COSC 70 Short Assignment 5

Min Hyung (Daniel) Kang

Due May 28th, 2015

1. Perform a principal components analysis (PCA) on 1799 256 x 256 grayscale images. After loading the file dat.mat you will have in your workspace a 65536 x 1799 data matrix D (note, 256 x 256 = 65536).

2. Perform a PCA on the 1799 images. Because the number of images is less than the dimensionality of each image, you should compute the eigenvectors and eigenvalues of the smaller 1799 x 1799 data matrix, transform each eigenvector to be those of the desired 65536 x 65536 data matrix, and normalize each eigenvector to be unit norm. (Note: Although matlab has a built-in PCA function, you should not use this.)

3. Reshape the first three largest eigenvalue eigenvectors from a 65536 x 1 vector to a 256 x 256 image (see reshape). Looking back on other basis representations that we have discussed in this class, what basis functions does this PCA basis resemble?

4. Now, reduce the dimensionality of each image to 3 by projecting each image onto the first three largest eigenvalue eigenvectors. Compute the center of mass of all 1799 3-D points and find the image that is furthest (in Euclidean distance) from the center of mass – that is, find the least typical image. Which image is it?

1 Short Assignment 5

In this assignment, we attempt to find the image that is most different from all the images. We do PCA projection of the images on a 3-d coordinate, find the center of gravity, and find the point that is most distant to find the most different image.

2 Methods

First, we are given a 65536 x 1799 matrix D, where each column represents a 256$*$256 image. Let's denote each image vector as $I_1, I_2, \cdots, I_{1799}$, and jth pixel of ith image as $I_{i,j}$.

$$D = \begin{bmatrix} I_{1,1} & I_{2,1} & I_{3,1} & \cdots & I_{1799,1} \\ I_{1,2} & I_{2,2} & I_{3,2} & \cdots & I_{1799,2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ I_{1,65536} & I_{2,65536} & I_{3,65536} & \cdots & I_{1799,65536} \end{bmatrix}$$

To perform PCA, we first have to normalize the dataset. Hence, we construct a new matrix Mnew where $Inew_{i,j} = I_{i,j} - mean(I_{i,j})$ where $mean(I_{i,j})$ is mean of the same pixel over all the images.

$$Dnew = \begin{bmatrix} I_{1,1} - mean(I_{i,1}) & I_{2,1} - mean(I_{i,1}) & \cdots & I_{1799,1} - mean(I_{i,1}) \\ I_{1,2} - mean(I_{i,2}) & I_{2,2} - mean(I_{i,2}) & \cdots & I_{1799,2} - mean(I_{i,2}) \\ \cdots & \cdots & \cdots & \cdots \\ I_{1,65536} - mean(I_{i,65536}) & I_{2,65536} - mean(I_{i,65536}) & \cdots & I_{1799,65536} - mean(I_{i,65536}) \end{bmatrix}$$

Note that we have far many dimensions (65536) than number of images(1799). Hence, we are going to compute the correlation matrix using smaller dimension matrix multiplication :

Let

$$SmallerC = Dnew' * Dnew$$

From this smaller covariance matrix, we find the eigenvectors and eigenvalues using *eig*. Then we multiply by MNew to get the eigenvectors of the actual covariance matrix (Dnew * Dnew'). We then normalize all the eigenvectors by dividing each eigenvector by its vector norm.

We search for three greatest eigenvalues and corresponding eigenvectors. We do this by ordering the eigenvalues and using its index, ordering the eigenvectors and getting the top 3. Let's denote the three eigenvectors with greatest eigenvalues as e_1, e_2, e_3.

We create a 65536 x 3 matrix

$$Max3Eigen = [e_1, e_2, e_3]$$

.

Note that we can now perform PCA on original data and project on the 3-d plane. We compute the following :

$$DProject = (Max3Eigen)' * Dnew$$

It then gives the 3x1799 DProject matrix, or a 3d PCA projection of the dataset. Let's denote the matrix as following :

$$DProject = \begin{bmatrix} x_1 & x_2 & \cdots & x_{1799} \\ y_1 & y_2 & \cdots & y_{1799} \\ z_1 & z_2 & \cdots & z_{1799} \end{bmatrix}$$

We compute the center of gravity for all the data points :

$$COM = \begin{bmatrix} x_{com} \\ y_{com} \\ z_{com} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{1799}(x_i)/1799 \\ \sum_{i=1}^{1799}(y_i)/1799 \\ \sum_{i=1}^{1799}(z_i)/1799 \end{bmatrix}$$

Then for each data point, we compute the euclidean distance from the COM.

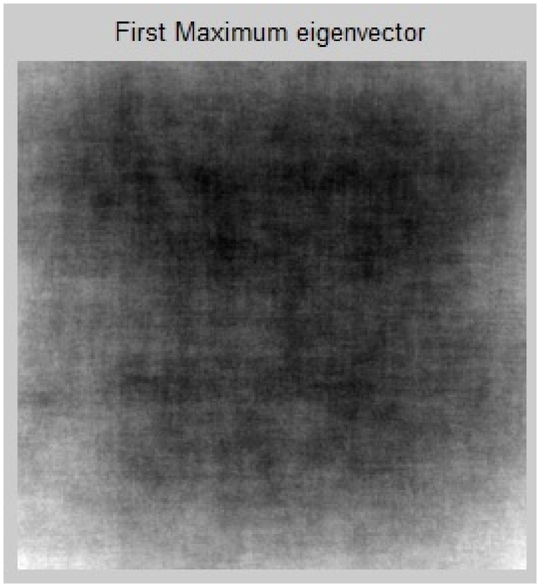
$$d_i = \sqrt{(x_i - x_{com})^2 + (y_i - y_{com})^2 + (z_i - z_{com})^2}$$

We find the point with farthest distance, and that point (image) should be the most different image.

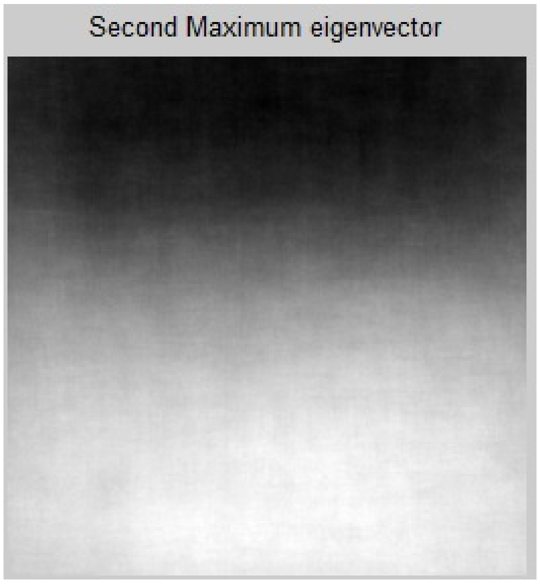
3 Results

We first look at the matrix form of 3 maximum eigenvectors.

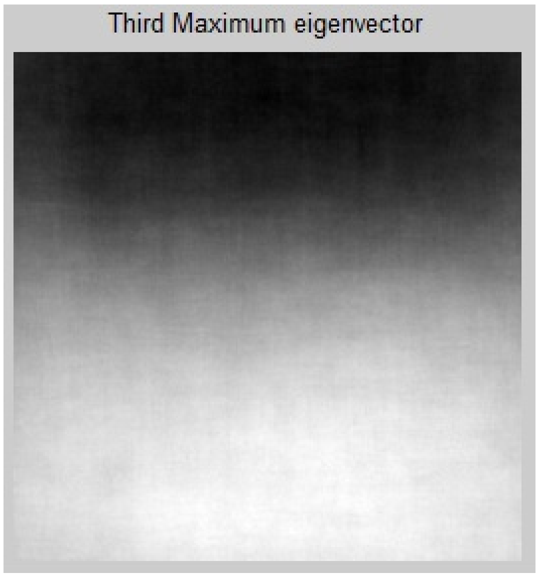
We see that the eigenvectors, especially the 2nd and third eigenvectors look like fourier basis, notably a single period of sinusoid grating.



(a) First Maximum Eigenvector

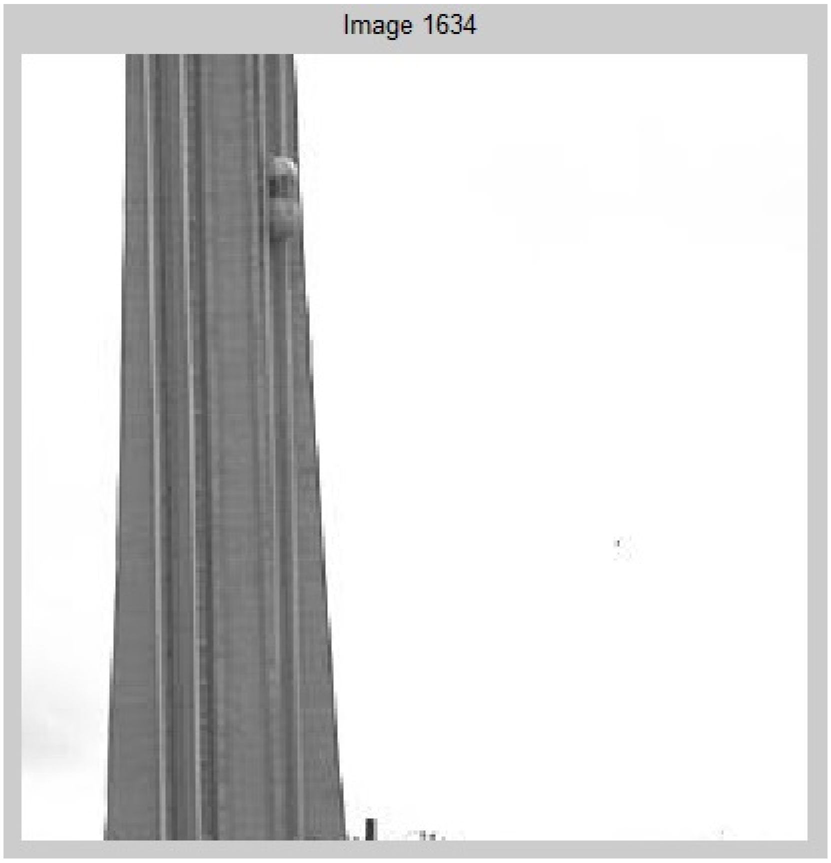


(b) Second Maximum Eigenvector



(c) Third Maximum Eigenvector

We then project all the images on to the basis, compute the center of mass of all the images, and find the point that is farthest from the point. We see that the image 1634 is the farthest.



4 Source Code

```
clear all;

%Read the data
load dat.mat;


%Get the dimension
%n = number of pixels in each image
%m = number of images
[n,m] = size(D);

%Cast for computation
D=im2double(D);

%Make the data center around zero
mu = mean(D')';
Dnew = zeros(n,m);
for i = 1:m
    Dnew(:,i) = (D(:,i) - mu);
end

%Since n>>m, We compute the covariance Matrix of M'M, and get its
%eigenvectors
SmallerC = Dnew.' * Dnew;

[V,E] = eig(SmallerC);
%Get the eigenvectors of desired covariance matrix
Eigen = Dnew*V;

%Normalize all the eigenvectors
for i = 1:m
    Norm = norm(Eigen(:,i));
    Eigen(:,i) = Eigen(:,i) / Norm;
end

%Sort the eigenvalues and corresponding eigenvectors,
%Get three largest eigenvalue eigenvectors
[a,b]=sort(diag(E),'descend');
Eigen = Eigen(:,b);
Max3Eigen = Eigen(:,1:3);
display(a(1:3));

%Reshape the max 3 eigen vectors in square matrix form and display them
%to see what they look like
```

```
first = reshape(Max3Eigen(:,1),256,256);
second = reshape(Max3Eigen(:,2),256,256);
third = reshape(Max3Eigen(:,3),256,256);
Ifirst = mat2gray(first);
Isecond = mat2gray(second);
Ithird = mat2gray(second);
imshow(Ifirst);
title('First Maximum eigenvector');
pause;
imshow(Isecond);
title('Second Maximum eigenvector');
pause;
imshow(Ithird);
title('Third Maximum eigenvector');
pause;

%Project onto three dimensions
%Each column represents a three dimensional representation of image
DProject = Max3Eigen.' * Dnew;

%Compute center of mass
COM = [mean(DProject(1,:));mean(DProject(2,:));mean(DProject(3,:))];
display(COM);

%Compute Euclidean Distance
Norm = zeros(1,m);
for i=1:m
   Norm(1,i) = norm(DProject(:,i) - COM) ;
end

%Show the image furthest from the COM
[Max,I] = max(Norm);
result = reshape(D(:,I),256,256);
str = sprintf('Image %d',I);
imshow(result);
title(str);
```