 矩阵，表示为

A =

1	1	1	1	1
2	4	1	4	5
3	6	1	0	7
9	3	0	5	1
4	7	1	0	3

在 MATLAB 命令窗口中输入：c = cond (A)，计算结果为：c = 21.5457。

在 MATLAB 命令窗口中输入：c = cond (A, 1)，计算结果为：c = 36.5983。

在 MATLAB 命令窗口中输入：c = cond (A, 2)，计算结果为：c = 21.5457。

在 MATLAB 命令窗口中输入：c = cond (A, inf)，计算结果为：c = 27.7799。

(10) 矩阵的标准正交基

- 若向量空间的基是正交向量组，则称其为向量空间的正交基，若正向向量组的每个向量都是单位向量，则称其为向量空间的标准正交基。

$$\alpha=(1,\ 0,\ 0)$$

$$\beta=(0,\ 1,\ 0)$$

$$\gamma=(0,\ 0,\ 1)$$

- **B=orth(A)**，矩阵B的列向量组成了矩阵A的一组标准正交基。

A =

1	4	5
8	2	9
1	0	4

在 MATLAB 命令窗口中输入： B = orth (A)，计算结果为：

B =

0.3986	0.8978	0.1873
0.8762	-0.4331	0.2112
0.2707	0.0800	-0.9593

2.3.7 稀疏矩阵

- 矩阵中非零元素的个数远远小于矩阵元素的总数），并且非零元素的分布没有规律，则称该矩阵为**稀疏矩阵(sparse matrix)**；
- 如果非零元素的分布存在规律（如上三角矩阵、下三角矩阵、对称矩阵），则称该矩阵为**特殊矩阵**

2.3.7 稀疏矩阵

- 在科学与工程领域中求解线性模型时经常出现大型的稀疏矩阵。
- 在使用计算机存储和操作稀疏矩阵时，经常需要修改标准算法以利用矩阵的稀疏结构。由于其自身的稀疏特性，通过压缩可以大大节省稀疏矩阵的内存代价。更为重要的是，由于过大的尺寸，标准的算法经常无法操作这些稀疏矩阵。

(1) 矩阵存储方式

- 在MATLAB中，对于矩阵的存储有两种方式：完全存储方式和稀疏存储方式。
- 完全存储方式是将矩阵的全部元素按照矩阵的列存储。以前讲到的矩阵的存储方式都是按这个方式存储的。如果矩阵中的元素只有少数不是零，会浪费大量的存储空间。

(1) 矩阵存储方式

- 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。用户可以创建整型、双精度、复数类型和逻辑类型的稀疏矩阵。稀疏矩阵不能自动生成。定义在完全存储方式下的运算只能产生完全存储的矩阵，不论多少个元素为0。在运算过程中，稀疏存储矩阵可以直接参与运算，产生的结果也是稀疏矩阵。

(2) 产生稀疏矩阵

- 通过函数sparse( )把普通矩阵转换为稀疏矩阵，该函数的调用格式为：
- S=sparse(A)，该函数将矩阵A转换为稀疏矩阵S。当矩阵A是稀疏存储方式时，则函数调用相当于S=A。
- S=sparse(m, n)：该函数产生大小为m行n列，所有元素都是0的稀疏矩阵。

- 通过函数full( )把稀疏矩阵转换为普通矩阵，该函数的调用格式为：B=full(A)，该函数将稀疏矩阵A转换为普通矩阵B。

(2) 产生稀疏矩阵

- 函数**nnz(S)**计算稀疏矩阵S中非零值的个数

  (Number of nonzero matrix elements)
- 函数**nonzeros( )**获取稀疏矩阵的非零值
- 函数**nzmax( )**获取存储非零值的空间长度
- 函数**spy( )**对稀疏矩阵中非零元素的分布进行图形化显示。
- 用函数**spalloc( )**为稀疏矩阵分配空间 。

(3) 特殊稀疏矩阵

- 单位矩阵只有对角线元素为1，其它元素都为0，是一种具有稀疏特征的矩阵。函数eye( )产生一个完全存储方式的单位矩阵。在MATLAB中，还有一个产生稀疏存储方式的单位矩阵的函数speye( )。该函数的调用格式为：
- **S=speye(n)** 产生一个n行n列的单位稀疏存储矩阵。
- **S=speye(m, n)** 产生一个m行n列的单位稀疏存储矩阵。

(3) 特殊稀疏矩阵

- **spones( ):** 将稀疏矩阵中的非零元素替换为1。
- **sprand( ):** 非零元素为均匀分布随机数的稀疏矩阵
- **sprandn( ):** 非零元素为高斯分布随机数的稀疏矩阵
- **sprandsym( ):** 非零元素为随机数的对称稀疏矩阵
- **spdiags( ):** 创建对角随机矩阵

2.3.8 矩阵的分解

矩阵分解是指根据一定的原理用某种算法将一个矩阵分解成若干个矩阵的乘积。

常见的矩阵分解有Cholesky分解、LU分解、QR分解，以及Schur分解和Hessenberg分解等。

(1) Cholesky分解

- 对于正定矩阵，可以分解为上三角矩阵和下三角矩阵的乘积，这种分解称为Cholesky分解。并不是所有的矩阵都可以进行Cholesky分解。能够进行Cholesky分解的矩阵必须是正定的，矩阵的所有对角元素必须是正的，同时矩阵的非对角元素不能太大。
- 通过函数chol( )进行矩阵的Cholesky分解。在采用函数chol( )进行Cholesky分解时，最好先通过函数eig( )得到矩阵的所有特征值，检查特征值是否为正。

X =

1	1	1	1	1	1
1	2	3	4	5	6
1	3	6	10	15	21
1	4	10	20	35	56
1	5	15	35	70	126
1	6	21	56	126	252

在 MATLAB 命令窗口中输入：R = chol (X)，

R =

1	1	1	1	1	1
0	1	2	3	4	5
0	0	1	3	6	10
0	0	0	1	4	10
0	0	0	0	1	5
0	0	0	0	0	1

(2) LU分解

- 在线性代数中，LU分解是矩阵分解的一种，可以将一个矩阵分解为一个下三角矩阵和一个上三角矩阵的乘积（有时是它们和一个置换矩阵的乘积，置换矩阵是一种系数只由0和1组成的方块矩阵）。
- LU分解主要应用在数值分析中，用来解线性方程、求反矩阵或计算行列式。

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}$$

(2) LU分解

- 高斯消去法又称为LU分解，将方阵A分解为下三角矩阵的置换矩阵L和上三角矩阵U的乘积。在MATLAB中，通过函数lu( )进行矩阵的LU分解。该函数的调用格式为：
- [L1, U1]=lu(A)：该函数将矩阵分解为下三角矩阵的置换矩阵L1和上三角矩阵U1。
- [L2, U2, P]=lu(A)：该函数将矩阵分解为下三角矩阵L2和上三角矩阵U2，以及置换矩阵P。
- Y=lu(A)：该函数将下三角矩阵和上三角矩阵合并在矩阵Y中，矩阵Y的对角元素为上三角矩阵的对角元素。

(3) QR分解

- 矩阵的正交分解，又称为QR分解。QR分解将一个m行n列的矩阵A分解为一个正交矩阵Q（m行m列）和一个上三角矩阵R（大小为m行n列）的乘积。
- [Q, R]=qr(A)，该函数将矩阵A进行QR分解，返回正交矩阵Q和上三角矩阵R。

(4) SVD分解

- 奇异值分解也是常用的矩阵分解之一。通过函数svd( )进行矩阵的svd分解（或奇异值分解）。
- **s=svd(A)**：该函数对矩阵A进行奇异值分解，返回由奇异值组成的列向量，奇异值按照从大到小的顺序进行排列。
- **[U, S, V]=svd(A)**：该函数对矩阵进行奇异值分解，其中U和V为酉矩阵，S为一个对角矩阵，对角线的元素为矩阵的奇异值的降序排列。

(5) Schur分解

- 对矩阵A的Schur分解公式为：A=U*S*U’，矩阵A必须是方阵，U为酉矩阵，S为块对角矩阵。在MATLAB中，通过函数schur( )进行矩阵的Schur分解。该函数的调用格式为：
- [U, S]=schur(A)：该函数将矩阵A进行Schur分解，返回酉矩阵U和块对角矩阵S。
- S=schur(A)：该函数仅返回块对角矩阵S。

(6) Hessenberg分解

- 对于任意一个n阶方阵可以进行Hessenberg分解，分解公式为：A=PHP’，其中P是酉矩阵，H的第一子对角线下的元素均为0，即H为Hessenberg矩阵。在MATLAB中，采用函数hess( )进行方阵的Hessenberg分解，该函数的调用格式为：
- **H=hess(A)**：该函数对方阵A进行Hessenberg分解，返回Hessenberg矩阵。
- **[P, H]=hess(A)**：该函数对方阵A进行Hessenberg分解，返回值为P和H，满足A=PHP’。

第4章 字符串分析

在使用MATLAB时经常会遇到对字符或字符串的操作。字符串是指1×n的字符数组。

在MATLAB软件中提供了很多的字符或字符串操作方法和函数，包括字符串的创建、字符串的属性、比较、查找以及字符串的转换和执行等。

2.4.1 字符串处理函数

字符和字符串是MATLAB语言的重要组成部分，MATLAB提供了强大的字符串处理功能。

- 在MATLAB中，单个字符是按照Unicode编码存储的，每个字符占两个字节。
- MATLAB内部按照字符的编码数值对字符串进行运算。

(1) 字符串基本属性

- 在MATLAB中，对字符串的设定非常的简单，只需要用单引号（’ ）将需要设定的字符串括起来，必须是在英文状态下输入的。

- 通过函数disp( )：对字符串进行显示。
- 通过函数size( )：取得该字符串的长度。
- 通过函数double( )：将字符串以ASCII码显示。
- 通过函数char( )：将ASCII码以字符串显示。

(2) 字符串的构造

在MATLAB中，多个字符串可以构成字符矩阵，但是矩阵的每行字符数必须相等。

函数strcat( )和函数strvcat( )对字符串进行连接：

- strcat( )将多个字符串连接成行向量，字符串首尾连接在一起，形成一个新的字符串。
- strvcat( )将多个字符串连接成列向量 。

(3) 字符串的比较

对两个字符串进行比较：

- **strcmp(str1, str2)**，该函数比较字符串str1和字符串str2是否相等，如果相等，函数返回值为1；当不相等时，返回值为0。
- **strncmp(str1, str2, k)**，该函数比较字符串str1和字符串str2的前k个字符是否相等，如果相等，返回值为1；当不相等时，返回值为0。该函数区分字符的大小写。
- **strncmpi(str1, str2, k)**，和函数strncmp( )基本一样。不同之处是该函数不区分字符的大小写。

(3) 字符串的比较

还可以通过字符的运算进行比较。当两个字符具有相同的长度时，可以采用matlab的运算符进行逐个字符的比较

6种关系运算符：

<（小于)、<=（小于或等于)、

>（大于)、>=（大于或等于）、

==（恒等于）、～=（不等于）

(4) 字符串的查找和替换

- 字符串的查找和替换是字符串操作的一项重要内容，在MATLAB中提供了函数strfind( )、和strrep( )等函数来实现对字符串的查找和替换操作。

- strfind( )的调用格式为

k=strfind(text, pattern)，在text字符串中查找pattern字符或字符串，当查找成功后返回第一个相同字符的具体位置。

(4) 字符串的查找和替换

- 函数strrep( )的调用格式为

S=strrep(S1, S2, S3)，该函数会将字符串S1中的子串S2都替换为S3，然后返回到字符串S中。如果没有找到字符串S2，就不进行字符串的替换，输出仍未原来的字符串。

(5) 字符串的转换

- 使用函数num2str( )、int2str( )、str2num( )和str2double( )等实现字符串和数值之间的相互转换：
- 函数t=num2str(X)：函数将数字X转换为字符串t。如果输入参数X为矩阵，则转换为一个字符串矩阵。该函数也可以指定数字的精度，调用格式为t=num2str(X, n)，其中t的精度为n位。

(5) 字符串的转换

- 函数t=int2str(X)，该函数将整数X转换为字符串。如果X不是整数，先将X取整，然后再转换为字符串。
- 函数x=str2num(S)，该函数将字符型矩阵S转换为一个数字矩阵。
- 函数x=str2double(S)，该函数将字符串转换为双精度的数值。
- 函数str=mat2str(mat)，该函数将数组或矩阵转换为对应的字符串。

(5) 字符串的转换

可以使用函数实现十进制、二进制、十六进制直接的转换：

- **hex2dec()、bin2dec()、dec2hex()、dec2bin()**
- 二进制和十六进制不能直接进行转换（没提供内部函数），需通过十进制进行间接转换。
- 二进制和十六进制是以字符串形式进行保存的。

4.2 字符串的其他操作

- 除了字符串的构造、查找、替换，以及字符串的转换等，在MATLAB中还有一些字符串处理函数，能够进行字符串的一些操作，例如字符的分类、字符串的执行等，还可以进行字符串大小写的转换等。

(1) 字符的分类

字符串中的字符通常可以分为空白字符、字母字符和其他类型的字符。可以通过函数isspace( )和函数isletter( )对字符串中的字符进行分类：

- 函数**isspace(S)**：字符串S进行分类，如果为空白字符，返回值为1,；否则返回值为0；
- 函数**isletter(S)**，对字符串S进行分类，如果为字母字符，返回值为1；否则返回值为0。

(2) 其他操作

- 函数upper( )可以将字符串转换为大写字母。
- 函数lower( )可以将字符串转换为小写字母。
- 利用函数ischar( )判断是否为字符，如果为字符则返回值为1，否则返回值为0。

- 在MATLAB的命令窗口，输入命令“help strfun”，可以显示所有的字符串操作函数。