FCI Fall 2013 Assignment III. 50 pts.
NAME(s): **Francisco Nardi** (16163892) and **Paulo Silva** (16164638)

Electronic submission on Blackboard is due latest by 11 pm on Monday, Oct 7th. Submissions received after the deadline will be graded only for effort for a maximum of 70% of the total grade (Refer to class syllabus for detailed grading policy). **State any assumptions you make, justify your answers, show intermediate steps and explain your results for maximum credit**. You may leave quantitative answers in the form of expressions unless a numeric value is required to address the question. All answers should be in your own words with any sources you refer to cited at the appropriate places. Any knowledge you acquire from the Internet should be written in your own words and be appropriately referenced. Copying and pasting from the Internet, each other or any other source will not count as your effort (Refer to class syllabus for detailed policy on plagiarism).
You may submit this assignment in groups of two each. Write your names on this sheet and include it as the cover page for your submission. You are encouraged to use MATLAB for this assignment. You may base your code on the samples provided on the textbook website (http://www.dcs.gla.ac.uk/~srogers/firstcourseml/); clearly indicate which sections of your answer are not original.

Q1. (10) Use all the data for any Olympic event to create a quadratic model with Gaussian noise for predicting Olympic gold medal winning times or distances as a function of calendar year.
(see http://www.databaseolympics.com/sport/sporteventlist.htm?sp=ATH for data)
(5) Create a graph that overlays the predicted winning times/distances on the observed data.
(5+5) Compare the loss calculated in two different ways: a) based on a noiseless prediction b) based on a noisy prediction.
(5) Add error bars to the line corresponding to your noiseless model

Q2. (20) Create 2D Gaussian plots (see pg 63) for the following scenarios: a) Independent random variables with unequal variance b) Dependent random variables with same variance c) the contour plot looks almost like a straight line and is parallel to an axis d) the contour plot looks almost like a straight line but is not parallel to either axis

Homework 3

Francisco Nardi - 16163892

Paulo Silva - 16164638

Exercise 3.1:

File used:

"400mHurdlesMen.txt"

Answer obtained:

"The mean loss for a noisy is 7.55208

Warning: Matrix is close to singular or badly scaled.

Results may be inaccurate. RCOND = 1.717229e-021.

> In hw31FranciscoPaulo at 84

Warning: Matrix is close to singular or badly scaled.

Results may be inaccurate. RCOND = 1.717229e-021.

> In hw31FranciscoPaulo at 88

The mean loss for a noiseless is 8.24456"

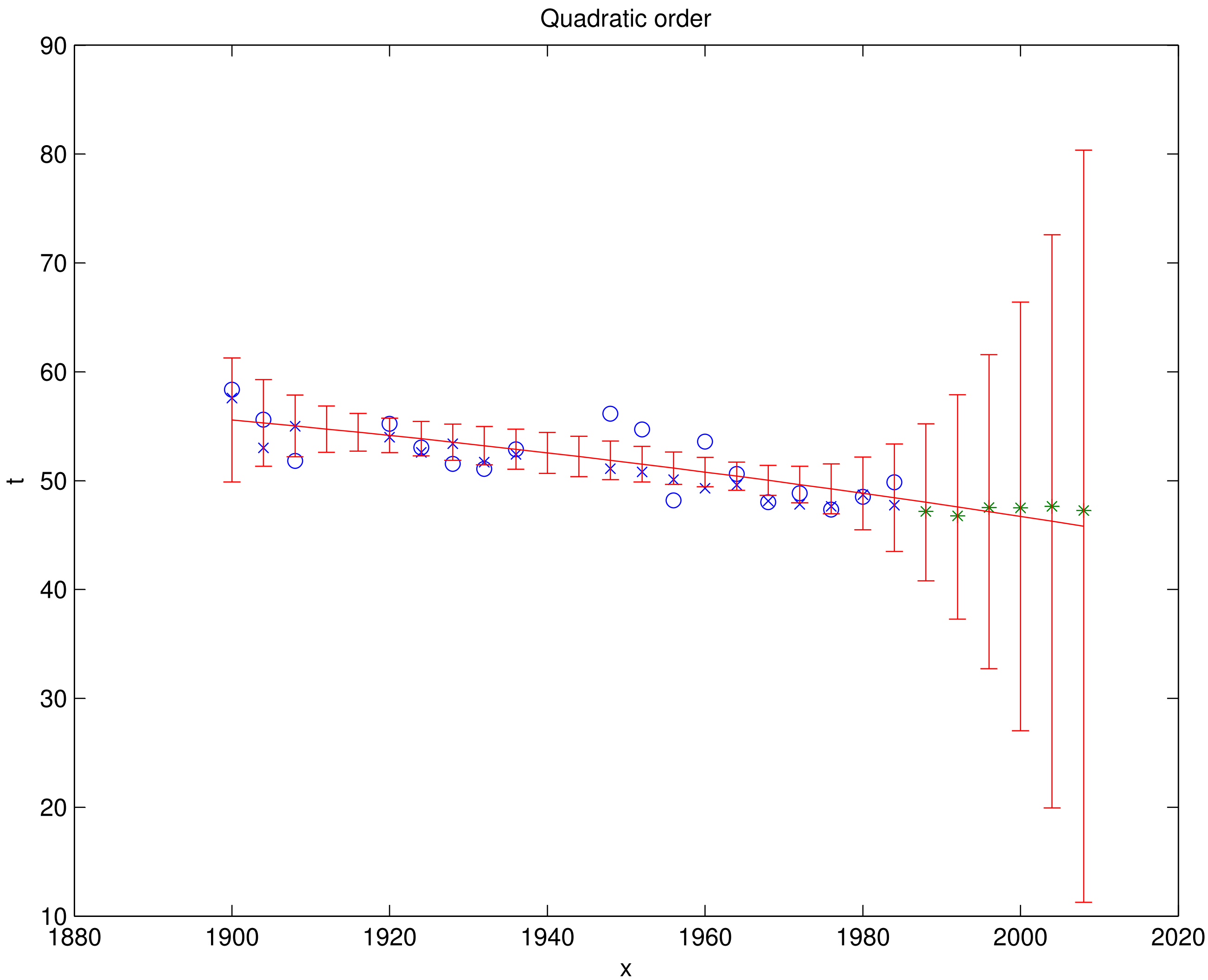
Exercise 3.2:

Answer obtained:

The four figures.

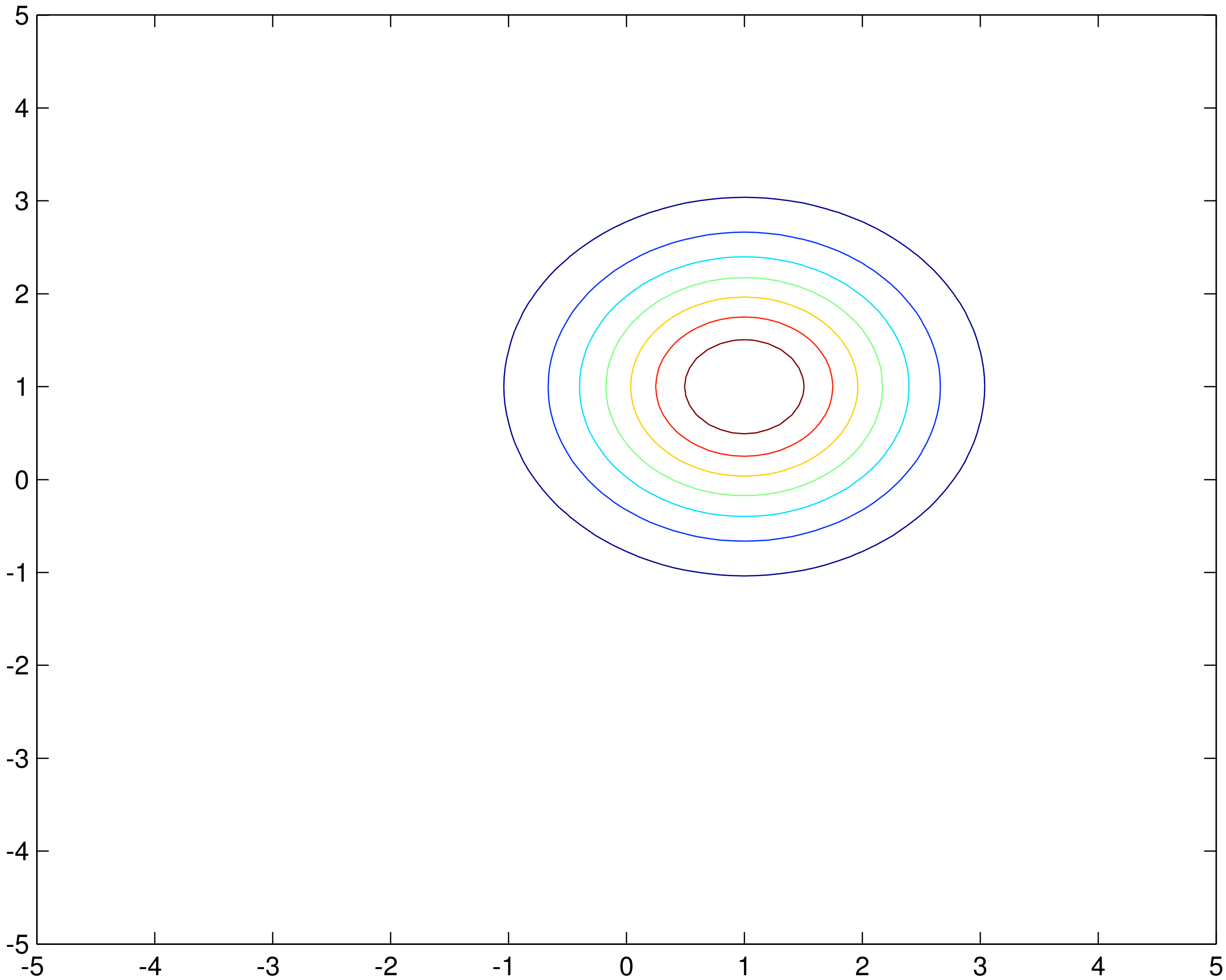
```
% Homework 3.1
% Francisco Nardi and Paulo Silva

% Cleaning the screen and variables as usual
clear all;
close all;

% Loading the files
filename = '400mHurdlesMen.txt';
data = importdata (filename);

% x receives the years, while t receives the winning times
x = data (: , 1);
t = data (: , 2);

% valid looks for x greater than 1986, which are N = 6
valid = find (x > 1986);
N = 6;

% valx has these x
% valt the related t
valx = x (valid: end);
valt = t (valid: end);

% it erases values of valx and valt from x and t, resp.
x (valid:end) = [];
t (valid:end) = [];

% the noise is set as 2
noise_var = 2;

% t2 will keep the original values t
t2 = t;

% adding noise to t
t = t + randn(size(x)).*sqrt(noise_var);

% It increases as the events happen,
% with space of 4 years
testx = [min(x):4:max(valx)]';

% The points marked with 'x' are the true values of the dataset
% The points marked with '*' are the predicted values
figure(1);
hold off
plot(x, t2,'x',valx, valt,'*');
xlabel('x');
ylabel('t');


% It draws the model
orders = [2:2];
for i = 1:length(orders)
    X = [];
    testX = [];
```

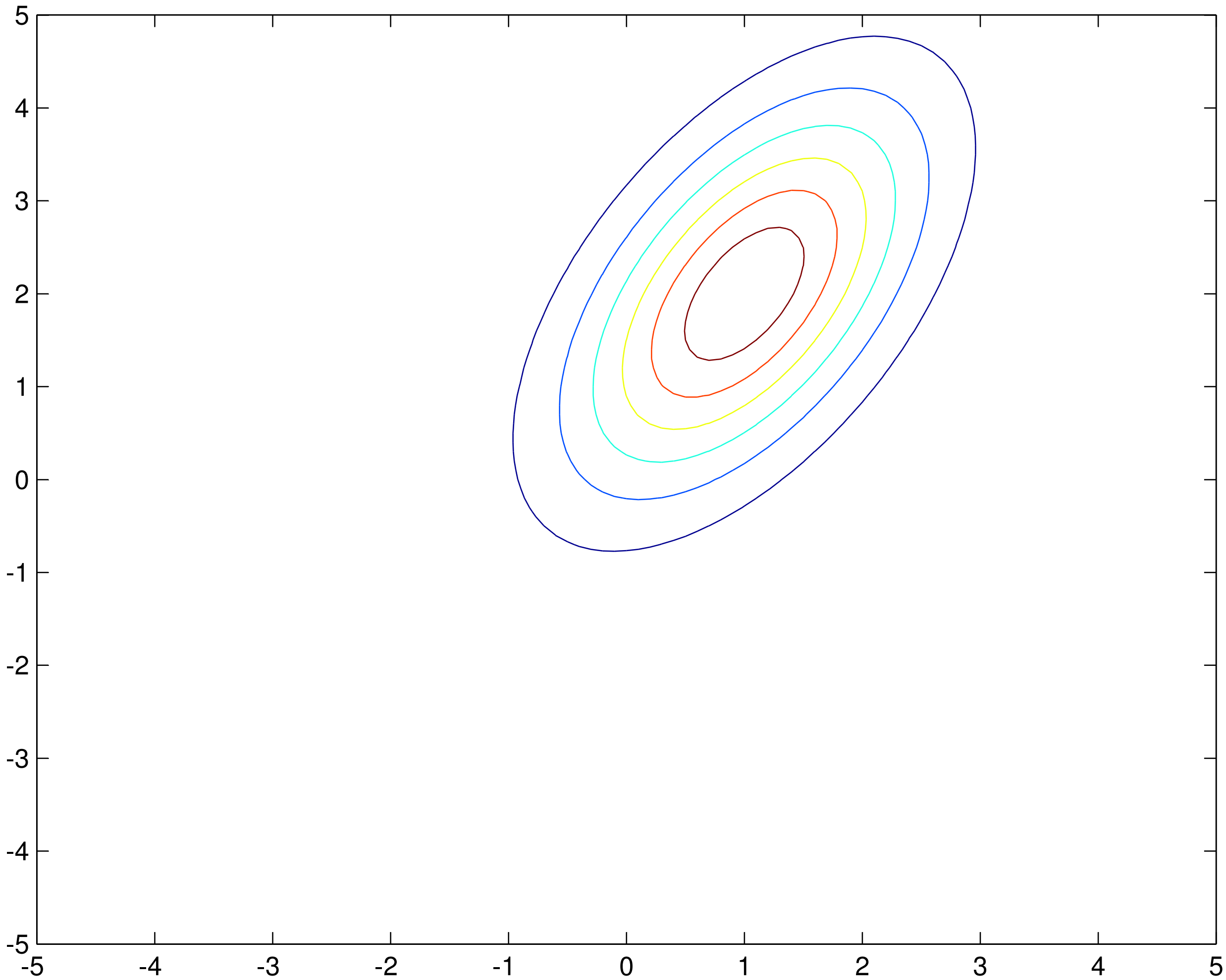
```
    for k = 0:orders(i)
        X = [X x.^k];
        testX = [testX testx.^k];
    end

    % It finds the w and covariance for the noisy function
    w = inv(X'*X)*X'*t;
    ss = (1/N)*(t'*t - t'*X*w);
    % It finds the mean and variance for the noisy function
    testmean = testX*w;
    testvar = ss * diag(testX*inv(X'*X)*testX');

    figure(1);
    hold on
    plot(x,t,'o');
    xlabel('x');
    ylabel('t');
    hold on
    errorbar(testx,testmean,testvar,'r')

    ti = sprintf('Quadratic order');
    title(ti);

    % loss returns the mean error for the noisy function
    loss = mean((testmean - testvar))/6;
    fprintf('\n\nThe mean loss for a noisy is %g\n\n', loss);

    % It finds the w and covariance for the noiseless function
    w2 = inv(X'*X)*X'*t2;
    ss2 = (1/N)*(t2'*t2 - t2'*X*w2);
    % It finds the mean and variance for the noiseless function
    testmean2 = testX*w2;
    testvar2 = ss2 * diag(testX*inv(X'*X)*testX');

    % loss returns the mean error for the noiseless function
    loss2 = mean((testmean2 - testvar2))/6;
    fprintf('\n\nThe mean loss for a noiseless is %g\n\n', loss2);

end
```



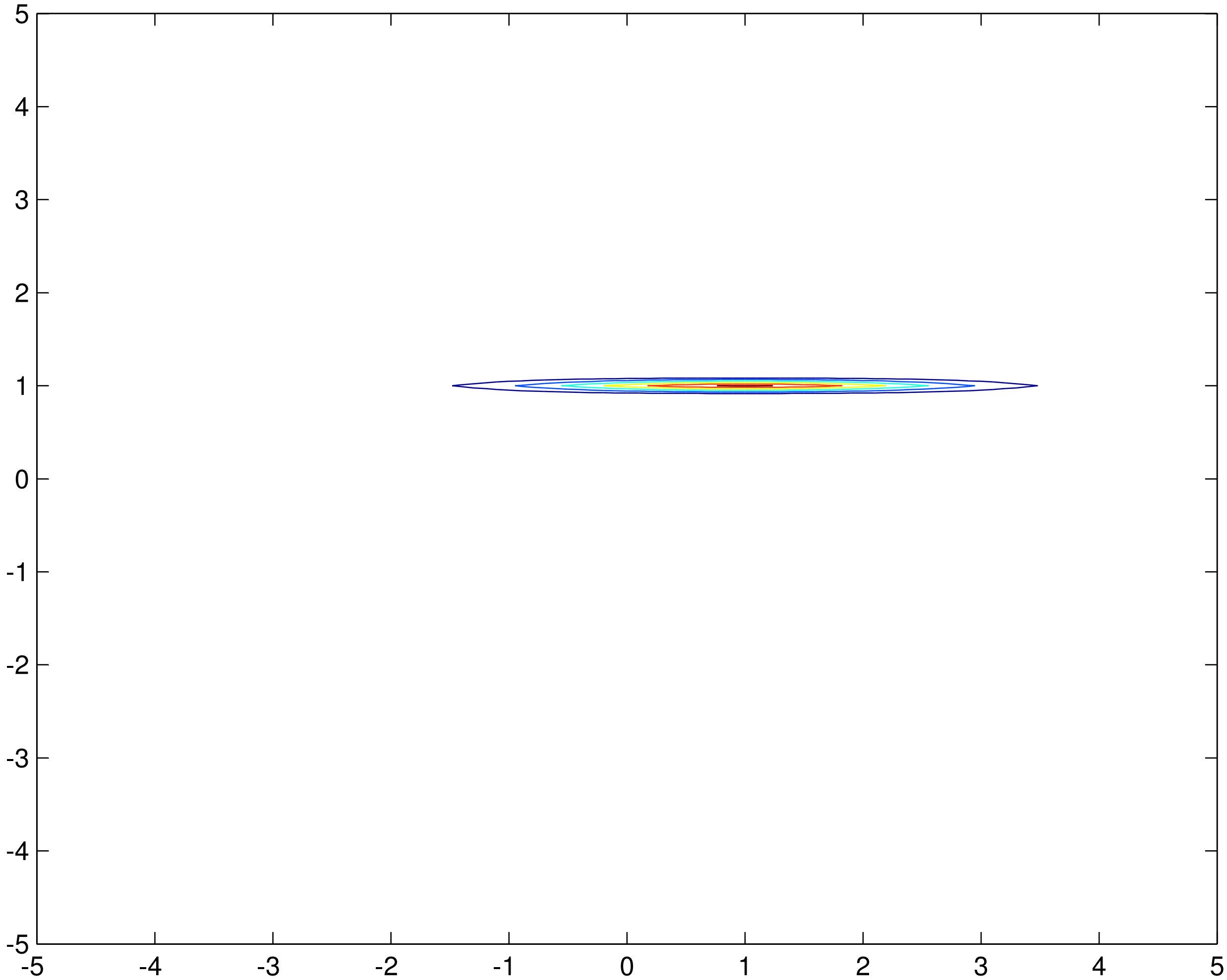
Quadratic order
90
80
70
60
50
40
30
20
10
t
1880
1900
1920
1940
1960
1980
2000
2020
x

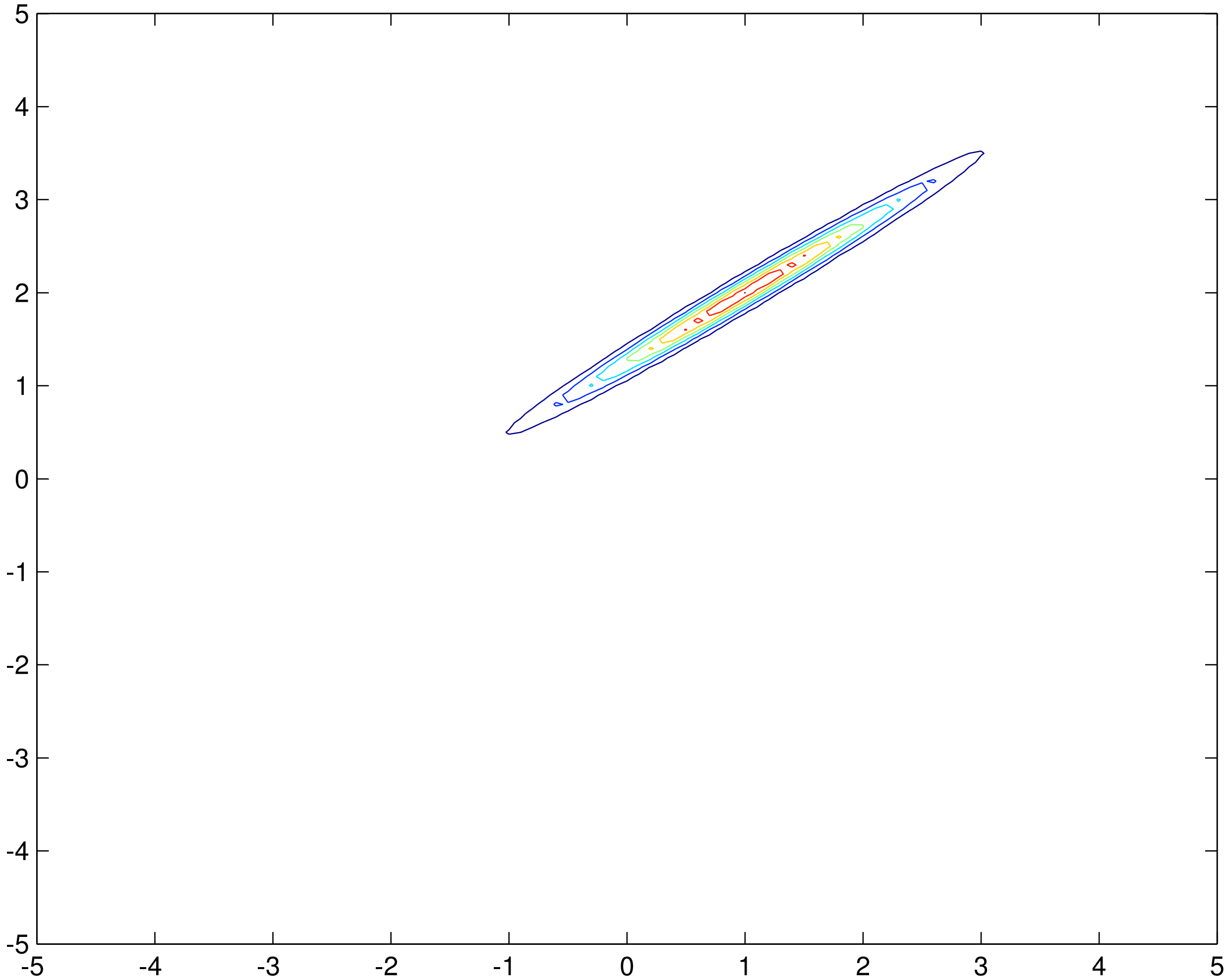
```
% Homework 3.2
% Francisco Nardi e Paulo Silva

%As usual, it is need to clean the screen and variables
clear all;
close all;

%Here we will fill the values of mu and sigma according to the draw we want

%Independent random variables with unequal variance
mu1 = [1;1];
sigma1 = [1 0;0 1];

%Here we call the function plotGauss
plotGauss(mu1,sigma1);

%Dependent random variables with same variance
mu2 = [1;2];
sigma2 = [1 0.8;0.8 2];

%Here we call the function plotGauss
plotGauss(mu2,sigma2);

%The contour plot looks almost like a straight line and is parallel
% to the x axis
mu3 = [1;1];
sigma3 = [1.7 0; 0 0.0001];

%Here we call the function plotGauss
plotGauss(mu3,sigma3);

%The contour plot looks almost like a straight line but is
% not parallel to either axis
mu4 = [1;2];
sigma4 = [1.07 0.8;0.8 0.61];

%Here we call the function plotGauss
plotGauss(mu4,sigma4);
```


5
4
3
2
1
0
-1
-2
-3
-4
-5
-5
-4
-3
-2
-1
0
1
2
3
4
5



5
4
3
2
1
0
-1
-2
-3
-4
-5
-5 -4 -3 -2 -1 0 1 2 3 4 5


5
4
3
2
1
0
-1
-2
-3
-4
-5
-5
-4
-3
-2
-1
0
1
2
3
4
5



5
4
3
2
1
0
-1
-2
-3
-4
-5
-5
-4
-3
-2
-1
0
1
2
3
4
5