Assignment-1

October 6, 2017

Team 08

Instructor
Prasanth LA

Vamsi Dikkala CS17M048
Vivek Kumar Agrawal CS17M049

Contents

1 Question 1 Projections

1.1 a) Projection Matrix on to the Column Space of A

Given Matrix

$$A = \begin{bmatrix} 3 & 6 & 6 \\ 4 & 8 & 8 \end{bmatrix}$$

1.1.1 Column Space

The column space of A is the linear combinations of the columns of A.In the Given matrix A, columns are dependent. Since columns are dependent, column space of A is linear combination of first column only. Hence

$$col(A) = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

1.1.2 Row Space

The row space of A is the linear combinations of the columns of A^{T}.In the Given matrix A^{T}, columns are dependent. Since columns are dependent, column space of A^{T} is linear combination of first column only. Hence

$$col(A^{\mathrm{T}}) = row(A) = \begin{bmatrix} 3 \\ 6 \\ 6 \end{bmatrix}$$

1.1.3 Projection Matrix

The projection matrix for any matrix A is given by

$$P = A(A^T A)^{-1} A^T$$

Projection matrix of column space of A is P_c, and row space is P_r

$$P_{\mathrm{c}} = \begin{bmatrix} 0.3600 & 0.4800 \\ 0.4800 & 0.6400 \end{bmatrix}$$

$$P_{\mathrm{r}} = \begin{bmatrix} 0.1111 & 0.2222 & 0.2222 \\ 0.2222 & 0.4444 & 0.4444 \\ 0.2222 & 0.4444 & 0.4444 \end{bmatrix}$$

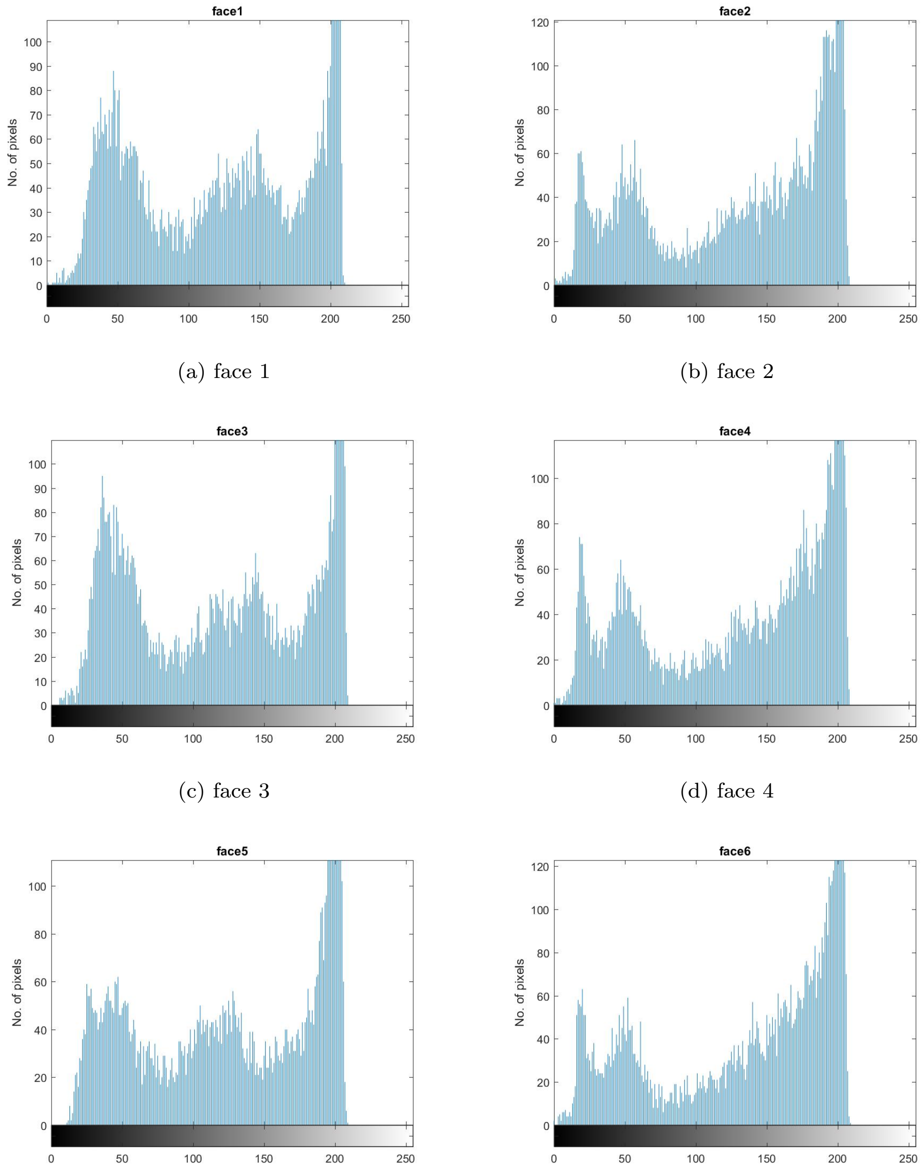
1.2 b) Finding $B=P_cAP_r$

While Computing B, it's coming equal to A.The Reason for this is that in first case we are projecting the matrix A on it's own column space,so the projection is coming out to be A itself.

In second case we are projecting the A on it's own row space, so again the projection is coming out to be A itself.

2 Question 2 Projections

2.1 Q2.a Histograms

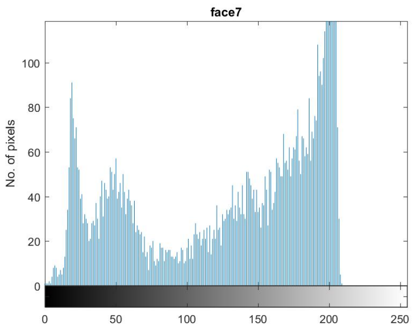


(a) face 1 (b) face 2

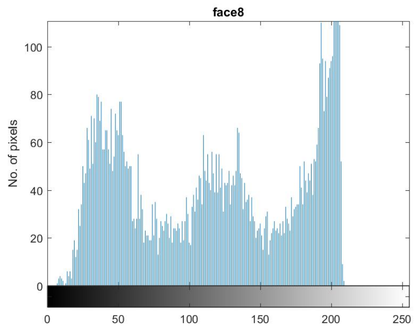
(c) face 3 (d) face 4

(e) face 5 (f) face 6

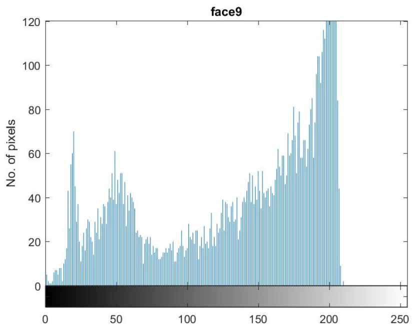
Figure 1: Intensity Histograms of Eigen-basis



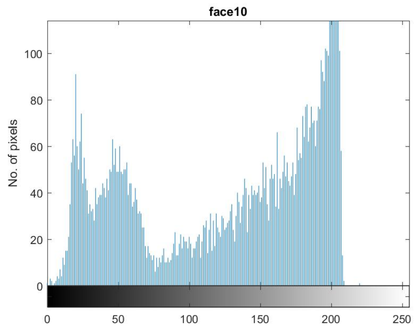
(a) face 7



(b) face 8



(c) face 9



(d) face 10

Figure 2: Intensity Histograms of Eigen-basis

2.2 Q 2.c Error plots

The given eigen basis matrix (EigenVectors.mat) contains orthonormal vectors q_1,q_2,q_3,....q_{8464} each of dimension 8464 representing the complete image space. Every image in this image space can be represented as linear combination of these orthonormal vectors. Let the given image B, therefore we can write B as

$$B = q_1x_1 + q_2x_2 + q_3x_3 + ...+ q_{8464}x_{8464}.$$

Finding the coefficents { x_1 , x_2 , x_3, ..., x_{8464} } Since q_1 , q_2 , q_3 ,.... ,q_{8464} are orthonormal.

$$q_i{}^T q_j = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

$$q_1{}^T B = q_1{}^T q_1 x_1 + q_1{}^T q_2 x_2 + q_1{}^T q_3 x_3 + ...+ q_1{}^T q_{8464} x_{8464}.$$

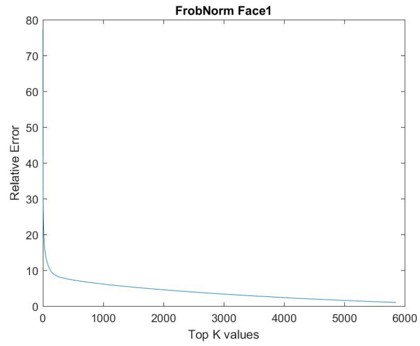
$$q_1{}^T q_2 x_2 + q_1{}^T q_3 x_3 + ...+ q_1{}^T q_{8464} x_{8464} = 0.$$

$$x_1 = q_1{}^T B$$

$$\hat{x} = \begin{bmatrix} | & | & | & | & | \\ q_1 & q_2 & . & . & q_{8464} \\ | & | & | & | & | \end{bmatrix}^T B$$

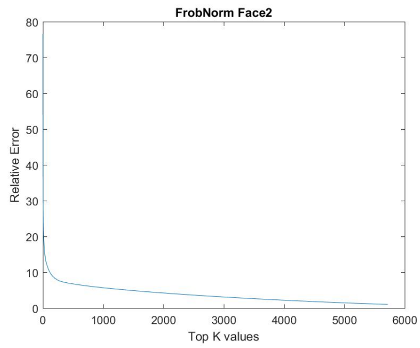
Now we got the Coefficent Vector . Coefficent Vector tells along which direction most of the Image features are present. Now we will select top K coefficent values and their corresponding eigen basis to reconstruct our image with less than 1% relative frobenius error between original and reconstructed image.

(a) Reconstructed Face 1



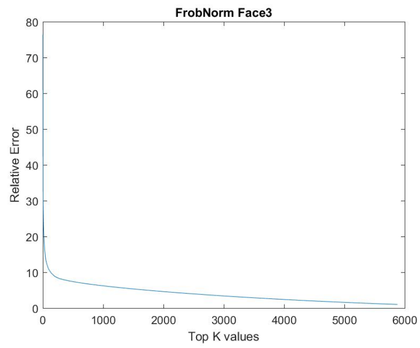
(b) Frobenius Error face 1

(c) Reconstructed Face 2



(d) Frobenius Error face 2

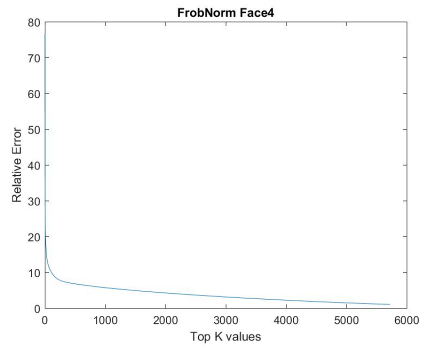
(e) Reconstructed Face 3



(f) Frobenius Error face 3

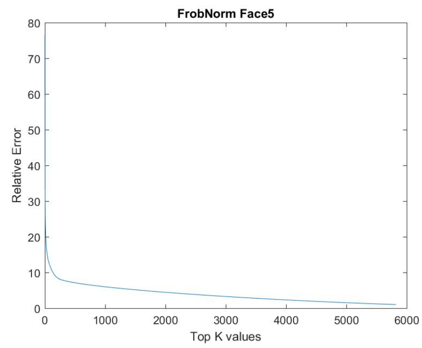
Figure 3: Reconstructed Faces and Corresponding Frobenius error plots

(a) Reconstructed Face 4



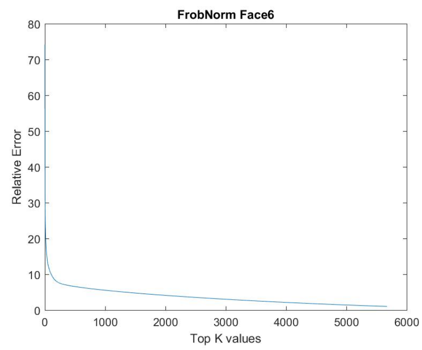
(b) Frobenius Error face 4

(c) Reconstructed Face 5



(d) Frobenius Error face 5

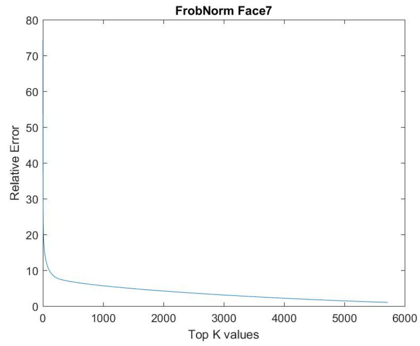
(e) Reconstructed Face 6



(f) Frobenius Error face 6

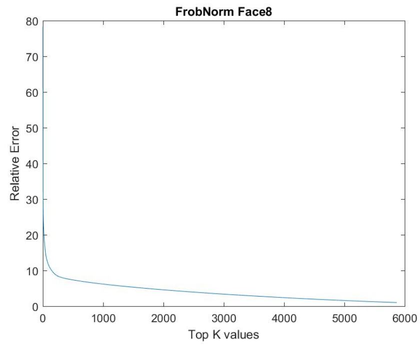
Figure 4: Reconstructed Faces and Corresponding Frobenius error plots

(a) Reconstructed Face 7



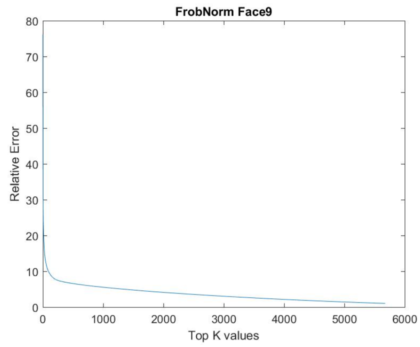
(b) Frobenius Error face 7

(c) Reconstructed Face 8



(d) Frobenius Error face 8

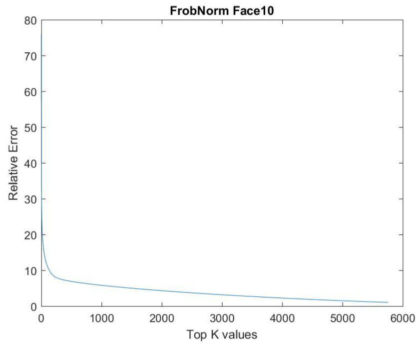
(e) Reconstructed Face 9



(f) Frobenius Error face 9

Figure 5: Reconstructed Faces and Corresponding Frobenius error plots

(a) Reconstructed Face 10



(b) Frobenius Error face 10

Figure 6: Reconstructed Faces and Corresponding Frobenius error plots

2.3 Code For Histogram(Q2.a)

```
for im=1:10
str1 = 's08/face';
str2 = int2str(im);
str = strcat(str1,str2);
str = strcat(str,'.pgm');
f1 = imread(str);
h = figure
imhist(f1);
str = strcat('face',str2);
ylabel('No. of pixels');
title(str);
end
```

2.4 Code For Converting Basis to Images(Q2.b)

```
load('EigenVectors.mat');
for i=1:3:75
 str = 'basis ';
 str2 = int2str(i);
 str = strcat(str,str2);
 h = figure
 imshow(basis2img(COEFF,i));
 title(str);
end
```

2.5 Code For Finding the top K Basis(Q2.c)

```
load('EigenVectors.mat');
for t=1:10
 str1 = 's08/face';
 str2 = int2str(t);
 str = strcat(str1,str2);
 str = strcat(str,'.pgm');
 f1 = imread(str);
 h = figure
 b1 = img2basis(f1);
 b1 = double(b1);
 xb1 = COEFF'*b1;
 abs_xb1 = abs(xb1);
 [xb1_sort,index] = sort(abs_xb1,'descend');

 relative_error_arr = zeros(8464,1);
 fnorm_original = 0;
 for j=1:8464
  fnorm_original = fnorm_original + b1(j,1)^2;
 end
 fnorm_original = sqrt(fnorm_original);

 for k=1:8464
  recon = zeros(8464,1);
```

```
  for i=1:k
   recon = recon +  xb1(index(i))*COEFF(:,index(i));
  end
  error = b1 - recon;
  fnorm_error = 0;
  for j=1:8464
   fnorm_error = fnorm_error + error(j,1)^2;
  end
  fnorm_error = sqrt(fnorm_error);
  relative_error = fnorm_error/fnorm_original;
  relative_error = relative_error*100;
  relative_error_arr(k,1) = relative_error;
  if(relative_error < 1)
   imshow(basis2img(recon,1));
   break;
  end
 end

 str = 'Top ';
 str3 = int2str(k);
 str = strcat(str,str3);
 str = strcat(str,' face');
 str = strcat(str,str2);
 title(str);
 str = strcat('LARP assignment\',str);
 print(h,str,'-djpeg');

 h = figure
 plot(1:k,relative_error_arr(1:k,1));
 xlabel('Top K values');
 ylabel('Relative Error');
 str = strcat('FrobNorm Face',str2);
 title(str);
 str = strcat('LARP assignment\',str);
 print(h,str,'-djpeg');

end
```

2.6 Functions

2.6.1 Basis to Image

```
function [img] = basis2img( COEFF,col )
count = 1;
for i=1:92
 for j=1:92
  img(j,i) = COEFF(count,col);
  count = count+1;
 end
end
 img = mat2gray(img);
end
```

2.6.2 Image to Basis

```
function [ b ] = img2basis( m )
 k = 0;
 for i=1:92
  b(k+1:k+92,1) = m(:,i);
  k = k+92;
 end
end
```

2.7 Individual Basis and Final Images

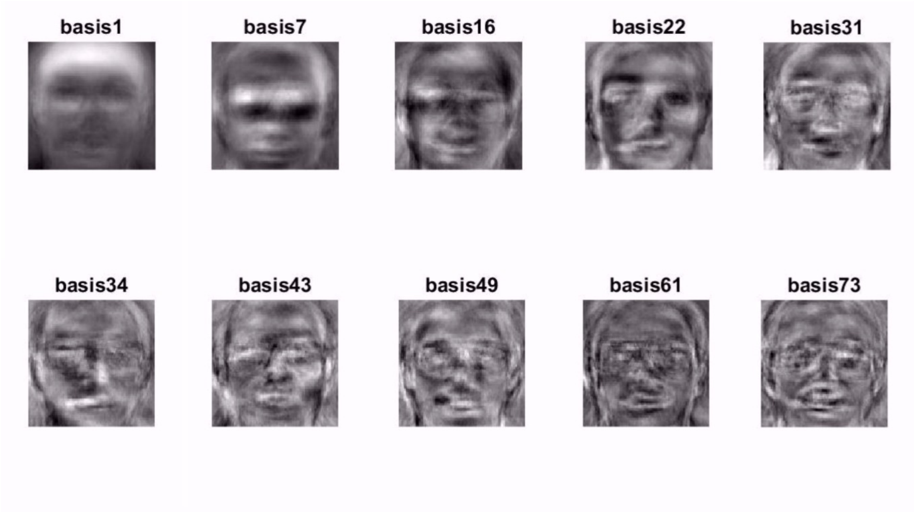


Figure 7: Individual Basis

;

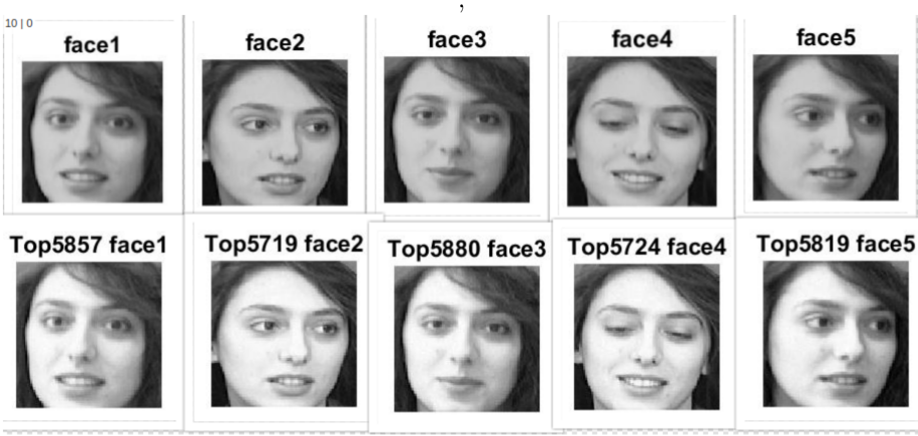


Figure 8: Comparision between original and reconstructed faces