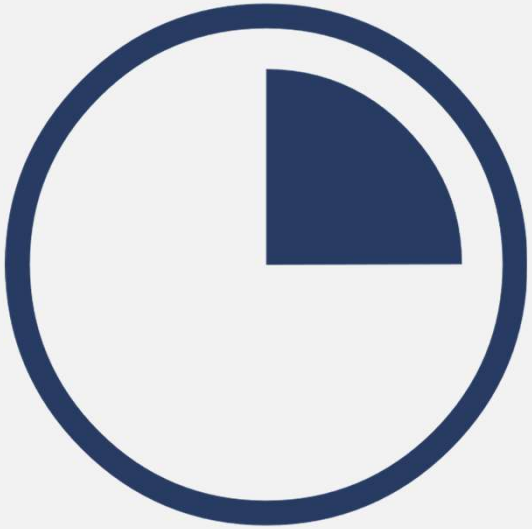
光电信息问题的MATLAB数学建模实验

School of Precision Instrument and Opto-electronics Engineering
Tianjin University

2024.09.02



PART 03
MATLAB矩阵的操作

一、复习矩阵的基础知识

【矩阵的定义】由$m\times n$个元素组成的m行n列的数表$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$称为一个$m\times n$阶矩阵，记为$A=(a_{ij})_{m\times n}$，当$m=n$时，我们称矩阵$A$为$n$阶方阵（或$n$阶矩阵）。

【同型矩阵】当矩阵A，B的行数和列数都相同时，我们称矩阵A,B为同型矩阵。

【转置矩阵】设$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$，$\begin{bmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{mn} \end{bmatrix}$称为矩阵$A$的转置矩阵，记为$A^T$或$A'$.

一、复习矩阵的基础知识

【向量的定义】由n个数构成的数表$[a_1,a_2,\cdots,a_n]$或$[a_1,a_2,\cdots,a_n]^T$称为n维行向量或n维列向量。显然，向量是矩阵的特例，行向量的行数为1，列向量的列数为1.

【向量的模】设向量$\alpha=[a_1,a_2,\cdots,a_n]^T$,称$\sqrt{a_1^2+a_2^2+\cdots+a_n^2}$为向量$\alpha$的模，记为$|\alpha|$.

【矩阵的加减法】设矩阵$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$, $B=\begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{bmatrix}$，则$A\pm B=$

$\begin{bmatrix} a_{11}\pm b_{11} & a_{12}\pm b_{12} & \cdots & a_{1n}\pm b_{1n} \\ a_{21}\pm b_{21} & a_{22}\pm b_{22} & \cdots & a_{2n}\pm b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}\pm b_{m1} & a_{m2}\pm b_{m2} & \cdots & a_{mn}\pm b_{mn} \end{bmatrix}$。显然，只有两个矩阵同型时才能相加减。

一、复习矩阵的基础知识

【数与矩阵的乘法】设 $A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$，则 $kA=\begin{bmatrix} ka_{11} & ka_{12} & \cdots & ka_{1n} \\ ka_{21} & ka_{22} & \cdots & ka_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ ka_{m1} & ka_{m2} & \cdots & ka_{mn} \end{bmatrix}$.

【矩阵与矩阵的乘法】设矩阵 $A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$，$B=\begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1s} \\ b_{21} & b_{22} & \cdots & b_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{ns} \end{bmatrix}$，则 $AB=$

$\begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1s} \\ c_{21} & c_{22} & \cdots & c_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{ms} \end{bmatrix}$，其中，$c_{ij}=a_{i1}b_{1j}+a_{i2}b_{2j}+\cdots+a_{in}b_{nj}(i=1,2,\cdots,m;j=1,2\cdots,s)$. 显然，两个矩阵的乘法必须满足左边矩阵的列数与右边矩阵的行数相等。

一、复习矩阵的基础知识

【单位矩阵】称$E=\begin{bmatrix}1 & 0 & \cdots & 0\\ 0 & 1 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & 1\end{bmatrix}$为单位矩阵。（部分教材用符号$I$表示单位矩阵。单位矩阵一定为方阵）

【矩阵的逆】设A是n阶矩阵，若存在n阶矩阵B，使得$BA=E$或$AB=E$，则称矩阵A可逆，矩阵B称为矩阵A的逆矩阵，记为$B=A^{-1}$.

【特征值和特征向量】设n阶方阵A满足以下条件：存在数λ（λ可为复数）和非零n维列向量x，使得$Ax=\lambda x$成立，则称数λ是方阵A的特征值，称x为方阵A对应于特征值λ的特征向量。注意，特征向量并不唯一，$kx(k\neq 0)$也是方阵A对应于特征值λ的特征向量。

二、MATLAB中的向量

天津大学 Tianjin University

1.向量的创建方法

（1）直接输入法

向量元素需要用英文的中括号“[ ]”括起来，元素之间用空格、逗号、分号或按回车键分隔，就可以创建对应的向量。若元素之间用空格、逗号分隔，则创建的是行向量；若用分号、回车键分隔，则创建的是列向量。（注意：这里的逗号和分号都是英文输入法下输入的，不能用中文的逗号或分号）

举例：a = [1 3 5]和 a = [1,3,5]都可以创建包含元素 1,3,5 的行向量，并将这个行向量的值赋值给 a；而 b = [1:3:5]创建的是包含元素 1,3,5 的列向量。

二、MATLAB中的向量

1.向量的创建方法

（2）冒号法：最常用

我们可以利用命令：A:step:B 来创建一个行向量。（冒号也要是英文的!）

其中，A 是起始值，step 是每次递增或递减的步长，B 是终止值（不一定刚好停在这里）。

若 step 等于 1，则可以直接简写成 A:B。

二、MATLAB中的向量

天津大学 Tianjin University

1.向量的创建方法

代码	结果和相应的解释
1:2:7	[1 3 5 7]　% 每次增加 2，直到最后到了 7
1:2:8	[1 3 5 7]　% 每次增加 2，到了 7 后，如果再增加 2 的话结果等于 9，比 8 要大，所以到了 7 就停止了。
1:2:9	[1 3 5 7 9]
2:3:18	[2 5 8 11 14 17]
0:0.1:1	[0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1]　% 每次增加 0.1
0:0.01:1	[0 0.01 0.02 0.03 …… 0.98 0.99 1]　% 每次增加 0.01
1:1:100 或简写成 1:100	[1 2 3 4 5 6 7 8 …… 98 99 100]　% 步长为 1 时可以省略
1:10:3	1　% 从 1 开始，增加 10 等于 11，比 3 还要大，所以返回 1
5:2:1	空的 1×0 double 行向量　% 若 A > B 且步长 step > 0，则会返回空的向量。
10:-1:6	[10 9 8 7 6]　% 步长为-1，因此会从 10 开始递减
10:-2:5	[10 8 6]　% 步长为-2，从 10 开始递减，到了 6 后，如果再减去 2 就等于 4，比 5 还要小，所以到了 6 就停止了。
10:-100:5	10　% 步长为-100,因为 10-100 = -90 比 5 还要小，所以返回 10
10:-10:50	空的 1×0 double 行向量　% 若 A < B 且步长 step < 0，则会返回空的向量。
1:0:2	空的 1×0 double 行向量　% 若 step = 0，则返回空的向量。

- （）表中，有三种情况都会导致MATLAB返回空的向量：空的1×0 double行向量。 “1×0”指的是向量的维度，可以理解为1行0列，即这个向量是空的，不存在元素。在**MATLAB**中，我们可以直接使用命令[ ]创建空的向量。
- MATLAB返回空的向量时，出现了double。这里的double表示双精度浮点型（本课程中并没有特意介绍数值的类型）。

二、MATLAB中的向量

1.向量的创建方法

（3）利用 MATLAB 函数创建

我们主要介绍两个函数: **linspace** 和 **logspace**，它们分别用来创建等差数列和等比数列。

首先介绍 **linspace** 函数，它有两种用法，区别在于是否给定第三个输入参数 n，如果我们不指定 n，则 MATLAB 会默认 n=100。这个函数使用的频率也很高，大家需要掌握。

- linspace(a, b)：该命令用来创建一个行向量，向量中的第一个元素为 a，最后一个元素为 b，形成总数为 100 个元素的线性间隔的向量。
- linspace (a, b, n)：该命令用来创建一个行向量，向量中的第一个元素为 a，最后一个元素为 b，形成总数为 n 个元素的线性间隔的向量。

二、MATLAB中的向量

1.向量的创建方法

代码	结果和相应的解释
linspace(1,100,10)	[1 12 23 34 45 56 67 78 89 100]　% 第一个数为 1，最后一个数为 100，整个向量构成了一个等差数列，由 10 个元素组成
linspace(1,99,10)	[1 11.8889 22.7778 33.6667 44.5556 55.4444 66.3333 77.2222 88.1111 99] % 第一个数为 1，最后一个数为 99，整个向量由 10 个元素组成，构成了一个等差数列，MATLAB 会自动计算等差数列的步长。
linspace(0.2*pi,10)	[0 0.69813 1.3963 2.0944 2.7925 3.4907 4.1888 4.8869 5.5851 6.2832] % 第一个数为 0，最后一个数为 2*pi，注意中间的乘号千万不能省略！
linspace(1,10)	[1 1.0909 1.1818 1.2727 1.3636　……　9.8182 9.9091 10] % 如果不指定第三个输入参数 n，则默认生成 100 个元素的等差数列
linspace(100,1,10)	[100 89 78 67 56 45 34 23 12 1]　% 如果 a>b,则步长是负数

二、MATLAB中的向量

2.向量元素的引用

索引（或下标）的概念

我们知道，向量分为行向量和列向量，它们在 MATLAB 中只有一个维度，因此我们可以利用向量中包含的元素个数来描述一个向量的大小。**在 MATLAB 中，可以使用 length 函数或 numel 函数来计算向量中包含的元素个数。**

例如：a = [1,3,8,9,7]; length(a)或 numel(a)的返回结果是 5，因为向量 a 中有五个元素。

假如我们有一个行向量 a，里面包含了 n 个元素（n 是大于等于 1 的常数），它们分别是 $a_1, a_2, \ldots, a_n$. 那么我们可以列一个表格：

向量的元素	a_1	a_2	a_3	$\cdots$	a_{n-1}	a_n
索引（下标）	1	2	3	$\cdots$	$n-1$	n

从上表可以看出，索引就是指某一个元素在向量中对应的位置，也可以称为元素在向量中所处的下标，在 MATLAB 中，向量的索引是从 1 开始的。

举个具体的例子，假设向量 a=[2 4 8 16 32 64 128 256 512 1024]，那么 a 中有 10 个元素，因此 a 的最大索引是 10。

2.向量元素的引用

(1) 单个元素引用

- 我们提取向量**a**中单个元素的方法很简单，只需要利用**a(ind)**命令，小括号中的**ind**就是你要提取的对应元素的索引。（注意：创建向量用中括号，提取元素要用小括号）

例如：a(1)的结果为2，因为a中第1个位置（索引或下标等于1）的元素是2；类似的，a(9)等于512，因为a中第9个位置的元素是512。

- 如果取索引为11，即输入a(11)会出现什么情况？

MATLAB会报错：“索引超出数组元素的数目(10)”，即告诉我们，现在这个向量中元素的数目只有10个，即最大索引是10，而你取了索引11的元素，超出了取值范围。

- 另外，如果我们将ind取成0、负数或者小数，例如输入a(0)、a(-1)、a(1.5)，MATLAB也会报错：“数组索引必须为正整数或逻辑值”。这里出现了“逻辑值”的概念，我们在本章后面小节中会介绍。

2.向量元素的引用

(1) 多个元素引用

- 类似的，我们也可以利用向量的索引来同时提取多个位置的元素，这时候只需要将**ind**设置成一个向量，**ind**中放入我们想要提取的元素的索引，然后使用**a(ind)**命令即可。

- 例如，我们令ind = [1 3 5 7 9]，那么a(ind)的结果为[2 8 32 128 512]，即我们提取了向量a中奇数位置的元素。熟悉向量冒号创建方法的同学应该能够看出，ind等于1:2:9，因此我们可以直接将a(ind)写成a(1:2:9)，这就表示提取a中奇数位置的元素；类似的，提取a中偶数位置元素的命令是a(2:2:10)，如果你不熟练的话，可以分成两步写，即先令ind=2:2:10，然后再使用a(ind)的命令。当然，对于同一个位置的元素，我们也可以提取多次，例如：ind = [1 2 2 3 3 3]，那么a(ind)得到的结果应该是[2 4 4 8 8 8]，以后熟悉的话可以直接写成a([1 2 2 3 3 3])。

二、MATLAB中的向量

3.向量元素的修改和删除

令向量a=[2 4 8 16 32 64 128 256 512 1024]

请依次执行下面的代码	修改后的向量 a
a(1) = 4 % 第一个元素改成 4	[4 4 8 16 32 64 128 256 512 1024]
a([1,3]) = [50 60] % 第 1 个位置元素改成 50；第 3 个位置元素改成 60	[50 4 60 16 32 64 128 256 512 1024]
a(1:3) = [5 6] % 赋值时，左右两侧的元素个数要相同，左边引用了 3 个位置，右侧的向量长度为 2	MATLAB 报错：无法执行赋值，因为左侧和右侧的元素数目不同。
a(2:4) = 100 % 如果右边为常数，则将指定位置的元素全部变成这个常数。	% 第 2 至 4 号位置的元素改为了 100 [50 100 100 100 32 64 128 256 512 1024]
a(13) = 88 % 把索引为 13 的元素赋值为 88，如果超过了最大索引，则会自动拓展向量的大小	[2 4 8 16 32 64 128 256 512 1024 0 0 88] % 索引 11 和 12 的位置会自动用 0 进行赋值

3.向量元素的修改和删除

如果我们将等号右侧变成空向量[ ]，则表示删除对应位置的元素。

a(1) = [ ]　% 删除 a 的第一个元素	[100 100 100 32 64 128 256 512 1024]
a(end-1:end) = [ ]　% 删除 a 中最后两个元素	[100 100 100 32 64 128 256]

三、MATLAB中的矩阵

1.矩阵的创建方法

（1）直接输入法

我们先来看直接输入法，直接输入法适用于矩阵中元素数量较少的情况。

输入矩阵时要以中括号“[ ]”作为标识符号，矩阵的所有元素必须都在中括号内。矩阵的同行元素之间用空格或逗号分隔，行与行之间用分号或回车键分隔。

例如：命令 *a = [1 2 3; 4 5 6];* 可以在工作区创建出变量名为 a 的矩阵 $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$。

三、MATLAB中的矩阵

1.矩阵的创建方法

（2）函数创建法

MATLAB 提供了一些函数，这些函数可以用来生成某些特定的矩阵，我们这里介绍几组最常用到的函数。

第一组函数： zeros、ones 和 eye。这三个函数分别用来创建全为 0 的矩阵、全为 1 的矩阵和单位矩阵。

以 zeros 函数为例，其常见的用法有两种：(1) zeros(n) 可以创建一个 n 行 n 列全为 0 的矩阵；(2) zeros(m,n) 可以创建一个 m 行 n 列全为 0 的矩阵。

例如：

命令	结果
a = zeros(3)	0 0 0 0 0 0 0 0 0
b = zeros(2,3)	0 0 0 0 0 0

ones和eye的用法类似

三、MATLAB中的矩阵

1.矩阵的创建方法

第二组函数： rand、randi 和 randn。这三个函数分别用来创建均匀分布的随机数、均匀分布的随机整数和标准正态分布的随机数，以后会大量用到，请大家熟记。（数据的分布是概率论里面的知识点，没学过的同学可以搜索关键词自学）

rand 函数用来创建区间 0 和 1 内均匀分布的随机数，其最常用的方法有两种：(1) rand(n) 可以创建一个 n 行 n 列的随机数矩阵；(2) rand(m, n) 可以创建一个 m 行 n 列的随机数矩阵。由 rand 函数创建的随机数矩阵的每个元素都随机取样自 0 和 1 之间的均匀分布。

randi 函数用来创建均匀分布的随机整数，其最一般的用法为：randi([imin, imax], m, n)，该命令可创建一个 m 行 n 列的随机数矩阵，随机数矩阵中的每个元素都是从区间[imin, imax]内随机抽取的整数。举个例子，假设我们要模拟投掷 100 次骰子，骰子有 6 个面，那么我们可以使用 randi([1, 6], 1, 100) 得到一个长度为 100 的行向量，向量中的每个元素都是取自 1, 2, 3, 4, 5, 6 中的一个整数。另外，如果 imin 等于 1，那么可以简写为 randi(imax, m, n)；如果 m 和 n 相同，即生成一个 n 行 n 列的方阵，那么可以直接写成 randi([imin, imax], n)。

randn 函数用来创建标准正态分布的随机数，其使用方法和 rand 函数类似：(1) randn(n) 可以创建一个 n 行 n 列的随机数矩阵；(2) randn(m, n) 可以创建一个 m 行 n 列的随机数矩阵。由 randn 函数创建的随机数矩阵的每个元素都随机取样自标准正态分布。

三、MATLAB中的矩阵

1.矩阵的创建方法

第三组函数：diag 和 blkdiag。

diag 函数可用来创建对角矩阵或者获取矩阵的对角元素

情况 1：如果输入的第一个参数是向量，则表示创建对角矩阵。

diag(v, k) 将向量 v 的元素放置在第 k 条对角线上，其他位置元素为 0。

k=0 表示主对角线，k>0 位于主对角线上方，k<0 位于主对角线下方。

如果 k=0，可以直接写成 diag(v)。

命令	结果
diag([1,2,3])	1 0 0 0 2 0 0 0 3
diag([1,2,3], -1)	0 0 0 0 1 0 0 0 0 2 0 0 0 0 3 0
diag([1,2,3], 2)	0 0 1 0 0 0 0 0 2 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0

三、MATLAB中的矩阵

1.矩阵的创建方法

情况 2：如果输入的第一个参数是矩阵，则表示获取矩阵的对角元素。

diag(A,k) 返回 A 的第 k 条对角线上元素的构成的列向量。

命令	结果
A = [1,2,3,4; 5,6,7,8; 9,10,11,12; 13,14,15,16]	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
diag(A)	1 6 11 16
diag(A, -1)	5 10 15
diag(A, 1)	2 7 12

三、MATLAB中的矩阵

1.矩阵的创建方法

blkdiag 函数可用来创建分块对角矩阵。

分块对角矩阵是相对于常规的对角矩阵而言的，常规的对角矩阵沿对角线具有单个元素，而分块对角矩阵的对角线的元素是矩阵。我们可采用以下形式表示一个分块对角矩阵：

$$\begin{bmatrix} A1 & O & \cdots & O \\ O & A2 & \cdots & O \\ \vdots & \vdots & \ddots & \vdots \\ O & O & \cdots & AN \end{bmatrix}$$ (A1,A2,…,AN各自可以是大小不同的矩阵)

命令	结果
A1 = [1,2,3;4,5,6]	1 2 3 4 5 6
A2 = [7,8;9,10]	7 8 9 10
A3 = [11,12;13,14;15,16]	11 12 13 14 15 16
blkdiag(A1,A2,A3)	1 2 3 0 0 0 0 4 5 6 0 0 0 0 0 0 0 7 8 0 0 0 0 0 9 10 0 0 0 0 0 0 0 11 12 0 0 0 0 0 13 14 0 0 0 0 0 15 16

三、MATLAB中的矩阵

1.矩阵的创建方法

（3）导入本地文件中的数据

MATLAB 可读取本地的文件，支持的常见格式如下：

- .txt、.dat 或 .csv（适用于带分隔符的文本文件）
- .xls、.xlsb、.xlsm、.xlsx、.xltm、.xltx 或 .ods（适用于电子表格文件）

三、MATLAB中的矩阵

2.矩阵元素的引用

在讲解矩阵元素的引用之前，我们先来回顾一下矩阵的表示方式：

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}_{m\times n}$$

上方给出了一个m行n列的矩阵，对于第i行第j列的元素，我们用a_{ij}表示。

因此，我们可以使用矩阵元素所处的行(row)和列(column)来进行引用矩阵的某一个元素，方式为：a(row_ind, column_ind).

这里的 row_ind 表示要引用的元素的行索引，column_ind 表示列索引。如果 row_ind 和 column_ind 都是一个常数，则表示提取矩阵中的单个元素；如果 row_ind 或 column_ind 是包含有多个元素的向量，则表示同时提取多个位置的元素。与向量类似，end 也可以用来替代最后一个索引，通常和冒号法一起使用。

三、MATLAB中的矩阵

2.矩阵元素的引用

命令	结果
a = randi([2,10],4,5) % 生成一个随机整数矩阵	7 9 3 6 5 5 3 4 4 2 5 8 6 6 8 7 10 10 7 6
a(1,2) % 第一行第二列的元素	9
a(2,end) % 第二行最后一列的元素	2
a(2,[1 3]) % 第二行、第一三列的元素	5 4
a(3,1:5) % 第三行、一至五列的元素	5 8 6 6 8
a([1 3], [1 3 5]) % 第一三行、第一三五列的元素	7 3 5 5 6 8
a(1:2:end, 1:2:end) % 奇数行且奇数列对应的元素	7 3 5 5 6 8

三、MATLAB中的矩阵

2.矩阵元素的引用

size函数两种常见的用法：

（1）size(A)返回一个行向量，其元素是A的各维度的长度。若A是一个3×4的矩阵，则size(A)返回向量[3 4]；如果让[r,c] = size(A)，那么r= 3,c= 4。

（2）size(A,dim)返回在维度dim上的长度。dim=1时表示行；dim=2时表示列。若A是一个3×4的矩阵，则size(A,1)返回3，size(A,2)返回4。

（length函数和numel函数也可以用在矩阵上。length函数会返回行和列的较大值：对上面的A矩阵，length(A)返回4；numel函数会返回矩阵中元素的总数，numel(A) 返回12）

2.矩阵元素的引用

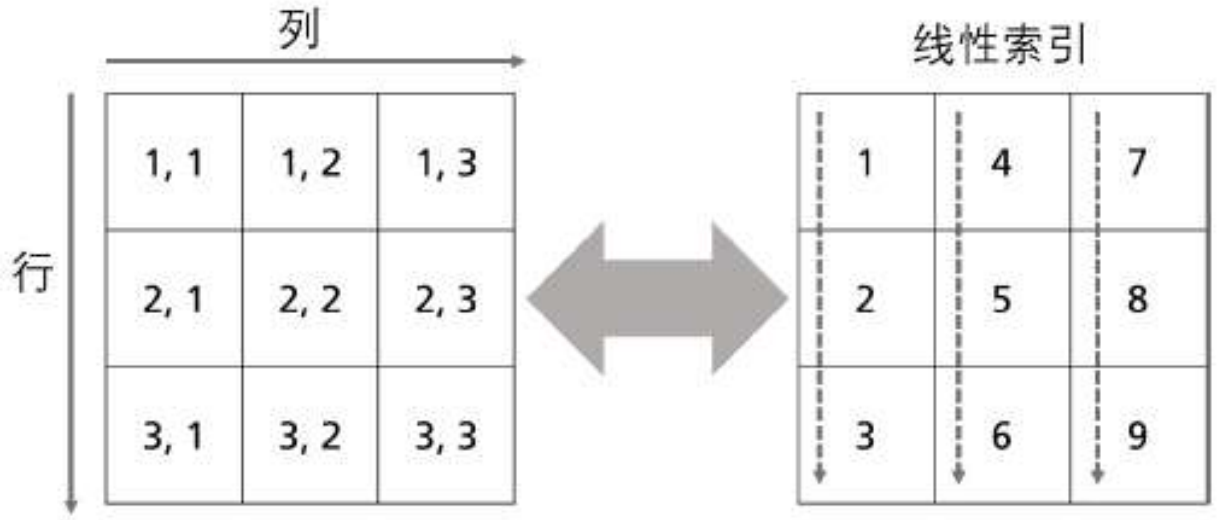
有时候我们需要取出矩阵的某一行或者某一列。以取出矩阵 A 的第一行为例，我们可以使用代码 A(1, 1:end)，即 row_ind 取 1 表示第一行，column_ind 取 1:end 表示从 1 到最后一列的索引。这时候我们可以直接将其简写为：A(1, :)，逗号后面是列索引的位置，加一个冒号就表示取出每一列的元素。同理，要取第一列的所有元素，我们可以使用代码：A(:, 1).

总结：

- A(:, n) 表示矩阵 A 的第 n 列的所有元素。
- A(m, :) 表示矩阵 A 的第 m 行的所有元素。

2.矩阵元素的引用

线性索引：



三、MATLAB中的矩阵

3.矩阵元素的修改和删除

依次运行下面的命令	结果
A = [1:4; 2:5; 3:6]	1 2 3 4 2 3 4 5 3 4 5 6
A(2,3) = 10 % 第二行第三列的元素修改成 10	1 2 3 4 2 3 10 5 3 4 5 6
A(3,:) = 100 % 如果右边为常数，则将指定位置的元素全部变成这个常数	1 2 3 4 2 3 10 5 100 100 100 100
% 左右两侧的大小要匹配 A([1,3],[2,3]) = [8 88; 888 8888]	1 8 88 4 2 3 10 5 100 888 8888 100

三、MATLAB中的矩阵

3.矩阵元素的修改和删除

依次运行下面的命令	结果
A = [1:4;2:5;3:6]	1 2 3 4 2 3 4 5 3 4 5 6
A(4) = 10 % 将线性索引为4的元素修改成10	1 10 3 4 2 3 4 5 3 4 5 6
A(1:2:end) = 0 % 如果右边为常数，则将指定位置的元素全部变成这个常数。	0 10 0 4 2 0 4 0 0 4 0 6
% 左右两侧的元素数量要匹配 A([3,5,6]) = [8 88 888]	0 10 0 4 2 88 4 0 8 888 0 6

3.矩阵元素的修改和删除

注意，如果你在赋值时将一个或多个元素置于矩阵现有的行和列索引的边界之外，则会将矩阵的大小进行拓展，MATLAB 会将没有赋值的位置的元素自动用 0 填充，使其保持为完整的矩形。

例如，A 是一个 2 行 3 列的矩阵，在 A 的第三行第四列的位置插入一个元素 88，矩阵 A 会自动进行拓展。

依次运行下面的命令	结果
A = [10 20 30; 60 70 80]	10 20 30 60 70 80
A(3,4) = 88	10 20 30 0 60 70 80 0 0 0 0 88

3.矩阵元素的修改和删除

此外，我们还可以通过在现有索引范围之外插入一个新的矩阵来扩展原始矩阵的大小。

```
% 接上面的代码，在 A 的四五行、五六列加上一个新的矩阵
A(4:5, 5:6) = [2 3; 4 5]
```

```
10    20    30     0     0     0
60    70    80     0     0     0
 0     0     0    88     0     0
 0     0     0     0     2     3
 0     0     0     0     4     5
```

以上就是修改矩阵元素的方法，下面我们再来介绍删除矩阵元素的方法。

如果我们将等号右侧变成空向量[ ]，则可以删除对应位置的元素。需要注意的是，通常只能删除矩阵的整行或者整列，否则会报错。

三、MATLAB中的矩阵

3.矩阵元素的修改和删除

命令	结果
A = [1:4; 2:5; 3:6]	1 2 3 4 2 3 4 5 3 4 5 6
A(:,1) = [ ] % 删除第一列	2 3 4 3 4 5 4 5 6
A = [1:4; 2:5; 3:6] A(:, [2,end]) = [ ] % 删除第二列和最后一列	1 3 2 4 3 5
A = [1:4; 2:5; 3:6] A(2,2) = [ ]	MATLAB 报错：空赋值只能具有一个非冒号索引。
A = [1:4; 2:5; 3:6] A([1,3],[1,2]) = [ ]	MATLAB 报错：空赋值只能具有一个非冒号索引。

4.矩阵的拼接和重复

有时候我们需要对多个矩阵进行拼接，变成一个大的矩阵。根据矩阵拼接的方向，我们可以分为横向(水平)拼接和纵向(垂直)拼接，如下图所示：

横向拼接(horizontal catenate)　　纵向拼接(vertical catenate)

如上图所示：横向拼接要求矩阵的行数相同；纵向拼接要求矩阵的列数相同。

4.矩阵的拼接和重复

在 MATLAB 中，我们可以使用命令[A, B] 或 [A B]对矩阵 A 和 B 进行横向拼接，也可以使用 MATLAB 中的内置函数：horzcat(A,B)；类似的，我们可以使用命令[A; B]对矩阵 A 和 B 进行纵向拼接，也可以使用 MATLAB 中的内置函数：vertcat(A,B)。

事实上，horzcat 和 vertcat 两个函数来源自 cat 函数，这里的 cat 不是猫的意思，而是单词 catenate 的缩写，可以翻译成连接。

cat 函数的用法如下：

命令 cat(dim,A,B)表示沿着维度 dim 方向将矩阵 B 拼接到矩阵 A 的末尾。

dim = 1 时表示沿着行方向从上往下进行拼接，即纵向拼接，因此 cat(1,A,B)等价于 vertcat(A,B)；

dim = 2 时表示沿着列方向从左自右进行拼接，即横向拼接，因此 cat(2,A,B)等价于 horzcat(A,B)。

（horzcat 函数中的 horz 取自英文单词 horizontal，表示水平的意思；vertcat 函数中的 vert 取自英文单词 vertical，表示竖直的意思）

三、MATLAB中的矩阵

4.矩阵的拼接和重复

（1） 横向拼接的例子

命令	结果
A = [1 6 7; 4 5 7] B = [3 1; 5 10]	A = 1 6 7 4 5 7 B = 3 1 5 10
[A, B] [A B] % 用空格隔开，可以有多个空格 horzcat(A,B) cat(2,A,B)	% 左侧四种方法都可以用于横向拼接 1 6 7 3 1 4 5 7 5 10

4.矩阵的拼接和重复

（2） 纵向拼接的例子

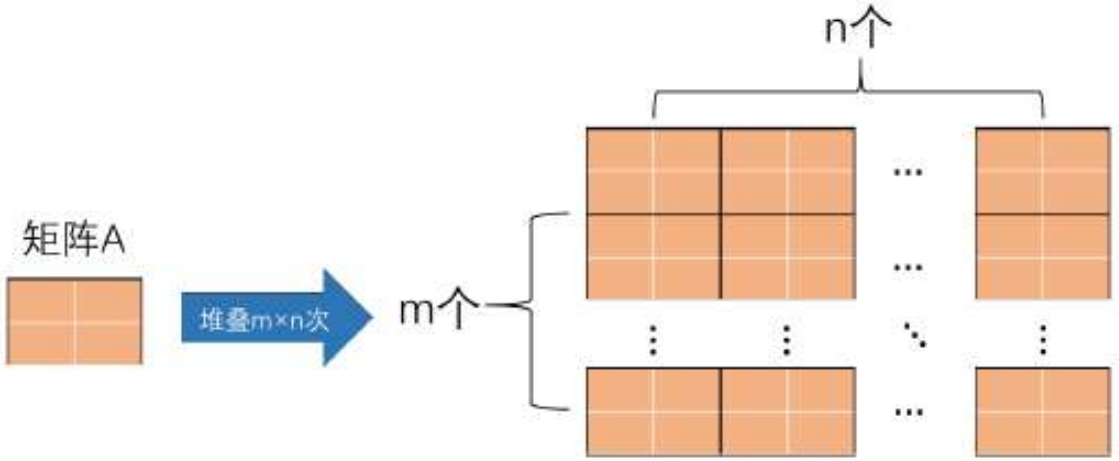
命令	结果
A = [2 4 5;2 2 4] B = [1 8 6;6 3 10;1 5 5]	A = 2 4 5 2 2 4 B = 1 8 6 6 3 10 1 5 5
[A; B] [A B] % 也可以使用回车键进行纵向拼接 vertcat(A,B) cat(1,A,B)	% 左侧四种方法都可以用于纵向拼接 2 4 5 2 2 4 1 8 6 6 3 10 1 5 5

三、MATLAB中的矩阵

4.矩阵的拼接和重复

命令	结果
A = ones(2) B = zeros(2) C = eye(2)	A = 1 1 1 1 B = 0 0 0 0 C = 1 0 0 1
[A B C] [A, B, C] cat(2,A,B,C) horzcat(A,B,C)	% 多个矩阵进行横向拼接 1 1 0 0 1 0 1 1 0 0 0 1
[A; B; C] [A B C] cat(1,A,B,C) vertcat(A,B,C)	% 多个矩阵进行纵向拼接 1 1 1 1 0 0 0 0 1 0 0 1

4.矩阵的拼接和重复



在 MATLAB 中，对同一个矩阵进行重复的堆叠的代码为 repmat(A,m,n)。

（如何记住 repmat 这个函数？ repeat 表示重复，matrix 表示矩阵）

4.矩阵的拼接和重复

命令	结果
A = 1:4; B = repmat(A,3,1)	1 2 3 4 1 2 3 4 1 2 3 4
C = [1,2; 3,4]; D = repmat(C,2,3)	1 2 1 2 1 2 3 4 3 4 3 4 1 2 1 2 1 2 3 4 3 4 3 4

4.矩阵的拼接和重复

除了对整个矩阵进行重复的堆叠外，MATLAB 还可以对向量或者矩阵中的元素进行重复，使用到的函数是 repelem。（如何记住 repelem: repeat 重复 + element 元素）

repelem 函数有两种用法：

（1） **重复向量 v 中的元素：repelem(v, n)**

当 n 为一个正整数时，表示把向量 v 中的每一个元素都重复 n 次；n 也可以为一个向量，其长度必须和 v 的长度相同，它可以将 v 中第 i 个位置的元素 v(i)重复 n(i)次，其中 n(i)表示 n 中第 i 个位置的元素。

命令	结果
v = [5,3,8]; repelem(v, 2) % 将 v 中的每个元素都重复 2 次	5 5 3 3 8 8
repelem(v, [2,1,4]) % 将 v 中的第一个元素重复 2 次，第二个元素重复 1 次，第三个元素重复 4 次	5 5 3 8 8 8 8

4.矩阵的拼接和重复

（2） 重复矩阵 A 中的元素：**repelem(A,m,n)**

m 和 n 分别表示沿着行方向(从上至下)以及沿着列方向(从左至右)将矩阵元素重复的次数，这里的 m 和 n 可以是正整数，也可以是向量。如果 m 是向量，则 m 的长度要和矩阵 A 的行数相同；如果 n 是向量，则 n 的长度要和矩阵 A 的列数相同。

命令	结果
A = [2,3,5; 8,4,7]; repelem(A,3,2) % 沿着行方向将 A 的元素重复 3 次，沿着列方向重复 2 次	2 2 3 3 5 5 2 2 3 3 5 5 2 2 3 3 5 5 8 8 4 4 7 7 8 8 4 4 7 7 8 8 4 4 7 7
repelem(A,2,[1,2,3]) % 沿着行方向：每一行都重复 2 次； 沿着列方向：第一列重复 1 次，第二列重复 2 次，第三列重复 3 次	2 3 3 5 5 5 2 3 3 5 5 5 8 4 4 7 7 7 8 4 4 7 7 7
repelem(A,[2,3],[1,2,3]) % 沿着行方向：第一行重复 2 次，第二行重复 3 次； 沿着列方向：第一列重复 1 次，第二列重复 2 次，第三列重复 3 次	2 3 3 5 5 5 2 3 3 5 5 5 8 4 4 7 7 7 8 4 4 7 7 7 8 4 4 7 7 7

5.矩阵的重构和重新排列

函数名	主要作用
reshape	更改矩阵的形状
sort	对向量或者矩阵进行排序
sortrows	基于矩阵的某一列对矩阵进行排序，同一行的元素不会改变
flip / fliplr / flipud	将矩阵进行翻转，fliplr 是左右翻转，flipud 是上下翻转
rot90	将矩阵按逆时针方向旋转 90 度或者 90 度的倍数

5.矩阵的重构和重新排列

reshape 函数可以改变矩阵的形状，其常用语法为 reshape(A, m, n)或者 reshape(A,[m,n])，这可以将矩阵 A 的形状更改为 m 行 n 列，前提是转换前后的两个矩阵的元素总数要相同。

例如有一个矩阵 A，它原来的形状是 2 行 6 列，如果我们需要将其形状变成 3 行 4 列，就可以使用命令：reshape(A, 3, 4).

命令	结果
A = randi(10,2,6)　% 生成随机整数矩阵	2 2 10 3 5 1 4 1 4 4 7 9
B = reshape(A,3,4)　% 改变形状	2 1 3 7 4 10 4 1 2 4 5 9

从上面的运行结果可以看出， reshape 函数实际上是按矩阵的线性索引来重新组织矩阵元素的。也就是说，它先取矩阵 A 的第一列，然后是第二列，依此类推，再按新的维度重新组织这些元素。因此，转换后的 B 矩阵中的元素和 A 矩阵中的元素是完全相同的，即 A(:)和 B(:)的结果完全相同。

5.矩阵的重构和重新排列

另外，我们不需要自己来计算转换后的矩阵有多少行或多少列。可以只给出转换后的行数，列数用空向量[ ]代替；或者只给出转换后的列数，行数用空向量[ ]代替。MATLAB 会自动帮我们计算转换后的矩阵大小。例如：若 A 是一个由 12 个元素组成的矩阵，命令 reshape(A,3,[ ])、reshape(A,[ ],4)可以实现和 reshape(A,3,4)一样的效果。

如果你给出的转换后的行数和列数的乘积不等于原始矩阵中元素的个数，那么 MATLAB 就会报错：

命令	结果
A = randi(10,3,6); B = reshape(A,5,8)	错误使用 reshape。元素数不能更改。请使用 [] 作为大小输入之一，以自动计算该维度的适当大小。
A = randi(10,3,6); B = reshape(A,[ ],8)	错误使用 reshape 已知维度的乘积 8 不能被元素总数 18 整除。

5.矩阵的重构和重新排列

（2）sort 函数

sort 函数是用于对向量或矩阵进行排序的。如果输入的参数是矩阵的话，还可以对矩阵的每一行或每一列分别进行排序。

①对向量排序

我们先来学习 sort 函数对向量排序，假设 v 是一个向量，有下面两种基础的用法：

- sort(v) 可以将向量 v 按照从小到大的顺序进行升序排列；
- sort(v, 'descend')可以将向量 v 按照从大到小的顺序进行降序排列。

命令	结果
v = [10 24 16 8 50 40]; v1 = sort(v)	8 10 16 24 40 50
v2 = sort(v,'descend')	50 40 24 16 10 8
% 换个列向量测试 sort 函数 v = [5;-5;7;0;4;5]; v1 = sort(v)	-5 0 4 5 5 7

5.矩阵的重构和重新排列

注意，上面的用法中，sort 函数只有一个返回值，即排序后的向量；事实上，sort 函数可以有两个返回值，基本用法为：`[sort_v, ind] = sort(v)`。这里，sort_v 是排序后的向量，ind 是排序后的向量（即 sort_v）中的每个元素在原向量（即 v）中的索引（即下标、位置）。我们来看一个具体的例子：

v = [10 24 16 8 50 40]; [sort_v , ind] = sort(v)	sort_v = [8 10 16 24 40 50] ind = [4 1 3 2 6 5]

在上面的例子中，我们让 sort 函数返回了两个变量：sort_v 和 ind。它们是两个长度相等的向量，向量的方向和 sort 函数中输入的 v 向量的方向一致，都是行向量。向量 v 中所有元素的最小值为 8，而 8 在 v 中的索引是 4，因此 sort_v 中第一个元素为 8，ind 的第一个元素为 4；向量 v 中第二小的值为 10，而 10 是 v 中的第 1 个元素，因此 sort_v 中第二个元素为 10，ind 的第二个元素为 1；依次类推，可以得到 sort_v 和 ind 向量的值。事实上，这里有一个恒等关系成立：v(ind)运行的结果和 sort_v 的结果完全一样，大家可以自行验证。

5.矩阵的重构和重新排列

② 对矩阵排序

上面介绍的是 sort 函数对向量进行排序的应用，下面我们再来介绍 sort 函数对矩阵 A 进行排序的用法：**sort(A, dim)**.

- dim = 1 时，沿着行方向(从上至下)对矩阵的每一列升序排列
- dim = 2 时，沿着列方向(从左至右)对矩阵的每一行升序排列

注意：（1）当 dim=1 时，sort(A,1)可以直接写成 sort(A)；（2）默认是升序排列的，我们可以在最后面加一个输入参数'descend'，变成从大到小的降序排列；（3）可以有两个返回值，代表的含义和对向量排序类似，表示排序后的元素在原矩阵所在行或所在列中的索引。

三、MATLAB中的矩阵

5.矩阵的重构和重新排列

命令	结果
A = randi(10,4,6)	4 1 8 7 8 2 3 7 8 6 4 2 8 7 8 4 7 6 1 6 3 1 8 5
% 对矩阵 A 的每一列升序排列 sort(A) sort(A,1)	1 1 3 1 4 2 3 6 8 4 7 2 4 7 8 6 8 5 8 7 8 7 8 6 % 每一列的元素都是按照升序进行排列
% 对矩阵 A 的每一行升序排列 sort(A, 2)	1 2 4 7 8 8 2 3 4 6 7 8 4 6 7 7 8 8 1 1 3 5 6 8 % 每一行的元素都是按照升序进行排列
% 对矩阵 A 的每一行降序排列 sort(A, 2, 'descend')	8 8 7 4 2 1 8 7 6 4 3 2 8 8 7 7 6 4 8 6 5 3 1 1 % 每一行的元素都是按照降序进行排列

5.矩阵的重构和重新排列

```
% 返回两个值
A = [4 1 8 7 8 2;
     3 7 8 6 4 2;
     8 7 8 4 7 6;
     1 6 3 1 8 5];
[sort_A, ind] = sort(A)
```

```
sort_A =
     1     1     3     1     4     2
     3     6     8     4     7     2
     4     7     8     6     8     5
     8     7     8     7     8     6
```

```
ind =
     4     1     4     4     2     1
     2     4     1     3     3     2
     1     2     2     2     1     4
     3     3     3     1     4     3
```

这里是对 A 中每一列进行升序排列。
我们先来解释 ind 的第一列四个元素的含义，第一列的四个元素分别是 4 2 1 3；其中第一个元素 4 表示排序后的 sort_A 中第一列的第一个元素 1 在原始矩阵 A 中第一列的第四个位置；-第二个元素 2 表示排序后的 sort_A 中第一列的第二个元素 3 在原始矩阵 A 中第一列的第二个位置；第三个元素 1 表示排序后的 sort_A 中第一列的第三个元素 4 在原始矩阵 A 中第一列的第一个位置；第四个元素 3 表示排序后的 sort_A 中第一列的第四个元素 8 在原始矩阵 A 中第一列的第三个位置。依次类推，可以解释其他各列 ind 的值的含义。

5.矩阵的重构和重新排列

（3）sortrows 函数

sortrows 函数可以基于矩阵的某一列对矩阵进行排序，排序后得到的新矩阵的同一行元素不会改变。这个函数的用法较多，下面我们直接用一个具体的实例来讲解它的主要用法。

假设有6名学生，下面这个矩阵保存着这六名同学在四门科目上的成绩。矩阵的每一行代表一名学生。这六名同学的四门科目的成绩对应着四列，例如第一名同学的第一科成绩为95，第二科成绩为80，依此类推。

$$\text{score} = \begin{bmatrix} 95 & 80 & 85 & 79 \\ 95 & 67 & 78 & 90 \\ 95 & 67 & 78 & 75 \\ 95 & 67 & 64 & 73 \\ 86 & 85 & 82 & 84 \\ 86 & 87 & 84 & 88 \end{bmatrix}$$

5.矩阵的重构和重新排列

（1） 请基于第一科的成绩按升序对这六名同学进行排序，得到排序后的成绩矩阵。若第一科成绩相同，则基于第二科成绩升序排列。如果第二科成绩还相同，就基于第三科成绩进行排序，依此类推。

```
score = [95 80 85 79;95 67 78 90;95 67 78 75;95 67 64 73;86 85 82 84;86 87 84 88];
sort_score1 = sortrows(score)
```

```
86  85  82  84
86  87  84  88
95  67  64  73
95  67  78  75
95  67  78  90
95  80  85  79
```

（2） 请基于第一科的成绩按升序对这六名同学进行排序。当第一科成绩相同时，请保持其在矩阵中出现的先后顺序。

```
sort_score2 = sortrows(score,1)
```

```
86  85  82  84
86  87  84  88
95  80  85  79
95  67  78  90
95  67  78  75
95  67  64  73
```

5.矩阵的重构和重新排列

（3） 请基于第二科的成绩按升序对这六名同学进行排序。当第二科成绩相同时，请保持其在矩阵中出现的先后顺序。

sort_score3 = sortrows(score,2)	95 67 78 90 95 67 78 75 95 67 64 73 95 80 85 79 86 85 82 84 86 87 84 88

（4） 请基于第一科的成绩按升序对这六名同学进行排序。当第一科成绩相同时，基于第三科成绩升序排列。如果第一科和第三科都相同，就保持在矩阵中出现的先后顺序。

sort_score4 = sortrows(score,[1,3])	86 85 82 84 86 87 84 88 95 67 64 73 95 67 78 90 95 67 78 75 95 80 85 79

事实上，sortrows(score)等价于 sortrows(score, 1:size(score,2))，即 sortrows(score, [1,2,3,4]).

5.矩阵的重构和重新排列

（5） 请基于第一科的成绩对这六名同学进行降序排列。当第一科成绩相同时，基于第三科成绩降序排列。如果第一科和第三科都相同，就保持在矩阵中出现的先后顺序。

```
sort_score5 =
  sortrows(score,[1,3],'descend')
```

```
    95    80    85    79
    95    67    78    90
    95    67    78    75
    95    67    64    73
    86    87    84    88
    86    85    82    84
```

（6） 请基于第一科的成绩对这六名同学进行降序排列。当第一科成绩相同时，基于第三科成绩升序排列。如果第一科和第三科都相同，就保持在矩阵中出现的先后顺序。

```
sort_score6 =
sortrows(score,[1,3],{'descend','ascend'})
% 使用元胞数组分别表示多个列的方向
```

```
    95    67    64    73
    95    67    78    90
    95    67    78    75
    95    80    85    79
    86    85    82    84
    86    87    84    88
```

5.矩阵的重构和重新排列

（7） 请基于第一科的成绩按升序对这六名同学进行排序。当第一科成绩相同时，请保持其在矩阵中出现的先后顺序，并返回索引值。

[sort_score7 , ind7] = sortrows(score,1)	sort_score7 = 86 85 82 84 86 87 84 88 95 80 85 79 95 67 78 90 95 67 78 75 95 67 64 73 ind7 = 5 6 1 2 3 4 ind7 的第一个元素为 5，这表示 sort_score7 的第一行元素在原矩阵中位于第 5 行；ind7 的第二个元素为 6，这表示 sort_score7 的第二行元素在原矩阵中位于第 6 行；依次类推，可以解释 ind7 每一个元素的含义。 因此 sort_score7 和 score(ind7, :)得到的结果完全相同。

5.矩阵的重构和重新排列

通过上面的例子可以看出，sortrows 函数和 sort 函数的区别在于：sort 函数会对矩阵的每一列分别进行排序；而 sortrows 函数是基于某一列进行排序的，排序后得到的新矩阵的同一行元素不会改变。

命令	结果
A = randi(10,4,6)	4 1 8 7 8 2 3 7 8 6 4 2 8 7 8 4 7 6 1 6 3 1 8 5
sort(A)	sortrows(A) % sortrows(A,1:6)
1 1 3 1 4 2 3 6 8 4 7 2 4 7 8 6 8 5 8 7 8 7 8 6	1 6 3 1 8 5 3 7 8 6 4 2 4 1 8 7 8 2 8 7 8 4 7 6

5.矩阵的重构和重新排列

（4）flip / fliplr / flipud 函数

下面我们来学习 flip / fliplr / flipud 这三个函数，它们可以用来对向量或矩阵进行翻转操作。其中，flip 函数是一个通用的翻转函数，而 fliplr 和 flipud 是其特例，分别用于从左到右和从上到下的翻转。flip 翻译成中文是翻转，而 fliplr 函数可以拆解为 filp+ 左边 left + 右边 right，flipud 则可以拆解为 flip + 上边 upper + 下边 down，大家可以根据英文来进行记忆。

5.矩阵的重构和重新排列

flip 函数有两种用法：

用法 1： **flip(A)**

- 如果 A 为向量，flip(A) 将翻转向量中各元素的顺序，向量的方向不变。
- 如果 A 为矩阵，flip(A) 将对矩阵进行上下翻转。

命令	结果
A = [5 2 7 8 9]; % 行向量 flip(A) % 等价于 A(end:-1:1)	9 8 7 2 5
A = [5;2;7;8;9]; % 列向量 flip(A)	9 8 7 2 5
A = [5 8 7; 4 2 6; 3 5 8; 6 4 1]; flip(A) % 对矩阵进行上下翻转	6 4 1 3 5 8 4 2 6 5 8 7

5.矩阵的重构和重新排列

用法 2： **flip(A, dim)**

flip(A,dim) 沿维度 dim 翻转 A 中元素的顺序。

- dim 为 1 时表示行，此时 flip(A,1) 将沿着行方向对矩阵 A 上下翻转。
- dim 为 2 时表示列，此时 flip(A,2) 将沿着列方向对矩阵 A 左右翻转。

命令	结果
A = [5 8 7; 4 2 6; 3 5 8; 6 4 1];	5 8 7 4 2 6 3 5 8 6 4 1
flip(A,1) % 对矩阵进行上下翻转 % 等价于 flip(A)或者 flipud(A)	6 4 1 3 5 8 4 2 6 5 8 7
flip(A,2) % 对矩阵进行左右翻转 % 等价于 fliplr(A)	7 8 5 6 2 4 8 5 3 1 4 6

5.矩阵的重构和重新排列

（5）rot90 函数

rot90 函数是对矩阵进行旋转的函数，它源于英文 rotate 一词，中文翻译为旋转。rot90 函数允许我们按 90 度或其倍数**逆时针**旋转矩阵。它的用法非常简单，rot90(A,k)将矩阵 A 按逆时针方向旋转 k*90 度，其中 k 是一个整数；不提供 k 时 k 默认取 1。我们来看几个例子：

命令	结果
C = [5 8 7; 4 2 6; 3 5 8; 6 4 1];	5 8 7 4 2 6 3 5 8 6 4 1
rot90(C) % 等价于 rot90(C,1)	7 6 8 1 8 2 5 4 5 4 3 6
rot90(C, 2) % 180°	1 4 6 8 5 3 6 2 4 7 8 5
rot90(C, 3) % 270° 等价于顺时针旋转 90°	6 3 4 5 4 5 2 8 1 8 6 7