

# QBUS6830 Lab1

## 读取数据

MATLAB 读取数据既可以通过 导入数据助手 也可通过代码解决。但本身 导入数据助手 也是通过调用代码处理数据的。

### 导入数据助手

1. 按菜单当中的 导入数据，选择需要导入的文件。
2. 进入导入界面后，修改导入文件的变量名，类型和数据范围

修改导入数据的类型

双击修改存储导入数据的变量名

修改导入数据范围

- 注意 导入数据的类型不同可能会造成数据的丢失，尤其是数值矩阵中的日期和时间数据

如果转化成数值矩阵，第一列的日期数据由于不能转化，最终被默认改为 NAN

- 如果想导入日期和时间数据，应该先导入成 table 然后再用 `table2array` 转化成 datetime

array, 或者用 table 的[点索引](#)获得 datetime array

## 代码导入数据

MATLAB 还可以用函数来读取各类数据文件, 但是不是很好用

TODO

## Array Matrix Cell

在 MATLAB 运算的所有数据类型, 都是按照 array 或 matrix 的形式进行存储和计算的。

### Array

#### 构建 Row vector 和 Column vector

用中括号将被空格或者逗号分割的数列赋值给 Row vector

```
1  %构建 行向量
2  >> a = [1 2 3 4] % 或者 [1,2,3,4]
3
4  a =
5
6      1      2      3      4
7
8  %构建 列向量
9  >> a = [1 ;2; 3; 4]
10 a =
11
12      1
13      2
14      3
15      4
```

### 引用向量元素

Array 有和 python 类似的索引引用

```
1  >> a(3)
2
3  ans =
4
5      3
```

- 注意这里的索引用的是 小括号, 而不是 python 里的中括号

同样用来代表全部的 冒号

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

以及引用一个范围的数据

```
1  >> a(2:4)
2
3  ans =
4
5      2
6      3
7      4
```

`end` 检索最后一位元素

```
1  >> a(end)
2
3  ans =
4
5      4
```

## 向量加减法

在 MATLAB 中当进行两个向量的加法与减法的时候，这两个向量的元素必须有相同的类型和数量。

```
1  >> b = [5;6;7;8]
2
3  b =
4
5      5
6      6
7      7
8      8
9
10 >> a + b
11
12 ans =
13
14      6
15      8
```

```

16      10
17      12
18  >> b = [5,6,7,8]
19
20  b =
21
22      5      6      7      8
23
24  >> a + b
25
26  ans =
27
28      6      7      8      9
29      7      8      9     10
30      8      9     10     11
31      9     10     11     12

```

## 标量向量乘法/加法

让一个数字乘以/加一个向量。标量乘法/加法会产生相同类型的新的一个向量，原先的向量的每个元素乘以/加以数量

```

1  >> m = 5 * a
2
3  m =
4
5      5
6      10
7      15
8      20
9
10 >> 5+m
11
12 ans =
13
14      10
15      15
16      20
17      25

```

## 转置

向量和矩阵 的转置 都是用的 `'` 运算符

```

1  m'
2  ans =
3  5    10    15    20

```

## 添加向量

将原向量  $m$  和要添加的向量  $n$  放在同一行:  $[m \ n]$  或者  $[m, n]$

将原向量  $m$  和要添加的向量  $n$  组合成新矩阵:  $[m;n]$ ,  $m$  和  $n$  的长度应该一样

```
1 >> m'
2
3 ans =
4
5      5      10      15      20
6
7 >> n = [3,6,9,12]
8
9 n =
10
11      3      6      9      12
12
13 >> [m' n]
14
15 ans =
16
17      5      10      15      20      3      6      9      12
```

## 点积 dot(a, b)

```
1 >> v1 = [2 3 4];
2 v2 = [1 2 3];
3 dp = dot(v1, v2);
4 >> dp
5
6 dp =
7
8      20
```

## 生成等差元素向量

`a = [s:l:f]`

生成第一个元素是  $s$ , 最后一个元素是  $f$ , 公差是  $l$  的向量

```
1 >> a = [1:2:19]
2
3 a =
4
5      1      3      5      7      9     11     13     15     17     19
```

## Matrix

TODO

# Cell

TODO

# 画图

## Plot 二维线形图

```
plot(X,Y,LineStyle)
```

[Plot](#) 函数生成二维线图

`x` 和 `y` 长度必须相同, `LineStyle` 字符串设置线的形状、标记和颜色

Color	Code
White	<b>w</b>
Black	<b>k</b>
Blue	<b>b</b>
Red	<b>r</b>
Cyan	<b>c</b>
Green	<b>g</b>
Magenta	<b>m</b>
Yellow	<b>y</b>

```
hand on
```

在同一张图上画线, 而不是生成新的图像

```
hand off
```

停止在同一张图上画线, 下一次画图将生成新的图像

## 随机数函数

# Rand

```
rand(s1,s2,...)
```

[rand](#)生成尺寸为(`s1 * s2 * ...`)的满足 $U(0,1)$ 均匀分布的随机数矩阵

```

1 > rand(5)
2
3 ans =
4
5     0.9058     0.2785     0.9706     0.4218     0.0357
6     0.1270     0.5469     0.9572     0.9157     0.8491
7     0.9134     0.9575     0.4854     0.7922     0.9340
8     0.6324     0.9649     0.8003     0.9595     0.6787
9     0.0975     0.1576     0.1419     0.6557     0.7577

```

```

1 rand(3,2)
2
3 ans =
4
5     0.7431     0.1712
6     0.3922     0.7060
7     0.6555     0.0318

```

## Randn

```
randn(s1,s2,...)
```

[rand](#)生成尺寸为( $s1 * s2 * \dots$ )的满足 $N(0,1)$ 正态分布的随机数矩阵

## normrnd

```
normrnd(mu,sigma,sz1,...,szN)
```

[normrnd](#)产生尺寸为( $s1 * s2 * \dots$ )的满足 $N(\mu, \sigma)$ 正态分布的随机数矩阵

## trnd

```
r = trnd(nu,[m,n,...])
```

[trnd](#)生成尺寸为( $s1 * s2 * \dots$ )的满足 $\nu$ 自由度的t分布的随机数矩阵

## Task 用到的几个函数

### .^2 和 ^2

`.^n` 按元素求幂

`^n` 矩阵幂

其他[运算符和特殊字符](#)

```

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

```

```

3  b =
4
5      1      2      3
6      4      5      6
7      7      8      9
8
9  >> b.^2
10
11 ans =
12
13      1      4      9
14     16     25     36
15     49     64     81
16
17 >> b^2
18
19 ans =
20
21     30     36     42
22     66     81     96
23    102    126    150

```

## diff

```

1  Y = diff(X)
2  Y = diff(X,n)
3  Y = diff(X,n,dim)

```

`Y = diff(X)` 计算沿大小不等于 1 的第一个数组维度的 `x` 相邻元素之间的差分：

`Y = diff(X,n)` 通过递归应用 `diff(X)` 运算符 `n` 次来计算第 `n` 个差分。

`Y = diff(X,n,dim)` 是沿 `dim` 指定的维计算的第 `n` 个差分。`dim` 输入是一个正整数标量。

以一个二维  $p \times m$  输入数组 `A` 为例：

- `diff(A,1,1)` 会对 `A` 的列中的连续元素进行处理，然后返回  $(p-1) \times m$  的差分矩阵。
- `diff(A,1,2)` 会对 `A` 的行中的连续元素进行处理，然后返回  $p \times (m-1)$  的差分矩阵。

## prctile

```
Y = prctile(X,p)
```

返回数组 `X` 中元素的百分位数。

## Lab 提到但目前没用到的函数



command	function
mean(x)	平均数
var(x)	方差
<a href="#">qqplot(x)</a>	分位数 - 分位数图
<a href="#">autocorr(x)</a>	绘制具有置信界限的单变量随机时间序列y的样本自相关函数（ACF）
<a href="#">parcorr(x)</a>	绘制具有置信界限的单变量随机时间序列y的样本部分自相关函数（PACF）。
<a href="#">price2ret(x)</a>	
length(x)	
log(x)	
log10(x)	
diff(x)	
<a href="#">normcdf</a>	正态分布函数
<a href="#">norminv</a>	正态逆分布函数
<a href="#">Tcdf</a>	t分布函数
<a href="#">Tinv</a>	t逆分布函数

## 一些有用技巧

1. 注释掉一段程序：%{、%}

2. help 命令名

调用官方的帮助文件

doc 命令名

调用官方的命令文档，比 help 更详细而且有例子

3. clc 清屏

清除命令窗口中的所有输入和输出信息，不影响命令的历史记录。

clear 和clear all

clear 变量名：可以清除workspace中的无用的变量，尤其是一些特别大的矩阵，不用时及时清理，可以减少内存占用。

clear all 清除所有的变量，使workspace一无所有，当重新开始一次算法验证时，最好执行一次，让workspace中的变量一目了然。。

4. Tab补全

5. 上箭头寻找以前的命令

