MATLAB基础操作1

冯银波

上海财经大学信息管理与工程学院

MATLAB起源

- MATLAB一词是 Matrix Laboratory 的缩写，可见其与矩阵计算的关系非常密切。
  - 最初，Cleve Moler博士为了方便讲授线性代数 和矩阵分析等课程，自己编写了一些程序库，供学生学习使用
- 随着这些程序代码被应用的越来越广泛，Cleve Moler及其合作伙伴推出了第一款商业软件MATLAB
- 之后，MATLAB得到了市场的广泛认可，尤其在科学与工程计算领域。MATLAB变得越来越成熟，其功能越来越强大。
- 大多数理工科专业的本科生都要求掌握MATLAB

MATLAB的主要功能

- 数值计算
- 符号计算
- 绘图
- 齐全的工具箱
  - ✓ 系统仿真
  - ✓ 科学与工程计算
  - ✓ 数据分析

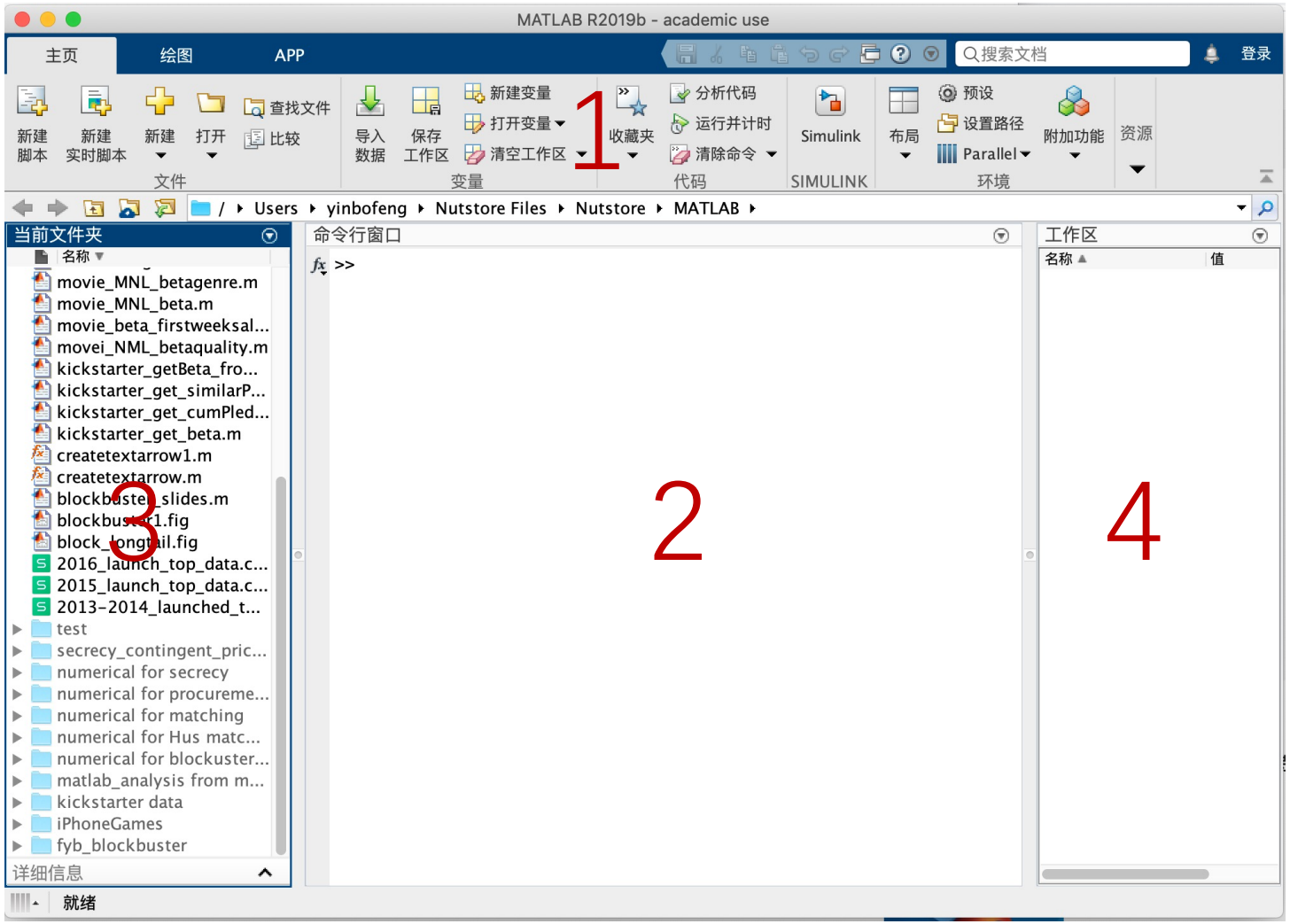
MATLAB的强大优势

MATLAB为什么受欢迎

- 简单易学（相对于C语言和Python）
- 计算功能强大，效率高
  - ✓ 基于几代数学家们在方法论上的贡献
- 不需要繁琐的底层编程
  - ✓ 工具箱齐全
- 可以与许多其它软件对接
  - ✓ 用MATLAB编写代码，运行时调用其它程序，如EXCEL,PYTHON等，让MATLAB变得如虎添翼

MATLAB操作界面

1. MATLAB工具栏
2. 命令行窗口
3. 当前文件夹窗口
4. 工作区窗口



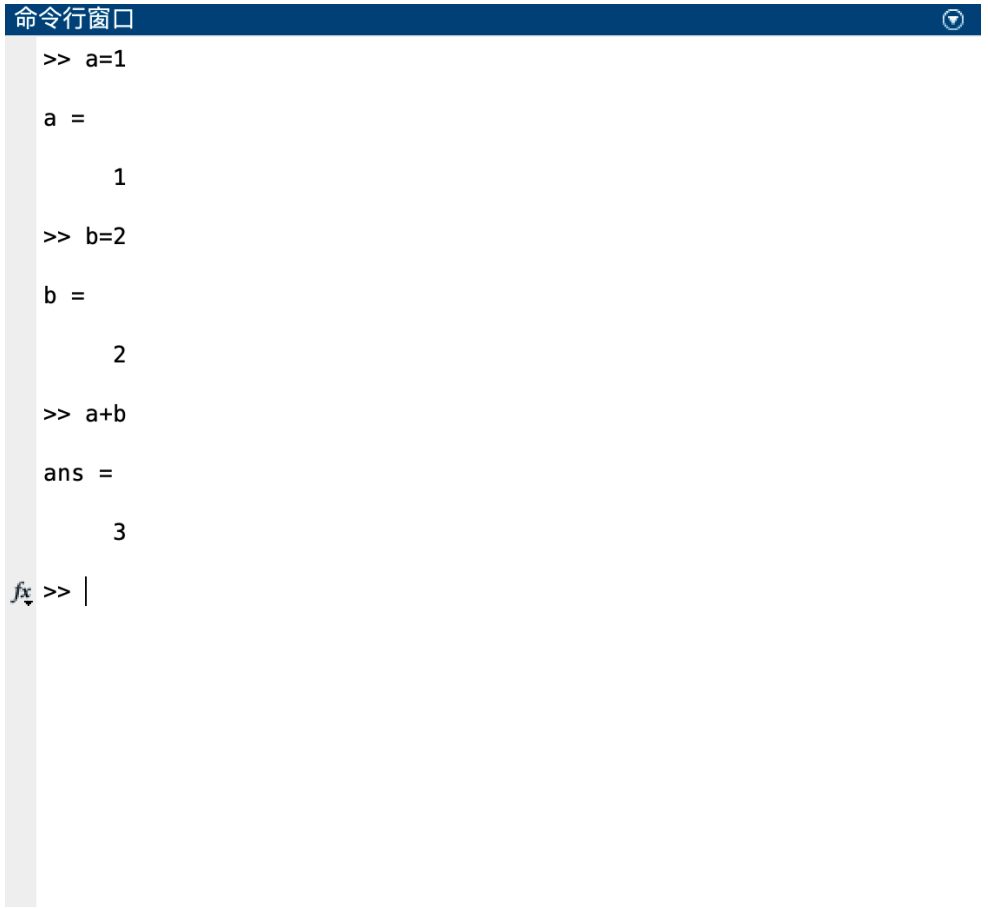
MATLAB工具栏

- 工具栏中最常用的命令按钮
  - 新建
  - 打开
  - 清除工作区
  - 清除命令
  - 布局
  - 设置路径
  - 复制、粘贴等

- 大家将在接下来的学习很快熟悉这些命令按钮

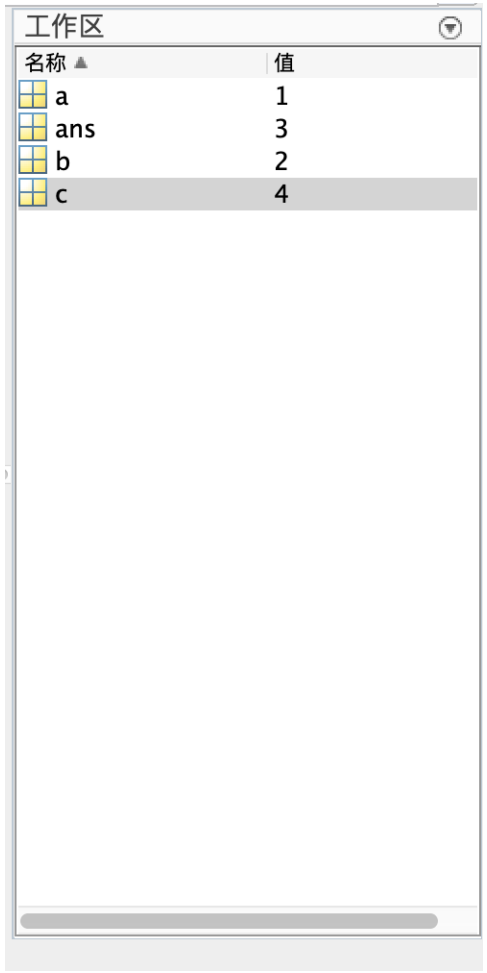
MATLAB命令行窗口

- 命令行窗口用户输入命令，并显示该命令的执行结果
  - 命令行窗口是MATLAB最重要最常用的窗口
  - 请尝试在命令行窗口中输入
    - ✓ >> a=1 +回车
    - ✓ >> b=2 +回车
    - ✓ >> a+b +回车
    - ✓ >> c=a+b +回车
    - ✓ >> c +回车
  - 重复以上命令，但每个命令行以分号结尾
    - ✓ 如： >> a=1; +回车



MATLAB工作区窗口

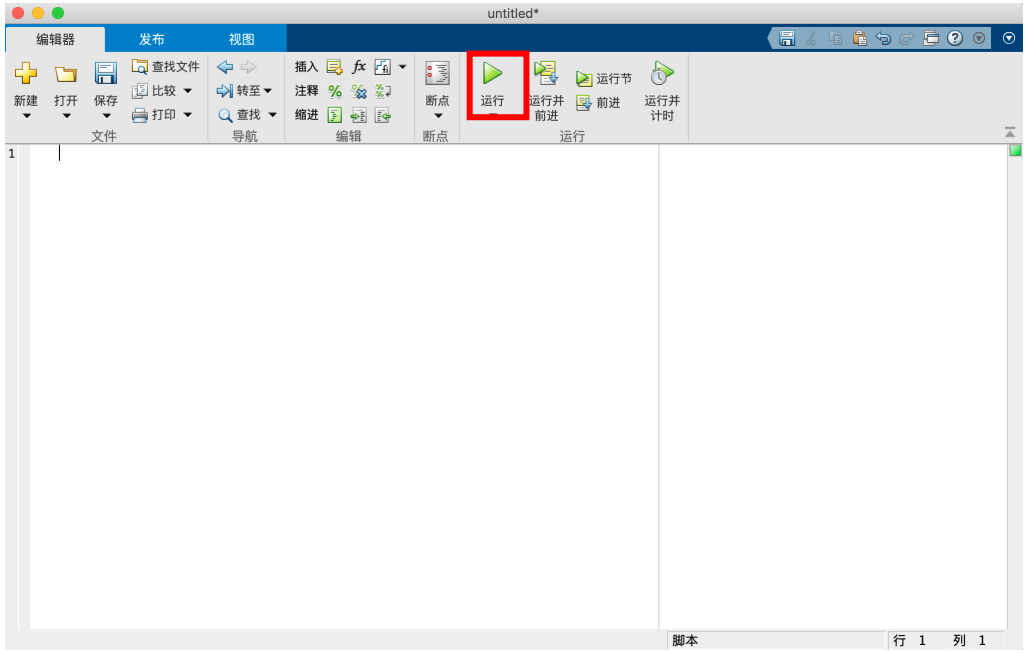
- 工作区窗口展示了MATLAB用来存储所有变量以及运算结果的内存空间
  - 将内存空间可视化
  - 可在工作区窗口中对已定义的变量重新赋值
  - 请在工作区窗口完成以下操作
    - ✓ 双击变量c，对c进行编辑
    - ✓ 将c的值改为4，然后关闭编辑窗口
    - ✓ 在工作区窗口中，c的值变成了4
    - ✓ 在命令行输入c+回车



MATLAB脚本文件

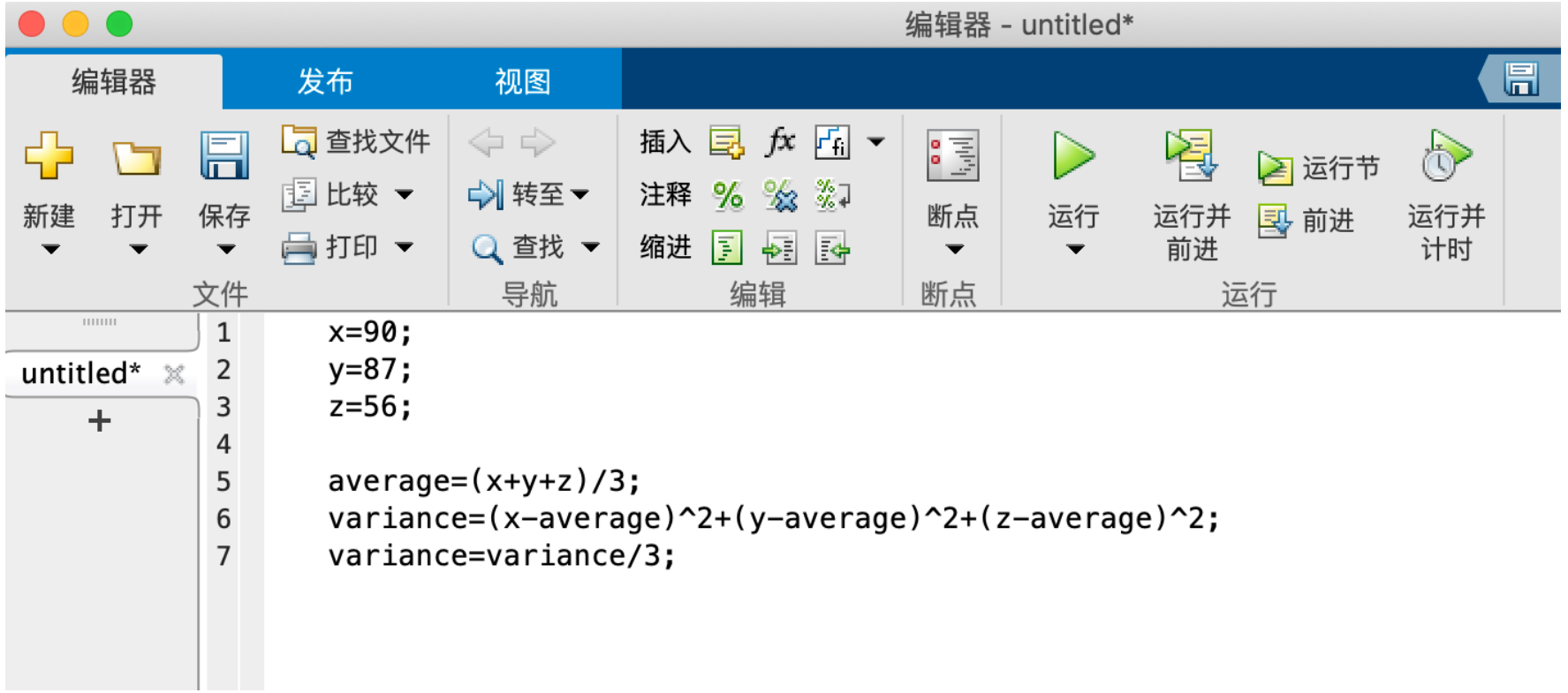
- 在命令行窗口中，每输入一行命令，敲回车后，系统都会自动执行改行的命令。对于比较简单的编程，可以这么做。但是对于比较复杂的编程，就需要启用脚本文件。
- 点击工具栏中的“新建”命令，选择新建“脚本文件”
- 脚本文件的作用和命令行窗口一样，区别在于：

脚本文件可以输入任意多行命令，而系统不会执行任何命令。当点击“运行”按钮之后，系统会逐行执行所有命令。



MATLAB脚本文件

- 首先在电脑中新建一个文件夹，用于存储你将要编写的代码文件。然后在matlab中新建脚本文件，输入以下命令，点击运行，系统会提示保存文件，选中已建好的文件夹，设好文件名，如first_test.m



MATLAB当前文件夹窗口

- 如果你保存的路径不属于MATLAB当前文件夹，则会出现以下提示，点击更改文件夹

- 为了方便起见，可在工具栏的“设置路径”中将你常用的文件夹添加到搜索路径，这样当你在该文件夹下保存/读取/运行程序时不会再出现以上提示

MATLAB脚本文件

- 运行完脚本文件之后，运行结果可以在工作区窗口中进行查看，也可在命令行窗口中输入average或者variance进行查看
- 大多数的编程工作都是在脚本文件中完成，而不是在命令行窗口中进行。命令行窗口一般用于进行非常简短的编程或者测试

进一步熟悉MATLAB操作界面

- 激活命令行窗口，敲击键盘上的“up键”（上下左右的上），可以选择输入某条历史命令，这样对于一些重复命令，无须重新输入
  - 注意：命令行窗口中所显示的运行结果无法编辑，如果不小心执行了错误的命令，无法回撤，只有重新再做一遍。这也是我们经常采用脚本文件进行编程的原因之一。
- 利用工具栏中的“清除命令”来清空工作区/命令历史记录
  - 注意：这里的清除，并没有完全删除这些变量；仍然可以在工作区窗口查看之前的变量/结果
- 利用工具栏中的“清除工作区”来清除工作区窗口中的所有变量
  - 清除之后，这些变量将从内存中消失，无法找回
  - 也可在工作区窗口中删除指定变量

进一步熟悉MATLAB操作界面

- 请自由拖动操作界面的各个子窗口（如：工作区窗口）
  - 拖动之后，如要回复原状，请点击工具栏中“布局”--->“默认”
- 命令行窗口中常用的快捷键命令：
  - >> clc: 相当于“清除命令”按钮
  - >> clear all：相当于“清除工作区”按钮
  - >> clearvars -except x y: 删除除变量x,y之外的所有变量 (x y之间用空格隔开)
  - >> who/whos: 快速查看当前工作区窗口的变量
- 请自由在MATLAB操作界面上进行操作，增加对MATLAB的熟悉感

变量命名及其操作

- 在高级编程语言中，任何被定义的变量都会被保存在内存中。要找到这个变量，我们不需要它的内存地址，只需要知道它的名字即可。
- 在MATLAB中，变量名必须以字母开头，后面可接字母、数字、下划线，变量名最长不超过63个字符，如：
  - >> X123=1;
  - >> intrest_rate=0.03;
  - >> My_name=张三’；
- 变量第一次被定义时必须给其赋值

命令行窗口

```
>> _sd=1;
 _sd=1;
 ↑
错误：文本字符无效。请检查不受支持的符号、不可见的字符或非 ASCII 字符的粘贴。

>> sd
函数或变量 'sd' 无法识别。
```

- 变量名中严格区分大小写，如：x2和X2是两个不同的变量

数据的描述性统计量

- 读取数据(count.dat是matlab自带的数据)
  - >> load count.dat 或者是 importdata('xxx.csv')
- 画图
  - >> plot(count)
- 数据统计信息
  - 从工具栏中选择 **Tools > Data Statistics**
- 也可以调用命令来得到这些统计量
  - >> mu=mean(count);　　mu1=mean(count(:,1));
  - >> sigma=std(count);　　sigma1=std(count(:,1));

变量及结果的存储

- 点击工具栏的“保存工作区”按钮，对工作区中的所有变量进行保存，保存为lishi.dat文件。当下次需要这些变量时，通过读取该文件可以使这些变量重新回到内存空间中去。
- 请进行如下操作
  - 点击“保存工作区”，对工作区进行保存
  - 清除工作区，之后，工作区窗口为空
  - 点击“导入数据”，选择刚才保存的lishi .dat 文件
  - 被清除的变量又回到了工作区当中

矩阵与子矩阵

- MATLAB中，所有变量都是一个矩阵，一般的矩阵变量定义如下：

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

X =

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

- 行向量或列向量在MATLAB中属于只有一行或一列的矩阵

```
>> row_x=[1,2,3]

row_x =

     1     2     3
```

```
>> col_y=[1;4;7]

col_y =

     1
     4
     7
```

- 标量在MATLAB中属于1x1的矩阵，如：x_2=1;(等价于x_2=[1];)

矩阵与子矩阵

- Matlab中生成一些特殊矩阵：
  - Y=zeros(3,5); 生成一个3行5列元素全为零的矩阵；
  - Y=ones(3,5); 生成一个3行5列元素全为1的矩阵；
  - Y=eye(4); 生成一个4行4列的单位阵；
  - Y=rand(3,5); 生成一个3行5列元素全为0～1之间随机数的矩阵。

矩阵与子矩阵

- MATLAB中，对矩阵的操作非常方便，对得起Matrix Laboratory这个名字
  - 以刚定义的矩阵X为例，X(2,3)表示该矩阵的第二行第三列元素

```
>> X(2,3)

ans =

     7
```

  - X(1,:)表示由X的第一行构成的行向量；X(:,2)表示由X的第二列构成的列向量；
  - X(1:3,2:4)表示由第1～3行第2～3列的元素构成的子矩阵；
  - 如果要提取矩阵X的第1、3列第2、4行构成的子矩阵，怎么办？

```
>> index1=[1,3];
>> index2=[2,4];
>> X(index1,index2)

ans =

     2     4
    10    12
```

编辑已有矩阵

- 可在工作区窗口中直接对某矩阵进行编辑（自行尝试）
- 利用命令进行编辑
  - 对原矩阵的末尾增加一行一列

```
>> new_row=[17,17,17,17];
>> Y=[X;new_row]

Y =

    -1     2     3     4
     5    -6     7     8
     9    10   -11    12
    13    14    15   -16
    17    17    17    17

>> new_col=[18;18;18;18;18];
>> Z=[Y,new_col]

Z =

    -1     2     3     4    18
     5    -6     7     8    18
     9    10   -11    12    18
    13    14    15   -16    18
    17    17    17    17    18
```

  - 删除矩阵的第3行：>> Z(3,:)=[]; 删除第3列：>> Z(:,3)=[];

编辑已有矩阵

- 将X的第1行第二列元素改为100
  - >> X(1,2)=100;
- X的第2行统一乘以2
  - >> X(2,:)=X(2,:)*2;
- X的第2行分别乘以[2,3,4,5]
  - >> X(2,:)=X(2,:).*[2,3,4,5];
- 令X的第1、3行，第2、4列的四个元素分别等于10086
  - >> X([1,3],[2,4])=10086
- 对矩阵进行运算时，“.*”“./”表示两个矩阵的对应元素进行加减乘除
  - 自行定义一个列向量X,一个行向量Y，体会X*Y与X.*Y的区别

编辑已有矩阵

- 在X的第1列与第2列中间插入一列 [-10;-10;-10;-10]
  - >> X=[X(:,1),[-10;-10;-10;-10],X(:,2:4)];

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

X =

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

>> X=[X(:,1),[-10;-10;-10;-10],X(:,2:4)]

X =

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

- 同理，如何在X的第1行与第2行之间插入一行呢?

编辑已有矩阵

- 交换X矩阵的第2行和第3行

```
X =

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

>> X([2,3],:)=X([3,2],:)

X =

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

- 交换两列怎么办?

练习

- 利用matlab求解下列线性方程组

$$\begin{bmatrix} 6 & 1 & & & & & \\ 8 & 6 & 1 & & & & \\ & 8 & 6 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 8 & 6 & 1 & \\ & & & & 8 & 6 & 1 \\ & & & & & 8 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ \vdots \\ x_{99} \\ x_{100} \end{bmatrix} = \begin{bmatrix} 7 \\ 15 \\ 15 \\ \vdots \\ \vdots \\ 15 \\ 14 \end{bmatrix}$$

- 输入系数矩阵A和右端向量b，然后通过

x=A^(-1)*b 或者 x=linsolve(A,b)

来得到方程组的解

对数组的操作

- 定义数组（向量）
  - >> X=[-1,7,-3,8,-5,9];
- 提取数组中大于等于7的元素
  - >> find(X>=7); 找出>=7的所有元素在数组的位置
  - >> X(X>=7); 提取数组中大于等于7的元素,构成一个新的数组
  - >> X(X>=7&X<=8); “&”表示“且”
  - >> X(X>=7|X<=-3); “|”表示“或”

```
>> X=[-1,7,-3,8,-5,9];
>> find(X>=7)

ans =

     2     4     6

>> X(X>=7)

ans =

     7     8     9

>> X(X>=7&X<=8)

ans =

     7     8

>> X(X>=7|X<=-3)

ans =

     7    -3     8    -5     9
```

字符的处理

- MATLAB中有两种基本的数据类型
  - 数值型数据
  - 字符型数据
- 字符型数组是用单引号括起来的数据
  - >> X='i love china'
    >> Y='i ''love'' china';
  - 单引号本身不作为字符，如果想要让单引号作为一个字符，需要用两个单引号来代替
- 字符型数组的操作和数值型数组相似
  - >> X(4:8) 取第4到第8个字符组成的子字符串
  - >> X(3:6)= ‘like’ 对X的第3到第6个字符重新赋值

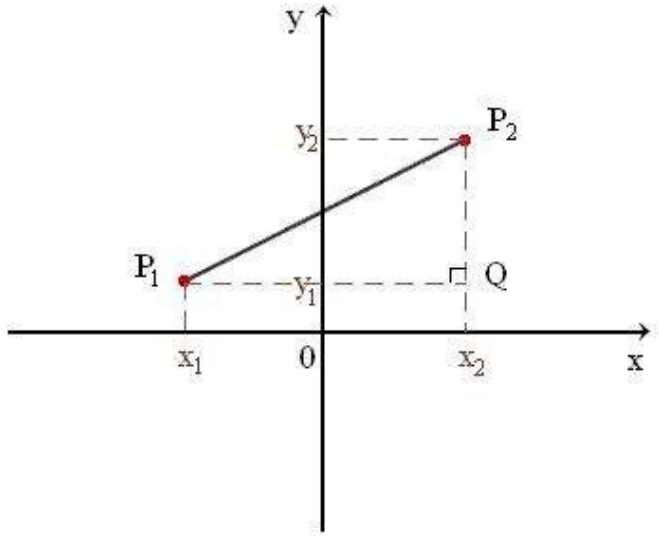
```
>> X='i love china'

X =

    'i love china'

>> Y='i ''love'' china'

Y =

    'i 'love' china'

>> X(4:8)

ans =

    'ove c'

>> X(3:6)='like'

X =

    'i like china'
```

脚本文件和函数文件

- 脚本文件是可在命令行窗口直接执行的文件，也叫命令文件。
- 函数文件是定义一个函数，不能直接执行，而必须以函数调用的方式来调用它。
- 函数名须与文件名一致
- 函数名须与matlab自带函数的名字区分开来
- 所有子函数须和主程序放在同一目录下

练习：求平面上任意两点的距离

- 平面上两点的坐标分别为$P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$.
  请用MATLAB编程，能够求解任意P_1, P_2之间的距离。



练习：求平面上任意两点的距离

- 首先新建函数文件（新建脚本文件也可以），输入以下代码并保存为distance.m

```
function y = distance(x1,x2)
%该函数用来求平面坐标上两点之间的距离，x1,x2均为二维数组，分别表示两个点的坐标
%distance为函数名
%y为返回值
y=(x1-x2).^2;
y=sum(y);
y=sqrt(y);
%上面三行命令完全可以用一行命令来代替，如下：
%y=sqrt(sum((x1-x2).^2));

end
```

- 注意：“%”是注释符号，一行中位于“%”后面的任意字符都不会被执行
  - 通常利用“%”来对程序进行注解，以便于他人能够读懂
- 在已知两点坐标的情形下，这里定义的distance函数可以求出两点之间的距离

练习：求平面上任意两点的距离

- 建立主程序文件，即新建脚本文件，输入以下代码，保存为qiu_juli.m

```
%该程序用于求解平面上任意两点的距离，每个点的坐标由他人指定

clear %用于清除内存空间中的其它变量，以避免冲突。通常不需要该命令

%接下来由他人制定每个点的坐标
x1=input('请输入平面上第一个点的横坐标,如:[-34.3,51.5],以回车键结束:');
x2=input('请输入平面上第二个点的横坐标,如:[-34.3,51.5],以回车键结束:');

%知道两点的坐标后，调用已经编写的distance函数来求出距离
disp('您所输入的两点之间的距离为：');
d=distance(x1,x2);
disp(d);

```

- 主程序文件所调用的所有函数必须和主程序文件在同一目录下
- 运行你所编的程序

if语句

- 当条件结果为标量时，非零表示条件成立，零表示条件不成立。
  - 例：建立脚本文件，并运行以下代码，保存为chisha.m

```
1 -    A='吃蛋糕';
2 -    B='吃烧烤';
3 -    x=rand(1,1);%随机产生一个0～1之间的数
4 -    if x<=1/2
5 -        disp(A);
6 -    else
7 -        disp(B);
8 -    end
```

练习：计算身份证最后一位

- 身份证号码由18位数组成，第1，2位数字表示所在省份的代码。
- 第3，4位数字表示所在城市的代码，第5，6位数字表示所在区县的代码，第7到14位表示出生年，月，日。
- 第15，16位数字表示所在地派出所的代码，第17位数字表示性别，奇数表示男性，偶数表示女性，第18位数字是校验码，由号码编制单位按统一公式计算而得。
- 如果我们知道前17位，如何计算第18位号码呢?

练习：计算身份证最后一位

1. 将前面的身份证号码17位数分别乘以不同的系数。从第一位到第十七位的系数分别为：
   7 9 10 5 8 4 2 1 6 3 7 9 10 5 8 4 2
2. 将这17位数字与系数分别相乘的结果加起来。
3. 用加出来和除以11，看余数是多少？
4. 余数只可能有0 1 2 3 4 5 6 7 8 9 10。其分别对应的最后一位身份证的号码为1 0 X 9 8 7 6 5 4 3 2。

练习：计算身份证最后一位

- 新建脚本文件，输入以下代码，保存为sfz_18_jisuan.m

```
%当知道身份证的前17位时，该程序用来计算第18位检验码
clear %用于清除内存中的变量

%由他人输入身份证前17位数字
sfz17_str=input('请输入你的身份证号码前17位,以回车键结束：','s');

sfz17_array=str2num(sfz17_str(:))';

%身份证号码前17位数分别乘以不同的系数，系数为：
xishu=[7 9 10 5 8 4 2 1 6 3 7 9 10 5 8 4 2];
yushu=mod(sfz17_array*xishu',11);%计算余数
code_list=[1 0 10 9 8 7 6 5 4 3 2]; %检验码序列,X用10代替
veri_code=code_list(yushu+1);
if veri_code~=10
    disp('您的身份证最后一位是：')
    disp(veri_code)
else
    disp('您的身份证最后一位是：')
    disp('X')
end
```

for语句

- for语句针对向量的每一个元素执行一次循环体。

```
for k=[1,3,2,5]
    k
end
```

循环四次

- 退出循环之后，循环变量的值就是向量中最后的元素值。

```
for k=1:2:10

end

k
```

1 3 5 7 9

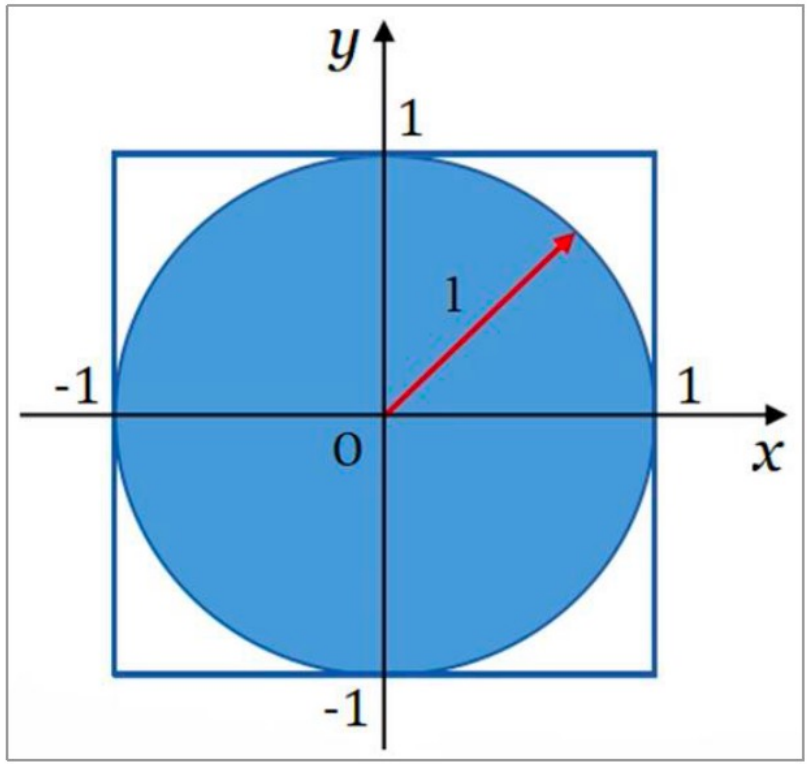
- 当向量为空时，循环体一次也不执行。

```
for k=1:-2:10
    k
end
```

空向量

for语句

（3）利用蒙特卡洛法求π 的近似值。



在正方形内随机投点，设点落在圆内的概率为P。

$P=\pi/4 \longrightarrow \pi=4P$

P=落在圆内的点数/所投点的总数

所投的点落在圆内的充要条件是$x^2+y^2\leqslant 1$。

for语句

```
S=0
n=10000
for i=1:n
        x=rand(1);
        y=rand(1);
        if x^2+y^2<=1
                s=s+1;
        end
end
pai=s/n*4
```

break语句和continue语句

- break语句用来跳出循环体，结束整个循环。
- continue语句用来结束本次循环,接着进行下一次是否执行循环的判断。

break语句和continue语句

- 例 求[100，200]之间第一个能被21整除的整数。

```
for n=100:200
    if rem(n,21)~=0
        continue
    end
    n
    break
end
```

While语句

- 例: 利用while循环求解上题

```
n=100;
while rem(n,21)~=0
    n=n+1;
end
disp(n);
```

有理式计算

- 有时，我们希望计算机把结果保存成分数而不是小数。可采用如下命令

```
format rat %使得接下来的计算结果都保留分数形式
a = 1;
b = 3;
disp(a/b+0.2);
format   %恢复默认形式
```

- 适用于小规模四则运算

符号运算

- 有时，计算公式中不全是数字，带有数学符号，这时的运算称为符号运算

```
syms x y z;
f=2*x^2+3*y-5;
g=x^2-z+7;
disp(f+g) %输出为 3*x^2 + 3*y - z + 2
```

符号运算

- 一些基本命令：
  - collect(S,x)—— 按指定变量x的次数对符号多项式S合并同类项

```
>> S = x^2*y+y*x–x^2–2*x;
collect(S) %此处默认x为符号变量
ans =
(y – 1)*x^2 + (y – 2)*x

>> collect(S,y) %此处选择y为符号变量
ans =
(x^2 + x)*y – x^2 – 2*x
```

符号运算

- expand(S) —— 将符号表达式S展开

```
>> S = (x - 2)*(x - 4);
>> expand(S)
ans =
x^2 - 6*x + 8
```

- factor(S) —— 将符号表达式S因式分解

```
>> S = x^2 - 6*x + 8;
>> factor(S)
ans =
 [x - 2, x - 4]
```

符号运算

- simplify(m)——对符号表达式m进行化简。如：

$$m = \begin{bmatrix} a^3 - b^3 & \sin^2\alpha + \cos^2\alpha \\ \frac{15xy - 3x^2}{x - 5y} & 78 \end{bmatrix}$$

```
>> syms a b x y alp
>> m=[a^3-b^3,sin(alp)^2+cos(alp)^2;(15*x*y-3*x^2)/(x-5*y),78] ;
>> simplify(m)        %对符号矩阵化简处理
ans =
[a^3 - b^3,  1]
[     -3*x, 78]
```

符号运算

- 求解带有符号的线性方程组

$$\begin{cases}(1-\lambda)x_1-2x_2+4x_3=1\\ 2x_1+(3-\lambda)x_2+x_3=1\\ x_1+x_2+(1-\lambda)x_3=1\end{cases}$$

- 代码如下：

```
syms lamda
A=[1-lamda,-2,4;2,3-lamda,1;1,1,1-lamda];
b=[1 1 1]';
linsolve(A,b)
ans =
-(- lamda^2 + 2*lamda + 6)/((- lamda^2 + 2*lamda)*(lamda - 3))
-(lamda^2 + lamda + 3)/(lamda^3 - 5*lamda^2 + 6*lamda)
-(lamda^2 - 2*lamda + 3)/(lamda^3 - 5*lamda^2 + 6*lamda)
```

符号运算

- symsum(S,v,a,b)：对符号表达式S中的指定变量v从a到b求和

例如计算：

$$1+2+3+\ldots\ldots+n=\sum_{k=1}^{n}k$$

- 代码如下：

```
syms k n

S = k;

F1 = symsum(S,k,1,n)

F1 =

(n*(n + 1))/2
```

符号运算

- Matlab可以对符号公式进行很多的运算，包括求极限、算不定积分、求导等等。留给大家自行摸索。

- 符号运算的用途：理论分析、学生写作业、家长辅导小孩写作业。