北太振寰

成 为 世 界 领 先 的 科 学 计 算 软 件 提 供 者

北太振寰（重庆）科技有限公司

北太天元语法基础

目录 CONTENTS

01- 数据类型

02- 数组和矩阵创建

03- 数组和矩阵运算

目录 CONTENTS

01 数据类型

数据类型

北太天元为用户提供多种数据类型，以便在不同的情况下使用。用户可以使用浮点型、整数类型、字符、字符串、逻辑型true或false、函数句柄、结构体（struct）和元胞（cell）数组等数据类型。函数句柄是可以间接调用函数的一种特殊的结构语言，也是一种数据类型。

在北太天元的数据类型中，除函数句柄以外，其它数据类型均可以用矩阵或者数组的形式来表示。

数据类型-数值型

北太天元中的数值型包括：有符号整数、无符号整数、单精度和双精度浮点数。

默认情况下，北太天元存储数据使用的是双精度浮点数。用户不可以更改默认的数据类型和精度，但可以选择用整数或者单精度来存储数组或矩阵。整数和单精度数组相比双精度能更高效地利用内存。

所有的数值型数组或矩阵都支持比如数组的拼接、重构等基本的操作。所有数值类型都可以使用数学运算符。

数据类型-数值型

例如:

```
>> a=1:10
a =
   1   2   3   4   5   6   7   8   9   10
```

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

数据类型-浮点型

北太天元以双精度(double)或单精度(single)格式表示浮点数。软件中默认为双精度，但可以通过一个简单的转换函数(single)将任何数值转换为单精度数值。

一般使用双精度来存储大于 3.4 x 10^38 或约小于 -3.4 x 10^38 的值。对于位于这两个范围之间的数值，可以使用双精度，也可以使用单精度，但单精度需要的内存更少。

数据类型-浮点型

例如:

```
>>a = 12.25

a =

   12.2500

>>whos
  Name  Size  Bytes  Class   Attributes

  a     1x1       8  double
```

```
>>b = single(12.25)

b =

  1x1 single

   12.2500

>>whos
  Name  Size  Bytes  Class   Attributes

  b     1x1       4  single
```

数据类型-整数

在北太天元中，支持整数型的数据类型主要包括8位、16位、32位和64位的有符号和无符号的整数数据类型。由于在本软件中默认的数据类型是双精度型，故在定义整型数据变量时，需要指定变量的数据类型。

数据类型	说明
uint8	8位无符号整数，数值范围为 0 ~ 255 (0 ~ 2^8-1)
int8	8位有符号整数，数值范围为 -128 ~ 127 (-2^7 ~ 2^7-1)
uint16	16位无符号整数，数值范围为 0 ~ 65535 (0 ~ 2^{16}-1)
int16	16位有符号整数，数值范围为 -32768 ~ 32767 (-2^{15} ~ 2^{15}-1)
uint32	32位无符号整数，数值范围为0 ~ 4294967295 (0 ~ 2^{32}-1)
int32	32位有符号整数，数值范围为 -2147483648 ~ 2147483647 (-2^{31} ~ 2^{31}-1)
uint64	64位无符号整数，数值范围为0 ~ 18446744073709551615 (0 ~ 2^{64}-1)
int64	64位有符号整数，数值范围为 -9223372036854775808 ~ 9223372036854775807 (-2^{63} ~ 2^{63}-1)

例如: 使用uint8定义整数

```
>> uint8(200)
ans =
  1x1 uint8
   200
```

```
>> uint8(300)
ans =
  1x1 uint8
   255
```

```
>> uint8(-10)
ans =
  1x1 uint8
   0
```

可以看到，当需要定义的数据在uint8函数数值范围之内时输出正确，而在范围之外时只显示上（下）限。

例如: 整数类型运算

```
>> x1=int8([1:9])
x1 =
  1x9 int8
   1   2   3   4   5   6   7   8   9
```

```
>> x2=int32([1:9])
x2 =
  1x9 int32
   1   2   3   4   5   6   7   8   9
```

>> x1+x2

错误：您输入的第 0 列附近：整数数组只能与相同类型整数或双精度类型数组运算。

函数执行中显示有错误信息，请反馈给开发团队。

复数

北太振寰

复数是一种特殊的数据结构，把形如a+bi (a,b均为实数）的数称为复数，默认情况是complex double 的类型，也可以建立complex single 的类型。其中，a称为实部，b称为虚部，i称为虚数单位。当虚部等于0（即b=0），这个复数可以视为实数；当z的虚部不等于0，实部等于0（即a＝0且b不为0）时，z=bi，常称z为纯虚数。

复数的四则运算规定如下：

加法法则：$$(a+bi)+(c+di)=(a+c)+(b+d)i$$

减法法则：$$(a+bi)-(c+di)=(a-c)+(b-d)i$$

乘法法则：$$(a+bi)*(c+di)=(ac-bd)+(bc+ad)i$$

除法法则：$$(a+bi)/(c+di)=[(ac+bd)/(c^2+d^2)]+[(bc-ad)/(c^2+d^2)]i$$

在北太天元中，提供有多个函数来方便得到复数的一些基本数值。如：

real(z)	计算复数的实部
imag(z)	计算复数的虚部
abs(z)	计算复数的模
angle(z)	以弧度为单位给出复数的幅角arctan(a/b)

数据类型-复数

例如：复数的显示。

```
%创建复数
>> a=1+3i
a =
  1x1 complex
double
1.0+ 3.0000i
```

```
%复数实部
>> real(a)
ans =
  1x1 double
   1
```

```
%复数虚部
>> imag(a)
 ans =
  1x1 double
   3
```

```
%复数相位角
>> angle(a)
ans =
  1x1 double
    1.2490
```

数据类型-逻辑型

逻辑数据类型就是仅具有“true”和“false”两个数值的一种数据类型。一般来说，逻辑“真”用1表示，逻辑“假”用0表示。在北太天元的逻辑运算中，任何数值都可以参与逻辑运算，北太天元将所有的非零值看做逻辑“真”，将零值看做逻辑“假”。

创建逻辑类型矩阵或者数组的函数主要包含以下3个：

(1)logical函数。可将任意类型的数组转换成逻辑类型数组。其中非零元素为“真”，零元素则为“假”。

(2)true函数。产生全逻辑真值数组。

(3)false函数。产生全逻辑假值数组。

北太振寰

例如: 利用函数建立逻辑类型数组示例

%产生单位矩阵

```
>> a=~eye(3,3)*3
a =
     0     3     3
     3     0     3
     3     3     0
```

%计算逻辑型矩阵b

```
>> b=logical(a)
b =
  3x3 logical
     0     1     1
     1     0     1
     1     1     0
```

%产生全为true的矩阵

```
>> c=true(size(a))
c =
  3x3 logical
     1     1     1
     1     1     1
     1     1     1
```

%产生全为false的矩阵

```
>> d=false(size(a))
d =
  3x3 logical
     0     0     0
     0     0     0
     0     0     0
```

在北太天元中，用单引号包含几个字符（Character）即可以构成一个字符向量。

一个字符向量被视为一个行向量，而字符向量的每一个字符（含空格符），则以其字符编码的形式存放于此向量的每一个元素中，只是它的对外显示形式仍然是可读的字符。也可以由双引号包含文本形成字符串，字符向量与字符串类型在数据的可视化、应用程序的交互方面，有着非常重要的作用。

北太振寰

例如： 一般字符向量与字符串的创建

在北太天元中，所有的字符向量都用两个单引号括起来，进行输入赋值。如在北太天元命令窗口输入：

```
>> a='baltamatica'
a =
  1x11 char
    'baltamatica'
```

字符向量的每个字符（空格也是字符）都是相应矩阵的一个元素。上述变量a是1×11阶的矩阵，可以用size(a)命令查得：

```
>> size(a)
ans =
     1    11
```

%1行11列

字符串则是用两个双引号括起来，创建一般字符串如下输入:

```
>> b="baltamatica"
b =
  1x1 string
    "baltamatica"
```

一个字符串为一个元素，使用size(b)命令可以查询到：

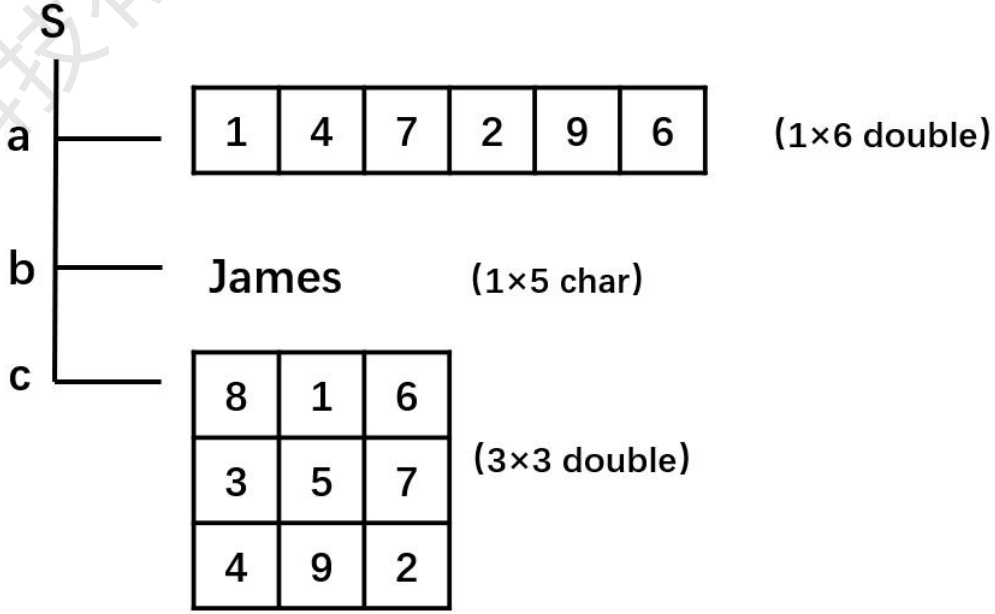
```
>> size(b)
ans =
     1     1
```

%1行1列

数据类型-结构体

结构体（struct）是北太天元提供的一种将选择的数据存储到一个实体的数据类型。一个结构体由数据容器组成，这种容器叫做域（field），每个域中可以存储北太天元支持的数据类型。用户能通过访问存储数据时指定的域名来对域中数据进行访问。

图3-1是一个包括了a、b和c三个域的结构体S的示意图。



数据类型-结构体

北太振寰

- 通常情况下，使用结构体（或者下面提到的元胞数组）的原因是在实际应用中需要存储多种混合的数据类型和大小。因为一般的北太天元数组只能存储同样大小的同种数据类型的元素。结构体和元胞数组就是重要的混合数据类型存储手段。

- 结构体提供了在一个实体中存储特定数据的方法，这可以使用户对数据进行整体或者部分访问与操作。同时用户可以将函数直接运用于结构体，在用户自定义的M文件函数之间进行数据传递，显示结构体任何域中的值，或者进行支持结构体类型的任何北太天元操作。

- 用户通过结构体可以给数据以文字标签，这样在应用中能清楚地对数据所包含的信息进行标注。

数据类型-结构体

例如：直接赋值法创建结构体实例，以结构体保存职工资料数据。

```
>> staff.name='jerry';
>> staff.gender='male';
>> staff.age='30';
>> staff.number='9797997';
```

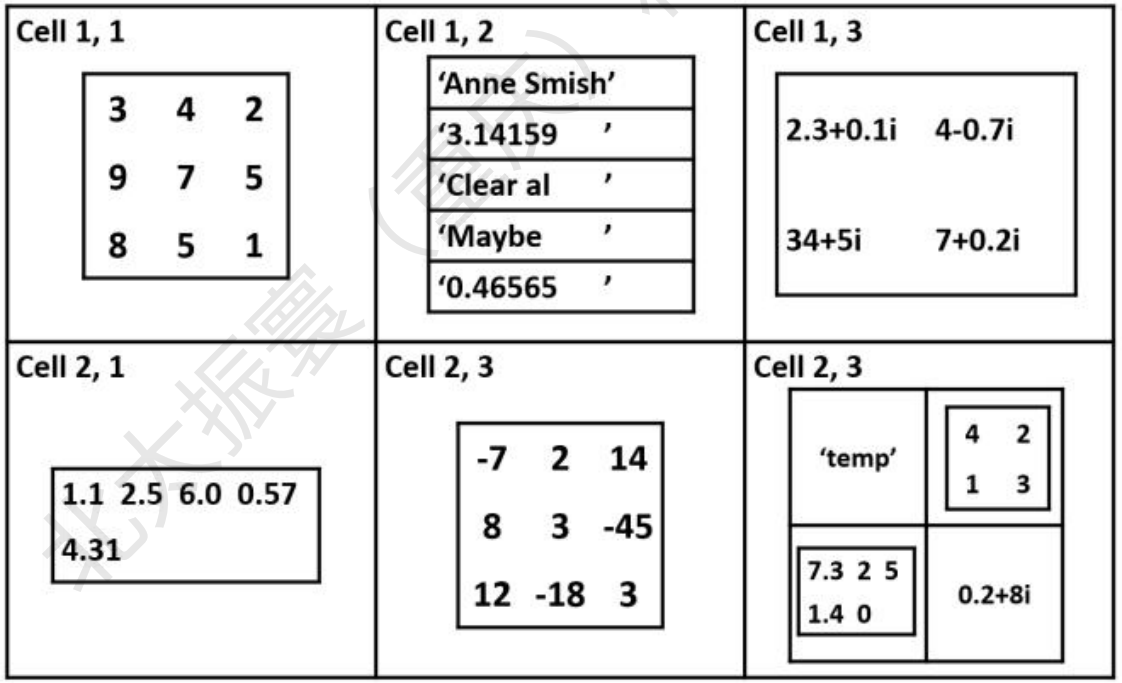
```
>> staff
staff =
  1x1 struct
    name: 'jerry'
  gender: 'male'
     age: '30'
  number: '9797997'
```

staff 即是以结构体类型存储的数据。

元胞数组（cell）是北太天元的一种特殊数据类型。可以将元胞数组看做一种无所不包的通用矩阵，或者叫做广义矩阵。组成元胞数组的元素可以是任何一种允许的数据类型的常数或者常量，每一个元素可以有不同的尺寸和内存占用空间，每一个元素的内容也可以完全不同。

数据类型-元胞数组

和一般的数值矩阵一样，元胞数组的内存空间也是动态分配的。下图是元胞数组的结构示意图，其表示的是一个2×3的元胞数组。元胞数组的第1行包括了无符号整数、字符向量数组和一个复数数组，第2行包括了其他3种类型的数组，其中最后一个是另外一个元胞数组的嵌套。



数据类型-元胞数组

元胞数组可以是一维的、二维的。对其中元素进行寻访，可以使用“单下标”方式或者“全下标”方式。

结构体和元胞数组的区别如下：

- 结构体和元胞数组在使用目的上类似，都是提供一种存储混合格式数据的方法。二者最大的区别在于：结构体存储数据的容器称为“域”，而元胞数组是通过数字下标索引来进行访问的。
- 结构体经常用于重要数据的组织存储，而元胞数组因为采用数字下标，所以经常在循环控制流中使用。元胞数组还经常被用来存储不同长度的字符向量。
- 在实际应用中，用户可以根据自己的习惯和实际应用来决定。

在表现形式上，元胞数组和一般矩阵一样，元胞数组的大小也必须是“长方形”。一般矩阵的创建使用中括号“[ ]”，而创建元胞数组使用的是花括号“ {} ”。元胞数组的创建方式与矩阵的创建方式类似，只需要将中括号“[ ]”替换为花括号“ {} ”即可。在元胞数组创建的过程中使用逗号或者空格来分隔元素，使用分号来分行。

例如：创建元胞数组实例。

```
>> A={[1 5 6 1 2;8 5 6 1 2;7 4 5 2 1],'hello hello';[2+5i, 6-4i],-pi:pi/4:pi}
A =
  2x2 cell array
  列 1
        {3x5 double }
        {[2.0000 + 5.0000i 6.0000 - 4.0000i]}
  列 2
        {'hello hello' }
        {[-3.1416 -2.3562 -1.5708 -0.7854 ...]}
```

注意：元胞数组获取时使用 () 和 {} 的区别

```
>>a={[1,2,3],'hello',magic(3)}

a =

  1x3 cell array

    {[1 2 3]}    {'hello'}    {3x3 double}
```

```
>> a(3) % 使用()获取的是子元胞

ans =

  1x1 cell array

    {3x3 double}
```

```
>>a{3} % 使用{}获取的的是子元胞的具体内容

ans =

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

数据类型-函数句柄

函数句柄是一种存储指向函数的关联关系的数据类型。函数句柄的创建可以通过特殊符号@引导函数名来实现。

实例--创建sin函数

```
>>fun_handle=@sin

fun_handle =

  1x1 function_handle

    @sin
```

使用句柄调用函数的方式与直接调用函数一样。例如下面两种调用方式是等价的：

```
>> fun_handle(20)

ans =

    0.9129

>> sin(20)

ans =

    0.9129
```

数据类型-函数句柄

函数句柄的调用可以通过feval进行，格式如下。

[y1,y2,...]=feval(fhandle,x1,...,xn)

其中，fhandle为函数句柄的名称，x1,...,xn为参数列表。

这种调用相当于执行以参数列表为输入变量的函数句柄所对应的函数。

实例--差值计算

北太天元程序如下：

（1）创建函数文件

```
function
f=test2(x,y)
    f=x-y;
end
```

（2）创建test函数的函数句柄

```
>> fhandle=@test2
fhandle=
  1x1 function handle
    test2
```

（3）调用该句柄

```
>> feval(fhandle,1,2)

ans =
     -1
```

匿名函数

匿名函数相对函数文件来讲比较简单，由一条表达式组成，能够接受多个输入或输出参数。使用匿名函数可以避免文件的管理和存储，但是匿名函数的执行效率比较低，会占用较多的时间。构建匿名函数的语法形式是：

fhandle = @(变量) expr

现在从右向左解释一下这个语法结构： expr为北太天元表达式，是函数的主体，即执行函数要完成的任务。变量为输入变量列表，用逗号分隔。@为北太天元的操作符，用于建立函数句柄。

实例--计算一个数的平方：

```
>>sqr = @(x) x.^2;
sqr =

  1x1 function_handle

    @(x)x.^2
```

实践：用匿名函数创建下列函数，并计算 x=2 的值

$$(a)\, f(x) = e^{x} - 2 - x$$

$$(b)\, f(x) = \cos(x) + 1 - x$$

$$(c)\, f(x) = \ln(x) - 5 + x$$

$$(d)\, f(x) = x^{2} - 10x$$

目录 CONTENTS

02 数组和矩阵创建

什么是数组？什么是矩阵？

数组（Array）是一种数据结构，用来存储一系列相同类型的元素，这些元素可以通过索引（即位置编号）来访问。

- 可以用方括号[]来表示数组，例如[1, 2, 3, 4, 5]。
- 每个元素都有一个索引，从1开始计数，所以上述数组的索引为1, 2, 3, 4, 5。

例子：

- 一维数组：假设表示学生分数的数组[85, 90, 78, 92]，可以通过索引来访问学生的分数。
- 二维数组：一个班级的座位表，每一行代表一组座位，每一列代表一排座位。

矩阵是由行和列组成的二维数组，它是数学和科学领域中非常重要的一个概念。

- 矩阵中的元素是通过行号和列号来定位的。
- 矩阵可以进行数学运算，比如加法、乘法等。

例子：

一个班级的学生身高和体重数据可以组成一个矩阵，每一行代表一个学生，第一列是身高，第二列是体重。

区别和联系：

- 矩阵是二维的数组，一维数组可以看作是 1×n 或 n×1 的矩阵（只有一行的矩阵或只有一列的矩阵）。
- 数组可以是多维的(0，1，2，3,…)，而矩阵是二维的。

一维数组的创建

概括而言，创建一维数组时，可以通过以下几种方法来进行。

(1)直接输入法：此时，可以直接通过空格、逗号和分号来分隔数组元素，在数组中输入任意的元素，生成一维数组。

(2)步长生成法：x=a:inc:b，在使用这种方法创建一维数组时，a和b为一维向量数组的起始数值和终止数值，inc为数组的间隔步长；如果a和b为整数时，省略inc可以生成间隔为1的数列。根据a和b的大小不同，inc可以采用正数，也可以采用负数来生成一维向量数组。

(3)等间距线性生成方法：x=linspace(a,b,n)，这种方法采用函数在a和b之间的区间内得到n个线性采样数据点。

(4)等间距对数生成方法：x=logspace(a,b,n)，采用这种方法时，在设定采样点总个数n的情况下，采样常用对数计算得到n个采样点数据值。（即：在 10 的幂 10 ^ a 和 10 ^ b（10 的 N 次幂）之间生成 n 个点。）

数组创建形式

(1) 要创建每行包含四个元素的数组，请使用逗号或空格分隔各元素，如下所示行向量的创建。

```
>>a = [1 2 3 4]
a =

   1   2   3   4
```

(2) 要创建包含多行的矩阵，请使用分号分隔各行。

```
>>a = [1 3 5; 2 4 6; 7 8 10]
a =

   1   3   5
   2   4   6
   7   8  10
```

数组创建形式

```
>> x1=[0,pi,0.3*pi,1.5,2]      %直接输入数据生成数组
x1 =
    0.0000    3.1416    0.9425    1.5000    2.0000
>> x2=0:0.6:6          %步长生成法
x2 =
  列 1 -- 6
    0.0000    0.6000    1.2000    1.8000    2.4000    3.0000
  列 7 -- 11
    3.6000    4.2000    4.8000    5.4000    6.0000
>> x3=linspace(1,6,7)      %等间距线性生成法
x3 =
  列 1 -- 6
    1.0000    1.8333    2.6667    3.5000    4.3333    5.1667
  列 7
    6.0000
>> x4=logspace(1,4,5)      %等间距对数生成法
x4 =
  1.0e+04 *
    0.0010    0.0056    0.0316    0.1778    1.0000
```

数组创建形式

创建矩阵的另一种方法是使用 ones、zeros 或 rand 等函数。例如，创建一个由零组成的 5×1 列向量。

```
>>z = zeros(5,1)
z =

     0
     0
     0
     0
     0
```

```
>>z = ones(5,1)
z =

     1
     1
     1
     1
     1
```

```
>>z = rand(5,1)
z =

    0.5928
    0.8443
    0.8579
    0.8473
    0.6236
```

矩阵的生成

北太振寰

直接输入法：

- 在命令行直接按行方式输入矩阵，适合较小的简单矩阵。
- **语法要求**：

1. 输入矩阵以“[ ]”为标识符号，矩阵的所有元素都必须包含在括号内；
2. 矩阵同行元素之间由空格或逗号分隔，行与行之间由分号分隔；
3. 矩阵大小不需要事先定义；
4. 若“[ ]”中无元素，表示空矩阵；
5. 如果不想显示中间结果，可以用分号“;”结束。

矩阵的生成

北太振寰

Eg.

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

A =

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

Eg.

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

ans =

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

矩阵的生成

北太振寰

M文件生成法：

- 当矩阵的规模比较大时，直接输入法就显得笨拙，出差错也不易修改。
- 为了解决这些问题，可以将所要输入的矩阵按格式先写入一文本文件中，并将此文件以 .m 为其扩展名，即M文件。
- M文件是一种可以在北太天元环境下运行的文本文件，它可以分为命令式文件和函数式文件两种。
- 在此处主要用到的是命令式M文件，用它的简单形式来创建大型矩阵。
- 在北太天元命令行窗口中输入M文件名，大型矩阵即可被输入到内存中。

实例：

（1）在脚本编辑器中编制一个名为 sample.m 的M文件，在文件中编制一个名为 gmatrix 的矩阵：

（2）运行M文件，在北太天元命令行窗口中输入文件名，创建矩阵

矩阵的生成

sample.m

```
% 在M文件中，输入大规模矩阵gmatrix
gmatrix = [378 89 90 83 382 92 29;
3829 32 9283 2938 378 839 29 ;
388 389 200 923 920 92 7478;
3829 892 66 89 90 56 8980;
7827 67 890 6557 45 123 35]
```

Eg.

```
>> sample

gmatrix=
  5x7 double matrix
  1.0e+03 *
    0.3780   0.0890   0.0900   0.0830   0.3820   0.0920   0.0290
    3.8290   0.0320   9.2830   2.9380   0.3780   0.8390   0.0290
    0.3880   0.3890   0.2000   0.9230   0.9200   0.0920   7.4780
    3.8290   0.8920   0.0660   0.0890   0.0900   0.0560   8.9800
    7.8270   0.0670   0.8900   6.5570   0.0450   0.1230   0.0350
```

矩阵的生成

北太振寰

csv文件生成法：

矩阵还可以由csv文件创建，在软件菜单栏中选择“数据”选项中的“导入数据”，选择对应的csv文件，选中要导入的数据，命名变量后即可导入文件数据。

北太振寰

常用函数

- eye(n)　　创建 $n \times n$ 单位矩阵
- eye(m, n)　　创建 $m \times n$ 单位矩阵
- ones(n)　　创建 $n \times n$ 全1矩阵
- ones(m, n)　　创建 $m \times n$ 全1矩阵
- zeros(m, n)　创建 $m \times n$ 全0矩阵

北太振寰

常用函数

- rand(n)　　在[0, 1]区间内创建一个 n×n 均匀分布的随机矩阵
- rand(m, n)　在[0, 1]区间内创建一个 m×n 均匀分布的随机矩阵
- diag(v)　　创建一向量v中的元素为对角的对角阵
- magic(n)　　生成 n 阶魔方矩阵

创建特殊矩阵

北太振寰

Eg.

```
>> zeros(3)

ans =

     0     0     0
     0     0     0
     0     0     0

>> zeros(2,3)

ans =

     0     0     0
     0     0     0
```

Eg.

```
>> ones(3)

ans =

     1     1     1
     1     1     1
     1     1     1

>> ones(2,3)

ans =

     1     1     1
     1     1     1
```

Eg.

```
>> magic(3)

ans =

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

Eg.

```
>> rand(3)

ans =

    0.5928    0.8473    0.2975
    0.8443    0.6236    0.0567
    0.8579    0.3844    0.2727

>> rand(2,3)

ans =

    0.4777    0.4800    0.8361
    0.8122    0.3928    0.3374
```

目录 CONTENTS

03 数组和矩阵运算

引用数组特定元素的方法

北太振寰

寻址方法	说明
A(r,c)	用定义的 r 和 c 索引向量来寻址 A 的子数组
A(r,:)	用 r 向量定义的行和对应行的所有列得到 A 的子数组
A(:,c)	用 c 向量定义的列和对应列的所有行得到 A 的子数组
A(:)	用列向量方式来依次寻址数组 A 的所有元素。如果 A(:) 出现在等号的左侧，表明用等号右侧的元素来填充数组。而 A 的形状不发生变化
A(k)	用线性索引（按列排序进行索引）向量 k 来寻找 A 的子数组
A(x)	用逻辑数组 x 来寻找 A 的子数组，x 的维数和 A 的维数必须一致

指定行和列下标的引用

北太振寰

北太天元中的每个变量都是一个可包含许多数字的数组。如果要访问数组的选定元素，就要用索引。

```
>>A = [1 2 3 4; 5 6 7 8; 9 10 11 12; 13
14 15 16]
A =

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

指定行和列下标的方式引用数组中的特定元素

```
>>A(1,3)
ans =

     3
>>A(4,2)
ans =

    14
```

指定行和列下标的引用

北太振寰

行列下标索引方式可以和“:”符号来结合使用，

可以获得数组中的多个元素值，比如

```
>> A = [1 2 3 4; 5 6 7 8; 9 10 11 12; 13
14 15 16];
>> A(2:4,3) % 对特定维度中某个范围的元素进行索引
ans =

     7
    11
    15
```

```
>> A(:,3)    %对特定维度中的所有元素进行索引

ans =

     3
     7
    11
    15
```

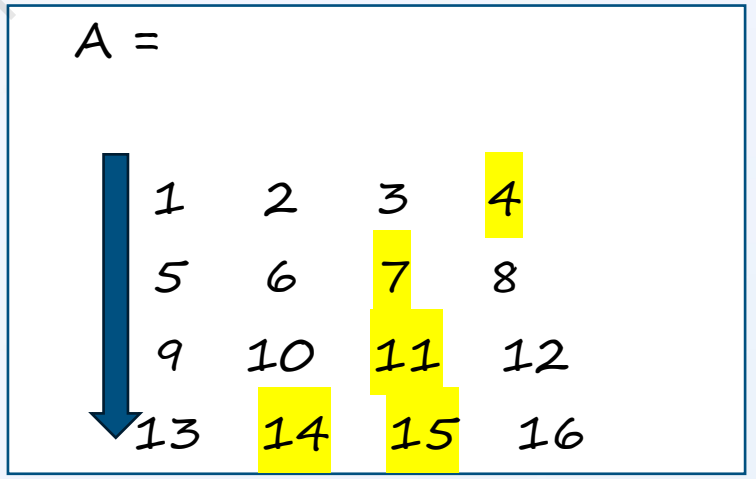
不太常用的引用形式

指定单一下标的方式引用数组中的特定元素，会优先按列的顺序取值。

```
>> A(8)
ans =

   14
>> A(10:13)  % 取10-13的元素
ans =

    7   11   15    4
```



使用单一下标引用数组中特定元素的方法称为线性索引。

逻辑 1 标识法

逻辑 1 标识法用来解决需要寻找数组中大于或小于某值的元素的问题。该方法是指用一个与需要寻访的数组同维度的逻辑数组来对数组进行寻访。逻辑数组中每一个true（1）值表示寻访数组对应位置上的元素。

```
>> B=A>10    %  返回逻辑下标

B =

  4x4 logical

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

```
>> C=A(B)        %  逻辑下标寻访

C =

    13
    14
    11
    15
    12
    16
```

```
A =

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

矩阵元素的修改

矩阵元素的修改：

命令名	说明
D=[A;B C]	A为原矩阵，B、C中包含要扩充的元素，D为扩充后的矩阵
A(m,:)=[]	删除A的第m行
A(:,n)=[]	删除A的第n列
A(m,n)=a	对A的第m行第n列的元素赋值
A(m,:)=[a b …]	对A的第m行赋值
A(:,n)=[a b …]	对A的第n列赋值

矩阵元素的增加或拼接

（1）矩阵可以像拼图一样进行元素的拼接形成新的一个矩阵。

```
>> A = [1 2 3;4 5 6];
>> B = eye(2);
>> C = zeros(2,1);
>> D = [A; B C]  % 矩阵拼接

D =

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

数组和矩阵可以进行扩充

```
>> A = [1 2 3 4 5]; % 向量扩充
>> A(end+1) = 6
A =

   1   2   3   4   5   6

>> A = [1 2 3;4 5 6]; % 矩阵扩充
>> A(end+1,:) = [7 8 9]

A =

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

矩阵元素的删除

（2）矩阵可以批量删除其中某些行或某些列元素值。

```
>> A = [1 2 3 4; 5 6 7 8; 9 10 11 12; 13
14 15 16]
>> A =

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

```
>> A(2,:)=[]  % 删除第二行

A =

    1    2    3    4
    9   10   11   12
   13   14   15   16

>> A(:,3)=[] % 删除第二列

A =

    1    2    4
    9   10   12
   13   14   16
```

使用线性索引进行单个删除：A(5)=[]

矩阵元素的修改

北太振寰

（3）矩阵还可以进行批量赋值修改。比如

```
>> A  = zeros(5,5) % 创建 5×5 的矩阵

A =

     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

>> A(2,3) = 1 % 单个赋值

A =

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

>> A(2:3,:) = 5 %对 2、3 两行都赋值为 5

A =

     0     0     0     0     0
     5     5     5     5     5
     5     5     5     5     5
     0     0     0     0     0
     0     0     0     0     0

>> A(:,2) = [1;2;3;4;5] % 对整列赋值

A =

     0     1     0     0     0
     5     2     5     5     5
     5     3     5     5     5
     0     4     0     0     0
     0     5     0     0     0
```

矩阵元素的大小

矩阵的变维：

矩阵的变维可以用符号“:”法和reshape函数法。

reshape函数的调用形式如下：reshape(X, m, n) 将已知矩阵变维成m行n列的矩阵。

Eg.

```
>> A = 1:12;
>> C = zeros(3,4);
>> C(:) = A(:);
>> C
```

此时第2、3行等价于C = reshape(A, [3,4])

```
C =

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

矩阵元素的运算

矩阵的变向：

命令名	说明
rot90(A)	将A逆时针方向旋转90°
rot90(A, k)	将A逆时针方向旋转90°*k，k可为正/负整数
fliplr(X)	将X左右翻转
flipud(X)	将X上下翻转
flipdim(X, dim)	dim=1时对行翻转，dim=2时对列翻转

矩阵元素的运算

北太振寰

Eg.

```
>> A = magic(3)

A =

   6   1   8
   7   5   3
   2   9   4

>> B = rot90(A,2) % 逆时针旋转180°

B =

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

Eg.

```
>> fliplr(A) % 左右翻转

ans =

   8   1   6
   3   5   7
   4   9   2

>> flipdim(A,1) % 对行进行翻转

ans =

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

矩阵元素的运算

北太振寰

矩阵元素的抽取：

对矩阵元素的抽取主要是指对角元素和上（下）三角阵的抽取。

命令名	说明
diag(X,k)	抽取矩阵X的第k条对角线上的元素向量。k=0时抽取主对角线， k为正整数时抽取上方第k条对角线上的元素，k为负整数时抽取下方第k条对角线上的元素
diag(X)	抽取主对角线
diag(v,k)	使得v为所得矩阵第k条对角线上的元素向量
diag(v)	使得v为所得矩阵主对角线上的元素向量
tril(X)	提取矩阵X的主下三角部分
tril(X,k)	提取矩阵X的第k条对角线下面的部分（包括第k条对角线）
triu(X)	提取矩阵X的主上三角部分
triu(X,k)	提取矩阵X的第k条对角线上面的部分（包括第k条对角线）

矩阵元素的运算

北太振寰

Eg.

```
>> A = magic(3)

A =

   6   1   8
   7   5   3
   2   9   4

>> v = diag(A,2)

v =

   8
```

Eg.

```
>> tril(A,-1)

ans =

   0   0   0
   7   0   0
   2   9   0

>> triu(A)

ans =

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

矩阵的算术和函数运算

(1) 单一的算术运算，则使用情况如下：

```
>>z = zeros(5,1)+2
z =

   2
   2
   2
   2
   2
```

(2)函数运算，则使用示例如下：

```
>>a=[1 3 5;2 4 6;7 8 10];
>>sin(a)
ans =

   0.8415    0.1411   -0.9589
   0.9093   -0.7568   -0.2794
   0.6570    0.9894   -0.5440
```

矩阵的加减法运算

设 $A=(a_{ij})$，$B=(b_{ij})$ 都是 $m\times n$ 矩阵，矩阵A和B的和记为A+B，规定为：

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

1. 交换律：A+B=B+A
2. 结合律：(A+B)+C=A+(B+C)

矩阵的加减法运算

```
>> A = [5,6,9;5,3,6]

A =

   5   6   9
   5   3   6
```

```
>> B = [3,6,7;5,8,9]

B =

   3   6   7
   5   8   9
```

```
>> C = [9,3,5;8,5,2]

C =

   9   3   5
   8   5   2
```

```
>> A+B

ans =

    8   12   16
   10   11   15

>> B+A

ans =
    8   12   16
   10   11   15
```

```
>> (A+B)+C

ans =
   17   15   21
   18   16   17

>> A+(B+C)
ans =

   17   15   21
   18   16   17
```

```
>> -B

ans =
   -3   -6   -7
   -5   -8   -9

>> A-B
ans =

    2   0   2
    0  -5  -3
```

矩阵的乘法运算

北太振寰

矩阵的乘法分为线性代数乘法和元素乘法，这两个在使用上有很大的区别，设 $A=(a_{ij})$，$B=(b_{ij})$ 都是 $m\times n$ 矩阵，矩阵A和B的代数乘积为A*B，矩阵A和B的元素乘积为A.*B，规定为：

矩阵代数乘法： $(A*B)_{(i,j)}=\sum_{k=1}^{n}A(i,k)*B(k,j)$

代数乘法要求两个矩阵的行和列必须满足矩阵乘法的条件，即第一个矩阵的列数必须等于第二个矩阵的行数。

元素乘法：

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

元素乘法要求两个矩阵的维度是要相同的

矩阵的乘法运算

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

A =

   1   3
   2   4
```

```
>> B = [3 0;1 5]

B =

   3   0
   1   5
```

```
>> C = [9,3; 5,2]

C =

   9   3
   5   2
```

```
>> A*B

ans =

    6   15
   10   20

>> A.*B

ans =

    3    0
    2   20
```

```
>> B*A

ans =

    3    9
   11   23

>> B.*A

ans =

    3    0
    2   20
```

```
>> A*B*C

ans =

   129   48
   190   70

>> A.*B.*C

ans =

   27    0
   10   40
```

矩阵的除法运算

同样地，符号 / 和 ./ 分别代表矩阵除法和元素除法。具体来说：

- 矩阵除法（/）：当使用 / 符号时，北太天元执行的是矩阵除法。这通常用于求解线性方程组。例如，如果 A 是一个 3x3 矩阵，b 是一个 1x3 向量，那么 x = b / A 将求解方程 x*A = b。
- 元素除法（./）：当使用 ./ 符号时，北太天元执行的是元素除法。这意味着两个矩阵的维度必须相同，结果矩阵中的每个元素是两个输入矩阵中对应位置元素的商。例如，如果 A 和 B 都是 3x3 矩阵，那么 C = A ./ B 将是一个 3x3 矩阵，其中 C(i,j) = A(i,j) / B(i,j)。

矩阵的除法运算

解以下线性方程组：

$$\begin{cases} 2x + 3y = 8 \\ 4x - y = 2 \end{cases}$$

将这个方程组表示为矩阵形式
x*A=b

```
>> A = [2 4; 3 -1];
>> b = [8  2];
>> x = b / A
x =

   1   2
```

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

计算 A./B,对矩阵 A 和 B 的对应元素进行逐个相除

```
>> c = A./B
c =

   0.5000   0.7500
   0.8333   0.8750
```

```
>> A = [2 4; 6 8];
>> b = 2;
```

标量与矩阵的元素除法,会将标量扩展为与矩阵相同维度的矩阵，然后逐元素除:

```
>> C = A ./ b
C =

   1   2
   3   4
```

培 训 反 馈 问 卷

QQ扫码入群 参与培训

入群即享 多重福利

•课程直播链接 •独家学习礼包 •专业技术支持

北太振寰

成 为 世 界 领 先 的 科 学 计 算 软 件 提 供 者

联系方式：market@baltamatica.com

官方网站

官方公众号