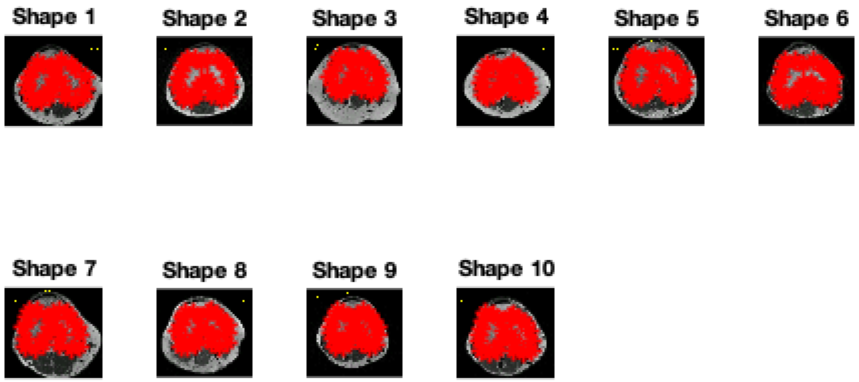
Table of Contents

```
%function Femur Categorization

% close all the opened windows
close all
clear all
% initialization of variables

N = 10;   % ...
p = 24;   % ...
d=2;      % ...
dp = d*p; % ...
```

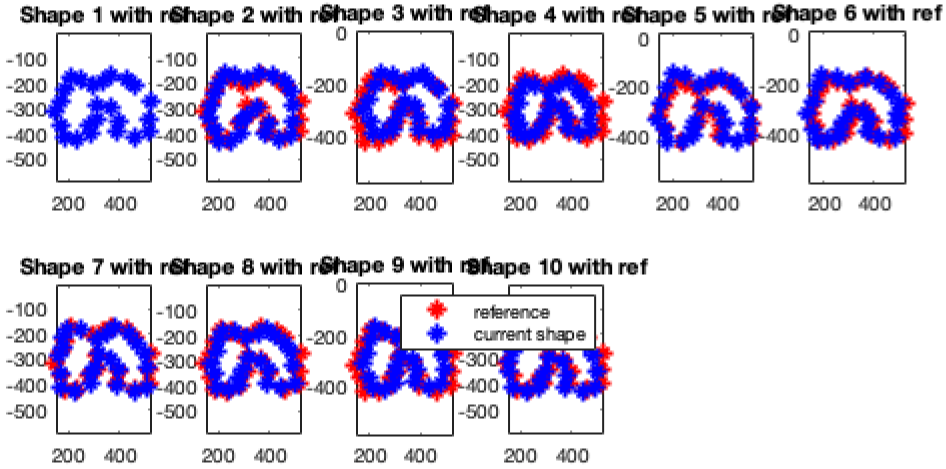
** 1.

```
figure;
shapeLandmarks = zeros(dp,N);

dirName = './FemurImages/';              %# folder path
files = dir( fullfile(dirName,'*.png') );   %# list all *.xyz files

files_table =struct2table(files);
files_table = [files_table(2:10,:); files_table(1,:)];
files = table2struct(files_table);

for i = 1:N
    % load images
    [pth,FileName,ext] = fileparts(files(i).name);
    shape = imread(['./FemurImages/', FileName,'.png']);
    % make landmarks
    %landmarkPickingFemur([dir, FileName, ext]);
    % load landmarks
    shapeLandmarks(:,i) = load(['./FemurImages/',
 FileName, '_coords.txt']);
    % visualize
    subplot(3,6,i), imshow(shape), axis off; hold on;
 plot(shapeLandmarks(1:dp/2,i), shapeLandmarks(dp/2+1:dp,i),'*r');
    title(['Shape ' num2str(i)])
end

% Q1: Landmarks are the locations that represent unique features that
% relate the features to something, like a shape.
% Figure visualizes landmarks on image.
```



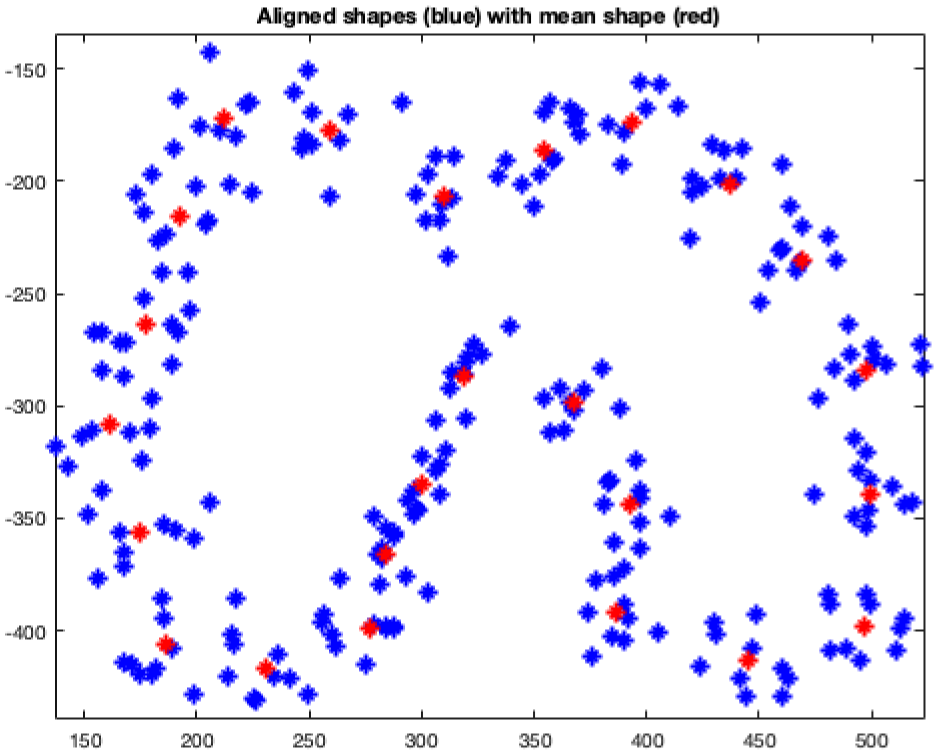
** 2.

```
% align the shapes
alignedShapeLandmarks = zeros(dp,N);

alignedShapeLandmarks(:,1) = shapeLandmarks(:,1);
for i = 2:N
    [t, newCoords] = procrustes2(shapeLandmarks(:,1),
 shapeLandmarks(:,i));
    alignedShapeLandmarks(:,i) = newCoords;
end

figure;
for i = 1:N
    subplot(3,6,i),plot(alignedShapeLandmarks(1:dp/2,1), -
alignedShapeLandmarks(dp/2+1:end,1), '*r'), hold on,
    subplot(3,6,i),plot(alignedShapeLandmarks(1:dp/2,i), -
alignedShapeLandmarks(dp/2+1:end,i), '*b'), hold on, axis equal;
    title(['Shape ' num2str(i) ' with ref'])
end
legend ('reference', 'current shape')


%
```



** 3. **

```
meanShape = mean(alignedShapeLandmarks, 2);

figure
for i = 1:N
    plot(alignedShapeLandmarks(1:dp/2,i), -
alignedShapeLandmarks(dp/2+1:end,i), '*b'), hold on, axis equal
end
plot(meanShape(1:dp/2), -meanShape(dp/2+1:end), '*r'), hold on,
 axis equal
title('Aligned shapes (blue) with mean shape (red)')
```

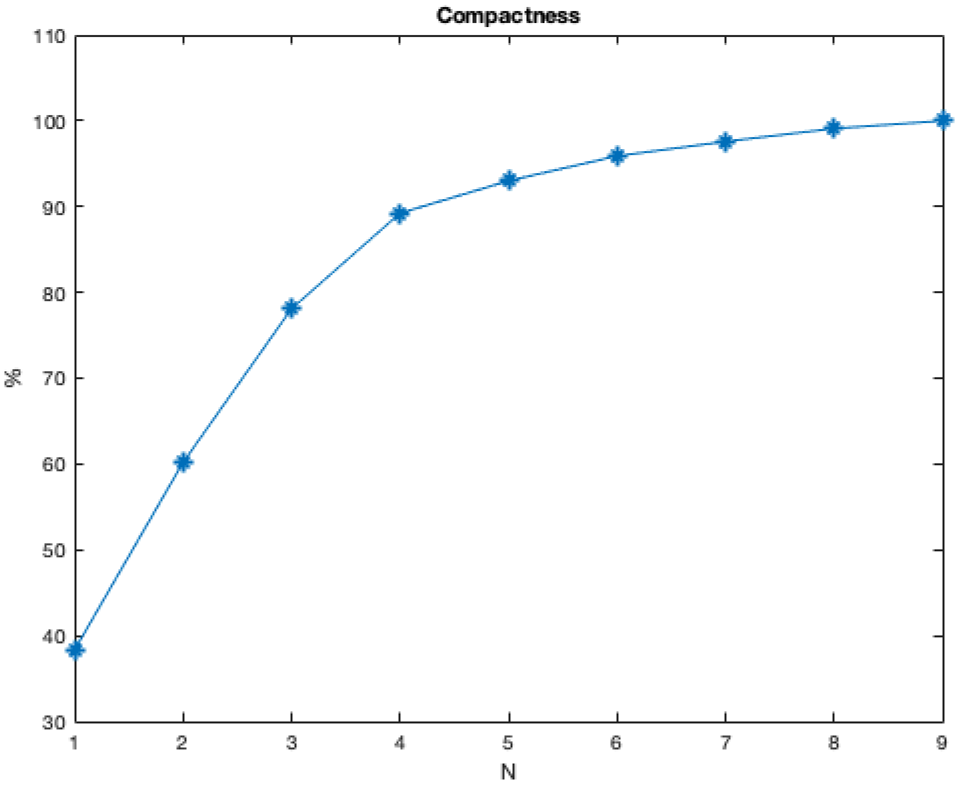


--- Q: complete the sentence below

subtract mean distance to the origin

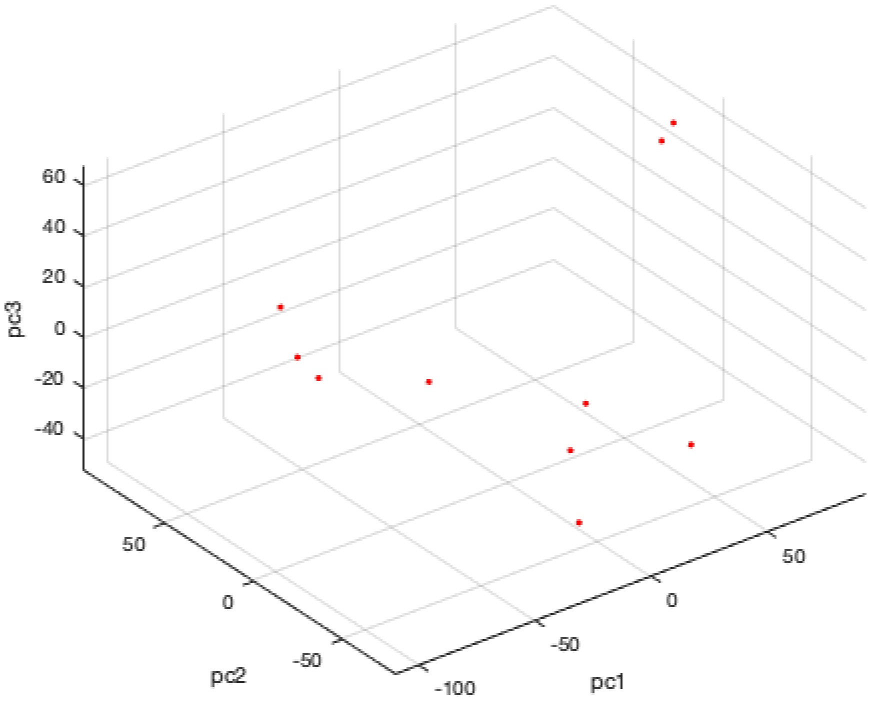
```
mean_matrix = repmat(meanShape,1,N);
centeredShapeLandmarks = alignedShapeLandmarks - mean_matrix;


C = 1/(N-1) * (centeredShapeLandmarks * centeredShapeLandmarks');

[U L UT] = svd(C);

evectors = U(:,1:N-1);
evalues = diag(L);
evalues = evalues (1:N-1);


evaluesNormalized = evalues ./ sum(evalues) *100;
compactness (1) = evaluesNormalized (1);
for i = 2:length(evaluesNormalized)
    compactness(i) = compactness(i-1) + evaluesNormalized (i);
end
figure, plot(compactness,'-*'), title('Compactness'), xlabel('N'),
 ylabel('%')
disp (['Compactness: ' num2str(compactness)])
```

```
Compactness: 38.31646        60.1698       78.09306       89.23142
 93.00263       95.89387       97.58596       99.09866           100
```

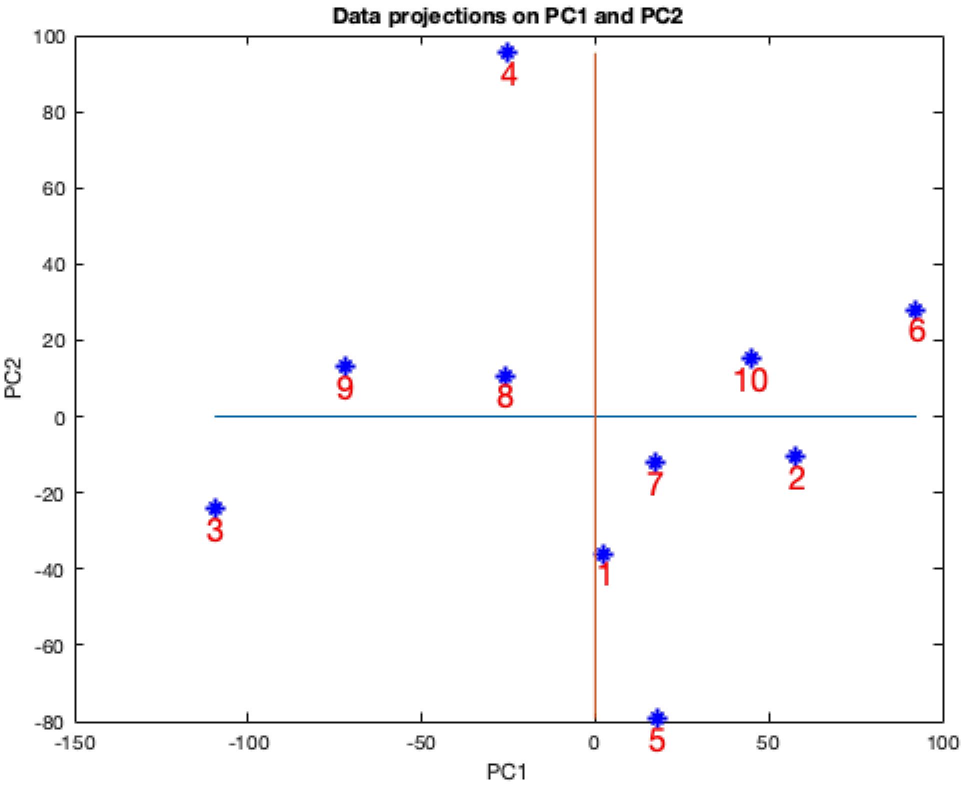


B. Project data on PC1, PC2 and PC3

```
for i = 1:N
    % projecting
    projectedPC1(i) = dot(centeredShapeLandmarks(:,i), evectors(:,1));
    projectedPC2(i) = dot(centeredShapeLandmarks(:,i), evectors(:,2));
    projectedPC3(i) = dot(centeredShapeLandmarks(:,i), evectors(:,3));
end


figure;
plot3(projectedPC1,projectedPC2,projectedPC3,'.r');
grid on;
axis equal;
xlabel('pc1');
ylabel('pc2');
zlabel('pc3');
% train perceptron
```

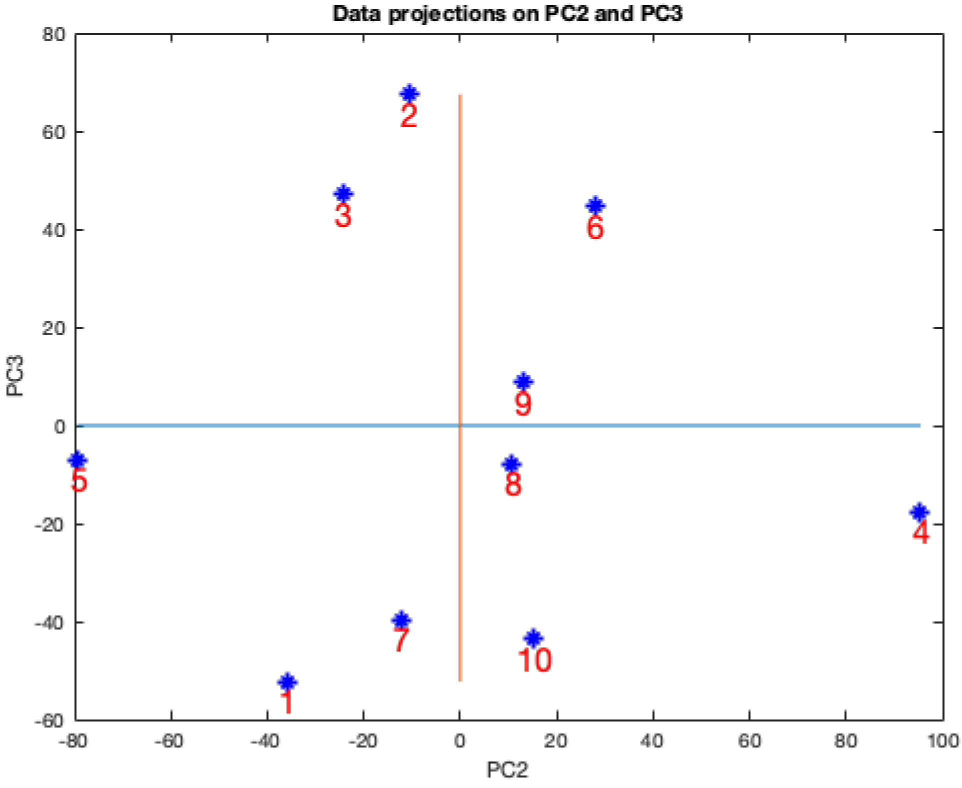


```
figure
for i = 1:N
    plot(projectedPC1(i),projectedPC2(i), '*b'); hold on
    h1 =
 text(projectedPC1(i),projectedPC2(i),num2str(i), 'HorizontalAlignment','center','
 [1 0 0], 'FontSize',16);
end
title ('Data projections on PC1 and PC2'), xlabel('PC1'),
 ylabel('PC2')
% axis though the origin
max_PC1 = max(projectedPC1);
min_PC1 = min(projectedPC1);
max_PC2 = max(projectedPC2);
min_PC2 = min(projectedPC2);
plot ([min_PC1 max_PC1], [0 0]); hold on;
plot ([0 0],[min_PC2 max_PC2]);
```



```
figure
for i = 1:N
    plot(projectedPC2(i),projectedPC3(i), '*b'); hold on
    h1 =
 text(projectedPC2(i),projectedPC3(i),num2str(i), 'HorizontalAlignment','center','
 [1 0 0], 'FontSize',16);
end
title ('Data projections on PC2 and PC3'), xlabel('PC2'),
 ylabel('PC3')
% axis though the origin
max_PC3 = max(projectedPC3);
min_PC3 = min(projectedPC3);
plot ([min_PC2 max_PC2], [0 0]); hold on;
plot ([0 0],[min_PC3 max_PC3]);


% ** C. Create new instances at -3 and +3

% bring mean to the origin
x_mean = mean(meanShape(1:dp/2));
y_mean = mean(meanShape(dp/2+1:dp));
meanShapeCentered(1:dp,1) = meanShape(1:dp) - x_mean;
meanShapeCentered(dp/2+1:dp,1) = meanShape(dp/2+1:dp) - y_mean;

% weights
% -3 and + 3 because we move 3 stdev from the mean (99.7%)
```

```
wLeft = -3;
wRight = +3;
```



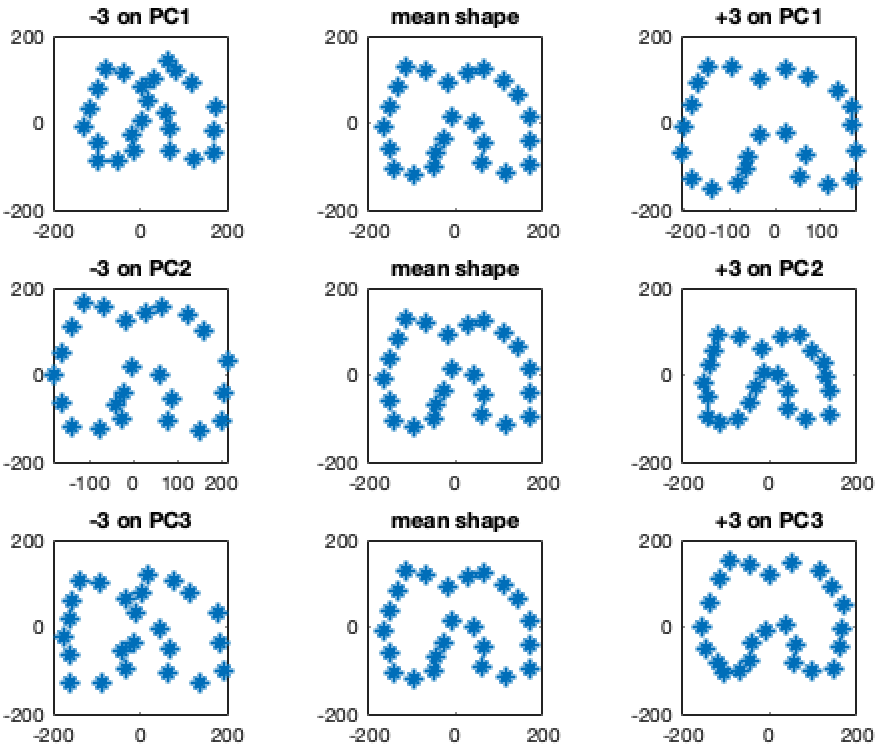
--- Q: describe the formula below

The formula subtracts and adds 3 standard deviations from the mean of each principle component based on its eigenvalues.

```
shapeLeft  = meanShapeCentered +
 evectors(:,1)*((-3)*sqrt(evalues(1)));
shapeRight = meanShapeCentered +
 evectors(:,1)*((+3)*sqrt(evalues(1)));
figure
subplot(3,3,1), plot(shapeLeft(1:dp/2), -shapeLeft(dp/2+1:dp), '*'),
 hold on
                axis equal, axis square, title ('-3 on PC1')
                ylim([-200, 200]);
subplot(3,3,2), plot(meanShapeCentered(1:dp/2), -
meanShapeCentered(dp/2+1:dp), '*'), hold on
                axis equal, axis square, title ('mean shape')
                ylim([-200, 200]);
subplot(3,3,3), plot(shapeRight(1:dp/2), -shapeRight(dp/2+1:dp), '*'),
 hold on
                axis equal, axis square, title ('+3 on PC1')
                ylim([-200, 200]);
```

```
% shape variation on PC2
shapeLeft  = meanShapeCentered +
 evectors(:,2)*((-3)*sqrt(evalues(2)));
shapeRight = meanShapeCentered +
 evectors(:,2)*((+3)*sqrt(evalues(2)));
subplot(3,3,4), plot(shapeLeft(1:dp/2), -shapeLeft(dp/2+1:dp), '*'),
 hold on
                axis equal, axis square, title ('-3 on PC2')
                ylim([-200, 200]);
subplot(3,3,5), plot(meanShapeCentered(1:dp/2), -
meanShapeCentered(dp/2+1:dp), '*'), hold on
                axis equal, axis square, title ('mean shape')
                ylim([-200, 200]);
subplot(3,3,6), plot(shapeRight(1:dp/2), -shapeRight(dp/2+1:dp), '*'),
 hold on
                axis equal, axis square, title ('+3 on PC2')
                ylim([-200, 200]);
ylim([-200, 200]);
% shape variation on PC3
shapeLeft  = meanShapeCentered +
 evectors(:,3)*((-3)*sqrt(evalues(3)));
shapeRight = meanShapeCentered +
 evectors(:,3)*((+3)*sqrt(evalues(3)));
subplot(3,3,7), plot(shapeLeft(1:dp/2), -shapeLeft(dp/2+1:dp), '*'),
 hold on
                axis equal, axis square, title ('-3 on PC3')
                ylim([-200, 200]);
subplot(3,3,8), plot(meanShapeCentered(1:dp/2), -
meanShapeCentered(dp/2+1:dp), '*'), hold on
                axis equal, axis square, title ('mean shape')
                ylim([-200, 200]);
subplot(3,3,9), plot(shapeRight(1:dp/2), -shapeRight(dp/2+1:dp), '*'),
 hold on
                axis equal, axis square, title ('+3 on PC3')
                ylim([-200, 200]);
% Q2:
% PCA's main goal is to reduce dimensionality to better characterize
% variance and covariance through a linear combination of these
 parameters.
% PCA is calculated by first making a covariance matrix in both
 directions
% by taking the mean of the landmarks and the transpose of the
 landmarks
% Then single variable decomposition finds the unit vector as the
% eigenvector the values of diagonalizing the matrix to find
 eigenvectors.
% The output of PCA analysis is your data projected onto the principle
% components, new feature axes.
% PC1 blows up the image when increased and shrinks when decreased.
% PC2 compresses vertically when increased and stretches vertically
 when
% decreased.
% PC3 stretches the image horizontally when decreased and compresses
 it
```

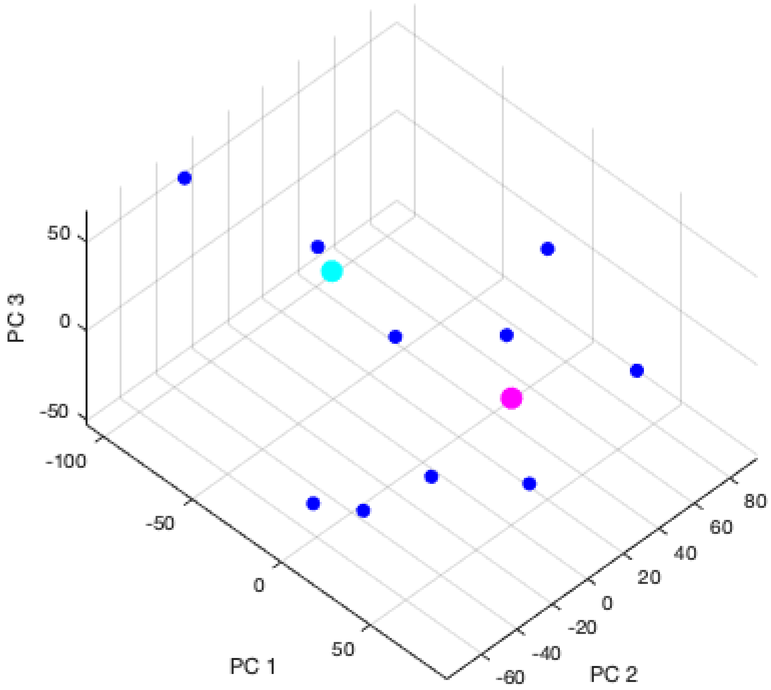
```
% when increased.
```



use K means to define 2 clusters the data

```
 close all
Project =  [projectedPC1;projectedPC2;projectedPC3]';
statrMatrix = [];
k = 2;

for i=1:k
        statrMatrix(i,1) = min(Project(:,1)) + (max(Project(:,1))-
min(Project(:,1)))*(i*2-1)/(k*2);
        statrMatrix(i,2) = min(Project(:,2)) + (max(Project(:,2))-
min(Project(:,2)))*1/2;
        statrMatrix(i,3) = min(Project(:,3)) + (max(Project(:,3))-
min(Project(:,3)))*1/2;
end

plot3(Project(:,1),Project(:,2),Project(:,3),'.b','MarkerSize',20);
xlabel('PC 1');
ylabel('PC 2');
zlabel('PC 3');
grid on
hold on
axis equal;
view(45,45)
```

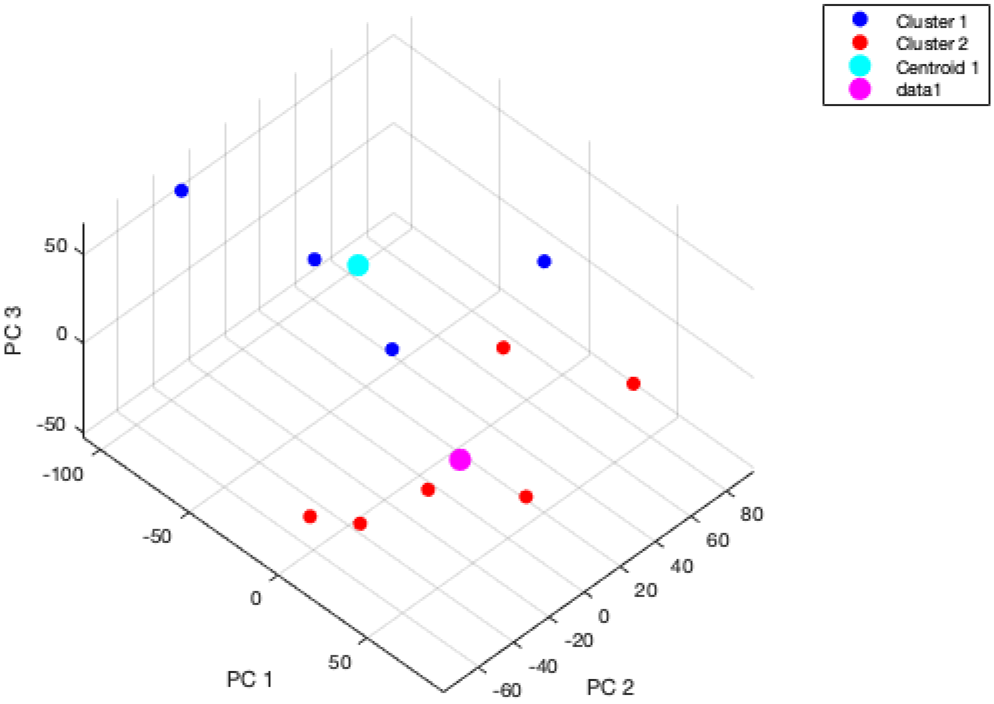
```
plot3(statrMatrix(1,1),statrMatrix(1,2),statrMatrix(1,3),'.c','MarkerSize',30)
plot3(statrMatrix(2,1),statrMatrix(2,2),statrMatrix(2,3),'.m','MarkerSize',30)
```



```
[idx,c] = kmeans(Project,k,'start',statrMatrix);


figure;
hold on
xlabel('PC 1');
ylabel('PC 2');
zlabel('PC 3');
grid on
tic

    idx1 = find(idx == 1);

 plot3(Project(idx1,1),Project(idx1,2),Project(idx1,3),'.b','MarkerSize',20)
    idx2 = find(idx == 2);

 plot3(Project(idx2,1),Project(idx2,2),Project(idx2,3),'.r','MarkerSize',20)
    view(45,45)
toc
axis equal;
hold on;

% plot the centroid of the 1st class
```

```
plot3(c(1,1),c(1,2),c(1,3),'.c','MarkerSize',30);
legend('Cluster 1','Cluster 2','Centroid 1')
plot3(c(2,1),c(2,2),c(2,3),'.m','MarkerSize',30);
```

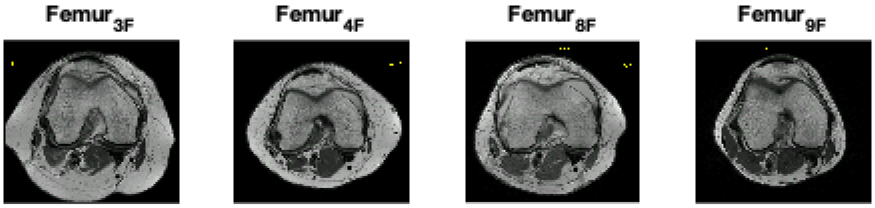
```
Elapsed time is 0.007375 seconds.
```

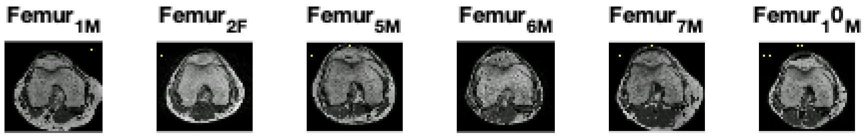


Show results

```
close all
wd ='./FemurImages/';

size(idx1)

figure(1);
for i = 1 : size(idx1,1)
     [pth,FileName,ext] = fileparts(files(idx1(i)).name);
     img = imread([wd,FileName,'.png']);
     subplot(1,size(idx1,1),i);
     imshow(img)
     title(FileName)
end
figure(2);
for i = 1 : size(idx2,1)
     [pth,FileName,ext] = fileparts(files(idx2(i)).name);
     img = imread([wd,FileName,'.png']);
     subplot(1,size(idx2,1),i);
```

```
    imshow(img)
    title(FileName)
end


ans =

     4     1
```





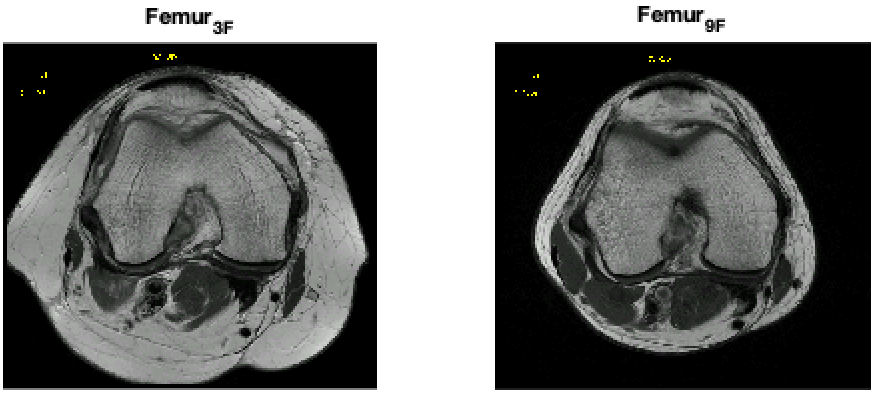
kmean 3 classes

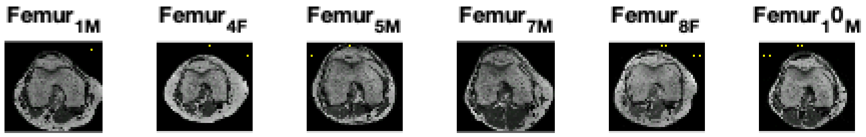
```
close all
statrMatrix = [];
k = 3;

for i=1:k
        statrMatrix(i,1) = min(Project(:,1)) + (max(Project(:,1))-
min(Project(:,1)))*(i*2-1)/(k*2);
        statrMatrix(i,2) = min(Project(:,1)) + (max(Project(:,1))-
min(Project(:,1)))*1/2;
        statrMatrix(i,3) = min(Project(:,1)) + (max(Project(:,1))-
min(Project(:,1)))*1/2;
end

plot3(Project(:,1),Project(:,2),Project(:,3),'.b');
xlabel('PC 1');
ylabel('PC 2');
zlabel('PC 3');
grid on
hold on
axis equal;
view(45,45)
plot3(statrMatrix(1,1),statrMatrix(1,2),statrMatrix(1,3),'.c','MarkerSize',20)
plot3(statrMatrix(2,1),statrMatrix(2,2),statrMatrix(2,3),'.m','MarkerSize',20)
```

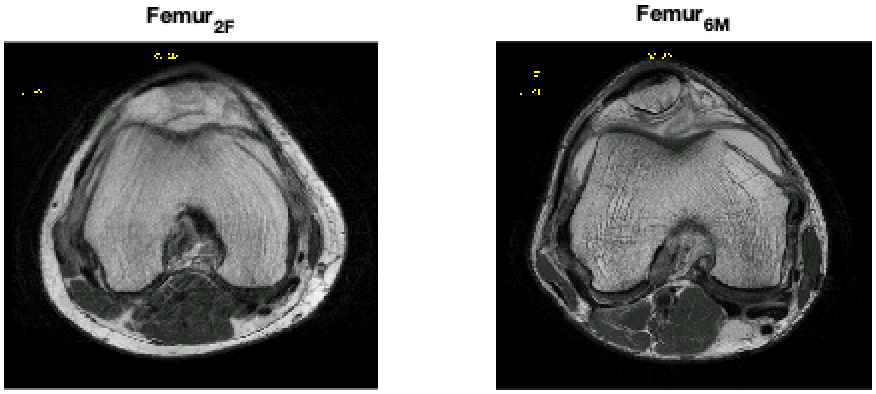
```
plot3(statrMatrix(3,1),statrMatrix(3,2),statrMatrix(3,3),'.y','MarkerSize',20)


[idx,c] = kmeans(Project,k,'start',statrMatrix);

figure;
hold on
xlabel('PC 1');
ylabel('PC 2');
zlabel('PC 3');
grid on
tic

    idx1 = find(idx == 1);
    plot3(Project(idx1,1),Project(idx1,2),Project(idx1,3),'.b')
    idx2 = find(idx == 2);
    plot3(Project(idx2,1),Project(idx2,2),Project(idx2,3),'.r')

    idx3 = find(idx == 3);
    plot3(Project(idx3,1),Project(idx3,2),Project(idx3,3),'.g')


    view(45,45)
toc
axis equal;
hold on;
%
% plot the centroid of the 1st class
plot3(c(1,1),c(1,2),c(1,3),'.c','MarkerSize',20);
plot3(c(2,1),c(2,2),c(2,3),'.m','MarkerSize',20);
plot3(c(3,1),c(3,2),c(3,3),'.y','MarkerSize',20);

% Show results
close all
wd ='./FemurImages/';

size(idx1)

figure(1);
for i = 1 : size(idx1,1)
     [pth,FileName,ext] = fileparts(files(idx1(i)).name);
     img = imread([wd,FileName,'.png']);
     subplot(1,size(idx1,1),i);
     imshow(img)
     title(FileName)
end
figure(2);
for i = 1 : size(idx2,1)
     [pth,FileName,ext] = fileparts(files(idx2(i)).name);
     img = imread([wd,FileName,'.png']);
     subplot(1,size(idx2,1),i);
     imshow(img)
     title(FileName)
```

```
end

figure(3);
for i = 1 : size(idx3,1)
     [pth,FileName,ext] = fileparts(files(idx3(i)).name);
     img = imread([wd,FileName,'.png']);
     subplot(1,size(idx3,1),i);
     imshow(img)
     title(FileName)
end
```

```
Elapsed time is 0.005899 seconds.

ans =

     2     1
```




Femur$_{1M}$
Femur$_{4F}$
Femur$_{5M}$
Femur$_{7M}$
Femur$_{8F}$
Femur$_{1}{}^{0}{}_{M}$

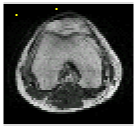

Femur$_{2F}$
Femur$_{6M}$

train with 3D features vector

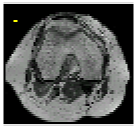
```
close all
clc

true = zeros(size(Project,1),1);
idx = [2,3,4,8,9]; % females
true(idx) = 1;

% train and test with all the images

[w,b,pass] = PerecptronTrn(Project,true);
[e,prediction] = PerecptronTst(Project,true,w,b);
disp(['Test_Errors=' num2str(e) '     Test Data Size= '
 num2str(size(Project,1))])


close all
wd ='./FemurImages/';

idx1 = find(prediction == 1);
idx2 = find(prediction == 0);

size(idx1)

figure(1);
for i = 1 : length(idx1)
    [pth,FileName,ext] = fileparts(files(idx1(i)).name);
     img = imread([wd,FileName,'.png']);
     subplot(1,length(idx1),i);
     imshow(img)
     title(FileName)
end
figure(2);
for i = 1 : length(idx2)
    [pth,FileName,ext] = fileparts(files(idx2(i)).name);
     img = imread([wd,FileName,'.png']);
     subplot(1,length(idx2),i);
     imshow(img)
     title(FileName)
end
```

```
Training_Errors=0     Training data Size=10
Elapsed time is 0.005955 seconds.
Elapsed time is 0.003171 seconds.
Test_Errors=0     Test Data Size= 10

ans =

     1     5
```

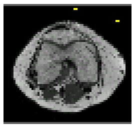
Femur_{2F}



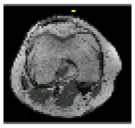
Femur_{3F}



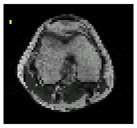
Femur_{4F}



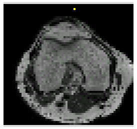
Femur_{8F}



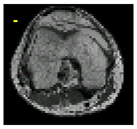
Femur_{9F}



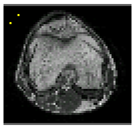
Femur_{1M}



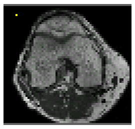
Femur_{5M}



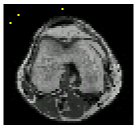
Femur_{6M}



Femur_{7M}



$\text{Femur}_{1}0_{M}$



test leave one out validation

```
for i = 1 : size(Project,1)

    training_data = Project;
    training_true = true;

    training_data(i,:) = [];
    training_true(i) = [];

    [w,b,pass] = PerecptronTrn(training_data,training_true);

    [e(i),prediction] = PerecptronTst(Project(i,:),true(i),w,b);

end

Performance = 100*(length(Project)-mean(e))/length(Project)
```

```
Training_Errors=0     Training data Size=9
Elapsed time is 0.001852 seconds.
Elapsed time is 0.000948 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.002868 seconds.
Elapsed time is 0.000253 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000788 seconds.
Elapsed time is 0.000382 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.002106 seconds.
Elapsed time is 0.000821 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000973 seconds.
Elapsed time is 0.000030 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000156 seconds.
Elapsed time is 0.000031 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000120 seconds.
Elapsed time is 0.000027 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000167 seconds.
Elapsed time is 0.000022 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000182 seconds.
Elapsed time is 0.000022 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000190 seconds.
Elapsed time is 0.000013 seconds.

Performance =

    99
```

training with nD coordinates

```
nD_coordinates = centeredShapeLandmarks';

% test leave one out validation
for i = 1 : size(nD_coordinates,1)

    training_data = nD_coordinates;
    training_true = true;

    training_data(i,:) = [];
    training_true(i) = [];

    [w,b,pass] = PerecptronTrn(training_data,training_true);

    [e(i),prediction] =
 PerecptronTst(nD_coordinates(i,:),true(i),w,b);

end

Performance_nD= 100*(length(nD_coordinates)-mean(e))/
length(nD_coordinates)
```

```
Training_Errors=0     Training data Size=9
Elapsed time is 0.000261 seconds.
Elapsed time is 0.000020 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000168 seconds.
Elapsed time is 0.000040 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000147 seconds.
Elapsed time is 0.000015 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000115 seconds.
Elapsed time is 0.000028 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000222 seconds.
Elapsed time is 0.000027 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000188 seconds.
Elapsed time is 0.000026 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000157 seconds.
Elapsed time is 0.000014 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000103 seconds.
Elapsed time is 0.000054 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000208 seconds.
Elapsed time is 0.000016 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000191 seconds.
Elapsed time is 0.000029 seconds.
```

```
Performance_nD =

   99.5833
```

train with 1D features vector

```
coordinates_1D = projectedPC1';

% test leave one out validation
for i = 1 : size(coordinates_1D,1)

    training_data = coordinates_1D;
    training_true = true;

    training_data(i,:) = [];
    training_true(i) = [];

    [w,b,pass] = PerecptronTrn(training_data,training_true);

    [e(i),prediction] =
 PerecptronTst(coordinates_1D(i,:),true(i),w,b);

end

Performance_1D= 100*(length(coordinates_1D)-mean(e))/length(Project)

% nD had better performance (99.58), 1D and 3D were similar (99). 1D
 and 3D
% are projected and not the actual landmark data, which may have lost
 some
% information.

Training_Errors=2     Training data Size=9
Elapsed time is 0.007458 seconds.
Elapsed time is 0.000965 seconds.
Training_Errors=0     Training data Size=9
Elapsed time is 0.000394 seconds.
Elapsed time is 0.000085 seconds.
Training_Errors=3     Training data Size=9
Elapsed time is 0.006228 seconds.
Elapsed time is 0.000086 seconds.
Training_Errors=2     Training data Size=9
Elapsed time is 0.005193 seconds.
Elapsed time is 0.000034 seconds.
Training_Errors=2     Training data Size=9
Elapsed time is 0.005121 seconds.
Elapsed time is 0.000037 seconds.
Training_Errors=2     Training data Size=9
Elapsed time is 0.005658 seconds.
Elapsed time is 0.000025 seconds.
Training_Errors=2     Training data Size=9
```

```
Elapsed time is 0.005231 seconds.
Elapsed time is 0.000028 seconds.
Training_Errors=2     Training data Size=9
Elapsed time is 0.005319 seconds.
Elapsed time is 0.000022 seconds.
Training_Errors=2     Training data Size=9
Elapsed time is 0.004420 seconds.
Elapsed time is 0.000044 seconds.
Training_Errors=2     Training data Size=9
Elapsed time is 0.004553 seconds.
Elapsed time is 0.000022 seconds.

Performance_1D =

    99
```

K - fold

```
c = cvpartition(10,'kfold',3);
idx = training(c,1);

trainCase = find(idx == 1);
testCase  = find(idx == 0);

[w,b,pass] = PerecptronTrn(Project(trainCase,:),true(trainCase));
[e,prediction] =
 PerecptronTst(Project(testCase,:),true(testCase),w,b);
```

```
Training_Errors=0     Training data Size=7
Elapsed time is 0.000215 seconds.
Elapsed time is 0.001827 seconds.
```

Published with MATLAB® R2018b