5 Octave导学

2022.10.21

5.1基本操作

运算

```
5+6
```

```
ans =
    11
```

```
3-2
```

```
ans =
     1
```

```
5*8
```

```
ans =
    40
```

```
4>1/2
```

```
ans = logical
   1
```

```
2^6
```

```
ans =
    64
```

```
1 == 2
```

```
ans = logical
   0
```

```
1 ~= 2
```

```
ans = logical
   1
```

```
1 && 0
```

```
ans = logical
   0
```

```
1 || 0
```

```
ans = logical
   1
```

```
xor(1,0)
```

```
ans = logical
   1
```

变量

```
a = pi;
a
```

```
a = 
   3.141592653589793
```

```
%命令: format long a %显示完整的变量值
disp(a)
```

```
   3.141592653589793
```

```
disp(sprintf('2 decimals:%0.2f',a))
```

```
2 decimals:3.14
```

```
disp(sprintf('6 decimals:%0.6f',a))
```

```
6 decimals:3.141593
```

矩阵

```
A = [1 2; 3 4; 5 6]
```

```
A = 3×2
     1     2
     3     4
     5     6
```

```
v = [1 2 3]
```

```
v = 1×3
     1     2     3
```

```
v = [1;2;3]
```

```
v = 3×1
     1
     2
     3
```

```
v = 1:0.1:2
```

```
v = 1×11
   1.000000000000000   1.100000000000000   1.200000000000000   1.300000000000000 ···
```

```
v = 1:6
```

```
v = 1×6
     1     2     3     4     5     6
```

```
ones(2,3)
```

```
ans = 2×3
     1     1     1
     1     1     1
```

```
c = 2*ones(2,3)
```

```
c = 2×3
     2     2     2
     2     2     2
```

```
w = zeros(1,3)
```

```
w = 1×3
     0     0     0
```

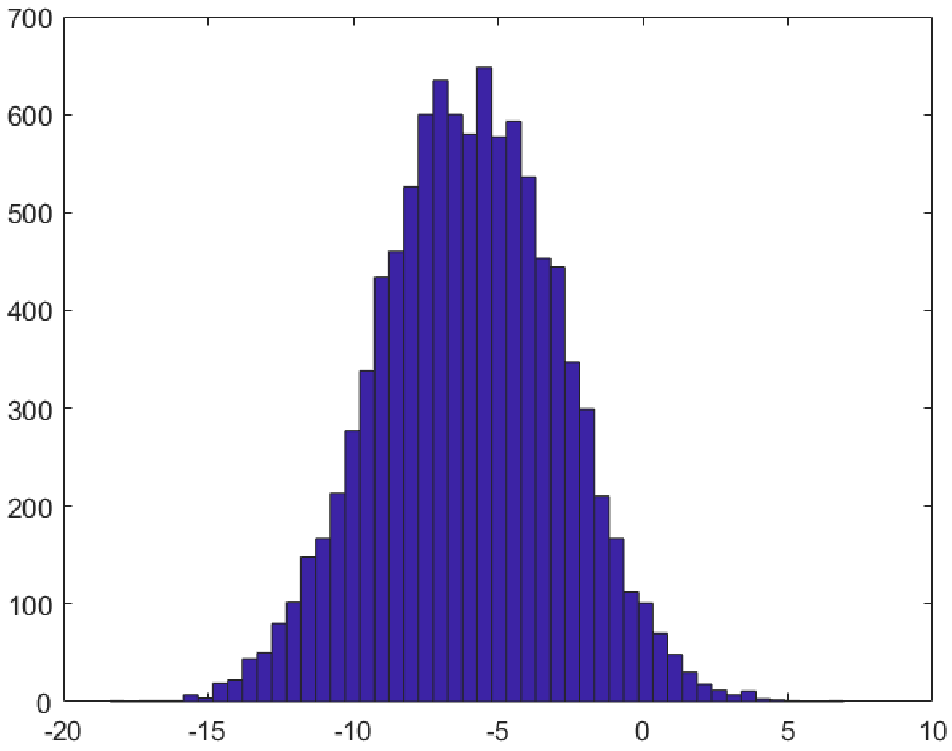
```
w = rand(1,3)
```

```
w = 1×3
   0.156003156891034   0.398367064139535   0.882504945784612
```

```
w = randn(1,3) %高斯随机分布
```

```
w = 1×3
   0.155205633826749   0.178595629002979  -0.337730845992083
```

```
w = -6 + sqrt(10)*(randn(1,10000));
hist(w,50) %绘制直方图
```



```
I = eye(4)
```

```
I = 4×4
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

5.2移动数据

```
size(A)
```

```
ans = 1×2
     3     2
```

```
size(A,1)
```

```
ans =
     3
```

```
length(v)
```

```
ans =
     6
```

```
length(A)
```

```
ans =
     3
```

```
length([1;2;3;4;5])
```

```
ans =
     5
```

```
%命令: load featuresX.dat %上传数据
%load(featuresx.dat)
%who %显示所有变量
%size(featuresx) %显示矩阵大小
%whos %显示变量的详细信息
%clear featuresx %清除变量
%v = featuresx(1:10) %将变量的前十个元素赋给v
%save hello.mat v; %将v保存在hello.mat文件中
%clear %删除空间中的所有变量
%save hello.text v -ascii %保存为用ascii编码的文档
```

```
%数据索引
A(3,2)
```

```
ans =
     6
```

```
A(2,:)%获得第二列所有元素
```

```
ans = 1×2
     3     4
```

```
A([1 3],:)%获得第一行和第三行的所有列的元素
```

```
ans = 2×2
     1     2
     5     6
```

```
A(:,2) = [10;11;12]%对A的第二列进行赋值
```

```
A = 3×2
     1    10
     3    11
     5    12
```

```
A = [A,[100;101;102]]%对A附加一列新向量
```

```
A = 3×3
     1    10   100
     3    11   101
     5    12   102
```

```
A(:)%将A中所有元素放入一个单列的向量
```

```
ans = 9×1
     1
     3
     5
    10
    11
    12
   100
   101
   102
```

```
A = [1 2;3 4;5 6];
B = [11 12;13 14;15 16];
C = [A B]
```

```
C = 3×4
     1     2    11    12
     3     4    13    14
     5     6    15    16
```

```
C = [A;B]
```

```
C = 6×2
     1     2
     3     4
     5     6
    11    12
    13    14
    15    16
```

5.3计算数据

```
A = [1 2; 3 4; 5 6];
B = [11 12; 13 14; 15 16];
C = [1 1; 2 2];
A*C
```

```
ans = 3×2
     5     5
    11    11
    17    17
```

```
A.*B
```

```
ans = 3×2
    11    24
    39    56
    75    96
```

```
A.^2
```

```
ans = 3×2
     1     4
     9    16
    25    36
```

```
v = [1; 2; 3];
1./v
```

```
ans = 3×1
    1.0000
    0.5000
    0.3333
```

```
log(v)
```

```
ans = 3×1
         0
    0.6931
    1.0986
```

```
exp(v)
```

```
ans = 3×1
    2.7183
    7.3891
   20.0855
```

```
abs([-1; 2; 3])
```

```
ans = 3×1
     1
     2
     3
```

```
-v
```

```
ans = 3×1
    -1
    -2
    -3
```

```
%将v中所有元素加1
v + ones(length(v),1)
```

```
ans = 3×1
     2
     3
     4
```

```
v+1
```

```
ans = 3×1
     2
     3
     4
```

```
A'
```

```
ans = 2×3
     1     3     5
     2     4     6
```

```
a = [1 15 2 0.5];
val = max(a)
```

```
val = 15
```

```
[val, ind] = max(a)
```

```
val = 15
ind = 2
```

```
a < 3
```

```
ans = 1×4 logical ##
   1   0   1   1
```

```
find(a < 3)%返回所有小于3的元素
```

```
ans = 1×3
     1     3     4
```

```
A = magic(3)%幻方矩阵
```

```
A = 3×3
     8     1     6
     3     5     7
     4     9     2
```

```
[r,c] = find(A>=7 )
```

```
r = 3×1
     1
     3
     2
c = 3×1
     1
     2
     3
```

```
sum(a)
```

```
ans = 18.5000
```

```
prod(a)
```

```
ans = 15
```

```
floor(a)%舍去小数部分
```

```
ans = 1×4
     1    15     2     0
```

```
ceil(a)%向上取整
```

```
ans = 1×4
     1    15     2     1
```

```
rand(3)
```

```
ans = 3×3
    0.0046    0.8687    0.2599
    0.7749    0.0844    0.8001
    0.8173    0.3998    0.4314
```

```
max(rand(3),rand(3))
```

```
ans = 3×3
    0.9106    0.5132    0.5797
    0.6221    0.4018    0.5499
    0.3510    0.8693    0.1839
```

```
max(A,[],1)%得到A每一列的最大值
```

```
ans = 1×3
     8     9     7
```

```
max(A,[],2)%每一行的最大值
```

```
ans = 3×1
     8
     7
     9
```

```
%求每一行每一列的最大元素
max(max(A))
```

```
ans = 9
```

```
max(A(:))
```

```
ans = 9
```

```
A = magic(9)
```

```
A = 9×9
    47    58    69    80     1    12    23    34    45
    57    68    79     9    11    22    33    44    46
    67    78     8    10    21    32    43    54    56
    77     7    18    20    31    42    53    55    66
     6    17    19    30    41    52    63    65    76
    16    27    29    40    51    62    64    75     5
    26    28    39    50    61    72    74     4    15
    36    38    49    60    71    73     3    14    25
    37    48    59    70    81     2    13    24    35
```

```
sum(A,1)%求每一列的总和
```

```
ans = 1×9
   369   369   369   369   369   369   369   369   369
```

```
sum(A,2)%每一行的总和
```

```
ans = 9×1
   369
   369
   369
   369
   369
   369
   369
   369
   369
```

```
A.*eye(9)
```

```
ans = 9×9
    47     0     0     0     0     0     0     0     0
     0    68     0     0     0     0     0     0     0
     0     0     8     0     0     0     0     0     0
     0     0     0    20     0     0     0     0     0
     0     0     0     0    41     0     0     0     0
     0     0     0     0     0    62     0     0     0
     0     0     0     0     0     0    74     0     0
     0     0     0     0     0     0     0    14     0
     0     0     0     0     0     0     0     0    35
```

```
sum(sum(A.*eye(9)))
```

```
ans = 369
```

```
sum(sum(A.*flipud(eye(9))))%反对角线求和
```

```
ans = 369
```

```
flipud(eye(9))%翻转函数
```

```
ans = 9×9
     0     0     0     0     0     0     0     0     1
     0     0     0     0     0     0     0     1     0
     0     0     0     0     0     0     1     0     0
     0     0     0     0     0     1     0     0     0
     0     0     0     0     1     0     0     0     0
     0     0     0     1     0     0     0     0     0
     0     0     1     0     0     0     0     0     0
     0     1     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0
```

```
A = magic(3)
```

```
A = 3×3
     8     1     6
     3     5     7
     4     9     2
```

```
temp = pinv(A)
```

```
temp = 3×3
    0.1472   -0.1444    0.0639
```

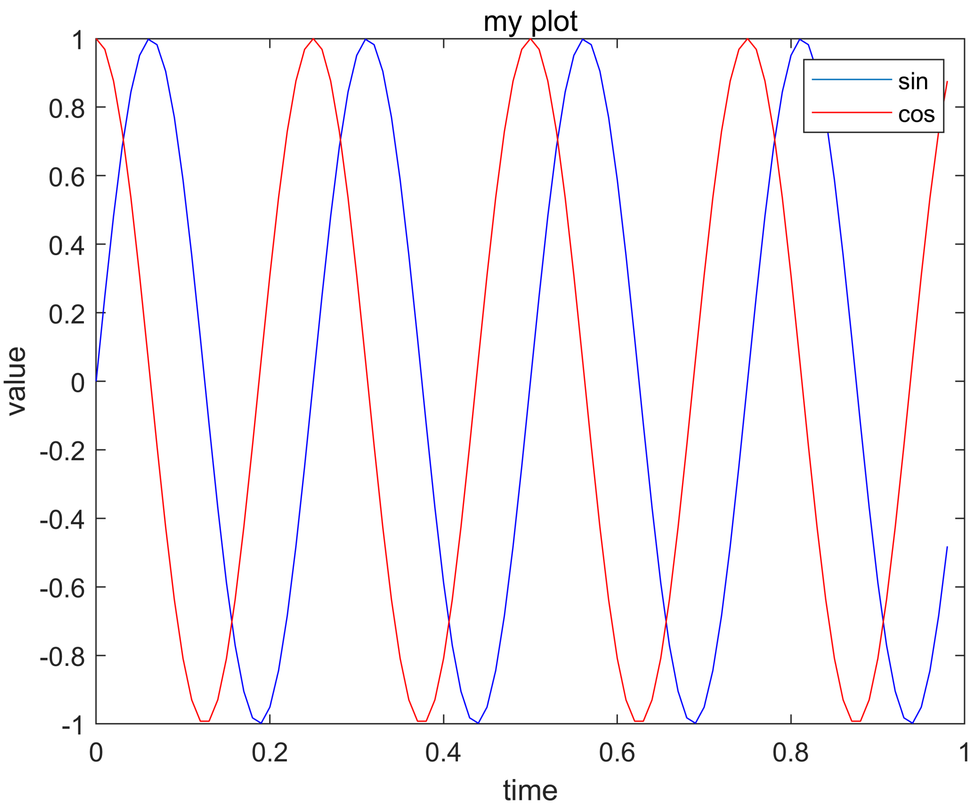
```
   -0.0611    0.0222    0.1056
   -0.0194    0.1889   -0.1028
```

```
%获得单位矩阵
temp*A
```

```
ans = 3×3
    1.0000    0.0000   -0.0000
   -0.0000    1.0000    0.0000
    0.0000   -0.0000    1.0000
```

5.4数据绘制

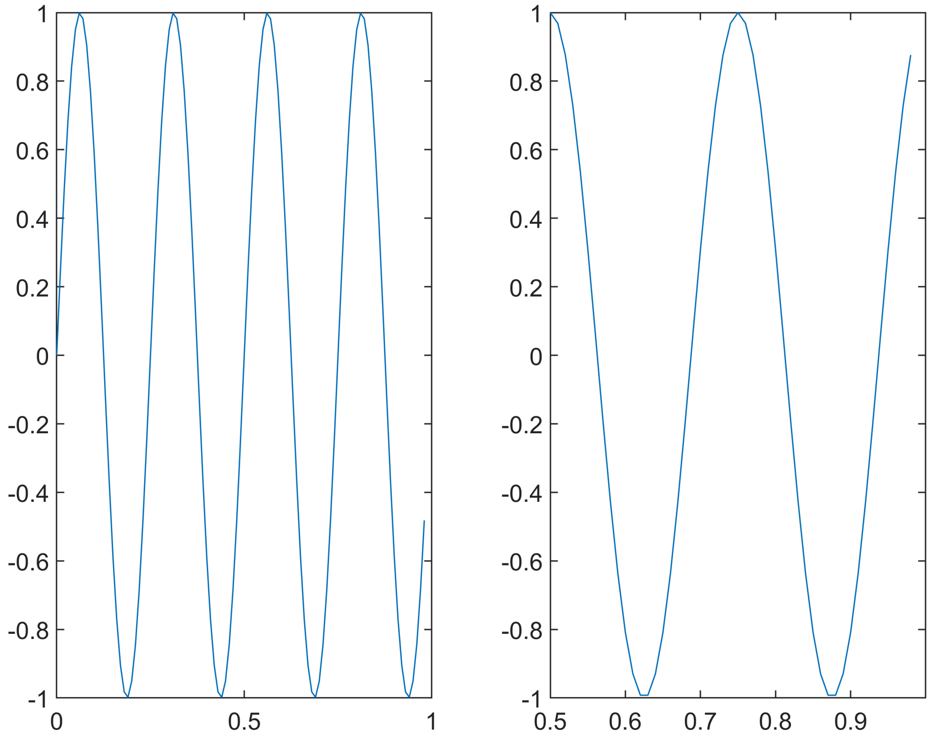
```
t = [0:0.01:0.98];
y1 = sin(2*pi*4*t);
plot(t,y1,'b');
hold on
y2 = cos(2*pi*4*t);
plot(t,y2,'r');
xlabel('time');
ylabel('value');
legend('sin','cos');
title('my plot');
```



```
%命令print -dpng 'myPlot.png' 导出图片格式
%figure(1)对绘制的图进行标号
```

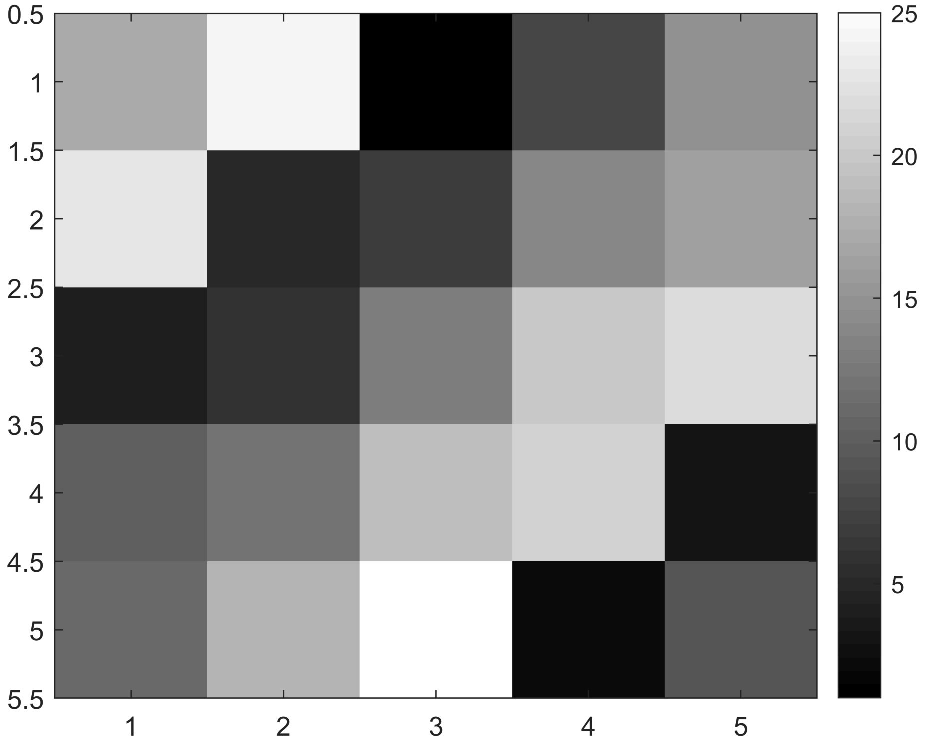
```
subplot(1,2,1);%将图像分为一行二列的格子，使用第一个各自
```

```
plot(t,y1);
subplot(1,2,2);
plot(t,y2);
axis([0.5 1 -1 1])%改变轴的刻度
```



```
%clf清除所有图像
```

```
clf
A = magic(5);
imagesc(A)%可视化矩阵
colorbar
colormap gray
```



5.5控制语句

```
v = zeros(10,1);
for i=1:10,
    v(i) = 2^i;
end
v
```

```
v = 10×1
           2
           4
           8
          16
          32
          64
         128
         256
         512
        1024
```

```
indices = 1:10;
for i = indices,
    disp(i);
end
```

```
     1

     2

     3
     4

     5

     6

     7

     8

     9

    10
```

```
i=1;
while i<=5,
    v(i) = 100;
    i = i+1;
end
v
```

```
v = 10×1
       100
       100
       100
       100
       100
        64
       128
       256
       512
      1024
```

```
i = 1;
while true,
    v(i)=999;
    i = i+1;
    if i == 6,
        break;
    end
end
v
```

```
v = 10×1
       999
       999
       999
       999
       999
        64
       128
       256
       512
      1024
```

```
if v(1)==1,
    disp('值是1');
elseif v(1) == 2,
    disp('值是2');
else
    disp('值不是1或2');
end
```

```
值不是1或2
```

```
squareThisNumber(5)
```

```
ans = 25
```

```
[a,b] = squareAndCubeThisNumber(5)
```

```
a = 25
b = 125
```

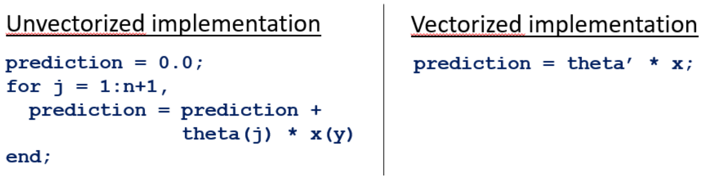
```
x = [1 1;1 2;1 3];
y = [1;2;3];
theta = [0;1];
j = costFunctionJ(x,y,theta)
```

```
j = 0
```

5.6矢量化

$$h_\theta(x) = \sum_{j=\theta}^{n} \theta_j x_j$$

$$= \theta^T x$$



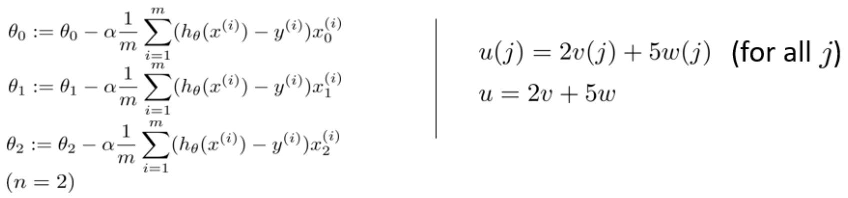
Unvectorized implementation

```
prediction = 0.0;
for j = 1:n+1,
  prediction = prediction +
               theta(j) * x(y)
end;
```

Vectorized implementation

```
prediction = theta' * x;
```

将变量矢量化后，代码会更加简洁。



$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)}$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)}$$

$(n = 2)$

$u(j) = 2v(j) + 5w(j)$ (for all j)

$u = 2v + 5w$

```
function y = squareThisNumber(x)
y = x^2;
end

function [y1,y2] = squareAndCubeThisNumber(x)
y1 = x^2;
```

```
y2 = x^3;
end

function J = costFunctionJ(X,y,theta)

% X is the "design matrix" containing our training examples.
% y is the class labels

m = size(X,1);            %number of training examples
predictions = X*theta;    %predictions of hypothesis on all m examples
sqrErrors = (predictions-y).^2; %suqared errors

J = 1/(2*m)*sum(sqrErrors);
end
```