1) Write a function to compute the 7 **similitude** moment shape descriptors. Test and compare results on the rectangle box images 'boxIm[1-4].bmp' on the website (provide the computed moment values). Normalize each image before computing the moments so that the range of grayscale values is between 0-1. How do the moments change across the box images? Why are some moments zero? Please make sure your function will work with non-binary (grayscale) imagery, as you will need this for later assignments (do not use Matlab's regionprops function). [4 pts]

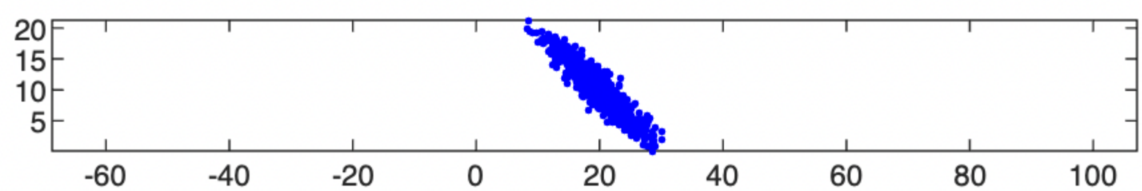
BoxIm1	[0.0422	0	0	0	0.1646	0	0]
BoxIm2	[0.0422	0	0	0	0.1646	0	0]
BoxIm3	[0.0423	0	0	0	0.1641	0	0]
BoxIm4	[0.1646	0	0	0	0.0422	0	0]

I used im2double to normalize the image.

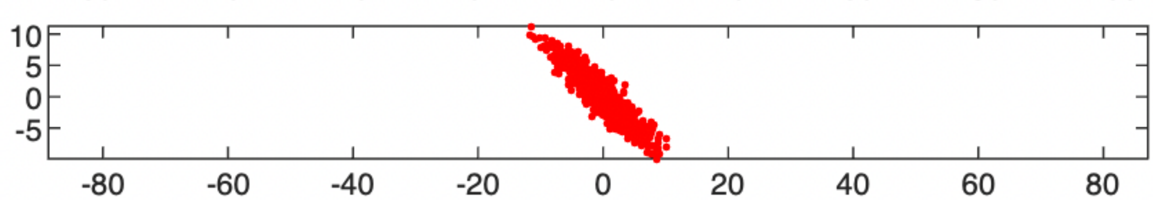
BoxIm1 and BoxIm2 has the same moment values. The moment value of BoxIm3 changed a little. But the moment values of BoxIm4 changed a lot.

I think pq value leads to some moments to be zero

2) Using the datafile (eigdata.txt) provided on the WWW site, perform the following MATLAB commands [1 pt]:

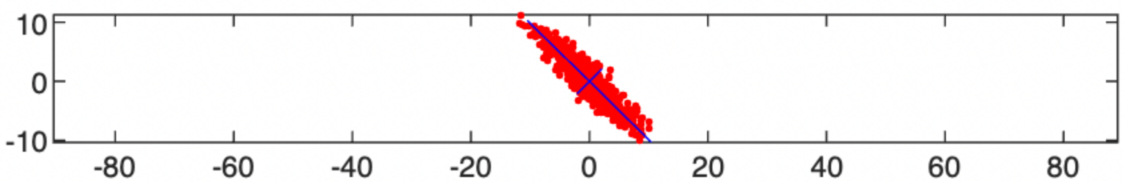


Plot for data points in the eigdata.txt.



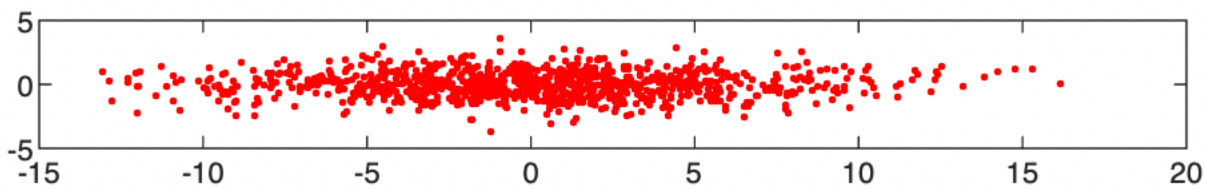
Plot for the mean-subtract data points.

3) Compute the eigenvalues (V) and eigenvectors (U) of the data (stored in Y) using the function eig() in Matlab (recall that you use either the covariance matrix or the inverse-covariance matrix of the data – see class notes). Plot the mean-subtracted data Y and the 2-D Gaussian ellipse axes for given the eigenvectors in U (you can use the plot command in Matlab for this. Make sure the axes have equal scale in the plot!). Use the eigenvalues in V to give the appropriate 3σ (standard deviation - not variance!) length to each axis (did you compute the eigenvalues from the covariance or inverse covariance of Y? The eigenvalues will be related but different! See class notes). [4 pts]



C should be set as 9 since we use 3σ. After get the points, use [0,0] to draw axes.

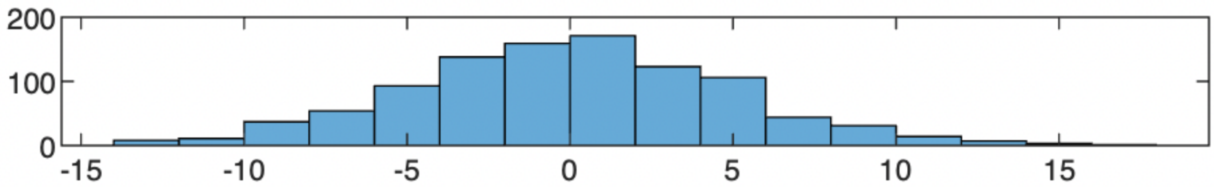
4) Rotate Y using the eigenvectors to make the data uncorrelated (i.e., project data Y onto the eigenvectors – see class slides). Plot the results (using equal scale axes as before). [2 pts]



I transpose Y when doing the calculation and then change the value back to the original format. The plot became horizontal.

5) Perform a simple data reduction technique by keeping only the values resulting from projection of Y onto the eigenvector corresponding the largest eigenvalue of the covariance (not inverse-

covariance) matrix. Plot a 1-D histogram of the values. Does it look like a 1-D Gaussian? [1 pt]



Yes! Because it is the reduction of 2-D Gaussian.

6) Submit a report containing all code, printouts of images, and discussion of results. Make a script to do the above tasks and call needed functions. Upload your report, code, and images to Carmen as usual. [No free points for this last step anymore!]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Problem 1

%similitude moments
function n_pq =  spatial_central_moments(image, p, q)
    m00 = sum(sum(image));
    [y,x] = size(image);

    m10 = 0;
    m01 = 0;
    for row = 1 :y
        for column=1:x
            m10 = m10 + column * image(row,column);
            m01 = m01 + row * image(row,column);

        end
    end
    x_bar = m10/m00;
    y_bar = m01/m00;

    u_pq = 0;
    for row = 1 :y
        for column=1:x
            u_pq = u_pq + ((column - x_bar)^p) * ((row - y_bar)^q) *
image(row,column);
        end
    n_pq = u_pq / (m00 ^ (1+ ((p+q)/2)));
    end
end
function Nvals =  similitudeMoments(image)

       Nvals = [spatial_central_moments(image,0, 2),
spatial_central_moments(image, 0, 3),...
           spatial_central_moments(image, 1, 1),
spatial_central_moments(image, 1, 2),...
```

```
            spatial_central_moments(image, 2, 0),
spatial_central_moments(image, 2, 1),...
            spatial_central_moments(image, 3, 0)];
end


Im1 = im2double(imread('boxIm1.bmp'));
NvalsIm1 = similitudeMoments(Im1);
disp(NvalsIm1);

Im2 = im2double(imread('boxIm2.bmp'));
NvalsIm2 = similitudeMoments(Im2);
disp(NvalsIm2);

Im3 = im2double(imread('boxIm3.bmp'));
NvalsIm3 = similitudeMoments(Im3);
disp(NvalsIm3);

Im4 = im2double(imread('boxIm4.bmp'));
NvalsIm4 = similitudeMoments(Im4);
disp(NvalsIm4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Problem 2

%% Load the data
clear; close all;
load eigdata.txt;
X = eigdata;
subplot(5,1,1);
plot(X(:,1),X(:,2),'b.');
axis('equal');
%% mean-subtract data
m = mean(X);
Y = X - ones(size(X,1),1)*m;
subplot(5,1,2);
plot(Y(:,1),Y(:,2),'r.');
axis('equal');


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Problem 3
cov_Y = cov(Y);
[U,V] = eig(cov_Y);
C = 9;
x_axis = U(:,1) * (sqrt(C *V(1,1)));
y_axis = U(:,2) * (sqrt(C *V(2,2)));
subplot(5,1,3);
plot(Y(:,1),Y(:,2),'r.');
hold on
plot([0,x_axis(1)],[0,x_axis(2)] , 'b');
plot([0,y_axis(1)],[0,y_axis(2)], 'b');
plot([0,-x_axis(1)],[0,-x_axis(2)] , 'b');
plot([0,-y_axis(1)],[0,-y_axis(2)], 'b');
axis('equal');
hold off

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%Problem 4

subplot(5,1,4);
Y = U' * Y';
Y = fliplr(Y');
plot(Y(:,1),Y(:,2),'r.');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Problem 5
subplot(5,1,5);
h = histogram(Y(:,1));
```