# AMATH482: Yale Faces and Music Classification

Shuying Zhang
email: shuying3@uw.com

**Abstract**

This projects explores the application of SVD on face recognition. This project also includes experiments of supervised learning in MatLab to classify music pieces with different genres from various artists.

## 1 Introduction and Overview

As discussed before, SVD is a powerful technique in linear algebra. We have already explored the application of SVD in principle components analysis. In this project, we will take a look at how SVD works in face recognition. The given data are divided into two parts: cropped and uncropped. We will discuss the difference between them as well.

In the second part of the project, we will do experiments on supervised learning by classify short music clips. The 5 seconds music clips are generated from various artists.

There are five sections in this report. The introduction and overview section gives a concise description of the topics in the report. The theoretical part provides background knowledge for SVD and Linear Discriminant Analysis (LDA). In algorithm implementation and development, we will introduce how to process images and audios data in MATLAB and analyze them. The computational results shows SVD results of analyzing the faces and the prediction of test music clips. A summary is concluded in summary and conclusions. All the MATLAB code and related MATLAB commands are in appendix. One can also take a look at the sample music in appendix.

## 2 Theoretical Background

We will briefly review the basic idea behind SVD here again. Some descriptions and figures are from the class notes [1].

## 2.1 Singular Value Decomposition (SVD)

SVD is a factorization of matrix into three components and use them in many applications. The full SVD takes the form

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* \tag{1}$$

where

$$\begin{aligned} &\mathbf{U} \in C^{m\times m} \text{ is unitary} \\ &\mathbf{V} \in C^{n\times n} \text{ is unitary} \\ &\mathbf{\Sigma} \in R^{m\times n} \text{ is diagonal} \end{aligned} \tag{2}$$

The diagonal entries of $\Sigma$ has the property that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$ where $p = \min(m, n)$. We can also do diagonalization via SVD.

In general, one can apply SVD on every matrix. To compute SVD, in MATLAB, use command `[U,S,V] = svd(A)`.

## 2.2 Classification: Linear Discriminant

LDA is a standard method for classification developed by Fisher in 1936. The algorithm finds a linear combination of features that separate each class. Figure 1 is an illustration of LDA from class notes. The goal of LDA is to find a suitable projection that maximizes the distance between the inter-class data while minimizing the intra-class data.

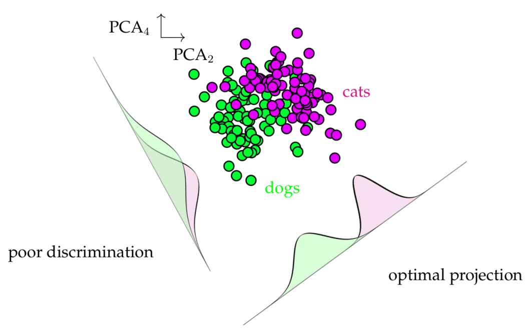


Figure 1: Illustration of LDA

To perform LDA, we need training sets to make the classifier learn the key features of each class by specifying labels. Then we test the classifier with our test sets. In MatLab, use command `class = classify(test, train, labels)`.

There is also a nonlinear discriminant algorithm called Quadratic Discriminant Analysis(QDA).

# 3 Algorithm Implementation and Development

## 3.1 Yale Faces

For both the uncropped and cropped image data, the analyzing process only differs in loading data from files as cropped data are located in multiple folders. We introduce the general steps here.

**Stack Data** Load every image and reshape the data into columns to form a matrix in which every column represents an image. Besides, in order to save memory and run time, we resize each image in to 60 * 40 pixels.

**Perform SVD** Subtract mean from the data and use `svd` command.

**Plot** Before we plot U and V, we need to reshape their columns back into matrix to visualize each image. Plot U, V and reconstruct images.

Notice, when plotting U, we use `flipud` command to make the image upright.

## 3.2 Music Classification

In these three tests, we are training the classifier with different data sets. The procedure of extracting music clips, forming training and testing sets, and performing classification are similar. We just show general steps here.

**Extracting Samples** We choose the relevant music collections online and download them using a plug in called `youtube-dl`. All the music data are used in this project only.

The music pieces have duration vary from 40 minutes to 90 minutes. For the pieces shorter than 60 minutes, we extract a 5-second clip every 20 seconds starting from 60 seconds. For longer pieces, we extract a 5-second clip every 30 seconds starting from 60 seconds.

Since the music signal are stereo, we turn them into mono data by taking the average.

**Stack Data and Perform SVD** Before we stack data into matrices, we first use `spectrogram` command to enhance the quality of the classification.

After short Fourier transformation, we reshape the data into columns and stack all the music clips together.

Perform SVD, use the absolute value of matrix V to do classification.

**Apply Linear Discriminant Analysis** We need to prepare the training sets and the test sets, since we have 100 samples, we randomly choose 80 of them to work as training sets, the rest 20 are test cases.

**Analyzing Results** In order to make results more representative, we run the experiments twice.

# 4 Computational Results

## 4.1 Yale Faces

After we decompose the image matrix into $\mathbf{U}$, $\Sigma$, $\mathbf{V}$. We can start doing SVD analysis. Here, $\mathbf{U}$ is a matrix and each column stores a feature or mode of the face space. $\Sigma$ contains the singular values corresponding to each mode. $\mathbf{V}$ tells us how each image is projected on those modes.

### 4.1.1 Cropped Faces

Figure 2 shows the main features on human faces. We are able to see features like eyes, nose and the shape of the face.

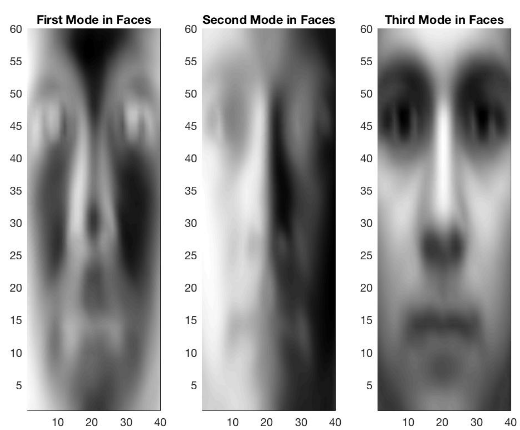


Figure 2: First Few Modes on Human Faces

To reconstruct the face, we try different rank r according to Figure 3. It tells us that the modes after 200 are almost zero. Hence, we may be able to use only the first 200 features to reconstruct a human face.

Figure 4 is the ranks we have tested. We could see that with 50 to 100 modes, we could construct a good image for face recognition.

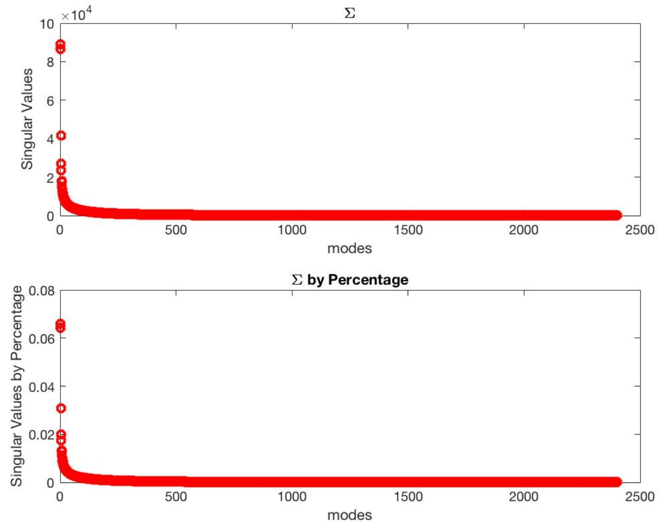


Figure 3: Singular Values of Each Node

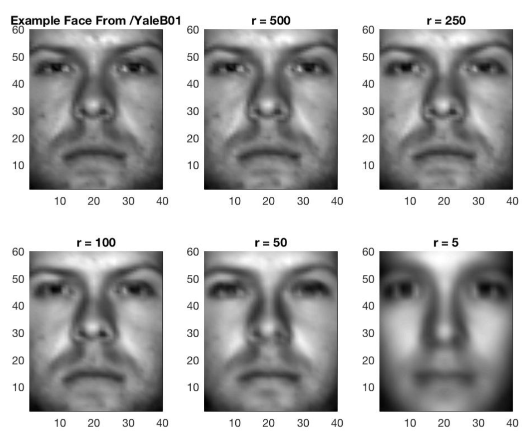


Figure 4: Reconstructed Faces by Differnet Rank r

#### 4.1.2 Uncropped Images

Figure 5 shows the main features on human faces. We are only able to see features like the shape of the face and the neck.

To reconstruct the face, we try different rank r according to Figure 6. It tells us that the modes after 80 are almost zero. Hence, we may be able to use only the first 80 features to reconstruct a human face.

Figure 7 is the ranks we have tested. We could see that with 75 to 100 modes, we could construct a good image for face recognition.

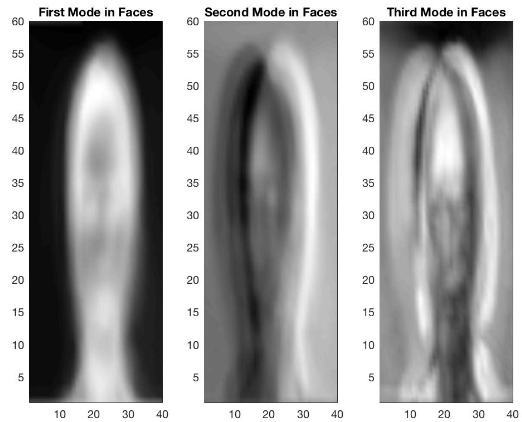


Figure 5: First Few Modes on Uncropped Human Faces

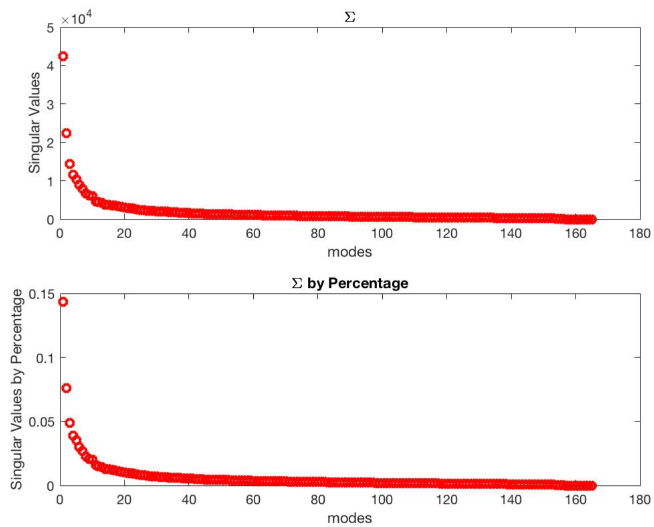


Figure 6: Singular Values of Each Node on Uncropped Faces

#### 4.1.3 Comparison

The cropped images contain ten times more features than uncropped (over 2000 versus less than 200). To construct a face, only 5 % to 10 % of the features (50 to 100 modes out of 2000 modes) on cropped faces are needed, whereas, a uncropped face requires more than half its features (75 to 100 modes out of 160 modes).

### 4.2 Music Classification

A list of music used in three tests can be found in appendix. The 5-sec music clip examples are in Figure 8.

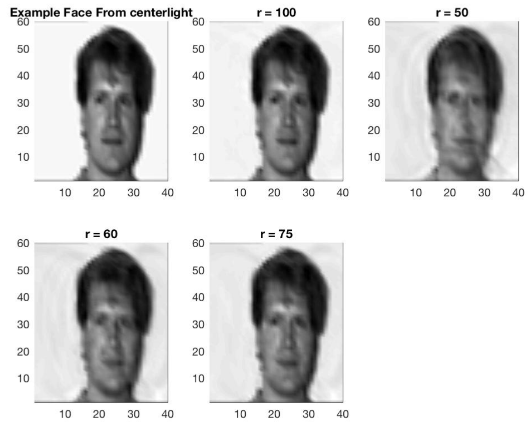


Figure 7: Reconstructed Faces by Differnet Rank r

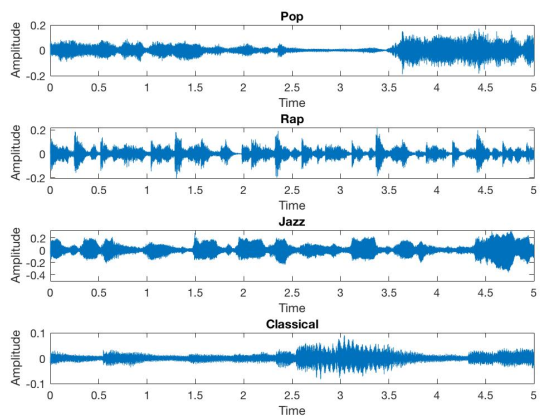


Figure 8: 5-second Music Clip Examples With Different Genres

#### 4.2.1 Test1: Band Classification

For test1, we choose three music artists with different genres: Michael Jackson(Pop), Louis Armstrong (Jazz), and Yiruma (Contemporary classic). After performing LDA twice, the results are in Figure 9 and Figure 10.

Base on two trials, the correctness rate for these three artists are ranging from 50 % to 80 %, which is within the tolerance since we are only dealing with 80 training sets. It brings my attention that Louis Armstrong and Yiruma are commonly misclassified. One naive guess about this phenomenon is that Yiruma is a pianist, and in jazz band, piano is a very important instrument. This shared feature may cause error.

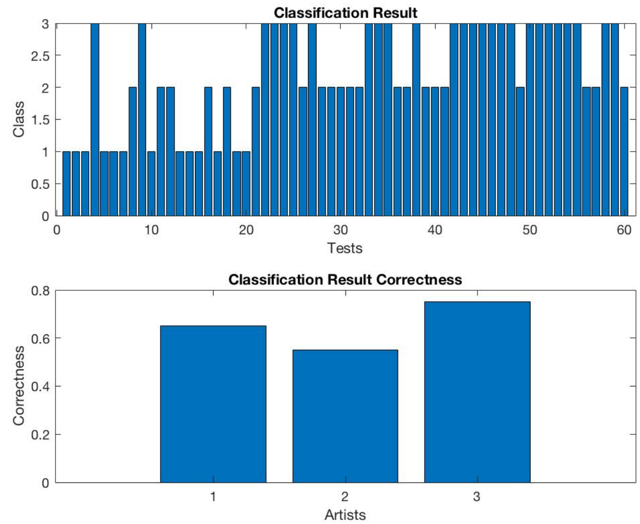


Figure 9: Trail 1: Band Classification Result and its Correctness; Top: first 10 bars Michael Jackson(Pop) class = 1, second 10 bars Louis Armstrong (Jazz) class = 2, third 10 bars Yiruma (Contemporary Classics) class = 3

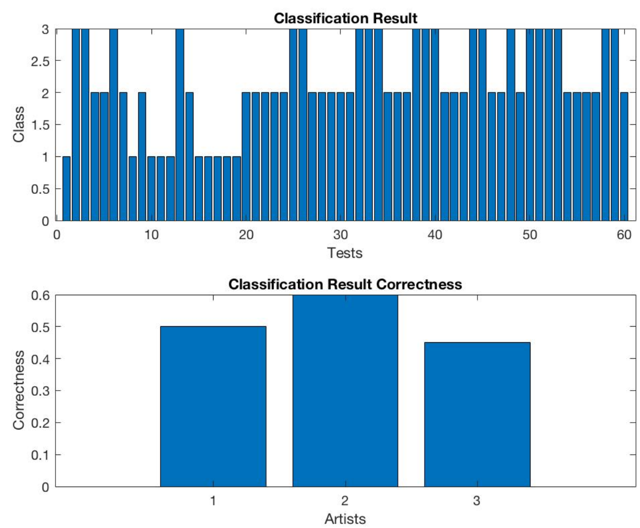


Figure 10: Trail 2: Band Classification Result and its Correctness; Top: first 10 bars Michael Jackson(Pop) class = 1, second 10 bars Louis Armstrong (Jazz) class = 2, third 10 bars Yiruma (Contemporary Classics) class = 3

## 4.3 Test 2: Band From Seattle Classification

For test2, we choose three music artists with same genre and from the same geographic area: Seattle. Hence, their music styles may share high similarity. They are: Soundgarden, Alice In Chains, and Pearl Jam. After performing LDA twice, the results are in Figure 11 and

Figure 12.

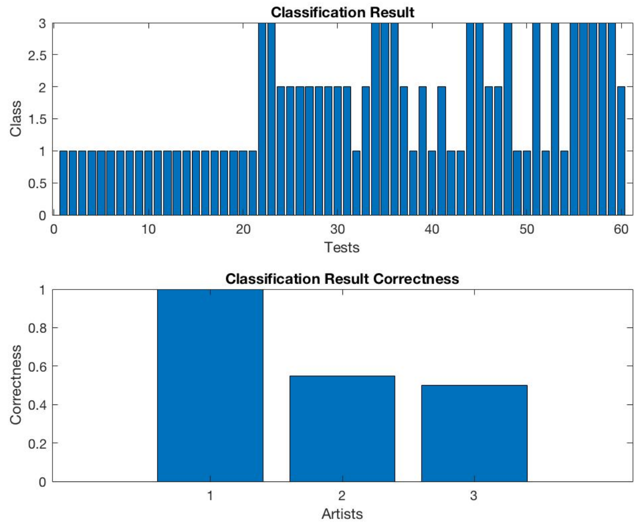


Figure 11: Trail 1: Seattle Band Classification Result and its Correctness; Top: first 10 Soundgarden class = 1, second 10 bars Alice In Chains class = 2, third 10 bars Pearl Jam class = 3

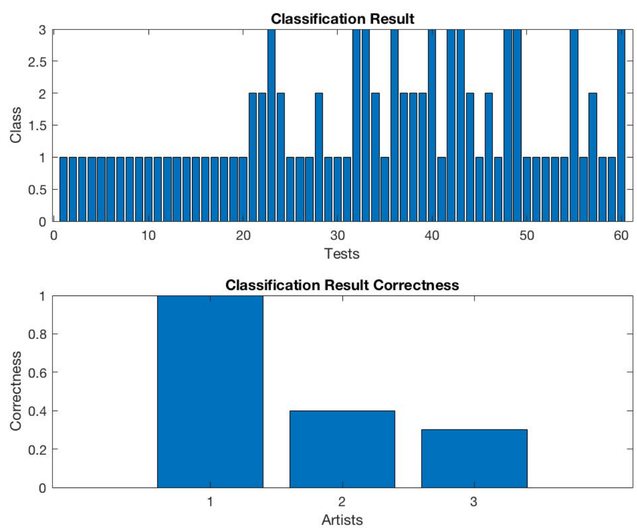


Figure 12: Trail 2: Seattle Band Classification Result and its Correctness; Top: first 10 Soundgarden class = 1, second 10 bars Alice In Chains class = 2, third 10 bars Pearl Jam class = 3

Based on the two trials, most of the sample clips are identified as class 1, which is Soundgarden. The correctness rate for soundgarden is 100 %, and the other two are less than 50 %. For this test, the result is a little frustrating. This is where we can improve in the future. For example, try larger number of samples. Now, It's difficult to make any conclusion.

## 4.4 Test 3: Genre Classification

For test3, we choose music collections of three music genres: Rap, Pop, and Classic. In those three genres, classic is more distinguishable. However, Rap and Pop may have some features overlapped because pop music is influenced by many music styles. After performing LDA twice, the results are in Figure 13 and Figure 14.

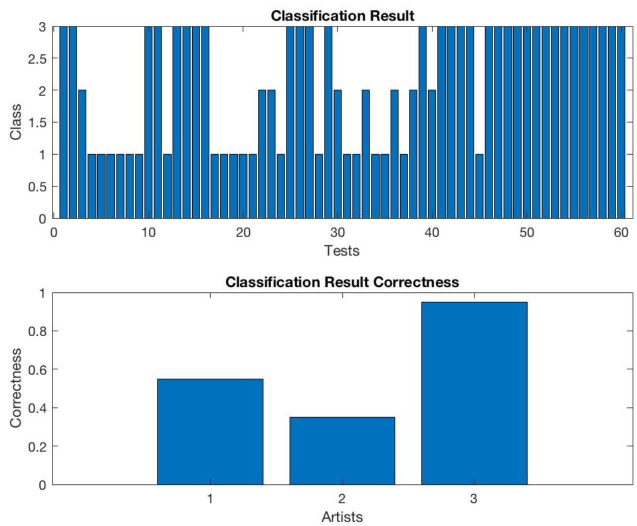


Figure 13: Trail 1: Genre Classification Result and its Correctness; Top: first 10 Rap class = 1, second 10 bars Pop class = 2, third 10 bars Classic class = 3

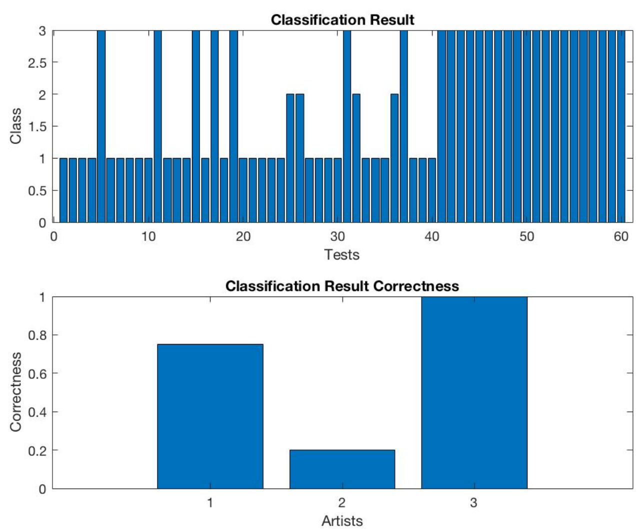


Figure 14: Trail 2: Genre Classification Result and its Correctness; Top: first 10 Rap class = 1, second 10 bars Pop class = 2, third 10 bars Classic class = 3

Based on the two trials, almost all the classic pieces are correctly classified, which is as expected. The correctness rate for rap is also beyond 50 %, which is acceptable due to the limitation of the number of the samples. Compared with these two classes, the result of classifying pop is not as good as them. The possible reason is stated above.

# 5 Summary and Conclusions

In this project, we explore another powerful application of SVD: Image recognition. It allows us to decompose the image data and reconstruct the images by important features. We can see that not all the modes are required to reconstruct a "good" recognizable image. In our experiments of supervised learning, we try the standard classification methods: LCA on various music data. Overall, the classifier returns expected results. Due to the size of the samples, the classifier doesn't correctly identify the music clips that share similarities. A Larger sample or other supervised learning methods can be applied in the future to improve the result.

# References


[1] Part3 : Computation Methods for Data Analysis. Data-Driven Modeling & Scientific Computation: Methods for Complex Systems Big Data, by J. Nathan Kutz, OUP Oxford, 2013, available at `http://faculty.washington.edu/kutz/582.pdf`.

# A MATLAB commands

Documentation for those commands can be found in MATLAB using `help command_name`.

## A.1 Image/Audio Loading

**imread(filename)** Reads images from files.

**audioread(filename)** Reads audios from files, return it's sample rate in Hertz.

**imresize(img, [X Y])** Resizes a image into given size.

**dir(directory)** Goes to specified directory.

## A.2 Data Modification

**size(X)** Returns the size of X.

**double(X)** Change the data type in X.

**repmat** Replicate arrays.

**svd(X)** find the SVD of matrix X. Returns u, s, v. One can add 'econ' to perform a reduced SVD rather a full SVD.

## A.3 Plotting

**bar** Draws bar plots of given data

**plot(X,Y)** Plots 2-dim data.

**subplot(i,j,k))** Draw multiple plots in one figure. i and j specifies the lay out, k is the index of the plots.

**legend, xlab, ylab, title** Labels figure information.

**pcolor(X)** Plot a matrix with colors.

**flipud(X)** flip the rows upside down.

### A.4 Classification

**classify(sample, xtrain, label)** Returns the class of sample data.

## B MATLAB code

```
%% Yale Faces

%cropped
%load images and stack them as a matrix
img_cropped = [];
for i = 1:39
    if (i <= 9)
     folder = ['CroppedYale/YaleB0' num2str(i) '/'];
    else
     folder = ['CroppedYale/YaleB' num2str(i) '/'];
    end

    files = dir([folder '*.pgm']);
    for j = 1:length(files)
         image = imresize(double(imread([folder files(j).name])), [60 40]);
         image = reshape(image, [],1); %reshape into column
         img_cropped = [img_cropped, image];
     end
end

%% svd
[u,s,v] = svd(img_cropped-mean(img_cropped(:)),'econ');

figure(1)
%plot s in values anf percentage (eigenvalues squared)
subplot(2,1,1)
plot(diag(s),'ro','Linewidth',[2])
xlabel('modes'); ylabel('Singular Values');
title('\Sigma');
subplot(2,1,2)
plot(diag(s)/sum(diag(s)),'ro','Linewidth',[2])
xlabel('modes'); ylabel('Singular Values by Percentage');
title('\Sigma by Percentage');
```

```
figure(2)
%plot u (eigenvectors corresponding eigenvalues)
subplot(1,3,1)
face = reshape(u(:,1),60,40);
pcolor(flipud(face)), shading interp, colormap(gray)
title('First Mode in Faces');
subplot(1,3,2)
face = reshape(u(:,2),60,40);
pcolor(flipud(face)), shading interp, colormap(gray)
title('Second Mode in Faces');
subplot(1,3,3)
face = reshape(u(:,3),60,40);
pcolor(flipud(face)), shading interp, colormap(gray)
title('Third Mode in Faces');

%%
figure(3)
%%reconstruct
subplot(2,3,1)
pcolor(flipud(reshape(img_cropped(:,1),60,40))), shading interp, colormap(gray)
title('Example Face From /YaleB01');
subplot(2,3,2)
fr500 = u(:,1:500)*s(1:500,1:500)*v(:,1:500).';
pcolor(flipud(reshape(fr500(:,1),60,40))), shading interp, colormap(gray)
title('r = 500');
subplot(2,3,3)
fr250 = u(:,1:250)*s(1:250,1:250)*v(:,1:250).';
pcolor(flipud(reshape(fr250(:,1),60,40))), shading interp, colormap(gray)
title('r = 250');
subplot(2,3,4)
fr100 = u(:,1:100)*s(1:100,1:100)*v(:,1:100).';
pcolor(flipud(reshape(fr100(:,1),60,40))), shading interp, colormap(gray)
title('r = 100');
subplot(2,3,5)
fr50 = u(:,1:50)*s(1:50,1:50)*v(:,1:50).';
pcolor(flipud(reshape(fr50(:,1),60,40))), shading interp, colormap(gray)
title('r = 50');
subplot(2,3,6)
fr5 = u(:,1:5)*s(1:5,1:5)*v(:,1:5).';
pcolor(flipud(reshape(fr5(:,1),60,40))), shading interp, colormap(gray)
title('r = 5');

%% Uncropped
clear all; close all; clc;

%load images
```

```
img_un = [];
folder = ['yalefaces_uncropped/yalefaces/'];
files = dir([folder 'subject*']);
for j = 1:length(files)
    image = imresize(double(imread([folder files(j).name])), [60 40]);
    image = reshape(image, [],1); %reshape into column
    img_un = [img_un, image];
end

%% svd
[u,s,v] = svd(img_un-mean(img_un(:)),'econ');

figure(1)
%plot s in values anf percentage (eigenvalues squared)
subplot(2,1,1)
plot(diag(s),'ro','Linewidth',[2])
xlabel('modes'); ylabel('Singular Values');
title('\Sigma');
subplot(2,1,2)
plot(diag(s)/sum(diag(s)),'ro','Linewidth',[2])
xlabel('modes'); ylabel('Singular Values by Percentage');
title('\Sigma by Percentage');

figure(2)
%plot u (eigenvectors corresponding eigenvalues)
subplot(1,3,1)
face = reshape(u(:,1),60,40);
pcolor(flipud(face)), shading interp, colormap(gray)
title('First Mode in Faces');
subplot(1,3,2)
face = reshape(u(:,2),60,40);
pcolor(flipud(face)), shading interp, colormap(gray)
title('Second Mode in Faces');
subplot(1,3,3)
face = reshape(u(:,3),60,40);
pcolor(flipud(face)), shading interp, colormap(gray)
title('Third Mode in Faces');

figure(3)
%% reconstruct
subplot(2,3,1)
pcolor(flipud(reshape(img_un(:,1),60,40))), shading interp, colormap(gray)
title('Example Face From centerlight');
subplot(2,3,2)
fr100 = u(:,1:100)*s(1:100,1:100)*v(:,1:100).';
pcolor(flipud(reshape(fr100(:,1),60,40))), shading interp, colormap(gray)
title('r = 100');
```

```
subplot(2,3,3)
fr50 = u(:,1:50)*s(1:50,1:50)*v(:,1:50).';
pcolor(flipud(reshape(fr50(:,1),60,40))), shading interp, colormap(gray)
title('r = 50');
subplot(2,3,4)
fr60 = u(:,1:60)*s(1:60,1:60)*v(:,1:60).';
pcolor(flipud(reshape(fr60(:,1),60,40))), shading interp, colormap(gray)
title('r = 60');
subplot(2,3,5)
fr75 = u(:,1:75)*s(1:75,1:75)*v(:,1:75).';
pcolor(flipud(reshape(fr75(:,1),60,40))), shading interp, colormap(gray)
title('r = 75');

%% music classification
% band classification

clear all; close all; clc;

%load data
mj = [];
la = [];
yiruma = [];

%get sample
info1 = audioinfo('hw4music/michaeljackson.wav');
fs = info1.SampleRate;
for i = 60:30:3045
    [y,~] = audioread('hw4music/michaeljackson.wav',[floor(i*fs)
        floor((i+5)*fs)]);
    y = (y(:,1) + y(:,2))./2;
    y = spectrogram(y);
    mj = [mj, y(:)];
end
%%
la = [];
info2 = audioinfo('hw4music/louisamstrong.wav');
fs = info2.SampleRate;
for i = 60:30:3045
    [y,~] = audioread('hw4music/louisamstrong.wav',[floor(i*fs) floor((i+5)*fs)]);
    y = (y(:,1) + y(:,2))./2;
    y = spectrogram(y);
    la = [la, y(:)];
end
%%
yiruma = [];
info3 = audioinfo('hw4music/Yiruma.wav');
fs = info3.SampleRate;
```

```
for i = 60:30:3045
    [y,~] = audioread('hw4music/Yiruma.wav',[floor(i*fs) floor((i+5)*fs)]);
    y = (y(1:220501,1) + y(1:220501,2))./2;
    y = spectrogram(y);
    yiruma = [yiruma, y(:)];
end

%%
% first 10 col mj, second 10 col la, last 10 col yiruma
spec = [mj,la,yiruma];
[u,s,v] = svd(spec,'econ');

%%
%randomly choose train and test sets
q1 = randperm(100);
q2 = randperm(100);
q3 = randperm(100);

%picking features
x1 = abs(v(1:100,2:4));
x2 = abs(v(101:200,2:4));
x3 = abs(v(201:300,2:4));

%train set
xtrain = [x1(q1(1:80),:); x2(q2(1:80),:); x3(q3(1:80),:)];

%test set
xtest = [x1(q1(81:end),:); x2(q2(81:end),:); x3(q3(81:end),:)];

%label trainning data
labels = [ones(80,1);2*ones(80,1);3*ones(80,1)];
class = classify(xtest,xtrain,labels);

subplot(2,1,1)
%pre = predict(class,xtest);
bar(class)
xlabel('Tests')
ylabel('Class')
title('Classification Result')

c1 = 0;c2 = 0;c3 = 0;
% calculate correctness rate
for i = 1:60
    if (i <= 20 && class(i,1) == 1 )
        c1 = c1 + 1;
    elseif (i <= 40 && i > 20 && class(i,1) == 2)
        c2 = c2 + 1;
```

```
    elseif (i <= 60 && i > 40 && class(i,1) == 3)
        c3 = c3 + 1;
    end
end
c1 = c1/20;
c2 = c2/20;
c3 = c3/20;
subplot(2,1,2)
bar([c1 c2 c3])
xlabel('Artists')
ylabel('Correctness')
title('Classification Result Correctness')

%% seattle classification
clear all; close all; clc;

%load data
sg = [];

%get sample
info1 = audioinfo('hw4music/Soundgarden.wav');
fs = info1.SampleRate;
for i = 60:30:3045
    [y,~] = audioread('hw4music/Soundgarden.wav',[floor(i*fs) floor((i+5)*fs)]);
    y = (y(:,1) + y(:,2))./2;
    y = spectrogram(y);
    sg = [sg, y(:)];
end
%%
aic = [];
info2 = audioinfo('hw4music/AliceinChains.wav');
fs = info2.SampleRate;
for i = 60:20:2045 %short collection
    [y,~] = audioread('hw4music/AliceinChains.wav',[floor(i*fs) floor((i+5)*fs)]);
    y = (y(:,1) + y(:,2))./2;
    y = spectrogram(y);
    aic = [aic, y(:)];
end

%%
pj = [];
info3 = audioinfo('hw4music/Pearljam.wav');
fs = info3.SampleRate;
for i = 60:20:2045
    [y,~] = audioread('hw4music/Pearljam.wav',[floor(i*fs) floor((i+5)*fs)]);
    y = (y(:,1) + y(:,2))./2;
    y = spectrogram(y);
```

```
    pj = [pj, y(:)];
end
%%
% first 100 col sg, second 100 col aic, last 100 col pj
spec = [sg, aic(1:229383,:) ,pj];
[u,s,v] = svd(spec,'econ');

%%
%randomly choose train and test sets
q1 = randperm(100);
q2 = randperm(100);
q3 = randperm(100);

%picking features
x1 = abs(v(1:100,2:7));
x2 = abs(v(101:200,2:7));
x3 = abs(v(201:300,2:7));

%train set
xtrain = [x1(q1(1:80),:); x2(q2(1:80),:); x3(q3(1:80),:)];

%test set
xtest = [x1(q1(81:end),:); x2(q2(81:end),:); x3(q3(81:end),:)];

%label trainning data
labels = [ones(80,1);2*ones(80,1);3*ones(80,1)];
class = classify(xtest,xtrain,labels);

subplot(2,1,1)
%pre = predict(class,xtest);
bar(class)
xlabel('Tests')
ylabel('Class')
title('Classification Result')

c1 = 0;c2 = 0;c3 = 0;
% calculate correctness rate
for i = 1:60
    if (i <= 20 && class(i,1) == 1 )
        c1 = c1 + 1;
    elseif (i <= 40 && i > 20 && class(i,1) == 2)
        c2 = c2 + 1;
    elseif (i <= 60 && i > 40 && class(i,1) == 3)
        c3 = c3 + 1;
    end
end
c1 = c1/20;
```

```
c2 = c2/20;
c3 = c3/20;
subplot(2,1,2)
bar([c1 c2 c3])
xlabel('Artists')
ylabel('Correctness')
title('Classification Result Correctness')

%% Genre Classification
clear all; close all; clc;

%load data
rap = [];

%get sample
info1 = audioinfo('hw4music/Rap.wav');
fs = info1.SampleRate;
for i = 60:20:2045
    [y,~] = audioread('hw4music/Rap.wav',[floor(i*fs) floor((i+5)*fs)]);
    y = (y(:,1) + y(:,2))./2;
    y = spectrogram(y);
    rap = [rap, y(:)];
end

%%
pop = [];
info2 = audioinfo('hw4music/Pop.wav');
fs = info2.SampleRate;
for i = 60:30:3045 %short collection
    [y,~] = audioread('hw4music/Pop.wav',[floor(i*fs) floor((i+5)*fs)]);
    y = (y(:,1) + y(:,2))./2;
    y = spectrogram(y);
    pop = [pop, y(:)];
end

%%
cl = [];
info3 = audioinfo('hw4music/classic.wav');
fs = info2.SampleRate;
for i = 60:30:3045 %short collection
    [y,~] = audioread('hw4music/classic.wav',[floor(i*fs) floor((i+5)*fs)]);
    y = (y(:,1) + y(:,2))./2;
    y = spectrogram(y);
    cl = [cl, y(:)];
end

%%
```

```
% first 100 col rap, second 100 col pop, last 100 col classic
spec = [rap, pop ,cl];
[u,s,v] = svd(spec,'econ');

%%
%randomly choose train and test sets
q1 = randperm(100);
q2 = randperm(100);
q3 = randperm(100);

%picking features
x1 = abs(v(1:100,2:7));
x2 = abs(v(101:200,2:7));
x3 = abs(v(201:300,2:7));

%train set
xtrain = [x1(q1(1:80),:); x2(q2(1:80),:); x3(q3(1:80),:)];

%test set
xtest = [x1(q1(81:end),:); x2(q2(81:end),:); x3(q3(81:end),:)];

%label trainning data
labels = [ones(80,1);2*ones(80,1);3*ones(80,1)];
class = classify(xtest,xtrain,labels);

subplot(2,1,1)
%pre = predict(class,xtest);
bar(class)
xlabel('Tests')
ylabel('Class')
title('Classification Result')

c1 = 0;c2 = 0;c3 = 0;
% calculate correctness rate
for i = 1:60
    if (i <= 20 && class(i,1) == 1 )
        c1 = c1 + 1;
    elseif (i <= 40 && i > 20 && class(i,1) == 2)
        c2 = c2 + 1;
    elseif (i <= 60 && i > 40 && class(i,1) == 3)
        c3 = c3 + 1;
    end
end
c1 = c1/20;
c2 = c2/20;
c3 = c3/20;
subplot(2,1,2)
```

```
bar([c1 c2 c3])
xlabel('Artists')
ylabel('Correctness')
title('Classification Result Correctness')
%% for plotting frequency only
% subplot(4,1,1)
% plot((1:length(y1(:,1)))/(length(y1(:,1))/5),y1(:,1));
% xlabel('Time'); ylabel('Amplitude'); title('Pop')
% subplot(4,1,2)
% plot((1:length(y2(:,1)))/(length(y2(:,1))/5),y2(:,1));
% xlabel('Time'); ylabel('Amplitude'); title('Rap')
% subplot(4,1,3)
% plot((1:length(y3(:,1)))/(length(y3(:,1))/5),y3(:,1));
% xlabel('Time'); ylabel('Amplitude'); title('Jazz')
% subplot(4,1,4)
% plot((1:length(y4(:,1)))/(length(y4(:,1))/5),y4(:,1));
% xlabel('Time'); ylabel('Amplitude'); title('Classical')
```

# C Music Collections

## C.1 Test 1: Band

**Michaeal Jackson** https://www.youtube.com/watch?v=z1fadkdxAX0

**Louis Armstrong** https://www.youtube.com/watch?v=SebmIELnZoQ

**Yiruma** https://www.youtube.com/watch?v=8Xjdj6-uFPY

## C.2 Test 2: Band in Seattle

**Soundgarden** https://www.youtube.com/watch?v=4NA2L8lTmX8

**Alice In Chains** https://www.youtube.com/watch?v=MHy1Yo9EzaM

**Pearl Jam** https://www.youtube.com/watch?v=bO9htD6gT0w

## C.3 Test 3: Genre

**Rap** https://www.youtube.com/watch?v=qGmSTlvnTGI

**Pop** https://www.youtube.com/watch?v=1zju3Cevz0o

**Classic** https://www.youtube.com/watch?v=xgs-laNZ0SE