Math 521 Homework 3

Calculating SVD Matrices with Eigenvalues and Eigenvectors
Principal Component Analysis and Snap Shot Method

Angel Silvar

Math 521
California State University - Long Beach
April 1, 2022

1 Theory

1.1 Problem 1: Gradient Direction with Symmetric Matrix

1) Show $\nabla_{\boldsymbol{v}}\langle \boldsymbol{v}, \boldsymbol{v}\rangle = 2\boldsymbol{v}$, and if C is symmetric matrix, then $\nabla_{\boldsymbol{v}}\langle \boldsymbol{v}, C\boldsymbol{v}\rangle = 2\boldsymbol{v}$

Let $\boldsymbol{v} \in \mathbf{R}^p$ Recall definition of Euclidean Distance:

$$||\boldsymbol{v}||_2^2 = \sum_{i=1}^{p} v_i^2 = \langle \boldsymbol{v}, \boldsymbol{v}\rangle$$

Then, we have

$$\frac{\partial\langle \boldsymbol{v}, \boldsymbol{v}\rangle}{\partial \boldsymbol{v}_k} = \frac{\partial}{\partial \boldsymbol{v}_k}\sum_{i=1}^{p} v_i^2$$

Then by linearity

$$= \sum_{i=1}^{p} \frac{\partial p_i^2}{\partial \boldsymbol{v}_k} = 2\sum_{i=1}^{p} v_i = 2\boldsymbol{v}$$

Note that we get $2\boldsymbol{v}_i$ when i = k

2) Now show,

$$(\phi^{(1)}, C\phi^{(2)}) = (C\phi^{(1)}, \phi^{(2)})$$

$$\frac{\partial\langle \boldsymbol{v}, \boldsymbol{v}\rangle}{\partial \boldsymbol{v}_k} = \frac{\partial}{\partial \boldsymbol{v}_k}\sum_{i=1}^{p} v_i^2$$

3) Given the data matrix $\mathbb{X}$, compute $\mathbb{X}\mathbb{X}^T$ and $\mathbb{X}^T\mathbb{X}$. For $u^{(j)}$, confirm

$$\mathbb{X} = \begin{pmatrix} -2 & -1 & 1 \\ 0 & -1 & 0 \\ -1 & 1 & 1 \\ 1 & -1 & 1 \end{pmatrix}$$

For $u^{(j)}$, confirm

$$u^{(j)} = \frac{1}{\sigma_j}\sum_{i=1}^{p} \boldsymbol{v}_i^j \boldsymbol{x}^k$$

where j = 1, ... rank(x)

$$\mathbb{X}\mathbb{X}^T = \begin{pmatrix} 6 & 1 & 3 & 0 \\ 1 & 1 & -1 & 1 \\ 3 & -1 & 6 & 3 \\ 0 & 1 & 0 & 3 \end{pmatrix}$$

$$\mathbb{X}^T\mathbb{X} = \begin{pmatrix} 6 & 0 & -3 \\ 0 & 4 & 0 \\ -3 & 0 & 6 \end{pmatrix}$$

Find Eigenvalues and Eigenvectors for $\mathbb{X}^T\mathbb{X}$

By definition of Eigenvalues $Ax = \lambda x$ where $A = \mathbb{X}^T\mathbb{X}$

$$Ax - I\lambda x \rightarrow det([A - I\lambda]x) = 0$$

$$|\mathbb{X}^{\mathbb{T}}\mathbb{X} - \mathbb{I}\lambda| = \begin{vmatrix} 6-\lambda & 0 & -3 \\ 0 & 4-\lambda & 0 \\ -3 & 0 & 6-\lambda \end{vmatrix} = -(6-\lambda)(4-\lambda)(\lambda-3) = 0$$

This gives us $\lambda = 9, 4, 3$. Note: $\sigma = \sqrt{9}, \sqrt{4}, \sqrt{3}$

$\lambda = 9$

$$\mathbb{X}^T\mathbb{X} - 9I = \begin{pmatrix} 6-\lambda & 0 & -3 \\ 0 & 4-\lambda & 0 \\ -3 & 0 & 6-\lambda \end{pmatrix} \longrightarrow \begin{pmatrix} 6-9 & 0 & -3 \\ 0 & 4-9 & 0 \\ -3 & 0 & 6-9 \end{pmatrix}$$

$$= \begin{pmatrix} -3 & 0 & -3 \\ 0 & -5 & 0 \\ -3 & 0 & -3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$x_1 = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \longrightarrow v_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$

$\lambda = 4$

$$\mathbb{X}^T\mathbb{X} - 4I = \begin{pmatrix} 6-\lambda & 0 & -3 \\ 0 & 4-\lambda & 0 \\ -3 & 0 & 6-\lambda \end{pmatrix} \longrightarrow \begin{pmatrix} 6-4 & 0 & -3 \\ 0 & 4-4 & 0 \\ -3 & 0 & 6-4 \end{pmatrix}$$

$$= \begin{pmatrix} 2 & 0 & -3 \\ 0 & 0 & 0 \\ -3 & 0 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$x_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \longrightarrow v_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$

$\lambda = 3$

$$\mathbb{X}^T\mathbb{X} - 3I = \begin{pmatrix} 6-\lambda & 0 & -3 \\ 0 & 4-\lambda & 0 \\ -3 & 0 & 6-\lambda \end{pmatrix} \longrightarrow \begin{pmatrix} 6-3 & 0 & -3 \\ 0 & 4-3 & 0 \\ -3 & 0 & 6-3 \end{pmatrix}$$

$$= \begin{pmatrix} 3 & 0 & -3 \\ 0 & 1 & 0 \\ -3 & 0 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$x_2 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \longrightarrow v_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$

Now, consider

Find Eigenvalues and Eigenvectors for XX^T. Let B = XX^T where

$$\mathbb{X}\mathbb{X}^{\mathbb{T}} = \begin{pmatrix} 6 & 1 & 3 & 0 \\ 1 & 1 & -1 & 1 \\ 3 & -1 & 6 & 3 \\ 0 & 1 & 0 & 3 \end{pmatrix}$$

By definition of Eigenvalues $Bx = \lambda x$ where $B = \mathbb{X}^T\mathbb{X}$

$$Bx - I\lambda x \rightarrow det([B - I\lambda]x) = 0$$

$$|\mathbb{X}\mathbb{X}^T - I\lambda| = \begin{vmatrix} 6-\lambda & 1 & 3 & 0 \\ 1 & 1-\lambda & -1 & 1 \\ 3 & -1 & 6-\lambda & 3 \\ 0 & 1 & 0 & 3-\lambda \end{vmatrix}$$

$$|B - \lambda I| = \lambda(\lambda - 9)(\lambda - 4)(\lambda - 3) = 0$$

$\lambda = 0, 9, 4, 3$

Consider: $\lambda = 9$

$$\mathbb{X}\mathbb{X}^T - I\lambda = \begin{pmatrix} 6-\lambda & 1 & 3 & 0 \\ 1 & 1-\lambda & -1 & 1 \\ 3 & -1 & 6-\lambda & 3 \\ 0 & 1 & 0 & 3-\lambda \end{pmatrix}$$

$$\mathbb{X}\mathbb{X}^T - 9I \begin{pmatrix} 6-9 & 1 & 3 & 0 \\ 1 & 1-9 & -1 & 1 \\ 3 & -1 & 6-9 & 3 \\ 0 & 1 & 0 & 3-9 \end{pmatrix} = \begin{pmatrix} -3 & 1 & 3 & 0 \\ 1 & -8 & -1 & 1 \\ 3 & -1 & -3 & 3 \\ 0 & 1 & 0 & -6 \end{pmatrix}$$

$$x_1 = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \longrightarrow v_1 = \tfrac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

$\lambda = 4$

$$\mathbb{X}\mathbb{X}^T - \lambda I = \begin{pmatrix} 6-\lambda & 1 & 3 & 0 \\ 1 & 1-\lambda & -1 & 1 \\ 3 & -1 & 6-\lambda & 3 \\ 0 & 1 & 0 & 3-\lambda \end{pmatrix}$$

$$\mathbb{X}\mathbb{X}^T - 4I = \begin{pmatrix} 6-4 & 1 & 3 & 0 \\ 1 & 1-4 & -1 & 1 \\ 3 & -1 & 6-4 & 3 \\ 0 & 1 & 0 & 3-4 \end{pmatrix} = \begin{pmatrix} 2 & 1 & 3 & 0 \\ 1 & -3 & -1 & 1 \\ 3 & -1 & 2 & 3 \\ 0 & 1 & 0 & -1 \end{pmatrix}$$

$$x_2 = \begin{pmatrix} 1 \\ 1 \\ -1 \\ 1 \end{pmatrix} \longrightarrow v_2 = \tfrac{1}{2} \begin{pmatrix} 1 \\ 1 \\ -1 \\ 1 \end{pmatrix}$$

$\lambda = 3$

$$\mathbb{X}\mathbb{X}^T - \lambda I = \begin{pmatrix} 6-\lambda & 1 & 3 & 0 \\ 1 & 1-\lambda & -1 & 1 \\ 3 & -1 & 6-\lambda & 3 \\ 0 & 1 & 0 & 3-\lambda \end{pmatrix}$$

$$\mathbb{X}\mathbb{X}^T - 3I = \begin{pmatrix} 6-3 & 1 & 3 & 0 \\ 1 & 1-3 & -1 & 1 \\ 3 & -1 & 6-3 & 3 \\ 0 & 1 & 0 & 3-3 \end{pmatrix} = \begin{pmatrix} 3 & 1 & 3 & 0 \\ 1 & -2 & -1 & 1 \\ 3 & -1 & 3 & 3 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$x_3 = \begin{pmatrix} -1/2 \\ 0 \\ 1/2 \\ 1 \end{pmatrix} \longrightarrow v_3 = \tfrac{1}{2} \begin{pmatrix} 1 \\ 1 \\ -1 \\ 1 \end{pmatrix}$$

$\lambda = 0$

$$\mathbb{X}\mathbb{X}^T - \lambda I = \begin{pmatrix} 6-\lambda & 1 & 3 & 0 \\ 1 & 1-\lambda & -1 & 1 \\ 3 & -1 & 6-\lambda & 3 \\ 0 & 1 & 0 & 3-\lambda \end{pmatrix}$$

$$\mathbb{X}\mathbb{X}^T - 0I = \begin{pmatrix} 6-0 & 1 & 3 & 0 \\ 1 & 1-0 & -1 & 1 \\ 3 & -1 & 6-0 & 3 \\ 0 & 1 & 0 & 3-0 \end{pmatrix} = \begin{pmatrix} 6 & 1 & 3 & 0 \\ 1 & 1 & -1 & 1 \\ 3 & -1 & 6 & 3 \\ 0 & 1 & 0 & 3 \end{pmatrix}$$

$$x_3 = \begin{pmatrix} 1 \\ -3 \\ -1 \\ 1 \end{pmatrix} \longrightarrow v_3 = \frac{1}{2\sqrt{3}} \begin{pmatrix} 1 \\ -3 \\ -1 \\ 1 \end{pmatrix}$$

Now we have singular values. For our right singular vectors:

$$\mathbb{U} = \begin{pmatrix} 1/\sqrt{2} & 1/2 & -\sqrt{6}/6 & \sqrt{3}/6 \\ 0 & 1/2 & 0 & \sqrt{3}/2 \\ 1/\sqrt{2} & -1/2 & \sqrt{6}/6 & -\sqrt{3}/6 \\ 0 & 1/2 & \sqrt{6}/3 & \sqrt{3}/6 \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & \sqrt{3} \\ 0 & 0 & 0 \end{pmatrix}$$

$$\mathbb{V} = \begin{pmatrix} -1/\sqrt{2} & 0 & 1/\sqrt{2} \\ 0 & 1 & 0 \\ 1/\sqrt{2} & 0 & 1/\sqrt{2} \end{pmatrix}$$

Where $B = \mathbb{U}\Sigma\mathbb{V}$ Now, to show

$$u^{(j)} = \frac{1}{\sigma_j} \sum_{i=1}^{p} \boldsymbol{v}_i^j \boldsymbol{x}^k$$

Now for

$$u^{(1)} = \frac{1}{3}\left[\frac{-1}{\sqrt{2}} \begin{pmatrix} -1 \\ 0 \\ -2 \\ -1 \end{pmatrix} + 0x^2 + \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 2 \\ 1 \end{pmatrix}\right] = \begin{pmatrix} 1/\sqrt{2} \\ 0 \\ 1/\sqrt{2} \\ 0 \end{pmatrix}$$

2 Problem 1: SnapShot Method

Here, the following Matlab code will show how to apply mean ensemble average on the entire data set, random column within the data set, apply eigen value and Eigen vector basis based on the mean -subtracted average. Then check for error from the actual data set and the truncated data set.

```
% Load image, store as double
load faces1
x=double(Y1);

%Display of ensemble-average Image
m=mean(x,2);
P= reshape(m,[120,160]);
figure;
imagesc(P), colormap(gray)
title('Display of Ensemble-Average Image')
```

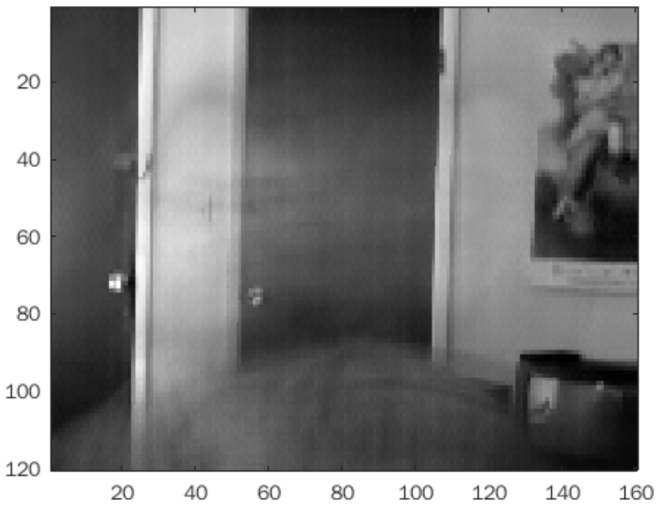

Figure 1 – Display of ensemble average.

```
k = randi(105);
x_tilda=x-repmat(m,1,size(x,2));
randomImg= x_tilda(:,k);
figure;
imagesc(reshape(randomImg, [120,160])), colormap(gray);
title('Mean Subtracted Image of kth column')
```

Figure 2 – 41st image - Ensemble average.

```

[U,Sigma,~]=svd(x_tilda,0);
u = rank(U);

a = 1 ;
b = randi(u);
c = randi(u);
d = randi(u);
```

```
9
10 s = strcat( 'Eigenvalue picture: ',num2str(a));
11 s2 = strcat( 'Eigenvalue picture: ',num2str(b));
12 s3 = strcat( 'Eigenvalue picture: ',num2str(c));
13 s4 = strcat( 'Eigenvalue picture: ',num2str(d));
14
15 % Store images
16 Ba=reshape(  U(:,a) ,120,160);
17 Bb=reshape( U(:,b) ,120,160);
18 Bc=reshape( U(:,c) ,120,160);
19 Bd=reshape(U(:,d) ,120,160);
20
21
22 % Display images as subplot
23 subplot(2,2,1), imagesc(Ba),  title(s) , colormap( gray )
24 subplot(2,2,2), imagesc(Bb), title(s2), colormap( gray )
25 subplot(2,2,3), imagesc(Bc), title(s3), colormap( gray )
26 subplot(2,2,4), imagesc(Bd), title(s4), colormap( gray )
```

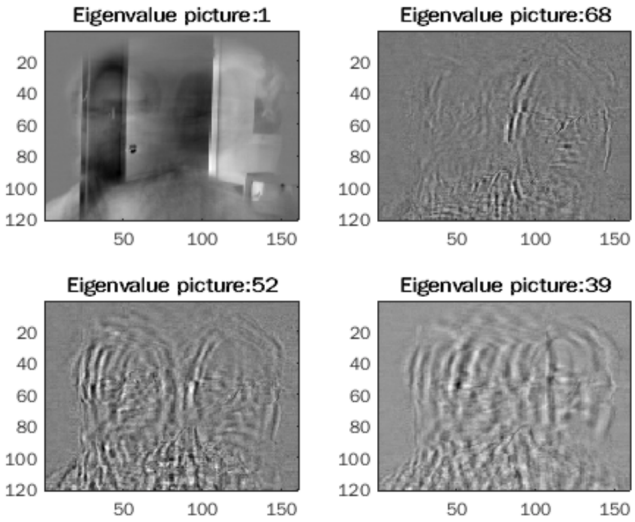

Figure 3 – Output of using Eigenvectors as a basis.

```
1
2 A=U'*x_tilda;
3
4 D=5;
5 xtrunc_5=U(:,1:D)*A(1:D,k);
6 error_5=norm(x_tilda(:,k)-xtrunc_5);
7
8 D=20;
9 xtrunc_20=U(:,1:D)*A(1:D,k);
10 error_20=norm(x_tilda(:,k)-xtrunc_20);
11
12 D=40;
13 xtrunc_40=U(:,1:D)*A(1:D,k);
14 error_40=norm(x_tilda(:,k)-xtrunc_40);
15
16 D=60;
```

```
xtrunc_60=U(:,1:D)*A(1:D,k);
error_60=norm(x_tilda(:,k)-xtrunc_60);

% Perfect reconstruction when D equal rank of data matrix
D=97;
xtrunc_97=U(:,1:D)*A(1:D,k);
error_97=norm(x_tilda(:,k)-xtrunc_97);

s5 = strcat( 'Truncation at D = 5; Column: ',num2str(k));
s6 = strcat( 'Truncation at D = 20; Column: ',num2str(k));
s7 = strcat( 'Truncation at D = 40; Column: ',num2str(k));
s8 = strcat( 'Truncation at D = 60; Column: ',num2str(k));


%Subplots
subplot(2,2,1), imagesc(reshape(xtrunc_5,120,160)), colormap(gray);
title(s5),  xlabel(sprintf('Relative Error: %d', error_5))

subplot(2,2,2), imagesc(reshape(xtrunc_20,120,160)), colormap(gray);
title(s6),  xlabel(sprintf('Relative Error: %d', error_20))

subplot(2,2,3), imagesc(reshape(xtrunc_40,120,160)), colormap(gray);
title(s7),  xlabel(sprintf('Relative Error: %d', error_40))

subplot(2,2,4), imagesc(reshape(xtrunc_60,120,160)), colormap(gray);
title(s8),  xlabel(sprintf('Relative Error: %d', error_60))
```

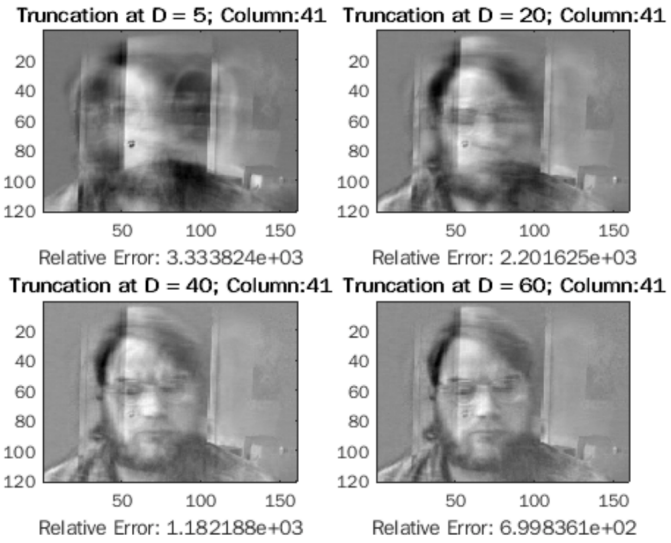

Figure 4 – Output of using transpose of Eigenvectors times Entire Data set .

```
lambda = diag(Sigma(1:109,1:109));
lambda_new = lambda/lambda(90);
index = 1:109;
figure
plot(index,lambda_new)
title( '\lambda_i/ \lambda_{max}')
```

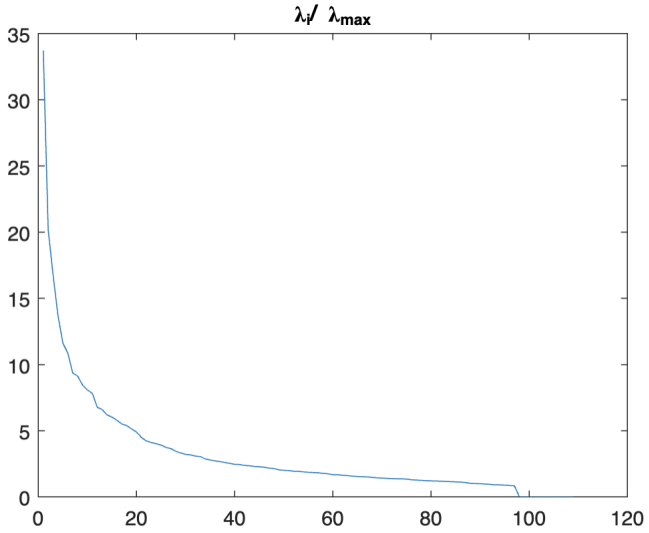

Figure 5 – Output of using Eigenvalues ratios against spectral radius.

The idea of this snapshot method is to use a subset of the data and avoid overloading our compiler with excessive data. This method uses an eigen vector as basis. We calculate our left singular values of our mean subtracted data ensemble. It was found that rank of 97 gives the most info, which is very close to the original rank of the data set. Ultimately, a basis that was able to recreate the images without using the entire data set was found

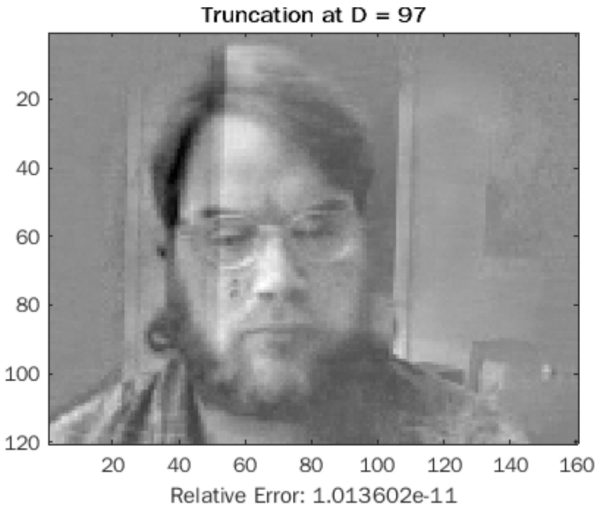

Figure 6 – Truncation at 97.

2.1 Problem 2: Principal Component Analysis

Here our data set is a column vector with 10 elements each of which is an integer from 1-10. Each integer has a 50 images. The idea is to create a basis for our 50 images then test our basis by projecting test images onto that data set and see how accurate the algorithm is in predicting the integer. This accuracy is based on norm of original to projected image.

We being with the mean subtracted ensemble before being projected. The following are the results of multiple numbers. The integer that was not recognized was 8. It is shown that the recreation of the original image has a lot of noise. This could be due to other numbers have straight bars, like 4 , 7, and 1. 9 has a tail. The best integer to compare this with would be 6 as it is curvy like 8; even so, it is missing the top part of 8.

```
load('Digits.mat');

G=double(Gallery);%rank(G) = 500
P=double(Probe);  %rank(P) = 10


errors = zeros(10,10);
pred = zeros(10,1);
A = zeros ( size(Gallery,1) ,50 ,10);

A_avg = zeros ( size(Gallery,1), 10);
A_tilda = zeros( size(Gallery,1) , 50 );
I = zeros ( 10 , 1);

for i = 1:10
   A ( :,: ,i)          =  G(:,((i-1)*50+1):(50*i));

   A_avg (:,i)          =  mean( A(:,:,i), 2);

   A_tilda(:,:,i)       =  A(:,:,i) - repmat(A_avg(:,i) ,1,50);

   [U(:,:,i) ,S,V]       =    svd(A_tilda(:,:,i),0);

   D           =  diag(S).^2;
   CE          =  cumsum(D)/sum(D);
   I(i)           =  find(CE >.99,1) ;
end

for i = 1:10
    for j = 1:10
        p           =  P(:,i)   - A_avg(:,j);
        y           =  U(:, 1:I(j) ,j) * U(:,1:I(j),j)' * p;

        errors(i,j)=  norm( P(:,i) - ( y  +  A_avg(:,j) ) );
    end

   [~,pred(i)]=min(errors(i,:));
end


P8_sub8 = P (: ,9)  - A_avg(: ,9);
P8_sub9 = P (: ,10) - A_avg(: ,10);

y8 = U( :, 1: I(9), 9  )   * U(:,  1: I(9), 9  )' * P8_sub8 ;

y9 = U( :, 1: I(10), 10)   * U(:,  1: I(10), 10  )' * P8_sub9 ;

Y8 = y8 + A_avg (: ,9);
Y9 = y9 + A_avg (: ,10);

figure (1)
subplot (1 ,3 ,1)
imagesc ( reshape ( P (: ,9) ,32 ,32)) , colormap ( gray ) , axis off
subplot (1 ,3 ,2)
imagesc ( reshape ( Y8 ,32 ,32)) , colormap ( gray ) , axis off
```

```
subplot (1 ,3 ,3)
imagesc ( reshape ( Y9 ,32 ,32)) , colormap ( gray ) , axis off


P7_sub7 = P (: ,8)  - A_avg(: ,8);
P7_sub6 = P (: ,9) - A_avg(: ,9);

y7 = U( :, 1: I(8), 8  )   * U(:,  1: I(8), 8  )' * P7_sub7 ;

y_8 = U( :, 1: I(9), 9)   * U(:,  1: I(9), 9  )' * P7_sub6 ;

Y7 = y7 + A_avg (: ,8);
Y_8 = y_8 + A_avg (: ,9);

figure (2)
subplot (1 ,3 ,1)
imagesc ( reshape ( P (: ,8) ,32 ,32)) , colormap ( gray ) , axis off
subplot (1 ,3 ,2)
imagesc ( reshape ( Y7 ,32 ,32)) , colormap ( gray ) , axis off
subplot (1 ,3 ,3)
imagesc ( reshape ( Y_8 ,32 ,32)) , colormap ( gray ) , axis off


P6_sub6 = P (: ,7)  - A_avg(: ,7);
P6_sub8 = P (: ,9) - A_avg(: ,9);

y6 = U( :, 1: I(7), 7  )   * U(:,  1: I(7), 7  )' * P6_sub6 ;

y__8 = U( :, 1: I(9), 9)   * U(:,  1: I(9), 9  )' * P6_sub8 ;

Y6 = y6 + A_avg (: ,7);
Y__8 = y__8 + A_avg (: ,9);

figure (3)
subplot (1 ,3 ,1)
imagesc ( reshape ( P (: ,7) ,32 ,32)) , colormap ( gray ) , axis off
subplot (1 ,3 ,2)
imagesc ( reshape ( Y6 ,32 ,32)) , colormap ( gray ) , axis off
subplot (1 ,3 ,3)
imagesc ( reshape ( Y__8 ,32 ,32)) , colormap ( gray ) , axis off

 pred
```

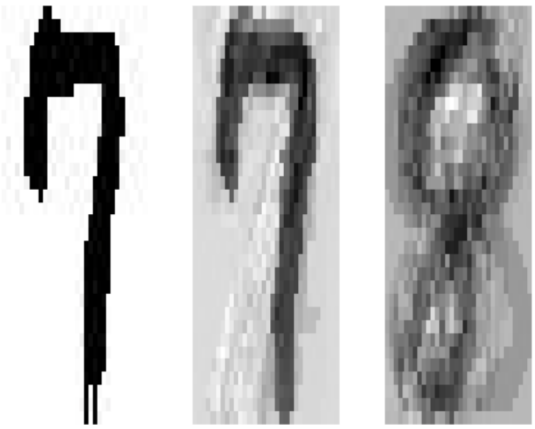

Figure 7 – Original Image , Probe projected onto 7, Probe projected on to 8 .

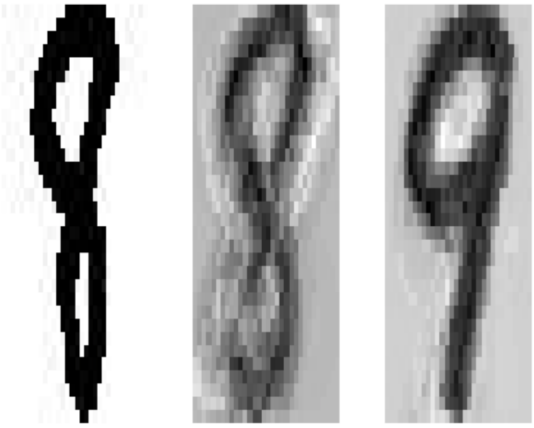

Figure 8 – Original Image , Probe projected onto 8, Probe projected on to 9 .

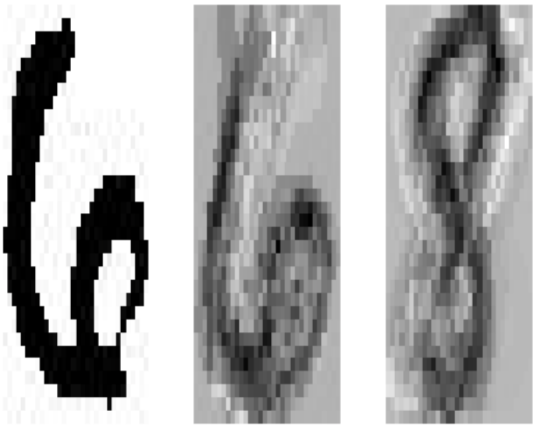

Figure 9 – Original Image , Probe projected onto 6, Probe projected on to 8 .