ECE 311 Lab 6

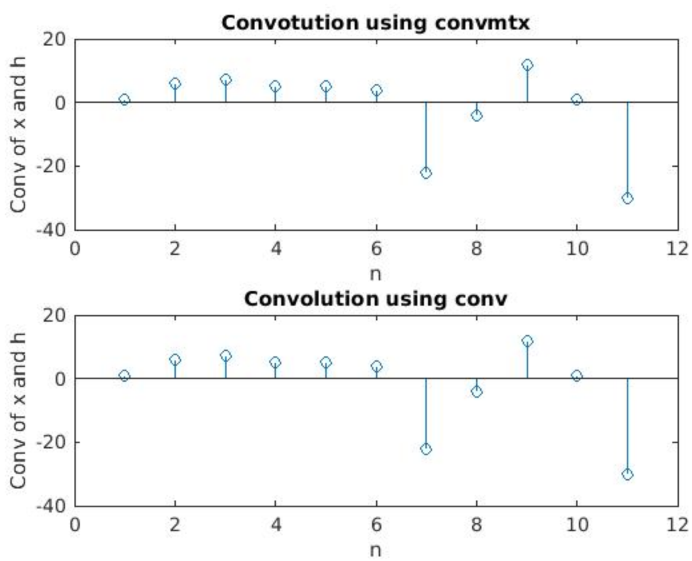
Jacob Hutter

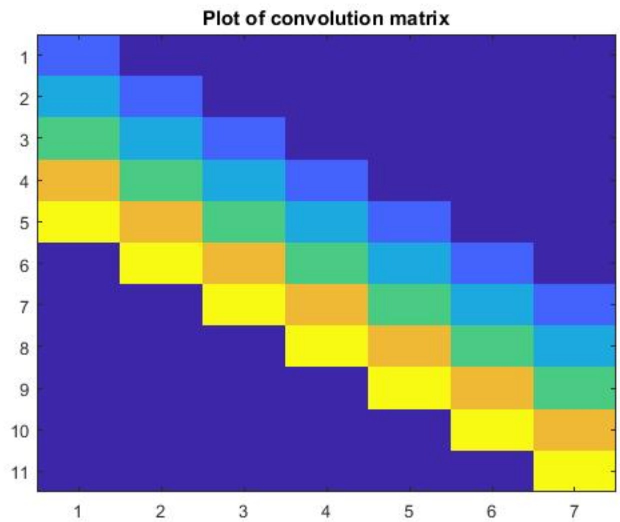
April 25, 2017

Report Item 1

```
x = [ 1 4 -4 -3 2 5  -6]';
h = [1 2 3 4 5]';
a = convmtx(h,length(x));
figure;
imagesc(a);
title('Plot of convolution matrix');

cx = a*x;
c = conv(x,h);

figure;
subplot(211);
stem(cx);
title('Convotution using convmtx');
ylabel('Conv of x and h');
xlabel('n');
subplot(212);
stem(c);
title('Convolution using conv');
ylabel('Conv of x and h');
xlabel('n');
```

report1.m





Report Item 2 $A = U\Sigma V^H$
$A^H A = V\Sigma^H U^H U \Sigma V^H$
$= V\Sigma^H \Sigma V^H$
$A^H A V = V\Sigma^H \Sigma = V\Sigma^2$

Report Item 3

```
clear all;
clc;
A = [1,4,-2; 3,11,5; 7,7,7];
AH = A';
AHA = AH*A;
AAH = A*AH;

[V1,D1] = eig(AAH);
[V2,D2] = eig(AHA);

[U3,S3,V3] = svd(A);

A*AH*U3 - U3*S3^2 % formula given, gives zero matrix

AH*A*V3 - V3*S3^2 % zero matrix returned
```

report2.m

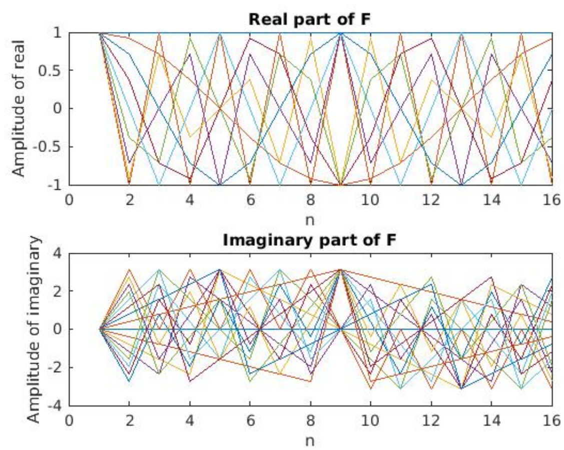
```
ans =

   1.0e-12 *

   -0.0142   -0.0107   -0.0009
    0.0853    0.0018   -0.0027
    0.1137    0.0178    0.0036


ans =

   1.0e-13 *

    0.9948    0.0888    0.0355
    0.8527   -0.2132    0.1171
    0.8527         0    0.0222
```

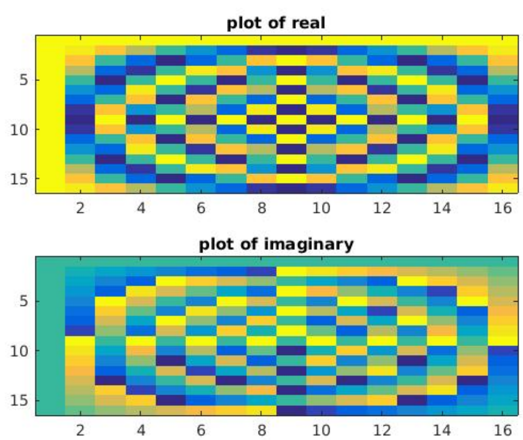
As you can see the resulting matricies are essentially all zeros.

Report Item 4

```
clc;
clear all;
x = [1 1 4 -4 -3 2 5 -6 3 2 4 -2 5 9 -8 4]';
F = dftmtx(length(x));
X = F*x;
r = real(F);
a = angle(F);
figure;
subplot(211);
plot(r);
title('Real part of F');
ylabel('Amplitude of real');
xlabel('n');
subplot(212);
plot(a);
title('Imaginary part of F');
xlabel('n');
ylabel('Amplitude of imaginary');

figure;
subplot(211);
imagesc(r);
title('plot of real');
subplot(212);
imagesc(a);
title('plot of imaginary');
```

report4.m



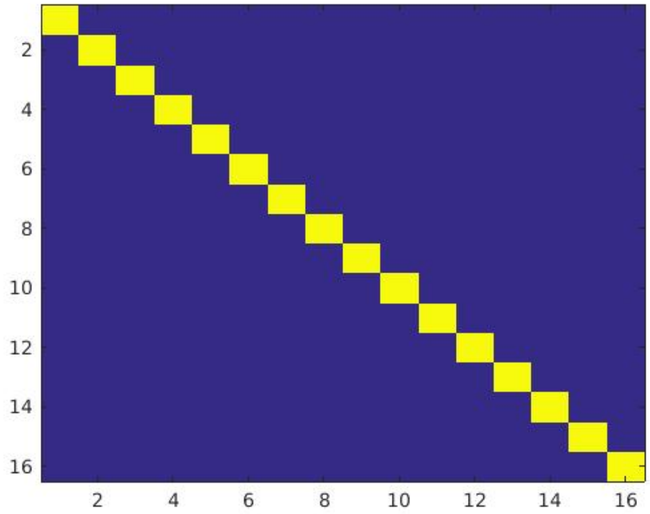


Above you can see that there is a cosine waveform in the real plot and a sine in the imaginary. As the row number goes up, we can see in the third and fourth plots that the frequency increases and then decreases. We can conclude that by dot product rules, the dot product formula is just the implementation of the DFT formula when the signal x is multiplied by the DFT matrix F.

Report Item 5

```
clc;
clear all;
x = [1 1 4 -4 -3 2 5 -6 3 2 4 -2 5 9 -8 4]';
F = dftmtx(length(x));
Fh = (1/length(x))*F';
A = Fh*F;
figure;
subplot(211);
plot(abs(A));
subplot(212);
plot(angle(A));

figure;
imagesc(abs(A));
```

report5.m



You can see by the plot I have included that the resulting relationship shows that they are orthonormal and orthoganol.

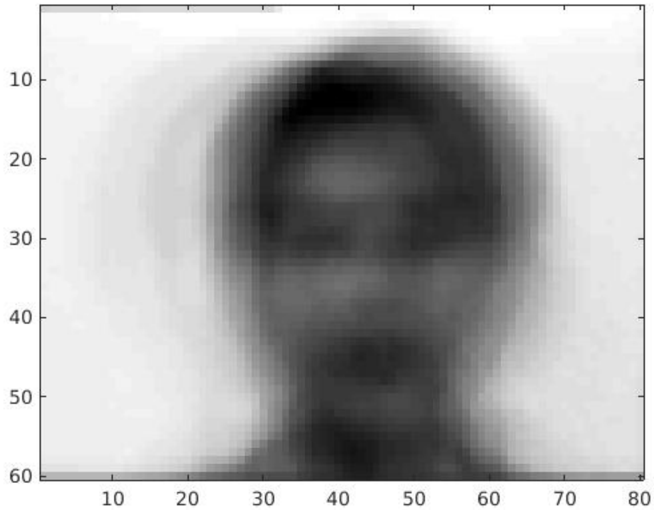
Report Item 6

```
clc;
clear all;
X = loadImages('yalefaces');
Y = compMeanVec(X);
Z = reshape(Y,[60,80]);
imagesc(Z);
colormap gray
```

report6.m

```
function [ Y ] = compMeanVec( X )
[height,width] = size(X);
sum = zeros(1,width);
for i=1:height
    sum = sum + X(i,:);
end
sum = sum/height;
Y = sum;
end
```

compMeanVec.m

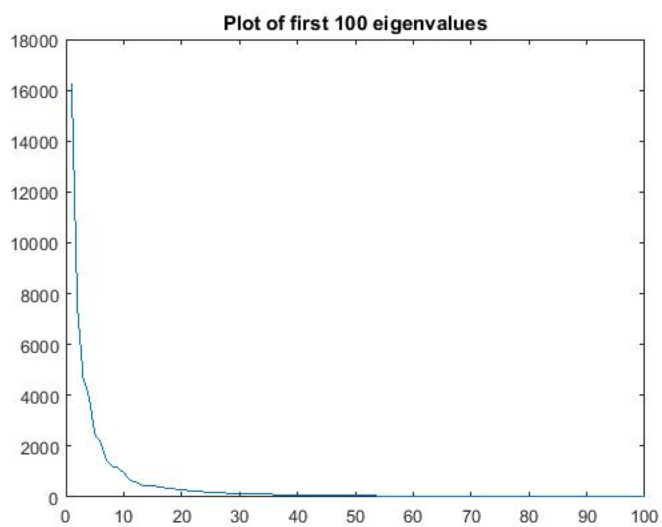


After the mean vector has been computed, the resulting image is the average of all of the images inside yalefaces.

Report Item 7

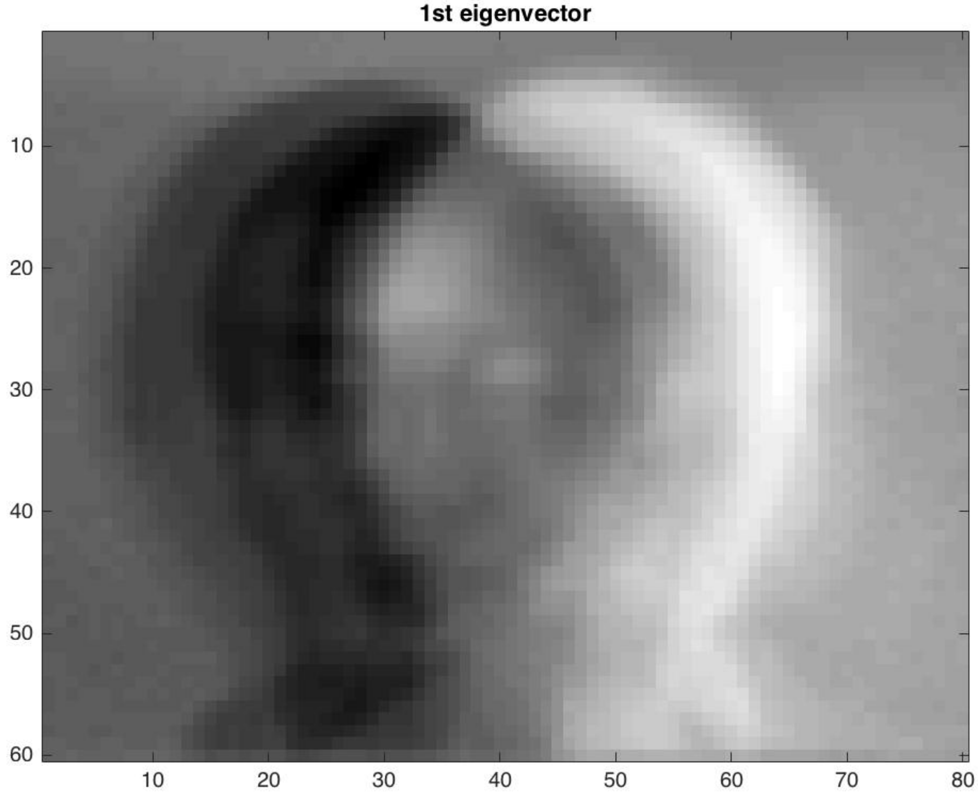
```
clc;
clear all;
X= loadImages('yalefaces');
Y = compMeanVec(X);
X_hat = zeros(165,4800);
for i=1:165
    X_hat(i,:) = X(i,:) - Y;
end
R = (X_hat')*(X_hat);
[U,S,V] = svd(R);
s = svd(R);
figure;
s = s(1:100,1);
plot(s);
title('Plot of first 100 eigenvalues');


figure;
imagesc(reshape(U(:,1),[60,80]));
colormap gray
title('1st eigenvector');

figure;
imagesc(reshape(U(:,2),[60,80]));
colormap gray
title('2nd eigenvector');

figure;
imagesc(reshape(U(:,3),[60,80]));
colormap gray
title('3rd eigenvector');

figure;
imagesc(reshape(U(:,4),[60,80]));
colormap gray
title('4th eigenvector');

figure;
imagesc(reshape(U(:,50),[60,80]));
colormap gray
title('50th eigenvector');

figure;
imagesc(reshape(U(:,100),[60,80]));
colormap gray
title('100th eigenvector');
```

report7.m

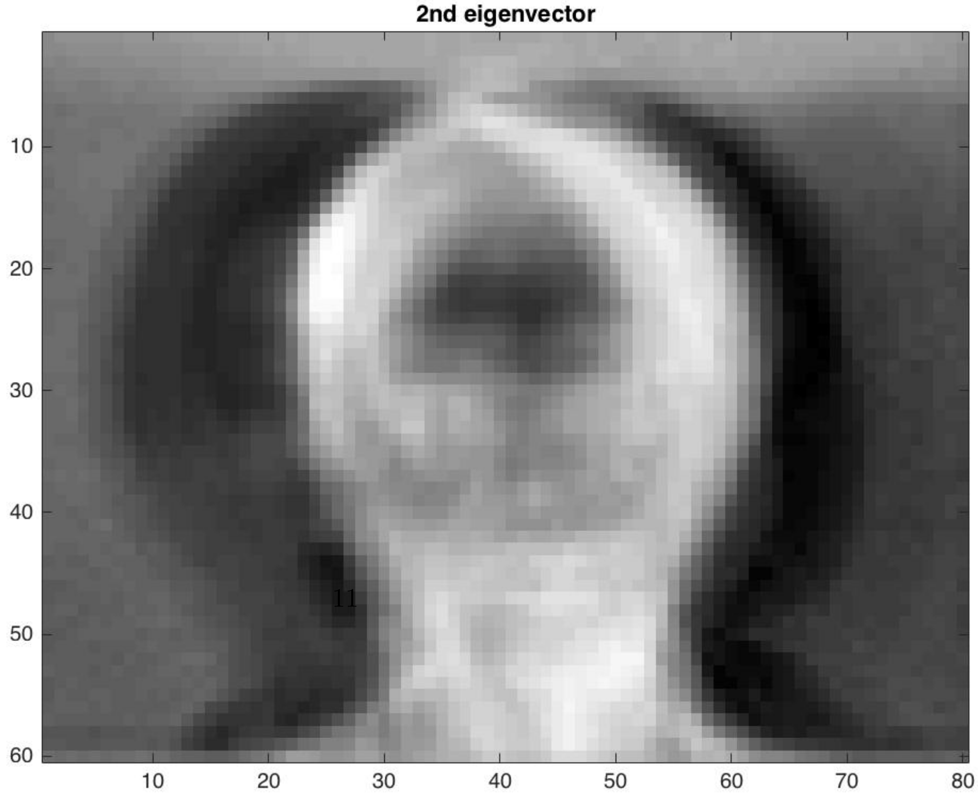


By the eigenvalue plot, the values sharply die off after the 20th value or so.

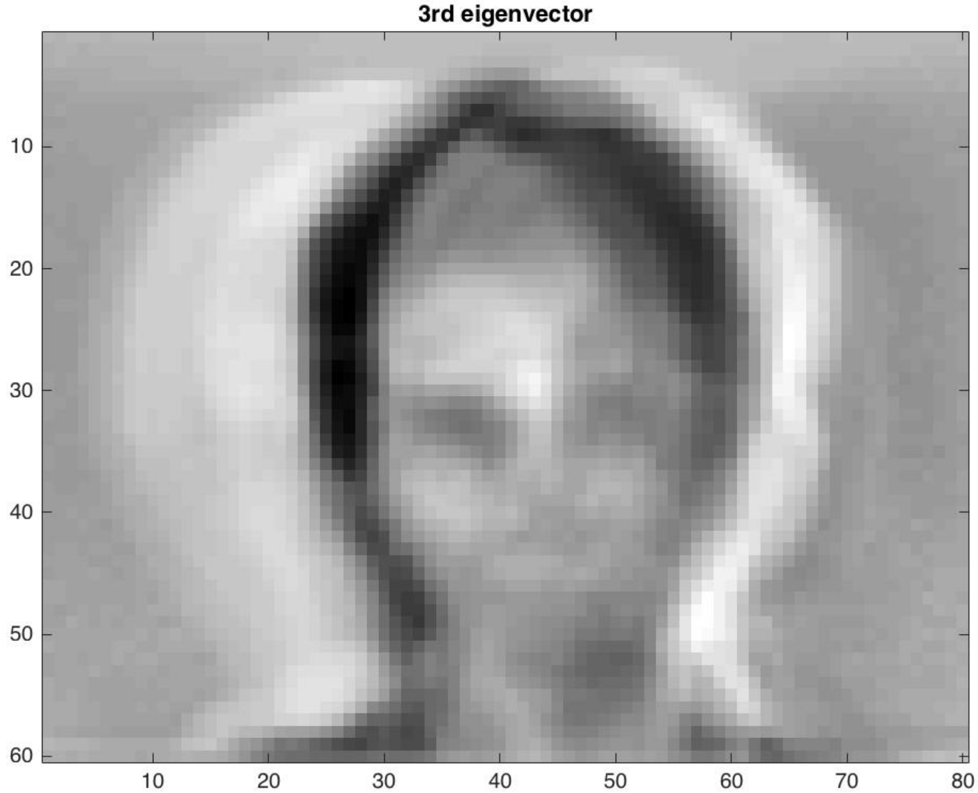
1st eigenvector



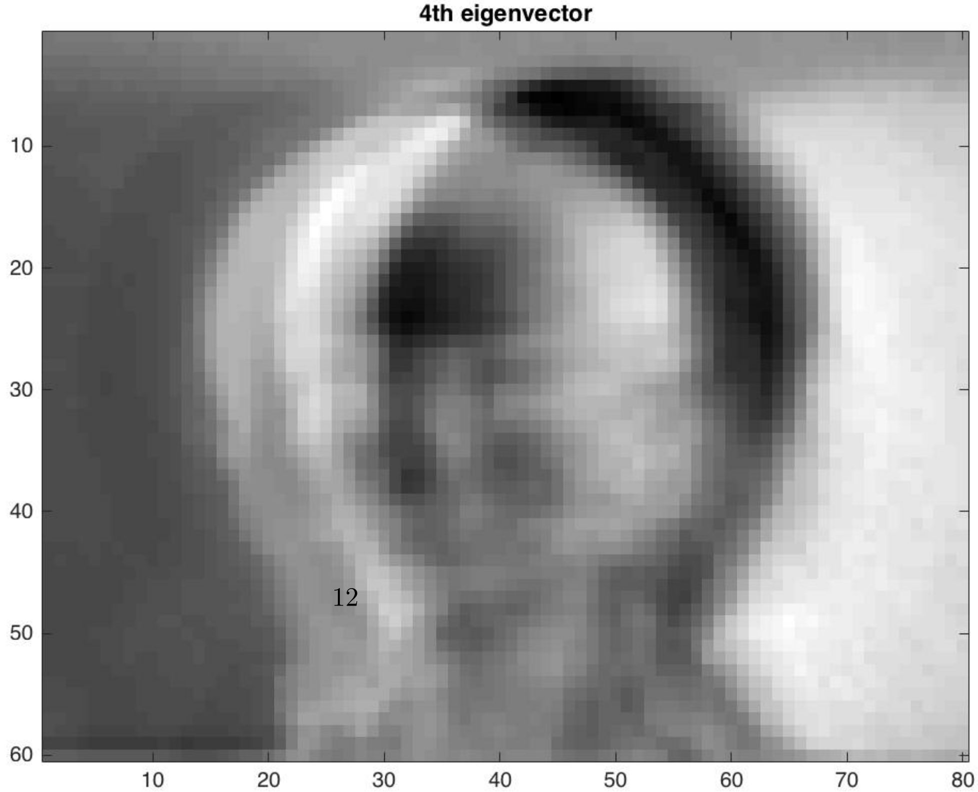
2nd eigenvector



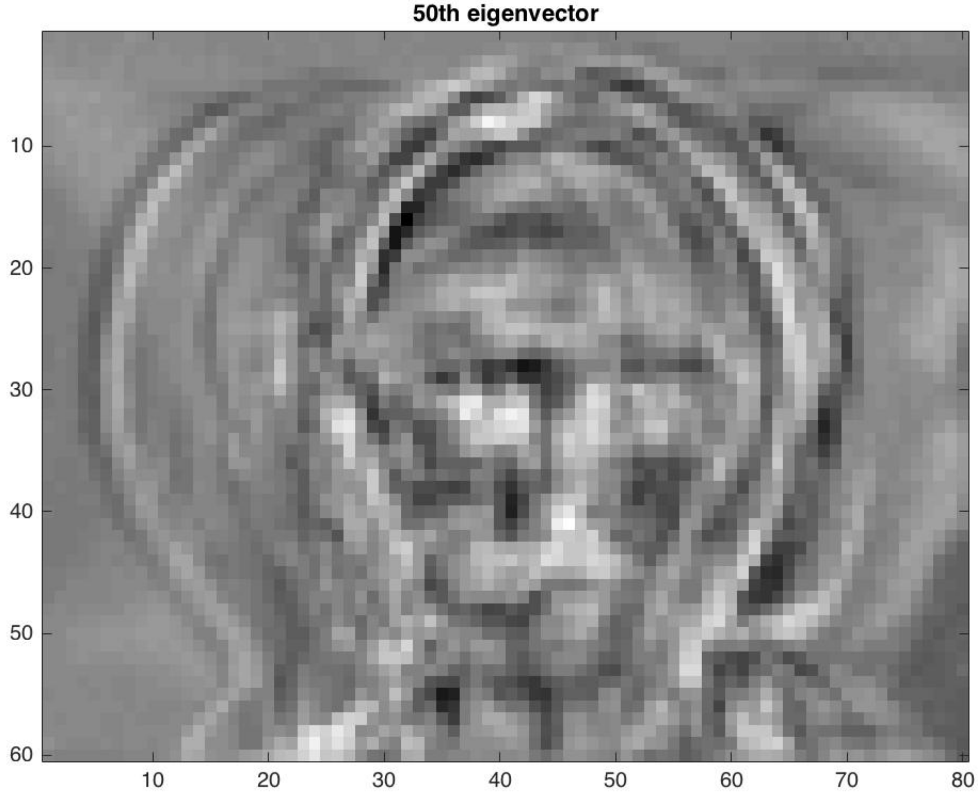
3rd eigenvector



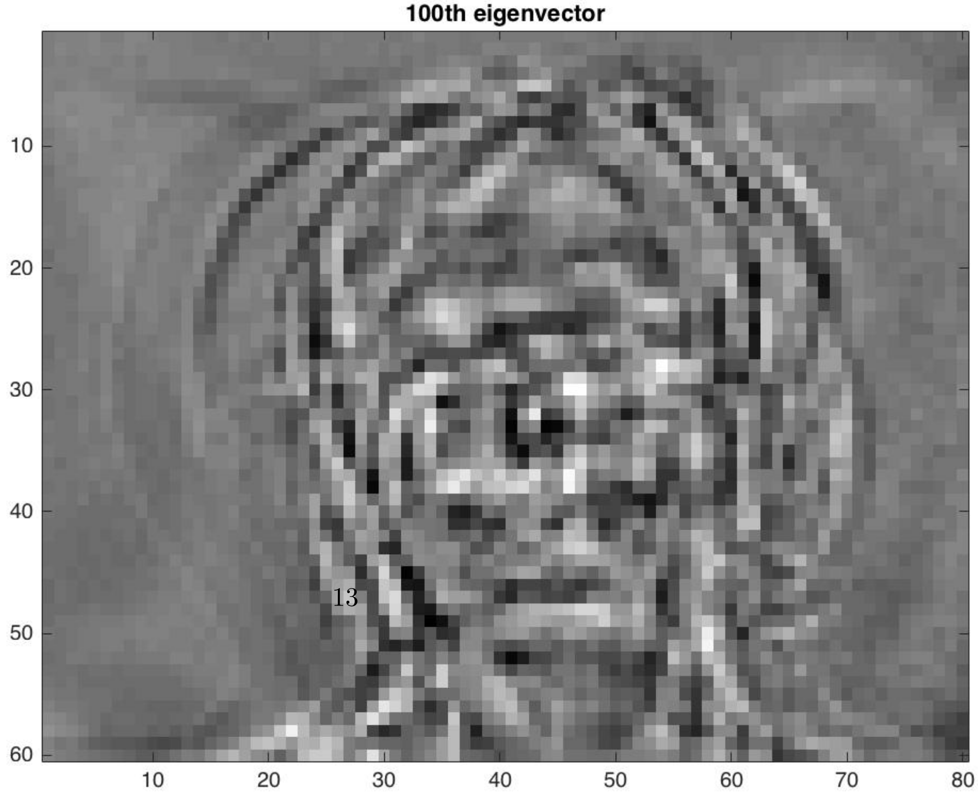
4th eigenvector



50th eigenvector



100th eigenvector



Report Item 8

Below are the implementations for PCAtransform and invPCAtransform

```
function [ x_pca ] = PCAtransform( Ux, V, x_orig )
x_hat = x_orig - Ux;
x_pca =  x_hat * V;
end
```

PCAtransform.m

```
function [ x_orig ] = invPCAtransform( Ux, V, x_pca )


x_orig = x_pca * V';
x_orig = x_orig + Ux;

end
```

invPCAtransform.m

Report Item 9

```
clc;
clear all;
X = loadImages('yalefaces');
Y = compMeanVec(X);
Ux = Y;
X_hat = zeros(165,4800);
for i=1:165
    X_hat(i,:) = X(i,:) - Y;
end
R = (X_hat')*(X_hat);
[U,S,V] = svd(R);

%%%%%%%%%% get U and Ux
A = imread('noisy_face.png');
A = im2double(A); % convert integer precision to double precision
    for mean
A = reshape(A,[1,4800]);

% start of PCA transform
A_pca = PCAtransform(Ux,U,A);
% end of PCA transform

A_pca(1,100:4800) = 0; % limit noise

%%%%%%%%%%%%%obtained pca version%%%%%%%%%%%%%%
%start of inv PCA transform
A_orig = invPCAtransform(Ux,U,A_pca);
%end of inv PCA transform
figure;
imagesc(reshape(A_orig,[60,80]));
colormap gray
```

report8.m