Task1

```
data=load('multi.txt');%load data
n=size(data,1);%count data number
randnum = randperm(n);%upset the oder
a=randnum(1:0.8*n);%training set
b=randnum(0.8*n+1:n);%test set
trainset = data(a,:);
testset = data(b,:);

%%=======================Train=============================
===

%declare variable X1,X2,X3,assign value from training set
X1 = trainset(:,1); X2 = trainset(:,2);X3=trainset(:,3);y = trainset(:,4);
m = length(y); % number of training examples

X = [ones(m, 1),X1,X2,X3]; % Add a column of ones to x
theta=zeros(4,1);%initalization matrix

% Some gradient descent settings
iterations = 1500;%
alpha = 0.001;

% compute and display initial cost
computeCost(X, y, theta)

% run gradient descent
theta = gradientDescent(X, y, theta, alpha, iterations);

% print theta to screen
fprintf('Theta found by gradient descent: ');
fprintf('%f %f %f %f \n', theta(1), theta(2),theta(3),theta(4));

%%=========================Test============================
======
%declare variable X1,X2,X3,assign value from training set
Xa = testset(:,1); Xb = testset(:,2);Xc=testset(:,3);y2 = testset(:,4);
u=length(y2); % number of testing examples

% print erro tate to screen
yp=[ones(u, 1),Xa,Xb,Xc]*theta;
errorate=sum((yp-y2).^2)/u;
fprintf('the errorate :');
```

```
fprintf('%f \n',errorate);
```

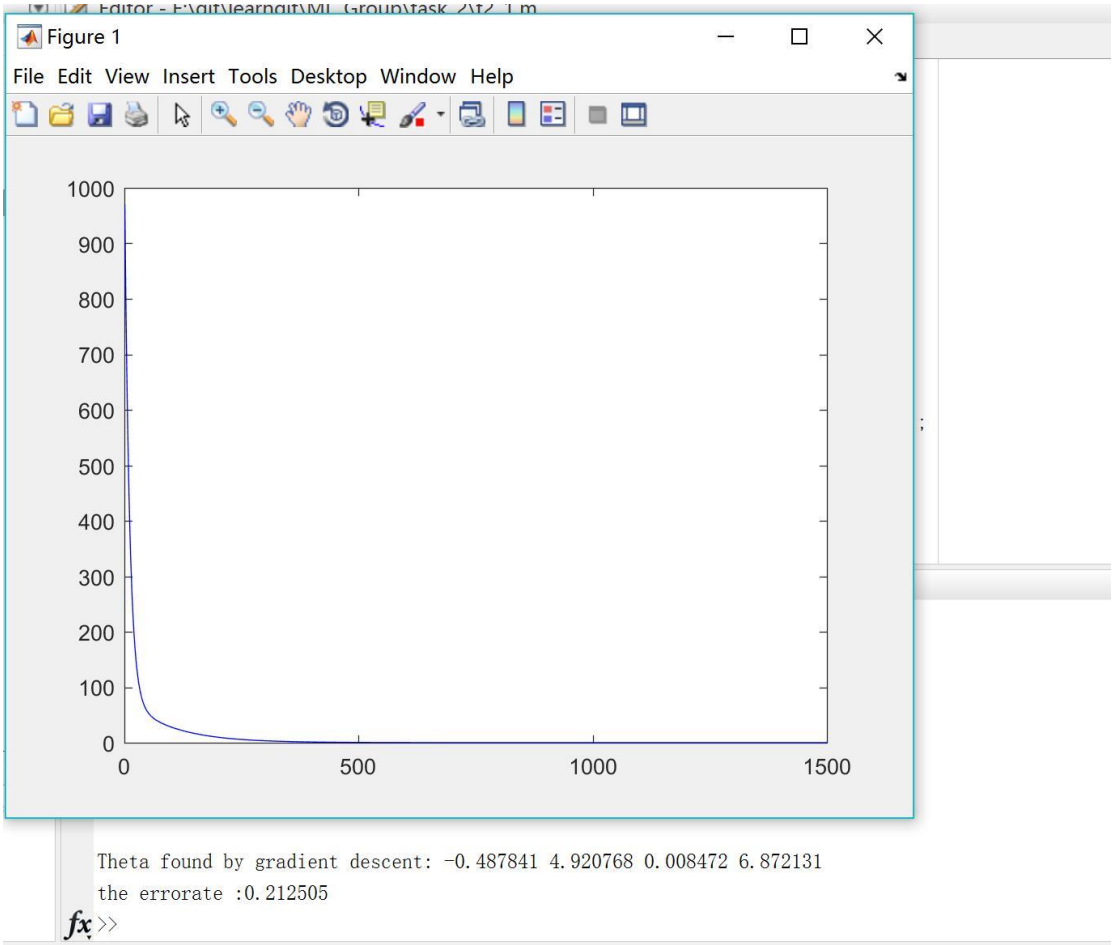
function gradientdescent

```
function [theta, J_history] = gradientDescent(X, y, theta, alpha, num_iters)
%GRADIENTDESCENT Performs gradient descent to learn theta
%   theta = GRADIENTDESENT(X, y, theta, alpha, num_iters) updates theta by
%   taking num_iters gradient steps with learning rate alpha

% Initialize some useful values
m = length(y); % number of training examples
J_history = zeros(num_iters, 1);
i=zeros(num_iters,1);
for iter = 1:num_iters

    %           ======================           YOUR           CODE           HERE
======================
    % Instructions: Perform a single gradient step on the parameter vector
    %                         theta.
    %
    % Hint: While debugging, it can be useful to print out the values
    %           of the cost function (computeCost) and gradient here.
    %









    %
==========================================================
=

    % Save the cost J in every iteration
    theta= theta - alpha*(X'*(X *theta - y))/m;
    J_history(iter) = computeCost(X, y, theta);
    i(iter)=iter;
end
    plot(i,J_history,'b-');
end
```

function computecost

```
function J = computeCost(X, y, theta)
```

```
%COMPUTECOST Compute cost for linear regression
%    J = COMPUTECOST(X, y, theta) computes the cost of using theta as the
%    parameter for linear regression to fit the data points in X and y

% Initialize some useful values
m = length(y); % number of training examples

% You need to return the following variables correctly
J = 0;

% ====================== YOUR CODE HERE ======================
% Instructions: Compute the cost of a particular choice of theta
%                       You should set J to the cost.
J=sum((X*theta - y).^2)/(2*m);




%
=============================================================
==============

End
```

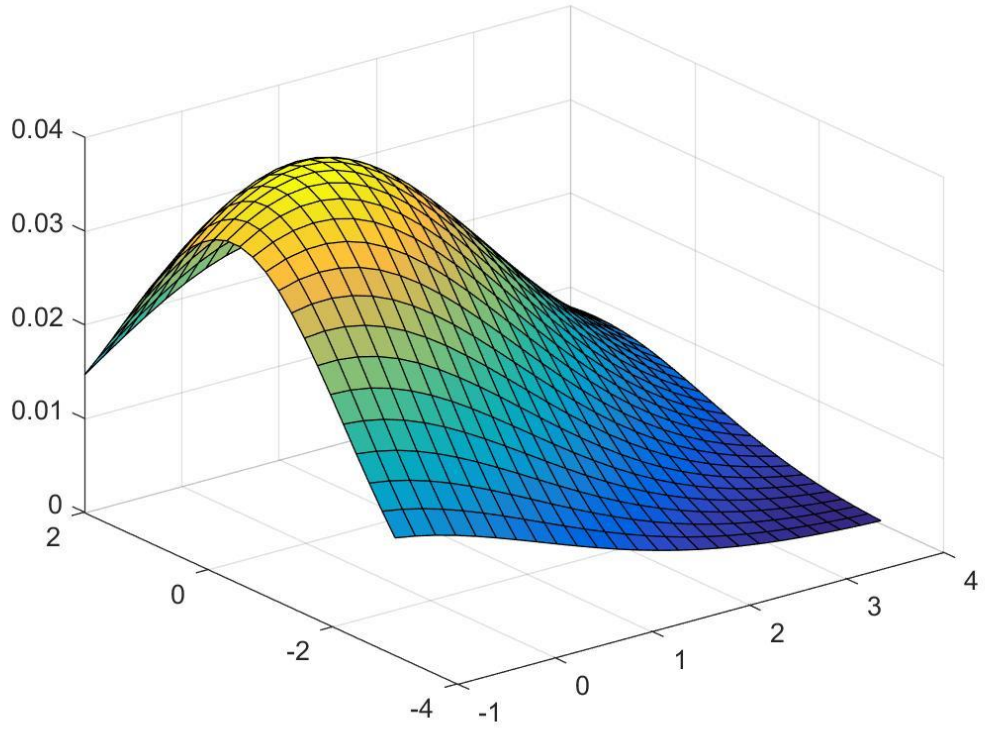


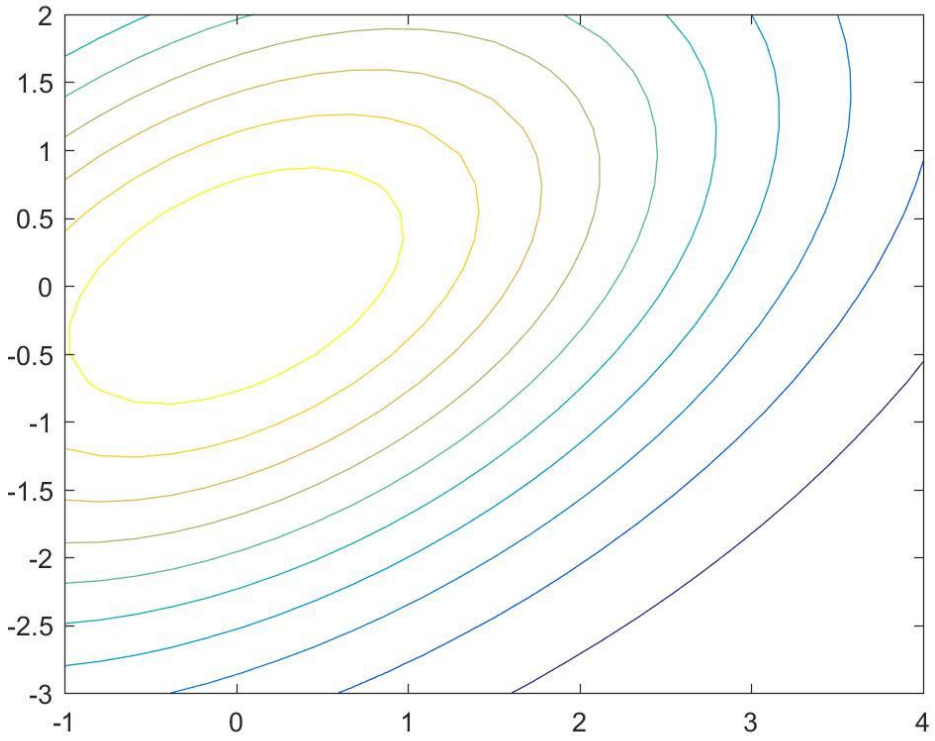
Task 2

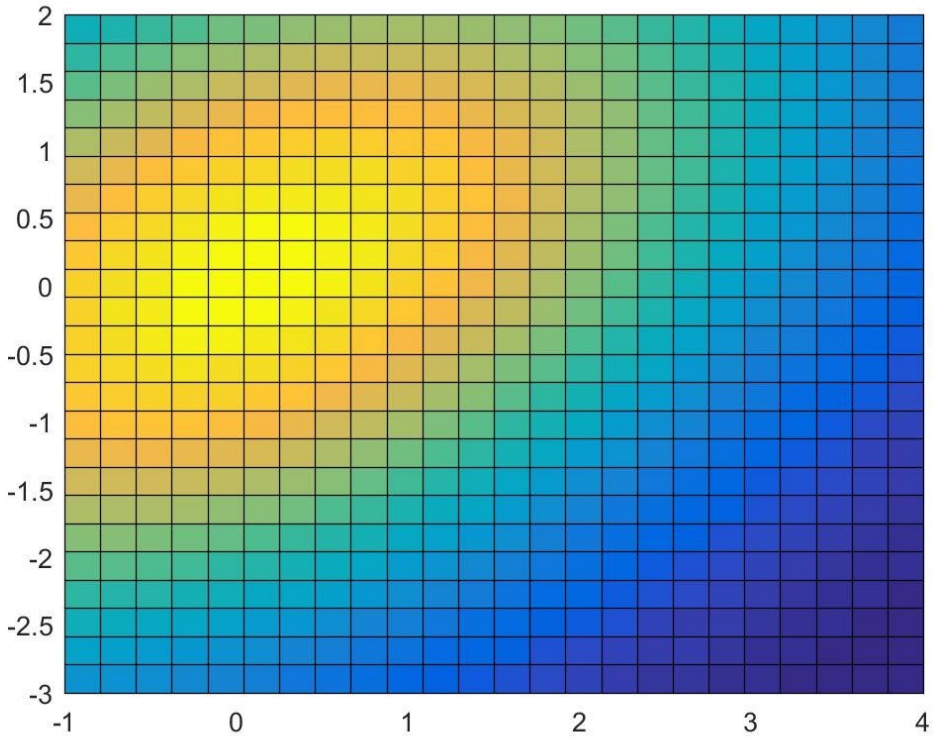
```
[X1,X2] = meshgrid(linspace(-1,4,25)', linspace(-3,2,25)');
X = [X1(:) X2(:)];
mu = [0 0]; sigma = [5 2;2 4];
p = Mvnpdf(X,mu,sigma);
surf(X1,X2,reshape(p,25,25));%三维图
figure (2);
contour(X1,X2,reshape(p,25,25),10);%等高线
figure(3);
pcolor(X1,X2,reshape(p,25,25));%色彩图
```

function mvnpdf

```
function [ p ] = Mvnpdf(X, mu,Sigma )
%UNTITLED3 Summary of this function goes here
%     Detailed explanation goes here
p = mvnpdf(X, mu, Sigma);

end
```


0.04
0.03
0.02
0.01
0
2
0
-2
-4
-1
0
1
2
3
4


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-3
-1
0
1
2
3
4


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-3
-1
0
1
2
3
4