Programming Exercise 7:K-means Clustering and Principal Component Analysis

```
>> submit()
== Submitting solutions | K-Means Clustering and PCA...
Use token from last successful submission (WxxShirley@outlook.com)? (Y/n): Y
==
==                                  Part Name |     Score | Feedback
==                                  --------- |     ----- | --------
==            Find Closest Centroids (k-Means) |   30 /  30 | Nice work!
==            Compute Centroid Means (k-Means) |   30 /  30 | Nice work!
==                                        PCA |   20 /  20 | Nice work!
==                          Project Data (PCA) |   10 /  10 | Nice work!
==                          Recover Data (PCA) |   10 /  10 | Nice work!
==                                  ---------- |
==                                             | 100 / 100 |
==
>>
```

1 K-means Clustering

Task: Implement the K-means algorithm and use it for image compression.

1.1 Implementing K-means

The K-means algorithm is as follows:

```
% Initialize centroids
centroids = kMeansInitCentroids(X,K);
for iter=1:iterations
   % Cluster assignment
   idx = findClosestCentroids(X,centroids);

   %Move centroids
   centroids = computeMeans(X,idx,K);
   end
```

1.1.1 Finding closest centroids

$$c^{(i)} := j \quad \text{that minimizes} \quad ||x^{(i)} - \mu_j||^2,$$

```
% Use a 2-dimensional loop
m = size(X,1);
temp = zeros(K,1);
for i=1:m,
for j=1:k,
temp(j) = sum((X(i,:)-centroids(j,:)).^2);
end
[val,ind]=min(temp);
idx(i)=ind;
 end
```

1.1.2 Computing centroid means

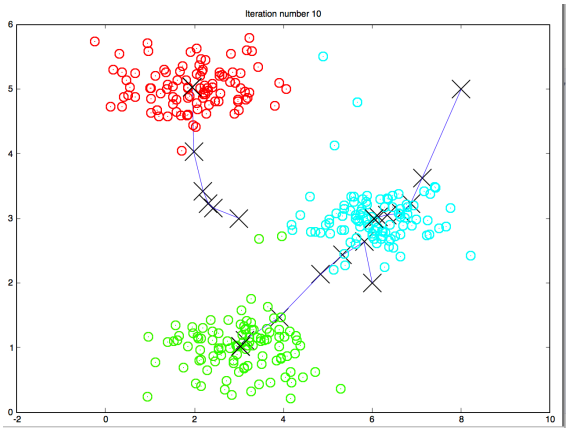
$$\mu_k := \frac{1}{|C_k|} \sum_{i \in C_k} x^{(i)}$$

```
function centroids = computeCentroids(X,idx,K)
```

```
[m n] = size(X);
centroids = zeros(K,n);
m = size(X,1);

for i=1:K,
count=0;
sum=zeros(1,n);
for j=1:m,
 if idx(j)==i
 count+=1;
 sum+=X(j,:);
 end
 centroids(i,:)=sum ./ count;
 end
```

1.2 K-means on example dataset

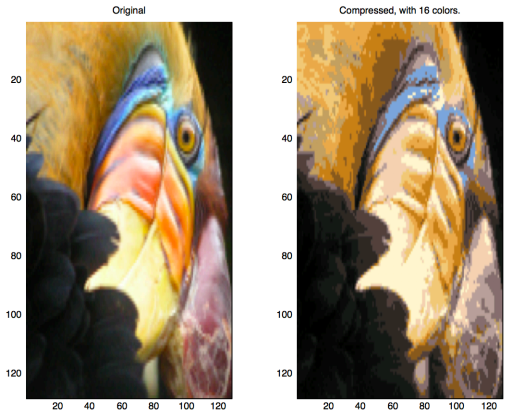
Figure 1:The expected output:



1.3 Random initialization

```
% Initialize the centroids to be random examples

% Randomly reorder the indices of examples
randidx = randperm(size(X,1));
centroids = X(randidx(1:K),:);
```

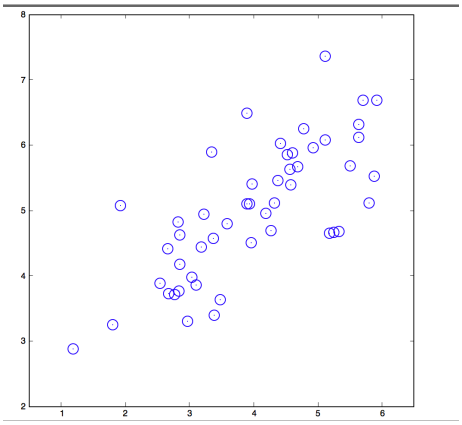
1.4 Image compression with K-means



2 Pricipal Component Analysis (PCA)

Task: Use PCA to perform dimensionality reduction.

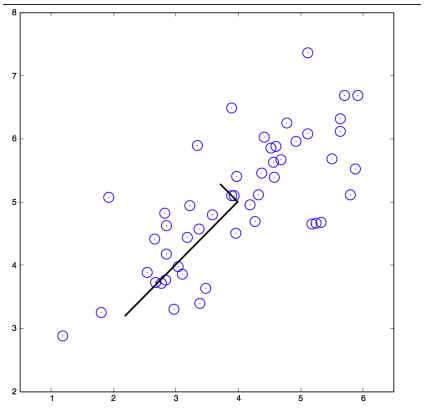
2.1 Example Dataset



2.2 Implementing PCA

Before using PCA, it is important to first normalize the data by subtract- ing the mean value of each feature from the dataset, and scalin each dimen- sion so that they are in the same range.

```
function [U,S] = pca(X)
[m n] = size(X)
U = zeros(n);
S = zeros(n);
sigma = X'*X./m;
[U,S,V] = svd(sigma);
end
```



2.3 Dimensionally Reduction with PCA

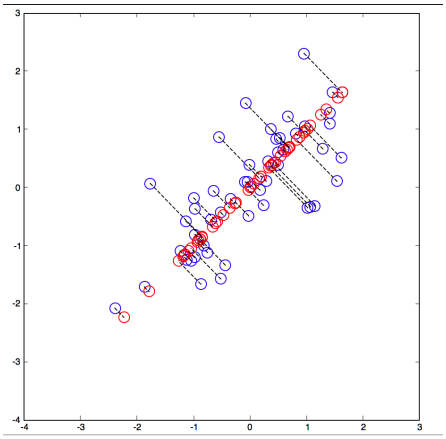
2.3.1 Projecting the data onto the principal components

```
function Z = projectData(X,U,K)
Z = zeros(size(X,1),K);
for i=1:size(X,1),
x=X(i,:)';
ureduce = U(:,1:K);
Z(i,:)=ureduce' * x;
end
end
```

2.3.2 Reconstructing an approximation of the data

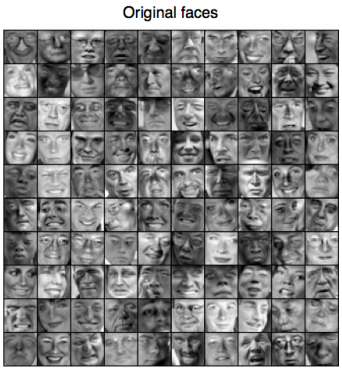
```
function X_rec=recoverData(Z,U,K)
X_rec = zeros(size(Z,1),size(U,1));

for i=1:size(Z,1),
v = Z(i,:)';
t = U(:,1:K)*v;
X_rec(i,:)=t';
end
end
```

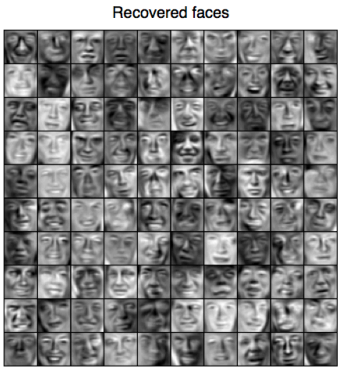
2.3.3 Visualizing the projections



2.4 Face Image Dataset

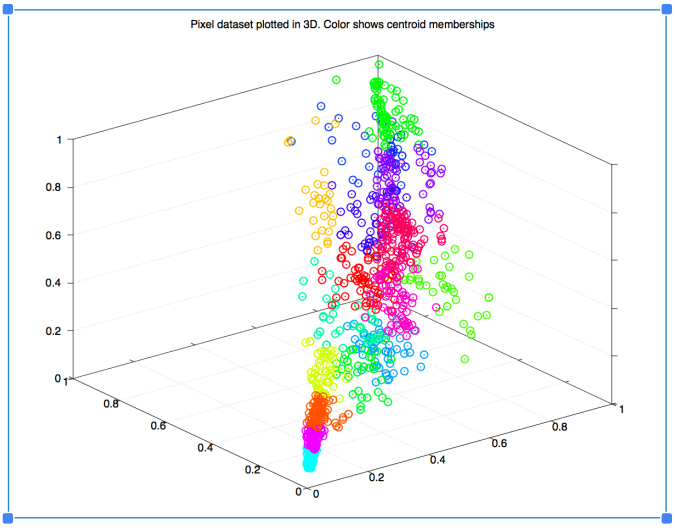
2.4.2 Dimensionality Reduction



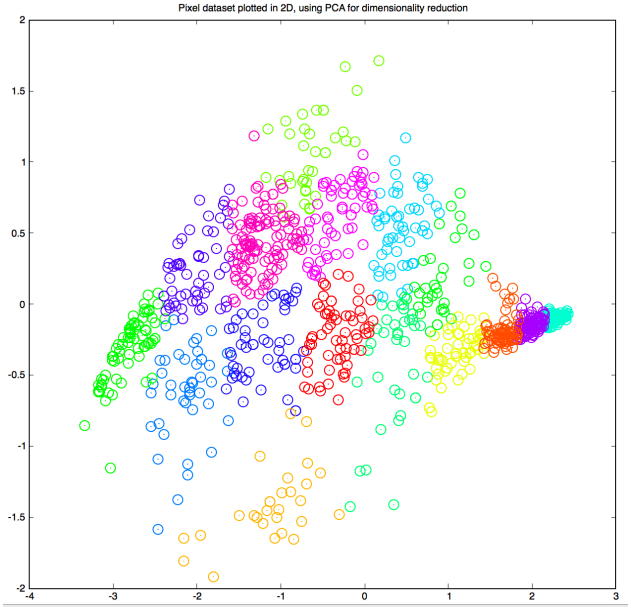


2.5 PCA for visualization

1) Original data in 3D



2) 2D visualization produced using PCA


Pixel dataset plotted in 2D, using PCA for dimensionality reduction
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-4
-3
-2
-1
0
1
2
3