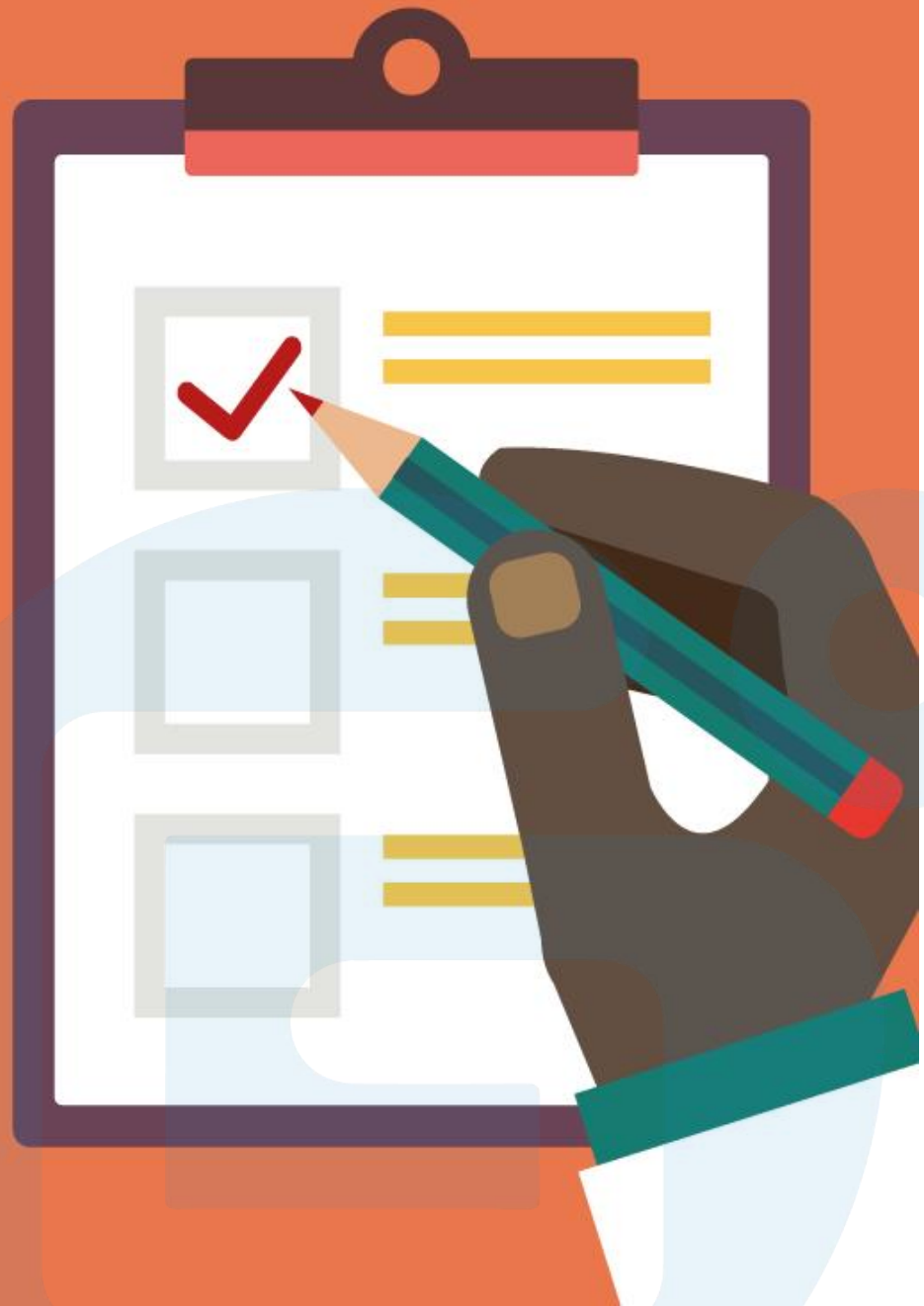
Talk is cheap, show me the code

第二课：矩阵的基本特性

Matlab入门课程系列



OUTLINE

- **Matlab矩阵的基本操作变量**
- 基本运算符
- 字符操作基础
- 数组和矩阵计算基础

一 Matlab矩阵的基本操作

Matlab矩阵的基本操作

在数学上，定义由 $m \times n$ 个数 $a_{ij}(i=1,2,\cdots,m; j=1,2,\cdots,n)$ 排成的 m 行 n 列的数表

$$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

为 m 行 n 列矩阵，并用大写黑体字母表示它。

只有一行的矩阵

$$A=\begin{pmatrix} a_1 & a_2 & \cdots & a_n \end{pmatrix}$$

称为行向量。

同理，只有一列的矩阵

$$A=\begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix}$$

称为列向量。

Matlab矩阵的基本操作

- $\begin{vmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{vmatrix}\begin{vmatrix}x1\\x2\\x3\end{vmatrix}=\begin{vmatrix}1\\2\\3\end{vmatrix}$ Ax=b, $AA^{-1}=\begin{vmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{vmatrix}$=$A^{-1}$A, x=$A^{-1}$b
- $\begin{vmatrix}-3 & -3 & -3\\4 & 5 & 6\\7 & 8 & 9\end{vmatrix}\begin{vmatrix}x1\\x2\\x3\end{vmatrix}=\begin{vmatrix}-1\\2\\3\end{vmatrix}$
- $\frac{\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}{\sqrt{2\pi}}\sim\exp(-kx^2)$

$$\frac{dS}{dt}=\Omega\times S$$

$$F(x>x_0)=\int_{-\infty}^{x_0}f(x)dx$$

Matlab矩阵的基本操作

复习：数组和矩阵的区别

- 数组是在程序设计中为了方便处理而把相同类型的一系列变量排列起来的集合。包括数值数组，字符数组，结构数组等多种形式。
- 而矩阵是一个**数学的概念**，矩阵运算有严格的数学规律，而数组运算则是matlab软件设定的规则，目的仅仅是为了方便数据管理，操作简单，命令形式自然。
- 二者的**联系**在于：
- matlab里矩阵是以数值数组的形式储存的，因此**一维数值数组可以视为向量，二维则是矩阵。**

Matlab矩阵的基本操作

创造一个矩阵有两种方法：

直接赋值的简单矩阵法，利用函数名的特殊矩阵法

命令行窗口

```
>> A=[1,2,3.22;5,6,8.88;1,2,1.111]

A =

    1.0000    2.0000    3.2200
    5.0000    6.0000    8.8800
    1.0000    2.0000    1.1110

>> A=[1,2,3.22;5,6,8.88;1,2,1.111,3]
要串联的数组的维度不一致。
```

简单矩阵法赋值的时候要注意保持每个维度一致

```
>> B=ones(4)

B =

     1     1     1     1
     1     1     1     1
     1     1     1     1
     1     1     1     1

>> C=eye(2,3)

C =

     1     0     0
     0     1     0
```

Matlab矩阵的基本操作

利用函数名的特殊矩阵法

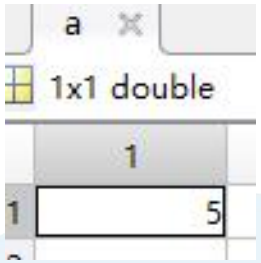
函数名称	函数功能
ones(n)	构建一个 $n\times n$ 的 1 矩阵（矩阵的元素全部是 1）
ones(m , n ,⋯, p)	构建一个 $m\times n\times\cdots\times p$ 的 1 矩阵
ones(size(A))	构建一个和矩阵 A 同样大小的 1 矩阵
zeros(n)	构建一个 $n\times n$ 的 0 矩阵（输出矩阵的元素全部是 0）
zeros(m , n ,⋯, p)	构建一个 $m\times n\times\cdots\times p$ 的 0 矩阵
zeros(size(A))	构建一个和矩阵 A 同样大小的 0 矩阵
eye(n)	构建一个 $n\times n$ 的单位矩阵
eye(m, n)	构建一个 $m\times n$ 的单位矩阵

Matlab矩阵的基本操作

函数名称	函数功能
eye(size(*A*))	构建一个和矩阵 *A* 同样大小的单位矩阵
magic(*n*)	构建一个 *n*×*n* 的矩阵，其每一行、每一列的元素之和都相等
rand(*n*)	构建一个 *n*×*n* 的矩阵，其元素为 0～1 之间均匀分布的随机数
rand(*m* , *n* ,⋯, *p*)	构建一个 *m*×*n*×⋯×*p* 的矩阵，其元素为 0～1 之间均匀分布的随机数
randn (*n*)	构建一个 *n*×*n* 的矩阵，其元素为零均值、单位方差的正态分布随机数
randn (*m* , *n* ,⋯, *p*)	构建一个 *m*×*n*×⋯×*p* 的矩阵，其元素为零均值、单位方差的正态分布随机数
diag(*x*)	构建一个 *n* 维的方阵，它的主对角线元素值取自向量 *x*， 其余元素的值都为 0
diag(*A* , *k*)	构建一个由矩阵 *A* 第 *k* 条对角线的元素组成的列向量 *k*= 0 为主对角线；*k*< 0 为下第 *k* 条对角线；*k*> 0 为上第 *k* 条对角线
diag(*x* , *k*)	构建一个(*n*+|*k*|)×(*n*+|*k*|)维的矩阵，该矩阵的第 *k* 条对角线元素取自向量 *x*，其余元素都为 0（关于参数 *k*，参考上个命令）
triu(*A*)	构建一个和 *A* 大小相同的上三角矩阵，该矩阵的主对角线上元素为 *A* 中相应元素，其余元素都为 0
triu(*A* , *k*)	构建一个和 *A* 大小相同的上三角矩阵，该矩阵的第 *k* 条对角线及其以上元素为 *A* 中相应元素，其余元素都为 0
tril(*A*)	构建一个和 *A* 大小相同的下三角矩阵，该矩阵的主对角线上元素为 *A* 中相应元素，其余元素都为 0
tril(*A*, *k*)	构建一个和 *A* 大小相同的下三角矩阵，该矩阵的第 *k* 条对角线上及其以下元素为 *A* 中相应元素，其余元素都为 0

Matlab矩阵的基本操作

通常一个矩阵是m行×n列的，如果m和n取1，则得到向量，全部取1则得到标量。Matlab里的标量，实际上是1 ×1的矩阵。



m和n取0，为空矩阵，不占据储存空间。注意和零矩阵的区别。

```
a =

     []

>> a

a =

     []
```

```
>> b=zeros(5)

b =

     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0    -0     0     0
     0     0     0     0     0
```

Matlab矩阵的基本操作

矩阵大小和结构改变

根据运算需要，有时需要对现存的矩阵进行大小和结构上的改变，例如旋转矩阵、改变矩阵的维度，删除部分元素等。

```
>> A=rand(1,4)

A =

    0.8147    0.9058    0.1270    0.9134

>> B=randn(1,4)

B =

    0.3188   -1.3077   -0.4336    0.3426

>> A1=reshape(A,2,2)

A1 =

    0.8147    0.1270
    0.9058    0.9134

>> B2=fliplr(B)

B2 =

    0.3426   -0.4336   -1.3077    0.3188
```

Matlab矩阵的基本操作

函数名称	函数功能
fliplr(*A*)	矩阵每一行均进行逆序排列
flipud(*A*)	矩阵每一列均进行逆序排列
flipdim(*A*, dim)	生成一个在 dim 维矩阵 *A* 内的元素交换位置的多维矩阵
rot90(*A*)	生成一个由矩阵 *A* 逆时针旋转 90° 而得到的新矩阵
rot90(*A*, *k*)	生成一个由矩阵 *A* 逆时针旋转 $k\times90°$ 而得到的新矩阵
reshape(*A* , *m* , *n*)	生成一个 $m\times n\times\cdots\times p$ 维的矩阵，其元素以线性索引的顺序从矩阵 *A* 中取得 如果矩阵 *A* 中没有 $m\times n\times\cdots\times p$ 个元素，将返回一个错误信息
repmat(*A*,[*m n*⋯*p*])	创建一个和矩阵 *A* 有相同元素的 $m\times n\times\cdots\times p$ 块的多维矩阵
shiftdim(*A* , *n*)	矩阵的列移动 *n* 步。*n* 为正数，矩阵向左移；*n* 为负数，矩阵向右移
squeeze(*A*)	返回没有空维的矩阵 *A*
cat(dim,*A*, *B*)	将矩阵 *A* 和 *B* 组合成一个 dim 维的多维矩阵
permute(*A*, order)	根据向量 order 来改变矩阵 *A* 中的维数顺序
ipermute(*A*, order)	进行命令 permute 的逆变换
sort(*A*)	对一维或二维矩阵进行升序排序，并返回排序后的矩阵； 当 *A* 为二维矩阵时，对矩阵的每一列分别进行排序
sort(*A*, dim)	对矩阵按指定的方向进行升序排序，并返回排序后的矩阵。当 dim=1 时，对矩阵的每一列排序； dim=2 时，对矩阵的每一行排序
sort(*A*, dim, mode)	mode 为'ascend'时，进行升序排序；mode 为'descend'时，进行降序排序
[*B*, IX] = sort(*A*,⋯)	IX 为排序后各元素在原矩阵中的行位置或列位置的索引

Matlab矩阵的基本操作

矩阵下标引用，又叫切片（slice）

分为一元引用和二元引用。二元引用较为简单，（m,n）为对应的行列数。

一元引用则是按照“列优先”原则，挨个数过去。

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

A =

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

```
>> A(2,3)

ans =

     6

>> A(5)

ans =

     5

>> A(4)

ans =

     2
```

Matlab矩阵的基本操作

索引表达式	函数功能
$A(\mathrm{l})$	将二维矩阵 A 重组为一维数组，返回数组中第一个元素
$A(:,j)$	返回二维矩阵 A 中第 j 列列向量
$A(i,:)$	返回二维矩阵 A 中第 i 行行向量
$A(:,j:k)$	返回由二维矩阵 A 中的第 j 列到第 k 列列向量组成的子矩阵
$A(i:k,:)$	返回由二维矩阵 A 中的第 i 行到第 k 行行向量组成的子矩阵
$A(i:k,j:l)$	返回由二维矩阵 A 中的第 i 行到第 k 行行向量 和第 j 列到第 l 列列向量的交集组成的子矩阵
$A(:)$	将矩阵 A 中的每列合并成一个长的列向量
$A(j:k)$	返回一个行向量，其元素为 $A(:)$中的第 j 个元素到第 k 个元素
$A([j_1 j_2 \cdots])$	返回一个行向量，其中的元素为 $A(:)$中的第 j_1、j_2 元素
$A(:,[j_1 j_2 \cdots])$	返回矩阵 A 的第 j_1 列、第 j_2 列等的列向量
$A([i_1 i_2 \cdots],:)$	返回矩阵 A 的第 i_1 行、第 i_2 行等的行向量
$A([i_1 i_2 \cdots],[j_1 j_2 \cdots])$	返回矩阵第 i_1 行、第 i_2 行等和第 j_1 列、第 j_2 列等的元素

Matlab矩阵的基本操作

矩阵信息获取

矩阵信息包括矩阵的结构，大小，维度以及内存占用等。

1.矩阵结构是指矩阵中元素的排列方式

函数名称	函数功能
isempty(A)	检测矩阵是否为空
isscalar(A)	检测矩阵是否是单元素的标量矩阵
isvector(A)	检测矩阵是否是只具有一行或一列元素的一维向量
issparse(A)	检测数组是否是稀疏矩阵

这类函数的返回值是逻辑类型的数据。返回值为“1”表示该矩阵是某一特定类型的矩阵；返回值为“0”表示该矩阵不是该特定类型的矩阵。

```
A =

     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0

>> B=isscalar(A)

B =

  logical

   0
```

Matlab矩阵的基本操作

2.矩阵大小(形状)是指矩阵中元素的数量和排列规律，包括维度、各维长度、元素总数

函　数	调用格式	描　述
ndims	n=ndims(X)	获取矩阵的维数
size	[m,n]=size(X)	获取矩阵在各维上的长度
length	n=length(X)	获取矩阵最长维的长度
numel	n=numel(X)	获取矩阵元素的个数

```
>> A=eye(5,3)

A =

     1     0     0
     0     1     0
     0     0     1
     0     0     0
     0     0     0

>> ndims(A)
size(A)
length(A)
numel(A)
```

```
ans =

     2


ans =

     5     3


ans =

     5


ans =

    15
```

Matlab矩阵的基本操作

3.矩阵占据内存

```
A =

    0.9575    0.9706    0.8003
    0.9649    0.9572    0.1419
    0.1576    0.4854    0.4218
```

```
>> whos A
  Name        Size                Bytes  Class     Attributes

  A           3x3                    72  double
```

```
>> B=["Alice","Bob","candy";"Dog","Elephant","Fish"]

B =

  2×3 string 数组

    "Alice"     "Bob"         "candy"
    "Dog"       "Elephant"    "Fish"
```

```
>> whos B
  Name        Size                Bytes  Class     Attributes

  B           2x3                   484  string
```

whos命令用于查看数组所占的内存

Matlab矩阵的基本操作

矩阵合并

函　数	基本调用格式	描　述
cat	cat(DIM,A,B)	在 DIM 指定的维度上合并矩阵 A 和 B。DIM=1 表示按行（竖直方向）合并；DIM=2 表示按列（水平方向）合并
horzcat	horzcat(A,B)	在水平方向上合并矩阵 A 和 B
vertcat	vertcat(A,B)	在竖直方向上合并矩阵 A 和 B
repmat	B = repmat(A,M,N)	通过复制 $M\times N$ 个矩阵 A 来构造新的矩阵 B
blkdiag	Y = blkdiag(A,B,...)	用已知的 A、B 等多个矩阵构造块对角化矩阵 Y，其中 $Y=\begin{bmatrix} A & 0 & \cdots & 0 \\ 0 & B & \cdots & 0 \\ 0 & 0 & \cdots & N \end{bmatrix}$

二 基本运算符

基本运算符

Matlab利用运算符执行计算，包括算术运算符、关系运算符、逻辑运算符。当一起出现时，优先级如下：算术>关系>逻辑，运算符可以看成是一种简写的函数

```
>> a=10;
b=12;
a+b>21&a-b>0

ans =

  logical

   0

>> a+b>21|a-b>0

ans =

  logical

   1
```

当然，在写代码的时候，为了保持较好的可读性，也可以在容易引起歧义的地方加上圆括号。

```
>> ((a+b)>21)&((a-b)>0)

ans =

  logical

   0
```

基本运算符

算术运算符，包括常见的加减乘除幂次方，注意点乘、点除、点幂次方

算术运算符	运算法则	算术运算符	运算法则
A+*B*	*A* 与 *B* 相加 （*A*、*B* 为数值或矩阵）	*A* - *B*	*A* 与 *B* 相减 （*A*、*B* 为数值或矩阵）
A * *B*	*A* 与 *B* 相乘 （*A*、*B* 为数值或矩阵）	*A*.* *B*	*A* 与 *B* 相应元素相乘 （*A*、*B* 为相同维度的矩阵）
A / *B*	*A* 与 *B* 相除 （*A*、*B* 为数值或矩阵）	*A*./ *B*	*A* 与 *B* 相应元素相除 （*A*、*B* 为相同维度的矩阵）
A ^ *B*	*A* 的 *B* 次幂 （*A*、*B* 为数值或矩阵）	*A*.^ *B*	*A* 的每个元素的 *B* 次幂 （*A* 为矩阵，*B* 为数值）

基本运算符

算术运算符之外，还有大量的内置函数用于处理基础运算法则

函　数	
exp(*x*)	求以 e 为底数的 *x* 次幂
log(*x*)	以 e 为底数对 *x* 取对数
log10(*x*)	以 10 为底数对 *x* 取对数
sqrt(*x*)	*x* 的平方根
sin(*x*)	*x* 的正弦函数
cos(*x*)	*x* 的余弦函数
tan(*x*)	*x* 的正切函数
asin(*x*)	*x* 的反正弦函数
acos(*x*)	*x* 的反余弦函数
atan(*x*)	*x* 的反正切函数
mod(*a*,*b*)	*a* 与 *b* 相除取余数
min(*a*,*b*)	返回 *a* 与 *b* 中较小的数值
max(*a*,*b*)	返回 *a* 与 *b* 中较大的数值
mean(*x*)	找出 *x* 阵列的平均值

median(*x*)	找出 *x* 阵列的中位数
sum(*x*)	计算 *x* 阵列的总和值
prod(*x*)	计算 *x* 阵列的连乘值
cumsum(*x*)	计算 *x* 阵列的累积总和值
cumprod(*x*)	计算 *x* 阵列的累积连乘值
sign(*x*)	*x* <0 时返回值为-1，*x* =0 时返回值为 0，*x* >0 时返回值为 1
rem(*x*,*y*)	返回 *x*/*y* 的余数
diff(*x*)	***x*** 向量的差分
sort(x)	对 ***x*** 向量进行排序
fft(x)	***x*** 向量的离散傅里叶变换
rank(x)	***x*** 矩阵的秩

基本运算符

关系运算符，共有6个，>,<,>=,<=,==,~=

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

```
>> A=1:10
B=10-A

A =

     1     2     3     4     5     6     7     8     9    10


B =

     9     8     7     6     5     4     3     2     1     0

>> C=(A>4)
D=(A==B)
```

```
C =

  1×10 logical 数组

   0   0   0   0   1   1   1   1   1   1


D =

  1×10 logical 数组

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

基本运算符

逻辑运算符，& 与，|或，~非

```
>> A=1:10
B=~(A>4)

A =

     1     2     3     4     5     6     7     8     9    10


B =

  1×10 logical 数组

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

函　数	运算法则
xor(x,y)	异或运算。x 与 y 不同时，返回 1；x 与 y 相同时，返回 0
any(x)	如果在一个向量 x 中，有任何元素是非零，返回 1；否则返回 0 如果矩阵 x 中的每一列有非零元素，返回 1；否则返回 0
all(x)	如果在一个向量 x 中，所有元素非零，返回 1；否则返回 0 矩阵 x 中的每一列所有元素非零，返回 1；否则返回 0

逻辑运算函数

基本运算符

运算优先级，除了算术>关系>逻辑之外，还有一些别的规则。

1. 括号 ()
2. 转置 (.')、幂 (.^)、复共轭转置 (')、矩阵幂 (^)
3. 带一元减法 (.^-)、一元加法 (.^+) 或逻辑求反 (.^~) 的幂，以及带一元减法 (^-)、一元加法 (^+) 或逻辑求反 (^~) 的矩阵幂。

> **注意**
>
> 尽管大多数运算符都从左至右运行，但 (^-)、(.^-)、(^+)、(.^+)、(^~) 和 (.^~) 按从右至左顺序从第二个运行。建议您使用括号显式指定包含这些运算符组合的语句的期望优先级。

4. 一元加法 (+)、一元减法 (-)、逻辑求反 (~)
5. 乘法 (.*)、右除 (./)、左除 (.\)、矩阵乘法 (*)、矩阵右除 (/)、矩阵左除 (\)
6. 加法 (+)、减法 (-)
7. 冒号运算符 (:)
8. 小于 (<)、小于或等于 (<=)、大于 (>)、大于或等于 (>=)、等于 (==)、不等于 (~=)
9. 按元素 AND (&)
10. 按元素 OR (|)
11. 短路 AND (&&)
12. 短路 OR (||)

一元加减法就是正负号

A&B

（1）首先判断A的逻辑值，然后判断B的值，然后进行逻辑与的计算。

（2）A和B可以为矩阵（e.g. A=[1 0],B=[0 0]）。

A&&B

（1）首先判断A的逻辑值，如果A的值为假，就可以判断整个表达式的值为假，就不需要再判断B的值。

（2）A和B不能是矩阵，只能是标量。

"|"与“||”同理。

三 字符操作基础

字符操作基础

单引号和双引号都可以用于创造字符串，区别在于单引号用于连接，双引号则是分隔

```
>> A=['alice','bob','sun']

A =

    'alicebobsun'

>> B=["alice","bob","sun"]

B =

  1×3 string 数组

    "alice"    "bob"    "sun"
```

```
>> B='this is your''s toy'

B =

    'this is your's toy'
```

```
>> A="nuclear bomb is so-called ""God Weapon"""

A =

    "nuclear bomb is so-called "God Weapon""
```

```
>> A='nuclear bomb is so-called "God''s Weapon"'

A =

    'nuclear bomb is so-called "God's Weapon"'
```

字符操作基础

字符串的比较，进行比较的两个字符串务必长度相等

```
>> A=['hello'=='world']

A =

  1×5 logical 数组

   0   0   0   1   0
```

```
>> B=strcmp('hello','hello')
C=strcmp('hello','world')

B =

  logical

   1


C =

  logical

   0
```

```
>> abs('中华人民共和国')

ans =

     20013     21326     20154     27665     20849     21644     22269
```

直接比较，是对每个字符进行比较，返回一串逻辑变量(0 or 1)，而采用strcmp函数则是进行整体比较，只要不完全一样，就返回0

字符操作基础

字符串查找和替换

```
>> A='This is a string'
B=findstr(A,'s')

A =

    'This is a string'


B =

     4     7    11
```

```
>> C=strrep(A,'string','sentence')

C =

    'This is a sentence'
```

```
>> C(9)='A'
C

C =

    'This is A sentence'


C =

    'This is A sentence'
```

字符操作基础

字符串的数值转换

函数名称	函数功能
abs	字符转换成 ASCII
dec2hex	十进制数转换成十六进制字符串
fprintf	把格式化的文本写到文件中或显示屏上
hex2dec	十六进制字符串转换成十进制数
hex2num	十六进制字符串转换成 IEEE 浮点数
int2str	整数转换成字符串
lower	字符串转换成小写
num2str	数字转换成字符串
setstr	ASCII 转换成字符串
sprintf	用格式控制数字转换成字符串
sscanf	用格式控制字符串转换成数字
str2mat	字符串转换成一个文本矩阵
str2num	字符串转换成数字
upper	字符串转换成大写

```
>>  rad=2.5;area=pi*rad^2;
string=['A circle of radius ' num2str(rad) ' has an area of ' num2str(area) '.'];
string

string =

    'A circle of radius 2.5 has an area of 19.635.'
```

四 数组和矩阵计算基础

数组和矩阵计算基础

数组的创建

- 直接赋值创建；
- 利用冒号的隐do循环创建；
- 利用logspace语句在10a和10b之间创建等比数列；
- 利用linspace语句创建；
- ……

```
>> A=[1,2,3.3;4,5,6.5;7,8,9.01]

A =

    1.0000    2.0000    3.3000
    4.0000    5.0000    6.5000
    7.0000    8.0000    9.0100
```

```
>> B=[1:5:6:10:11:0.1:11.4]

B =

    1.0000    2.0000    3.0000    4.0000    5.0000
    6.0000    7.0000    8.0000    9.0000   10.0000
   11.0000   11.1000   11.2000   11.3000   11.4000
```

```
>> C=logspace(1,2,10)

C =

   10.0000   12.9155   16.6810   21.5443   27.8256   35.9381   46.4159   59.9484   77.4264  100.0000
```

```
>> D=linspace(1,51,10)

D =

    1.0000    6.5556   12.1111   17.6667   23.2222   28.7778   34.3333   39.8889   45.4444   51.0000
```

```
>>  D=linspace(1,51,11)

D =

     1     6    11    16    21    26    31    36    41    46    51
```

```
>> D=linspace(1,51)

D =

  1 至 14 列

    1.0000    1.5051    2.0101    2.5152    3.0202    3.5253    4.0303    4.5354    5.0404    5.5455
```

数组和矩阵计算基础

数组的算术运算

- 包括加减乘除（左除和右除）以及乘方
- 要求两个数组形状相同，是对应元素之间执行加减乘除
- 乘除的符号使用“.*”“./”“.\”，不加点则执行矩阵乘法
- 乘方可以两个数组，也可以数组对标量，同样要加.^

```
>> A=[1,2;3,4]

A =

     1     2
     3     4

>> B=[3,5;6,8]

B =

     3     5
     6     8

>> A*B

ans =

    15    21
    33    47

>> A.*B

ans =

     3    10
    18    32
```

数组和矩阵计算基础

数组的比较运算和逻辑运算

- 同样包括6个，>,<,>=,<=,==,~=
- 数组的关系运算符返回同维数组，用布尔量表示逻辑是否成立
- 逻辑运算包括与或非，矩阵逻辑运算规则同关系运算

```
>> A=[1,3;5,4]
B=[2,2.2;4,4.5]

A =

     1     3
     5     4


B =

    2.0000    2.2000
    4.0000    4.5000
```

```
>> A>B

ans =

  2×2 logical 数组

   0   1
   1   0
```

```
>> A&B

ans =

  2×2 logical 数组

   1   1
   1   1

>> ~A

ans =

  2×2 logical 数组

   0   0
   0   0
```

数组和矩阵计算基础

矩阵的转置和共轭转置

‘和.’用于表示共轭转置和普通转置，对于实数，二者没有区别

```
>> A=[0-1i,2+2i;4+2i,5-3i]

A =

   0.0000 - 1.0000i   2.0000 + 2.0000i
   4.0000 + 2.0000i   5.0000 - 3.0000i

>> A'

ans =

   0.0000 + 1.0000i   4.0000 - 2.0000i
   2.0000 - 2.0000i   5.0000 + 3.0000i

>> A.'

ans =

   0.0000 - 1.0000i   4.0000 + 2.0000i
   2.0000 + 2.0000i   5.0000 - 3.0000i
```

数组和矩阵计算基础

矩阵的运算法则

加减与数组没有区别，要求俩矩阵必须形状一致

乘除要遵循矩阵乘除法的规则

两个矩阵的乘法必须满足被乘矩阵的列数与乘矩阵的行数相等。设矩阵 A 为 $m\times h$ 矩阵，B 为 $h\times n$ 矩阵，则两矩阵的乘积 $C=A\times B$ 为一个矩阵，且 $C_{mn}=\sum_{k=1}^{n}A_{mk}\times B_{kn}$ 。

矩阵之间的乘法不遵循交换律，即 $A\times B\neq B\times A$。但矩阵乘法遵循下列运算律。

- 结合律：$(A\times B)\times C=A\times(B\times C)$。
- 左分配律：$A\times(B+C)=A\times B+A\times C$。
- 右分配律：$(B+C)\times A=B\times A+C\times A$。
- 单位矩阵的存在性：$E\times A=A$，$A\times E=A$。

数组和矩阵计算基础

矩阵的除法是乘法的逆运算，分为左除和右除两种，分别用运算符号“\”和“/”表示。如果矩阵 *A* 和矩阵 *B* 是标量，那么 *A/B* 和 *A\B* 仍然是不等价的。对于一般的二维矩阵 *A* 和 *B*，当进行 *A\B* 运算时，要求 *A* 的行数与 *B* 的行数相等；当进行 *A/B* 运算时，要求 *A* 的列数与 *B* 的列数相等。

矩阵如果是方阵，可以执行乘方的运算，其实质是A×A×A×A×A×A...

```
>> A=[1,2;3,4]
A2=A^2
A3=A^3

A =

     1     2
     3     4
```

```
A2 =

     7    10
    15    22


A3 =

    37    54
    81   118
```

数组和矩阵计算基础

矩阵排序

```
>> A=[5,3,45;2,33,51;22,30,4]

A =

     5     3    45
     2    33    51
    22    30     4

>> sort(A)

ans =

     2     3     4
     5    30    45
    22    33    51
```

```
>> sort(A,1)

ans =

     2     3     4
     5    30    45
    22    33    51

>> sort(A,2)

ans =

     3     5    45
     2    33    51
     4    22    30
```

```
>> sort(A,'ascend')

ans =

     2     3     4
     5    30    45
    22    33    51
```

```
>> sort(A,'descend')

ans =

    22    33    51
     5    30    45
     2     3     4

>> sort(A,2,'descend')

ans =

    45     5     3
    51    33     2
    30    22     4
```

数组和矩阵计算基础

矩阵求和

```
>> A=[5,3,45;2,33,51;22,30,4]

A =

     5     3    45
     2    33    51
    22    30     4
```

```
>> B=sum(A)
C=sum(A,2)
D=cumsum(A)
E=cumsum(A,2)
F=sum(sum(A))

B =

    29    66   100


C =

    53
    86
    56
```

```
D =

     5     3    45
     7    36    96
    29    66   100


E =

     5     8    53
     2    35    86
    22    52    56


F =

   195
```

数组和矩阵计算基础

矩阵求积(product)

```
>>  B=prod(A)
C=prod(A,2)
D=cumprod(A)
E=cumprod(A,2)
F=prod(prod(A))
B =
         220        2970        9180
C =
         675
        3366
        2640
D =
           5           3          45
          10          99        2295
         220        2970        9180
E =
           5          15         675
           2          66        3366
          22         660        2640
F =
   5.9982e+09
```

下节课预习

下节课预习

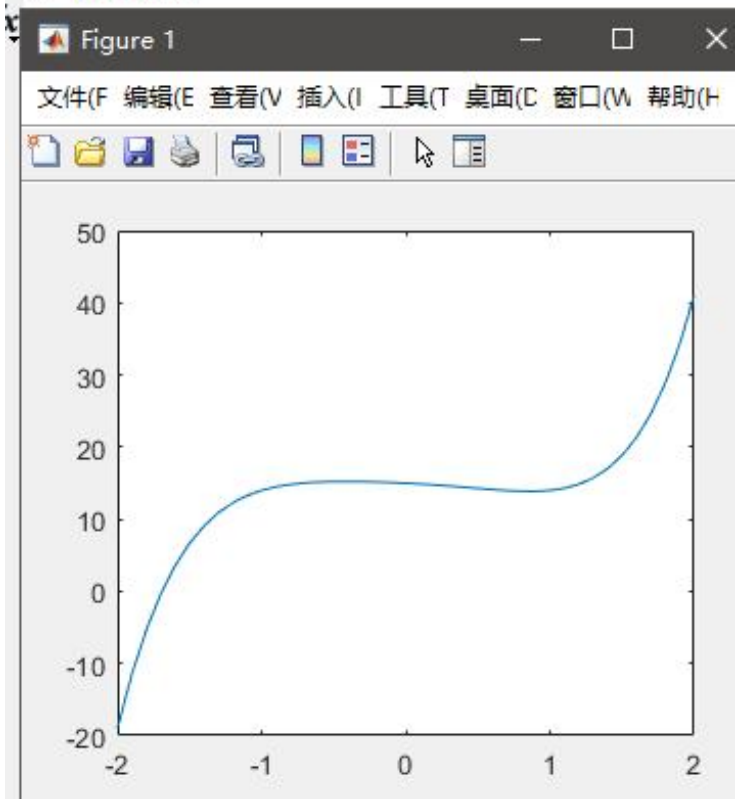
- 矩阵运算中，还包括一些复杂运算，例如**求逆，求范数，矩阵分解，矩阵特征值，特征向量**等，由于该部分内容与线性代数相关性较强，本课程不再一一说明，有兴趣的同学可以课后结合帮助文档进行学习。
- 下节课我们将学习Matlab编程基础，包括程序控制流，基本语法，脚本和函数，变量传递，程序调试等相关知识。

下节课预习

思考题：形如x5-x2-x+15=0这样的高阶方程组，如何对其进行求解？

```
>> syms x;
>> y=@(x)x^5-x^2-x+15;
>> fsolve(y,0)

Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the default value of the function tolerance, and
the problem appears regular as measured by the gradient.

<stopping criteria details>

ans =
   -1.6911
>> x=ans;
>> y(x)
ans =
  -5.0297e-09
```

```
>> x=[-2:0.1:2];
>> y=x.^5-x.^2-x+15;
>> plot(x,y)
```



感谢参与 下堂课见