实验一

**function A = warmUpExercise()**

返回5x5单位矩阵

代码：

A = eye(5,5);

**function plotData(x, y)**

作单变量数据的平面直角坐标图

代码：

xlabel('population size in 10,000s');

ylabel('profit in $10,000s');

plot(x, y, 'rx', 'MarkerSize', 10);

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

返回损失函数值

代码：

J = sum((X \* theta - y) .^ 2 ) / 2 / m;

**function [theta, J\_history] = gradientDescent(X, y, theta, alpha, num\_iters)**

梯度下降，迭代num\_iters次

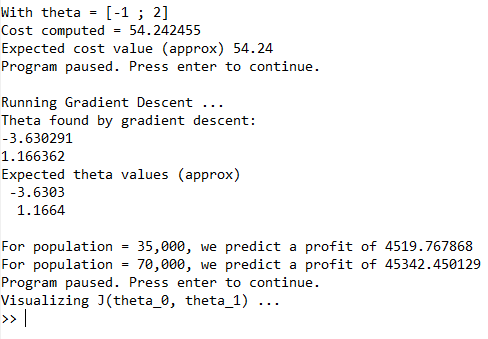
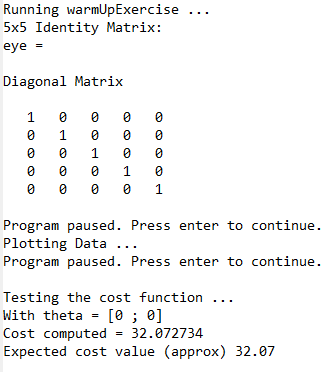
代码：

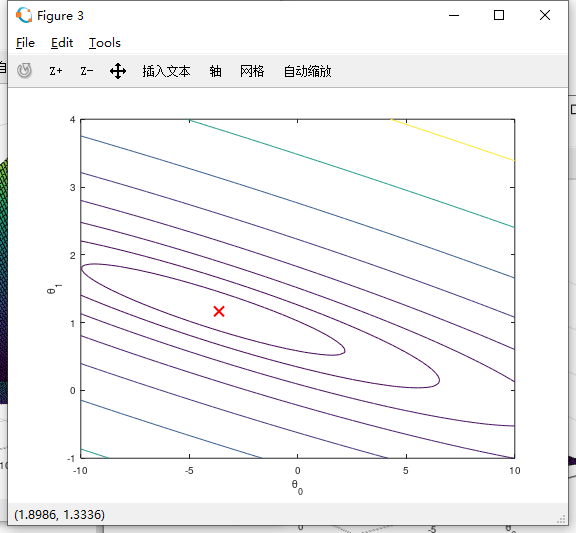
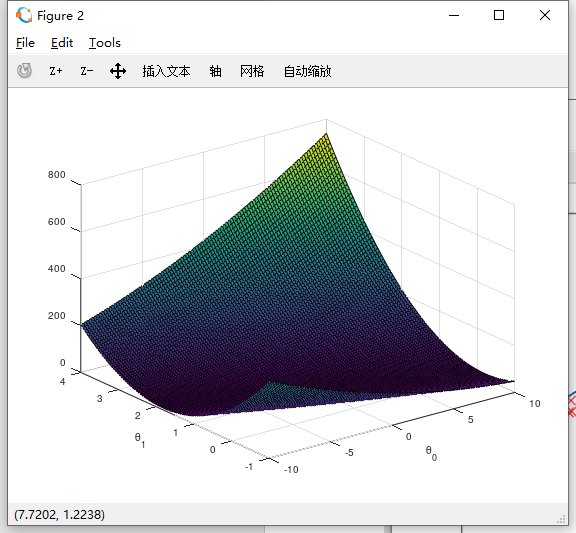
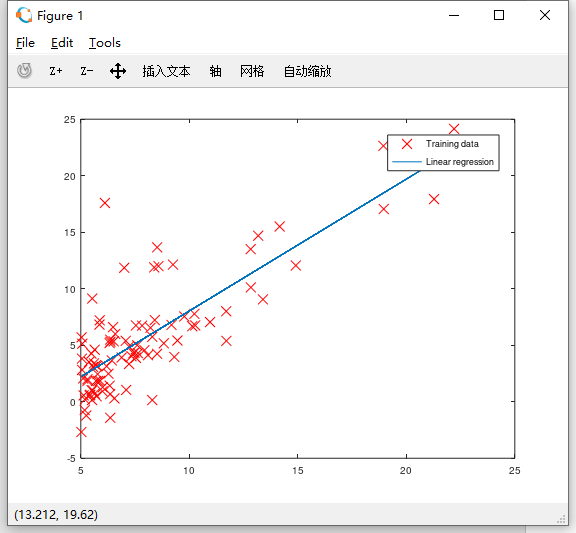
% (n+1)xm \* mx1 = (n+1)x1

% theta i = sum of j from 1 to m {x(j)i \* (x(j)\*theta - y(j))} devided by m

theta = theta - alpha / m \* X' \* (X \* theta - y);

**ex1运行结果**





**function J = computeCostMulti(X, y, theta)**

计算多元线性回归的损失函数值

代码：

同computeCost(X, y, theta)

**function [theta, J\_history] = gradientDescentMulti(X, y, theta, alpha, num\_iters)**

多元变量梯度下降，迭代num\_iters次

代码：

同gradientDescent(X, y, theta, alpha, num\_iters)

**function [X\_norm, mu, sigma] = featureNormalize(X)**

特征标准化

代码：

mu = mean(X); % (n+1)x1

sigma = std(X); % (n+1)x1

X\_norm = (X - mu) ./ sigma; % use broadcast mechanism

**function [theta] = normalEqn(X, y)**

正规方程

代码：

theta = pinv(X' \* X) \* X' \* y;

**ex1multi运行结果**

